NUMERICAL PREDICTION OF FILM COOLING OF TURBINE BLADES USING MULTIBLOCK CURVILINEAR GRIDS

By

Pingfan He

B.Sc., ZhengZhou University, M.Sc., IAPCM, China

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

PH. D

in

THE FACULTY OF GRADUATE STUDIES

INSTITUTE OF APPLIED MATHEMATICS

DEPARTMENT OF MECHANICAL ENGINEERING

DEPARTMENT OF MATHEMATICS

We accept this thesis as conforming

to the required standard

.....	.....
.....	.....
.....	.....

THE UNIVERSITY OF BRITISH COLUMBIA

1995

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Institute of Applied Mathematics
Department of Mathematics
Department of Mechanical Engineering
The University of British Columbia
2075 Wesbrook Place
Vancouver, Canada
V6T 1W5

Date: Sept 19, 1995

Abstract

In the present thesis, a computational capability is developed for the prediction of film cooling of turbine blades. This includes the development of two comprehensive numerical codes and the associated methods. To address the difficulties associated with complex configurations of turbine blades and convergence problems, several numerical techniques are used, including curvilinear coordinate-based calculations, multigrid acceleration, domain segmentation and grid generation. Investigation and validation of these methods are carried out and novel techniques are proposed to achieve an efficient numerical solver.

Two computer codes have been developed. One is a 3D curvilinear coordinate-based CFD code, called CMGFD, which can be used to calculate laminar/turbulent flows in arbitrary geometries using non-structured curvilinear grids. The methods developed here have been implemented into the codes. To support the application of the CMGFD code, a multigrid elliptic grid generation code, MBEGG, was developed which can be used to generate multi-block curvilinear grids for the CMGFD code. A multigrid method is used to solve the three elliptic grid generation equations thus providing an efficient grid generator. The developed computational codes are applied to study film cooling of an experimental turbine blade model. The computational domain follows the physical geometry which includes a curved blade surface and a number of injection holes. A block-structured curvilinear grid is generated by the MBEGG code which exactly represents the inclined, round film-holes and the curved blade surface. The computational results show that the developed numerical tool has the potential to accurately model the complex cooling process in actual blade geometries.

Table of Contents

List of Tables

List of Figures

Nomenclature

C_μ, C_1, C_2	The $k-\epsilon$ turbulence model constants.
F^i	Volume flow rates across cell faces.
G	Generation rate of turbulent kinetic energy.
$\mathbf{J}$	Generalized flow flux.
M	Mass flow ratio, $M = \dfrac{\varrho_c U_c}{\varrho_\infty U_\infty}$.
M_{15}	Mass flow ratio of the first row coolant orifices.
M_{44}	Mass flow ratio of the second coolant orifices.
M_∞	Overall mass flow ratio averaged from all coolant holes.
P_i, i=1,2,3	Control functions in grid generation equations.
$\mathbf{S^i}$	Surface area vectors.
T	Temperature.
U^i	Contravariant velocities.
U^{ξ_i}	tangential velocity components.
u_τ	Friction velocity, $u_\tau = \dfrac{\tau_w}{\varrho}$
V	Volume of a control cells.
X	Streamwise distance measured from stagnation line.
Z	Spanwise distance measured from one edge of a first row hole.
$\mathbf{a_i}$	Covariant vectors.
$\mathbf{a^i}$	Contravariant vectors.
a_{ij}	Coefficients of the grid generation equations.
a_{nb}	Coefficients of the discretized governing equations.

$\mathbf{e_i}$	Tangential vectors.
$\mathbf{e^i}$	Normalized surface vectors.
d	Injection orifice diameter.
g^{ij}	Surface area metric tensor.
k	Turbulent kinetic energy.
p	Mean static pressure.
$\mathbf{u}$	Velocity vector.
x_i	Cartetian coordinates.
x, y, z	Cartetian coordinates.
y^+	Dimensionless distance from the solid wall, $y^+ = \frac{\varrho y U_\tau}{\mu}$.
α_i	Non-orthogonal angles.
ϵ	Dissipation rate of turbulent kinetic energy.
η	Film cooling effectiveness, $\eta = \frac{T_\infty - T_{aw}}{T_\infty - T_c}$.
κ	Von Karman constant, $\kappa = 0.41$.
μ	Dynamic viscosity.
ξ_i	Curvilinear coordinates.
ϱ	Fluid Density.
σ_k, σ_ϵ	Turbulence model constants.
τ_w	Wall shear stress.
θ	Non-dimensional temperature, $\theta = \frac{T_{aw} - T_c}{T_\infty - T_c}$.
ϕ	General dependent variable.
Γ	General diffusivity coefficient.

Subscript

aw	Adiabatic wall.
c	Coolant injection.

∞	Main stream.
e, w, ...	Center of control volume surfaces.
E, W, ...	Control volume center.
15	Referring to row 1.
44	Referring to row 2.

Superscript

c	Coarser grid.
f	Finer grid.
ξ_i	ξ_i oriented variables.

Acknowledgment

I wish to express a deep sense of gratitude to my supervisors Dr. M. Salcudean and I. S. Gartshore for their encouragement, guidance and support throughout the course of this research. Working with them has been an enjoyable experience which I will cherish throughout my life.

I would like to thank my co-supervisor Professor B. Seymour in the Institute of Applied Mathematics, for the numerous pieces of advice he has give me. Thanks are also due to Professor J. Varah and Professor M. Quick.

I would also like to thank all my colleagues of the CFD group in the Department of Mechanical Engineering, especially Dr. Z. Abdullah and Dr. P. Nowak. Their suggestions and discussions have been in valuable. Thanks to Mr. M. Findlay for his proofreading of this thesis.

Finally, I wish to thank my wife, Hui, for her patience and support.

Chapter 1

Introduction

1.1 Background

One of the most effective ways to improve gas turbine engine efficiency is to increase the working fluid temperature. Consequently, new turbine designs are continually being pushed to operate at higher temperatures. The modern gas turbine engines operate under working temperatures of $1400-1500^0C$ which are beyond the allowable metal temperatures. Therefore, to maintain reasonable engine life and safety, critical components, such as the turbine blades, have to be protected.

Film cooling is a widely used technique to protect turbine blades from the high temperature of the surrounding gas stream. This technique involves injection of coolant at one or more discrete locations along a surface exposed to a high temperature environment in order to protect the surface in the downstream region. A discrete-hole cooling arrangement for a turbine blade is illustrated in Figure 1.1. In this blade there are a number of discrete, small injection holes on the surface from which coolant is injected into the boundary layer on the blade surface to form a thin layer of coolant which protects the blade surface from the high temperature environment. The cooling process occurs in a complex 3D configuration which includes not only a curved airfoil but also a number of small injection holes. The aim of film cooling design is to minimize the coolant injected while maximizing the thermal protection for the blades. A large number of experimental and computational studies have been devoted to the understanding of the film cooling

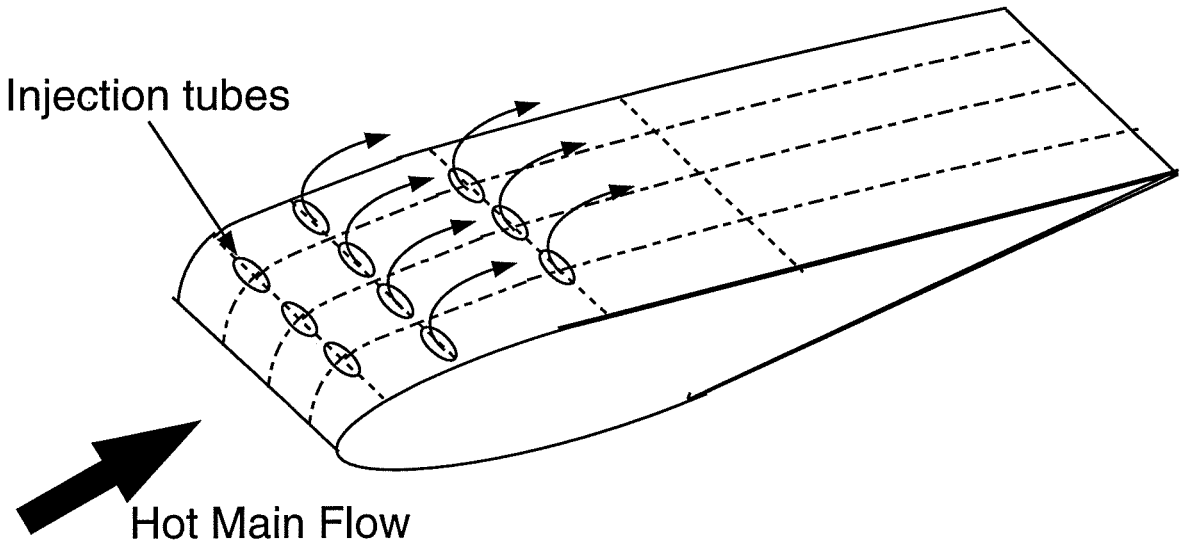

Figure 1.1: Illustration of a discrete-hole cooling arrangement for a turbine blade.

process in order to achieve higher film cooling effectiveness. A brief literature review is provided in the following paragraphs.

1.2 Motivation for the Present Study

Film cooling has been a subject of research for many years. There are a large number of research papers on film cooling of turbine blades in the open literature. Goldstein (1971) gave a detailed review of film cooling research and summarized the experimental results on film cooling effectiveness with normal, tangential or nearly tangential injection up to 1971. There are two major techniques in film cooling studies; one is experimental investigation and the other is numerical prediction. Most early investigations of film cooling were carried out using experimental techniques. Current engine design is highly empirical and relies on a large experimental database. Recent works are more concerned with discrete-hole film cooling on a flat plate or a curved blade surface. There have been numerous experimental investigations of discrete hole cooling. However, a review of experimental studies is not the subject of the present thesis. Instead, only a selection of some experimental works associated with the physical model of the present numerical study are mentioned.

In the film cooling of a turbine blade, the stagnation region near the leading edge is a particularly critical region as it suffers the most intense exposure to the hot free stream and therefore requires particular attention. A number of studies have been carried out for film cooling at the leading edge. Ou & Han (1991, 1992) and Mehendale & Han (1992) carried out a systematic investigation of the effects of high mainstream turbulence and the effects of injection orifice geometry on the leading edge film cooling. They measured the film cooling effectiveness over a blunt body with a semi-cylindrical leading edge and

a flat afterbody. Here, the film cooling effectiveness is defined as follows,

$$\eta = \frac{T_\infty - T_{aw}}{T_\infty - T_c} \tag{1.1}$$

where T_{aw} is the adiabatic wall temperature, T_∞ is the mainstream temperature and T_c is the coolant temperature. Two rows of orifices at $\pm 15^0$ and $\pm 40^0$ were used with spanwise blowing and with spacing-to-diameter ratios of 4 and 3. The effectiveness was found to decrease with increasing mainstream turbulence; however, this effect decreased with increasing blowing rate. Their study also indicates that the spacing-to-diameter ratios have a significant influence on the film cooling effectiveness.

Recently, a systematic experimental investigation of film cooling effectiveness near the leading edge of a turbine blade was carried out by Gartshore *et al* (1993) and Salcudean *et al* (1994) at the University of British Columbia. This study was carried out to fulfill a research collaboration between Pratt & Whitney Canada and the University of British Columbia. A large turbine blade model was used in this experimental study. This model has a semi-circular leading edge with four rows of laterally-inclined film cooling orifices positioned symmetrically about the stagnation line. Both air and CO_2 were used as coolant. A heat-mass transfer analogy was used to measure the cooling effectiveness. Their experimental results indicate that the film cooling effectiveness changes substantially with respect to the overall mass flow ratio. Here, the mass flow ratio is defined as $\varrho_c U_c / \varrho_\infty U_\infty$, where ϱ_c and ϱ_∞ are the coolant density and the mainstream flow density respectively. The best effectiveness was obtained in a range of mass flow ratios $0.5 - 0.6$. A continuous decrease in the film cooling effectiveness was found when the mass flow ratio was further increased.

The above experimental studies indicate that the film cooling effectiveness is influenced by a number of factors, such as, spacing-to-diameter ratios, mass flow ratios and

free stream turbulence. Therefore, in order to achieve an optimal design of the film cooling system, a parametric analysis has to be carried out to determine the film cooling performance with various blade geometries and flow parameters. For this purpose, experimental measurements are too expensive and time-consuming to obtain the detailed flow and heat transfer data. In addition, measurement is also limited by the availability and accuracy of the measuring instruments for many situations. An alternative method in film cooling research is numerical prediction.

Recent advances in high speed digital computers and computational fluid dynamics make it possible to model the flow field and heat transfer phenomena in film cooling of turbine blades. Numerical simulation can provide detailed information about flow fields and constitutes a fast and inexpensive tool for understanding the film cooling process. The present study is motivated by the above need and is devoted to the numerical simulation of film cooling of turbine blades. It is our aim to develop a computational capability for the prediction of film cooling of a turbine blade with a realistic geometry.

1.3 Previous Numerical Studies

Numerical simulation of the film cooling of turbine blades is not new. There are a number of numerical studies in the literature and the first research paper can be traced back to Kacker *et al* (1969). However, most of the early numerical studies were confined to two-dimensional flows with wall jet configurations. Among the three-dimensional studies, Bergeles *et al* (1978) used a partially parabolic numerical scheme to predict the mean velocity and temperature of laminar flow for a single row of inclined holes and for a surface with multiple rows of holes. Later, they used the same method in conjunction with an anisotropic turbulence model to calculate a single jet injected into a crossflow at 90^0 and 30^0. Good agreement with measured results for an inclined injection hole was

obtained over most of the domain with the exception of the region in the vicinity of the jet orifice where the agreement was poor especially for high mass flow ratios. This is partly caused by the assumption of a uniform velocity profile at the jet exit plane. Demuren *et al* (1986) carried out a systematic study for a single row of holes in a flat plate inclined at different angles using the locally-elliptic procedure. As in the treatment of Bergeles *et al* (1978), uniform velocities at the jet exit plane were assumed in order to exclude the coolant orifice ducts from the computational domain. The predicted film cooling effectiveness was in poor agreement with the measured data in some cases, especially for high mass flow ratios. Sathyamurthy and Patankar (1990) investigated discrete-hole film cooling over a flat plate with lateral injection. A laterally periodic parabolic procedure was used in their calculation. Their results indicate that lateral injection can operate at high blowing rates and can achieve better film coverage than streamwise injection.

Computations of film cooling by solving elliptic Navier-Stokes equations have also been reported. White (1981) solved the fully elliptic transport equations for a single injection hole on a flat plate. The novel feature of the method is its use of separate computational domains in the injection pipe and cross-stream regions. Comparison of jet exit profiles with measurements showed good agreement. Jubran (1989) used a commercial CFD code, PHOENICS, to predict the film cooling effectiveness and the velocity field for two rows of holes inclined in the streamwise and spanwise directions over a flat plate. More recently, Zhou *et al* (1993) used a multigrid and a segmentation technique for the prediction of film cooling for the case of discrete holes with lateral injection. With this technique, they were able to use a very fine grid and the solution also extended into the injection hole regions for the vertical injection hole case. The computational results showed good agreement with the experiments.

More recently, computations have been reported for more complex geometries using body-fitted coordinates to approach real blade configurations. This includes the use of

some commercial CFD codes. Leylek and Zerkle (1993) conducted computations for a streamwise inclined orifice on a flat plate. Their solutions include flow in the plenum, coolant tube and cross-stream regions. A curvilinear grid was used to represent the circular injection tube which formed an angle with the blade wall in the streamwise direction. They reported important information on the flow in the coolant hole which contains counter-rotating vortices and local jetting effects. The results showed that for a hole length of about 4 times the hole diameter, there is a large separation region in the film cooling tube which leads to redistribution of the coolant flow at the hole exit. However, the influence of wall curvature was not considered since the computations did not include the curved blade surface.

Garg and Gaugler (1993, 1994) used general body-fitted coordinates to model the complex flow domain around a real turbine airfoil. The fully three-dimensional Navier-Stokes equations were solved together with closure proposed by the Baldwin and Lomax (1978) algebraic turbulence model. The computational domain included the whole curved blade surface but not the coolant ducts. Turbulent (1/7 power law) profiles were specified at the duct exits, in conformity with observations of Leylek and Zerkle (1993). Experimentally measured wall temperatures were specified as boundary conditions at the airfoil surface and wall heat fluxes were calculated. Fair agreement with the experimental results was reported. Their study provided important guidelines for numerical simulations of film cooling for real turbine blades.

Vogel (1994) carried out calculations of turbine flows with film cooling using multi-block body-fitted coordinates. Two cases are reported in Vogel's paper, the first dealing with two-dimensional film cooling with a slot configuration. A curvilinear grid was used to represent the curved leading edge. The second calculation deals with film cooling from a single orifice on the suction side of the blade. Since the calculation deals only with a single streamwise inclined injection orifice, the author was able to use a symmetry

condition through the middle of the injection orifice. The computational domain included the suction side of a blade surface and half of the streamwise injection orifice. The paper focussed on the detailed flow field and did not describe the temperature field.

1.4 Remarks on Film Cooling Studies

Actual measurements provide the most reliable information about a physical process. However, it should be mentioned that there are serious difficulties with measurements in many situations and the measuring instruments are not completely free from errors. In addition, experimental investigations are usually expensive and time-consuming in obtaining the complete flow data. Numerical simulation can furnish detailed information about fluid flow and heat transfer phenomena and provide a fast and inexpensive tool for understanding the film cooling process. However, numerical simulation has not reached a stage where it can be directly applied in engine design. One of the major problems is that there is lack of appropriate methods to model the cooling process for the complex geometry of a turbine blade. Therefore the majority of the previous calculations were limited to simplified geometries.

The complex configuration of an actual turbine blade presents a number of difficulties for numerical simulation. First, the flow region includes a curved blade surface and a number of circular injection holes for which an ordinary coordinate system, such as Cartesian or cylindrical coordinates can not be applied. Secondly, the flow occurs in different flow regions: a main flow region around the blade surface and a number of sub-regions in the small coolant injection holes for which ordinary one-segment methods are ineffective. Thirdly, the diameters of the injection holes are very small compared with the characteristic dimension of the main flow region. When a fairly dense grid is used in the injection holes, the number of grid nodes needed overall will be too large to be used

with the present computer capacities without using excessively nonuniform grids. The use of a very non-uniform and dense grid near the cooling orifices can cause additional convergence difficulty. Therefore, a solution method should be developed to address both the convergence and geometrical difficulties.

The use of simplified geometries, such as a flat plate with discrete wall jets, can provide useful flow information about the complex cooling process. However, the curvature of a real blade is not accounted for in such a model; consequently the film cooling might be significantly miscalculated. Due to the shape of turbine passages, the flow through them experiences very strong streamline curvature. This curvature can significantly influence the cooling process. Furthermore, as reported by Leylek and Zerkle (1993), the velocity profile at the orifice exit is very complex and is neither uniform nor in alignment with the coolant tube. Our study also showed that the velocity, temperature and $k-\epsilon$ profiles at a coolant-hole exit plane strongly depend on the hole location and the mass flow ratio. Thus, an assumption of constant or power-law solution profiles may be very inaccurate. Therefore, in order to obtain accurate flow and heat transfer data, the simulation should include both the curved blade surface and the coolant ducts.

1.5 Objective and Scope

The present work has two main objectives. (1). One objective is to address the need discussed above to develop an efficient numerical tool suitable for modeling the film cooling process over realistic physical geometries. (2). The other objective is to use this numerical tool to study film cooling of the UBC experimental turbine blade for which considerable measured data has been obtained. This provides not only a validation of the experimental results but also a detailed flow and temperature field.

The first objective involves both code and method developments. To address the difficulties associated with complex geometries and convergence problems, several numerical techniques are used including grid generation, curvilinear coordinate-based calculation, multigrid method and domain segmentation. Two computer codes are developed. One is a 3D curvilinear coordinate-based CFD code, called CMGFD, which can be used to calculate laminar/turbulent flows in arbitrary geometries using non-structured curvilinear grids. The other is a multigrid elliptic grid generation code, called MBEGG, which can be used to generate multi-block curvilinear grids for the CMGFD code.

The CMGFD code is developed based on an existing CFD code, MGFD, developed by Nowak (1992) in our research group. The MGFD code is a comprehensive 3D numerical code with multigrid and domain segmentation techniques. However, its application is still limited to rectangular type geometries since it is based on the Cartesian coordinate system. To address the present need for the prediction of film cooling of turbine blades, the CMGFD code is developed using block-structured curvilinear grids. All the techniques used in the MGFD code, such as the multigrid method and domain segmentation feature, are generalized to the CMGFD code. It should be mentioned that due to the comprehensive nature of the MGFD code, substantial work was required to implement the curvilinear grid-based calculation. In addition to changing the basic grid system, other significant changes, such as inclusion of the multigrid method, are also made.

The CMGFD code involves several numerical techniques. These techniques are implemented in the code in four steps. In each step, considerable effort is involved in developing efficient methods suitable for the code. A detailed investigation of the proposed method and validation of the code is carried out. The development of these methods comprises one of the important tasks in the present study and is described in the following paragraphs.

In the first step, the calculation of laminar flow in complex geometries using general

curvilinear grids is investigated. This step is necessary to develop an efficient, curvilinear coordinate-based finite-volume method while avoiding the additional complexity of turbulence modeling and domain segmentation treatments. Finite-volume methods in conjunction with curvilinear grids have been become popular in recent years for flow simulation in complex geometries. However, problems still exist in terms of the accuracy and efficiency of such computations. In the present study, a finite volume method using general curvilinear grids is proposed to overcome some of these difficulties. The coordinate invariant conservation equations and the physical geometric quantities of the control cells are used directly to formulate the numerical scheme without reference to the commonly-used covariant and contravariant vectors. The definitions of the physical geometrical quantities and the tangential velocity unknowns are introduced. A new difference scheme for the non-orthogonal diffusion and pressure terms is proposed. This scheme contributes to the main diagonal terms in the resulting coefficient matrix and allows an implicit treatment of the non-orthogonal quantities without increasing the number of computational molecules. A coupled equation solver is used in place of the commonly-used pressure-correction equation associated with grid non-orthogonality. The developed method is implemented in the CMGFD code. Several two- and three-dimensional laminar flows are computed and compared with other numerical, experimental and analytical results to validate the solution method and code. The main work in this step is presented in chapter 2.

In the second step, the method developed in the first step is generalized to turbulent flows. It is well known that turbulent flows are more difficult to handle. This step is intended to increase our knowledge in dealing with turbulent flows in the environment of general curvilinear grids and achieve an efficient numerical method for turbulent flows. The mathematical model adopted for the present study is introduced. The $k - \epsilon$ two-equation model together with the 'wall function' is used to simulate turbulent flows.

The discretization of the $k-\epsilon$ equations and the formulation of the 'wall function' are presented. The treatment of the source terms in the $k-\epsilon$ equations is discussed in detail. The performance of the method is investigated through several three-dimensional turbulent flows to validate the method as well as the CMGFD code. This is presented in chapter 3.

The third step is concentrated on the development of an efficient multigrid method for flow simulation in general curvilinear grids. The multigrid method is an efficient iterative solution procedure which exhibits convergence rates insensitive to grid refinement. It has been widely used in the field of computational fluid dynamics. However, there have been relatively few studies carried out using general curvilinear grids. There are additional difficulties for multigrid acceleration in curvilinear coordinate systems which require careful consideration.

In the present study, a multigrid method for calculating laminar/turbulent flows using general curvilinear grids is developed. First, the multigrid calculation of laminar flows is studied with emphasis on the influence of complex flow boundaries. It is found that the discrete governing equations on different grid levels can become inconsistent in some curvilinear grids due to complex flow boundaries, thus reducing the efficiency of the multigrid algorithm. A novel treatment is proposed to solve this problem. Secondly, the difficulties associated with the multigrid acceleration of turbulent flows are discussed and several techniques which help the performance of the multigrid method are introduced. Particularly, it is found that the formulation of the coarse-grid defect equation using the 'wall function' can cause inconsistency of the governing equations between fine and coarse grids. This problem is discussed in detail and a novel approach is proposed to allow the successful implementation of the multigrid method for turbulent flows. Work in this step is presented in chapter 4.

In the last step, the method developed in the three previous steps is generalized to

non-structured curvilinear grids by using a domain segmentation technique. This technique divides the domain of interest into different sub-domains. Solutions are obtained by iteratively applying the solver described in the previous steps to each sub-domain. The implementation of the domain segmentation strategy in the method using general curvilinear grids is studied. Particular attention is given to the communication between neighbouring sub-domains. The performance of the multi-block method is investigated through several computational examples.

The methods described above have been implemented in the CMGFD code. With the above developments the code can be used to deal with arbitrary complex geometries if proper block-structured curvilinear grids are generated. However, generating appropriate grids is often a difficult task. The existing difficulty in generating appropriate grids is one of the reasons that few numerical studies are carried out over complex three dimensional domains. In the present study an elliptic grid generation code, MBEGG, is developed to support the application of the CMGFD code. Even though the code is initially developed to address the need for the prediction of film cooling of a turbine blade it can be used to generate multi-block curvilinear grids for various complex flows. An elliptic grid generation method is used in the code, which solves three nonlinear elliptic grid generation equations. The source terms in the equations are problem-dependent functions and can be used to control the grid characteristics, such as orthogonality and grid stretching. In the present study, methods to achieve certain grid quantities, such as skewness, through the control functions are implemented in the code. The multigrid method is an ideal solver for the elliptic grid generation equations due to the fully elliptic nature of the equations. In the present study, a multigrid method is developed to solve these equations. A domain segmentation technique is adopted in the code to generate block-structured curvilinear grids. Block-structured curvilinear grids are generated for flows over a cooled turbine blade using the code.

As mentioned earlier, the main objective of the present thesis is to develop an efficient numerical tool for the prediction of fluid flows in complex geometries with emphasis on calculation of film cooling problems. To this end, the developed computational code is applied to study film cooling of a UBC experimental turbine blade model. This model has a semi-circular leading edge with four rows of film cooling orifices positioned symmetrically about the stagnation line. It uses a lateral injection and the cooling orifices are inclined spanwise 30^0 to the turbine blade surface. The computational domain is taken directly from the physical geometry without simplification and includes not only the curved blade surface but also the entrance ducts of the circular coolant holes. The physical domain is segmented into a number of sub-domains and separate curvilinear grids are generated for different flow regions. Grids are generated by the MBEGG code which represents exactly the curved blade surface as well as the circular injection orifices. Computations have been carried out for a low overall mass flow ratio ($M = \frac{\varrho_c U_c}{\varrho_i n f t y U_\infty}$) of 0.52 and a high mass flow ratio of 0.97, where ϱ is density, U is velocity and the subscript c and ∞ indicate the coolant and hot flow respectively. The computational results are compared with the available experimental results.

1.6 Contributions of the Thesis

In summary, the present thesis contributes to both the engineering application and theory of CFD in the following areas:

- A numerical method for computing fluid flow in complex three-dimensional geometries using curvilinear grids is developed. Unlike previous numerical studies, the original coordinate-invariant conservation equations and the physical geometric quantities of the control cells are used directly to formulate the numerical scheme

without reference to the coordinate derivatives of transformation. The physical geometric quantities, including volumes, surface areas and surface normal directions of control cells, are introduced to formulate the governing equations in place of the commonly-used covariant and contravariant vectors. A calculation and interpolation procedure for the physical geometric quantities is developed. This method allows the use of significantly non-smooth grids.

- A new scheme for handling the non-orthogonal terms in the momentum and pressure equations is developed. This scheme contributes to the main diagonal terms in the resulting coefficient matrix and allows an implicit treatment of the non-orthogonal quantities without increasing the number of computational molecules. This treatment of the non-orthogonal terms enhances computational stability and convergence rate.

- The physical tangential velocity components resulting from the velocity expansion in the unit tangent vector basis are proposed as dependent variables in the momentum equations. The discrete momentum equations using the tangential velocity components are obtained by an algebraic manipulation which avoids the tensor expression of Christoffel symbols. A coupled solution procedure is used in place of the complicated pressure-correction equation associated with grid non-orthogonality.

- Numerical simulation of turbulent flows in complex geometries is studied. The $k-\epsilon$ two-equation model together with the 'wall function' treatment is used as the turbulence closure. Discretization of the $k-\epsilon$ equations is presented with particular attention to the linearization of the source terms and calculation of the turbulence energy generation rate. Methods are introduced to enhance the computational stability and facilitate the calculation of the energy generation rate. The formulation of the 'wall function' in general curvilinear grids is developed.

- A multigrid procedure for the computation of three-dimensional laminar/turbulent flows in general curvilinear grids is developed. It is shown that the discrete governing equations can become inconsistent between fine and coarse grids in certain curvilinear grids, as well as when the $k-\epsilon$ model with the 'wall function' is used for turbulent flows. Novel procedures have been proposed to solve these problems. Computational results using fine grids of up to approximately one million grid nodes showed that the developed multigrid algorithm is very efficient and a reduction of up to 99% in CPU time was achieved.

- A domain segmentation strategy in conjunction with general curvilinear grids is developed. This approach allows the treatment of arbitrary three-dimensional geometries using non-structured curvilinear grids. Particular attention has been given for the communication between different sub-domains. The performance of the multi-block method is investigated through several computational examples which show that the developed method has the capability to deal with complex flow domains using multi-block structured curvilinear grids and allows the use of significantly discontinuous grid slopes at the interface.

- A block-structured curvilinear coordinate-based finite-volume code, CMGFD, is developed based on an existing Cartesian coordinate-based CFD code. The new code has the capability to deal with arbitrary geometries by using separate curvilinear grids in different flow regions. There is no restriction for curvilinear grids to be used and no limitation for the number of segments to be used in a flow region. An efficient multigrid method is implemented in the code which promises fast convergence for both laminar and turbulent flows.

- A multi-block elliptic grid generation code, MBEGG, is developed for generating block-structured curvilinear grids in complex geometries. The code uses an elliptic grid generation method which solves three elliptic partial differential equations. A multigrid method is developed to solve these equations which provides an efficient grid generator. Techniques to control grid quantities, such as skewness, are introduced. A multi-block grid generation strategy is adopted to allow the generation of block-structured curvilinear grids in arbitrary geometries. This code has the capability to generate block-structured curvilinear grids in arbitrary 2D or 3D domains.

- Computations are carried out for film cooling of a UBC turbine blade model. Block-structured grids are generated by the developed MBEGG code which exactly represent the inclined, round film-holes and the curved blade surface. Computations over the cooled turbine blade model are carried out for overall mass flow ratios of 0.52 and 0.97. The detailed flow field is obtained which improves our understanding of film cooling. The structure of vortices is investigated numerically. This provides information on the mixing of coolant with hot flow. The predicted results are compared with the experimental results of Salcudean *et al* (1994) and show good agreement. The present computations show that the developed numerical tool has the potential to model the complex cooling process in actual blade geometries.

Chapter 2

A Computational Method Using Curvilinear Grids

In this chapter, a finite-volume method for laminar flow in complex 3D geometries using general curvilinear grids is presented. A new second-order accurate difference scheme for the cross-derivative terms is proposed to describe the non-orthogonal components, allowing parts of those terms to be treated implicitly without increasing the number of computational molecules. The coordinate-invariant conservation equations and the physical geometric quantities of control cells are used directly to formulate the numerical scheme without reference to the coordinate derivatives of transformation. A coupled equation solver is used in place of the complicated pressure-correction equation associated with grid non-orthogonality. Several two- and three-dimensional laminar flows are computed and compared with other available numerical, experimental and analytical results to validate the solution method. Part of the work presented in this chapter has been reported previously in He and Salcudean (1994).

2.1 Introduction

For problems with complex geometries, finite-element methods appear to be the natural choice due to their intrinsic geometric flexibility. Finite-volume/difference methods, however, are well established in computational fluid dynamics and an alternative approach would be to use these methods with an appropriate body-fitted coordinate system. Such methods can be developed based on well-established solution algorithms and numerical codes for the regular geometries.

Curvilinear coordinates, with orthogonal and non-orthogonal grids, have been used extensively for fluid flow in complex geometries and a number of research papers exist in the open literature; for examples, see Maliska and Raithby (1984) and Shyy *et al* (1985). The governing equations are considerably simpler in orthogonal coordinates; however these methods have serious geometric limitations. Orthogonal grids are difficult to generate, especially for three-dimensional domains. For complex three-dimensional geometries, non-orthogonal grids are often necessary because they provide greater flexibility in the distribution of the grid points.

The use of a curvilinear coordinate system can entail a number of difficulties which have to be addressed. First, most grid generation techniques provide discrete grid points rather than the analytic functions of transformation. In such cases, the coordinate derivatives of transformation (commonly called covariant base vectors), $\mathbf{a}_i = (\frac{\partial x}{\partial \xi_i}, \frac{\partial y}{\partial \xi_i}, \frac{\partial z}{\partial \xi_i})$, are usually computed approximately and other metric quantities, such as contravariant base vectors and the Jacobian determinant, are then evaluated from these coordinate derivatives. The definition and calculation of these derivatives, however, becomes ambiguous when a non-smooth grid is used. The procedure used to approximate these quantities is critical and may lead to significant numerical errors or even unrealistic solutions, as demonstrated by Segal *et al* (1992) and Lee *et al* (1992) for example.

Secondly, significant deterioration of the convergence rate of some of the available iterative solution procedures can occur with a non-orthogonal grid, as demonstrated by Peric (1990), especially when the grid is highly non-orthogonal. This is partially due to the explicit treatment of the large non-orthogonal diffusion terms, which can be described analytically by cross-derivative terms such as $\frac{\partial}{\partial \xi_2}(\Gamma g^{12} \frac{\partial u}{\partial \xi_1})$ and $\frac{\partial}{\partial \xi_3}(\Gamma g^{13} \frac{\partial u}{\partial \xi_1})$ in the momentum equations, where Γ is the diffusion coefficient and g^{ij} is the controvariant metric tensor.

The choice of dependent variables in the momentum equations also requires careful

consideration. When Cartesian velocity components are used as dependent variables the conservation of momentum is considered along fixed directions everywhere in the field and no curvature terms appear in the momentum equations. The applicability and performance of the scheme, however, depend on the orientation of the computational grid relative to the reference Cartesian coordinate system. In the staggered approach, zero convective fluxes across the control volume faces may occur when the grid turns by 90^0 and one Cartesian velocity component is stored on each cell face. Grid-oriented velocities can be used as dependent variables in order to eliminate this difficulty. The use of grid-oriented velocity components, however, leads to grid-sensitive curvature terms in the momentum equation. Such terms depend on the second derivative of grid coordinates, and thus are difficult to discretize in a conservative manner, possibly leading to severe inaccuracies as noted by Segal *et al* (1992).

Most of the work reported in the literature uses a variant of the SIMPLE algorithm of Patankar (1980) to couple the pressure and velocity fields. The extension of this technique to non-orthogonal grids results in a complex pressure correction equation, especially for three-dimensional cases (see Peric (1990)).

In the present study a method for the computation of fluid flow in complex geometries is proposed that attempts to solve some of the problems discussed above. Efforts are made to achieve greater flexibility in grid design using significantly non-orthogonal and non-smooth grids. To solve the problem related to the non-orthogonal grid, a new second-order accurate numerical scheme is proposed to describe the cross-derivative terms. This scheme allows implicit treatment of part of the non-orthogonal terms, without increasing the size of the computational molecules. To avoid inaccurate discretization for a non-smooth grid, the physical geometric quantities, i.e., volumes, surface areas and surface normal vectors, are calculated directly and used to formulate the numerical schemes.

These geometric quantities are calculated in such a way as to satisfy the geometric conservation laws as described by Vinokur (1989). When the divergence theorem is used over a control cell, the conservation equation can be expressed accurately for an arbitrary grid using these geometric quantities. Such an approach yields an overall conservative approximation for any grid and provides a better physical understanding of the resulting formulation.

The proposed method uses the physical tangential velocity components as dependent variables in the momentum equations. These variables are the contravariant-type velocity unknowns and are volume flow rates across cell faces with appropriate normalization. Similar to other grid-oriented velocity unknowns, the choice of such dependent variables gives rise to additional curvature terms. These terms can be expressed by Christoffel symbols using tensor notation and discretized directly as done by Segal *et al* (1992), for instance. In the present study, a different approach is followed which avoids the explicit discretization of the second-order coordinate derivatives. A coupled equation solver, combined with a staggered grid approach, is used to solve the momentum and continuity equations directly, eliminating the need for the pressure correction equation.

The proposed method is described by first presenting the derivation of the discretized general governing equations and the numerical scheme for the non-orthogonal terms. This scheme is then proven to be second-order accurate. Next, the treatment of the momentum equations and curvature terms using the tangential velocity components as the dependent variables is described, followed by a description of the overall coupled solution procedure in the curvilinear coordinate system. Finally, four computational examples are presented to demonstrate the capabilities of the method.

2.2 Numerical Formulations

Only laminar flows in single domains are considered in this chapter in order to avoid the complexity of turbulence modeling and multi-domain treatment. The generalization to those more complex cases will be presented in the following chapters. The flow is assumed steady, incompressible and Newtonian. Under such assumptions, the laminar flow is governed by the following Navier-Stokes equations,

$$\varrho \mathbf{u} \cdot \bigtriangledown \mathbf{u} - \bigtriangledown \cdot (\mu \bigtriangledown \mathbf{u}) = - \bigtriangledown p, \tag{2.1}$$

$$\bigtriangledown \cdot \mathbf{u} = 0, \tag{2.2}$$

where ϱ is the flow density, μ is the dynamic viscosity, p is pressure and $\mathbf{u}$ is the velocity vector. Equations (2.1) and (2.2) express the momentum and mass conservations, respectively, for steady incompressible flows. It should be noted that these equations are written in a coordinate-free form independent of coordinate systems.

The governing equations (2.1-2.2) can be transformed to a general curvilinear coordinate system with the aid of tensor calculus. The transformed equations can then be discretized in the transformed rectangular domain. In the present study, a direct discretization method is developed. Instead of using the transformed governing equations, the coordinate-free governing equations are used directly to formulate the numerical schemes in a general curvilinear grid. All the numerical schemes are formulated using the physical geometric quantities without using the commonly-used covariant and contravariant vectors. This method is more desirable for non-smooth grids since the coordinate derivatives, which are not well defined in irregular grids, are avoided.

This section describes the discretization of a general scalar governing equation, which forms the basis for the discretization of the momentum equations. The geometric quantities used in our discretization are described first. The divergence theorem is then used

to integrate the governing equation. Finally, the new discretization method for the non-orthogonal terms is described and the second order accuracy is proven.

2.2.1 Physical Geometric Quantities

For curvilinear coordinates $\xi_i = \xi_i(\mathbf{r})$, where $\mathbf{r} = \mathbf{r}(x_1, x_2, x_3)$, is a vector and x_i are the components in a Cartesian coordinate system, the commonly-used covariant vector and contravariant vector in the literature are defined as $\mathbf{a}_i = \dfrac{\partial \mathbf{r}}{\partial \xi_i}$ and $\mathbf{a}^i = \bigtriangledown \xi_i$ respectively. These vectors and associated metric tensors are required in the transformed governing equations and subsequent numerical formulation. For computations using curvilinear grids, the covariant vectors are usually calculated approximately from the discrete grid points of a given grid. Other quantities required in the numerical formulation such as the contravariant metric tensor and Jacobian determinant are then evaluated from these vectors. In the solution procedure, interpolations are necessary to obtain these quantities in various locations, especially for a staggered grid system. The procedure of calculation and interpolation of these quantities is critical and may violate certain geometric conservation laws (see Vinokur (1989)), leading to significant numerical errors.

In the present study, instead of using coordinate derivatives $\mathbf{a}_i$, the physical geometric quantities, which include cell surface areas, cell volumes and surface normal directions, are used directly to formulate the numerical scheme. A uniform grid with mesh size $\triangle \xi_i = 1$ is assumed in the transformed computation domain whenever the notations of coordinate transformation are used. The geometric quantities used in the present study are illustrated in Figure 2.1. The unit tangent vectors, denoted by $\mathbf{e}_i$ (i= 1, 2, 3), are calculated at the centers of the control volume surfaces and are locally parallel to the coordinate lines ξ_i. These tangent vectors correspond to the normalized covariant base vectors $\mathbf{a}_i$ in the literature. The surface area vectors, denoted by $\mathbf{S}^i$ (i = 1, 2, 3), are defined at the same point as $\mathbf{e}_i$ and are normal to the control volume surfaces, with

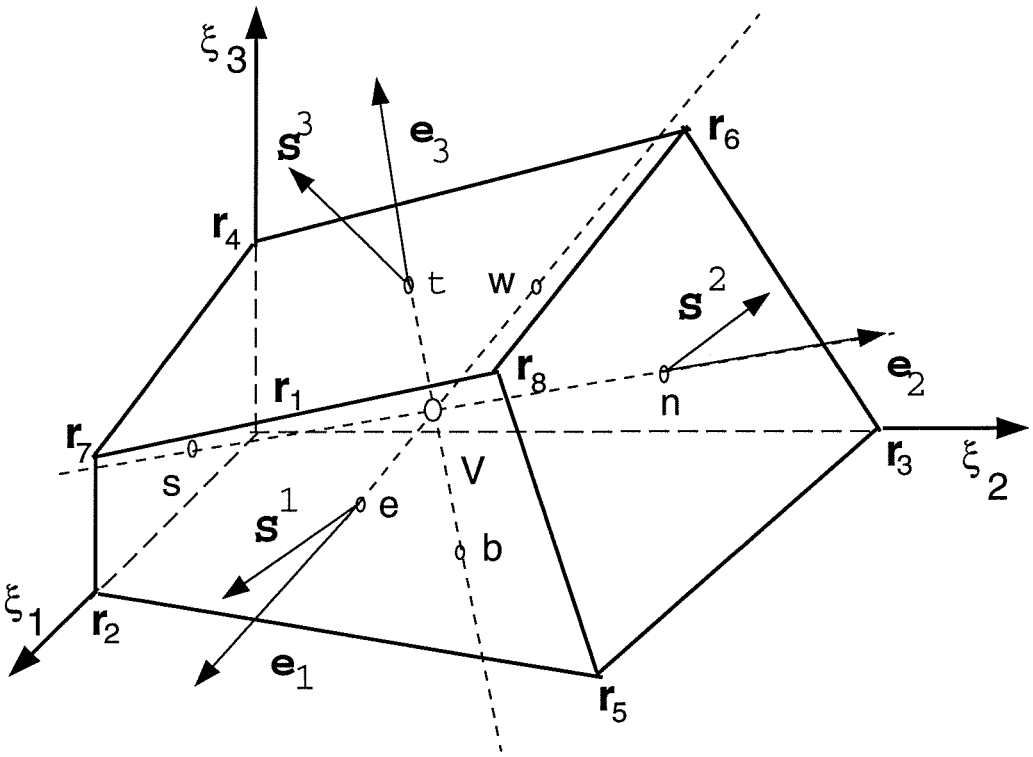

Figure 2.1: Illustration of the physical geometric quantities for a control cell.

magnitude $|\mathbf{S}^i|$ equal to the corresponding surface area. The volume of the control cell is denoted by V.

$\mathbf{e}_i$, $\mathbf{S}^i$ and V are the basic grid quantities and are calculated directly using discrete grid points. For the convenience of formulation, two additional quantities are defined from the above geometric quantities. The non-orthogonal angles, denoted by α_i are defined as the angles between the surface area vector, $\mathbf{S}^i$, and the tangential vector $\mathbf{e}_i$. These angles are a measure of the degree of grid non-orthogonality; for an orthogonal grid these angles are zero. The surface area vectors, $\mathbf{S}^i$ are rescaled and denoted as: $\mathbf{e}^i = \dfrac{\mathbf{S}^i}{|\mathbf{S}^i| cos\alpha_i}$.

2.2.2 Calculation of the Geometric Quantities

In general, the geometric quantities have to be evaluated from discrete grid points. The unit tangent vectors $\mathbf{e}_i$ are calculated by second order accurate averaging (using the center points of two neighboring control cells).

The faces of a control cell are generally surfaces rather than planes, as illustrated in Figure 2.2. In fact, it is not always possible to fit a plane through four points. The surface area vectors and volume of a control cell can be approximated directly by various formulas.

To approximate the surface area vectors and volume of a cell, the following formulas for a triangle and a pyramid are used. A properly oriented surface area vector for a triangular face with vertices $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$, is given by

$$\mathbf{S} = (\mathbf{r}_2 - \mathbf{r}_1) \times (\mathbf{r}_3 - \mathbf{r}_1)/2, \tag{2.3}$$

and the volume for a pyramid with vertices $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$ and $\mathbf{r}_4$ is given by

$$\frac{1}{6}(\mathbf{r}_2 - \mathbf{r}_1) \times (\mathbf{r}_3 - \mathbf{r}_1) \cdot (\mathbf{r}_4 - \mathbf{r}_1) = \frac{1}{3}\mathbf{S} \cdot (\mathbf{r}_4 - \mathbf{r}_1). \tag{2.4}$$

Using equations (2.3) and (2.4) we can use various averaging processes to calculate the geometric quantities for an arbitrary cell having straight line edges. Each polygonal

face is divided into plane triangular faces, and the total volume treated as a sum of tetrahedra. In order to obtain an accurate discretization, the geometric quantities are calculated according to the geometric conservation laws described by Vinokur (1989) and the following formulas for surface area vectors and volume are used:

$$\mathbf{S}_w^1 = (\mathbf{r}_6 - \mathbf{r}_1) \times (\mathbf{r}_3 - \mathbf{r}_4)/2 \tag{2.5}$$

$$V = \frac{1}{3}(\mathbf{S}_w^1 + \mathbf{S}_s^2 + \mathbf{S}_b^3) \cdot (\mathbf{r}_8 - \mathbf{r}_1), \tag{2.6}$$

where the subscripts w, s, b indicate locations, west, south, bottom, respectively (see Figure 2.1), of the vectors. From the two basic vectors, $\mathbf{S}$ and $\mathbf{e}_i$, the quantities $cos\alpha_i$ and $\mathbf{e}^i$ can be calculated.

Noting that $\mathbf{r}_2 - \mathbf{r}_1$, $\mathbf{r}_3 - \mathbf{r}_1$ and $\mathbf{r}_4 - \mathbf{r}_1$ are the approximations of covariant vectors $\mathbf{a}_1 = \frac{\partial \mathbf{r}}{\partial \xi_1}$, $\mathbf{a}_2 = \frac{\partial \mathbf{r}}{\partial \xi_2}$ and $\mathbf{a}_3 = \frac{\partial \mathbf{r}}{\partial \xi_3}$ respectively, we can see $\mathbf{S}^1 \approx \mathbf{a}_2 \times \mathbf{a}_3$, and $V \approx (\mathbf{a}_1 \times \mathbf{a}_2) \cdot \mathbf{a}_3$ from the formulas (1) and (2). Since the volume element $\sqrt{g}$ and contravariant vectors $\mathbf{a}^i$ satisfy formulas $\sqrt{g} = (\mathbf{a}_1 \times \mathbf{a}_2) \cdot \mathbf{a}_3$ and $\mathbf{a}^1 = \frac{\mathbf{a}_2 \times \mathbf{a}_3}{\sqrt{g}}$ (see Thompson *et al* (1985)), we have, $\mathbf{S}^1 \approx \sqrt{g}\mathbf{a}^1$ and $V \approx \sqrt{g}$. The volume element $\sqrt{g}$ is the inverse of the Jacobian determinant.

The geometric quantities at the positions described in section 2.2.1 are calculated from known discrete points. For other locations, the values for $\mathbf{e}^i$ and V are computed by a second order accurate interpolation and the unit tangent vectors are calculated from $\mathbf{e}^i$ by orthogonal relations, $\mathbf{e}^i \cdot \mathbf{e}_j = 0$ $(i \neq j)$, instead of calculation from $\mathbf{e}_i$ by averaging.

2.2.3 Discretization

The general transport equation for a dependent variable ϕ is written as follows:

$$\nabla \cdot (\varrho \mathbf{u} \phi) - \nabla \cdot (\Gamma \nabla \phi) + C = 0, \tag{2.7}$$

where $\mathbf{u}$, ϱ and Γ, are the velocity, density and diffusion coefficient respectively. The first term represents transport of variable ϕ by convection and the second term represents

transport by molecular diffusion. The last term, C, is a source term which generally depends on ϕ. This equation is the natural expression of the transport of variable ϕ and takes different forms in various coordinate systems. The discretized equations were obtained by first integrating equation (2.7) over a control volume in the physical space, and then deriving an algebraic approximation to this integral equation. Geometric quantities associated with the control volume are used directly to evaluate the fluxes on the control cell surfaces.

