NASA/CR—1999-209155

Euler Flow Computations on Non-Matching Unstructured Meshes

Udayan Gumaste
University of Colorado, Boulder, Colorado

Prepared under Grant NAG3–1425

National Aeronautics and
Space Administration

Glenn Research Center

June 1999

Acknowledgments

I would like to take this opportunity to express my heartfelt gratitude for my advisor, Prof. Carlos A. Felippa for his guidance and support during the course of my doctoral studies at the University of Colorado. His deep insight, knowledge and vision were a key factor in making this thesis possible. I would also like to acknowledge the suggestions and contributions of the members of the thesis committee, namely Prof. K.C. Park, Prof. C. Farhat, Prof. C.-Y. Chow and Prof. O.A. McBryan. Additional thanks are also due to Prof. Farhat for allowing me to use programs developed by his research group and giving me access to local high-performance machines for my research. Support for this work came from NASA Glenn Research Center under Grant NAG3-1425 monitored by Dr. C.C. Chamis, the National Science Foundation under Grant ECS-9217394 and from NSF GRT Grant GER-9355046 of which Prof. O.A. McBryan was the principal investigator. Bruno Koobus deserves special thanks for explaining to me the mysteries of the fluid solver I used in this research. The many discussions I had with him served as the foundation for this work. Mark Sarkis introduced me to the mortar element method and advised me on its implementation in the present work. While the contributions of these two persons were invaluable to me, the responsibility for any flaws in this work rests with me alone. My work benefited greatly from the ideas and suggestions of many researchers and visitors at the Center for Aerospace Structures: Alain Dervieux, Catherine Lacour, Stéphane Lanteri, Nathan Maman, and Bijan Mohammadi, to name a few. It has been a pleasure to work with Scott Alexander, Po–Shu Chen, Christopher Degand, Rabia Djellouli, Gyula Greschik, Manoel Justino, Scott Keating, Teymour Manzouri, Greg Reich, Hai Tran, and David Vandenbelt.

Table of Contents

List of Figures

List of Tables

Euler Flow Computations on Non-Matching Unstructured Meshes

Udayan Gumaste
Center for Aerospace Structures
College of Engineering
University of Colorado

Chapter 1

Introduction

Speed and economics of air transportation have been revolutionized by the introduction of high-performance engines on aircraft systems. Increasing demands on performance have necessitated higher rotational speeds, thinner airfoils, higher pressure ratios per stage and increased operating temperatures, requiring more optimized designs.

The need to consider several interacting physical disciplines, in addition to very complex geometries, makes engine design a daunting task. Traditionally, engine designers have relied heavily on empirical methods based on past experience and on extensive use of rig and full-scale engine testing. This approach is not only expensive both in terms of time and financial resources, but also dangerous as a few catastrophic failures have been reported during testing. Also, newer and more novel designs require extrapolation of empirical results beyond earlier levels of experience, emphasizing the need for an efficient, economical and reliable analysis system to compliment experimental techniques and evaluate behavior and performance of aircraft engines beforehand.

Rapid developments in computational technology both in terms of high-performance hardware and development of efficient and advanced nu-

merical methods have lead to the application of computational tools to predict engine performance. Combined with measurements and experimental data, these methods provide additional tools for simulation, design, optimization and the calculation of three dimensional flows in highly complex geometries. In many instances, computational methods are the only tools available for simulation, because the actual testing of aircraft engines with detailed measurements in rotating passages is cumbersome, and in many cases, impossible.

While development of methods to predict aerodynamic behavior of engines has received considerable attention, it has been only recently that attempts have been made at performing multidisciplinary analyses of complete aircraft engine systems taking into account coupled interaction of multiple fields.

One particular such interaction is that between the fluid and structural components of the system, generally referred to as *fluid-structure interaction* or particularly as *aeroelasticity* for problems involving high-speed flows.

1.1 Turbomachinery Aeroelasticity

Classically [25], aeroelasticity is defined as the study of the effect of aerodynamic forces on elastic bodies. Pressing demands for improved aerodynamic efficiency of engines has resulted in dynamic problems involving structural integrity, particularly those for the bladed components of the engine. Such problems are generally classified as problems of *turbomachinery aeroelasticity*.

Historically [8], it has been noted that engines that incorporated novel structural or aerodynamic configurations often suffered from flow-induced

blade vibrations in service. In many cases, these problems could not be foreseen in advance, neither analytically nor during the original developmental testing. In some cases, persistent flow-induced vibrations ultimately lead to premature blade failure, both in compressor and turbine stages. Such failures were usually sudden and catastrophic, as even if only a portion of a single blade failed on account of flow-induced vibration, the result would be an instantaneous and total loss of engine power [24].

1.1.1 Types of Blade Vibrations

Aerodynamically induced vibrations are usually classified into two categories, namely flutter and forced vibration. Each of these will be described next.

1.1.1.1 Flutter

Under some conditions, a blade row operating in a completely uniform flow-field can get into a self-excited oscillation called *flutter*. A characteristic of flutter is that the aerodynamic forces are solely dependent on the structural motion, which is sustained by the extraction of energy from flow during each vibratory cycle. The flutter frequency generally corresponds to one of the lower blade of coupled blade-disk natural frequencies.

Blade failure due to flutter occurs predominantly in the compressor and fan stages of engines and to a lesser extent in turbine blading. The outstanding feature of flutter is that very high stresses are generated within the blades leading to very short-term, high-cycle fatigue failures.

1.1.1.2 Forced Vibrations

Destructive forced vibrations can occur in fan, compressor and turbine blading when a periodic aerodynamic forcing function, with frequency close to a system resonant natural frequency, acts on the blades in a given row.

Such forcing functions, which are independent of the vibrational motions of the structure, are generated at multiples of the engine rotational frequency and arise from a variety of sources. For example, aerodynamic disturbances resulting from the presence of upstream and downstream struts, stator vanes, and rotor blades, and disturbances because of inlet flow nonuniformities, rotating stall patterns and compressor surge often lead to forced vibration of blades.

Another important source of resonant forced vibration is the aerodynamic interaction between adjacent blade rows. The two principal types of such interaction are potential flow or static pressure interaction and wake interaction. The former results from the variations in the velocity potential or pressure field associated with the blades of a given row and their effect on the blades of a neighboring row moving at a different rotational speed. This type of interaction is of serious concern when the axial spacings between neighboring blade rows are small or flow Mach numbers are high. Wake interaction is the effect on the flow through downstream blade rows of the wakes shed by one or more upstream rows and persists over considerable distances.

1.1.2 Turbomachinery vs. External Aeroelasticity

While considerable progress has been made in the computational analysis of aeroelastic phenomena for flows around external bodies, such as wings, wing-bodies or complete aircraft, that for aircraft engines and turbomachinery did not gather much momentum until the late 1970s and early 1980s [8]. One of the reasons for this delay was the complex nature of problems encountered

in turbomachinery aeroelasticity which are summarized below [56] :

1. Large multiplicity of closely spaced mutually interfering blades, giving rise to both aerodynamic and structural coupling.
2. Presence of centrifugal loading terms both in the fluid and structural components.
3. Flow in blade cascades is much more complex than that in external flow cases on as it may be sub-sonic, sonic or supersonic depending upon the inlet Mach number and stagger angle giving rise to an intricate Mach reflection pattern.
4. Structural mistuning, which refers to slight differences in mode shapes or frequencies between the blades and can cause localized mode vibrations, in which all the energy in the system is concentrated on one or two blades leading to blade loss.
5. Aerodynamic mistuning, which refers to differences in blade-to-blade spacing and pitch angles altering the unsteady flow characteristics in blade passages.
6. For turbine blades, thermal effects will also have to be considered in addition to the interaction between fluid and structures.
7. The treatment of boundary conditions for fluid solvers is more complicated for internal flows than for external flows.
8. On account of moving components, structural analysis has to have geometric non-linearity capability.

1.2 Thesis Outline

The aim of this research is to apply and develop modern computational tools to simulate and analyze problems in and related to turbomachinery

aeroelasticity of a aircraft engines. This is seen as a first step in the effort to perform a fully coupled multidisciplinary analysis of complete propulsion systems. The central task of current research is to develop a method for simultaneous computer analysis of rotating and non-rotating components in turbomachines, such as in the analysis of a rotor stage and a stator stage.

Chapter 2 begins with a description of the state-of-the-art in turbomachinery aeroelasticity, paying particular attention to development of flow solvers to predict aerodynamic behavior and the coupled treatment of fluid and structural components, highlighting the advantages and drawbacks of recent attempts.

Chapter 3 starts with an explanation of the partitioned analysis approach to solving multidisciplinary coupled problems in engineering and outlines the application of this methodology to the development of software for aeroelastic computations. Details of model preparation and preliminary results obtained from the use of these programs for turbomachinery simulations is also given.

Chapter 4 describes in detail, the two dimensional fluid solver that is used in current research.

Chapter 5 leads to the core of the thesis and deals with the simulation of rotor-stator interaction phenomena in aircraft engines. The problems encountered in such simulations will be presented followed by a summary of recent efforts in this direction.

The method developed during the course of this research is explained in detail in Chapter 6.

Chapter 7 presents results from numerical experiments that were performed to assess the developed method.

Finally, Chapter 8 summarizes current research and makes recommendations for future work.

1.3 Summary

1. Computational methods have been gaining acceptance in design of aircraft engines, especially with the advent of high-performance hardware and efficient numerical algorithms.
2. While considerable progress has been made in the development of advanced fluid solvers to predict aerodynamic performance, coupled treatment of multiple fields has received attention only recently.
3. The interaction between the fluid and structural components in the bladed regions of the engine is of importance as blades have been known to fail by either flutter or forced vibrations induced by aerodynamic loads.
4. Aeroelastic phenomena for internal flows such as in turbomachinery are more complex than for those for external flows on account of a number of reasons, the predominant being increased geometric complexity, mutual interaction between adjacent structural components and presence of thermal and geometric loading.

Chapter 2

A Review of Computational Methods for Turbomachinery Aeroelasticity

This chapter provides an overview of methods currently employed to perform aeroelastic analysis of aircraft engines. Typically, a system for aeroelastic analysis consists of two modules, namely a structural analysis module to determine the structural response to an aerodynamic load and a fluid or aerodynamic analysis module which predicts the unsteady aerodynamic loads based on the geometric boundary defined by the structure. Each of these two modules is essential in any computational setup; however, because of the complexity of the aerodynamic environment within aircraft engines, hitherto greater attention has been given to the development of advanced fluid solvers than their structural counterparts.

This chapter begins with the requirements for accurate modeling of fluid behavior and mentions the assumptions and approximations made to make computer analysis more tractable. Recent developments in the field of computational fluid dynamics (CFD) for turbomachinery applications will be highlighted. Geometric details of blade structures and their general behavior in a rotating environment will be given next, followed by a description of modeling approximations used in some cases. A summary of research in the area of turbomachinery aeroelasticity will conclude the chapter.

2.1 Aerodynamic Modeling for Turbomachinery Applications

Turbomachinery flows are among the most complex encountered in fluid dynamic practice. The characteristics of flow change from region to region within a single turbomachine. For example, flows may be either laminar, transitional or turbulent and locally incompressible, sub-sonic, transonic or supersonic depending upon the location within an aircraft engine [35]. They are also subject to large pressure gradients in all directions and are subject to centrifugal forces and thermal effects on account of combustion. The geometries through which flow occurs are highly complex and simultaneously include both rotating and non-rotating components. In many cases, treatment of boundary conditions becomes very difficult and complicated.

Ideally, to capture all effects, a fluid solver would need to solve the full Navier-Stokes equations with an appropriate turbulence model. However, in order to reduce the total problem size, usually several simplifying assumptions are made, depending upon which types of flows are of interest, mainly to make these problems more amenable to computational solutions. For example, viscous flows through two dimensional cascades with sub-sonic attached flow can be most efficiently predicted using inviscid techniques (such as panel codes or potential codes) coupled with a good boundary layer solver.

A comprehensive review of all the different types of fluid solvers used in practice and the assumptions made therein would be beyond the scope of this thesis and the reader is directed to [35] for further details. This section will review general trends in flow solvers and their assumptions relevant to aeroelastic applications alone. These can be broadly classified into the following types :

1. Approximations made in mathematical modeling of fluid, i.e. the gov-

erning fluid equations.

2. Those made with respect to the three dimensional nature of flow.
3. Assumptions made to reduce the total problem size.
4. Steady-state assumptions.

Each of these assumptions are further clarified below.

2.1.1 Simplified Flow Models

As reported earlier, a Navier-Stokes solver with an appropriate turbulence model would be ideal to capture complete flow physics. However, to reduce storage and computational cost, the governing differential equations are approximated depending upon the type of application and the resolution desired. Some of these approximations are as follows :

1. Inviscid flow : In these approximations, the viscous terms are neglected completely and the resulting equations are solved using any of the following techniques :
    a. panel method (incompressible, irrotational, two dimensional flow)
    b. potential equation (irrotational, two dimensional or three dimensional flow)
    c. stream function equation (two dimensional flow)
    d. Euler equations (two dimensional and three dimensional flow)
2. Boundary layer approximations : When the viscous layers are thin compared to the blade spacing, the streamwise diffusion terms are neglected. The pressure field is assumed to be imposed by the inviscid layer and is prescribed from an inviscid analysis.

3. Parabolized Navier-Stokes equations : This assumption is similar to the boundary layer assumption above except that it allows for normal and transverse pressure gradients. The streamwise pressure gradient is obtained from an inviscid analysis and is continuously updated to capture flow physics correctly.
4. Thin Layer or Reduced Navier-Stokes equations : In this, the streamwise diffusive term is neglected. This is valid only when the viscous layers are thin and is useful when the computational grids are too coarse to resolve the streamwise diffusion terms.
5. Zonal techniques : Zonal techniques enable the use of different approximations in different regions of the engine and their linkage gives a global flow field.

For aeroelastic analysis there is a general consensus that viscous effects can be neglected except in stall and choke flutter [8]. Thus a three dimensional Euler solver would suffice. However, there is no general agreement on the ability of various formulations to capture the important features and stability characteristics of a given problem. Again, some assumptions are made to simplify the solution. These include :

1. Linearized Potential Flow : Two different classes of linearized unsteady cascade theories have been developed :
    a. Theories that linearize about a uniform mean flow.
    b. Theories that linearize about a non-uniform, deflected mean flow.

    Of course, all these theories make the fundamental assumption that the flow is inviscid and of a perfect gas with no shocks. Bendik-

sen [8] has reviewed a large number of such flow solvers and these will not be repeated here.

2. Non-linear Flow Models : Some calculations of flow around cascades with non-linear potential models were reported in the early 1970s. However, these were rare and met with limited success. Nowadays, with great advances being made both in the development of numerical algorithms and availability of powerful computing platforms, both Euler and Navier-Stokes solvers have become quite common and have been reported in significant numbers, for example [14, 15, 19, 27, 32] to name a few.

2.1.2 Assumptions Made with respect to the Three Dimensional Nature of Flow

It should be noted that flow through turbine, compressor and fan rotors is inherently unsteady and three dimensional in nature. For example, large fan rotors have a velocity gradient from the hub, where the flow is sub-sonic, to the tip, where flow is supersonic as a result of blade rotation [9]. This, in addition to the variation of Coriolis forces in the radial direction gives rise to a very complex shock structure from hub to tip. Thus, in order to capture the true nature of flow, a fully three dimensional model is required.

However, on account of limitations in computing power, early researchers used simplified two-dimensional cascade models for flow computations. These models yielded sufficiently good results, in fact, to quote Bendiksen [8], "... it is surprising that [two dimensional] cascade theories have been successful in 'explaining' — if not exactly predicting — the occurrence of flutter in supersonic fans ..."

While some purely two dimensional computations were carried out, more advanced flow solvers were developed based on a theory proposed by

Wu [68] in 1952. In Wu's model, the flow is assumed to follow an axisymmetric streamsurface. The radius and thickness of this streamsurface are assumed to be known as a function of streamwise distance. These quantities are usually obtained from an axisymmetric throughflow or meridional analysis. The equations governing the flow along the streamsurface combine the axial and radial components into one streamwise component and are thus two dimensional. The true three dimensional characteristics of flow can be extracted from this two dimensional approximation as the shape of the streamsurface is known. Specification of the streamsurface allows modeling of blades with variable heights and thicknesses, unlike that for the purely two dimensional solvers which had problems modeling blades of arbitrary shapes. As this approach uses two dimensional analysis to capture three dimensional phenomena, it is called "quasi three dimensional" and is common to many turbomachinery analysis programs.

2.1.3 Assumptions Made to Reduce the Total Problem Size

This assumption is common to many aeroelastic and fluid solvers of all types. For non-aeroelastic fluid solvers, it is obvious that flow through all interblade passages will be identical on account of similarity in geometry. Based on an interesting proposition of Lane [36] in 1957, even aeroelastic analyses, in which there is a change in geometry for each blade passage, can also be performed considering only one or a few interblade passages. This is highly beneficial as the total problem size is reduced by an order of magnitude.

Lane observed that at flutter, adjacent blades vibrate approximately 180 degrees out of phase with respect to each other. He considered the possible flutter mode shapes of a perfect rotor with identical blades and

showed that the flutter mode shapes are remarkably simple : each blade vibrates with identical modal amplitudes but with a constant phase angle σ between adjacent blades. For a rotor with N blades, the possible interblade phase angles are given by :

$$\sigma_n = 2\pi n/N, \quad n = 0, 1, 2, \ldots, N-1$$

Thus the flutter mode is a traveling wave with respect to the rotor. This simple structure of the flutter mode is a direct consequence of the periodicity of cyclic symmetry in geometry which leads to important cyclic properties for both the structure and fluid. From a computational standpoint, Lane's Theorem, which assumes linear structural behavior, allows a full blade row of N blades to be modeled using only a single blade or a few blades.

2.1.4 Time-Accuracy Assumptions

This is an assumption only when a fluid solver is used for aeroelastic analysis. Aeroelasticity is truly an unsteady phenomenon, yet at times, some researchers employed steady-state flow solvers for aeroelastic analysis. This is done by obtaining steady-state solutions from a flow solver and using that to perform a 'static' aeroelastic analysis.

2.1.5 Development of Advanced Flow Solvers

This is to give a very brief overview of the state-of-the-art in CFD for turbomachinery applications.

Keeping in phase with the development of CFD tools for external flows, commendable progress has been made in the development of advanced flow solvers for turbomachinery. Particular emphasis has been laid to develop sophisticated analysis methods to deal with the complex geometries of

aircraft engines, difficulties arising out of that and the modeling of effects that other disciplines have on fluid flow.

In order to obtain fast steady-state flow solutions through complex aircraft engine geometries, advanced solvers are developed to reduce the large diversity between the length and time scales of flow. Prominent amongst these is the work of Adamczyk [2, 16] who uses advanced averaged models to compute flow in multistage turbomachinery. Three averaging operators are developed. The first averaging operator, namely the ensemble average, is introduced to eliminate the need to resolve the detailed turbulent structure of flow. The second operator is used for time-averaging and allows fast computation of steady flows. The last operator, namely the passage-to-passage averaging operator allows simultaneous simulation of flows through blade-rows having variable number of blades and/or rotating speeds. Details of these operators are lengthy and complex and will not be dealt with here. The reader is referred to [2] for the full mathematical formulation.

With growing interest in treating aircraft engines on a more unified basis, particular emphasis is laid on modeling interdisciplinary interaction between fluid and other components. Stewart [61] has developed a program `ENG10` which takes into account the effect of blade forces, loss, combustor, heat addition, blockage, bleeds and convective mixing. This program, in the writer's opinion, represents the true state-of-the-art in turbomachinery flow solvers and can be viewed as an efficient synthesis of existing models for multidisciplinary interaction. An approach similar to that of Adamczyk is used, in which the right-hand sides of Euler equations include averaged terms for blade forces, combustor and other effects mentioned above.

Other notable works in this area are those of Koya and Kotake [32] and Gerolymos [27, 28]. Of these Koya and Kotake are credited the first

truly three dimensional time-dependent Euler calculation for flow through a turbine stage. Gerolymos has developed advanced methods for investigation of flutter in vibrating cascades, employing assumptions made about linear structure behavior and spatial periodicity.

2.2 Structural Modeling and Solvers

While not as complex or involved as modeling the fluid behavior, modeling of blades of a turbomachine requires particular attention on account of the presence of a rotating environment and geometric details. As in the case of aerodynamic modeling, several assumptions are made in the structural modeling of engines. This section will first review the modeling requirements and then mention the approximations made and recent advances in this area.

2.2.1 Requirements for Modeling Turbomachine Blades

Before explaining the requirements for accurate modeling of turbomachinery blades, it is worthwhile to consider how blades are mounted in a turbomachine and their general behavior in a rotating environment.

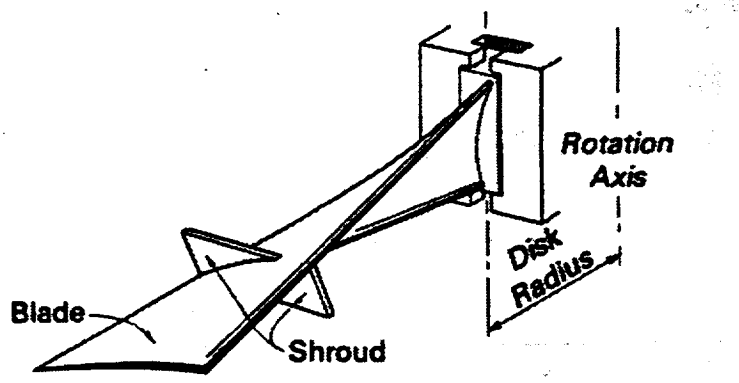

Figure 2.1 : A Typical Shrouded Fan Blade

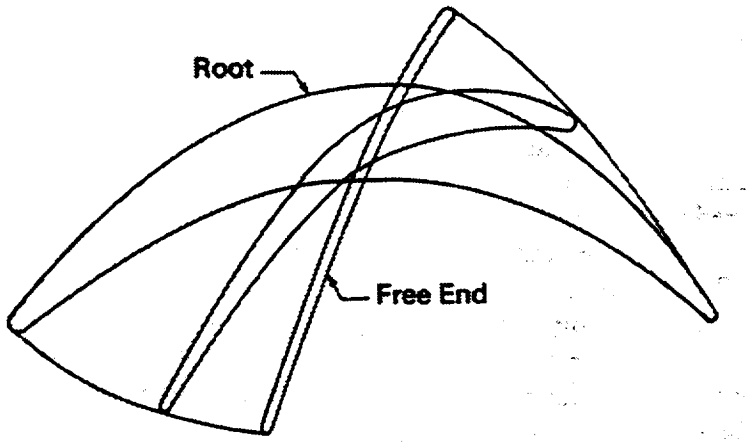

Figure 2.2 : Typical Blade Cross-Section

Typically, a blade is attached to a circular disk by a dovetail joint, as seen in Figure 2.1 from [40]. Usually a single stage of a turbomachine comprises of 30-40 blades attached circumferentially on the periphery of a disk. The disk and its attached blades rotate at a large angular velocity about an axis perpendicular to the plane of the disk. On account of the centrifugal forces the blade experiences, there is a small, but significant, amount of blade untwisting. This causes the shrouds of the adjacent blades to lock resulting in an added stiffness for each individual blade.

Blade geometries by themselves are from simple and vary greatly from hub to tip both in terms of thickness and cross-sectional profile. Figure 2.2 shows a highly twisted, actual turbine blade, with cross-sections taken at the root, the free end and midway along the blade [40].

In addition to the changes in blade geometry because of centrifugal loading as mentioned above, blades undergo large vibrational motion on account of aerodynamic loads imposed by the fluid.

With this background, the following requirements can be identified

for structural analysis of turbomachinery blades :

1. The structural analysis program should capable of handling centrifugal loads.
2. Non-linear geometric effects should be included to be able to trace changes in blade geometry on account of centrifugal loading and vibrations on account of aerodynamic loads.
3. Another geometric feature required would be the capacity of handle contact problems such as the locking of shrouds and their effect on overall structural stiffness.
4. Thermal effects will need to be considered in turbine blading.

2.2.2 Modeling Assumptions and Approximations

Several assumptions are made with regard to the aeroelastic behavior of blades in a turbomachine which leads to a few approximations as outlined below.

