The simulation of incompressible flow using the artificial compressibility method.

Sabbagh-Yazdi, Saeid Reza

Department of Civil Engineering
University of Wales, Swansea

The Simulation of Incompressible Flow Using the Artificial Compressibility Method

by

Saeid Reza Sabbagh Yazdi

B.Sc. , M.Sc.

Thesis submitted to the University of Wales
in candidature for the degree of
Doctor of Philosophy

May/ 1997

C/Ph/213/97/

CERTIFICATE OF RESEARCH

This work has not previously been accepted in substance for any degree and is not currently submitted in candidature for any degree.

Candidate signature :

Date : May 1997

STATEMENT 1

This thesis is the result of my own investigations, except otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Candidate signature :

Date : May 1997

STATEMENT 2

I hereby consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Candidate signature :

Date : May 1997

ACKNOWLEDGEMENT

I would like to express my sincere appreciation towards my supervisor, professor N.P.Weatherill for his constructive suggestions, guiding comments and informative discussions which have gratefully stimulated the development of this work.

It is my pleasure to acknowledge helps I received from several staff and students in the Department of Civil Engineering, University of Wales Swansea, particularly Dr. O.Hassan, Dr. M.Marchant for providing me some meshes and post/pre-processing soft wares, Dr. F.Beaven, Dr. D.K.Natakusumah and Dr. M.T.Manzari for conceptual assistance, Dr. N.Verhoeven, Mr. R.Said, Dr. Y.Zheng, Mr E.T.Smith who helped generate some meshes, and finally all the administrators, specially staffs of computer section and secretaries, Mrs. L.Jenkins and Mrs. A.E.Davies for their cooperations. I greatly enjoyed working with my colleagues, especially my Iranian friends who made me feel at home during my study in Swansea.

I would also like to gratefully acknowledge the financial support provided by the Ministry of Culture and Higher Educations of Islamic Republic of Iran. I take this opportunity to thank all the members of staff in the Education department of the Embassy of Islamic Republic of Iran, particularly Dr. H. S. Amoli due to the provision of services.

My deepest thanks to all my family, particularly my wife, Maryam, for their moral support, understanding, companionship and constant optimistic encouragements. Finally, I would like to dedicate this thesis to my daughter, Shiva, who deserves the most appreciation for her endurance and lasting patience.

SUMMARY

In this work an algorithm is developed for simulating incompressible steady flow on two and three-dimensional unstructured meshes. The Navier-Stokes equations are briefly reviewed as the basic governing equation for fluid flow. Using this set of equations, in the limit of incompressible flow, the problem of imposing the time independent continuity equation on the momentum equations arises. This difficulty can be removed by employing the Artificial Compressibility approach. This approach modifies the continuity equation by adding a pseudo pressure time derivative. This modification makes the set of equations well conditioned for numerical solution. If the set of modified equations is used for the solution of steady state problems, the added pressure derivative tends to zero and the set of equations reduces to the steady state incompressible Navier-Stokes equations.

The cell-vertex finite volume method is employed for solving the modified equations on unstructured triangular and tetrahedral meshes. The principles of central difference space discretisation are described and the basic ideas behind adding artificial dissipation term are reviewed. A normalisation procedure for the computation of the artificial dissipation term is adopted. Two different formulations based upon a cell-vertex finite volume method and a Galerkin finite element method for the discretisation of the viscous terms on unstructured triangular meshes are employed in two and three dimensions. A modification to the finite volume formulation is introduced for improving accuracy on unstructured grids. The issues relating to multi-stage time stepping, boundary conditions and some techniques for increasing computational efficiency are described. A general review of several methods for generating regular and irregular unstructured triangular and tetrahedral meshes are presented.

The proposed algorithm is validated by solving several inviscid and viscous two-dimensional test cases. Extension of the algorithm to three dimensions are studied by using some further benchmark examples. Some engineering applications are considered to present the ability of the developed flow solver to simulate more complicated real world problems. Finally, some conclusions are drawn and a few guidelines for further research work are suggested.

TABLE OF CONTENTS

Chapter 1 : Introduction

Chapter 2 : Basic Governing Equations of Fluid Mechanics

Chapter 3 : Formulation of the Incompressible Flow Equations

Chapter 4 : Finite Volume Formulation

Chapter 5 : Discretisation of Governing Equations

Chapter 8 : Simulation of Two-Dimensional Flow

Chapter 9 : Three-Dimensional Flow Simulation

Chapter 10 : Concluding Remarks

1

Introduction

1.1 Over View

Simulation of the physics of fluid flow plays an important role in understanding and adapting flow phenomena for engineering purposes. For over 200 years, researchers have studied fluid flow in an attempt to help solve realistic engineering problems. In many instances, a fluid interacts with objects which results in forces on the object, and in some cases, these forces change the form and position of the objects which can then have a significant effect on the flow itself. Examples of fluid-structure interaction problems includes the flow over automobiles, low speed aircraft and structures, such as tall buildings, cooling towers and large cylindrical storage tanks.

The behaviour of the fluid can be described by a system of equations which is known as the Navier-Stokes equations. Due to considerable mathematical difficulties encountered in solving this non-linear system of equations, it is very difficult, for general flows, to find an analytical method of solution. By applying simplifying assumptions, some comprehensive analytical methods for fluid mechanics have been developed. However, these methods are generally limited in their ability to simulate complicated flows over complex geometrical configurations.

Experimental methods have historically been used to measure the fluid flow parameters over complex geometries. Although wind tunnel experiments have the capability to produce information on realistic flows, they are expensive. The task of matching flow conditions, the difficulties associated with accurate measurements and the time consuming nature of the procedures are other disadvantages associated with wind tunnel experiments. Furthermore, there are many cases in which the use of wind tunnel experiments are practically impossible.

The development of high speed digital computers has led to the development of Computational Fluid Dynamics (CFD) for simulating complex fluid flows. Nowadays, the simulation of complex flow fields is possible by numerically solving the mathematical equations which govern fluid flow. The control over fluid properties and the ability to simulate complicated flow fields are some advantages offered by CFD. Although wind tunnel experiments are still very important, computer simulations can be used to provide a significant insight into fluid behaviour. Since CFD is now a rapidly maturing discipline, many different techniques have been developed and a complete review of these methods is beyond the scope of this thesis. However, a description of Computational Fluid Dynamics, together with an historical background, can be found in the survey papers by Chapman [1], Kutler [2] and Jameson [3].

The main focus of this work is to develop a numerical solution algorithm for the simulation of incompressible viscous flow suitable for simulating aerodynamic and environmental flows. Interest in this topic originates from the need to achieve accurate predictions of pressure and velocity fields to improve the aerodynamic design of objects. Such simulations have been used for improving the design of proposed aerodynamic configurations such as cars and low speed aircraft. In addition, today, there is considerable interest in using CFD to predict flow fields around environmental structures, such as buildings.

In an effort to achieve this, inviscid and viscous flow solvers are developed, firstly in two dimensions, then extended for three-dimensional flow simulation.

Validation of the scheme is accomplished by comparing the numerical results with the results from available analytical, experimental and independent numerical results published in the literature. Finally, the flow solver is applied to study realistic flows.

1.2 Background

In order to simulate fluid flow the set of Navier-Stokes equations can be used as an appropriate mathematical model. Unfortunately, these equations are highly non-linear and there are several difficulties in obtaining their solution. Hence, to make some simplifications to these equations, appropriate approximations have to be made. At the first level of approximation, the Reynolds averaged form of the equations can be applied for many practical applications. This hypothesis assumes that the flow variables can be determined in terms of a mean value plus a value representing the fluctuation due to turbulence effects. Then, laminar flows can be regarded as a special case of the Reynolds Averaged Navier-Stokes equations, where the turbulence is zero. For high Reynolds number flows about streamlined bodies, the entire outer flow field is effectively inviscid and the effects of viscosity are confined to a thin layer adjacent to the solid wall and in the wake of the body. Hence, the second level of approximation can be achieved by completely neglecting the effects of viscosity and thermal conductivity and this leads to the Euler equations. Further simplification can be made by making the assumption of irrotational flow and thus a velocity potential function can be introduced. The resulting mathematical formulations are called the Potential Flow equation [4,5,6].

Using methods developed for the solution of the compressible Navier-Stokes equations gives rise to some numerical difficulties in the limit of the incompressible flow. Since density remains constant, the time derivative in the continuity equation is zero. On the other hand, the speed of sound dominates the system of equations and destroys the stability and accuracy of many numerical schemes. Therefore, the development of computational algorithms for incompressible flows implies that techniques are required to overcome the difficulties of extracting the pressure from the combined continuity and momentum equations. These problems can be resolved using the artificial compressibility approach which casts the set of equations in pseudo transient form by introducing an artificial compressibility term into the continuity equation. The resulting equations can be solved using methods which have been developed for the computation of compressible flows.

Computer based methods which have been developed to solve the Reynolds Averaged Navier-Stokes equations can be divided into three main categories; Finite Difference Methods (FDM), Finite Element Methods (FEM), and Finite Volume Methods (FVM). The FDM has been adopted by many CFD developers for use on regular curvilinear structured grids. Implicit and explicit schemes have been successfully applied to solve the set of Navier-Stokes equations [7-9]. The FEM provides great flexibility at the expense of a more complicated mathematical formulation. Various FEM schemes have been applied to handle inviscid and viscous flows in two and three dimensions [10-13]. The FVM has a simple physical interpretation when applied to the flow equations. The flexibility and simplicity of the FVM facilitate its implementation on unstructured meshes for solving complex problems. FVM practitioners have developed several algorithms which have demonstrated the applicability of the approach to simulate flow over arbitrary complicated geometries. One particular approach, which has been developed for compressible flow solvers, is central space discretisation coupled with multi-stage time stepping [14-19]. The considerable efficiency achieved using this approach makes it a suitable method for adopting for the simulation of incompressible flows.

Apart from the choice of numerical method, devising a suitable mesh is another important requirement of any numerical method. For simple geometries, an appropriate grid can be generated using methods based on mapping techniques [20]. For more complex geometries, the generation of suitable grids has proved to be quite troublesome as the shapes do not readily conform to a single global structured grid. To solve this problem, two common approaches have been proposed; the multi-block structured approach [21] and the unstructured triangular approach [22]. The multi-block approach decomposes the global domain into several regions and within each sub-region a local structure grid is generated. However, the subdivision of the domain into a number of sub-regions which must be topologically equivalent to blocks is a non-automated process, and hence, is time consuming. The use of unstructured grid generation, however, overcomes many of these problems. Several unstructured triangular mesh generation techniques have been recently developed, which can generate appropriate meshes for two and three-dimensional domains. The main advantage of generating meshes in this way is that they provide a flexible manner for discretising complex domains without recourse to large scale domain decomposition [22]. However, in the unstructured approach, a connectivity matrix must be used to define the elements and this leads to the requirement to use indirect addressing in the flow solver.

In this work, effort has been devoted to developing a cell vertex Finite Volume algorithm which uses multistage integration and central space discretisation to solve the incompressible flow equations. The ability of the algorithm to deal with complex geometrical configuration is maintained by utilising side-based data structure on unstructured triangular and tetrahedral meshes. This algorithm can be applied for the simulation of inviscid and viscous incompressible flows over solid bodies.

1.3 Scope of the Research

Difficulties associated with the numerical solution of the set of Navier-Stokes equations in the limit of the incompressible flow has motivated the development of several methods. The incompressible flow equations can be solved using derived or primitive variable formulations of the Navier-Stokes equations.

The derived variable approach uses new dependent variables. As some examples, Stream Function/Vorticity and Vorticity/Velocity approaches have been used [6,24]. However, these methods can not easily be extended to three dimensions since there are difficulties in expressing boundary conditions and there is a need for a separate equation for the solution of the pressure field [6].

The primitive variable approach utilises the independent variables of velocity and pressure which can be classified into three groups; the Pressure Poisson method, the Penalty method, the Fractional Step technique and the Artificial Compressibility approach.

The first, which historically was one of the most commonly used, is the Pressure Poisson method [5]. In this method the velocity field is advanced in time by using the momentum equation. Then, pressure is obtained from the Poisson equation such that the continuity equation will be satisfied at the next time step. The method can be applied to steady state problems and it is referred to as the Pressure Correction method. In this method, the velocity and pressure are indirectly coupled.

The second group of primitive variable approaches is known as Fractional Step or the Velocity Correction technique which was first introduced by Chorin [25]. This technique starts the computation by solving an intermediate velocity field for the momentum equations with omitted pressure gradients. Then, the pressure field is solved by inserting the intermediate velocity field into the continuity and the momentum equations. The pressure field is usually determined separately in the second stage of computations. This technique has been widely used in the context of FDM and FEM [28-34]. Application of the method was extended to the solution of both steady incompressible and slightly compressible flows [30-32]. Recently, attempts have been made to use this technique to develop a unique algorithm for solving both incompressible and compressible flows [32,33].

The third method which uses primitive variables was proposed by Chorin [34] and is known as the method of Artificial Compressibility. In this approach, a pseudo time derivative of pressure is added to the continuity equation, which directly couples the pressure and velocity fields. The equations are advanced in time until a divergence-free velocity field is obtained for steady state conditions. Similarity exists between this method and the Fractional Step (Velocity Correction) method [25], which maps the intermediate velocity field to a divergence-free velocity field using an Artificial Compressibility parameter. Hence, because time accuracy of the set of equations has been destroyed, both approaches are only suitable for steady state problems. However, the Artificial Compressibility approach has the advantage of a direct coupling between the pressure and velocity fields as they are advanced in pseudo time. This makes the approach computationally less expensive than the other primitive variable approaches. The Artificial Compressibility approach has been widely used in a FDM context [35-41]. Some FDM developers have extended the approach to the solution of the time dependent two and three-dimensional problems [42-45]. Turkel suggested a more sophisticated approach based on mathematical analysis [46]. Among FEM users, Peraire et al applied the approach for the simulation of three-dimensional flows on unstructured grids [47].

In the FVM community, Rizzi and Erikson [48] used the approach in applications to rotational inviscid flows on three-dimensional structured meshes. Farmer [50] applied the approach to ship wave problems considering free surface boundaries and Belov et al [51] extended the approach to solve time dependent problems by using a semi-implicit scheme on three-dimensional structured grids. Dreyer [49] used the approach utilising Jameson's cell-vertex Finite Volume scheme [14-16] for solving inviscid flow problems on two-dimensional unstructured triangular grids. The Jameson scheme, which employs multistage Runge-Kutta time stepping together with separate central difference space discretisation and added artificial dissipation term, provides a flexible procedure to simulate flow over complex configurations. In the present work, an attempt is made to adapt the Artificial Compressibility approach to the solution of the incompressible flow equations and to solve the resulting equations using the Jameson's cell-vertex Finite Volume scheme [14,16] on two and three-dimensional unstructured meshes.

1.4 Outline of Presentation

The main aspect of this work is the development of an algorithm for the simulation of incompressible laminar flows. Towards this goal, the set of Navier-Stokes equations is modified according to the Artificial Compressibility approach. More flexibility is provided by adopting a cell-vertex Finite Volume discretisation on unstructured triangular meshes and utilising a side-based algorithm in three-dimensional generalised coordinates. The solution procedure follows the multi-stage Runge-Kutta time stepping approach together with a separate central difference space discretisation with added artificial dissipation. The numerical solution of the modified equations is applied on two-dimensional regular and irregular unstructured meshes. The approach is then extended to three dimensions. In this thesis, the procedures followed are reported through ten chapters.

Chapter 2 reviews the mathematical description of the laminar Navier-Stokes equations together with the appropriate boundary conditions.

Chapter 3 considers the modification of the Navier-Stokes equations required for the Artificial Compressibility approach.

Chapter 4 summarises the cell-vertex Finite Volume formulation and its application to the integral form of the Navier-Stokes equations.

Chapter 5 describes the discretisation of the governing equations, including the convective, artificial dissipation and viscous terms, on unstructured triangular and tetrahedral meshes.

Chapter 6 discusses the details of the numerical procedures employed for the solution of the modified equations using a multi-stage time-stepping scheme and discusses issues related to computational efficiency.

Chapter 7 briefly reviews the different mesh generation techniques which have been applied for creating unstructured triangular meshes.

Chapter 8 presents results which assess the performance of the algorithm on appropriate test cases for the simulation of two-dimensional inviscid and viscous incompressible flows.

Chapter 9 presents extensions to the flow solver for the simulation of three-dimensional incompressible flows and shows the accuracy and efficiency of the algorithm to deal with different complicated test geometries.

Chapter 10 draws the final conclusions and proposals for further work.

References

[1] D. R. Chapman, 'Computational Aerodynamics Development and Outlook', Dreyden Lectureship in Research. AIAA J. **17**, No 12, 1293–1313, (1979).

[2] P. Kutler, 'A Perspective of Theoretical and Applied Computational Fluid Dynamics', AIAA J., Vol. **23**, 328-341, (1985).

[3] A. Jameson, 'Successes and Challenges in Computational Aerodynamics', AIAA Paper 87-1184, (1987).

[4] C. Hirsch, 'Numerical Computation of Internal and External Flows', Vol. **1**, Fundamentals of Numerical Descritization, Wiley, (1989).

[5] F. H. Harlow and J. E. Welch, 'Numerical Calculation of Time Dependent Viscous Incompressible Flow with Free Surface', Physics of Fluids, **8**, pp. 2182–2189, (1965).

[6] D. A. Anderson, J. C. Tannehill and R. H. Pletcher, 'Computational Fluid Mechanics and Heat Transfer', Hemisphere Publishing Co. and McGraw-Hill Co., (1984).

[7] MacCormack, R. W. 'A Numerical Method for Solving the Equations of Compressible Viscous Flows', AIAA Paper 81-110, (1981).

[8] R. Beam and R. F. Warming,, 'An Implicit Factored Scheme for the Compressible Navier-Stokes Equations, AIAA J., **16**, (1978).

[9] T. H. Pulliam and J. L. Steger, 'Recent Improvements in Efficiency, Accuracy and Convergence for Implicit Approximate Factorisation Algorithms', AIAA Paper 85-0360, January (1985).

[10] R. Lohner, K. Morgan, J. Peraire and O. C. Zienkiewicz, 'Finite Element Methods for High Speed Flows', In Proceeding of the AIAA 7th Computational Fluid Dynamics Conference, Cincinnati, Paper AIAA-85-1531-CP, (1985).

[11] J. Peraire, J. Peiro, L. Formaggia, K. Morgan and O. C. Zienkiewicz,

'Finite Element Euler Computations in Three Dimensions, AIAA Paper 88–0032, (1988).

[12] O. Hassan, K. Morgan and J. Peraire, 'An Implicit/Explicit Finite Element Method for Compressible Viscous High Speed Flows', AIAA Paper 89-0363, (1989).

[13] O. Hassan, K. Morgan and J. Peraire, 'An Implicit Finite Element Method for High Speed Flows', AIAA Paper 90-0402, (1990).

[14] A. Jameson, W. Schmidt and E. Turkel, 'Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes', AIAA Paper 81-1256, June (1981)

[15] W. Haase, B. Wagner, A. Jameson and W. Schmidt, 'Development of a Navier-Stokes Method Based on Finite Volume Techniques for Solving the Time Dependent Euler Equations',5'th GAMM-Conference on Numerical Methods in Fluid Mechanics', Roma, Italy, (1983).

[16] A. Jameson, T. J. Baker and N. P. Weatherill, 'Calculation of Inviscid Transonic Flow over a Complete Aircraft', AIAA Paper 86-0103, (1986).

[17] F. Grasso, A. Jameson and L. Martinelli, A Multistage Multi-grid Method for the Compressible Navier-Stokes Equations', in Numerical Simulation of Compressible Navier-Stokes Flows: a GAMM workshop, eds. M.O Bristeau et al, Vieweg and Sohn, (1987).

[18] D. J. Mavriplis, 'Accurate Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes', AIAA J., Vol 28, No. 2, pp. 213–221, (1990).

[19] D. J. Mavriplis and A. Jameson, 'Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes', AIAA J. , Vol **28**, No. 8, pp. 1415–1425, (1990).

[20] J. F. Thompson, A. U. Warsi and C. W. Mastin, 'Numerical Grid Generation , Foundations and Applications', North Holand, Elsevier Science Publishing Co., (1985).

[21] J. Shaw, C. R. Forsey, N. P. Weatherill and K. E. Rose, 'A Block Structured Mesh Generation Techniques for Aerodynamics Geometries', In

Numerical Grid Generation in Computational Fluid Dynamics, eds. J. Hauser and C. Taylor, Pineridge Press (1986).

[22] N. P. Weatherill, 'A Method for Generating Irregular Computational Grids in Multiply Connected Planar Domains', Int.. J. Num. Meth. in Fluids, **8**, pp. 181–197, (1988).

[23] N. P. Weatherill, 'Grid Generation', VKI Lecture Series 13–14, (1994).

[24] R. Peyret, T. D. Taylor, 'Computational Methods for Fluid Flow', Springer Verleg New York, pp. 142–155, (1983).

[25] A. Chorin , 'Numerical Solution of Navier-Stokes Equations' , Math. of Comp. , **22**, pp 745–762 (1968).

[26] A. Chorin , 'On the Convergence of Discrete Approximations to the Navier-Stokes Equations' , Math. of Comp. , **23**, pp 341–353 (1969).

[27] G. Comini and S. Del Guidice, 'Finite Element Solution of Incompressible Navier-Stokes Equations, Numerical Heat Transfer, pp 463–478, (1972).

[28] G. E. Schneider, G. D. Raithby, M. M. Yovanovich, 'Finite Element Analysis of Incompressible Fluid Flow Incorporating Equal Order Pressure and Velocity Interpolation', In Numerical Methods in Laminar and Turbulent Flow, C. Taylor, K. Morgan , C. A. Brebia , Pentech Press, Plymouth, (1978).

[29] M. Kavahra, K. Ohmiya, 'Finite Element Analysis of Incompressible Density Flow Using Velocity Correction Method', Int. J. for Num. Meth. in Fluids, **5**, pp 981–993 , (1985).

[30] J. Szmelter, 'Computational Methods for Compressible and Incompressible Fluid Mechanics' ,Ph.D Thesis, University of Wales, Swansea, (1989).

[31] O. C. Zienkiewicz, J. Szmelter, J. Paraire, 'Compressible and Incompressible Flow, An Algorithm for All Seasons' ,Comp. Math. Appl. Mech. Eng. , **78**, pp.105–121, (1990).

[32] J. Wu, 'Compressible and Incompressible Flow Problems – A New Finite Element Algorithm and Adaptivity' , Ph.D Thesis, University of Wales, Swansea, (1992).

[33] O. C. Zienkiewicz, K. Morgan and B. V. Satya Sai, 'A General Algorithm for Compressible and Incompressible Flow, Part I and II', Int. J. for Num. Meth. in Fluids, **20**, pp 887–913 , (1995).

[34] A. Chorin , 'A Numerical Method for Solving Incompressible Viscous Flow Problems', J. of Comp. Physics , **2**, pp. 12–26, (1967).

[35] J. Steger, P. Kutler, 'Implicit Finite Difference Procedures for Computation of Vortex Wake', AIAA J. , **15**, pp. 581–90, (1977).

[36] J. L. C. Chang, D. Kwak, 'On The Method of Pseudo Compressibility for Numerically Solving Incompressible Flows', AIAA 22nd Aerospace Sciences Meeting, AIAA Paper 84-0252, (1984).

[37] J. L. C. Chang, D. Kwak, S. C. Dao, R. Rosen, 'A Three-dimensional Incompressible Flow Simulation Method and its Application to Space-Shuttle Main Engine - Part I - Laminar Flow', AIAA 3rd Aerospace Meeting, AIAA Paper 85-0175, (1985a).

[38] J. L. C. Chang, D. Kwak, S. C. Dao, R. Rosen, 'A Three-Dimensional Incompressible Flow Simulation Method and its Application to Space-Shuttle Main Engine, Part II-Turbulent Flow', AIAA 18th Fluid Dynamics , Plasma-dynamics Conference, AIAA Paper 85-1670, (1985b).

[39] D. Choi, C. L. Merkle, 'Application of Time Iterative Scheme to Incompressible Flow', AIAA J., **23**, pp. 1518–24, (1985).

[40] D. Kwak, L. J. Chang, S. P. Shanks, S. Chakaravarthy, 'A Three-dimensional Incompressible Navier-Stokes Flow Solver Using Primitive Variables', AIAA Journal, **24**, pp. 390–6, (1986).

[41] W. Y. Soh, 'Time Marching Solution of Incompressible Navier-Stokes Equations for Internal Flow', J. of Comp. Physics, **70**, pp. 232–52, (1987).

[42] C. L. Merkle, M. Athaval, 'Time-accurate Unsteady Incompressible Flow Algorithm Based on Artificial Compressibility ', AIAA Paper 87-1137, (1987).

[43] W. Y. Soh, J. W. Goodrich, 'Unsteady Solution of Incompressible Flow Navier-Stokes Equations', J. of Comp. Physics, **79**, pp. 113–114, (1988).

[44] S. E. Rogers and D. Kwak, 'Upwind Differencing Scheme for Time-accurate Incompressible Navier-Stokes Equations', AIAA paper 87-1137, (1987).

[45] S. E. Rogers, 'A Comparison of Implicit Schemes for the Incompressible Navier-Stokes Equations with Artificial Compressibility', AIAA paper 95-0567, (1995).

[46] E. Turkel, 'Preconditioned Method for Solving the Incompressible and Low Speed Compressible Equations', ICASE Report 86–14, NASA Langley Centre, (1986).

[47] J. Peraire, K. Morgan and J. Piero, 'The Simulation of 3D Incompressible Flows Using Unstructured Grids', Frontiers of Computational Fluid Dynamics, John Wiley and Sons, pp 281–301, (1994).

[48] A. Rizzi, L. E. Eriksson, 'Computation of Inviscid Incompressible Flow With Rotation', J. of Fluid Mech., **153**, pp. 275–312, (1985).

[49] J. Dreyer, 'Finite Volume Solution to The Steady Incompressible Euler Equations on Unstructured Triangular Meshes', M.Sc. Thesis, MAE Dept., Princeton University, (1990).

[50] J. Farmer, 'A Finite Volume Multigrid to the Three-dimensional Nonlinear Ship Wave Problem', Ph.D. Thesis, Princeton University, (1993).

[51] A. Belov, L. Martinelli, and A. Jameson, 'A New Implicit Algorithm with Multigrid for Unsteady Incompressible Flow Calculations', AIAA paper 95-0049 , (1995).

2

Basic Governing Equations of Fluid Mechanics

2.1 Introduction

The behaviour of a physical system can be expressed by conservation laws. It means that the quantities such as mass, momentum and energy are conserved regardless of the geometry of the system. These three conditions without any additional dynamic law completely describe the behaviour of the flow field. The conservation of mass, momentum and energy for the motion of an unsteady flow can be expressed by the system of Navier-Stokes equations.

Mathematically, the Navier-Stokes equations may be written in two forms; differential and integral form. However, the differential form is equal to the integral form with the assumption of the smoothness of the solution. The laminar viscous form of the Navier-Stokes equations can be applied by neglecting the turbulence effects. Far from solid bodies and wake regions, viscous effects can normally be neglected and the Navier-Stokes equations then reduce to the Euler equations. The system of Euler equations constitutes the most complete description of inviscid, non-heat-conducting flow and hence is the highest level of approximation for non-viscous fluids. Although the Euler equations are obviously not universally valid, the importance of their accurate numerical simulation resides in the dominating convective character of the Navier-Stokes equations at high Reynolds numbers.

In this chapter, in order to give a short review of the laminar viscous incompressible flow and related equations, the mathematical formulation of the general form of Navier-Stokes equations is described. General conservative form of the Navier-Stokes equations is presented in Section 2.2. Section 2.3 contains the vector form of these equations. The modifications for the isothermal and incompressible flow are described in Sections 2.4. Section 2.5 and 2.6 present sets of equations for viscous and inviscid flows, respectively. Finally, in Section 2.7, a general description of the equations which are applied to implement solid wall, far field and symmetry boundary conditions are discussed.

2.2 The Navier-Stokes Equations

Complete description of the flow behaviour, at any specified time, requires information about velocity field and static properties at all points of the domain. For compressible flow, the static properties for any real flow in thermodynamic equilibrium are density and pressure. In the region of low speed flow, with the assumption of isothermal flow, the changes in density are negligible so that the flow can be considered incompressible. Hence, in this case, the static property of the flow is only specified by pressure.

In order to describe the governing flow equations, consider a domain Ω, enclosed by boundary Γ, fixed with respect to a Cartesian frame of reference, with $\vec{\mathbf{n}}$ being the outward directed unit vector normal to the boundary Γ (Figure 2.1).

If the local variation of a quantity per unit volume $\mathbf{W}$ is acting in the domain Ω, the variation of $\mathbf{W}$ is due to the effect of the internal sources S and the fluxes from surrounding boundaries. If the quantity $\mathbf{W}$ obeys the conservation law, the rate of its variation within the domain Ω, is equal to the net contribution of the fluxes $\vec{\mathbf{F}}$ passing through boundary Γ plus any surface source S_Γ and volume source $\vec{S}_\Omega$. They can be written as

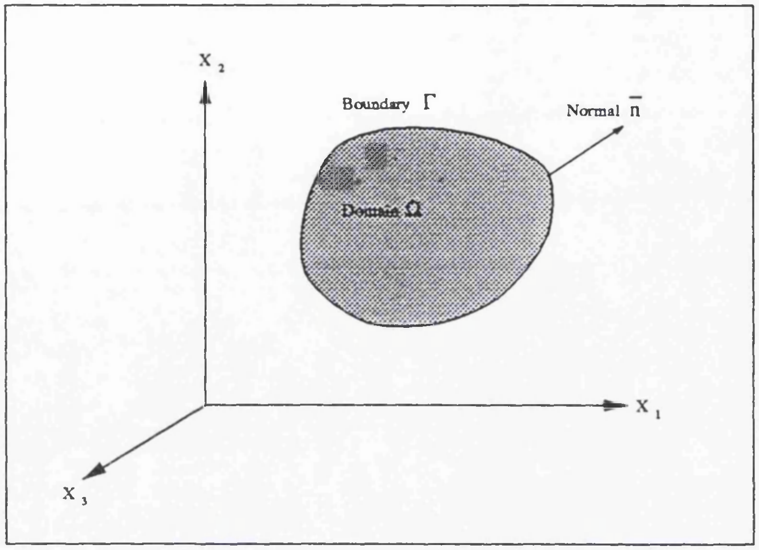

Figure 2.1 *Description of the three dimensional flow domain Ω*

$$\frac{\partial}{\partial t}\int\limits_{\Omega} \mathbf{W}\, d\Omega = -\oint\limits_{\Gamma} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma + \oint\limits_{\Gamma} \vec{S}_{\Gamma} \cdot \vec{\mathrm{n}}\, d\Gamma + \int\limits_{\Omega} S_{\Omega}\, d\Omega \qquad (2.1)$$

The equation of the conservation law can be written in volume integral form using Gauss divergence theorem by converting the surface integrals to volume integrals. After rearranging, we have

$$\frac{\partial}{\partial t}\int\limits_{\Omega} \mathbf{W}\, d\Omega + \int\limits_{\Omega} \vec{\nabla} \cdot \vec{\mathbf{F}}\, d\Omega = \int\limits_{\Omega} \vec{\nabla} \cdot \vec{S}_{\Gamma}\, d\Omega + \int\limits_{\Omega} S_{\Omega}\, d\Omega \qquad (2.2)$$

The differential form of the conservation law for the domain Γ can be written as

$$\frac{\partial \mathbf{W}}{\partial \mathrm{t}} + \vec{\nabla} \cdot \vec{\mathbf{F}} = \vec{\nabla} \cdot \vec{S}_{\Gamma} + S_{\Omega} \qquad (2.3)$$

where $\vec{\nabla}$ is the divergence of the vector field defined by $\vec{\mathbf{F}}$ which includes partial derivatives $\partial/\partial x_i \mathbf{I}_i$. Here, x_i is the coordinates in the i direction of the cartesian coordinates and $\mathbf{I}_i$ is the unit vectors in the i direction ($j = 1, 2, 3$).

The vector $\vec{\mathbf{F}} = \mathbf{W}\vec{\mathbf{U}}$ consists of the flux passing through the boundary of the domain. The surface source $\vec{S}_\Gamma$ can be considered as the stresses acting on the surfaces of the boundary which is called the diffusive flux. The vector of internal sources $\vec{S}_\Omega$ describes the contribution due to the molecular motion and thermal reaction of the fluid. The computations to be considered here are for flows in the absence of volume sources. Hence, the term S_Ω is assumed to be omitted from equations (2.2) and (2.3) for the rest of this chapter.

2.2.1 Conservation of Mass

For conservation of mass, the quantity $\mathbf{W}$ is considered as the specific mass ρ. The specific mass flux is a vector quantity because of the scalar nature of ρ. For a single phase fluid, any variation of ρ is due to the convective flux only. With the assumption of no source or sink of mass within the domain, the rate of increased specific mass inside the domain is equal to the rate of flow which crosses the domain. Therefore, using equation (2.2), the integral form of the mass conservation law can be considered as

$$\frac{\partial}{\partial t}\int_\Omega \rho \, d\Omega = \int_\Omega \rho(\vec{\nabla} \cdot \vec{\mathbf{U}}) \, d\Omega \tag{2.4}$$

where ρ and $\vec{\mathbf{U}}$ are the specific mass and velocity vectors, respectively.

From equation (2.3), the differential form of the Navier-Stokes equations may be written in the following conservation form

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0 \qquad (i = 1, 2, 3) \tag{2.5}$$

where u_i is the velocity component in i direction of cartesian coordinates ($i = 1, 2, 3$).

2.2.2 Conservation of Momentum

For conservation of momentum, the conserved quantity is considered as the momentum, $\rho\vec{\mathbf{U}}$. Note that, the momentum flux is a tensor because of the vector nature of the momentum. The complete form of the conservation equation consists of both convective and diffusive parts and the total flux can be considered as the summation of these two parts.

The variation of momentum within the domain must be equal to the net contributions from any sources. In the absence of chemical reactions, the integral form of this equilibrium may be expressed by using equation (2.2) as

$$\frac{\partial}{\partial t}\int_{\Omega} \rho\vec{\mathrm{U}}\,d\Omega + \int_{\Omega} \rho\vec{\mathrm{U}}(\vec{\nabla}\cdot\vec{\mathrm{U}})\,d\Omega - \int_{\Omega} (\vec{\nabla}\cdot\vec{\sigma})\,d\Omega = 0 \qquad (2.6)$$

where the convective part $\rho\vec{\mathrm{U}}(\vec{\nabla}\cdot\vec{\mathrm{U}})$ represents the amount of momentum transported by the fluid motion and the diffusive part $\nabla\cdot\vec{\sigma}$ describes the flux contribution due to the molecular motion and thermal reaction of the fluid acting on the boundary surface.

The vector of internal stresses, $\vec{\sigma}$ is dependent on the nature of the fluid. The components of total internal stresses for a Newtonian isotropic fluid can be taken as

$$\sigma_{ij} = \tau_{ij} - \delta_{ij}p \qquad (i,j = 1,2,3) \qquad (2.7)$$

where τ_{ij}, δ_{ij} and p are the shear stress, Kronecker delta and isotropic pressure, respectively.

The stress tensor, τ_{ij} expresses the viscous stresses in terms of velocity variations. It is assumed that the stress tensor is a linear function of the deformation tensor for a Newtonian viscous fluid. Also the relation between the stresses and deformation is independent of the direction for an isotropic fluid. Following these assumptions, the stress tensor components τ_{ij} can be written as

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \lambda\,\delta_{ij}\,\frac{\partial u_k}{\partial x_k} \qquad (i,j,k = 1,2,3) \qquad (2.8)$$

To eliminate λ in the present analysis, it is assumed that the Stokes hypothesis is valid so that λ and the viscosity coefficient of the fluid μ are related by

$$\lambda = -\frac{2}{3}\,\mu \tag{2.9}$$

Hence, by applying the divergence theorem to the surface integrals, for laminar viscous flow, the conservation of momentum can be written in the component differential form

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}\left(\,\rho u_i u_j + p\,\delta_{ij} - \tau_{ij}\,\right) = 0 \qquad (i,j = 1,2,3) \tag{2.10}$$

2.2.3 Conservation of Energy

The conservation of energy implies that the total energy per unit mass E is equal to the summation of the action of work on a body and surface forces plus the variation of energy due to any heat flux vector $\vec{\mathrm{q}}_h$ entering inside the domain Ω. A mathematical representation of the above statement is

$$\frac{\partial}{\partial t}\int\limits_{\Omega} \rho E\, d\Omega + \int\limits_{\Omega} \rho E(\vec{\nabla}\cdot\vec{\mathrm{U}})\, d\Omega - \int\limits_{\Omega} \vec{\mathrm{U}}(\vec{\nabla}\cdot\vec{\sigma})\, d\Omega + \int\limits_{\Omega} (\vec{\nabla}\cdot\vec{\mathrm{q}}_h)\, d\Omega = 0 \tag{2.11}$$

The total energy per unit mass, E is defined as the sum of the internal energy e and the kinetic energy,

$$E = e + \frac{1}{2}\,(\,\vec{\mathrm{U}}\cdot\vec{\mathrm{U}}\,) \tag{2.12}$$

The components of the heat flux vector $\vec{\mathrm{q}}_h$ can be adopted by Fourier's law of heat conduction. It relates the heat flux to temperature gradients as follows

$$q_i = -k\,\frac{\partial T}{\partial x_i} \qquad (i = 1,2,3) \tag{2.13}$$

The conservation of energy equation can be written in differential component form as $(i, j = 1, 2, 3)$

$$\frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_j}\left(\rho\,(Eu_j + p) \; - \tau_{ij}\, u_i + q_j \right) = 0 \tag{2.14}$$

If the fluid is a perfect gas, then the pressure p is related to the temperature T through the equation of state,

$$p = \rho RT \tag{2.15}$$

with R as the gas constant. Assuming further that the gas is calorically perfect (i.e. constant specific heat C_p and C_v), then the following relations exist [2,3]

$$\begin{aligned} e &= C_v T \\ C_v &= \frac{R}{\gamma - 1} \\ \gamma &= \frac{C_p}{C_v} \end{aligned} \tag{2.16}$$

where C_v is the specific heat at constant volume, C_p is the specific heat at constant pressure and γ is the ratio of the specific heats. For air at standard conditions, $\gamma = 1.4$ and $R = 287\, m^2/s^2\, K$.

Combining equations (2.12), (2.15) and (2.16), the following equations are obtained which relate the pressure p and temperature T to the variables $(\rho, \vec{\mathbf{U}}, E)$

$$\begin{aligned} p &= (\gamma - 1)\rho \left[E - \frac{1}{2}\,(\, \vec{\mathbf{U}} \cdot \vec{\mathbf{U}}\,) \right] \\ T &= \frac{p}{C_v \rho(\gamma - 1)} \end{aligned} \tag{2.17}$$

It is to be noted that for a fluid which cannot be considered perfect, the required equations of state can be found in the form of tables, charts or curve fits.

2.3 Vector Form of the Navier-Stokes Equations

For numerical formulations, it is often convenient to write the Navier-Stokes equations in a compact vector form. In this form, the Navier-Stokes equations in the absence of body forces may be written as

$$\frac{\partial \mathbf{W}}{\partial t} + \left(\frac{\partial \mathbf{f}^c}{\partial x_1} + \frac{\partial \mathbf{g}^c}{\partial x_2} + \frac{\partial \mathbf{h}^c}{\partial x_3}\right) - \left(\frac{\partial \mathbf{f}^d}{\partial x_1} + \frac{\partial \mathbf{g}^d}{\partial x_2} + \frac{\partial \mathbf{h}^d}{\partial x_3}\right) = 0 \qquad (2.18)$$

where

$$\mathbf{W} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho E \end{bmatrix}$$

$$\mathbf{f}^c = \begin{bmatrix} \rho u_1 \\ \rho {u_1}^2 + p \\ \rho u_1 u_2 \\ \rho u_1 u_3 \\ \rho u_1 E + p \end{bmatrix} \qquad \mathbf{f}^d = \begin{bmatrix} 0 \\ \tau_{11} \\ \tau_{12} \\ \tau_{13} \\ u_1\tau_{11} + u_2\tau_{12} + u_3\tau_{13} - q_1 \end{bmatrix}$$

$$\mathbf{g}^c = \begin{bmatrix} \rho u_2 \\ \rho u_2 u_1 \\ \rho {u_2}^2 + p \\ \rho u_2 u_3 \\ \rho u_2 E + p \end{bmatrix} \qquad \mathbf{g}^d = \begin{bmatrix} 0 \\ \tau_{21} \\ \tau_{22} \\ \tau_{23} \\ u_1\tau_{21} + u_2\tau_{22} + u_3\tau_{23} - q_2 \end{bmatrix}$$

$$\mathbf{h}^c = \begin{bmatrix} \rho u_3 \\ \rho u_3 u_1 \\ \rho u_3 u_2 \\ \rho {u_3}^2 + p \\ \rho u_3 E + p \end{bmatrix} \qquad \mathbf{h}^d = \begin{bmatrix} 0 \\ \tau_{31} \\ \tau_{32} \\ \tau_{33} \\ u_1\tau_{31} + u_2\tau_{32} + u_3\tau_{33} - q_3 \end{bmatrix} \qquad (2.19)$$

The vector **W** represents the vector of conserved variables, $\mathbf{f}^c$ $\mathbf{g}^c$ and $\mathbf{h}^c$ constitute the flux vectors of convective and pressure terms, while $\mathbf{f}^d$, $\mathbf{g}^d$ and $\mathbf{h}^d$ represent the diffusive flux vectors. Note that the first row of equation (2.18) corresponds to the continuity equation as given by equation (2.5). Similarly, the three other rows are given by equations (2.10) while the fifth row is related to the energy equation (2.14). The elements of the viscous stress tensor τ_{ij} can be calculated from equation (2.8) as

$$\tau_{12} = \mu\left(\frac{\partial u_2}{\partial x_1} + \frac{\partial u_1}{\partial x_2}\right)$$

$$\tau_{23} = \mu\left(\frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2}\right)$$

$$\tau_{31} = \mu\left(\frac{\partial u_1}{\partial x_3} + \frac{\partial u_3}{\partial x_1}\right)$$

$$\tau_{11} = \mu\left[\frac{4}{3}\frac{\partial u_1}{\partial x_1} - \frac{2}{3}\left(\frac{\partial u_2}{\partial x_2} + \frac{\partial u_3}{\partial x_3}\right)\right]$$

$$\tau_{22} = \mu\left[\frac{4}{3}\frac{\partial u_2}{\partial x_2} - \frac{2}{3}\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_3}{\partial x_3}\right)\right]$$

$$\tau_{33} = \mu\left[\frac{4}{3}\frac{\partial u_3}{\partial x_3} - \frac{2}{3}\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2}\right)\right] \tag{2.20}$$

2.4 Isothermal Equation of State

For low speed flows, the assumption that the changes in the temperature field are very small can be made and the flow can be considered isothermal. For this case, the equation of state (2.15) is replaced by

$$c^2 = \frac{\partial p}{\partial \rho} = \frac{\gamma p}{\rho} \tag{2.21}$$

where the constant c is the speed of sound and γ is the specific heat coefficient under constant pressure for perfect gas [1].

This assumption simplifies the set of equations by decoupling the energy equation (2.14) from other conservation laws. In this case, equations (2.5), (2.10) and (2.21) are the complete set of equations. Note that, for incompressible flow, since there is no change in specific mass ρ then the value of the speed of sound c tends to infinity.

2.5 The Incompressible Viscous Flow Equations

For a low speed flow the changes of specific mass ρ from the mean value ρ_0 are very small. The low speed flow is a flow in which the maximum velocity is less than three times of the speed of sound. In this case, unlike compressible flow, where all the equations are fully coupled, the energy equation (2.14) can be solved independently after establishing the velocity field.

In the case of truly incompressible flow, the changes in specific mass ρ are negligible. Therefore, by using the mean value of specific mass ρ_0, the following approximations can be made $(i, j = 1, 2, 3)$

$$\begin{aligned} \frac{\partial \rho}{\partial t} &\simeq 0 \\ \frac{\partial}{\partial x_j}(\rho u_j) &\simeq \rho_0 \frac{\partial}{\partial x_j}(u_j) \\ \frac{\partial}{\partial x_j}(\rho u_i u_j) &\simeq \rho_0 \frac{\partial}{\partial x_j}(u_i u_j) \end{aligned} \tag{2.22}$$

Taking these approximations to equation (2.18) and dividing both sides of the equations by ρ_0, the simplified vector form of the Navier-stokes equations can be obtained for incompressible flow as

$$\frac{\partial \mathbf{W}}{\partial t} + \left(\frac{\partial \mathbf{f}^c}{\partial x_1} + \frac{\partial \mathbf{g}^c}{\partial x_2} + \frac{\partial \mathbf{h}^c}{\partial x_3}\right) - \left(\frac{\partial \mathbf{f}^d}{\partial x_1} + \frac{\partial \mathbf{g}^d}{\partial x_2} + \frac{\partial \mathbf{h}^d}{\partial x_3}\right) = 0 \tag{2.23}$$

where

$$\mathbf{W} = \begin{bmatrix} 0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

$$\mathbf{f}^c = \begin{bmatrix} u_1 \\ {u_1}^2 + p' \\ u_1 u_2 \\ u_1 u_3 \end{bmatrix} \qquad \mathbf{f}^d = \begin{bmatrix} 0 \\ \tau'_{11} \\ \tau'_{12} \\ \tau'_{13} \end{bmatrix}$$

$$\mathbf{g}^c = \begin{bmatrix} u_2 \\ u_2 u_x \\ {u_2}^2 + p' \\ u_2 u_3 \end{bmatrix} \qquad \mathbf{g}^d = \begin{bmatrix} 0 \\ \tau'_{21} \\ \tau'_{22} \\ \tau'_{23} \end{bmatrix}$$

$$\mathbf{h}^c = \begin{bmatrix} u_3 \\ u_3 u_1 \\ u_3 u_2 \\ {u_3}^2 + p' \end{bmatrix} \qquad \mathbf{h}^d = \begin{bmatrix} 0 \\ \tau'_{31} \\ \tau'_{32} \\ \tau'_{33} \end{bmatrix} \qquad (2.24)$$

where $p' = p/\rho_0$ and $\tau_{ij}' = \tau_{ij}/\rho_0$ and the stress tensor τ_{ij} is the same as equation (2.8). Hence, in equations (2.8) and (2.9) μ can be replaced by the kinematic viscosity ν.

2.6 Incompressible Inviscid Flow Equations

For attached flows at large Reynolds numbers, the important viscous effects are confined to a thin layer in the vicinity of solid boundaries. If the boundary layer is very thin compared to the characteristic length of the flow field, then the interaction between the boundary layer and inviscid portion of the flow field is often neglected. In this case, the viscous terms can be dropped completely from the Navier-Stokes equations and the resulting equations are called the Euler equations which represent inviscid flow condition. The vector form of the governing equations for truly incompressible inviscid flow can be simplified to the following equation

$$\frac{\partial \mathbf{W}}{\partial t} + \left(\frac{\partial \mathbf{f}^c}{\partial x_1} + \frac{\partial \mathbf{g}^c}{\partial x_2} + \frac{\partial \mathbf{h}^c}{\partial x_3} \right) = 0 \qquad (2.25)$$

where the components $\mathbf{W}$, $\mathbf{f}^c$, $\mathbf{g}^c$ and $\mathbf{h}^c$ are given by equation (2.24)

As can be seen in both equations (2.23) and (2.25), by omitting the time derivative of the specific mass from the continuity equation, there is a time independent constraint on the momentum equations for both unsteady and steady cases. This condition causes one of the main difficulties in the numerical solution of the incompressible flow equations.

However, by considering constant entropy the isothermal equation of state, the time derivative of the specific mass can be replaced with a transient pressure term by using

$$\frac{\partial \rho}{\partial t} = \frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} = \frac{1}{c^2}\frac{\partial p}{\partial t} \tag{2.26}$$

In this case, for steady state conditions, the temporal terms are zero, and hence equations (2.23) and (2.25) are independent of c and the set of equations represents the incompressible flow equations. However, for time dependent cases, the local speed becomes very small comparing with the speed of sound c. This fact is more significant in very low speed regions, i.e. near stagnation points. Hence, in such a case, the speed of sound dominates the system of equations and makes the set of equations ill conditioned for numerical solution.

2.7 Boundary Conditions

In order to pose the problem in complete form, boundary conditions need to be provided along the whole boundaries of the domain of interest. In general, the domain boundary can be divided into three categories, the first is the wall boundary, the second is the flow domain symmetry boundary and the third is the flow boundary which must be specified to produce a bounded computational domain.

For viscous flows in general, the nature of the boundary conditions imposed on the wall boundaries are set on the basis of physical experiences. All known experiments for viscous flows indicate that the relative velocity between the fluid and solid wall is zero. This is called the no-slip condition and is expressed by

$$\begin{aligned} (\vec{\mathbf{U}} \cdot \vec{\mathbf{n}})_{wall} &= 0 \\ (\vec{\mathbf{U}} \cdot \vec{\mathbf{t}})_{wall} &= 0 \end{aligned} \tag{2.28}$$

where $\vec{\mathbf{U}} = (u_1, u_2, u_3)$, $\vec{\mathbf{n}} = (n_1, n_2, n_3)$ and $\vec{\mathbf{t}} = (t_1, t_2, t_3)$ are velocity vector, unit vector normal and unit vector tangential to the wall boundary, respectively.

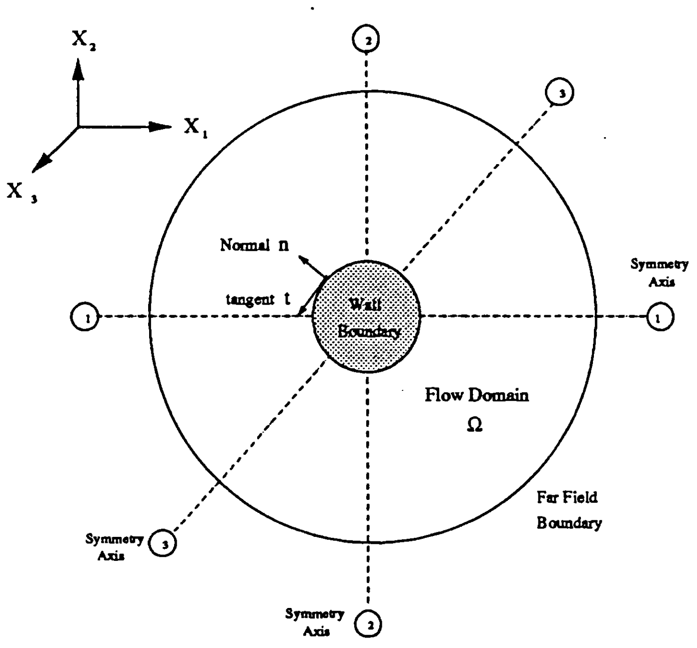

Figure 2.2. *Far field and Wall boundaries plus tangential and normal vectors*

For inviscid flow, the only boundary condition applied at the wall boundary is the flow tangency or no flow normal to the wall boundary and is expressed as

$$(\vec{\mathrm{U}} \cdot \vec{\mathrm{n}})_{wall} = 0 \tag{2.29}$$

For the flow domain symmetry boundaries the flow tangency can be imposed by considering no flow normal to the wall boundary. Therefore, the same principal as inviscid flow can be applied. Hence, equation (2.29) is valid for this type of boundary condition. Special care should be considered when perpendicular symmetry boundaries have to be used or the symmetry boundaries are connected to the wall boundaries.

The far field boundary conditions for the Navier-Stokes equations are more difficult to specify in a way that facilitates the computation. In practice the far field boundary is placed typically 12–15 reference length away from the body. The common approach to impose the flow boundary condition is to differentiate between inflow and outflow boundary conditions which can be determined easily by considering the sign of the dot product of the velocity and normal vectors at flow boundaries.

For external flows about a body in a uniform stream, provided that the far field boundary is placed far from the body the viscous terms at the far field boundaries are usually very small. Consequently, in these regions, the governing equations reduce to the Euler equations and hence various approaches devised for the treatment of inviscid flow boundaries can be used. The convective nature of the inviscid equations implies that information in the flow domain propagates along characteristic lines corresponding to the local wave directions.

For internal viscous flow computations, where open boundaries occur close to the wall boundary, the effects of the far field boundary condition on the implementation of the non-slip wall boundary condition and viscous gradients close to the solid wall cannot be ignored. The appropriate definition of the boundary conditions for such a case is not yet theoretically established. Some discussion is provided by Rudy and Strickwerda [6], Bayliss and Turkel [7]. A more detailed discussion of the far field boundary conditions for viscous flows can be found in references [2,5-8].

2.8 Summary

In this chapter, the integral and differential form of the Navier-Stokes equations has been reviewed using the conservation laws. In addition, the details of the mathematical formulation of the equations and the essential information to specify the properties of the proposed fluid are described. Then, possible simplifications to the flow equations have been discussed. The specific case of the equations for incompressible flow, which is applicable for low speed flows, is studied for both inviscid and viscous flows. Finally, the appropriate boundary conditions for wall, symmetry and far field boundaries are defined. For more detailed discussions on these equations, readers may be referred to the reference texts on fluid dynamics [1-4].

The problems associated with the solution of the set of incompressible flow equations cause some difficulties in the accurate computation of the velocity and pressure fields. Therefore, it is necessary to investigate techniques to modify the set of incompressible flow equations in a form which make them better conditioned for using established computational methods.

References

[1] C. Hirsch, 'Numerical Computation of Internal and External Flows', **1**, Fundamentals of Numerical Descritization, Wiley, (1989)

[2] C. Hirsch, 'Numerical Computation of Internal and External Flows', **2**, Computational Methods for Inviscid and Viscous Flows, Wiley, (1989)

[3] D. A. Anderson, J. C. Tannehill and R. H. Pletcher, 'Computational Fluid mechanics and Heat Transfer', Hemisphere Publishing Co. and McGraw-Hill Co., (1984).

[4] W. J. Minkowycz, E. M. Sparrow, G. E. Schneider., R. H. Pletcher.,'Handbook of Numerical Heat Transfer', John Wiley and Sons, Inc., (1988).

[5] R. Peyret and T. T. Taylor, 'Computational Methods for Fluid Flow', Springer-Verlag, (1983).

[6] D. Rudy and J. Strickwerda, 'A Non-reflecting Outflow Boundary Condition for Subsonic Navier-Stokes Calculation', J. of Comp. Physics, Vol 36, (1980).

[7] A. Bayliss and E. Turkel, 'Outflow Boundary Conditions for Fluid Dynamics', ICASE Report 80-21, (1980).

[8] M. Giles, 'Non-Reflecting Boundary Conditions for Euler Equations', CFDL-TR-88-1 Report, (1988).

[9] W. Y. Soh, 'Time Marching Solution of Incompressible Navier-Stokes Equations for Internal Flow', J. of Comp. Physics, **70**, pp. 232–52, (1987).

3

Formulation of the Incompressible Flow Equations

3.1 Introduction

For the condition of incompressible flow the changes in specific mass are negligible and, therefore, the time derivative in the continuity equation tends to zero. This implies a time independent constraint on the momentum equations which is a major problem in solving the Navier-Stokes equations due to the difficulty in extracting the pressure from the coupled continuity and momentum equations. The issues related to the solution of the incompressible flow equations under these conditions motivates this investigation on appropriate numerical solution methods.

In Section 3.2, various mathematical models for the solution of the incompressible flow equations are reviewed. Among them, the Artificial Compressibility approach is identified as one of the efficient techniques for resolving the problems associated with the solution of the incompressible form of the Navier-Stokes equations. This approach modifies the set of equations to pseudo unsteady form which are then suitable for steady state solution using standard numerical methods developed for solving the compressible form of Navier-Stokes equations. In Section 3.3, after a mathematical discussion about the Artificial Compressibility approach, hyperbolicity properties of the modified system of equations and its relevant parameters are studied. Finally, in Section 3.4, the non-dimensional form of the modified equations and the parameters of the adopted approach are described.

3.2 Review of Models for Simulating Incompressible Flow

A number of methods have been developed for the numerical solution of the incompressible flow equations using derived or primitive variable formulations. The application of each method depends upon the condition of the flow, i.e. inviscid/viscous, steady/transient, irrotational/rotational and two/three dimensional flows. They can be divided into two general categories of derived and primitive variable approaches. In this section, the most important techniques for numerical solution of the incompressible flow equations are reviewed.

3.2.1 Derived Variable Approach

The derived variable approach defines the Navier-Stokes equation (2.10) in terms of new dependent variables. These methods can be listed as, the Stream Function/Vorticity method [1], and the Vorticity/Velocity method [2]. For some particular applications, when the flow can be considered as steady, inviscid and irrotational Panel methods may be used [3].

The Stream Function/Vorticity method is a technique which casts the momentum equations in terms of a distribution of vorticity, ϑ. This form is obtained by taking the curl of the momentum equations and a derived Poisson equation for the stream function Ψ. The resulting set of equations for inviscid flow can be written in two-dimensional form as

$$\begin{aligned} &\frac{\partial \vartheta}{\partial t} + \left(u_1 \frac{\partial \vartheta}{\partial x_1} + u_2 \frac{\partial \vartheta}{\partial x_2} \right) = 0 \\ &\frac{\partial^2 \Psi}{\partial x_1^2} + \frac{\partial^2 \Psi}{\partial x_2^2} = -\vartheta \end{aligned} \qquad (3.1)$$

where

$$u_1 = \frac{\partial \Psi}{\partial x_2} \quad , \quad u_2 = -\frac{\partial \Psi}{\partial x_1}$$

and

$$\vartheta = \frac{\partial u_2}{\partial x_1} - \frac{\partial u_1}{\partial x_2} \tag{3.2}$$

The pressure can be computed separately from an equation, which is called the Poisson equation. This equation is obtained by taking the divergence of the momentum equation and introducing the divergence-free velocity condition of the continuity equation. This equation for inviscid flow is given by

$$\frac{\partial^2 p}{\partial x_1^2} + \frac{\partial^2 p}{\partial x_2^2} = \left[\frac{\partial u_1}{\partial x_1}\frac{\partial u_2}{\partial x_2} - \frac{\partial u_2}{\partial x_1}\frac{\partial u_1}{\partial x_2} \right] \tag{3.3}$$

The Vorticity/Velocity approach applies the vorticity equation for the computation of the vorticity field. A combination of continuity and the definition of vorticity, equation (3.2), can be used for the velocity field which for inviscid flow is described as

$$\begin{gathered} \frac{\partial^2 u_1}{\partial x_1^2} + \frac{\partial^2 u_1}{\partial x_2^2} = -\frac{\partial \vartheta}{\partial x_2} \\ \frac{\partial^2 u_2}{\partial x_1^2} + \frac{\partial^2 u_2}{\partial x_2^2} = \frac{\partial \vartheta}{\partial x_1} \end{gathered} \tag{3.4}$$

Again, the pressure can be computed separately using equation (3.3) as an additional equation.

Panel methods may be used only for the numerical solution of steady, inviscid and irrotational flow using Laplace equation in the velocity potential Φ namely

$$\frac{\partial^2 \Phi}{\partial x_1^2} + \frac{\partial^2 \Phi}{\partial x_2^2} = 0 \tag{3.5}$$

where

$$\frac{\partial \Phi}{\partial x_1} = u_1 \quad , \quad \frac{\partial \Phi}{\partial x_2} = u_2 \tag{3.6}$$

For the solution of this form of equation many standard solution techniques exist. One of these techniques is called the singularity method which can be developed using the linearity property of the equation. In this method, a linear superposition of known elementary flow fields, such as source singularities, is defined. The unknown coefficients of this linear superposition can be obtained by considering the condition that in the absence of wall suction or blowing the resultant velocity field satisfies the condition of zero normal velocity on solid wall surfaces. The Panel method can be used for the computation of the flow over three-dimensional complex geometries. By using this methods, a solution for the pressure on the body can be obtained without solving for the flow field throughout the domain.

Both the Stream Function/Vorticity and Vorticity/Velocity formulations have several restrictions and difficulties. Neither of them can easily be extended to three dimensions and there are difficulties in expressing boundary conditions for both of them. The separate equation for the solution of the pressure field is another weak point for these methods. Panel methods are restricted to the solution of steady, inviscid and irrotational flow. The problems associated with derived variable approaches motivate the use of primitive variable formulations.

3.2.2 Primitive Variable Approach

The primitive variable approach utilises the independent variables of velocity components and pressure. This requires the solution of equation (2.23) for viscous flow or equation (2.25) for inviscid flow. For incompressible flow, the time derivative of the specific mass in the continuity equation tends to zero. Hence, the time independent continuity equation serves as a constraint for the time dependent momentum equations which must be satisfied for all time. This fact precludes the use of the simple explicit time marching procedures for the solution of the equations of motion. However, there are three common techniques for overcoming this problem. These methods are the Pressure Poisson technique, Fractional Step technique, the Penalty method and the Artificial Compressibility approach.

The first method which is called Pressure Poisson method [3] is able to solve the time dependent incompressible inviscid or viscous equations of motion. In this method, the momentum equation (2.10) is solved for the velocity components and the Poisson equation (3.3) is solved for pressure. Continuity is satisfied indirectly through the solution of the Poisson equation. The momentum equations (2.10) are marched in time utilising the boundary condition on the velocity. Then the Poisson equation (3.3) is solved for the pressure utilising the Neumann boundary conditions. This method can also be applied to stationary problems and is referred to as the Pressure Correction method [1]. For steady state computations, an explicit method can be applied to the momentum equations, followed by several iterations applied to the Poisson equation for the pressure. Since there is no primitive equation for pressure, the lack of natural boundary condition for pressure presents severe difficulties. However, there have been several attempt to overcome boundary condition problem of the pressure [30]. A brief summary of some of common pressure-based methods are given in Appendix E.