Let $\mathbf{J} = \varrho \mathbf{u}\phi - \Gamma \bigtriangledown \phi$ denote the total flux. Equation (2.7) can be then written as

$$\bigtriangledown \cdot \mathbf{J} + C = 0 \tag{2.8}$$

Equation (2.8) is integrated over a general control volume in the physical space, δV, as shown in Figure 2.2, and the divergence theorem is applied,

$$\begin{aligned} \iiint_{\delta V} \bigtriangledown \cdot \mathbf{J} dv &= \oint_{\delta S} \mathbf{J} \cdot d\mathbf{S} \qquad (2.9) \\ \approx \mathbf{J} \cdot \mathbf{S}^1|_e - \mathbf{J} \cdot \mathbf{S}^1|_w + \mathbf{J} \cdot \mathbf{S}^2|_n - \mathbf{J} \cdot \mathbf{S}^2|_s + \mathbf{J} \cdot \mathbf{S}^3|_t - \mathbf{J} \cdot \mathbf{S}^3|_b \end{aligned}$$

Next consider transport of variable ϕ along the ξ_i coordinate direction:

$$\begin{aligned} \mathbf{J} \cdot \mathbf{S}^i &= \varrho \phi \mathbf{u} \cdot \mathbf{S}^i - \Gamma \bigtriangledown \phi \cdot \mathbf{S}^i \qquad (2.10) \\ &= \varrho \phi F^i - \Gamma \bigtriangledown \phi \cdot \mathbf{S}^i \end{aligned}$$

where

$$F^i = \mathbf{u} \cdot \mathbf{S}^i, \ (i = 1, 2, 3)$$

are the volume flow rates across cell faces. By the definition of gradient,

$$\bigtriangledown \phi = (\oint_S \phi d\mathbf{S})/V$$

Applying Gauss divergence theorem, we obtain

$$(\bigtriangledown \phi)_P = [\mathbf{S}^1(\phi_e - \phi_w) + \mathbf{S}^2(\phi_n - \phi_s) + \mathbf{S}^3(\phi_t - \phi_b)]/V \tag{2.11}$$

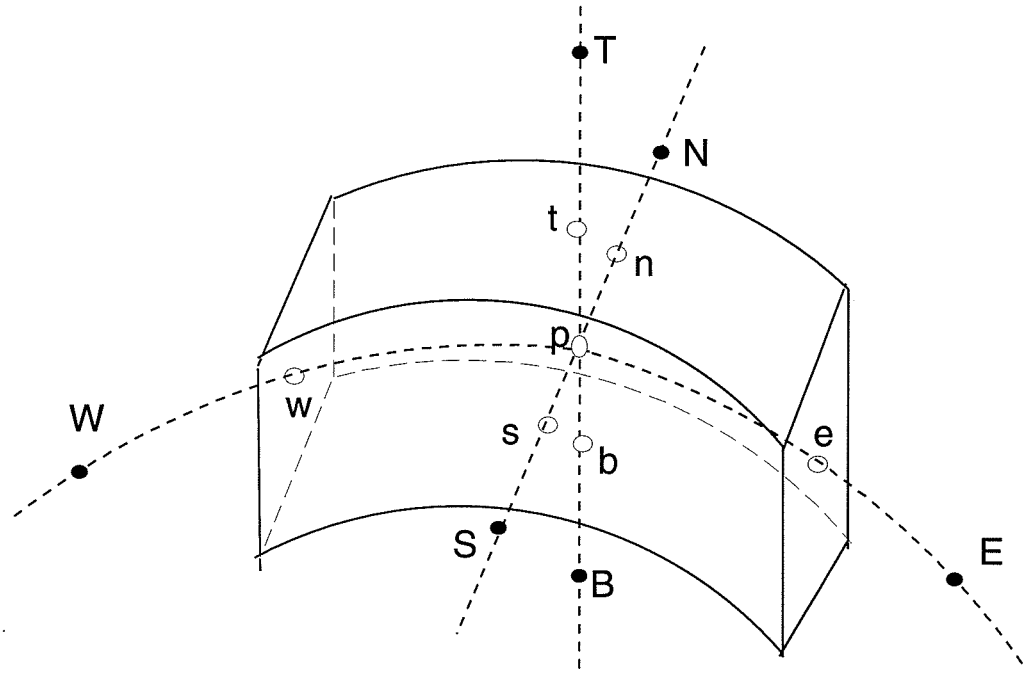

Figure 2.2: A general control volume. Nodes to the north, south, east, west, top and bottom are represented by N, S, E, W, T and B respectively. Lower case denotes control volume surfaces.

where $\phi_e - \phi_w$, $\phi_n - \phi_s$ and $\phi_t - \phi_b$ are approximations of $\frac{\partial \phi}{\partial \xi_1}$, $\frac{\partial \phi}{\partial \xi_2}$, and $\frac{\partial \phi}{\partial \xi_3}$, respectively, since $\triangle \xi_i = 1$ in the transformed space. Hence, the gradient of the dependent variable ϕ can be expressed as:

$$\nabla \phi = \frac{1}{V}(\mathbf{S}^1 \frac{\partial \phi}{\partial \xi_1} + \mathbf{S}^2 \frac{\partial \phi}{\partial \xi_2} + \mathbf{S}^3 \frac{\partial \phi}{\partial \xi_3}) \tag{2.12}$$

Substituting equation (2.12) into (2.10) we obtain the following expression for the transport of variable ϕ:

$$\mathbf{J} \cdot \mathbf{S}^i = \varrho \phi F^i - \Gamma g^{ii} \frac{\partial \phi}{\partial \xi_i} - \sum_{i \neq j} \Gamma g^{ij} \frac{\partial \phi}{\partial \xi_j} \tag{2.13}$$

where

$$g^{ij} = \frac{\mathbf{S}^i \cdot \mathbf{S}^j}{V}, \ (i, j = 1, 2, 3)$$

is the surface area metric tensor. When the grid is orthogonal, $g^{ij} = 0$ for $i \neq j$, therefore, the last term in (2.13) is the result of grid non-orthogonality. The total transport of variable ϕ can be decomposed into an orthogonal component and a non-orthogonal component:

$$\mathbf{J} \cdot \mathbf{S}^i = J_O^i + J_N^i, \ i = 1, 2, 3 \tag{2.14}$$

where

$$J_O^i = \varrho \phi F^i - \Gamma g^{ii} \frac{\partial \phi}{\partial \xi_i}$$

and

$$J_N^i = -\sum_{i \neq j} \Gamma g^{ij} \frac{\partial \phi}{\partial \xi_j}.$$

Substituting equation (2.14) into (2.9), the conservation equation over the control cell can be written as:

$$(J_O^1 + J_N^1)|_e - (J_O^1 + J_N^1)|_w + (J_O^2 + J_N^2)|_n - (J_O^2 + J_N^2)|_s \tag{2.15}$$

$$+(J_O^3 + J_N^3)|_t - (J_O^3 + J_N^3)|_b + V \cdot C = 0.$$

The orthogonal components, J_O^i, have the same form as for the Cartesian coordinate system. Therefore, schemes such as the hybrid scheme, or the power-law scheme of Patankar (1980) for regular geometries can be applied to these terms. The non-orthogonal components, however, cause some numerical difficulties. Ordinary discretization of the non-orthogonal terms, involving only the corner points, results in a 19-diagonal coefficient matrix which is not unconditionally diagonally dominant. This may result in numerical instability and unrealistic solutions. Direct solution methods are impractical for this 19-diagonal coefficient matrix as demonstrated by Braaten and Shyy (1986). Hence, the common practice is to treat the non-orthogonal components of the equation explicitly by combining them into the source term. This may cause serious deterioration in the convergence rate if these explicit components become large. It is therefore necessary to investigate other treatments. In the present study, a new numerical scheme is proposed to address this difficulty and is presented in the following section.

2.2.4 Discretization of the Nonorthogonal Fluxes

Referring to Figure 2.3, consider the approximation of one of the non-orthogonal terms, $g^{12}\Gamma\frac{\partial\phi}{\partial\xi_2}$, at the mid-point of the east surface. When the central difference scheme is used, there is no contribution to the main diagonal terms of the resulting coefficient matrix. Also, four additional corner points are involved, with at least two negative coefficients in the resulting discretization for the cross-derivative term, $\frac{\partial}{\partial\xi_1}(g^{12}\frac{\partial\phi}{\partial\xi_2})$, making an implicit treatment difficult. The following alternative approximation was introduced to solve this problem:

If $g^{12}|_e \leq 0$, then

$$\frac{\partial\phi}{\partial\xi_2}|_e = (\phi_{NE} - \phi_E + \phi_P - \phi_S)/2h \tag{2.16}$$

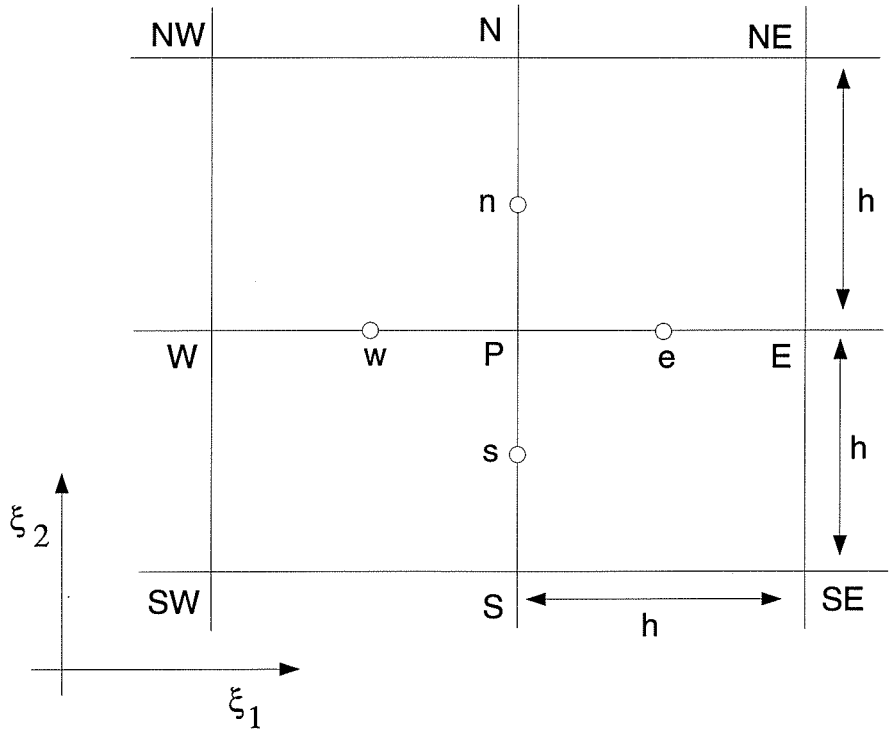

Figure 2.3: Illustration of discretization of cross-derivatives.

and if $g^{12}|_e > 0$, then

$$\frac{\partial \phi}{\partial \xi_2}|_e = (\phi_N - \phi_P + \phi_E - \phi_{SE})/2h \tag{2.17}$$

Different numerical schemes are chosen depending on the sign of the coefficient g^{12}, based on the preference for a positive contribution to the main diagonal term. The idea is similar to the upwind scheme for the convection terms in which different expressions are employed for different signs of the velocity. In contrast to the one-side difference, this scheme is symmetrical around point e and is second-order accurate. This can be shown through the following order analysis.

With reference to Figure 2.3, the following is obtained using Taylor expansions:

$$\phi_{NE} - \phi_E = \frac{\partial \phi}{\partial \xi_2}|_E h + \frac{1}{2}\frac{\partial^2 \phi}{\partial {\xi_2}^2}|_E h^2 + O(h^3),$$

and

$$\phi_P - \phi_S = \frac{\partial \phi}{\partial \xi_2}|_P h - \frac{1}{2}\frac{\partial^2 \phi}{\partial {\xi_2}^2}|_P h^2 + O(h^3).$$

Combining the above expressions yields,

$$\phi_{NE} - \phi_E + \phi_P - \phi_S = (\frac{\partial \phi}{\partial \xi_2}|_E + \frac{\partial \phi}{\partial \xi_2}|_P)h + \frac{1}{2}(\frac{\partial^2 \phi}{\partial {\xi_2}^2}|_E - \frac{\partial^2 \phi}{\partial {\xi_2}^2}|_P)h^2 + O(h^3). \tag{2.18}$$

Since

$$\frac{\partial \phi}{\partial \xi_2}|_E + \frac{\partial \phi}{\partial \xi_2}|_P = 2\frac{\partial \phi}{\partial \xi_2}|_e + \frac{1}{4}\frac{\partial^3 \phi}{\partial \xi_1^2 \partial \xi_2}h^2 + O(h^3), \tag{2.19}$$

$$\frac{\partial^2 \phi}{\partial {\xi_2}^2}|_E - \frac{\partial^2 \phi}{\partial {\xi_2}^2}|_P = \frac{\partial^3 \phi}{\partial \xi_1 \partial {\xi_2}^2}|_e h + O(h^2) \tag{2.20}$$

we can write equation (2.16) as

$$\frac{\partial \phi}{\partial \xi_2}|_e = (\phi_{NE} - \phi_E + \phi_P - \phi_S)/2h + O(h^2). \tag{2.21}$$

A similar estimate exists for approximation (2.17):

$$\frac{\partial \phi}{\partial \xi_2}|_e = (\phi_N - \phi_P + \phi_E - \phi_{SE})/2h + O(h^2). \tag{2.22}$$

The above estimates show that the schemes (2.16-2.17) presented here are second-order accurate. The other derivatives involved in the expressions for the non-orthogonal components can be treated similarly. This approach provides a second order approximation for the non-orthogonal term $\frac{\partial}{\partial \xi_i}(g^{ij}\frac{\partial \phi}{\partial \xi_j})$. Also, using this method, the number of corner points used is reduced by half and the main diagonal term of the resulting coefficient matrix, a_P, is augmented by the following amount:

$$2\Gamma(|g^{12}| + |g^{13}| + |g^{23}|).$$

2.2.5 Formulations of the Discretization Coefficients

After treating the non-orthogonal diffusion as described above, the general governing equation can be discretized following the methods developed for standard Cartesian coordinates. In the present study, the Power Law profile of Patankar (1980) is used for the orthogonal fluxes which ensures the central difference operator at low Peclet number and gradually changes to upwind differencing at high Peclet numbers.

A general algebraic equation is obtained by substituting the discretized forms of the orthogonal and non-orthogonal fluxes into equation (2.15). The resulting equation can be written as

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + a_T\phi_T + a_B\phi_B + \sum_{nc} a_{nc}\phi_{nc} + b \qquad (2.23)$$

where the a_E, a_W, etc. denote the combined convection-diffusion coefficients, including the non-orthogonal terms. The summation index "nc" represents the corner points from the discretization of non-orthogonal fluxes (such as ϕ_{WN}, and ϕ_{WS}) and b includes only the source term C.

2.3 Calculation of The Flow Field

2.3.1 Variable Arrangement and The Choice of Velocity Unknowns

A staggered grid arrangement is adopted in which the scalar quantities, such as pressure p, are located in the geometric center of each control cell and the tangential velocity components are located in the middle of the cell surfaces. There are several options for choosing velocity unknowns for the method using non-orthogonal grids. The use of curvilinear velocity unknowns is preferred here since the applicability of such a method is not limited by grid-orientation, which is a requirement for later calculations.

Different sets of velocity unknowns may be chosen. A discussion and review of the various possibilities is given by Rodi *et al* (1989). In the present study, the physical tangential velocity components are used as the dependent variables for the momentum equation. These variables, denoted as $U^{\hat{\xi}_i}$, as defined by Borisenko and Tarpov (1968), are the resulting coefficients from the velocity expansion in the unit tangent basis vectors $\mathbf{e}_i$:

$$\mathbf{u} = U^{\hat{\xi}_1}\mathbf{e}_1 + U^{\hat{\xi}_2}\mathbf{e}_2 + U^{\hat{\xi}_3}\mathbf{e}_3. \tag{2.24}$$

From the above equation and the orthogonal relation of $\{\mathbf{e}_i\}$ and $\{\mathbf{e}^i\}$, the following expression for $U^{\hat{\xi}_i}$ can be obtained:

$$U^{\hat{\xi}_i} = \frac{\mathbf{u}\cdot\mathbf{e}^i}{\mathbf{e}_i\cdot\mathbf{e}^i}. \tag{2.25}$$

From the above formula, it can be seen that the physical tangential velocity components are similar to the physical contravariant velocity components $\mathbf{u}\cdot\mathbf{e}^i$ which were used by Demirdzic *et al* (1987) as the primary velocity unknowns and differ from them only by a factor of $\mathbf{e}_i\cdot\mathbf{e}^i$. Unlike the non-physical contravariant velocity components, these physical velocity unknowns have the same order of magnitude as the Cartesian velocity components which are independent of the mesh size. As pointed out by Demirdzic *et*

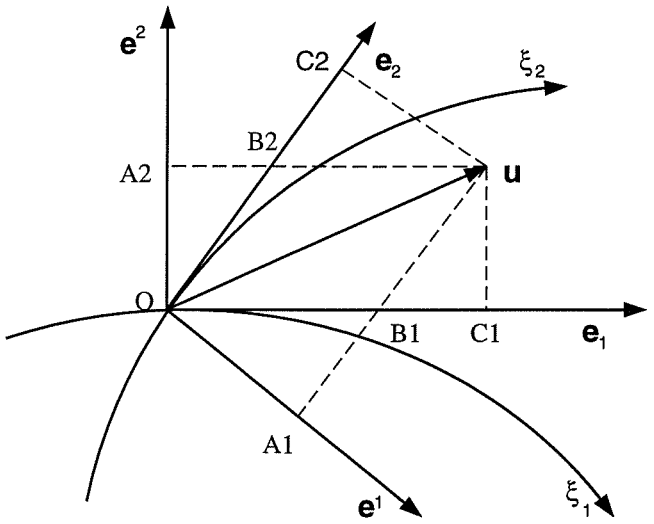

Figure 2.4: Illustration of the three types of physical velocity components in curvilinear grids, OA1 and OA2: physical contravariant velocity components, OB1 and OB2: physical tangential velocity components, and OC1 and OC2: physical covariant velocity components.

al (1987), the use of physical velocity unknowns is preferred since there are drawbacks inevitably associated with the non-physical velocities. There are three different types of physical curvilinear velocities which are illustrated in Figure 2.4. In this figure, the line segments $OA1$ and $OA2$ represent the physical contravariant velocity components and are the velocity projections on the unit surface-normal vectors. The line segments $OB1$ and $OB2$ represent the physical tangential velocity components and are the results of velocity expansion in the tangential vector basis. Segments $OC1$ and $OC2$ represent the physical covariant velocity components and are the velocity projections on the unit tangential vectors $\mathbf{e}_1$ and $\mathbf{e}_2$. The advantage of using the physical tangential velocity components is that if the velocity is tangent to one of the grid lines, the corresponding physical tangential velocity component will become identical with the velocity while other tangential velocity components vanish. From Figure 2.4, it can be seen that only

the physical tangential velocity components have such a property, which reduces the effects of false diffusion when one of the grid lines is aligned with the flow streamline.

From equation (2.25) it can be seen that the physical tangential velocity components are the volume fluxes, $\mathbf{u}\cdot\mathbf{S}^i$, normalized by appropriate geometric quantities. The volume fluxes $\mathbf{u}\cdot\mathbf{S}^i$ were used as velocity unknowns by Rosenfeld *et al* (1988), and are equivalent to the quantities $\sqrt{g}U^i$. As reported by Segal *et al* (1992), the use of quantities $\sqrt{g}U^i$ as velocity unknowns provides a more accurate solution than the use of the contravariant velocity projections U^i.

The tangential velocity components are contravariant unknowns which are equivalent to the quantities $\frac{\sqrt{g}U^i}{|\mathbf{S}^i|cos\alpha_i}$. Unlike the contravariant velocity projections U^i or volume fluxes, U^{ξ_i} are the physical components and their physical lengths are of the same order of magnitude as the velocity. This facilitates the implementation of the boundary condition in programming.

Similar to the use of volume fluxes, the use of the tangential velocity components satisfies the velocity-recovery requirement as described by Segal *et al* (1992). In fact, from the calculations of geometric quantities described in section 2.2.2 and the formula (2.24) and (2.25), it is easy to see that the transformation $\mathbf{u} \rightarrow U^{\xi_i} \rightarrow \mathbf{v}$ gives exactly $\mathbf{v} = \mathbf{u}$ if $\mathbf{u}$ is a constant vector. Computational tests also show that the use of the tangential velocity components as velocity unknowns provides satisfactory performance.

2.3.2 Discretization of Momentum Equations

As with other types of grid-oriented velocity components, the use of tangential velocity components as dependent variables gives rise to additional curvature terms which can be expressed as Christoffel symbols using tensor notation. Since these curvature terms involve second order derivatives of the grid coordinates, they are difficult to discretize accurately when grids become non-smooth. There are two basic approaches for the

discretization of momentum equations for the use of curvilinear velocity unknowns. One method is based on the transformed momentum equations for the curvilinear velocity unknowns. Discretization is carried out in a rectangular domain in the transformed space. The other is based on the discrete momentum equations written in a locally fixed coordinate system. The present study follows the second approach and derives the discretized equations based on algebraic equations for Cartesian velocity components. This approach was first used by Karki and Patankar (1988) for calculations of two-dimensional flows using the physical covariant velocity unknowns as the primary variables in the momentum equations. It seems more desirable for non-smooth grids since the explicit discretization of second order derivatives of the grid coordinate is avoided.

In order to discretize the momentum equations, auxiliary discretizations for the Cartesian velocity components are considered. Suppose that all three Cartesian velocity components are located at the U^{ξ_1} position. The governing equations for Cartesian velocity components can be viewed as special cases of the general governing equation (2.7). The discretization can be obtained according to the method outlined in the last section by integrating the governing equations over the control cell for U^{ξ_1}. Since the Cartesian velocity components are assumed to share the same control cell as for U^{ξ_1}, their discretized equations will have identical coefficients. Therefore, the discretization can be written, using vector notation, as:

$$a_p^{\xi_1}\mathbf{u}_p = a_e^{\xi_1}\mathbf{u}_e + a_w^{\xi_1}\mathbf{u}_w + a_n^{\xi_1}\mathbf{u}_n + a_s^{\xi_1}\mathbf{u}_s + a_t^{\xi_1}\mathbf{u}_t + a_b^{\xi_1}\mathbf{u}_b + \sum_{nc} a_{nc}^{\xi_1}\mathbf{u}_{nc} + \mathbf{b}^{\xi_1}, \qquad (2.26)$$

where the lower case is used for the subscripts instead of the upper case to indicate that the positions of U^{ξ_1} are at cell faces instead of at cell centers. The source term $\mathbf{b}^{\xi_1}$, contains only the pressure gradient term, which is discretized by applying the divergence theorem, as discussed later. The above discretization for the momentum equation is coordinate-invariant and independent of the choice of dependent variables.

To find the discretization using the tangential velocity components as dependent variables, the velocity expansion in the local tangential vector basis at point "p", $\{\mathbf{e}_i|_p\}$, is considered. Taking the inner product of vector $\mathbf{e}_p^1$ and vector equation (2.26) and applying formula (2.25), the following equation is obtained:

$$a_p^{\xi_1} U_p^{\xi_1} = \sum_{nb} a_{nb}^{\xi_1} (U^{\xi_1})'_{nb} + \sum_{nc} a_{nc}^{\xi_1} (U^{\xi_1})'_{nc} + (b^{\xi_1})', \tag{2.27}$$

where

$$(U_{nb}^{\xi_1})' = \mathbf{e}_p^1 \cdot \mathbf{u}_{nb} \qquad (U_{nc}^{\xi_1})' = \mathbf{e}_p^1 \cdot \mathbf{u}_{nc} \qquad (b^{\xi_1})' = \mathbf{e}_p^1 \cdot \mathbf{b}^{\xi_1}, \tag{2.28}$$

the index "nb" represents the six nearest neighbors of the node p, namely, e (east), w (west), n (north), s (south), t (top) and b (bottom), and the index "nc" represents the corner points, as previously defined. The primed velocities, $(U_{nb}^{\xi_1})'$ and $(U_{nc}^{\xi_1})'$, are velocity projections of neighboring velocities over the vector, $\mathbf{e}^1$, at point p. The vector $\mathbf{e}^1$ changes from point to point in the flow field, and therefore, the velocities $(U_{nb}^{\xi_1})'$ generally differ from the actual neighboring variables, $U_{nb}^{\xi_1}$. Since $(U_{nb}^{\xi_1})'$ are not dependent variables, they must be replaced by $U_{nb}^{\xi_1}$. This can be done by rearranging equation (2.27) as follows:

$$a_p^{\xi_1} U_p^{\xi_1} = \sum_{nb} a_{nb}^{\xi_1} U_{nb}^{\xi_1} + \sum_{nc} a_{nc}^{\xi_1} U_{nc}^{\xi_1} + b^{\xi_1,c} + b^{\xi_1} \tag{2.29}$$

where

$$b^{\xi_1,c} = \sum_{nb} a_{nb}^{\xi_1} ((U_{nb}^{\xi_1})' - U_{nb}^{\xi_1}) + \sum_{nc} a_{nc}^{\xi_1} ((U_{nc}^{\xi_1})' - U_{nc}^{\xi_1}). \tag{2.30}$$

Equation (2.30) represents the curvature terms when the tangential velocity components are used as velocity unknowns. Using equations (2.25) and (2.28), equation (2.30) can be rewritten as

$$b^{\xi_1,c} = \sum_{nb} a_{nb}^{\xi_1} (\mathbf{e}_{nb}^1 - \mathbf{e}_p^1) \cdot \mathbf{u}_{nb} + \sum_{nc} a_{nc}^{\xi_1} (\mathbf{e}_{nc}^1 - \mathbf{e}_p^1) \cdot \mathbf{u}_{nc}. \tag{2.31}$$

The above equation shows that the curvature terms for U^{ξ_1} are produced by the change of vector $\mathbf{e}^1$ from point "p" to the neighboring points. Figure 2.5 illustrates the formation of curvature source terms.

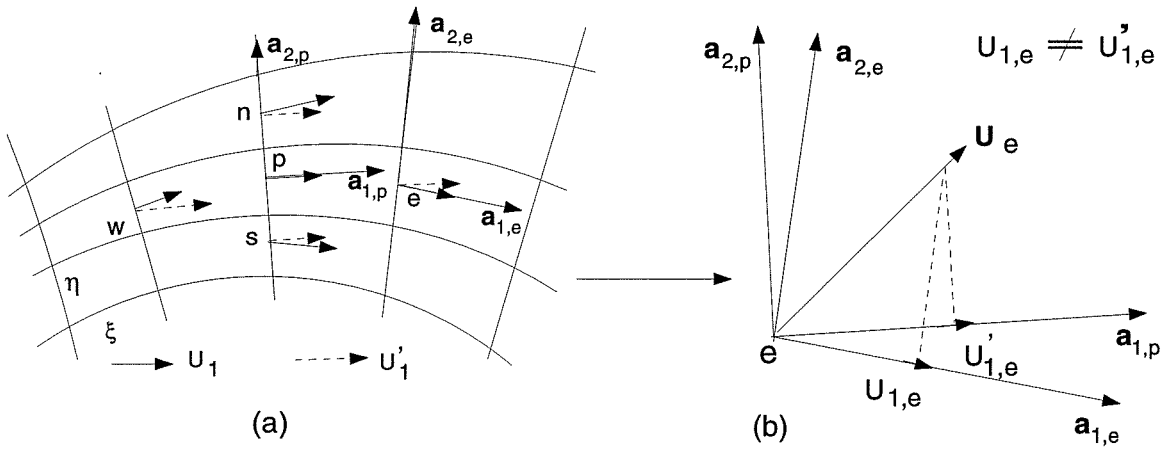

Figure 2.5: Velocity decomposition in a locally fixed coordinate system.

For an actual computation, the velocity $\mathbf{u}$ at neighboring points is calculated from equation (2.24), and the curvature term $b^{\xi_1,c}$ is then calculated from equation (2.31). For a staggered grid the velocity unknowns U^{ξ_2} and U^{ξ_3} are not defined at the U^{ξ_1} position Therefore, some interpolation is needed for the calculation of velocities $\mathbf{u}_{nb}$ and $\mathbf{u}_{nc}$. In the present study, full weighting is used for the interpolation.

The discretized equations for the other two velocity unknowns can be derived in a similar manner. The resulting discretization for the momentum equations, with part of the pressure difference term written explicitly, can be expressed as follows:

$$a_w^{\xi_1} U_w^{\xi_1} = \sum_{nb} a_{nb}^{\xi_1} U_{nb}^{\xi_1} + \sum_{nc} a_{nc}^{\xi_1} U_{nc}^{\xi_1} + b^{\xi_1,c} + A_{\xi_1}(p_P - p_W) + b^{\xi_1} \tag{2.32}$$

$$a_s^{\xi_2} U_s^{\xi_2} = \sum_{nb} a_{nb}^{\xi_2} U_{nb}^{\xi_2} + \sum_{nc} a_{nc}^{\xi_2} U_{nc}^{\xi_2} + b^{\xi_2,c} + A_{\xi_2}(p_P - p_S) + b^{\xi_2} \tag{2.33}$$

$$a_b^{\xi_3} U_b^{\xi_3} = \sum_{nb} a_{nb}^{\xi_3} U_{nb}^{\xi_3} + \sum_{nc} a_{nc}^{\xi_3} U_{nc}^{\xi_3} + b^{\xi_3,c} + A_{\xi_3}(p_P - p_B) + b^{\xi_3} \tag{2.34}$$

where the superscripts indicate the different coefficients associated with different velocity

components and the second summation on the right hand side includes partial non-orthogonal flux terms. The third term represents the curvature body force and the last term contains only pressure terms.

2.3.3 Treatment of Pressure Gradient and Associated Boundary

In equations (2.32-2.34), the pressure term is concealed in the source term. However, the pressure is also an unknown in a flow field and must be found together with the velocity variables. The pressure term, $b^{\xi_i,p}$, in these equations can be obtained from the above derivation of the discrete momentum equations. The $\bigtriangledown p$ can be expressed using the physical geometric quantities as follows,

$$\bigtriangledown p = \frac{1}{V}(\mathbf{S}^1\frac{\partial p}{\partial \xi_1} + \mathbf{S}^2\frac{\partial p}{\partial \xi_2} + \mathbf{S}^3\frac{\partial p}{\partial \xi_3}). \tag{2.35}$$

The pressure source term, $b^{\xi_i,p}$, in equations (2.32-2.34) can be written explicitly as

$$b^{\xi_i,p} = -\frac{1}{(|\mathbf{S}^i|\mathbf{e}^i \cdot \mathbf{e}_i)_p}(g^{ii}\frac{\partial p}{\partial \xi_i} + \sum_{i\neq j} g^{ij}\frac{\partial p}{\partial \xi_j})_p, \tag{2.36}$$

where

$$g^{ij} = \mathbf{S}^i \cdot \mathbf{S}^j, \ \ (i,j = 1,2,3)$$

is the surface area metric tensor. The first term in equation (2.36) is the orthogonal component, and the last term in equation (2.36) is the non-orthogonal component which vanishes for an orthogonal grid. The orthogonal component is discretized naturally using the central difference scheme. For the non-orthogonal component, the scheme proposed for the non-orthogonal diffusion terms is applied. Referring to Figure 2.3, the discretization of $g^{12}\frac{\partial p}{\partial \xi_2}$ at point 'e' takes the following form.

$$g^{12}\frac{\partial p}{\partial \xi_2}|_e = \begin{cases} g^{12}(p_{NE} - p_E + p_P - p_S)/2 & \text{if } g^{12}|_e < 0 \\ g^{12}(p_N - p_P + p_E - p_{SE})/2 & \text{otherwise} \end{cases}$$

With such a discretization, a quantity $|g^{12}|_e(p_E - p_P)/2$ is generated which appears as a direct pressure force to drive the velocity component U^{ξ_1}. This quantity, which is equivalent to an orthogonal component $|g^{12}|_e \frac{\partial p}{\partial \xi_1}/2$, can be treated in a similar manner as the orthogonal term. Such a treatment of non-orthogonal pressure terms will increase the implicit portion in an iteration procedure and contribute to the main diagonal terms in the resulting pressure equation, thereby enhancing the numerical stability and convergence speed.

The discretization of the pressure gradient requires pressure values outside the flow domain. This can be seen from the discretization of non-orthogonal pressure terms presented in the above formulation. For example, the discretization of cross-derivative $g^{12}\frac{\partial p}{\partial \xi_2}$ in the $U^{\xi_1}-$ momentum equation next to the north boundary requires the pressure values beyond this boundary. In the present study, a linear extrapolation is used to calculate the pressure values outside the boundary using the pressures inside the domain.

2.3.4 Discretization of Continuity Equations

Integrating the continuity equation

$$\bigtriangledown \cdot \varrho \mathbf{u} = 0 \tag{2.37}$$

over a scalar control volume, as shown in Figure 2.2 and applying the divergence theorem, we obtain

$$(\varrho \mathbf{u} \cdot \mathbf{S}^1)_e - (\varrho \mathbf{u} \cdot \mathbf{S}^1)_w + (\varrho \mathbf{u} \cdot \mathbf{S}^2)_n - (\varrho \mathbf{u} \cdot \mathbf{S}^2)_s + (\varrho \mathbf{u} \cdot \mathbf{S}^3)_t - (\varrho \mathbf{u} \cdot \mathbf{S}^3)_b = 0. \tag{2.38}$$

Substituting equations (2.24) into the above equation yields the following discretized equation:

$$(c_1 U^{\xi_1})_e - (c_1 U^{\xi_1})_w + (c_2 U^{\xi_2})_n - (c_2 U^{\xi_2})_s + (c_3 U^{\xi_3})_t - (c_3 U^{\xi_3})_b = 0, \tag{2.39}$$

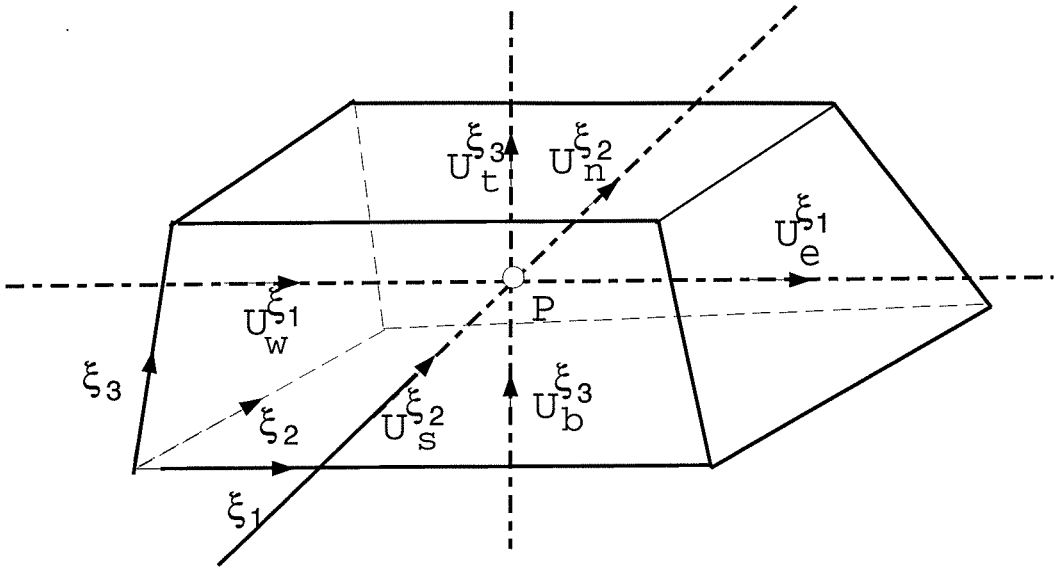

Figure 2.6: A scalar control volume with 7 unknowns.

where

$$c_i = \varrho |\mathbf{S}^i| cos\alpha_i$$

With this formulation, the mass conservation equation is expressed exactly using the tangential velocity components, without any extra source terms.

2.4 Solution Procedure

2.4.1 Coupling of the Velocity and Pressure

The evaluation of the pressure field has always been a difficult issue in the primitive-variable approach for incompressible flow since the pressure is indirectly involved in the continuity equation. This difficulty becomes more acute when a curvilinear coordinate

system is used since expressions for the pressure gradient and the continuity equation become much more complicated. In this section, a block-implicit pressure-velocity coupled solution procedure is presented which essentially updates the velocities and pressure simultaneously. This method was first used by Vanka (1986a) to solve two-dimensional, steady, incompressible flow using a Cartesian coordinate system and was observed to provide good convergence rates.

2.4.2 Symmetric Coupled Gauss-Seidel Iteration Method

The pressure and velocities for a typical control volume are shown in Figure 2.6. The pressure, located at the center, and the velocities, located at the control volume surfaces, are treated as the unknowns, and the pressure and velocities at all the other points are treated explicitly. The momentum equations (2.32-2.34) provide the six algebraic equations for the six unknown velocities. These equations can be simplified by treating only two variables implicitly for each equation and are written as follows:

$$a_{p,w}^{\xi_1} U_w^{\xi_1} - A_w^{\xi_1} p_P = R_1, \tag{2.40}$$

$$a_{p,e}^{\xi_1} U_e^{\xi_1} + A_e^{\xi_1} p_P = R_2, \tag{2.41}$$

$$a_{p,s}^{\xi_2} U_s^{\xi_2} - A_s^{\xi_2} p_P = R_3, \tag{2.42}$$

$$a_{p,n}^{\xi_2} U_n^{\xi_2} + A_n^{\xi_2} p_P = R_4, \tag{2.43}$$

$$a_{p,b}^{\xi_3} U_b^{\xi_3} - A_b^{\xi_3} p_P = R_5, \tag{2.44}$$

$$a_{p,t}^{\xi_3} U_t^{\xi_3} + A_t^{\xi_3} p_P = R_6, \tag{2.45}$$

where the R_i $(i = 1, 2, ...6)$ include all the other terms left in the equations, calculated using the currently-available values of the velocity and pressure at the neighboring points.

This approach is intended to provide a simple algebraic system which can be solved efficiently. The continuity equation (2.39) provides an additional algebraic equation, i.e.,

$$c_{1,e}U_e^{\xi_1} - c_{1,w}U_w^{\xi_1} + c_{2,n}U_n^{\xi_2} - c_{2,s}U_s^{\xi_2} + c_{3,t}U_t^{\xi_3} - c_{3,b}U_b^{\xi_3} = 0. \tag{2.46}$$

These equations are arranged in a block structure as follows:

$$\begin{bmatrix} a_{p,w}^{\xi_1} & 0 & 0 & 0 & 0 & 0 & -A_w^{\xi_1} \\ 0 & a_{p,e}^{\xi_1} & 0 & 0 & 0 & 0 & A_e^{\xi_1} \\ 0 & 0 & a_{p,s}^{\xi_2} & 0 & 0 & 0 & -A_s^{\xi_2} \\ 0 & 0 & 0 & a_{p,n}^{\xi_2} & 0 & 0 & A_n^{\xi_2} \\ 0 & 0 & 0 & 0 & a_{p,b}^{\xi_3} & 0 & -A_b^{\xi_3} \\ 0 & 0 & 0 & 0 & 0 & a_{p,t}^{\xi_3} & A_t^{\xi_3} \\ c_{1,e} & -c_{1,w} & c_{2,n} & -c_{2,s} & c_{3,t} & -c_{3,b} & 0 \end{bmatrix} \times \begin{bmatrix} U_w^{\xi_1} \\ U_e^{\xi_1} \\ U_s^{\xi_2} \\ U_n^{\xi_2} \\ U_b^{\xi_3} \\ U_t^{\xi_3} \\ P_P \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \\ R_4 \\ R_5 \\ R_6 \\ 0 \end{bmatrix}$$

The above block of equations is solved analytically using Gaussian elimination. The procedure is repeatedly applied in the checkerboard order to all cells in the flow field. After sweeping the whole field, the coefficients $a_{nb}^{\xi_i}$, etc., are recalculated and the entire procedure is repeated until the residues become sufficiently small.

2.5 Validation of the Method

The proposed method has been implemented in the CMGFD code. This code was written for general three-dimensional geometries, and can treat both two- and three-dimensional problems. A number of examples are reported in this chapter. They are computed using the CMGFD code to validate the code and the proposed method using non-orthogonal grids.

2.5.1 Cavity flow with inclined side walls

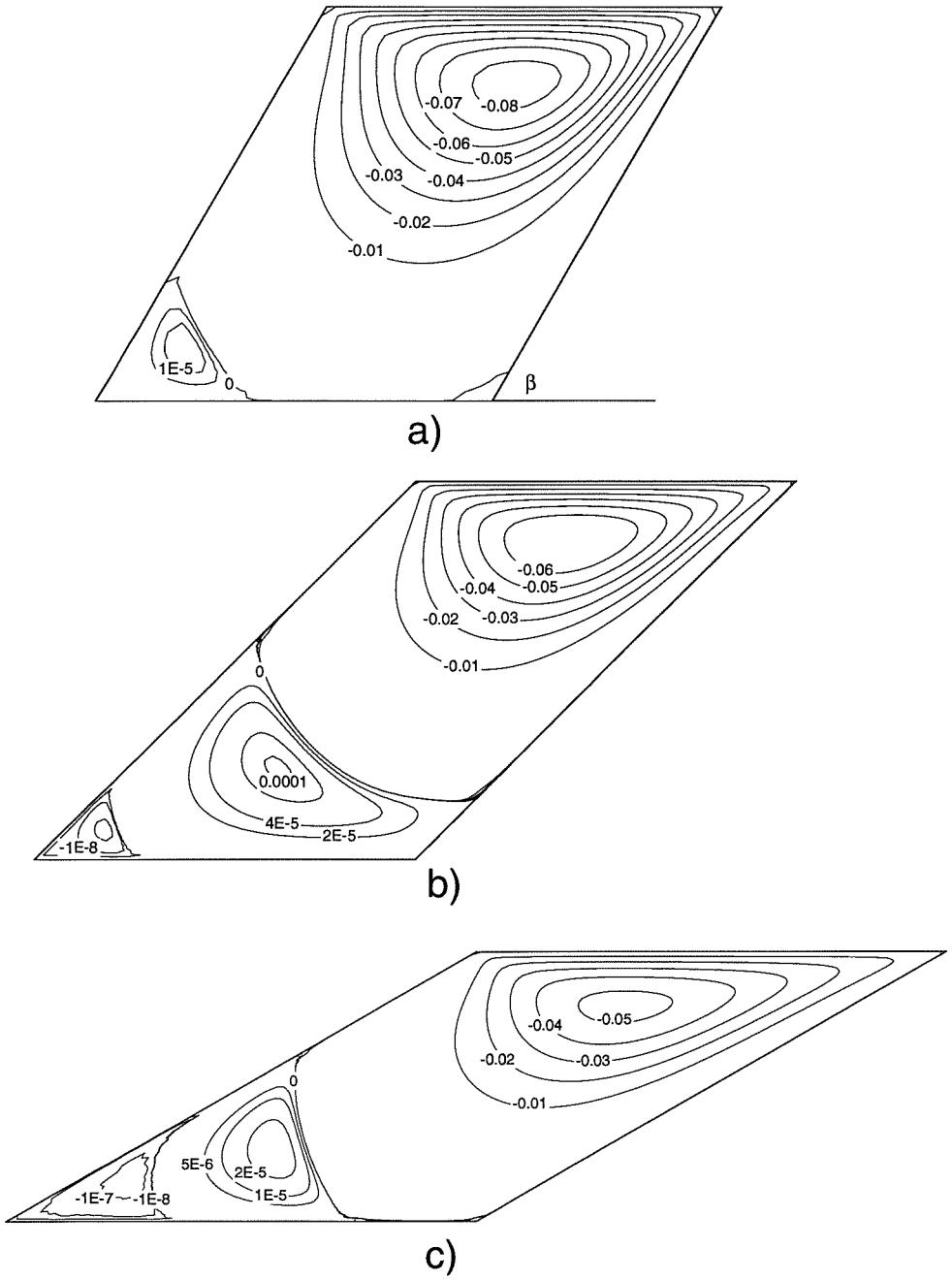

Figure 2.7: Streamlines for flow in cavities with inclined side wall, (a) $\beta = 60°$, (b) $\beta = 45°$, (c) $\beta = 30°$.

Grid	20×20			40×40			80×80		
Inclination β	60^o	45^o	30^o	60^o	45^o	30^o	60^o	45^o	30^o
# of Iterat.	75	85	110	141	160	190	310	396	450

Table 2.1: Number of iterations necessary for convergence.

The flow in a cavity with the top wall moving at a constant velocity, U_w, is often used as a case study for testing the efficiency of a solution algorithm. Here, the side walls of the cavity are taken to be inclined, forming an angle, β, with the horizontal plane. Three cases are computed with inclinations of $\beta = 60^o, 45^o$ and 30^o, at a Reynolds number $Re = 100$, where the Reynolds number is defined as $Re = \frac{\varrho U_w}{\mu}$. Calculations were performed for various grid densities and Table 2.1 lists the convergence rates for each test calculation. The convergence criterion was a maximum residual of less than 10^{-4} for each equation. The momentum residuals were normalized by ϱU_w^2 and the mass residual by ϱU_w. Unlike the results reported by Perić (1990), the present method allows a wide range of under-relaxation factors with fast convergence for significantly non-orthogonal grids. Computational results using the finest grid, 80×80, are plotted in Figure 2.7. They show a strong main vortex, driven by the lid movement, and a sequence of weaker vortices in the sharp corner between the bottom plate and the upstream side wall. This second vortex system is the main difference between the square cavity and cavities with inclined walls. These results are very similar to the previous numerical results of Perić (1990), which used a central difference scheme for non-orthogonal fluxes.

2.5.2 Laminar flow through a tube with a constriction

As the second example, laminar flow through a tube with an axisymmetric constriction was considered. Such flows were studied experimentally by Young *et al* (1973). The

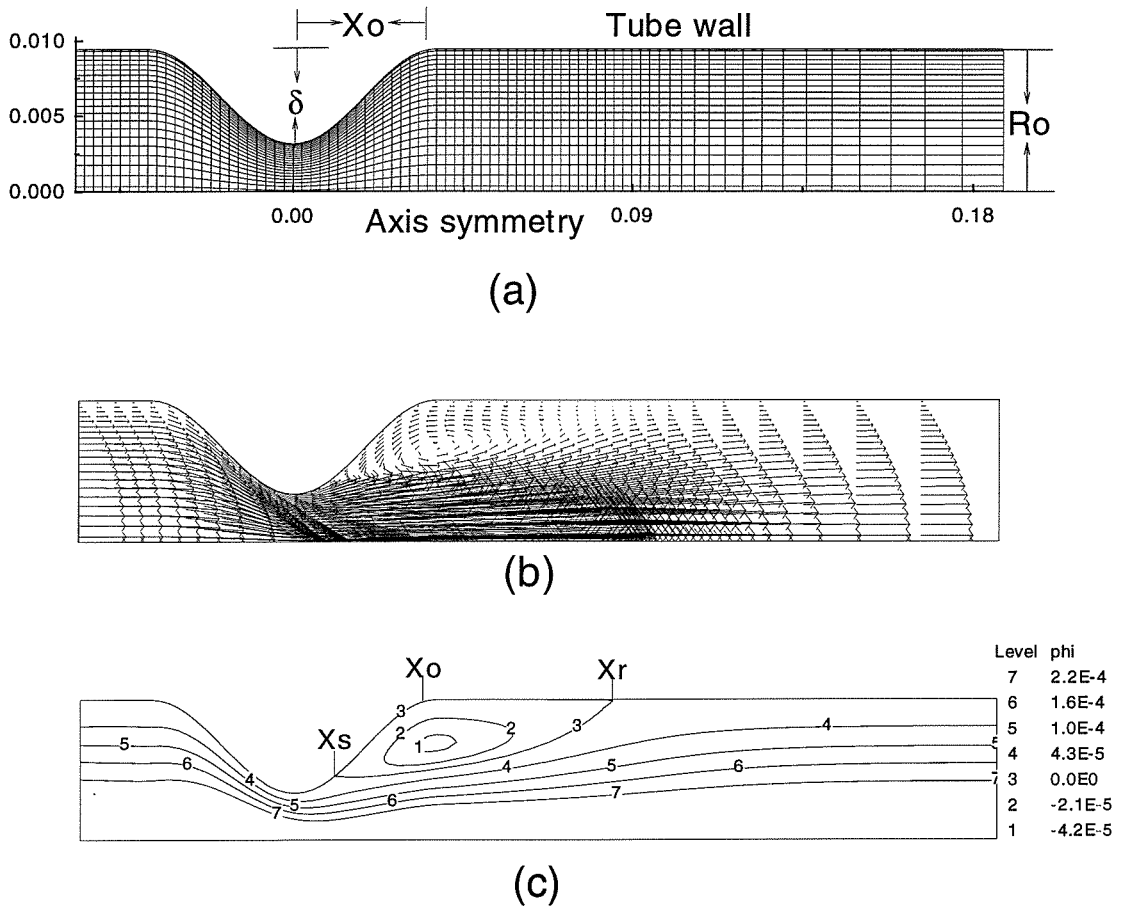

Figure 2.8: Flow through tube with constriction: (a) geometry and grid, (b) velocity field, (c) streamlines.

general shape of the axisymmetric constriction was specified as a cosine curve:

$$\frac{R}{R_0} = (1 - \frac{\delta}{R_0} cos \frac{\pi x}{2X_0}), \quad -X_0 < x < X_0$$

as shown in Figure 2.8a, where the length, X_0, and height, δ, of the constriction are parameters which can be varied for different flow situations.

The present work investigates geometries corresponding to "Model-2" of the experimental work of Young *et al* (1973). For this model, the geometry is specified as

$$R_0 = 9.45 \times 10^{-3} m, \qquad \frac{\delta}{R_0} = \frac{2}{3}, \qquad \frac{X_0}{R_0} = 4.$$

80×20		Separation point		Reattachment point	
Re	# of Iterations	Experiment	Prediction	Experiment	Prediction
50	210	0.33	0.32	2.28	2.27
100	241	0.34	0.33	4.19	4.10

Table 2.2: Comparison with experimental results and numbers of iterations for calculation.

The boundary conditions imposed were no-slip on the wall, zero stream-wise gradient at the outlet, fully developed parabolic flow at the inlet, and symmetry conditions at the axis. An 80×20 grid in x-r coordinates was used for the calculations (see Figure 2.8a). The density of the grid points was higher near the wall than at the centre of the tube, and the grid was stretched in the axial direction, with more grid points in the constricted region. Computations were carried out at Re=50 and 100 where the Reynolds number was defined as

$$Re = \frac{2\bar{U}R_0}{\nu},$$

and $\bar{U}$ is the mean velocity of the inflow. The predicted flow field and streamlines for $Re = 50$ are plotted in Figures 2.8b & 2.8c respectively. In these figures, X_s and X_r are separation and reattachment points respectively. The flow pattern for $Re = 100$ is very similar, except for the size of the recirculation zone.

The predicted separation and reattachment lengths, together with experimental results from Young *et al* (1973), are presented in Table 2.2. Excellent agreement is shown between the present computations and experimental results.

2.5.3 Developing laminar flow in a pipe

For the third computational example, laminar flow through a pipe was considered. This flow can be considered to be two-dimensional because of axial symmetry. However, the

use of a non-orthogonal grid in the present calculation makes the problem appear fully three-dimensional (see Figure 2.9a). This problem was chosen because the available analytical solution makes it possible to test the accuracy of the proposed method. Furthermore, such a grid will be used as part of the mesh for subsequent computations of the film cooling process for turbine blades.

The $16 \times 16 \times 50$ mesh illustrated in Figure 2.9a was generated by solving an elliptic grid generation system of equations.

The three-dimensional, incompressible Navier-Stokes equations were solved, with the no-slip condition at the wall, uniform velocity profile ($w = 1$) at the inlet, and zero stream-wise gradient condition at the outlet. The pipe length was chosen as twelve times the diameter (D) for the computations and the Reynolds number was $Re_D = 100$. Using the same convergence criterion as in the previous problem, a solution was obtained after 480 iterations. Figure 2.9b shows the velocity vectors in the symmetry plane. Figure 2.9c shows the calculated velocity contours in a cross-plane located ten diameters downstream ($x/D = 10$). Figure 2.9d compares the calculated solution in the well-developed region ($x/D = 10$) with the analytical solution $u(r) = 2.0(1 - r)^2$. The result shows good agreement between the analytic and the numerical solutions. Figure 2.9e compares the maximum velocity in the developing region with the results reported by Shah and London (1978) which also shows a good agreement.

2.5.4 Flow in a pipe with a smooth 90 degree bend

For this example, laminar flow in a pipe with a smooth 90° bend was considered. This is a strongly three-dimensional flow, where the main flow in the streamwise direction along the pipe is influenced by strong secondary flows in the pipe cross section, arising from the centrifugal forces due to the bend curvature. This flow was studied experimentally by Enayet *et al* (1982). The same geometry used in the experimental study was adopted

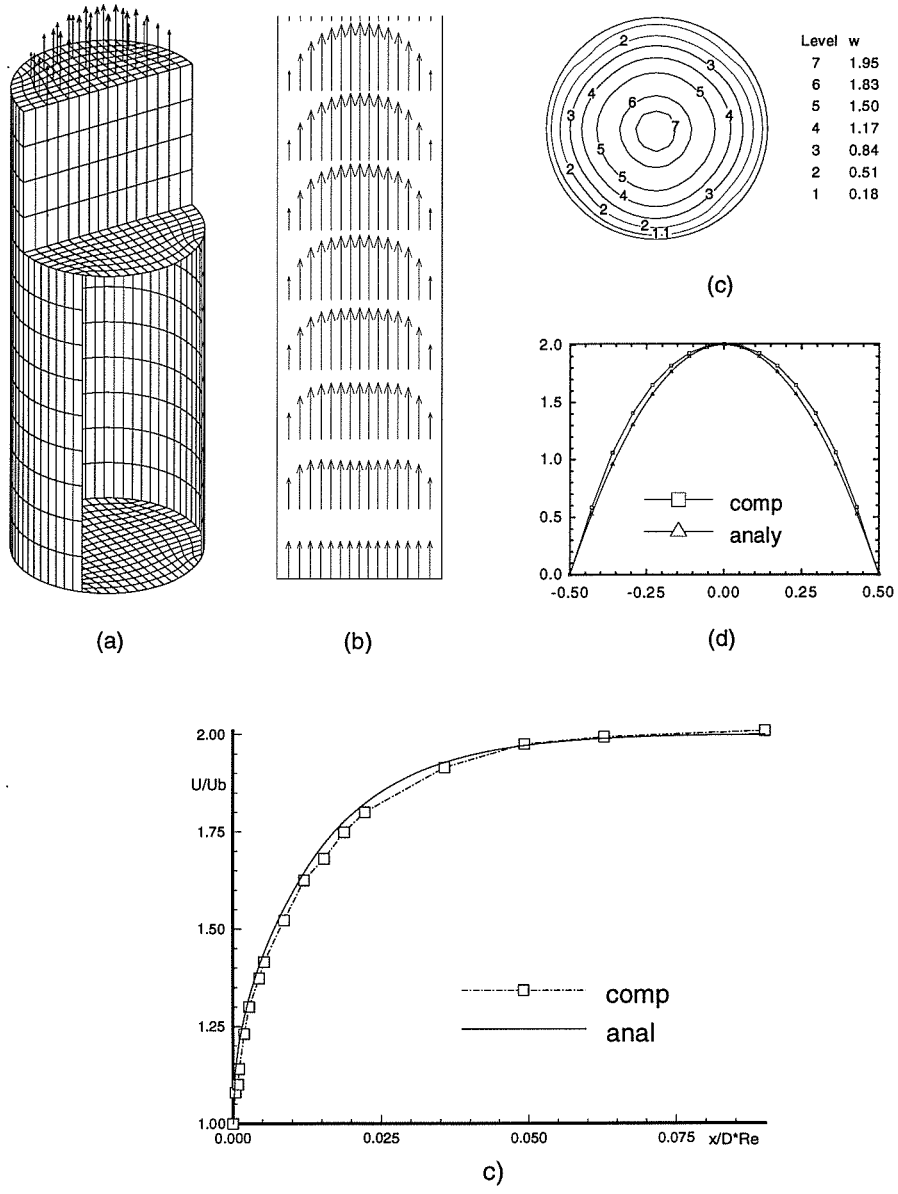

Figure 2.9: (a) Illustration of grid and velocity, (b) development of velocity profile, (c) velocity contour in the cross-pipe plane at $x = 10D$, (d) comparison with analytic solution at $x = 10D$, (e) Comparison for flow in the developing region, where $\frac{U}{U_b} = \frac{U_{max}}{U_{bulk}}$.