Primary among these is the use of only a few blades to model an entire stage, in accordance with Lane's theorem. This restricts the use of these models only to linear analysis.

Another historical trend is the use of a single torsional degree-of-freedom to model a blade in 2-dimensions. The idea that turbomachinery aeroelasticity is a single degree-of-freedom phenomenon appears to have taken root from the very beginning of interest in this subject [8]. This stemmed from the observation that flutter in blading does not occur by the coalescence of bending and torsional modes but by the adjustment of modal amplitudes and phase angles so as to extract energy from the fluid, usually in the torsional mode. Thus to capture flutter correctly, using only

a torsional degree-of-freedom in two dimensional studies was thought to be sufficient.

However, in reality, as adjacent blades are not in phase with each other, the flutter mode is a traveling wave and it would be possible for the bending mode to alter the extraction of energy from the fluid. Thus, even though flutter may not occur by the coalescence of bending and torsional modes, both bending and torsional modes need to be modeled to investigate the possibility of flutter. The simplest models should (and did) therefore have both bending and torsional degrees-of-freedom.

The next level of modeling approximation was the use of beam models for blades [8, 56]. In this, blades are modeled as straight, slender, twisted elastic beams with a symmetric varying cross-section. Blades are assumed to be rigid in the radially, thus eliminating the equation of motion in that direction. The degrees-of-freedom in this case consist of (a) bending in the plane of rotation, (b) bending in the plane perpendicular to the plane of rotation and (c) a torsional degree-of-freedom about the elastic axis of the beam. This model was based on the geometric non-linear theory of elasticity and gave rise to a set of coupled, but linear, equations of motion. The beam model is adequate only when the blade is relatively thick and has a large aspect ratio. If this is not the case, then beam models are found to be inadequate to capture chordwise bending modes and a two dimensional model is called for [40].

The cross-sections of blades vary greatly from the hub to the tip and hence means have to be found by which they can be modeled in 2-dimensions using what are called equivalent sections in which changes in the spanwise direction are accounted for in an average sense.

More recently, with the development in finite element technology for structural analysis, elaborate finite element models have been used, which take into account geometric non-linear effects on account of large displacements of the blades. However, developing an accurate model of engine blades with complete details such as shrouds remains a challenge even in the age of advanced finite element solvers.

2.2.3 Development of Structural Solvers

The development of structure solvers for aircraft engine applications has not been much different from that for any other structural analyses. In fact, Reddy *et al* [56] mention that most of the structural calculations at NASA LeRC have been performed using NASTRAN.

Some specific stand-alone programs, especially those which take into account thermal and other effects such as bird and ice impacts and also the effects of composites used have also been used for blade analysis though not directly coupled with a fluid solver for aeroelastic analyses [55].

2.3 Summary of Aeroelastic Analysis Programs for Turbomachinery

Some of the assumptions made for aeroelastic analysis of turbomachinery have been mentioned above. The following is a concise summary of research in turbomachinery aeroelasticity till now [6, 56, 59] :

1. Use of potential or Euler solvers with simplifying assumptions.
2. Purely two dimensional, quasi three dimensional or axisymmetric fluid solvers.
3. Only one or a few blades are modeled.
4. To compute structural response, linear structural behavior is assumed. This makes it possible to use quicker frequency domain analysis.

5. Very often, it is found that only static structure response is considered, neglecting inertia effects, even when the method of analysis is for unsteady analysis.
6. For cases in which inertia effects are considered, a very simplified structural model is used, with as few as 2 DOFs.
7. Even though aeroelasticity is an unsteady phenomenon, steady-state methods are used to compute fluid flow and structure loads are computed at each time step from this steady state solution. As time-accuracy of fluid solvers is sacrificed in order to obtain a fast steady-state solution, this may not yield correct results.
8. Transfer of loads between fluid and structure is done through lift coefficients, thus losing spatial accuracy in computing structural loads.
9. Some fluid solvers use moving meshes for analyzing vibrating blades. Exact details of algorithms for mesh updating are not given and it is probable that these algorithms do not satisfy the geometric conservation law, which will be discussed later (Section 4.1.4).

2.4 Summary

1. Problems in interior aeroelasticity are more difficult to analyze than those in exterior aeroelasticity. Consequently, a number of simplifying assumptions are made in order to make computational treatment possible. On account of the more complex behavior of fluids within turbomachinery than structural behavior, till now greater attention has been paid to development of flow models and flow solvers than their structural counterparts.

2. To simplify the process of flow solution, approximations are made both at the level of physical modeling (two dimensional instead of three dimensional, a single or a few blades instead of a complete cascade) and mathematical modeling (linearized and potential equations instead of complete Navier-Stokes or Euler calculations). More recently, efficient and advanced flow solvers have been developed.
3. Likewise, approximations are also made in structural modeling. These include restriction to the case of linear behavior making frequency domain analysis possible and the use of simplified models (beams and equivalent sections instead of complete blades). Use of advanced finite element structural solvers have resulted in more realistic simulations.
4. In many cases, details of coupling between the structure and fluid components have not been given which could result in discrepancies.
5. A complete system for aeroelastic analysis of aircraft engines without assumptions other than those in the process of discretization still remains to be developed.

Chapter 3

Partitioned Analysis Procedures for the Aeroelastic Problem

This chapter deals with the formulation of coupled field problems for aeroelasticity and their solution using the partitioned analysis approach. It begins by introducing the concept of partitioned analysis and the motivation behind this methodology. Use of partitioned analysis for aeroelastic applications will be mentioned and elaborated upon. The individual software components used for solving the coupled field aeroelastic problem will be briefly overviewed. Lastly, a brief description of initial attempts at using existing technology for external aeroelasticity for internal aeroelasticity applications will be given.

3.1 Partitioned Analysis for Coupled Field Problems

Many current problems in engineering require the integrated treatment of multiple interacting fields. These include, for example, fluid-structure interaction for submerged structures and in pressure vessels and piping, soil-water-structure interaction in geotechnical and earthquake engineering, thermal-structure-electromagnetic interaction in semi- and superconductors and fluid-structure interaction (FSI) in aerospace structures and turbomachinery, the last of which is the focus of attention for current research.

Nowadays, sophisticated and advanced analysis tools are available for individual field analysis. For example, for FSI, advances in the last few decades have resulted in the development of powerful and efficient flow analyzers, which is the realm of interest of computational fluid dynamics (CFD). Equally robust structural analysis tools are available, which is a result of development of advanced finite element methods (FEM). Computer analysis of coupled field problems is a relative newcomer and no standard analysis methodology has been established. One natural alternative is to tailor an existing single-field analysis program to take into account multidisciplinary effects. As an example, fluid volume elements could be added to a FEM structure solver. Another approach would be to unify the interacting fields at the level of governing equations and formulate analysis methods thereupon, for example, as suggested by Sutjajho and Chamis [62].

Both these methods suffer from drawbacks. From a programming point-of-view, addition of modules of different fields leads to an uncontrolled growth in complexity of necessary software. It becomes increasingly difficult to modify existing codes to incorporate improved formulations. For users, a monolithic code can impose unnecessary restrictions in modeling and mesh generation. For example, in FSI, forcing equal mesh refinement on the fluid-structure interface may either cause the structure elements to be too small, making analysis more expensive, or cause fluid cells to be too large, resulting in a loss of accuracy or stability or both.

Partitioned analysis [48] offers an attractive approach in which diverse interacting fields are processed by separate field analyzers. The solution of the coupled system is obtained by executing these field analyzers, either in

sequential or parallel manner, periodically exchanging information at synchronization times. This approach retains modularity of software and simplifies development. It also allows to exploit well established discretization and solution methods in each discipline and does not enforce any specific mesh refinement requirements.

3.2 Mathematical Model for Aeroelasticity

Aeroelasticity deals with the interaction of high speed flows with flexible structures. Thus, in a physical sense, it is a two-field phenomenon. The first field is the structure, for example, the blades of a turbomachine or an entire aircraft, and the second is the fluid flowing around the structure. During coupled interaction, the structure defines the geometric boundary for flow while the aerodynamic load imposed by the fluid on the structure initiates and sustains the structural response. The goal of computational aeroelasticity is to predict or analyze this two-way coupling. Different treatments required for each field pose a challenge for coupled field analysis.

3.2.1 Field Equations

This section introduces the governing differential equations used to describe the structural and fluid components in aeroelastic analysis.

3.2.1.1 Structural Equations

The structure is governed by equations from classical theory of elasticity :

$$\mathrm{div}(\boldsymbol{\sigma}(\epsilon)) + \mathbf{b} = -\rho\ddot{\mathbf{u}} \tag{3.1}$$

where $\boldsymbol{\sigma}$ is the stress tensor and ϵ is the strain tensor which is function of structural displacements. $\mathbf{b}$ represents body forces (such as gravity) acting on the structure and $\mathbf{u}$ denotes structural displacement.

3.2.1.2 Fluid Equations

As mentioned in Section 2.1, ideally, the fluid field should be described by the Navier-Stokes equations. However, in this research, only the Euler equations are considered. The primary reason for this is to speed-up computations and the methods developed can be extended for Navier-Stokes or even turbulence computations. Furthermore, there is a consensus amongst researchers in this area (Section 2.1.1) that for aeroelastic analysis involving turbomachinery, Euler equations suffice. These are written in 3-dimensions as follows :

$$\frac{\partial}{\partial t} W(\vec{x},t) + \vec{\nabla} \cdot \vec{\mathcal{F}}(W(\vec{x},t)) = 0 \tag{3.2}$$

where $\vec{x}$ and t denote the spatial and temporal variables, and

$$W = (\rho, \rho\vec{U}, E)^T, \qquad \vec{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)^T$$

and

$$\vec{\mathcal{F}} = \begin{pmatrix} F_x(W) \\ F_y(W) \\ F_z(W) \end{pmatrix}$$

where $F_x(W)$, $F_y(W)$ and $F_z(W)$ denote the convective fluxes in the x, y and z directions respectively and are given by :

$$F_x(W) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(E+p) \end{pmatrix}, \quad F_y(W) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(E+p) \end{pmatrix} \quad F_z(W) = \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(E+p) \end{pmatrix}$$

In the above expressions, ρ is the density, $\vec{U} = (u, v, w)$ is the velocity vector, E is the total energy per unit volume and p is the pressure. The velocity, energy and pressure are related by the equation of state for a perfect gas :

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho\|\vec{U}\|^2\right)$$

where γ is the ratio of specific heats, $\gamma = 1.4$ for air.

For aeroelastic computations, the flow boundary is defined by the structure (Γ_b) and inflow and outflow boundaries ($\Gamma_{I/O}$). Thus

$$\Gamma = \Gamma_b \cup \Gamma_{I/O}$$

Let $\vec{n}$ denote the outward normal at any point of Γ. Then, the no-slip condition (i.e., no flow normal to the boundary) at the fluid-structure interface is given by

$$\vec{U} \cdot \vec{n} = 0$$

At the same boundary, on the structural partition, the boundary condition is given by

$$\boldsymbol{\sigma} \cdot \vec{n} = -p\vec{n}$$

which prescribes the load on the structure imposed by the fluid pressure.

The flow is assumed to be uniform at the inflow and outflow boundaries and is thus prescribed. The free-stream state vector W_∞ is given by

$$\rho_\infty = 1, \qquad \vec{U}_\infty = (u_\infty, v_\infty, w_\infty) \quad \text{with} \quad ||\vec{U}_\infty|| = 1, \qquad p_\infty = \frac{1}{\gamma M_\infty^2}$$

where M_∞ denotes the free-stream Mach number. The velocity components u_∞, v_∞ and w_∞ are obtained from the angle of attack and the yaw angle.

3.2.2 Discretization of the Field Equations

This section briefly overviews methods used to discretize the field equations described in Section 3.2.1.

3.2.2.1 Structural Equations

The finite element method (FEM) has become more or less the standard to discretize structural equations. Neglecting damping, the discretized structural equations are written as :

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{f}^{ext}$$

or

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{f}^{int}(\mathbf{q}) = \mathbf{f}^{ext}$$

In the above equations, $\mathbf{M}$ is the mass matrix, $\mathbf{K}$ is the stiffness matrix, $\mathbf{q}$ are the structural degrees-of-freedom and $\mathbf{f}^{ext}$ are the external applied loads. In the second of the above equations, the term $\mathbf{f}^{int}(\mathbf{q})$ represents the vector of internal forces within the structure which includes the elastic forces $\mathbf{Kq}$ and could also include other forces such as those due to damping.

3.2.2.2 Fluid Equations

Numerous methods are popular for the discretization, mainly the finite-difference (FD), finite-volume (FV) and finite element (FE) methods. In the current approach, the finite-volume discretization is adopted to discretize the convective fluxes. Details of this process are omitted at present but will be explained in Chapter 4.

3.3 Partitioned Analysis for Aeroelastic Applications

Aeroelasticity deals with the interaction of high-speed flows with flexible structures. Thus, in a physical sense, it is a two-field phenomenon. However, on account of different formulation methods used for the fluid and structure components, computationally, it becomes more convenient to treat this as a three-field coupled problem.

3.3.1 Aeroelasticity as a Three-Field Coupled Problem

Traditionally, structural equations are formulated in Lagrangian co-ordinates, in which the mesh is embedded in the material and moves with it; while the fluid equations are typically written using Eulerian co-ordinates, in which the mesh is treated as a fixed reference through which the fluid moves.

Therefore, in order to apply the partitioned analysis approach in which the fluid and the structure components are treated separately, it becomes essential to move at each time step, at least the portions of the fluid mesh that are close to the moving structure. One of the approaches which obviates the need for partial remeshing of the fluid mesh is one where the moving fluid mesh is modeled as a pseudo-structural system with its own dynamics. Thus, the *physical* two-field aeroelastic problem can be *computationally* formulated as three-field system, comprising of the fluid, the structure and the dynamic mesh. This is the Adaptive Lagrangian-Eulerian (ALE) [17, 41] formulation. The semi-discrete equations governing this three-way coupled problem can be written as follows :

$$\frac{\partial}{\partial t}\left(\mathbf{A}(\mathbf{x},t)\mathbf{w}(t)\right) + \mathbf{F}^c\left(\mathbf{w}(t),\mathbf{x},\dot{\mathbf{x}}\right) = \mathbf{R}\left(\mathbf{w}(t)\right) \tag{3.3a}$$

$$\mathbf{M}\frac{\partial^2 \mathbf{q}}{\partial t^2} + \mathbf{f}^{int}(\mathbf{q}) = \mathbf{f}^{ext}\left(\mathbf{w}(t),\mathbf{x}\right) \tag{3.3b}$$

$$\tilde{\mathbf{M}}\frac{\partial^2 \mathbf{x}}{\partial t^2} + \tilde{\mathbf{D}}\frac{\partial \mathbf{x}}{\partial t} + \tilde{\mathbf{K}}\mathbf{x} = \mathbf{K}_c\mathbf{q} \tag{3.3c}$$

where t designates time, $\mathbf{x}$ is the position of a moving fluid node, $\mathbf{w}$ is the fluid state vector, $\mathbf{A}$ results from the finite-element/finite-volume discretization of the fluid equations, $\mathbf{F}^c$ is the vector of convective ALE fluxes, $\mathbf{R}$ is the vector of diffusive fluxes, $\mathbf{q}$ is the structural displacement vector, $\mathbf{f}^{int}$ denotes the vector of internal forces in the structure, $\mathbf{f}^{ext}$ the vector of external forces, $\mathbf{M}$ is the finite element mass matrix of the structure, $\tilde{\mathbf{M}}, \tilde{\mathbf{D}}$ and $\tilde{\mathbf{K}}$ are fictitious mass, damping and stiffness matrices associated with the moving fluid mesh and $\mathbf{K}_c$ is a transfer matrix that describes the action of the motion of the structural side of the fluid-structure interface on the dynamic fluid mesh. For example, $\tilde{\mathbf{M}} = \tilde{\mathbf{D}} = 0$ and $\tilde{\mathbf{K}} = \tilde{\mathbf{K}}^R$ where $\tilde{\mathbf{K}}^R$ is a rotation matrix corresponds to a rigid mesh motion of the fluid mesh around an oscillating structure, while $\tilde{\mathbf{M}} = \tilde{\mathbf{D}} = 0$ includes the spring-based mesh updating scheme proposed by Batina [7] and Tezduyar *et al* [63].

It should be noted that the three components of the coupled field system described in (3.3) exhibit different mathematical and numerical properties and hence require different computational treatments. For Euler and Navier-Stokes flows, the fluid equations are non-linear. The structural equations may be either linear or non-linear depending upon the type of application. The fluid and structure interact only at their interface, via the pressure and motion of the structural interface. However, the pressure variable cannot be easily isolated from the fluid equations or the fluid state vector $\mathbf{w}$, making the coupling in this three-field problem implicit rather than explicit.

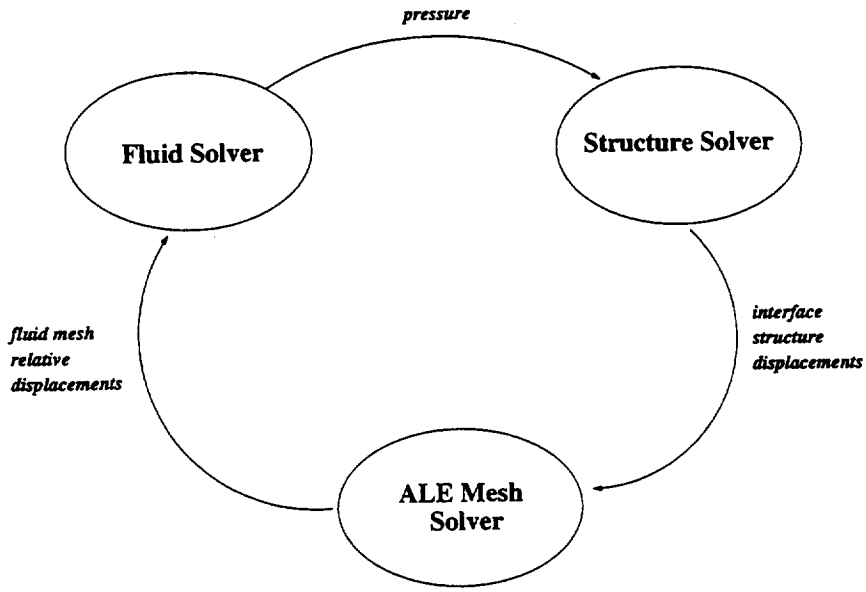

Figure 3.1 : Interaction between Programs for FSI

The simplest possible partitioned analysis procedure for transient aeroelastic analysis is as follows :

- Advance the structural system under a given pressure load.
- Update the fluid mesh according to the movement of the fluid structure interface.
- Advance the fluid system and compute the new pressure load.

This procedure is carried out in cyclic order until the desired end of computations is reached, see Figure 3.1.

3.3.2 Geometric Conservation Law

An interesting feature that arises out of the use of the three-field ALE formulation is the need to take into consideration the motion of fluid volume cells while computing fluxes in the fluid solver. It is shown in [64] that in order to compute flows correctly on a dynamic mesh, it is essential that the

selected algorithm preserves the trivial solution of an uniform flow-field even when the underlying mesh is undergoing arbitrary motions. The necessary condition for the flow solver to accomplish this is referred to in literature as the Geometric Conservation Law (GCL). Failure to satisfy the GCL results in spurious oscillations although the system for which solution is sought is physically stable. Further details about GCL and its current implementation will be given in Section 4.1.4.

3.4 The PARFSI System for Unsteady Aeroelastic Computations

A system of locally developed programs for unsteady aeroelastic computations, PARFSI (**Par**allel **F**luid-**S**tructure **I**nteraction) will be described next. This system consists of a fluid solver, a structure solver, a dynamic ALE mesh solver and a few preprocessing programs for parallel computations.

3.4.1 Fluid Solver

For flow computations, a three dimensional fluid solver for unstructured dynamic meshes is used. This discretizes the conservative form of the Navier-Stokes equations using a mixed finite-element/finite-volume (FE/FV) method. Convective fluxes are computed using Roe's [57] upwind scheme and a Galerkin centered approximation is used for viscous fluxes. Higher order accuracy is achieved through the use of a piecewise linear interpolation method that follows the principle of MUSCL (**M**onotonic **U**pwind **S**cheme for Scalar **C**onservation **L**aw) proposed by Van Leer [66]. Time integration can be performed either explicitly using a 3-step variant of the Runge-Kutta method, or implicitly, using a linearized implicit formulation. An elaborate description of the three dimensional fluid solver can be found in [37].

3.4.2 Structure Solver

A parallel structural analysis program, PARFEM has been developed by Farhat and co-workers over the last few years. This program has a wide range of one-dimensional to three-dimensional finite elements for structural analysis. Time-integration is implicit based on Newmark's method [26]. For parallelization, the FETI (**F**inite Element **T**earing and **I**nterconnecting) [20, 21] domain-decomposition method is used.

3.4.3 ALE Mesh Solver

The fluid mesh is assumed to be a network of springs based on a method proposed by Batina [7]. The solver used to update the fluid mesh is integrated into the fluid code as a subroutine which is called every time there is an exchange of information between the structure and fluid. At each time step t^{n+1}, displacements at the interior nodes are predicted by extrapolating the previous displacements at time steps t^n and t^{n-1}. Nodes on the far-field boundaries are held fixed, while the motion of fluid nodes on the fluid-structure interface is obtained by interpolation of structural displacements.

3.4.4 Matching Non-conforming Fluid and Structure Meshes

Two preprocessing programs have been developed to enable parallel aeroelastic computations. To decompose the fluid and structure meshes, a mesh decomposition software TOP-DOMDEC [23] is used. This is equipped with a range of mesh decomposition algorithms and can also be used as a visualization tool.

As fluid and structure computations are performed by independent programs adhering to the partitioned analysis methodology, the fluid and

structure meshes need not coincide along their interfaces. Hence an interpolation procedure is followed to transfer pressures from the fluid to the structure and displacements from the structure to the fluid. Interpolation information (in terms of interpolation coefficients within elements and association of fluid/structure nodes/elements across the fluid structure interface) necessary for parallel execution of solvers is set up by a preprocessing program MATCHER, described in [45].

3.4.5 Subcycling between Fluid and Structure Solvers

The fluid and structure meshes may have varied degrees of refinement and will hence have different time steps. Subcycling [50] allows the fluid and structure solvers to run concurrently with different time steps by periodic exchange of information at synchronization times. This also makes structural computations more efficient as usually the implicit structure time step is an order of magnitude higher than the explicit fluid time step.

3.5 Application of PARFSI for Turbomachinery Simulations

As a beginning, the existing programs for aeroelastic analysis were used to simulate the aeroelastic response for the blades of the GE-EEE fan stage [30]. Disregarding modifications made to some pre- and post-processing programs, no major modifications were required for any of the field analyzers in computing the response to internal flow using codes primarily designed for external aeroelastic computations. This highlights a major benefit of adopting the partitioned analysis methodology.

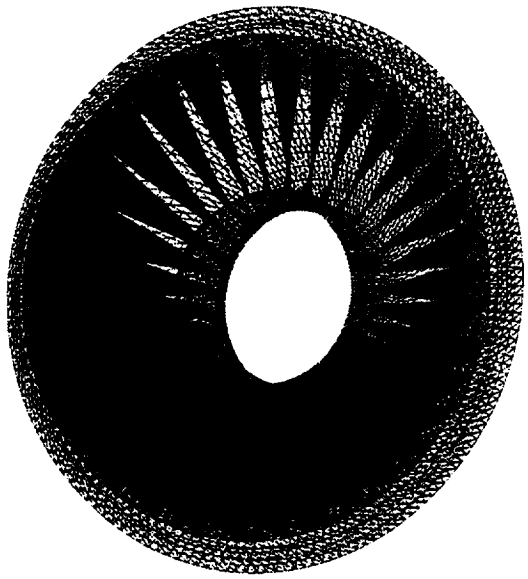

Figure 3.2 : Fluid and Structure Meshes for the GE-EEE Fan Stage

3.5.1 Model Preparation

Two physical models have been used in the aeroelastic simulations.