The second primitive variable approach is known as the Fractional Step or Velocity Correction method, which was first introduced by Chorin [4]. Numerically, this technique splits the operators of the Navier-Stokes equations into two parts, which are solved in two separate stages [4,5]. This technique starts by solving an approximate velocity field utilising the momentum equations for incompressible flow in which the pressure term is decomposed. The fractional momentum equation (2.10) can be written as

$$\frac{(\Delta u_i)^\diamond}{\Delta t} + \frac{\partial}{\partial x_j}\left(u_i u_j - \tau'_{ij} \right)^\diamond = 0 \qquad (i,j = 1,2,3) \tag{3.7}$$

where x_j, u_j, and τ'_{ij} are the cartesian coordinates, velocity components and shear stress, respectively. Here, '$\diamond$' denotes the variables at the first stage. Then, considering '$\diamond\diamond$' for the variables at the second stage, they follow as

$$\begin{aligned} (u_i)^{n+\Theta_1} &= (u_i)^n + \Theta_1 \left((\Delta u_i)^\diamond + (\Delta u_i)^{\diamond\diamond} \right) \\ (p')^{n+\Theta_2} &= (p')^n + \Theta_2 \left((\,0\,)^\diamond + (\Delta p')^{\diamond\diamond} \right) \end{aligned} \tag{3.8}$$

where $p' = p/\rho$ is the isotropic pressure per specific mass. The parameters Θ_1 and Θ_2 are two relaxation parameters in the range of 0.5 to 1 [12]. The computation for the set of incompressible equations can be completed by rearranging the rest of the momentum equation in conjunction with mass conservation equation (2.5) as $(i, j = 1, 2, 3)$

$$\begin{aligned} \frac{1}{c^2}\frac{(\Delta p')^{\circ\circ}}{\Delta t} + \Theta_1 \frac{\partial}{\partial x_i}(\Delta u_i)^{\circ\circ} &= -\frac{\partial}{\partial x_i}(u_i)^n - \Theta_1 \frac{\partial}{\partial x_i}(\Delta u_i)^{\circ} \\ \frac{(\Delta u_i)^{\circ\circ}}{\Delta t} + \Theta_2 \frac{\partial}{\partial x_i}(\Delta p')^{\circ\circ} &= -\frac{\partial}{\partial x_i}(p')^n \end{aligned} \tag{3.9}$$

The approximate velocity field is solved from the first part, equation (3.7), then the pressure field can be solved by inserting the approximate velocity into the the second part, equations (3.9). Since, for incompressible flow the speed of sound c tends to infinity, it is suggested to utilise the artificial speed of sound. Therefore, the time accuracy of the system of equations is destroyed and the method can be applied only for steady state flow. However, time stepping can be applied as iterative marching to the steady state condition. Appropriate time step limits can be found by eliminating $(\Delta u_i)^{\circ\circ}$ from equation (3.9) [12]. As can be seen, two iterative procedures are necessary for every single stage of the computations, which makes the method computationally expensive.

The third method is Penalty method which adds a value of ηp_η to the right hand side of the continuity equations where subscript η denotes the penalty solution. Eliminating the pressure from the momentum equations using the new continuity equations, we get a system involving the velocity only. After velocity is calculated from the momentum equations, one can determine the pressure p_η from the new continuity equation. An iterative procedure can be set up such that the difference between the penalty solution and the exact solution can be as small as desired for any $0 < \eta < 1$ [30-32]. In Penalty method pressure is eliminated by penalizing the continuity equations, although it is retained in the boundary conditions. Involving only the velocities, a considerable saving in computing time and computer memory is achieved due to direct coupling between velocity and pressure.. However, in many engineering applications the pressure may be the most important design parameter, but the pressure is recovered by using perturbed continuity equation exhibits oscillations due to the ill-conditioned pressure matrix. Another disadvantage is the penalty parameter which for small values, causes loss of accuracy and for too large value sometimes prevents convergence to the solution [32].

The fourth technique is the Artificial Compressibility approach which was proposed by Chorin [14]. In this approach, the incompressible equations are cast in pseudo-unsteady form. The approach can be applied to the viscous and inviscid set of equations in two and three dimensions and it is attractive because of the efficiency of the algorithm. The Artificial Compressibility approach has received the attention of several workers utilising Finite Difference [15-24], Finite Element [25] and Finite volume [26-29] Methods. Recently, some workers [24,26] applied the approach to solve unsteady problems. Although, there are some similarities between the second and fourth techniques, the third technique has the advantage of direct coupling between the pressure and velocity fields as the equations are advanced in time. In this work, the Artificial Compressibility approach is adopted for the computation of two and three-dimensional inviscid and laminar viscous incompressible flows.

Recently considerable progress has been made to solve incompressible form of Navier-Stokes equations using various finite element method. A brief review of some of these method is given in Appendix **F**.

3.3 Artificial Compressibility Approach

The fundamental idea of Chorin's Artificial Compressibility approach is to introduce an unsteady pressure term into the continuity equation, which resolves the problem of the time independent velocity constraint on the momentum equation. When the solution converges to steady state conditions, the pseudo temporal derivative of pressure tends to zero, thus the set of equations satisfies the incompressible equations. The Artificial Compressibility approach allows techniques developed for compressible flow simulations to be used for solving the stationary incompressible equations. In this approach an artificial compressibility term under the form of the pressure time derivative is added to the continuity equation. The added transient pressure term is formed similar to equation (2.26). In the limit of incompressible flow the local velocities may be very small compare to the speed of sound c. Using an empirical parameter β instead of c resolves the problem of dominating sound speed in the system of equations. Hence, the modified set of incompressible viscous flow equations may be written as $(i, j = 1, 2, 3)$

- Conservation of mass :

$$\frac{1}{\beta^2}\frac{\partial p}{\partial t} + \frac{\partial u_j}{\partial x_j} = 0 \tag{3.10}$$

- Conservation of momentum :

$$\frac{\partial u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(u_i u_j + p\,\delta_{ij} - \sigma_{ij} \right) = 0 \tag{3.11}$$

The set of equations may be written in vector form as

$$\frac{\partial \mathbf{W}}{\partial t} + \mathbf{P}\Big[\Big(\frac{\partial \mathbf{f}^c}{\partial x_1} + \frac{\partial \mathbf{g}^c}{\partial x_2} + \frac{\partial \mathbf{h}^c}{\partial x_3} \Big) - \Big(\frac{\partial \mathbf{f}^d}{\partial x_1} + \frac{\partial \mathbf{g}^d}{\partial x_2} + \frac{\partial \mathbf{h}^d}{\partial x_3} \Big) \Big] = 0 \tag{3.12}$$

where the vectors $\mathbf{W}$, $\mathbf{f}^c$, $\mathbf{g}^c$, $\mathbf{h}^c$, $\mathbf{f}^d$, $\mathbf{g}^d$ and $\mathbf{h}^d$ are the same as in equation (2.24), and the preconditioning matrix $\mathbf{P}$ is

$$\mathbf{P} = \begin{bmatrix} \beta^2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.13}$$

The preconditioning matrix, $\mathbf{P}$ can be used for casting the system of equations in hyperbolic-parabolic form. Here, β^2 may be viewed as a relaxation parameter for pseudo time marching of the pressure iteration. Analogous to the compressible flow equations (2.21), the equation of state is considered as

$$\beta^2 = \frac{\gamma p}{\rho_0} \tag{3.14}$$

where ρ_0 is the mean value of specific mass, γ is the ratio of the specific heats and the parameter β^2 can be called the Artificial Compressibility parameter.

Note that, because of the added pseudo transient pressure term, the proposed system of equations can only be applied for steady state flow. In this case, the added temporal derivatives of the pressure tend to zero when the solution converges to the steady state condition. Thus, the set of equations satisfy the incompressible form of the Navier-Stokes equations.

3.3.1 Mathematical Character of the Modified Equations

Application of the Artificial Compressibility approach casts the system of equations in a form similar to the compressible Navier-Stokes equations. Since the effort to develop methods to solve the compressible Navier-Stokes equations has been very intensive, a substantial assortment of computer based methods exist which can be used for the solution of the modified set of equations. Practically, most of these methods are based upon the hyperbolic character of the convective part of the equations. Here, the hyperbolicity of the modified set of equation is studied using their differential form. Rizzi and Erikson [26] investigated the effect of the preconditioning matrix **P** on the mathematical character of the system of equations by writing the inviscid form of equation (3.12) in quasi-linear form, namely

$$\frac{\partial \mathbf{W}}{\partial t} + \mathbf{G}_1 \frac{\partial \mathbf{W}}{\partial x_1} + \mathbf{G}_2 \frac{\partial \mathbf{W}}{\partial x_2} + \mathbf{G}_3 \frac{\partial \mathbf{W}}{\partial x_3} = 0 \tag{3.15}$$

where

$$\mathbf{G}_1 = \mathbf{P}\frac{\partial \mathbf{f}}{\partial \mathbf{W}} = \begin{bmatrix} 0 & \beta^2 & 0 & 0 \\ 1 & 2u_1 & 0 & 0 \\ 0 & u_2 & u_1 & 0 \\ 0 & u_3 & 0 & u_1 \end{bmatrix}$$

$$\mathbf{G}_2 = \mathbf{P}\frac{\partial \mathbf{g}}{\partial \mathbf{W}} = \begin{bmatrix} 0 & 0 & \beta^2 & 0 \\ 0 & u_2 & u_1 & 0 \\ 1 & 0 & 2u_2 & 0 \\ 0 & 0 & u_3 & u_2 \end{bmatrix} \tag{3.16}$$

$$\mathbf{G}_3 = \mathbf{P}\frac{\partial \mathbf{h}}{\partial \mathbf{W}} = \begin{bmatrix} 0 & 0 & 0 & \beta^2 \\ 0 & u_3 & 0 & u_1 \\ 0 & 0 & u_3 & u_2 \\ 1 & 0 & 0 & 2u_3 \end{bmatrix}$$

In this case, a matrix **H** can be formed by taking a linear combination of individual flux Jacobian matrices, such as

$$\mathbf{H} = \mathbf{G}_1\zeta_1 + \mathbf{G}_2\zeta_2 + \mathbf{G}_3\zeta_3 \tag{3.17}$$

where the variables ζ_1, ζ_2 and ζ_3 are arbitrary real variables, representing the slope of the characteristic waves of the system.

The eigenvalues of the matrix **H**, which can be taken as characteristic convection waves of the Jacobian matrices of the modified set of equations, are found as

$$\lambda_1 = \mathrm{U} \qquad \lambda_2 = \mathrm{U} + a \qquad \lambda_3 = \mathrm{U} + a \qquad \lambda_4 = \mathrm{U} - a \tag{3.18}$$

where

$$\mathrm{U} = u_1\zeta_1 + u_2\zeta_2 + u_3\zeta_3 \tag{3.19}$$

and

$$a^2 = \mathrm{U}^2 + \beta^2({\zeta_1}^2 + {\zeta_2}^2 + {\zeta_3}^2) \tag{3.20}$$

Because the eigenvalues of the matrix **H** are real, it can be stated that the inviscid part of equation (3.12) is guaranteed to be hyperbolic. The convergence rate and stability of the numerical scheme is dictated by the slowest and the fastest characteristic convection waves, respectively. Equation (3.20) highlights the important effect of the choice of β^2 on the convergence and stability of the numerical scheme. Notice that the eigenvalues can be adjusted according to the value of the parameter β^2.

3.3.2 Artificial Compressibility Parameter β^2

Initially, the Artificial Compressibility parameter, β^2, was constant in the preconditioning matrix proposed by Chorin. There have been several attempts to give a precise value for this parameter. In the Finite Difference community, Chang and Kwak [16-18] tried to find the lower limit, upper limit, and optimum magnitude of β^2 for both laminar and turbulent flow problems. Kwak et al [19] suggested a bound for parameters and Soh [20] investigated limits of the parameter and appropriate boundary conditions.

Rizzi and Eriksson [26], in their Finite Volume formulation, verified that the hyperbolicity of the problem may be degraded if the different characteristic convection waves in the system, equation (3.18), become too disparate by specification of an inappropriate value for the parameter β^2. By two-dimensional measurements of the condition of the system of equations for any specified $\vec{\mathrm{U}}$ and β^2, they verified that the condition of the system of equations depends upon the ratio of $C_{\beta^2} = \beta^2/(\vec{\mathrm{U}} \cdot \vec{\mathrm{U}})$. Finally, they concluded that with the value of $C_{\beta^2} > 1$ the pressure waves dominate over the convection waves and the system is less directionally dependent and better conditioned. Hence, it was concluded that for solving the flow field the parameter β^2 need not to be a global constant. Then, in order to adapt the scheme to the concept of local time stepping, it was suggested that β^2 be computed to be proportional to the local velocity squared as

$$\beta^2 = \textbf{Max.} \; [\; \beta^2_{min} \; , \; C_\beta(\vec{\mathbf{U}} \cdot \vec{\mathbf{U}}) \;] \tag{3.21}$$

where β^2_{min} and C_β are the lower bound of β^2 and the constant ratio C_{β^2}, respectively. It is shown that, the hyperbolicity of the system of equations depends upon the value of C_{β^2}. After examining several values for this ratio, a range of $1 < C_\beta < 5$ was suggested for this ratio and $\beta^2_{min} = 0.3$ as the lower limit of the Artificial Compressibility parameter.

However, Turkel [15] after some mathematical considerations, suggested a more general preconditioning matrix, **P** which contains modifications not only to the continuity equation, but also to the unsteady term of the momentum equations. Thus, the preconditioning matrix is taken to be of the following form

$$\mathbf{P} = \begin{bmatrix} \beta^2 & 0 & 0 & 0 \\ (\alpha+1)u_1 & 1 & 0 & 0 \\ (\alpha+1)u_2 & 0 & 1 & 0 \\ (\alpha+1)u_3 & 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

where the parameters α and β^2 have to be found in such a way that render the system better conditioned for numerical solution. Following the same procedure for ensuring the hyperbolicity of the new system of equations, Turkel gave the definition of β^2 as

$$\begin{aligned} &for \;\; \alpha < 1 \;\; \beta^2 = \;\; max. \; [\; \beta^2_{min} \; , \; (2-\alpha)(\,\vec{\mathbf{U}} \cdot \vec{\mathbf{U}}\,) \;] \\ &for \;\; \alpha \geq 1 \;\; \beta^2 = \;\; max. \; [\; \beta^2_{min} \; , \; \alpha(\,\vec{\mathbf{U}} \cdot \vec{\mathbf{U}}\,) \;] \end{aligned} \tag{3.23}$$

Finally, Turkel concluded that the best results for the governing equations can be obtained using equation (3.23) with $\alpha = 1$ and $C_\beta = 1$. More details of this approach and the extension to three dimensions may be found in Appendix **A**.

Dreyer [27] applied Turkel's preconditioning matrix using the Finite Volume formulation by using equation (3.23) for the parameter β^2 for the simulation of incompressible flow on two dimensional triangular meshes,. It can be concluded from his work that $-1 \leq \alpha \leq 1$ and values of $0.2 \leq C_\beta \leq 6.0$ should be used.

Note that, by considering $\alpha = -1$ in equation (3.23), Turkel's approach leads to the scheme of Rizzi and Eriksson which requires less computational effort. Hence, the preconditioning matrix proposed by Rizzi and Eriksson is adopted for the present algorithm using expression (3.21) for the parameter β^2. This formulation gives Chorin's initial Artificial Compressibility approach with $\beta^2 = Constant$.

3.4 Non-dimensional Form of the Modified Equations

For practical purposes, it is often convenient to write equations (3.12) in a non-dimensional form. Also, by non-dimensionalising the equations, the flow variables are normalised so that their values fall between certain prescribed limiting values. For the present work, the following non-dimensionalisation has been adopted [1-3,14-16]

$$\begin{aligned} t^* &= \frac{t\,|\vec{\mathrm{U}}_0|}{\ell} & p^* &= \frac{p}{\rho_0|\vec{\mathrm{U}}_0|^2} & & \\ u_i^* &= \frac{u_i}{|\vec{\mathrm{U}}_0|} & x_i^* &= \frac{x_i}{\ell} & (i &= 1,2,3) \\ \rho^* &= \frac{\rho}{\rho_0} & \mu^* &= \frac{1}{\Re} & & \end{aligned} \tag{3.24}$$

where '*' denotes a non-dimensional value, free stream values are denoted by the index '0' and ℓ is the reference length. The Reynolds Number is defined by

$$\Re = \frac{\rho_0\,\ell\,|\vec{\mathrm{U}}_0|}{\mu_0} \tag{3.25}$$

Note that, because of conditions pertaining to low speed flow, i.e. the isothermal conditions, there is no change in viscosity μ_0 so the Reynolds number remains constant throughout the flow domain.

If this non-dimensionalisation is applied to the modified equations, the resulting system of equations may be written as

$$\frac{\partial \mathbf{W}^*}{\partial t^*} + \mathbf{P}^*\Big[\big(\frac{\partial \mathbf{f}^{c*}}{\partial x_1{}^*} + \frac{\partial \mathbf{g}^{c*}}{\partial x_2{}^*} + \frac{\partial \mathbf{h}^{c*}}{\partial x_3{}^*}\big) - \big(\frac{\partial \mathbf{f}^{d*}}{\partial x_1{}^*} + \frac{\partial \mathbf{g}^{d*}}{\partial x_2{}^*} + \frac{\partial \mathbf{h}^{d*}}{\partial x_3{}^*}\big)\Big] = 0 \qquad (3.26)$$

where the vector of conserved variables $\mathbf{W}^*$, preconditioning matrix

$\mathbf{P}^*$, connective flux vectors $\mathbf{f}^{c*}$, $\mathbf{g}^{c*}$ and $\mathbf{h}^{c*}$ and viscous flux vectors $\mathbf{f}^{d*}$, $\mathbf{g}^{d*}$ and $\mathbf{h}^{d*}$ are given by .

$$\mathbf{W}^* = \begin{bmatrix} p^* \\ u_1^* \\ u_2^* \\ u_3^* \end{bmatrix} \qquad \mathbf{P}^* = \begin{bmatrix} \beta^{*2} & 0 & 0 & 0 \\ 0 & 1 & 0. & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{f}^{c*} = \begin{bmatrix} u_1^* \\ u_1^{*2} + p^* \\ u_1^* u_2^* \\ u_1^* u_3^* \end{bmatrix} \quad \mathbf{g}^{c*} = \begin{bmatrix} u_2^* \\ u_2^* u_1^* \\ u_2^{*2} + p^* \\ u_2^* u_3^* \end{bmatrix} \quad \mathbf{h}^{c*} = \begin{bmatrix} u_3^* \\ u_3^* u_1^* \\ u_3^* u_2^* \\ u_3^{*2} + p^* \end{bmatrix}$$

$$\mathbf{f}^{d*} = \begin{bmatrix} 0 \\ \tau^*{}_{11} \\ \tau^*{}_{12} \\ \tau^*{}_{13} \end{bmatrix} \quad \mathbf{g}^{d*} = \begin{bmatrix} 0 \\ \tau^*{}_{21} \\ \tau^*{}_{22} \\ \tau^*{}_{23} \end{bmatrix} \quad \mathbf{h}^{d*} = \begin{bmatrix} 0 \\ \tau^*{}_{31} \\ \tau^*{}_{32} \\ \tau^*{}_{33} \end{bmatrix} \qquad (3.27)$$

with the parameter β^{2*} defined as a function of the local velocity vector

$$\beta^{2*} = \ \textbf{Max.}\ [\ \beta^2_{min}\ ,\ C_\beta(\vec{\mathbf{U}}^* \cdot \vec{\mathbf{U}}^*)\ ] \qquad (3.28)$$

The shear stresses can be considered as

$$\tau^*{}_{ij} = \frac{\tau_{ij}}{\rho_0|\vec{\mathbf{U}}_0|} \qquad (i,j = 1,2,3) \qquad (3.29)$$

We can rewrite the dimensionless form of the stress tensor τ_{ij} described by equation (2.8) as

$$\tau^*_{ij} = \mu^*\Big(\frac{\partial u_i^*}{\partial x_j^*} + \frac{\partial u_j^*}{\partial x_i^*}\Big) + \lambda\,\delta_{ij}\,\frac{\partial u_i^*}{\partial x_i^*} \qquad (i,j = 1,2,3) \qquad (3.30)$$

Note that, by omitting the '*', the non-dimensional form of the equation is identical to the original dimensional form (except for the time derivative of pressure in the continuity equation). For convenience, the asterisks can normally be dropped from the non-dimensional equations.

3.5 Summary

In this chapter, after a review of various methods for solving the incompressible flow equations, the Artificial Compressibility approach was chosen. This technique modifies the set of incompressible equations by introducing a pseudo time derivative pressure term into the continuity equation. In the set of modified equations, direct coupling of the pressure and velocity fields make them well conditioned to apply explicit numerical methods previously developed for the solution of the compressible form of the Navier-Stokes equations. This additional transient term affects the time accuracy of the set of equations, and hence, the set of modified equation can be used for solving the steady state flows. In this case, the time marching procedure can be viewed as an iterative relaxation to the steady state condition.

From the mathematical study of the hyperbolicity of the convective part of the set of the modified equations, it was concluded that the system of equations can be better conditioned for numerical solution by using appropriate values for the parameter β^2. There is an effect of this parameter on the stability and convergence of the numerical solution which appears in the eigenvalue of the convective part of the system of equations. In order to have a better convergence rate, β^2 can be scaled using the local velocity in the computational domain.

In summary, the mathematical model for the present incompressible flow solution algorithm consists of a set of non-dimensional equations given by equation (3.26), in conjunction with the relaxation parameter β^2 defined by equation (3.28). Following the suggestion of other workers, the scaling coefficient C_β of the artificial compressibility parameter is considered in the range of $1 < C_\beta < 5$ and the lower limit for β^2 is considered as $0.2 < \beta^2 < 0.5$.

References

[1] D. A. Anderson, J. C. Tannehill and R. H. Pletcher, 'Computational Fluid mechanics and Heat Transfer', Hemisphere Publishing Co. and McGraw-Hill Co., (1984).

[2] R. Peyret and T. T. Taylor, 'Computational Methods for Fluid Flow', Springer-Verlag, (1983).

[3] C. Hirsch, 'Numerical Computation of Internal and External Flows', **1**, Fundamentals of Numerical Descritization, Wiley, (1989)

[4] A. Chorin , 'Numerical Solution of Navier-Stokes Equations' , Math. Comp. , **22**, pp 745–762 (1968).

[5] A. Chorin , 'On the Convergence of Discrete Approximations to the Navier-Stokes Equations' , Math. Comp. , **23**, pp 341–353 (1969).

[6] G. E. Schneider, G. D. Raithby, M. M. Yovanovich, 'Finite Element Analysis of Incompressible Fluid Flow Incorporating Equal Order Pressure and Velocity Interpolation', in Numerical Methods in Laminar and Turbulent Flow, C. Taylor, K. Morgan , C. A. Brebia , Pentech Press, Plymouth, (1978).

[7] M. Kavahra, K. Ohmiya, 'Finite Element Analysis of Incompressible Density Flow Using Velocity Correction Method', Int. J. for Num. Meth. in Fluids, **5**, pp 981–993 , (1985).

[8] O. C. Zienkiewicz, K. Morgan and B. V. Satya Sai, 'A General Algorithm for Compressible and Incompressible Flow-Part I and II', Int. J. for Num. Meth. in Fluids, **20**, pp 887–913 , (1995).

[9] G. Comini and S. Del Guidice, 'Finite Element Solution of Incompressible Navier-Stokes Equations, Numerical Heat Transfer, pp 463–478, (1972).

[10] J. Szmelter, 'Computational Methods for Compressible and Incompressible Fluid Mechanics' , Ph.D Thesis, University of Wales, Swansea, (1989).

[11] O. C. Zienkiewicz, J. Szmelter, J. Paraire, 'Compressible and Incompressible Flow, An Algorithm for All Seasons', Comp. Math. Appl. Mech. Eng. , **78**, pp.105–121; (1990).

[12] J. Wu, 'Compressible and Incompressible Flow problems – A New Finite Element Algorithm and Adaptivity' ,Ph.D Thesis, University of Wales, Swansea, (1992).

[13] O. C. Zienkiewicz, K. Morgan and B. V. Satya Sai, 'A General Algorithm for Compressible and Incompressible Flow-Part I and II', Int. J. for Num. Meth. in Fluids, **20**, pp 887–913 , (1995).

[14] A. Chorin, 'A Numerical Method for Solving Incompressible Viscous Flow Problems', J. of Comp. Physics , **2**, pp. 12–26, (1967).

[15] E. Turkel, 'Preconditioned Method For Solving The Incompressible And Low Speed Compressible Equations', ICASE Report 86–14, NASA Langley Centre, (1986).

[16] J. L. C. Chang, D. Kwak, 'On The Method of Pseudo Compressibility for Numerically Solving Incompressible Flows', AIAA 22nd Aerospace Sciences Meeting, AIAA Paper 84-0252, (1984).

[17] J. L. C. Chang, D. Kwak, S. C. Dao, R. Rosen, 'A Three-dimensional Incompressible Flow Simulation Method and its Application to Space-Shuttle Main Engine-Part I-Laminar Flow', AIAA 3rd Aerospace Meeting, AIAA Paper 85-0175, (1985a).

[18] J. L. C. Chang, D. Kwak, S. C. Dao, R. Rosen, 'A Three-dimensional Incompressible Flow Simulation Method and its Application to Space-Shuttle Main Engine-Part II-Turbulent Flow', AIAA 18th Fluid Dynamics ,Plasma-dynamics Conference, AIAA Paper 85-1670, (1985b).

[19] D. Kwak, L. J. Chang, S. P. Shanks, S. Chakaravarthy, 'A Three-dimensional Incompressible Navier-Stokes Flow Solver Using Primitive Variables' , AIAA Journal, **24**, pp. 390–6, (1986).

[20] W. Y. Soh, 'Time Marching Solution of Incompressible Navier-Stokes Equations for Internal Flow', J. of Comp. Physics, **70**, pp. 232–52, (1987).

[21] S. E. Rogers and D. Kwak, 'Upwind Differencing Scheme for Time-Accurate Incompressible Navier-Stokes Equations', AIAA paper 87-1137, (1987).

[22] C. L. Merkle, M. Athaval, 'Time-Accurate Unsteady Incompressible Flow Algorithm Based on Artificial Compressibility ', AIAA Paper 87-1137, (1987).

[23] W. Y. Soh, J. W. Goodrich, 'Unsteady Solution of Incompressible Flow Navier-Stokes Equations', J. of Comp. Physics, **79**, pp. 113–114, (1988).

[24] S. E. Rogers, 'A Comparison of Implicit Schemes for the Incompressible Navier-Stokes Equations With Artificial Compressibility', AIAA 95-0567 paper, (1995).

[25] J. Peraire, K. Morgan and J. Piero, 'The Simulation of 3D Incompressible Flows Using unstructured Grids', Frontiers of Computational Fluid Dynamics, John Wiley and Sons, pp 281–301, (1994).

[26] A. Rizzi, L. E. Erikson, 'Computation of Inviscid Incompressible Flow with Rotation', Journal of Fluid Mechanics, **153**, pp. 275–312, (1985).

[27] J. Dreyer, 'Finite Volume Solution to the Steady Incompressible Euler Equations on Unstructured Triangular Meshes', M.Sc. Thesis, MAE Dept., Princeton University, (1990).

[28] J. Farmer, L.Martinelli, A. Jameson, 'A Fast Multi-grid Method for Solving Incompressible Hydrodynamic Problems with Free Surfaces ', AIAA Paper 93-0767, (1993).

[29] A. Belov, L.Martinelli, A. Jameson, 'A New Implicit Algorithm with Multi-grid for Unsteady Incompressible Flow Calculations', AIAA Paper 95-0049, (1995).

[30] R. Peyret, 'Computational Fluid Mechanics', Academic Press INC, (1996).

[31] R. Temam, ' Navier-Stokes Equations', North holland, Amesterdam, (1977).

[32] S.K. Hannani, M Stanislas, P. Dupont, 'Incompressible Navier-Stokes Computations with SUPG and GLS Formulations', Compt. Meth. in Appl. Mech. Eng., **124**, pp 153–170, (1995).

4

Finite Volume Formulation

4.1 Introduction

In computer-based methods for solving fluid flow problems, a limited number of points is specified to represent the domain of interest. Then, the assumption is made that the solution of the equations at these points presents the solution of the equations within the entire field. If an appropriate solution method for the set of governing equations can be devised, the computational approach offers a powerful means of studying fluid flow. Such a solution can be achieved by application of the Finite Volume formulation which is based upon the integral form of the Navier-Stokes equations. This method is a powerful approach for computation. Furthermore, it can be efficient if appropriate data structures can be used. Using a side-based algorithm, considerable flexibility and computational efficiency can be incorporated into the method. Here, the term *side* is applied to edges of triangular cells and faces of tetrahedral cells of two and three-dimensional meshes, respectively.

This chapter presents an algorithm for computing the solution of the flow equations using the Finite Volume Method. In Section 4.2, after a brief reference to some concepts of the Finite Volume Method, the approach used for the discretisation of the flow equations is reviewed. A cell vertex scheme is chosen and the concept of overlapping control volumes is discussed. Then, discretisation of the integral form of Navier-Stokes equations using a cell-vertex approach is studied in Section 4.3 and the essential formulations for the computation of side areas and volume of the control volumes are presented. For the sake of simplicity and ease of understanding, most of the figures throughout this chapter are explained in a two-dimensional framework.

4.2 The Finite Volume Integration

A brief introduction to some of the important concepts of the Finite Volume Method is presented in this section. Although a more detailed discussion on the method can be found elsewhere [1-4], it has been considered appropriate to include this introductory section with the objective of providing a quick reference to some of the important concepts of this method which are relevant to issues discussed in the following sections.

The Finite Volume Method was proposed in the field of Computational Fluid Dynamics independently by MacDonald [5], and MacCormack and Paulay [6] for the solution of two-dimensional, time dependent Euler equations and extended to three dimension by Rizzi and Inouye [7]. Since then, this method has been used for solving both compressible and incompressible flows [7-10]. The Finite Volume Method is based upon the integral form of the equations and the global conservation law for the whole domain is satisfied for every sub-domain or control volume. In this respect, the Finite Volume Method is similar to the Finite Element Method. However, the Finite Volume Method assumes that the variables are piecewise constant within each sub-domain, and hence, by omitting the test function from the formulations, uses a simple formulation based upon volume averaging gradients. Therefore, the Finite Volume Method has the same flexibility as the Finite Element Method, whilst using a mathematical expression which clearly describes the physics of the fluid flow.

The Finite Volume Method has the ability to deal with complicated geometries and irregular sub-domains. The Finite Volume Method can be implemented using a side-based data structure to carry out all the required mathematical operations. In this data structure the relationship between a cell and its neighbours is defined using the side connectivity matrix. Such a data structure is suitable for defining unstructured meshes which provide a flexible means for discretising complex domains. Therefore, the Finite Volume Method can handle any type of grid. Hence, unlike the Finite Difference Method which is usually applied to the differential form of the governing equations and is usually applied on structured grids, the Finite Volume method can be easily applied on unstructured meshes.

Now consider a domain Ω which is enclosed by the boundary Γ (Figure 2.1). For this sub-domain Ω, in the absence of body forces S_Ω, equation (2.1) can be written as

$$\frac{\partial}{\partial t}\int_\Omega \mathbf{W}\, d\Omega + \oint_\Gamma \vec{\mathbf{F}} \cdot \vec{\mathbf{n}}\, d\Gamma - \oint_\Gamma \vec{S}_\Gamma \cdot \vec{\mathbf{n}}\, d\Gamma = 0 \tag{4.1}$$

For simplicity, assume that $\vec{\mathbf{F}}$ includes both the flux vector $\vec{\mathbf{F}}$ and the surface stresses S_Γ on the boundary Γ. Hence, equation (4.1) can be written as

$$\frac{\partial}{\partial t}\int_\Omega \mathbf{W}\, d\Omega + \oint_\Gamma (\vec{\mathbf{F}} \cdot \vec{\mathbf{n}})\, d\Gamma \; = 0 \tag{4.2}$$

where $\vec{\mathbf{n}}$ is the normal vector to the boundary and $\vec{\mathbf{F}}$ is a general flux vector acting on the boundary of the domain. If the domain is divided into a large number of small arbitrary sub-domains, then, the governing equations can be used to estimate the average rate of variation of $\mathbf{W}$ in each sub-domain. This method of obtaining discrete approximations preserves the conservation form. The implementation of the discretisation of the computational domain can be done by dividing the domain into the sub-domains within which the integrations are performed.

The first step in the implementation of the Finite Volume Method is to divide the computational domain into a number of sub-domains, which are called control volumes. The control volumes can be formed by a set of non-overlapping cells in two dimensions or tetrahedral cells in three dimensions . In an unstructured grid, the control volumes can consist of triangular or tetrahedral cells in two and three dimensions, respectively. The control volume can be formed by either a cell or a number of cells sharing a vertex (Figure 4.1). Hence, the two schemes, cell-centred and cell-vertex, have been developed [8,11].

4.2.1 Different Schemes for the Finite Volume Discretisation

It is necessary to assign the flow variables to a fixed location within the control volume. The approach to this depends on the choice of the adopted scheme. The Finite Volume Method can generally be divided into cell-centred and cell-vertex schemes. Then, the discretisation may be obtained by storing values of flow variables at either the cell centres [8,9] or the cell vertices [10,11] which, in both cases, are considered to be the centre of appropriate control volumes.

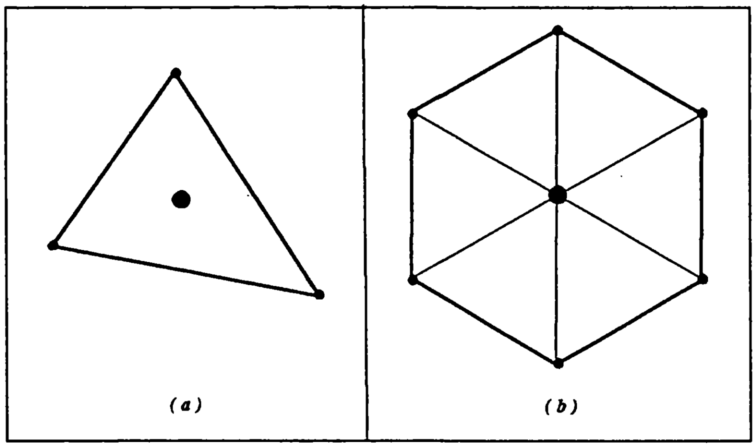

Figure 4.1.a *A triangular cell-centred control volume*
Figure 4.1.b *A cell-vertex control volume consist of triangles*

In the cell-centred scheme, the exact location of the variables inside the control volume is unknown. Therefore, the variables are assigned to the centre of the cells. Thus, for this approach, the control volumes are a set of non-overlapping computational cells equivalent to the mesh cells (Figure 4.1.a). This scheme requires an additional array to store values of flow variables at the centre of the cells and out side the boundaries of the domain.

In the cell-vertex scheme the variables are assigned to the grid nodes. For this approach, the control volume for a node is typically taken as the union of all the cells sharing that node (Figure 4.1.b). Note that, a boundary node lies on the boundary of its own control volume. In a general unstructured mesh there are less nodes than elements. This is manifestly the case in three dimensions where there are approximately five times the number of elements than nodes. Hence, the cell-vertex scheme provides a more efficient algorithm. Therefore, in the present work the cell-vertex scheme is adopted.

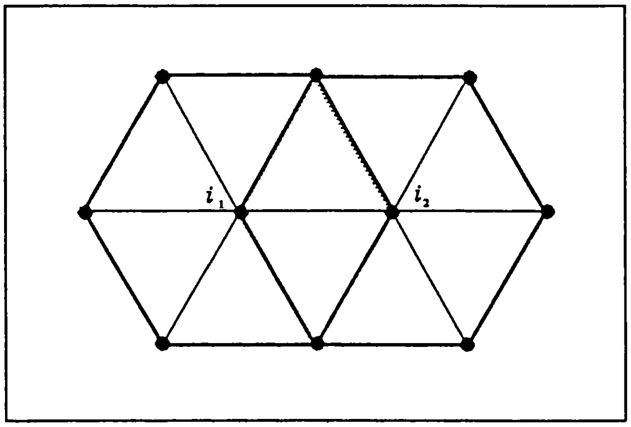

Figure 4.2 *Two-dimensional overlapping control volumes.*

In present Finite Volume scheme, the control volume can be considered as the union of all the cells sharing a node. Hence, control volumes of every two neighbouring nodes i_1 and i_2 share the cells which include those nodes (Figure 4.2). Therefore, this results in a set of overlapping computational control volumes. However, the alternative of the non-overlapping control volumes around every vertex can be developed [12]. But for the sake of simplicity of programming, present cell-vertex scheme involves the set of overlapping control volumes.

4.2.2 Concept of Overlapping Control Volumes

Consider three arbitrary sub-divisions of Ω_1, Ω_2 and Ω_3 for the domain Ω (Figure 4.3). Equation (4.2) for each sub-domain can be written as

$$\frac{\partial}{\partial t}\int_{\Omega_1} \mathbf{W}\, d\Omega + \oint_{ABDA} \vec{\mathbf{F}} \cdot \vec{\mathbf{n}}\, d\Gamma = 0 \tag{4.3}$$

$$\frac{\partial}{\partial t}\int_{\Omega_2} \mathbf{W}\, d\Omega + \oint_{BCDB} \vec{\mathbf{F}} \cdot \vec{\mathbf{n}}\, d\Gamma = 0 \tag{4.4}$$

$$\frac{\partial}{\partial t}\int_{\Omega_3} \mathbf{W}\, d\Omega + \oint_{CADC} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma = 0 \tag{4.5}$$

The contour integral for each sub-domain can be written in terms of a summation of line integrals surrounding that sub-domain. For instance

$$\oint_{ABDA} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma = \oint_{AB} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma + \oint_{BD} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma + \oint_{DA} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma \tag{4.6}$$

$$\oint_{BCDB} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma = \oint_{BC} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma + \oint_{CD} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma + \oint_{DB} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma \tag{4.7}$$

$$\oint_{CADC} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma = \oint_{CA} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma + \oint_{AD} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma + \oint_{DC} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma \tag{4.8}$$

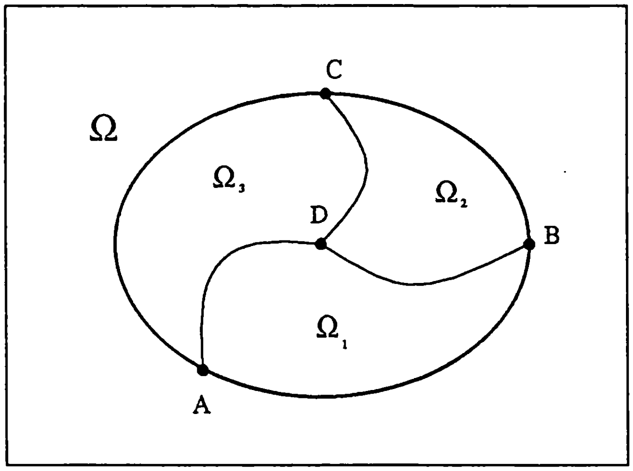

Figure 4.3 *Three arbitrary sub-divisions for the domain Ω*

Clearly, the line integral along a line between two sub-domains appears twice but in different directions. Now, the following relations are evident

$$\begin{aligned}
\oint_{AD} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma &= -\oint_{DA} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma \\
\oint_{BD} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma &= -\oint_{DB} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma \\
\oint_{CD} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma &= -\oint_{DC} \vec{\mathbf{F}}\cdot\vec{\mathbf{n}}\, d\Gamma
\end{aligned} \tag{4.9}$$

Hence, summing all the contour integrals inside the domain Ω, the interior line integrals appear twice with an opposite sign. Since these contributions cancel, the resulting contour integral for the main domain can be written as

$$\oint_{ABCA} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma = \oint_{AB} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma + \oint_{BC} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma + \oint_{CA} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma \qquad (4.10)$$

Adding the time dependent volume integrals of the sub-domains to the sum of contour integrals leads to the global conservation law for the whole domain, namely,

$$\frac{\partial}{\partial t}\int_{\Omega} \mathbf{W}\, d\Omega + \oint_{ABCA} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma = 0 \qquad (4.11)$$

It is important that all the contour integrals be taken in a consistent direction otherwise the internal flux contributions will not be cancelled. In order to have the outward direction for normals, the direction of contour integrals is chosen as anti-clockwise.

4.3 Discretisation Using Cell-vertex Finite Volume Method

The discretised form of equation (4.11) can be applied over each control volume. It is apparent that the flux for a side must be defined in order to calculate the flux integral for a control volume. Here, a side-based discretisation of the governing equations follows two-dimensional procedure proposed by Jameson et al [8,10]. In this approach, the control volume associated with the vertex i is formed by gathering the volumes of the cells meeting at the vertex i (Figure 4.4). The flow variables are stored at the vertices and they are assumed to represent average values over the entire control volume surrounding each vertex. Using a side-based algorithm, the contour integral of equation (4.2) can be applied for the sides surrounding every control volume i which consist of several triangles or tetrahedra.

The contour integral in equation (4.2) can be approximated by a summation over the boundary of the control volume (Figure 4.4)

$$\oint_{\Gamma_i} \vec{\mathbf{F}} \cdot \vec{\mathrm{n}}\, d\Gamma = \sum_{k=1}^{N_{side}} \vec{\mathbf{F}} \cdot \vec{\Delta \mathrm{s}} \qquad (4.12)$$

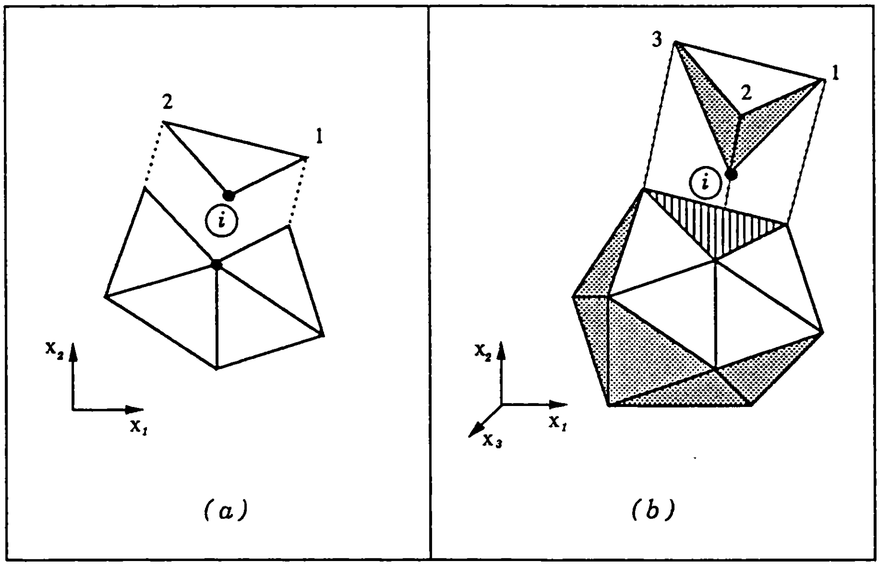

Figure 4.4.a *An elementary 2D cell-vertex control volume i*
Figure 4.4.b *An elementary 3D cell-vertex control volume i*
(by gathering all the cells surrounding node i)

where N_{side} is the number of sides surrounding the control volume i. Defining the the cartesian components of the flux vector $\vec{\mathbf{F}}$ as

$$\vec{\mathbf{F}} = \mathbf{f}\,\hat{\mathbf{I}}_1 + \mathbf{g}\,\hat{\mathbf{I}}_2 + \mathbf{h}\,\hat{\mathbf{I}}_3 \tag{4.13}$$

and the vector of projected areas of the side k , $\vec{\Delta \mathbf{s}}$ is defined as

$$\vec{\Delta \mathbf{s}} = \Delta s_1\,\hat{\mathbf{I}}_1 + \Delta s_2\,\hat{\mathbf{I}}_2 + \Delta s_3\,\hat{\mathbf{I}}_3 \tag{4.14}$$

where $\mathbf{I}_j$ is the unit vector in the x_j direction and Δs_j is the projected area of the side k perpendicular to the x_j axis of the cartesian coordinate system $(j = 1, 2, 3)$.

By assuming $\mathbf{W}_i$ is an average value of the quantity $\mathbf{W}$ over the control volume i and Ω_i is the volume of the control volume associated with node i, the variable $\mathbf{W}_\mathrm{i}$ at node i can be defined as

$$\mathbf{W}_i = \frac{1}{\Omega_i} \int_{\Omega_i} \mathbf{W}\, d\Omega \tag{4.15}$$

Then, for a domain discretisation fixed in time, the semi-discretised form of equation (4.11) can be expressed as

$$\frac{\partial}{\partial t}\mathbf{W}_i + \frac{1}{\Omega_i}\sum_{k=1}^{N_{side}} \left(\mathbf{f}\,\Delta s_1 + \mathbf{g}\,\Delta s_2 + \mathbf{h}\,\Delta s_3 \right)_k = 0 \qquad (4.16)$$

Note that, only the fluxes across external boundary sides of a control volume contribute to the flux balance of equation (4.16). The flux balance at each node can be evaluated directly by summing the contributions from its surrounding sides. The fluxes $\mathbf{f}$, $\mathbf{g}$ and $\mathbf{h}$ are estimates of the mean fluxes normal to the cartesian components of the side k. This averaged flux of each side $\vec{\mathbf{F}}_k$ can be formulated using the flux $\vec{\mathbf{F}}_i$ associated with the end nodes of the proposed side as

$$\vec{\mathbf{F}}_k = \frac{1}{N_{node}}\sum_{i=1}^{N_{node}} \vec{\mathbf{F}}_i \qquad (4.17)$$

where N_{node} is the number of nodes of the side. As can be seen, equation (4.16) describes the evolution of the average value $\mathbf{W}_i$ in time. This equation provides great flexibility to apply the formulation to polygonal control volumes, which make it suitable for irregular meshes around complex geometries.

Equation (4.16) shows that, in the absence of source terms, the variation of average value $\mathbf{W}_i$ over a time step Δt should be balanced with the sum of fluxes passing through sides. When the neighbouring control volumes i_r and i_l are considered, the flux through the side between these two control volume contributes to both of them but with opposite signs. By adding all the contributions to the control volumes, there is no uncancelled contributions from inside the domain and the global conservation is satisfied and the scheme is said to be conservative. Therefore, the scheme can be developed based upon a side-based algorithm in which the computation can be proceed using all sides throughout the computational domain.

It is interesting to note that using a side-based algorithm, the grid coordinates appear only in the determination of the volume of the control volume and side areas. Hence, for the flux balance computation for each control volume, only the coordinates of vertices of boundary sides of the control volume are needed. Therefore, it is more convenient to use a side-based data structure for unstructured grids in which the connectivity data can be defined in terms of the end nodes and nodes associated with neighbour control volumes.

4.3.1 Area of the Side

Since the process of flux balance computation between the control volumes utilises the sides surrounding the control volumes, the sides can be considered as the basic element in the discretisation procedure. The area and the orientation of the sides can be defined in terms of a single vector in which its components consist of the projected areas normal to the cartesian axis.

For two-dimensional problems, using two end points n_1 and n_2 of an edge, the projected length, Δs_j, is defined as

$$\Delta s_j = (x_j)_{n_2} - (x_j)_{n_1} \qquad (j = 1, 2) \tag{4.18}$$

For three-dimensional problems one of the alternatives to calculating the projected area of boundary sides is to use the vector product of two edges of a boundary face as two vectors in cartesian coordinates (Figure 4.5).

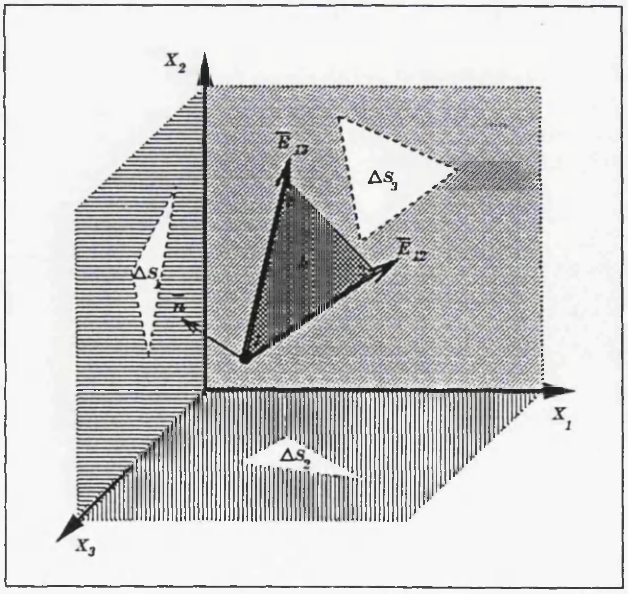

Figure 4.5 *Projected areas Δx_j perpendicular to cartesian axis*

Considering the vector product of two edges $\vec{\mathbf{E}}_{12}$ and $\vec{\mathbf{E}}_{13}$ meeting at node n_1 we can write

$$\begin{aligned}\vec{\Delta \mathrm{s}} &= \frac{1}{2}\left[\vec{\mathbf{E}}_{n_{12}} \times \vec{\mathbf{E}}_{n_{13}}\right] \\ &= \frac{1}{2}\begin{bmatrix} \hat{\mathbf{I}}_1 & \hat{\mathbf{I}}_2 & \hat{\mathbf{I}}_3 \\ (\Delta x_1)_{n_{12}} & (\Delta x_2)_{n_{12}} & (\Delta x_3)_{n_{12}} \\ (\Delta x_1)_{n_{13}} & (\Delta x_2)_{n_{13}} & (\Delta x_3)_{n_{13}} \end{bmatrix}\end{aligned} \tag{4.19}$$

Therefore, considering equation (4.19), the components of the projected areas can be defined as

$$\begin{aligned}\Delta s_1 &= \frac{1}{2}\Big[(\Delta x_2)_{n_{12}}(\Delta x_3)_{n_{13}} - (\Delta x_3)_{n_{12}}(\Delta x_2)_{n_{13}}\Big] \\ \Delta s_2 &= \frac{1}{2}\Big[(\Delta x_3)_{n_{12}}(\Delta x_1)_{n_{13}} - (\Delta x_1)_{n_{12}}(\Delta x_3)_{n_{13}}\Big] \\ \Delta s_3 &= \frac{1}{2}\Big[(\Delta x_1)_{n_{12}}(\Delta x_2)_{n_{13}} - (\Delta x_2)_{n_{12}}(\Delta x_1)_{n_{13}}\Big]\end{aligned} \tag{4.20}$$

4.3.2 Volume of the Control Volumes

The volume of the control volume associated with node i, which is the summation of the volumes of the cells meeting at that node, can be calculated directly using

$$\Omega_i = \oint_{\Gamma_i} x_j \cdot \vec{\mathbf{n}}\, d\Gamma \qquad (j = 1,2,3) \tag{4.21}$$

For two and three-dimensional grids the number of nodes which define each side is equal to the order of dimension N_{dim}. Thus, generally the volume of the control volume can be computed using its surrounding sides by utilising Δs_j projected areas and coordinates of the nodes n of the surrounding sides k as

$$\Omega_i = \sum_{k=1}^{N_{side}} (\bar{x}_j)_k\,(\Delta s_j)_k \qquad (j = 1,2,3) \tag{4.22}$$

where $(\bar{x_j})_k = \sum_{n=1}^{N_{dim}} (x_j)_n / N_{dim}$ is the average value of (x_j) at the proposed side. Note that, the computation of Ω_i can be done using a cyclic summation of the above formula over all the sides. The contribution of each side to the computation of the control volume is added to the two neighbouring nodes of that side but with opposite signs (Figure 4.6).

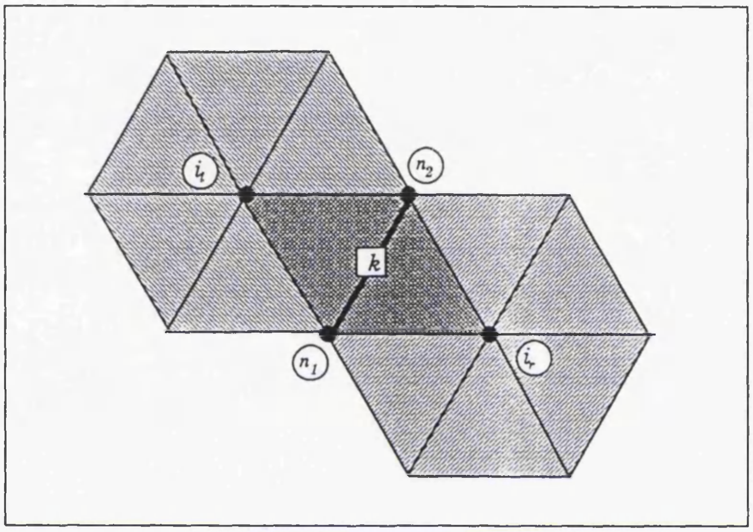

Figure 4.6 *Side k between two neighbouring control volumes*

For two-dimensional problems, the control volume area Ω_i of the triangles meeting at the node i, can be computed from

$$\Omega_i = \sum_{k=1}^{N_{edge}} \left[\frac{(x_1)_{n_2} + (x_1)_{n_1}}{2} \left((x_2)_{n_2} - (x_2)_{n_1} \right) \right]_k \tag{4.23}$$

where N_{edge} indicates the number of cells surrounding each control volume and subscripts n_1 and n_2 indicate the two end points of the edges k which enclose the node i.

In three dimensions the control volume can be computed using the summation of the volumes of the tetrahedra meeting at the node i as

$$\Omega_i = \sum_{k=1}^{N_{face}} \left[\frac{(x_1)_{n_1} + (x_1)_{n_2} + (x_1)_{n_3}}{3} \Delta s_1 \right]_k \tag{4.24}$$

where subscripts n_1, n_2 and n_3 indicate the three points of the faces k which surround the node i.

4.5 Summary

Some concepts of the Finite Volume formulation for discretisation of the Navier-Stokes equations on the unstructured meshes have been presented. After an introduction to the cell-centred and the cell-vertex schemes, the cell-vertex Finite Volume scheme for discretising governing has been chosen as the basis of the method. The discretisation of the integral form of the Navier-Stokes equation using a side-based algorithm has been described. Finally, the details required to compute the geometrical parameters of triangular and tetrahedra control volumes has been presented.

Following the cell-vertex Finite Volume scheme of Jameson et al [8,10], the governing equations can be discretised in space and time separately. More details of the formulation and numerical techniques for the solution of the incompressible flow equations are described in the next two chapters. Chapter 5 describes the details of the space discretisation of the fluxes associated with the boundary sides of the control volumes and Chapter 6 reviews the technique for implementing the time integration and boundary conditions.

References

[1] C. Hirsch, 'Numerical Computation of Internal and External Flows', Vol. 1, Fundamentals of Numerical Discretisation, Wiley, (1989).

[2] W. J. Minkowycz., E. M. Sparrow., G. E. Schneider., R. H. Pletcher.,'Hand Book of Numerical Heat Transfer', John Wiley and Sons, Inc., (1988).

[3] R. Peyret and T. T. Taylor, 'Computational Methods for Fluid Flow', Springer-Verlag, 1983.

[4] D. A. Anderson, J. C. Tannehill and R. H. Pletcher, 'Computational Fluid Mechanics and Heat Transfer', Hemisphere Publishing Co. and McGraw-Hill Co., (1984).

[5] P. W. MacDonnald, 'The Computation of Transonic Flow through Two-dimensional Gas Turbine Cascades.' ASME Paper 71-GT-89,(1971).

[6] R. W. MacCormack, and , A. J. Paullay, 'Computational Efficiency Achieved by Time Splitting Finite Difference Operators', AIAA Paper 72-154, San Diego, (1972).

[7] A. W. Rizzi, and M. Inouye, 'Time Split Finite Volume Method for Three-dimensional Blunt-body Flows', AIAA Journal, (1973).

[8] A. Jameson, W. Schmidt and E. Turkel, 'Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes', AIAA Paper 81-1256, June (1981).

[9] D. K. Natakusumah, 'Finite Volume Solution of the Two-Dimensional Compressible Flow Equations on Structured and Unstructured Hybrid Grids', Ph.D. Thesis, Civil Eng. Dept., University of Wales, (1992).

[10] A. Jameson, T. J. Baker and N. P. Weatherill, 'Calculation of Inviscid Transonic Flow over a Complete Aircraft', AIAA Paper 86-0103, (1986).

[11] F. Beaven, 'Numerical Solution of the Navier-Stokes Equations on Generalised Grids', Ph.D. Thesis, Civil Eng. Dept., University of Wales, (1994).

[12] P. Tammamidis, 'A New Upwind Scheme on Triangular Meshes using The Finite Volume', J. of Comp. Meth. in Appl. Mech. , **128** (1995).

5

Discretisation of Governing Equations

5.1 Introduction

The spatial discretisation procedure of the governing equations follows the numerical solution of the Navier-Stokes equations proposed by Jameson et al [1,14]. Following this approach, the discretisation in space and time is done separately. The discretisation of both the viscous and convective fluxes are important tasks in the cell-vertex finite volume formulation. In this work, the fluxes are discretised in space following the cell-vertex finite volume approach. The proposed algorithm is developed for use on unstructured triangular/tetrahedral meshes utilising a side-based data structure. In this chapter, after describing the spatial discretisation of the governing equations, space discretisation of the convective and the diffusive fluxes is studied. The techniques regarding the numerical solution, including time stepping, boundary conditions and computational issues are described in Chapter 6.

The evaluation of the components of the convective flux term along the sides of the mesh depends on the selected numerical scheme as well as on the location of the flow variables with respect to the grid. In the present work, a central-difference scheme is adopted, which can be efficiently implemented for both two and three-dimensional applications. Because of the application of this scheme for space discretisation, an artificial dissipative operator must be added to the flux vector to ensure the stability of the numerical method. Due to the irregularity of node spacings in unstructured grids, the implementation of an inappropriate artificial dissipation may degrade the accuracy and stability of the scheme. A good formulation of the artificial dissipation is essential for accurate simulations.

The computation of the viscous stresses implies employing proper formulations for the evaluation of the first order derivatives of the velocity components. Two formulations for the discretisation of the viscous terms are studied. These formulations are derived from the cell-vertex finite volume and the Galerkin finite element formulations for triangular meshes, respectively. In order to improve the accuracy of the cell-vertex finite volume formulation, a modification is introduced for the stencil employed for the computation of the viscous terms.

This chapter presents the formulation for the spatial discretisation of the convective, viscous and artificial dissipation terms. In Section 5.2 the standard cell-vertex finite volume formulation, which was reviewed in Chapter 4, is employed for discretising the modified incompressible equations. In Section 5.3 a brief review of the different space discretisation schemes is given. Section 5.4 describes the spatial discretisation of the convective term. Some aspects of the artificial dissipation are studied in Section 5.5 and the application of appropriate formulations for scaling and the normalisation of the artificial dissipation are discussed. In section 5.6, after an investigation of various alternatives for the computation of the viscous terms, a new modified stencil for computing the contour integral for viscous stresses in the cell vertex finite volume formulation, is introduced.

5.2 Spatial Discretisation of the Governing Equations

The implementation of the finite volume method for the solution of the incompressible Navier-Stokes equations is described in this section. The numerical method for the solution of these equations relies largely on the methods developed for the compressible equations. The cell-vertex finite volume scheme is developed on the basis of the scheme proposed by Jameson et al [1] for the solution of the compressible Euler equations. Here, a discretised form of the set of modified equations, which were presented in Chapter 3, is formulated using the procedure described in Chapter 4.

The first step is the implementation of the finite volume method for the solution of the proposed modified incompressible equations, including preconditioning matrix **P**. The governing set of equations of incompressible laminar flow can be obtained by dropping the asterisks of the set of non-dimensional form of equations (3.26). Integrating this set of equations within the control volume i and applying the Gauss-Divergence theorem, the integral form of the proposed equations can be written similar to the equation (4.11), as

$$\frac{\partial}{\partial t}\int_{\Omega} \mathbf{W}\, d\Omega + \mathbf{P} \oint_{\Gamma} \vec{\mathbf{F}} \cdot \vec{\mathbf{n}}\, d\Gamma = 0 \tag{5.1}$$

where $\vec{\mathbf{n}}$ is an outward unit vector normal to the boundary Γ enclosing the sub-domain Ω. The cartesian components of $\vec{\mathbf{F}}$ are defined as $\mathbf{f}, \mathbf{g}$ and $\mathbf{h}$. If the convective and viscous elements of fluxes are considered as $\vec{\mathbf{F}}^c$ and $\vec{\mathbf{F}}^d$, respectively, then the integral form of this equation may be written for three dimensions as

$$\begin{aligned}\frac{\partial}{\partial t}\int_{\Omega_i} \mathbf{W}\, d\Omega + \mathbf{P}\Big[&\oint_{\Gamma_i}(\mathbf{f}^c\, d\Gamma_1 + \mathbf{g}^c\, d\Gamma_2 + \mathbf{h}^c\, d\Gamma_3) \\ &- \oint_{\Gamma_i}(\mathbf{f}^d\, d\Gamma_1 + \mathbf{g}^d\, d\Gamma_2 + \mathbf{h}^d\, d\Gamma_3)\Big] = 0\end{aligned} \tag{5.2}$$

Now, similar to equation (4.15), we can approximate equation (5.2) as contour integrals around the sides k surrounding each control volume as

$$\begin{aligned}\frac{\partial}{\partial t}\mathbf{W}_i + \frac{\mathbf{P}}{\Omega_i} \sum_{k=1}^{N_{side}} \Big[&(\mathbf{f}^c\, \Delta s_1 + \mathbf{g}^c\, \Delta s_2 + \mathbf{h}^c\, \Delta s_3) \\ &- (\mathbf{f}^d\, \Delta s_1 + \mathbf{g}^d\, \Delta s_2 + \mathbf{h}^d\, \Delta s_3)\Big]_k = 0\end{aligned} \tag{5.3}$$

The vector of conserved variables **W**, the preconditioning matrix **P** and the convective and viscous component of fluxes vectors are given by equation (3.27).

to higher order accuracy increases the complexity of numerical schemes and they often need additional formulations for the computation of the fluxes.