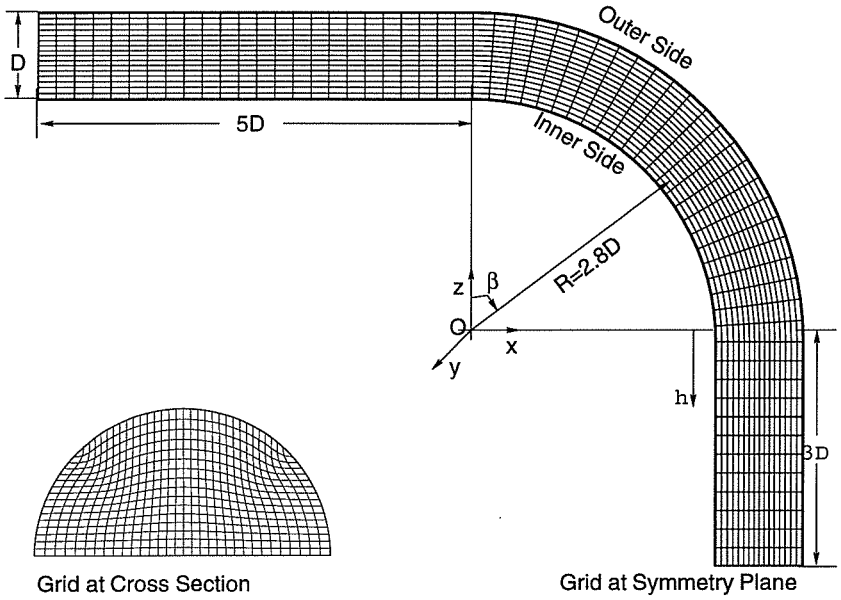

Figure 2.10: Grid and flow geometry for a pipe with a smooth 90° bend.

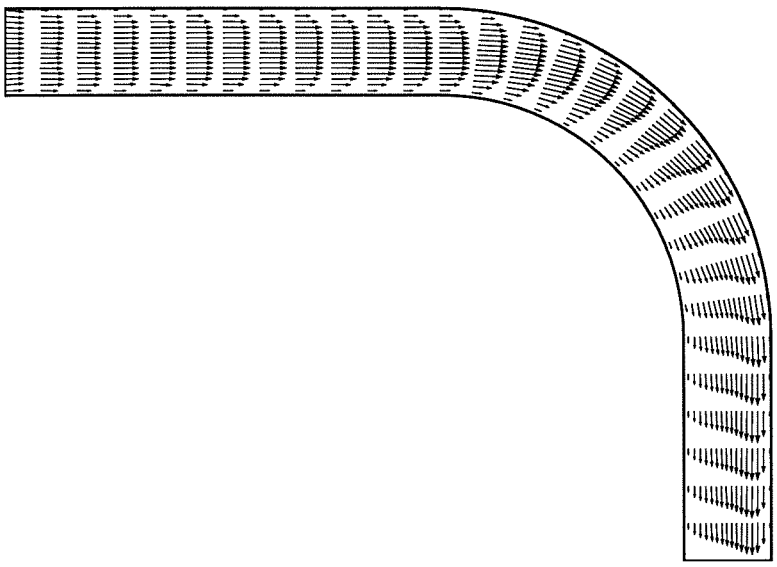

Figure 2.11: Velocity vectors on the symmetry plane.

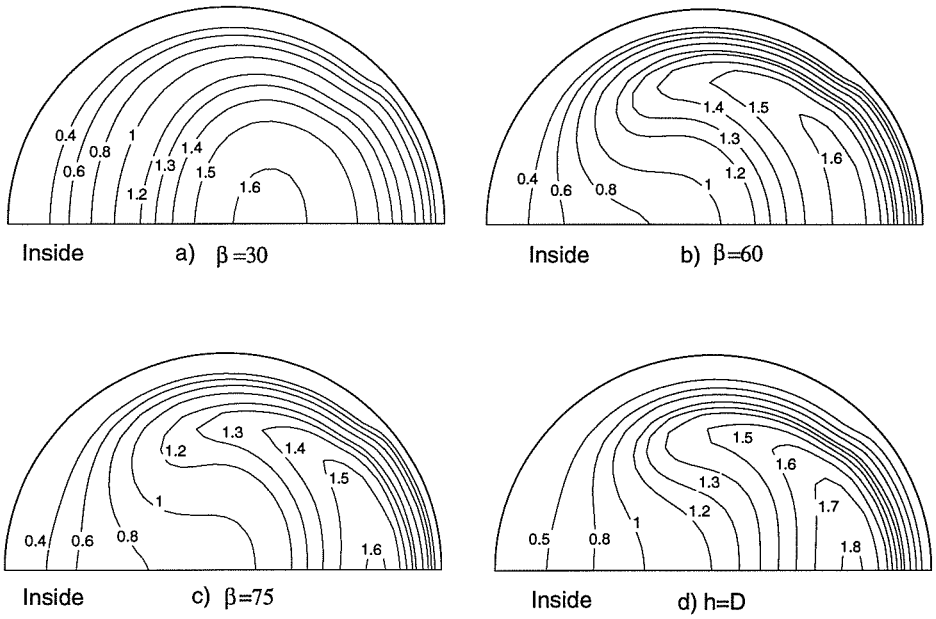

Figure 2.12: Contours of mean velocity U/U_B in the bend.

for the present computation. The geometry and grid are illustrated in Figure 2.10. Due to the symmetry about the x-z plane, the computational domain has been limited to the top half of the pipe only. A $32 \times 16 \times 64$ mesh was used in the computations, which were carried out at $Re = \varrho U_b D/\mu = 500$, corresponding to the experiment, where U_b is the bulk velocity at the entrance and D is the pipe diameter. The convergence criterion was the same as that used in the first example, with convergence satisfied after 620 iterations. Figure 2.11 shows the calculated velocity vectors in the plane of symmetry. Contours of streamwise velocity in four cross-stream locations are plotted in Figure 2.12. The predicted velocity at these same four locations in the plane of symmetry, together with the results measured by Eanyet *et al* (1982), are plotted in Figure 2.13, where reasonable agreement can be seen.

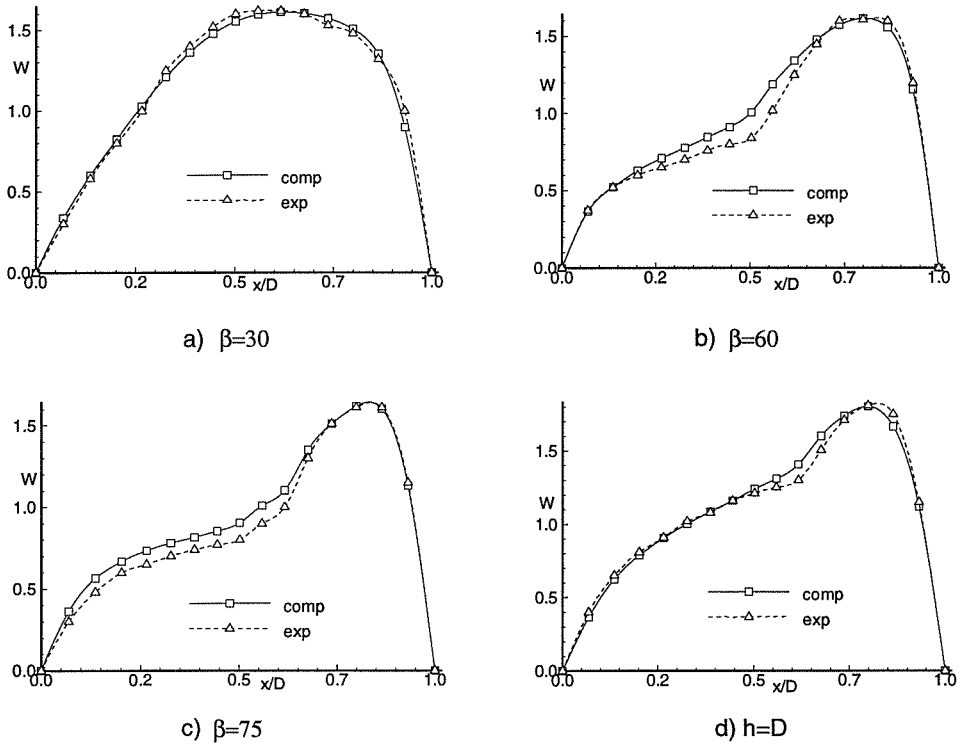

Figure 2.13: Comparison with the experimental results of Eanyet *et al* (1982).

2.6 Closure

A numerical method for computing laminar flows in complex three dimensional domains is presented. Computational examples show that the proposed method can be used to solve flow problems in complex three-dimensional geometries using significantly non-orthogonal and moderately non-smooth grids. A new scheme for handling the non-orthogonal fluxes is proposed which is useful when the grid is significantly non-orthogonal. Efforts have been made to allow for the use of moderately non-smooth grids by directly employing the geometric quantities of control cells and avoiding explicit discretization of the derivatives of the grid coordinates.

Chapter 3

Computation of Turbulent Flows Using Curvilinear Grids

In this chapter, the method developed for laminar flows in the last chapter is generalized to solve turbulent flows in complex three-dimensional geometries. The Reynolds-averaged Navier-Stokes equations together with the $k-\epsilon$ turbulence model are solved to simulate turbulent flows. The 'wall function' is used to take into account the near-wall region where the $k-\epsilon$ model is not valid. The discretization of the $k-\epsilon$ equations and the treatment of the source terms are discussed. Special attention is given to the treatment of wall boundaries. The performance of the numerical method is investigated through several three-dimensional turbulent flows. Comparisons with available experimental results are made which show that the present solution method for turbulent flows is reliable.

3.1 Introduction

In this chapter, the ideas used in chapter 2 are generalized to solve turbulent flows in complex three-dimensional geometries. It is well known that turbulent flow is more difficult to handle than laminar flow. Turbulence modeling is one of the difficult areas in CFD. However, most engineering flows, such as film cooling of a turbine blade, are turbulent. This chapter presents work intended to increase our knowledge of computing turbulent flows in the environment of general curvilinear grids and to develop an efficient solver for turbulent flows.

The 3D, incompressible, time-averaged Navier-Stokes equations are solved. Turbulence closure is attained by the use of the standard $k-\epsilon$ model with the 'wall function' treatment described by Launder and Spalding (1974). The $k-\epsilon$ model is the most widely used turbulence model in the CFD community. Its capability has been demonstrated by many authors. However, the $k-\epsilon$ model also has some shortcomings for turbulent flows with curvature, recirculation, etc. due to anisotropic and non-equilibrium turbulence. It is our aim to show how the $k-\epsilon$ model performs with our solution method for curvilinear grids.

The $k-\epsilon$ turbulence model was derived for fully turbulent flows and is not valid in the near-wall region in which the influence of laminar viscosity is important. In the present study, the 'wall function' described by Launder and Spalding (1974) and others is used to link the equations in the fully turbulent flow and in the near-wall region. The method evaluates the turbulence quantities near the wall by assuming different velocity profiles (log-law or linear) in the near-wall region. The advantage of such a method is that there is no need for excessively fine grids to take into account the steep gradients in the near-wall region. In this chapter, the detailed formulations of the 'wall function' using the grid geometric quantities and the physical tangential velocity components are presented. The velocity boundaries on the wall are implemented by treating the wall shear stress as an auxiliary force in the momentum equations.

For turbulence calculations, the $k-\epsilon$ equations do not generate additional complexity for discretization. They can be viewed as special cases of a general scalar equation and discretized according to the method described in previous chapter. However, the numerical treatment of the source terms in these equations is important and deserves special attention. In this chapter, the treatment of these source terms is reported in somewhat more detail. In non-orthogonal curvilinear grids, the turbulence energy generation rate becomes difficult to calculate from the tangential velocity unknowns. To overcome this

difficulty and facilitate the calculation of the curvature source terms in the momentum equations, the Cartesian velocity components are used as auxiliary dependent variables. The turbulence energy generation rate can be easily calculated using the Cartesian velocity components.

The performance of the developed method is investigated through several three-dimensional turbulent flows, including benchmark computations for developing turbulent pipe flow, and developing turbulent flows in a square duct and a tube, each with a 90^0 bend. Computational results are compared with available experimental results to validate the method.

3.2 Governing Equations

In this section, the governing equations and turbulence model used in the present study are described. The full, Reynolds-averaged Navier-Stokes equations together with the standard $k-\epsilon$ equations are solved for turbulent flows. Using the eddy viscosity concept and the closure of the $k-\epsilon$ two-equation model of Launder and Spalding (1974), the averaged governing equations for steady, incompressible flows can be written in the following coordinate-free form:

$$\varrho \mathbf{u} \cdot \bigtriangledown \mathbf{u} - \bigtriangledown \cdot (\mu_{eff} \bigtriangledown \mathbf{u}) = - \bigtriangledown p, \tag{3.1}$$

$$\bigtriangledown \mathbf{u} = 0, \tag{3.2}$$

$$\bigtriangledown \cdot (\varrho \mathbf{u} k - \frac{\mu_t}{\sigma_k} \bigtriangledown k) = G - \varrho \epsilon, \tag{3.3}$$

$$\bigtriangledown \cdot (\varrho \mathbf{u} \epsilon - \frac{\mu_t}{\sigma_\epsilon} \bigtriangledown \epsilon) = C_1 \frac{\epsilon}{k} G - \varrho C_2 \frac{\epsilon^2}{k}, \tag{3.4}$$

where k is the turbulence kinetic energy, ϵ is the turbulence energy disspation rate, ϱ is the flow density, G is the turbulence energy generation rate, C_1, C_2, σ_k and σ_ϵ are

empirical constants which, following Launder and Spalding (1974), are taken as

$$C_1 = 1.44, \quad C_2 = 1.92, \quad \sigma_k = 1.0, \quad \sigma_\epsilon = 1.3, \quad C_\mu = 0.09.$$

Equations (3.1) and (3.2) represent the momentum and mass conservations, respectively, for steady, incompressible flows, and are the well-known Reynolds-averaged Navier-Stokes equations. Equations (3.3) and (3.4) are the transport equations for turbulence kinetic energy and its dissipation rate respectively. In these equations, $\mathbf{u}$ is the mean velocity vector which is time-averaged from the instantaneous velocity, $\bigtriangledown\mathbf{u}$ is a tensor, and μ_{eff} is the effective viscosity which is given by:

$$\mu_{eff} = \mu_t + \mu_l,$$

where μ_l and μ_t are the laminar and the turbulent viscosities respectively. The turbulent viscosity is evaluated from the relation

$$\mu_t = \varrho C_\mu k^2/\epsilon, \tag{3.5}$$

where C_μ was found empirically to be approximately constant at high Reynolds numbers.

3.3 Discretization of the $k-\epsilon$ Equations

The momentum equations and the continuity equation are same as that for laminar flows presented in Chapter 2 except that the viscosity μ_{eff} in equation (3.1) includes the additional turbulent viscosity. Therefore, the velocity components and pressure can be treated in the same way as in the last chapter. The discretization of equations (3.1-3.2) can be obtained from the equations derived in the previous chapter. Therefore, only the discretization of the $k-\epsilon$ equations requires further consideration.

The $k-\epsilon$ equations (3.3-3.4) can be treated together as a general scalar equation as follows:

$$\bigtriangledown\cdot(\varrho\mathbf{u}\phi - \Gamma\bigtriangledown\phi) + S = 0, \tag{3.6}$$

where the dependent variable, ϕ, is a scalar which designates the turbulence energy k or dissipation rate ϵ. The corresponding diffusion coefficient and source terms in the above equation are

$$\Gamma = \frac{\mu_t}{\sigma_k}, \qquad S = \mu_t G - \epsilon, \tag{3.7}$$

and

$$\Gamma = \frac{\mu_t}{\sigma_\epsilon}, \qquad S = C_1 \mu_t G \frac{\epsilon}{k} - C_2 \frac{\epsilon^2}{k}. \tag{3.8}$$

for the k and ϵ equations respectively.

The discretization of the scalar equation (3.6) has been presented in the last chapter. Therefore, the discretization of the $k - \epsilon$ equations can be obtained without additional difficulties if the source terms can be solved appropriately. There is not much information in the literature regarding the efficient treatment of these source terms. It was found, however, that the treatment of these source terms is critical in terms of computational stability and efficiency. In the present study, the numerical treatment of these terms is reported.

The source terms in the $k - \epsilon$ equations include the turbulence energy generation rate which has to be evaluated from the velocity. The evaluation of the turbulence energy generation rate G becomes complex for computations using curvilinear velocity unknowns. In a Cartesian coordinate system, the generation rate G can be expressed using the derivatives of mean velocity components:

$$G = \mu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \frac{\partial U_i}{\partial x_j}, \tag{3.9}$$

where (U_1, U_2, U_3) are the Cartesian mean velocity components. In a curvilinear coordinate system, the generation rate is usually calculated using the transformational formula. The expression of the turbulence generation rate based on the physical tangential velocity components can be obtained with the aid of tensor calculus. However, the formulation

includes the second-order derivatives of grid coordinates and becomes complex and difficult to calculate. To save computational efforts, Cartesian velocity components are used as auxiliary variables in the solution procedure in addition to the primary velocity variables. Such an approach allows efficient calculation of the turbulence generation rate without using the transformed formulas for the physical tangential velocity components. In addition, the curvature source terms in the momentum equations can be efficiently and accurately evaluated. The disadvantage is the requirement of additional computer storage for these auxiliary variables. This disadvantage can be remedied using a subdomain solution procedure as introduced in a later chapter.

The generation rate can be evaluated directly from the auxiliary Cartesian velocity components as follows. First, the gradients of Cartesian velocity components $\bigtriangledown U_i$ are calculated from the formula for $\bigtriangledown U_i$ presented in equation (2.12). From the gradient $\bigtriangledown U_i$, the derivatives $\partial U_i/\partial x_i$ can be obtained easily and the energy generation rate G can then be calculated from equation (3.9). Such an approach allows efficient evaluation of the energy generation rate.

After calculating the turbulence energy generation rate, the source terms can be treated in the following manner. For the turbulence energy equation, integrating the source term S_k over a control volume δV gives

$$S = \int_{\delta V} (G - \varrho\epsilon) \sim (G - \varrho\epsilon)_p V_p \tag{3.10}$$

where the subscript p indicates that the value is associated with the control-volume center and V_p is the volume of the control cell with center 'p'. There are two approaches which can be used to deal with the source term. One is to move all the quantities in equation (3.10) to the right-hand side in the discrete equation and treat them as source terms in the solution procedure. The other is to split the source term into two different portions by appropriate linearization and to treat one of them implicitly. For the first approach,

however, the source term may become negative. As discussed by Patankar (1980) for an always-positive variable, the source term in the discretization equation should be positive in order to avoid unrealistic solutions. To achieve a positive source term, the dissipation rate ϵ is replaced by $C_\mu \varrho k^2/\mu_t$ using equation (3.5) and the negative term $-G\epsilon$ can then be linearized with respect to the dependent variable k. Therefore, the total source term can be split as follows:

$$S = -S_p k_p + S_c, \tag{3.11}$$

where

$$S_p = (\frac{C_\mu \varrho^2 k V}{\mu_t})_p, \qquad S_c = (GV)_p \tag{3.12}$$

The term S_p positively contributes to the main diagonal term a_P in the discretization of the k-equation, which enhances the numerical stability. The source term of the ϵ-equation can also be linearized in a similar manner according to the always-positive rule described by Patankar (1980).

After calculating the source terms, the final discrete equations for k and ϵ can be written as the following seven-point algebraic equations:

$$a_P^k k_P = a_E^k k_E + a_W^k k_W + a_N^k k_N + a_S^k k_S + a_T^k k_T + a_B^k + b^k, \tag{3.13}$$

$$a_P^\epsilon \epsilon_P = a_E^\epsilon \epsilon_E + a_W^\epsilon \epsilon_W + a_N^\epsilon \epsilon_N + a_S^\epsilon \epsilon_S + a_T^\epsilon \epsilon_T + a_B^\epsilon + b^\epsilon, \tag{3.14}$$

where the subscripts indicate the seven scalar positions, namely the center point 'P' and the six neighbors; the superscripts indicate coefficients associated with k and ϵ.

For a non-orthogonal grid, the source terms in the above two equations contain the non-orthogonal diffusion quantities which may become negative which allows the possibility that the physical positive variables could acquire an erroneous negative value. To solve this problem, an artificial damping quantity is added to both sides of the discrete equations. Taking the $k-$equation as an example, a positive quantity βk_P is added to

both the source term and the main diagonal term at the same time, where β is a positive constant and is chosen depending on the problem to be solved. Therefore, the source term is augmented by an amount βk_P while the coefficient of the main diagonal term, a_P, is increased to $a_P + \beta$. This artificial damping technique is different from the ordinary under-relaxation in the literature. The former increases both the main diagonal and source terms and the latter reduces the change of the dependent variables. Such a practice is found to be very useful for turbulent calculations using non-orthogonal grids.

3.4 Wall Boundaries

The $k - \epsilon$ model is valid only for fully turbulent regions and does not take into account the laminar viscosity effect which is important in the near-wall region. To take this effect into account without using an excessive number of grid points near the wall, the 'wall function' is adopted in the present study.

The 'wall function' has been used in the literature for turbulent flow calculations in curvilinear coordinates. Demirdzic *et al* (1987) presented a calculation procedure for two-dimensional turbulent flows based on the physical contravariant velocity unknowns. A formulation of the 'wall function' for two-dimensional curvilinear grids was also presented in this paper. Agouzoul *et al* (1990) presented a calculation for a three-dimensional turbulent flow using a non-staggered and non-orthogonal grid based on the Cartesian velocity unknowns. To avoid the complication associated with the formulation of momentum equations near the wall for three-dimensional flows, an 'equivalent viscosity' approach was proposed in the paper to take the variations of the velocity in the near-wall region into account. In the present study, the velocity boundary at the wall is implemented by taking the wall shear stress as an auxiliary force in the momentum equations and appropriately altering the flow fluxes on the control-volume surfaces near the wall. The

formulation in a general curvilinear coordinate system based on the physical tangential velocity unknowns and the grid physical geometric quantities is described below.

We consider the formulation at a scalar node 'p' next to the wall. Following Launder and Spalding (1974), two different velocity profiles near the wall are assumed depending on the values of the dimensionless distance y^+,

$$\frac{U_p}{u_\tau} = \frac{1}{\kappa} ln(Ey^+), \text{ if } y^+ > 11.36, \tag{3.15}$$

$$\frac{U_p}{u_\tau} = y^+, \text{ if } y^+ \leq 11.36 \tag{3.16}$$

where the dimensionless distance y^+ is defined as:

$$y^+ = \frac{\varrho y_p u_\tau}{\mu},$$

and the friction velocity is defined as:

$$u_\tau = \sqrt{\frac{\tau_w}{\varrho}}. \tag{3.17}$$

Here, y_p is the normal distance of the node 'p' from the wall, U_p is the velocity component parallel to the wall at the node 'p', and κ and E are empirical constants which are taken as $\kappa = 0.41$, $E = 9.97$ following Launder and Spalding (1974).

The normal distance, y_p, is evaluated directly from a given grid. To calculate U_p, the velocity vector $\mathbf{u}_p$ which is calculated from formula (2.24) is decomposed into the normal vector, $\mathbf{u}_p^n$, and the tangential vector, $\mathbf{u}_p^\tau$, at the wall,

$$\mathbf{u}_p^n = (\mathbf{e}^w \cdot \mathbf{u}_p)\mathbf{e}^w, \qquad \mathbf{u}_p^\tau = \mathbf{u}_p - \mathbf{u}_p^n \tag{3.18}$$

where $\mathbf{e}^w$ is the unit surface normal vector at the wall. Here the magnitude of the tangential velocity vector $\mathbf{u}_p^\tau$ is U_p.

After U_p and y_p are obtained, the only unknown in equation (3.15) or (3.16) is the friction velocity u_τ. Therefore, u_τ can be solved from these equations iteratively using

Newton's method. The values of k and ϵ at the first node from the wall are then specified by the following algebraic expressions:

$$k_p = \frac{u_\tau^2}{\sqrt{C_\mu}} \tag{3.19}$$

$$\epsilon_p = \frac{u_\tau^3}{\sqrt{\kappa y_p}} \tag{3.20}$$

Instead of using the 'no-slip' wall condition, the wall boundary for velocity is implemented by appropriately modifying the flux transport terms at the cell surfaces adjoint to the boundary and then treating the wall shear stress as an auxiliary force in the momentum equations. The force due to the wall shear stress is in the opposite direction to the tangential velocity $\mathbf{u}_p^\tau$ and can be expressed as: $-\tau_w A \mathbf{u}_p^\tau / U_p$, where A is the wall surface area of the control cell which can be obtained from the surface area vector, and the wall shear stress can be obtained from the friction velocity using equation (3.17). This term is treated as a force in the momentum equations and is used to modify the source terms in the discretized equations. Taking the bottom wall as an example, the source terms due to the wall shear stress for the U^{ξ_1} and U^{ξ_2} momentum equations are $-\tau_w \mathbf{u}_p^\tau \cdot \mathbf{e}_p^1 |\mathbf{S}^3| / U_p$ and $-\tau_w \mathbf{u}_p^\tau \cdot \mathbf{e}_p^2 |\mathbf{S}^3| / U_p$ respectively.

3.5 Application to Turbulent Flows

The method described in this chapter has been implemented in the CMGFD. In this section, the code is applied to several three-dimensional problems to investigate the performance of the developed method and the $k - \epsilon$ model.

3.5.1 Developing Turbulent Pipe Flow

In the first example, the turbulent developing flow in a smooth pipe was computed. Such a flow has been studied extensively by many authors. A review paper by Klein (1981)

summarized most of the experimental results for the turbulent developing pipe flow. As pointed out by Klein, developing pipe flow is very complex and, unlike earlier studies which believed that turbulent developing pipe flow was solely determined by the growth of wall-boundary layers, the velocity profile peak in the pipe centerline may reach a maximum and then decrease again.

Since turbulent developing pipe flow has been extensively studied and many reliable results are available, this problem was chosen to test the proposed method. As in the last chapter, a non-orthogonal rectangular type grid (see Figure 3.1a) is used to replace the ordinary cylindrical coordinate grid. Such non-orthogonal grids are very useful for solving many internal flows.

Based on the experimental results for this flow, the pipe length L was chosen as 80 times the diameter D from the inlet, within which fully developed flow is generally achieved. Flows with two different Reynolds numbers, namely $Re = \varrho U_b D/\mu = 10^5$ and 3.0×10^5, are calculated, where U_b is the bulk velocity. At the inlet, a uniform velocity, a constant turbulence kinetic energy, and a constant dissipation rate are prescribed. k and ϵ are evaluated from the following equations:

$$k = 1.5U^2(u/U)^2, \qquad \epsilon = C_\mu^{\frac{3}{4}} k^{\frac{3}{2}}/l \tag{3.21}$$

where u/U is the turbulence intensity and l is the turbulence length scale. Here, the turbulence intensity u/U was taken as 0.02 and 0.04 for $Re = 10^5$ and 3.0×10^5 respectively, and the turbulence length scale was taken as one-tenth of the pipe diameter.

For flow with $Re = 10^5$, the computed velocity profile at the centerline, together with experimental results from Richman *et al* (1973) are plotted in Figure 3.2a. A velocity peak was found at about 29 pipe diameters from the entrance which differs from the measurements. However, the overall agreement is reasonable if one notices that the large velocity scale exaggerates the difference which is under 6%. Further, this phenomenon

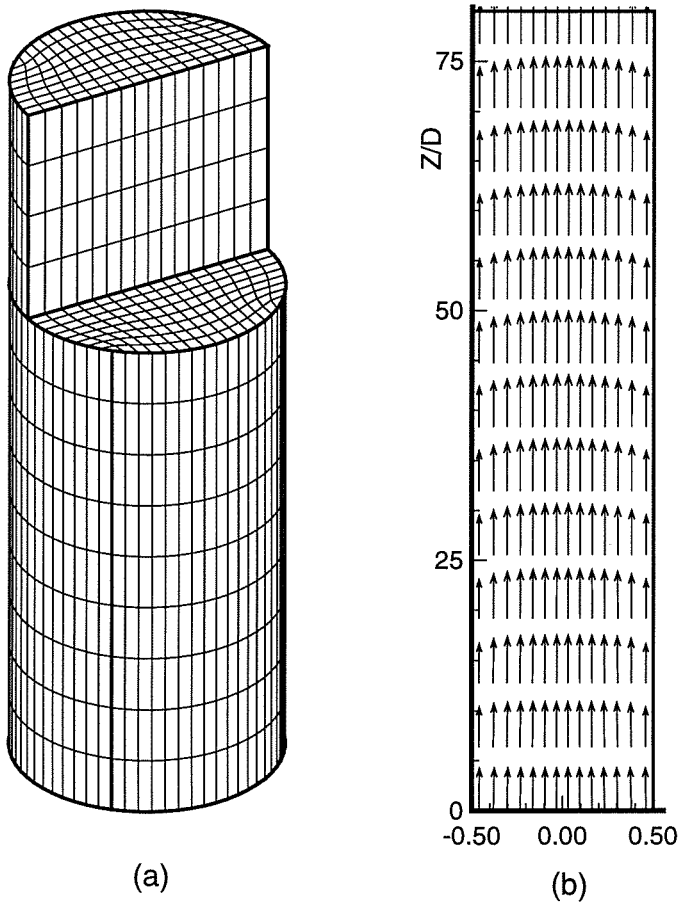

Figure 3.1: (a) The non-orthogonal rectangular type grid. (b) Predicted velocity field at the symmetry plane.

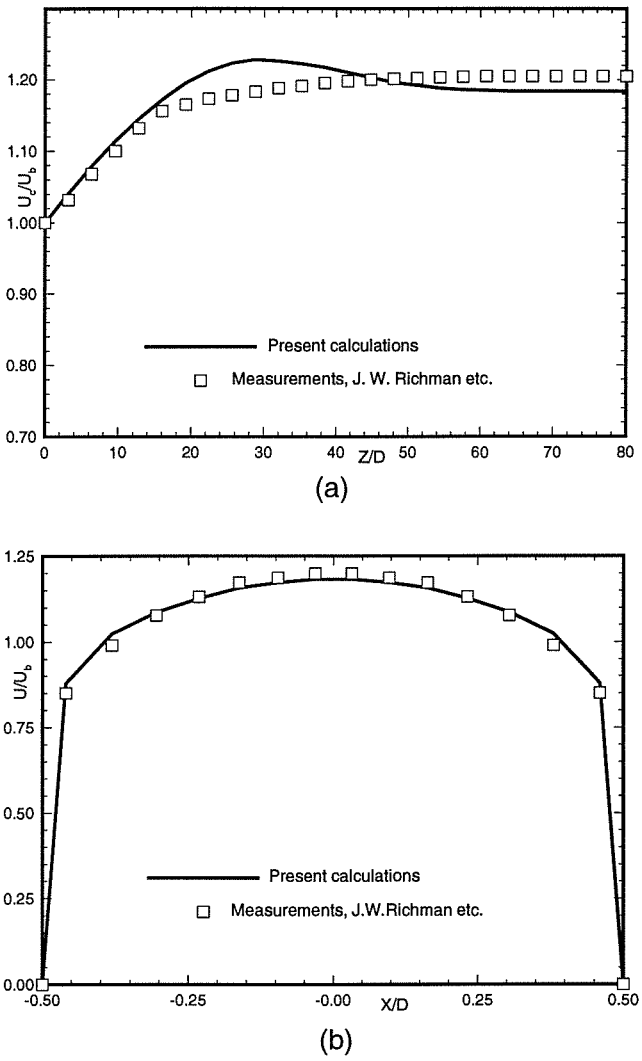

Figure 3.2: Predicted velocity profiles for $Re = 10^5$, (a) developing velocity profile at the pipe center, (b) developed velocity profile at cross-section where $z/D = 60$.

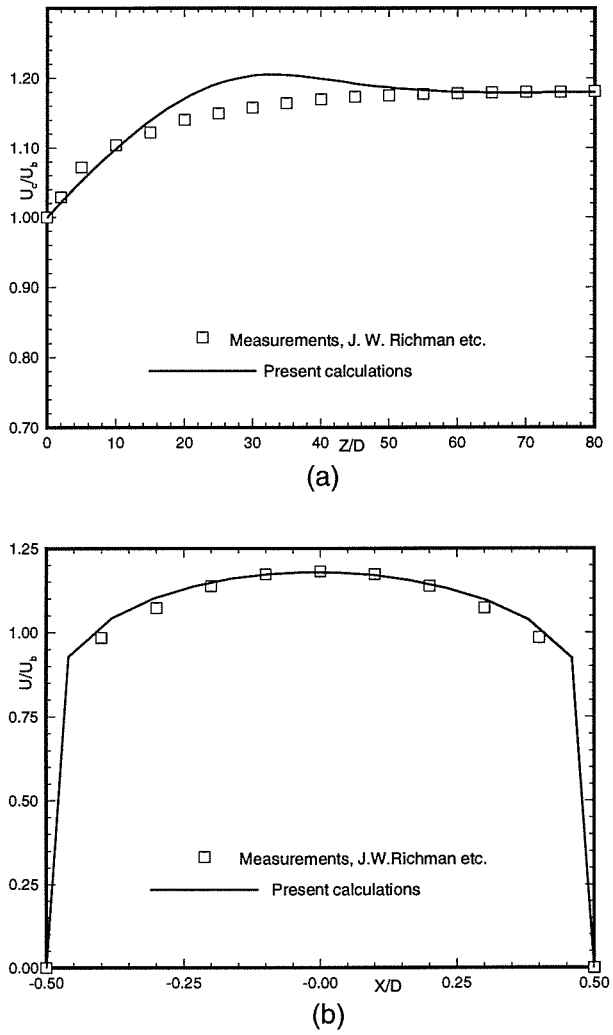

Figure 3.3: Predicted velocity profiles for $Re = 3.0 \times 10^5$, (a) developing velocity profile at the pipe center, (b) developed velocity profile at cross-section where $z/D = 60$.

was confirmed by other experimental results as discussed by Klein (1981). The flow becomes fully developed after 50 pipe diameters. The profile of the predicted fully-developed velocity is plotted in Figure 3.2b with the experimental data. This figure shows that the fully-developed velocity profile is in good agreement with the experimental results.

Similar computational results for the higher Reynolds number 3.0×10^5 together with the experimental measurements of Richman *et al* (1973) are plotted in Figure 3.3. The velocity peak occurs at about 32 pipe diameters; a little later than the case for the lower Reynolds number. Good agreement between the predicted and experimental results is also seen here at large z/d.

From both computations, accurate solutions were obtained for the flow in the well-developed region. This is because the $k - \epsilon$ model and the 'wall function' describe the well-developed turbulent pipe flows very well. Thus the method can be very accurate if a suitable turbulence model is used.

3.5.2 Developing Turbulent Flow in a 90^0 Curved Square Duct

As the second computation, turbulent flow in a 90^0-bend square duct was simulated. Experimental measurements were reported by Taylor *et al* (1982). The geometry which was used in the experiment was also used for the computation, which consisted of a 90^0 curved square duct with a mean bend radius of 92 mm with an upstream straight duct 0.3 m long and a downstream straight duct 0.2 m long. The dimensions of the cross-section are $40\times$ 40 mm^2. The Reynolds number defined by the bulk velocity, U_b, and the cross-sectional dimension is $40,000$.

Exploiting the advantage of flow symmetry, only half of the physical domain was computed to save computational time. The configuration and grid used for the computation are illustrated in Figure 3.4. A grid with $21 \times 11 \times 81$ nodes was generated, which had

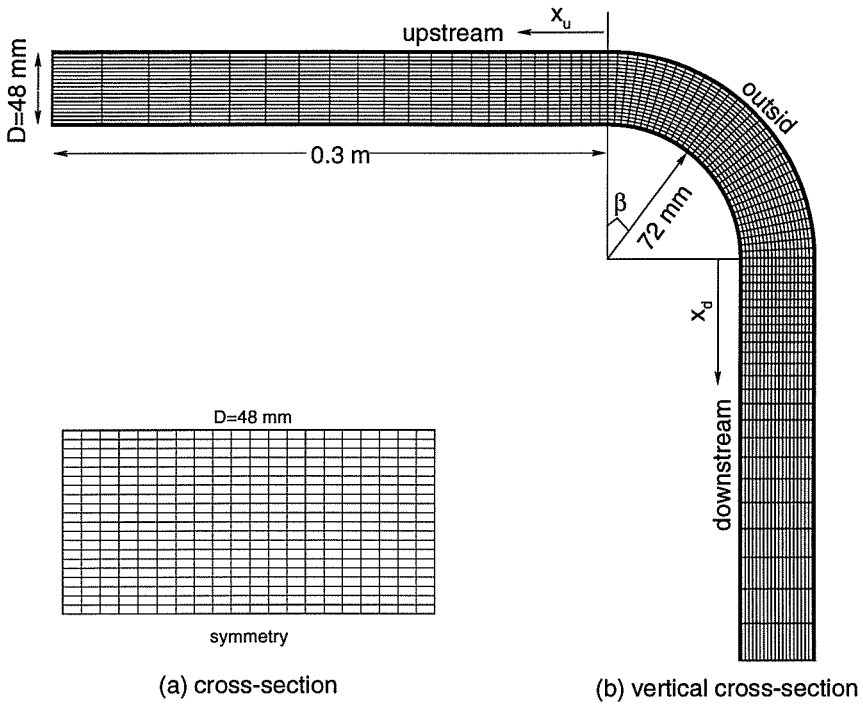

Figure 3.4: Illustration of computational grid for the curved duct.

a higher density of grid points in the curved section and less grid points near the outlet and inlet.

As in the last example, a uniform velocity and constant $k-\epsilon$ values were prescribed at the inlet. Computational results are plotted for different cross-sections. The flow field in the middle vertical plane is plotted in Figure 3.5. The maximum velocity occurs initially near the inner side wall in the upper portion of the curved section and gradually shifts to the outside wall due to the radial pressure difference arising from the bend curvature. As in the experimental results, strong secondary flows are found in the present computation which influence the whole flow pattern. The secondary current is developed gradually and becomes strongest at about $\beta = 60^0$, where the maximum secondary velocity reaches approximately thirty percent of the bulk velocity. The secondary flow fields at two different duct cross-sections together with the streamline velocity contours are plotted in Figure 3.6.

The predicted velocity profiles at four different locations are compared in Figure 3.7 with the measured results of Taylor *et al* (1982).

The results agree well at the upstream portion of the curved section and deteriorate with increasing downstream distance. However, the overall agreement is still reasonable. The discrepancy in the downstream section is believed to be due to limitations of the $k-\epsilon$ model for flows with streamline curvature and strong secondary flows. A higher-order scheme and denser grid spacing may also improve the computational results.

3.5.3 Developing Turbulent Flow in a 90^0 Curved Pipe

The last example simulates developing turbulent flow in a 90^0 curved pipe. Measurements of such a flow was reported by Enayet *et al* (1982). Laminar flow with the same geometry was studied in the last chapter where a good agreement between the experimental and computational results was obtained. In the present study, the turbulent flow is computed

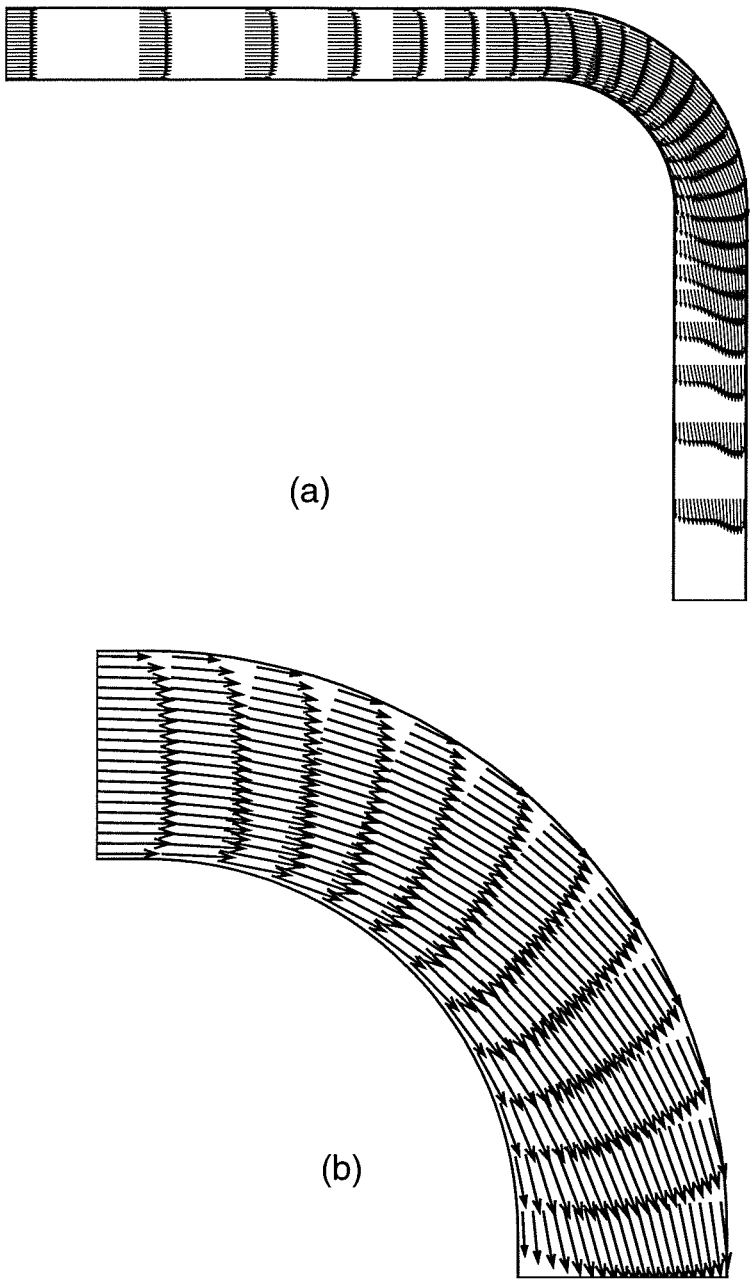

Figure 3.5: Flow field in the middle vertical plane, $Re = 40,000$, (a) flow field in the whole duct, (b) flow field in the curved section.

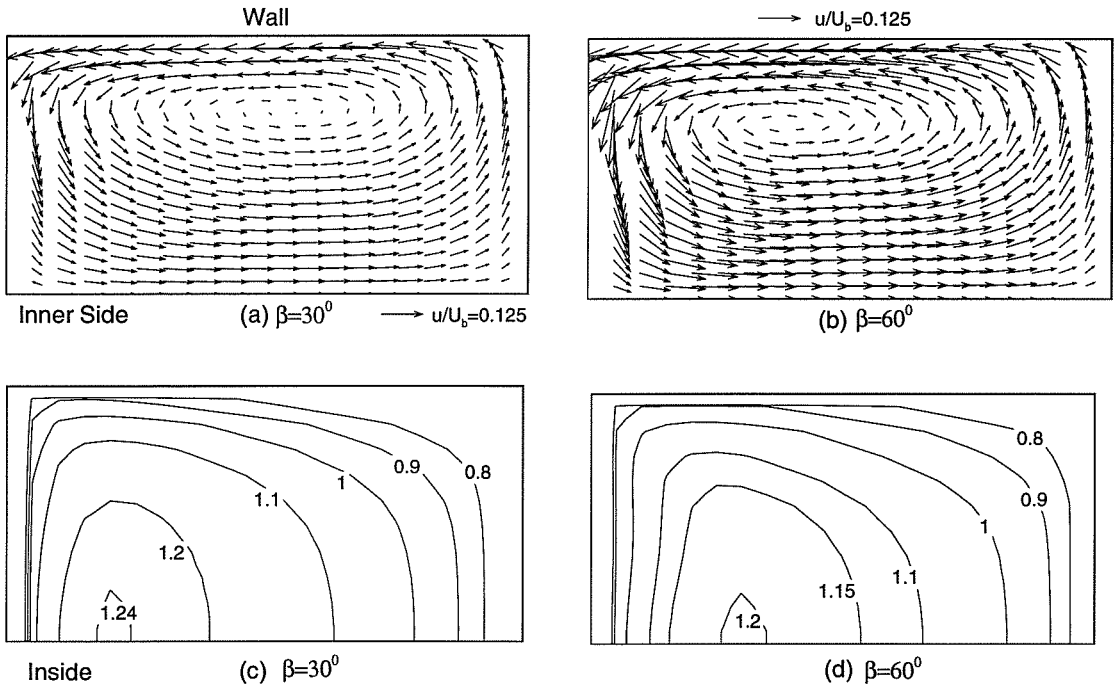

Figure 3.6: Secondary flow field and primary flow velocity contours of u/U_b at two different cross-sections.

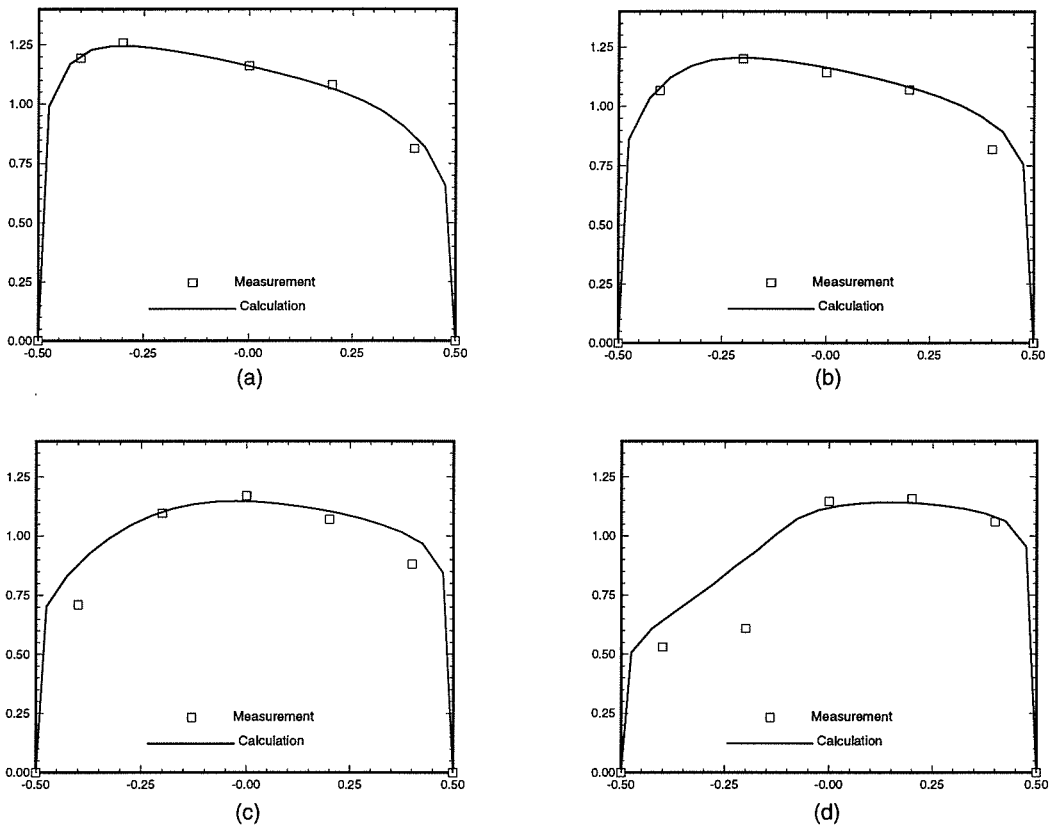

Figure 3.7: Comparisons of velocity profiles with the measurements at four different locations, (a) $\beta = 30^0$, (b) $\beta = 60^0$, (c) $\beta = 75^0$, (d) downstream $x_d = 0.25D$.

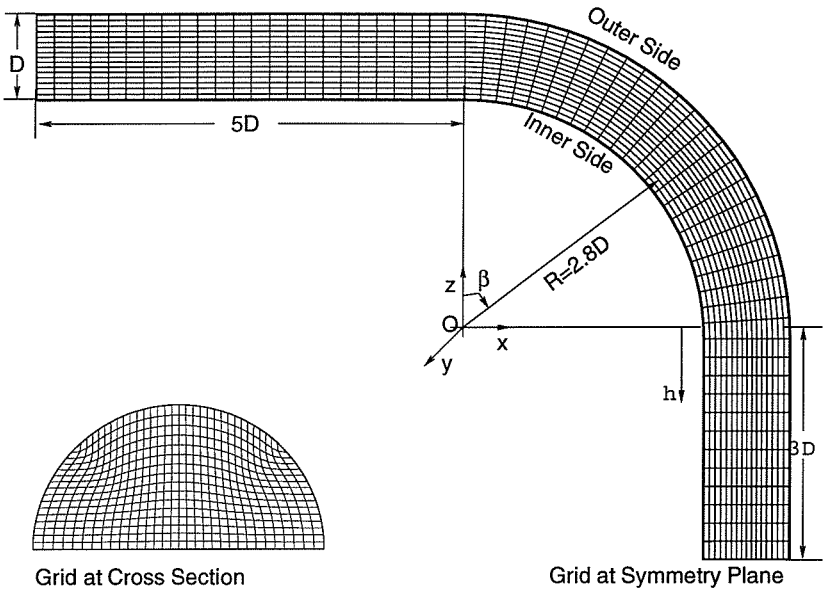

Figure 3.8: Illustration of computational grid for the curved pipe.

to test the present method. The geometry is similar to that in the previous example except that the cross-section is taken to be a circular disk. The diameter of the cross-section is 48 *mm* and the radius of the pipe curvature is equal to 2.8 times the pipe diameter. The bend was fitted with a straight pipe 240 *mm* long upstream and 480 *mm* long downstream.

The grid and geometry are illustrated in Figure 3.8. A non-uniform and non-orthogonal grid was used which was generated using an elliptical grid generation technique, to be described in chapter 6. As was the case for the previous example, only half of the flow domain was used for the computation.