1. The first model is a single row of blades from the compression stage of the GE-EEE turbofan engine, which serves as a test case for most computational methods at NASA LeRC. This model consists of 32 blades along the circumference. Details of blade geometry were obtained from a NASTRAN FE model. This model has approximately 60,000 fluid nodes and 1,600 structure nodes. For parallel analysis, the fluid mesh was decomposed into 32 subdomains and the structure mesh into 4 subdomains.
2. The second test model was a hypothetical two-row stage which was obtained by using the GE-EEE model mentioned above, setback along

the longitudinal axis of the engine and half-way shifting it in the circumferential direction. In this case, the fluid mesh consisted of approximately 45,000 nodes and the structure mesh has approximately 3,200 nodes. For parallel analysis, 16 sub-domains were used for the fluid and 4 for the structure.

The fluid mesh was generated by first constructing a mesh for a typical cell block which was the passage between two blades. A hexahedral mesh was obtained in the interblade passage by algebraic interpolation and tetrahedra were generated by dividing each individual hexahedron into six tetrahedra. The structure mesh was composed of triangular shell elements having uniform thickness for the sake of simplicity.

Once the structure mesh for a single blade and the fluid mesh for a single interblade passage were obtained, models for the entire stage were easily constructed by rotating these around the circumference.

Wireframe plots of the fluid and structure meshes generated for each of the above cases are shown in Figures 3.2 and 3.3.

3.5.2 Results

It was observed that the blades tend to vibrate in phase with similar amplitudes. A slight coupling effect was observed between the bending and torsional modes of vibration for the blades.

Results for the two-row case were more interesting. The first row appeared to act as a screen and absorbed most of the impact of the aerodynamic load. This caused it to vibrate with a much greater frequency and amplitude than the second row. Again, some bending-torsion coupling was observed in blade vibrations.

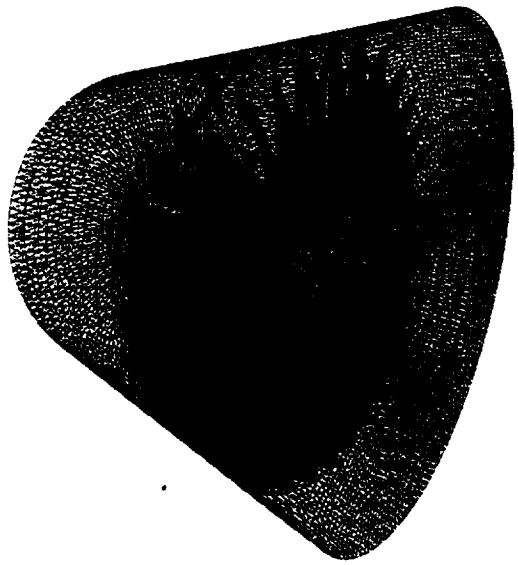

Figure 3.3 : Fluid and Structure Meshes for the Hypothetical 2-Row Stage

3.5.3 Comparison of Current Approach with Other Methods

It is felt that the current approach to turbomachinery aeroelasticity offers the following advantages over other methods :

1. No approximations or assumptions are made regarding either of the structural or fluid components except those in the discretization of the governing equations.
2. Independent modeling of the fluid and the structure allows the use of arbitrary computational procedures for each field without affecting the other. Meshes can be refined independently without having to take into consideration conformity at the fluid-structure interface.
3. Additional field analyzers can be added without greatly increasing the programming complexity of existing software.

4. The use of massively parallel computations allows a fast turn-around which is of great importance in the design process.

3.5.4 Shortcomings

While the use of the PARFSI for turbomachinery aeroelasticity demonstrates the advantages of the partitioned analysis approach and massively parallel computations, several enhancements are required to make the system more applicable to problems of this type :

1. Addition of centrifugal loading and geometric non-linearity capability to the structure solver.
2. Addition of viscous and turbulence effects to the fluid code and the ability to handle differential rotations between rotating and non-rotating components of an engine such as that between a rotor and a stator.
3. Addition of thermal effects.

3.6 Summary

1. Multidisciplinary problems often need to be solved in modern engineering. This involves the simultaneous solution of single field problems for each individual field and the interaction between multiple fields. The partitioned analysis approach allows the analyst to divide and conquer by eliminating unnecessary overheads and restrictions in both software development and physical modeling.
2. Governing equations for the structural and fluid fields need to be considered in a computational treatment of aeroelasticity. However,

on account of a discrepancy in which these equations are formulated, a third fictitious field is introduced as a bridge between the Lagrangian description for the structure and the Eulerian description for the fluid.

3. An advanced, massively parallel system based on the partitioned analysis approach has been developed at the Center for Aerospace Structures for computational simulation of external aeroelasticity problems.
4. Satisfactory results were obtained by applying this system to simulate the aeroelastic response of a single and multirow fan stage of an aircraft engine.
5. While the current technology can be used to a limited extent to perform analyses of these types, several enhancements are required for more realistic simulations.

Chapter 4

Euler Flow Computations on Two-Dimensional Unstructured Meshes

Preliminary results from fully three dimensional simulations of aeroelasticity phenomena in turbomachinery were presented in Chapter 3. As mentioned in that chapter, there are several issues that need to be addressed to make the current computational setup capable of handling more realistic cases typically found in aircraft engines. One identified requirement was the need to allow mutual slipping between meshes associated with different parts of a turbomachine, such as those employed in the analysis of interaction between adjacent rotating and non-rotating components.

This chapter gives a description of a two dimensional Euler solver for unstructured meshes, paying close attention specially to the spatial discretization using the finite-volume method.

4.1 A Finite-Volume Approach to Euler Calculations on Unstructured Meshes

A two dimensional Navier-Stokes [22, 38] solver using a mixed finite element/finite volume formulation on unstructured triangular meshes is described here. For the case under study, namely discontinuous unstructured meshes, the viscous terms have been neglected because they are relatively

unimportant in turbomachinery applications, particularly so in aeroelasticity [8]. Consequently, a description of the finite-element discretization of viscous terms will be omitted for brevity.

4.1.1 Governing Equations

Let $\Omega(t) \subset \mathbb{R}^2$ be the flow domain of interest and $\Gamma(t)$ be its moving and deforming boundary.

A mapping function is introduced between the configuration at time t, $\Omega(t)$ in which time is denoted by t and the mesh point co-ordinates by $\vec{x}$, and a reference configuration $\Omega(0)$ in which time is denoted by τ and the mesh point co-ordinates by $\vec{\xi}$.

$$\vec{x} = \vec{x}(\xi, \tau); \qquad t = \tau \tag{4.1}$$

The non-dimensional conservative form of the Euler equation in an ALE formulation (Section 3.3) can be written as

$$\frac{\partial}{\partial t} W(\vec{x}, \dot{\vec{x}}, t) + J\vec{\nabla} \cdot \vec{\mathcal{F}}^c \left(W(\vec{x}, \dot{\vec{x}}, t) \right) = 0 \tag{4.2}$$

where

$$\vec{\mathcal{F}}^c \left(W(\vec{x}, \dot{\vec{x}}, t) \right) = \vec{\mathcal{F}} \left(W(\vec{x}, t) \right) - \dot{\vec{x}} W$$

in which $\vec{x}$ and t denote the spatial and temporal variables, and W is the state vector given by

$$W = (\rho, \rho u, \rho v, E)^T, \qquad \vec{\nabla} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right)^T$$

The flux vector is

$$\vec{\mathcal{F}} = \begin{pmatrix} F(W) \\ G(W) \end{pmatrix}$$

where $F(W)$ and $G(W)$ denote the convective fluxes given by

$$F(W) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix}, \qquad G(W) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}$$

In the above expressions, $J = \det(d\vec{x}/d\vec{\xi})$ is the Jacobian of the frame transformation $\vec{x} \rightarrow \vec{\xi}$, $\dot{\vec{x}} = (\partial \vec{x}/\partial \tau)_{\vec{\xi}}$ is the ALE mesh velocity, ρ is the density, $\vec{U} = (u, v)$ is the velocity vector, E is the total energy per unit volume and p is the pressure.

The velocity, energy and pressure are related by the equation of state for a perfect gas

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho \|\vec{U}\|^2\right)$$

where γ is the ratio of specific heats, $\gamma = 1.4$ for air.

4.1.2 Boundary Conditions

Three types of boundary conditions can be specified :

1. Inflow boundary conditions : These are specified at the inlet for internal flow calculations.
2. Outflow boundary conditions : These are specified at the exit for internal flow calculations.
3. Slip boundary conditions : These are no through-flow boundary conditions imposed weakly at fixed walls.

The location of these boundary conditions define the limits of the computational flow domain.

4.1.3 Spatial Discretization

There are two inherently linked aspects to the finite-volume spatial discretization :

1. The geometrical discretization of the computational domain by a fluid-volume unstructured triangular mesh and
2. The numerical discretization of the Euler equations on the fluid-volume mesh.

4.1.3.1 Geometrical Discretization

At any time t, the flow domain Ω is assumed to be a polygonal bounded region of $\mathbb{R}^2$. Let $\mathcal{T}_h$ be a standard triangulation of Ω and h the maximal length of the edges of $\mathcal{T}_h$. The vertices of any triangle T are denoted by S_i and the set of its neighboring vertices by $K(i)$. A cell C_i for each vertex S_i is constructed by the union of the sub-triangles resulting from the subdivision by means of the medians of each triangle of $\mathcal{T}_h$ that is connected to S_i, see Figure 4.1. The boundary of C_i is denoted by ∂C_i and the unit outward normal to ∂C_i by $\vec{\nu} = (\nu_{ix}, \nu_{iy})$. The union of all these control volumes constitutes a discretization of domain Ω :

$$\Omega = \bigcup_{i=1}^{N_v} C_i$$

where N_v denotes the total number of triangle vertices in the mesh. Figure 4.2 depicts the dual finite-volume mesh associated with a typical unstructured triangulation.

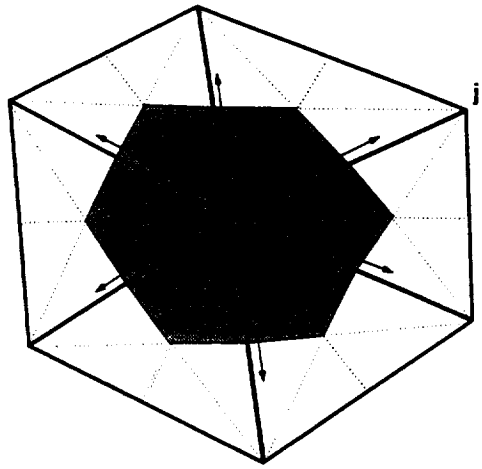

Figure 4.1 : Cell Definition in an Unstructured Mesh

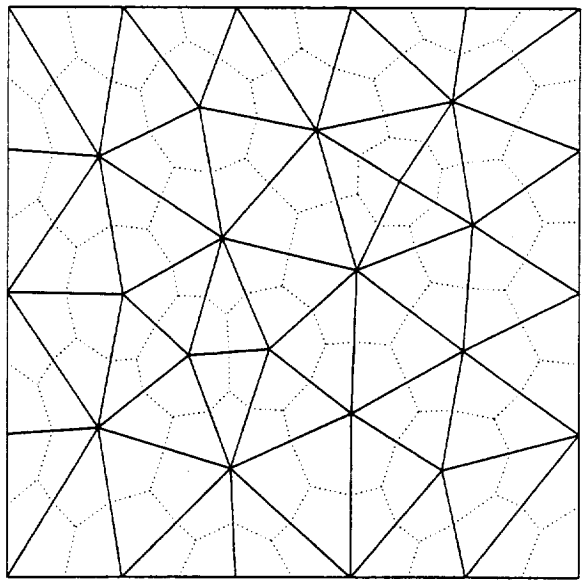

Figure 4.2 : Dual Mesh Associated with a Typical Unstructured Triangulation

4.1.3.2 Discretization of the Euler Equations

With the computational domain discretized as explained in Section 4.1.3.1, integrating (4.2) over the individual control-volume cells in a reference ξ space yields

$$\int_{C_i(0)} \frac{\partial JW}{\partial t}|_\xi \, d\Omega_\xi + \int_{C_i(0)} \nabla \cdot \vec{\mathcal{F}}^c(W, \dot{\vec{x}}) J \, d\Omega_\xi = 0 \qquad (4.3)$$

Since the partial time derivative is evaluated at a constant value of ξ, it is moved outside the integral sign in (4.3)

$$\frac{d}{dt}\int_{C_i(0)} JW\, d\Omega_\xi + \int_{C_i(0)} \nabla\cdot\vec{\mathcal{F}}^c(W,\dot{\vec{x}})J\, d\Omega_\xi = 0$$

Switching from the ξ reference space to the x space at time t,

$$\frac{d}{dt}\int_{C_i(t)} W\, d\Omega_x + \int_{C_i(t)} \nabla\cdot\vec{\mathcal{F}}^c(W,\dot{\vec{x}})\, d\Omega_x = 0$$

Integrating the last of the above equations by parts yields

$$\frac{d}{dt}\int_{C_i(t)} W\, d\Omega_x + \int_{\partial C_i(t)} \vec{\mathcal{F}}^c(W,\dot{\vec{x}})\cdot\vec{n}\, d\sigma = 0 \tag{4.4}$$

where $\partial C_i(t)$ is the cell boundary.

4.1.3.3 Approximation of the Convective Fluxes

The integral of the convective fluxes in (4.4) is approximated using a Riemann solver based on Roe's approximate Riemann solver [57], while the Steger-Warming flux decomposition [60] is employed for the far-field boundaries :

$$\begin{aligned}\int_{\partial C_i(t)} \vec{\mathcal{F}}^c(W,\dot{\vec{x}})\cdot\vec{n}\, d\sigma = &\sum_{j\in K(i)} \Phi^{\mathrm{Roe}}(W_i,W_j,\vec{n}_{ij},\sigma_{ij}) \\ &+ \Phi^{\mathrm{SW}}(W_i,W_\infty,\vec{n}_{i\infty},\sigma_{i\infty})\end{aligned}$$

Here Φ^{Roe} and Φ^{SW} denote the numerical fluxes of Roe and Steger-Warming, respectively, Γ_∞ is the far-field boundary and W_∞ is the vector of physical variables associated with the far-field uniform flow. $\vec{n}_{ij}$ and $\vec{n}_{i\infty}$ are integrals of the normal to the cell boundary defined as

$$\begin{aligned}\vec{n}_{ij} &= \int_{\partial C_i(t)\cap\partial C_j(t)} \vec{n}\, d\sigma \\ \vec{n}_{i\infty} &= \int_{\partial C_i(t)\cap\Gamma_\infty} \vec{n}\, d\sigma \\ \sigma_{ij} &= \frac{1}{||\vec{n}_{ij}||}\int_{\partial C_i(t)\cap\partial C_j(t)} \dot{\vec{x}}\cdot\vec{n}\, d\sigma \\ \sigma_{i\infty} &= \frac{1}{||\vec{n}_{i\infty}||}\int_{\partial C_i(t)\cap\Gamma_\infty} \dot{\vec{x}}\cdot\vec{n}\, d\sigma\end{aligned}$$

Roe's numerical flux for first-order spatial accuracy is defined by

$$\Phi^{\text{Roe}}(U,V,\vec{\nu},\sigma) = \frac{1}{2}\left[\mathcal{F}(\vec{\mathcal{U}})\cdot\vec{\nu} - \sigma||\vec{\nu}||U + \mathcal{F}(\vec{\mathcal{V}})\cdot\vec{\nu} - \sigma||\vec{\nu}||V\right] - \left|\tilde{A}(U,V,\vec{\nu}) - \sigma||\vec{\nu}||I_d\right|\left(\frac{V-U}{2}\right) \tag{4.5}$$

where $\tilde{A}(U,V,\vec{\nu})$ is Roe's mean value of the flux Jacobian matrix $\dfrac{\partial\vec{\mathcal{F}}}{\partial W}$.

4.1.3.4 Extension to Second-Order Accuracy

As mentioned above, the upwind numerical integration scheme is only first-order accurate. In order to achieve second-order accuracy, an extension of Van Leer's MUSCL [66] method is developed for unstructured meshes.

Based on the spatial approximation used in this method, the gradient of any function is constant over each cell of the mesh. In the MUSCL approach, second-order accuracy is achieved by extrapolating the values of W_i and W_j at the cell interfaces $\partial C_i \cap \partial C_j$ to get W_{ij} and W_{ji} respectively given by

$$W_{ij} = W_i + \frac{1}{2}(\nabla W)_i^{\beta}\cdot\overrightarrow{S_iS_j}$$
$$W_{ji} = W_j - \frac{1}{2}(\nabla W)_j^{\beta}\cdot\overrightarrow{S_iS_j}$$

Here the approximate nodal gradients $(\nabla W)_{i,j}^{\beta}$ are obtained via a β-combination of centered and fully upwind gradients :

$$(\nabla W)_i^{\beta} = (1-\beta)(\nabla W)_i^{\text{CENT}} + \beta(\nabla W)_i^{\text{UPW}}$$

The centered gradient $(\nabla W)_i^{\text{CENT}} = (\nabla W)_{i,j}^{\beta=0}$ can be chosen as any vector satisfying

$$(\nabla W)_i^{\text{CENT}}\cdot\overrightarrow{S_iS_j} = W_j - W_i$$

To compute the upwind gradient, note that $(\nabla W)_i^{\text{UPW}} = (\nabla W)_{i,j}^{\beta=1}$. Then it follows that

$$(\nabla W)_i^{\text{UPW}} = 2(\nabla W)_i^{\beta=\frac{1}{2}} - (\nabla W)_i^{\text{CENT}}$$

The half upwind gradients ($\beta = 1/2$) are computed via a linear interpolation of the Galerkin gradients computed in each triangle of C_i so that

$$(\nabla W)_i^{\beta=1/2} = \frac{\iint_{C_i} \nabla W|_{\triangle}\, dxdy}{\iint_{C_i} dxdy}$$

$$= \frac{1}{\text{area}(C_i)} \sum_{\triangle \subset C_i} \frac{\text{area}(\triangle)}{3} \sum_{k=1, k\subset\triangle}^{3} W^k \nabla N_k^{\triangle}$$

where $N_k^{\triangle}$ is the P1 shape function associated with node k of triangle $\triangle$.

To damp or eliminate spurious oscillations which may occur in the vicinity of discontinuities, a slope limitation procedure is enforced. First, the fictitious states W_{ij}^* and W_{ji}^* are defined as

$$W_{ij}^* = W_i - 2(\nabla W)_i \cdot \overrightarrow{S_i S_j} + (W_j - W_i)$$

$$W_{ji}^* = W_j - 2(\nabla W)_j \cdot \overrightarrow{S_j S_i} + (W_i - W_j)$$

The slopes are obtained via a van Albada [65] average :

$$dW_{ij} = \text{Ave}(W_j - W_i, W_i - W_{ij}^*)$$

$$dW_{ji} = \text{Ave}(W_i - W_j, W_j - W_{ji}^*)$$

For two scalars a and b, the van Albada average is given by

$$\text{Ave}(a,b) = \begin{cases} \dfrac{a(b^2+\epsilon^2) + b(a^2+\epsilon^2)}{a^2+b^2+\epsilon^2} & \text{if } a \cdot b > 0 \\ 0 & \text{otherwise} \end{cases}$$

In the above expression, ϵ is a small number introduced to avoid a zero denominator.

The new extrapolated values for the flux function Φ^{Roe} are computed by

$$W_{ij} = W_i + \frac{1}{2} dW_{ij}$$

$$W_{ji} = W_j + \frac{1}{2} dW_{ji}$$

4.1.4 Time Integration and the Geometric Conservation Law

The resulting semi-discrete form of (4.4) is

$$\frac{d}{dt}(A_i W_i) + F_i(W, \vec{x}, \dot{\vec{x}}) = 0 \tag{4.6}$$

where $A_i = \int_{C_i(t)} d\Omega_x$, W_i denotes the average value of W over the cell $C_i(t)$, $\vec{x}$ is the vector of time-dependent mesh point positions, W is the vector formed by the collection of W_i over all the mesh points and F_i are the ALE discretized convective fluxes.

Let $t = n\Delta t$ be the time at the n^{th} time step. The integration of (4.6) leads to

$$A_i^{n+1} W_i^{n+1} - A_i^n W_i^n + \int_{t^n}^{t^{n+1}} F_i(W, \vec{x}, \dot{\vec{x}})\, dt = 0 \tag{4.7}$$

The question now arises as to where the convective fluxes are to be evaluated at each step. Choices include the initial configuration $(t^n, \vec{x}^n)$ or the final configuration $(t^{n+1}, \vec{x}^{n+1})$ or an intermediate one between these two.

The same ambiguity also arises for the velocity computation, $\dot{\vec{x}}$. It has been shown by Lesoinne and Farhat [42] that any proposed method chosen to integrate the fluxes must satisfy the condition that the state of uniform flow must be preserved under arbitrary mesh motion. This requires that the change in area of each control volume between steps t^n and t^{n+1} should be equal to the area swept by the cell boundary during time $\Delta t^n = t^{n+1} - t^n$. Laws of this type are generally referred to in literature as Geometric Conservation Laws (GCL). The GCL derived in [42] stipulates that the flux integrand in (4.7) has to integrated at the mid-point configuration between

$(t^n, \vec{x}^n)$ and t^n and t^{n+1}.

$$\int_{t^n}^{t^{n+1}} F_i(W, \vec{x}, \dot{\vec{x}})\, dt = \Delta t F_i(W^n, \vec{x}^{n+\frac{1}{2}}, \dot{\vec{x}}^{n+\frac{1}{2}})$$

$$\vec{x}^{n+\frac{1}{2}} = \frac{\vec{x}^n + \vec{x}^{n+1}}{2}$$

$$\dot{\vec{x}}^{n+\frac{1}{2}} = \frac{\vec{x}^n + \vec{x}^{n+1}}{\Delta t^n}$$

Once the fluxes are computed, (4.6) is integrated in time using a 3-step variant of the Runge-Kutta method [22, 38].

4.1.5 Implementation of Boundary Conditions

The present implementation handles three types of boundary conditions : (i) the slipping boundary condition at the solid boundary, (ii) the inflow and outflow boundary conditions at the far-field boundaries and (iii) a periodic boundary condition for rotor-stator interaction calculations. Implementation of each of these boundary conditions is described next.

4.1.5.1 Slip Boundary Condition

The slip boundary condition ($\vec{U} \cdot \vec{n} = \dot{\vec{x}} \cdot \vec{n}$) is prescribed at the solid boundary Γ_b. This is imposed weakly by modifying the fluxes appropriately and it can be shown that at the wall boundary

$$\int_{\partial C_i(t) \cap \Gamma_b(t)} \vec{\mathcal{F}}^c(W, \dot{\vec{x}}) \cdot \vec{n}\, d\sigma = \left\{ \begin{array}{c} 0 \\ p_i n_{i\Gamma_{b_x}} \\ p_i n_{i\Gamma_{b_y}} \\ p_i ||\vec{n}_{i\Gamma_b}|| \sigma_{i\Gamma_b} \end{array} \right\}$$

with

$$\vec{n}_{i\Gamma_b} = \int_{\partial C_i(t) \cap \Gamma_b(t)} \vec{n}\, d\sigma \quad \text{and} \quad \sigma_{i\Gamma_b} = \frac{1}{||\vec{n}_{i\Gamma_b}||} \int_{\partial C_i(t) \cap \Gamma_b(t)} \dot{\vec{x}} \cdot \vec{n}$$

4.1.5.2 Inflow and Outflow Boundary Conditions

At the inflow and outflow boundaries, a precise set of compatible exterior data values that depend upon the flow regime and velocity direction need to be specified. The flux integral from (4.6) is evaluated at the far-field boundaries (Γ_f) using a non-reflective version of the flux-splitting scheme of Steger and Warming [60] :

$$\int_{\partial C_i(t)\cap\Gamma_f} \vec{\mathcal{F}}^c(W,\dot{\vec{x}})\cdot\vec{n}\,d\sigma = A^{c+}(W_i,\vec{n},\sigma)\cdot W_i + A^{c-}(W_i,\vec{n},\sigma)\cdot W_f$$

where

$$A^c(W,\vec{n},\sigma) = \frac{\partial\vec{\mathcal{F}}(U)}{\partial W}\cdot\vec{n} - \sigma||\vec{n}||I_d$$

and W_f is the vector of state variables corresponding to the exterior flow regime.