Unlike upwind-differencing schemes, central-differencing schemes do not distinguish between upstream and downstream influences. For computation of the convection dominated flows, the central-differencing schemes do not provide enough diffusive effects to damp unwanted oscillations and can lead to instabilities in the solution. Hence, it is necessary to add explicit artificial dissipation fluxes to maintain stability of the scheme and to prevent odd-even point oscillations. Such oscillations can be controlled with the addition of a numerical viscosity. Therefore, any central-difference discretisation scheme should contain a dissipative operator in order to ensure stability of the numerical method.

In the present work, a central-difference scheme is employed, which implies a requirement for an additional artificial dissipative term. The convergence and accuracy of the scheme is directly related to appropriate formulations for the artificial dissipation.

5.4 Convective Term

In order to evaluate the convective flux term within the control volume, the approximation to the convective flux integral $C(\mathbf{W})$ is given by

$$C(\mathbf{W}_i) \;=\; \sum_{k=1}^{N_{side}} [\mathbf{f}^c \Delta s_1 + \mathbf{g}^c \Delta s_2 + \mathbf{h}^c \Delta s_3]_k \tag{5.5}$$

where N_{side} is the number of sides enclosing the control volume i. $C(\mathbf{W}_i)$ is the convective operator, which represents the discrete approximation to the convective (inviscid) flux balance across all boundaries of the control volume i. By introducing the velocity flux Q for each side as

$$Q = u_1 \Delta s_1 + u_2 \Delta s_2 + u_3 \Delta s_3 \tag{5.6}$$

the convective operator for control volume i can be written as

$$C(\mathbf{W}_i) = \sum_{k=1}^{N_{side}} \begin{bmatrix} Q \\ Q\,u_1 + p\,\Delta s_1 \\ Q\,u_2 + p\,\Delta s_2 \\ Q\,u_3 + p\,\Delta s_3 \end{bmatrix}_k \tag{5.7}$$

In practice, each side is accessed only once and the flux balance is carried out by adding the flux contribution to the left control volume and subtracting the same amount of flux from the right control volume. By accumulating flux contributions of the convective operator from each boundary side of the control volume, the residual due to the convective term at the central node of the control volume will be obtained.

The conserved variables on the sides enclosing the control volume can be computed using the simple procedure of a non-centred weighting. In this case the flow variables at the side k delimiting two control volumes is given by

$$\mathbf{W}_k \;=\; \sum_{n=1}^{N_{dim}} \omega_n \, \mathbf{W}_n \tag{5.8}$$

where N_{dim} is the number of vertices of each side. Here, ω_n are weighting coefficients at node n of side k defined as

$$\omega_n \;=\; \frac{\Omega_n}{\sum_{n=1}^{N_{dim}} \Omega_n} \tag{5.9}$$

These weighting coefficients are defined using Ω_n , the volume of the control volumes associated with the end nodes of the side k. Other options for defining these weighting coefficients are also possible. For the meshes with smooth variation of mesh spacings, a special case of equation (5.8) in which $\omega_n = 1/N_{dim}$ has been implemented in the present work as

$$\mathbf{W}_k \;=\; \frac{1}{N_{dim}} \sum_{n=1}^{N_{dim}} \mathbf{W}_n \tag{5.10}$$

It means that, the flow variables along the side k are taken as a simple average of values at the nodes associated with that side.

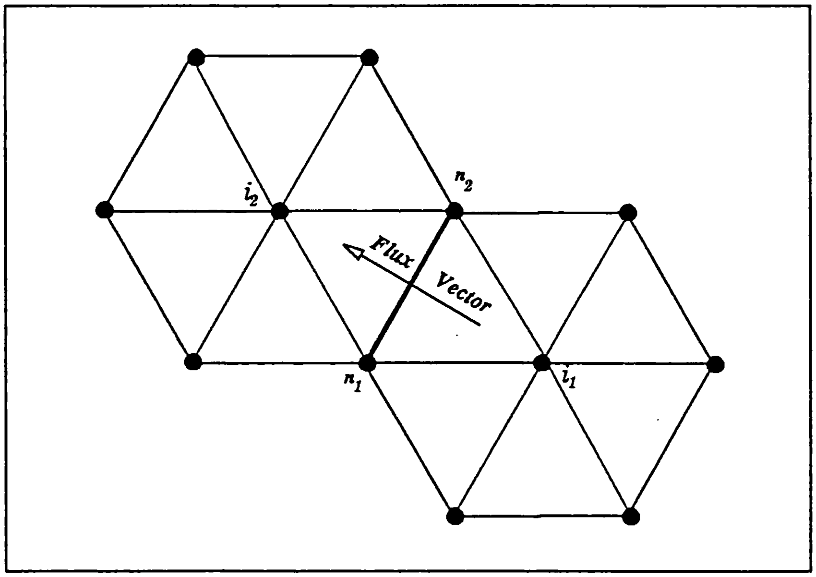

Figure 5.1 *Accumulation of convective flux in two dimensions*

The computation of the convective fluxes can be done using a loop over all the sides by adding the convective flux associated with every side, equation (5.7), to its two neighbouring control volumes. The two-dimensional description of the convective flux accumulation of the present cell-vertex approach is presented in Figure (5.1). Note that, if k represents a boundary side, the centre node of a control volume lies on its own boundary. Therefore, it is necessary to add the fluxes related to this control volume to the nodes of the proposed side. The values of flow variables on these boundary sides are determined according to the boundary conditions defined on that side.

5.5 Artificial Dissipation

It is shown that, for nodal based schemes on equilateral triangular grids a three level oscillation is permitted by the discrete equations as the solution of the system of equations proceeds to the steady state [14]. Any explicit solution of second order central-difference scheme of the linear convection equation generates oscillations. Indeed these oscillations make the central-difference schemes unstable. The instability of the central-differencing schemes can be controlled with the addition of a numerical viscosity. These additional terms should simulate the effect of the physical viscosity, on the scale of the mesh, especially around discontinuities and high gradient regions. Hence, this term must be of higher order than the truncation error in order to be negligible in smooth regions. Note that, the artificial dissipative operator has no physical meaning, and it is only used to control numerical oscillations.

Even in viscous flow computations, outside the boundary layer and wake area the effect of the diffusive terms is negligible, so there is not sufficient natural dissipation to damp the oscillations which appear due to the central-difference discretisation of the governing equations. Although the boundary conditions should suppress such modes at steady state, their existence during the (pseudo) transient solution process may adversely affect convergence [13]. Before describing the details of the formulation of artificial dissipation for the set of modified equations, a brief review of the basic idea behind the formulation of this term is presented in this section.

The central-difference schemes include a form of artificial dissipation for two main purposes :

i) to control the odd-even decoupling of grid points which is typical of central-differencing,

ii) to maintain stability in the essentially inviscid region of the flow field.

For the solution of the governing equations, the artificial dissipative term should be constructed in such a way that satisfies the following conditions:

i) the dissipative operator should be capable of eliminating all physically meaningless oscillations, but must remain small enough so as not to reduce the accuracy of the original scheme.

ii) it should be conservative. This means that when it summed over the entire flow-field all dissipative terms cancel. Therefore, no net production of mass and momentum occurs. This criterion can be achieved by ensuring that, for each contribution to a particular control volume, there exists an equal and opposite contribution to the opposite control volumes.

iii) its construction should be in a simple and inexpensive manner. This means that, although the dissipative operator is to be calculated at each time-step, its calculation should not consume too much CPU time.

The artificial dissipative term $D(\mathbf{W}_i)$ can be formulated in a number of different ways [1,15-18]. Here a very brief summary of related schemes is given in one-dimensional form.

5.5.1 Review of Useful Artificial Dissipation Models

As a modification to the convective term, Von Neumann and Richtmyer [19] suggested adding a dissipation term to the central-difference discretisation in space using a first difference term as

$$D(\mathbf{W}) \approx -d_2 \, \Delta x_1 \frac{\partial \mathbf{W}}{\partial x_1} \tag{5.11}$$

In order to enhance the effect of the dissipation in the presence of strong pressure gradients and to reduce it in smooth flow regions, MacCormak and Baldwin [33] made d_2 proportional to a second derivative of the pressure field as

$$d_2 = \nu_2(|\vec{\mathbf{U}}| + c)(\Delta x_1)^2 |\frac{\partial^2 p}{\partial x_1^2}| \tag{5.12}$$

where c is the speed and ν_2 is a global constant. This pressure scaling term is generally second order, except in regions of strong pressure gradients, where it reduces to the order of one.

Another form of artificial viscosity is expressed by Steger [21] and applied in the Beam and Warming implicit scheme [22]. This form introduces higher order derivatives while scaling the dissipation with the maximum eigenvalue of the convective part of the system of equations, equation (3.18). This form presents a dissipation proportional to the third difference as

$$D(\mathbf{W}) \approx \nu_4(|\vec{\mathbf{U}}| + c)\,(\Delta x_1)^3 \frac{\partial^3 \mathbf{W}}{\partial x_1^3} \tag{5.13}$$

where ν_4 is a global constant. In viscous flow computations, in order to reduce the magnitude of the artificial dissipation inside the viscous layer, a weighting function can also be used [23].

Jameson et al [1] combined a second order dissipation term introduced by Von Neumann and Richtmyer [19], using the pressure scaling of MacCormak and Baldwin [20], with the fourth order dissipation term of Steger [21] as

$$D(\mathbf{W}_i) \approx -d_2\,\Delta x_j \frac{\partial \mathbf{W}}{\partial x_j} + \nu_4(|\vec{\mathbf{U}}| + c)(\Delta x_j)^3 \frac{\partial^3 \mathbf{W}}{\partial x_j^3} \tag{5.14}$$

In the presence of strong pressure gradients or discontinuities, this blend of undivided Laplacian and Biharmonic operators is able to damp high-frequency oscillations, while for smooth flow the fourth differencing is sufficient to stabilise the computations. Note that, expressions for undivided Laplacian and Biharmonic operators are used, since the difference in the flow variables is not divided by the corresponding Δx_j.

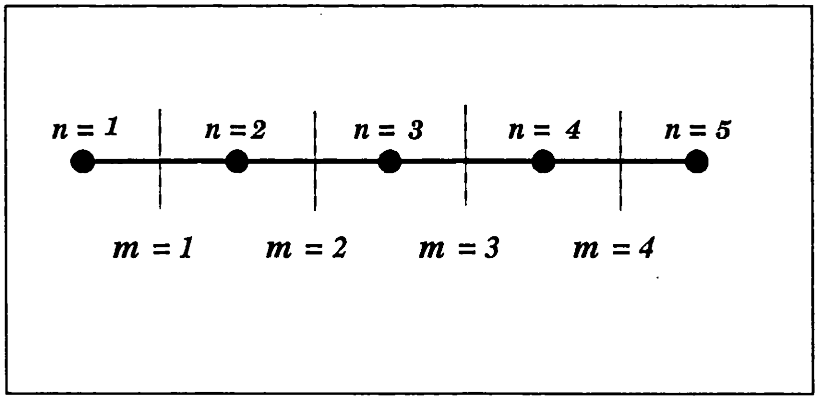

Figure 5.2 *Designation of one-dimensional stencil*

At this point, some simplified discretisation of the first, second, third and fourth differences are given [24] using a one-dimensional stencil (Figure 5.2). The expressions $\nabla, \nabla^2, \nabla^3$ and ∇^4 are vectors of undivided first, second, third and fourth order differences of flow variables, respectively. The first difference $\Delta x_1(\partial \mathbf{W}/\partial x_1)$ for the edge m can be approximated by

$$\Delta x_j \frac{\partial \mathbf{W}}{\partial x_1} \approx (\nabla \mathbf{W})_m = \mathbf{W}_{n+1} - \mathbf{W}_n \tag{5.15}$$

Summation of $\nabla\mathbf{W}$ over edges surrounding node n gives an approximation for the second difference $\Delta x_1^2(\partial^2\mathbf{W}/\partial x_1^2)$ as

$$\begin{aligned}\nabla^2\mathbf{W}_n &= (\nabla\mathbf{W})_{m+1} - (\nabla\mathbf{W})_m \\ &= \mathbf{W}_{n+1} - 2\mathbf{W}_n + \mathbf{W}_{n-1}\end{aligned} \tag{5.16}$$

Then, the third difference $\Delta x_1^3(\partial^3\mathbf{W}/\partial x_1^3)$ along the edge m can be approximated using the second differences at neighbouring nodes as

$$\begin{aligned}\Delta x_1^3\frac{\partial^3\mathbf{W}}{\partial x_1^3} \approx (\nabla^3\mathbf{W})_m &= \nabla^2\mathbf{W}_{n+1} - \nabla^2\mathbf{W}_n \\ &= \mathbf{W}_{n+2} - 3\mathbf{W}_{n+1} + 3\mathbf{W}_n + \mathbf{W}_{n-1}\end{aligned} \tag{5.17}$$

Summing all the third differences along the edges around node n leads to an expression equal to the approximation for the fourth difference at that node as

$$\begin{aligned}\nabla^4\mathbf{W}_n &= (\nabla^3\mathbf{W})_{m+1} - (\nabla^3\mathbf{W})_m \\ &= \mathbf{W}_{n+2} - 4\mathbf{W}_{n+1} + 6\mathbf{W}_n - 4\mathbf{W}_{n-1} + \mathbf{W}_{n-2}\end{aligned} \tag{5.18}$$

5.5.2 Artificial Dissipation Operator for the Present Algorithm

For incompressible smooth flows the second term of equation (5.14) is sufficient to provide artificial dissipation. Therefore, the first term of this equation can be neglected, and hence, the fourth differencing term introduced by Steger [21], equation (5.13), and formulated by Jameson et al [1] for the cell-vertex finite volume method, is adopted. The edge-based dissipation formulation, which was originally developed for the inviscid flow equations on unstructured triangular grids, is implemented in the present work. The formulation can be applied with minor modifications and parameter tuning for both two and three-dimensional computations.

The fourth differencing term or biharmonic operator can be constructed by using a flux balance formulation which maintains the original conservative form.

The artificial dissipative term is constructed using the undivided Biharmonic operator which corresponds to a fourth difference of the conserved variables

$$D(\mathbf{W}_i) = \nabla^4 \mathbf{W}_i \tag{5.19}$$

For both two and three-dimensional cases, the artificial dissipation term will be computed by taking differences along edges of the mesh. For each central node of control volume i, the undivided Biharmonic operator can be defined as

$$\nabla^4 \mathbf{W}_i \approx \nu_4 \sum_{j=1}^{N_{edge}} \Lambda_{ij} (\nabla^2 \mathbf{W}_j - \nabla^2 \mathbf{W}_i) \tag{5.20}$$

where N_{edge} is the total number of edges connected to the node i and j is the other end node of that edge, and Λ_{ij} is a scaling factor associated with edge m located between nodes i and j. In practice, Λ_{ij} is calculated using the value of Λ_i and Λ_j, which are scaling factors of the two end nodes of the edge.

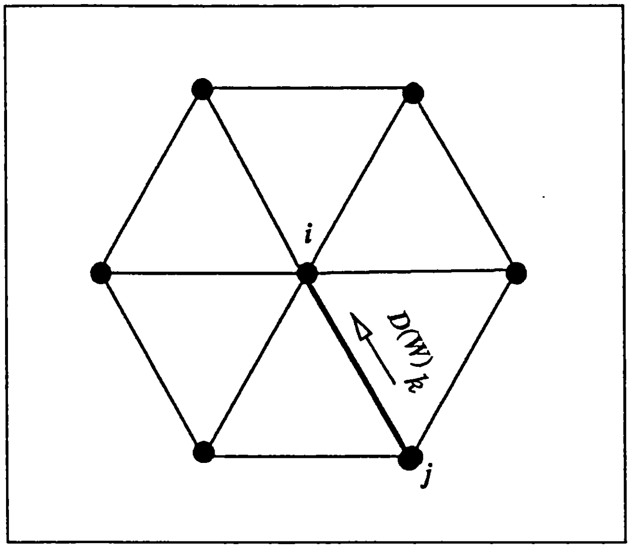

Figure 5.3 *Two dimensional description of dissipative flux*

It is necessary to calculate the undivided Laplacian term, $\nabla^2 \mathbf{W}_i$, for the computation of the biharmonic operator. The computation can be started by a summation of the differences between the flow variables across each edge inside the control volume (Figure 5.3). These are the edges that share the central node of control volume i. For each node i, the biharmonic operator can be expressed as

$$\nabla^2 \mathbf{W}_i = \sum_{n=1}^{N_{edge}} (\mathbf{W}_i - \mathbf{W}_j) \tag{5.21}$$

where j represents the neighbouring nodes of node i, with the total number of edges, N_{edge}, and $\mathbf{W}_j$ represents the flow variables as given in equation (3.27).

It should be noted that like other central-difference schemes, accurate results can be achieved by tuning the coefficient ν_4, the global constant of artificial dissipation, which appears in equation (5.20). The minimum possible value of the coefficient ν_4, which is needed to guarantee convergence, produces the most accurate results. Although the empirical coefficients of the artificial dissipation formulation should be tuned for given problems, the ability to minimise numerical dissipation provides great flexibility for improving the accuracy of the results.

5.5.3 Scaling the Artificial Dissipation

For edge-based dissipation, the Biharmonic operator can be scaled using the flux bounds associated with the particular edge. This bound can be obtained from the spectral radii of the flux Jacobian matrices associated with equation (3.15). The scaling factor of the dissipation along each edge, Λ_{ij}, can be taken as the minimum or average of the flux bounds Λ_i and Λ_j of two end nodes i and j of the edges. Hence, Λ_i can be evaluated proportional to an estimate of the maximum local eigenvalue of the flux Jacobian matrices [25,26], equations (3.18). In practice, the scaling factor at each node can be computed proportional to the summation of the spectral radii associated with the sides connected to that node. Thus, the spectral radii of the modified flux Jacobian matrices, equation (3.18) can be applied to formulate the scaling factor of each node as

$$\Lambda_i = \sum_{k=1}^{N_{side}} \left[\mid \vec{\mathrm{U}} \cdot \vec{\Delta \mathrm{s}} \mid + \sqrt{(\vec{\mathrm{U}} \cdot \vec{\Delta \mathrm{s}})^2 + \beta^2 (\vec{\Delta \mathrm{s}} \cdot \vec{\Delta \mathrm{s}})} \right]_k \tag{5.22}$$

where $\vec{\Delta s}$ is the vector of the projected area, which is defined in Chapter 4, and $\vec{\mathbf{U}}$ is the average velocity vector of the proposed side k.

Another form of the scaling factor was introduced by Farmer [27]. This formulation is formed analogous to the scaling factor for the formulations of the cell-vertex finite volume compressible flow solver [11]. This form of scaling factor was considered as

$$\Lambda_i = \sum_{k=1}^{N_{side}} \left[\left| \vec{\mathbf{U}} \cdot \vec{\Delta \mathbf{s}} \right| + \sqrt{\beta^2 (\vec{\Delta \mathbf{s}} \cdot \vec{\Delta \mathbf{s}})} \right]_k \tag{5.23}$$

Note that, if the parameter β^2 is considered as a function of local velocity, equation (3.21), there may be an effect on the value of the scaling factors expressed by equations (5.22) and (5.23). This effect will be more pronounced with different choices of the empirical coefficient of C_{β^2} for the proposed expression for β^2. Hence, the value of dissipation may increase in the high velocity regions and decrease in the low velocity regions. This fact may degrade the accuracy of the computations, especially if the value of the C_{β^2} is chosen different from unity. Imposing the preconditioning matrix **P** at each pseudo time step, allows the artificial dissipation formulation to be completely independent of the preconditioning matrix **P** [25]. Hence, in order to minimise the computational load, this term may be considered constant during every pseudo time stepping procedure. Thus, the spectral radii of the unmodified flux Jacobian matrices, equation (3.18) can be applied. Therefore, the formulation of the scaling factor Λ_i is identical to equation (5.22) or (5.23) by taking $\beta^2 = 1$.

However, using these formulations for scaling the artificial dissipation may make the scheme dissipative if the mesh spacing is not fine enough in the high velocity regions. In such positions, in order to reduce the artificial dissipation, analogous to the formulation of speed of sound for a compressible perfect gas, the artificial speed of sound, β^2, may be considered proportional to the pressure as

$$\beta^{2\star} \approx \frac{\gamma p}{\rho_0} \tag{5.24}$$

where γ is the ratio of the specific heat and ρ_0 is the mean value of the specific mass. Thus, two other forms of dissipation scaling factor can be formed by replacing β^2 with $\beta^{2\star}$ in equations (5.22) and (5.23). By considering the parameter $\beta^{2\star}$ as a function of local pressure, the value of the scaling factor not only becomes independent of the choice of the empirical coefficient C_{β^2} but also it helps reduce the value of the artificial dissipation at positions where the high velocities accrue in the regions of coarse mesh spacings.

5.5.4 Normalisation of the Artificial Dissipation

Application of edge-based dissipations on unstructured meshes produces non-uniform dissipation due to the irregularity in the number of edges connected to each computational node. This effect degrades the convergence and accuracy of the computations. Application of appropriate normalisation can help reduce the problem.

For two-dimensional compressible cell-vertex finite volume schemes on unstructured triangular meshes, Sykes [28] proposed a normalisation procedure for diminishing the effect of the variable number of edges which can be connected to each nodal point. Sykes divided the Laplacian operator computed for each node, equation (5.21), by the number of edges connected to that node. Then the left hand side of equation (5.20), is normalised by dividing the scaling factor of the proposed edge by $(N_i+N_j)/(2\times 6)$, where N_i and N_j are the number of edges connected to the two end points of the edge k. The factor 6 is obtained from the number of edges connected to each node of a completely regular and uniform two-dimensional mesh, which consists of of equilateral triangles.

In the present work, the normalisation of the dissipation for three dimensions is adopted using the following steps. The Laplacian operator, equation (5.21), can be normalised by multiplying by $1/N_i$, where N_i is the number of edges connected to the central node i. Finally, the Biharmonic operator, which is computed along the edges, is normalised by multiplying by $(N_i+N_j)/(2\times 14)$, where 14 is the number of edges connected to the centre of a regular equilateral control volume which consists of 24 tetrahedral cells.

5.6 Viscous Term

The computation of the diffusive flux components, $\mathbf{f}_i^d$, $\mathbf{g}_i^d$ and $\mathbf{h}_i^d$ is more complex than that required for the convective flux. This complication is due to the methods of computation of the shear stress components, equation (3.30), which are proportional to the first derivatives of the velocity components. The approximation to the viscous flux integrals for each control volume i, that appears in equation (5.4), can be written as

$$V(\mathbf{W}_i) \;=\; \sum_{k=1}^{N_{side}} (\,\mathbf{f}^{\mathrm{d}}\,\Delta s_1 + \mathbf{g}^{d}\,\Delta s_2\, + \mathbf{h}^{d}\,\Delta s_1\,)_k \tag{5.25}$$

where $V(\mathbf{W}_i)$ is the discrete approximation of the viscous terms. The elements of the viscous flux operator can be written as

$$V(\mathbf{W}_i) \;=\; \sum_{k=1}^{N_{side}} \begin{bmatrix} 0 \\ \tau_{11}\,\Delta s_1 + \tau_{12}\Delta s_2 + \tau_{13}\Delta s_3 \\ \tau_{21}\,\Delta s_1 + \tau_{22}\Delta s_2 + \tau_{23}\Delta s_3 \\ \tau_{31}\,\Delta s_1 + \tau_{32}\Delta s_2 + \tau_{33}\Delta s_3 \end{bmatrix}_k \tag{5.26}$$

where the viscous stresses τ_{ij} can be calculated from equation (3.30) as

$$\tau_{ij} = \frac{1}{\Re}\Big(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\Big) + \delta_{ij}\,\frac{2}{3\,\Re}\,\frac{\partial u_i}{\partial x_i} \qquad (i,j = 1,2,3) \tag{5.27}$$

In order to calculate the viscous stresses, it is necessary to calculate the gradients of the velocity components $\partial u_i/\partial x_j$. These derivatives can be computed assuming that the velocities are piecewise linear over a discrete domain Ω', so that their first order derivatives will be piecewise constant and may be evaluated as

$$\int_{\Omega'} \frac{\partial f}{\partial x_j} d\Omega' = \oint_{\Gamma'} f\, d\Gamma' \;\rightarrow\; \frac{\partial f}{\partial x_j} = \frac{1}{\Omega'} \oint_{\Gamma'} f\, d\Gamma' \qquad (j = 1,2,3) \tag{5.28}$$

where the integral is performed around a domain Ω' bounded by contour Γ'. Taking f as the average value of a variable, such as velocity component u_j at boundary side, this contour integral can be approximated by

$$\frac{\partial f}{\partial x_j} \approx \frac{1}{\Omega'} \sum_{l=1}^{N_{side}} \left[\frac{1}{N_{dim}} \sum_{n=1}^{N_{dim}} f_n \; \Delta s_j \right]_l \qquad (j = 1, 2, 3) \tag{5.29}$$

Here, Δs_j is the projected area of the sides l surrounding domain Ω' and N_{side} is the number of sides surrounding the secondary cell. The volume of the domain Ω' for each side is approximated using by

$$\Omega' \approx \sum_{l=1}^{N_{side}} \left[\frac{1}{N_{dim}} \sum_{n=1}^{N_{dim}} (x_1)_n \; \Delta s_1 \right]_l \tag{5.30}$$

Note that, the above approximation involves a cyclic summation over all the boundary sides of the domain Ω'.

There is an important effect on the accuracy of the numerical solution due to the choice of the formulation for evaluation of the velocity gradients. Workers on finite volume methods have introduced different formulations for the evaluation of velocity gradients and viscous stresses. Here, a brief review of some of the formulations is presented.

5.6.1 Cell-centred Formulations of the Viscous Stresses

For cell-centred finite volume schemes, a number of formulations have been adopted for the computation of velocity gradients around each side. Beaven [6] examined different stencils on both quadrilateral and triangular two-dimensional regular grids. In the first stencil (Figure 5.4.a), he used the average of the velocity components stored in the neighbouring cells to each side. Then he computed the velocity gradients of each cell using the average velocity at its surrounding sides. Finally, the average value of the velocity gradient of two neighbouring cells was considered.

The second stencil (Figure 5.4.b) was formed by joining two neighbouring cells. In this case, the mean value of the variables stored at the centre of neighbouring cells of sides surrounding the secondary cell is used for the direct computation

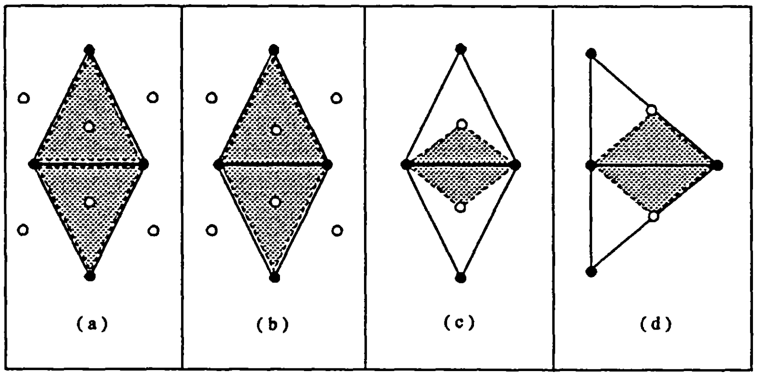

Figure 5.4 *Cell-centred stencils for computing viscous stresses*

of contour integral of velocity gradients associated with the central side of the secondary cell.

Natakusumah [7] applied another type of secondary cell by joining the centre of two neighbouring cells to nodes of the sharing sides (Figure 5.4.c). Therefore, in order to compute viscous stresses, he utilised both the variables at the centre of two neighbouring cells and the variables obtained for the nodes of the proposed side. This stencil produces good results for the two-dimensional meshes in which the cells are inverse symmetric about the edges.

The stencil proposed by Natakusumah was modified by Beaven [6] for the meshes with symmetric cells about edges (Figure 5.4.d). In order to find closer points to the centre of the required side, he used two mid points of the surrounding sides of the neighbouring cells instead of their vertices. After comparing the results for cell-centred stencils on given test cases, he concluded that the fourth stencil (Figure 5.4.d) produces more accurate results. Note that, this formulation is introduced only for two-dimensional triangular meshes.

5.6.2 Cell-vertex Formulations of the Viscous Stresses

The first formulation for the computation of the viscous stresses is applied by several numerical workers [8-10]. In this type, the secondary cell around each side is formed by a combination of two cells sharing that side (Figure 5.5). The geometry of this stencil is very similar to the second stencil type of the cell-centred scheme. The computation of the velocity gradients can be completed by using the average values of the variable at surrounding sides of the secondary cell. In this case, the equation (5.29) can be used to complete the velocity gradients with the value of N_{side}, i.e. the number of sides surrounding each secondary cell taken as

$$2D \ : \ N_{side} \ = \ [\ 4 \ \textit{inside domain} \ / \ 3 \ \textit{on boundary}\ ]$$
$$3D \ : \ N_{side} \ = \ [\ 6 \ \textit{inside domain} \ / \ 4 \ \textit{on boundary}\ ]$$

In practice, the computation of the diffusive fluxes are performed in two steps. The first step, involves the computation of the first order derivatives of the velocity components u_1, u_2 and u_3, which is necessary for the computation of the viscous stresses τ_{ij} at boundary side k of every control volume. The velocity gradients can be evaluated directly at the sides of the control volumes. Since first order derivatives were obtained by using the cell-vertex type stencil, the components of the viscous stress tensor associated with each control volume can be computed by accumulating the contribution of all the boundary sides. Hence, the second step involves the computation of the viscous flux associated with the central node of every control volume by utilising stresses τ_{ij} at its boundary sides. Similar to the convective flux computations, the viscous flux of each side is added to its neighbouring control volumes with opposite sign (Figure 5.1). This procedure can be done using a loop over all the sides using a side-based data structure. This formulation produces acceptable results on the meshes in which the cells are inverse symmetric about sides.

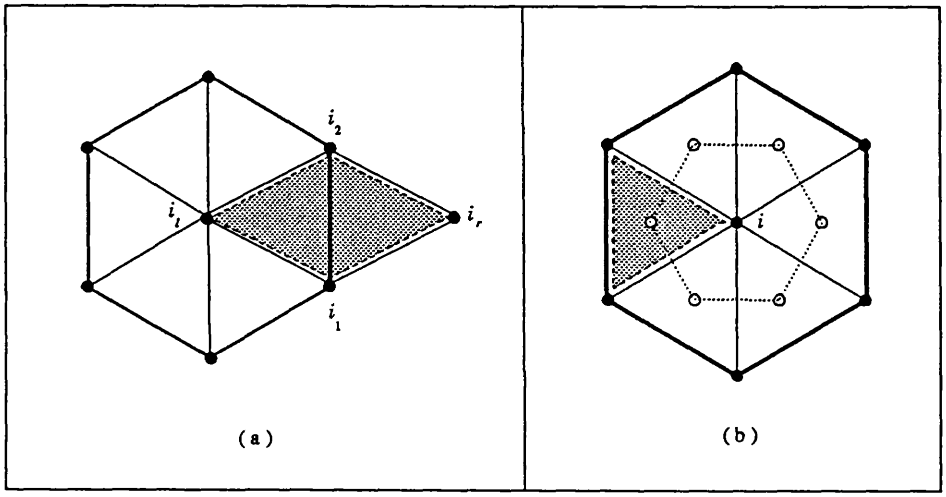

Figure 5.5 *Two different cell-vertex stencils for computing viscous term*

The second formulation for the cell-vertex scheme is derived from a Galerkin Finite Element formulation for triangular meshes [12]. This formulation considers the value of a variable within the triangular cell to be given by a combination of the values defined at the vertices of the cell, weighted according to the geometric shape function defined at each node (Figure 5.5.b). Although the computations of fluxes are made for each cell, gathering all the contour integrals of the cells associated with a control volume will cancel the contributions from interior sides. Hence, similar to the cell-vertex finite volume scheme, only the contribution of boundary sides of the control volume will be effective in the flux computations. This formulation is applied for the cell-vertex finite volume scheme in two dimensions [11]. In order to make the thesis self-contained, a brief description of the formulation, and extension to three dimensions, is given in Appendix **B**.

In order to employ the Galerkin finite element formulation for the present cell-vertex scheme, equation (5.29) can be used. The computations are performed by a loop through the mesh cells calculating the contributions to equation (5.29) and assigning this to the control volumes associated with all the nodes of that particular cell. Hence, the calculations can be done either by using side-based or cell-based connectivity data. This formulation is able to produce acceptable results for all types of regular and irregular meshes.

Another alternative is to modify the stencil for the computation of the velocity gradients in the cell-vertex finite volume formulation. This formulation is developed based on the idea of the fourth cell-centred two-dimensional type stencil (Figure 5.4.d) to improve the shape of the cell-vertex secondary cells (Figure 5.5.a), associated with regular criss-cross and irregular meshes. The vertices of this stencil are formed by using two auxiliary points m_1 and m_2 located on the intersection of line normal to the mid point of side k and boundary of the neighbouring cells (Figure 5.6). Thus, these points are different from the neighbouring nodes of the proposed side i_1 and i_2. Having found the coordinates of these auxiliary points, the velocities at these two points can be interpolated from the nodes on the related boundary side of the secondary cell. The position of the extra points and the interpolation coefficients can be computed prior to starting the computations. The derived formulations for the position of points and interpolation coefficients can be found in Appendix **C**.

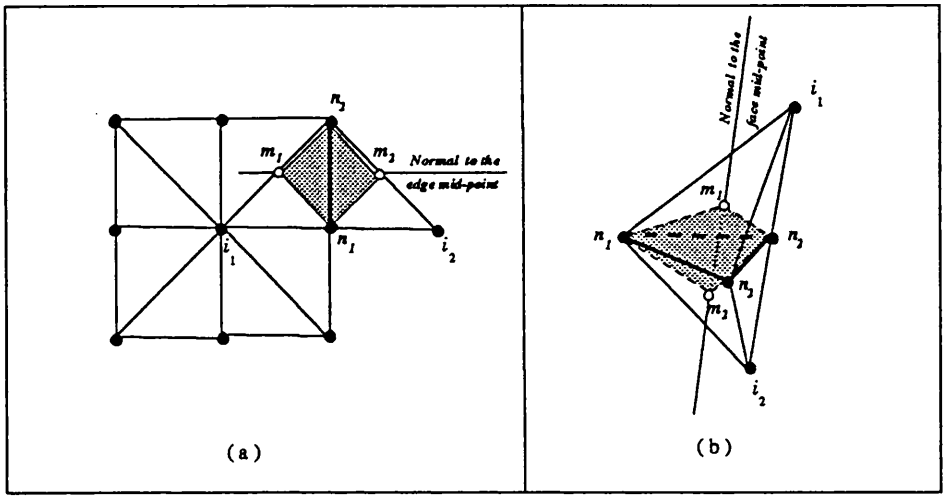

Figure 5.6 *Modified cell-vertex type secondary cell*
a) Two dimensional and b) Three dimensional

For this stencil (Figure 5.6), the velocity gradient for each side k can be computed similar to the cell-vertex finite volume formulation. The only difference is that instead of two neighbouring nodes i_1 and i_2 of each side, two auxiliary points of m_1 and m_2 are utilised for the computation of viscous stresses associated with side k. Then the computed stresses τ_{ij} are added to the central nodes of neighbouring control volumes of that side. This method can be applied for both two and three-dimensional irregular triangular meshes and produces acceptable results.

5.7 Summary

In this chapter, the space discretisation of the convective and viscous terms are described. The discretisation of the convective term is achieved by using a central-differencing scheme. This scheme requires the addition of an artificial dissipation to achieve stability.

For the present algorithm, the edge-based dissipation is applied by computing the Biharmonic operator, equation (5.21), for both two and three-dimensional cases. The Biharmonic operator can be scaled proportional to the maximum local eigenvalue of the flux Jacobian matrices. In order to minimise the value of the artificial dissipation, the Biharmonic operator can be scaled using the minimum scaling factors associated with two end nodes of the edges. Generally, in order to improve the accuracy and convergence of the computations on unstructured meshes, application of the normalisation of the dissipation term is recommended.

The accuracy of the viscous term is largely dependent upon the formulation used for the computation of first order velocity derivatives of equation (5.27). For cell-vertex schemes, generally two common formulations of the finite volume and Galerkin finite element are applicable for the evaluation of the viscous terms on triangular and tetrahedral meshes. The cell-vertex finite volume formulation produces accurate results, on the meshes in which the cells are inverse symmetric about the sides. Such mesh can be easily generated in two dimensions. The Galerkin finite element formulation presents good results for all types of irregular triangular meshes. Another alternative is proposed by modifying the stencil for the computation of the viscous terms. Application of this modification facilitates the accurate computation of the viscous terms on any type of two and three-dimensional irregular triangular and tetrahedral meshes. However, this formulation requires an additional set of points for the modification of the stencil for the computation of the viscous stresses. Hence, the Galerkin finite element formulation has been used to simulate viscous flow on three-dimensional unstructured irregular tetrahedral grids.

References

[1] A. Jameson, W. Schmidt and E. Turkel, 'Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes', AIAA Paper 81-1256, June (1981)

[2] C. Hirsch, 'Numerical Computation of Internal and External Flows', Vol. 1, Fundamentals of Numerical Discretisation, Wiley, (1989).

[3] S. R. Chakravarthy and S. Osher, 'A New Class of High Accuracy TVD Schemes for Hyperbolic Conservation Laws', AIAA-85-0363 , (1985).

[4] N. T. Frink, P. Parikh and S. Pirzadeh, ' A Fast Upwind Solver for the Euler Equations on Three-Dimensional Unstructured Meshes' , AIAA paper 91-0102, 29th Aerospace Sciences Meeting , (1991).

[5] M. Aftosmis, D. Goitonde, and T. S. Tavares, 'On the Accuracy, Stability and Monotonicity of Various Reconstruction Algorithms for Unstructured Meshes', AIAA paper 94-0415, (January 1994).

[6] F. Beaven, 'Numerical Solution of the Navier-Stokes Equations on Generalised Grids', Ph.D. Thesis, Civil Eng. Dept., University of Wales, (1994).

[7] D. K. Natakusumah, 'Finite Volume Solution of the Two-dimensional Compressible Flow Equations on Structured and Unstructured Hybrid Grids', Ph.D. Thesis, Civil Eng. Dept., University of Wales, (1992).

[8] N. P. Weatherill, L. J. Johnston, A. J. Peace and J. A. Shaw, 'Development of a Method for the Solution of the Reynolds-Averaged Navier-Stokes Equations on Triangular Grids', ARA Report 70, (1986).

[9] R. C. Swanson, E. Turkel, 'A Multistage Time-stepping Scheme for the Navier-Stokes Equations', AIAA paper 85-0035 (1985).

[10] L. Martinelli, A Jameson, F. Grasso, 'A Multi-grid Method for the Navier-Stokes Equations', AIAA paper 86-0208 (1986).

[11] D. J. Maviriplis, A. Jameson, L. Martinelli, 'Multi-grid Solution of the Navier-Stokes Equations on Triangular Meshes', AIAA-89-0120 , (1989).

[12] O. C. Zienkiewicz, 'The Finite Element Method ', New York, McGraw-hill, (1977).

[13] A. Jameson, 'A Vertex-based Multi-grid Algorithm for Three-dimensional Compressible Flow Calculations', ASME Symposium on Numerical Methods for Compressible flow, Annahim, (December 1986).

[14] A. Jameson, T. J. Baker and N. P. Weatherill, 'Calculation of Inviscid Transonic Flow over a Complete Aircraft', AIAA Paper 86-0103, (1986).

[15] H. Luo, J. D. Baum, R. Lohner and J. Cabello, 'Adaptive Edge-based Finite Element Schemes for the Euler and Navier-Stokes Equations on Unstructured grids', AIAA-93-0336, (January 1993)

[16] A. Jameson, 'Transonic Aerofoil Calculations Using Euler Equations','In Numerical Methods in Computational Aerodynamics', Ed. P.L. Roe, Academic Press, (1987).

[17] D. J. Mavriplis and A. Jameson, 'Multi-grid Solution of the Navier-Stokes Equations on Triangular Meshes', AIAA Journal, **28**, No. 8, pp. 1415–1425, (1990)

[18] N. P. Weatherill, O. Hassan, D. L. Marcum, 'Calculation of Steady Compressible Flow-fields with the Finite Element Method', AIAA-93-0341, (January 1993).

[19] J.Von Nuemann and R. D. Richtmyer, 'A Method for the Numerical Calculations of Hydro-dynamical Shock' , J. of Math. Physics,**20** ,(1950)

[20] R. W. MacCormak and B. S. Baldwin, ' A Numerical Method for Solving The Navier-Stokes Equations with Application to Shock-Boundary layer Interaction' , AIAA paper 75-1, (1975)

[21] J. L. Steger, ' Implicit Finite Difference Simulation of Flow about Two-Dimensional Geometries' AIAA Journal **16**, pp. 679-686, (1978)

[22] R. M. Beam and R. F. Warming, ' Implicit Numerical Methods for the Compressible Navier-Stokes and Euler Equations' , Von Karman Institute Lecture Series 1982-04, Rhode Saint Genese, Belgium, (1982).

[23] C. H. Sung, C. W. Lin, 'Numerical Investigation on Effect of Fairing on the Vortex Flow around Airfoil/ Flat-plate Junctures', AIAA paper 88-0615 , (1988).

[24] R. C. Swanson, E. Turkel, 'Artificial Dissipation and Central Difference Schemes for the Navier-Stokes Equations', ICASE Report 87-0029 (1987).

[25] J. Dreyer, 'Finite Volume Solution to the Steady Incompressible Euler Equations on Unstructured Triangular Meshes', M.Sc. Thesis, MAE Dept., Princeton University, (1990).

[26] A. Belov, L.Martinelli, A. Jameson, 'A New Implicit Algorithm with Multi-grid for Unsteady Incompressible Flow Calculations', AIAA Paper 95-0049, (1995).

[27] J. Farmer, L.Martinelli, A. Jameson, 'A Fast Multi-grid Method for Solving Incompressible Hydrodynamic Problems with Free Surfaces ', AIAA Paper 93-0767, (1993).

[28] L. A. Sykes, 'Development of A Two-dimensional Navier-Stokes Algorithm for Unstructured Triangular Grids', ARA Report 80, April 1990.

6

Numerical Solution Technique

6.1 Introduction

After the space discretisation of the fluxes, the time integration of the governing equations can be performed following the explicit multi-stage Runge-Kutta time stepping proposed by Jameson et al [1,8]. The details of the central-difference approximation to the space discretisation of the convective and diffusive flux vectors on unstructured triangular meshes were presented in Chapter 5. In this chapter, the details of the numerical procedure for the time integration of the governing equations is described. The description is given with reference to the general formulation of the cell-vertex finite volume descritization of the integral form of the Navier-Stokes equations, which was presented in Chapter 4. The time stepping of the set of discretised equations is described in Section 6.2. In Section 6.3, a general introduction to different boundary conditions is presented. Finally, some numerical techniques for accelerating convergence and increasing computational efficiency of the numerical scheme are reviewed in Sections 6.4 and 6.5, respectively.

6.2 Time Stepping

An explicit multi-stage Runge-Kutta scheme [1] has been implemented to integrate the equations to steady state. After spatial discretisation, equation (5.4) can be transformed into a set of coupled ordinary differential equations

$$\frac{d}{dt}(\mathbf{W}_i) \ + \ \frac{\mathbf{P}_\mathrm{i}}{\Omega_i} R(\mathbf{W}_i) \ = 0 \qquad\qquad i = 1, 2, \ldots, N_{total} \qquad (6.1)$$

where Ω_i is the control volume associated with node i and N_{total} is the total number of nodes. The preconditioning matrix $\mathbf{P}_\mathrm{i}$ is computed using the local velocities at i. Here, $R(\mathbf{W}_i)$ is the total residual of the convective and viscous terms defined as

$$R(\mathbf{W}_i) \ = \ \Big[C(\mathbf{W}_i) - V(\mathbf{W}_i)\Big] \qquad (6.2)$$

As mentioned before, the application of the artificial compressibility approach to the set of incompressible Navier-Stokes equations destroys the time accuracy of the system of equations. This is because of the introduction of the pseudo transient term in the continuity equation. Hence, for the present algorithm the time stepping can be viewed as an explicit relaxation to steady state.

Various integration methods have been developed for a system of ordinary differential equations as in equation (6.1). Details of such approaches as linear multiple-step methods, predictor corrector schemes Runge-Kutta schemes can be found elsewhere [1-5]. In the present work, the explicit multi-stage Runge-Kutta time stepping scheme of Jameson et al [1] has been implemented.

Consider $\mathbf{W}_i^n$ as the value of $\mathbf{W}_i$ after n pseudo time steps. The basic idea of the Runge-Kutta scheme is to evaluate the differential system (6.1) at several values of $\mathbf{W}$ in the interval between $n\Delta t$ and $(n+1)\Delta t$ and to combine them in order to obtain an approximation for $\mathbf{W}_i^{n+1}$. The general form of an m–stage Runge-Kutta scheme is described as

$$\mathbf{W}_i^{(0)} = \mathbf{W}_i^n$$
$$\mathbf{W}_i^{(1)} = \mathbf{W}_i^{(0)} - \theta_1 \frac{(\Delta t_i \mathbf{P}_\mathrm{i})^{(0)}}{\Omega_i} R(\mathbf{W}_i^{(0)})$$

$$\mathbf{W}_i^{(2)} = \mathbf{W}_i^{(0)} - \theta_2 \frac{(\Delta t_i \mathbf{P}_\mathrm{i})^{(1)}}{\Omega_i} R(\mathbf{W}_i^{(1)}) \tag{6.3}$$

$$\cdot$$

$$\cdot$$

$$\begin{aligned}
\mathbf{W}_i^{(m)} &= \mathbf{W}_i^{(0)} - \theta_m \frac{(\Delta t_i \mathbf{P}_\mathrm{i})^{(m-1)}}{\Omega_i} R(\mathbf{W}_i^{(m-1)}) \\
\mathbf{W}_i^{n+1} &= \mathbf{W}_i^{(m)}
\end{aligned}$$

where Δt is the discrete time step, and $\theta_1 \ldots \theta_m$ are coefficients particular to the number of stages.

The computation of the residual terms at every stage can be very expensive. A cheaper alternative is to use the hybrid formulation. With this formulation, the convective operator is evaluated at every stage in the time step, but the time step limit, the viscous operator $V(\mathbf{W}_i)$, and preconditioning matrix $\mathbf{P}_\mathrm{i}$, are only evaluated at the first stage. Thus, the computation of time stepping will be in the form of

$$\begin{aligned}
\mathbf{W}_i^{(0)} &= \mathbf{W}_i^{n} \\
\mathbf{W}_i^{(1)} &= \mathbf{W}_i^{(0)} - \theta_1 \frac{(\Delta t_i \mathbf{P}_\mathrm{i})^{(0)}}{\Omega_i} \left[C(\mathbf{W}_i^{(0)}) - V(\mathbf{W}_i^{(0)}) \right] \\
\mathbf{W}_i^{(2)} &= \mathbf{W}_i^{(0)} - \theta_2 \frac{(\Delta t_i \mathbf{P}_\mathrm{i})^{(0)}}{\Omega_i} \left[C(\mathbf{W}_i^{(1)}) - V(\mathbf{W}_i^{(0)}) \right] \qquad (6.4)
\end{aligned}$$

$$\cdot$$

$$\cdot$$

$$\begin{aligned}
\mathbf{W}_i^{(m)} &= \mathbf{W}_i^{(0)} - \theta_m \frac{(\Delta t_i \mathbf{P}_\mathrm{i})^{(0)}}{\Omega_i} \left[C(\mathbf{W}_i^{(m-1)}) - V(\mathbf{W}_i^{(0)}) \right] \\
\mathbf{W}_i^{n+1} &= \mathbf{W}_i^{(m)}
\end{aligned}$$

where $\theta_1 \ldots \theta_m$ are coefficients of the proposed multi-stage scheme. Two schemes of five and three-stage time stepping have been tested for calculations in the present work. For five-stage schemes the coefficients $\theta_1 \ldots \theta_5$ are defined as

$$\theta_1 = 1/4 \qquad \theta_2 = 1/6 \qquad \theta_3 = 3/8 \qquad \theta_4 = 1/2 \qquad \theta_5 = 1$$

and for three-stage schemes the coefficients $\theta_1 \ldots \theta_3$ are defined as

$$\theta_1 = 0.6 \qquad \theta_2 = 0.6 \qquad \theta_3 = 1.0$$

The time step Δt must be chosen to be consistent with the stability limitation of the governing equations [4,5]. Since the system of modified equation is cast similar to the Navier-Stokes equations, a time stepping limit can be adopted which originally was developed for the compressible equations [8]. Thus, the time step limit for each control volume can be defined as

$$\Delta t_i = \frac{1}{\lambda_i^c/\Omega_i + \lambda_i^d/\Omega_i^2} \tag{6.5}$$

here λ_i^c can be defined as the characteristic of convection by using the system wave of the governing equations [9] as

$$\lambda_i^c = \sum_{k=1}^{N_{side}} \left[\mid \vec{\mathrm{U}} \cdot \Delta\vec{\mathrm{S}} \mid + \sqrt{\left(\vec{\mathrm{U}} \cdot \Delta\vec{\mathrm{s}}\right)^2 + \beta^2 \left(\Delta\vec{\mathrm{s}} \cdot \Delta\vec{\mathrm{s}}\right)} \right]_k \tag{6.6}$$

where $\Delta\vec{\mathrm{s}}$ is the vector of projected areas of the sides surrounding the control volume given by equation (4.14). $\vec{\mathrm{U}}$ is the average velocity vector of the proposed side k as

$$\vec{\mathrm{U}} = u_1 \hat{\mathbf{I}}_1 + u_2 \hat{\mathbf{I}}_2 + u_3 \hat{\mathbf{I}}_3$$

$$u_i = \frac{1}{N_{dim}} \sum_{n=1}^{N_{dim}} (u_i)_n \tag{6.7}$$

where $\hat{\mathbf{I}}_1$, $\hat{\mathbf{I}}_2$ and $\hat{\mathbf{I}}_3$ are the unit vectors in the x_1, x_2 and x_3 directions, respectively. This velocity can be used in equation (3.28) for computing β^2. Here, N_{dim} is number of nodes of each side which in triangular meshes is 2 and in tetrahedral meshes is 3.

λ_i^d represents the maximum eigenvalue of the viscous operator and can be defined as

$$\lambda_i^d = \sum_{k=1}^{N_{side}} \frac{2\mu}{\rho} \left(\Delta\vec{\mathrm{s}} \cdot \Delta\vec{\mathrm{s}} \right)_k \tag{6.8}$$

where N_{side} is the number of sides surrounding control volume i.

Note that the time step Δt_i is computed for the centre node of the control volumes by cyclic summation over sides surrounding the control volumes. These computations can be done prior to the multi-stage procedure.

The global time step limit of the system is dictated by the minimum value of the time step Δt_i in the computational domain. However, for the explicit schemes the limit on time step for which the integration of convective dominated flow is stable, is dictated by the CFL condition. This condition implies that $\Delta t \leq \mathrm{CFL}/|\lambda^c_{max}|$. The constant CFL depends on the particular applied multi-stage scheme. A conservative estimate for the general time step limit of the flow computation leads to following condition

$$\Delta t_{max} \leq \frac{CFL}{\left(\lambda^c_i/\Omega_i + \lambda^d_i/\Omega^2_i\right)_{max}} \tag{6.9}$$

With the application of the artificial compressibility approach, the stability of the numerical scheme is directly related to the choice of the parameter β^2, which appears in the preconditioner **P**. Several suggestions about the choice of parameters associated with β^2 are reviewed in Section 3.3.2. In order to verify these parameters, independent choices of β^2 have to be considered by examining proper parameters for the expression (3.28).

6.3 Boundary Conditions

The construction of appropriate boundary conditions represents one of the important tasks in the development of an algorithm. In general, for most incompressible problems with which we are dealing with in this work, the domain boundary can be divided into two major categories; first is a solid wall boundary and second is a far field boundary, which must be specified to produce a bounded computational domain. However, there are other types of boundaries such as symmetry boundaries, which may be used for limiting the domain i.e. in symmetric flow. Numerical formulations for this type of boundary is similar to the inviscid wall boundary conditions. The stability and the accuracy of the solution can be disturbed by improper treatment of the flow conditions specified on these boundaries.

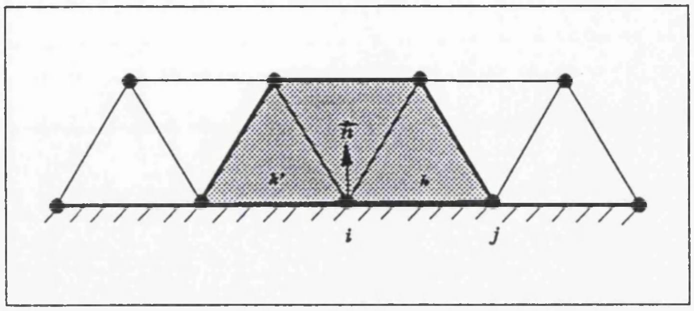

Figure 6.1 *Outward normal vector on the boundary*

Let k represent a boundary side, i represents a node associated with a control volume attached to the boundary and j represents a node on the boundary, then the outward unit normal vector $\vec{\mathbf{n}}$, (Figure 6.1) can be expressed as

$$\vec{\mathbf{n}} = n_1 \hat{\mathbf{I}}_1 + n_2 \hat{\mathbf{I}}_2 + n_3 \hat{\mathbf{I}}_3 \tag{6.10}$$

Using the projected area of the boundary sides perpendicular to the j cartesian axis, Δs_j, the component of $\vec{\mathbf{n}}$ can be defined as

$$\begin{aligned} n_1 &= \frac{\Delta s_1}{\mid \vec{\Delta \mathbf{s}} \mid} \\ n_2 &= \frac{\Delta s_2}{\mid \vec{\Delta \mathbf{s}} \mid} \\ n_3 &= \frac{\Delta s_3}{\mid \vec{\Delta \mathbf{s}} \mid} \end{aligned} \tag{6.11}$$

where

$$|\vec{\Delta \mathbf{s}}| = \sqrt{\vec{\Delta \mathbf{s}} \cdot \vec{\Delta \mathbf{s}}} \tag{6.12}$$

The components of the vector of area $\vec{\Delta \mathbf{s}}$ are defined in Chapter 4.

By cyclic summation of all the contributions of the boundary sides k, associated with the node i on the boundary, the resultant unit normal at node i can be found as

$$\vec{\mathbf{n}}_i = \frac{1}{N_{side}} \sum_{k=1}^{N_{side}} \vec{\mathbf{n}}_k \tag{6.13}$$

where N_{side} is number of sides sharing node k.

6.3.1 Solid Wall Boundary Conditions

The surface of the solid wall boundary is normally considered as a rigid impermeable wall, so that no mass flux passes through the boundary. Depending upon the condition of the inviscid or viscous flow, the flow tangency or no slip condition may be implemented.

We can use the components of the normal at a boundary node to define the component of the velocity vector normal to the boundary at node i as

$$\begin{aligned} \vec{\mathbf{U}}_i^n &= \vec{\mathbf{U}} \cdot \vec{\mathbf{n}} \\ &= \Big[(u_1 n_1)\hat{\mathbf{I}}_1 + (u_2 n_2)\hat{\mathbf{I}}_2 + (u_3 n_3)\hat{\mathbf{I}}_3\Big]_i \end{aligned} \tag{6.14}$$

For inviscid or viscous flow, the normal component of the velocity flux across that boundary should be zero. In order to impose zero velocity flux through wall boundaries, it is necessary to set all the components of normal velocity $\vec{\mathbf{U}}_i^n$ to zero. This can be easily done by

$$(\vec{\mathbf{U}}_i)_{Inviscid} = \vec{\mathbf{U}}_i - \vec{\mathbf{U}}_i^n \tag{6.15}$$

where $\vec{\mathbf{U}}_i^*$ is the velocity at wall boundary node i, after zero normal velocity correction.

In viscous flow, because of the no slip condition on solid walls, there should be no velocity component on solid wall boundary nodes. Thus,

$$(\vec{\mathbf{U}}_i)_{Viscous} = \vec{\mathbf{U}}_i - \vec{\mathbf{U}}_i \tag{6.16}$$

In practice, the process of computing the corrected velocity flux through side k is performed by introducing the corrected $\vec{\mathbf{U}}_i$ through the stages of the time stepping.

6.3.2 Symmetry and Side-slip Boundary Conditions

For symmetric problems, in order to reduce the computational load and memory overhead, it is reasonable to compute a part of the flow domain which can be separated by symmetric lines in two dimension or symmetric surfaces in three dimensions. The inviscid flow over a sphere is an example of this type where the flow is symmetric to the line parallel to the free stream flow direction and passing through the centre of the sphere. In some cases, it is necessary to impose side-slip boundaries, which can be used to separate a part of a flow domain to compute the details of the flow over a certain part. Imposing side-slip boundaries on the sides of a flat plate for simulation of the laminar boundary layer is an application of this type. Using equations (6.14) and (6.15), the procedure of computing inviscid solid body boundary conditions can be applied to impose zero normal velocity through this imaginary wall.

6.3.3 Far Field Boundary Conditions

A complication arises when the solution of the governing equations is sought on an unbounded domain. However, for practical purposes a numerical calculation must be performed in a finite domain. Therefore, far field boundary conditions which are capable of simulating the properties of the infinite flow-field should be implemented.

Since imposing a free stream value at the outer boundary will not allow information to propagate out of the domain, this information will reflect back to the domain and disturb the computations. Extrapolations may however introduce unwanted reflection back into the domain. Thus, they must be applied with caution. Therefore, it is necessary to use elaborate techniques such as non-reflecting boundary conditions, based upon the elimination of information propagating along the characteristic lines.

The direction of propagation of information can be deduced from the sign of the eigenvalues of the convective part of the Jacobian matrix, which are described in Chapter 3. Remember that, although the eigenvalues λ of the convective part of the governing equations, equations (3.18), are real, they are not all positive.

Hence, according to the sign of the eigenvalues, there are three incoming waves and one outgoing wave. Therefore, in three-dimensional problems, there are three right running waves at the inflow and one left running wave at the outflow (Figure 6.2). Note that, for two-dimensional flow, the number of right running waves decreases to two. Dependent upon these directions, the values at the far field boundary nodes can be determined using either free stream values or values of variables inside the domain.

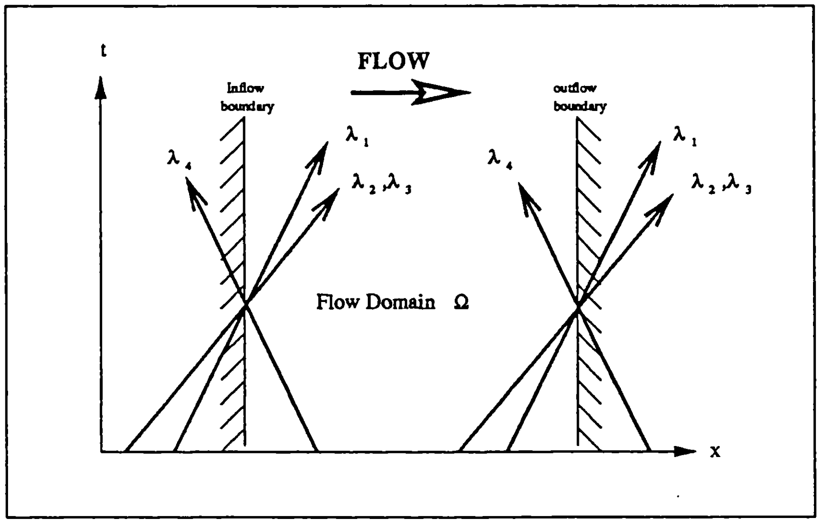

Figure 6.2 *Incoming and outgoing characteristic waves*

In the present work, the proposed far field boundary condition of Soh [10] has been adopted for imposing free stream values or extrapolating variables on the inflow or outflow far field boundaries. This technique enforces free stream velocity components as incoming characteristics at inflow and pressure as incoming characteristic at outflow. Since, the upstream dependency is dominant in the situation of incompressible flow, to prevent the effect of outgoing waves to the computational domain the velocities at outflow boundary can be extrapolated from neighbouring nodes inside the domain. Thus, the velocities at the outflow boundary can be extrapolated from upstream. For the inflow boundary, considering zero gradient for the pressure close to boundary, the velocity components may be extrapolated from inside the domain. It should be noted that, the inflow and outflow boundary can be distinguished from the sign of the dot product of computed velocity vector and normal vector to the boundary surfaces, $(\vec{n}_i \cdot \vec{U}_i)$.

6.3.4 General Remarks about Boundary Conditions

In the early stages of time stepping, the sudden implementation of solid wall boundary conditions, will result in high discontinuities in flow variables. Thus, imposing solid wall boundary conditions should be done gradually. The boundary conditions should be imposed over a number of iterations using a relaxation parameter. At the intersection of inviscid or symmetry walls, the resulting normal vectors will not be in the direction perpendicular to either of the walls. Hence, a special treatment for these boundaries is required. The zero normal velocity can be imposed individually for each of the wall boundaries associated with the intersection of the wall and symmetry boundaries.

Note that, even for viscous flows, where the flux velocity $\vec{\mathbf{U}}_i$ at the wall boundary side is set to zero, the components of the momentum flux remains non-zero due to contributions from the pressure term. Therefore, for the flux distribution of side k which lies on the boundary a special formulation is required. This is done by adding the contribution of the values of the computed fluxes of the boundary side k to its end nodes, which are located on the boundary of their own control volumes (Figure 6.1).

6.4 Convergence Acceleration

Various indicators can be used to measure the speed of convergence, such as root mean square of the pressure and velocity residuals, or calculating convergence rate r_c at each time $t = t^n$. For instance for pressure, the formulation can be adopted as

$$\left(\frac{R_p}{\Delta t_{min}}\right)^{t=n\Delta t} = r_c{}^{n-1}\left(\frac{R_p}{\Delta t_{min}}\right)^{t=0} \tag{6.17}$$

where Δt_{min} is the minimum time step computed throughout the nodes of the domain, and R_p is the root mean square of the pressure residuals. The general formulation for the calculation of the root mean square of the residuals can be defined as

$$R_{\mathbf{W}_i} = \sqrt{\frac{1}{N_{nodes}} \sum_{i=1}^{N} [R(\mathbf{W}_i)]^2} \qquad (6.18)$$

where N_{nodes} is the number of nodes inside the domain and the residuals $R(\mathbf{W}_i)$ are given in equation (6.1).