The computation was carried out for a Reynolds number of 43,000; the same as that used in the experiments of Enayet *et al* (1992). The results are plotted in a similar manner as for the previous example. The velocity vectors at the middle vertical plane

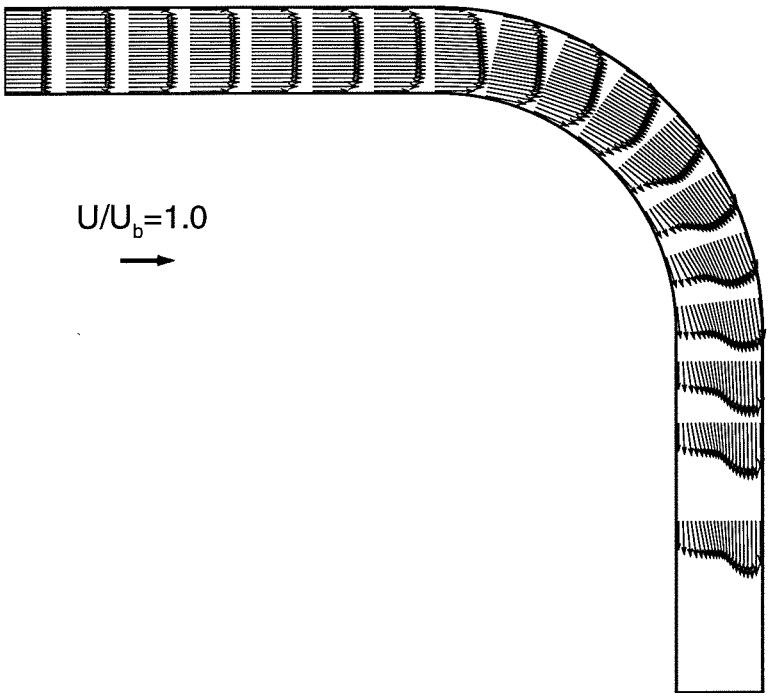

Figure 3.9: Flow field in the middle vertical plane, $Re = 40,000$.

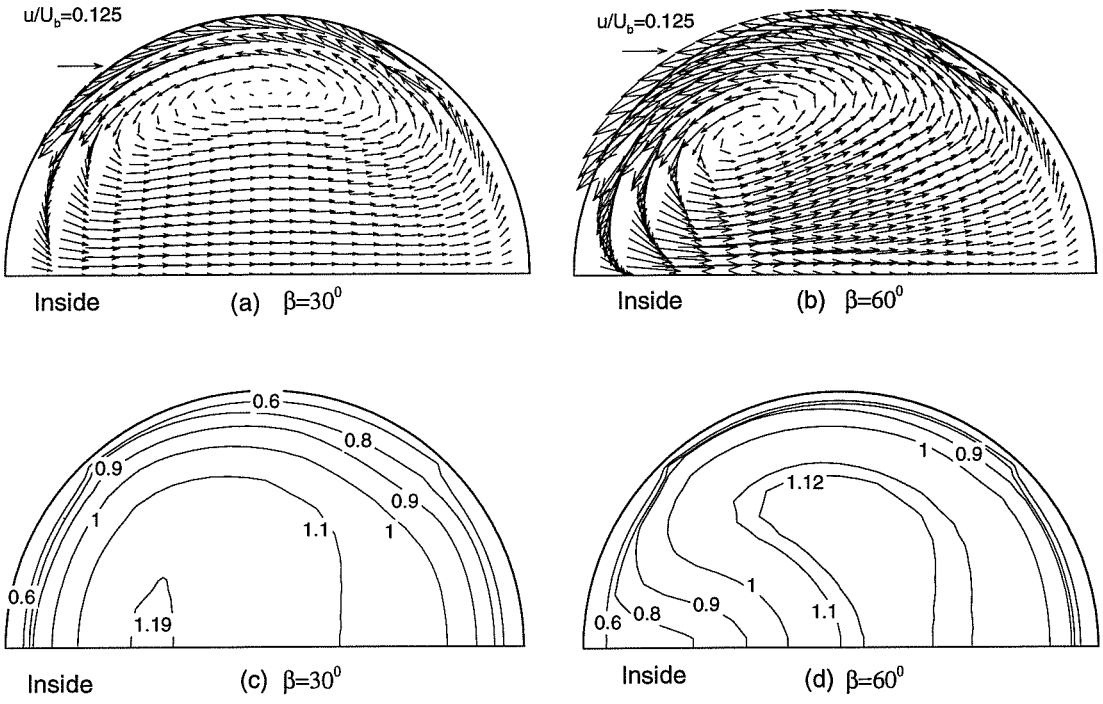

Figure 3.10: Secondary flow field and primary flow streamline velocity contours at two different cross-sections.

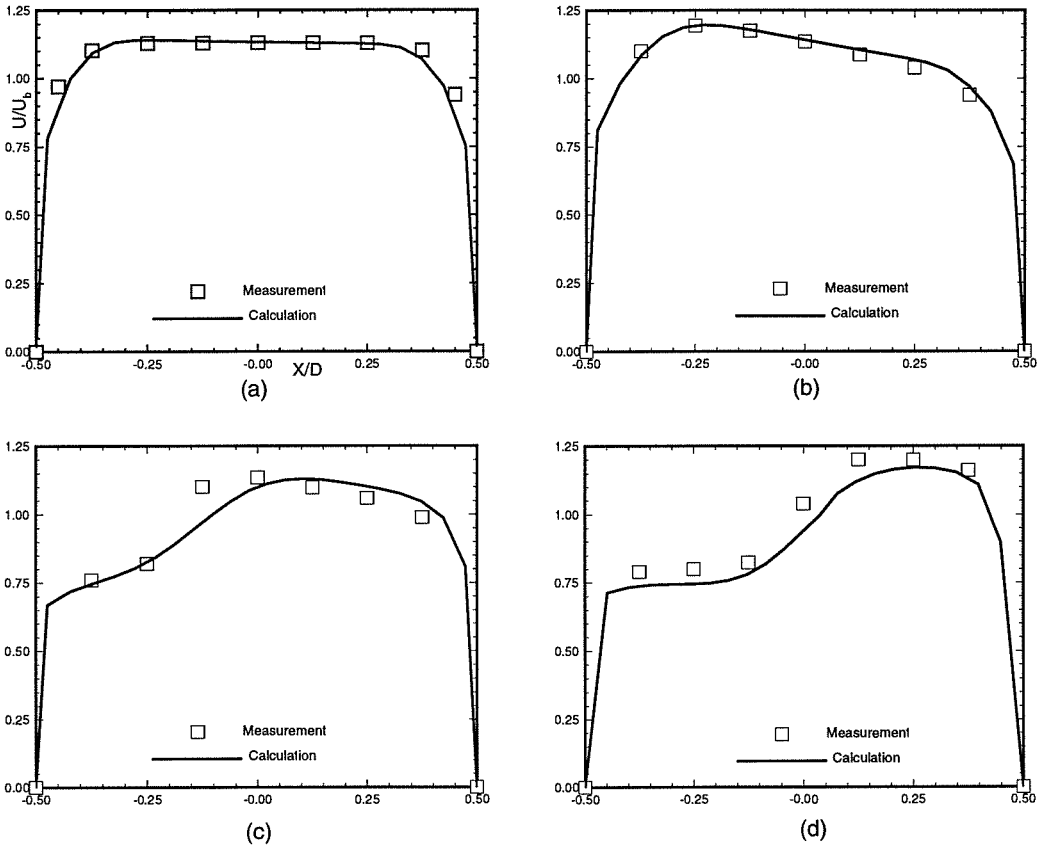

Figure 3.11: Comparisons of velocity profiles for different cross-sections, (a) upstream 0.5D, (b) $\beta = 30^0$, (c) $\beta = 60^0$, (d) $\beta = 75^0$.

are plotted in Figure 3.9. Unlike the pattern of laminar flow in the same geometry (see Figure 2.11), the velocity maximum in the curved, upstream section occurs near the inner side and not near the outside wall as observed in laminar flow. As in the previous example, the pressure-driven secondary flow extends over the entire flow region in the curved and downstream sections. The secondary flow fields together with the streamline velocity contours at two different locations are plotted in Figure 3.10 for reference. The predicted velocity at different locations with the measured results of Enayet (1982) *et al* are plotted in Figure 3.11. The overall agreement is good, even though there are again some discrepancies in the downstream portion of the curved section.

3.6 Closure

Numerical simulation of turbulent flows in complex geometries using general curvilinear coordinates is studied. The curvilinear coordinate-based method developed in the last chapter is generalized to turbulent flows using the $k-\epsilon$ two equation model with the 'wall function' treatment. The discretization of the $k-\epsilon$ equations and the treatment of the 'wall function' in curvilinear grids are presented. Methods enhancing the computational stability are introduced.

Several three-dimensional turbulent flows are computed using the developed solution method and detailed results are presented. Computational tests show that the method is efficient and reliable. Accurate solutions can be obtained for flows without curvature and recirculation. The computational results for the last two examples show certain discrepancies with the experimental results. This is attributed to the limitations of the $k-\epsilon$ turbulence model and the 'wall function' treatment for flows with strong secondary flows and streamline curvature. However, the overall agreement observed for both examples is still acceptable.

Chapter 4

Multigrid Acceleration

In this chapter, the development and performance of a multigrid procedure for the computation of three-dimensional laminar/turbulent flows using general curvilinear grids are presented. Brandt's (1981) Full Approximation Storage (FAS) in conjunction with the Full Multigrid cycling (FMG) is adopted. The numerical method for laminar/turbulent flows described in the last two chapters is used as the smoother. The present study shows that the discrete governing equations can become inconsistent between fine and coarse grids in curvilinear grids, as well as when the $k-\epsilon$ model with the 'wall function' is used for turbulent flows. Novel procedures are proposed to solve these problems. Several laminar/turbulent flows are calculated using the multigrid method to illustrate its performance.

4.1 Introduction

The convergence rate of traditional single-grid iterative solution methods seriously deteriorates as grids are refined. By using a Fourier analysis, Brandt (1980) showed that iterative solution methods are only efficient in smoothing those error components whose wavelengths are comparable to the mesh size. The multigrid method is an efficient iterative solution procedure which exhibits convergence rates insensitive to grid refinement. The method has been widely used in the field of computational fluid dynamics. More recently, a few studies have been reported on the use of multigrid acceleration in

curvilinear coordinates. Those works differ mostly in the choice of the basic numerical algorithms (coupled or decoupled), grid layout (staggered or collocated grids), and primary velocity unknowns on non-orthogonal grids (Cartesian, contravariant velocity components etc.). Rayner (1991) and Shyy *et al* (1993a,b) conducted multigrid calculations for two-dimensional flows in complex geometries based on decoupled sequential solution procedures. Those methods were based on the staggered grid system and the pressure-correction scheme. Rayner (1991) used the contravariant velocity components as the primary velocity unknowns, while Shyy *et al* (1993a) adopted a combination strategy in which both the Cartesian and contravariant velocity components were used to facilitate the formulation of the mass and momentum equations. In addition to the decoupled solution procedure, Joshi *et al* (1991) and Oosterlee *et al* (1992) reported multigrid algorithms for flows in two-dimensional complex geometries based on a coupled solution solver. These methods were based on the staggered grid system and the contravariant-type velocity unknowns. In contrast to the above two-dimensional works which are based on the staggered grid system, Smith *et al* (1993) presented a multigrid procedure for three-dimensional flows on non-orthogonal collocated grids. The algorithm used by the authors solves the momentum equations for the Cartesian velocities as the dependent variables and stores all variables at the control volume centers.

The implementation of multigrid algorithms for turbulent flows faces many additional difficulties, even in a Cartesian coordinate system. The $k-\epsilon$ two-equation model with the 'wall function' treatment is widely used for simulation of turbulent flows. The multigrid acceleration for turbulent flows using such a two-equation turbulence model has been investigated by a few authors. Based on experience with the difficulties of multigrid cycling associated with the $k-\epsilon$ equations and the 'wall function', Vanka (1987) developed a multigrid method which did not include the $k-\epsilon$ equations within the multigrid cycle, but instead solved them on the fine grid and transferred only the turbulent viscosity. This

method was also adopted by Yokota (1990) who reported good convergence rates. Even though this method can speed-up the convergence rate for turbulent flows, the multigrid efficiency may be limited by the slow convergence of the $k-\epsilon$ equations on very fine grids. A more effective solution procedure can be expected if the set of governing equations as a whole are solved together using the multigrid cycle.

Rubini *et al* (1992) presented a multigrid calculation for turbulent, variable density flow over a three-dimensional, backward-facing step using the $k-\epsilon$ model with 'wall function' treatment. The authors used a staggered grid system and the SIMPLE algorithm as the smoother. The convergence of the $k-\epsilon$ equations was accelerated together with the momentum and continuity equations using the multigrid strategy. The authors found that additional linearization of the source terms was necessary to prevent negative values of k and ϵ. In addition, the negative values of k and ϵ can arise due to negative correction. This problem was circumvented by simply not updating values in locations that would result in a negative value. Their results indicate a dramatic reduction in the computer time required to obtain a solution by a factor of over 25 for the finest grid considered. Shyy *et al* (1993b) presented a multigrid solution procedure for two-dimensional turbulent flows in conjunction with the $k-\epsilon$ two-equation turbulence model in general curvilinear grids. The method used a staggered grid arrangement and the Cartesian velocity components as the primary variables in the momentum equations. They reported some special treatments which facilitate the successful implementation of the multigrid algorithm. Particularly, they noted the inconsistency of the 'wall function' among different grid levels which may decrease the efficiency of the multigrid method.

In the present study, the development of a multigrid procedure for computation of three-dimensional laminar/turbulent flows in general curvilinear grids is presented. Problems associated with multigrid calculation in curvilinear grids are analyzed. Novel techniques are proposed to solve these problems.

It was reported by Shyy *et al* (1993b) that the geometrical complexities can impose an upper limit on the number of effective grid levels useful for the MG method. The present study shows that this limitation is caused by the inconsistency between the fine- and coarse-grid governing equations. When the coarse-grid defect governing equations are directly formulated using the coarse grid nodes only, they will degrade relative to the fine-grid equations with the degradation of the spatial resolution of the coarse grid. When the degraded coarse-grid equations are solved, the resulting correction will be inadequate to correct the fine-grid solutions thus reducing the efficiency of the multigrid algorithm. In the present study, this problem is analyzed and a new method is proposed to solve it.

As mentioned in the previous paragraph, when the multigrid acceleration is applied to all the governing equations, including the $k-\epsilon$ equations, it is found that the formulation of the coarse-grid defect equation using the 'wall function' may cause inconsistency between the governing equations of fine and coarse grids. This problem is discussed in this chapter and a novel practice is presented to allow the successful implementation of the multigrid method for turbulent flows using the $k-\epsilon$ equations. Several techniques which improve the multigrid solution efficiency for turbulent flows are introduced.

Consistent restriction of residuals and solutions between coarse and fine grids is important for the nonlinear multigrid algorithm. It is impossible to preserve both the mass and momentum conservations in the course of the transfer of solutions. However, mass conservation can be maintained between grids by deliberately choosing the interpolating formulation. In general, mass conservation between grids is usually not maintained in restriction for methods using curvilinear grids. In the present study, a mass-preserving restriction for the physical tangential velocity components is developed.

In order to validate the multigrid performance, computations were carried out for several laminar/turbulent flows with significant non-orthogonal and strongly curved grids.

Comparisons were made between the multigrid and single-grid methods through measurement of the number of finest grid iterations, the equivalent work units and CPU time. The computational results show that the developed multigrid method is very efficient.

4.2 The Multigrid Algorithm

The idea behind using multigrid methods is to remove the low-frequency errors by solving the governing equations on various auxiliary coarse grids. For a multigrid method, it is critical to use consistent governing equations between the coarse and fine grids. As discussed later, this criterion may be violated in certain curvilinear grids. Therefore, additional care should be taken in the development of a multigrid method for general curvilinear grids.

In the present study, the Full Approximation Storage (FAS) of Brandt (1981) in conjunction with the Full Multi-Grid (FMG) cycling is adopted. The solution procedure is initiated on the coarsest grid and subsequent initial solutions on finer grids are obtained by interpolating converged solutions. On each finer grid and on the finest grid, the FAS solution procedure is applied to obtain a converged solution.

To illustrate the problem to be solved, the FAS procedure is briefly described in this section using a two-grid (c, f) system. The discrete governing equations on a fine grid 'f' are written as an operator equation as follows:

$$L^f(\phi^f) = 0, \tag{4.1}$$

where L is the non-linear operator designating the discrete governing equations, ϕ is the solution vector which includes all dependent variables, and the superscript 'f' indicates that the solution and operator are associated with the fine grid. Suppose ψ^f is an approximate solution of equation (4.1). Unless ψ^f is the exact solution, there will be a

residual r^f such that

$$L^f(\psi^f) = r^f. \tag{4.2}$$

Let c^f be a correction of the approximation solution ψ^f so that $\psi^f + c^f$ satisfies equation (4.1), namely,

$$L^f(\psi^f + c^f) = 0. \tag{4.3}$$

Subtracting equation (4.3) from equation (4.2), the following equation is obtained:

$$L^f(\psi^f + c^f) - L^f(\psi^f) = -r^f. \tag{4.4}$$

If the correction c^f can be found from the above equation, it is obvious that the exact solution ϕ^f can be obtained. The idea of the multigrid method is to solve for the correction c^f, rather than solve for the solution on a coarse grid. In order to do so, the above equation is restricted to the coarse grid 'c',

$$L^c(I_f^c\psi^f + c^c) - L^c(I_f^c\psi^f) = -I_f^c r^f, \tag{4.5}$$

where the superscript 'c' indicates that the quantities are associated with the coarse grid, while I_f^c is the restriction operator. For linear problems, the approximation solution $I_f^c\psi^f$ cancels out in equation (4.5) and a correction equation can be obtained. For non-linear problems, the full approximation equation

$$L^c(\phi^c) = L^c(I_f^c\psi^f) - I_f^c r^f \tag{4.6}$$

is solved in the FAS procedure and the correction c^c is then obtained from the solution difference

$$c^c = \phi^c - I_f^c\psi^f.$$

The auxiliary equation (4.6) containing the restricted fine-grid residual (or defect) is frequently referred to as the defect equation in the literature. After the correction is

solved from equation (4.6), it is prolongated to the fine grid to adjust the old solution ψ^f, namely, $\phi^f = \psi^f + I_c^f(\phi^c - I_f^c\psi^f)$ where I_c^f is the prolongation operator. If the adjusted solution ϕ^f does not satisfy equation (4.1), the above procedure can be re-applied iteratively until the residue is below the required criterion. The above procedure forms a two-level FAS procedure.

4.3 Consistent Formulation of Governing Equations

From the above multigrid procedure, it can be seen that the coarse-grid defect equation (4.6) used for the calculation of the correction is formulated from the fine-grid equation (4.1). Therefore, the defect equation should be a faithful representation of the fine-grid equation on the coarse grid. Three components, the residual r^f, approximation solution ψ^f, and solution operator L^f have to be transferred to the coarse grid to formulate the defect equation. Attention has been given to the accurate restriction of the solution and the residual in the literature. However, the restriction of solution operators also requires careful consideration. The defect equation (4.6) is similar to the solution equation (4.1) except for additional source terms which include the residual restricted from the fine grid. Therefore, the coarse-grid operator L^c is usually formulated by discretizing the governing equation on the coarse grid in the same manner as that used for the solution operator L^f. Such a practice is usually adequate for computations in Cartesian coordinates. However, this may result in inconsistent operators between coarse and fine grids for computations using curvilinear grids.

Shyy *et al* (1993b) reported that geometrical complexities can impose an upper limit on the number of effective grid levels for the multigrid method. Their computations demonstrated that the efficiency of a multigrid method is not improved by using higher grid levels if the spatial resolution on coarser grids becomes too degraded. The present

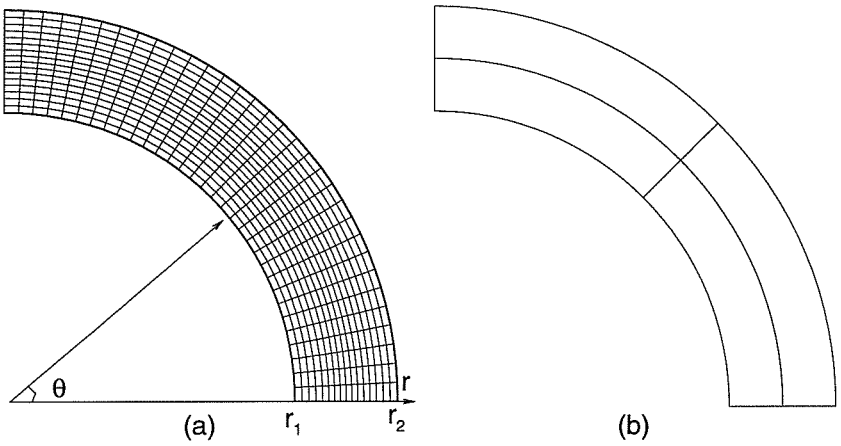

Figure 4.1: Grids in cylindrical coordinates: (a) 16×16 grid; (b) 2×2 grid.

study indicates that this limitation is caused by the direct formulation of the defect operator L^c on the coarse grid which is no longer consistent with the fine-grid operator L^f due to the degradation of the coarse grid. On a curvilinear grid, a coarse grid operator L^c depends on the grid geometric quantities associated with the coarse grid. In general, the analytic function of the coordinate transformation is unknown and the physical geometric quantities have to be calculated from the discrete grid nodes. Due to the deterioration of the spatial resolution on the coarse grids, however, the physical geometric quantities calculated from the coarse-grid nodes can become inconsistent with those on the fine grid in the sense that they are not defined by the same coordinate functions.

In order to illustrate this problem, a two-dimensional flow domain, namely, a curved channel is considered. For this particular problem, an analytic coordinate transformation exists:

$$x = rcos\theta, y = rsin\theta, \qquad r_1 \leq r \leq r_2, 0 \leq \theta \leq \frac{\pi}{2}, \tag{4.7}$$

which transforms the curved channel domain in the $x - y$ plane to a rectangular domain in the $r - \theta$ plane. The corresponding curvilinear grids are shown in Figures 4.1a & 4.1b for a fine grid 16×16 and a coarse grid 2×2, respectively. From Figure 4.1, it can be seen

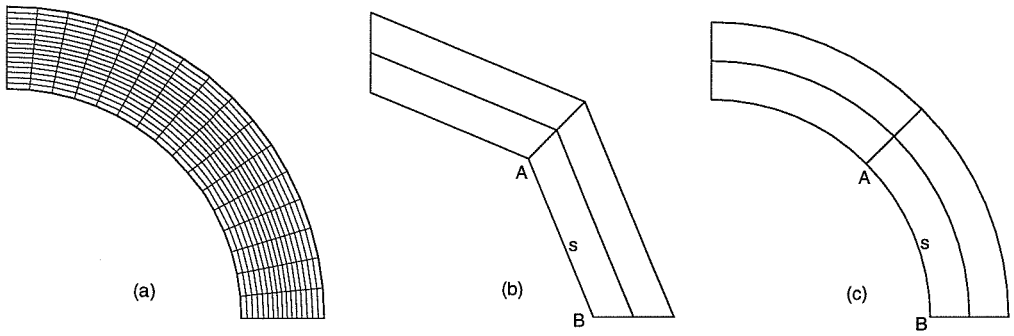

Figure 4.2: Curvilinear grids: (a) 16×16 grid; (b) coarsest 2×2 grid; (c) 2×2 grid restricted from the fine grid 16×16.

that even the 2×2 coarse grid still fully represents the curved channel geometry. This is because the grid lines on the coarse grid are defined by the analytical function (4.7), which is independent of the number of grid nodes. When the physical geometric quantities for fine and coarse grids are calculated from the coordinate transformation function (4.7), they are consistent since they are defined by the same coordinate system. In general, however, the physical geometric quantities have to be calculated from the discrete grid nodes When the physical geometric quantities are calculated from the discrete grid nodes using the linear profile assumption, such as that used in chapter 1, the grid lines used for the calculation are actually the line segments produced by connecting the neighbouring grid nodes, as shown in Figures 4.2(a) &4.2(b) for the 16×16 fine grid and 2×2 coarse grid, respectively. From these figures, it can be seen that the fine grid in Figure 4.2a is a reasonable approximation to the corresponding grid in the cylindrical coordinates shown in Figure 4.1a. Thus, the physical geometric quantities calculated from the fine-grid nodes are approximately equal to those defined by the coordinate transformation (4.7). However, the coarse grid in Figure 4.2b can no longer give a faithful representation of the corresponding grid in the cylindrical coordinate system shown in Figure 4.1b, and thus the physical geometric quantities calculated from the coarse-grid nodes are different

from those calculated from the coordinate transformation (4.7). Therefore, the physical geometric quantities on the coarse and fine grids are inconsistent since they are essentially equivalent to those defined by two different coordinate systems.

It should be noted that the grid lines on a coarse grid always coincide with those on the fine grid in an analytical coordinate system, such as the cylindrical coordinate system, no matter how coarse the grid is. Similarly, this argument should also be valid for general curvilinear grids, namely, the grid lines on a coarse grid used for the computation of the physical geometric quantities should coincide with those on the fine grid in order to be defined by the same coordinate transformation. For the curved channel geometry, the grid lines in a 2×2 grid defined by the equivalent transformation of the 16×16 fine grid in Figure 4.2a should be the restriction of the fine-grid lines as shown in Figure 4.2c, rather than those shown in Figure 4.2b. However, the physical geometric quantities such as the surface areas (lengthes in 2D) and cell volumes defined in Figure 4.2c can not be obtained from the coarse grid nodes alone. Calculation of the grid geometric quantities from the coarse-grid nodes is equivalent to using the low-resolution grid lines as shown in Figure 4.2b rather than the curvilinear grid lines as shown in Figure 4.2c. For instance, the length (area in 3D) of the cell-surface 's' calculated from the grid nodes in Figure 4.2b is equal to the distance of the two neighbouring nodes $\bar{AB}$ rather than the length of the corresponding curved segment $\hat{\text{AB}}$ in Figure 4.2c. Therefore, the physical geometric quantities calculated from the coarse-grid nodes only will be inconsistent with those on the fine grid. When the defect equations on the coarse grid are formulated using the coarse-grid geometric quantities, an inconsistency will arise between the governing equations on the fine and coarse grids. The correction calculated from the inconsistent coarse-grid defect equations will be inadequate to correct the fine-grid solution, thus, reducing the multigrid efficiency.

4.4 Consistent Calculation of the Grid Geometric Quantities

From the above discussion, we know that in order to obtain consistent governing equations between fine and coarse grids, consistent physical geometric quantities have to be used. In the present study, a calculation procedure is developed to achieve this goal. The idea of the method is to use the same grid resolution in the formulation of governing equations for all the grid levels. Instead of using the coarse-grid nodes to discretize the equations, the finest grid is used to formulate the discrete governing equations on all the grid levels. All the coarse-grid geometric quantities required to formulate the defect governing equations are calculated from the finest grid instead of from the coarse-grid nodes. First, the grid geometric quantities on the finest grid, namely, the surface areas, cell-volumes and surface normal vectors, are calculated using the method presented in chapter 2. The geometric quantities in the subsequent coarser grids are calculated in the following manner. The surface areas for a coarse grid are calculated by summing the areas of the relevant cell surfaces of the finest grid,

$$A_c^i = \sum_{finest} A_f^i, \tag{4.8}$$

and the unit surface normal vector $\mathbf{e}^i$ on a coarse grid is calculated as the area-mean of the relevant unit surface normal vectors $\mathbf{e}^i$ in the finest grid,

$$\mathbf{e}_c^i = \frac{1}{A_c^i} \sum_{finest} A_f^i \mathbf{e}_f^i, \tag{4.9}$$

where the subscripts 'c' and 'f' indicate that the quantities are associated with the coarse and the finest grid respectively, and the summation is over all the cell surfaces of the finest grid coincident with the coarse-grid surface. The cell volume of a coarse grid is calculated as the summation of the relevant cell volumes of the finest grid,

$$V_c = \sum_{finest} V_f, \tag{4.10}$$

where the summation is over all the finest-grid cells coincident with the coarse-grid cell. Such a calculation procedure of physical geometric quantities is equivalent to computations using the coordinate transformation designated by the finest grid through all the grid levels. For instance, the length of the cell-surface 's' for the 2×2 coarse grid shown in Figure 4.2 will be the length of the curved segment $\widehat{AB}$ in Figure 4.2c rather than the line segment $\bar{AB}$ in Figure 4.2b. With the present method, the coarse grids are actually not involved in the numerical formulations. Therefore, the deterioration of the geometric representation of coarse grids does not influence the multigrid efficiency.

4.5 Multigrid Acceleration of Turbulent Flows

The problem discussed in the last section occurs for both laminar and turbulent flows. In this section, the problems particularly associated with multigrid acceleration of turbulent flows are discussed. As mentioned earlier, the $k-\epsilon$ equations are solved together with the momentum and continuity equations in the present study using the multigrid procedure. The multigrid acceleration of the $k-\epsilon$ equations faces additional difficulties which have to be carefully addressed. First, the use of the 'wall function' to link the flow in the near wall region to the main flow region may not be suitable for all grid levels. The 'wall function' is valid only within a certain region near the wall. The calculation of k and ϵ near the wall relies on the assumption of the local equilibrium of production of and generation of turbulence energy. In a six-level multigrid solution procedure, for instance, the distance from the first scalar node to the wall on the coarsest grid will be $2^5 = 32$ times that of the finest grid. When the first scalar node of the finest grid is adequately located in the equilibrium region, the first grid node of the coarsest grid may be too far from the wall to obtain a reasonable solution profile. Secondly, the use of the 'wall function' in different grid levels will cause inconsistent governing equations

between grids as will be demonstrated later. Thirdly, the strong nonlinearity and the dominance of the source terms in the k and ϵ equations make these equations difficult to handle in the multigrid cycling. The $k-\epsilon$ equations have a strong nonlinear coupling through the production and dissipation in the source terms. These source terms are often dominant in the discrete governing equations. Furthermore, unlike the velocities and pressure, certain k and ϵ values updated by iteration or by adding corrections will cause complete failure of the algorithm. These problems are discussed in the following sections and novel treatments are introduced to solve them.

4.6 The Treatment of Wall Boundary on the Coarse Grids

In this section, the problem associated with the inconsistent governing equations caused by the use of the 'wall function' in different grid levels is considered. To illustrate this problem, a two-level multigrid procedure for a two-dimensional turbulent flow is considered. Figure 4.3 shows a two-level multi-grid near a wall boundary, where the big and small circles indicate the scalar nodes next to the wall on which the 'wall function' is applied. The $k-\epsilon$ equations are applied to the cells marked with an 'x'. If the 'wall function' is used to formulate the coarse-grid defect equation, the restricted residual $I_f^c r^f$ in equation (4.6) at the scalar node 'E', for instance, will be the average of the fine-grid residual R^f at the nodes 'A', 'B', 'C' and 'D'. However, the residuals at nodes 'A' and 'B' are calculated from the $k-\epsilon$ equations which are inconsistent with the operator L^c at node 'E' formulated by the 'wall function'.

In equation (4.6), the restricted residual I_f^c includes the residuals of the $k-\epsilon$ equations on the fine grid, but the solution operator L^c has to be formulated using the 'wall function'. Therefore, it is unlikely that a reasonable correction can be obtained by applying the 'wall function' in certain areas while the residuals in the fine grid are calculated from

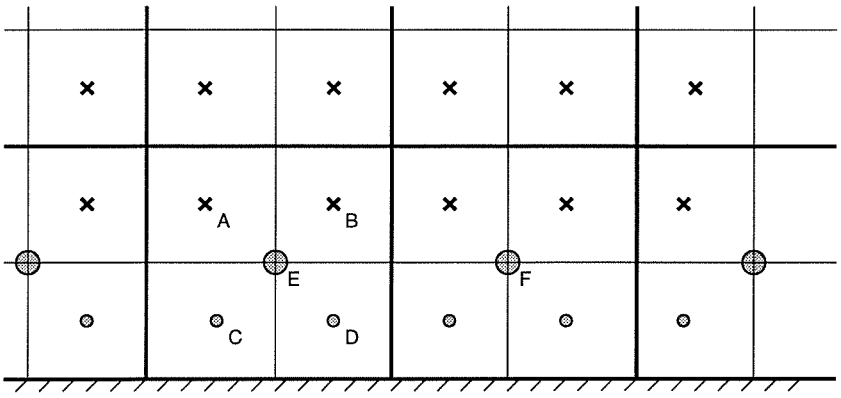

Figure 4.3: Illustration of using inconsistent defect equations at interior scalar nodes 'x'.

the $k-\epsilon$ equations there. This will cause an inconsistency of the governing equations between the fine and coarse grid levels.

The deterioration of speed-up from the single-grid to the multigrid procedure for turbulent flow calculations using the $k-\epsilon$ two-equation model with 'wall function' was also observed by Shyy *et al* (1993b). To solve this problem, they used a special grid coarsening system in which the first grid lines away from the wall are retained during the course of grid restriction. With this practice, the treatment of wall boundary for turbulent flows is adequately handled at the coarser grid levels, resulting in an increased speed-up from the single-grid to the multigrid procedure. However, this non-standard grid restriction will cause additional complexity in programming. Furthermore, the much-increased mesh expansion ratios among the adjacent cells on the coarser grids is not well handled.

In the present study, a new method is proposed to solve this problem. The idea is based on the argument that consistent governing equations through all grid levels have to be used. To avoid the inconsistency between the use of $k-\epsilon$ equations and the use of

the 'wall function' treatment on different grid levels, the 'wall function' is only applied on the finest grid. For the wall boundary on the coarser grids, a Dirichlet boundary condition for the $k-\epsilon$ equations is used in place of the 'wall function' treatment. Instead of solving k and ϵ from the balance between the local production and dissipation of k in the equilibrium region near the wall, the restricted k and ϵ from the finer grid are prescribed at the scalar nodes next to the wall as the boundary condition. A similar treatment to the above has been developed simultaneously and independently by Sun (1994).

For the momentum equations, the wall boundary on the coarse grids can be treated in the same way as for the $k-\epsilon$ equations, that is, the velocity components next to the wall can be simply prescribed using the restricted velocity components from the fine grid. Such a treatment is observed to provide better performance than the standard treatment. However, this treatment will result in no correction for the tangential velocity components on the cells next to the wall which limits the potential speed-up of the convergence rate of the multigrid method. In the present study, an alternative approach is used which solves the momentum equations in all the velocity positions without using the prescribed values from the fine grid. The tangential velocity components next to the wall are strongly dependent on the wall shear stress. If the wall shear stress can be determined, a governing equation for the tangential velocity components next to the wall can be formulated. In the 'wall function' treatment, the wall shear stress τ_w is calculated from the friction velocity u_τ which is defined by the 'wall function'. Since the 'wall function' will not be used on the coarser grids in the proposed method, the wall shear stress can only be obtained from the finest grid. Fortunately, it is possible to calculate the wall shear stress on a coarse grid from that on the finest grid. It is known that the wall shear stress is independent of the grid level. Therefore, the correct wall shear stress on the coarser grid can be calculated from values on the finest grid. In the present study, the wall shear

stress on a coarser grid is taken as the area-weighted average of the corresponding wall shear stresses on the finest grid,

$$\tau_w^c = \frac{1}{A_c} \sum_{finest} A_f \tau_w^f \tag{4.11}$$

where the scripts 'c' and 'f' indicate the association between the coarse and finest grids respectively. The summation is over all the finest-grid wall surfaces coincident with the coarse-grid surface. After the wall shear stress is obtained, the momentum equations for the tangential velocity components next to the wall can be formulated by considering the wall shear stress as an auxiliary force in the momentum equation. The components resulting from the vector expansion of the wall shear stress τ_w^c in the unit tangential vector basis $\{\mathbf{e}_i\}$ are used to modify the source terms in the discrete momentum equations (2.34-2.36). With such a treatment, the governing equations on the coarse grids can be solved to provide accurate correction.

4.7 The Treatment of $k - \epsilon$ Quantities and Source Terms

In this section, several special treatments facilitating the multigrid calculation of turbulent flows are introduced. In chapter 3, a treatment of the source terms of the $k - \epsilon$ equations was presented for calculation in general curvilinear grids. A key feature of the treatment is to assure positive source terms in both the discrete k and ϵ equations. This practice is useful in avoiding negative values for turbulence energy and dissipation since it is known that any negative k and ϵ acquired during the iteration will lead to complete failure of the solution method.

In the coarse-grid defect equation, the source terms contain additional quantities from the restricted residuals and solutions which may become negative. Thus the values of k and ϵ possibly acquire an erroneous negative value. A similar treatment to that reported by Rubini *et al* (1992) is used to solve this problem. However, different from

their practice in which only the resultant error of the restricted solution was considered, the source term as a whole is considered. Suppose the coarse-grid defect equations on a coarser grid H for k and ϵ can be written as

$$a_p^c \phi_p^c = \sum_{nb} a_{nb}^c \phi_{nb}^c + S^c, \tag{4.12}$$

where S^c is the source term containing quantities from various sources, such as grid non-orthogonality and restricted residual. If the source term S^c becomes negative, the following discrete equation is used for the iterative solution solver:

$$(a_p^c - \frac{S^c}{\phi_p^c})\phi_p^c = \sum_{nb} a_{nb}^c \phi_{nb}^c. \tag{4.13}$$

This practice will eliminate any negative source term in the $k - \epsilon$ equations.

In a multigrid method, the values of k and ϵ may also become negative after adding a correction. Rubini *et al* (1992) and Shyy *et al* (1993b) adopted the same strategy by which they simply do not update any locations that would result in a negative value. In the present study, this strategy is modified by including an additional limitation. Since both k and ϵ are involved in the denominators of some terms in the governing equations, they should not be too small. For instance, if k becomes zero after adding the correction, the source term in ϵ equation (4.12) will become infinite. This will also lead to the failure of the multigrid algorithm. Therefore, the correction scheme is modified for k and ϵ in the present study. Let

$$q = \psi^f + I_c^f(\phi^c - I_f^c \psi^f),$$

then for k and ϵ

$$\phi^f = \begin{cases} q & \text{if } q > 10^{-4}\psi^f \\ \psi^f & \text{otherwise} \end{cases}$$

where ψ^f is the old approximation solution for k and ϵ, $\phi^c - I_f^c \psi^f$ is the correction on the coarse grid. The present practice implies that if the k and ϵ at some locations become

unreasonably small by adding a correction, then the values at these locations will not be updated.

Due to the strong nonlinearity and the dominance of the source terms in both the k and ϵ equations, under-relaxation of both the solution and source terms is adopted.

4.8 Restriction and Prolongation

In addition to the consistent formulation of governing equations on different grid levels, the consistent restriction of solutions and residuals is also important for the multigrid algorithm. In a nonlinear problem, the discrete solution operator L^c on a coarse grid depends on the restricted velocity unknowns from the corresponding fine grid. Therefore, the restricted solution should be a good representation of the fine-grid solution. It is impossible to conserve both mass and momentum in the course of the restriction of solutions. However, mass conservation can be maintained between grids by deliberately choosing the interpolating procedure. The mass preserving restriction has been emphasized and adopted in several previous works, such as Sivaloganathan (1988).

Mass conservation requires the mass flux through a control-volume surface on a coarse grid to be equal to the total mass flux through the relevant surfaces of the fine grid, namely,

$$Flux = (\varrho \mathbf{u} \cdot \mathbf{e}^i A^i)_c = \sum_{fine} (\varrho \mathbf{u} \cdot \mathbf{e}^i A^i)_f, \tag{4.14}$$

where the subscripts 'c' and 'f' indicate the association with the coarse and fine grids, respectively, and the summation is over all the control-volume surfaces on the fine grid coincident with the coarse-grid surface. Using equation (2.25) for the physical tangential velocity components, equation (4.14) can be rewritten as

$$(\varrho U^{\xi_i} A^i \mathbf{e}^i \cdot \mathbf{e}_i)_c = \sum_{fine} (\varrho U^{\xi_i} A^i \mathbf{e}^i \cdot \mathbf{e}_i)_f. \tag{4.15}$$

From the above equation, the physical tangential velocity component $U_c^{\xi_i}$ on the coarse grid can be defined as follows:

$$U_c^{\xi_i} = \frac{1}{(\varrho \mathbf{e}^i \cdot \mathbf{e}_i A^i)_c} \sum_{fine} (\varrho U^{\xi_i} A^i \mathbf{e}^i \cdot \mathbf{e}_i) \tag{4.16}$$

For the cell-centered variables and residuals the coarse-grid values are taken as the volume-weighted mean of the eight corresponding fine-grid values. Taking the pressure p as an example, we have:

$$p_c = \frac{1}{V_c} \sum_{fine} (V.p)_f, \tag{4.17}$$

where the summation is over all the fine-grid control cells enclosed in the corresponding control cell on the coarse grid. In order to preserve the high-frequency contents of the fine-grid residuals, an interpolation of higher order than that used for velocities is employed. The residuals of momentum equations on a coarse grid are taken as the area-weighted mean of the relevant twelve values on the fine grid. Taking the residual of the $U^{\xi_3}-$ momentum equation as an example, the following formula is taken with reference to Figure 4.4:

$$\begin{aligned} r^c|_{i^c,j^c,k^c} &= \frac{1}{\sum A^f}[A^f r^f|_{i^f,j^f,k^f-1} + A^f r^f|_{i^f-1,j^f,k^f-1} + A^f r^f|_{i^f,j^f-1,k^f-1} \\ &+ A^f r^f|_{i^f-1,j^f-1,k^f-1} + 2A^f r^f|_{i^f,j^f,k^f} + 2A^f r^f|_{i^f,j^f-1,k^f} + 2A^f r^f|_{i^f-1,j^f,k^f} \\ &+ 2A^f r^f|_{i^f-1,j^f-1,k^f} + A^f r^f|_{i^f,j^f-1,k^f+1} + A^f r^f|_{i^f-1,j^f-1,k^f+1} \\ &+ A^f r^f|_{i^f,j^f,k^f+1} + A^f r^f|_{i^f-1,j^f,k^f+1}, \end{aligned} \tag{4.18}$$

where the summation is over all the coefficients of the residuals. A bilinear interpolation is used as the prolongation operator. In all formulations for restriction and prolongation, the area-weighted average procedure is used for all quantities defined on the control-volume surface, such as velocity unknowns, surface normal vectors, and momentum residuals. The volume-weighted average procedure is adopted for all quantities defined at the control-volume center, such as pressure and mass residual.

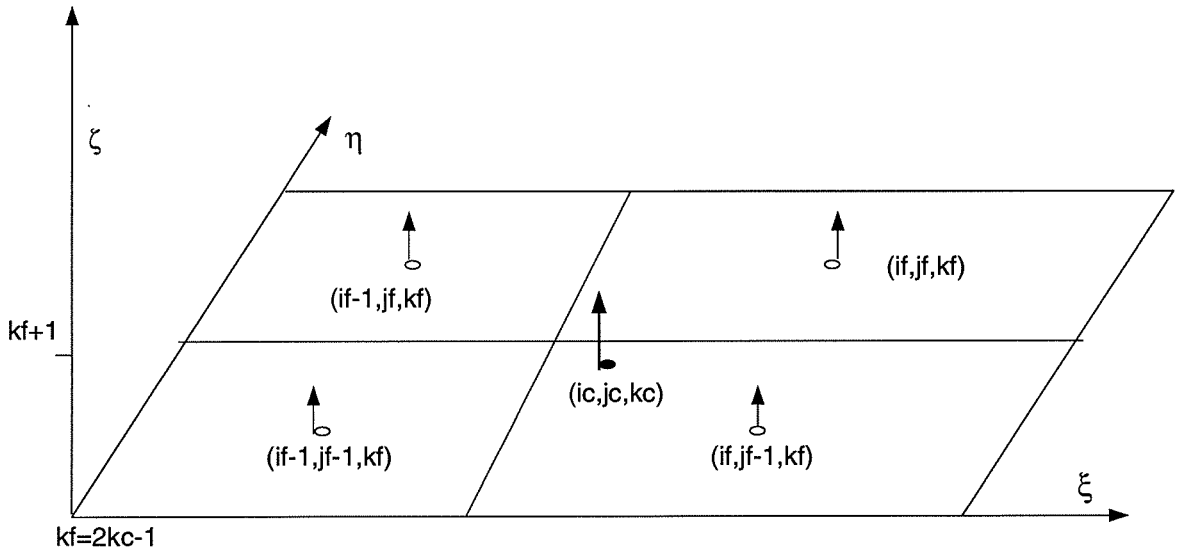

Figure 4.4: Illustration of locations of $U^{\xi_3}-$ residuals on fine and coarse grids for restriction.

4.9 Test Calculations

The developed multigrid algorithm in general curvilinear grids has been implemented in the CMGFD code. As shown in chapters 2 and 3, the code has been validated through a number of model problems for both laminar and turbulent flows and has been proven to be a reliable solver. Since the objective of this chapter is to develop an efficient multigrid algorithm, comparisons with experimental or other computational results are not made. Instead, the detailed convergence history for various grid sizes and multigrid levels is presented. In order to provide a uniform measure of computational effort, a work unit, WU, is defined as being equivalent to the computational effort for one traditional single-grid iteration on the finest grid. For instance, for three-dimensional problems, eight iterations on the next coarse grid are equivalent to one iteration on a fine grid. The convergence criterion is based on the absolute maximum residual of the four equations, i.e.,

$$R = max_{i,j,k}(R^u, R^v, R^w, R^p, R^k, R^\epsilon)$$

where the maximum is taken over all computational cells. A solution is considered to be converged in the present study when the maximum residual is below 10^{-6}. The reason for using such a tight convergence criterion is that certain calculations require lower residuals than others in order to show some flow phenomena. For instance, the third vortices in the skewed cavity contain very small velocities which can not be seen without obtaining lower order residuals (see Demirdzic *et al* (1992)).

The computations are started at the coarsest grid and the converged solution is prolongated to the next finer grid where the FAS procedure is applied. This procedure is continued until the finest grid is reached. A fixed V-cycle FAS procedure with two pre-smooth and two post-smooth iterations is used through all the computations. On the coarsest correction level, ten iterations are applied to solve the defect equation.

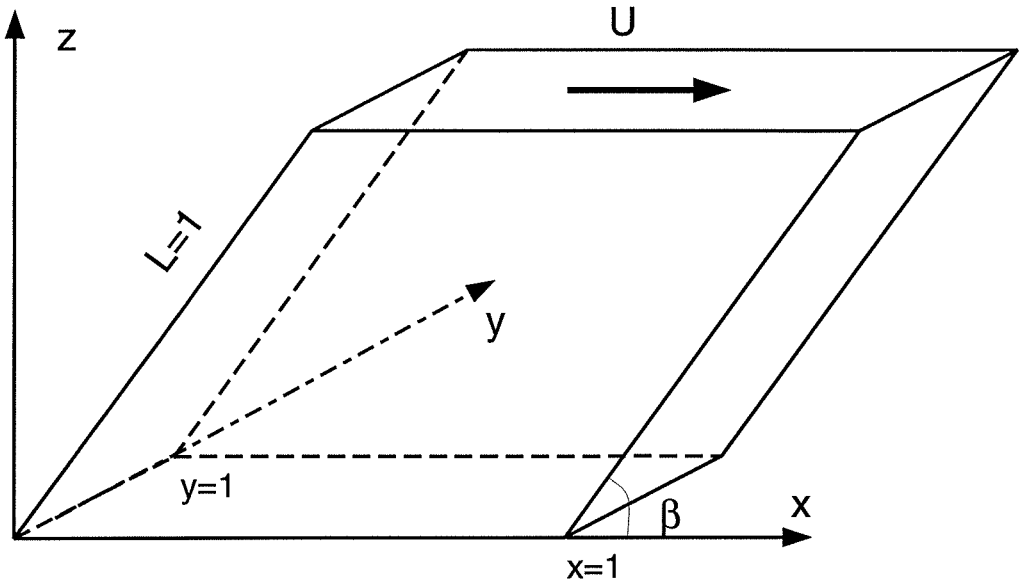

Figure 4.5: Physical geometry of the skewed cube.

4.9.1 Laminar Flow in a Skewed Cubic Cavity

In the first application, the multigrid algorithm is applied to the laminar flow in a skewed cubic cavity. This example tests the performance of the developed multigrid method for non-orthogonal grids. Flow in a cavity (or cube in 3D) has been considered as a standard test case for numerical algorithms due to the characteristic nonlinear elliptic nature it shares with many flows of practical interest. In the present study, the laminar flow in a skewed cubic cavity is considered. The counterpart of this problem in two dimensions, i.e., the flow in a skewed driven cavity, was proposed recently as a bench-mark problem by Demirdzic (1992) for testing numerical methods on non-orthogonal grids and has been studied by a number of authors. Figure 4.5 shows the problem currently being considered. The side wall is inclined at an angle $\beta = 45^0$ with the top wall moving at a velocity U.

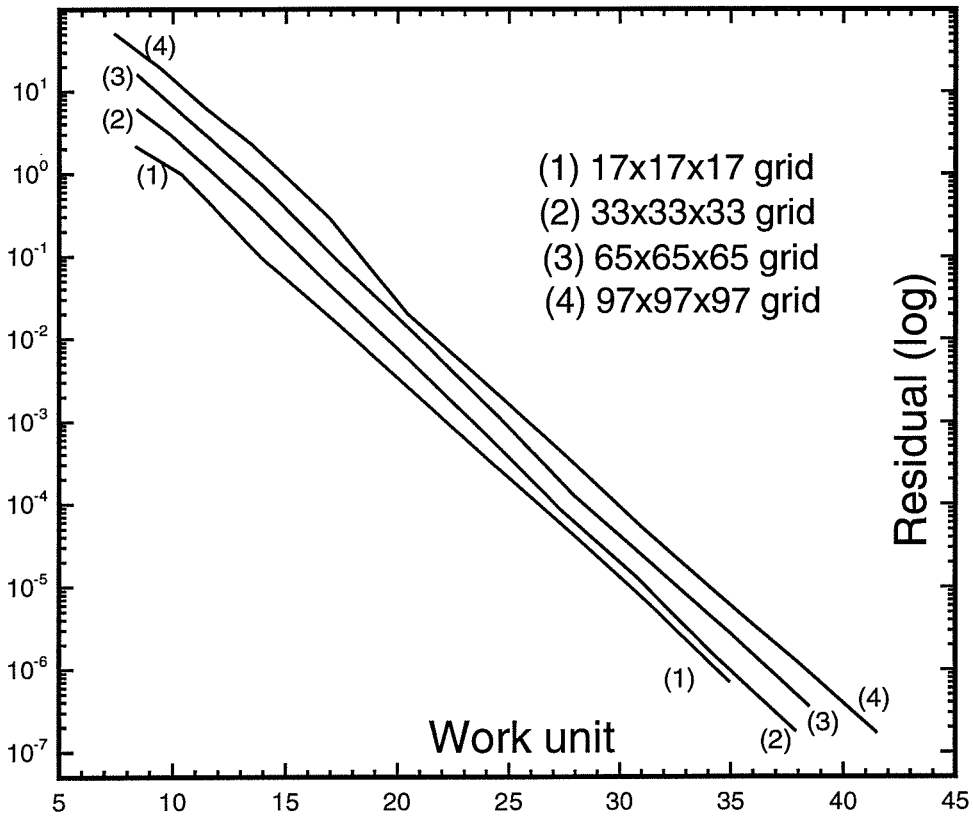

Figure 4.6: Convergence history of MG calculations using various grids.