4.1.5.3 Periodic Boundary Condition

Another type of boundary condition incorporated into the present implementation is the periodic boundary condition used typically in computations such as rotor-stator interaction simulations, where only one or a few airfoils are employed to model an entire stage. An assumption made for this purpose is that the boundaries of the mesh on which periodic boundary conditions are imposed have to be identical. This assumption is not that restrictive as it would otherwise be not possible to have a periodic flow on a non-periodic geometry.

For example, consider Figure 4.3 in which periodic boundary conditions are applied along the interfaces AA' and BB'. Interface AA' is chosen or defined as the *master* interface, i.e., the interface along which the fluxes and variables will actually be computed, as opposed to interface BB' which

is defined as the *slave* interface where values of new variables are obtained from the corresponding mesh points on AA'. In order for this mapping of variables to be possible, the placement of mesh points on AA' has to be exactly identical to that on BB'.

To compute the fluxes at mesh points along the master interface, it is essential to know not only the values of variables at all mesh points connected to the point under consideration, but also the gradients which are required for second order spatial accuracy. On account of this, it becomes necessary to consider all mesh points connected up to two levels to the point on the master boundary.

In the implementation of the periodic boundary condition, ghost triangles are constructed on both the master and slave interfaces as seen in Figure 4.4. In this, the shaded triangles are attached to the master interface, lying 'inside' towards the slave interface. These are exactly duplicated on the slave interface. Likewise, triangles connected to the slave interface lying towards the master interface are duplicated on the master interface. It should be noted that triangles are duplicated up to only a single level of connection at the slave interface but up to two levels of connection at the master interface. At duplicated mesh points, variables are obtained from the corresponding original points. This allows the exact computation of both gradients and fluxes.

4.2 Summary

This chapter reviewed a finite-volume approach to Euler flow computations on unstructured triangular meshes. Details for spatial discretization, computation of convective fluxes, implementation of boundary conditions, time integration and the geometric conservation law were given.

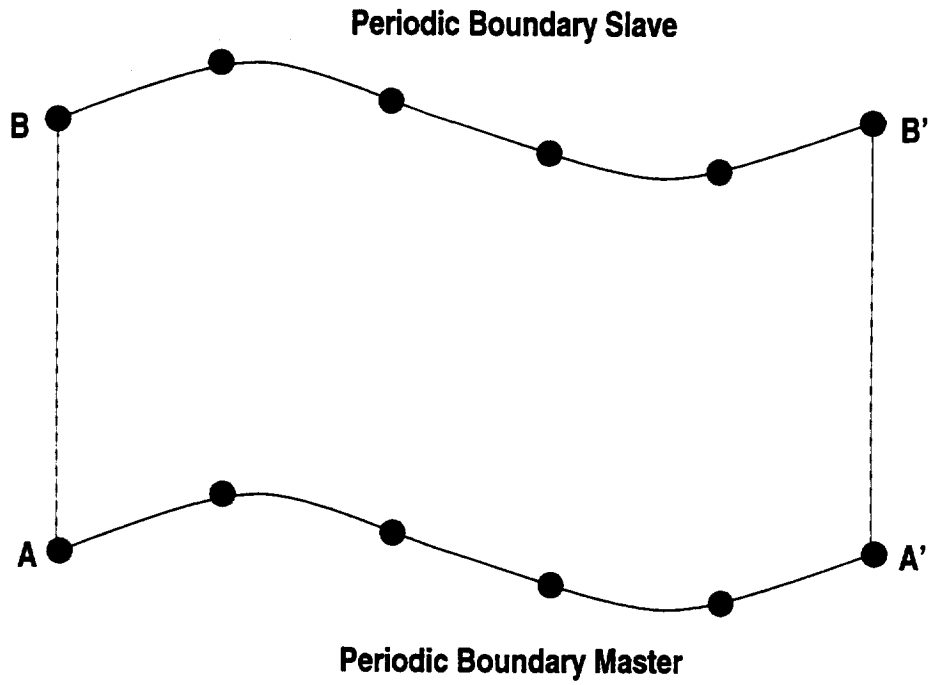

Figure 4.3 : Mesh Requirements for Periodic Boundary Conditions

This concludes the description of a finite-volume approach to performing Euler flow computations on unstructured meshes. The next section examines the main challenges involved in extending the capabilities of the current method to discontinuous unstructured meshes.

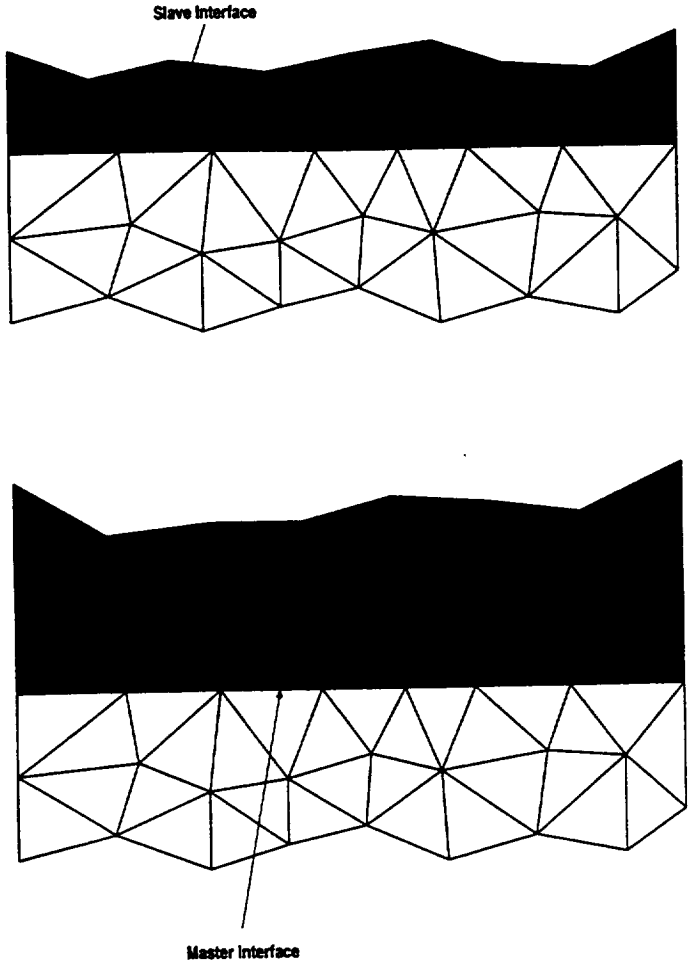

Figure 4.4 : Implementation of the Periodic Boundary Condition

Chapter 5

A Review of Flow Computations on Non-Matching Meshes

As mentioned in earlier chapters, the main research objective is to develop a method to perform flow computations on non-matching unstructured meshes. This chapter discusses the motivation for such problems and reviews recent efforts taken in this direction, highlighting their merits and drawbacks.

5.1 Rotor-Stator Interaction in Turbomachinery

The compressor and turbine components of an aircraft engine are usually comprised of several successive *stages*, each stage being made up of a rotating component, the *rotor*, and a non-rotating component, the *stator*. The function of the rotor is to add energy to the flow by mechanical interaction of the fluid with the blades, during which the fluid acquires angular momentum. The stator removes this angular momentum and diffuses flow to raise pressure. This combined action of the rotor and the stator is of fundamental importance to the performance and efficiency of the engine and hence is a matter of key research interest. An engine usually contains several such rotor-stator stages in succession and hence the ability to analyze such stages forms the first building block in an attempt to simulate a whole aircraft engine.

From an aerodynamic point of view, an understanding of the effect that an aft airfoil has on the flow in the region of the airfoil behind it is crucial in mainly two respects. The first, namely inviscid interaction between adjacent airfoils and second, interaction on account of viscous effects [18]. Gradients due to inviscid flow have effects on both the upstream and the downstream sides of the airfoil up to a length scale equal to the pitch or chord of the cascade. This can cause unsteadiness in flows both in upstream and downstream airfoils if the axial spacing between them is less than approximately the airfoil chord. Viscous effects, on the other hand, only affect downstream blades by creating wakes in the flow field. However, disturbances caused by wakes are stronger and do not decay as fast as those caused by inviscid flow and can be felt several chords downstream and even in cases where the airfoils are spaced far apart. In most cases, both inviscid and viscous effect are equally dominant and occur simultaneously.

As mentioned in Chapter 1, interaction between adjacent airfoils results in forced resonant vibration, which can be of importance from an aeroelastic perspective.

5.2 Issues in Computer Analysis of Rotor-Stator Interaction

The main hurdle to overcome in performing computer analysis of rotor-stator interaction is the need to compute flows across mesh interfaces when meshes no longer remain continuous.

A method to perform Euler flow computations on unstructured triangular meshes was described in Chapter 4. Finite-volume cells are constructed around each node by joining the medians of triangles to their centroids as shown in Figures 4.1 and 4.2. Fluxes are computed by constructing Riemann problems at the interfaces of adjacent cells which are processed by

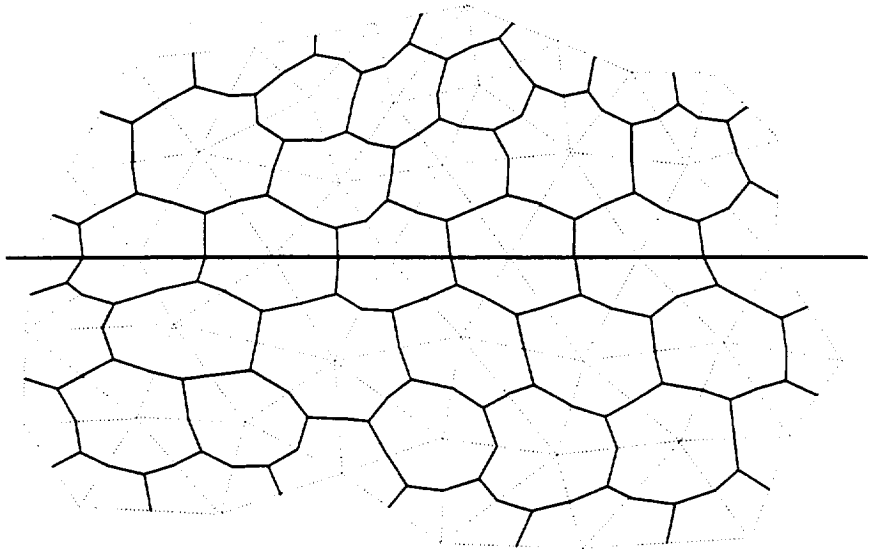

Figure 5.1 : Finite-Volume Cells on Matching Meshes

using Roe's approximate Riemann solver. Difficulties that arise on account of the choice of spatial discretization in the current method and the challenges that they pose while extending the present method to the case of non-matching unstructured meshes will be examined in this section.

5.2.1 Finite-Volume Cells on Matching and Non-Matching Unstructured Triangular Meshes

Consider two unstructured triangular meshes aligned at an interface as shown in Figure 5.1. The bold horizontal line demarcates the interface between these meshes while the dotted lines represent the mesh triangles. Finite-volume cells constructed around mesh points are shown by solid lines. In this figure, it is observed that cell boundaries on both sides of the interface are continuous, and fluxes for points lying on the interface can be easily computed by adding fluxes computed on each of the half-cells.

Next, consider the situation when the mesh on the bottom is given some displacement to the right as shown in Figure 5.2. In this case, the cell

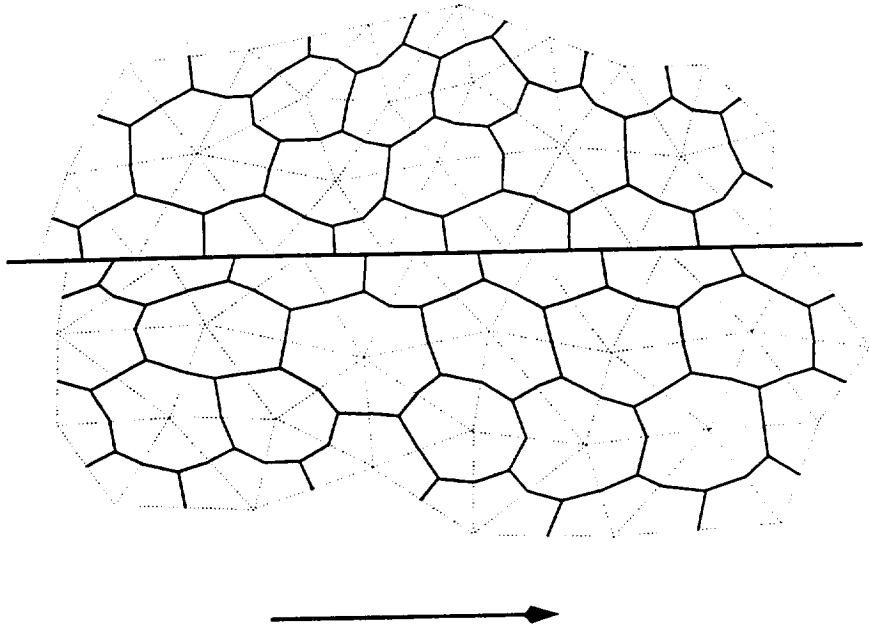

Figure 5.2 : Finite-Volume Cells on Non-Matching Meshes

boundaries are no longer aligned and computation of fluxes at points lying on the mesh interface becomes non-trivial. This relative sliding of mesh cells along the interface represents the key difficulty in the development of a method to treat non-conforming unstructured meshes. An example of slipping meshes and arising mesh discontinuity can be seen in Figure 5.3.

5.2.2 Goals for Current Research

The focus of current research is to develop a method to enable flow computations to be performed on unstructured triangular meshes even when meshes are no longer continuous. While motivations for such application are numerous, stress here will be laid mainly on analyzing multistage turbomachinery flows such as in rotors and stators.

Other cases of interest in turbomachinery applications where such mismatch could occur would be interaction of flow and mechanical components between non-rotating components such as inlets and diffusers with a

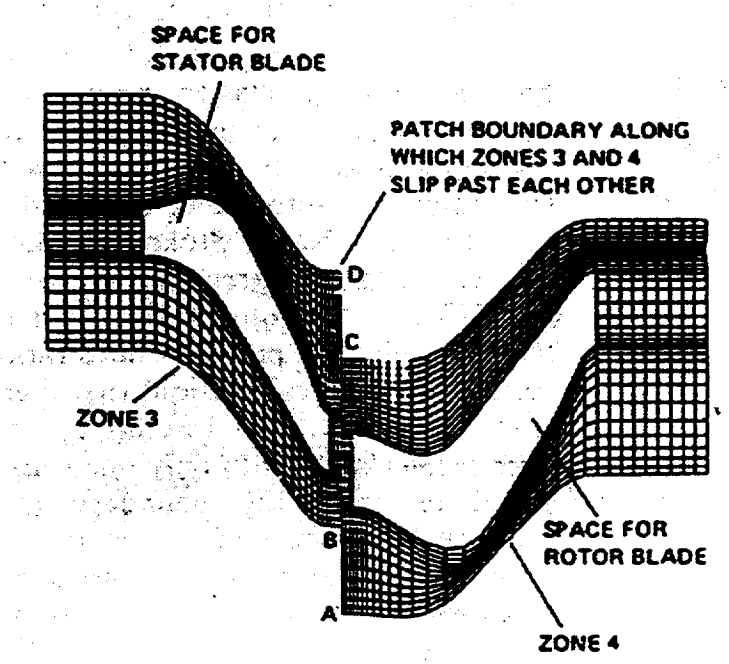

Figure 5.3 : Slipping Meshes in Rotor-Stator Interaction

rotating component such as a large fan. Relative motion of meshes would also have to be accounted for in the motion of control surfaces on aircraft wings and also in store separation simulations.

The method developed should satisfy the following requirements :

1. **Accuracy** : The method should preserve the accuracy of the present finite-volume scheme without introducing any additional sources of error on account of mesh mismatch.
2. **Computational Efficiency** : The long-term goal of current research is to carry out high-performance three dimensional analysis of aircraft engine systems using massively parallel processing. The method should be amenable to parallelization. Thus, the schemes should favor locality to reduce communication overhead as much as possible.
3. **Computational Tractability** : Several solutions have been sug-

gested for the problem at hand. However, some of them have major drawbacks in the sense that they are either not parallelizeable or are so involved in nature that programming them for complex geometries becomes almost impossible. The method developed should be computationally simple and easy to program, making extensions to more complicated and three dimensional cases possible.

5.3 Review of Earlier Work

Several solution strategies have been proposed to the problem of having non-matching meshes for rotor-stator interaction. These can be grouped into the following two categories :

1. Dynamic remeshing at the interface between rotating and non-rotating components.
2. A multiple mesh or zonal approach in which meshes are generated separately for each of the rotating and non-rotating components and mesh mismatch is handled by transferring and exchanging information from one mesh to another.

Each of these approaches will be described next.

5.3.1 Dynamic Meshes

A natural approach to deal with non-matching meshes is to perform remeshing at the interface whenever mesh triangles become too distorted. Giles [29] employed this to perform rotor-stator interaction for turbine blades. Referring to Figure 5.4, meshes attached to the stator and the rotor are constructed separately and are kept disjoint. A set of fictitious cells or glue

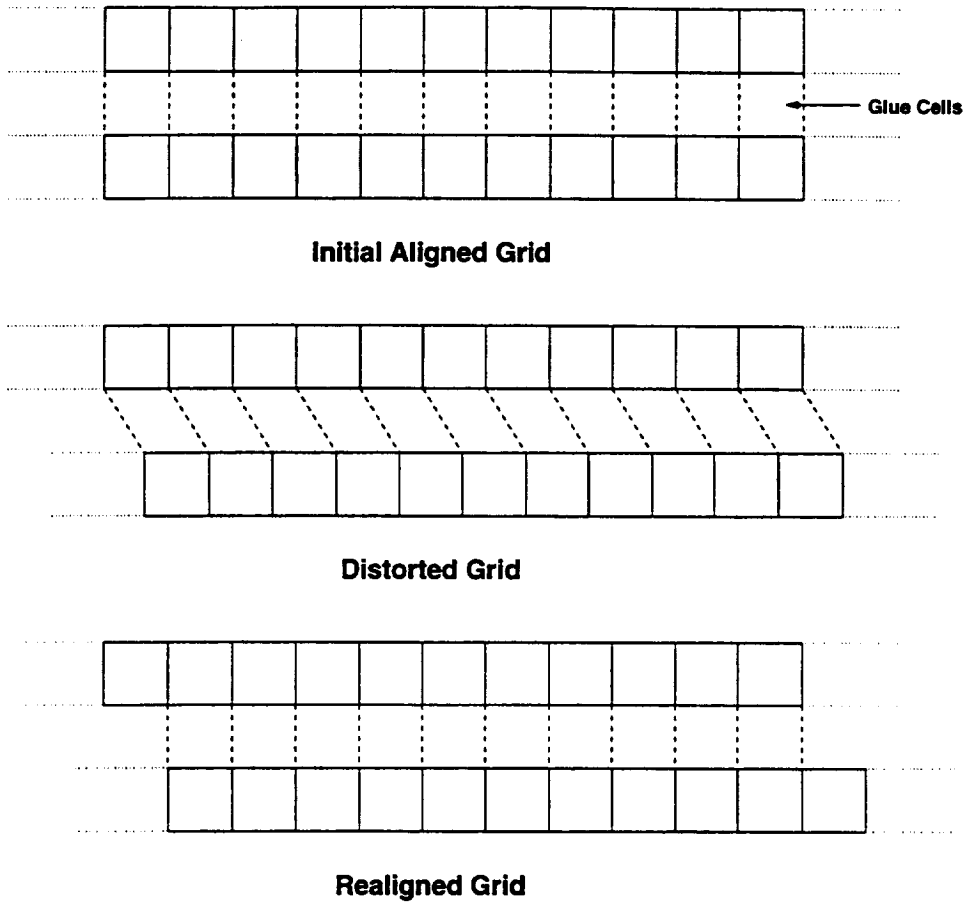

Figure 5.4 : Dynamic Remeshing of Fluid Meshes on the Rotor/Stator Interface

cells is constructed at the interface between the individual meshes, so as to connect corresponding points on either side of the interface.

Once mesh motion is initiated (in this case, by moving the lower mesh to the right), the glue cells become distorted as seen in Figure 5.4. After a limiting value of distortion is reached, a new set of glue cells is constructed so that the moving meshes are either completely realigned or their degree of distortion is minimized.

This idea is conceptually simple, however, it requires identical spacing of mesh points on the mesh interface, which is generally impossible in the case of unstructured meshes. An extension of this method to unstructured

meshes would require multiple-layer remeshing after every few time steps, which hinders parallelization. This problem would be exacerbated in three dimensions.

5.3.2 Multiple Patched Meshes or Zonal Approach to RSI

To obviate the problem of remeshing and having specific mesh requirements, several examples can be found in literature in which mesh attached to the rotor and the stator are *patched* at the interface and allowed to slide freely. Flow computations are made possible by an exchange of information between individual meshes at each time step.

Rotor-stator interaction is one example in which multiple patched meshes are used. Other instances are domain decomposition for parallel processing, Chimera or overset grids, localized mesh refinement and coupling of solutions obtained by different methods on different parts of the mesh. Of these, domain decomposition and Chimera or overset grids are more relevant to the present problem and will be briefly explained next for future reference.

5.3.2.1 Domain Decomposition

Within the context of parallel processing for large scale physical problems, it is common to divide the discretized computational domain (like a mesh or grid) into smaller subdomains and assign each subdomain to a processor of the parallel machine. Each processor performs computations on the individual subdomain assigned to it, treating it as a separate problem with its own boundary conditions. The key issue in this case becomes the development of appropriate boundary conditions at the interface between two or more adjacent subdomains and the exchange of information. The problem at hand is similar except that there is the additional relaxation of the requirement of the mesh lines to be continuous.

5.3.2.2 Chimera or Overset Grids

Development of Chimera or overset grids took place to allow the construction of structured grids in complex two dimensional and three dimensional geometries by overlapping blocks of body-fitted structured grids.

For example, a major grid is generated around a main body element and minor grids are then overset on top of the major grid with a common area of overlap in which solutions are matched across grid interfaces. The manner in which grids overlap can be quite arbitrary without requiring the grid boundaries to join in any special way.

As in the case of domain decomposition, it becomes important to impose the right boundary conditions on the boundaries of the individual meshes and transfer information from one mesh to another.

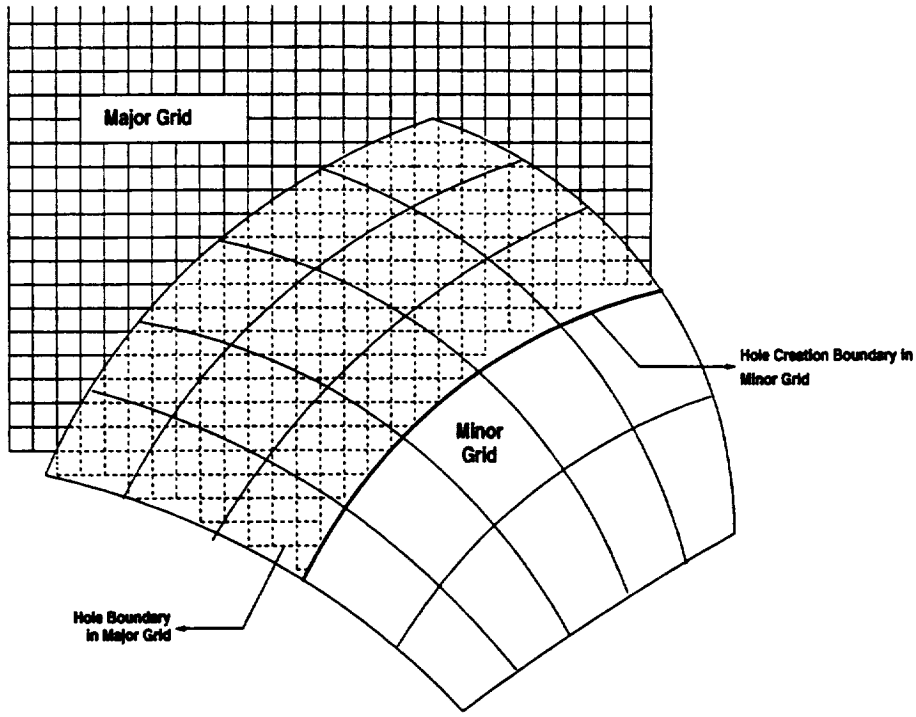

Figure 5.5 : Example of Chimera Grid Construction

As mentioned above, the central issue in the use of multiple meshes, whether for domain decomposition or in Chimera or overset grids is the application of appropriate boundary conditions at the grid interfaces and the transfer of information from one computational domain to another. Methods developed for such applications differ primararily in the way this information exchange takes place and can be broadly classified into the following two categories :

1. Exchange of information occurs by *conservatively* transferring fluxes from one mesh to another. These will henceforth be referred to as *flux conservative* or simply *conservative* methods. The exact meaning and significance of conservation will be explained in a later section.
2. Another approach is to exchange physical variables between meshes as opposed to fluxes which are thus no longer conserved. Such methods are usually referred to as *non-conservative.*

The concept of flux conservation and its significance to computational fluid dynamics will be discussed next.