To accelerate the convergence, several standard techniques have been applied. Those implemented in the present work involve the use of local time stepping and residual smoothing.

6.4.1 Local Time Stepping

It is common to use minimum Δt_i in order to guarantee the time accuracy of the numerical procedure, but it will reduce the speed of convergence to the minimum time step of the domain.

For an explicit scheme, the maximum permissible time step at each control volume associated with node i, which is determined by its stability limit, may be used to accelerate the convergence to steady state. In practice, the time step Δt_i is computed by looping over each node using expression (6.5), which results in an average time step for each control volume. The average time step limit is then used as a local time step to advance the solution in that particular control volume. This will ensure that the disturbance in the solution is propagated across the domain in a number of time steps proportional to the control volume size. In such a case, although the minimum time step of the domain affects the global convergence rate, the convergence of other control volumes are not restricted to the minimum time step.

In the present algorithm, since only steady state solutions are of interest, the time accuracy of the scheme is no longer important, and hence, modification to the time stepping scheme can accelerate the convergence to steady state.

6.4.2 Residual Smoothing

Another method to modify the time step limit is to replace the residual at one control volume, defined in equation (6.2), by a weighted average of the residuals of neighbouring control volumes. The average is calculated implicitly as

$$R_i' = R_i + \epsilon \nabla^2 R_i' \tag{6.19}$$

where ϵ is a weighting parameter, R_i' is the smoothed residual and $\nabla^2 R_i'$ is the undivided Laplacian of the residual, which can be defined as

$$\nabla^2 R_i' = \sum_{m=1}^{N_{node}} (R_m' - R_i') \tag{6.20}$$

where N_{node} is the total number of neighbouring nodes inside the control volume. Substituting equation (6.20) into (6.19) gives the following expression for the smoothed residuals

$$R_i' = \frac{R_i + \epsilon \sum_{m=1}^{N_{node}} R_m'}{1 + \epsilon N_{node}} \tag{6.21}$$

This implicit system is then solved by performing several Jacobi iterations

$$R_i^{(n)} = \frac{R_i^{(0)} + \epsilon \sum_{m=1}^{N_{node}} R_m^{(n-1)}}{1 + \epsilon N_{node}} \tag{6.22}$$

where $R_i^{(0)}$ refers to the unsmoothed residuals R, and with n as the final iteration, $R_i^{(n)}$ will be the smoothed residuals R'. Here, ϵ is an empirical coefficient.

6.5 Computational Efficiency

In order to have a practical flow solver, it is necessary to ensure that the algorithm is efficient. This can be done by applying appropriate data structures to facilitate access to the essential set of geometrical data, which are coordinates of nodes of each side and the nodes of neighbouring control volumes. This issue motivates the use of an indirect addressing system based on a side data structure. In addition, the algorithm should be able to take advantage of the capability of modern super computers, in particular, hardware which has a vector architecture. These issues will be summarised in the rest of this section.

6.5.1 Side-based Data Structure and Indirect Addressing

In the proposed algorithm, fluxes are to be calculated across sides of the grid, and hence, the side is the most basic element of the grid. The flux computation of the present method involves a single loop over all the sides of the mesh, assigning the calculated flux at a side to both control volumes adjacent to that side.

A side-based data structure can be constructed for expressing the connectivity data, and the flux balances can be performed by a single loop over sides, using an indirect addressing system. The mesh information can be stored to two matrices. The first one includes coordinates of nodes and the second one consists of a side connectivity matrix, which stores all necessary information concerning the sides of control volumes.

To define a side-based data structure, each row of the connectivity matrix contains specific integer labels of information about one side. Since the cell-vertex scheme works in terms of sides and neighbouring nodes, only triangles or tetrahedra can be treated within the grid. To ensure that the correct sense of contour integration is taken, the numbering system must preserve the anti-clockwise sense in each triangle. In order to impose this role, the order of the nodes associated with each side in the data structure should conform to the following arrangement.

In two-dimensional grids the side-based data structure includes the numbering of four nodes. The first and the second are the two end points of the edges. The third and the fourth indicate the nodes which form the triangular cells located to the left and the right hand of the proposed edge, respectively. Note that an anti-clockwise direction should be considered for the definition of the nodes of the triangular cell which is located in the left hand side.

For three-dimensional meshes, the set of data for each face includes five node numbers. The first three numbers define the nodes of the proposed triangular face. The fourth and fifth nodes indicate the node of the tetrahedral cells located on the left and right hand sides of the proposed face, respectively. Therefore, for each cell the appropriate sign for the flux contribution for each tetrahedral cell is automatically determined.

Using indirect addressing system, a loop over all grid sides can be devised to calculate the flux balance of all control volumes. The contribution of each side is calculated only once in this loop. The results are then added to or subtracted from the results stored in the neighbouring control volumes of each side. As an example, part of the code, which performs evaluation of the volume of the control volumes, is listed here.

Example code *File:* volume

```
c      Evaluate the volume of each computational control volume

      do 10 is = 1 , nsides
        n1    =   Isides(is,1)
        n2    =   Isides(is,2)
        n3    =   Isides(is,3)
        nl    =   Isides(is,4)
        nr    =   Isides(is,5)
*
        dx = ((Coord(n3,2) -Coord(n1,2)) * (Coord(n2,3) -Coord(n1,3)) -
     &        (Coord(n2,2) -coord(n1,2)) * (Coord(n3,3) -Coord(n1,3)))/2.
        dv =   ( Coord(n1,1) + Coord(n2,1) + Coord(n3,1)) * dx / 3.
*
       Volume ( nl ) = Volume ( nl ) + dv
       Volume ( nr ) = Volume ( nr ) - dv
  10   continue
```

End of file

6.5.2 Code Vectorisation

The modern vector computer is capable of vectorising the computational loops which are required in the calculation of the flux balance and time stepping. For vectorisation, it is essential that the address labels of the control volumes should not be accessed more than once in a proposed single loop. Therefore, the procedure is limited by the appearance of recurrences within the loops used by the finite volume method. Thus, the sides can be reordered and subdivided into a number of groups such that, within each group, all side addresses are accessed no more than once. This process is known as colouring and is normally performed before the execution process. Note that, for practical reasons, the boundary sides are separated from the internal sides of the computational domain. In the present work the colouring process is performed by looping over all sides and then assigning each side to a particular colour. As an example, part of the vectorised code, which performs the evaluation of the volume of the control volume is listed here.

Example code *File:* `vectorised.volume`

```
c      Evaluate the volume of each computational control volume

      do 20 ic = 1 , ncolsides
       nstr  =   Icolsid ( ic*2-1 )
       nend  =   Icolsid ( ic*2   )
c
c$dir no_recurrence
cdir$ ivdep
c
       do 10 is = nstr , nend
        n1    =   Isides(is,1)
        n2    =   Isides(is,2)
        n3    =   Isides(is,3)
        nl    =   Isides(is,4)
        nr    =   Isides(is,5)
*
        dx = ((Coord(n3,2) -Coord(n1,2)) * (Coord(n2,3) -Coord(n1,3)) -
     &        (Coord(n2,2) -coord(n1,2)) * (Coord(n3,3) -Coord(n1,3)))/2.
        dv =   ( Coord(n1,1) + Coord(n2,1) + Coord(n3,1)) * dx / 3.
*
       Volume ( nl ) = Volume ( nl ) + dv
       Volume ( nr ) = Volume ( nr ) - dv
  10   continue
  20   continue
```

End of file

For increasing the efficiency of the computations, the colouring is controlled to ensure that all individual groups of colours contain approximately the same number of sides. In order to optimise the number of sides in groups of colours the following procedure is implemented. A loop is taken through the sides to check if any potential colour does not already exist in the two adjacent control volumes. An existing colour is used if the difference between the number of sides within that colour and all other colours will not be greater than one. If, after the checks, the side can not be assigned to a colour group then a new colour is created. This procedure is repeated until all sides have been assigned to a colour. A matrix containing the start and end numbers of each colour allows the division of single loop to multiple loops.

6.6 Summary

The aspects of the time integration and boundary conditions for the numerical solution of the governing equations are discussed in this chapter. The time stepping was used as a relaxation iteration to steady state. This is done by adopting a multi-stage Runge-Kutta scheme, which has been developed for the solution of the compressible form of the Navier-Stokes equations. Two multistage schemes were described. In the present algorithm, the three-stage scheme has been adopted for the case studies.

The implementation of the inviscid and viscous wall and far field boundary conditions has been discussed. A brief review of the techniques for increasing the computational efficiency, such as local time stepping, residual smoothing, code vectorisation and using a side-based data structured are described. Since the implicit procedure of the residual smoothing is somewhat expensive, it has not been used for three-dimensional computations. However, the application of the procedure of residual smoothing improves the convergence rate for computations in two dimensions.

References

[1] A. Jameson, W. Schmidt and E. Turkel, 'Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes', AIAA Paper 81-1256, June (1981)

[2] H. Luo, J. D. Baum, R. Lohner and J. Cabello, 'Adaptive Edge-Based Finite Element Schemes for the Euler and Navier-Stokes Equations on Unstructured grids', AIAA-93-0336, (January 1993)

[3] A. Jameson, 'Transonic Aerofoil Calculations Using Euler Equations', in 'Numerical Methods in Computational Aerodynamics', ed. P.L. Roe, Academic Press, (1987).

[4] L. A. Sykes, 'Development of A Two-dimensional Navier-Stokes Algorithm For Unstructured Triangular Grids', ARA Report 80, April 1990.

[5] N. P. Weatherill, L. J. Johnston, A. J. Peace and J. A. Shaw, 'Development of A Method for the Solution of the Reynolds-Averaged Navier-Stokes Equations on Triangular Grids', ARA Report 70, (1986).

[6] A. W. Rizzi, and M. Inouye, 'Time Split Finite Volume Method for Three-dimensional Blunt-body Flows', AIAA Journal, (1973).

[7] E. Turkel, 'Accuracy of Schemes with Non-uniform Meshes for Compressible Fluid Flows', ICASE Report No. 85-43, NASA Langley Research Centre, (1987).

[8] A. Jameson, T. J. Baker and N. P. Weatherill, 'Calculation of Inviscid Transonic Flow over a Complete Aircraft', AIAA Paper 86-0103, (1986).

[9] J. Dreyer, 'Finite Volume Solution to The Steady Incompressible Euler Equations on Unstructured Triangular Meshes', M.Sc. Thesis, MAE Dept., Princeton University, (1990).

[10] W. Y. Soh, 'Time Marching Solution of Incompressible Navier-Stokes Equations for Internal Flow', J. of Comp. Physics, **70**, pp. 232–252, (1987).

[11] D. J. Maviriplis, 'Accurate Multi-grid Solution of the Euler Equations on Unstructured Adaptive Meshes', AIAA Journal **28**, pp 213-221 , (1990).

7

Unstructured Mesh Generation

7.1 Introduction

The governing equations of fluid dynamics are formulated on the basis of the continuum hypothesis. In computational engineering, it is impossible to consider all the points in the continuum domain because of the restriction of memory and speed of computers. Thus, only a set of selected points can be chosen for the calculation of flow variables. These sets of points and their connections form the mesh. With the assumption that the information at these points presents the complete description of the domain, this mesh can be used by a numerical method for solving the flow equations. Application of the finite volume method for the solution of the governing equations requires that the flow domain be divided into a set of contiguous, non-overlapping cells. This method facilitates the use of unstructured meshes, which provide flexibility for flow simulation over complex geometries. In unstructured meshes, the cells can be defined in terms of objects, which for a two-dimensional mesh includes nodes, edges and triangular cells, whilst for a three-dimensional mesh includes nodes, edges, faces and tetrahedral cells.

This chapter describes the basic principals behind discretisation procedures of the flow domain and the generation of regular and irregular unstructured meshes around arbitrary configurations. Section 7.2 reviews the two approaches of structured and unstructured mesh generation. Section 7.3 describes the stages of producing regular grids by using methods appropriate for the generation of structured grids. Finally, Section 7.4 summarises the methods applied to generate irregular unstructured triangular and tetrahedral meshes. Section 7.5 gives a brief description of the Delaunay triangulation method for generating triangular and tetrahedral unstructured meshes.

7.2 Mesh Generation Approaches

Recently, considerable effort has been focused on the domain discretisation process, which is called mesh generation. For some problems, the generation of a suitable mesh can be demanding. With the increase of computer power and improvements in flow simulation techniques, there is a greater focus on accurate solutions to real problems. Therefore, this places constraints on the mesh generation. The grid must have the ability to reflect the geometrical boundaries of the domain and resolve the physics of the problem accurately. In addition, the number of points should should not be excessive and their distributions should meet the demand of flow solver (i.e. smoothness and regularity). Any mesh must meet these requirements. Hence, the grid generation procedure can be thought of as a set of iterations cycling through the grid generation and evaluation steps. Thus, the grid generator should be fast, efficient and automatic.

An important classification of grid generation techniques is based upon the structured and unstructured philosophies. For structured grids the set of points map to a regular array, and the cell connectivities are implicitly defined in the ordering of the matrix array of the set of points. Unstructured meshes, in addition to the set of points, require the explicit definition of connectivities between nodes to form cells. Although, a survey of grid generation methods can be found in the comprehensive review given by Weatherill [12], a brief discussion about the advantages and disadvantages of the structured versus unstructured grids is presented.

Structured grids have been widely used because of the ease of use and implementation of boundary conditions. The generation of the rectilinear meshes provides great control over the quality of the grid in terms of the position of the points and the size of the cells but they are applicable for simple configurations. For the complex configurations multi-block structured grids can be applied [1] in which the computational domain is subdivided into a set of blocks. The arrangement of the blocks defines the topology which is consistent with the boundary shape. Such an approach has been used by Shaw, Georgala and Weatherill [25] to predict transonic inviscid flow about an aircraft with pylon and wing configurations, as well as military aircraft with chin intake and propelling nozzles. The problems with multi-block structured grids are the time required to set up the block connectivities, the considerable difficulty of defining the mesh blocks and ensuring the contiguity of mesh lines at the various interfaces [2]. Another approach that has been suggested is the use of separate overlapping meshes. In this approach the flow variables are interpolated between these meshes using an iterative solution procedure. This approach also presents some difficulties when applied to closely coupled components, such as nacelle, pylon and wing regions of modern aircraft. As an alternative, the use of non-align meshes which do not conform to the shape of the configuration has been proposed and for relatively simple flow problems some success has been reported. However, the application of the solid wall boundary conditions leads to complex interpolation formulas and the accuracy obtainable by such an approach is questionable [11].

For complex geometries, the generation of an appropriate grid often causes a major problem [6]. The structure imposed by rows and columns of rectangular grid points can become very restrictive. Therefore, the rectangular grids may not seem to be the best way to discretise a complicated domain. In addition, there remains some doubts about the possibility to automatically generate a structured grid and maintain low cell skewness, a smooth point distribution and adequate grid resolution of important regions [16]. The motivation for overcoming these problems has lead to the use of entirely unstructured meshes comprising triangles in two dimensions and tetrahedra in three dimensions [7,8]. This flexible approach is free of many of the problems that have been associated with other approaches. Using this approach, the grid can be adapted to irregular boundaries and a smooth distribution of triangle and tetrahedral size can be achieved without distortion of the cells. In 1986, Jameson, Baker and Weatherill [11] demonstrated how three-dimensional unstructured tetrahedra meshes may be used to efficiently solve the Euler equations over a complete aircraft. Unstructured grids had been initially employed in finite element codes [11]. In order to use unstructured meshes it is necessary to include the ability to apply additional information about the connectivity of the grid and an indirect addressing system which must be used to locate cell neighbours. This requires additional memory and may seem less efficient than structured grid methods. However, unstructured grids have the ability to discretise complex domains without having to use an excessive number of computational nodes. Therefore, there may be a compromise between efficiency and memory.

The accurate simulation of viscous high Reynolds number flows is seen as a critical problem of the unstructured approach and require a major development effort to overcome the difficulties. Unstructured mesh generation methods do not naturally generate highly stretched elements appropriate for the resolution of boundary layers in viscous flow simulation. Unstructured triangular and tetrahedral meshes have been seldom employed for the solution of Navier-Stokes equations. In 1986, Weatherill, Johnston, Peace and Shaw [26] developed of a finite volume method for the solution of the Reynolds-Averaged Navier-Stokes equations on a triangular grid consisting of small directly triangulated structured regions around the aerofoil and an unstructured mesh elsewhere. Since then, the solution of the Navier-Stokes equations on triangular unstructured meshes has been the subject of many exercises and investigations, particularly in the field of the compressible flow simulation [27,28].

Difficulties associated with solving viscous flows using the triangular meshes for complex geometries has resulted in the hybrid structured and unstructured meshing strategies. In this approach, quadrilateral meshes are employed in the viscous regions, and unstructured meshes are employed in the inviscid regions [29]. Although this approach has proven valuable for computing flows over various types of configurations, it lacks the generality required for arbitrarily complex geometries [27]. consequently, mesh generation developers have begun to focus on an approach which combines some concepts of structured grid technology with the traditional unstructured methods. Hence, an approach was developed, the so-called method of Advancing Layers or Advancing Normals. This approach which was advocated by Weatherill [20], automatically creates a layer of semi-structured (regular) triangular cells with arbitrary aspect ratio in viscous regions and unstructured (irregular) cells in the inviscid dominated flow fields. In this approach, as the layer advances, the cell aspect ratio grows until the elements have unit aspect ratio, whereupon the grown grid is interfaced with a traditional unstructured grid [21,22]. Such an approach, which was readily extended to three-dimensions by Pirzadeh [30], can be easily modified to generate tetrahedra, prisms or hexagon.

The interesting aspect of unstructured triangular and tetrahedral meshes is the ability to accommodate local source points to provide a variable grid resolution within the domain. In addition, cells can be added or removed locally as dictated by the flow or geometrical features, and thus, provides a natural environment for mesh smoothing and grid adaptation [6-9]. However, the required criterion for the grid adaptation can be obtained from the feature of the flow field. Various techniques have been developed for grid adaptation i.e. using error estimation schemes [8] and a streamline based approach [31]. Adaptivity techniques were initially developed for unstructured grids which were traditionally generated using isotropic cells. Since the demand for viscous flow simulations over complex geometries has presented a challenge which could not be accurately resolved by *a priori* adopted structured grids, the adaptivity strategies have been extended to the generation of mixed regular and irregular meshes [21].

In the present work, regular, irregular and mixed regular/irregular unstructured triangular and tetrahedral meshes are employed and the adaptivity procedure is considered. Note that since the present flow solver has been developed using a side-based algorithm, both the regular and irregular meshes are treated in the same manner. The following sections give a brief summary of some approaches for the generation of the triangular and tetrahedral meshes which have been employed in this work.

7.3 Regular Unstructured Meshes

In regular grids, the produced set of grid points can have a high degree of structure which in some cases can be exploited. For instance, when the solution of the viscous flow equations is of interest, because of the rapid variation of the velocity components parallel to a solid body inside the boundary layer, accurate solutions of the flow requires a number of grid points normal to the solid body. In this case, application of irregular unstructured meshes using equilateral cells leads to huge number of nodes close the solid wall surface. However, regular grids have the ability to incorporate a few high aspect ratio cells inside the boundary layer, adjacent to the solid wall surface. Hence, for the computation of viscous flow over simple geometries, structured quadrilateral and hexagonal grid points can be used to create ideal regular unstructured triangular and tetrahedral meshes in two and three dimensions, respectively.

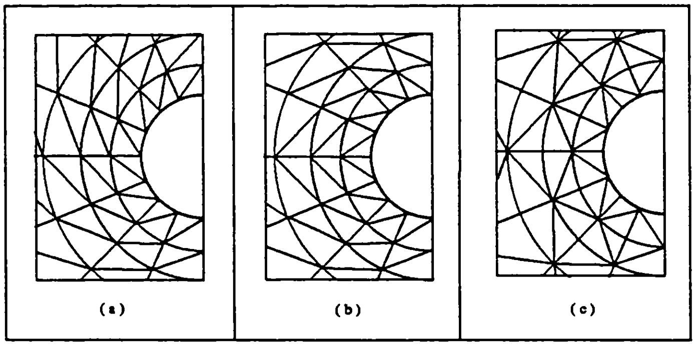

Figure 7.1 *Directly triangulated regular meshes,*
(a) Uni-direction diagonals (inverse symmetric),
(b) Symmetric diagonals to the flow axis
(c) Criss-cross diagonals (non-inverse symmetric).

Two-dimensional regular unstructured triangular meshes can be generated by dividing the quadrilateral grid spacings into the two triangles using diagonal edges (Figure 7.1.a). In viscous flow simulations over a closed configurations, the solution may exploit the symmetric direction of diagonal sides on upper and lower surfaces (Figure 7.1.b). In this case, considerable improvement of the numerical solution may be achieved. However, for both Figures (7.1.a) and (7.1.b), the direction of the diagonal edges directly affects the shape of the control volumes. This may affect the accuracy of the computations. The problem can be overcome using a crisscross form of regular meshes (Figure 7.1.c). The criss-cross meshes can be formed by altering the diagonal edges within the neighbouring quadrilateral grid. However, in these types of regular meshes, the shape of the control volume and the number of edges connected to its centre vary. Hence, even with applying appropriate scaling and normalisation techniques for the edge-based artificial dissipation term, oscillatory effects can appear in the solution [28].

Similar to two-dimensional meshes, three-dimensional regular unstructured tetrahedral meshes can be created using structured hexagonal grid. Within each hexagon, either five or six tetrahedral cells can be accommodated by defining appropriate connectivities. If the alternative of six tetrahedra cells within each hexagonal grid spacing are considered, the criss-cross meshes can be produced by altering the diagonal faces in neighbouring hexahedra. In this work, regular triangular meshes are used for some simple geometries, and hence a brief summary of some of the methods for generation of structured grid points is given.

The structured grids in two and three dimensions can be generated by mapping a single set of computational coordinates (ξ_1 , ξ_2 , ξ_3) to the curvilinear coordinates of the physical space (x_1, x_2, x_3). Various mapping techniques are available for this boundary value problem. These include algebraic schemes [4], partial differential equations [5] and conformal mapping techniques [1].

Algebraic grid generation system can be referred to as interpolation for generating values of cartesian coordinates in the interior of the rectangular transformed region from the specified values at the boundaries. This technique generates grids directly, therefore, it is computationally fast and provides control over grid spacing due to its explicit nature. The algebraic system can be viewed as an interpolation scheme in which curves are fitted between points in opposite boundaries or intermediate surfaces in the field. Evaluation of an interpolation function at constant values of the curvilinear coordinates defines the coordinate system. Uni-directional and multi-directional interpolation schemes can be applied for two and three-dimensional domains in which there are four and six sides of boundary data, respectively.

In the methods based on the solution of partial differential equations, the solution procedure and the properties of the generated grid are closely related to the properties of the applied partial differential equation, i.e. elliptic, parabolic or hyperbolic. Hyperbolic methods are used for problems in which the exact position of outer boundaries is not critical. Nowadays, the elliptic equation technique appears to be the most predominantly used in the field of structured grid generation [4]. This is due to their natural smoothing properties which prevents discontinuities. This property makes them ideally suited for grid generators. Although the generated grids are not necessarily orthogonal, they will be smoothly varying even if the boundary of the domain has a slope discontinuity. Following this approach, the curvilinear coordinates are generated by solving the non-homogeneous Laplace equations (Poisson equations). The solution of equations can be obtained by the use of central differencing and using control functions for controlling the grid point spacing. This system of equations can be numerically solved by application of finite difference approximations in both two and three dimensions [3].

An alternative approach to structured grid generation is conformal mapping which transform a geometrically complicated domain into a simple domain within which a mesh can be generated. Then, this mesh is converted to physical domain as a computational mesh. The concept of a mapping in two dimensions is to define a transformation, which takes a physical domain defined in the plane of (x_1, x_2), onto the rectangular domain on the plane (ξ_1 , ξ_2). A curvilinear coordinate system generated by conformal mapping is very rigid in the sense that little control can be exerted over the distribution of the grid points. The coordinate system tends to be difficult to construct compared with algebraic or elliptic systems. However, the complex variable techniques by which conformal transformations are usually generated, are inherently two-dimensional. In spite of some progress in using mapping techniques for generation of structured grid points, these methods are not easily applicable to three dimensions for all geometries. Surveys of the various techniques are given in [19]. However, for simple geometries three dimensional structured grid points can be formed by stacking or rotating two dimensional grid points.

7.4 Unstructured Meshes

There are different methods for creating genuinely unstructured meshes. The two major methods are the Advancing Front and Delaunay triangulation which have been developed over the last 10 years [10,11]. In three dimensions, the Advancing Front method begins with a set of triangles, known as the front, which encloses a domain. Elements are created on each triangle of the front by either creating a point or using an existing point. In this way, cells are created in the domain and the front advances until the domain is filled with tetrahedra. More recently, in line with other unstructured generation grid methods, grid point spacing is controlled by sources placed in the domain. Effort has seen the development of fast search algorithms [8,10,24]. The Delaunay method is different in the sense that, given an enclosing surface triangulation, the Delaunay triangulation approach connects all boundary points to form a coarse triangulation prior to the insertion of points using some form of refinement strategy. The recovery of the given set of boundary triangles within the tetrahedral construction has been a major issue in Delaunay triangulation approach. However, major advances have been made in this approach and today point insertion strategies, combined with a Delaunay triangulation algorithm, can efficiently generate very large meshes for complex geometries [15,23]. Although, appropriate data structures have been applied to perform searches for the Advancing Front, these searches are not as fast as those which can be performed within the Delaunay tetrahedral structure. Therefore, a major development in recent years has been the speed advantage of Delaunay triangulation method as compared with Advancing Front methods [12,15].

Due to the major development in speed of the Delaunay triangulation, the codes based on this algorithm were found very efficient for iterative procedure of mesh generation and evaluation, particularly for three-dimensional problems. In order to achieve better results of geometry definition and surface grid generation, also to speed up the time consuming grid generation and evaluation iterations, a single environment which contains all the tools required for pre-processing computational simulation has been utilised. This open environment is called the Parallel Simulation User Environment (PSUE) which has the potential to provide an all encompassing environment for unstructured mesh generation applied in computational engineering. The PSUE framework integrates modules such as ; Geometry Builder/Viewer, CAD Input, Geometry Repair, Grid Generation and Adaptation and Analysis of data [12]. Hence, the quality of both regular and irregular meshes can be evaluated using the PSUE package. In this work, this package is widely used, especially for the generation and evaluation of three-dimensional inviscid unstructured meshes. Here, a brief description of Delaunay triangulation method, which is employed for generating the irregular unstructured meshes, is presented.

7.5 Delaunay Triangulation

The Delaunay triangulation is one of the most effective methods of creating irregular unstructured triangular meshes, which has recently been the subject of many studies. Weatherill [16] and Jameson, Baker and Weatherill [11] implemented the Delaunay triangulation technique for aerodynamics applications of single and multi-connected domains using a modified Bowyer triangulation algorithm [17]. Here, a very brief description of the two-dimensional Delaunay triangulation algorithm is presented.

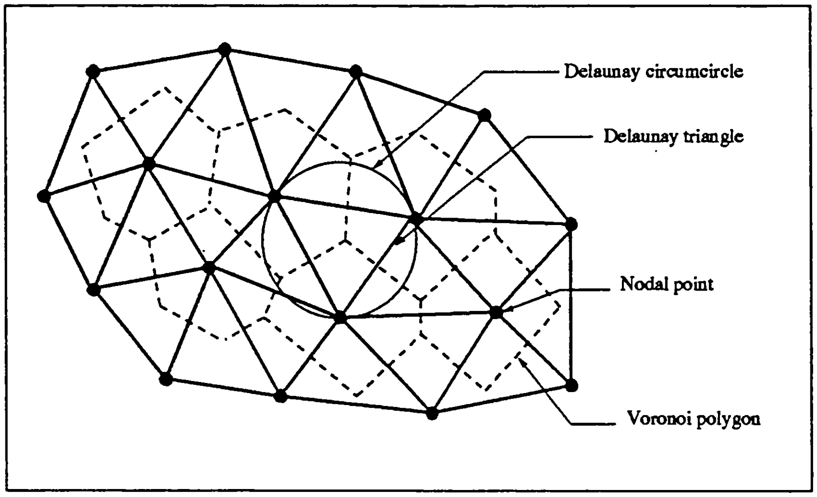

Figure 7.2 *Voronoi diagram and Delaunay triangulation*

Given a set of points in the plane, the Dirichlet tessellation [18] can be constructed which assigns to each points a territory that is the area of the plane closer to the certain point than any other point in the set. This tessellation of a closed domain results in a set of non-overlapping convex polygons, called Voronoi regions [19], which covers the entire domain, Figure (7.2). The territorial boundary, which forms a side of a Voronoi polygon, must be midway between the two points. This side is a segment of the perpendicular bisector of the line joining these two points. The definition of the Voronoi regions ensures that the triangulation produces triangles of reasonable aspect ratio. The Delaunay triangulation is formed by joining the points which have some segments of boundary in common. Note that, each Delaunay triangle has a unique vertex of the Voronoi diagram and no other vertex within the Voronoi structure lies within the circle centred at this vertex. Figure (7.2) shows a two-dimensional schematic of a Voronoi diagram and associated Delaunay triangulation of a small set of points.

In practice, since the initial set of points is not readily defined, the Delaunay algorithm can be done through a sequential process of introducing new points into the existing structure. Then, this broken structure is reconnected to form a new Delaunay triangulation. This approach facilitates the introduction of local source points with prescribed resolution and domain of influence. These sources impose local clustering of grid points and provide variable resolution of the mesh. Thus, the process starts by setting up an initial triangulation, which can be adjusted in such a way that all points to be triangulated are contained within one or two super-triangles. The next step is to ignore the presence of the boundary and thus triangulate the entire domain treating the boundary points in the same way as internal points. Then, all triangles which happen to be located outside of the boundaries are identified and subsequently removed. This simple process can be achieved providing that all adjacent boundary points are found to be contiguous in the Delaunay algorithm [13,14,15].

Today, point insertion strategies, combined with a Delaunay algorithm enable the method to generate very large meshes for complex geometries. The attractive aspect of this type of mesh generation is the fact that the concept of Delaunay triangulation can easily be extended to three dimensions. The major differences are that the notations of Delaunay triangles is replaced by

Delaunay tetrahedra in three dimensions [15]. Hence, the territory of each data point is a convex polyhedron and each vertex of the Voronoi structure is at the circumcentre of a sphere defined by four points, which describe the Delaunay tetrahedron. A brief review of the Delaunay triangulation algorithm for three-dimensional mesh generation is given in Appendix D.

7.6 Summary

Several methods of unstructured mesh generation techniques are reviewed in this chapter. For simple geometries, the regular unstructured triangular meshes can be produced using structured grid methods. For complex configurations, an alternative grid generation procedure is to use irregular unstructured grids. The approach of Delaunay triangulation is an efficient method for generating unstructured triangular and tetrahedral meshes. In this technique, local sources can be introduced to define variable mesh resolution. The advantage of the resulting mesh is that cells are forced to be equilateral which make the mesh more suitable for accurately solving the inviscid flow equations.

In the present work, since the regular unstructured meshes provide more control over the size of the cells, they are used at early stages of computation for simple problems. Note that, in spite of their name, the number of edges or cells connected to each node can be irregular, especially in three dimensions. Hence, they have been used to verify the ability of flow solver to deal with such a problem. This fact has considerable effect on the computation of edge-based artificial dissipation term. However, in order to demonstrate the performances of the flow simulator and have a comparison, most of the inviscid test cases are examined on the irregular unstructured meshes as well.

In boundary layer and wake regions of the viscous flows, the regular grids are the most appropriate meshes for resolving high gradient of the flow variables. Therefore, the accuracy of the viscous flow solver will be examined by developing individual regular unstructured mesh generators for individual problems, which are studied in the following chapters.

References

[1] J. F. Thompson, Z. U. A. Warzi, and W. Mastin, 'Numerical Grid Generation ; Foundations and Applications' Special Lecture, Advances in Finite Element Technology, North-Holland, (1985).

[2] N. P. Weatherill, C. Forsey, 'Grid Generation and Flow Calculations for Aircraft Geometries' , J. Aircraft , Vol. 22, No. **10**, pp 855-860, (1985).

[3] W. J. Minkowikz, E. M. Sparrow, G. E. Schneider and R. H. Pletcher, 'Handbook of Numerical Heat Transfer', John Willy and Sons, (1988).

[4] L. E. Erikson, 'Generation of Boundary Conforming Grids around Wing Body Configuration Using Transfinite Interpolation', AIAA Journal, Vol 20, pp. 1313-1320, No. 10, (1982).

[5] J. F. Thompson, 'Grid Generation Techniques in Computational Fluid Dynamics', AIAA Journal, Vol 22, 1505–1523, (1984).

[6] N. P. Weatherill, M. J. Marchant, O. Hassan, D. L. Marcum 'Grid Adaptation Using a Distribution of Sources Applied to Inviscid Compressible Flow Simulations', Int. J. for Num. Meth. in Fluids, **19**, pp. 739–794, (1994).

[7] M. J. Marchant, N. P. Weatherill, 'A Streamline Based Approach to the Adaptation of Unstructured Grids for Viscous Flow Simulation', ICFD Conference, Oxford U.K. April 1995.

[8] J. M. Peraire, M. Vahdati, K. Morgan and O. C. Zienkiewicz, 'Adaptive Remeshing for Compressible Flow Computations', J. of Comp. Physics, **72**, pp. 449–446, (1987).

[9] D. J. Mavriplis, 'Accurate Multi-grid Solution of The Euler Equations on Unstructured and Adaptive Meshes', AIAA Journal, Vol 28, No. 2, pp. 213–221, (1990).

[10] R. Lohner, K. Morgan, J. M. Peraire and O. C. Zienkiewicz, 'Finite Elements for High Speed Flows', AIAA paper 85-1531-CP, 17th AIAA CFD Conference, (1985).

[11] A. Jameson, T. J. Baker and N. P. Weatherill, 'Calculation of In-viscid Transonic Flow over a Complete Aircraft', AIAA Paper 86-0103, (1986).

[12] N. P. Weatherill, 'A Review of Mesh Generation' Special Lecture, Advances in Finite Element Technology, Civil-Comp Press. Edinburgh. pp 1-10, (1996).

[13] S. W. Sloan and G. T. Houlsby, 'An Implementation of Watson's Algorithm for Computing Two-dimensional Delaunay Triangulation', Advanced Engineering Software, **6**, No. 4, (1984).

[14] N. P. Weatherill, 'Grid Generation', VKI Lecture Series 1990–10, (1990).

[15] N. P. Weatherill and O. Hassan, 'Efficient Three-dimensional Delaunay Triangulation with Automatic Point Creation and Imposed Boundary Constraints' Int. J. for Num. Methods in Eng., **37**, pp. 2005–2039, (1994). (1986).

[16] N. P. Weatherill, 'A Method for Generating Irregular Computational Grid in Multiply Connected Planar Domain', Int. J. for Num. Methods in Fluids, **8**, pp. 181–197, (1988).

[17] A. Bowyer, 'Computing Dirichlet Tessellation', The Computer Journal, **24**, pp. 162–166, (1981).

[18] G. L. Dirichlet, ' Uber Die Reduction der Positiven Quadratischen Formen mit drei Unbestimmten Ganzen Zahlen', Z. Reine Angew. Math, **40** (3), pp. 209–227 (1850).

[19] P. G. Green and R. Sibson, 'Computing Dirichlet Tessellation in the Plane', Comput. J., **21** (2), 168–173 (1978).

[20] N. P. Weatherill, 'Mixed Structured and Unstructured Grids for Aerodynamic Flow Simulation' , The Aeronautical J., 111–123, Vol 94, (1990).

[21] O. Hassan, E. J. Probert, N. P. Weatherill, M. J. Marchant, k. Morgan, and D. L. Marcum, ' The Numerical Solution of Viscous Transonic flow Using Unstructured Grids', AIAA paper 94-2346, (1994).

[22] N. P. Weatherill, 'Mesh Generation for Aerospace Computations - the Last 10 Years and Prospects for the Future' Proceedings of the Basel World CFD User Days 1996, Freiburg, Germany, PP. 19–23, (1996).

[23] D. L. Marcum and N. P. Weatherill, ' Unstructured Grid Generation

Using Iterative Point Insertion and Local Reconstruction', AIAA Journal Vol 33, **9**, pp. 1619–1625, (1995).

[24] J. M. Peraire, J. Peiro, L. Formaggia, K. Morgan and O. C. Zienkiewicz, 'Finite Element Euler Computations in Three-dimensions', Int. Num. Meth. in Eng. , Vol 26, **72**, pp. 2135–2159, (1987).

[25] J. A. Shaw, J. M. Georgala and N. P. Weatherill, 'The Construction of Component Adaptive Grids for Aerodynamic Geometries' In Numerical Grid Generation in Computational Fluid Dynamics 1988, PP. 383–894, Pineridge Press, (1988).

[26] N. P. Weatherill, L. J. Johnston, A. J. Peace and J. A. Shaw, 'Development of a Method for the Solution of the Reynolds-Averaged Navier-Stokes Equations on Triangular Grids', ARA Report 70, (1986).

[27] D. J. Maviriplis, A. Jameson, L. Martinelli, 'Multi-grid Solution of the Navier-Stokes Equations on Triangular Meshes', AIAA-89-0120 , (1989).

[28] L. A. Sykes, 'Development of A Two-dimensional Navier-Stokes Algorithm for Unstructured Triangular Grids', ARA Report 80, April 1990.

[29] J. A. Shaw, J. M. Georgala and P. N. Child sand, 'Automatic Generation of Block Structures - Progress and Challenges' Numerical Grid Generation in Computational Field Simulation and Related Fields, B. Soni, J. F. Thompson, J. Hauser and P. R. Eiseman (Eds), PP. 403, Proceeding of the 5th Int. Grid Gen. Conf., NSF Research Centre, Mississippi State Uni. (1996).

[30] S. Pirzadeh, 'Viscous Unstructured Three-dimensional Grids by the Advancing Layers Method', AIAA Paper, 94-0417, (1994)

[31] M. J. Marchant and N. P. Weatherill, ' A Streamline Based Approach Grids for Viscous Flow Simulation', ICFD Conference, Oxford, UK, April (1995).

8

Simulation of Two-Dimensional Flows

8.1 Introduction

The results and comments on the implementation of the algorithm for simulating two-dimensional incompressible inviscid and viscous flows using the artificial compressibility approach on cell-vertex finite volume method are studied in this chapter. The results will be presented in three sections. Section 8.2 presents the results of inviscid flow computations and verifies the accurate implementation of the convective term, artificial dissipation term and proper boundary conditions. Section 8.3 presents the viscous results. After demonstrating the validation of the algorithm by using geometrically simple test cases, Section 8.4 presents applications to simulate some realistic engineering problems. For each case, the results presented show the performance and accuracy of the solver. The performance of the flow solver is shown using the convergence behaviour of the computations in terms of the residual of pressure and velocity components. The accuracy of the method is assessed by comparing some of the computed results with those obtained from available exact solutions or other experimental and independent results reported in the literature.

Typical pressure and velocity fields are shown using contours of variables or stream lines. For all the test cases of this chapter, the computations started from the free-stream values. Far field boundary conditions were implemented by imposing free-stream values of velocity components at inflow and pressure at outflow. Other values at far field boundaries were extrapolated from inside the domain.

8.2 Inviscid Flow Simulation

The first test case is the flow around a circular cylinder. This test case, for which an analytical solution is available, is a challenging case, and hence has been used by many researchers to examine the accuracy of flow solvers. Therefore, it is employed to verify the accuracy of the discretisation of the convective and artificial dissipation terms in the present flow solver. The second test case presents results of an internal flow computation in a channel with a 10% arc bump. The case can be used for examining the implementation of inflow and outflow boundaries which intersect solid wall boundaries. The computed results of this test case are compared with independent results from other incompressible flow solvers. The third test case is the simulation of the flow over a NACA0012 airfoil. The ability of the flow solver to deal with sharp edges at different angles of incidences are studied using this test case. The results are compared with other independent results from the aerodynamics literature.

For all the inviscid cases, the distribution of pressure on the solid wall is measured using pressure coefficient C_p as

$$C_p = \frac{2.0(p_i - p_0)}{\rho_0 |\vec{\mathrm{U}}_0|^2} \tag{8.1}$$

where p_i is the pressure at wall boundary node i and p_0, ρ_0 and $\vec{\mathrm{U}}_0$ are the free stream pressure, specific mass and velocity, respectively.

8.2.1 Flow Around a Circular Cylinder

The test case involves the inviscid flow computation around a circular cylinder. For this test case, numerical and analytical results are available [1,8]. The exact incompressible solution for this example has been developed using a complex potential formulation [1]. This test case is used to verify the accuracy of the discretisation of the convective and artificial dissipation terms. For this case, the most difficult part of the flow simulation is to compute an accurate pressure distribution on the cylinder surface, especially at its down stream side.

Figure (8.2.1.a) shows the view of the unstructured mesh used in this computation, which was created by Delaunay triangulation. The outer boundary was placed at a distance of 10 radii away from the centre of cylinder. The grid is composed of 2529 points, 4662 cells, and 7192 edges. There are 360 grid points distributed on the cylinder surface, which can be seen in Figure (8.2.1.b).

The free-stream values were considered as $(u_1)_i^0 = 1.0, (u_2)_i^0 = 0.0$ and $(p)_i^0 = 1.0\,(i = 1, N_{node})$. In this test case, without imposing zero wall dissipation, slight dissipative effects appeared in the flow field near the cylinder surface. Enforcing zero wall dissipation [23] reduces this dissipative effect and produces more accurate results. However, lack of dissipation on solid wall allows some oscillations of the values of the flow parameters propogates on the solid wall nodes. This effect prevents achieving maximum convergence of the residuals of the flow parameters on the SUN work-station. However, there is approximately 4.5 orders of magnitude reduction in pressure residuals after 2000 iterations. The convergence history of the pressure and velocity residuals are shown in Figure (8.2.1.c). The computed pressure coefficient on the cylinder surface and the analytical solution are compared in Figure (8.2.1.d). As can be seen, the computed pressure coefficient is in close agreement with the analytical solution.

The computed velocity vectors are shown in Figure (8.2.1.e). In order to give a general sense about the computed flow field, pressure and velocity contour plots are shown in Figures (8.2.1.f) and (8.2.1.g), respectively. The results presented provide some confidence in the capability of the algorithm to accurately simulate two-dimensional inviscid flows.

Inviscid Flow Around a Circular Cylinder

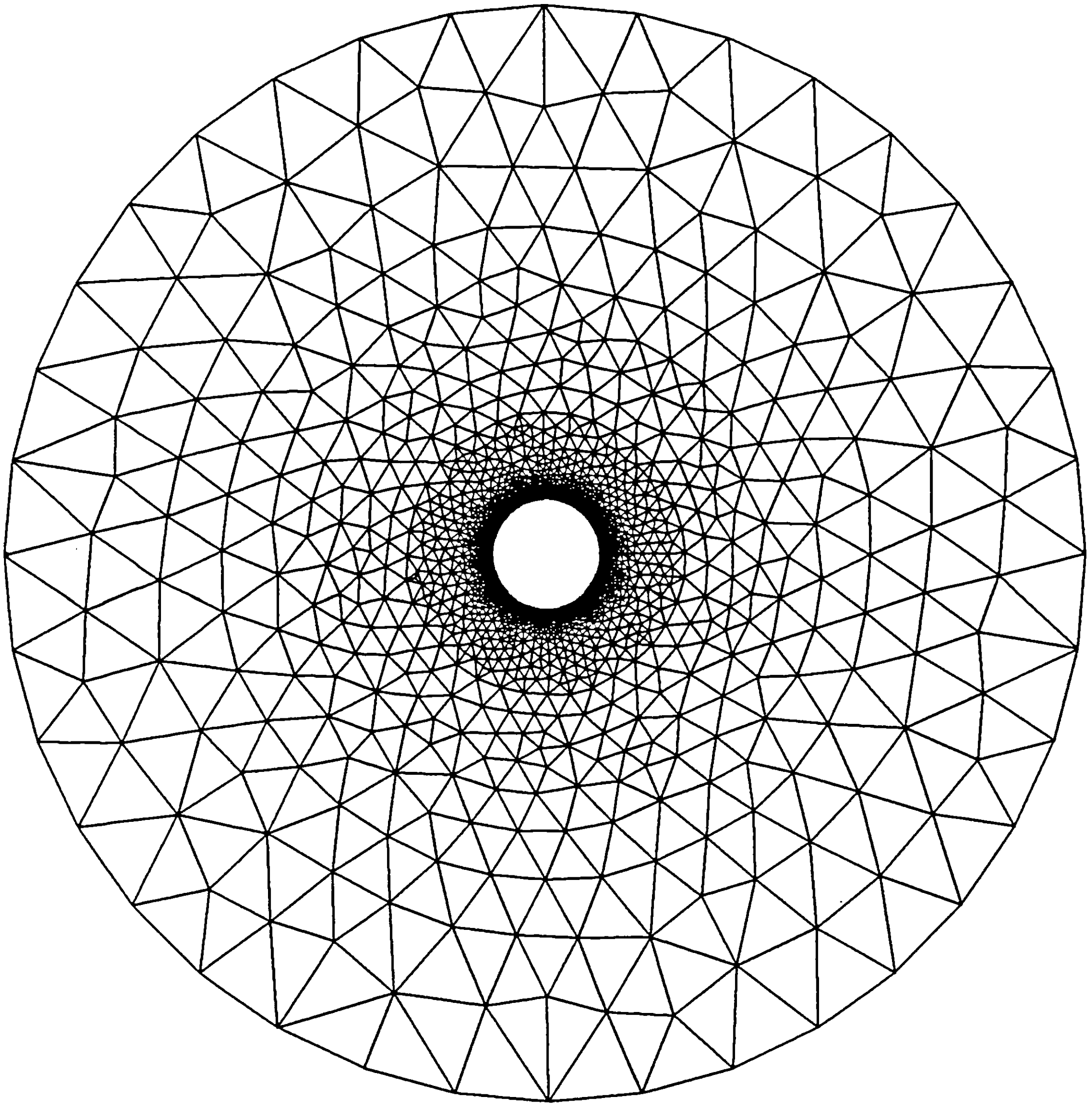

Figure 8.2.1.a *Triangular mesh for the flow around a circular cylinder generated by the Delaunay triangulation method*

Inviscid Flow Around a Circular Cylinder

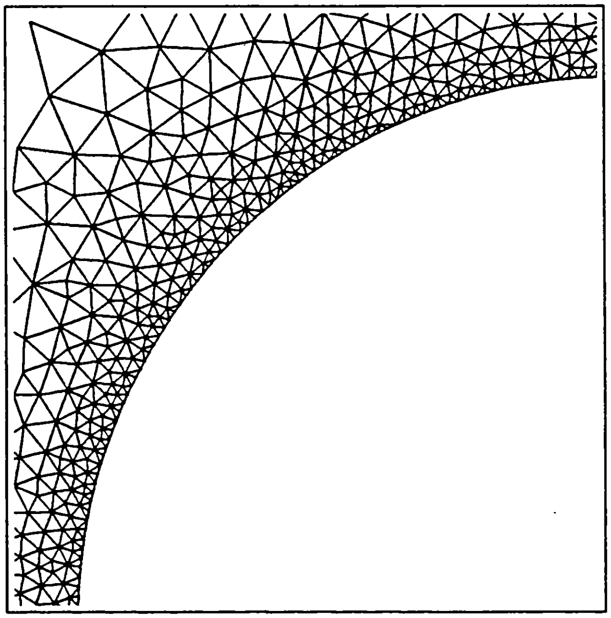

Figure 8.2.1.b *Partial view of the mesh around the cylinder.*

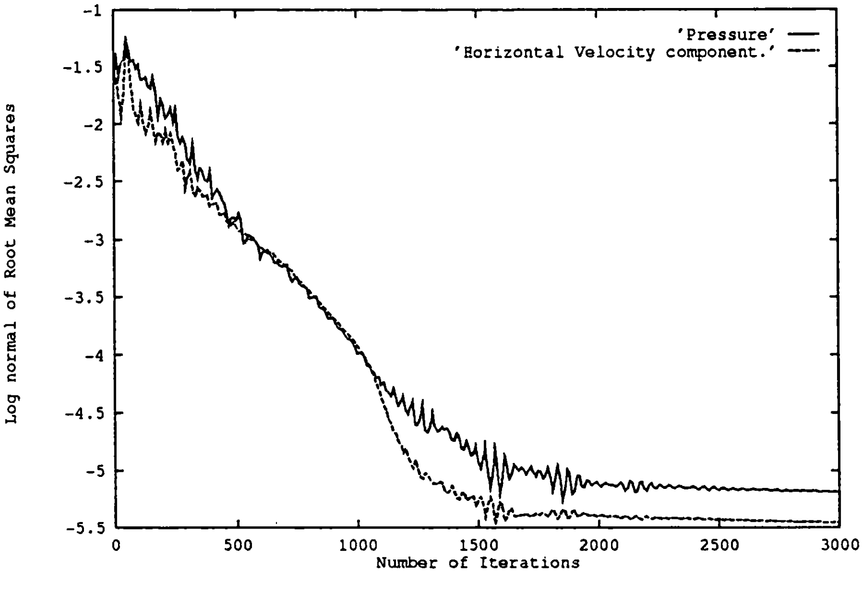

Figure 8.2.1.c *Convergence of pressure and velocity residuals.*

Inviscid Flow Around a Circular Cylinder

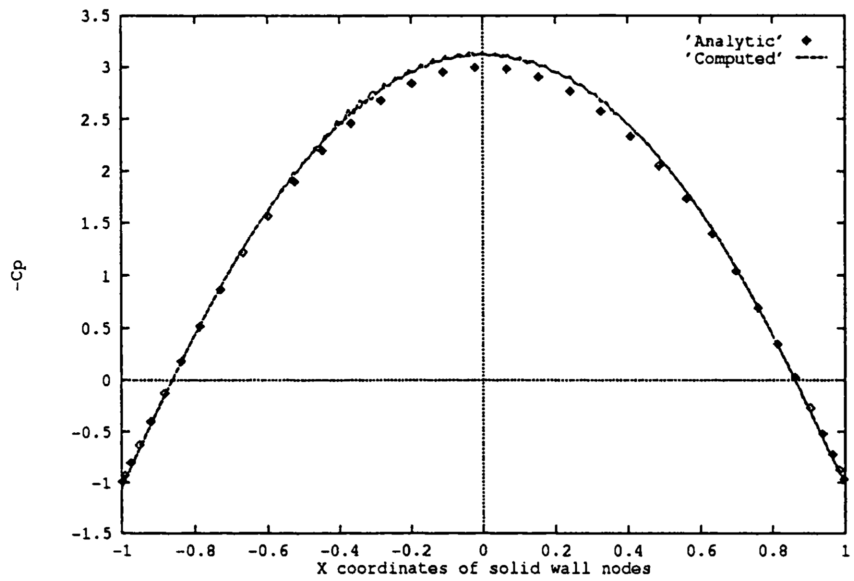

***Figure 8.2.1.d** Coefficient of pressure on the surface of cylinder (comparison with the analytical solution [1]).*

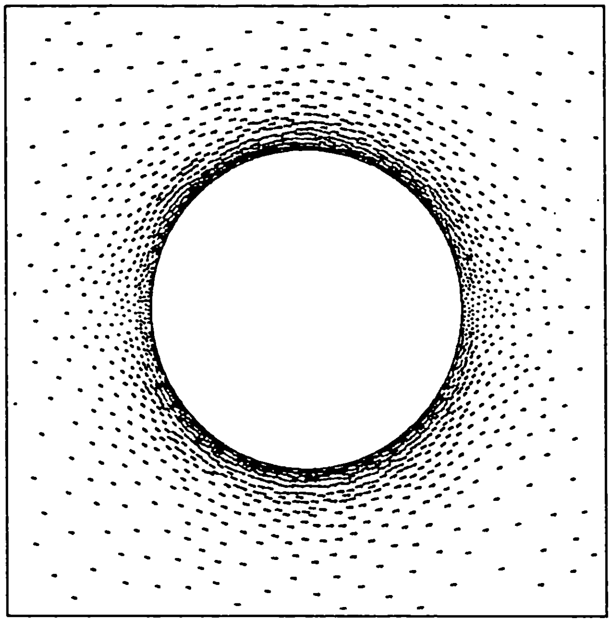

***Figure 8.2.1.e** Velocity vectors around the circular cylinder.*

Inviscid Flow Around a Cylinder

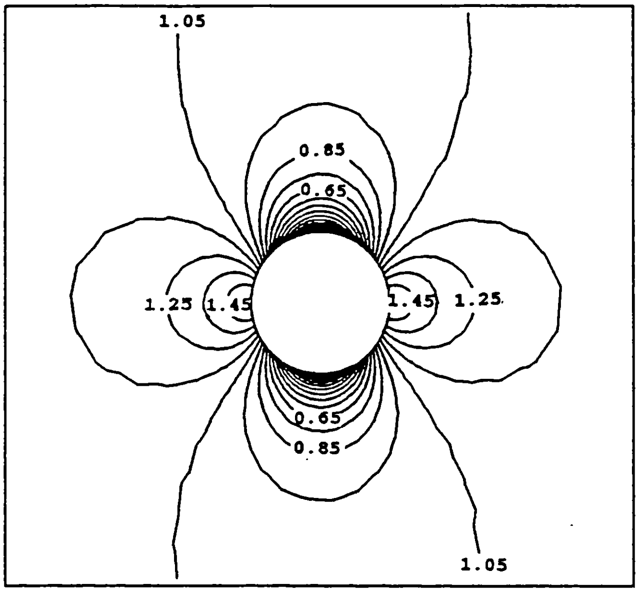

Figure 8.2.1.f *Typical pressure contours around the cylinder (Min. = 0.55, Max. = 1.525, Intervals =0.1).*

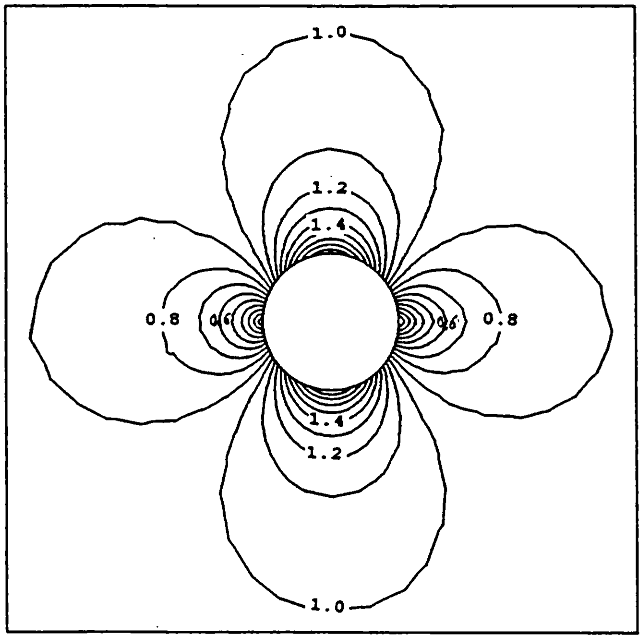

Figure 8.2.1.g *Typical velocity contours around the cylinder (Min. = 0.0, Max. = 2.0 , Intervals =0.1).*

8.2.2 Flow Through a Channel with a Circular Arc

The problem considered here is the internal flow through a channel with a bump consisting of a 10% arc. The test is used by researchers in both fields of compressible and incompressible flow. The intersection of the solid wall with inflow and outflow boundaries and the short distance of these two boundaries from the bump at the centre of the channel are two main points which make the case sensitive to a correct implementation of the inflow/outflow boundary conditions. Hence, this case which was purposed by Ni [2], can be used for verifying the accurate implementation of inflow and outflow boundary conditions. Later Choi-Merkle [3] performed incompressible flow simulation. He applied the artificial compressibility method [4], with constant β^2 using the finite difference scheme of Briley-McDonald [5]. The velocity and pressure contours resulting from this work are used for comparison.

The geometry of this channel is defined by considering the length of the channel to be three times its width and the length of the bump equal to the channel width. The computational grid employed is shown in Figure (8.2.2.a). The grid is composed of (65 × 17) points (or 1100 points in total), 2052 cells and 3151 edges. For the computation of this test case, the free-stream values were considered as $(u_1)_i^0 = 1.0$, $(u_2)_i^0 = 0.0$ and $(p)_i^0 = 1.0 \;\; (i = 1, N_{node})$.

Figure (8.2.2.d) shows the typical computed velocity vectors. The computed pressure and velocity contours are depicted in Figures (8.2.2.b) and (8.2.2.e), respectively. To assess the accuracy of the present computations, the pressure and velocity contours obtained by the incompressible flow solver of Choi-Merkle [3] are shown in Figures (8.2.2.c) and (8.2.2.f). In general, there is a close agreement between the two results.

Figure (8.2.2.g) shows the distribution of the coefficient of pressure on the solid wall. The computation was performed by starting the calculation from initial values on the inflow and outflow boundaries. After 1400 explicit iterations on a SUN work-station which uses single precision by default, approximately 6 orders of magnitude in the reduction of the pressure and velocity residuals was reached (Figure 8.2.2.h).