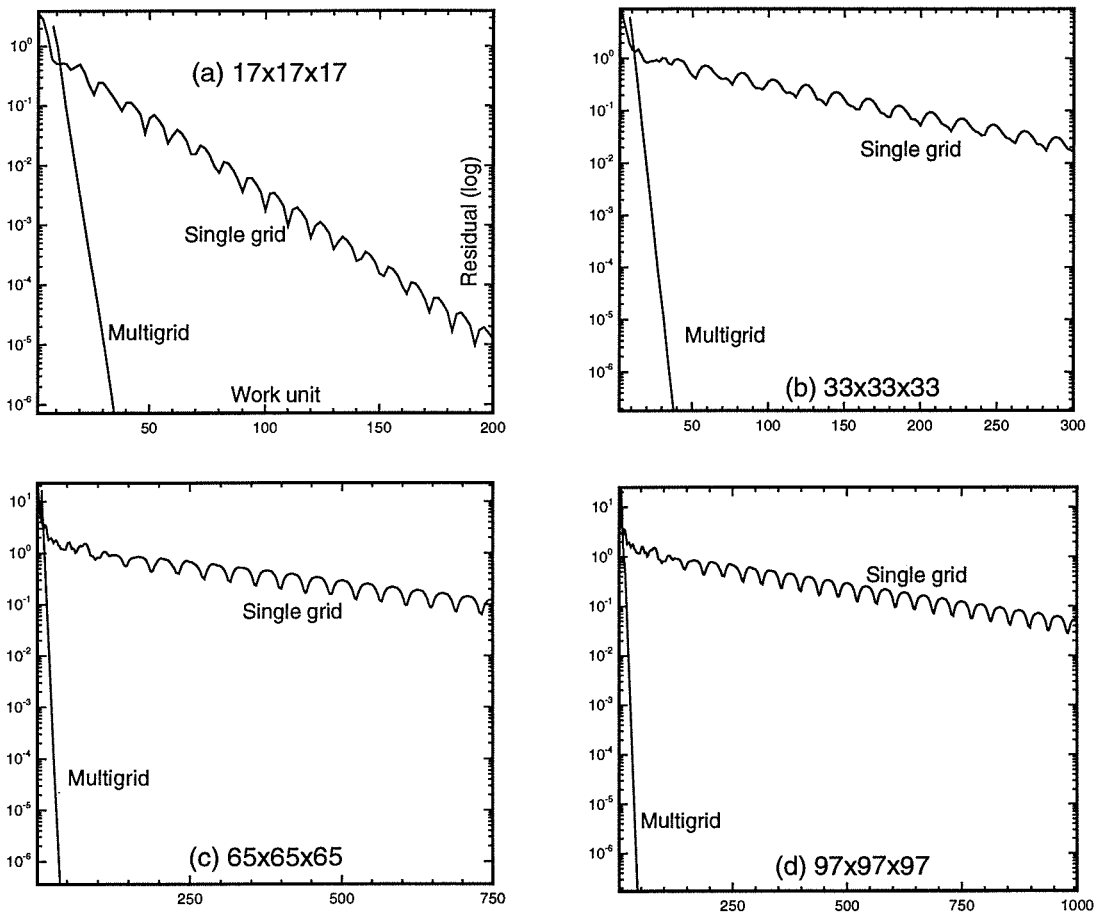

Figure 4.7: Comparison of convergence behaviors of multigrid and single grid calculations.

All the walls are squares of dimension $D \times D$ where $D = 1$ in Figure 4.5. The Reynolds number, defined as $\rho UD/\mu$, was taken to be 100 in the present study.

Multigrid calculations have been carried out for grids consisting of (1) $16 \times 16 \times 16$, (2) $32 \times 32 \times 32$, (3) $64 \times 64 \times 64$, and (4) $96 \times 96 \times 96$ cells using 4-level, 5-level, 6-level and 6-level grids respectively. In these multigrid calculations the coarsest grid consisted of only $2 \times 2 \times 2$ cells for the first three grids and $3 \times 3 \times 3$ cells for the last grid on which the converged solution can be obtained with just a few iterations. All calculations were initiated from zero velocity and pressure fields. The convergence histories of the multigrid calculations for the four different grids are plotted in Figure 4.6 with reference to the work unit (WU). It can be seen that, unlike the performance of single-grid methods, the rate of convergence is independent of the grids and convergence is achieved for all the grids in 40 work units. Vanka (1986b,1991) reported multigrid calculations for laminar flows in a cubic cavity. His calculations showed that the residuals decreased by about three orders of magnitude in roughly twenty finest-grid iterations. Considering the difference between the work unit and fine grid iteration, and different efforts used on one single-grid iteration, the performance of the present multigrid calculation using a non-orthogonal grid is found to be comparable with the results reported by Vanka (1991) for flow in a cube where the Cartesian coordinates were used.

In order to compare the performance of the multigrid and single-grid methods, computations with the traditional single-grid were also made. The convergence behavior of the multigrid and the traditional single-grid methods are plotted in Figures 4.7 (a), (b), (c) and (d) with reference to the equivalent work units for grids of $17 \times 17 \times 17$, $33 \times 33 \times 33$, $65 \times 65 \times 65$ and $97 \times 97 \times 97$, respectively. The convergence characteristics of these computations are also summarized in Table 4.1. CPU times for an IRIX(SGI) machine are reported. Single-grid computations for the fine grids $65 \times 65 \times 65$ and $97 \times 97 \times 97$ were terminated after the first 2000 and 1000 iterations respectively as

Finest Grid	MG Levels	MG CPU seconds	SG CPU seconds	CPU time speed-up	Finest grid iterations	MG WU	SG WU
$17 \times 17 \times 17$	4	47.2	213.9	4.5	18	34.91	242
$33 \times 33 \times 33$	5	390.9	7356.2	18.8	20	35.71	1092
$65 \times 65 \times 65$	6	4843.5	$3.8 \times 10^{5*}$	78*	21	37.46	4300*
$97 \times 97 \times 97$	6	16495.2	$1.6 \times 10^{6*}$	100*	24	41.46	6000*

Table 4.1: Convergence characteristics of all calculations (the superscript $*$ indicates that the values are estimated).

too much computational time is required to bring the residuals below 10^{-6} as specified for the multigrid method. However the number of iterations required for convergence can be estimated from the logarithmic variation after a certain number of iterations as observed in Figure 4.7. The estimated speed-up ratio was 78 and 100 for the fine grids $65 \times 65 \times 65$ and $97 \times 97 \times 97$ respectively. From these results, it can be seen that the multigrid algorithm produced a substantial improvement in the convergence rate. The speed-up ratio was increased when the grid was refined. It should be noted that it is impractical to seek a converged solution using the traditional single-grid method in the IRIX (SGI) machine for the fine grid (4) which would require months to finish. However, a solution can be obtained overnight on the same computer when the multigrid method is used. This result clearly shows the necessity of using multigrid techniques for flow calculations when very fine grids are required.

4.9.2 Laminar Flow in a Duct with Obstruction

The previous example requires the use of significantly non-orthogonal grids. However, the problem does not involve other difficulties associated with grid curvature and the degradation of spatial resolution on coarse grids as the grid is uniform and the grid geometric quantities are constant everywhere. In the following example, the laminar flow

in a duct with an arc on the bottom wall is computed for which a curved grid is required. The geometry shown in Figure 4.8 consists of a duct with a square cross-section modified by a circular arc on the bottom wall near the inlet. The counterpart of this problem in two dimensions, i.e., flow in a planar channel with a circular arc on the bottom wall, was recently computed by Shyy *et al* (1993b) using a multigrid algorithm. Here, the maximum height of the arc is taken as 30% of the inlet dimension which is higher than that used by Shyy *et al* (1993b). Multigrid calculations were carried out for a fine grid consisting of $129 \times 33 \times 33$ nodes using different grid levels in order to observe the efficiency of the multigrid method using higher grid levels. Figure 4.8 shows a coarser grid $65 \times 17 \times 17$ used in the present study where a non-uniform grid was adopted downstream. Figure 4.9 shows the coarsest grid, $9 \times 3 \times 3$, used in the present study, where Figure 4.9a was obtained by the restriction of grid lines on the finest grid and Figure 4.9b was obtained directly from the coarsest grid nodes. It can be seen that the grid shown in Figure 4.9b is not a faithful representation of the flow domain. When the defect equation (4.6) is directly formulated on the coarsest grid, it is equivalent to using the grid geometric quantities in Figure 4.9b. Therefore, the formulated governing equation is inconsistent with that on the fine grid due to the degradation of the geometric representation of the coarse grid. However, the physical geometric quantities calculated from the finest grid by the proposed method are equivalent to those in the curvilinear grid shown in Figure 4.9a. Thus, the representation of the flow domain does not deteriorate on the coarse grids.

Figure 4.10 shows the convergence histories of multigrid (MG) calculations using different grid levels for the grid of $129 \times 33 \times 33$. From this plot, it can be seen that the computational efficiency is improved when a higher grid level is used even though the coarsest grid was seriously degraded. Therefore, the effective grid levels useful for the MG method are not limited by the geometric complexity, unlike the results reported by

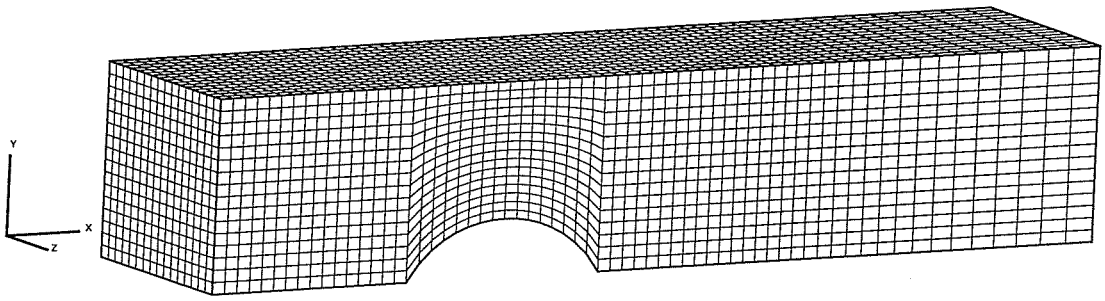

Figure 4.8: Illustration of the grid $65 \times 17 \times 17$ used in the study.

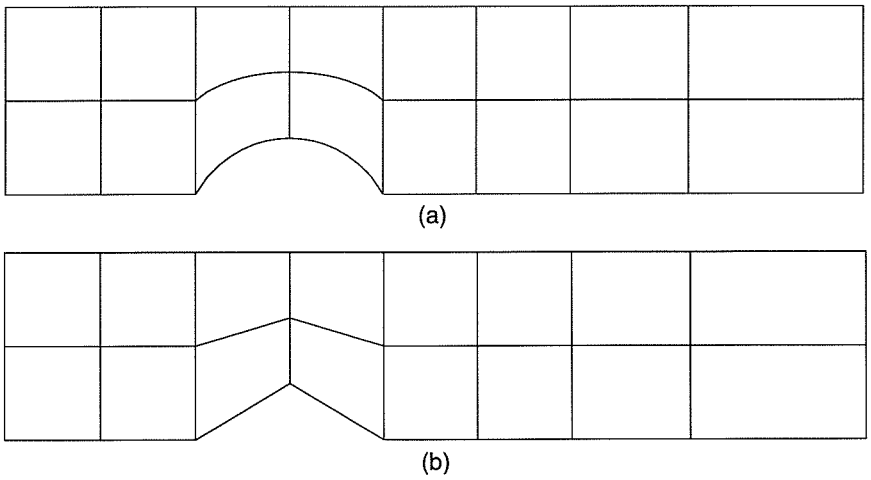

Figure 4.9: Illustration of the coarsest grid $5 \times 3 \times 3$ in a $xy-$ plane.

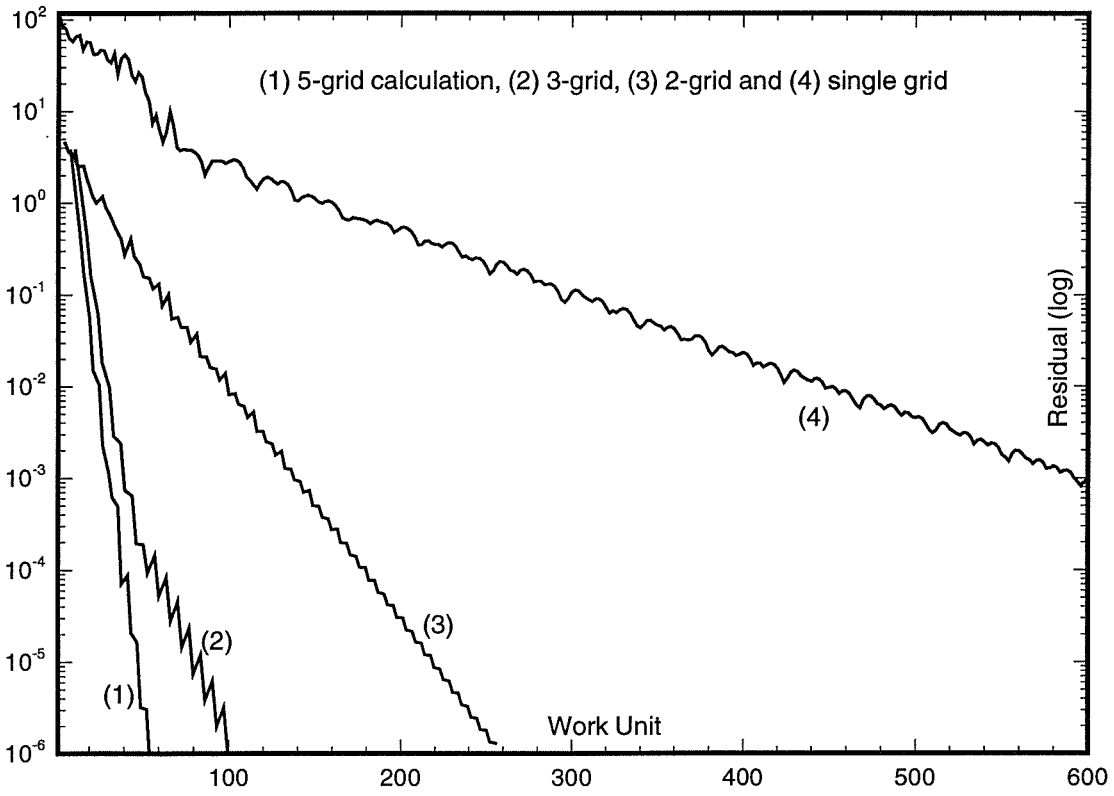

Figure 4.10: Convergence history for computation of laminar flow in a duct using finest grid $129 \times 33 \times 33$.

Shyy *et al* (1993b). The higher-level MG computations generated a substantial improvement in the convergence rate and a 5-level MG method resulted in a speed-up of the convergence rate which was two times that of the 3-level MG calculation. This clearly shows the advantage of the proposed method for multigrid calculations when the coarse grid becomes degraded in spatial resolution.

4.9.3 Laminar Flow in a Strongly Curved 90^0 Pipe

In the last example, flow in a strongly curved pipe with a 90^0 bend was computed. Such flows have been studied experimentally and numerically by a number of authors (for example Smith *et al* (1993)) and contain important characteristics, including strong secondary flow and strong grid curvature. The adopted geometry is similar to that used in chapter 2 but with a more strongly curved bend of 90^0 with the radius of the bend curvature equal to 1.5 times the pipe diameter (see Figure 4.11). The bend was fitted with a straight pipe of two diameters length in the upstream direction and a sufficiently long pipe downstream.

Figure 4.11 shows the finest grid of $33 \times 33 \times 97$ used for the present calculation. A non-uniform grid in the downstream direction was adopted in order to place the outlet boundary far enough into the uniform flow region. Computations were carried out using three different multigrid levels. For the 5-level multigrid calculation, the coarsest grid consisted of $2 \times 1 \times 6$ cells (as shown in Figure 4.12). As in the last example, Figure 4.12b was obtained from the finest grid by restriction of the grid lines, and Figure 4.12a was obtained directly from the the coarsest grid nodes. It can be seen that the geometric representation of the coarsest grid in Figure 4.12a is seriously degraded.

The convergence behavior for computations using 4 different grid levels is shown in Figure 4.13. Again, the present results show that the number of effective grid levels useful for the multigrid method is not limited by the geometrical complexity. The acceleration

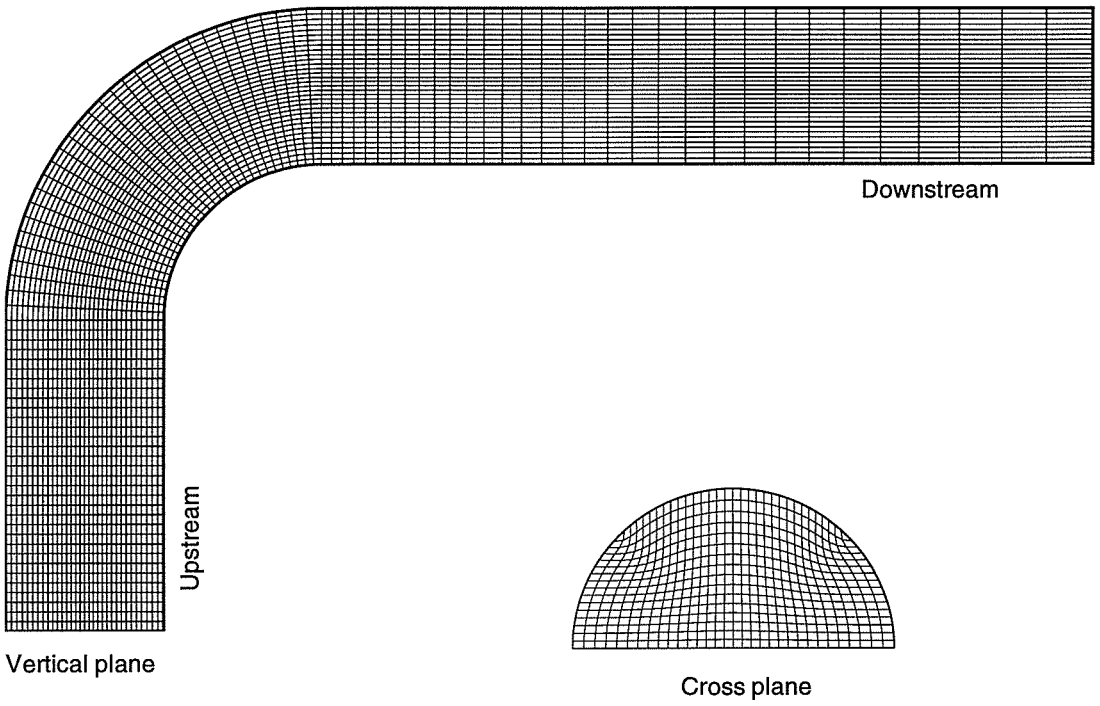

Figure 4.11: Finest grid in the MG calculation of curved pipe flow, $33 \times 33 \times 97$.

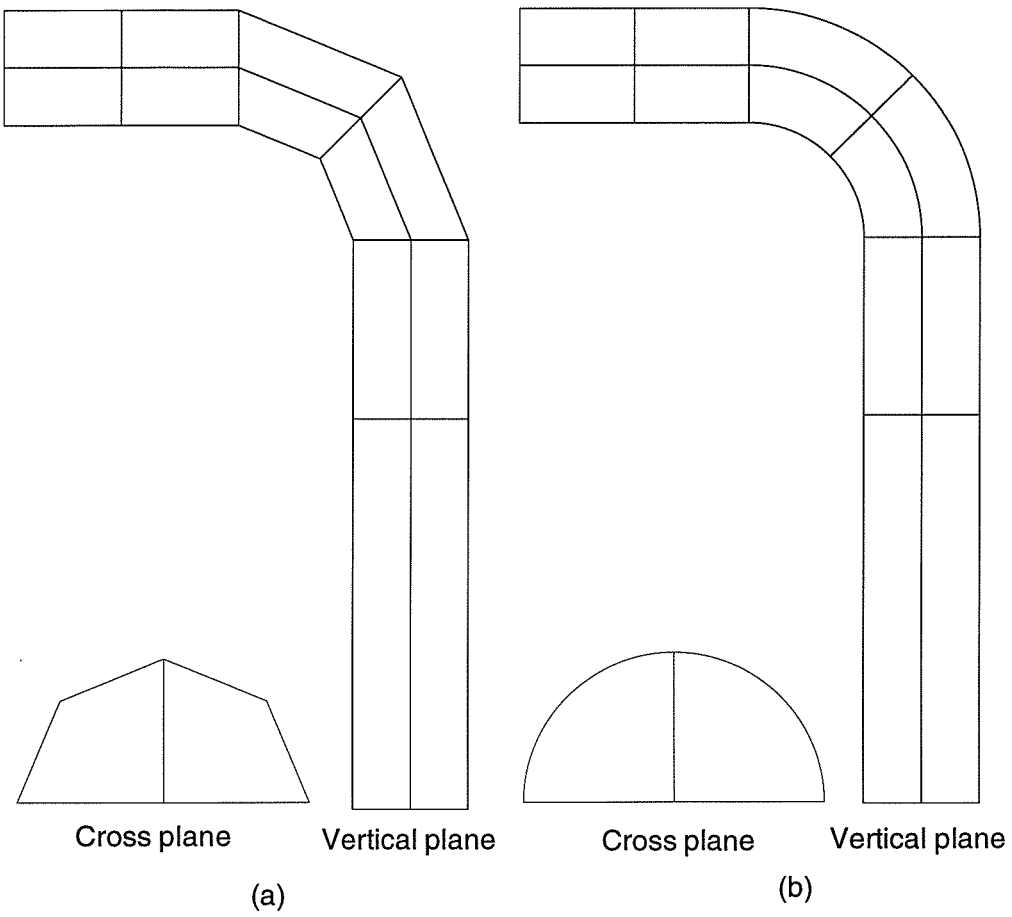

Figure 4.12: Coarsest grid $3 \times 2 \times 7$: (a) grid restricted from the finest grid; (b) grid obtained from the coarsest grid nodes.

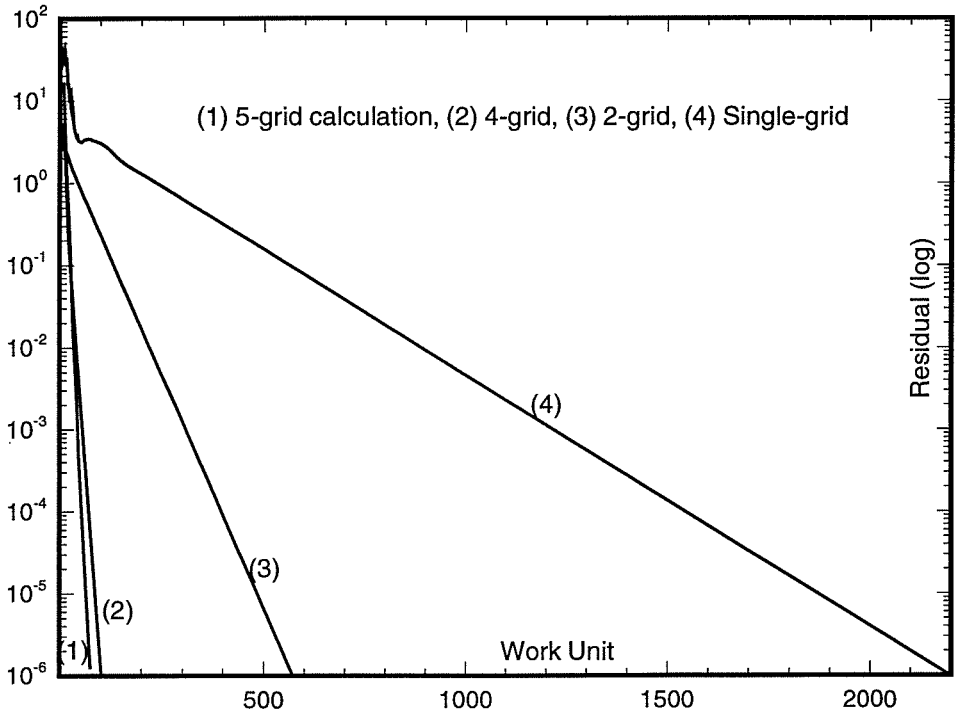

Figure 4.13: Convergence histories for computations using different grid levels.

ratio of the multigrid method increases when the number of multigrid levels is increased. With the 5-level multigrid calculation, the convergence rate accelerates more than 40 times compared to the traditional single-grid method. However, when the number of multigrid levels is reduced to 4, the speed-up ratio is reduced to about 28. This result shows the necessity of using higher grid levels for an efficient multigrid method. This type of flow was also studied by Smith *et al* (1993) using a multigrid algorithm and a reduction of the overall CPU time up to 20% was observed compared with the single-grid method. The speed-up ratio of the multigrid method obtained using the 5-level grid in the present study is found to be much higher than that reported by Smith *et al* (1993).

4.9.4 Developing Turbulent Flow in A Strongly Curved 90^0 Pipe

This example considers the developing turbulent flow in a strongly curved pipe. The same geometry as that used for laminar flow in the previous example was adopted. Computations were carried out for a Reynolds number of 10^5 with grids consisting of $17\times17\times49$ and $33\times33\times129$ nodes using four and five-level multigrid cycles respectively. Figure 4.14 presents a typical plot of the convergence behavior for computations using the multigrid and single grid methods for the $17 \times 17 \times 49$ grid. The fast convergence of the multigrid method can be seen. A reduction of 85% in CPU time was achieved. The converged solution for the fine grid $33 \times 33 \times 129$ was not carried out using the single-grid method due to the long computer time required, but the estimated speed-up ratio of the multigrid acceleration based on the first 500 iterations of the single grid calculation is more than 20 (or 95% reduction). This result is quite encouraging for multigrid calculations of turbulent flows.

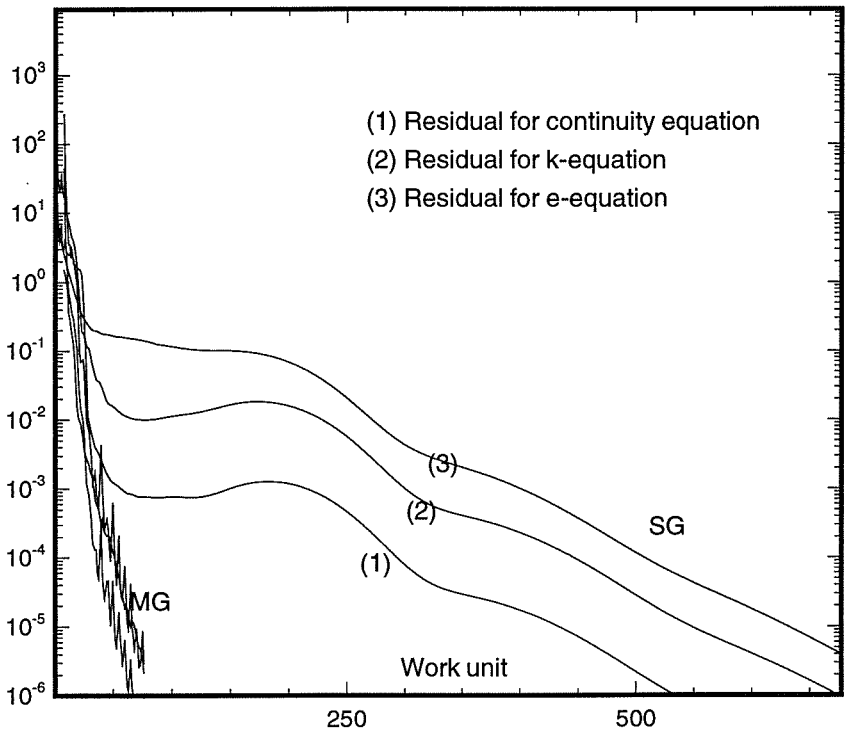

Figure 4.14: Convergence history for the computation of a developing turbulent flow in a strongly curved pipe.

4.10 Closure

An efficient multigrid procedure for three-dimensional laminar/turbulent flows in general curvilinear grids is presented based on the curvilinear coordinate-based method developed in the previous two chapters. It is shown that in certain cases, the ordinary formulation of the coarse-grid equations may result in inconsistent governing equations between fine and coarse grids which reduces multigrid efficiency. Special procedures need to be taken for the formulation of the coarse-grid equations in curvilinear grids and the treatment of the 'wall function' in turbulent flows. Computational tests show that the multigrid algorithm developed here efficiently reduces overall CPU time for both laminar and turbulent flows. On a very fine grid, the CPU time for the multigrid method was found to be about one hundred times less than that for a single-grid method. This clearly shows the necessity of multigrid method for computations using dense grids. Unlike the observations reported in the literature, the number of effective grid levels useful for the multigrid method is not limited by the geometrical complexities with the present method, even for seriously degraded coarse grids.

Chapter 5

Domain Segmentation and the CMGFD Code

In this chapter, the methods developed in the previous three chapters are generalized to non-structured curvilinear grids by using a domain segmentation technique (multi-block method). This technique divides the domain of interest into different sub-domains. Solutions are obtained by iteratively applying a solver to each sub-domain. The domain segmentation strategy adopted here is described in this chapter. Particular attention is given to the communication among neighbouring sub-domains. The performance of the multi-block method is investigated through several computational examples which show that the developed method has the capability to deal with complex configurations using multi-block curvilinear grids. A curvilinear, coordinate-based CFD code, called CMGFD, is developed based on the methods described in this and previous chapters, and an existing CFD code called MGFD.

5.1 Introduction

With the methods developed in the previous chapters, laminar/turbulent flows in arbitrary single-block geometries can be solved with appropriate curvilinear grids. However, only structured, curvilinear grids can be used with those methods. It would be difficult to treat multiple flow regions using single block grids, such as that in film cooling of a turbine blade.

One approach to deal with multiple flow regions with finite volume methods is the use of a domain decomposition technique. This technique has become popular in dealing

with complex geometries in recent years. Leschziner and Dimitriadis (1989) presented a calculation procedure for flows in a domain consisting of a main duct and a side branch which may intersect at any angle. They used a multi-block, non-orthogonal grid to represent domains in two conjunction ducts. At the duct interface, the computational sub-domains overlap one set of finite volumes; those pertaining to the velocities normal to the interface. Perng *et al* (1991) and Hinatsu *et al* (1991) developed a multigrid domain-splitting technique for simulating incompressible flows in complex geometries. In this method, the momentum equations are solved independently on each subdomain while the pressure field is computed simultaneously on the entire flow field by a multigrid method. However, they only applied the method to several two-dimensional flows in relatively simple geometries.

In this chapter, a domain segmentation technique in conjunction with the methods developed in the previous three chapters are developed to allow the use of block-structured (multi-block) curvilinear grids. Particular attention is given to the data transfers between neighbouring blocks. Several computational examples are carried out to validate the method. The methods described in this and the previous three chapters are implemented in the curvilinear, coordinate-based CFD code CMGFD. The development of this code is summarized.

5.2 Domain Segmentation Technique

The idea of the domain segmentation technique is not complicated. The domain of interest is segmented into an arbitrary number of sub-domains, called computation blocks, and each block is covered by its own Cartesian or curvilinear grid. A solver is then repeatly applied to each block to obtain a converged solution in the whole domain. However, the development of a generally-applicable 3D multi-block CFD code is not trivial. A 3D

multi-block and multigrid CFD code, called MGFD, was developed by Nowak (1992). The code is written generally which allows an arbitrary number of blocks to be used. It can be used to solve flows in arbitrary geometries which may be decomposed into a number of rectangular sub-domains. In the present study, the domain segmentation technique of Nowak (1992) is generalized to curvilinear grids by using the methods developed in the previous chapters. This practice will allow the use of non-structured, curvilinear grids to deal with arbitrary geometries with multiple flow regions. The developed multi-block method is described in the following paragraphs.

First, the domain of interest is segmented into a number of subdomains, which are called computational blocks. The segmentation of a given domain can be arbitrary and depends on the configuration. Each block overlaps with its neighbouring blocks by two sets of cells to facilitate the communication among blocks.

In each block, the single-block solution method developed in chapters 2 and 3 is applied to solve the governing equations. First, from either an initial-guess solution or an intermediate solution obtained from the last iteration, the auxiliary Cartesian velocity components and the turbulence viscosity are calculated. Next, the coefficients and source terms in all the discrete governing equations are calculated using the formulations presented in chapter 2 and 3. The calculation of the coefficients and source terms requires the geometric quantities and dependent variables at neighbouring control cells. This procedure needs special treatment at block boundaries. For boundaries of the block which fall on flow boundaries, such as a wall or an inlet boundary, the actual boundary conditions are applied. For boundaries which fall in a neighbouring block, the dependent variables and the physical geometric quantities are obtained from the neighbouring block. The details will be discussed in the next section. After the discrete governing equations are obtained, the alternative line-coupled iterative procedure described in chapter 3 can be used to update the dependent variables.

The above procedure is applied to all the blocks sequentially. After sweeping all the blocks once, the turbulence viscosity and auxiliary Cartesian velocity components are recalculated for each block and all the coefficients and source terms are recalculated based on the updated turbulence viscosity and velocity unknowns. The entire procedure is cycled through all the blocks until the residuals become sufficiently small.

Since the unknowns in only one block are solved at one time, the storage required for the quantities in the discrete equations is much less than that required when the whole flow domain is solved together. The number of cells contained in each block is usually small and is independent of the number of cells used in the whole flow domain, so the computer storage required can be reduced substantially. More importantly, this solution procedure is able to deal with flows in multiple flow regions.

5.3 Communication Between Blocks

Efficient communication between the blocks has to be established in order to obtain accurate solutions and fast convergence. This is done by appropriately transferring data between adjacent blocks in each iteration. There are different methods to transfer data between neighbouring blocks. In the present study, the method used by Perng (1991) and Nowak (1992) is followed. In this method, each computational block is extended into its neighbouring blocks by two sets of cells. The discrete governing equations on the extended cells are formulated using the physical geometrical quantities and dependent variables obtained from the neighbouring blocks. The discrete governing equations on the extended domains are solved in the same way as those on ordinary cells by using a line-coupled solver.

In the CMGFD code, three types of multi-block curvilinear grids are used which require different treatments: (1) the grid lines in adjacent blocks match at the interface

with complete continuity, (2) the grid lines match at the interface but with discontinuous grid line slopes, and (3) the grid lines do not match at interface. For the first type of grid, an interior boundary condition is used which ensures that both mass and momentum conservation are satisfied at the interface. This interior boundary condition in each block is implemented by using the overlapping cells in each of the neighbouring blocks. For curvilinear grids, the formulation of governing equations on the extended cells in a block requires the grid physical geometric quantities in its neighbouring blocks. The transfer of the grid geometric quantities from one block to another is carried out at the beginning of the solution procedure. The unknowns on the extended boundaries are prescribed by using the intermediate solution of the associated neighbouring blocks. When the coupled Gauss-Seidel solution solver has swept over all the computational blocks, the unknowns in the whole flow domain, including all the interfaces, are updated. The converged solution satisfies both mass and momentum conservation near the interfaces. Therefore, such an approach will not produce any additional inaccuracy near the interface.

For the second type of grid, since the grid lines change directions abruptly across the interfaces (see in Figure 5.1), the derivatives of the grid coordinates are not continuous across the interface. Therefore, the ordinary numerical formulations using derivatives of grid coordinates may produce significant numerical errors. With the numerical formulations described in chapter 2, the coordinate derivatives are avoided. However, interpolations are required to obtain the physical geometrical quantities and the velocity components at the positions where they are not defined. These interpolations near the interface are important as they may cause significant numerical error. This is discussed as follows.

For convenience of discussion, the scheme is presented for the two-dimensional case but it can be generalized to three-dimensional problems without additional difficulties. Figure 5.1 shows a two-block grid with discontinuous grid line slopes at the block interface. Due

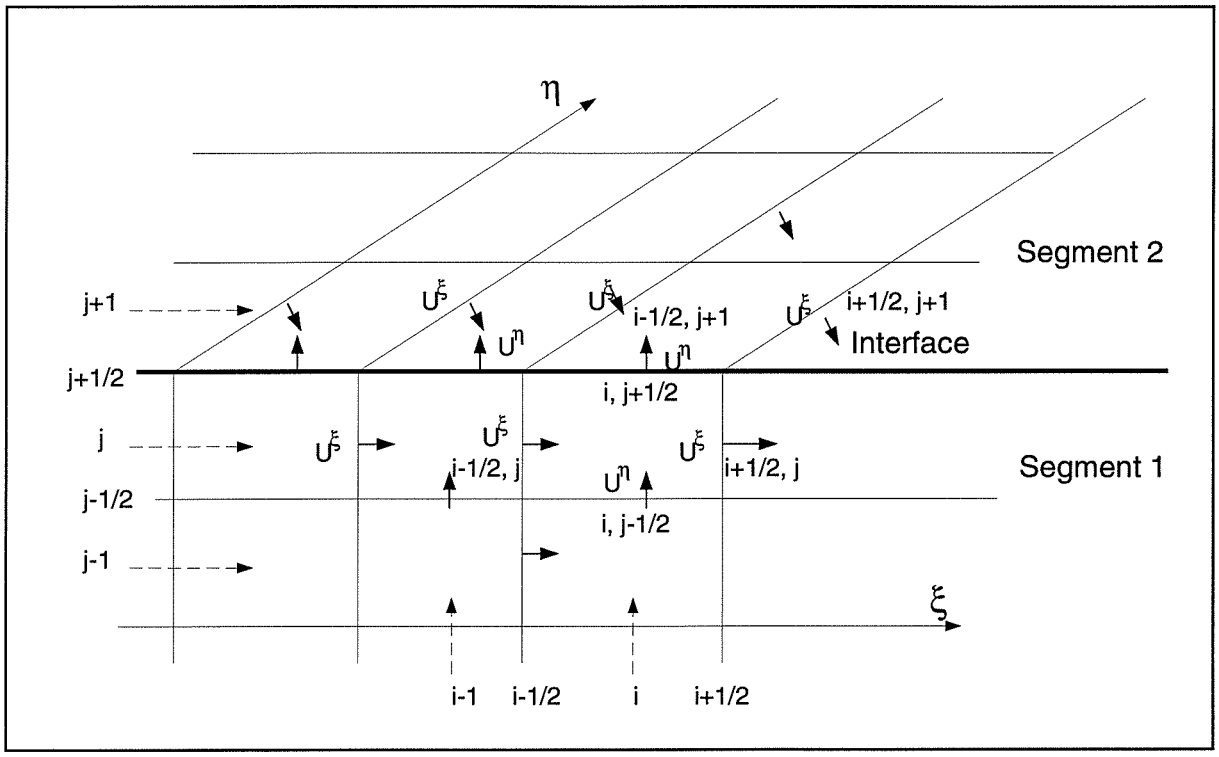

Figure 5.1: Illustration of multi-block curvilinear grids with discontinuous grid line slopes across the interface.

to the change of η grid line slopes, the surface normal vectors $\mathbf{e}^1$ (or contravariant vector $\mathbf{a}^1$) change direction abruptly across the interface. From equation (2.25) we know that the velocity component U^ξ is grid oriented and depends on the surface normal vector $\mathbf{e}^1$. Therefore, the change of grid line direction can produce a large jump of U^ξ across the interface even with constant a velocity field, and the ordinary interpolation becomes inadequate.

The numerical formulation of the momentum equations needs two (three for 3D) velocity components at each velocity position. Since only one velocity component is defined at a velocity position (control volume surface) in a staggered grid arrangement, other components have to be obtained by interpolation. In particular, interpolation is needed to obtain the velocity U^ξ at each U^η position on the interface. As an example, we consider the interpolation procedure at the point (i, j+1/2) on the interface. Here, the $U^\xi-$ velocity positions are marked with the (i-1/2, j), (i+1/2, j), etc., and the $U^\eta-$ velocity positions are marked with (i, j-1/2), (i, j+1/2), etc. On the point (i, j+1/2), only U^η is defined and U^ξ has to be obtained from its neighbouring values by interpolation. One natural interpolation scheme is to take the average of the four neighbouring values, i.e.,

$$U^\xi(i, j+\frac{1}{2}) = \frac{U^\xi(i-\frac{1}{2}, j) + U^\xi(i+\frac{1}{2}, j) + U^\xi(i-\frac{1}{2}, j+1) + U^\xi(i+\frac{1}{2}, j+1)}{4} \quad (5.1)$$

As discussed in the previous paragraph, the values of $U^\xi(i-1/2, j)$ and $U^\xi(i+1/2, j)$ in segment 1 can be significantly different from the values at the neighbouring points $U^\xi(i-1/2, j+1)$ and $U^\xi(i+1/2, j+1)$ in segment 2 due to the sudden change of the surface normal vectors $\mathbf{e}^1$ across the interface. Averaging these sometimes significantly large different values unavoidablely produces certain numerical errors. Furthermore, the velocity unknowns in scheme (5.1) have inconsistent base vectors. As discussed in chapter 2, each velocity component U^ξ pertains to a base vector. The surface normal vector $\mathbf{e}^1$ is

used in the present thesis as the base vector to define U^ξ. A base vector for the averaged velocity unknown $U^\xi(i, j+1/2)$ has to be properly defined by interpolation. However, a proper interpolation scheme to define the surface normal vector $\mathbf{e}^1(i, j+1/2)$ at the interface can not be justified in view of the suddenly changed surface normal vectors $\mathbf{e}^1$ across the interface.

In the present study, an interpolation scheme is proposed to avoid these uncertainties of interpolation near the interface. Instead of the full average scheme (5.1), a one-side interpolation is used, namely, the interpolation at the interface is only taken from one side of the interface. For example, the velocity component $U^\xi(i, j+1/2)$ at the interface is averaged only from one segment, either from segment 1,

$$U^\xi(i, j+\frac{1}{2}) = \frac{U^\xi(i-\frac{1}{2}, j) + U^\xi(i+\frac{1}{2}, j)}{2} \tag{5.2}$$

or from segment 2,

$$U^\xi(i, j+\frac{1}{2}) = \frac{U^\xi(i-\frac{1}{2}, j+1) + U^\xi(i+\frac{1}{2}, j+1)}{2} \tag{5.3}$$

If the interpolation is taken in segment 1, for instance, using equation (5.2), the surface normal vector $\mathbf{e}^1$ will be calculated in the same way by averaging the two surface normal vectors in segment 1 without using surface normal vectors from segment 2. Even though the scheme (5.2) has only first order of accuracy, the base vectors (surface normal vectors) for the three velocity components $U^\xi(i, j+1/2)$, $U^\xi(i-1/2, j)$ and $U^\xi(i+1/2, j)$ are consistent since there is no significant difference among them. This avoids the inconsistency of the base vectors (the surface normal vectors) associated with the velocity components used in the full average scheme (5.1).

The choice of scheme (5.2) or (5.3) can follow the idea of the upwind difference scheme. If $U^\xi(i, j+1/2) > 0$, the fluid flows from segment 1 to segment 2. The velocity components U^ξ at point (i,j+1/1) are more likely to be influenced by those in segment

1. Therefore, the interpolation scheme (5.2) based on segment 1 is used. Otherwise the scheme (5.3) based on segment 2 is adopted.

For the third type of grid, since the grid lines do not meet at the interface the extended cells can not be obtained directly from their neighbouring blocks. To reduce the inaccuracy caused by interpolation, only semi-matched grids at the interface are considered, namely grid lines in one block which meet with a coarser or finer grid of another block (the same concepts as those used in a multigrid method). An illustration of this type of grid is given in Figure 5.2. In this figure, the grid in Segment 1 (bottom) is denser than that in Segment 2. For such grids, the extended cells for a block can be constructed using a finer or coarser grid from its neighbouring blocks as illustrated in Figure 5.2. From Figure 5.2 it can be seen that the extended cells of Segment 1 are just the finer grid of Segment 2. The grid physical geometric quantities and the unknowns in the additional cells are obtained from the corresponding values in the neighbouring blocks by restriction or prolongation with the same strategy as that used in the multigrid method described in chapter 4 (see Section 4.4). Such an assumption reduces the computational effort significantly for non-matched curvilinear grids.

5.4 The CMGFD Code

As mentioned earlier, the CMGFD code was developed based on an existing CFD code, MGFD, at UBC. The code was developed in four steps. In each step, detailed investigation of the method and validation of the code were carried out in order to achieve an efficient and accurate code. The methods described in chapters 2-5 have been implemented in the code. This is discussed in the following paragraphs.

Unlike Cartesian coordinate-based calculations, the computational grid for a complex

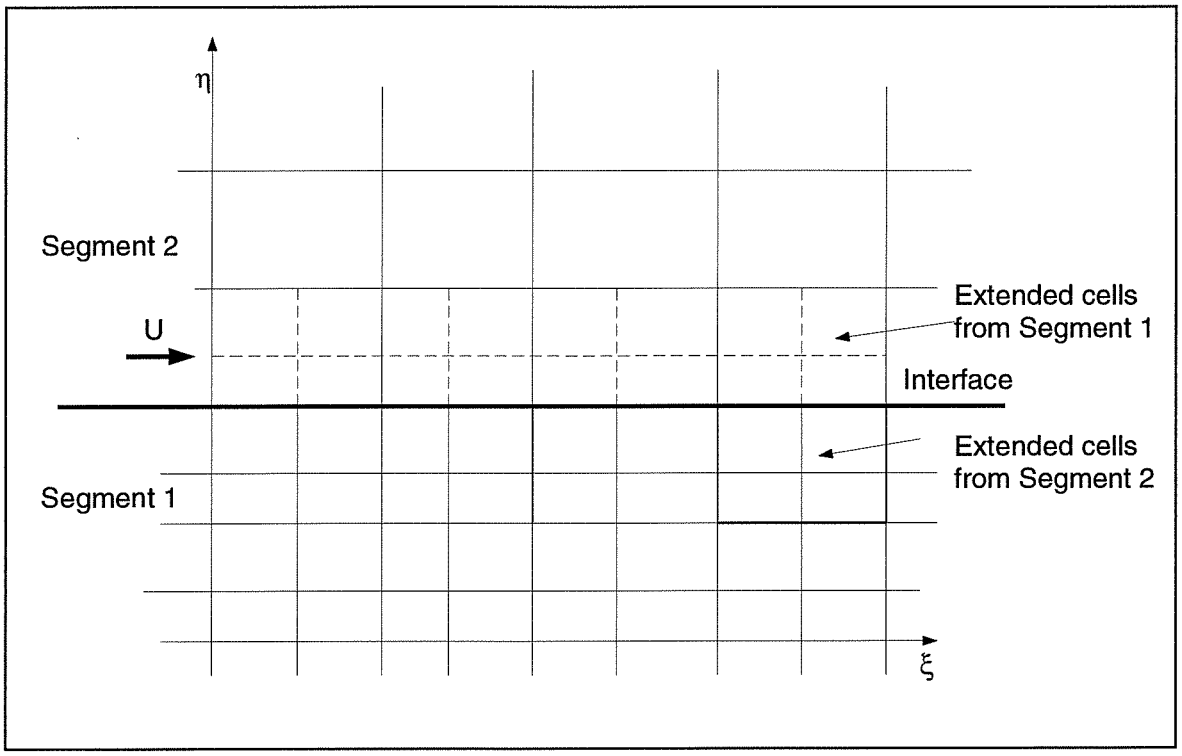

Figure 5.2: Illustration of the semi-matched multi-block curvilinear grids.

geometry is usually not simple to generate. For generality, the code assumes that a curvilinear grid is available for a given problem and leaves the grid generation task for various grid generation techniques, such as one which is described in chapter 6. Therefore, the program GEO developed for generating various non-uniform Cartesian grids in MGFD code actually will not be used in the CMGFD code.

For a given grid, the Cartesian coordinates of all the grid nodes are loaded into the CMGFD code. A subroutine package has been developed to deal with the curvilinear grids and arrange the basic block structures. The grid geometric quantities at cells in each block are calculated from the given grid nodes. First, the surface normal vectors, surface areas and volumes are calculated using the grid nodes. The unit tangent vectors are then calculated from the surface normal vectors by the orthogonal relations. Secondly, the physical geometric quantities are extended two sets of cells outside the block to facilitate the formulation of the governing equations. For a boundary coinciding with a flow boundary, the physical geometric quantities outside the boundary are taken to be equal to the nearest values. For a block boundary falling in a neighbouring block, the geometric quantities are directly taken from this neighbouring block.

There are two different approaches for storing the geometric quantities. One approach is calculating only one set of basic geometric quantities, for instance the surface area vectors, and storing them in the computer memory. All other quantities used in the formulation of governing equations are calculated from these basic geometric quantities in each iteration. The advantage of such an approach is that substantial computer memory can be saved. However, recalculation of these quantities in each iteration requires significant computer time. In the present study, an alternative approach is followed and all the geometric quantities are calculated at the beginning and fixed throughout the solution procedure.

As a by-product, the grid is checked using the geometric quantities, un-usable grids

such as the grids with zero or negative volume cells, are reported to the user. This often happens due to the wrong output of the grid generation code or inconsistent geometric information loaded to the CMGFD code. In addition, there are a number of physical characteristics that affect the quality of the grid, namely orthogonality, smoothness, cell expansion ratios, etc. All these quantities are checked to identify low quality grids for possible improvement.

The computed geometric quantities are used to formulate the governing equations. This was done by properly modifying the MGFD code. All the code involved with grid information (such as cell dimensions dx, dy, dz) are modified throughout the code. The formulations for the coefficients and source terms are substantially changed from those in the MGFD code. The source terms in the discrete momentum and continuity equations are particularly complex in the present formulation as they include non-orthogonal pressure, non-orthogonal diffusion, and grid curvature terms. Their calculation requires various interpolations of velocity, pressure and geometric quantities. It is found that the interpolation procedures are important in terms of computational accuracy and efficiency. An interpolating procedure was developed through testing a number of problems.

The multigrid method proposed in chapter 4 is implemented in the CMGFD code by using a new subroutine package. This package contains three subroutines namely, 'FMG', 'FAS' and 'GEOM'. The FMG and FAS subroutines execute the standard FMG-FAS procedure. They are written generally which allows easy implementation of multigrid methods to other single-grid CFD codes. This package will also be used in the multigrid elliptic grid generation code, MBEGG (see the next chapter). The GEOM subroutine is written to calculate the physical geometric quantities on all the coarse grids using the method developed in chapter 4.

So far, the code is limited to 3D steady, incompressible laminar/turbulent flows with scalar transfer. For turbulent flows, the standard $k - \epsilon$ model with the wall function

treatment is used for the turbulence closure. The CMGFD code has the capability to deal with arbitrary 2D/3D geometries by using separate curvilinear grids in different flow regions. There is no restriction for curvilinear grids to be used and no limitation for the number of segments in each flow region. An efficient multigrid method is implemented in the code and promises fast convergence for both laminar and turbulent flows.

5.5 Test Computations

The validation of the CMGFD code has been carried out throughout previous chapters for various geometries using single-block curvilinear grids. In this section, several calculations using multi-block curvilinear grids are presented.

5.5.1 Computations of a L-Shaped Duct

In the first example, laminar flow in a L-shaped duct is calculated. This example allows one to validate the solution method using multi-block curvilinear grids with discontinuous grid slopes across block interfaces. In this study, two computational grids are adopted, one is a two-segment non-orthogonal grid with discontinuous grid slopes and the other is a two-block Cartesian type grid. Figure 5.3 shows the two-block non-orthogonal grid used for the present study. It can be seen that the grid lines change direction significantly (by 90^0) at the interface. For comparison, computation is also carried out using a two-block Cartesian grid as shown in Figure 5.4. Figure 5.5 shows the predicted flow field in the symmetry plane of the duct using the two-block non-orthogonal grid. Two recirculation zones are found in the two corner regions. Figure 5.6 shows the predicted flow field using the orthogonal grid at the same plane. From Figures 5.5 and 5.6 the similarity of results predicted by the non-orthogonal and Cartesian grids can be seen. For

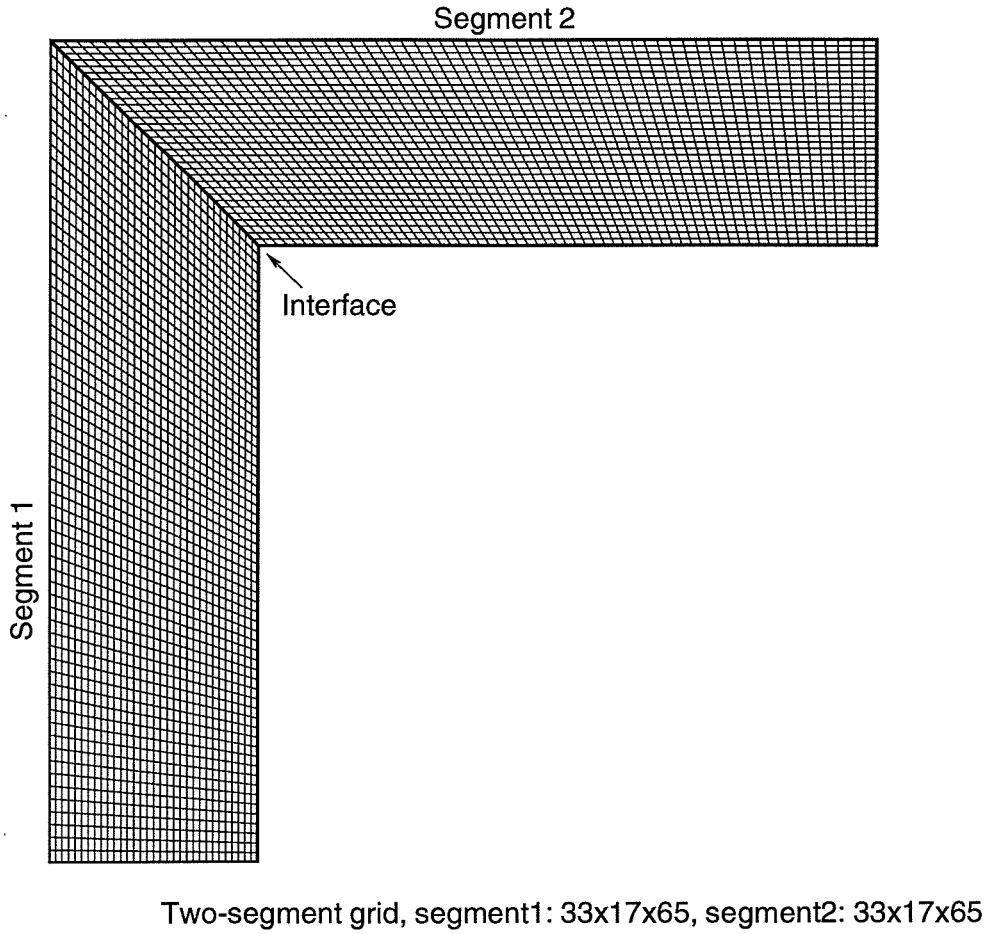

Figure 5.3: Illustration of the adopted two-block non-orthogonal grid at the symmetry plane for the L-shaped duct.