5.3.3 Flux Conservation

On account of the highly non-linear nature of the Euler and compressible Navier-Stokes equations, it is a well known fact that discontinuities may develop in the solution even if smooth initial conditions have been given. Such solutions that violate the smoothness requirements of the differential form at a discontinuity manifold, but which satisfy the integral form everywhere are called *weak* solutions. Because of the non-linearity of the governing equations, multiple weak solutions may be mathematically possible. A solution which is physically relevant can be selected through criteria proposed by

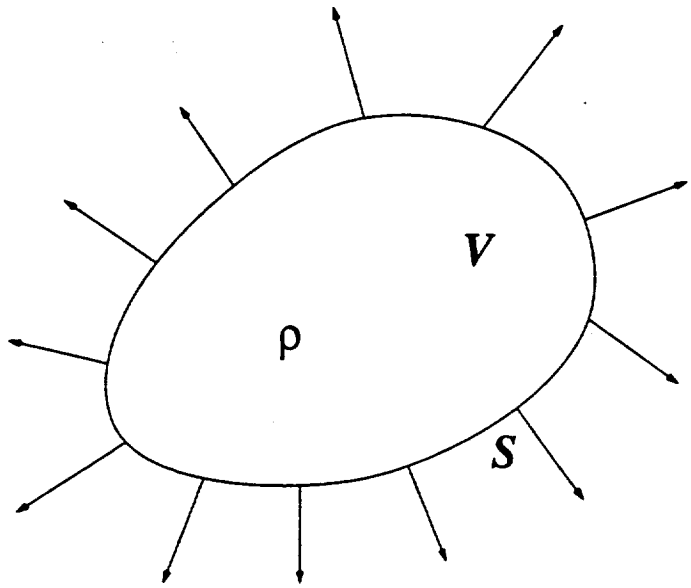

Figure 5.6 : Flux conservation for a Control Volume

Lax [39], namely conservation and the entropy condition. This condition and its significance will not be discussed here and the reader is referred to [39, 43] for details.

As its name implies, the basic principle behind a conservation law is that the total quantity of the conserved variables in any region changes only due to the flux through its boundaries. To clarify this, consider the conservation of mass or the continuity equation of the Euler equations :

$$\frac{\partial}{\partial t}\iiint_{\mathrm{V}} \rho \, \mathrm{dV} + \iint_{S} \rho \, \mathbf{V} \cdot \mathbf{dS} = 0 \tag{5.1}$$

Referring to Figure 5.6, this implies that the time rate of change of mass inside the control volume V is equal to the net flow of mass in or out of the control volume (i.e., the flux) through surface S. Hence, to prevent an unnatural creation or destruction of the quantity ρ by a numerical scheme, it is essential that the flux be computed correctly. This is commonly referred to in literature as flux conservation. Failure to conserve fluxes in numerical

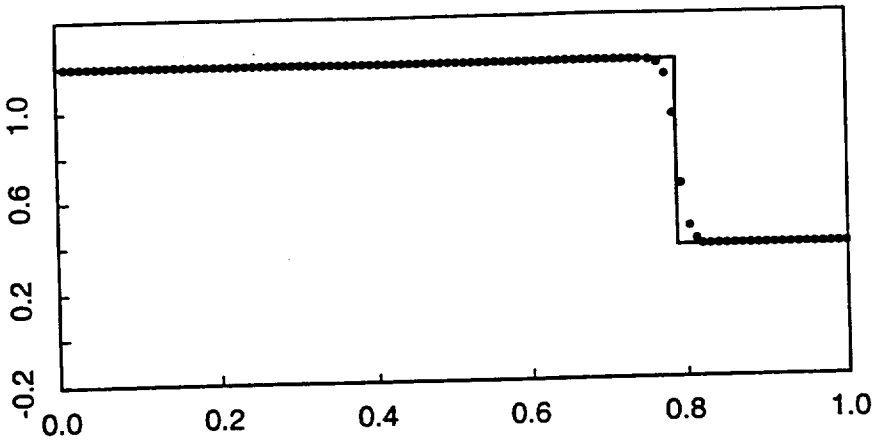

Figure 5.7 : True and Computed Solutions to the Burgers' Equation using a Conservative Method

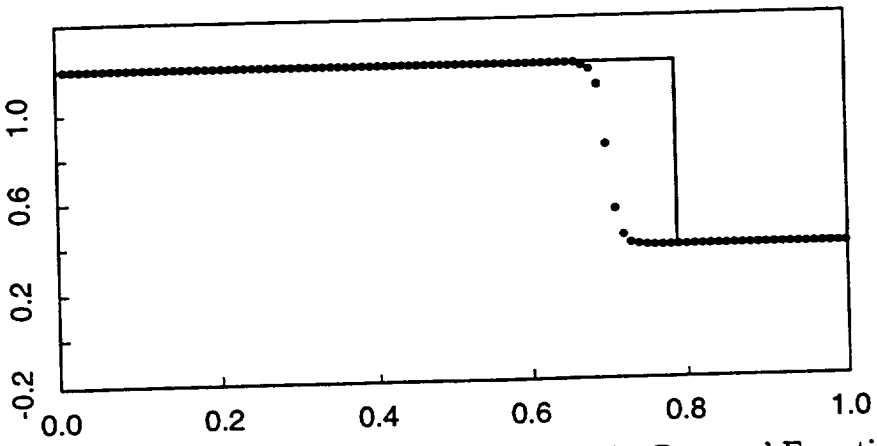

Figure 5.8 : True and Computed Solutions to the Burgers' Equation using a Non-Conservative Method

schemes gives rise to artificial source terms for the governing equations, which in turn may lead to the incorrect positioning of shocks and discontinuities. This is illustrated in Figures 5.7 and 5.8 (from [43]) for the Burgers' equation $(u_t + uu_x = 0)$.

For the spatial discretization adopted in the present numerical scheme, this implies that the numerical flux through the cell boundaries is such that the total mass (and other conserved variables) remains constant in each cell.

Several flux conservative methods proposed for patched meshes are described below.

5.3.4 Flux Conservative Methods for Patched Meshes

Flux conservative methods have been developed for patched meshes, both in the context of rotor-stator interaction and overset meshes. Rai and co-workers [51, 53] have been credited with the development of a flux conservative method for patched structured grids for rotor-stator interaction applications. This was extended to the case of unstructured triangular meshes by Mathur and others [46, 47]. Berger [10] proposed a flux conservative method for overset grids, with particular applications to adaptive grid refinement. Wang [67] proposed a flux conservative scheme for overset grids which guaranteed conservation locally at each interface of grid overlap and thus globally for the overlap as a whole. This section will briefly go over the relevant details of these methods.

5.3.4.1 Rai's Conservative Approach for RSI

In the mid-1980s, Rai [44, 51, 52, 53, 54] developed a method for accurate and efficient computation of flows using patched grids for rotor-stator interaction. The key feature of this approach was the emphasis laid on the conservative treatment of the grid interface.

For the purpose of illustration, assume that only two grids are to be considered, having an interface as shown in Figure 5.9. In this figure, the solid lines are the grid lines for the individual grids, the bold solid line denotes the grid interface and the dotted lines indicates the cells constructed for finite-volume computation. In this example, fluxes have to be computed at a point O, lying on the grid interface. To obtain this flux correctly, Rai extends the grid of Zone 2 into Zone 1 such that a finite-volume cell like RSTU can be constructed. This cell is so constructed to allow its boundary

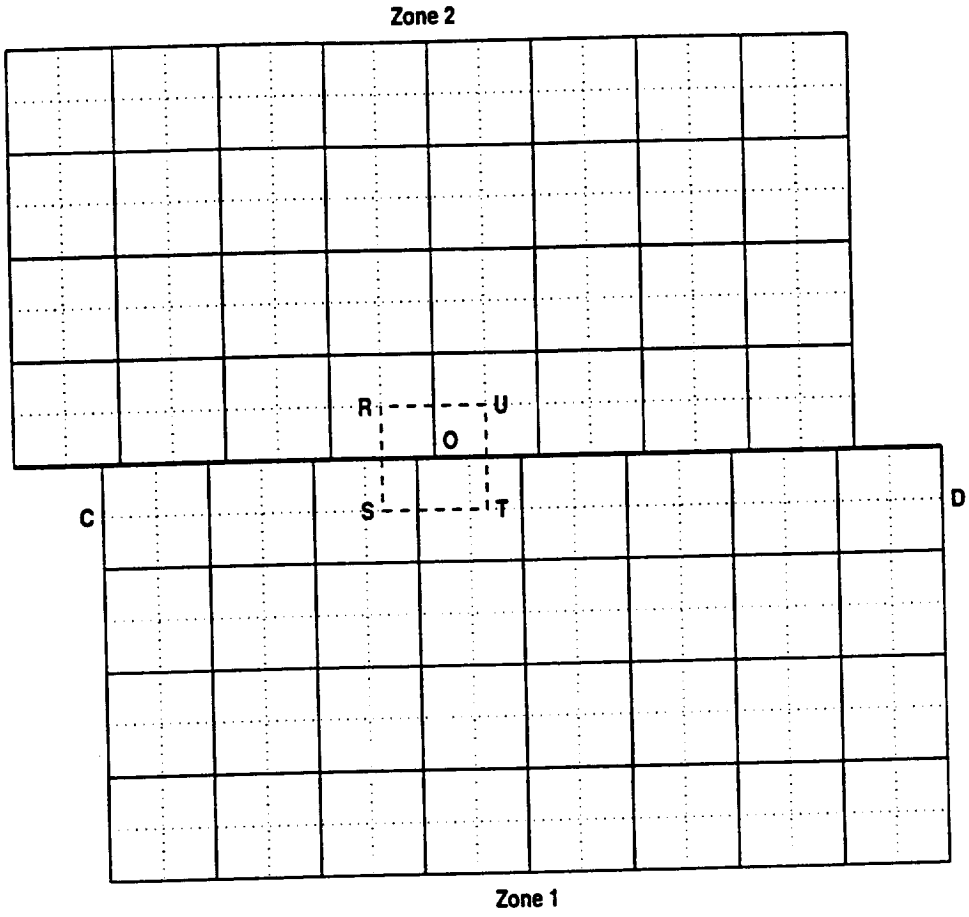

Figure 5.9 : Grid Extrapolation for Rai's Conservative Method

ST to align with boundary of the finite-volume cells of Zone 1, namely, line CD.

Fluxes through the boundary ST are obtained by a conservative interpolation of computed fluxes computed on line CD. Following a conservative flux interpolation procedure, fluxes through segments such as ST are obtained in such a way that the sum of these fluxes equals the sum of the total fluxes across CD. This interpolation procedure involves the determination of the location of segment ST on line CD and fractions of cell boundaries of line CD (volume weights) which contribute fluxes to ST. Details of this interpolation scheme may be found in [51]. In order to establish continuity

along the grid interface, values of variables on the interface for Zone 1 are obtained by linear interpolation of those values for Zone 2.

This method has been shown to work well on a variety of problems where non-matching grids arise, including rotor-stator interaction. However, the grid construction and extrapolation as required by this method are possible only for structured grids. Furthermore, the flux interpolation is valid only if fluxes are assumed to be constant in each finite-volume cell. This would not be the case for the second-order accurate treatment as described in Section 4.1.3.4.

5.3.4.2 Berger's Method for Overset Grids

Berger [10] proposed a method for conservative transfer of fluxes for overset grids. This method considers the region of overlap between grids and fluxes are transferred based on the manner in which grid cells of the individual grids overlap each other.

Consider two overlapping grids as shown in Figure 5.10. Assume that a cell-centered scheme is adopted for computations, that is, the grid lines themselves represent the cell boundaries and variables are computed for each cell (typically at its centroid) rather than at the grid vertices. Let the grid represented by the horizontal and vertical lines be the major or master grid. A cell from the minor or slave grid is shown by the oblique lines. To compute the fluxes for this minor cell, the method requires the determination of the extent of overlap (volume weight) this cell has with the cells from the major grids which it overlaps or intersects as shown by the areas A_1, A_2, A_3 and A_4 in Figure 5.10. This would guarantee flux conservation as the sum of fluxes transferred to the minor grid would be exactly equal to that in the region of overlap for the major grid.

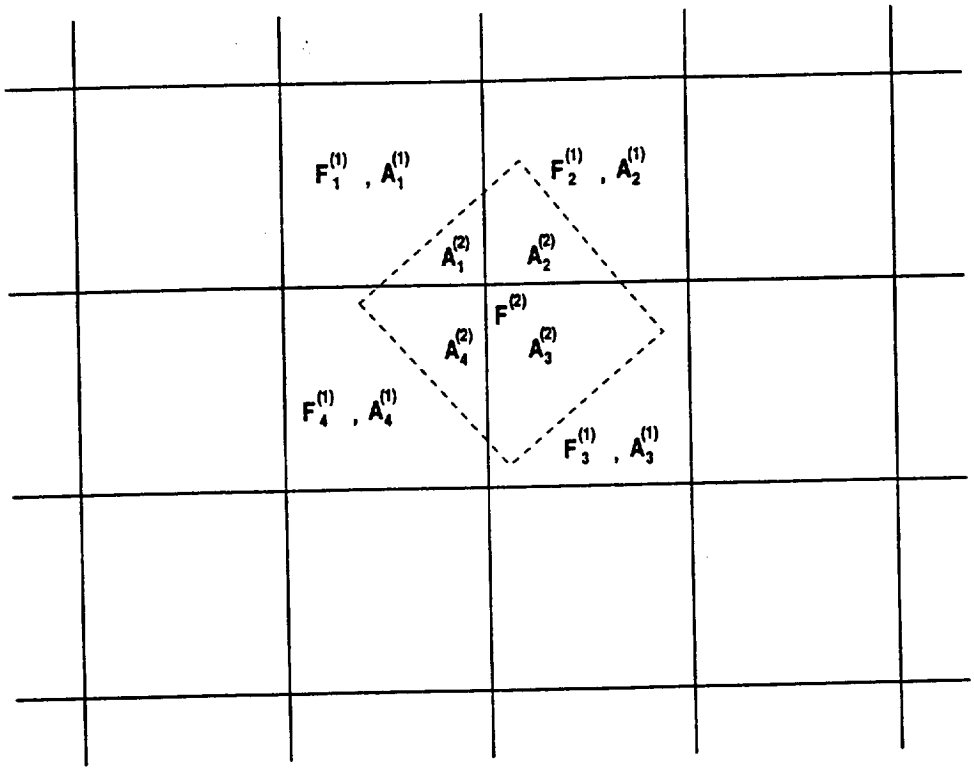

Figure 5.10 : Cell Intersection for Berger's Method

Although this method is fully conservative, it is complex and computationally intensive. Determining the volume weights for intersecting rectangular cells alone is a very difficult problem in computational geometry. An extension to cells like those obtained from the finite-volume construction described in Section 4.1.3.1 for triangular meshes and for cells obtained similarly for tetrahedral meshes would be even more cumbersome, if not impossible. Again, this method is valid only if fluxes are assumed constant over the finite-volume cells, which is not the case with the present spatial discretization method. However, one advantage of this method should be cited : its ability to couple solutions obtained by different discretization methods on different grids because fluxes are conserved on the basis of grid geometries alone.

5.3.4.3 Wang's Method for Overset Grids

Another flux conservative method for overset grids is that of Wang [67]. This method is similar to that of Rai (Section 5.3.4.1) except that it allows the grid overlap to take place arbitrarily.

In Figure 5.11, the horizontal and vertical lines mark the cell boundaries of the major grid while the oblique lines show the cell boundaries of the minor grid. For the sake of illustration, assume that the flux through line 1-4 is to be obtained.

Line 1-4 intersects the lines of the major grid at points 2 and 3 as shown. Clearly, the total flux through 1-4 would be equal to the sum of fluxes through the partial segments 1-2, 2-3 and 3-4. These fluxes are obtained in the following two-step procedure :

1. Reconstruction : Obtain the values of the physical variables on either side of the partial segments (such as 1-2) by interpolation of cell-centered values from the major and minor grids.
2. Riemann solution : The reconstruction step sets up Riemann problems on each of the partial segments which are then solved to obtain the fluxes.

This method is the most attractive amongst the ones seen so far for structured grids. It involves less geometrical computations than Berger's method (Section 5.3.4.2) and could be used in many applications.

5.3.4.4 Mathur's Conservative Method for Unstructured Meshes

Mathur and co-workers [46] developed a conservative scheme for unstructured meshes for applications in both rotor-stator interaction and adaptive mesh refinement.

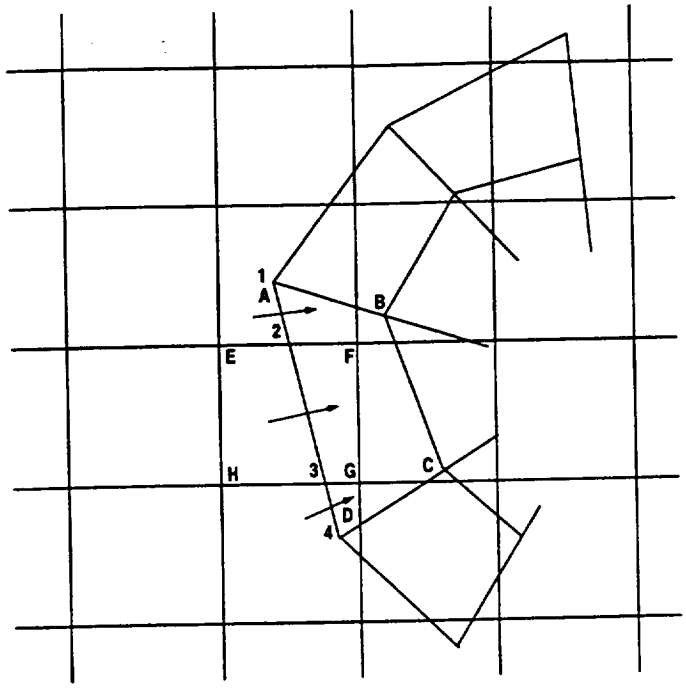

Figure 5.11 : Construction of Riemann Problems at Grid Interface for Wang's Method

The control volume around a point for the difference scheme is obtained by constructing cells formed by the union of the edges of all triangles to which the point belongs, excluding the edges which connect the point to other triangle vertices. Details of the resulting difference scheme can be obtained in [46]. Two typical cells on either side of the mesh interface are shown in Figure 5.12.

In order to compute conserved fluxes, partial fluxes at points such as A and D are first obtained by considering the contribution of fluxes obtained through their respective control volumes. In the next step, these partial fluxes are exchanged to and from the other side of the interface through the flux interpolation scheme described in Section 5.3.4.1. The total flux then is the sum of the flux computed through the control volume and the flux obtained from exchange with the other side. This facilitates both flux

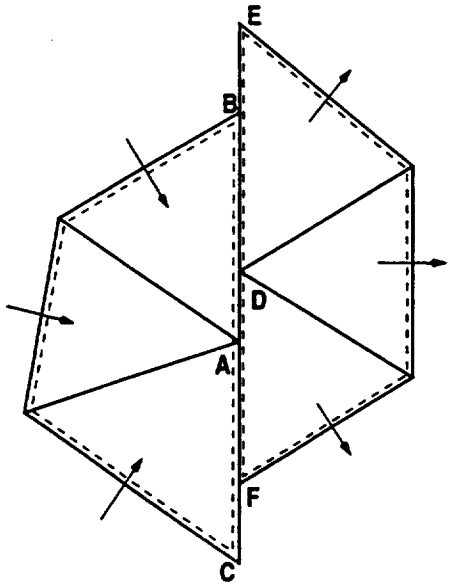

Figure 5.12 : Mesh Interface for Mathur's Method

conservation and independence of flux computation for each mesh.

This method is the only one available in literature for unstructured meshes. It is computationally efficient and allows computations to be done independently for each mesh, which facilitates parallelization. The only possible drawback is the loss in accuracy at the mesh interface by making the assumption of fluxes being constant within each control volume, even in the case of a higher order scheme such as in Section 4.1.3.4.

5.3.5 Comments on Flux Conservative Methods

The need for flux conservation between patched grids and methods which have been designed to do this have been discussed so far. However, it is felt that while explicit global flux conservation, if achieved, would render a method suitable for application to patched grids, it may be possible to get away with this requirement by careful usage of the so-called non-conservative schemes.

First, flux conservative methods do have drawbacks. For instance, it has been reported [49] that attempts to enforce flux conservation by methods such as those described above have resulted in the creation of instabilities. Secondly, from a computational point of view, almost all flux conservative methods need to determine some kind of volume weights in the process of flux transfer. Methods reported so far have been either only for structured grids with rectangular cells or for cases where the flux is assumed to be constant in each cell. The latter assumption is also made in the lone case for unstructured meshes reported in Section 5.3.4.4.

It should be noted that the final goal of current work is to develop robust methods which can be implemented in both two and three dimensions. Determination of volume weights in three dimensions would be computationally intensive and very hard to implement. Therefore, another approach, similar to overset grids is followed. Prior to giving details about this, some light is shed on overset grids and the use of non-conservative schemes.

5.3.6 Overset Grids and Non-Conservative Schemes

Recent progress in parallel processing for CFD has given rise to interest in treatment of patched grids and the development of artificial boundary conditions at the grids' interface. Several flux conservative solutions have been proposed to these problems, some of which have been presented in Section 5.3.4. This section reviews non-conservative methods used typically in the context of overset or Chimera grids and particularly at their reported inability to capture the position of shocks and discontinuities correctly.

In an interesting review article, Keeling and others [31] show that if used correctly, non-conservative methods can give results comparable to

those of conservative methods with less computational expense. First, it is shown that it *is* possible to construct unique single valued solutions to the conservation laws in fluid dynamics using overlapping meshes. Details of this are lengthy and have been omitted here for brevity. Secondly, it is established that local convergence with proper continuity requirements suffices to ensure correct convergent solutions. This is briefly explained in the next subsection.

5.3.6.1 Discrete Conservation

Systems of conservation laws of interest to fluid dynamics, namely, the Euler equations, can be concisely written in the classical differential form as

$$\frac{\partial W}{\partial t} + \nabla \cdot F = 0 \tag{5.2}$$

Details regarding vector notation and initial and boundary conditions have been omitted in (5.2) to avoid clutter. Also, these follow from the equations in Section 4.1.1.

These equations can be discretized using a finite-volume scheme such as in Section 4.1 to yield the following semi-discrete form of (5.2) :

$$A_i[W_i^{n+1} - W_i^n] = -\Delta t \sum_{j \in k(i)} F_i(W_i, W_j) \tag{5.3}$$

Here, n denotes the current time step, Δt the current increment in physical time, A_i the area of the cells associated with node i, $k(i)$ the set of nodes connected to node i via triangle edges and F_i, the numerical approximation to the fluxes in (5.2). Now, in order to achieve discrete conservation, the required condition is that if the terms in (5.3), following Section 5.3.3, are summed over all cells, fluxes across opposite sides of interior cells should cancel each other in a telescopic fashion and only the fluxes on the boundary

of the domain remain. Thus if Ω is the discretized domain and $\partial\Omega$ is its boundary, then,

$$\sum_i A_i W_i^{n+1} = \sum_i A_i W_i^n - \Delta t \sum_i \sum_{j \in k(i), j \in \partial\Omega} F_i(W_i, W_j) \qquad (5.4)$$

It has been shown [13] that if the discrete conservation requirement is satisfied only approximately, such that any discrepancy vanishes with mesh refinement, then a correct convergent solution is obtained. Otherwise, the solution converges to that of (5.2) with a fictitious source term.