Inviscid Flow Through a Channel with 10% Bump

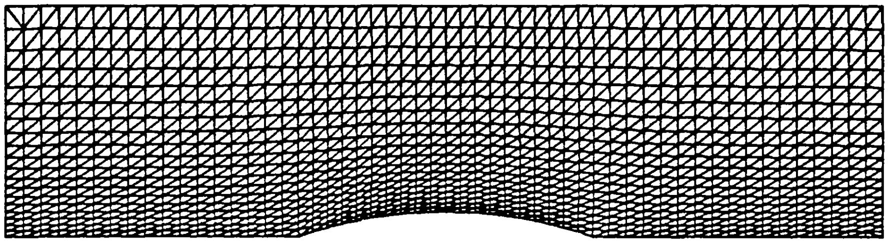

Figure 8.2.2.a *Triangular mesh for the channel with bump.*

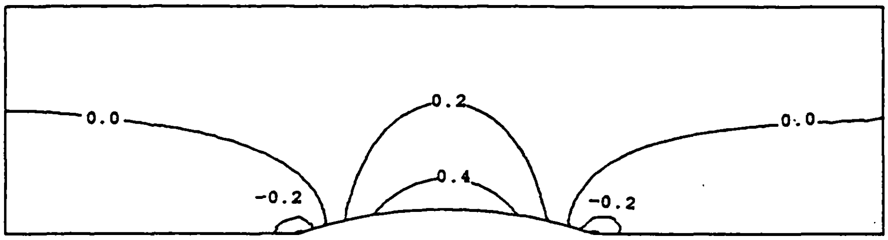

Figure 8.2.2.b *Typical pressure contours inside the channel (Min. = -0.4, Max. = 0.4, Intervals = 0.2).*

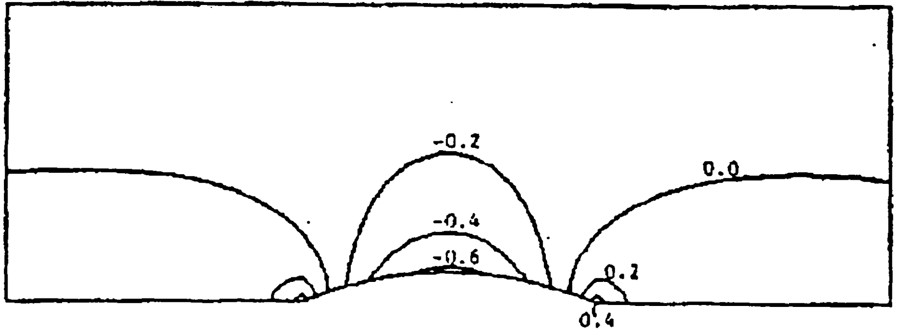

Figure 8.2.2.c *Pressure contours after Choi-Merkle [3].*

Inviscid Flow Through a Channel with 10% Bump

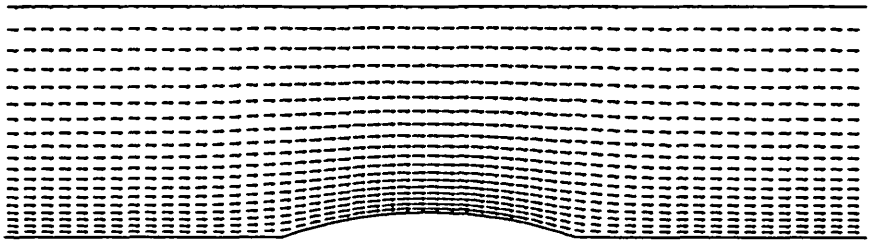

Figure 8.2.2.d *Typical velocity vectors inside the channel.*

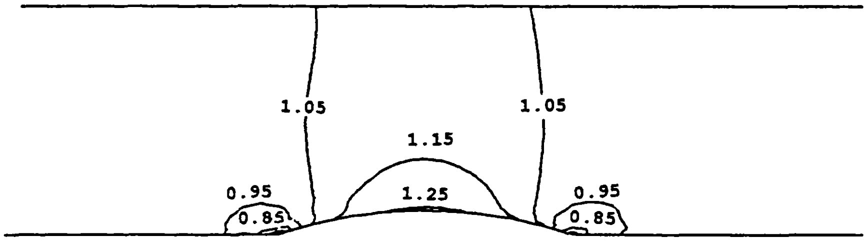

Figure 8.2.2.e *Typical velocity contours inside the channel (Min. = 0.75, Max. = 1.25, Intervals = 0.1).*

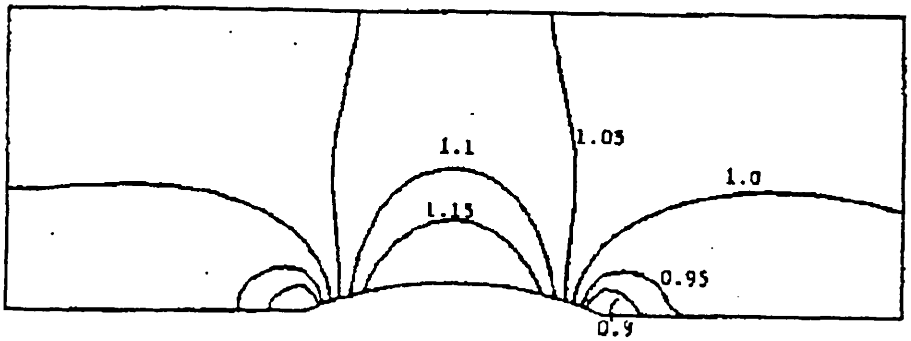

Figure 8.2.2.f *Velocity contours after Choi-Merkle [3].*

Inviscid Flow Through a Channel with 10% Bump

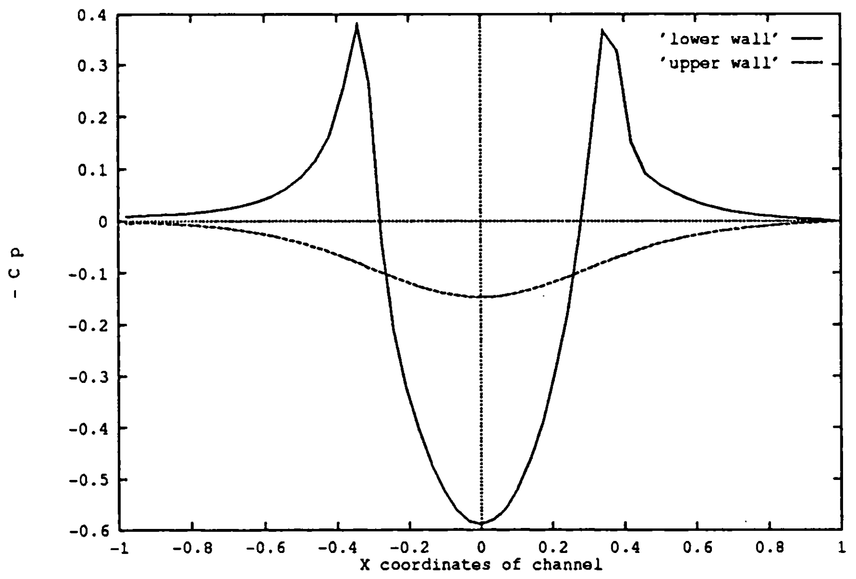

Figure 8.2.2.g *Coefficient of pressure on the channel walls.*

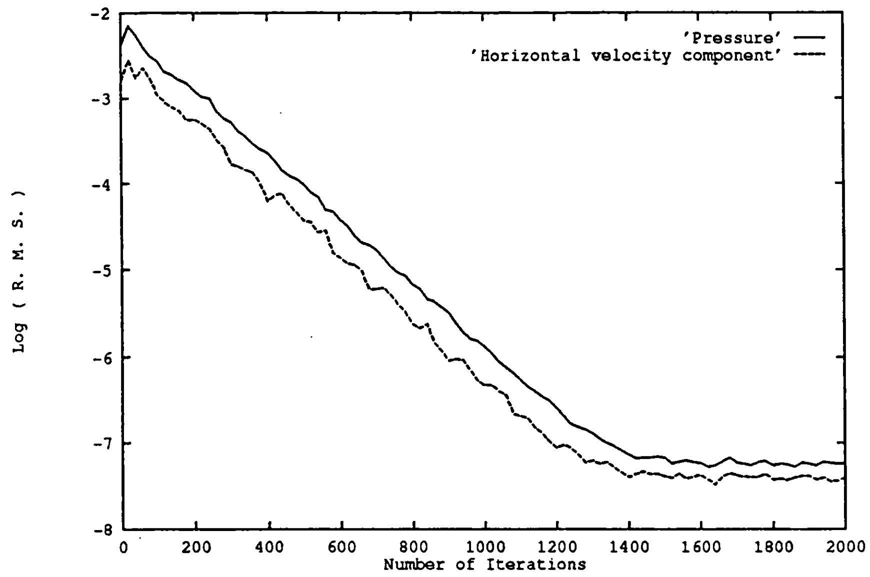

Figure 8.2.2.h *Convergence of Pressure and Velocity residuals.*

8.2.3 Flow Over a NACA0012 Aerofoil

The problem considered is the simulation of the flow around a NACA0012 aerofoil which is a well known aerodynamics test case. This symmetric aerofoil is widely used for flow calculations in many CFD workshops. There are several results which can be used for comparison [6-9]. Some of the available results can be used for validation of the developed flow solver to deal with sharp trailing edges. The pressure and velocity fields close to the trailing edge of the aerofoil can be used for verifying the accuracy of the computation of the flow field close to the sharp end of the object at different angles of incidences. In this work the calculations were performed for 0.0, 2.0, and 5.0 degree angles of incidence.

The view of the irregular grid used for this calculation is shown in Figures (8.2.3.a) and (8.2.3.b). The outer boundary is at distance of 10 chords from the aerofoil. The grid employed was fully unstructured, which was created by the Delaunay triangulation. The computational domain consists of 3406 triangles. The number of the points is 1805 with 168 points located on the aerofoil and the total number of the edges is 5211.

The computation of this test case started from the free-stream values as $(u_1)_i^0 = Cos(\alpha)$, $(u_2)_i^0 = Sin(\alpha)$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$. Here α is the angle with the horizontal cartesian axe x_1. The computation was performed starting from free stream values. For all the simulations of inviscid flow over the NACA0012 aerofoil the computation took less than 2000 iterations to converge. A typical converged history of computed pressure and velocity residuals are plotted in Figure (8.2.3.c). Generally, there is approximately 6 orders of magnitude reduction in pressure and velocity residuals, so the solution can be considered to be adequately converged on the SUN work-station which uses single precision by default. For $\alpha = 0.0^o$ and $\alpha = 2.0^o$, the computed surface pressure coefficient C_p on the aerofoil is compared with the result of the simulation using potential flow solution [7] in Figures (8.2.3.d) and (8.2.3.e). In Figure (8.2.3.f), the computed coefficient of pressure on the aerofoil for the angle of incidence $\alpha = 5.0^o$ is compared with the results of the Rizzi et al [8] computed using a finite volume method and Eriksson [9] computed using a panel method. Typical computed pressure and velocity contours around the aerofoil are presented in Figures (8.2.3.g) and (8.2.3.h). In general, the results of the present flow solver look very similar to those results obtained by other workers.

Inviscid Flow Over a NACA0012 Aerofoil

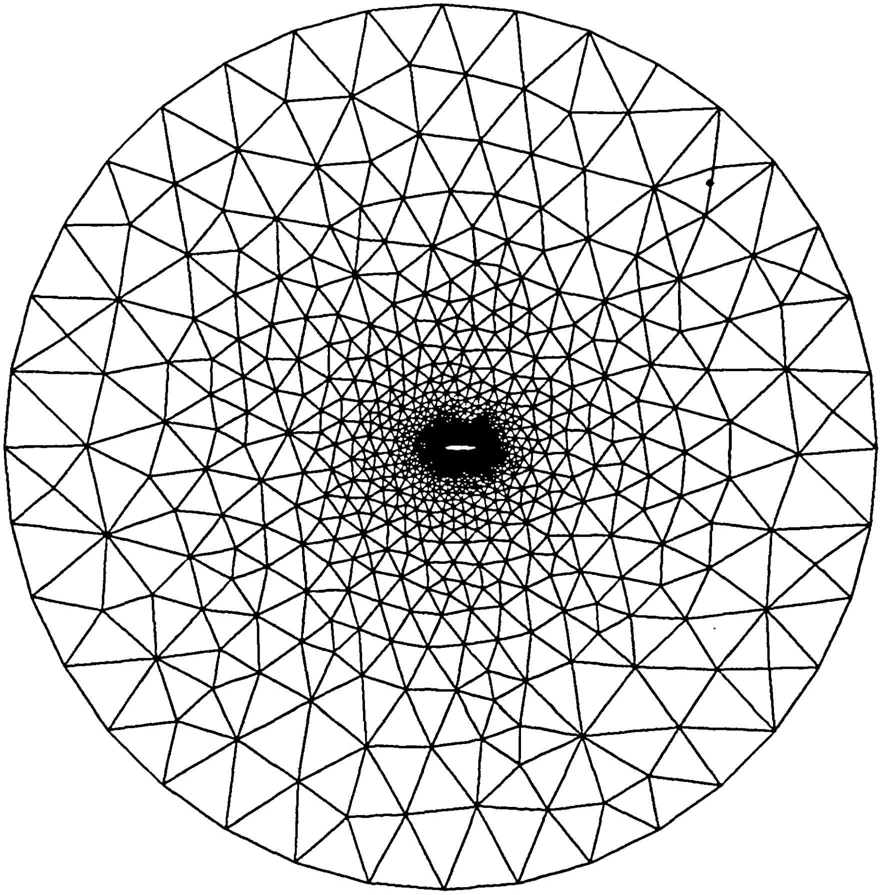

Figure 8.2.3.a *Triangular mesh for flow around the NACA0012 profile generated by the Delaunay triangulation method.*

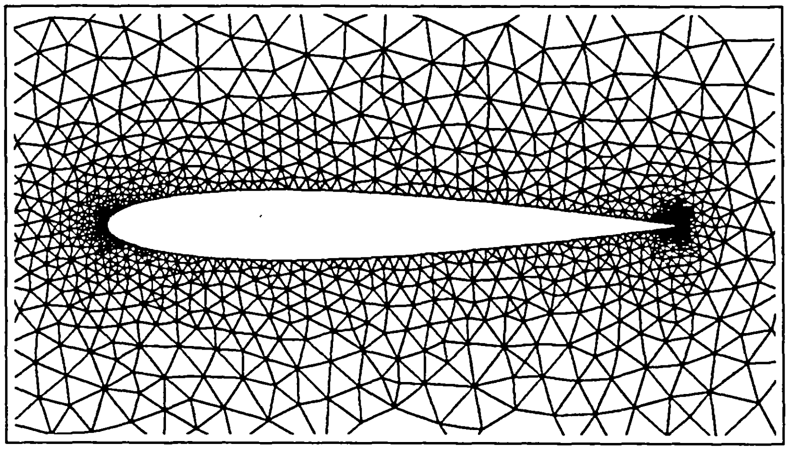

Figure 8.2.3.b *Partial view of the mesh around the aerofoil.*

Inviscid Flow Over a NACA0012 Aerofoil

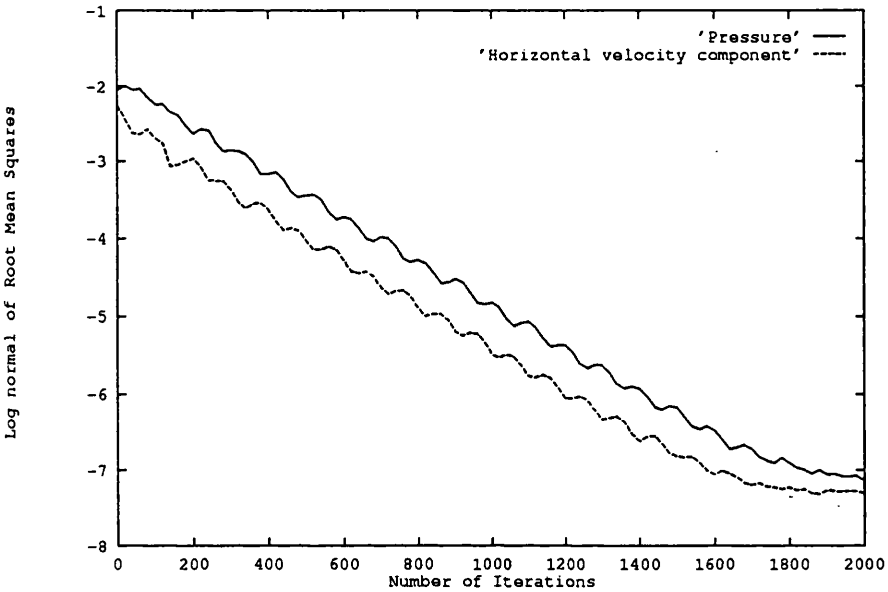

Figure 8.2.3.c *Convergence of pressure and velocity residuals.*

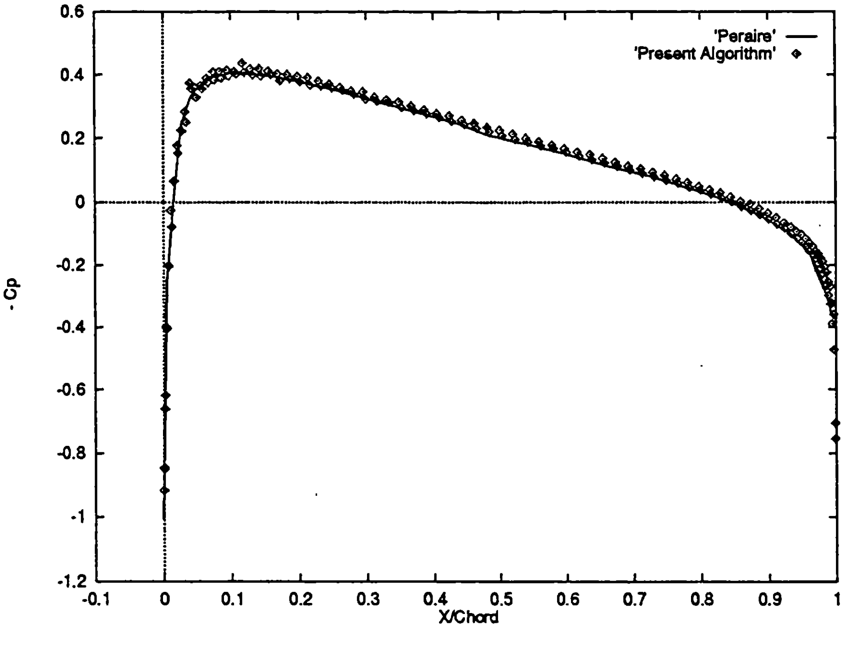

Figure 8.2.3.d *Coefficient of pressure on the aerofoil body for $\alpha = 0.0°$, comparison with the results from potential flow solution by Peraire [7].*

Inviscid Flow Over a NACA0012 Aerofoil

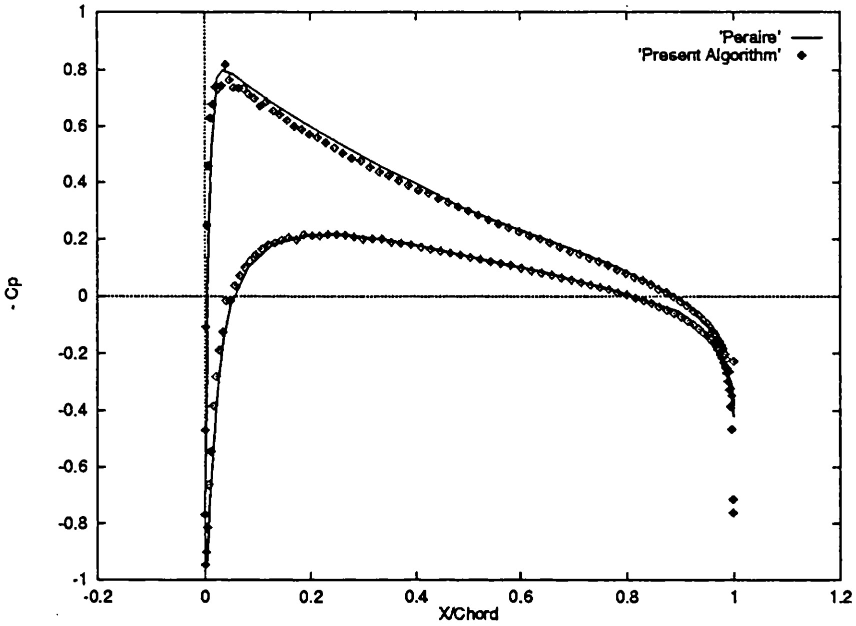

Figure 8.2.3.e *Coefficient of pressure on the aerofoil body for $\alpha = 2.0°$, comparison with the results from potential flow solution by Peraire [7].*

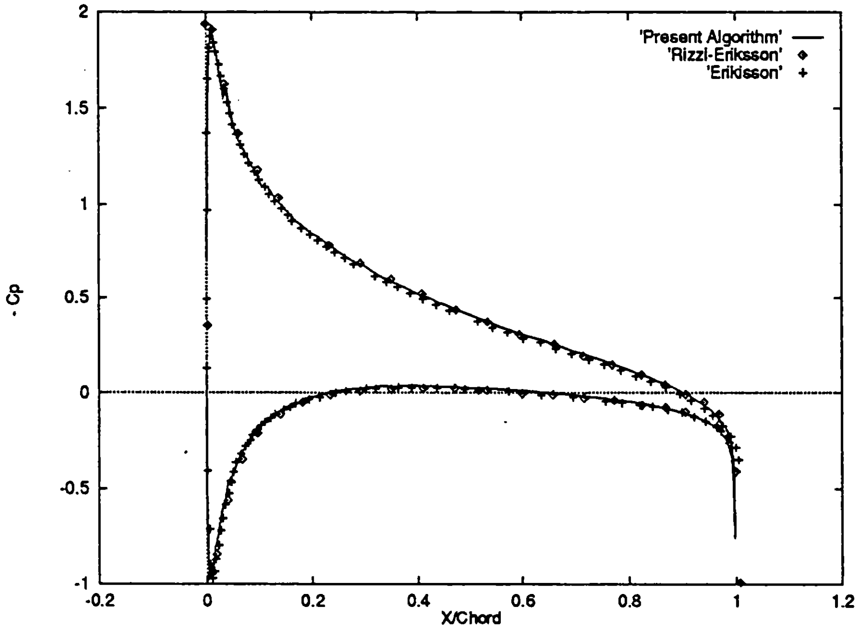

Figure 8.2.3.f *Coefficient of pressure on the aerofoil body for $\alpha = 5.0°$, comparison with other incompressible flow computations by other workers [7,9].*

Inviscid Flow Over a NACA0012 Aerofoil ($\alpha = 5.0^o$)

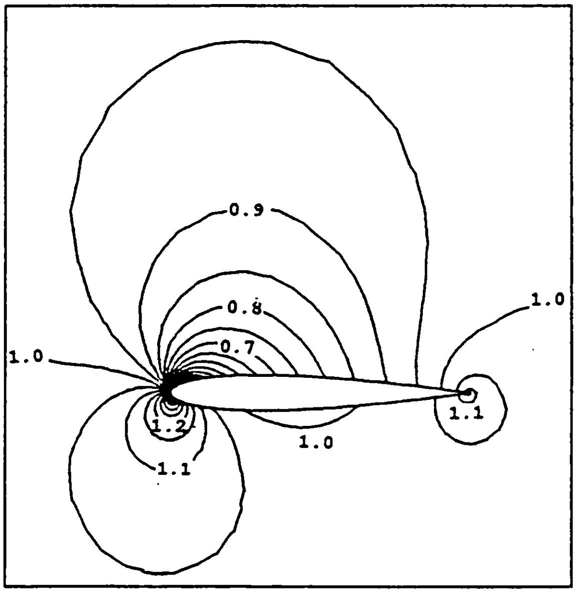

Figure 8.2.3.g *Typical pressure contours around the aerofoil (Min. = 0.011, Max. = 1.55, Intervals = 0.05).*

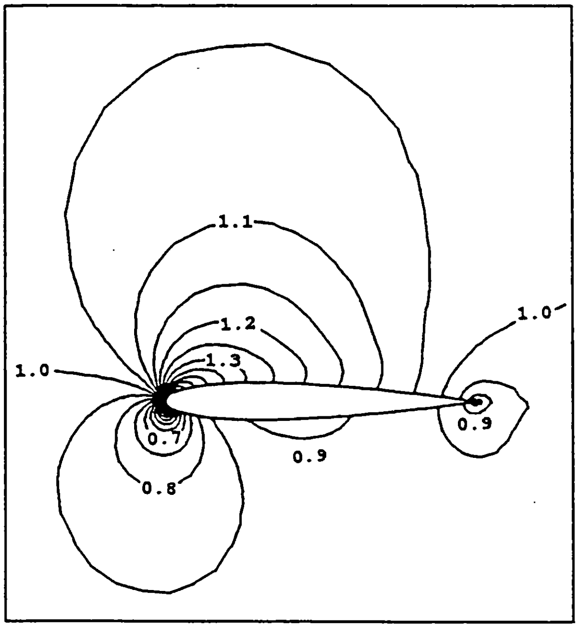

Figure 8.2.3.h *Typical velocity contours around the aerofoil (Min. = 0.035, Max. = 1.685, Intervals = 0.075).*

8.3 Viscous Flow Simulation

In this section, the accurate implementation of the viscous terms is verified by selecting some test cases which provide a good test for the viscous flow problems. For all the two-dimensional viscous cases, the viscous stresses have been evaluated using the cell-vertex finite volume formulation, together with suitable meshes which meet the requirements of this formulation. The first test case is the flow over a flat plate, which is a well known benchmark example for incompressible laminar viscous flow. For this case, the accuracy of the computed viscous stresses and resulted velocity components inside a boundary layer region can be verified by some analytical dimensionless relations. The next test case is the steady flow around a circular cylinder at Reynolds numbers below 150. This case is used to study the ability of the flow solver to simulate viscous wake regions behind an object. Some of the available results from experimental works are used to assess the accuracy of the computed results. In order to capture high gradient velocity variations inside the boundary layer and in wake regions, the minimum mesh spacing normal to the direction of dominant velocity should be less than $1/\sqrt{\Re}$.

8.3.1 Flow Over a Flat Plate

For a flat plate parallel to the free stream the flow solution was found by one of the Prandtl's first students, Blasius [10]. His mathematical similarity relationship is a unique relation between the horizontal component of the velocity u_1 and the dimensionless parameter y. This relation is valid for all the Reynolds numbers in the limit of incompressible flow. White [11] demonstrates that the comparison between the Blasius relation and the experimental results of Liepmann [12] for several low Reynolds numbers are quite good.

For this test case, the length of the flow domain is considered to be twice the height. In order to provide a proper starting point for the viscous region, a length of inviscid flow equal to half the plate length was applied. Two different types of meshes are created using two different stream-wise and anti-stream-wise direction of diagonal edges. Present computations show that the result from the two different meshes are not significantly different for these two types of meshes. Here, the results on a regular mesh are presented. The general and partial view of the applied mesh are shown in Figures (8.3.1.a) and (8.3.1.b). This grid is composed of (132 × 70) points (or 9240 points in total) and 27317 edges. The aspect ratio of the triangles located on the leading edge of the plate is equal to one. The mesh spacing is expanded from the leading edge in both directions leading to triangles with aspect ratios of 108 at the end of the plate.

For this test case, the free-stream values were considered as $(u_1)_i^0 = 1.0$, $(u_2)_i^0 = 0.0$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$. The free stream velocity components are imposed at the inflow boundary and free stream pressure at the outflow boundary. The remaining variables at inflow and outflow boundaries are obtained by extrapolation from inside the domain. Hence, all the results presented were obtained without imposing any special treatment at the intersection of the flat plate wall and the far field boundaries. The computation was performed on the grid, starting from free stream initial values. After 5000 explicit iterations, the residuals based upon pressure and velocity components reduced by seven orders of magnitude. Typical velocity and pressure contours for a Reynolds number of $\Re = 10000$ are shown in Figures (8.3.1.c) and (8.3.1.d), respectively.

The accuracy of the present computations can be assessed by comparing the computed results with the results of Blasius [10]. He derived a dimensionless coordinate, which makes the results independent of the horizontal position on the plate and Reynolds number. This relation is defined as

$$y = x_2\sqrt{\frac{\rho_0|\mathbf{U}_0|}{2\mu_0 x_1}} \tag{8.2}$$

where x_1 and x_2 are cartesian coordinates parallel and perpendicular to the direction of the free stream velocity $|\mathbf{U}_0|$, respectively. Here μ_0 and ρ_0 are the mean value of the viscosity coefficient and specific mass, respectively. For the horizontal and vertical components of velocity, u_1 and u_2, at different points normal to the plate, two parameters u and v are defined as

$$\begin{aligned} u &= \frac{u_1}{|\mathbf{U}_0|} \\ v &= \frac{u_2}{|\mathbf{U}_0|\sqrt{\Re_{x_1}}} \end{aligned} \tag{8.3}$$

Figure (8.3.1.e) presents the independency of the results from the choice of Reynolds number, using the parameter u of Blasius similarity solution for Reynolds numbers of 1000, 3000, 5000 and 10000 at a certain position of the plate, $x_1 = 0.45$. For comparison of the computed results at different positions of the plate length, a constant Reynolds number of 10000 is considered for the rest of the work. The comparison of computed u and v with the Blasius solution are presented in Figures (8.3.1.f) and (8.3.1.g). The shear stress and the skin friction coefficient at different positions on the plate length are scaled by applying the parameters

$$s = \tau_{12}\sqrt{\Re_{x_1}} \tag{8.4}$$

and

$$C_f = \frac{2\tau_{12}}{\rho_0|\mathbf{U}_0|^2} \tag{8.5}$$

The comparison of the results of s and C_f are shown in Figures (8.3.1.h) and (8.3.1.i), respectively. In general, good agreement between computational results and the parameters of Blasius similarity solution are obtained.

Flat Plate Viscous Mesh

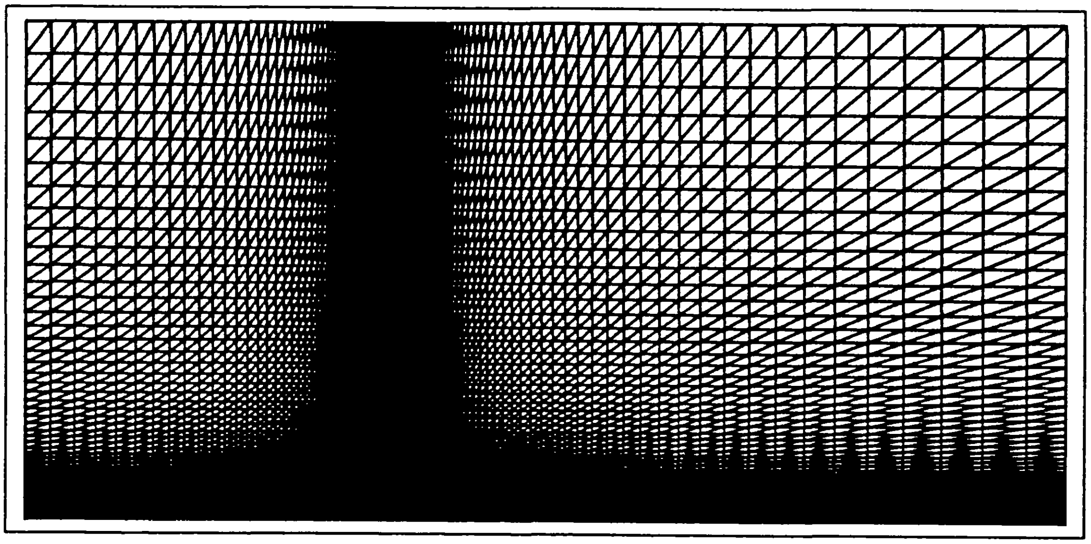

Figure 8.3.1.a General view of the regular mesh for a Flat Plate.

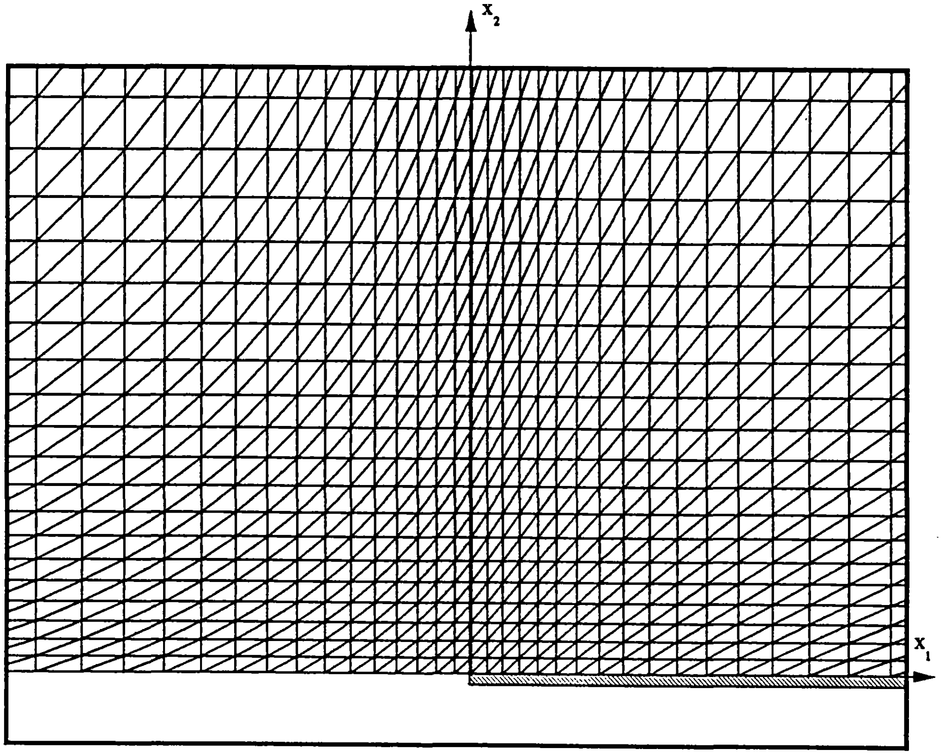

Figure 8.3.1.b Partial view of the mesh close to the Flat Plate.

Viscous Flow Over a Flat Plate

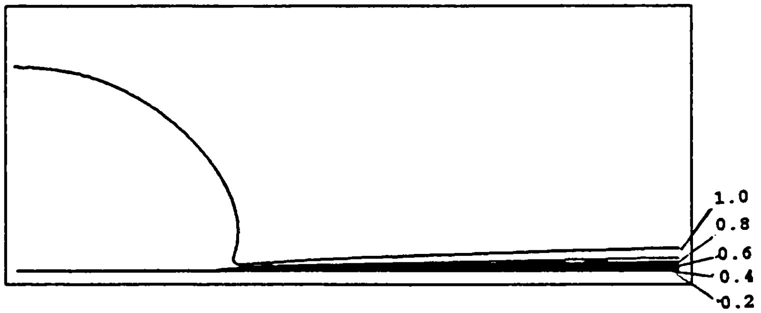

Figure 8.3.1.c *General view of velocity contours,* $\Re = 10000$ *(Min. = 0.0, Max. = 1.00, Intervals = 0.1)*

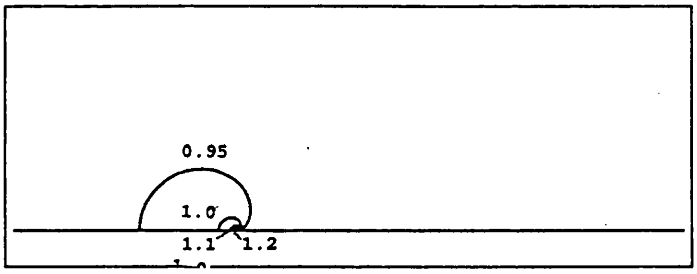

Figure 8.3.1.d *General view of pressure contours,* $\Re = 10000$ *(Min. = 0.95, Max. = 1.30, Intervals = 0.05).*

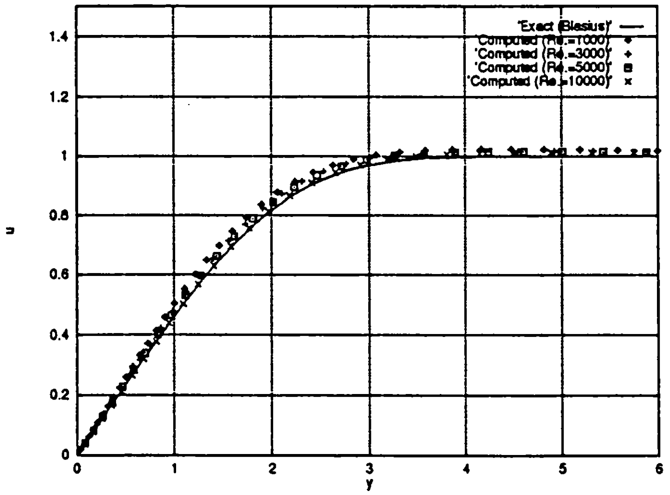

Figure 8.3.1.e *Comparison of computed* $u = u_1/|\mathbf{U}_0|$*, with Blasius similarity solution at* $x_1 = 0.45$ *for different* $\Re$ *(*$y = x_2\sqrt{\rho_0|\mathbf{U}_0|/2\mu_0 x_1}$*).*

Viscous Flow Over a Flat Plate, $\Re = 10000$

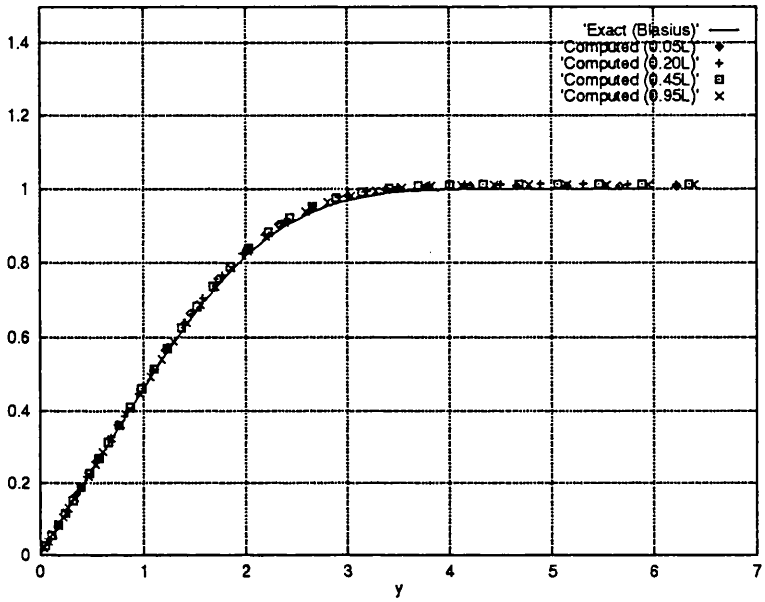

Figure 8.3.1.f *Comparison of $u = u_1/|\mathrm{U}_0|$ normal to the plate with Blasius similarity solution for different positions on the plate ($y = x_2\sqrt{\rho_0|\mathrm{U}_0|/2\mu_0 x_1}$).*

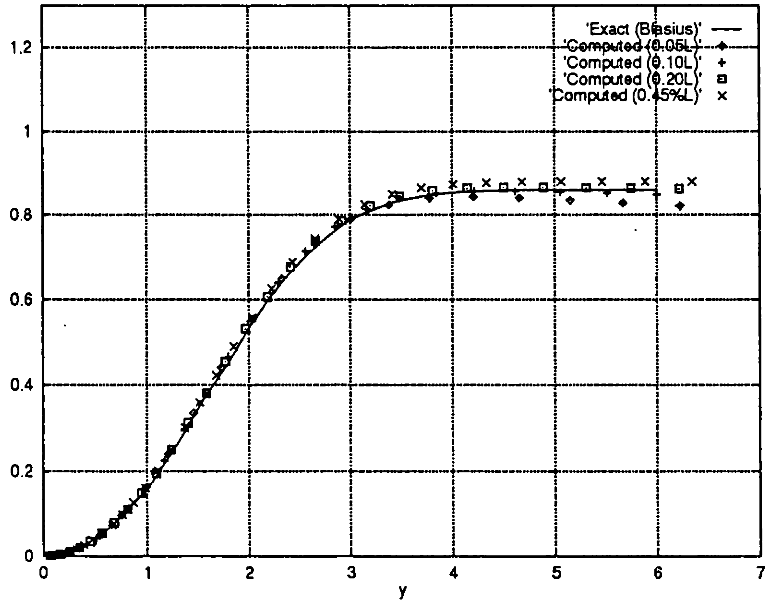

Figure 8.3.1.g *Comparison of $v = u_2/|\mathrm{U}_0|\sqrt{\Re_{x_1}}$ normal to the plate with Blasius similarity solution for different positions on the plate ($y = x_2\sqrt{\rho_0|\mathrm{U}_0|/2\mu_0 x_1}$).*

Viscous Flow Over a Flat Plate, $\Re = 10000$ (Continued)

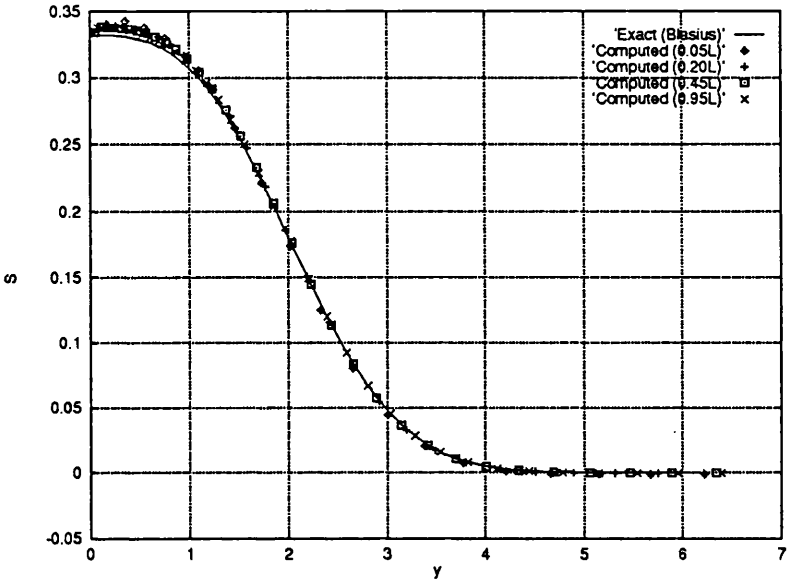

Figure 8.3.1.h *The shear stress parameter ($S = \tau_{12}\sqrt{\Re_{x_1}}$), normal to the plate compared with the Blasius similarity solution ($y = x_2\sqrt{\rho_0|\mathrm{U}_0|/2\mu_0 x_1}$).*

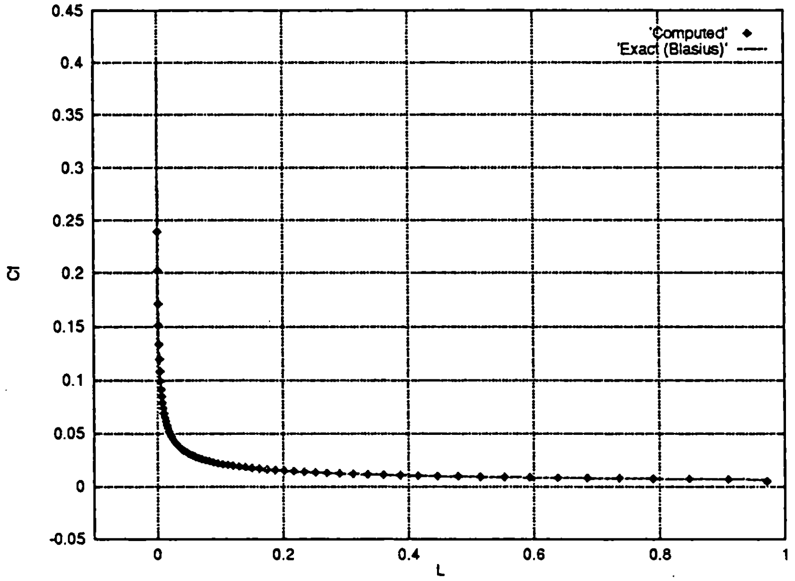

Figure 8.3.1.i *Skin friction coefficient ($C_f = 2\tau_{12}/\rho_0|\mathrm{U}_0|^2$) compared with the Blasius exact solution at different positions on the plate length $L = x_1/\ell$.*

8.3.2 Flow Around a Circular Cylinder

This test case, the viscous laminar flow around a cylinder, can be used to study the ability of the solver to simulate wakes. Experiments show that the wake flow behind a cylinder remains steady up to a certain Reynolds number. The laminar vortex streets will appear for $50 < \Re < 150$ and the flow can not then be considered steady. There will be turbulent vortex shedding with Reynolds numbers $\Re \geq 150$ [13]. Hence, for the present laminar flow computations, simulations are performed for $\Re < 150$.

Flows of this kind are applicable to simulate the flow around large scale structures. Structures such as large empty storage tanks are subject to external wind loading. The pressure on this cylindrical geometry, which develops due to the wind flow, may cause disastrous results. An example of this problem is the wind induced collapse of oil storage tanks at Haydock, Lancashire, England, in 1967, Figure (8.3.2.a). Therefore, this problem has been subject of the several theoretical and experimental studies, Figure (8.3.2.b).

For this test case, a two-dimensional theoretical formulation has been developed by Acrivos et al [14] and experimental results are available after the work of same author [15]. The experimental test involves measurements of the length of the wake bubbles, as well as measurements of the rear pressure behind the cylinder, for Reynolds numbers below 150. In this experimental work the steady flow was enforced by imposing a splitter plate behind the cylinder to remove the vortex street effects.

In our simulations for this test case, both regular and irregular meshes have been used. Figure (8.3.2.c) shows a view of the regular mesh used in the computation. The outer boundary was placed at a distance of 10 chords away from the centre of the cylinder. The regular grid is composed of 7200 nodes, 14400 cells, and 21720 edges. There are 120 grid points distributed on the cylinder surface, which can be seen in Figure (8.3.2.d). A finer regular mesh of 240×60 nodes is also examined but there was no significant improvement in accuracy of the results. In this mesh, to simulate the experiments, a symmetry line is imposed at rear side of the cylinder. For the computations of this test case, the free-stream values were considered as $(u_1)_i^0 = 1.0$, $(u_2)_i^0 = 0.0$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$.

The accurate and efficient computation of the flow fluid for the case on irregular unstructured meshes requires not only an appropriate solution algorithm but also adaptive refinement of the mesh. Hence, the manual adaptivity based upon the features of the flow field is developed for this steady flow problem in which the grid in regions with steep gradients of velocity components are defined. The procedure can involve a complete redefinition of the mesh, which is called remeshing. This process requires information to generate a new mesh in an attempt to achieve a solution which satisfies an intended quality of the results. Adaptive remeshing helps to confine the high resolution grid clusterings only in required regions, and hence provides considerable efficiency.

For a Reynolds number of 25, two irregular meshes were employed using mesh adaptivity procedure. These irregular meshes which were generated used the method of Advancing Layers or Advancing Normals [14,15], and consist of regular layers with minimum aspect ratio cells equal to 0.1. The adapted meshes were generated by manually refining the viscous wake region behind the cylinder. The refinement of the mesh was driven by the previous computed flow fields. The adaptivity procedure was achieved using sources points and lines, which were placed within the viscous wake region behind the cylinder to control local point spacing of the mesh [16]. By considering appropriate amplification and decay parameters for each point or line source adequate grid clustering can be defined within the regions of steep gradients of the flow variables in the wake region. Partial views of two adapted meshes, are presented in Figure (8.3.2.e) and Figure (8.3.2.f). The first mesh contains 15925 nodes, 31493 elements and 47418 edges and the second mesh contains 30061 nodes, 59708 elements and 89769 edges. Unlike the case with the regular mesh, no symmetric horizontal wall boundary condition was imposed at the rear of the cylinder.

For all the cases, the computations took less than 5000 iterations to converge and approximately more than 7 orders of magnitude in the reduction of the pressure residuals was reached on SGI machine by using double precision. The convergence behaviour of the pressure and velocity on the regular mesh for Reynolds number of 25 are shown in Figure (8.3.2.g). The resulting stream lines, velocity and pressure contours are presented in Figures (8.3.2.h), (8.3.2.i) and (8.3.2.j), respectively.

In order to evaluate the accuracy of the algorithm, the measured distance of the wake stagnation point from centre of the cylinder, $L = l_{wake}/\ell$, where ℓ is the cylinder diameter, can be compared with the computed results. In Figure (8.3.2.k) the predicted position of the wake stagnation point for the Reynolds number 17.5, 25, 50, 75, and 150 on regular and irregular meshes are compared with experimental results. The computed skin friction and pressure coefficient on the cylinder surface are shown in Figure (8.3.2.l). It is observed that the pressure at the rear stagnation point of the cylinder is nearly equal -0.5, which is identical to the experimental measurement [15]. The comparison of the results obtained by the present algorithm are in good agreements with experimental data. Hence, the design of structures can particularly take advantage of the pressure loads computed from the CFD algorithm.

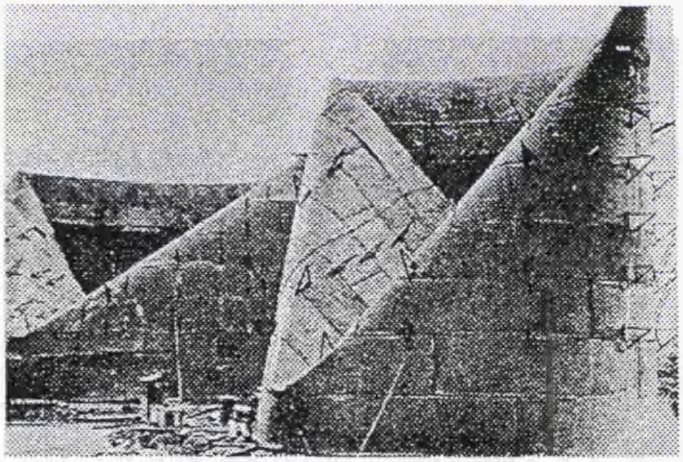

Figure 8.3.2.a *Wind-induced collapse of oil storage tanks, Haydock, Lancashire, England, in 1967.*

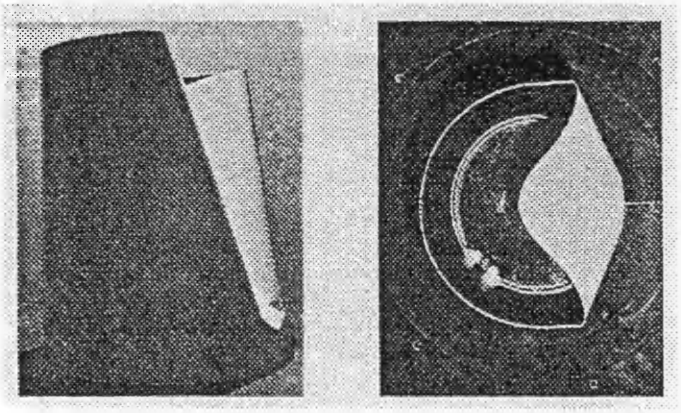

Figure 8.3.2.b *Result of a wind tunnel simulation of the collapse of the shells of oil storage tanks.*

Viscous Regular Mesh around a Circular Cylinder

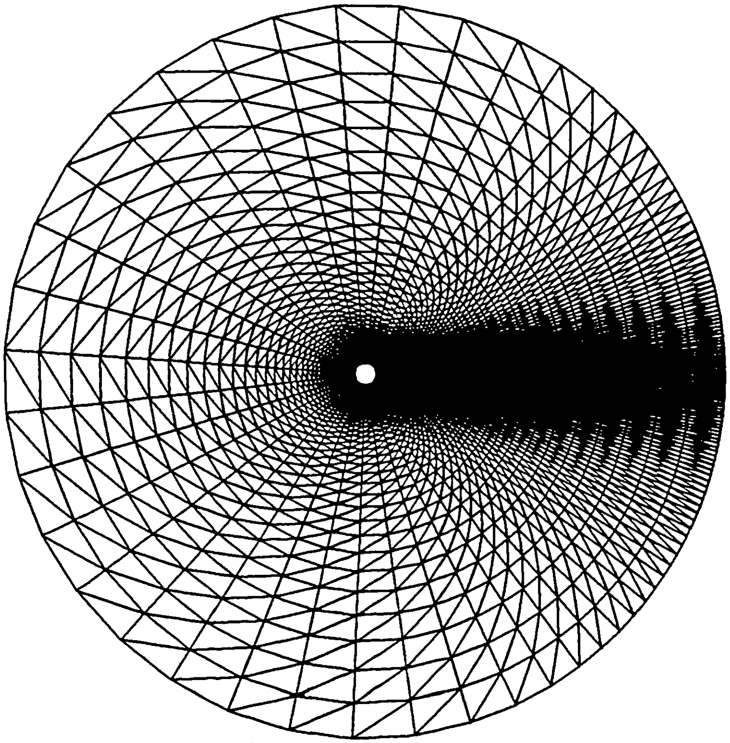

Figure 8.3.2.c *The mesh for viscous flow around a cylinder*

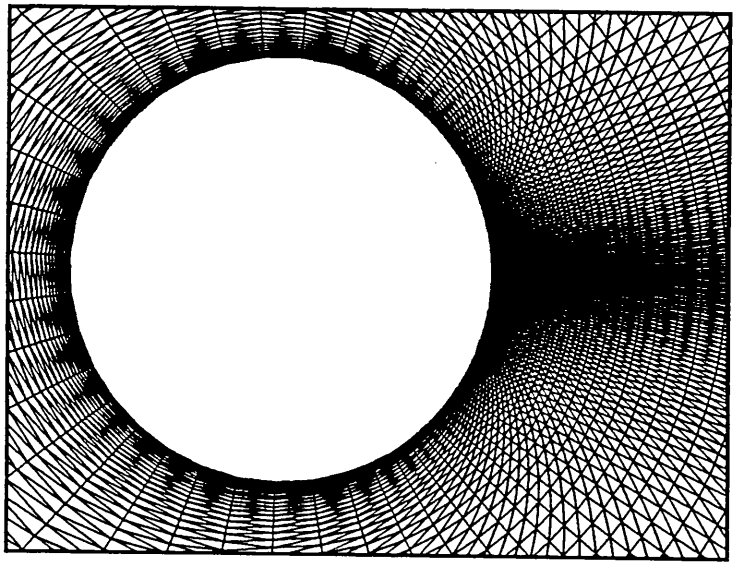

Figure 8.3.2.d *Partial view of the regular triangular mesh*

Irregular Adapted Meshes around a Circular Cylinder

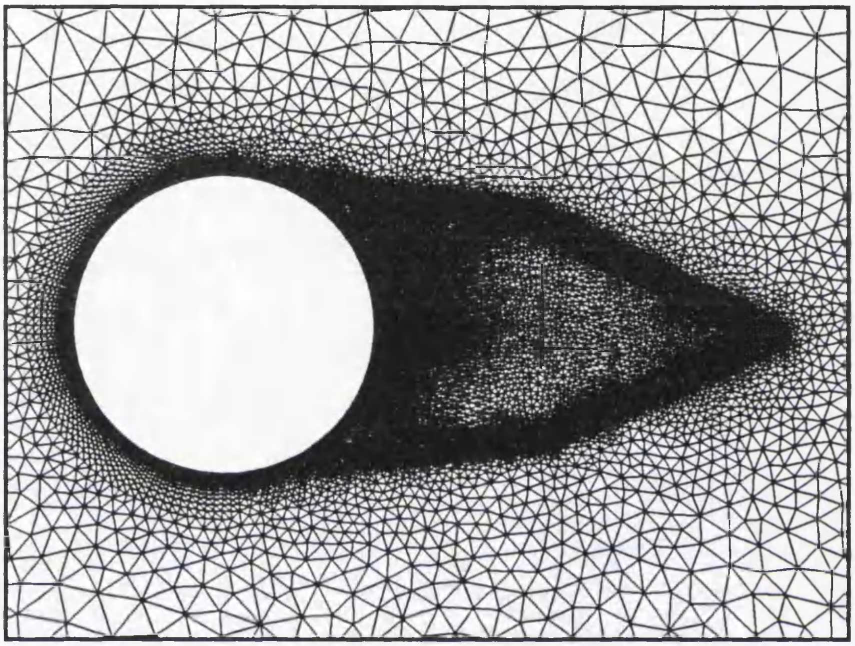

Figure 8.3.2.e *Partial view of the first mesh adapted for* $\Re = 25$.

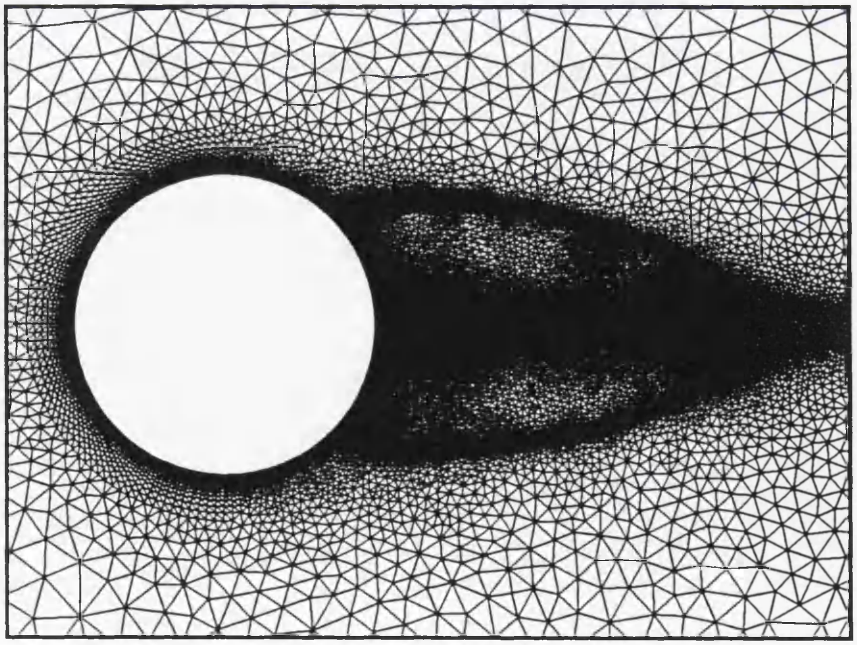

Figure 8.3.2.f *Partial view of the second mesh adapted for* $\Re = 25$.

Viscous Flow around a Circular Cylinder, $\Re = 25$

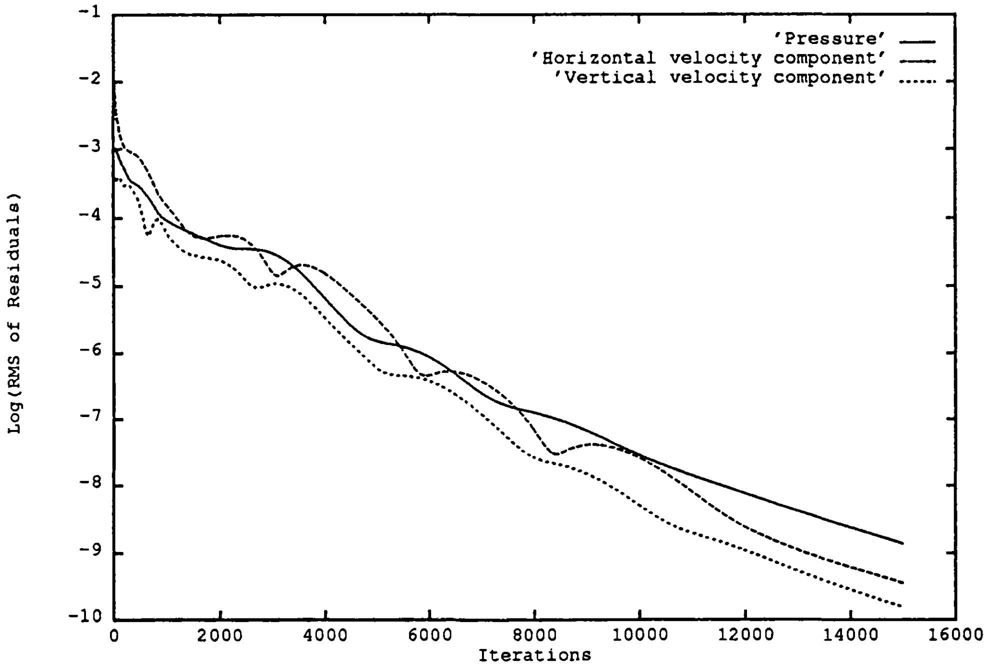

Figure 8.3.2.g *Typical convergence behaviour of velocity components and pressure residuals.*

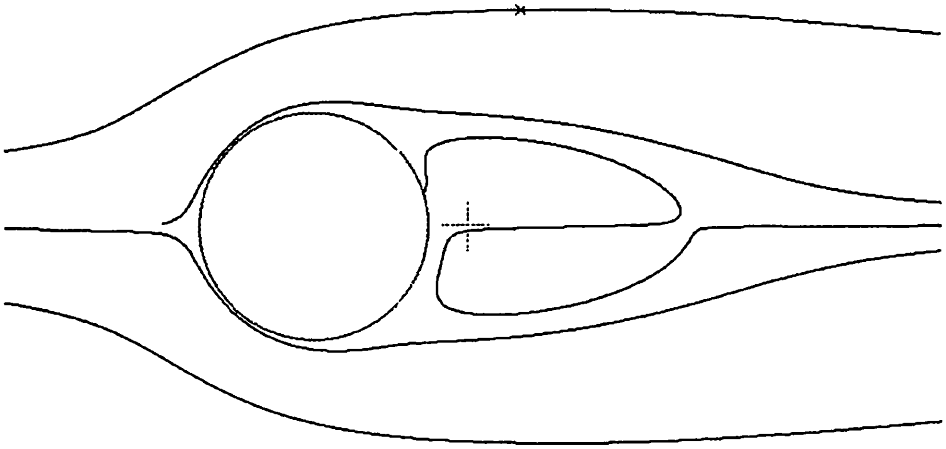

Figure 8.3.2.h *Typical stream lines which show wake region behind the circular cylinder.*

Viscous Flow around a Circular Cylinder, $\Re = 25$

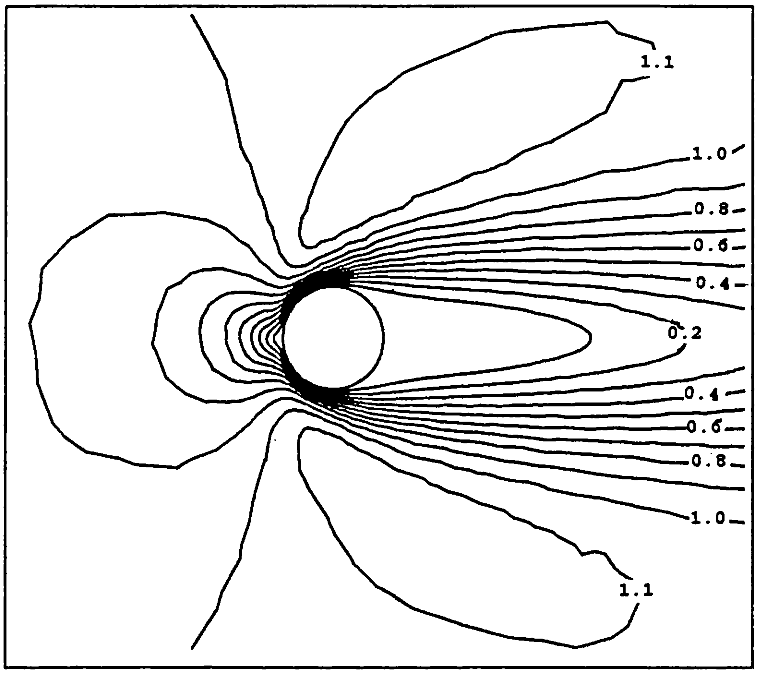

Figure 8.3.2.i *Typical velocity contours around the cylinder (Min.= 0.0, Max. = 1.16, Intervals =0.1)*

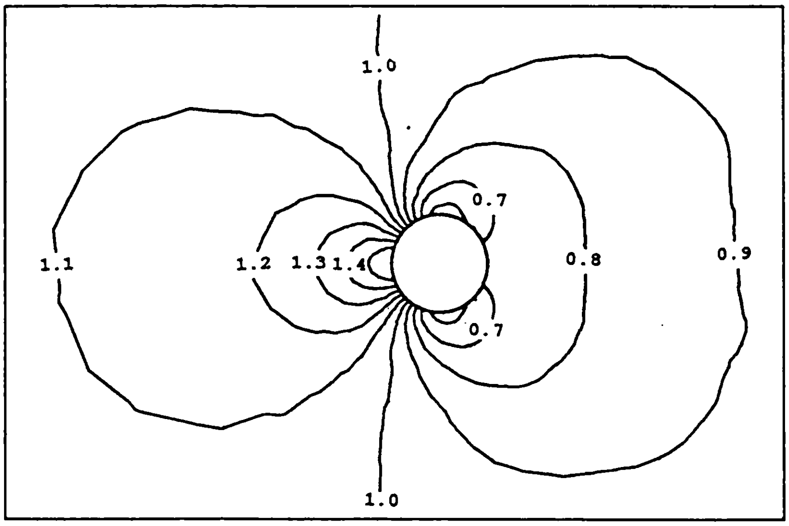

Figure 8.3.2.j *Typical pressure contours around the cylinder (Min.= 0.54, Max. = 1.63, Intervals =0.1)*

Viscous Flow around a Circular Cylinder

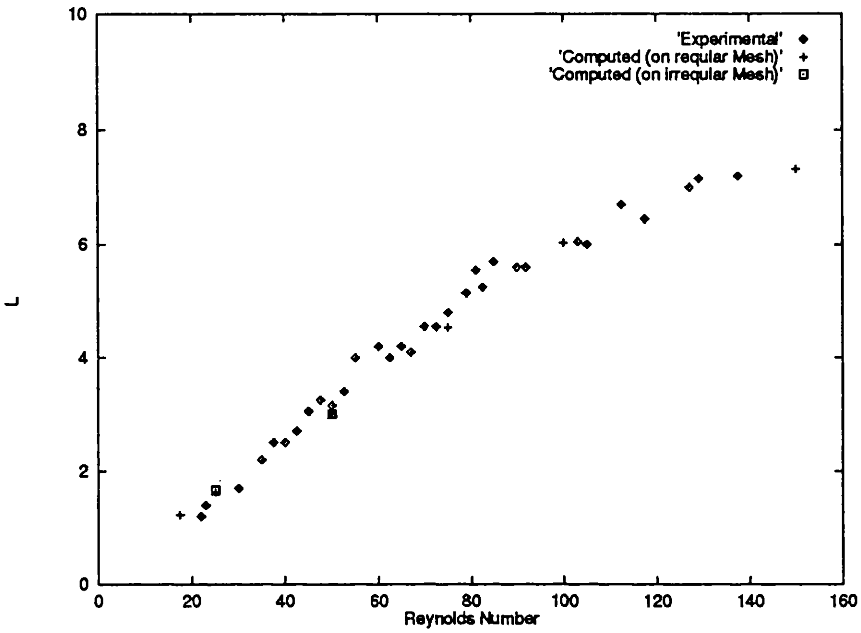

Figure 8.3.2.k *The computed wake stagnation point, $L = l_{wake}/\ell$, compared with the experimental results [15].*

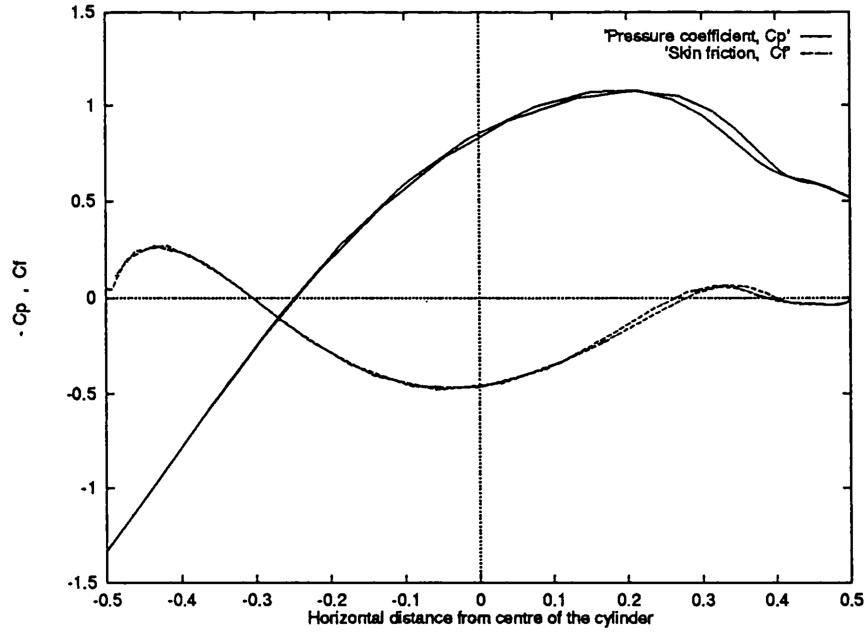

Figure 8.3.2.l *Coefficient of Pressure, $C_p = 2(p_i - p_0)/\rho_0|\mathrm{U}_0|^2$ and Skin friction, $C_f = 2\tau_{12}/\Re_0\rho_0|\mathrm{U}_0|^2$ on the cylinder for $\Re = 25$.*

8.4 Applications

In this section, the two-dimensional incompressible flow solver is employed to simulate some of the flow problems in the field of engineering. As an application for the two-dimensional inviscid flow solver, the first case involves the simulation of wind flow over Swansea University Campus. This case demonstrates the ability of flow solver to compute the flow field which can be used for environmental planning purposes. The effect of changing the arrangement of the buildings was presented by considering the wind flow from different directions. The next application case is considered as the viscous flow over an aerodynamic configuration. The simulation of flow over a NACA0012 aerofoil presents the validation of the viscous flow solver to deal with a sharp trailing edges geometry. The results of the computation of this case can be used to investigate the ability of the flow solver to capture boundary layer separation and resulting recirculation near trailing edge of the aerofoil.