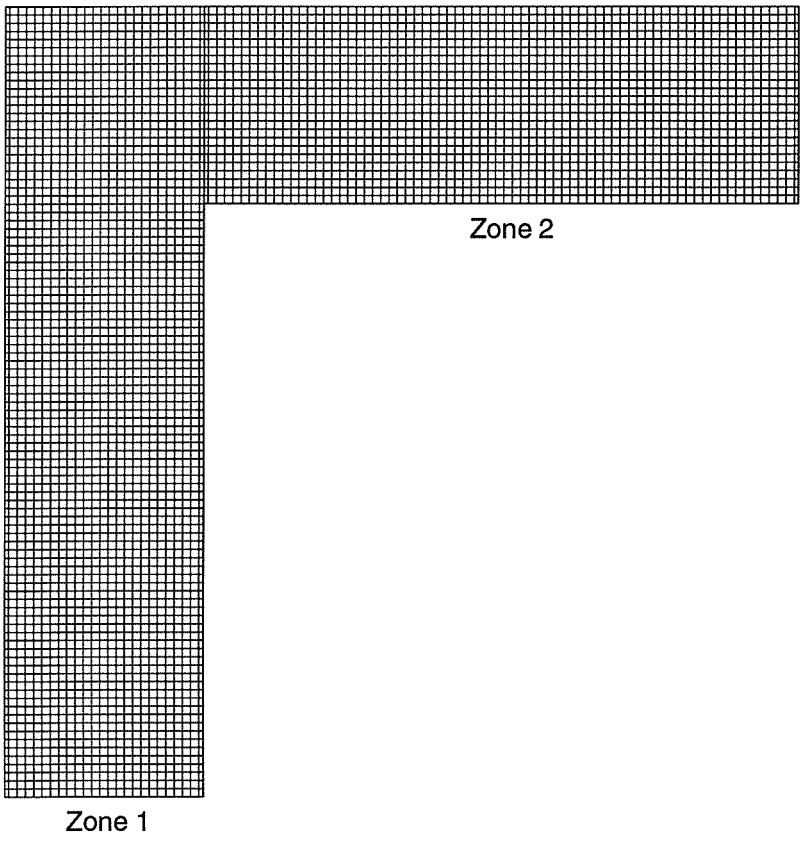

Figure 5.4: The two-block Cartesian grid at the symmetry plane for the L-shaped duct.

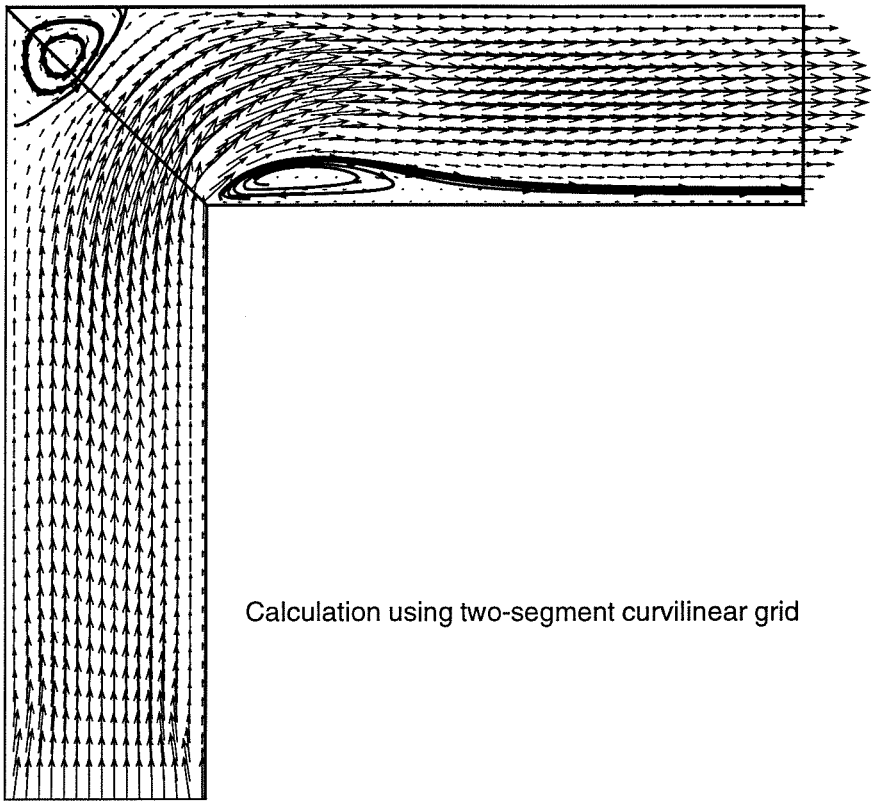

Figure 5.5: Predicted velocity field at the symmetry plane by the non-orthogonal grid.

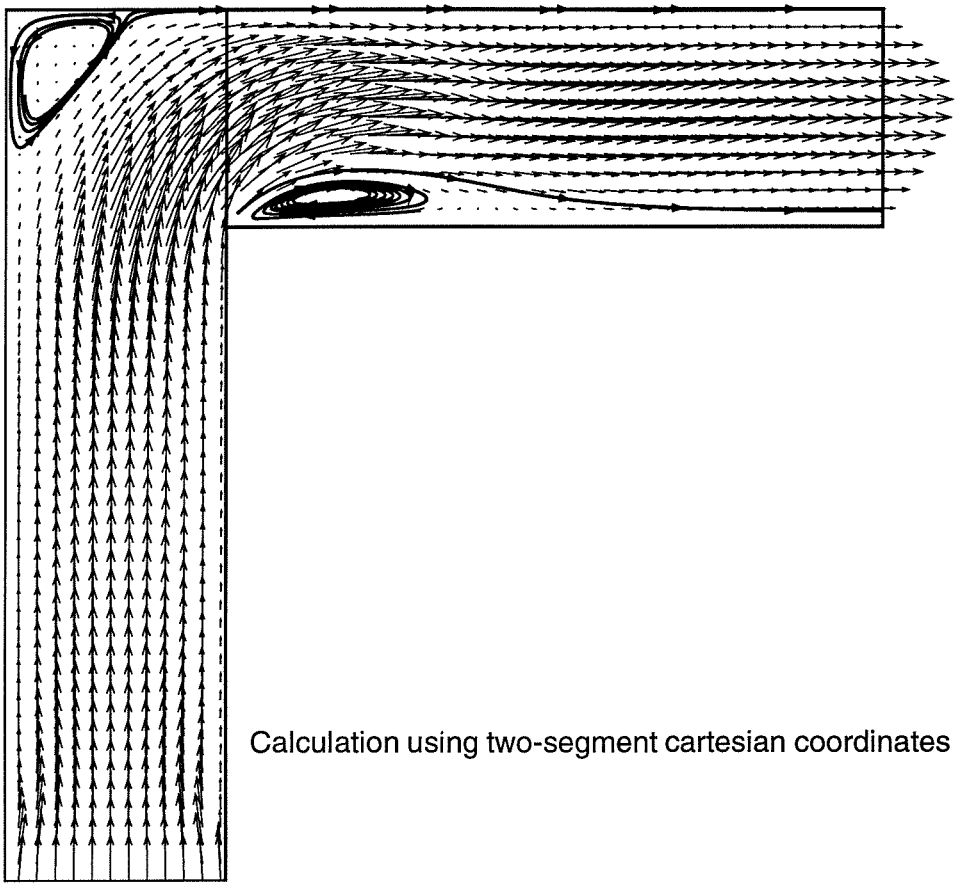

Figure 5.6: Predicted velocity field at the symmetry plane by the orthogonal grid.

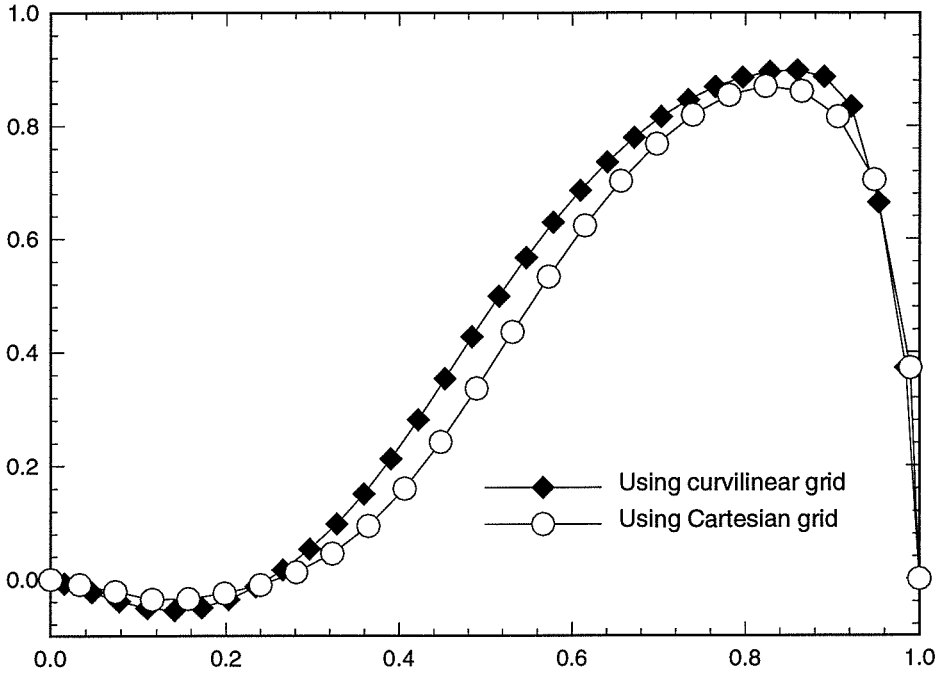

Figure 5.7: Comparison of the u-velocity profiles at the interface shown in Figure 5.3.

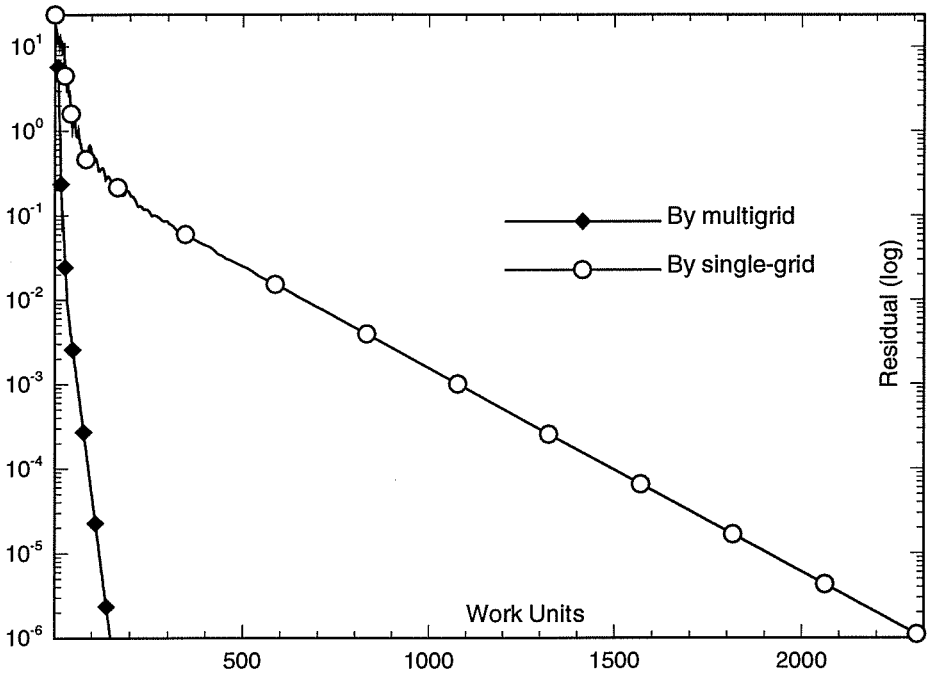

Figure 5.8: Comparison of maximum residuals by multigrid and single-grid calculations for the non-orthogonal grid.

a quantitative comparison, the predicted U-velocity (U is the Cartesian velocity component in the outflow direction) profiles at the centerline of the interface from the two grids are compared in Figure 5.7. Good agreement can be seen. These results show that the developed method allows the use of multiblock grids with significantly discontinuous grid line slopes. Computations are carried out for the non-orthogonal grids with both multigrid and single grid methods. The convergence behaviour of multigrid and single grid calculations are shown in Figure 5.8. It can be seen that the multigrid method converged much faster than the single grid method. This shows the advantage of the multigrid method for computations using multi-block curvilinear grids.

5.5.2 Calculation of A Hydrocyclone

Hydrocyclones have many applications in industry. Figure 5.9 illustrates a conventional cone-cylindrical hydrocyclone. The cylindrical part is closed at the top by a cover, through which the vortex finder protrudes some distance into the cyclone body. The underflow leaves through the opening in the apex of the cone. The liquid enters the hydrocyclone through a tangential inlet (feed opening) and leaves through the vortex finder and underflow orifice.

The developed CMGFD code is applied to simulate fluid flow in a hydrocyclone. Except for the region in and just around the tangential inlet duct, the motion of the fluid within the hydrocyclone has axial symmetry. Therefore, the flow can be treated approximately as two-dimensional, axi-symmetrical, while the angular velocity component is treated as a scalar quantity according to a discussion with Abdullah (1993). Figure 5.10a shows the non-orthogonal grid used for the present calculation. The grid contains three blocks and the third block grid is generated by solving a two-dimensional grid generation system. The predicted velocity field in a vertical plane is shown in Figure 5.10b. From this figure, the separation of fluid can be seen. Most of the incoming fluid moves in an

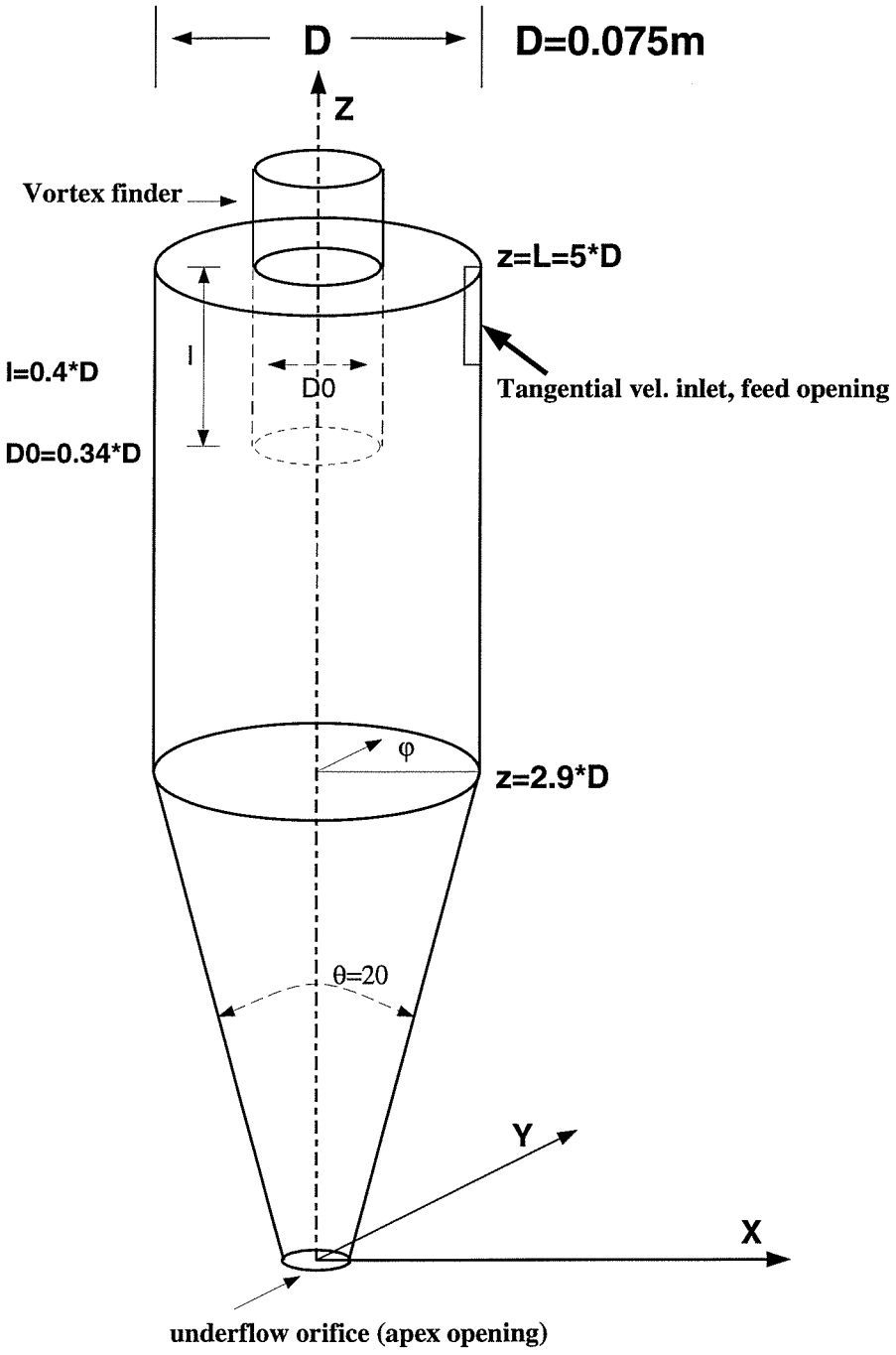

Figure 5.9: Illustration of a conventional hydrocyclone.

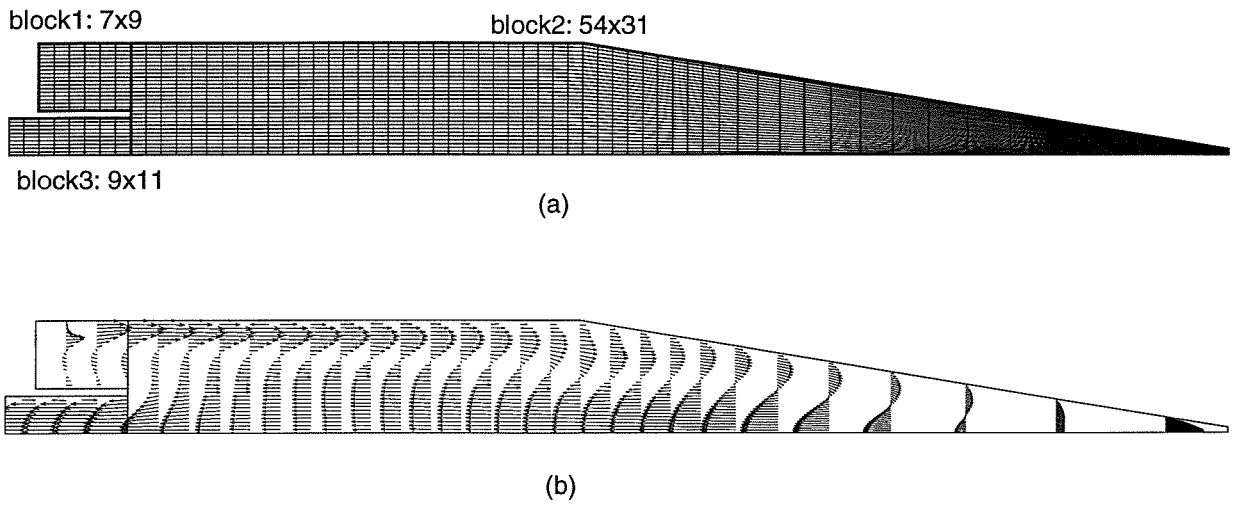

Figure 5.10: (a) 3-block mesh used in the present study, (b) predicted velocity field.

outer helical flow into the outer portion of the inverted cone where it begins to feed across towards the center. Some of the downward flow leaves through the underflow orifice in the apex of the cone while the rest reverses its vertical direction and goes up via the inner helical flow and out through the vortex finder.

5.5.3 Laminar Flows Over A Turbine Blade

The last example in this chapter is a computation for laminar flows over a turbine blade. Such a flow has no direct application to turbine engines since the flow in film cooling of turbine blades is actually turbulent. Nevertheless, the present calculation is useful in providing guidelines for calculation of flows in complex configurations using multi-block curvilinear grids as well as the prediction of film cooling of an actual turbine blade.

The computational domain is chosen according to the UBC experimental turbine blade model of Salcudean *et al* (1993). This model uses four rows of circular injection

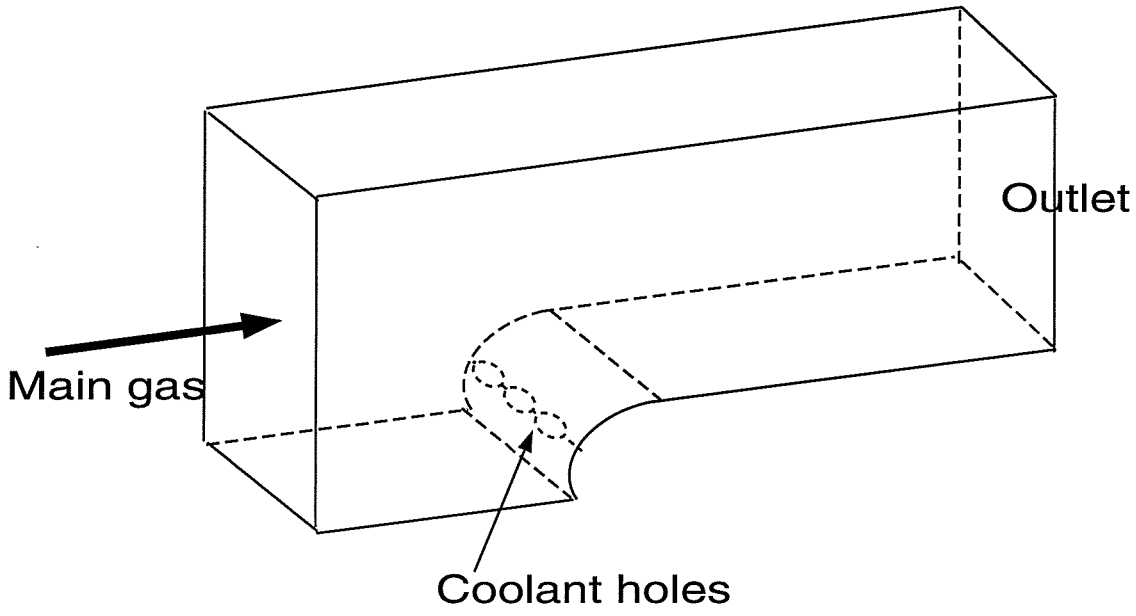

Figure 5.11: Computational domain for a turbine blade model.

tubes located at a semi-circular nose coupled to a flat after-body. Exploiting the advantage of its symmetric property about the stagnation line, only half of the flow domain is taken as the computational domain. Figure 5.11 shows the physical domain selected for the present numerical study where three injection tubes are adopted in a row of circular cooling holes in order to simulate flows with multi-injection holes. The physical domain is segmented into four subregions, a main flow region and three injection coolant regions. In order to obtain a continuous grid at the interface, a rectangular type grid is used for the injection holes instead of cylindrical grids. A multi-block curvilinear grid has been generated using the MBEGG code described in the next chapter. This grid is shown in Figure 5.12.

Computations were carried out for a Reynolds number of 1000 and mass flow ratio of 0.5, where the Reynolds number is defined with respect to the radius of the semi-circular

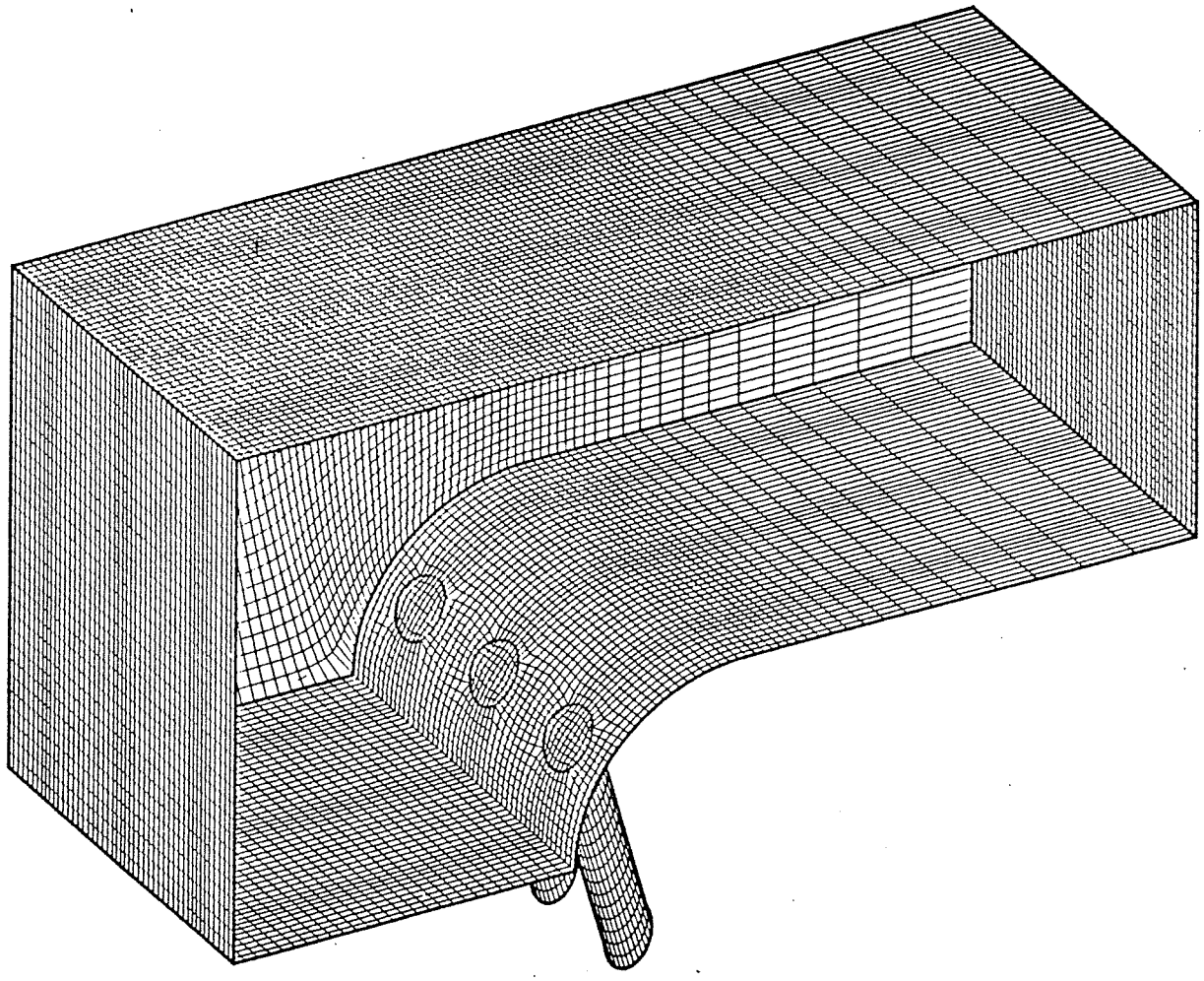

Figure 5.12: A 4-block curvilinear grid generated for the domain shown in Figure 5.11.

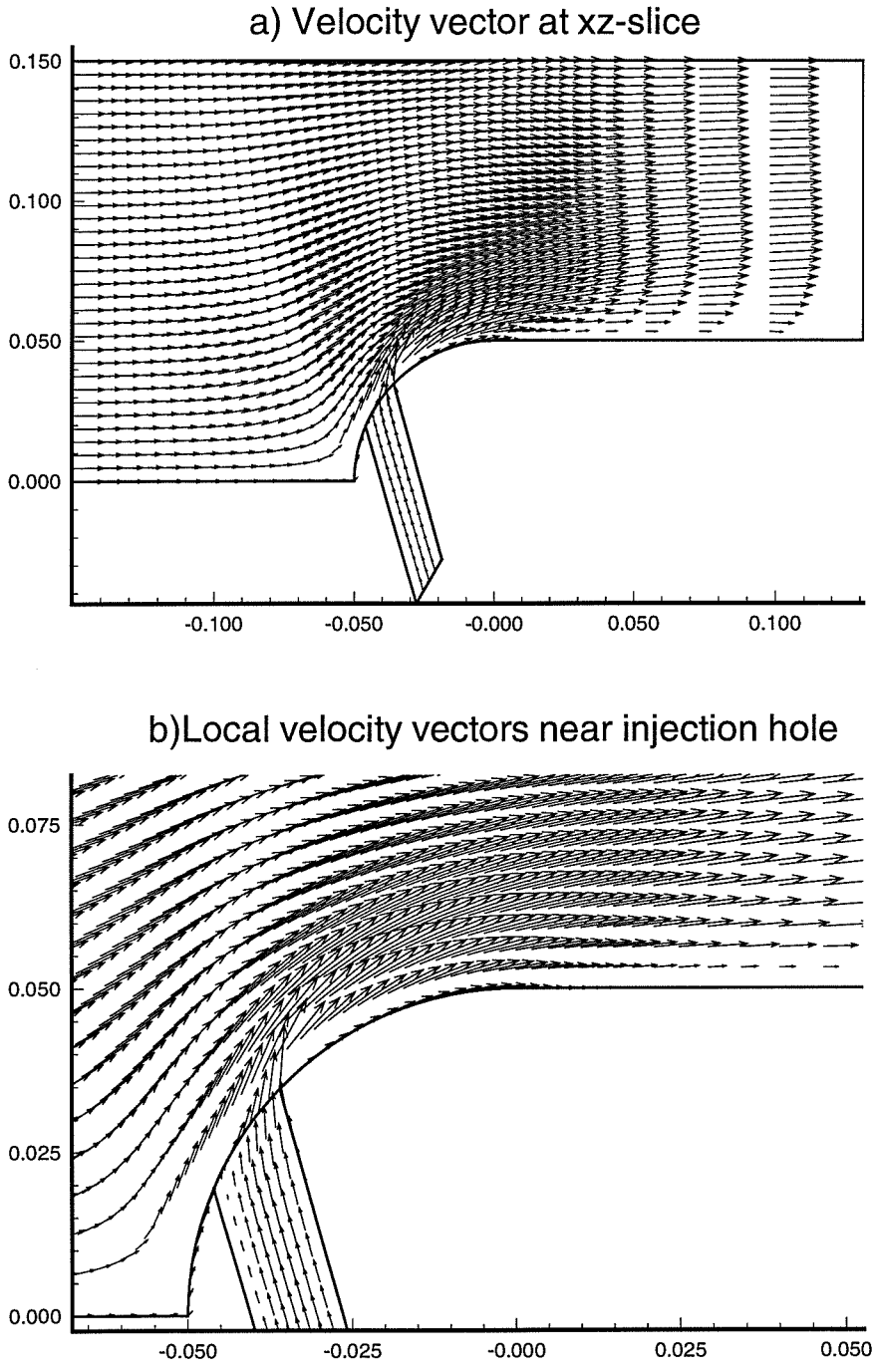

Figure 5.13: The velocity field at the xz-plane through the middle of an injection hole.

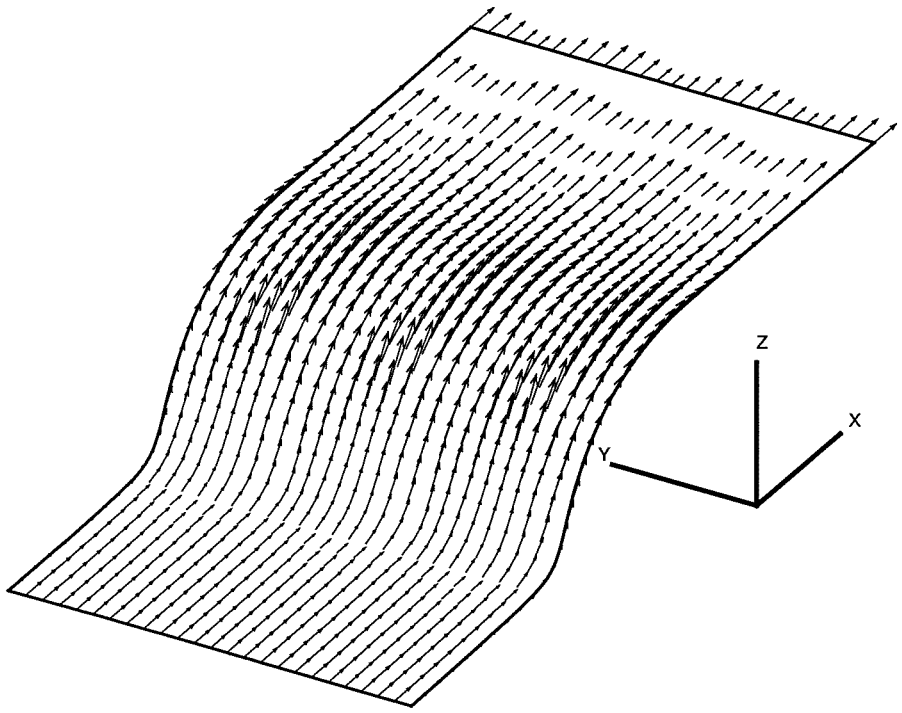

Figure 5.14: The velocity at a surface distanced from the turbine wall by two cells

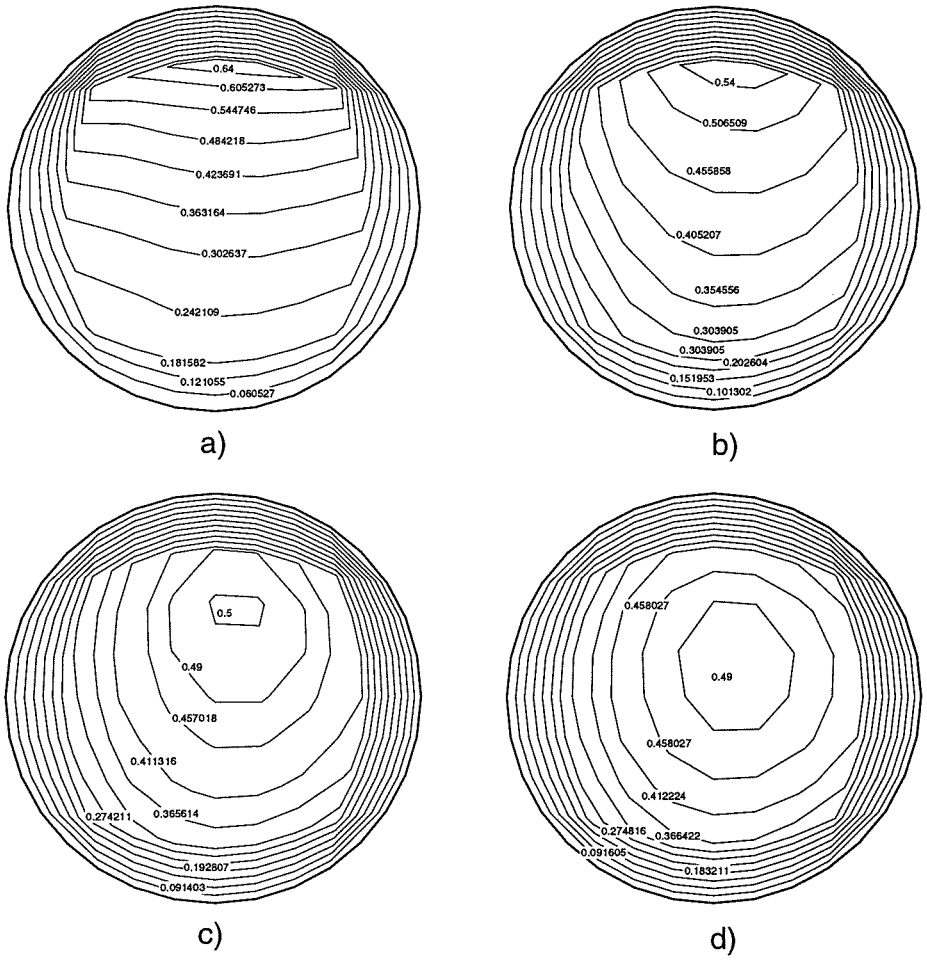

Figure 5.15: The streamline-velocity contour at the cross-plane of an injection hole; a) one cell size distance from the turbine surface; b) two cell size distance from the turbine surface; c) three cell size distance from the turbine surface; d) four cell size distance from the turbine surface.

body and main-stream velocity. Figure 5.13 shows the calculated velocity vectors on a xz-plane through the middle of an injection hole, from which it can be seen that the velocity is very non-uniform at the coolant hole exits. The flow in the injection hole exits is affected by the main stream velocity and strong pressure gradient and, thus, changed rapidly with respect to magnitude, direction, etc. The velocities at a surface distanced from the turbine wall by two cells are plotted in Figure 5.14. Figure 5.15 shows the development of the non-uniform profile near the exit of injection holes. The streamline velocity contour is plotted at the cross-planes of an injection hole at four different locations. A strongly non-uniform velocity profile near the injection hole can be clearly seen, which shows that the use of a uniform velocity profile as the boundary condition at the injection exit, used in many previous studies, could be very inaccurate.

5.6 Closure

A domain segmentation technique in conjunction with the curvilinear coordinate-based method developed in the previous three chapters is described. Communication between different blocks is emphasized. The developed multi-block method allows the treatment of arbitrary complex configurations using non-structured curvilinear grids. A multi-block and multigrid code is developed based on an existing CFD code. The multi-block technique in conjunction with the methods developed in the previous three chapters is implemented in the code. Computational examples show that the developed method and code can be used to solve flows in complex geometries using block-structured curvilinear grids with significant discontinuities in grid line slopes.

Chapter 6

Numerical Grid Generation

In this chapter, the development of a 3D elliptic grid generation code is presented. The code is developed to address the need to generate appropriate grids for analyzing the film cooling of turbine blades. An elliptic grid generation method is used in the code, which solves three nonlinear, elliptic, grid generation equations. A multigrid method is developed to solve these equations which provides an efficient grid generator. Techniques to control grid quantities are introduced. A multi-block grid generation strategy is adopted to allow the generation of block-structured curvilinear grids in arbitrary geometries.

6.1 Introduction

In order to simulate the film cooling process over a turbine blade, suitable grids have to be generated to represent the actual blade geometry. One of the important tasks in the present thesis is to develop an efficient grid generator to generate suitable grids over turbine blades. This includes the development of a multigrid and multi-block grid generation code, called MBEGG.

Grid generation has been investigated extensively in recent years and a large number of research papers have been published on the generation of body-fitted grids for a wide variety of geometries in two as well as three dimensions. A review paper by Thompson (1982) summarized most of the work done before 1981. The grid generation methods used in the literature can be classified into three categories: (1) conformal mapping, (2) algebraic grid generation methods and, (3) elliptic grid generation methods. Among

these methods, the third type of method has the best potential for the generation of grids in complex three-dimensional configurations. In the present study, the elliptic grid generation method proposed by Thompson (1985) is adopted. The method is based on three elliptic partial differential equations. When proper grid nodes are specified at the boundary of a domain, a smooth grid can be generated by solving these equations.

In the grid generation equations, the inhomogeneous terms are problem-dependent functions (which are commonly called control functions). The choice of an appropriate scheme for the evaluation of the control functions is one of the important considerations in an elliptic grid generation method. Methods to achieve certain grid quantities through the control functions, such as stretching and orthogonality, have been implemented in the code. When these grid quantities are initially specified by users, the code can choose the appropriate source terms to obtain the desired grids automatically.

In an elliptic grid generation method, a suitable solution method has to be developed to solve the three nonlinear, elliptic, differential equations. The multigrid method is an ideal solver for the grid generation equations due to the fully elliptic nature of these equations. In the present study, a multigrid method is developed to solve these equations. It is found that the method produces a much faster convergence rate than the traditional single grid method.

For complex configurations, it becomes necessary to partition the physical domain into a number of subregions. Grids are generated in the different subregions. A number of papers have been reported in the literature for generating multi-block (composite) grids in various complex configurations. A detailed discussion can be found in the review paper of Thompson *et al* (1982). In the MBEGG code, a multi-block method is used to allow the treatment of multi-region geometries. The development of the MBEGG code together with the methods adopted is described in the following sections.

6.2 Elliptic Grid Generation

The elliptic grid generation method proposed by Thompson (1985) is based on the following well-known Possion equations:

$$\nabla^2 \xi_k = P_k(x_1, x_2, x_3), \qquad k = 1, 2, 3, \tag{6.1}$$

where (x_1, x_2, x_3) are the Cartesian coordinates in a physical domain and ξ_i (i=1, 2, 3) are the curvilinear coordinates in a regular domain. Equation (6.1) has the same form as the heat conduction equation, where $P_i(x_1, x_2, x_3)$ are heat source terms. The equations establish a correspondence between the Cartesian coordinates (x_1, x_2, x_3) in a complex physical domain and the curvilinear coordinates (ξ_1, ξ_2, ξ_3) in a cuboid domain. However, the equations in this form, yielding (ξ_1, ξ_2, ξ_3) for given (x_1, x_2, x_3) would be difficult to use, especially when specifying the boundaries. Therefore, the following transformed Poisson equations are usually used in the actual grid generation procedure:

$$\begin{aligned} &a_{11}\frac{\partial^2 x_k}{\partial \xi_1^2} + a_{22}\frac{\partial^2 x_k}{\partial \xi_2^2} + a_{33}\frac{\partial^2 x_k}{\partial \xi_3^2} + 2a_{12}\frac{\partial^2 x_k}{\partial \xi_1 \partial \xi_2} + 2a_{13}\frac{\partial^2 x_k}{\partial \xi_1 \partial \xi_3} + 2a_{23}\frac{\partial^2 x_k}{\partial \xi_2 \xi_3} \\ &+ J^2(P_1(\xi_1, \xi_2, \xi_3)\frac{\partial x_k}{\partial \xi_1} + P_2(\xi_1, \xi_2, \xi_3)\frac{\partial x_k}{\partial \xi_2} + P_3(\xi_1, \xi_2, \xi_3)\frac{\partial x_k}{\partial \xi_3}) = 0, \end{aligned} \tag{6.2}$$

where the coefficients a_{ij} are given by the following expression:

$$a_{ij} = \sum_{m=1}^{3} b_{mi} b_{mj}$$

and where b_{ij} is the co-factor of the (i, j) element in the transformation matrix

$$M = (\frac{\partial x^i}{\partial \xi^j})_{3\times 3}$$

with J=det(M) as the Jacobian determinant. In the above equations, the P_i are problem-dependent functions which can be used to control the grid characteristics, such as orthogonality and grid stretching. Thus, an important part of the work in the elliptic grid

generation method is to develop an appropriate scheme for the evaluation of the control functions. When a grid is specified at the boundary of a domain, the smoothest possible grid can be obtained by solving the above grid generation equations.

6.3 Control Functions

To obtain the desired grids, proper control functions have to be constructed. In the MBEGG code, two types of control functions are calculated from the boundary grid nodes which may be specified by the user. The first type of control function controls the grid stretching, and the second controls the grid orthogonality near the boundaries. Such control functions are implemented in the code to allow the users to generate the required grids. The two types of control functions are calculated from the boundary grid nodes prior to the solution procedure. The calculation of the two types of control functions is described in the following paragraphs.

For the convenience of our discussion, the surfaces in a three-dimensional block are numbered. This is illustrated in Figure 6.1. Face 1, for example, is the boundary surface where ξ_1 is fixed to zero.

The stretching functions used here are based on the method of Thomas and Middlecoff (1980). These functions have the following form,

$$P_i^s = -(x_{\xi_i} x_{\xi_i\xi_i} + y_{\xi_i} y_{\xi_i\xi_i} + z_{\xi_i} z_{\xi_i\xi_i})/(x_{\xi_i}^2 + y_{\xi_i}^2 + z_{\xi_i}^2), \qquad i = 1,2,3 \tag{6.3}$$

where the superscript 's' indicates that the functions control stretching and the subscripts ξ_i indicate differentiation. With these stretching functions, the stretching of the grid at boundaries can be transmitted into the interior.

In the MBEGG code, the P_i^s functions are calculated on all the boundaries where ξ_i is not a constant. For instance, P_1^s is calculated on faces 3) to 6) but not on face 1) or 2). This is because the P_1^s function is defined by the deratives with respect to ξ_1 which can

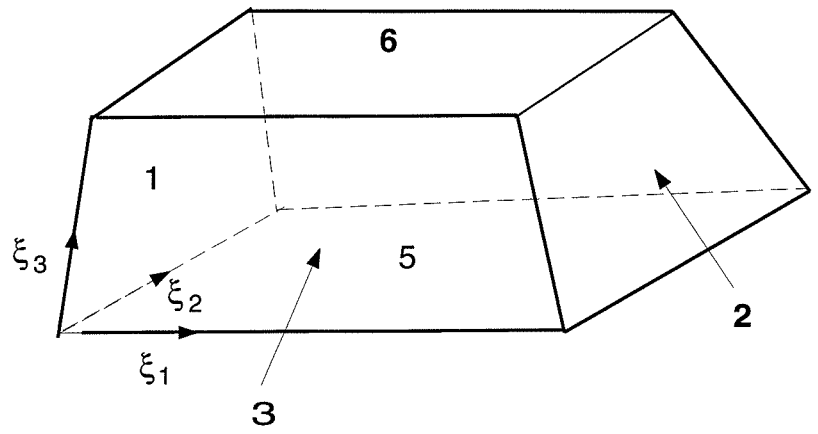

Figure 6.1: Illustration of the face numbers in a 3D block.

not be calculated from the boundary nodes on surfaces 1) and 2). This function can be calculated on faces 3) to 6) by using certain difference approximations. In the MBEGG code, a second order difference scheme is used to discretize all the derivatives involved in the P_i^s functions. After the P_i^s functions are calculated, they are then interpolated into the interior of the grid using certain interpolation methods. These control functions are calculated only once at the beginning of the solution process. This type of control function is very useful for most grid generation problems.

The control functions used to control grid orthogonality near the boundaries used in the MBEGG code are based on those proposed by Steger and Sorenson (1979). The control functions are assumed to have the following form:

$$P_i(\xi_1, \xi_2, \xi_3) = P_i^s(\xi_1, \xi_2, \xi_3) + \sum_{n=1}^{6} R_i^n(\xi_1, \xi_2, \xi_3), \tag{6.4}$$

where the index n is the surface number from 1) to 6) (see Figure 6.1). The first term on the right-hand side of equation (6.4) is the stretching function described in the last paragraph, and the second term is the function to control the orthogonality near the

boundary. Each R_i^n is a function which can be used to control the cell height and orthogonality on face 'n'. The R_i^n are determined as follows. Taking R_i^1 (i=1,2,3) as examples, the functions are assumed to have the following form,

$$R_i^1(\xi_1, \xi_2, \xi_3) = Q_i(\xi_2, \xi_3)e^{-a^i\xi_1} \tag{6.5}$$

where a^i are positive constants. To define appropriate Q_i, equations (6.2) are applied on face 1) to obtain three linear equations for Q_1, Q_2 and Q_3. These equations can be solved to obtain Q_i (i=1,2,3) if all the derivatives involved in equation (6.2) can be calculated from grid nodes on face 1). This means that all possible first and second partial derivatives of x_i with respect to ξ_i must be determined. Derivatives involving only ξ_2 or ξ_3 can be found by simply differencing the given boundary grid nodes. However, derivatives involving ξ_1 can not be calculated in an obvious manner from the boundary grid nodes. These derivatives are found from the constraints imposed upon a line of varying ξ_1 which intersects face 1) as follows: (a) it must be normal to the tangent vector along the ξ_2 direction, (b) it must be normal to the tangent vector along the ξ_3 direction, (c) the length along that line from the surface to the next node must be controlled. These geometric constraints can be expressed in the following equations:

$$\frac{\partial x_1}{\partial \xi_1}\frac{\partial x_1}{\partial \xi_2} + \frac{\partial x_2}{\partial \xi_1}\frac{\partial x_2}{\partial \xi_2} + \frac{\partial x_3}{\partial \xi_1}\frac{\partial x_3}{\partial \xi_2} = 0, \tag{6.6}$$

$$\frac{\partial x_1}{\partial \xi_1}\frac{\partial x_1}{\partial \xi_3} + \frac{\partial x_2}{\partial \xi_1}\frac{\partial x_2}{\partial \xi_3} + \frac{\partial x_3}{\partial \xi_1}\frac{\partial x_3}{\partial \xi_3} = 0, \tag{6.7}$$

$$\frac{\partial x_1}{\partial \xi_1}\frac{\partial x_1}{\partial \xi_1} + \frac{\partial x_2}{\partial \xi_1}\frac{\partial x_2}{\partial \xi_1} + \frac{\partial x_3}{\partial \xi_1}\frac{\partial x_3}{\partial \xi_1} = S^2, \tag{6.8}$$

where S is the desired distance to the first point from face 1). From these equations, the first derivatives with respect to ξ_1 can be solved. The only derivatives remaining to be determined are the second partial derivatives with respect to ξ_1. Those can be found by differencing the existing solution in the iteration process. After all the derivatives

involved in equation (6.2) are calculated, these equations are treated as a 3×3 linear algebraic system for the unknowns Q_1, Q_2 and Q_3. The orthogonal control functions R_i^n can then be found using equation (6.5).

With the control functions expressed in equation (6.4), boundary stretching can be conserved in the interior and orthogonal grids at boundaries can be obtained. These two types of control functions can be used separately or in combination.

6.4 Multigrid Solver

The numerical solution of equation (6.2) can be obtained by iterative methods, such as Gauss-Seidel iteration. Although iterative solution procedures have been widely used, they become computationally expensive for finer grids and for higher accuracy. Thus, there is a need to devise faster solution procedures. The multigrid method is an ideal solver for the grid generation system, due to the fully elliptic nature of these equations. In order to obtain an efficient grid generator, a multigrid method is developed to solve these equations. The Full Approximation Scheme (FAS) of Brandt (1980) in conjunction with Full Multigrid Cycling (FMG) is adopted. The basic idea of the FMG scheme is to provide a good initial solution for the finer grids by prolongating from the solutions on coarse grids. This avoids the need for an algebraic grid generation procedure for initial solutions, which is often necessary for single-grid methods in order to avoid singularities.

A multigrid method includes four major components: discretization, smoothing (solution solver), restriction, and prolongation, which are summarized in the following paragraphs.