5.3.6.2 Discrete Conservation on Overset Grids

On overlapping grids, the discrete equation (5.3) for a cell lying in subdomain Ω_i can be written as

$$A_i^{(i)}[W_i^{n+1,(i)} - W_i^{n,(i)}] = -\Delta t \sum_{j \in k(i)} F_i^{(i)}(W_i^{(i)}, W_j^{(i)}) \qquad (5.5)$$

where superscript (i) indicates that all terms pertain to subdomain Ω_i. In addition to the usual physical initial and boundary conditions, (5.5) must be supplemented with another *artificial* boundary condition

$$W_i^{n,(i)} = (\tau_i W^{n,(j)})_i \qquad (5.6)$$

for grid points of Ω_i lying in the region of overlap in Ω_j. In (5.6), τ_i is the trace operator (which can be thought of a boundary operator on $W^{(j)}$ with a few additional definitions to account for discontinuities which fall in the realm of functional analysis). This means that the value of $W_i^{n,(i)}$ for nodes i lying on a line of overlap used in (5.5) to compute fluxes, must be such that it approximates the value of $W^{n,(j)}$ on the boundary of the overlap of Ω_i into

Ω_j. Computationally, τ_i is a transfer operator, usually involving some form of interpolation from Ω_j to Ω_i.

Another point to note is that the computation of fluxes may depend upon the values W in the overlap region which have to be accurately obtained for correct computation of fluxes. Once these values of W in the overlap region are obtained to the desired level of accuracy, fluxes across the physical interface between adjacent grids can be computed as in the case for interior grid points, and a condition similar to (5.4) results :

$$\sum_i A_i^{(i)} W_i^{n+1,(i)} = \sum_i A_i^{(i)} W_i^{n,(i)} - \Delta t \sum_i \sum_{j \in k(i), j \in \partial\Omega} F_i^{(i)}(W_i^{(i)}, W_j^{(i)}) \quad (5.7)$$

Thus, it is seen that the overset grid scheme is *piecewise* conservative for each individual subdomain. It has been suggested in [31], that such piecewise conservation with a proper treatment of artificial boundary conditions as mentioned above, would be sufficient to obtain correct convergent solutions on multiple overset domains. Overall discrete conservation would not be essential.

5.3.6.3 Further Remarks on Overset Grids

Recent developments have shown that overset grids can be used with confidence in CFD. At the same time, it has also been mentioned that the use of overset grids resulted in incorrect positioning of shocks and discontinuities, which is attributed to the non-conservative manner in which interpolation is performed. However, Keeling *et al* [31] emphasize that while overset grids are not a priori conservative, their reported inability to treat shocks and discontinuities properly is often blamed incorrectly on their non-conservative nature and several clarifying explanations are given. It has been noticed

that in many cases where claims are made about the dissatisfactory performance of non-conservative schemes, proper experimental techniques are often lacking leading to several sources for numerical error :

1. The performance of CFD methods depend crucially on the grids on which they are tested and robust methods are known to fail on grids that are inadequately refined or improperly positioned. Errors may thus arise if grids of largely varying refinement are patched and an overset approach is followed.
2. In some cases, an inadequate interpolation scheme is used, resulting in the creation of spurious waves which eventually pollute the solution.
3. Insufficient overlap extent can also give rise to incorrect flux computation. To ensure correct computation of fluxes, grids have to be overlapped consistently with the spatial discretization employed.

As seen above, great importance is placed on the need to obtain accurate values of variables at points of grid overlap. One suggested method to obtain these has been the use of an ENO (essentially non-oscillatory) interpolating scheme [49]. In the present work, another highly accurate projection scheme which is growing in popularity, namely the mortar method, has been used. This method is described in the next section.

5.4 The Mortar Element Method

The mortar element method is being increasingly used in domain decomposition for parallel processing of problems in the physical sciences. Originally developed to couple solutions obtained by different methods on adjacent meshes, it now finds applications in solid and fluid mechanics and is also a fertile area for research in applied mathematics. This section gives a brief

overview of the mortar method, starting with a historical background, mathematical statement, properties, and merits and demerits over other comparable methods.

5.4.1 Historical Background

The mortar element method was first developed for the purpose of coupling different discretizations on different subdomains, namely spectral and finite element methods [11]. On account of its remarkable properties, it has also been used for non-conforming meshes for elliptic problems [12, 34], in fluid mechanics [1] and also for sliding mesh applications with spectral methods [3]. In each case, it provides an efficient way to glue solutions on interfaces of non-conforming meshes.

Later on, the method has attracted attention in the field of domain decomposition for parallel processing mainly on account of its ability to provide optimal accuracy in the gluing process. Another application of interest has been localized mesh refinement [34] for elliptic and parabolic problems and also sliding meshes such as in the context of rotor-stator interaction. In these cases, the mortar element method provides an auxiliary set of equations to solve for conformity in the solution variables.

While earlier research has focused on the use of the mortar element method for non-overlapping domain decomposition, it has only recently found use in overlapping domain decomposition [12]. In this case, the mortar element method provides an optimally accurate way to interpolate or project values of variables from one mesh to another. This of crucial importance for robustness as seen in Section 5.3.6.3. It will be in this role that the method will be presented here.

5.4.2 Mathematical Background

As mentioned in the introduction to this section, the mortar element method has attracted the attention of mathematicians on account of its ability to provide optimal accuracy in gluing solutions at interfaces. This section presents theory of the mortar element method emphasizing its role as a projector.

Assume that a piecewise linear solution u_1 is known on an interface Γ_1 as seen in Figure 5.13. A piecewise solution u_2 is sought on the interface Γ_2. The end points of Γ_1 and Γ_2 match whereas interior points need not do so, as is illustrated in the figure. Although u_2 on Γ_2 can be obtained in a trivial manner by matching linear interpolation, this recovery would be only first-order accurate.

In the terminology of the mortar element method, Γ_1 is known as the master interface (on which the solution u_1 is known) and Γ_2 is known as the slave interface, the solution u_2 on which is obtained from u_1, u_2 being completely dependent on u_1.

The first step in the mortar element method consists of defining piecewise linear test (hat) functions (ϕ_i) on the slave interface Γ_2, such that

$$\phi_i(x) = \begin{cases} 1, & \text{if } x = x_i; \\ 0, & \text{otherwise.} \end{cases}$$

except on for $i = 1$ and $i = (n-1)$ in which case the test functions are defined such that there is zero slope at the end-points of the interface.

This being done, the mathematical statement of the mortar element method becomes :

Find u_2 on Γ_2 such that

$$\int_\Gamma \phi(u_1 - u_2)\, d\Gamma = 0 \tag{5.8}$$

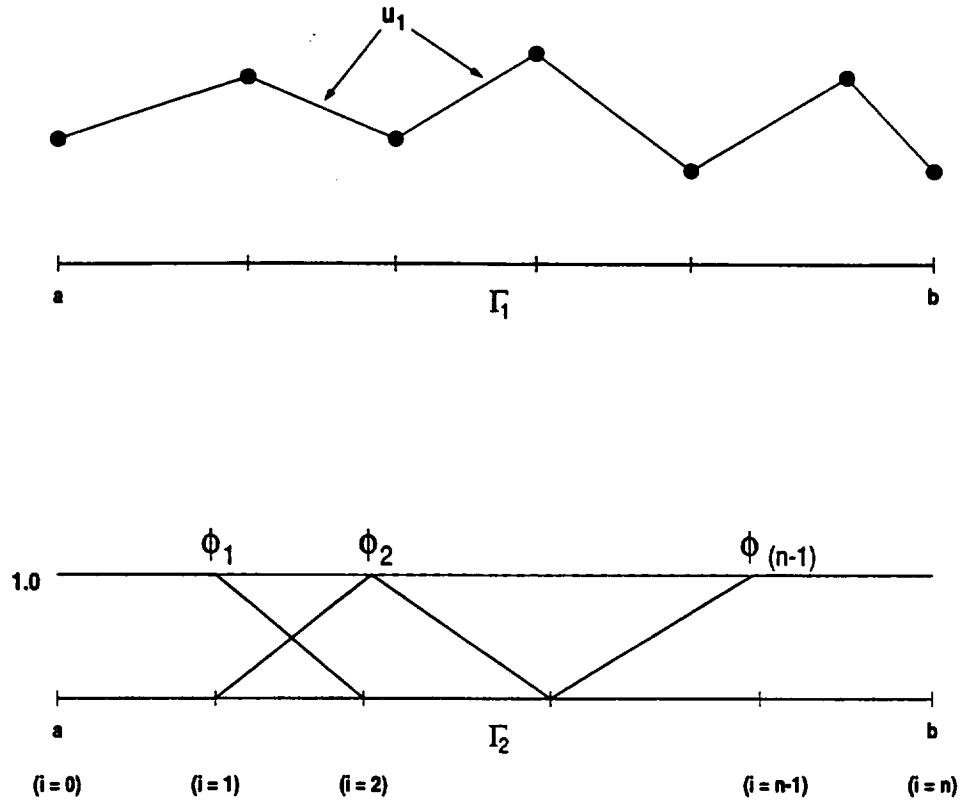

Figure 5.13 : Definition of Test Functions for the Mortar Method

with

$$u_1(a) = u_2(a)$$

$$u_1(b) = u_2(b)$$

Expression (5.8) is similar to what is called the Petrov-Galerkin (PG) method in mathematical literature or the Method of Weighted Residuals (MWR) in engineering literature. The point of difference is that in PG or MWR methods, the test functions ϕ are defined on the same interface as u_1.

5.4.3 Numerical Implementation

The integral equation for the mortar element method, (5.8) can be equivalently written as

$$\underbrace{\int_\Gamma u_1 \phi \, d\Gamma}_{\mathcal{B}_1} = \underbrace{\int_\Gamma u_2 \phi \, d\Gamma}_{\mathcal{B}_2} \tag{5.9}$$

This section will explain how to compute $\mathcal{B}_1$ and $\mathcal{B}_2$. On account of its simplicity, computation of $\mathcal{B}_2$ will be explained first.

5.4.3.1 Computation of $\mathcal{B}_2$

It should be noted that both u_2 and ϕ are defined as piecewise linear functions on Γ_2. This makes the computation of $\mathcal{B}_2$ fairly straightforward and easy to implement. It can be shown that :

- for $i = 1$

$$\int_\Gamma u_2 \phi \, d\Gamma = \frac{1}{2}(x_1 - x_0) u_0^{(2)} + \left[\frac{1}{2}(x_1 - x_0) + \frac{1}{3}(x_2 - x_1)\right] u_1^{(2)} + \frac{1}{6}(x_2 - x_1) u_2^{(2)} \tag{5.10}$$

- for $i = (n-1)$

$$\int_\Gamma u_2 \phi \, d\Gamma = \frac{1}{6}(x_{n-1} - x_{n-2}) u_{n-2}^{(2)} + \left[\frac{1}{3}(x_{n-1} - x_{n-2}) + \frac{1}{2}(x_n - x_{n-1})\right] u_{n-1}^{(2)} + \frac{1}{2}(x_n - x_{n-1}) u_n^{(2)} \tag{5.11}$$

- for all others

$$\int_\Gamma u_2 \phi \, d\Gamma = \frac{1}{6}(x_i - x_{i-1}) u_{i-1}^{(2)} + \frac{1}{3}(x_{i+1} - x_{i-1}) u_i^{(2)} + \frac{1}{6}(x_{i+1} - x_i) u_{i+1}^{(2)} \tag{5.12}$$

In (5.10), (5.11) and (5.12), the superscript (2) indicates that the variables u_i are defined on Γ_2. Also, it is observed that (5.10), (5.11) and (5.12), if written in a matrix form, can be compactly represented as

$$\mathcal{B}_2 = \mathbf{A} \cdot \mathbf{u}^{(2)} \tag{5.13}$$

where $\mathbf{A}$ is a tridiagonal matrix.

5.4.3.2 Computation of $\mathcal{B}_1$

The computation of $\mathcal{B}_1$ is more involved than that of $\mathcal{B}_2$ because u_1 is defined on Γ_1 while ϕ is defined on Γ_2. Referring to Figure 5.14, the integral on the right hand side of (5.9) is computed in the following manner :

1. First an ordered union of points from both the master and slave interfaces is assembled as shown on the intermediate line in Figure 5.14.
2. Mathematical expressions for both u_1 and ϕ are obtained in each of the segments formed by the points in the union, knowing that u_1 and ϕ are both piecewise linear on Γ_1 and Γ_2 respectively.
3. $\mathcal{B}_1$ is computed by the piecewise integration of the product of ϕ and u_1 obtained from step (2) on each of the segments obtained in step (1).

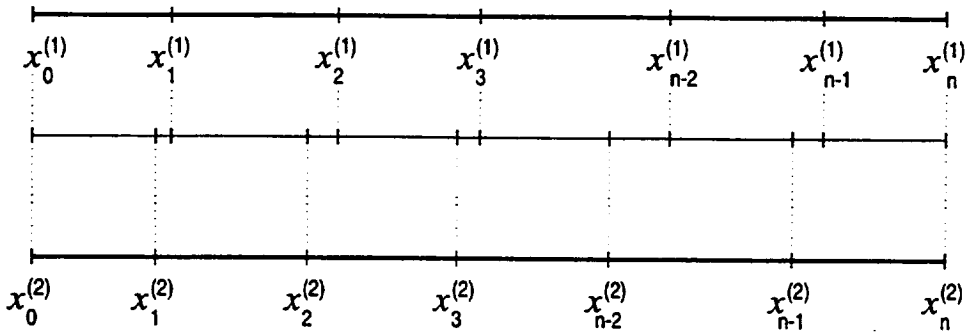

Figure 5.14 : Interface Construction for Computation of $\mathcal{B}_1$

5.4.3.3 Obtaining Slave Variable Values

Computation of $\mathcal{B}_1$ and $\mathcal{B}_2$ as shown above, results in the creation of a tridiagonal system of equations, (5.13), which can be solved efficiently to get u_2.

5.4.4 Properties of the Mortar Element Method

Without going into too many mathematical details, the following are the salient features of the mortar method as obtained from analyses of the method applied to Poisson's equation [11, 12] :

1. The discretization error is of second-order and is independent of the size of overlap as long as the extent of overlap is not smaller than the size of the coarser mesh [12].
2. The mortar method is capable of handling strong gradients in the solution unlike other higher-order interpolation methods which experience spurious oscillations following Gibbs' phenomenon.
3. Computationally, it is fairly easy to implement and inexpensive in case of one dimensional interfaces.

This concludes the discussion of the mortar element method.

5.5 Summary

1. The significance of rotor-stator interaction in turbomachinery and the need for its computational treatment has been outlined. The central issues that would arise in rotor-stator interaction computations are highlighted, especially that of non-matching meshes. This led to the identification of the goals of current research.
2. Earlier work in the area of non-matching meshes for CFD was reviewed, both for rotor-stator interaction and domain decomposition and overset meshes. Two popular methods for rotor-stator interaction are dynamic remeshing and the use of a zonal approach. Details of each of these are given.

3. The problem of non-matching meshes in rotor-stator interaction is similar in nature to those encountered in domain decomposition and overset meshes. Each of these requires the transfer of information from one mesh to another.
4. Information exchange schemes may be classified as conservative and non-conservative. A conservative exchange of information ensures that the total flux across the mesh interface is conserved. In non-conservative schemes, the flux balance is not handled explicitly but an interpolation of conserved variables is used to get information across.
5. A possible drawback of non-conservative schemes is their lack of ability to locate shocks and discontinuities correctly. Hence conservative methods have been preferred. However, these methods can be computationally intensive and hard to implement, especially for unstructured meshes. Also, in some cases, they have been shown to give rise to numerical instabilities.
6. Some examples of conservative methods for rotor-stator interaction and overset meshes are reviewed.
7. A mathematical treatment of non-conservative schemes is given pointing out the requirements for a higher-order interpolation scheme. It was also seen that drawbacks attributed only to the non-conservative interpolation scheme actually had several sources of error and in many cases the process of interpolation was not the culprit.
8. The mortar element method and its use as a projector along an interface is described.

Chapter 6

An Overlapping Mesh Method for Non-Matching Unstructured Meshes

Chapter 5 reviewed problems that arise in the context of rotor-stator interaction and the need for non-matching mesh methods in computational fluid dynamics. Several approaches to this problem were described highlighting their strengths and weaknesses. This chapter presents a new method developed in the present research.

6.1 Review of Requirements

The central issues arising in non-matching unstructured meshes with a finite-volume discretization were discussed in Section 5.2. The primary concerns were the evaluation of convective fluxes at the mesh interface and the determination of gradients required for extension to second-order accuracy.

The main requirements for a non-matching mesh method are the correct evaluation of fluxes and gradients throughout the computational domain, particularly at mesh interfaces, consistent with the spatial discretization adopted in the current fluid solver. This requires an exchange of information between sub-meshes which can be performed either conservatively or non-conservatively as discussed in Chapter 5.

In the present work, both overlapping and non-overlapping methods were investigated. The non-overlapping method is described in Section 6.2. The description is terse because the method did not perform satisfactorily. This is followed by a more detailed description of the new overlapping method in Sections 6.3 through 6.5. This method, identified as SUM (Slipping Unstructured Meshes) was successful on all benchmark examples.

6.2 A Non-Overlapping Flux Conservative Method for Non-Matching Unstructured Meshes

A method similar to that of Wang [67] (Section 5.3.4.3) was investigated. Fluxes were directly computed at the interface between meshes to satisfy the conservation requirement. This method, although conservative, was inconsistent with the spatial discretization and erroneous results were obtained. Prior to giving details about these difficulties, the method will be sketched briefly.

6.2.1 Method Description

Figure 6.1 shows the interface between two non-matching unstructured meshes. Boundaries of the finite-volume cells constructed from the mesh triangulations on either side of the interface are shown by solid lines whereas the triangles themselves are shown by dashed lines. In order to compute fluxes at the mesh interface, Riemann problems are set up by considering segments formed by the union of cell boundaries on the interface, such as the segments $AB, BC, CD, \ldots, LM$, as shown in Figure 6.1. Riemann problems are constructed using the conserved variables on either side of the interface at the mid-points of these segments. Variable values at the mid-points are obtained either by linear interpolation or by mortar projection. Fluxes through the

interface segments are then computed by treating these Riemann problems by Roe's approximate Riemann solver. Depending upon the way in which the interface segment normals are defined, the computed interface fluxes are added to or subtracted from the fluxes computed for the corresponding points on the interface considering the cell interfaces with other points of the triangulations. For example, in the mesh of Figure 6.1, fluxes through segments BC, CD, DE and EF will contribute to the total flux at point D. Because fluxes are added and subtracted in equal amounts for each interface segment, conservation is achieved locally on each interface segment and consequently globally on the whole interface.

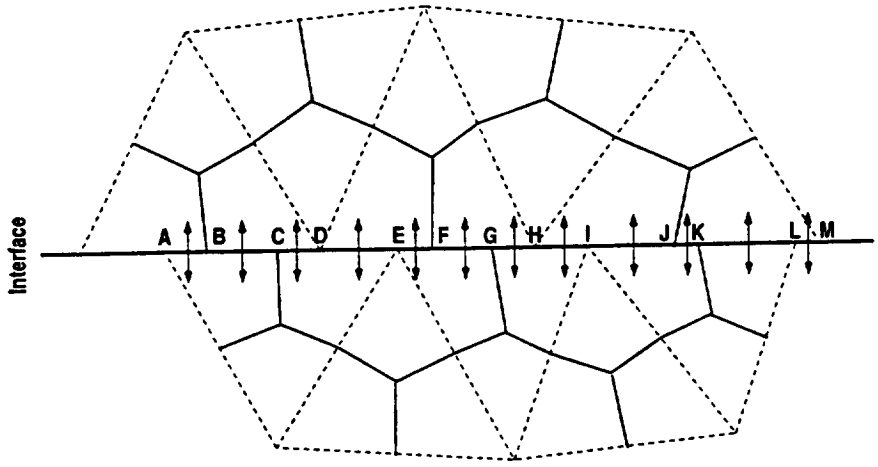

Figure 6.1 : Flux Computation at Mesh Interface

6.2.2 Inconsistency with Spatial Discretization

The method described in Section 6.2.1 is conservative. However, it is inconsistent with the spatial discretization adopted in the current finite-volume method as will be seen next.

To display this inconsistency, consider a typical node in an unstructured mesh, such as node A in Figure 6.2. Fluxes at A can be computed

in a standard manner by setting up Riemann problems at cell interfaces as indicated by the two-way arrows. Let the sum of these fluxes be denoted by ψ and W be the values of the conserved variables at A. Then, the semi-discretized equation (Section 4.1.4) for A can be written as

$$\text{Area}(A)\frac{dW}{dt} + \psi = 0 \tag{6.1}$$

where Area(A) is the area of the finite-volume cell around A.

Next, the mesh is partitioned along line AB in Figure 6.2 to get two meshes as shown in Figure 6.3. Nodes A_l and B_l define the interface of the left mesh whereas nodes A_r and B_r define the interface for the right mesh.

Fluxes for A_l and A_r through cell boundaries not aligned with the interface are computed following the standard procedure for finite-volume cells. Let these fluxes be denoted by ψ_l and ψ_r, respectively. The flux through the cell interface, ψ_i is computed by solving the Riemann problem at point C on the interface as described in Section 6.2.1. The total flux for A_l will thus be $\psi_l + \psi_i$ whereas that for A_r will be $\psi_r - \psi_i$. The semi-discretized equations for A_l and A_r can then be written as

$$\text{Area}(A_l)\frac{dW_l}{dt} + (\psi_l + \psi_i) = 0 \tag{6.2a}$$

$$\text{Area}(A_r)\frac{dW_r}{dt} + (\psi_l - \psi_i) = 0 \tag{6.2b}$$

where W_l and W_r are the conserved variables for A_l and A_r and Area(A_l) and Area(A_r) are the cell areas of finite-volume cells around A_l and A_r respectively.

From (6.1) and the individual equations in (6.2), it is seen that $W_l \neq W_r$ after time-integration even though points A_l and A_r coincide in space.

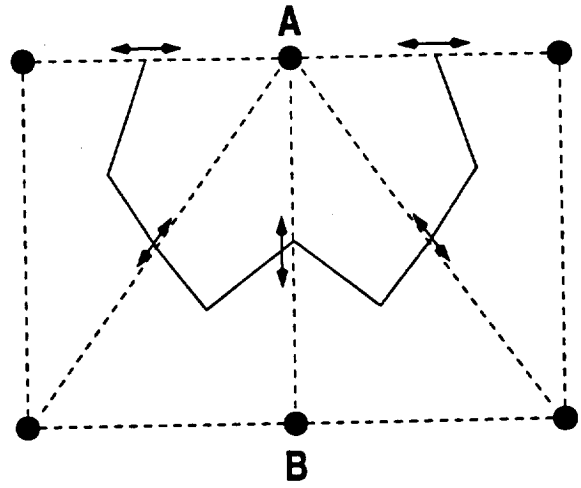

Figure 6.2 : An Unpartitioned Finite-Volume Cell

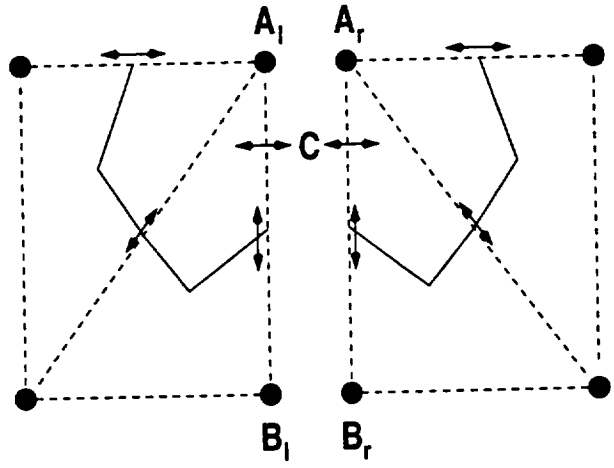

Figure 6.3 : A Partitioned Finite-Volume Cell

Furthermore, each of these values of W_l and W_r would be different from W for the unpartitioned cell, which would be in error.