8.4.1 Wind Flow Through University Campus

The velocity and direction of the wind streams between buildings in urban area is a major consideration. This situation can produce unpleasant conditions for pedestrians and may even cause damage to the structures. The results of the present flow solver can be used to help predict the flow field. Such an evaluation helps to design the optimal position of buildings and can be used in the field of environmental engineering. Assuming that the wind flow is in a very high Reynolds number, the viscous regions will be confined in thin layer close to the solid wall, and hence, the rest of the flow field can be considered inviscid. Therefore, the wind flow over two-dimensional plan of Swansea University Campus is simulated using the inviscid flow solver. Considering the wind from different directions, the calculations are made for 0.0 and 180.0 degree angles of incidence.

The grid employed is fully unstructured and was created by the Delaunay triangulation. The computational domain consists of 21453 triangles. The number of points is 11519 with 1635 points located on the walls of different buildings. A general and the partial view of the grid used for this calculation are shown in Figures (8.4.1.a) and (8.4.1.b). The computation of this test case started from the free-stream values as $(u_1)_i^0 = Cos(\alpha)$, $(u_2)_i^0 = Sin(\alpha)$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$ where α is the angle with the horizontal cartesian axe x_1. Figures (8.4.1.c) and (8.4.1.d) are the velocity vectors and streamlines resulting from wind flow at 0.0 degree of incidence to the x_1 axis. Figures (8.4.1.e) and (8.4.1.f) present the same results for a flow at 180.0 degree of incidence to the x_1 axis. As can be seen, the arrangement of the buildings with respect to the flow direction has an important effect on the flow direction and velocity magnitude at different positions of the campus. Although there is not any documented measurements for the local direction of wind between the main buildings, local observations show that the results from the inviscid solver are qualitatively realistic.

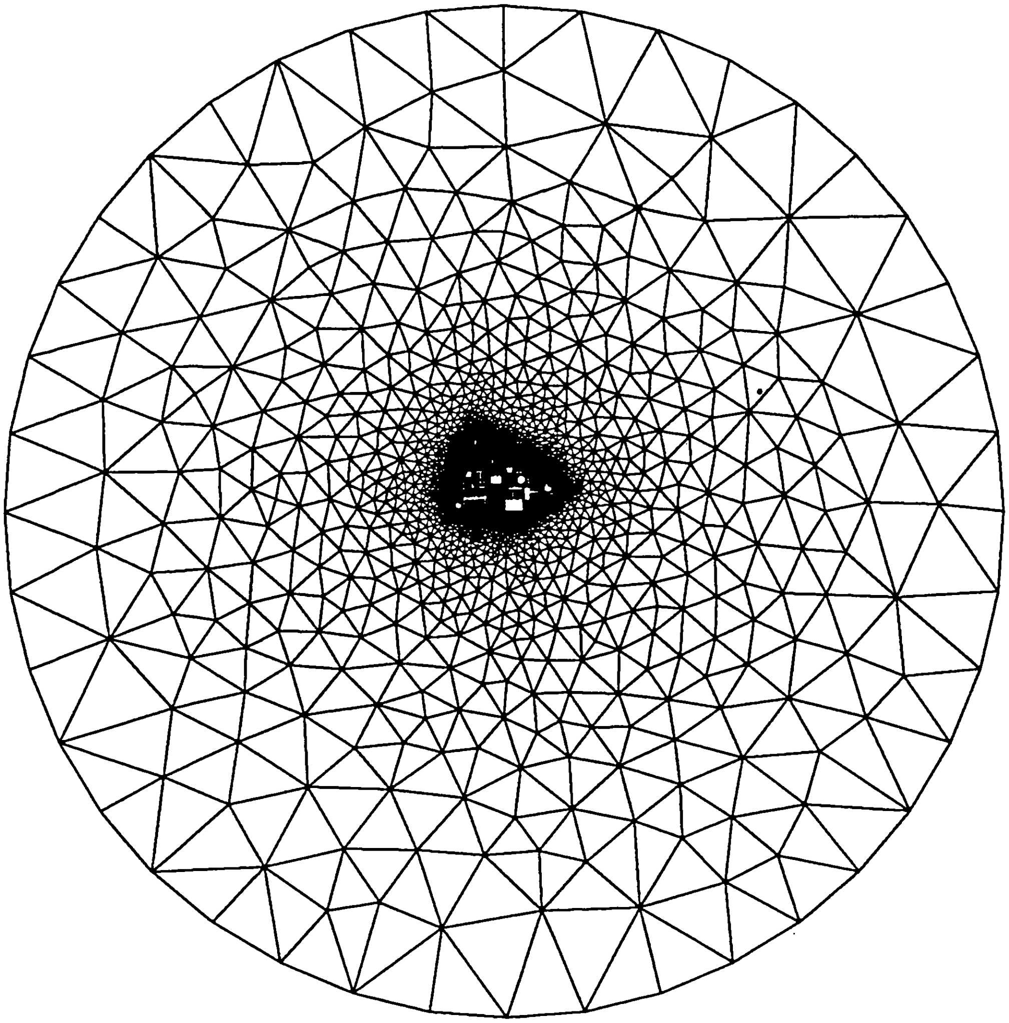

Figure 8.4.1.a *Unstructured mesh for Swansea University Campus*

Wind Flow over Simulation University Campus

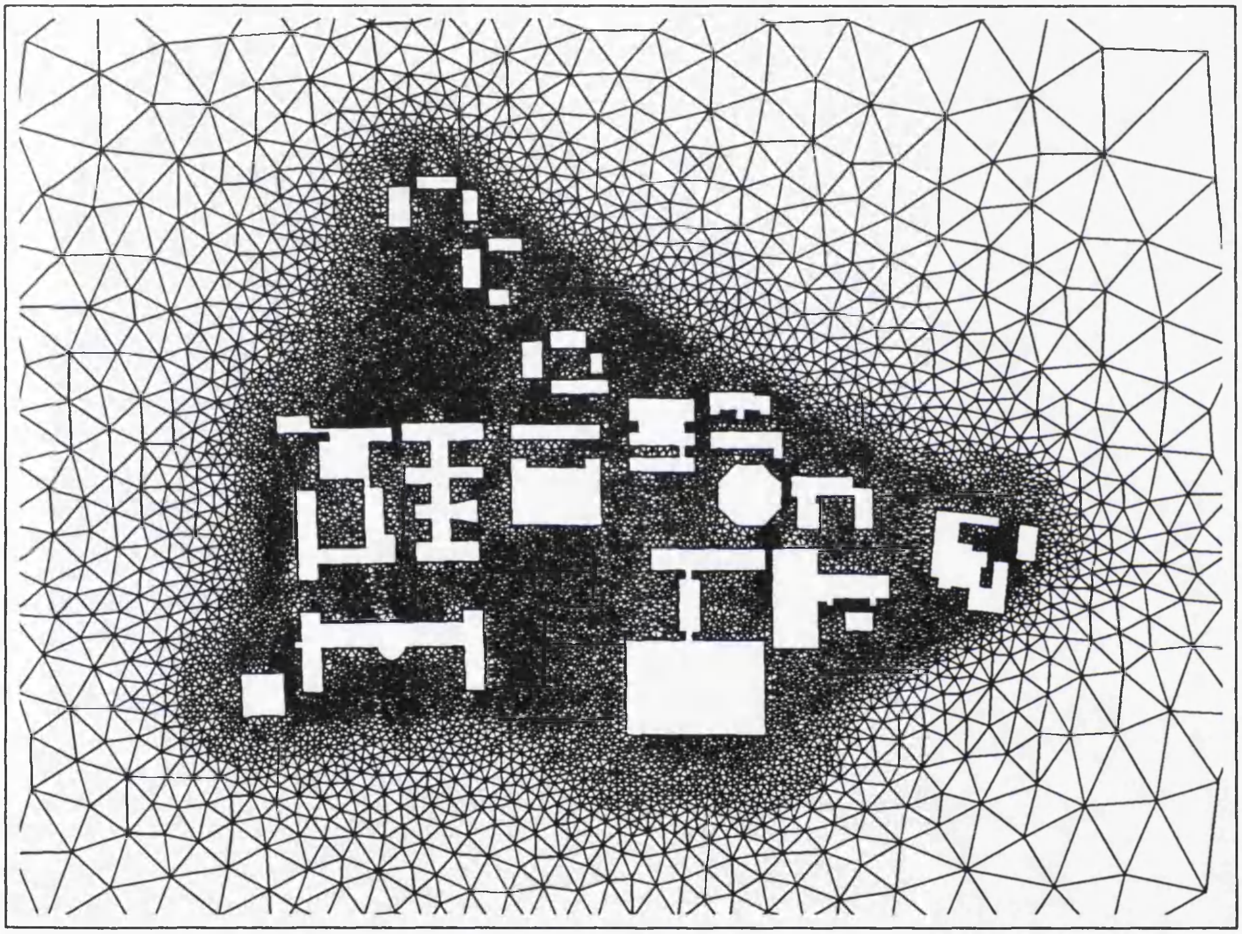

Figure 8.4.1.b *Close view of the mesh for University Campus, generated by Delaunay triangulation method.*

Wind Flow Simulation over University Campus

(Angle of Incidence $\alpha = 0.0°$)

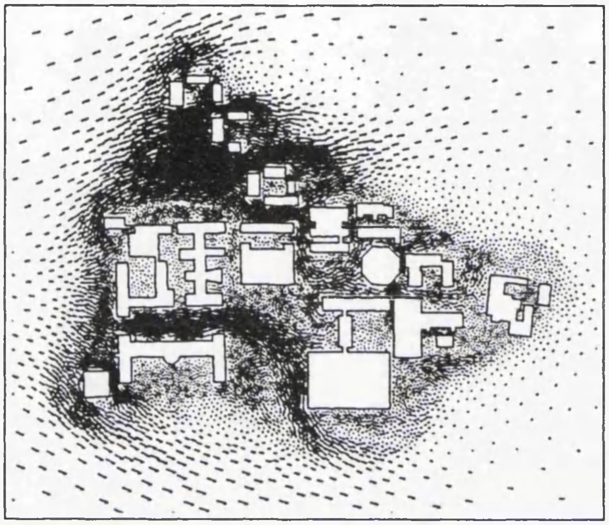

Figure 8.4.1.c *Typical velocity vectors through campus.*

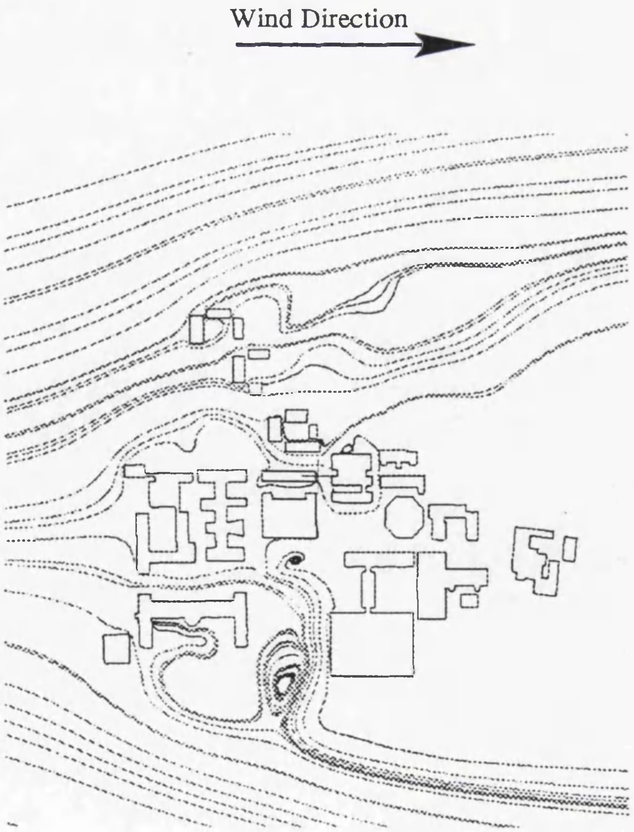

Figure 8.4.1.d *Resulting streamlines from inviscid flow simulation.*

Wind Flow Simulation over University Campus

(Angle of Incidence $\alpha = 180.0°$)

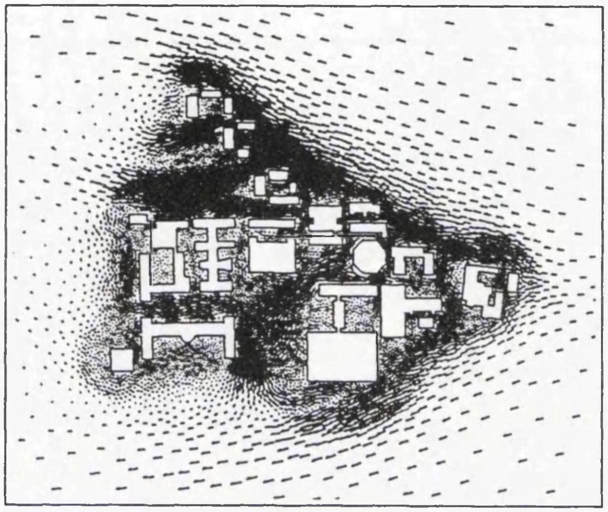

Figure 8.4.1.e *Typical velocity vectors through the campus.*

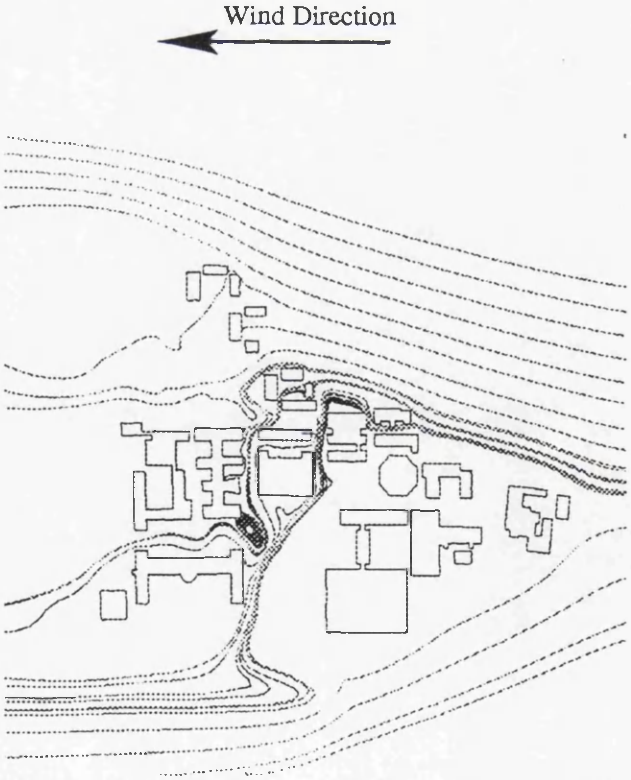

Figure 8.4.1.f *Resulting streamlines from inviscid flow simulation.*

8.4.2 Viscous Flow Over The NACA0012 Aerofoil

Here, the ability of the flow solver to simulate the viscous flow over an aerodynamic configuration is investigated using a NACA0012 aerofoil. For this aerofoil which is widely used in many CFD experimental and numerical investigations, boundary layer separation and recirculation takes place close to the trailing edge. A well documented set of results is available for simulating inviscid and viscous flow over this aerofoil [17-22]. This case was considered to examine the behaviour of the flow solver to deal with this sharp ended configuration. The view of the regular grid, which is used for this calculation is shown in Figures (8.4.2.a) and (8.4.2.b). The outer boundary is at a distance of 12 chords from the aerofoil. The computational domain consists of 4480 triangular cells. The number of the points is 8704 with 128 points located on the aerofoil surface and the total number of the edges is 13184.

The computation of this test case started from the free-stream values as $(u_1)_i^0 = Cos(\alpha)$, $(u_2)_i^0 = Sin(\alpha)$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$ where The computation for the Reynolds number of 5000 and 0.0 angle of incidence was started from free stream values. The typical convergence of this case took less than 5000 iterations for a reduction of eight orders of magnitude. The convergence history of the computed pressure and velocity residuals are plotted in Figure (8.4.2.c). In order to provide a general sense of flow field around the aerofoil, the results such as typical pressure contours Figure (8.4.2.d) , velocity contours Figure (8.4.2.e) and velocity vectors Figure (8.4.2.f) around the aerofoil are presented. Details of recirculation due to thickening of the viscous region, which cause flow separation, can be clearly seen in Figure (8.4.2.g). As can be observed in Figure (8.4.2.g), the recirculation near the trailing edge appears due to flow separation. This separated flow region appeared in the predictions of Swanson and Turkel [19]. The computed coefficients of surface pressure C_p and skin friction C_f on the aerofoils surface are presented in Figures (8.4.2.h) and (8.4.2.i), respectively. The formulation for C_p is given by equation (8.1) and C_f is defined by equation (8.5). Although most of the results of computations for $\Re = 1000$ are presented, the calculation was successfully made for several Reynolds numbers in the range of $50 < \Re < 5000$ at 0.0 degree angles of incidence. For different Reynolds numbers of 50, 200 and 5000, the coefficients of surface pressure C_p and skin friction C_f on the both side of aerofoils surface are presented in Figures (8.4.2.j) and (8.4.2.k) .

Viscous Regular Mesh around a NACA0012 Aerofoil

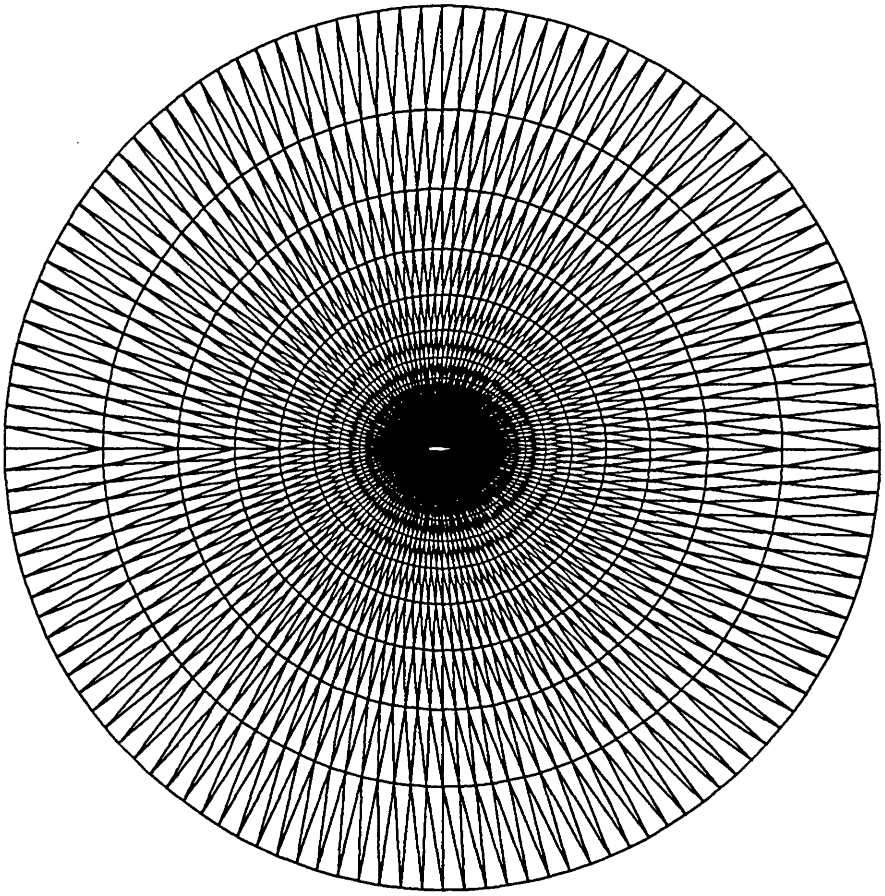

Figure 8.4.2.a *Regular symmetric mesh around the aerofoil.*

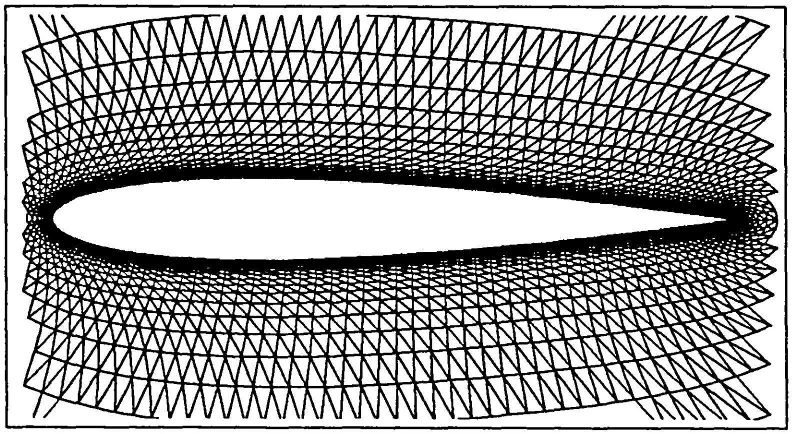

Figure 8.4.2.b *Partial view of the mesh around the aerofoil.*

Viscous Flow over a NACA0012 Aerofoil ($\alpha = 0.0°, \Re = 5000$)

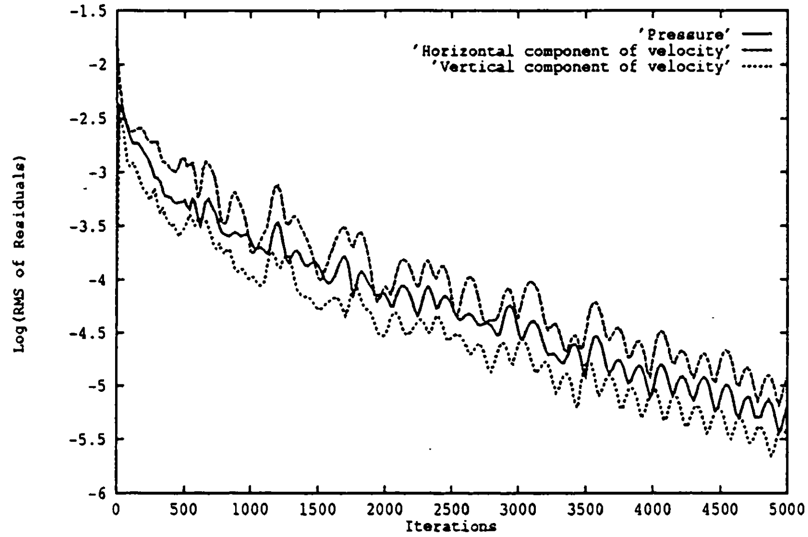

Figure 8.4.2.c *Convergence of Pressure and Velocity residuals.*

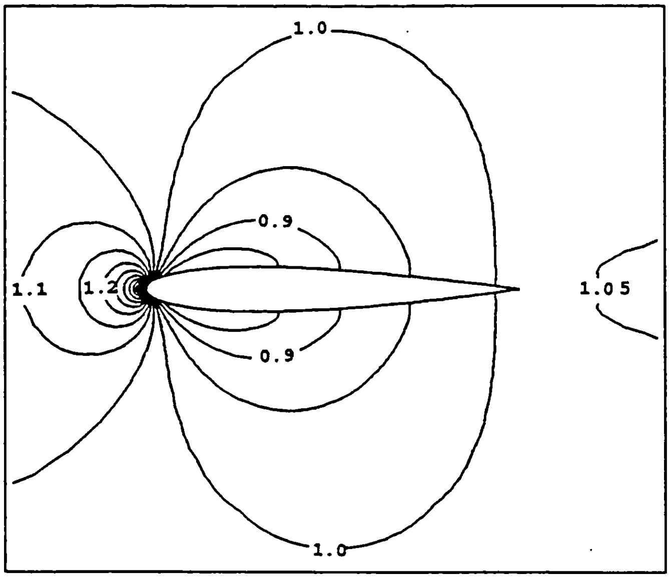

Figure 8.4.2.d *Typical pressure contours around the aerofoil (Min. =0.811 Max.= 1.57 Intervals = 0.1).*

Viscous Flow over a NACA0012 Aerofoil ($\alpha = 0.0^o, \Re = 5000$)

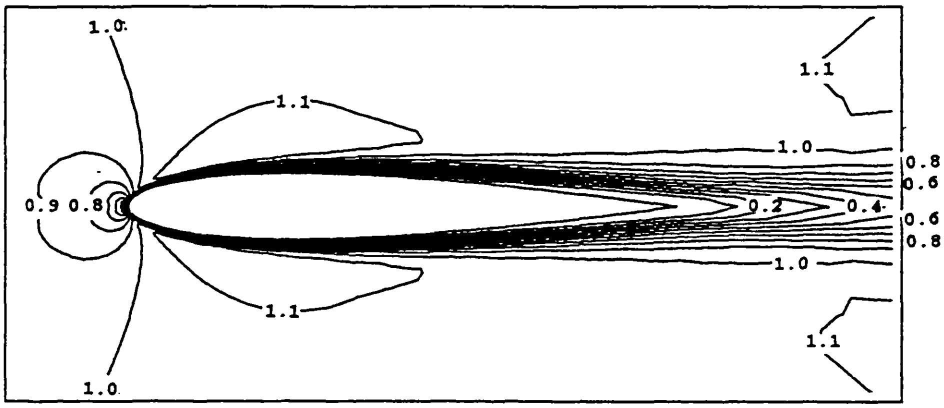

Figure 8.4.2.e *Typical velocity contours around the aerofoil (Min. =0.0 Max.= 1.16 Intervals = 0.1).*

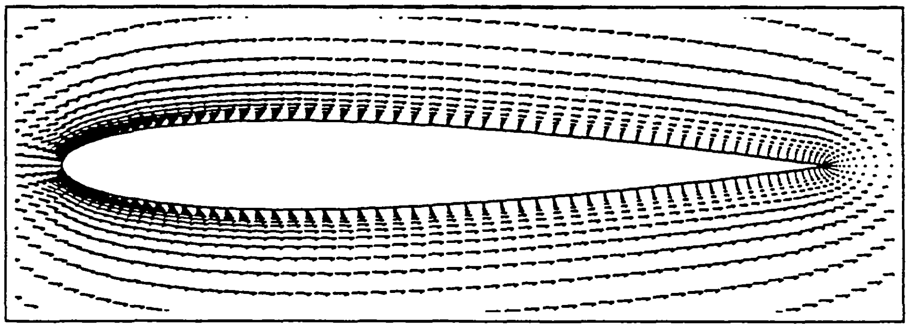

Figure 8.4.2.f *Typical velocity vectors around the aerofoil.*

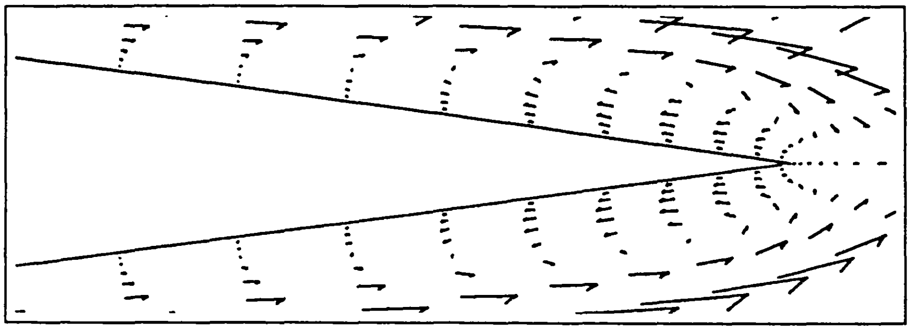

Figure 8.4.2.g *Details of the velocity vectors at the trailing edge of the aerofoil.*

Viscous Flow over a NACA0012 Aerofoil ($\alpha = 0.0^o, \Re = 5000$)

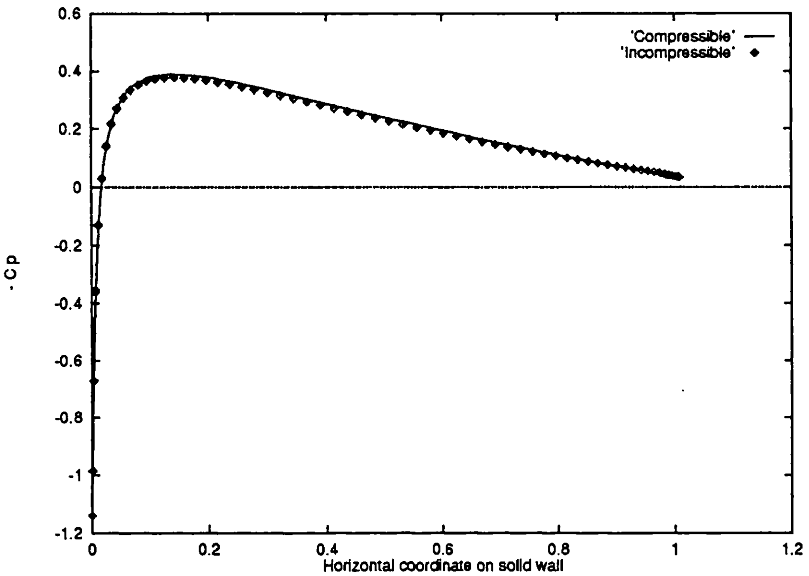

Figure 8.4.2.h *Pressure coefficient on the aerofoil surface.*

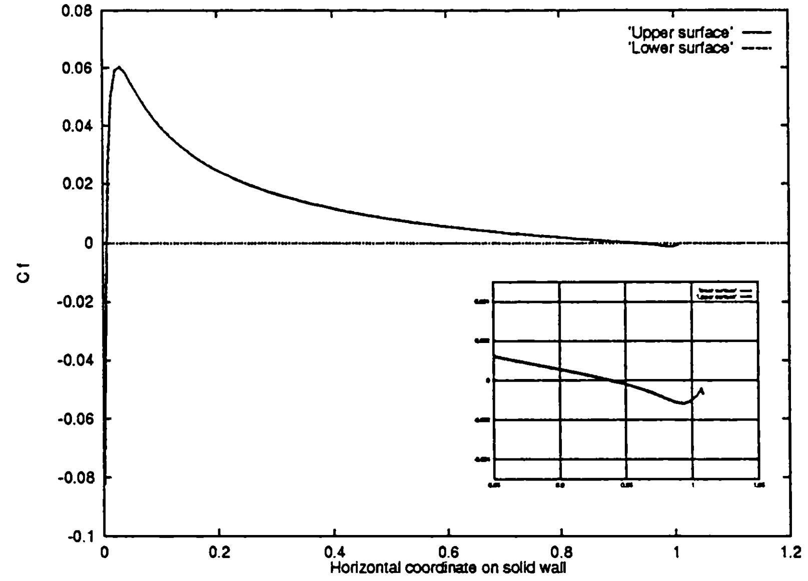

Figure 8.4.2.i *Skin friction coefficient on the aerofoil body surface.*

Viscous Flow over a NACA0012 Aerofoil ($\alpha = 0.0^o$)

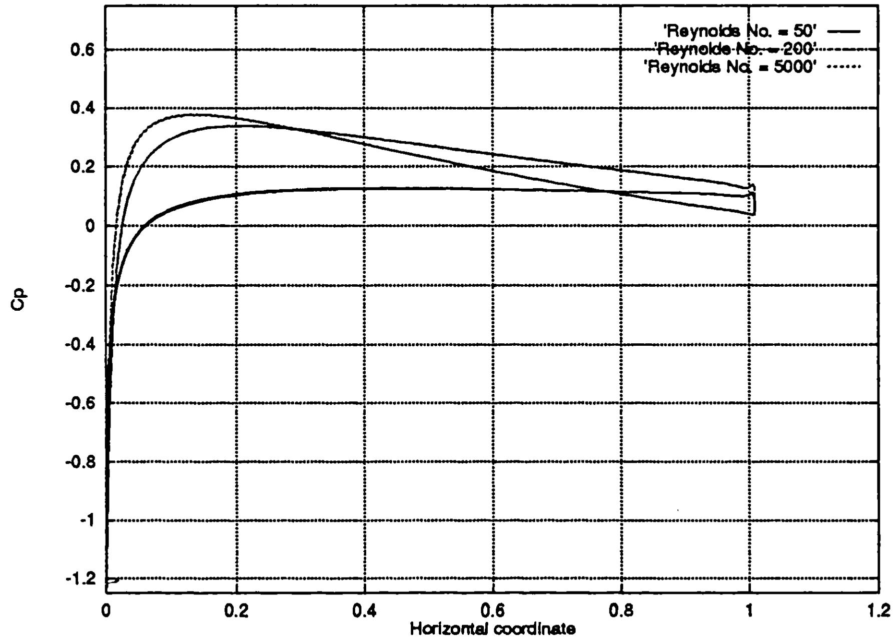

Figure 8.4.2.j *Pressure Coefficient on the aerofoil surface, for different Reynolds numbers.*

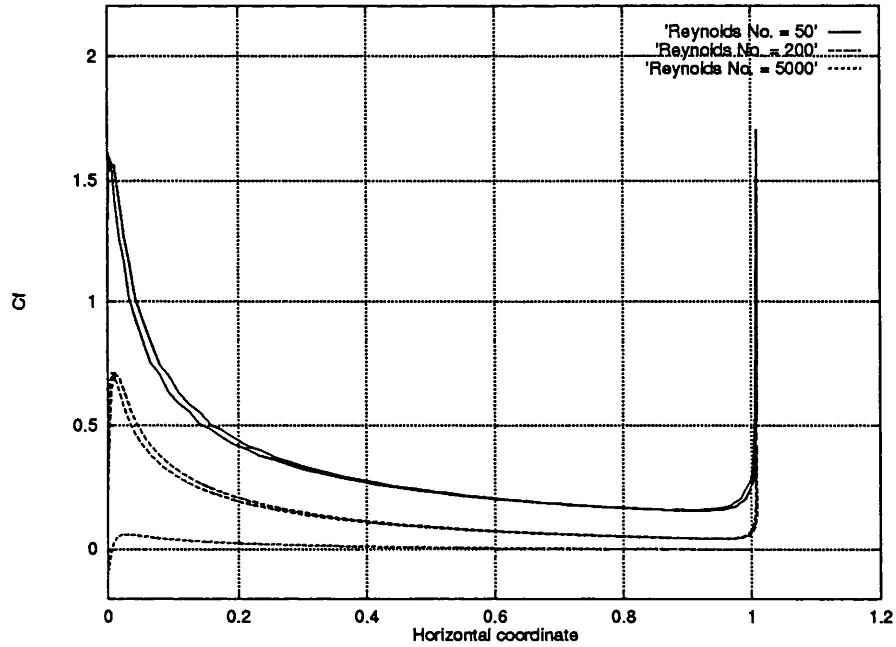

Figure 8.4.2.k *Skin friction Coefficient on the aerofoil surface, for different Reynolds numbers.*

8.5 Summary

In this chapter, results from the two-dimensional inviscid and viscous algorithms have been presented. From the test calculations the following points can be made.

For two-dimensional inviscid flow computations acceptable results have been predicted on irregular unstructured meshes without normalisation of the artificial dissipation term. The accuracy of the results have been improved by refining the mesh spacing close to the solid wall boundary.

In two-dimensional viscous flow simulations, diffusive terms have been computed by implementing a cell-vertex finite volume formulation for computing viscous stresses and satisfactory results have been obtained.

Both inviscid and viscous flow solvers have been employed for simulation of some applied problems in the field of engineering.

References

[1] L. M. Milne-Thomson, 'Theoretical Hydrodynamics', The Macmillan Press, 5th Edition, pp.158 (1979).

[2] R. H. Ni, 'A Multiple Grid Scheme for Solving the Euler Equations', AIAA Journal, **20**, No. 11, 1981.

[3] D. Choi, C. L. Merkle, 'Application of Time Iterative Scheme to Incompressible Flow', AIAA Journal, **23**, pp. 1518–24, (1985).

[4] A. Chorin , 'A Numerical Method for Solving Incompressible Viscous Flow Problems',J. of Comp. Physics , **2**, pp. 12–26, (1967).

[5] W. R. Briley, H. MacDonald, 'On the Structure and Use of Linearised Block Implicit Schemes', J. of Comp. Physics, Vol **34**, pp. 54–73, (1980).

[6] J. Wu, 'Compressible and Incompressible Flow Problems - A New Finite Element Algorithm and Adaptivity' , Ph.D, University of Wales, Swansea, (1992)

[7] J. Peraire, 'Potential Flow Past Arbitrary Aerofoils Using Conformal Mapping' , Private Communication.

[8] A. Rizzi, L.E.Eriksson, 'Computation of Inviscid Incompressible Flow with Rotation', J. of Fluid Mech., **153**, pp. 275–312, (1985).

[9] L. E. Eriksson, 'Calculation of Two-dimensional Flow Wall Interface for Multi Component Aerofoils in Closed Low Speed Tunnels', FFATN AU-1116, Part 1, Stockholm, (1975).

[10] H. Blasius, NACA Technical Memorandum 1256, Vol **56**, pp 1-37, (1908).

[11] F. White 'Viscous Fluid Flow', McGraw-Hill, pp 261-267, (1974).

[12] H. W. Liepmann,'NACA Wartime', Report W107, ACR3H30, (1943).

[13] T. Sarpkaya, 'Vortex-Induced Oscillations, A Selective Review', J. of applied Mech., Vol **46**, pp. 241–257, (1979).

[14] A.S. Grove, F.H. Shair, E.E. Peterson and A. Acrivos 'The Steady Separated Flow Past Circular Cylinder at Large Reynolds Numbers', J. of Fluid Mech., **21** , pp. 737–760, (1965).

[15] A. Acrivos, L.G. Leal, D.D. Snowden and F. Pan, 'Further Experiments on Steady Separated Flow Past Bluff Objects', J. of Fluid Mech., **34**, pp. 25–48, (1968).

[16] N. P. Weatherill, M. J. Marchant, O. Hassan and D. L. Marcum, 'Grid Adaptation Using a Distribution of Source Applied to Inviscid compressible Flow Simulation', Int. J. of Num. Meth. in Fluids, Vol **9**, pp. 739-764, (1994).

[17] N. P. Weatherill, L. J. Johnston, A. J. peace, J. A. Shaw , 'Development of a Method for the Solution of the Reynolds-Averaged Navier-Stokes Equations on Triangular Grids', ARA, Report 70, (1986)

[18] J. J. Thibert, M. Granjaques, and L. Ohman. 'NACA0012 Airfoil Pressure Distribution Measurements'. Technical Report AGARD AR–138, AGARD Advisory Report, (1979)

[19] R. C. Swanson and E. Turkel, 'A Multigrid Time-stepping Scheme for Navier-Stokes Equations', AIAA Paper 85-0035, (1985).

[20] L. Martinelli, A Jameson, F. Grasso, 'A Multi-grid Method for the Navier-Stokes Equations', AIAA paper 86-0208 (1986).

[21] D. J. Mavriplis and A. Jameson, 'Multi-grid Solution of the Navier-Stokes Equations on Triangular Meshes', AIAA Journal, **28**, No. 8, (1990).

[22] R. C. Swanson and E. Turkel, 'Artificial Dissipation and Central Difference Scheme for the Euler and Navier-Stokes Equations', AIAA Paper 87-1107, (1987).

[23] L. A. Sykes, 'Development of A Two-dimensional Navier-Stokes Algorithm for Unstructured Triangular Grids', ARA Report 80, April 1990.

9

Simulation of Three-Dimensional Flows

9.1 Introduction

In this chapter, some test cases are selected to study the ability of the proposed finite volume algorithm to simulate both inviscid and viscous flows on unstructured tetrahedral meshes. Some available analytical and independent numerical solutions or other experimental results are used to assess the accuracy of the algorithm.

In Sections 9.2, the accurate implementation of the convective and artificial dissipation terms are discussed. Sections 9.3 presents the results of the viscous flow computation using the Galerkin finite element formulation for computation of the viscous stresses. Section 9.4 presents results of the flow simulation for complex problems in the field of engineering. Note that in order to to save computational CPU time, the procedure of implicit residual smoothing is not applied for the three-dimensional tests cases which include more number of nodes than two-dimensional cases.

9.2 Inviscid Flow Simulations

In this section the results of the implementation of the convective and artificial dissipation terms are discussed. For inviscid flow simulations, the accuracy and convergence of the algorithm are dependent on the implementation of appropriate artificial dissipation term, which is necessary due to application of central difference space discretisation. Since the artificial dissipation term is computed along the edges, the irregularity of the number of edges connected to each node in three-dimensional meshes causes some difficulty. Hence, it is necessary to normalise the artificial dissipation term.

The inviscid flow around a sphere is considered as a test case. The main motivation for selecting this case comes from the fully three-dimensional behaviour of flow around a sphere. Hence, this test case can be used to verify the performance and accuracy of the three-dimensional inviscid flow solver. An analytical incompressible solution for this example has been developed using a complex potential formulation [1]. Here both regular and irregular meshes have been used to assess the performance of the inviscid part of the flow solver on different types of unstructured tetrahedral meshes. By taking advantage of the symmetry of the flow, both meshes are generated only around a quarter of a complete sphere.

A specific regular mesh generator is developed for this test case using spherical system of coordinates. The grid points for each quarter of the sphere are constructed by putting a number of coupled Khiiam (Pascal) triangulation patches on top of each other and mapping into the spherical coordinates. Connection of the corresponding nodes of the patches in the radial direction forms a set of prismatic volumes, each of which consists of three tetrahedra. These elements can be arranged in a symmetrical manner in neighbouring prisms. Regular refinements were imposed near the stagnation points on the sphere surface. The size of the tetrahedra were increased using an expansion parameter in all directions from the two poles of the sphere. Figure (9.2.a) shows the view of the regular mesh of a quarter of sphere used in the computation. The volume mesh is surrounded by one solid wall, one far field and two symmetry surfaces. The outer boundary was placed at a distance of 12 radii away from the centre of cylinder. The grid is composed of 60840 nodes, 337896 cells, and 684608 edges. There are 17632 grid points distributed on the wall surface. Some details of the mesh are shown in Figure (9.2.b).

An irregular mesh was also created using the Delaunay triangulation method [2], which is embedded inside the PSUE package [3]. The mesh was adapted to the flow condition by refining regions in front and behind the sphere surface. The manual adaptivity procedure was developed by local refinement of the mesh. This refinement is enforced by positioning source points within the domain close to the stagnation points. By defining the grid resolution in certain regions around the source points grid clustering can be controlled. Partial views of the irregular mesh are presented in Figure (9.2.c) and Figure (9.2.d). The mesh contains 37542 nodes, 196463 elements and 402425 faces.

The computation of this test case started from the free-stream values as $(u_1)_i^0 = 1.0$, $(u_2)_i^0 = 0.0$, $(u_3)_i^0 = 0.0$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$. For both types of meshes, the computations took less than 10000 iterations to converge and since the procedure of residual smoothing is not applied, approximately four orders of magnitude in the reduction of the pressure residuals was reached. The typical convergence behaviour of the pressure and velocity are shown in Figure (9.2.e). Typical computed velocity vectors are presented in Figure (9.2.f). The contours of pressure on the sphere and symmetry surfaces of regular and irregular unstructured meshes are shown in Figures (9.2.g) and (9.2.h), respectively.

The pressure coefficient, C_p, on the sphere surface for the regular mesh and the analytical solution [1] are shown in Figure (9.2.i). For the irregular mesh the computed results of C_p on the sphere surface are compared with the analytical solution in Figure (9.2.j). Note that, the analytical solution of C_p is plotted only for the sphere wall, whilst the computed C_p is plotted for all the nodes on the sphere and intersection of the two symmetry walls. In general, for both regular and irregular meshes, the agreement between the computed C_p and analytical solution [1] are quite acceptable.

Regular Mesh around a Sphere

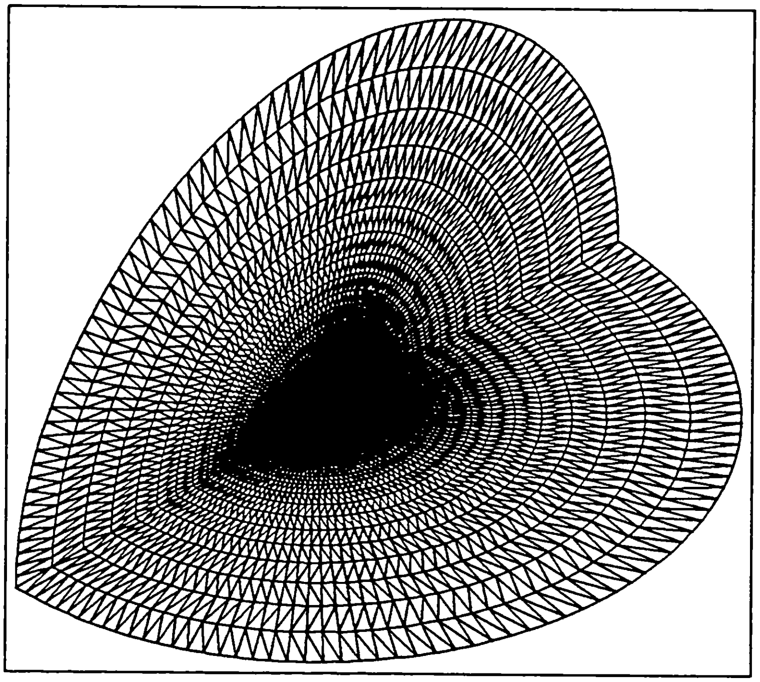

Figure 9.2.a *Regular mesh around a quarter of sphere.*

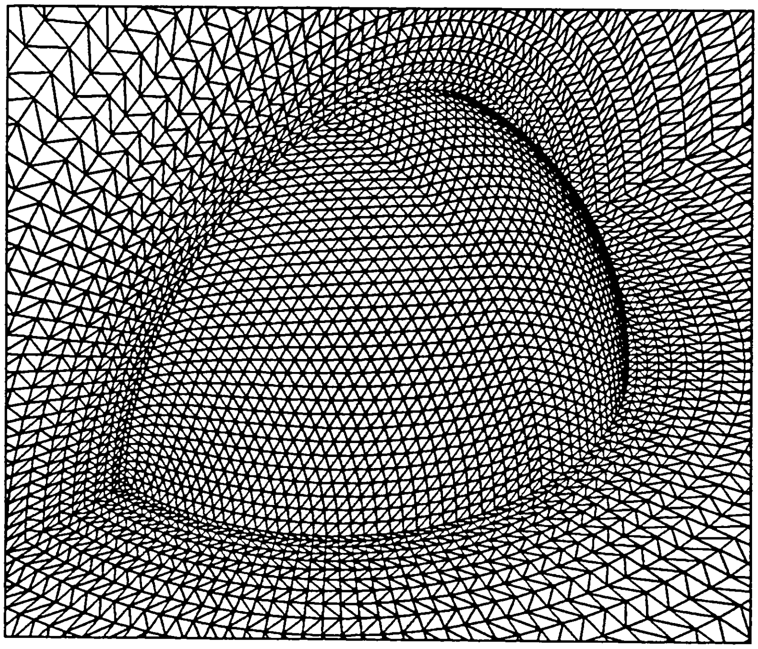

Figure 9.2.b *Regular mesh around a quarter of a sphere.*

Irregular Mesh around a Sphere

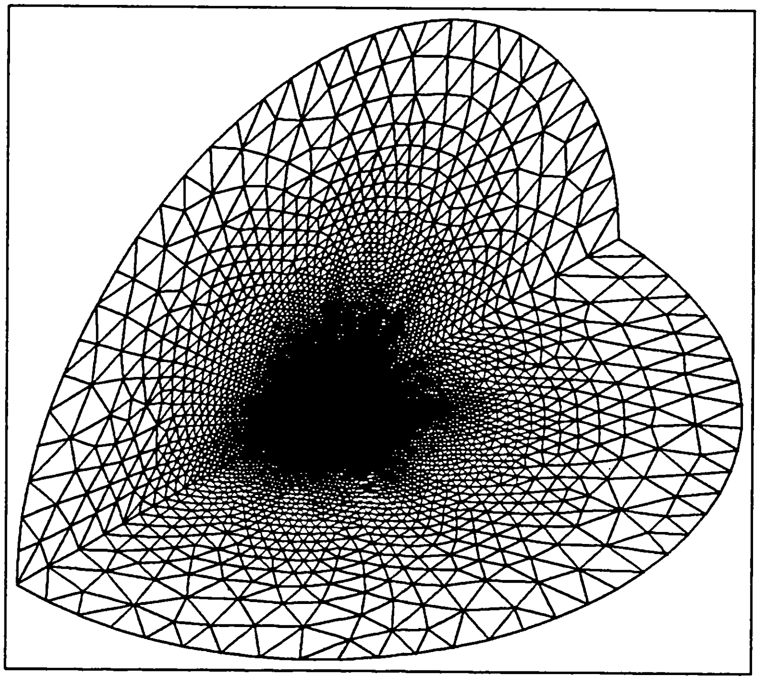

Figure 9.2.c *Irregular mesh around a quarter sphere.*

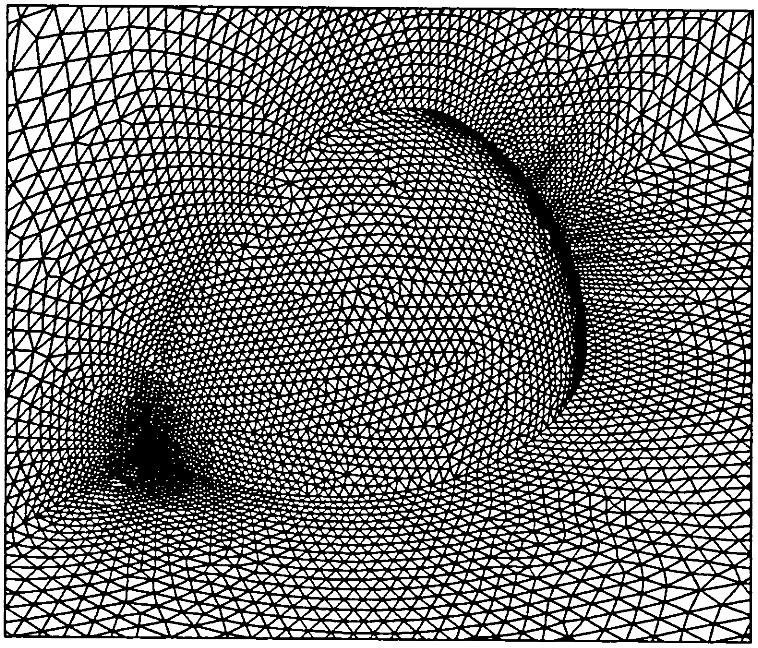

Figure 9.2.d *Irregular mesh around a quarter sphere.*

Inviscid Flow around a Sphere

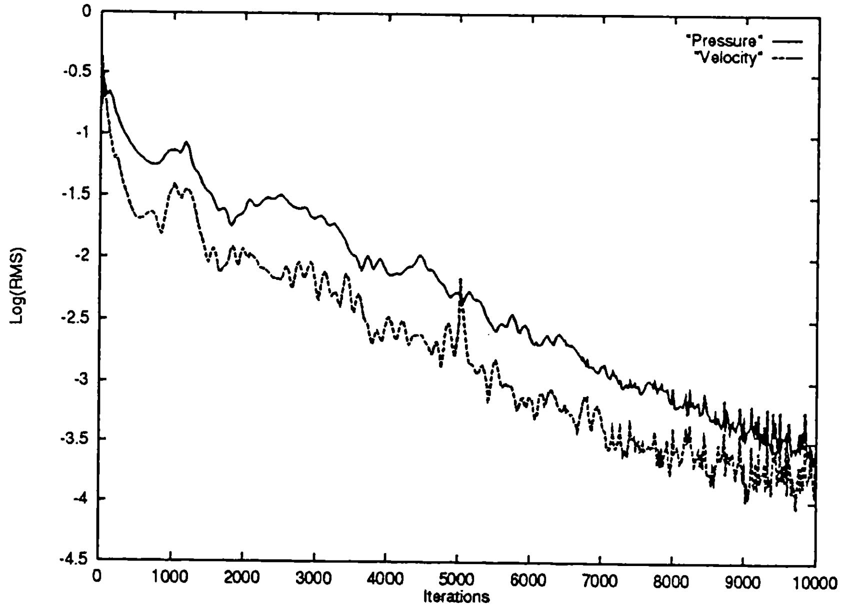

Figure 9.2.e *Typical convergence history of pressure and velocity residuals (computed on the irregular mesh).*

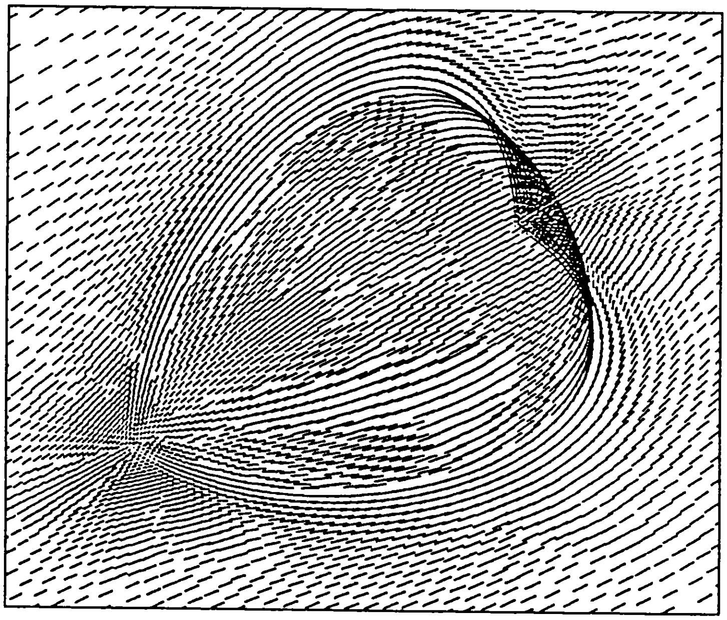

Figure 9.2.f *Typical computed velocity vectors on the sphere and symmetric surface.*

Inviscid Flow around a Sphere

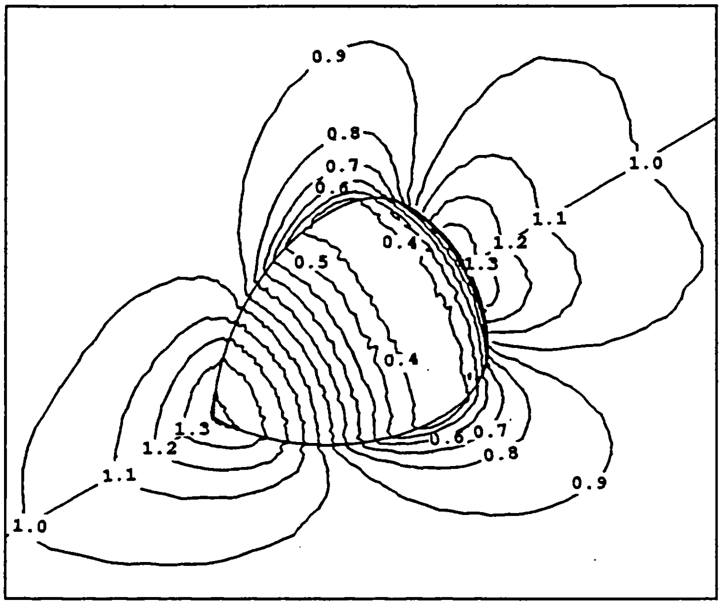

Figure 9.2.g *Typical computed pressure contours on the sphere and symmetric surface for the regular mesh (Min.=0.335 , Max.=1.57 , Intervals=0.1).*

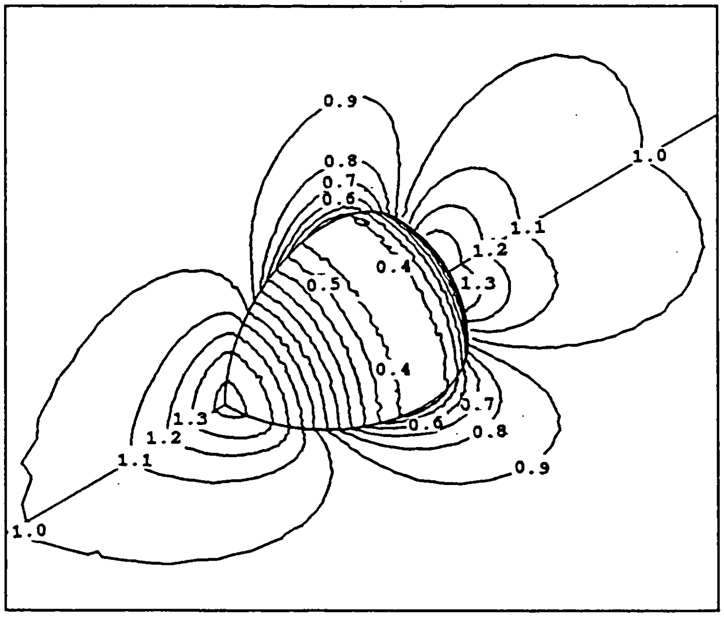

Figure 9.2.h *Typical computed pressure contours on the sphere and symmetric surface for the irregular mesh (Min.=0.334 , Max.=1.54 , Intervals=0.1).*

Inviscid Flow around a Sphere

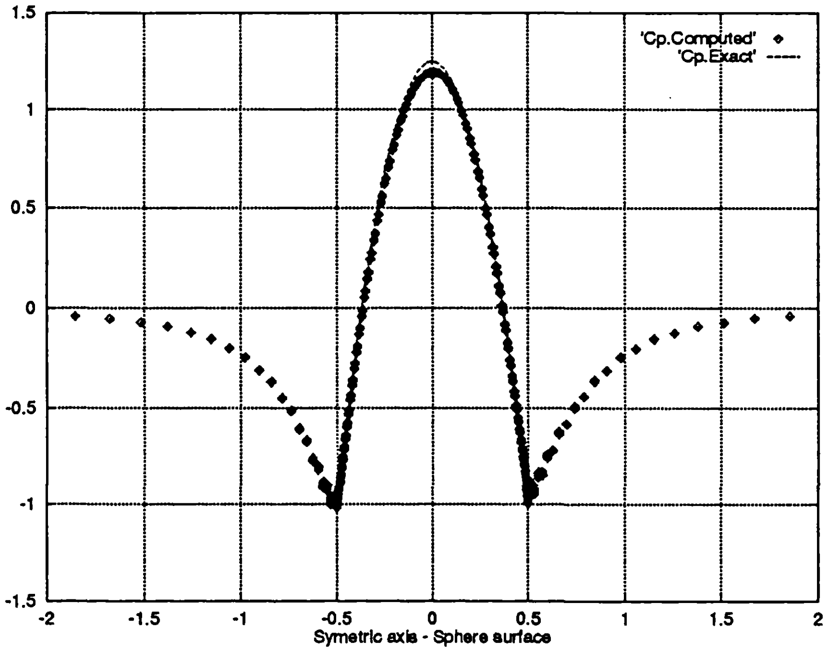

Figure 9.2.l *Computed pressure Coefficient for the regular mesh compared with the analytical solution [1].*

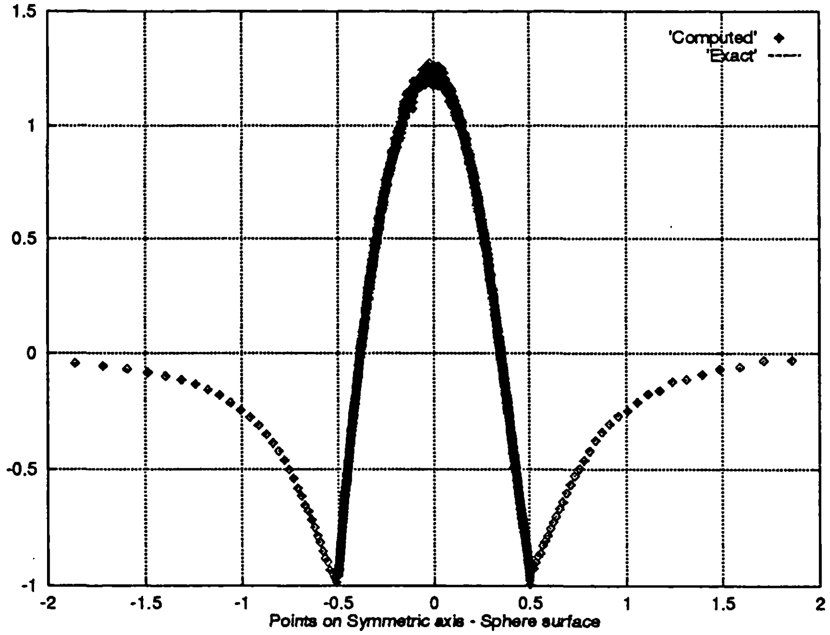

Figure 9.2.j *Computed pressure Coefficient for the irregular mesh compared with the analytical solution [1].*

9.3 Viscous Flow Simulations

In this section the implementation of the viscous term of the three-dimensional flow solver is examined. For the evaluation of viscous stresses, the application of the Galerkin finite element formulation is studied. Flow over a flat plate, which is widely used for validation of incompressible laminar flow solvers, is chosen as a test case to investigate the accuracy of the computation of the viscous term. The comparison of the computed results with the results from Blasius similarity solution are used to assess the accuracy of the velocity components inside the boundary layer region. The mathematical similarity solution for a flat plate parallel to the free-stream is found by Blasius [5]. The relations, which were described in Section (8.3.1), present unique relations between components of velocity u_1 and u_2 and the dimensionless parameter $y = x_2 f(x_1)$. These parameters are valid for all the Reynolds numbers in the limit of incompressible flow. Here, u_1 and u_2 are the components of the velovity in x_1 and x_2 direction.