Discretization of equations (6.2) is obtained using the second-order, central difference scheme for all derivatives involved. The resulting discrete equations are solved on a cuboid domain with a Dirichlet boundary condition. The discrete governing equations

are then written as seven-point formulas as follows,

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + a_T\phi_T + a_B\phi_B + C \tag{6.9}$$

where the variable ϕ expresses the three unknown x_k (k=1,2,3) and the subscripts W, E, etc. indicate the six neighbouring positions corresponding to node P.

Since the coefficients a_{ij} in equation (6.2) are functions of the coordinate derivatives, the singular case, i.e. J=0, and the anisotropic case where the coefficients a_{11},a_{22} and a_{33} have different orders of magnitude, e.g., $a_{11} \ll a_{22}$, may occur. The point Gauss-Seidel type relaxation is not efficient for the anisotropic case, such as $a_{11} \ll a_{22}$, since it is only smoothed in the ξ_2 direction and not in the ξ_1 direction. This can be remedied partially by simultaneously solving for those unknowns along a line parallel to the ξ_2 direction. To prevent all the possible anisotropic cases, the alternating line-coupled Gauss-Seidel iterative procedure is used as the smoother.

Singular cases may also happen during iteration (for example, zero initial guess solution will result in $a_{11} = a_{22} = a_{33} = 0$). For such cases $a_P = 0$, and the Gauss-Seidel type relaxation fails as a smoother. To avoid this case, an artificial damping quantity is introduced. That is, a positive quantity $\beta\phi_P$ is added to both sides of equation (6.9), where β is a positive constant and is chosen depending on the problem to be solved. Therefore, the source term C is augmented by an amount of $\beta\phi_P$ while the coefficient a_P of the main diagonal term is increased to $a_P + \beta$.

Restriction is a process by which contents of a fine grid are transferred to a coarser grid after a few sweeps of relaxation. Two quantities, namely, the approximate solution (x_1, x_2, x_3) and the residual are transferred to the coarse grid by two different types of operators. Pure injection is used for the restriction of the solution. In order to preserve the high frequency contents of the fine grid residuals, full weighting is applied in the transfer operator for residue restriction.

Prolongation is a process by which the corrections and solutions (x_1, x_2, x_3) in the coarse grid are interpolated onto the fine grid. Its order depends on the order of the differencing scheme and for a second order difference scheme, it is sufficient to apply linear interpolation for prolongation to avoid large magnification of the high frequency components. In the program, bilinear interpolation is used.

6.5 Multi-Block Grid Generation

Although in principle it is possible to establish a correspondence between any physical region and a single rectangular type domain by solving the elliptic grid generation equations, the resulting grid is likely to be too skewed and irregular to be usable for certain complex 3D configurations, such as that for the film cooling of a turbine blade. For complex configurations it becomes necessary to partition the physical domain into a number of subregions. Grids are generated in different subregions.

In the MBEGG code, a multi-block grid generation method is used to generate grids for complex geometries. In this method, the entire complex physical domain is segmented into different sub-domains called grid blocks. Grids are generated in each block and patched together to produce a multi-block grid.

The topology to define the block structure in a multi-block method can be complicated for complex configurations. In the MBEGG code, the multiblock topology is defined in the following manner. First, a data file is created to provide the basic geometric information, such as the number of blocks, the neighbouring blocks, the grid dimensions of each block, and block types. Secondly, a Cartesian coordinate system is chosen referring to the given configuration to define the coordinates of each point in the domain.

A subroutine is used to define the block structure from the information provided by the user. Each block is defined by six boundary surfaces. Each surface is defined by

four edges. The shape of each edge is specified by the user as either a line or a curve segment. For line segments, only the coordinates of the two end points are required. For a curved segment, a function or certain constraints have to be given. A rule for the grid distribution in each edge is specified by the user, such as uniform distribution or nonuniform distribution with a given cell-dimension ratio. From this information, the grid nodes in an edge are calculated. After all the edges have been defined, the grid for each boundary surface can be generated. There are two types of boundary surfaces; one is a plane surface and the other is a curved surface. For a plane surface, the two-dimensional, elliptic, grid generation equations are solved to generate the smoothest grid on the surface. The grid generation for a curved surface is somewhat more involved. In the present study, an indirect technique is used. First, the curved surface is mapped into a 2D plane area according to certain rules. A grid is then generated for the plane area by solving the 2D, elliptic, grid generation equations. The grid for the curved surface is obtained by mapping the 2D plane grid back onto the curved surface.

With the specified grid coordinates on the six surfaces of each block, the inner grid in each block can be obtained by solving equation (6.2) while the grid coordinates of the six surfaces are used as boundary conditions. For multi-block grids, the blocks share common boundary surfaces called interfaces. The grids near the interfaces have to be controlled in order to establish good relations between adjacent blocks. The grid lines in adjacent blocks might be made to meet at the interface with complete continuity, with continuous grid line but with discontinuous slope, or perhaps not to meet at all. Matched grid lines across the interface can be achieved by simply specifying the same grid coordinates for two adjacent blocks at the interface. This type of grid is adopted in all our calculations. For a multi-block grid, smooth grid lines across the interfaces are preferred. This can be achieved by specifying both the same grid coordinates and same grid line slopes at the interface. Unfortunately, the elliptic grid generation system only permits one type

of boundary condition to be specified in order to have a solution. In the present study, generating grids with smooth grid lines across the interfaces was not sought. Instead, efforts are made in the CMGFD code to allow for the use of discontinuous grid line slopes across the interface.

6.6 The Code Structure

This code includes a 2D, elliptic, grid generation package and a 3D, elliptic, grid generation package which can be used to generate grids in arbitrary 2D or 3D single-block domains. The 2D grid generation code solves a two-dimensional, elliptic, grid generation system and can be used to generate grids on the boundary surfaces of each block which serve as boundary conditions for three-dimensional grid generation. The 3D grid generation code solves the three-dimensional, elliptic, grid generation system and can be used to generate grids in hexahedral domains of arbitrary shape. The automatic choice of control functions is implemented in both packages. Designed grids can be obtained by specifying the control functions. Both the 2D and 3D code packages are written using a full multigrid and full approximation scheme. The code package described in chapter 5 for the multigrid method is modified for the present use. The MBEGG code is constructed by using the two single-block generation codes and some subroutines for domain decomposition and linkage.

6.7 Application of the Code

The code has been applied to generate grids in various geometries, such as curved pipes and turbine blades, to support the application of the CMGFD code. The generation of those grids, such as the multi-block grids over a turbine blade, have been shown in the chapters where the problems are solved. In this section, only the generation of a grid in

a 2D domain is presented to demonstrate the performance of the multigrid method.

Figure 6.2 shows the domain to be considered. This domain is related to the configuration of a turbine blade to be studied in the next chapter. Here, the circular edge represents the blade surface in the cross-blade section. Such a domain will be used as a boundary surface in a multi-block grid over the turbine blade.

Computation is carried out using a 6-level multigrid with the coarsest grid containing only 2×2 cells. Convergence was considered to be achieved when the maximum residues for all the three governing equations were less than a prescribed tolerance, which for these tests was set at 5×10^{-5}. The converged solution is plotted for a finer grid (32×32) in Figure 6.3.

It was found that the full multigrid method which first started on the coarsest grid had significant contributions to numerical stability for the grid generation. Because of the strong nonlinearity of the grid generation equation, an inadequate initial guess will cause numerical instability. Computations using the single-grid method showed that it was difficult to converge for fairly dense grids. An algebraic grid generation procedure was used in the single-grid method to achieve a good initial solution. With heavy relaxation a solution was obtained for a grid of 32×32 cells. However, convergence could not be obtained for grid dimensions beyond 32×32. The convergence behavior for the multigrid and the single-grid calculations are compared in Figure 6.4. It can be seen that the multigrid method produces a much faster convergence rate.

6.8 Closure

A multigrid and multi-block grid generation code was developed. An elliptic grid generation method is used, which generates grids by solving three, elliptic, differential equations. Methods to achieve certain grid quantities through the control functions are implemented

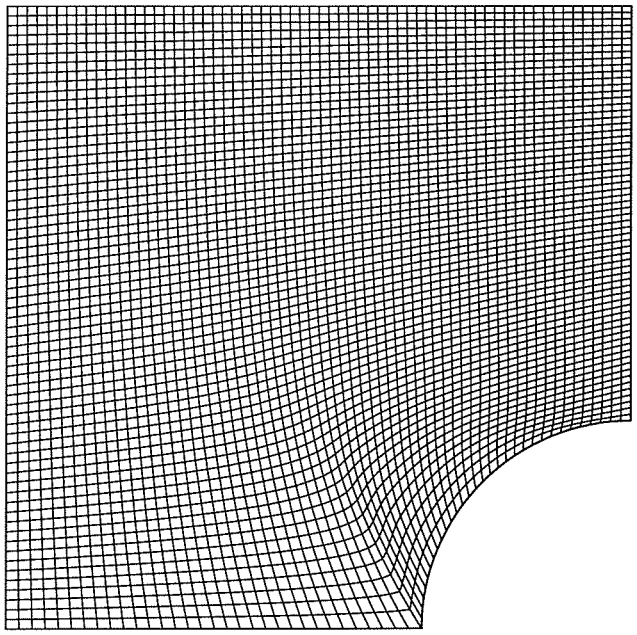

Figure 6.2: The mesh generated for a finer grid with 32×32 cells.

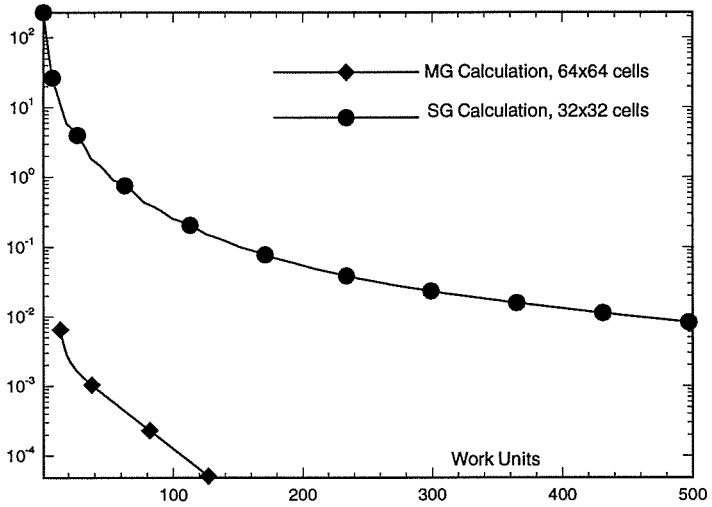

Figure 6.3: Convergence behavior, (a) multigrid calculation with the finest grid of 64×64. (b) single-grid calculation with 32×32 grid.

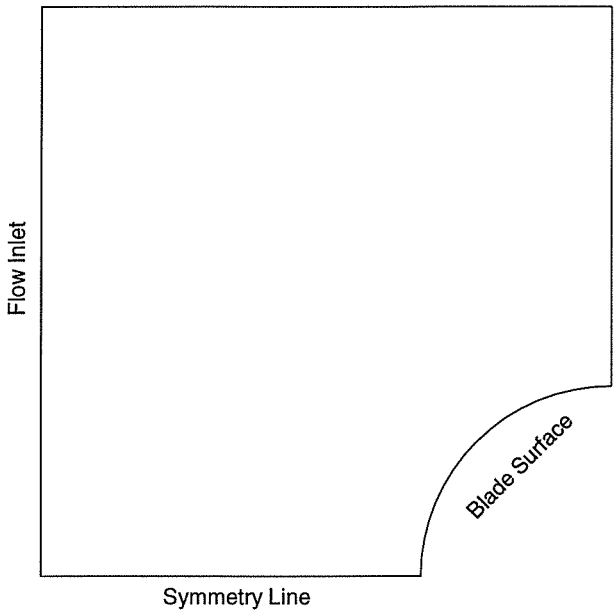

Figure 6.4: A two-dimensional domain to be considered.

in the code. A multigrid method was developed to solve these equations. The developed multigrid algorithm dramatically reduces the computational effort required to solve the system of elliptic grid generation equations. Fast convergence has been achieved with a zero initial guess. A domain segmentation technique was developed to generate block-structured grids. The code can be used to generate grids in arbitrary 2D or 3D domains. The combination of the MBEGG and CMGFD codes allows for flow simulations in arbitrary geometries.

Chapter 7

Prediction of Film Cooling Over A Turbine Blade

Extensive experimental studies of leading edge film cooling have been carried out at UBC by Gartshore *et al* (1993) and Salcudean *et al* (1993,1994a) based on a large turbine blade model. In this chapter, computations are carried out based on such a blade model and are compared with the experimental data. This model has a semi-circular leading edge with four rows of film cooling orifices positioned symmetrically about the stagnation line. Lateral injection is used and the cooling orifices are inclined at 30^0 to the blade surface in the spanwise direction.

The computational methods and codes described in previous chapters are used to analyze the flow and related heat transfer in this complex geometry. The computational domain follows the blade geometry and includes not only the curved blade surface but also the inclined circular coolant orifices. The computational domain is segmented into a number of sub-domains and separate curvilinear grids are employed for different flow regions. Grids are generated using the methods and codes described in the previous chapter. A block-structured, non-orthogonal grid is used to exactly represent the curved blade surface as well as the circular injection orifices. Computations over the cooled turbine blade model are carried out for overall mass flow ratios of 0.52 and 0.97. The relative mass flow ratios from each orifice are specified to match experimental values. Density ratios of coolant to free stream were taken to be unity (constant density). The predicted results are compared with the experimental results of Salcudean *et al* (1994).

7.1 Problem Description

The film cooling model under consideration is that of an experimental turbine blade model at UBC, as shown in Figure 7.1. This experiment was designed to study the film cooling effectiveness near the leading edge of turbine blades. The model has a semi-circular leading edge diameter of 127 mm, spanwise top and bottom lengths of 1.2 m (between fences), and a chordwise length of 2.3 m. Coolant was injected through four rows of circular orifices which are inclined at 30^0 in the spanwise direction and perpendicular to the blade surface in the streamwise direction on the cylindrical leading edge. These four rows of orifices are located symmetrically at $\pm 15^0$ and $\pm 44^0$ with respect to the stagnation line. Each injection tube has a diameter d of 12.7 mm and each row has orifices whose spanwise spacing was $4d$.

Exploiting the advantage of the flow symmetry about the stagnation line and periodicity in the spanwise direction, only half of the physical region with one period in the spanwise direction was taken as the computational domain. Figure 7.2 shows the domain used in the present study, where d is the diameter of the injection orifice. The computational domain includes a main flow region, two half injection orifices in the first row, and one injection orifice in the second row. The main flow region extends $10d$ upstream from the leading edge, 40d downstream (in the positive x-direction) from the end of the semi-cylinder, and 30d in the z-axis vertical direction from the leading edge. Since the circular injection tubes are inclined at 30^0 to the spanwise direction, the orifices become ellipses with a semi-major axis of $2d$ in the y-direction, as shown in Figure 7.2. The length of the injection hole is 4d.

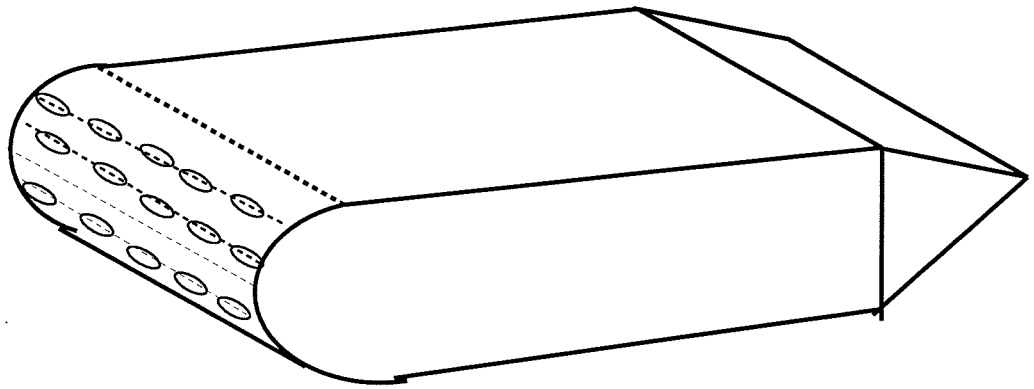

Figure 7.1: Illustration of the turbine blade model used in the experimental study of Salcudean *et al* (1994).

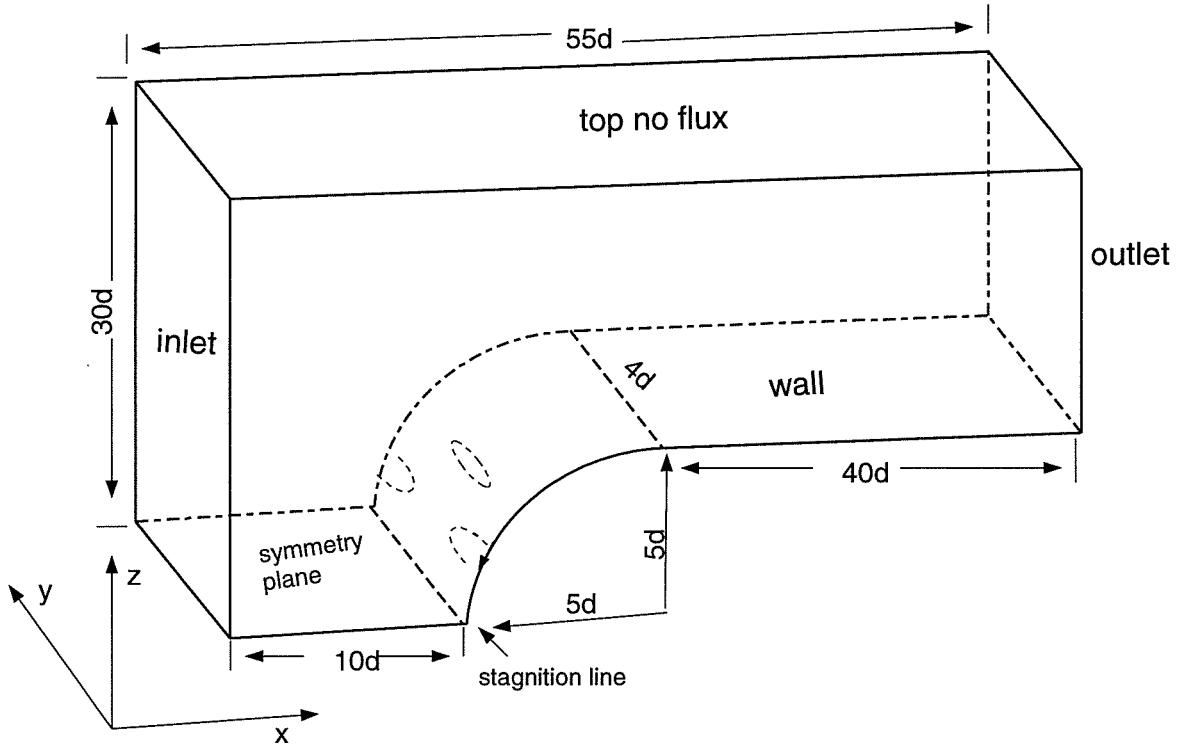

Figure 7.2: Illustration of the computational domain, including a main flow region, two half injection-hole ducts in the first row and one injection hole duct in the second row.

7.2 Computational Grid

The grid was generated using the MBEGG code described earlier. The computational domain is segmented into four subdomains, a "hot" flow region over the blade, a coolant hole in the second row and two half coolant holes in the first row. In order to obtain a continuous grid at the interface, a rectangular type grid is used for the injection holes instead of cylindrical grids. The generated grid is shown in Figure 7.3. The fine grid used for the present study contains $101 \times 17 \times 37$ nodes in the main flow region and $9 \times 9 \times 19$ nodes for each injection orifice duct. Figure 7.4 shows the grid on the blade surface and in the injection orifice regions. The grids in the orifice ducts match with the grids in the main flow region at the interface.

7.3 Solution Model

Since the comparable film cooling experiments were done in a low, speed incompressible flow with isothermal conditions and a heat/mass transfer analogy, the computations to compare with the experiments are made with a steady, incompressible, constant density code. The 3D, incompressible, Reynolds-averaged, Navier-Stokes equations together with the energy equation are solved. Turbulence closure is attained by the use of the standard $k-\epsilon$ model with the 'wall function' treatment described by Launder and Spalding (1974). The shortcomings of the $k-\epsilon$ model are well known for the prediction of film cooling problems. These are mainly due to the anisotropic and non-equilibrium turbulence caused by flow curvature, film jet spreading, flow complexity, and unsteadiness near the injection orifice exits, as well as the heat transfer to the wall in the near-wall, viscosity-affected sublayer. However, the present calculations are for an adiabatic wall and, therefore, do not involve heat transfer to the wall. It is useful to determine how well the standard $k-\epsilon$ model can predict the film cooling over an actual geometry with a curved blade surface

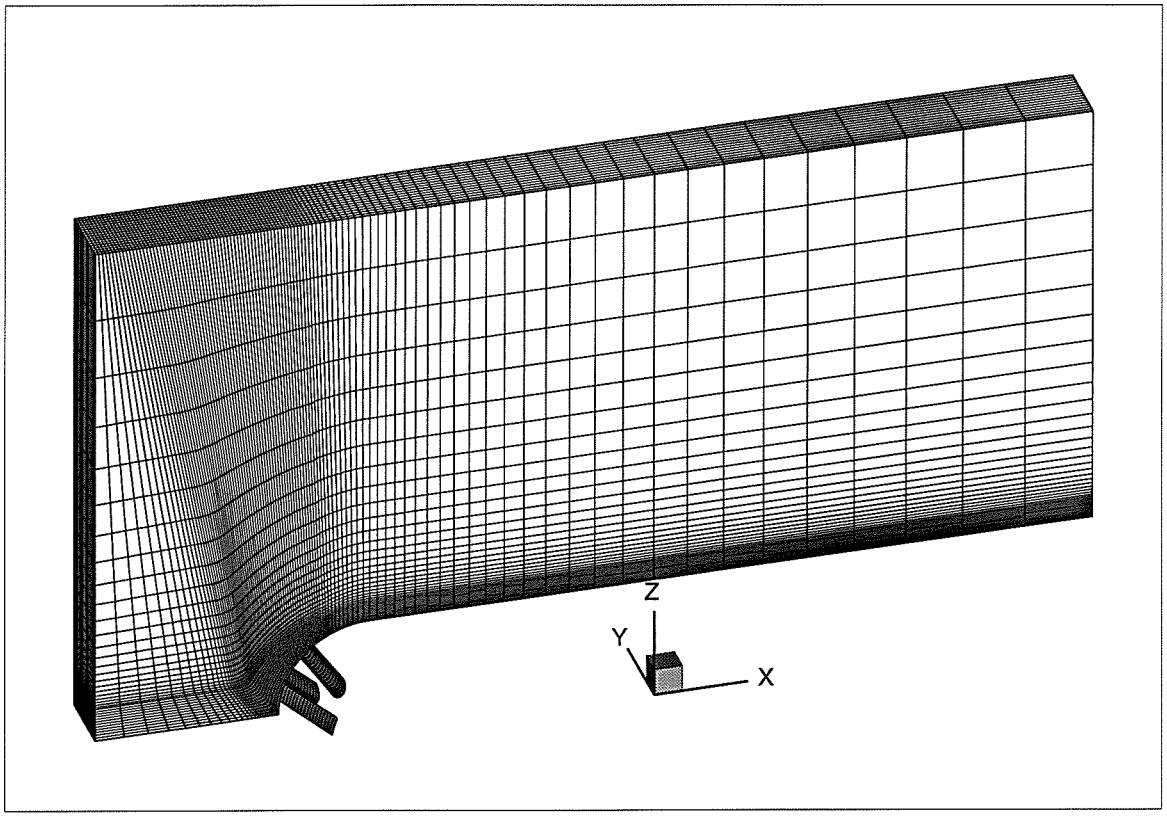

Figure 7.3: The grid generated for the present calculation containing $101 \times 17 \times 37$ grid nodes in the main flow region and $9 \times 9 \times 17$ grid nodes in one injection hole.

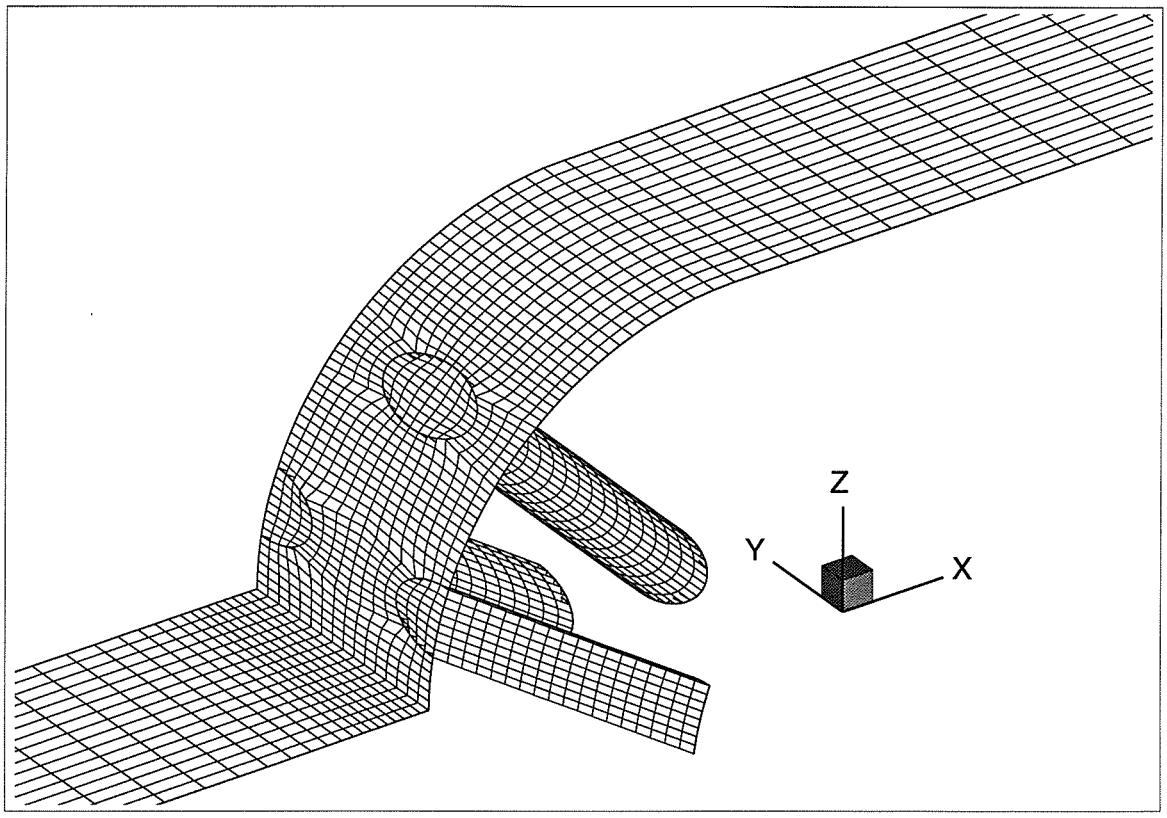

Figure 7.4: Computational grid at the curved blade surface and the injection hole ducts.

and the injection orifice ducts.

In addition to the Reynolds-averaged Navier-Stokes equations and the $k-\epsilon$ equations described in chapter 3, the following thermal energy equation is solved:

$$\bigtriangledown \cdot(\varrho \mathbf{u} T - (\frac{\mu_l}{Pr} + \frac{\mu_t}{Pr_t}) \bigtriangledown T) = 0, \tag{7.1}$$

where T is temperature, Pr and Pr_t are the Prandtl number and turbulent Prandtl number respectively.

7.4 Boundary Conditions

Five types of boundary conditions; inlet, outlet, wall, no-flux, and periodic, are used. The treatment of boundary conditions on each side can be described as follows:

- Mainstream:

  The inlet plane of the mainstream (the west plane of the main flow region) is located $10d$ upstream from the leading edge where all the dependent variables except pressure are prescribed to match the experimental values. The mean velocity is taken as $U_\infty = 10\ m/s$ following the experiments. The turbulence energy and turbulence dissipation were evaluated using the following formulas:

  $$k = 1.5U^2(u/U)^2, \qquad \epsilon = C_\mu^{\frac{3}{4}} k^{\frac{3}{2}}/l \tag{7.2}$$

  where u/U is the turbulence intensity and l is the turbulence length scale. The turbulence intensity is assigned a value of $u/U = 0.5\%$ and the turbulence dissipation is calculated based on a length scale equal to the height of the wind tunnel $l = 1.6\ m$. Since the turbulence intensity is very small, these assumptions should not affect the results significantly. The temperature is assigned a non-dimensional value of 1.

- Coolant Inlets:

  The velocities at coolant duct inlets are specified from the mass flow ratios and discharge rates measured in the experimental study of Salcudean *et al* (1994a). The k and ϵ values at the duct inlets are evaluated from equation (7.2). The turbulence intensity u/U is specified as 5% and the lenth scale l is taken to be the diameter d of the injection orifice. The temperature is assigned a non-dimensional value of 0.

- Adiabatic Wall:

  The wall at the turbine blade surface (from the leading edge line to the outlet) is assumed to be adiabatic and impermeable with zero tangential velocity. The standard 'wall function' described by Launder and Spalding (1974) is used. The front section before the stagnation line at the bottom of the main flow region is treated as a zero flux and free-slip condition due to the flow symmetry about the stagnation line.

- Periodic Condition:

  The periodic (cyclic) condition is used on the boundaries in the spanwise direction (i.e., the south and north planes). Such a boundary condition assumes that there are an infinite number of orifices in the spanwise direction.

- No-Flux Condition:

  The top boundary is located at a distance $30d$ in the z- direction from the symmetry planes, where the impermeable, free-slip condition is imposed.

- Outlet Condition:

  The outflow boundary is located at a distance $40d$ downstream from the semi-cylindrical end. The zero-gradient condition in the streamwise direction is imposed for all dependent variables at the outlet boundary. It was sufficiently far downstream to ensure that the flow in the upstream region was not affected by downstream conditions.

7.5 Computational Procedure

The CMGFD code described in Chapter 5 was used to carry out computations of flow and film cooling effectiveness for mass flow ratios of $M_\infty = 0.52$ and $M_\infty = 0.97$, where M_∞ is the overall mass flow ratio averaged from all injection orifices which is defined as $M_\infty = \dfrac{U_c}{U_\infty}$. Here U_infty is the free-stream velocity and U_c is the average coolant injection velocity. Due to the strong pressure gradient near the leading edge, the mass flow from the first row (located at $\pm 15^0$ of the semi-circle from the stagnation line) is less than that in the second row (located at $\pm 44^0$). Following the notation of Salcudean *et al* (1994a), the mass flow ratio for the first and the second rows are denoted as M_{15} and M_{44}, respectively. The mass flow ratios M_{15} and M_{44} were obtained by Salcudean *et al* (1994a) by measuring the discharge flow rate and are given by the following relations.

(1) if $M_\infty = 0.52$, then $M_{15} = 0.3$ and $M_{44} = 0.75$.

(2) if $M_\infty = 0.97$, then $M_{15} = 0.86$ and $M_{44} = 1.08$.

These values were used to calculate the turbulence kinetic energy and dissipation rate at the coolant duct inlets. For both mass flow ratios, the coolant duct Reynolds number Re is equal to 4200 as in the experimental study, where $Re = \dfrac{\rho U_c d}{\mu}$ is based on the overall

average coolant velocity U_c, injection orifice diameter d, and the viscosity and density of the air at STP.

A two-level multigrid was used for the computations of the mass flow ratio $M_\infty = 0.52$. No special effort was made for the initialization of the solution. On the coarse grid, all the dependent variables except k and ϵ were set to zero, and k and ϵ were set equal to the values in the free stream. The converged solution was prolongated to the fine grid where the FAS procedure was applied.

It was observed that heavy under-relaxation was necessary to obtain a converged solution. This is mainly due to the first row of injection holes where the jet exits are located in the high pressure region. Convergence was not obtained until α_u and α_p were reduced to 0.5 and 0.3 respectively, where α_u is the under-relaxation factor for momentum equations, and α_p is the under-relaxation factor for pressure and the $k-\epsilon$ equations. Furthermore, under-relaxation of the turbulence production rate, the turbulence viscosity, and near wall turbulence quantities was applied to facilitate convergence of the computations. For the mass flow ratio $M_\infty = 0.97$ convergence was even more difficult to obtain. To overcome this difficulty, the converged solution for $M_\infty = 0.52$ was used as the initial solution for this calculation. This approach was found to be very useful. The maximum residuals of all the discrete governing equations dropped by 4 orders of magnitude in about 400 iterations for $M_\infty = 0.52$ and 480 iterations for $M_\infty = 0.97$. Further iterations were carried out to achieve a residual drop of 8 orders of magnitude. Grid independence studies have shown that, for the number of computational nodes equal to half of that used in the final calculations, the film cooling effectiveness was lowered as much as 20% while decreasing the number of nodes by 30% lowered effectiveness by 5%. One can conclude that the results predicted here for the finest grid are approaching grid independence. Full grid independence can not be achieved with the standard wall treatment in turbulent flows.

7.6 Results and Discussion

The computational results are presented in this section. It should be mentioned that the solutions are not presented in the complete computational domain in most of the figures for reasons of clarity. The flow region near the curved blade surface and the injection orifice exits contains the most rapid changes of dependent variables, while the region far from the curved blade surface tends to be more uniform. Therefore, most of the figures show the region near the blade surface and injection orifice exits.

Figure 7.5 presents the calculated solutions in the xz-plane and in a plane through the centerline of the injection orifice (that is, the north or south plane) in the first row for $M_\infty = 0.52$. It should be noted that the orifice length projected on the xz-plane is only half of the length of the orifice as viewed in the figures due to the inclination angle in the spanwise direction. Here, Figure 7.5a shows a local velocity field around the curved blade surface. Figure 7.5b shows the pressure distribution in the xz-plane for the entire computational domain. The non-dimensional pressure is expressed as $\frac{p}{p_{ref}}$, where p_{ref} is the pressure at the center of the stagnation line. Figures 7.5c & 7.5d show close-up views of the velocity and pressure fields, respectively, near the orifice exit and curved blade surface. It can be seen that the orifice exit is located in a high pressure region. The coolant in the front region of the orifice is not blown out directly into the main flow due to the high pressure near the leading edge which causes some of the hot fluid to flow into the coolant orifice. The velocity in this region is nearly parallel to the blade surface. Figure 7.5e shows the distribution of non-dimensional temperature θ, where $\theta = \frac{T - T_c}{T_\infty - T_c}$. It can be seen that the temperature is higher than the coolant temperature in a small interior region near the coolant tube exit. Non-dimensional temperature in this case is equal to non-dimensional concentration, the number 0 being assigned to the pure coolant and the number 1 being assigned to the free stream fluid. Figure 7.5f shows

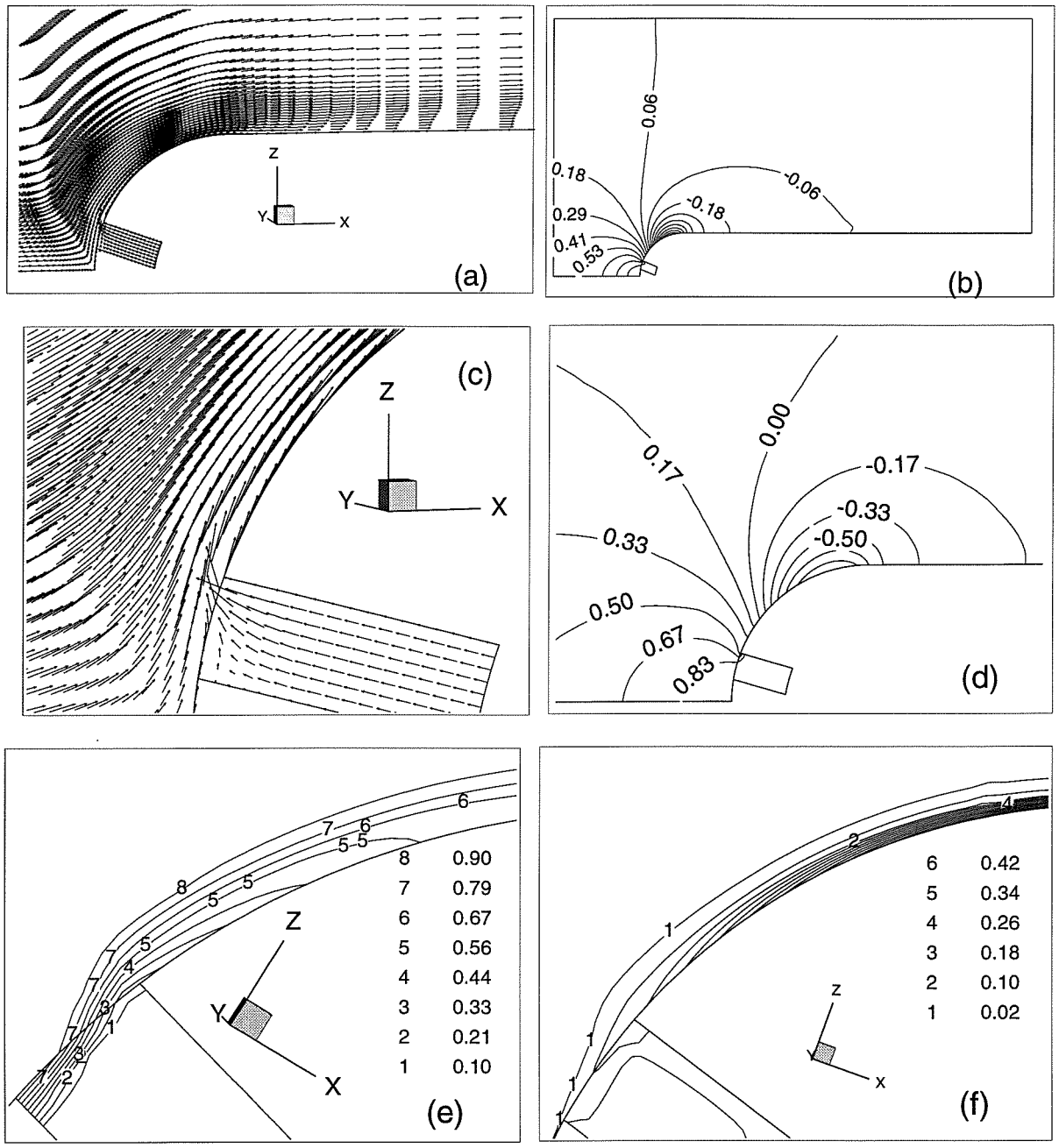

Figure 7.5: $M_\infty = 0.52$; Solutions at the streamwise plane (xz-plane) through the centerline of the 15^0 coolant orifice: (a) local velocity field around the curved blade surface, (b) pressure distribution in the whole xz-plane, (c) local velocity field (near the orifice exit) and (d) local pressure, (e) local temperature, (f) local turbulence kinetic energy.

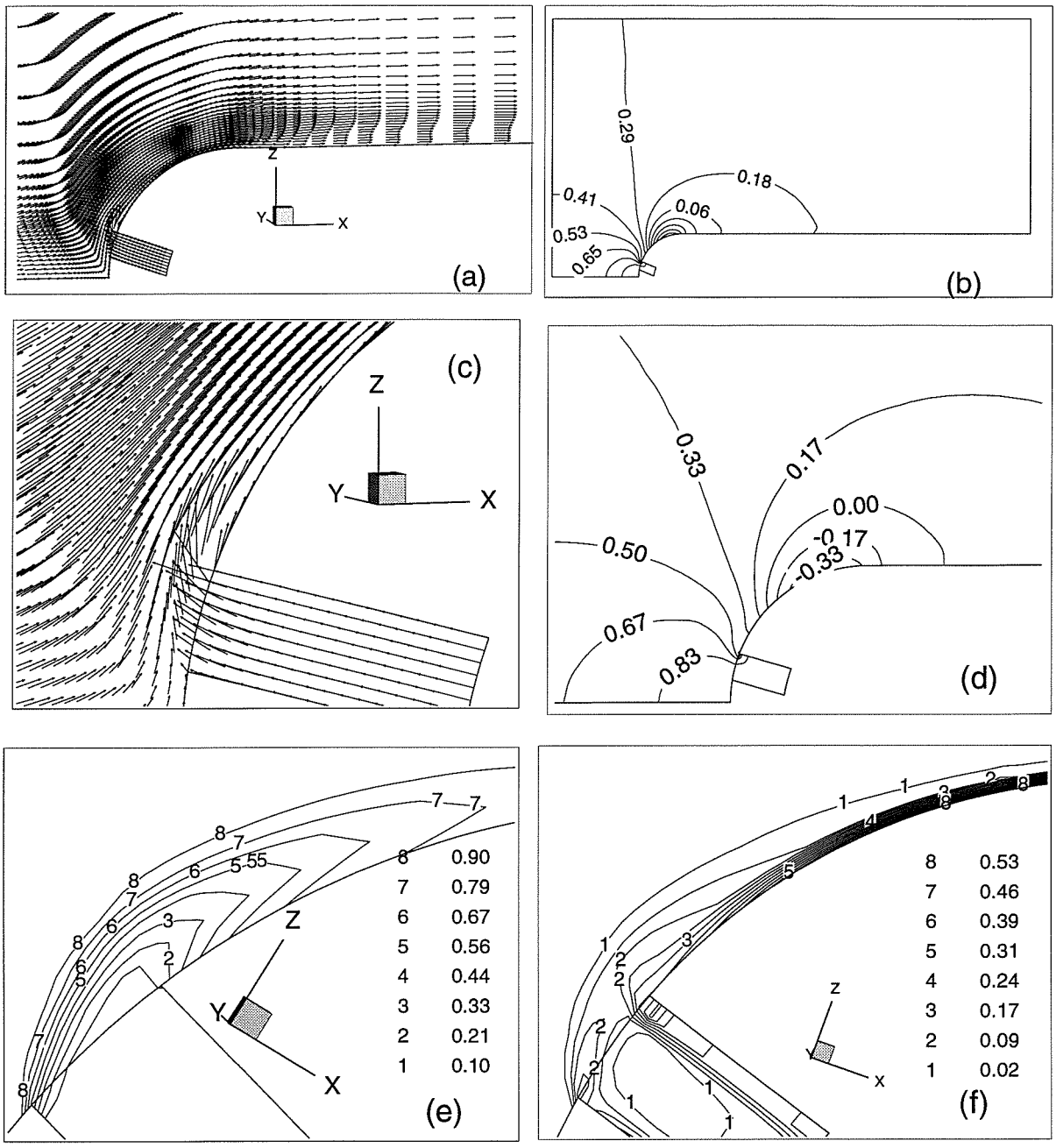

Figure 7.6: $M_\infty = 0.97$; Solutions at the streamwise plane through the centerline of the 15^0 coolant orifice: (a) local velocity field around the curved blade surface, (b) pressure distribution in the whole xz-plane, (c) local velocity field (near the orifice exit) and (d) local pressure, (e) local temperature, (f) local turbulence kinetic energy.

the distribution of the turbulence kinetic energy normalized as $\frac{10k}{U_\infty^2}$. It can be seen that the turbulence is mainly produced in the region downstream of the orifice and is close to the blade wall.

Figure 7.6 presents the results for $M_\infty = 0.97$ in the same manner. Unlike the previous result for $M_\infty = 0.52$, the coolant in the upstream region of the orifice is blown directly into the hot flow field, although the velocity profile at the exit is still strongly non-uniform. From Figure 7.6e it can be seen that there is little dilution of the coolant flow in the coolant duct. From both the distribution of non-dimensional temperature and turbulence kinetic energy, it can be seen that the coolant penetrates farther into the hot air for $M_\infty = 0.97$ than for $M_\infty = 0.52$. This is one of the reasons why a higher mass flow ratio may not produce higher film cooling effectiveness. Figures 7.7 and 7.8 present similar results in the xz-plane through the coolant tube of the second row for $M_\infty = 0.52$ and $M_\infty = 0.97$ respectively. Unlike the first row, the second row injection is located in a lower pressure region. The lowest pressure occurs just downstream of the injection orifice.

The turbulence kinetic energy is larger at $M_\infty = 0.97$ than that at $M_\infty = 0.52$. It is also larger for the second row than for the first row. It should be noted that the turbulence kinetic energy at the entrance of the coolant orifice which is specified as the inlet boundary condition is not convected to the interior of the orifice. This is because the source terms are dominant in the $k-\epsilon$ equations and the convection terms are small. The inlet k and ϵ values have little influence on the interior turbulence and the turbulence is mainly produced by the generation term (contained in the source terms of $k-\epsilon$ equations) and 'wall function'. Therefore, the use of estimated, uniform $k-\epsilon$ values at the inlet boundaries is likely adequate.

Comparison of Figures 7.5-7.8 shows that, as expected, the coolant penetrates further

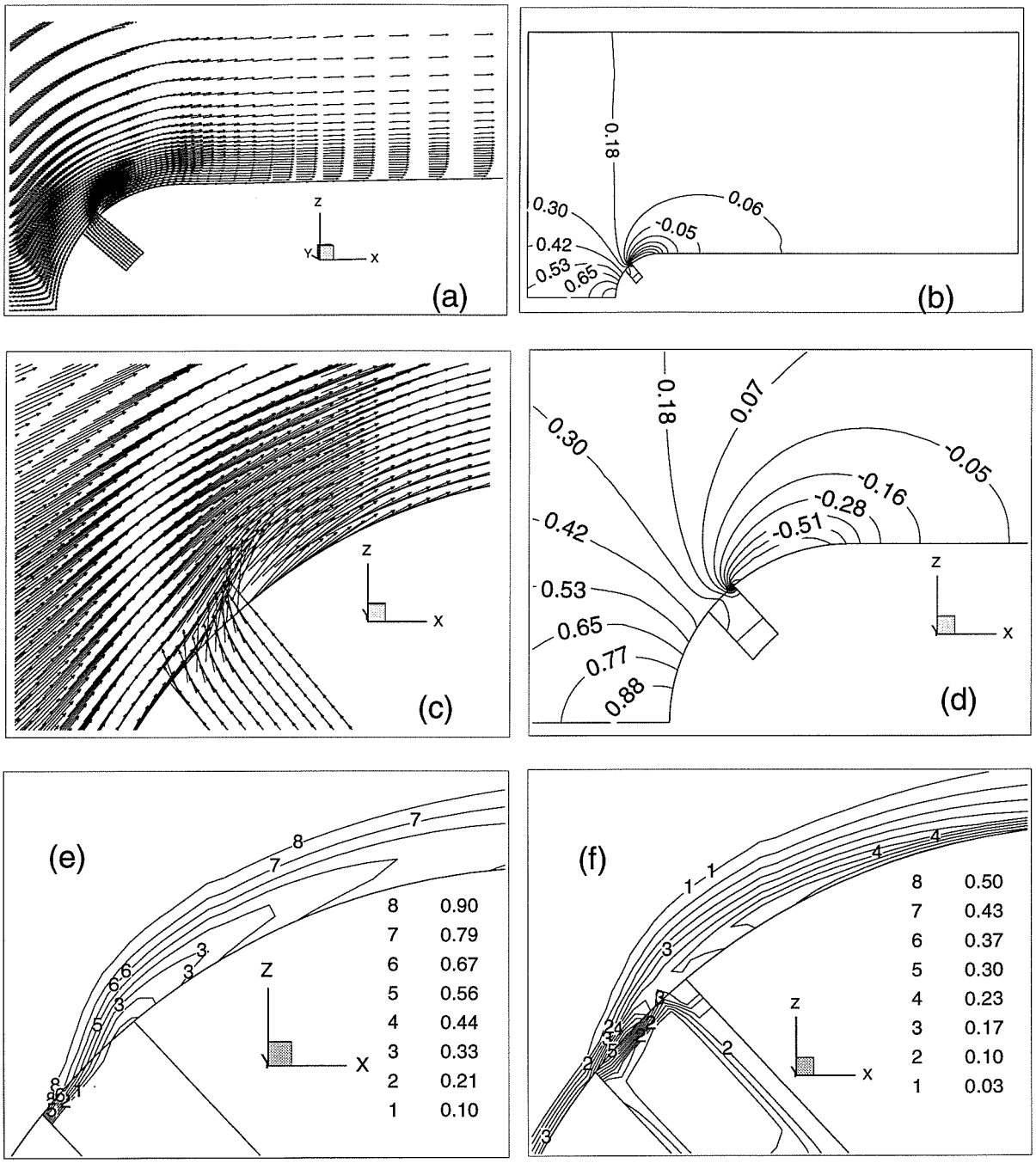

Figure 7.7: $M_\infty = 0.52$; Solutions at the streamwise plane through the centerline of the 44^0 coolant orifice: (a) local velocity field around the curved blade surface, (b) pressure distribution in the whole xz-plane, (c) local velocity field (near the orifice exit) and (d) local pressure, (e) local temperature, (f) local turbulence kinetic energy.

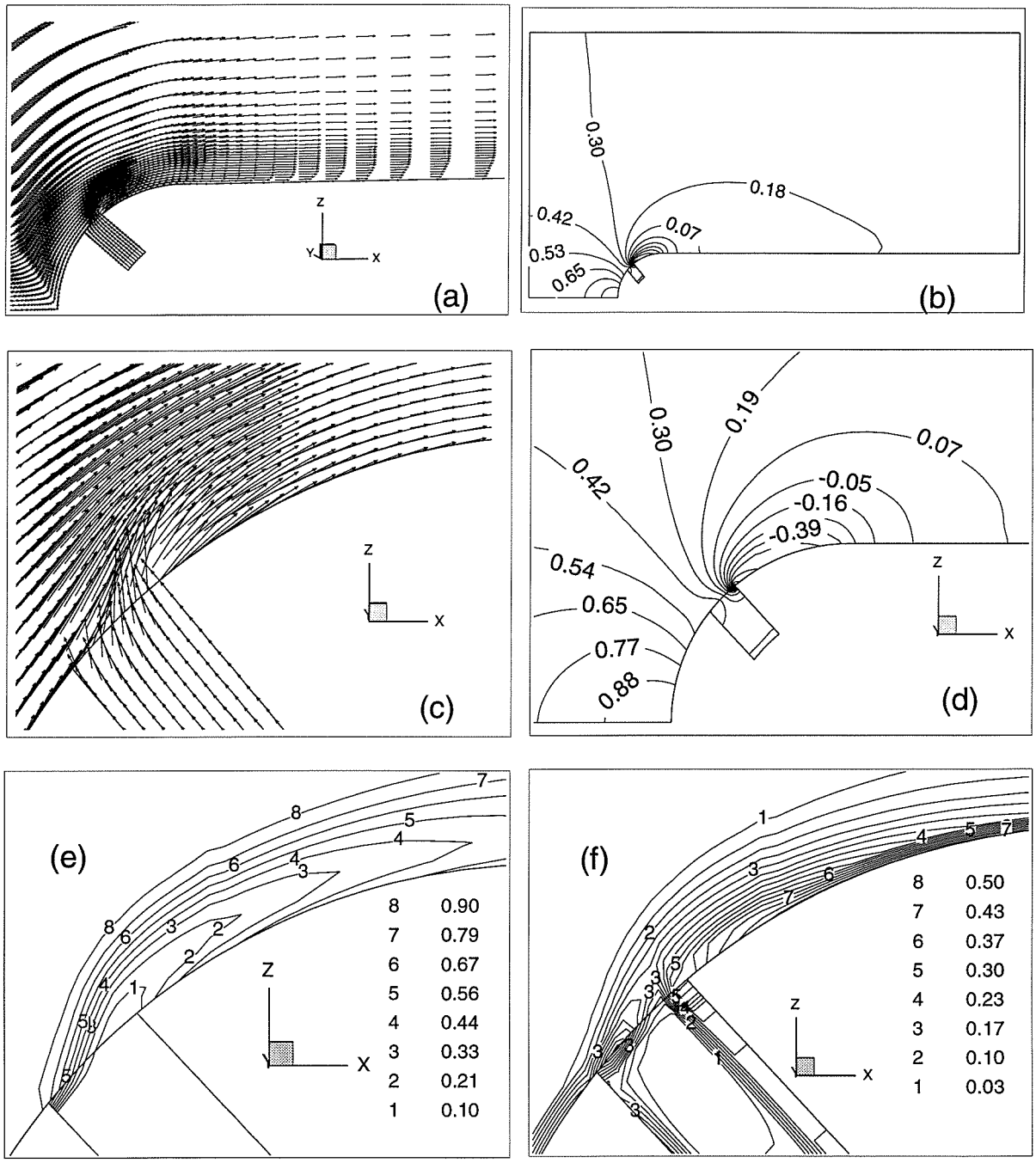

Figure 7.8: $M_\infty = 0.97$; Solutions at the streamwise plane through the centerline of the 44^0 coolant orifice: (a) local velocity field around the curved blade surface, (b) pressure distribution in the whole xz-plane, (c) local velocity field (near the orifice exit) and (d) local pressure, (e) local temperature, (f) local turbulence kinetic energy.

into the hot fluid with increasing mass flow ratio. The velocity at the orifice exit is very complex and depends on the orifice location and mass flow ratio; an assumption of a uniform velocity profile at the exit plane can be very inaccurate. The pressure distribution shows that the highest pressure occurs upstream of the first row and the lowest pressure occurs at the downstream edge of the second row. This produces a more non-uniform distribution of the flow at the coolant tube exit in the first row than in the second row. This effect is more noticeable at low mass flow ratios than at haigh mass flow ratios.