Another aspect of the this method which makes it practically unsuitable is its inability to consider information upstream and downstream of the interface for points lying on the interface. To show this, referring to Figure 6.4, assume that uniform states exist on either side of the interface, that

is, for all nodes to the left of the interface, including nodes on the interface itself, $W = W_l$, and $W = W_r$ for all points to the right of the interface. If fluxes are to be computed for A_l, there will be a constant value $W = W_l$ on boundaries of the cell around it and hence the total flux for A_l will be zero, because the divergence of a constant is zero. However, for A_r, the flux will be non-zero because $W = W_l$ on the cell boundary aligned with the interface but $W = W_r$ for nodes lying to the right of the interface. Consequently different values of variables will be obtained for A_l and A_r after time integration.

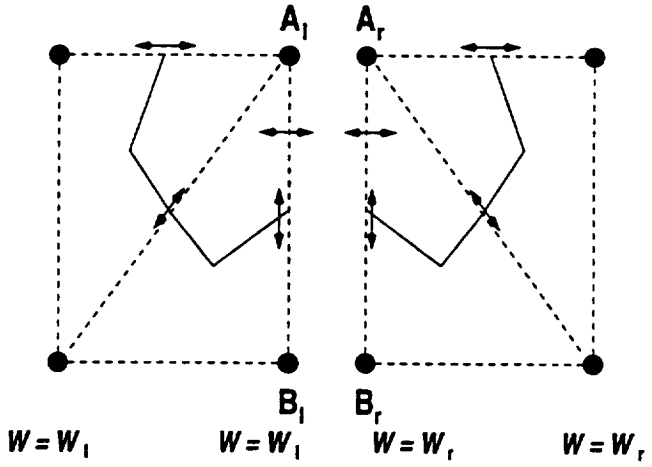

Figure 6.4 : Inability of the Non-Overlapping Method to Compute Fluxes

The last drawback of the non-overlapping method is its inability to compute gradients at the mesh interface that are required for extension to second-order accuracy. This requirement, however, can be relaxed by accepting first-order spatial accuracy at the interface.

Summarizing, it is not possible to devise a consistent, non-overlapping, non-matching mesh method for the spatial discretization adopted in the current finite-volume scheme. To compute flows correctly, it is essential not

only to consider variable values at the mesh interface, but also those in a region slightly away from the interface so as to guarantee proper transfer of information. These requirements mandate the use of an overlapping scheme. Details of various ways in which overlapping schemes can be implemented for unstructured meshes will be discussed in the following sections.

6.3 Overlapping Schemes for Unstructured Meshes

The need to overlap meshes in developing a method for non-matching unstructured meshes was brought out in Section 6.2. Overlapping schemes can be implemented both in conservative and non-conservative forms. This section discusses ways in which overlapping schemes can be implemented for the current finite-volume spatial discretization. Before discussing such schemes, a generalized overlapping finite-volume scheme will be presented.

6.3.1 A Generalized Overlapping Scheme

Two non-matching triangular meshes separated at an interface are displayed in Figure 6.5. Both meshes, in general, are formed by unstructured triangulations, but are shown as structured for clarity. Non-overlapping portions of each mesh are indicated by solid lines whereas the overlapping portions (projecting meshes henceforth) are indicated by dashed lines.

The purpose of the projecting meshes is to gather information about the state of flow in the region of overlap and enable computations to be performed consistently with the spatial discretization. In the present case, overlapping is essential for flux and gradient computations at the interface, as discussed in Section 6.2.2.

Apart from the choice of conservative or non-conservative schemes in which information is transferred between meshes, overlapping schemes differ

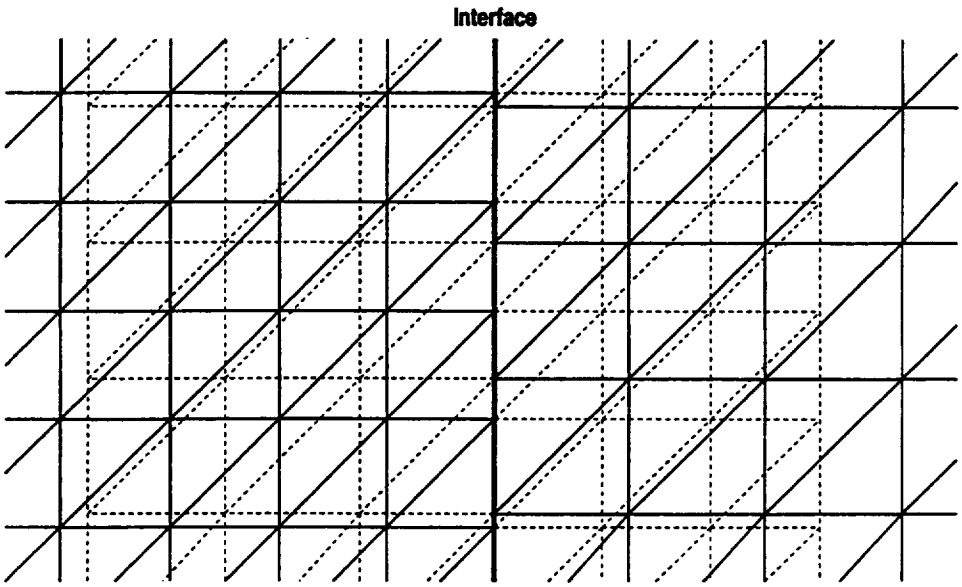

Figure 6.5 : A Generalized Representation of Overlapping Unstructured Triangular Meshes

with respect to the manner in which the actual mesh overlap takes place. The following points need to be addressed while developing an overlapping scheme :

1. Wider overlapping increases the total computational load. Consequently, an optimal extent of overlap has to be determined, as a tradeoff between accuracy and computational expense.
2. Efficient implementation requires not only the overlap distance but also the distribution and arrangement of mesh nodes in the region of overlap.
3. Once an optimal distance and arrangement of mesh nodes has been determined, a decision must be made regarding the treatment of interface nodes on either side. The main question to be answered here is whether variables are to be updated on either side of the interface independently of each other, or is this process mutually dependent. The key requirement is the satisfaction of continuity at the interface.

4. Finally, it is important that computations be performed at the interface and in the region of overlap consistently with those in the non-overlapping regions. That is, no errors other than those associated with information transfer, arise on account of the mesh overlap.

Each of these points will be further discussed in the subsequent sections.

6.3.2 Optimal Mesh Overlap

The need for careful selection of mesh overlap was stressed in the previous section. Here, the significance of the extent of overlap and the arrangement of nodes in the projecting mesh is explained further.

Consider a node P belonging to the left mesh and lying on the mesh interface as shown in Figure 6.6. In this figure, the left mesh projects onto the right mesh in an arbitrary, unstructured manner. The right mesh is displayed as structured and rectangular for clarity.

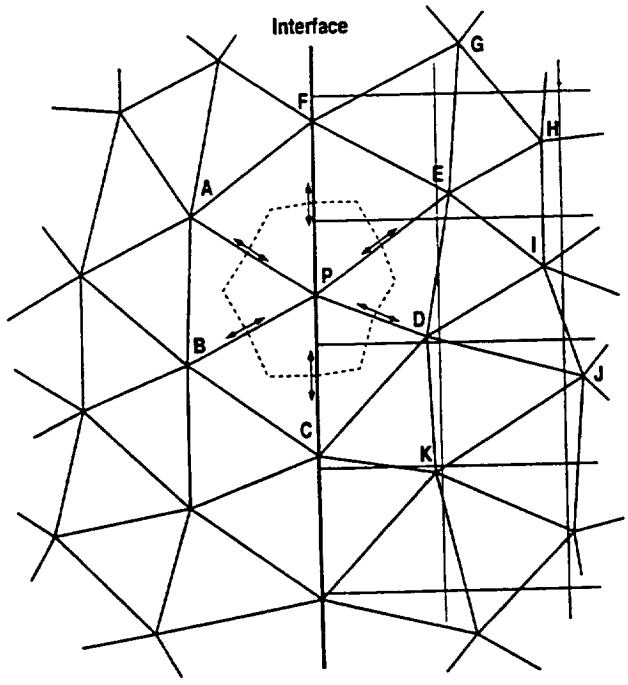

Figure 6.6 : Overlapping for Left Mesh

Overlapping the left mesh onto the right one allows construction of complete finite-volume cells for points lying on the interface. For example, the finite-volume cell about P is drawn in dashed lines Figure 6.6. To update the flow at P, it is essential to know the values of conserved variables at points A, B, C, D, E and F. In addition to the conserved variables, gradients at these points are also required for extension to second-order accuracy. This requires knowing the values of conserved variables at all nodes attached to nodes such as A, B, C, D, E and F. For example, to compute the gradient at D, values of conserved variables at points E, P, C, K, J and I are needed. Similarly, to determine the gradient at point E, values at G, F, P, D, I and H are required. Once values of all variables and gradients are known, fluxes for points such as P can be computed by solving the Riemann problems established at the cell interfaces as indicated by the two-way arrows in Figure 6.6.

From this, it can be concluded that to compute flows for points of the left mesh lying on the interface with second-order accuracy, it is necessary to extend the left mesh two levels into the right mesh. A similar analysis for the right mesh will indicate that the right mesh has to be extended two levels into the left mesh.

The question of the distance or extent over which projection should take place still remains unanswered. Qualitatively, the projecting mesh should penetrate deeply enough to be able to transfer adequate amount of information required for flux computation. Thus, if both the left and right meshes are roughly equally refined, the second level of projection of the left mesh onto the right mesh should approximately coincide with the layer of nodes of the right mesh separated by two triangle levels from the interface. The exact manner in which overlapping occurs in the method developed here is explained in Section 6.4.1.

This concludes the discussion on mesh overlaps. The next section studies the enforcement of continuity on the mesh interface.

6.3.3 Enforcement of Interface Continuity

A natural requirement for any non-matching mesh method is the enforcement of continuity at the mesh interface. Failure to do this gives rise to interface errors that can propagate through the entire mesh and pollute the computed solution. Two interface computation schemes may be followed :

1. In the first method, meshes are overlapped two levels onto each other. Interface fluxes are computed and flow variables are updated independently. This, scheme does not guarantee interface continuity, however, because there is no explicit enforcement of relationship between the right and left flow variables.

   To illustrate this point, consider two coincident nodes located at at O in the sketch of Figure 6.7. Two finite-volume cells are constructed around each one, one delimited by nodes A, B, C, D, E and F and the other by nodes P, Q, R, S, T and U. Unless these two sets of nodes coincide, values of conserved variables at these points will generally be different. Furthermore, because of different cell geometries, the normals at cell interfaces which are used to compute fluxes will also differ. Consequently, the total sum of fluxes at O will depend upon the set of nodes used to compute fluxes. Different values of state variables will be obtained after time-integration, depending upon the set of fluxes and the cell used. It can be concluded that this scheme will give rise to an artificial discontinuity in conserved variables at the interface.

2. Another way to to compute variables is to update the values of variables at points of the left mesh lying on the interface, followed by projection of these values for interface points of the right mesh. This scheme enforces continuity at the interface by accounting the dependence of interface points for the right mesh on those for the left mesh.

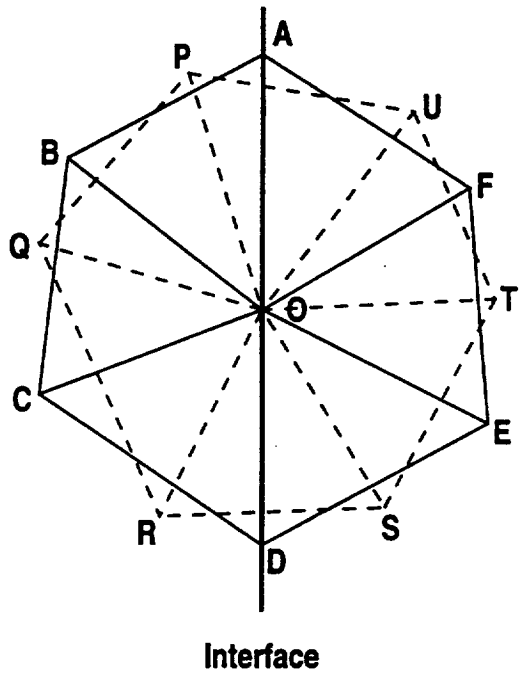

Figure 6.7 : Creation of Discontinuities for Coinciding Nodes on an Interface

If second-order accuracy is to be kept, the projection of interface variables from one mesh to another has an interesting effect on the way in which meshes need to be overlapped. To illustrate this point, consider Figure 6.8 in which the mesh on the right is to be projected onto the mesh on the left. Mesh representation conventions are analogous to those discussed for Figure 6.6.

To begin, note that values of variables at points P, Q, R and S of the right mesh in Figure 6.8 are obtained from the values at A, B, C, D and

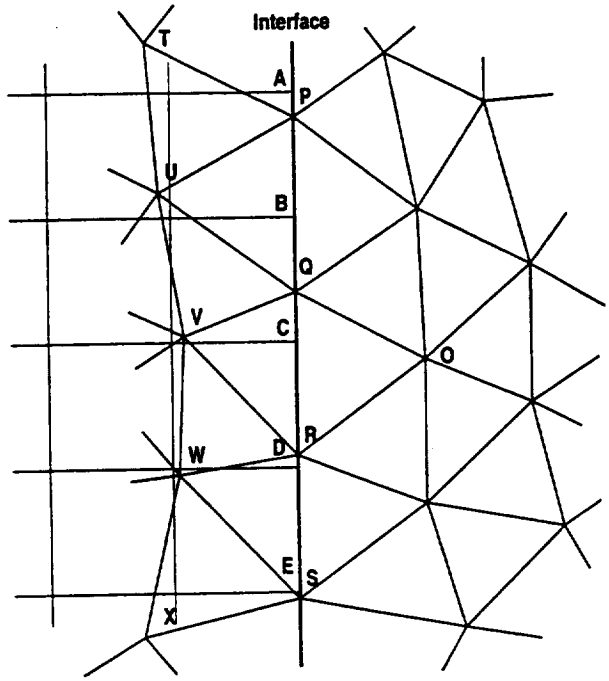

Figure 6.8 : Overlapping for Right Mesh

E of the left mesh. Hence fluxes at P, Q, R and S need not be computed. However, in order to update flow variables at a point in the interior of the right mesh one level away from the interface with second-order accuracy, such as O, it is essential to know the gradient values at points P, Q, R and S. For this, values of variables at points T, U, V, W and X are required. Hence, even though fluxes need not be computed at the interface points of the right mesh, it is necessary to project the right mesh one level onto the left mesh in order to retain second-order accuracy throughout the entire computational domain. Note, however, that if first order accuracy is adequate, a direct interface projection is sufficient. The extent and manner in which the right mesh needs to be projected is similar to that for the left mesh as described in Section 6.3.2.

6.3.4 Information Exchange between Meshes

The key issue concerning the exchange of information between meshes remains unaddressed. This section examines possible approaches to this topic and discusses the relative ease and difficulty in the implementation of several of these methods.

6.3.4.1 Conservative Methods

As mentioned in Chapter 5, methods which guarantee the conservation of fluxes across interfaces are attractive on account of their proven ability to capture the position of shocks and discontinuities correctly. It was also noted, however, that these methods are amenable to computer implementation only for two-dimensional structured grids. It is very hard, if not impossible, to develop an effective implementation for two-dimensional unstructured or three-dimensional meshes. This section examines potential difficulties involved in developing a conservative method for non-matching unstructured meshes.

It should be remembered that even for conservative methods, it is essential to overlap meshes to correctly compute fluxes and gradients at the interface as mentioned in Section 6.2.2. To ensure conservation, it is essential to balance fluxes between finite-volume cells of the projecting mesh and the finite-volume cells into which the mesh overlaps.

One way to do this is to try to generate the meshes in such a way that boundaries between cells of the projecting mesh and the overlapped mesh are aligned in the overlap region and continue to remain aligned even after relative motion between meshes, as is done in the method of Rai and co-workers described in Section 5.3.4.1. This alignment is, in general, very hard to achieve for unstructured triangular meshes.

If cells cannot be aligned, then there are a number of options, similar to those presented in Sections 5.3.4.2 and 5.3.4.3. Each of these requires the precise determination of the manner in which cells overlap each other. This is very hard to implement and computationally expensive for the spatial discretization employed in the present fluid solver, because each finite-volume cell is an arbitrary shaped polygon with the number of polygon vertices generally changing from cell to cell.

On account of these implementational difficulties, it was decided to follow a non-conservative approach in which exact conservation is not specified a priori. Conservation follows in the limit of mesh refinement, provided that the consistency requirement is satisfied on the interface and in the region of overlap.

6.3.4.2 Non-Conservative Methods

Non-conservative methods in CFD were discussed in detail in Chapter 5. In these methods, information exchange occurs by the interpolation of conserved variables from one mesh to another. As the process of variable interpolation does not ensure that fluxes are balanced, conservation is not strictly enforced.

However, as mentioned in Section 5.3.6, the error in conservation on account of non-conservative interpolation depends crucially on the interpolation procedure used and can be greatly reduced by choosing a higher order method. In the present method, the second-order accurate mortar element method (Section 5.4) is used for this purpose.

6.4 SUM : A Method for Slipping Unstructured Meshes

This section describes in detail the method developed during the course of this research, SUM, paying close attention to the following issues :

1. The implementation of optimal overlap between meshes,
2. The enforcement of continuity at the mesh interface, and
3. The exchange of information between meshes either by linear interpolation or the mortar element method.

6.4.1 Implementation of Mesh Overlaps

The present implementation of SUM can handle two non-matching meshes separated at an interface which is aligned with the y-axis. The mesh to the left of the interface is referred to as the left mesh and the one to the right as the right mesh. As described in Section 6.3.2, the left mesh projects two levels onto the right mesh, and the right mesh one level onto the left one.

A simple mesh overlapping pre-processor is implemented which performs an automatic optimal mesh overlap. The extent of overlap is determined by considering the minimum and maximum distances of mesh nodes connected to the mesh interface for each level of extension, as shown in Figure 6.9. In this figure, $\delta^{(1)}_{min}$ and $\delta^{(1)}_{max}$ are the minimum and maximum distances of points directly connected to points on the mesh interface from the mesh interface, whereas $\delta^{(2)}_{min}$ and $\delta^{(2)}_{max}$ are the distances of points connected through one other point to the mesh interface. The extent of mesh overlap for each level, $\delta^{(1)}$ and $\delta^{(2)}$ is then determined by

$$\delta^{(1)} = \frac{\delta^{(1)}_{min} + \delta^{(1)}_{max}}{2}$$

$$\delta^{(2)} = \frac{\delta^{(2)}_{min} + \delta^{(2)}_{max}}{2}$$

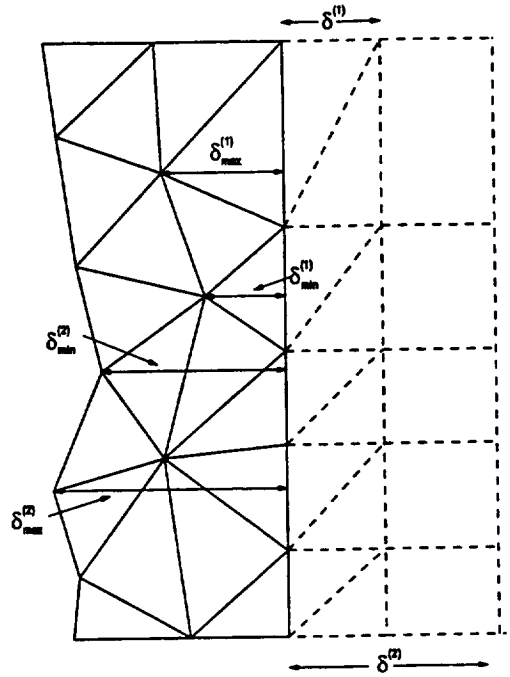

Figure 6.9 : Implementation of Mesh Overlaps

6.4.2 Projection of Variables

To implement the interface continuity requirements discussed in Section 6.3.3, the values of conserved variables at the right mesh interface nodes are obtained from the values at the left mesh interface nodes. To update the flow variables at the left interface nodes, the left mesh is projected two levels deep onto the right mesh. Furthermore, to retain second-order accuracy, the right mesh is extended one level deep onto the left. The extension is performed using the algorithm given in the previous subsection.

In Figure 6.10, triangles of the left and right meshes are indicated by dashed lines. Projecting meshes are shown using solid lines. Lines 1, 2, 3 and 4, on the nodes of which projection of variables is required, are defined as follows :

1. Line 1 is the actual interface between the two meshes. It is used to

enforce continuity by projecting the values of variables from the left interface mesh points to the right as pointed out in Section 6.3.3. In addition to enforcing continuity at the interface, line 1 supplies artificial boundary conditions for the right mesh in accordance with equation (5.6).

2. Lines 2 and 3 are formed by extending the left mesh into the right mesh. Variables on line 2 and 3 are used to supply the artificial boundary conditions for the left mesh.
3. Line 4 results by the overlap of the right mesh into the left mesh. Variables on line 4 supply artificial boundary conditions for the right mesh.

These mesh projection lines will be referred to as lines 1, 2, 3 and 4 in the subsequent discussion.

Values of variables can be obtained at these points either by linear interpolation (henceforth referred to as SUM/LI) or by using a higher-order projection scheme based on the mortar method (SUM/MP standing for SUM/Mortar Projection). Both methods were implemented and tested in the present work.

6.4.2.1 SUM/LI : Linear Interpolation

In SUM/LI, linear interpolation is easily implemented by simply considering the location of the point at which variables are to be obtained in the mesh in which overlap occurs and assuming a linear variation of variables over triangles.

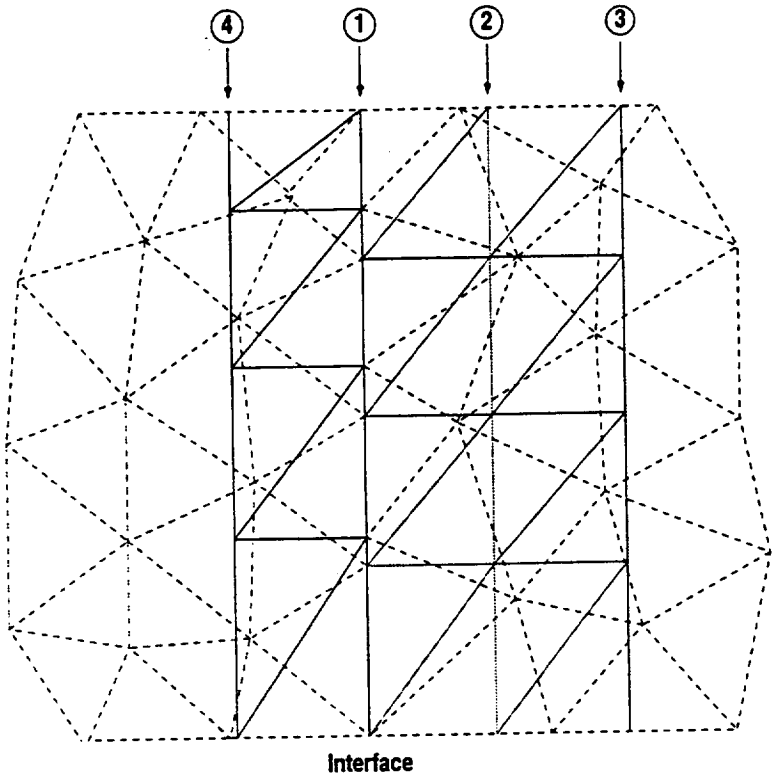

Figure 6.10 : Mesh Interfaces for Variable Projection

6.4.2.2 SUM/MP : Mortar Projection

The basic ideas behind the mortar method are described in Section 5.4. Application of the mortar method requires the specification of a master interface on which the values of variables are regarded as known. In SUM/MP, the master interface is constructed by considering the intersections of the projecting mesh with lines of the mesh into which overlap occurs as indicated in Figure 6.11.