For this test case, a regular mesh is generated by stacking two-dimensional regular grid points and then five tetrahedra are considered inside every hexagonal grid spacing. The width and height of the flow domain are considered equal to 50% and 100% of the plate length, respectively. In this test case, the free stream velocity components are imposed at the inflow boundary and free stream pressure at the outflow boundary. The remaining variables at inflow and outflow boundaries are obtained by extrapolation from inside the domain. Although the computation are performed without imposing any special far field boundary condition treatment, i.e. enforcing velocity profile at the outflow boundary, symmetric walls are imposed at the two edges of the plate in the flow direction. In order to reduce the effect of free-stream values, the symmetry boundaries are considered equal to 50% and 100% of the plate length at inflow and outflow, respectively. The general and partial view of the applied mesh are shown in Figures (9.3.a) and (9.3.b). The grid is composed of 18480 points, and 97977 edges. The aspect ratio of the tetrahedra located at the leading edge of the plate are equal to one with an expansion of the mesh in both directions leading to triangles with aspect ratio of 100 at the end of the plate.

The computation was performed on the grid, starting from free stream values. The free-stream values are considered as $(u_1)_i^0 = 1.0$, $(u_2)_i^0 = 0.0$, $(u_3)_i^0 = 0.0$ and $(p)_i^0 = 1.0$ $(i = 1, N_{node})$. Although the residual smoothing is not applied, after less than 5000 iterations, the residuals based upon pressure and velocity components reduced by 4.5 orders of magnitude. The convergence behaviour of the pressure and velocity residuals are shown in Figure (9.3.c). In order to present the independency of the results from the choice of Reynolds number, the results of $u = u_1/|\mathrm{U}|$ at a certain position of the plate, $x_1 = 0.40$, are compared in Figure (9.3.d) with Blasius similarity solution for different Reynolds numbers of 1000, 2500, 5000 and 10000. Note that all the computations, for different Reynolds numbers are done on the same mesh. Typical velocity and pressure contours for $\Re = 1000$, computed by the present algorithm, are shown in Figures (9.3.e) and (9.3.f), respectively.

The accuracy of the computations can be assessed by comparing the computed results with the results of Blasius [5]. For comparison of the computed results at different position of the plate length, a constant Reynolds number of 5000 is considered. The accuracy of the horizontal and vertical components of velocity u_1 and u_2 at different points normal to the plate are studied using the parameters u and v defined by equations (8.3). The comparison of these parameters with the Blasius solution are presented in Figures (9.3.g) and (9.3.h). The horizontal components of velocity u_1 is plotted for different x_3 values in Figure (9.3.i). In general there is agreement between the computed results and the Blasius similarity solutions.

The Flat Plate Viscous Mesh

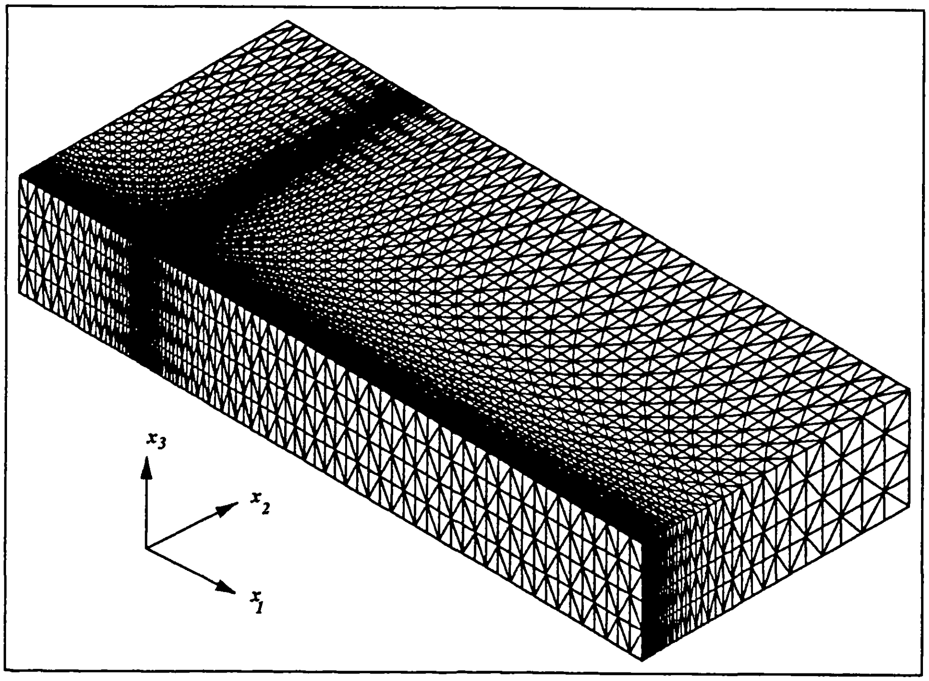

Figure 9.3.a *General view of the regular 3D mesh for a Flat Plate.*

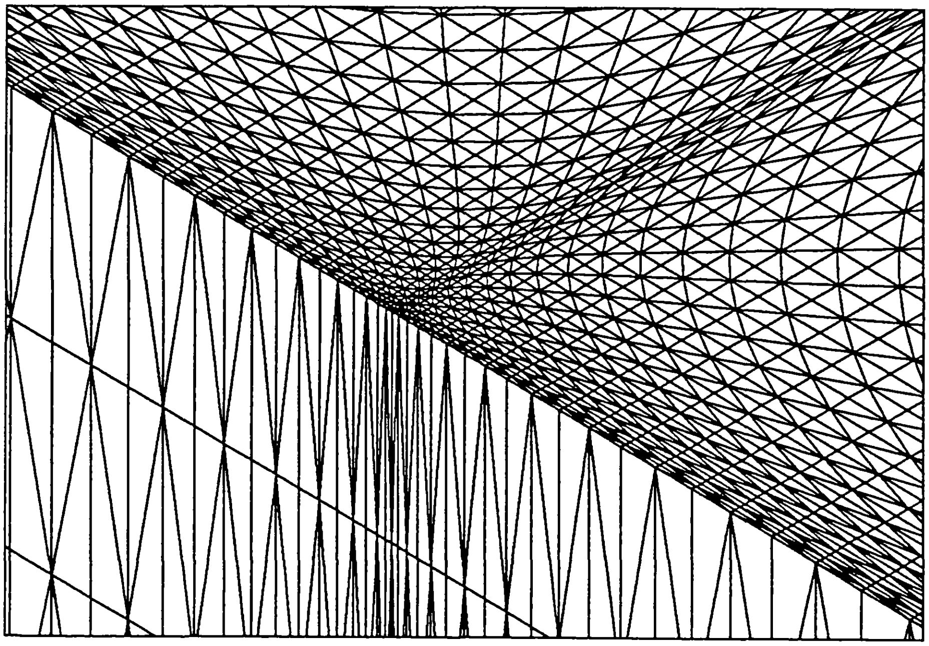

Figure 9.3.b *Partial view of the mesh close to the Flat Plate edge.*

Viscous Flow over a Flat Plate

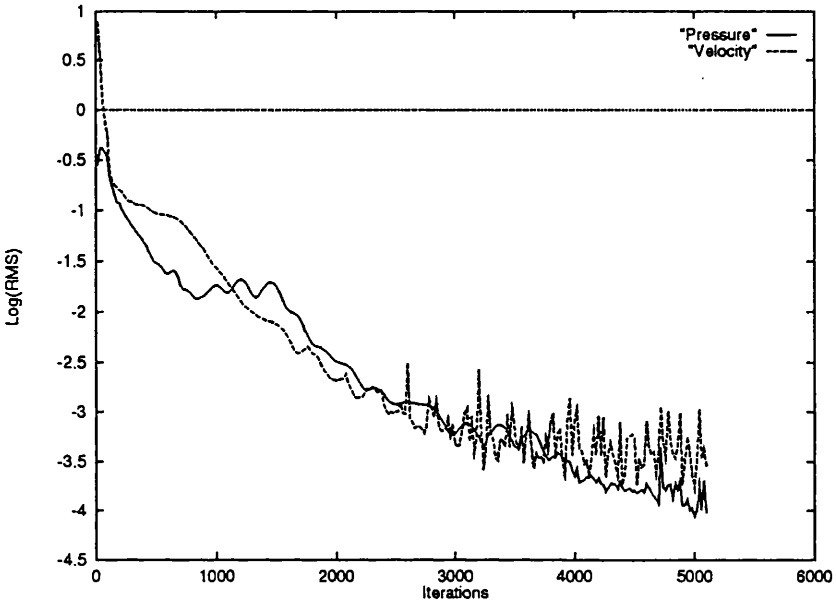

Figure 9.3.1.c *The convergence behaviour of the pressure and velocity residuals.*

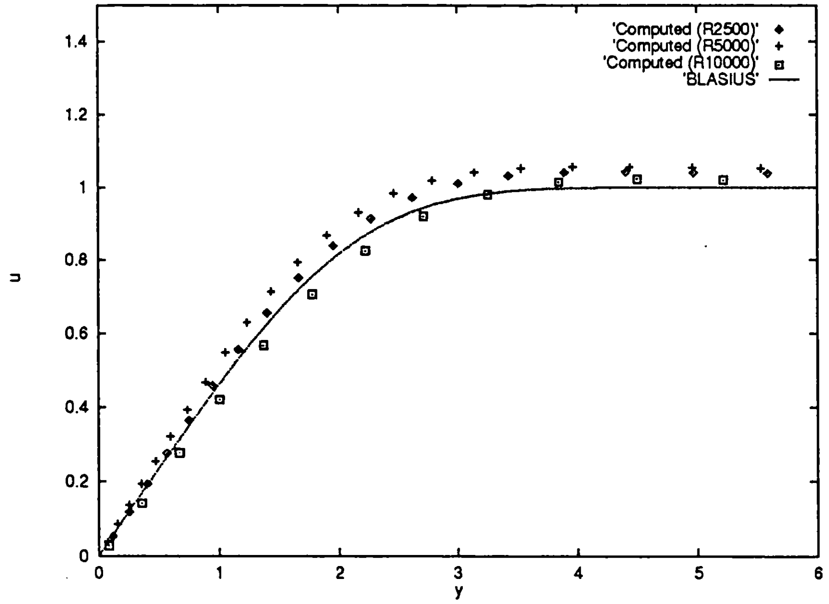

Figure 9.3.d *Comparison of computed* $u = u_1/|\mathrm{U}|$, *with Blasius similarity solution at* $x_1 = 0.40$ *for different* $\Re$ $(y = x_2\sqrt{\frac{\rho|\mathrm{U}|}{2\mu x_1}})$.

Flow over a Flat Plate, $\Re = 1000$

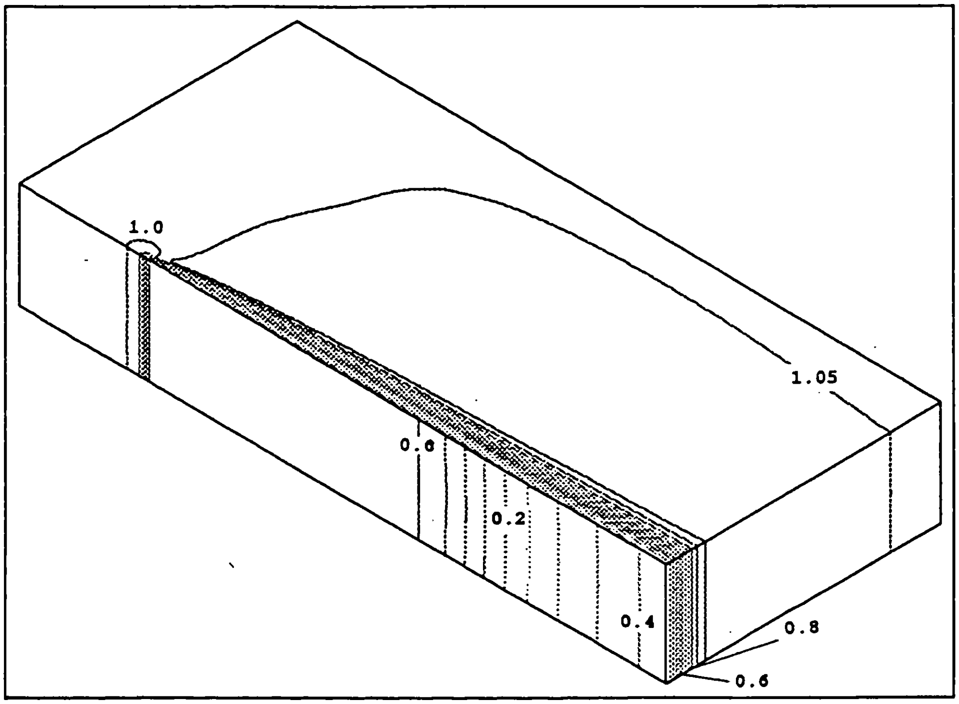

Figure 9.3.e *General view of the velocity contours (Min. = 0.0, Max. = 1.1, Intervals = .05)*

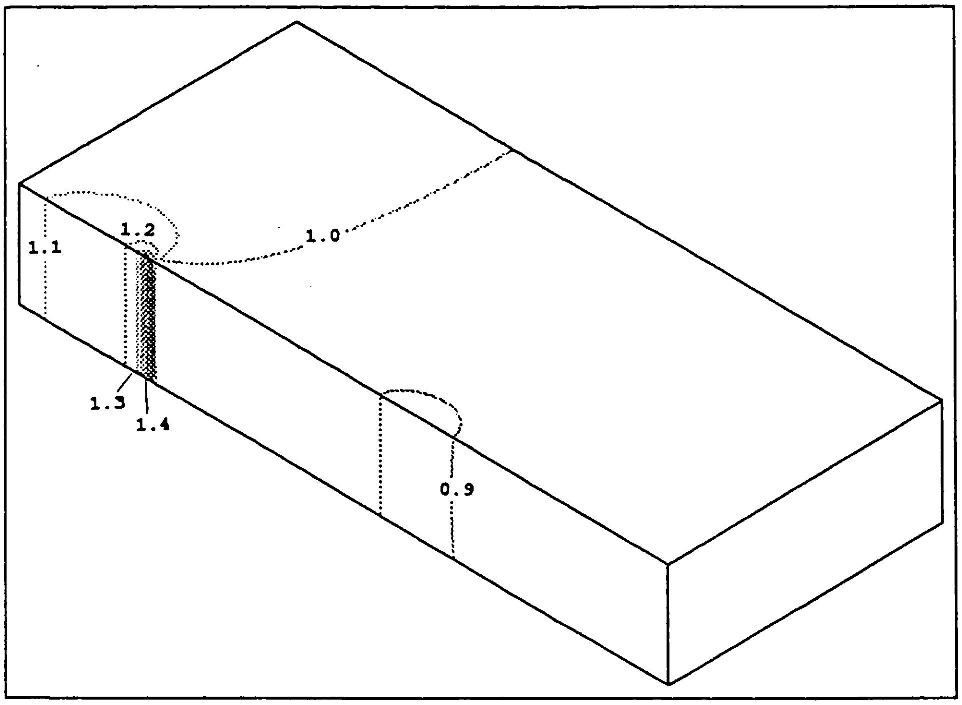

Figure 9.3.f *General view of the pressure contours (Min. = 0.9, Max. = 1.41, Intervals = 0.1).*

Viscous Flow over a Flat Plate, $\Re = 5000$

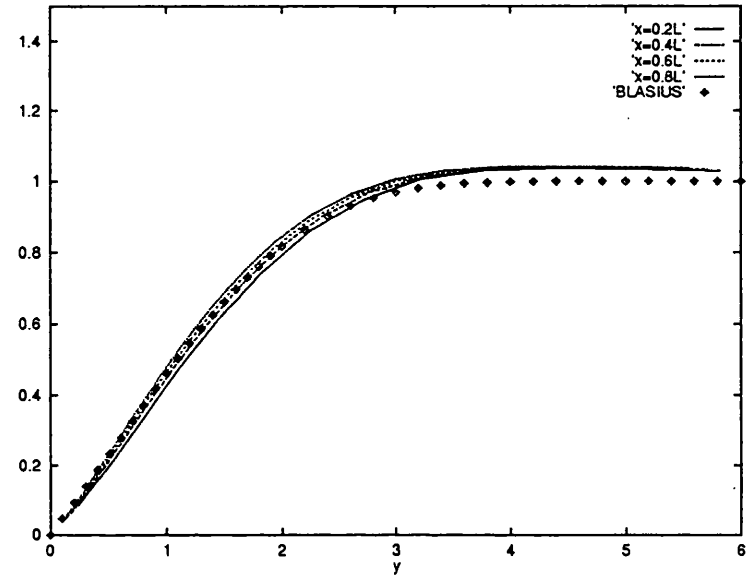

Figure 9.3.g *Comparison of $u = u_1/|\mathrm{U}|$ normal to the plate with Blasius similarity solution for different positions on the plate ($y = x_2\sqrt{\rho|\mathrm{U}|/2\mu x_1}$).*

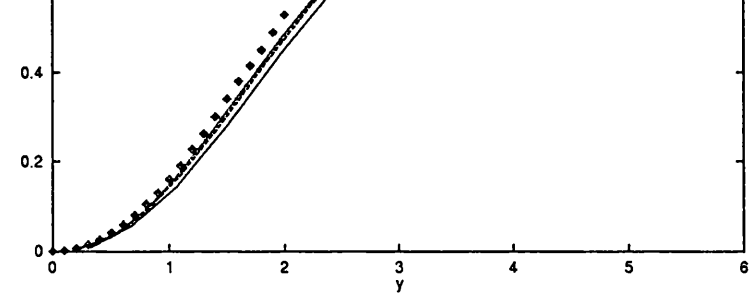

Figure 9.3.h *Comparison of $v = u_2/|\mathrm{U}|\sqrt{\Re_{x_1}}$ normal to the plate with Blasius similarity solution for different positions on the plate ($y = x_2\sqrt{\rho|\mathrm{U}|/2\mu x_1}$).*

Viscous Flow over a Flat Plate, $\Re = 5000$

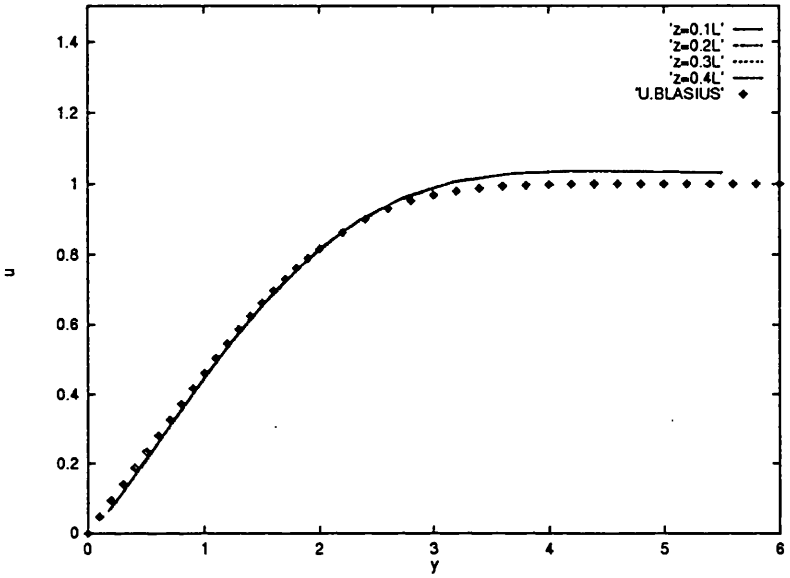

Figure 9.3.i *Comparison of computed $u = u_1/|\mathrm{U}|$, with Blasius similarity solution at $x_1 = 0.40$ for different $\Re$ ($y = x_2\sqrt{\rho|\mathrm{U}|/2\mu x_1}$).*

9.4 Application to Complex Problems

In order to assess the ability of the solver to deal with geometrically complex problems, some test cases were selected which relate to problems associated with Civil Engineering. The first case is the external wind flow over a tall building. The pressure distribution on the building surface can be used for environmental and structural design purposes. The next test case is chosen as the flow over a single cooling tower. can be achieved. The last test case is the flow over three cooling towers. The results from this flow simulation can be used to evaluate the interference effects between towers. The results can be applied in both fields of the Structural and Environmental Engineering. For each case, the flow field is presented in terms of the pressure and velocity contours as well as velocity vectors and stream lines. Some of the results are compared with experimental data.

9.4.1 Wind Flow over a Tall Building

This test case involves the environmental and structural problem of the simulation of steady inviscid wind flow around buildings. Wind is one of the main climate factors influencing the design of buildings. A tall building, by its nature, is an obstacle to the wind and deflects air along various streams and channels, which can produce unpleasant conditions for pedestrians and may even lead to structural damage. Experiments show that the buildings, which are substantially taller than their surroundings, present a large obstruction to the wind, deflecting it both horizontally and vertically. One of the effects of this deflection is to create vortices, and increased wind speeds near ground level in the area around the building. In these regions some problems may be presented by the locally accelerated flow.

The model which is chosen here consists of two separate buildings. For this case some experimental results are available [7]. In order to simulate the experimental model, the second building is considered to be the taller one. The height and width, as well as the distance between the two buildings, are considered to be equal to unity. All other dimensions are assumed to be equal to one quarter of the height of taller building. The layout of the simulated model is shown in Figure (9.4.1.a).

An irregular unstructured mesh was created using Delaunay triangulation method [2], which is embedded inside the PSUE package [3]. The zone of refinement behind the main building was adapted to the vortex wake region behind the taller building. The adaptivity procedure was developed by refining the mesh and imposing point and line sources for local refinement within the horseshoe shape wake region behind the tall building and close to the walls [4]. The volume mesh is surrounded by ten solid walls, one far field and one symmetry boundary. The mesh contains 39797 nodes, 220878 elements and 447863 faces. The partial view of the adapted mesh which is used for the simulation is presented in Figure (9.4.1.b).

For this case, the computations took about 7000 explicit iterations to converge and approximately 4.5 orders of magnitude in the reduction of the pressure residuals was achieved. The velocity distribution around such a building and the wind pressures on the walls and roof are all closely linked. The pressure load on the surfaces of the building can be used for structural design purposes. The computed pressure coefficient Figure (9.4.1.c) and the results from experimental model Figure (9.4.1.d) show that the distribution of the computed pressure coefficient on the wind ward building wall is in quantitative agreement with experimental data.

The experimental data indicates that the flow close to the windward face of the tall building radiates from a central region at the height of some three quarters of the building height. The computed velocity vectors on the surfaces of the main building are shown in Figure (9.4.1.e) and can be compared with the related experimental results, which are created using smoke flow, in Figure (9.4.1.f).

The vertical vortices between and behind the buildings were made visible by injecting smoke in the experimental test. The computational results at an imaginary intersecting surface parallel to the flow direction passing through buildings and the experimental results can be compared in Figures (9.4.1.g) and (9.4.1.h).

Below the level of the flow stagnation point, at the front face of the building, a considerable quantity of air flows downwards and outwards to be concentrated near ground level at the wind ward corners. These accelerated air-streams pass around the corners to form two jets of air, which stretch downwind for a considerable distance, forming horizontal and vertical vortices [7]. The computed velocity vectors at ground surface are shown in Figure (9.4.1.i) and indicate a horizontal horseshoe vortex at ground level behind the main building. This horse-show vortex was made visible using a film of pigmented oil in the experiment Figure (9.4.1.j). Experimental measurements show that the distance between two buildings has a great effect on the horseshoe vortex patterns. The slight difference in location of this vortex between the computed and the experimental data is due to viscous effects and the distance between the buildings.

As can be seen the flow solver predicts flow behaviour which is in general qualitative agreement with experimental data.

Wind Flow over a Tall Building

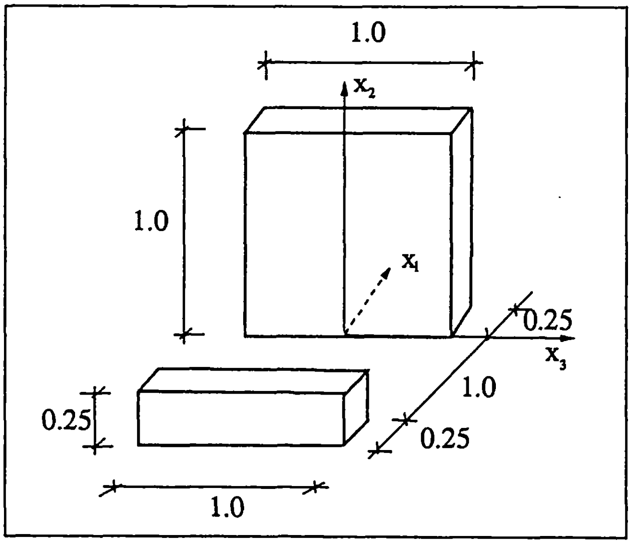

Figure 9.4.1.a *The layout of the simulated model of buildings [7]*

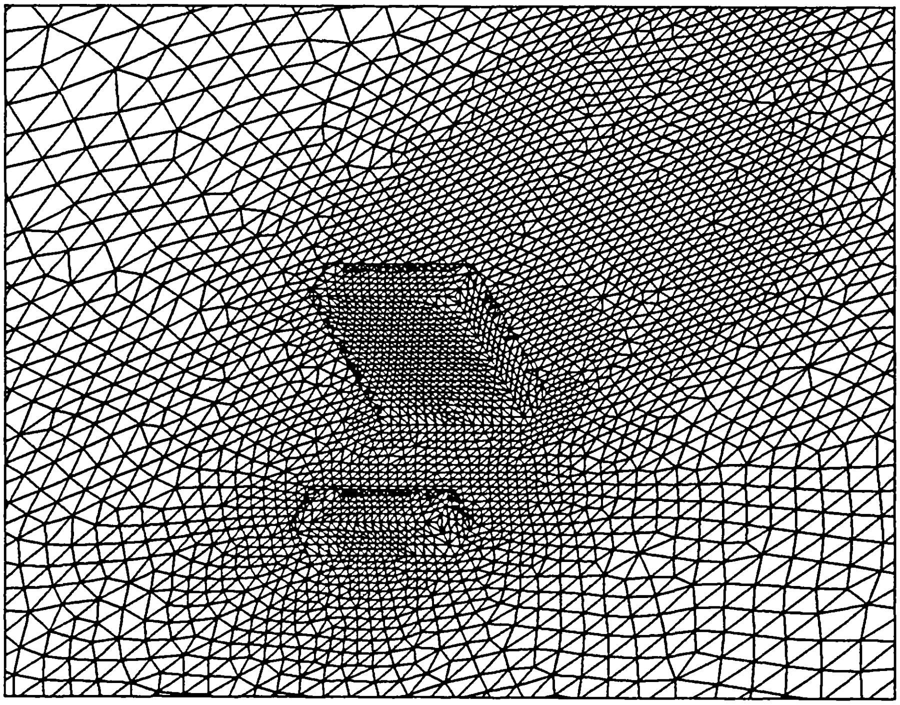

Figure 9.4.1.b *Partial view of triangular surface mesh for wind flow over tall buildings*

Wind Flow over a Tall Building

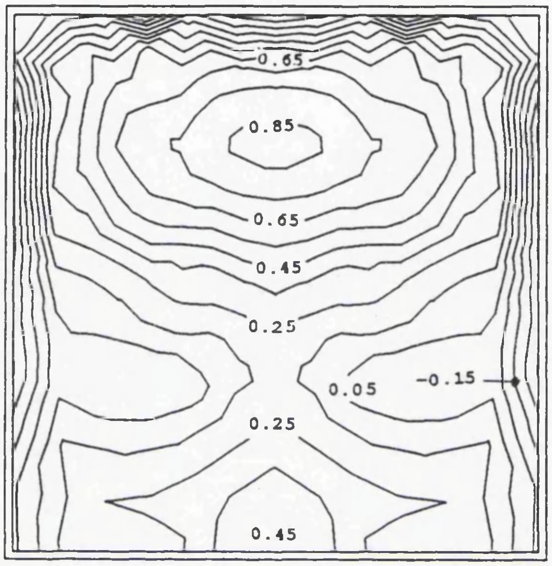

Figure 9.4.1.c *Typical computed pressure coefficient C_p on building surfaces (Min.=-0.15, Max.=0.85, Intervals =0.1).*

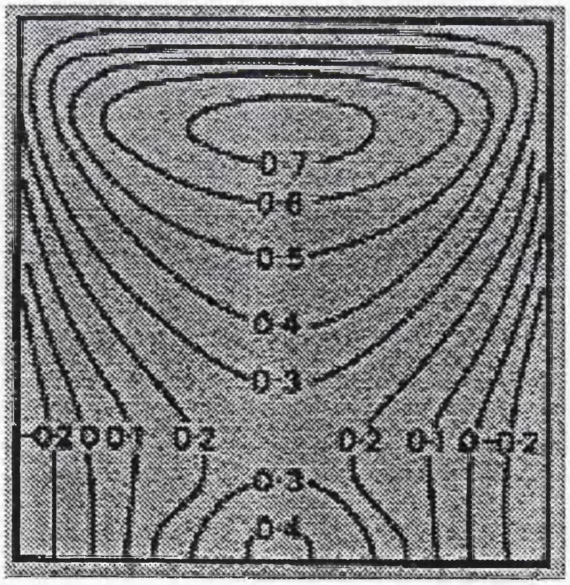

Figure 9.4.1.d *Experimental results for the pressure coefficient C_p on the main windward wall*

Wind Flow over a Tall Building

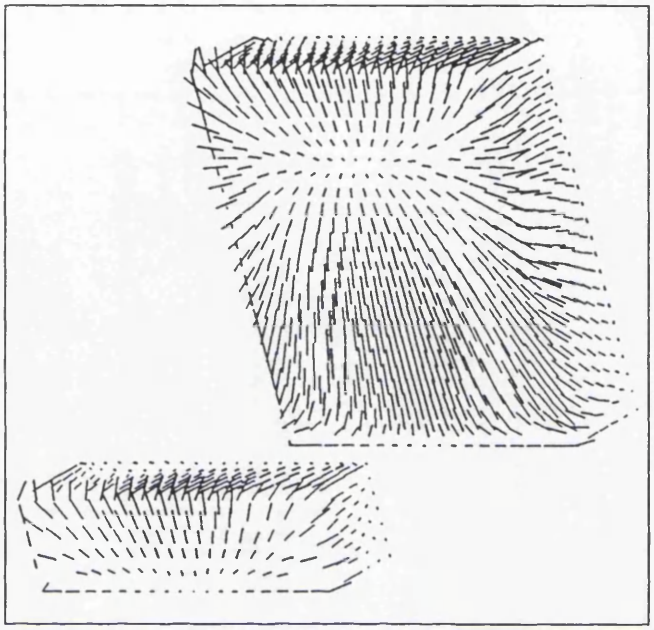

Figure 9.4.1.e *Typical computed pressure contours on the building and ground surfaces*

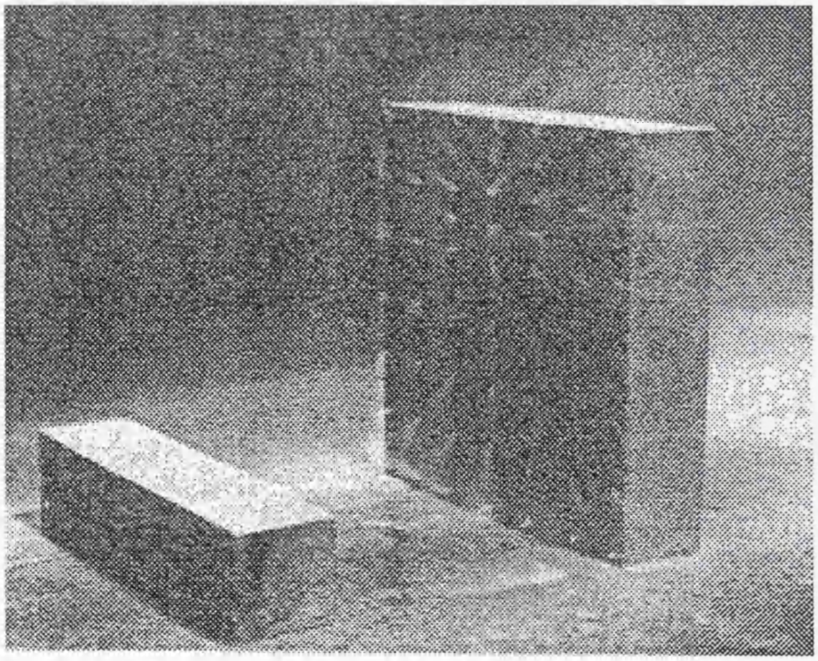

Figure 9.4.1.f *Experimental results for the pressure coefficient on the main building surfaces*

Wind Flow over a Tall Building

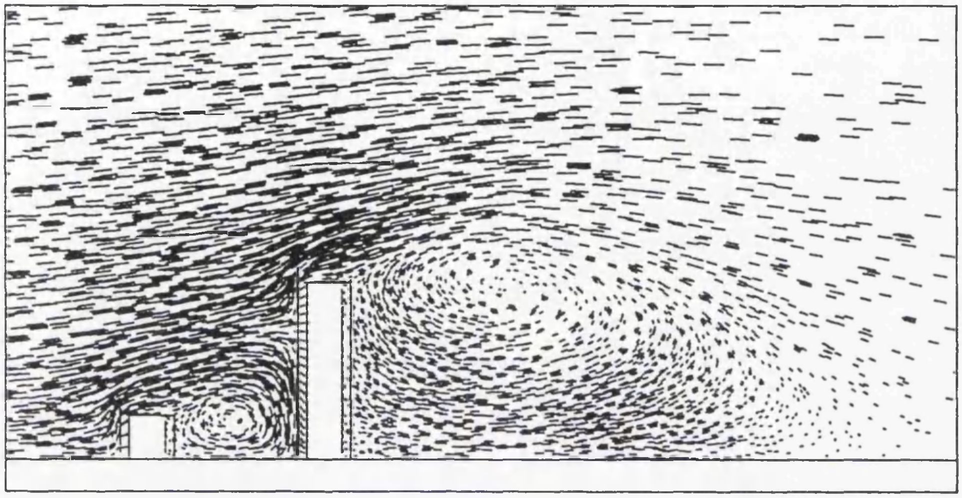

Figure 9.4.1.g *Typical velocity vectors passing the tall building on a section of the domain parallel to the flow direction*

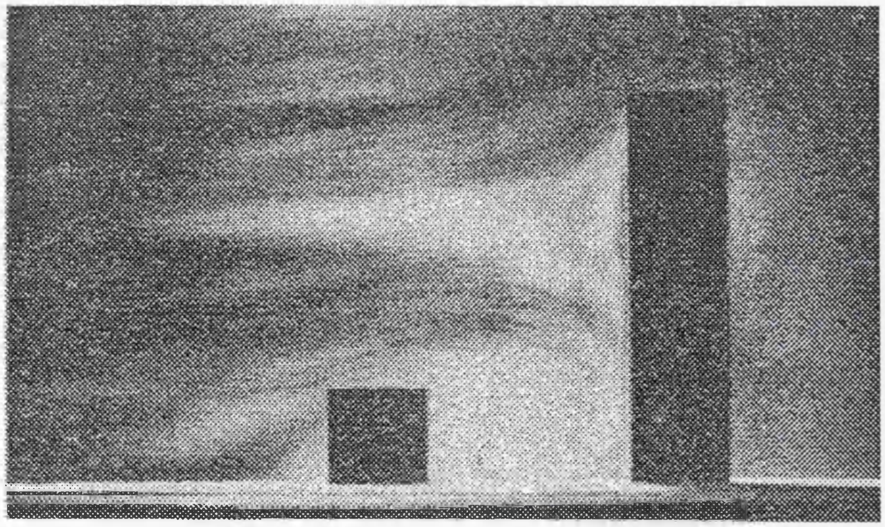

Figure 9.4.1.h *Experimental results showing the flow over tall building*

Wind Flow over a Tall Building

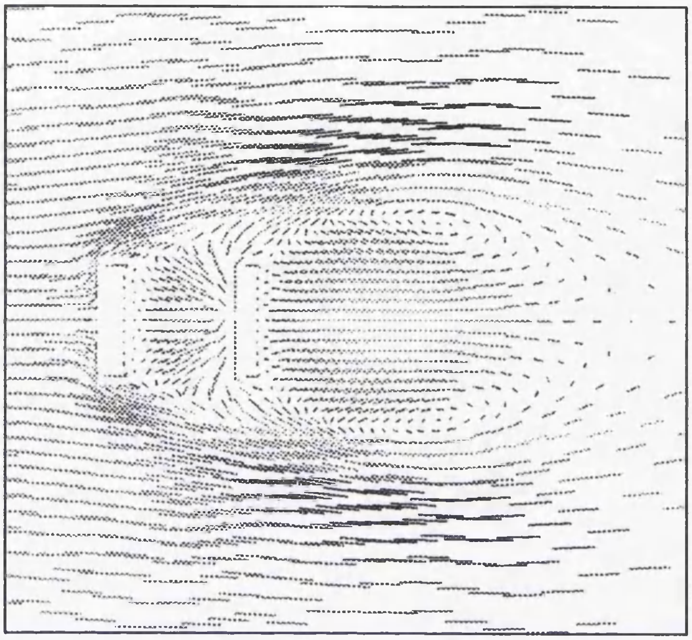

Figure 9.4.1.i *Typical velocity field around the buildings at ground level*

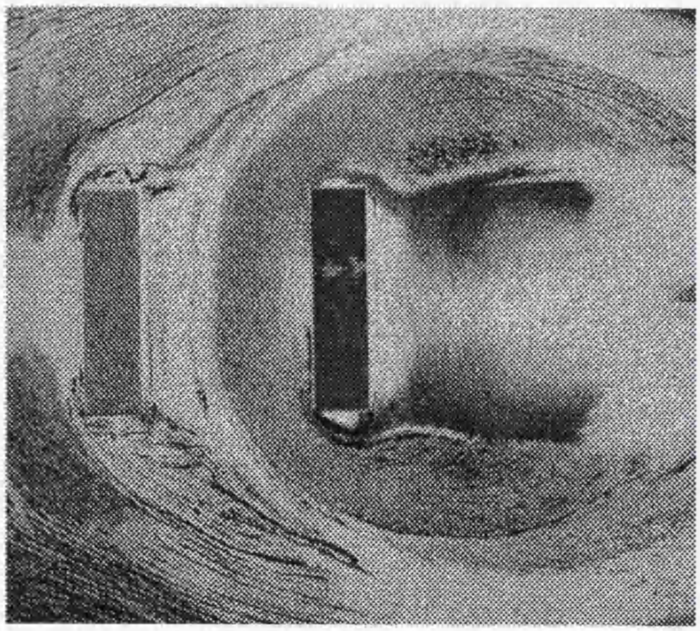

Figure 9.4.1.j *The flow pattern at ground level obtained by experiment*

9.4.2 Wind Flow over a Single Cooling Tower

The next case is the application of the three-dimensional inviscid flow solver in the field of structural engineering. This case is the flow simulation over a single cooling tower. The collapse of three cooling towers of Ferry Bridge in North England due to wind effects absorbed the attention of many researchers in the field of fluid dynamics and structural engineering. This interesting test case involves different fluid phenomena, i.e. different direction of the flow patterns due to the slopes in different directions. Here, a single cooling tower, which contains no vertical curvature, is chosen. The details of dimensions of the tower which are normalised by the base diameter are given by Figure (9.4.2.a).

An unstructured mesh was created by the Delaunay triangulation utilising the PSUE package. The surface meshes comprise of a far field boundary, three solid wall boundaries and three symmetry boundaries. The far field boundary is located 12.5 times the base diameter of the tower from its centre. The grid consists of 367179 faces and 179987 tetrahedra cells. The number of points used is 33582 with 14410 points located on the boundaries. A general view of the grid used for this calculation is shown in Figure (9.4.2.b). For all the results presented in this section the direction of wind flow is from the left to the right side.

The computations take 10000 explicit iterations until a 3.5 order of magnitude reduction for pressure and velocity residuals is achieved. Figure (9.4.2.c) presents the top view of the velocity vectors on different surfaces and Figure (9.4.2.d) shows the velocity vectors on the tower surface and an imaginary plane passing through the centre line of the structure. Figures (9.4.2.e) and (9.4.2.f) present general views of the pressure and velocity contours on the tower and ground surface. The results of computation show the application of the inviscid flow solver for evaluating the pressure distribution on the structure.

Wind Flow over a Single Cooling Tower

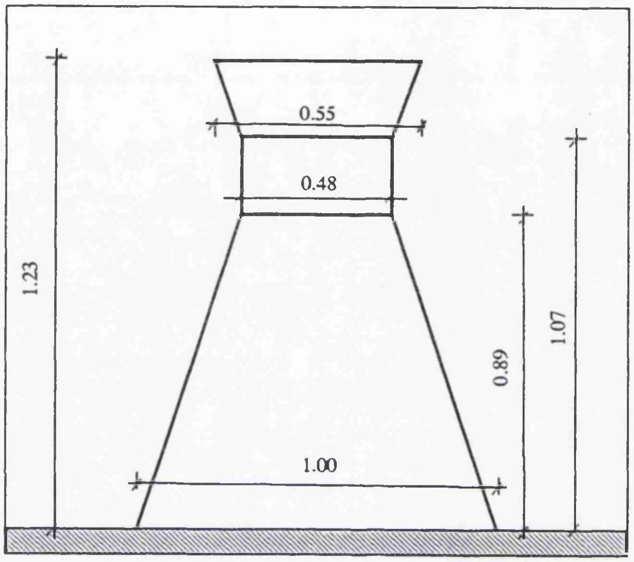

Figure 9.4.2.a *General dimensions considered for the tower.*

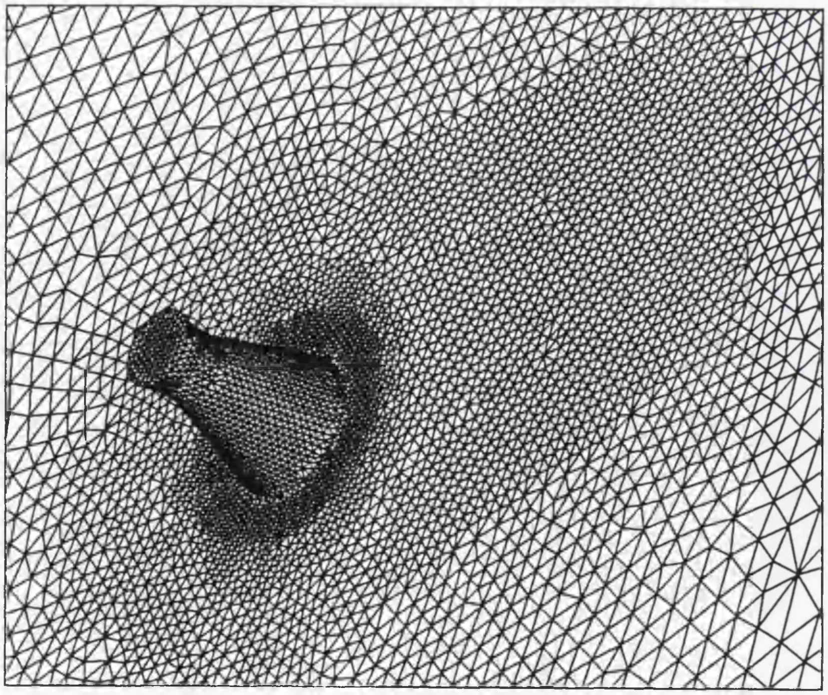

Figure 9.4.2.b *Unstructured triangular mesh for cooling tower.*

Wind Flow over a Single Cooling Tower

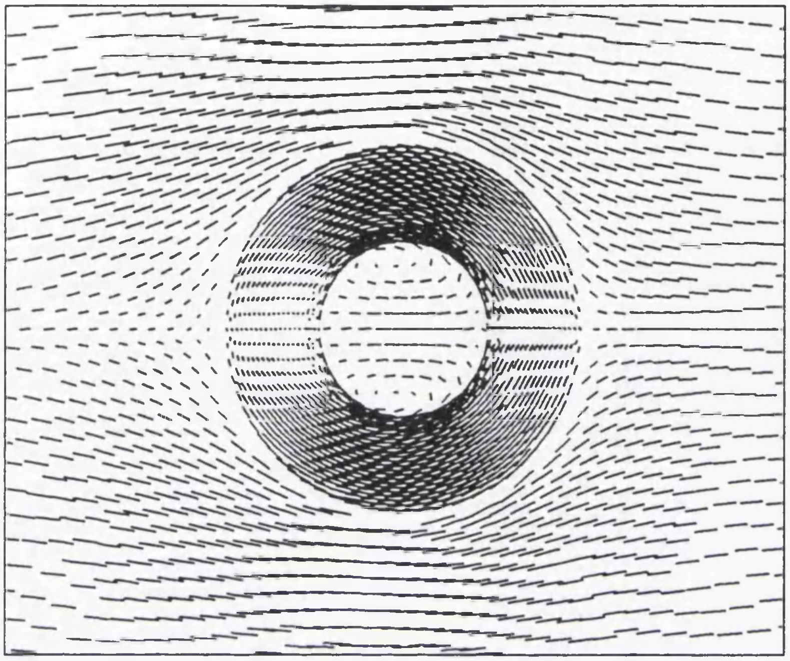

Figure 9.4.2.c *Velocity vectors resulting from inviscid computations.*

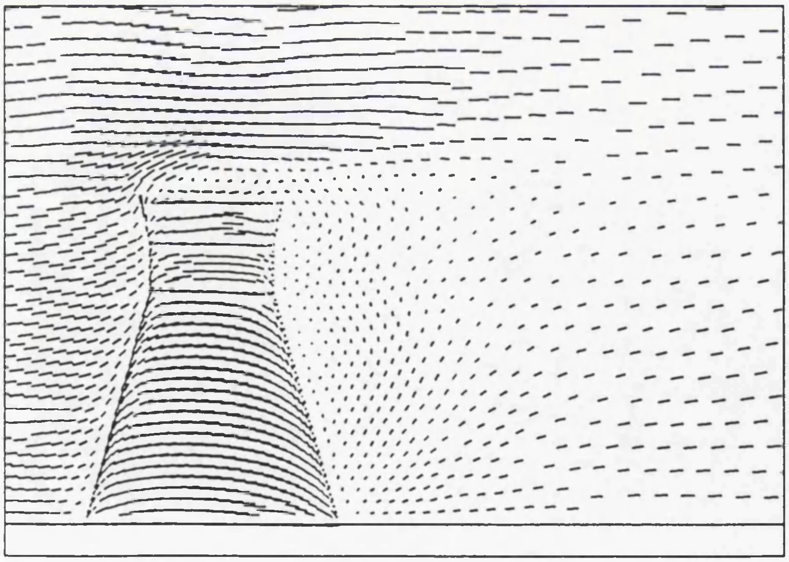

Figure 9.4.2.d *Velocity vectors resulting from inviscid computations.*

Wind Flow over a Single Cooling Tower

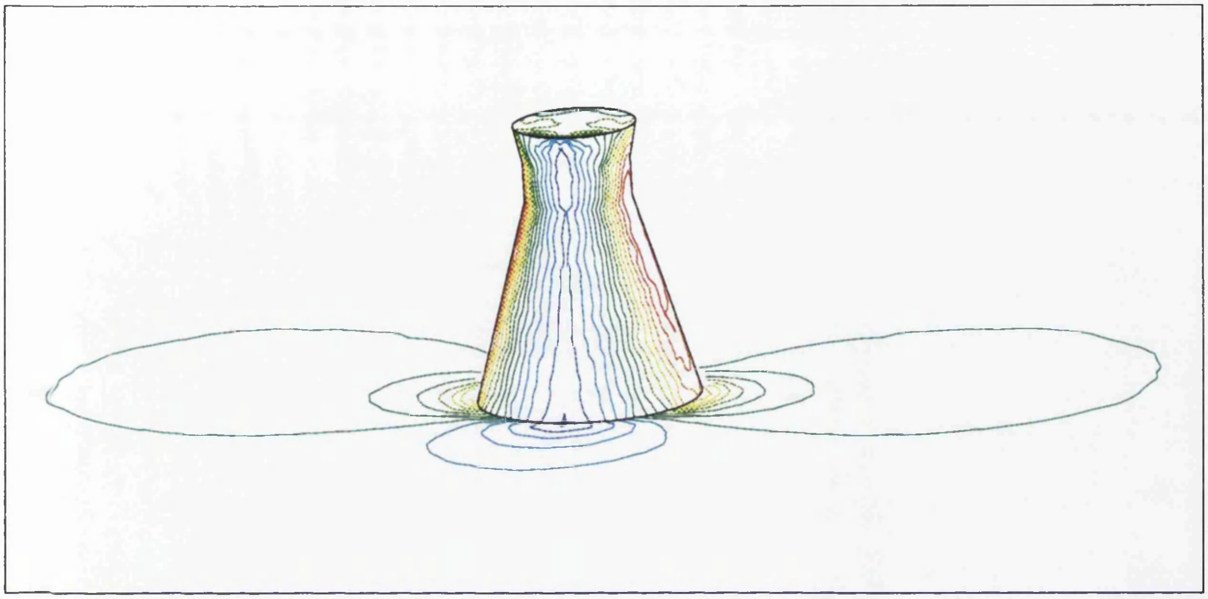

Figure 9.4.2.e *Wind ward view of pressure contours on the ground and tower surface (Min.=0.08, Max.=1.58, Intervals =0.1).*

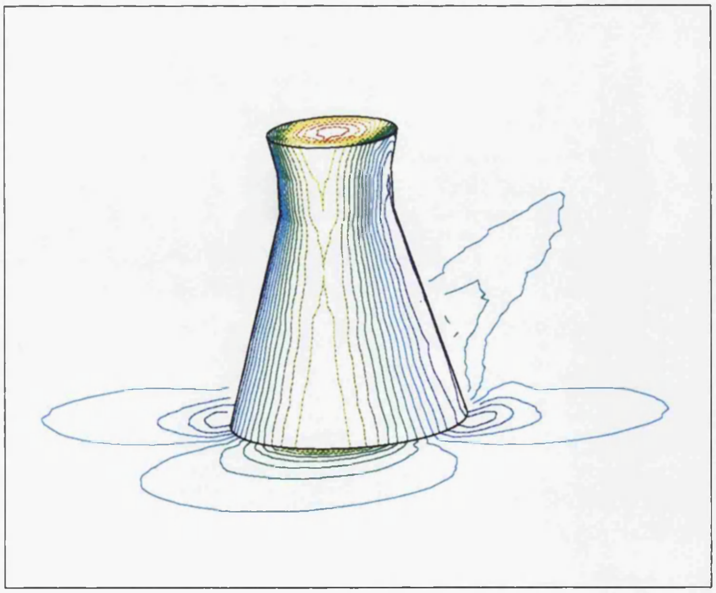

Figure 9.4.2.f *Wind ward view of velocity contours on the ground and tower surface (Min.=0.0, Max.=1.63, Intervals =0.1).*

9.4.3 Wind Flow Over Three Cooling Towers

The last case to be considered is the flow simulation over three cooling towers, which is the application of the three-dimensional inviscid flow solver in the field of both environmental and structural engineerings. The investigations of the effect of grouping cooling towers is of practical interest. This is an interesting test case, which involves several fluid flow phenomena, including different directions of the flow due to the grouping the structures, curvatures of the towers, expansion and contraction of flow and the existence of several stagnation points.

The grid applied was fully unstructured and was created by Delaunay triangulation utilising PSUE package [3]. The computational domain consists of 283957 faces and 137824 tetrahedra cells. The number of points is 25764 with 16618 points located on the boundaries. A general view of the grid used for this calculation is shown in Figures (9.4.3.a).

After 7500 explicit iterations a reduction 3.5 orders of magnitude is observed for the pressure and velocity residuals. Figure (9.4.3.b) presents the top view of the velocity vectors on different surfaces of the mesh. Figures (9.4.3.c) and (9.4.3.d) present wind ward views of typical pressure and velocity fields on the ground and on the surfaces of the towers, respectively. Figures (9.4.3.e) and (9.4.3.f) present two different view of typical three-dimensional stream lines around the set of towers. The visualisation package ENSIGHT was used to produce these plots. The results from the solver can be used for obtaining the design parameters such as forces on the the towers due to different arrangements.

Wind Flow over Cooling Towers

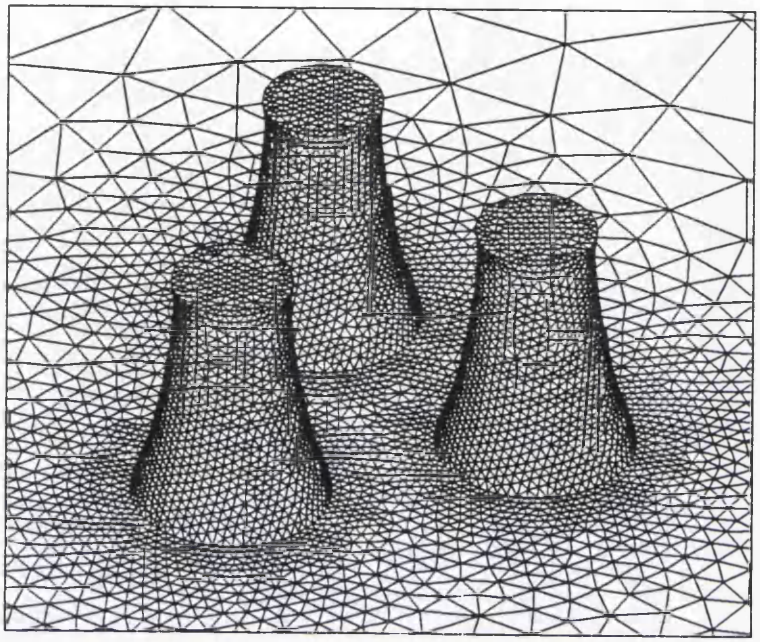

Figure 9.4.3.a *Unstructured mesh for three cooling towers.*

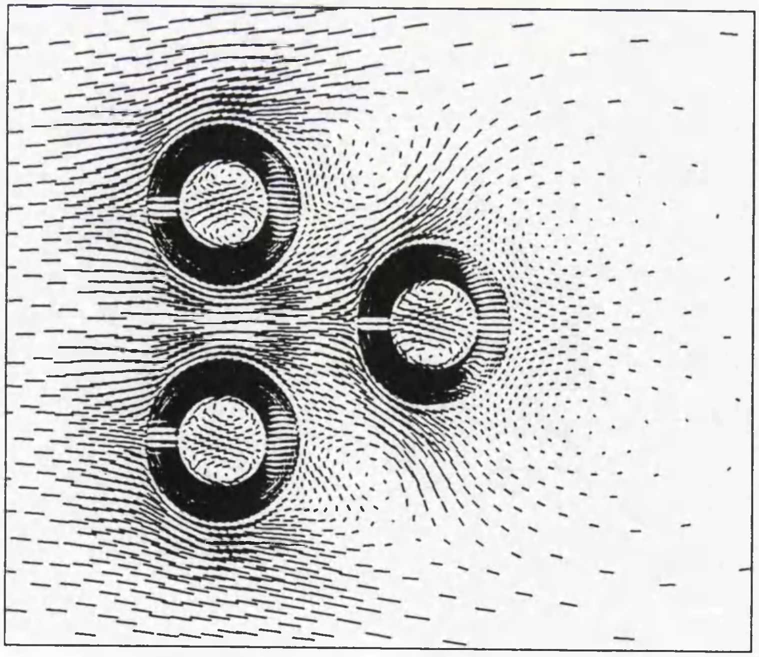

Figure 9.4.3.b *Velocity vectors on the towers and ground surfaces (Wind flow direction is from the left to the right).*

Wind Flow over Cooling Towers

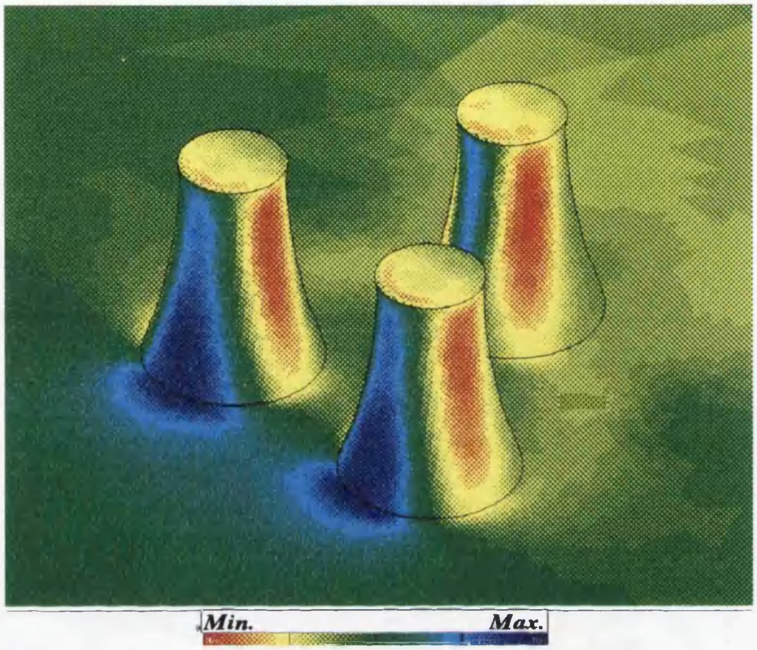

Figure 9.4.3.c *Wind ward view of typical pressure field on the ground and on the sufaces of the towers (Min.=0.335 , Max.=1.66).*

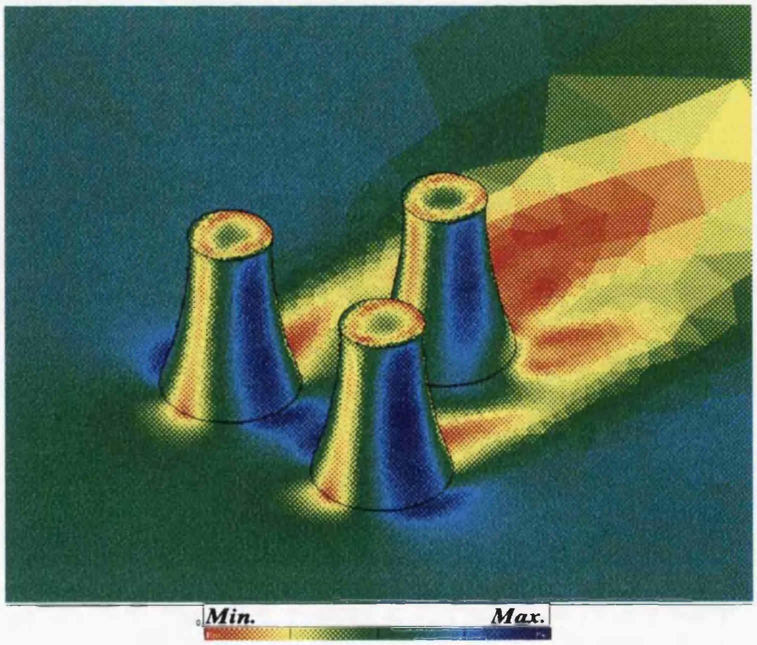

Figure 9.4.3.d *Wind ward view of typical velocity field on the ground and on the sufaces of the towers (Min.=0.014 , Max.=2.59).*

Wind Flow over Cooling Towers

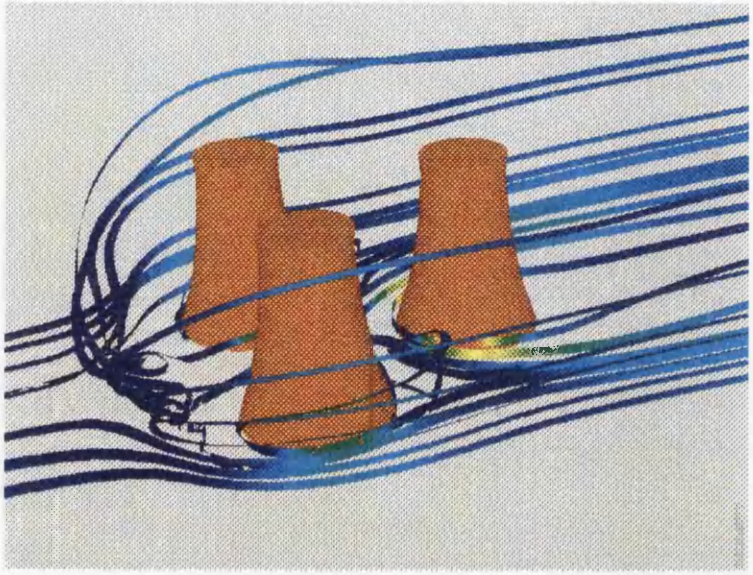

Figure 9.4.3.e *General view of stream lines*

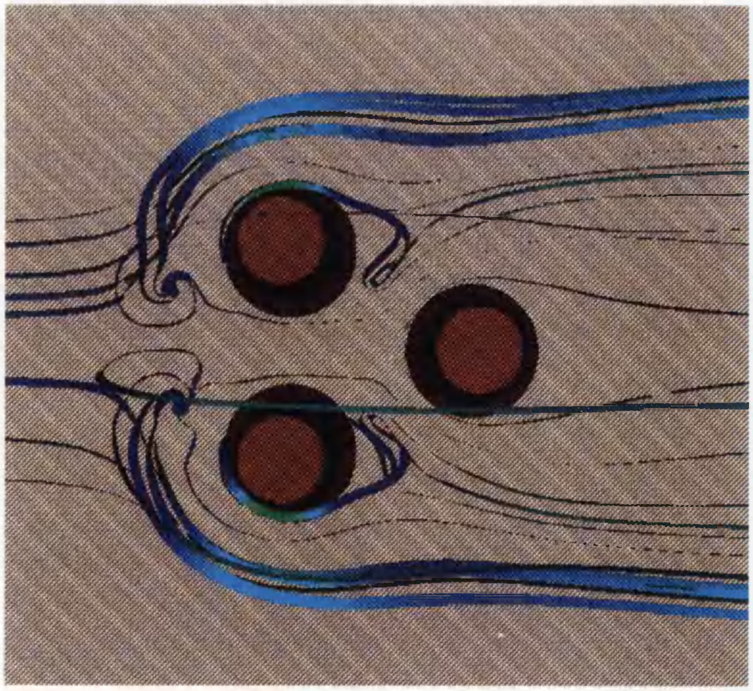

Figure 9.4.3.f *Top view of stream lines*

9.5 Summary

In this chapter, appropriate test cases were selected to study the performance of the algorithm for both inviscid and viscous three-dimensional flow simulations. Acceptable results are achieved from the inviscid flow computations by applying the normalisation procedure for the edge-based artificial dissipation term. Application of the Galerkin finite element formulation for the computation of viscous terms produces acceptable results on unstructured tetrahedral meshes.