Figure 7.9 shows the development of vortices for $M_\infty = 0.52$ in planes which are perpendicular to the blade surface and which are between the first and the second rows. Here, angle "α" indicates the angular position of the plane on the semi-circular blade surface. The figure is shown with a longer domain in the y-direction for reasons of clarity by extending the computational domain using the periodic boundary condition. Figure 7.9a shows the velocity field in the plane through the centerline of the first row of orifices. The velocity is generally directed downward except in the region near the blade surface. This is not surprising since the plane through the first row forms a sharp angle (15^0) with the free stream direction. No vortices are observed in this plane. The high pressure at the exit of the orifices causes the field to flow in both directions and the effect of the lateral injection can not be observed.

Figure 7.9b shows the velocity field in the cross plane at $\alpha = 22^0$, which is approximately at $x/d = 0.6$ downstream from the centerline of the first row of orifices. Figures 7.9c and 7.9d show the velocity vectors in the cross planes at $\alpha = 28^0$ $(x/d = 1.2)$ and $\alpha = 36^0$ $(x/d = 1.8)$ respectively, which clearly show the streamwise vortices. Figure 7.10 presents velocity vectors for $M_\infty = 0.97$ in the same manner. Unlike the previous case, the lateral injection is obvious at $\alpha = 15^0$ $(x/d = 0)$ and in the positive $y - axis$ direction close to the wall. No vortices can be observed in this plane, but they can be seen along

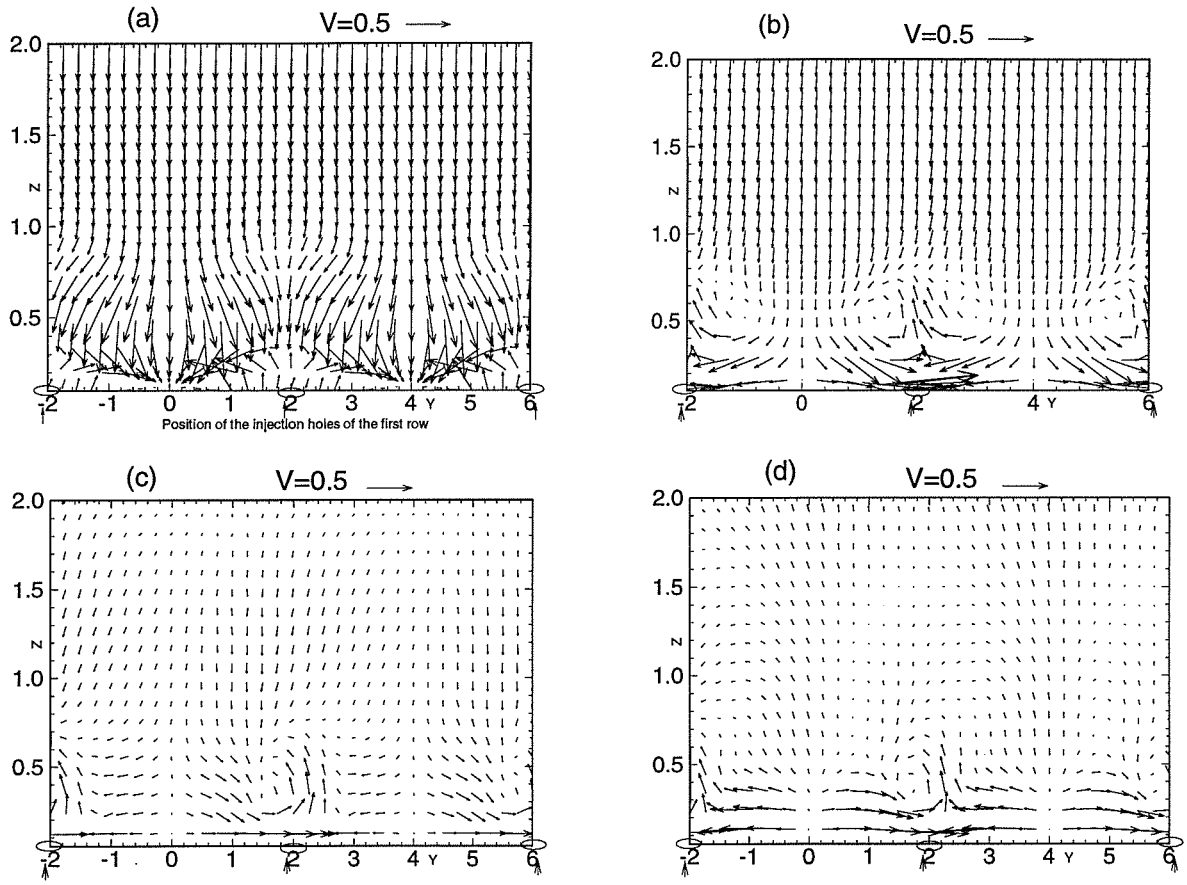

Figure 7.9: $M_\infty = 0.52$; Velocity fields at cross-stream planes which are perpendicular to the blade surface: (a) $\alpha = 15^0$, (b) $\alpha = 22^0$, (c) $\alpha = 28^0$ and (d) $\alpha = 36^0$.

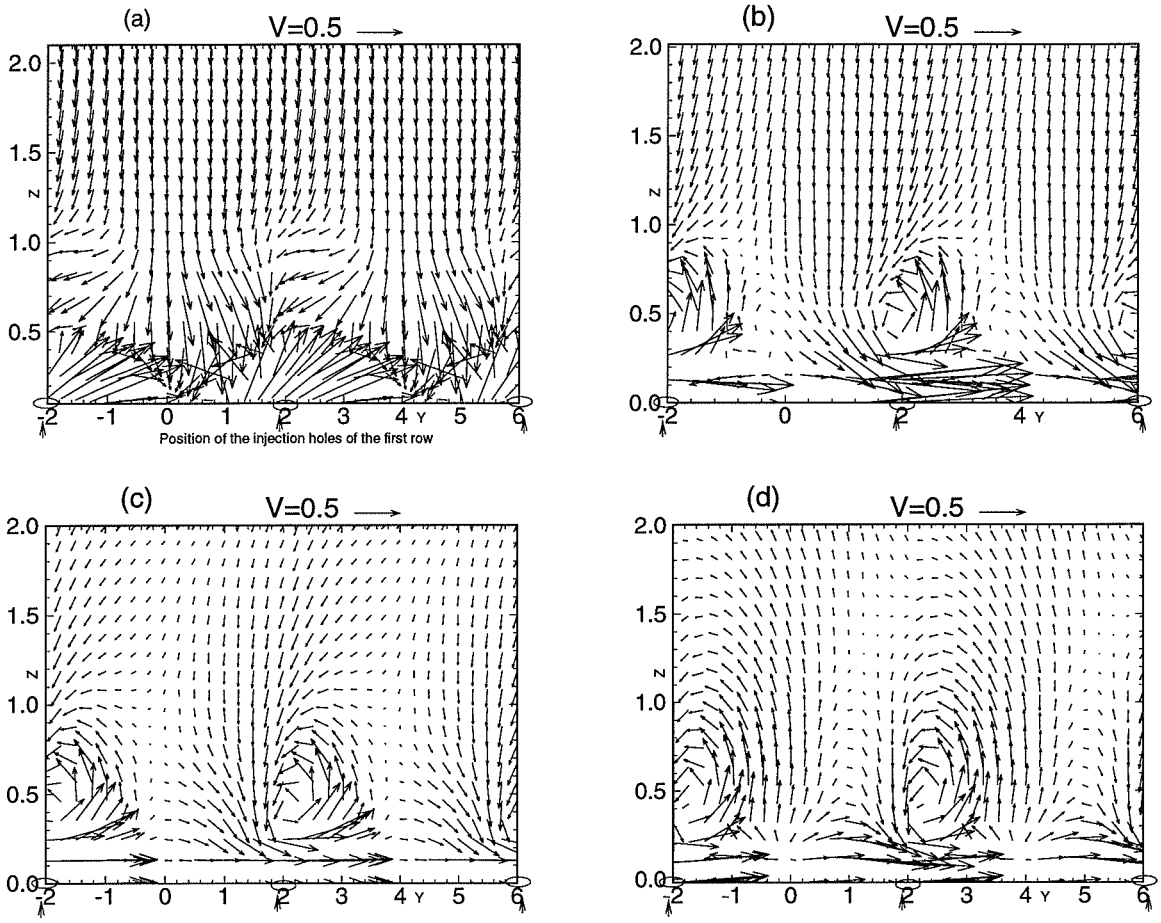

Figure 7.10: $M_\infty = 0.97$; Velocity fields at cross-stream planes which are perpendicular to the blade surface: (a) $\alpha = 15^0$, (b) $\alpha = 22^0$, (c) $\alpha = 28^0$ and (d) $\alpha = 36^0$.

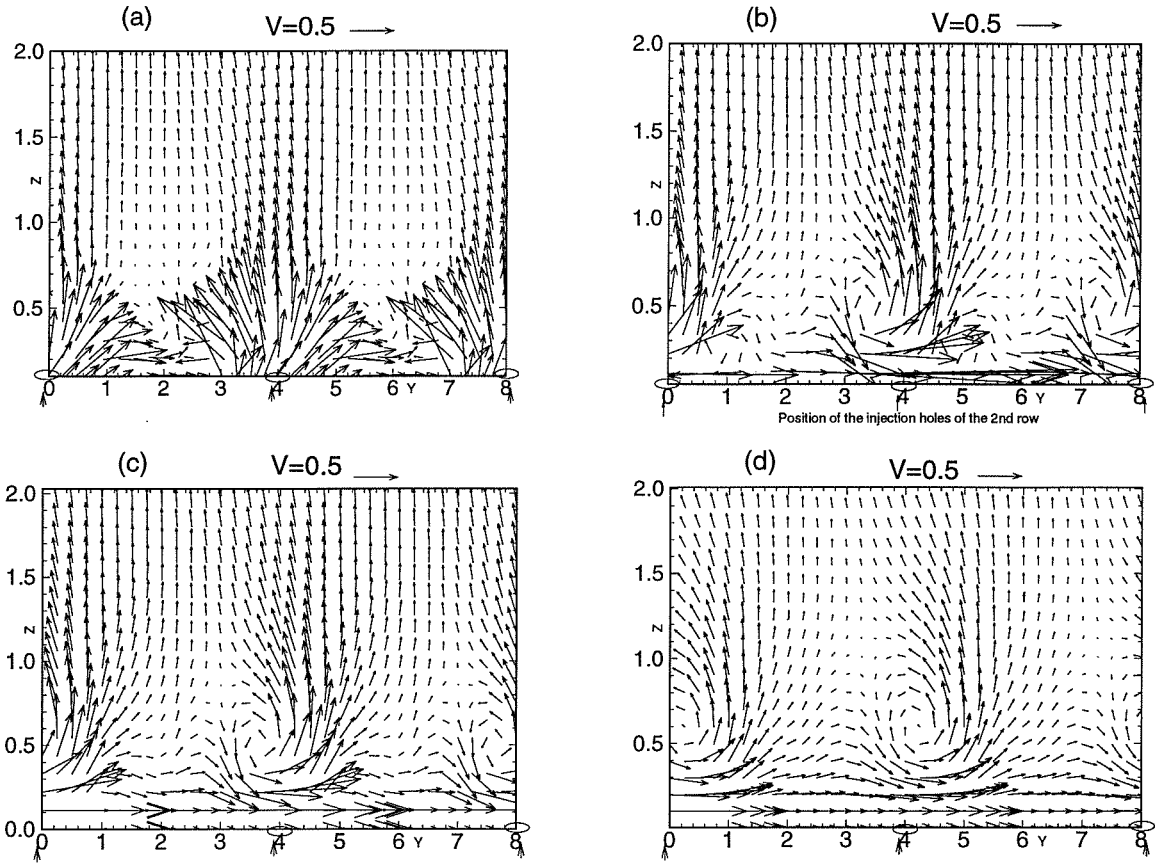

Figure 7.11: $M_\infty = 0.52$; Velocity fields at cross-stream planes which are perpendicular to the blade surface: (a) $\alpha = 44^0$, (b) $\alpha = 55^0$, (c) $\alpha = 60^0$ and (d) $\alpha = 80^0$.

the other three locations. It can be seen that the vortices formed above the orifices are displaced spanwise to the right side as the plane moves downstream.

Figures 7.11 and 7.12 show the velocity fields in cross planes downstream of the second row of coolant orifices. The flow pattern differs significantly from those presented in the previous figures. Vortices are now formed closer to the orifice centerline and are stronger. It is worthwhile mentioning that the formation of vortices is quite different from that reported by Vogel (1994) for streamwise injection. In the latter, the large

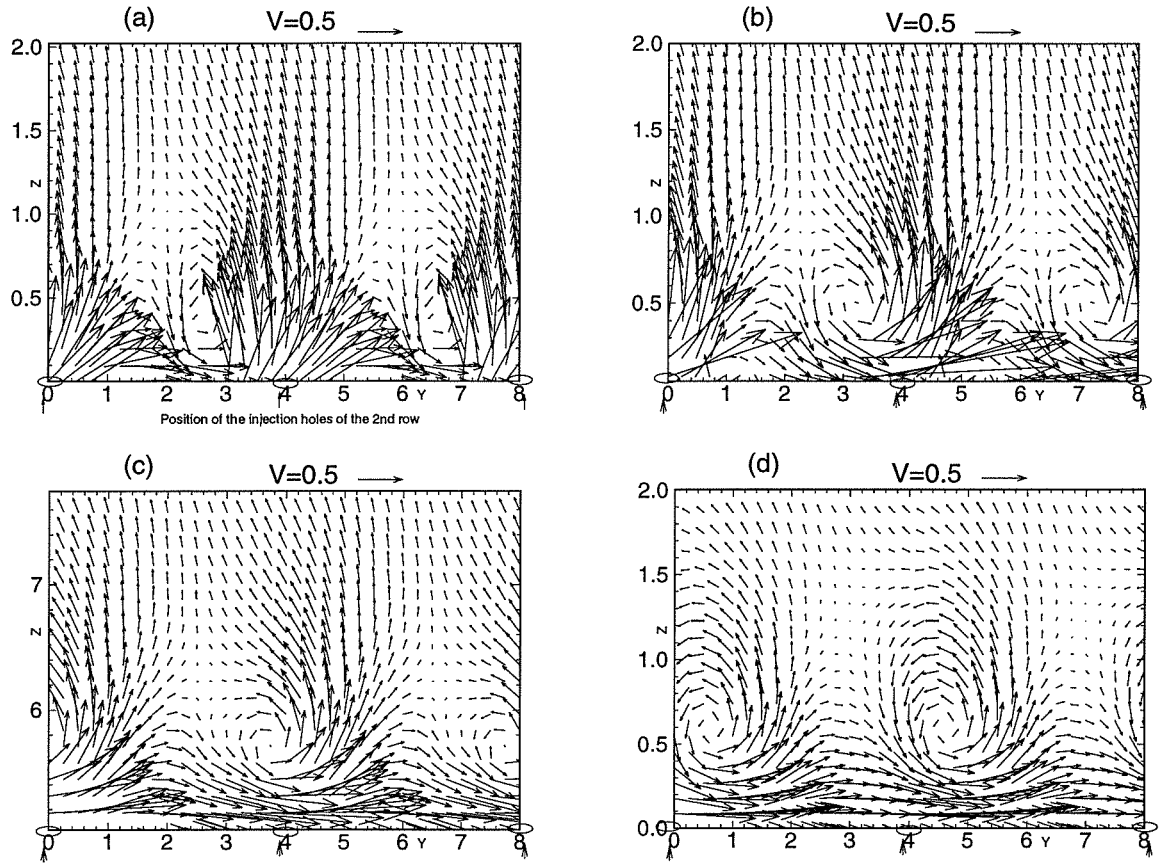

Figure 7.12: $M_\infty = 0.97$; Velocity fields at cross-stream planes which are perpendicular to the blade surface: (a) $\alpha = 44^0$, (b) $\alpha = 55^0$, (c) $\alpha = 60^0$ and (d) $\alpha = 80^0$.

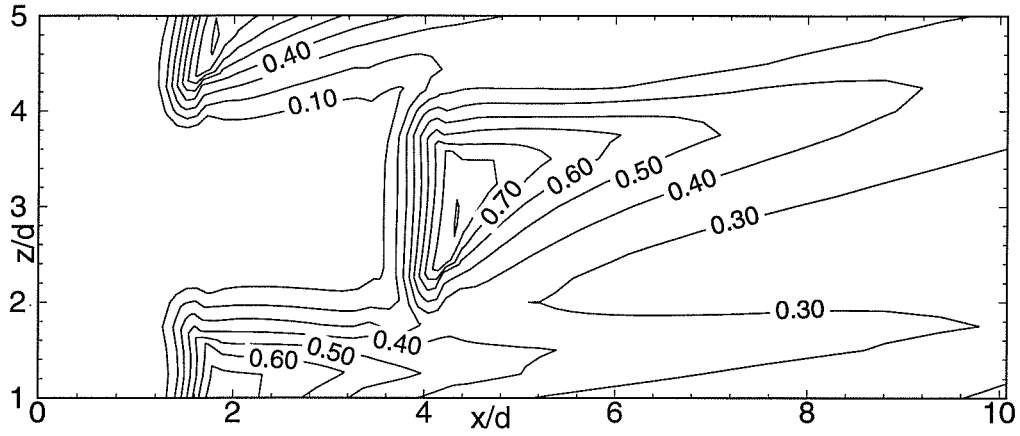

Figure 7.13: Contours of predicted FCE for $M_\infty = 0.52$ at the blade surface.

vortices tend to lift the coolant from the surface, enhance mixing and thereby decreasing cooling effectiveness. For the present, lateral injection, the vortices are smaller, form further downstream, and further from the blade surface than with streamwise injection.

The predicted film cooling effectiveness on the blade surface for $M_\infty = 0.52$ is shown in Figure 7.13. It can be seen that the coolant from the first row of orifices flows through the surface located between the cooling orifices of the second row, providing good coverage of the blade surface. This broad coverage, with flow from the first row covering the span between second row of orifices was also obtained by Salcudean *et al* (1994a). The measured film cooling effectiveness is shown in Figure 7.14 for comparison. The spanwise average effectiveness is shown in Figure 7.15 for $M_\infty = 0.52$ together with the experimental curve. Here the x-axis is the curved distance x/d from the stagnation line and the y-axis is the spanwise averaged film cooling effectiveness. The agreement between the two results is good.

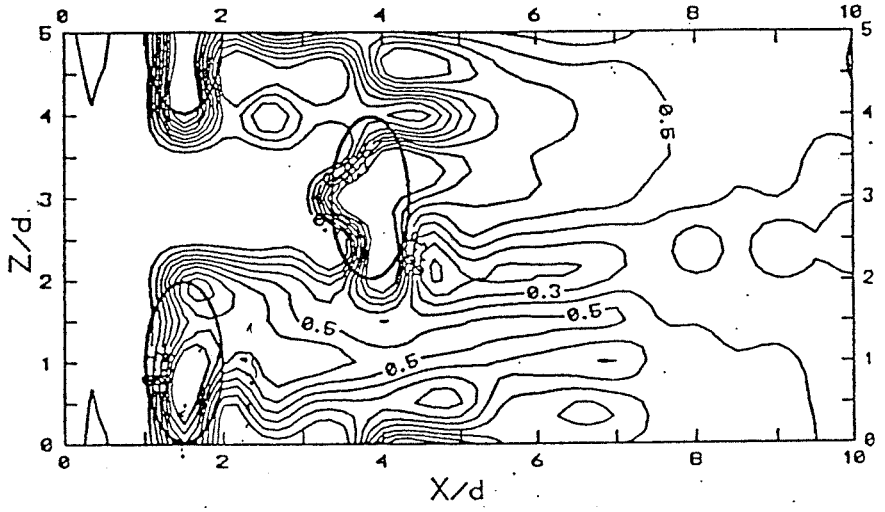

Figure 7.14: Contours of measured FCE for $M_\infty = 0.52$ at the blade surface.

The film cooling effectiveness on the blade surface for $M_\infty = 0.97$ is shown in Figure 7.16. For comparison, the experimental contours of film cooling effectiveness are shown in Figure 7.17 for a mass flow ratio of $M_\infty = 0.97$. There are some discrepancies between the predicted and measured results. It can be seen that the coolant from the first row of orifices is blown toward the orifices of the second row, leaving a significant region of the blade surface with poor coolant coverage. This observation is in agreement with experiments (Salcudean *et al* 1994a). The predicted spanwise averaged film cooling effectiveness together with the experimental curve is shown in Figure 7.18. The agreement is fair but not as good as for $M_\infty = 0.52$. At high mass flow ratios the coolant penetrates deeper into the outer flow field and produces a much more complex flow involving anisotropic film jet spreading and non-equilibrium turbulence. Therefore, the treatment of turbulence using the standard $k - \epsilon$ model does not represent the flow as accurately.

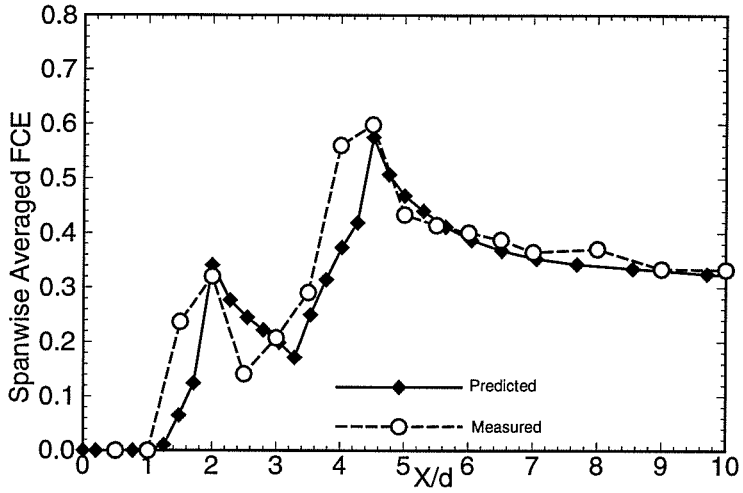

Figure 7.15: $M_\infty = 0.52$; Comparison of the spanwise averaged film cooling effectiveness with the measured result.

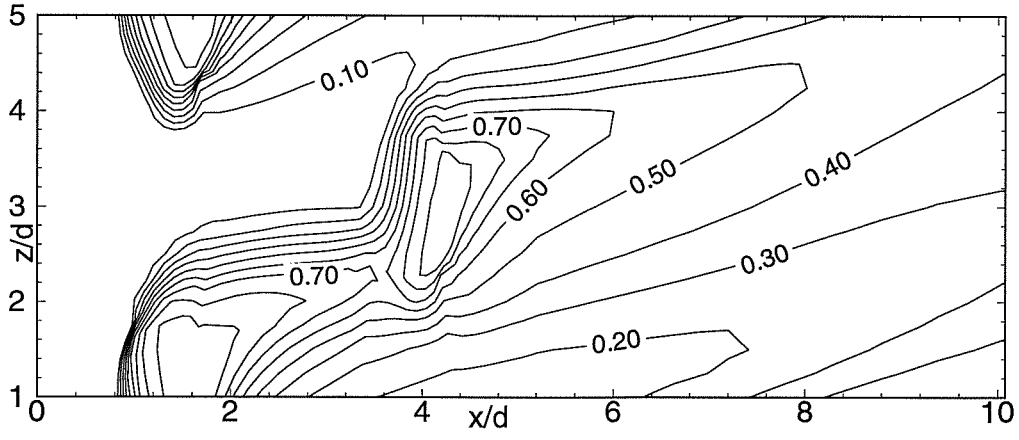

Figure 7.16: Contours of predicted FCE for $M_\infty = 0.97$ at the blade surface.

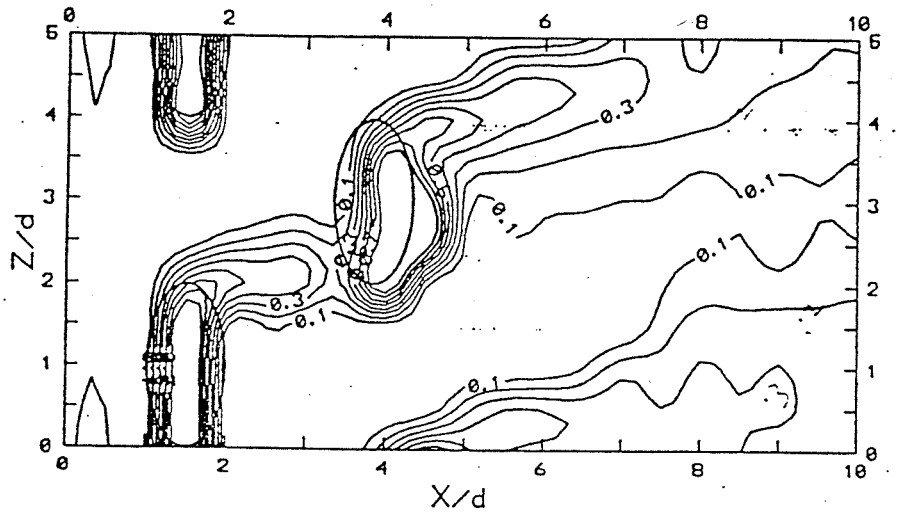

Figure 7.17: Contours of measured FCE for $M_\infty = 0.97$ at the blade surface.

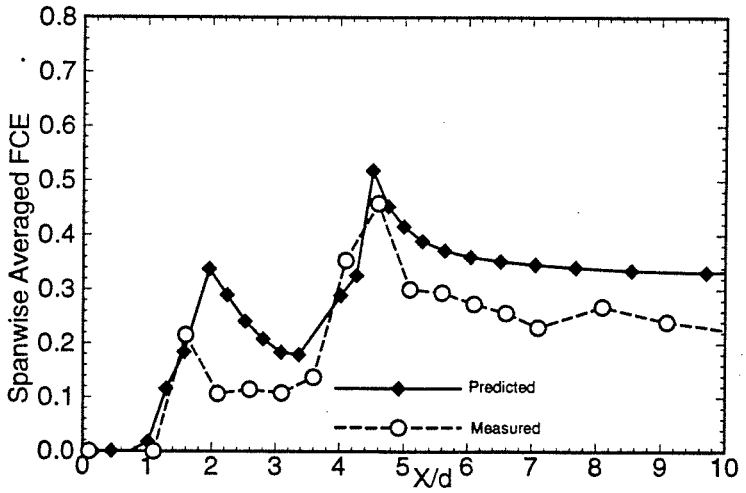

Figure 7.18: $M_\infty = 0.97$; Comparison of the spanwise averaged film cooling effectiveness with the measured result.

7.7 Closure

Computations of film cooling are carried out for a UBC experimental blade model using the methods described in this thesis. From the present calculations, the following observations are made:

- The flow field shows significant differences between the first row and the second row because of the pressure variation on the blade surface. Flow at the exit of the orifices is very non-uniform, particularly at the front row.

- The interaction between the main flow and coolant shows a different vortex formation in the present spanwise injection from that observed in streamwise injection. The vortices are weaker, form further downstream, and are further from the blade surface. Therefore, the lifting of the coolant from the surface which occurs for streamwise injection is not as significant for spanwise injection.

- Examination of the film cooling effectiveness shows good coverage for $M_\infty = 0.52$, where the coolant from the first row flows between the orifices of the second row. For $M_\infty = 0.97$ the coolant from the first row of orifices blows toward the orifices of the second row and therefore the coverage is poor. A comparison of film cooling effectiveness with experimental observations shows good agreement for $M_\infty = 0.52$ and fair agreement for $M_\infty = 0.97$.

- The present study shows that the computational methods and codes developed in this thesis have the potential to model the complex cooling process in an actual turbine blade geometry.

Further improvements of the computational results may be possible with improvements of the turbulence and near wall modeling, and extension of the computational domain to include a part of the plenum.

Chapter 8

Conclusions And Recommendations

8.1 Conclusions

A computational capability has been developed for the prediction of film cooling of turbine blades. This includes the development of two comprehensive numerical codes and the associated methods. The codes involve several numerical methods, including curvilinear coordinate-based calculations, multigrid acceleration, domain segmentation, and grid generation. A detailed investigation of these methods is carried out and novel techniques are proposed to improve the methods. Conclusions on theory development and application are given in the following.

- A computational method using general curvilinear grids is proposed. The method uses a novel discretization to overcome the difficulties associated with the use of non-smooth grids. The numerical scheme is formulated directly using the coordinate-invariant governing equations and the physical geometric quantities which include the cell-surface areas, the surface normal vectors, and the cell volumes, instead of the commonly-used covariant and contravariant vectors. This method avoids the coordinate derivatives of transformation which are not well defined for non-smooth grids and allows for the use of significantly non-smooth grids. A new scheme for handling the non-orthogonal terms in the momentum equations is proposed. This scheme contributes to the main diagonal terms in the resulting coefficient matrix and allows for an implicit treatment of the non-orthogonal quantities,

without increasing the number of computational molecules. This treatment of the non-orthogonal terms enhances computational stability and convergence rate. The physical tangential velocity components, resulting from the velocity expansion in the unit tangent vector basis, are proposed as dependent variables in the momentum equations. A coupled solution procedure is used in place of the pressure-correction equation associated with grid non-orthogonality. A number of flow problems have been studied using the proposed method which include flows in a skewed cavity, flows in a curved pipe and ducts, and flows through a circular cylinder.

- Calculations of turbulent flow in curvilinear coordinates have been investigated. The curvilinear coordinate-based method developed for laminar flows in complex geometries has been generalized to turbulent flows. The $k-\epsilon$ two-equation model together with the 'wall function' treatment is used as the turbulence closure. Discretization of the $k-\epsilon$ equations is presented with particular attention to the linearization of the source terms and calculation of the turbulence energy generation rate. Methods are introduced to facilitate the calculation of the energy generation rate and enhance the computational stability. A formulation for the 'wall function' in general curvilinear grids is presented. The performance of the developed method is studied through several three-dimensional, turbulent flows.

- Multigrid acceleration of three-dimensional, laminar flows in general curvilinear grids is studied. It was found that the discrete governing equations on different grid levels can become inconsistent in some curvilinear grids due to complex flow boundaries, thus reducing the efficiency of the multigrid algorithm. A novel technique has been proposed to solve this problem. Computations using grids with both significant non-orthogonality and strong curvature showed that the developed

multigrid algorithm is very efficient. Unlike the phenomenon reported in the literature, the number of effective grid levels useful for the multigrid method was not limited by the geometrical complexities of the computational domain with the present method, even for seriously degraded coarse grids.

- Several techniques which help the performance of the multigrid method in turbulent flows are introduced. Particularly, it is found that the formulation of the coarse-grid defect equation using the 'wall function' can cause inconsistency of the governing equations between fine and coarse grids. This problem is discussed in detail and a novel practice is proposed to allow the successful implementation of the multigrid method in turbulent flows. A similar approach was also developed independently by Sun (1994).

- Calculations using block-structured (multi-block) curvilinear grids are investigated. The method developed for single-domain computations in general curvilinear grids is generalized to multi-domains by using a domain segmentation technique. Particular attention has been given to the communication between different domains. The performance of the multi-block method is investigated through several computational examples which show that the developed method works well and allows for the use of significantly discontinuous grid slopes at interfaces.

- One of the contributions of the present thesis is that a block-structured, curvilinear coordinate-based, finite-volume code, CMGFD, has been developed based on an existing Cartesian coordinate-based CFD code. The methods described in this thesis, such as the curvilinear coordinate-based method, have been implemented in the code in several steps, from simple cases to complex ones. At each step, detailed investigations of the method and validations of the code are performed in order to achieve an efficient and accurate code. An efficient multigrid method

is implemented in the code and promises fast convergence for both laminar and turbulent flows. This code has the capability to deal with arbitrary geometries by using separate curvilinear grids in different flow regions. There is no restriction for curvilinear grids to be used and no limitation on the number of segments which can be used in the flow region.

- A multi-block and multigrid grid generation code, MBEGG, has been developed to support the application of the CMGFD code. The code is based on an elliptic grid generation method which solves three elliptic, partial differential equations. Methods to achieve certain grid quantities, such as stretching and orthogonality, through the use of control functions have been implemented in the code. When these grid quantities are initially specified by users, the code can choose the appropriate source terms to obtain the desired grid. A multigrid method is developed to solve the three elliptic equations which provides an efficient grid generator. The capability of the code is demonstrated through the application to a number of problems. It can be used to generate block-structured curvilinear grids in arbitrary 2D or 3D domains.

- Computations are carried out for film cooling of a turbine blade model at UBC. In these calculations, the flow and the associated heat transfer are solved using a block-structured, curvilinear grid generated by the MBEGG code which exactly represents the inclined, round film-holes and the curved blade surface. Detailed flow field and heat transfer data are obtained which improves the understanding of the complex flow. The predicted vortex structures show how the coolant is lifted and mixed with hot flows. Comparisons with experiments show good agreement at low mass flow ratios, however, the agreement deteriorates at higher mass flow ratios due to deficiencies of the $k - \epsilon$ turbulence model with 'wall function' treatment.

Further improvement is expected by using more sophisticated turbulence models.

8.2 Recommendations for Future Work

Some recommendations for future work are made as follows:

- The turbulence modeling is a major source of numerical errors for many engineering flow calculations. The present study shows that the $k - \epsilon$ model with the 'wall function' treatment is inadequate for predicting film cooling of turbine blades with high mass flow ratios. Further studies are needed in this area, possibly using non-isotropic, higher-order closure turbulence models or multiple-time-scale models. The near-wall turbulence treatment to represent the increased mixing occurring at high mass flow ratios should be further investigated.

- At the time of this study the CMGFD code is limited to incompressible flows with constant density. However, many engineering flows involve variable density, such as the film cooling problem with cool air or CO_2 as coolant. Therefore, it is worthwhile to generalize the CMGFD code to include variable density flows.

- The film cooling efficiency of a turbine blade is influenced by many factors, such as the mass flow ratio, and the shape and location of the injection orifices. Therefore, in order to achieve an optimal design for film cooling, a parametric study should be carried out. This is possible with the numerical codes developed here.

- The transfer of data between subdomains is critical for multi-domain calculations. It is often the source of numerical error and slow convergence, especially for non-matched grids at interfaces. Some methods use one set of overlapping cells while other methods use two sets of overlapping cells to facilitate the data transfer between neighbouring subdomains. However, there is a lack of detailed information

about the efficiency of these methods. Further investigations would be useful in order to achieve an efficient and accurate solver.

- The numerical scheme adopted in this thesis is based on the power-law profile of Patankar (1980) for convection-diffusion which is a low-order scheme. Most of the applications of higher-order schemes are limited to Cartesian coordinates. The development and investigation of higher-order schemes in curvilinear grids would be very worthwhile.

- Grid generation is one of the major tasks in numerical simulation for flows in complex geometries. The developed grid generation code, MBEGG, can be used for generating block-structured, curvilinear grids in complex geometries. However, there are several aspects which require further consideration. In particular, the code has no capability for generating grids with continuous grid slopes across the interfaces for multi-domains. One possible method is to iteratively adjust the source terms near the interface to achieve smooth grids.

- Although the CMGFD and MBEGG codes are able to simulate laminar/turbulent flows in arbitrary 2D/3D geometries, the application of these codes is problem/user dependent. It would be very useful to develop these codes into a user-friendly package.

Bibliography

[1] Z. Abdullah, 1993, "Calculations of a hydro-cyclone", Private Communication.

[2] M. Agouzoul, M. Reggio and R Camarero, 1990, "Calculation of turbulent flows in a hydraulic turbine draft tube", J. Fluids Eng., Vol. 112, pp.257-263.

[3] B.S. Baldwin and H. Lomax, 1978, "Thin-layer approximation and algebraic model for separated turbulent flows," AIAA paper, pp.78-257.

[4] G.Bergeles, A.D.Gosman and B.E.Launder, 1978, "The turbulent jet in a cross stream at low injection rates: a three dimensional numerical treatment", Numerical Heat Transfer, Vol.1, pp.217-242.

[5] M.E. Braaten and W. Shyy, 1986, "Comparison of iterative and direct solution methods for viscous flow calculations in body-fitted coordinates", Int. J. Nume. Meth. in Fluids, Vol. 6, pp. 325-349.

[6] M. E. Braaten and S. V. Patankar, 1989, "A block-corrected subdomain solution procedure for recirculating flow calculation", Nume. Heat Transfer, Part B, Vol. 15, pp.1-20.

[7] A. Brandt, 1981, *Guide to Multigrid Development, Lecture Notes in Mathematics*, 960, Springer-Verlag, New York.

[8] A.I. Borisenko and I. E. Tarapov, 1968, *Vector and Tensor Analysis with Applications*, Prentice-Hall, Inc.

[9] L. S. Caretto, R. M. Curr, and D. B. Spalding, 1972 "Two numerical methods for three-dimensional boundary layers", Comp. Meth. Appl. Mech. Eng., Vol.1, pp.39-57.

[10] L. Davidson and P. Hedberg, 1989, "Mathematical derivation of a finite volume formulation for laminar flow in complex geometries", Int. J. Numer. Methods Fluids, Vol.9, pp.531-540.

[11] I. Demirdzic, A. D. Gosman, R. I. Issa and M. Peric, 1987, "A calculation procedure for turbulent flow in complex geometries", Compu. & Fluids, Vol.15, pp.251-273.

[12] I. Demirdzic, Z. Lilek and M. Peric, 1992, "Fluid flow and heat transfer test problems for non-orthogonal grids: bench-mark solutions", Int. J. Numer. Methods in Fluids, Vol.15, pp.329-354.

[13] A.D.Demuren, W.Rodi and B.Schonung, 1986, "Systematic study of film cooling with a three dimensional calculation procedure", Journal of Turbomachinery, Vol. 108, 124-130.

[14] Q. V. Dihn, R. Glowinsi, and J. Periaux, *Solving Elliptic Problems by Domain Decomposition Methods with Applications*, in G. Birkhoff and A. Schoenstadt (eds.), Elliptic Problem Solvers II, Academic Press, New York, 1984.

[15] M.M. Eanyet, M.M. Gibson, A.M.K.P. Taylor and M. Yianneskis, 1982, "Laser-doppler measurements of laminar and turbulent flow in a pipe bend', Int. J. Heat and Fluid Flow, Vol.3 pp.213-219.

[16] M. Faghri, E.M. Sparrow, and S.T. Prata, 1984, "Finite difference solutions of convection diffusion problems in irregular domains using nonorthogonal coordinate transformation', Numer. Heat Transfer, Vol.7, pp.183-209.

[17] P. F. Galpin, J. P. Van Doormaal, and G. D. Raithby, 1985, "Solution of the incompressible mass and momentum equations by application of a coupled line solver", Int. J. Numer, Methods Fluids, Vol.15, pp.615-625.

[18] V.K. Garg, and R.E. Gaugler, 1993, "Heat transfer in film-cooled turbine blades", ASME paper 93-GT-81.

[19] V.K. Garg and R.E. Gaugler, 1994, "Prediction of film cooling on gas turbine airfoils", ASME paper 94-GT-16.

[20] I. Gartshore, M. Salcudean, Y.Barnea, K.Zhang, and Aghdasi, F., 1993, "Some effects of coolant density on film cooling effectiveness", ASME 93-GT-76.

[21] R.J.Goldstein, *Advances in Heat Transfer*, Vol.7, pp.321-379, 1971.

[22] P. He and M. Salcudean, "A numerical method for 3D viscous incompressible flows using non-orthogonal grids", J. of Numer. Method in Fluids, Vol.18, pp.449-469.

[23] P. He, M. Salcudean and I.S. Gartshore, 1994, "Multigrid calculation of laminar and turbulent flows using general curvilinear grids", Proc. of CFD94 of Canada, Toronto.

[24] P. He, M. Salcudean and I.S. Gartshore, " Computations of film cooling for the leading edge region of a turbine blade model", Preprint, (1994).

[25] P. He, M. Salcudean and I. S. Gartshore, 1993, "A nonlinear multigrid algorithm for grid generation", Technical Report, The Department of Mechanical Engineering, UBC.

[26] P. He, *CMGFD: A 3D Curvilinear Coordinate-Based Multi-Block Multigrid CFD Code*, Technique, CFD group, Dept. of Mech. Eng., The University of British Colubia.

[27] M. Hinatsu and J. H. Ferziger, 1991, "Numerical computation of unsteady incompressible flow in complex geometry using a composite multigrid technique", Int. J. Numer. Methods Fluids, Vol.13, pp.971-997.

[28] D. S. Joshi and S. P. Vanka, 1991, "Multigrid calculation procedure for internal flows in complex geometries", Numer. Heat Transfer, B, Vol.20, pp.61-80.

[29] B.A. Jubran, 1989, "Correlation and prediction of film cooling from two rows of holes", J. Turbomachinery, Vol.111, pp.502-509.

[30] S.C.Kacker, B.R.Par and J.H.Whitelaw, 1969, "The prediction of wall jets flows with particular reference to film cooling", Progress in Heat Transfer, Vol.2, pp.163-186.

[31] K.C. Karki and S.V. Patankar, 1988, "Calculation procedure for viscous incompressible flows in complex geometries", Numer. Heat Transfer, Vol.14, pp.295-307.

[32] A. Klein, 1981, "REVIEW: Turbulent developing pipe flow", J. Fluids Eng., Vol.103, pp.242-249.

[33] B. E. Launder and D. B. Spalding, 1974, "The numerical computation of turbulent flows", Comp. Meth. Appl. Mech. Eng., Vol. 3, pp.269-275.

[34] D. Lee and J.J. Chiu, 1992, "Covariant velocity-based calculation procedure with nonstaggered grids for computation of pulsatile flows", Numer. Heat Transfer, Part B, Vol.21, pp.269-286.

[35] M. A. Leschziner and K. P. Dimitriadis, 1989, "Computation of three-dimensional turbulent flow in non-orthogonal junctions by a branch-coupling method", Comp. Fluids, Vol.17, pp.371-396.

[36] J.H. Leylek and R.D. Zerkle, 1993, "Discrete-jet film cooling: A comparison of computational results with experiments", ASME 93-GT-207.

[37] C.R.Maliska and G.D.Raithby, 1984, "A method for computing three dimensional flows using non-orthogonal boundary-fitted coordinates", Int. J. Numer. Methods Fluids, Vol.4, pp.518-537.

[38] A.B. Mehendale and J.C. Han, 1992, "Influence of high mainstream turbulence on leading edge film cooling heat transfer" ASME Journal Of Turbomachinery, Vol. 114, pp.707-715.

[39] M.C. Melaaen, 1992, "Calculation of Fluid Flows with Staggered and Nonstaggered Curvilinear Nonorthogonal Grids- Theory", Numer. Heat Transfer, Part A, Vol. 21, pp.1-20.

[40] W.J. Mick and R.E. Mayle, 1988, "Stagnation film cooling and heat transfer including its effect within the hole pattern," ASME Journal of Turbomachinery, Vol. 110, pp.66-72.

[41] P. Nowak, 1992, "A multigrid and multi-block method", Technique Report, The University of British Columbia.

[42] C. W. Oosterlee and P. Wesseling, 1992, "A multigrid method for an invariant formulation of the incompressible navier-stokes equations in general co-ordinates", Comm. Applied Nume. Methods, Vol.8, pp.721-734.

[43] S. Ou and J.C. Han, 1991, "Influence of mainstream turbulence on leading edge film cooling heat transfer through two rows of inclined film slots", ASME 91-GT-254.

[44] S. Ou and J.C. Han, 1992, "Influence of mainstream turbulence on leading edge film cooling heat transfer through two rows of inclined film slots", Transactions of the ASME, Vol. 114, pp.724-733.

[45] S.V. Patankar, *Numerical Heat Transfer and Fluid Flow*, Hemisphere, Washington, D.C., 1980.

[46] C.Y. Perng and R.L. Street, 1991, "A coupled multigrid-domain-splitting technique for simulating incompressible flows in geometrically complex domains", Int. J. Numer. Methods Fluids, Vol.13, pp.269-286.

[47] M. Perić, 1990, "Analysis of pressure-velocity coupling on nonorthogonal grids", Numer. Heat Transfer, Part B, Vol.17, pp.63-82.

[48] D. Rayner, 1991, "Multigrid flow solutions in complex two-dimensional geometries", Int. J. Numer. Methods in Fluids, Vol.13, pp.507-518.

[49] J.W. Richman and R.S. Azad, 1973 "Developing turbulent flow in smooth pipes", Apl. Sci. Res., Vol.28, pp.419-441.

[50] W. Rodi, S. Majumdar, and B. Schonung, 1989, "Finite volume methods for two-dimensional incompressible flows with complex boundaries", Comp. Meth. in Appl. Mech. and Eng., Vol.75, pp.369-392.

[51] M. Rosenfeld, D. Kwak and M. Vinokur, 1988, "A solution method for the unsteady incompressible Navier-Stokes equations in generalized coordinate systems", AIAA Paper AIAA-88-0718.

[52] P. A. Rubini, H. A. Becker and E. W. Grandmaison, A. Pollard, A. Sobiesiak and C. Thurgood, 1992, "Multigrid acceleration of three-dimensional, turbulent, variable-density flows", Numer. Heat Transfer, Part B, Vol.22, pp.163-177.

[53] M. Salcudean, I. Gartshore, K.Zhang, and Y.McLean, 1993, "An experimental study of film cooling effectiveness near the leading edge of a turbine blade", J. of Turbomachinery, Vol. 116, pp.71-76.

[54] M. Salcudean, I. Gartshore, K.Zhang, and Y.Barnea, 1994a, "Leading edge film cooling of a turbine blade model through single and double row injection: effects of coolant density", ASME 94-GT-2.

[55] M. Salcudean, P. He, P. Nowak, I.S. Gartshore and Z. Abdullah, 1994b, *CMGFD: A Curvilinear Coordinate-based, Multigrid and Multi-Block CFD Code*, The University of British Columbia.

[56] P. Sathyamurthy and S.V. Patankar, 1990, "Prediction of film cooling with lateral injection", Heat Transfer in Turbulent Flow, pp. 61-70.

[57] A. Segal, P. Wesseling, J. Van Kan, C.W. Oosterlee and K. Kassels, 1992, "Invariant discretization of the incompressible Navier Stokes equations in boundary fitted coordinates", Int. J. Numer. Meth. Fluids, Vol.15, pp.411-426.

[58] R.K. Shah and A.L. London, *Advances in Heat Transfer*, Supplement 1, Academic Press, Inc., 1978.

[59] W. Shyy, S.S. Tong and S.M. Correa, 1985, "Numerical recirculating flow calculations using a body-fitted coordinate system", Numer. Heat Transfer, Vol.8, pp.99-113.

[60] W. Shyy and M. Braaten, 1996, "Three-dimensional analysis of the flow in a curved hydraulic turbine draft tube", Int. J. Numer. Methods Fluids, Vol.6, pp.861-882.

[61] W. Shyy and C. Sun, 1993a, "Development of a pressure-correction/staggered-grid based multigrid solver for incompressible recirculating flows", Comp. Fluids, Vol.22, pp.51-76.

[62] W. Shyy and C. Sun, 1993b, "Multigrid computation for turbulent recirculating flows in complex geometries", Numer. Heat Transfer, B, Vol.23, pp.79-98.

[63] S. Sivaloganathan and G. J. Shaw, 1988, "A multigrid method for recirculating flows", Int. J. Nemer. Method in Fluids, Vol.8, pp.417-440.

[64] K. M. Smith, W. K. Cope and S. P. Vanka, 1993, "A Multigrid procedure for three-dimensional flows on nono-orthogonal collocated grids", Int. J. Numer. Methods in Fluids, Vol.17, pp.887-904.

[65] Y. Sun, 1994, "The wall function treatment for multigrid calculations", Private Communication.

[66] J.L. Steger and R.L. Sorenson, 1979, "Automatic mesh-point clustering near a boundary in grid generation with elliptic partial differential equations". J. Comp. Phys., Vol. 33, pp.405-410.

[67] A. M. K. P. Taylor, J. H. Whitelaw and M. Yianneskis, 1982, "Curved ducts with strong secondary motion: velocity measurements of developing laminar and turbulent flow", J. Fluids Eng., Vol.104, pp.350-359.

[68] J.F.Thompson, Z.U.A.Warsi and C.W.Mastin, 1982, "Boundary- fitted coordinate systems for numerical solution of partial differential equations - A Review", J. of Comp. Phys., Vol.47, pp.1-108.

[69] J.F.Thompson, 1984, "Grid generation techniques in computational fluid dynamics", AIAA Journal, Vol.22, pp. 1505-1521.

[70] J.F. Thompson, Z.U.A. Warsi and C.W.Mastin, *Numerical Grid Generation, Foundations and Applications*, Elsevier, New York, 1985.

[71] P.D. Thomas and J.L. Middlecoff, 1979, "Direct control of the grid point distribution in meshes generated by elliptic equations", AIAA Journal, Vol.18, pp.652-656.

[72] S.P. Vanka, 1986a, "Block-implicit multigrid solution of Navier- Stokes equations in primitive variables", J. Compu. Phys., Vol.65, pp.138-158.

[73] S.P. Vanka, 1986b, "A calculation procedure for three-dimensional steady recirculating flows using multigrid methods", Comp. Meth. Applied Mech. Eng., Vol.55, pp.321-338.

[74] S. P. Vanka, 1987, "Block-implicit computation of viscous internal flows-recent results", AIAA paper No. AIAA-87-0058, New York.

[75] S.P. Vanka, 1991, "Fast numerical computation of viscous flow in a cube", Numer. Heat Transfer, Part B, Vol.20, pp.255-261.

[76] M. Vinokur, 1989, "An analysis of Finite-difference and Finite-Volume Formulations of conservation laws", J. Compu. Phys., Vol.81.

[77] D.T. Vogel, 1994, "Navier-stokes calculation of turbine flows with film cooling", ICAS 94-2.5.3.

[78] A.J. White, 1981, "The prediction of the flow and heat transfer in the vicinity of a jet in crossflow", ASME 80-WA/HT-26.

[79] D.F. Young and F.Y. Tsai, 1973, "Flow characteristics in models of arterial stenoses-I. steady flow", J. Biomech., Vol.6, pp. 395-410.

[80] J. W. Yokota, 1990, "Diagonally inverted lower-upper factored implicit multigrid scheme for the three-dimensional navier-stokes equations", AIAA J., Vol. 29, pp.1642-1649.

[81] J. Zhou, M. Salcudean and I.S. Gartshore, 1993, "Prediction of film cooling by discrete-hole injection", ASME 93-GT-75.