Some application were employed to evaluate the ability of the flow solver to predict some demanding flows in the field of engineering. From the results it can be concluded that the flow solver is able to predict the pressure and velocity fields over complex configurations, which is applicable for the structural design and environmental planning. It seems that the algorithm can be taken as a corner-stone for further development to accurately solve realistic engineering problems.

References

[1] L. M. Milne-Thomson, 'Theoretical Hydrodynamics', The Macmillan Press, 5th Edition, pp.158 (1979).

[2] N. P. Weatherill, 'Delaunay Triangulation in Computational Fluid Dynamics', Compute. Math. Appl., No. **24**, (5-6), PP. 129–150, (1992).

[3] N. P. Weatherill, 'A Review of Mesh Generation' Special Lecture, Advances in Finite Element Technology, Civil-Comp Press. Edinburgh. pp 1-10, (1996).

[4] N. P. Weatherill, M. J. Marchant, O. Hassan and D. L. Marcum, 'Grid Adaptation Using a Distribution of Source Applied to Inviscid Compressible Flow Simulation', Int, J. of Num. Meth. in Fluids, Vol **9**, pp. 739-764, (1994).

[5] H. Blasius, NACA Technical Memorandum 1256, Vol **56**, pp 1-37, (1908).

[6] F. White 'Viscous Fluid Flow', McGraw-Hill, pp 261-267, (1974).

[7] A. D. Penwarden and A. F. E. Wise, 'Wind Environment Around Buildings', Building Research Establishment Report, BRE publications (1975).

[8] A. Jameson, T. J. Baker and N. P. Weatherill, 'Calculation of Inviscid Transonic Flow over a Complete Aircraft', AIAA Paper 86-0103, (1986).

[9] N. P. Weatherill, L. J. Johnston, A. J. Peace and J. A. Shaw, 'Development of a Method for the Solution of The Reynolds-Averaged Navier-Stokes Equations on Triangular Grids', ARA Report 70, (1986).

[10] L. Martinelli, A Jameson, F. Grasso, 'A Multi-grid Method for the Navier-Stokes Equations', AIAA paper 86-0208 (1986).

10

Concluding Remarks

10.1 Conclusions

A contribution in the area of computational fluid dynamics has been made to the development of an incompressible flow solver capable of simulating two and three-dimensional flows. The incompressible Navier-Stokes equations has been modified using the Artificial Compressibility approach. A side-based algorithm was adopted based upon a cell-vertex finite volume scheme. The algorithm was designed to work on unstructured meshes consisting of triangular and tetrahedral cells in two and three dimensions, respectively. The development was achieved in two stages. At the first stage, a code has been developed to solve inviscid flow problems, and then, it has been extended for computing laminar viscous flows. In the second stage, the solution algorithm was extended to three dimensions in order to simulate flow problems on unstructured tetrahedral meshes. Various flow problems were used to verify the accuracy and performance of the two and three-dimensional flow solvers. In this chapter, after a review of the important points in development of the algorithm, a brief conclusion is made about the choice of parameters which are used in the present work. Finally, suggestions about further work can be found in the last section. study.

10.2 Development of the Flow Solver

Compressible flow solvers are able to simulate flows with a Mach number above approximately 0.2. Extensive numerical practices show that, when the compressible set of the equations have been applied in the field of slightly compressible flow, the speed of sound becomes much larger than the speed of convection. Hence, the efficient numerical methods which have been established to solve the compressible Navier-Stokes equations do not work well for an incompressible flow problem. On the other hand, for a truly incompressible flow problem, because of the very small changes in the value of the specific mass, the time derivative of the specific mass in the continuity equation tends to zero. Therefore, a divergence free velocity constraint is imposed on the momentum equations. The absence of a time derivative of specific mass destroys the hyperbolicity of the set of Navier-Stokes equations, and hence, common numerical methods for solving the compressible flow equations fail to solve the incompressible flow problems. In order to overcome this problem, several methods have been introduced using derived and primitive variables.

If, the interest is only in steady flow, artificial compressibility approach is an efficient way around the difficulty. This approach removes the high speed of sound from the system of equations by prescribing a pseudo-temporal evaluation for the pressure which makes the set of equations hyperbolic. This modification casts the set of equations well conditioned for using numerical methods which have been established for the solution of the compressible Navier-Stokes equations. However, due to the introduction of a pseudo transient pressure term the time accuracy of the equations is destroyed and they can only be used when the steady state flow simulation is of interest. Note that, the pseudo temporal derivative of pressure tends to zero when the solution converges to steady state and then the set of modified equations satisfies the incompressible Navier-Stokes equations. This approach is attractive because of its efficiency due to direct coupling between the pressure and velocity fields as they are advanced in pseudo time. Hence, the proposed modified equations is capable of working only with the continuity and momentum equations and there is no need to apply an additional equation for the computation of the pressure field.

The numerical solution procedure was adopted using a cell-vertex scheme finite volume scheme which discretises the integral form of the governing equations. Application of such a scheme on unstructured triangular and tetrahedral meshes uses a side-based algorithm which provides the ability to deal with complex geometries in both two and three dimensions. In the present algorithm, the governing equations were discretised in space using the second order central differencing scheme. Thus, in order to control the instability of the scheme for both inviscid and viscous flow simulations, the application of the artificial dissipation was necessary.

In the present algorithm, the artificial dissipation term was evaluated by applying the Biharmonic operator which was computed using an edge-based algorithm. Hence, the size of the grid spacings and number of edges connected to the nodes directly affected the value of the artificial dissipation at different nodes. Although the worse effects of mesh spacing were improved by refining the mesh near the regions of high gradient flow variables and derefining in the zones of the low gradient flow variables, using unsmooth meshes, which includes sudden changes of mesh spacing degraded the effectiveness of the artificial dissipation. However, the mesh dependency of the artificial dissipation term was recovered using appropriate scaling factors. Hence, the artificial dissipation term was scaled by applying suitable scaling factors proportional to the maximum eigenvalue of the flux Jacobian matrices of the convective part of the governing equations. The effect of the variation of the number of edges connected to every node was resolved by normalising the artificial dissipation term. However, for two-dimensional cases the present simulator gave acceptable results using appropriate meshes, but for three-dimensional problems the application of normalisation proved demanding for the computation of the artificial dissipation. In the present work, proper normalisation techniques for unstructured triangular and tetrahedral meshes were adopted. In general, the accuracy of the results were improved by reducing the value of the coefficient of the artificial dissipation whilst the convergence behaviour of the pressure and velocity residuals showed some, although insignificant, deterioration. Although the tuning of this parameter was somewhat time consuming, it provided the ability to apply the minimum required numerical dissipation.

In the simulation of viscous problems, the accuracy of the results depended upon the applied formulation for computation of viscous term. For the present cell-vertex scheme three different formulations were studied. The first one was the cell-vertex finite volume formulation which produced accurate results on the two-dimensional regular meshes. An alternative formulation was developed by modifying the stencil for computing the viscous stresses. The next formulation was the Galerkin finite element formulation which produced accurate results for the triangular and tetrahedral meshes. Since this formulation was independent of the shape of the cells in the control volume, it was found suitable for simulating viscous flow problems on three-dimensional unstructured meshes. However, for three-dimensional cases, the last formulation was found more efficient than the modified cell-vertex finite volume formulation which required an additional set of points on both side of element faces.

In the present algorithm, the time stepping was used as relaxation iteration to the steady state. The multi-stage Runge-Kutta schemes which have originally been developed for the solution of the compressible Navier-Stokes equations were adopted for integrating the governing equations to steady state. Two schemes of three and five-stages are investigated. The three-stage Runge-Kutta schemes was chosen for the computation of all the case studies because of its efficient performance in terms of speed of convergence.

Acceptable results throughout all the domain were achieved when free stream values and extrapolated values of the flow variable were used together. In the present algorithm, the free stream values of velocity components and pressure were imposed as the value of flow variables on inflow and outflow boundaries, respectively. The values of the remaining flow variables on the far field boundaries are extrapolated from inside the domain. For external flow computations, far field boundaries were considered to be far enough from solid body.

The proposed algorithm has been studied for general unstructured grids which were used for flow simulation around arbitrary geometries. The computations were started from free-stream conditions without any pre-knowledge about the flow field and flow conditions. Adopting a side-based data structure, together with code vectorisation provided a good performance on high-speed vectorised computers, particularly in three dimensions. The developed algorithms were applied to standard test cases. The ability and accuracy of the flow solver on unstructured triangular and tetrahedral meshes were presented for different test cases. The developed algorithms have shown their ability to simulate inviscid and laminar viscous flows.

10.3 Choice of Parameters

The stability of the present incompressible flow solver was directly related to the parameter of the artificial compressibility β^2. The present artificial compressibility approach was adapted with the idea of local time stepping by considering the parameter β^2 proportional to the local velocity field and introducing an empirical coefficient C_{β^2}. The empirical coefficient taken as $C_{\beta^2} = 3$ presented stable convergence for both inviscid and viscous flow simulations. However, the minimum limit of $\beta^2 = 0.3$ provided a more efficient performance.

For the present central difference scheme, the minimum and maximum bound for the coefficient of the artificial dissipation dictated the accuracy and convergence of the computations. Therefore, in the present scheme, the constant of the artificial dissipation term ν_4 was found as the most sensitive parameter. Accurate tuning of this parameter ensured the accuracy and stability of the scheme. In general, acceptable results were obtained using the value of $\nu_4 = 3/256$, where the normalisation of the artificial dissipation term was not applied, whilst the value of this parameter was reduced to $\nu_4 = 1/256$ where the normalisation of the artificial dissipation was employed.

By improving the quality of the mesh, larger values of CFL number could be applied which defined the maximum time stepping limit and obviously affected the rate of convergence. In this work, the CFL number was chosen as $1.0 \leq CFL \leq 3.5$ for various test cases. At early stages of the computations, the rate of the convergence was affected by large disturbances in of the solution inside the flow domain, especially close to the solid bodies. Hence, the wall boundary conditions, the maximum time stepping limit, CFL number and the minimum required value of the constant of the artificial dissipations ν_4, were enforced through a number of iterations. In this way, accurate results were achieved without destroying the stability and convergence speed of the numerical solution.

Although the procedure of residual smoothing improved the rate of the convergence of the two-dimensional computations, it was not employed for the three-dimensional computations because of its expensive explicit nature. In the case of applying residual smoothing as a convergence accelerator, the scheme had better performance using the weighting parameter $\epsilon = 0.25$.

10.4 Recommendations for Further Work

The results of the research carried out in this thesis suggest that the proposed approach is very flexible and is capable of producing realistic flow predictions. It is felt that the material covered in this thesis is a good start for further study in the area of incompressible inviscid and viscous flow simulations using the cell vertex finite volume method on unstructured triangular meshes. Nonetheless, much remains to be done to improve both accuracy and efficiency of incompressible flow solver developed in the present work. The following sections give some suggestions for further research in these directions.

(1) Artificial Dissipation

Extensive numerical experimentations suggest that one of the factors affecting the accuracy and convergence of the central differencing discretisation scheme is the formulation of the artificial dissipation term. Application of this term requires appropriate mesh and parameter tuning which involves extensive effort. It could be beneficial, to find simple formulation which does not include tuning procedure. An alternative is the implemention of other form of space discretisation, such as higher order upwind or TVD schemes.

(2) Turbulence Model

The most pressing area requiring additional work is the inclusion of a turbulence model. In most real problems it is important to predict the turbulent effects on the flow solution.

(3) Transient Flow Simulations

Very recently, the approach of the artificial compressibility has been extended to the transient problems using semi-implicit schemes. It seems that, with some effort, the present scheme can be extended to the solution of transient problems on unstructured meshes. This can be done by adding an implicit loop for time stepping after the convergence of every explicit pseudo time marching to steady state conditions.

(4) Parallel Processing

An interesting area of work is the development of the capability to take advantage of parallel processing techniques. Mesh decomposition approaches have been already developed for the compressible flow solvers. Adopting the same techniques for the present solver facilitates a further development of the incompressible algorithm and would make it more applicable to large engineering problems.

(5) Free Surface Boundary Condition

Adding a free surface boundary conditions would provide the ability to solve more applied problem in the field of shallow water simulation and hydraulics.

(6) Grid Adaptation

Preliminary results indicate that more optimal results can be achieved using the combination of grid refinement and derefinement procedures. Automatic adaptation based upon an appropriate error estimation scheme, would help to obtain more accurate numerical solutions.

(7) Multi-grid Techniques

Numerical experimentations indicates that, as the physics of the flow increase in complexity, the size of the local time steps reduces. In such circumstances the dependency on convergence accelerating techniques will increase and multi-grid technique is an approach which should be investigated.

A

Preconditioned Method of Turkel for Incompressible Flow

Turkel [2] introduced different methods for solving the system of the equations using the conserved and primitive variables for both incompressible and low speed compressible flows. In this appendix, Turkel's preconditioner method for solving the system of equations using the conserved variables for incompressible flow is presented in three-dimensional form. However, the intent is to use methods that take advantage of the hyperbolicity of governing equations. Hence, the work is concentrated on the inviscid part of the equations.

Turkel considered a generalised method of Chorin's artificial compressibility approach [1]. The method allows the artificial time derivatives in all the equations, and not just the continuity equation, to provide faster convergence to the steady state conditions and facilitate the use of both the conservative and primitive variables. This generalised form of equations is formed by introducing two arbitrary variables of α and β in the general preconditioning matrix.

Then, it was shown that this form in results in a symmetric hyperbolic system and so it is well-posed for numerical solution.

This set of the equations which contains modifications in the unsteady terms of continuity and the momentum equations, can be written in the three-dimensional form as

$$\mathbf{P}^{-1}\frac{\partial \mathbf{W}}{\partial t} + \left(\frac{\partial \mathbf{f}^c}{\partial x_1} + \frac{\partial \mathbf{g}^c}{\partial x_2} + \frac{\partial \mathbf{h}^c}{\partial x_3}\right) = 0 \qquad (A.1)$$

where

$$\mathbf{P}^{-1} = \begin{bmatrix} \frac{1}{\beta^2} & 0 & 0 & 0 \\ \frac{(\alpha+1)u_1}{\beta^2} & 1 & 0 & 0 \\ \frac{(\alpha+1)u_2}{\beta^2} & 0 & 1 & 0 \\ \frac{(\alpha+1)u_3}{\beta^2} & 0 & 0 & 1 \end{bmatrix} \qquad (A.2)$$

Its inverse is given by

$$\mathbf{P} = \begin{bmatrix} \beta^2 & 0 & 0 & 0 \\ -(\alpha+1)u_1 & 1 & 0 & 0 \\ -(\alpha+1)u_2 & 0 & 1 & 0 \\ -(\alpha+1)u_3 & 0 & 0 & 1 \end{bmatrix} \qquad (A.3)$$

The preconditioning matrix is to be used as a device to create a well posed hyperbolic system of equations. Note that, $\alpha = -1$ gives Chorin's artificial compressibility approach. The system of equations can be integrated to a steady state solution the the incompressible inviscid equations.

In order to present the effect of the preconditioner matrix of P, equation (A.1) can be written in a quasilinear form

$$\frac{\partial \mathbf{W}}{\partial t} + \mathbf{G}_1'\frac{\partial \mathbf{W}}{\partial x_1} + \mathbf{G}_2'\frac{\partial \mathbf{W}}{\partial x_2} + \mathbf{G}_3'\frac{\partial \mathbf{W}}{\partial x_3} = 0 \qquad (A.4)$$

where the modified flux Jacobian matrices $\mathbf{G}_1'$, $\mathbf{G}_2'$ and $\mathbf{G}_3'$ are given by

$$\mathbf{G}_1' = \mathbf{P}\frac{\partial \mathbf{f}^c}{\partial \mathbf{W}} = \begin{bmatrix} 0 & \beta^2 & 0 & 0 \\ 1 & (1-\alpha)u_1 & 0 & 0 \\ 0 & -\alpha u_2 & u_1 & 0 \\ 0 & -\alpha u_3 & 0 & u_1 \end{bmatrix}$$

$$\mathbf{G}_2' = \mathbf{P}\frac{\partial \mathbf{g}^c}{\partial \mathbf{W}} = \begin{bmatrix} 0 & 0 & \beta^2 & 0 \\ 0 & u_2 & -\alpha u_1 & 0 \\ 1 & 0 & (1-\alpha)u_2 & 0 \\ 0 & 0 & -\alpha u_3 & u_2 \end{bmatrix} \qquad (A.5)$$

$$\mathbf{G}_3' = \mathbf{P}\frac{\partial \mathbf{h}^c}{\partial \mathbf{W}} = \begin{bmatrix} 0 & 0 & 0 & \beta^2 \\ 0 & u_3 & 0 & -\alpha u_1 \\ 0 & 0 & u_3 & -\alpha u_2 \\ 1 & 0 & 0 & (1-\alpha)u_3 \end{bmatrix}$$

Similar to the artificial compressibility approach, the hyperbolicity of equation (A.4) can be determined by examining the eigenvalues of matrix H' which is a linear combination of the modified flux Jacobian matrices taken as

$$\mathbf{H}' = \mathbf{G}_1'\zeta_1 + \mathbf{G}_2'\zeta_2 + \mathbf{G}_3'\zeta_3 = \qquad (A.6)$$

$$\begin{bmatrix} 0 & \zeta_1\beta^2 & \zeta_2\beta^2 & \zeta_3\beta^2 \\ \zeta_1 & (1-\alpha)\zeta_1 u_1 + \zeta_2 u_2 + \zeta_3 u_3 & -\zeta_2\alpha u_1 & -\zeta_3\alpha u_1 \\ \zeta_2 & -\zeta_1\alpha u_2 & \zeta_1 u_1 + (1-\alpha)\zeta_2 u_2 + \zeta_3 u_3 & -\zeta_3\alpha u_2 \\ \zeta_3 & -\zeta_1\alpha u_3 & -\zeta_2\alpha u_3 & \zeta_1 u_1 + \zeta_2 u_2 + (1-\alpha)\zeta_3 u_3 \end{bmatrix}$$

It can be shown that, for arbitrary real values of ζ_1, ζ_2 and ζ_3, the eigenvalues of H' are real and are given by

$$\lambda_1 = \frac{\mathrm{U}}{(1-\alpha)} \qquad \lambda_2 = \frac{\mathrm{U}+a}{(1-\alpha)} \qquad \lambda_3 = \frac{\mathrm{U}+a}{(1-\alpha)} \qquad \lambda_4 = \frac{\mathrm{U}-a}{(1-\alpha)} \qquad (A.7)$$

where

$$\mathrm{U} = (1-\alpha)(u_1\zeta_1 + u_2\zeta_2 + u_3\zeta_3) \qquad (A.8)$$

and

$$a = \mathrm{U}^2 + 4\beta^2({\zeta_1}^2 + {\zeta_2}^2 + {\zeta_3}^2) \qquad (A.9)$$

The convergence and stability of the explicit solution of this system of the hyperbolic equations depends upon the eigenvalues of the flux Jacobian matrices. In addition, the eigenvalues λ_1, λ_2, λ_3 and λ_4 can be used for the construction of the far field boundary conditions. However, multiplication of the unsteady terms by the preconditioning matrix P provides a control over the eigenvalue magnitudes, permitting the alternation of the system in such a way that renders it more appropriate for using for explicit numerical solution techniques. The system of equations can be well conditioned for numerical solution by selecting parameters α and β^2 in such a way that minimises the maximum wave speed of the system. The wave speed of the system of equations can be found by taking the Fourier transform of the quasilinear equation (A.4) and then substituting the x_1, x_2 and x_3 components of the Fourier transform variables for ζ_1, ζ_2 and ζ_3 into equation (A.7), respectively. From the expression of these eigenvalues, the maximum ratio of wave speeds can be determined algebraically in Fourier space. An approximation must then be made for application in physical space. Following this procedure, Turkel proposed the choice for an optimal value of β^2 given in an a priori choice of α as

$$\begin{aligned} &for \quad \alpha < 1 \quad \beta^2 = \quad (2-\alpha)(u_1^2 + u_2^2 + u_3^2) \\ &for \quad \alpha \geq 1 \quad \beta^2 = \quad \alpha(u_1^2 + u_2^2 + u_3^2) \end{aligned} \qquad (A.10)$$

Note that, for artificial compressibility preconditioning corresponding to $\alpha = -1$, the following scaling for β^2 results

$$\beta^2 = \quad 3(u_1^2 + u_2^2 + u_3^2) \qquad (A.11)$$

B

Galerkin Finite Element Formulation for Triangular Meshes

An alternative formulation for evaluating the viscous term for the cell-vertex finite volume scheme is developed [3] based on a Galerkin finite element approximation on triangular meshes [4]. In this appendix a brief description of Galerkin finite element formulation is given in two and three-dimensional form.

The spatial discretisation procedure begins by storing the variables at the vertices of the triangular and tetrahedral cells in two and three dimensions, respectively. In this formulation, the vector form of Navier-Stokes equation are multiplied by a test function φ, and integrated by parts to give the following form of governing equations

$$\frac{\partial}{\partial t}\int_{\Omega} \varphi \mathbf{W}\, d\Omega + \int_{\Omega} \varphi(\nabla \cdot \mathbf{F}^{c})\, d\Omega = \int_{\Omega} \varphi(\nabla \cdot \mathbf{F}^{d})\, d\Omega \qquad (B.1)$$

Integrating the flux integrals by part, and neglecting boundary terms gives

$$\frac{\partial}{\partial t}\int_{\Omega} \varphi \mathbf{W}\, d\Omega = \int_{\Omega} (\mathbf{F}^c \cdot \nabla\varphi)\, d\Omega - \int_{\Omega} (\mathbf{F}^d \cdot \nabla\varphi)\, d\Omega \qquad (B.2)$$

The discretisation takes place over each cell rather than around each side (Figure 6.5.b). Consider a typical triangular/tetrahedral cell inside a control volume associated with node i (Figure 4.4). In order to evaluate the flux balance equations at a vertex, the test function φ, has the value of unity at node i, zero at all other nodes and varies linearly across the cell. Therefore, the integrals in the above equation are non-zero only over node i, are zero at all other nodes and vary linearly across the cell.

For the left hand side of equation (B.2) which contains no derivative of φ, the test function can evaluated by using the following exact integration formulas

$$\begin{aligned} &for\ 2D: \int_{\Omega} \varphi_1^a \varphi_2^b \varphi_3^c \, d\Omega = \frac{a!\, b!\, c!}{(a+b+c+2)!}\, 2\Omega' = \frac{\Omega'}{3} \\ &for\ 3D: \int_{\Omega} \varphi_1^a \varphi_2^b \varphi_3^c \varphi_4^d \, d\Omega = \frac{a!\, b!\, c!\, d!}{(a+b+c+d+3)!}\, 6\Omega' = \frac{\Omega'}{4} \end{aligned} \qquad (B.3)$$

where Ω' is the volume of the cell. Thus, using equation (B.3), the left hand side of the equation (B.2) can be written as

$$\begin{aligned} &for\ 2D: \frac{\partial}{\partial t}\int_{\Omega'} \varphi \mathbf{W}\, d\Omega \approx \frac{\Omega'}{3}\frac{d\mathbf{W}}{dt} \\ &for\ 3D: \frac{\partial}{\partial t}\int_{\Omega'} \varphi \mathbf{W}\, d\Omega \approx \frac{\Omega'}{4}\frac{d\mathbf{W}}{dt} \end{aligned} \qquad (B.4)$$

The right hand side of equation (B.2) includes the derivatives of φ. They may be evaluated by integration of φ over the cell as

$$\nabla\varphi = \frac{1}{\Omega'}\oint_{\Gamma'}\varphi\, d\Gamma \approx \frac{1}{\Omega'}\sum_{k=1}^{M_{side}} \bar{\varphi}_k \vec{\Delta \mathrm{s}}_k \qquad (B.5)$$

where $\vec{\Delta s}_k$ is the vector of projected areas of side k, which is defined by equation (4.19). $\bar{\varphi}_k$ is the average of the test function of side k, as $\bar{\varphi}_k = \sum_{n=1}^{N_{dim}} \varphi_n / N_{dim}$ where N_{dim} is the number of nodes n at each side. Since the value of the test function is non-zero only at node i, only the contribution of terms which include φ_i remains in equation (B.5). Therefore, the derivatives of test function at every node i (Figure 4.4) can be written as

$$for\ 2D: \quad (\nabla\varphi)_i = \frac{1}{\Omega'}\left[\frac{\vec{\Delta \mathrm{l}}_{1i} + \vec{\Delta \mathrm{l}}_{2i}}{2}\right] = \frac{\vec{\Delta \mathrm{l}}_{12}}{2\Omega'}$$

$$(B.6)$$

$$for\ 3D: \quad (\nabla\varphi)_i = \frac{1}{\Omega'}\left[\frac{\vec{\Delta \mathrm{s}}_{12i} + \vec{\Delta \mathrm{s}}_{23i} + \vec{\Delta \mathrm{s}}_{31i}}{3}\right] = \frac{\vec{\Delta \mathrm{s}}_{123}}{3\Omega'}$$

As can be seen for each node n only $\vec{\Delta \mathrm{s}}_{12}$ and $\vec{\Delta \mathrm{s}}_{123}$, the two and three-dimensional vectors of projected areas of the side opposite to that node, is effective (Figure 4.4). For simplicity, these vector can be denoted by $\vec{\Delta \mathrm{l}}_n$ and $\vec{\Delta \mathrm{s}}_n$, respectively.

The convective fluxes $\mathbf{F}^c$ are taken as piecewise linear functions in space. Thus, they can be taken equal to their average values at the centre of the cell as $\bar{\mathbf{F}}^c = \sum_{n=1}^{N_{node}} \mathbf{F}^c_n / N_{node}$, where N_{node} is the number of nodes n at each cell. Then, by employing equation (B.6), the convective part of equation (B.2) for every cell can be written as

$$for\ 2D: \quad \int_{\Omega'} (\mathbf{F}^c \cdot \nabla\varphi)\, d\Omega \approx \left[\Omega'\left(\frac{\mathbf{F}^c_{n_1} + \mathbf{F}^c_{n_2} + \mathbf{F}^c_{i}}{3}\right)\right]\left[\frac{1}{2\Omega'}\sum_{n=1}^{3}\vec{\Delta \mathrm{l}}_n\right]$$

$$(B.7)$$

$$for\ 3D: \quad \int_{\Omega'} (\mathbf{F}^c \cdot \nabla\varphi)\, d\Omega \approx \left[\Omega'\left(\frac{\mathbf{F}^c_{n_1} + \mathbf{F}^c_{n_2} + \mathbf{F}^c_{n_3} + \mathbf{F}^c_{i}}{4}\right)\right]\left[\frac{1}{3\Omega'}\sum_{n=1}^{4}\vec{\Delta \mathrm{s}}_n\right]$$

The viscous flux includes first derivatives of φ. It can be shown that the discretisation of the viscous term is conservative inside each cell by considering the contribution of all the nodes of the cell. Assuming that the viscous flux is piecewise constant over each cell, the viscous stresses τ are calculated at the centre of cells. Hence, by using equation (B.6) for derivatives of φ, the diffusive part of equation (B.2) can be defined for each cell as

$$\begin{aligned} for\ 2D : \quad & \int_{\Omega'} (\mathbf{F}^d \cdot \nabla\varphi)\, d\Omega \approx \mathbf{F}^d \Big[\frac{1}{2\Omega'} \sum_{n=1}^{3} \vec{\Delta \mathbf{l}}_n\Big] \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad (B.8) \\ for\ 3D : \quad & \int_{\Omega'} (\mathbf{F}^d \cdot \nabla\varphi)\, d\Omega \approx \mathbf{F}^d \Big[\frac{1}{3\Omega'} \sum_{n=1}^{4} \vec{\Delta \mathbf{s}}_n\Big] \end{aligned}$$

Now, for a cell-vertex scheme, the discretisation of the convective and diffusive fluxes for each control volume can be achieved by integrating equations (B.7) and (B.8) over the cells sharing central node i of every control volume. Since the contribution of internal sides of the control volume appears twice, they cancel each other and sum of contour integrals of the interior cells equals to the contour integral over the control volume. By summing over all cells of control volume, the spatial discretisation of governing equations can be written as

$$\begin{aligned} for\ 2D : \quad & \Omega' \frac{d\mathbf{W}}{dt} = \Big[\sum_{k=1}^{N_{side}} \Big(\frac{\mathbf{F}^c_{n_1} + \mathbf{F}^c_{n_2}}{2}\Big)_k \vec{\Delta \mathbf{l}}_k - \frac{3}{2} \sum_{k=1}^{N_{side}} \mathbf{F}^d \vec{\Delta \mathbf{l}}_k\Big] \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad (B.9) \\ for\ 3D : \quad & \Omega' \frac{d\mathbf{W}}{dt} = \Big[\sum_{k=1}^{N_{side}} \Big(\frac{\mathbf{F}^c_{n_1} + \mathbf{F}^c_{n_2} + \mathbf{F}^c_{n_3}}{3}\Big)_k \vec{\Delta \mathbf{s}}_k - \frac{4}{3} \sum_{k=1}^{N_{side}} \mathbf{F}^d \vec{\Delta \mathbf{s}}_k\Big] \end{aligned}$$

where subscript $\Omega = \sum \Omega'$ represents the volume of control volume and N_{side} is the number of sides of the control volume.

This form leads to a similar discretisation form for the cell-vertex finite volume method. The only difference between this finite element formulation and the finite volume formulation for triangular and tetrahedral grids is in the method of evaluation of viscous terms and the constant coefficients. It can be shown that for equilateral triangular cells the discretisation of viscous term resulting from finite element formulation is equivalent to the cell-vertex finite volume formulation [5].

C

Modified Stencil for Viscous Stress Computations

The cell vertex finite volume approach employs a secondary cell around each boundary side of control volumes for the computation of contour integrals associated with the computation of the velocity gradients of the viscous stresses [6-8]. This secondary cell is formed by two neighbouring cells sharing the same side (Figure 5.5.a). The shape of this secondary cell has an important effect on the accuracy of the numerical solution of the governing equations. This stencil evaluates viscous stresses accurately on meshes in which their cells are inverse symmetric about the centre of the shared side. Although for two-dimensional meshes such a constraint can be satisfied it is very difficult to generate a three-dimensional mesh which meets such a requirement. Therefore, the initial secondary cell around the sides can be modified to form an appropriate secondary cell. This can be done by finding two extra points on the boundary of the initial secondary cell in the direction normal to the centre of the proposed side (Figure 5.6). In order to find the position of these points, it is necessary to solve the vector equation of the straight line passing through the central point of the proposed side coupled with the vector equation of the plane surface for every surrounding side of the initial secondary cell.

The vector equation of the straight line normal to the centre point c of a side can be defined as

$$\vec{\mathbf{r}}_m = \vec{\mathbf{r}}_c + h\vec{\mathbf{n}}_s \qquad (C.1)$$

where $\vec{\mathbf{r}}_c$ is the position vector of the central point of the interior side of the secondary cell and $\vec{\mathbf{r}}_m$ is the position vector of the desired point which is located on the normal line at the distance h from $\vec{\mathbf{r}}_c$. Generally any position vector can be defined as

$$\vec{\mathbf{r}} = x_1\hat{\mathbf{I}}_1 + x_2\hat{\mathbf{I}}_2 + x_1\hat{\mathbf{I}}_2 \qquad (C.2)$$

The components of $\vec{\mathbf{r}}_c$ can be obtained using a simple average of the coordinates of the proposed side s. Here, the unit vector $\vec{\mathbf{n}}_s$ can be considered equal to the normal unit vector of the proposed side s. The components of $\vec{\mathbf{n}}_s$ can be defined by equation (6.11).

On the other hand, the vector equation of the plane for every surrounding side k can be defined by the following relation

$$\vec{\mathbf{r}}_m \cdot \vec{\mathbf{n}}_k = D_k \qquad (C.3)$$

where $\vec{\mathbf{n}}_k$ is the unit normal vector and of side k and D_k is the characteristic constant of the particular plane which includes side k. This constant describes the normal distance of the plane from the origin. In practice, D_k can be found by introducing the position vector of one of the end nodes $\vec{\mathbf{r}}_n$ of the proposed side.

Note that, any surrounding side k which has an oblique angle with the central side s of the secondary cell, has no intersecting point with the normal straight line $'h\vec{\mathbf{n}}_s'$ which passes through the point c of the side s. Thus, the computation should only be made for those boundary sides of the secondary cell which satisfy the following criterion

$$\vec{\mathbf{n}}_s \cdot \vec{\mathbf{n}}_k < 0 \qquad (C.4)$$

However, there is a point associated with every surrounding side of the initial secondary cell which has an angle where the central side is not oblique. But, at left and right hand sides of the central side, only m_1 and m_2, two closest points to the mid point of the central side locate on the boundary of the initial secondary cell. The position vector $\vec{\mathbf{r}}_m$ and height h for every boundary side k of the secondary cell can be found by coupling the equation (C.3) with equation (C.1). Having found $\vec{\mathbf{r}}_m$ and h for all the boundary sides of the secondary cell, the two points m_1 and m_2 are those with the minimum distances h_l and h_r to the centre of the proposed side s. Now, the modified stencil can be completed by connecting these points to the end nodes of the side s.

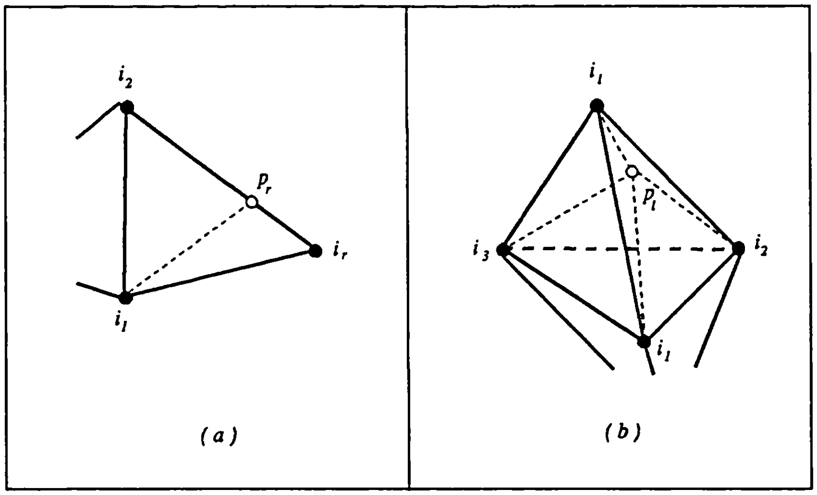

Figure C.1 *(a) 2D and (b) 3D description of the sub-cell Ω_n^**

By finding the position of the two auxiliary points m_1 and m_2, the flow variables $\mathbf{W}_m$ at these points can be interpolated from their neighbouring nodes within the cell. This can be done using the volume Ω_n^* of sub-cells which are formed by connecting the nodes of the cell to the auxiliary point m (Figure C.1).

$$\mathbf{W}_{\mathrm{m}} = \sum_{n=1}^{Nnode} \frac{\mathbf{W}_n \times \Omega_n^*}{\Omega'} \tag{C.5}$$

where $\mathbf{W}_n$ is the value of flow variables at node n and Ω_n^* is the sub-cell opposite to the node n which preclude that node. The volume of the computational cell Ω' can be computed by the summation of these sub-cells as

$$\Omega' = \sum_{n=1}^{Nnode} \Omega_n^* \tag{C.6}$$

In order to reduce the computational load, the position of auxiliary points m_1 and m_2 and the coefficient of interpolation related to every node Ω_n^*/Ω' can be computed once prior to the stages of computations.

The procedure for the computation of viscous stresses could be performed exactly in the same manner as cell-vertex finite volume stencil by replacing two neighbouring nodes i_l and i_r of every side surrounding control volume (Figure 6.5.a), with the two auxiliary computed point m_1 and m_2 (Figure 6.6). Note that, similar to the cell-vertex finite volume formulation, the computed viscous flux of every side must be added to the central nodes of the two neighbouring control volumes i_l and i_r, with opposite sign. The whole procedure can easily be done by using a side-based algorithm using a single do loop over all the sides of the computational domain. This modified secondary cell can be applied on both two and three-dimensional meshes.

D

The Unstructured Grid Generation Procedure Using the Delaunay Algorithm

The Delaunay triangulation and its geometrical properties have been widely known for a considerable time. However, the application to mesh generation techniques has only relatively recently been explored. In general the use of the Delaunay triangulation for grid generation requires a set of points interior to a given domain. Unfortunately, the Delaunay criterion does not give any indication as to how points should be defined. Therefore, it is necessary to construct a way of grid point generation for arbitrary geometries. A key requirement of a mesh generation procedure is to ensure that the mesh is boundary conforming i.e. in two dimensions, edges in the triangulation form the boundaries and, in three dimensions, triangular faces of the assembly of tetrahedra conform to the boundary surface. Unfortunately, given a set of points n_i and corresponding triangulation $T(n_i)$, obtained from the Delaunay triangulation of the points, there is no guarantee that the bounding edges, or faces, will be contained within $T(n_i)$. The boundary connectivity constraint, required of a boundary conforming mesh generator, is not naturally built into Delaunay construction. The length scale and point distribution determine the point connectivities, irrespective of the boundary surfaces. Hence, it is necessary to ensure that the problem is well defined and that the boundary data is appropriately included into any Delaunay triangulation algorithm. Generally, in the development of a Delaunay triangulation grid generation, three problems must be overcome.

i) Connecting the points which can be resolved using Voronoi diagram

ii) Generating the position of interior points.

iii) Ensuring that the resulting triangulation is boundary conforming.

The global procedure for the generation of three-dimensional grids can be defined as follow.

1) Input boundary points n_i, $i = 1, N_{Boundary}$ and boundary point connectivities of the faces C_j, $J = 1, M_{Boundary}$.

2) Derive boundary edges E_k, $k = 1, N_{Edges}$, N_{Edges} from boundary face connectivities C_i.

3) Perform the Delaunay triangulation of n_i to obtain tetrahedra $T_m, m = 1, N_{T_{(total)}}$.

4) Create interior field points and connect using the Delaunay triangulation to form tetrahedra T_l, $l = 1, N_{T_{(interior)}}$.

5) Ensure that the surface triangulation of C_j, $j = 1, N$ is contained in the volume triangulation. Recover any missing faces by following the steps; recover boundary edges E_k in T_l, and recover boundary faces C_j in T_l.

6) Identify all tetrahedra outside the domain of interest T_{0_h}, $h = 1, N_L$.

7) Delete tetrahedra T_{0_h}, $h = 1, L$ to give the final grid $T_l, l = 1, N_S$.

Following the grid construction, post-processing can be applied to smooth the grid, check consistencies and derive grid quality statistics prior to use with an analysis module.

Two algorithms has been reported on the construction of the Delaunay triangulation follows by Bowyer [9] and Watson [10]. The approach discussed by Bowyer [9] is favoured over that of the companion paper by Watson [10] because it is more readily applied to the construction in three dimensions [11]. Here, the algorithm used to generate the Delaunay triangulation follows the work of Bowyer. This algorithm, which is based on the in-circle criterion, is a sequential process; each point is introduced into an existing Delaunay satisfying structure, which is broken and then reconnected to form a new Delaunay triangulation. In three dimensions the algorithm, in step-by-step format is as follows.

1) Define a set of points which form a convex hull within which all points will lie. An appropriate Delaunay data structure is established for this construction. It should be noted that some vertices of the associated Voronoi diagram are not strictly defined. Since they lie outside the convex hull and therefore do not possess four forming points of a tetrahedron, thus do not have four neighbour Voronoi vertices. Default values are used in this case.

2) Introduce a new point anywhere within the convex hull.

3) Determine all vertices of the Voronoi diagram to be deleted. A point which lies within a sphere, centred at a vertex of the Voronoi diagram and which passes through its four forming points, results in the deletion of that vertex. This follows from the 'in-circle' criterion of the Voronoi construction.

4) Find the forming points of all the deleted Voronoi vertices. These are the contiguous points to the new point.

5) Determine the neighbouring Voronoi vertices to the deleted vertices which have not themselves been deleted. These data provide the necessary information to enable valid combinations of the contiguous points to be constructed.

6) Determine the forming points of the new Voronoi vertices. The forming points of new vertices must include the new point together with three points which are contiguous to the new point and form a face of a neighbouring tetrahedra (these are the possible combinations obtained from Step 5).

7) Determine the neighbouring Voronoi vertices. Following Step 6, the forming points of all new vertices have been computed. For each new vertex, perform a search through the forming points of the neighbouring vertices, as found in Step 5, to identify common triples of forming points. When a common combination occurs, neighbours of the Voronoi diagram have been found.

8) Reorder the Voronoi diagram data structure, over-writing the entries of the deleted vertices.

9) Repeat Steps 2-8 for the next point.

E

Pressure-based Methods For Solving Incompressible Flow Problems

The solution procedure for the velocity and pressure in pressure-based methods can be performed in either an un-coupled or coupled manner. The computational strategy for solving the conservation equations in an un-coupled manner, sequentially for one variable at a time, is known as the segregated approach. Most of the solution methods reported in the literature have followed the segregated strategy.

One of the most general segregated methods is known as the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) procedure [12]. This procedure is based on a cyclic series of guess-and-correct operations to solve for the primitive variables. The velocity components are first calculated from the momentum equations using a guessed pressure field. The pressure and velocities are then corrected so as to satisfy the continuity equation. This process continues until the solution converges. In this procedure the actual pressure is defined as a combination of pressure and a pressure correction value. However, for some problems the SIMPLE procedure over-estimates the value of the pressure correction. Hence, an under-relaxation parameter is defined.

Another method is known as PISO (Pressure-Implicit with Splitting the Operators) algorithm which uses slightly different formulations from the SIMPLE procedure and adds another correction for the velocity which is computed from the predicted pressure field [13]. The method utilises the spilitting of the operations in the solution of the discretised momentum and pressure equations in such a way that the flow fields obtained at each time step are close approximations of the exact solution of the difference equations with a formal order of accuracy. Since this method uses a true pressure equation, it may be preferable.

Since it is not possible to readily determine the exact value of the under-relaxation parameter of SIMPLE procedure, the SIMPLER (SIMPLE Revised) procedure is developed to improve the convergence. In this procedure the velocity is guessed initially and the pressure is computed from the Poisson equation then the corrected velocities are computed using the momentum equations [12]. In spite of the improved convergence behaviour of SIMPLER over SIMPLE, these procedures do not appear to be free from difficulties. Under-relaxation is usually necessary especially on fine grids and for complex geometries. In some cases the use of the pressure correction process in SIMPLER procedure seems to destabilize the convergence process [14].

Several methods have been developed for improving the convergence of the segregated pressure-based methods which involve solving a system of equations. These methods are classified as

I) Approximate Factorisation Technique:

1) Implicit Alternative Direction technique (ADI)

2) Implicit Lower-Upper technique (ILU)

II) Conjugate Gradient type accelerations (CG) :

1) Generalised Minimal Residual (GMRES)

2) Quasi-Minimal Residual (QMR)

3) Two norm Minimal Residual (Bi-CG)

4) Conjugate Gradient Square method (CGS)

5) Conjugate Gradient Stable method (CGSTAB)

6) Conjugate Gradient applied to Normal equations (CGN)

7) Transport Free Quasi-Minimal Residual (TFQMR)

III) Multi-Grid accelerations (MG) :

1) Full Multi-Grid (FMG)

2) Discretisated Coarse Grid Approximation (DCGA)

3) Galerkin Coarse Grid Approximation (GCGA)

IV) Block Correction acceleration (BC)

The users of segregated methods believe that it is very costly to solve velocity-pressure coupling when nonlinearity of the equations has not been solved. Although fully coupled solvers increase the robustness, improve the convergence and consume less computational time than segregated methods, they are based upon a more complex linear solver for the governing equations. The first attempt to solve simultaneously the governing equations in a coupled way on staggered grids was the so-called SIVA algorithm [15]. The pressure and the six surrounding staggered velocities on the control volume were solved simultaneously. Momentum equations were combined yielding a Poisson-like equation for the pressure. Owing to the point-wise nature, SIVA was found to converge rather slowly. Similar attempt lead to CELS algorithm [16]. More recent methods follow the boundary layer practice in which the domain is swept from upstream to downstream, implicit differencing of momentum being used for stability. Upstream influence through the pressure field has been accounted by introducing some form of forward differencing for the stream-wise gradient which allows departure-free behaviour. Such methods have been developed in the framework of the so-called partially parabolic assumption [17]. Unfortunately, in such methods the coupling is performed only in sub-domains so that the resulting matrices to solve are easy to handle, but poor convergence rate is sustained, especially on fine grids mainly because of the weak coupling between sub-domains. This situation is in some respect improved with multi-grid methods [18,19]. Another alternative to the one-step method (non-segregated) has been developed using so-called influence matrix techniques. This method is very complex and the cost of solving the influence matrix system is very high in three dimensions. Although the fully coupled method does not appear to bring sufficient improvements with respect to standard Poisson based methods [14].

It should be noted that in pressure-based approaches, the appropriate discrete boundary condition for pressure is obtained in the form of Neumann conditions which uses pressure derivatives. For a complete implementation of the Neumann boundary condition for the pressure, it is necessary to use a staggered computational grid for the solution of the Poisson equation. A staggered grid is a grid where the velocity components and the pressure are defined on separate overlapping grids. Recently, an alternative grid arrangement has been introduced for this method, known as a collocated grid, which requires less computational storage allocation and less interpolation than the staggered grid approach but requires two adjustable under-relaxation factors [20].

F

Finite Element Methods for Solving Incompressible Flow Problems

In the finite element computation of incompressible flows, using a standard Galerkin formulation, there are two main sources of potential numerical instabilities. One is due to the presence of the convection term. Symmetric treatment of the convection term by standard Galerkin formulation, in which the test and trial functions are similar, is identified as the sources of numerical instabilities. This is true in a sense that Galerkin finite element discretisation (equal to second order differencing) of the convection term produces a set of equations that are decoupled between adjacent nodes and the scheme can result in spurious node-to-node oscillations. Such oscillations become more significant for high Reynolds numbers with sharp internal or boundary layers in the solution. Although these oscillations can be suppressed by severely refining the mesh, the necessary degree of refinement is not often economical. In the finite volume and finite difference contexts, such instabilities can be removed by the use of dissipation or upwind operators. In the finite element community, some efforts to stabilise the Galerkin finite element method have lead to implementation of the Petrov-Galerkin formulation, in which the trial and test functions are different. The test functions are chosen equal to trial function plus a higher-order function in order to give more weight to upwind nodes. This method is first order accurate and the extension to unsteady cases or when a source term is present does not give a consistent formulation [21].

Soon it was pointed out that to reduce the oscillations, it is sufficient to add an artificial diffusion term acting only in the direction of the streamlines. Then it was observed that a streamline diffusion term can be introduced naturally in the standard Galerkin method by perturbing the weighting functions with their derivatives without modifying the original governing equations. This method which is called SUPG (Streamline Upwind Petrov Galerkin) or SDM (Streamline Diffusion Method) results in a higher order accurate method with good stability properties. Nevertheless some oscillation remain, due to the absence of control gradient in directions other than the streamlines. However, the SUPG is more stable than the standard Galerkin formulation in which the presence of sharp layers globally pollutes the solution. In order to remove the shortcoming of SUPG, a discontinuity capturing term can be added to the scheme. The additional term enhances the ability of the method to produce smooth approximations to internal and boundary layers.

Later the SUPG formulations was generalised by adding the least squares form of the residuals to the Galerkin formulation [22]. This approach is called GLS (Galerkin Least Squares) and coincides with the SUPG method in the hyperbolic case, or piecewise linear elements.

The other source of instability is produced if an inappropriate combination of interpolation function for the velocity and pressure is used to solve the Navier-Stokes equations. After employing equal order interpolation it was noticed that a good velocity solution was usually accompanied by a very poor meaningless pressure solution [23]. When mixed interpolation (or unequal interpolation) was employed velocity and pressure made sense. Equal order interpolation generally results in a singular matrix [24]. To overcome the above deficiency, the velocity and pressure interpolating functions have to satisfy certain conditions of compatibility called the Babuska-Brezzi conditions. When the steady state equations for incompressible flow are cast in terms of the primitive variables u and p, it can be readily seen that the appearance of zero diagonal terms renders the system singular unless the number of velocity variables is greater than the pressure variables. The Babuska-Brezzi condition restricts the use of equal order of interpolation for u and p, and hence, considerable limits the combination of the velocity and pressure functions, and complicates the programming and data handling procedure [25].

Several methods have been developed for circumventing this restriction using pairs of elements for velocity and pressure or high degree polynomials. An alternative to satisfy the Babuska-Brezzi condition are the use of a pair of elements or different polynomials for the velocity and pressure approximations. These create considerable programming headaches. However the same mesh for velocity and pressure can be applied using high degree polynomial elements like Taylor-Hood element which the assembly cost, consequently, is higher than ordinary elements. There are other forms of divergence-free elements which either are inaccurate (involving low degree finite element pairs) or cumbersome to program (involving high degree elements pairs) [14].

An alternative technique which relaxes the well known Babuska Brezzi stability condition in the velocity-pressure discretisation using equal order of interpolation is Fractional Step technique (Velocity Correction method) [26,27]. In this method the momentum equations is splited into two parts which are solved in two stages and a pseudo transient term is added to the continuity equation. In this method which is suitable for steady state computations, the discrete steady state equations have no zero diagonal and the difficulties associated with equal order interpolation coupled will not appear [28].

Some other methods have been developed and analysed for circumventing Babuska-Brezzi in the sense of equal order polynomials with respect to the same mesh may be used for the velocity and pressure. These methods which directly couples the velocity and pressure solution involves a modification of the discrete incompressibility constraint. The coupling between the velocity and pressure may be achieved by methods like Penalty and Artificial Compressibility methods [14].

The system of equations discretised by the Finite Element method is a system of non-linear equations in terms of velocity, pressure and corresponding finite element space. In order to solve the non-linear system of Navier-Stokes equations, some linearisation procedure can also be invoked. Thus the problem can be solved by applying some relaxation procedures like Newton, the Simple Iteration and Continuation methods.

In Newton's method, the discretised primitive variable formulation is defined in a linear form using the following procedure. Given an initial guess u^0 for velocities, one can generate the sequence of u_i^n and p_i^n for $n = 1, 2, 3, ...$ by solving the linear problem of

$$\begin{gathered} h(u^n) = 0 \\ f(u^n) + g(u^{n-1}, u^n) + g(u^n, u^{n-1}) + h(p^n) = c + g(u^{n-1}, u^{n-1}) \end{gathered} \qquad (F.1)$$

where f, g, h and c are the oprators which form the continuity and momentum equations. Newton's method requires the evaluation and solution of a new Jacobian matrix at every iteration. At the price of a reduced convergence rate one can lessen these costs by employing a quasi-Newton or updated method such as Broyden's method [30].

The Simple Iteration method can be applied for sufficiently low Reynolds numbers and it can be shown that it is globally and linearly convergent [31]. Given an initial guess u^0 for the velocity, one generates the sequence of u_i^n and p_i^n for $n = 1, 2, 3, ...$ by solving the linear problem of

$$h(u^n) = 0$$

$$f(u^n) + g(u^{n-1}, u^n) + h(p^n) = c \tag{F.2}$$

A hybrid method can be defined to take advantages of both Newton's and the Simple Iteration methods by defining

$$h(u^n) = 0$$

$$f(u^n) + g(u^{n-1}, u^n) + \alpha_g g(u^n, u^{n-1}) + h(p^n) = c + \alpha_g g(u^{n-1}, u^{n-1}) \tag{F.3}$$

The procedure for the computation can be started using the Simple Iteration method $\alpha_g = 0$, and close to the solution it can switch to the Newton's method $\alpha_g = 1$.

The Continuation methods can be used for high Reynolds numbers. Since the convergence of the solution depends upon the initial guess, the method for solving low Reynolds numbers can be used for obtaining the appropriate initial guess. Then by using information obtained by solving the problem for desired Reynolds number the computations can be proceed using the iterative methods. For solving very large sparse, linear systems of algebraic equations, particularly for high Reynolds numbers or three dimensions, gauss elimination solvers are not often adequate. As a result there have been numerous iterative methods for solving the linear systems. Including in this category are variants of Conjugate Gradient, Multi-Grid, Reduced Basis, Generalised Minimum Residuals, and related algorithms. It should be noted that no general consensus has been reached with regard to which approach is the best. Finally, for time dependent problems, the system of semi-discretised equations can be solved by one of the methods namely ; Single-Step Fully Implicit, Single-step Semi-Implicit, Backward Differentiation Multi-Step schemes. An alternative to solve a time independent problem is to determine the steady state solution as the large limit of the transient problem [14].

References

[1] A. Chorin, 'A Numerical Method for Solving Incompressible Viscous Flow Problems',J. of Comp. Physics , **2**, pp. 12–26, (1967).

[2] E. Turkel, 'Preconditioned Method for Solving the Incompressible and Low Speed Compressible Equations', ICASE Report 86–14, NASA Langley Centre, (1986).

[3] D. J. Maviriplis, A. Jameson, L. Martinelli, 'Multi-grid Solution of the Navier-Stokes Equations on Triangular Meshes', AIAA-89-0120 , (1989).

[4] O. C. Zienkiewicz, 'The Finite Element Method ', New York, McGraw-hill, (1977).

[5] L. A. Sykes, 'Development of a Two-Dimensional Navier-Stokes Algorithm For Unstructured Triangular Grids', ARA Report 80, April 1990.

[6] N. P. Weatherill, L. J. Johnston, A. J. Peace and J. A. Shaw, 'Development of a Method for the Solution of the Reynolds-Averaged Navier-Stokes Equations on Triangular Grids', ARA Report 70, (1986).

[7] R. C. Swanson, E. Turkel, 'A Multistage time-stepping Scheme for the Navier-Stokes Equations', AIAA paper 85-0035 (1985).

[8] L. Martinelli, A Jameson, F. Grasso, 'A Multi-grid Method for the Navier-Stokes Equations', AIAA paper 86-0208 (1986).

[9] A. Bowyer, 'Computing Dirichlet Tessellation', The Computer Journal, **24**, pp. 162–166, (1981).

[10] D. F. Watson, 'Computing the n-dimensional Delaunay Tessellation with Application to Voronoi Polytopes', The Computer Journal, **24**, pp. 167–172, (1981).

[11] N. P. Weatherill and O. Hassan, 'Efficient Three-Dimensional Delaunay Triangulation with Automatic Point Creation and Imposed Boundary Constraints' Int. J. for Num. Meth. in Eng., **37**, pp. 2005–2039, (1994). (1986).

[12] S.V. Patankar, 'Numerical Heat Transfer and Fluid Flow', Hemsphere Publishing Corp, pp. 126–130, (1980).

[13] R.I. Issa, 'Solution of the Implicitly Discretised Fluid Flow Equations by Operator-Splitting', J. of Comp. Phys., **62**, pp 40–65 (1985).

[14] R. Peyret, 'Computational Fluid Mechanics', Academic Press INC, (1996).

[15] L.S. Caretto, A.D. Gosman, S.V.Patankar, and D.B.Spalding, Proceeding of Third International Cont. Numerical Methods in Fluid Dynamics, **60**, (1972).

[16] P.F. Gaplin, J.P. Van Doormaal, and G.D. Raithby, ' Solution of the Incompressible Mass and Momentum Equations by Application of a Coupled Equation Line Solver', Int. J. of Num. Meth. in Fluids, **5**, pp. 615–626, (1985).

[17] M. Rosenfeld, and D. Kwak, 'Time-Dependent Solution of Viscous Incompressible Flows in Moving Coordinates', Int. J. of Num. Meth. in Fluids, **13**, pp. (1991).

[18] S.P. Vanka, 'Block-Implicit Multi-grid Solution of Navier-Stokes Equation in Primitive Variables', J. of Comp. Physics, **65**, pp. 138–158 (1985).

[19] P.M. Sockol, 'Multigrid Solution of the Navier-Stokes Equations on Highly Stretched Grids', Int. J. of Num. Meth. in Fluids, **17**, pp. 543–566 (1993).

[20] P. Tamamidis, G. Zhang, D. N. Assanis, 'Comparison of Pressure-Based and Artificial Compressibility Methods for Solving 3D Steady Incompressible Flows', J. of Comp. Physics, **124**, pp. 1–13 (1996).

[21] J.C. Heinrich and O.C. Zienkiewicz, 'Quadratic Finite Element Schemes for Two-dimensional Convective-transport Problem', Int. J. of Num. Meth. in Eng., **11**, pp 1831–1844, (1977).

[22] T.J.R. Hughes, L.P. Franca and G.M. Hulbert 'New Finite Element Formulation for Computational Fluid Dynamics: VIII. The Galerkin/Least-Squares method for advective-diffusive equations', Comp. Meth. in Appl. Mech. Eng., **73**, pp 173–189, (1989).

[23] C.Taylor and P.Hood, 'A Numerical solution of the Navier-Stokes Equations using FEM Technique', J. of Computer and Fluids, **1**, pp 73–100, (1973).

[24] R.L. Sani, P.M. Gresho, R.L. Lee and D.F. Griffiths 'The Cause and Cure of the Spurious Pressures Generated by Certain FEM Solutions of the incompressible Navier-Stokes Equations, Part 1', Int. J. of Num. Meth. in Fluids, **1**, pp 17–43, (1981) [Part 2: **2**, pp 171–204, (1981)].

[25] S.K. Hannani, M Stanislas, P. Dupont, 'Incompressible Navier-Stokes Computations with SUPG and GLS Formulations', Compt. Meth. in Appl. Mech. Eng., **124**, pp 153–170, (1995).

[26] G. E. Schneider, G. D. Raithby, M. M. Yovanovich, 'Finite Element Analysis of Incompressible Fluid Flow Incorporating Equal Order Pressure and Velocity Interpolation', in Numerical Methods in Laminar and Turbulent Flow, C. Taylor, K. Morgan , C. A. Brebia , Pentech Press, Plymouth, (1978).

[27] M. Kavahra, K. Ohmiya, 'Finite Element Analysis of Incompressible Density Flow Using Velocity Correction Method', Int. J. for Num. Meth. in Fluids, **5**, pp 981–993 , (1985).

[28] O. C. Zienkiewicz, K. Morgan and B. V. Satya Sai, 'A General Algorithm for Compressible and Incompressible Flow-Part I and II', Int. J. for Num. Meth. in Fluids, **20**, pp 887–913 , (1995).

[29] R. Peyret and T. T. Taylor, 'Computational Methods for Fluid Flow', Springer-Verlag, 1983.

[30] H. Matthies and G. Strang,' The Solution of the Nonlinear Finite Element Equations',Int. J. of Num. Meth. in Eng. **14**, pp 1613–1626 (1977).

[31] V. Girault and P.A. Raviart, 'Finite Element approximation of the Navier-Stokes Equations ', Springer-Verlag, Berlin, (1979).