In this figure, dashed lines represent lines of the projecting mesh, whereas lines of the mesh into which overlap occurs are represented by solid lines. Nodes of the projecting mesh where values of variables are sought are indicated by squares (which form the slave interface) and intersection points between the lines of the two meshes are shown by circles (which form the

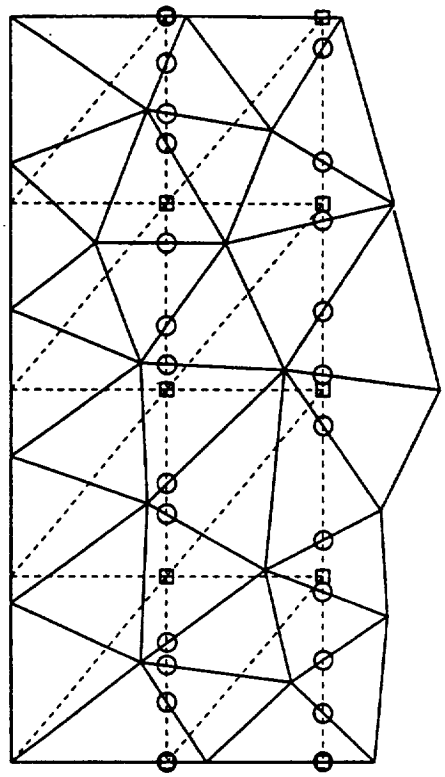

Figure 6.11 : Construction of Master Interfaces for the Mortar Method

master interface).

Values of variables at points belonging to the master interface can be obtained by assuming a linear distribution of variables over each triangle of the mesh into which overlap occurs. Based on these values, all information required for the construction of $\mathcal{B}_1$ (Section 5.4.3.2) is available and it can be computed. $\mathcal{B}_2$ (Section 5.4.3.1) is assembled by knowing the co-ordinates of points on the slave interface. Once $\mathcal{B}_1$ and $\mathcal{B}_2$ are available, values of variables on the slave interface can be easily computed by solving the resulting tridiagonal system of equations as described in Section 5.4.3. This process is applied on lines 2–4. For line 1, the master interface is constructed directly from the interface points of the left mesh.

At first glance, the non-symmetric way in which the meshes overlap

may give the impression that the left mesh gets more preference than the right mesh. However, it should be noted that each mesh depends equally on the other to get information used to prescribe artificial boundary conditions.

The left mesh depends on the right mesh to prescribe artificial boundary conditions on lines 2 and 3, whereas the right mesh depends on the left mesh for artificial boundary conditions on lines 1 and 4. Thus, line 1 can be viewed as an extension of the right mesh onto the left mesh. However, instead of projecting onto the left mesh, it coincides with the physical interface between the two meshes. This serves dual purpose : enforcing continuity at the interface and providing artificial boundary conditions for the right mesh.

This completes the description of the method to project variables at points belonging to the projecting meshes and at the interface between the non-matching meshes. The following section describes the incorporation of this methodology into a time-stepping algorithm.

6.4.3 An Algorithmic Description of SUM

At present, SUM accepts two non-matching meshes which are pre-processed following the procedure outlined in Section 6.4.1. This section describes in further detail the steps devised to handle non-matching unstructured meshes in an algorithmic fashion. The description focuses on aspects particular to the treatment of non-matching meshes. Implementation details standard to the finite-volume fluid solver described in Chapter 4, such as the computation of cell areas and boundary normals, incorporation of the geometric conservation law into the Riemann solver, and treatment of boundary conditions are omitted for brevity.

6.4.3.1 Step 1 : Initialization

Initial conditions to the problem being solved are provided either by starting from a uniform flow solution or a pre-computed solution which is used for the restart option. Several other initializations for particular problems (such as for the shock tube problem) are also possible.

Flow is initialized by prescribing the values of the conserved variables at all mesh points. Values of pressure at each node are obtained from

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho||\vec{U}||^2\right) \tag{6.3}$$

This gives knowledge of the flow field over the entire computational domain.

In addition to the initialization of flow variables, a list of 'candidate' segments belonging to the overlapped mesh is prepared during initialization to test for intersections with each line of the projecting mesh. This is required for the construction of the master interface as described in Section 6.4.2.2. Referring to Figure 6.12, if sliding motions are restricted to take place along the y-axis, segments of line AA' of the projecting mesh will intersect only those segments of the overlapped mesh indicated by the thicker lines. This optimizes the determination of segment intersections. If a stationary problem (that which does not involve moving meshes) is to be solved, all the segment intersections are computed and interpolation coefficients stored for reuse.

6.4.3.2 Step 2 : Mesh Motion and Geometrical Update

At each time-step, meshes undergo rigid-body displacements with a user-prescribed velocity. In addition to the updating of the co-ordinates of nodes, computation of mesh velocities and recomputation of cell normals, intersections of segments of the projecting mesh with those of the overlapped mesh must be computed at each step.

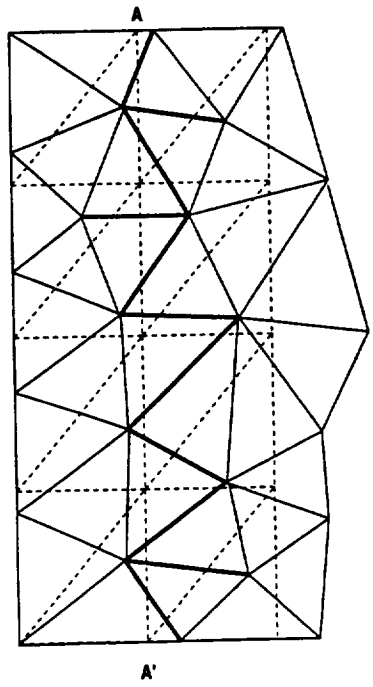

Figure 6.12 : Candidates for Segment Intersection

A point to note here is the inclusion of the effects of periodic boundary conditions, if they appear. Their treatment is illustrated in the rotor-stator benchmark problem considered in Chapter 7. A model configuration for rotor-stator interaction simulation is shown in Figure 6.13. Meshes generated around the two airfoils are separated at the interface indicated by the bold line. The left airfoil represents the rotor and moves downward as computations proceed. For each airfoil, dashed lines indicate lines of the corresponding projecting mesh. For clarity, only one line of projection is shown for each mesh. Periodic boundary conditions are imposed on the upper and lower boundaries of the mesh around each airfoil.

Let the position of the rotor airfoil be that as shown in the lower part of Figure 6.13. In this state, values of variables are required along line $B_1B_3B_2$. Note that on account of periodicity, flow at B_2 is the same as that

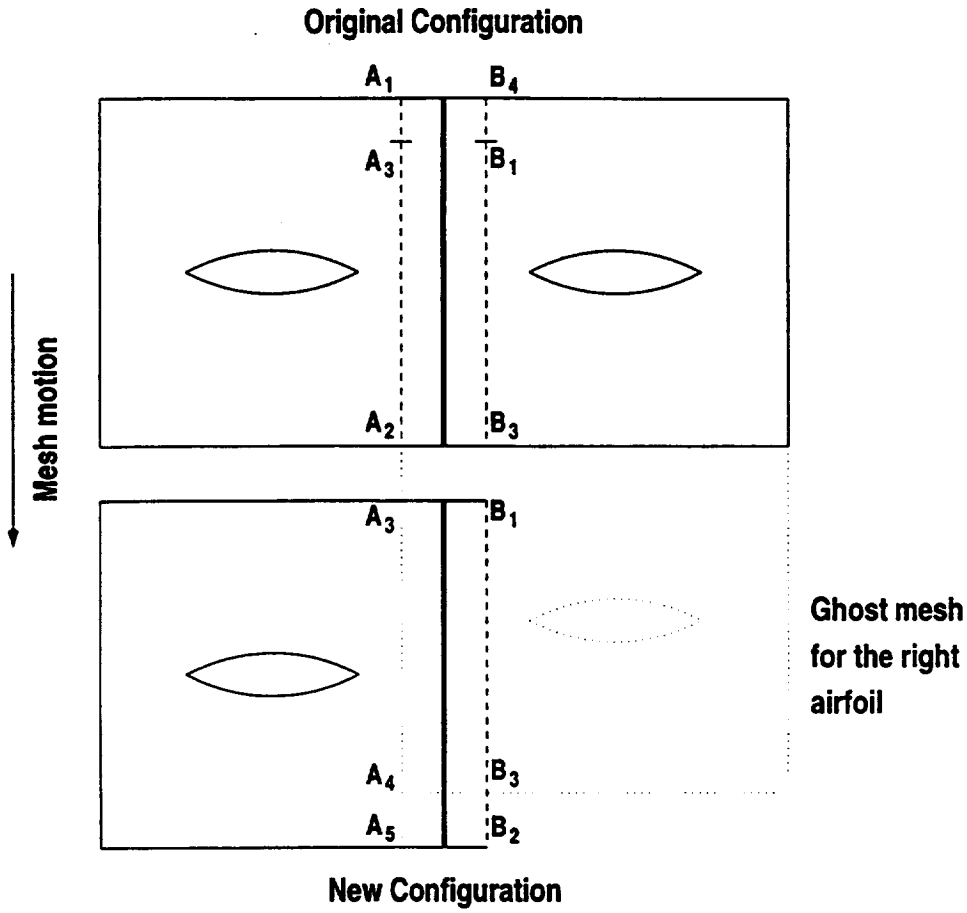

Figure 6.13 : Construction of Master Interfaces with Periodic Boundary Conditions

at B_1. However, no mesh is available to the right of the rotor airfoil in this configuration. Taking periodicity into consideration, flow variables can be obtained at B_3 from the original configuration, where B_3 lies on the lower boundary of the rear airfoil. Thus, the master interface on $B_1B_3B_2$ can be constructed by using information from B_1B_3 in the original configuration for B_1B_3 in the new configuration, and that from B_4B_1 in the original configuration for B_3B_2 in the new configuration. Similarly, information along $A_1A_3A_2$ for the rear airfoil can be obtained from A_3A_4 in the new configuration for A_3A_2 and A_4A_5 for A_1A_3.

Co-ordinates of intersection points are obtained for each segment

which enables the determination of interpolation coefficients. The list of nodes (node numbers of the end points of segments of the overlapped mesh which intersect a segment of the projecting mesh) from which interpolation occurs and the interpolation coefficients are stored for each intersection of segments. This process will give the co-ordinates of the points indicated by circles in Figure 6.11. Note that for line 1, segment intersections do not have to be computed because the interface nodes of the left mesh themselves provide the master interface.

Once segment intersections are determined, Step 3 calculations described in the next subsection are executed at each station of the Runge-Kutta time-stepping procedure.

6.4.3.3 Step 3 : Construction of the Master Interface

The next step in the algorithm is to determine the values of conserved variables at points forming the master interface, the co-ordinates of which are obtained in Step 2.

This is implemented knowing the values of conserved variables at the end points of the segment of the overlapped mesh, which is intersected by the segment of the projecting mesh and the interpolation coefficients. Thus, if $\mathbf{W}_1$ and $\mathbf{W}_2$ are the values of conserved variables at the segment end-points, and ξ_1 and ξ_2 are the interpolation coefficients, values of conserved variables at the corresponding point on the master interface, $\mathbf{W}_m$, are obtained by

$$\mathbf{W}_m = \xi_1 \mathbf{W}_1 + \xi_2 \mathbf{W}_2$$

Repeating this process over all points on lines 2, 3 and 4 of Figure 6.10, enables the construction of master interfaces along these lines. These are the values used to supply the artificial boundary conditions discussed in Section 6.4.2.

6.4.3.4 Step 4 : Information Exchange between Meshes

Once the master interfaces are constructed as in Step 3 above, information is exchanged between meshes at lines 2, 3 and 4. This can be done either by linear interpolation (SUM/LI) or via projection using the mortar element method (SUM/MP).

Linear interpolation is performed by considering the location of slave points, indicated by squares in Figure 6.11, within the corresponding master interface line. Once the location is determined, values of variables at the slave points are obtained by linear interpolation of the values of variables from the end-points of segments of the master interface.

Mortar projection is harder to implement. It is carried out by following the general technique of Section 5.4, which is specialized for this problem as follows :

1. Knowing the co-ordinates of points on the slave interfaces, construct the $\mathcal{B}_2$ matrix as described in Section 5.4.3.1 for each slave interface.
2. From the co-ordinates of the master points and the values of conserved variables at these points obtained in Step 3, construct the $\mathcal{B}_1$ matrix for each master interface.
3. Having computed both $\mathcal{B}_1$ and $\mathcal{B}_2$ for lines 2, 3 and 4 of Figure 6.11, values of conserved variables at points of the projecting mesh are obtained by equating $\mathcal{B}_2 = \mathcal{B}_1$. Solution of the tridiagonal system of (5.13) gives the required values.

Once the values of conserved variables are known, pressure values at each point of the projecting mesh are updated following (6.3).

At this juncture, values of all variables at all points of each mesh are known at the current step of the Runge-Kutta iteration. Thus each mesh can

be considered separately for the computation of fluxes, which is described next.

6.4.3.5 Step 5 : Flux and Gradient Computation

Because the values of conserved variables are known at all points of each mesh, the process of computing fluxes and gradients is the same as that for a single mesh computation as described in Sections 4.1.3.3 and 4.1.3.4. That is, the convective fluxes through the cell boundary along each mesh segment are approximated by

$$\int_{\partial C_i(t)} \vec{\mathcal{F}}^c(W, \dot{\vec{x}}) \cdot \vec{n}\, d\sigma = \sum_{j \in K(i)} \Phi(W_i, W_j, \vec{n}_{ij}, \sigma_{ij})$$

where the integral of $\vec{\mathcal{F}}^c(W, \dot{\vec{x}}) \cdot \vec{n}$ over the boundary of cell around node i, C_i, given by $\partial C_i(t)$ at time t, denotes the convective flux through $\partial C_i(t)$, $K(i)$ is the set of nodes of the triangular mesh connected to node i, W_i and W_j are the conserved variables at nodes i and j, n_{ij} is the integral of the cell normal over $\partial C_i(t)$ and σ_{ij} is a term for the inclusion of the effects of the geometrical conservation law as defined in Section 4.1.3.3. Φ, in this case denotes a 'generalized' flux which can be either Roe's approximation of the convective flux alone or can also include the Steger-Warming flux at far-field boundaries.

Fluxes at points of the projecting mesh and for interface points of the right mesh are set to be zero since values of variables are not updated at those points.

6.4.3.6 Step 6 : Time Integration

Once fluxes are computed at the end of Step 5, the following semi-discretized equation is obtained :

$$\frac{dW}{dt} + \psi(W) = 0 \tag{6.4}$$

In (6.4), ψ represents the computed fluxes and encapsulates other details like the imposition of boundary conditions and treatment to include stipulations of the geometric conservation law. The Runge-Kutta algorithm used to numerically integrate (6.4) can be summarized as follows :

$$\begin{cases} W^{(0)} = W^n \\ W^{(k)} = W^{(0)} - \dfrac{\Delta t}{4-k}\psi(W^{(k-1)}) \qquad k = 1,2,3 \\ W^{n+1} = W^{(3)} \end{cases}$$

In the above representation, the superscript (k) denotes the k^{th} step of the Runge-Kutta algorithm whereas n is the n^{th} time-step.

6.4.3.7 Step 7 : Enforcement of Interface Continuity

At the end of each Runge-Kutta step, conserved variables on the interface of the left mesh (line 1) are projected onto the interface nodes for the right mesh, in order to enforce continuity at the interface as discussed in Section 6.3.3. This, again, can be done either by linear interpolation or using mortar projection, as for lines 2, 3, and 4 described earlier. However, no separate construction of the master interface is required in this case because interface points of the left mesh themselves form the master interface.

This concludes the algorithmic description of SUM. A schematic representation of the algorithm is given in the box in Figure 6.14.

- **Initialization**
  - initialize flow variables.
  - pre-process segment intersections.
- **Start time loop.**
  - move the meshes and compute segment intersections.
  - start Runge-Kutta integration loop.
    - interpolate master variables on lines 2, 3, and 4.
    - project variables on lines 2, 3 and 4.
    - compute gradients and fluxes for separately for each individual mesh.
    - perform a Runge-Kutta integration step.
    - project variables on line 1 to enforce continuity at the interface.
  - end Runge-Kutta integration loop.
- **End time loop.**

Figure 6.14 : Schematic Representation of the SUM Algorithm

6.5 Analysis of Conservation Error

Details about SUM and its algorithm were given in Section 6.4. The primary requirement on any scheme for non-matching meshes is its ability to conserve fluxes between meshes. Since flux conservation is not imposed in SUM a priori, the performance of SUM with regard to global conservation error is assessed here. This gives a quick picture of its its applicability to problems in CFD. For this purpose, a simple numerical experiment was performed. The aim of this experiment is merely to analyze SUM for conservation. The

results of a more extensive set of benchmark experiments are presented in Chapter 7.

6.5.1 Experimental Procedure

A simple but effective way of quantifying the conservation error of a non-matching scheme is to measure the flux imbalance across the mesh interface at steady state. This follows from the fact that flux over a closed volume must vanish at steady state.

In Figure 6.15, dotted lines denote mesh triangles, the thin solid line is the mesh interface, the bold solid line shows the cell interfaces through which fluxes are computed to analyze the conservation error. Dashed lines are the cell boundaries which are not considered in the present analysis. Thus, the difference of fluxes (indicated by arrows) through lines AA' and BB', normalized to the total flux through AA' will give an estimate of the loss of accuracy in conservation for the numerical scheme.

For the present investigation, the problem of supersonic flow over a ramp (Section 7.1) is used with varying degrees of mesh refinement on either side of the mesh discontinuity seen in Figure 6.16. Let n_1 be the number of points on the mesh discontinuity to the left and n_2 the same to the right. Unstructured meshes are generated on each side of the discontinuity so that mesh spacing on the boundaries of each mesh corresponds to n_1 for the left mesh and n_2 for the right mesh. The simulation was run until the system attained steady state within numerical tolerances.

6.5.2 Results and Conclusion

Table 6.1 shows the percentage imbalance in mass fluxes across the mesh discontinuity for varying values of n_1 and n_2 with both first and second

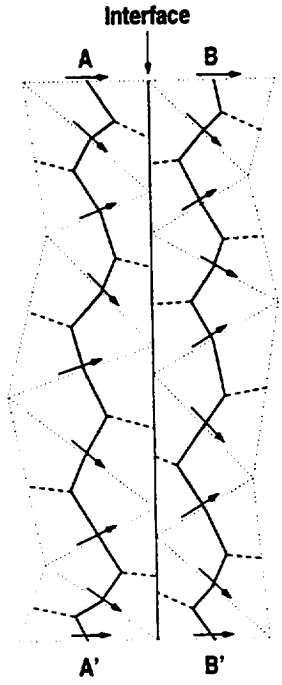

Figure 6.15 : Experimental Setup for Analysis of Conservation Error

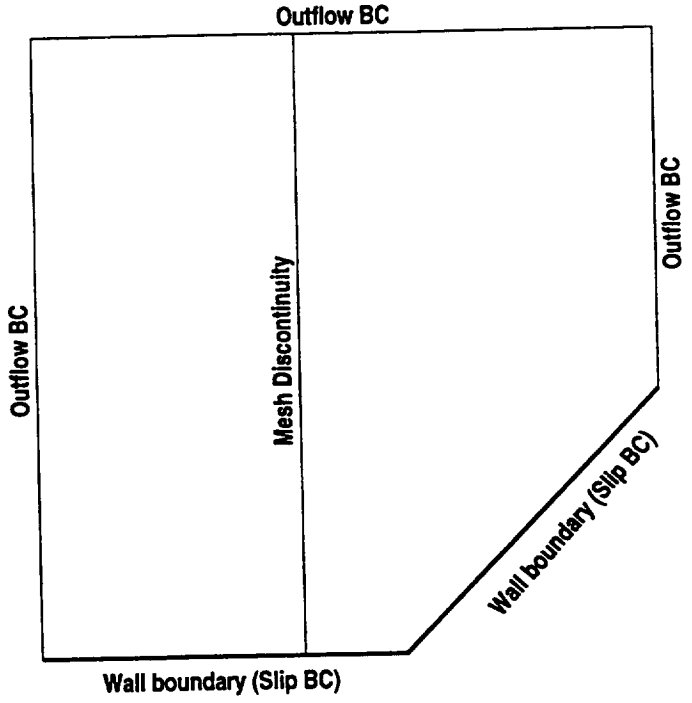

Figure 6.16 : Mesh Model for Supersonic Flow over a Ramp

order accuracy using linear interpolation (SUM/LI) and mortar projection (SUM/MP) for variable transfer. The overall flux conservation error does not exceed 3.3% and 1.6% for the first and second-order methods respectively.

For a constant ratio of n_1/n_2, the conservation error decreases as both n_1 and n_2 increase, i.e. the meshes are refined. Keeping a fixed value of n_1, the conservation error initially decreases to low value for the case where $n_1 = n_2$ and then increases as n_2 is increased, i.e. the mesh on the right is finer than that on the left. However, a drop in error occurs when n_2 is further increased. In most cases, better conservation is achieved when the mortar method is used for projection.

The principal conclusion to be drawn from these results is that for sufficiently refined meshes with similar degrees of refinement on each side of the interface, high accuracy in flux conservation can be achieved with the present method. This makes the method particularly suitable for CFD applications such as rotor-stator interaction where meshes have to be sufficiently and almost equally refined on each side of the mesh discontinuity. The use of the mortar method, however, has not significantly reduced the conservation error at the interface as seen in Table 6.1. A similar level of accuracy is achieved if the more inexpensive linear interpolation is used.

6.6 Summary

1. Requirements for computing flows with non-matching unstructured meshes using the finite-volume spatial discretization employed in the present fluid solver are reviewed.
2. An attempt made to develop a non-overlapping, flux conservative method is described and the reasons for its non-applicability explained.

Table 6.1 : Percentage Error in Conservation for the Ramp Problem

Accuracy :	First Order		Second Order	
n_1, n_2	SUM/LI	SUM/MP	SUM/LI	SUM/MP
12, 10	3.33%	2.93%	1.29%	1.61%
24, 20	1.84%	1.77%	1.24%	0.91%
48, 40	0.66%	0.64%	0.44%	0.38%
31, 16	1.22%	0.92%	1.96%	1.56%
31, 25	0.51%	0.57%	0.19%	6.96×10^{-2}%
31, 31	0.66%	0.64%	9.51×10^{-3}%	5.51×10^{-2}%
31, 37	0.12%	0.17%	0.14%	0.29%
31, 46	0.12%	0.17%	0.11%	0.19%

The need to overlap meshes, both for conservative and non-conservative methods is emphasized.

3. Technical aspects pertaining to overlapping schemes for unstructured meshes, such as the extent of mesh overlap and the enforcement of continuity at the mesh interface, are put forth.
4. Consistency difficulties in achieving flux conservation are brought out, thus giving reasons for using non-conservative methods.
5. The method developed during this research, SUM (Slipping Unstructured Meshes) is described in detail, and its ability to conserve fluxes assessed.
6. The interface conservation test shows that although the mortar element method allows efficient and accurate transfer of variables between meshes, its use does not give much improvement over the linear interpolation method as regards to interface flux conservation.

Chapter 7

Results

Several numerical experiments have been performed to assess the method developed, SUM. This chapter describes the details of these experiments, which includes the motivation for the problems being solved, problem physics, mesh generation, details of solution method, the results obtained and the conclusions that could be drawn from these results.

As mentioned in Chapter 5, a key measure of success or acceptability of SUM is the ability to allow the smooth and undisturbed passage of shocks through mesh discontinuities. An estimate of the flux-conserving ability of SUM is given in Section 6.5. This chapter reports results from simulations performed on standard benchmark problems for which analytical solutions are available. In addition, as a representative of rotor-stator interaction, a more realistic problem involving the relative motion between two successive airfoils is treated. In all but one of the experiments performed, SUM/MP is used.

7.1 Supersonic Flow Over a Ramp

Prior to testing SUM on unsteady problems, it is used on a simple two dimensional problem commonly found in literature on compressible flow, namely that of a supersonic flow over a ramp.

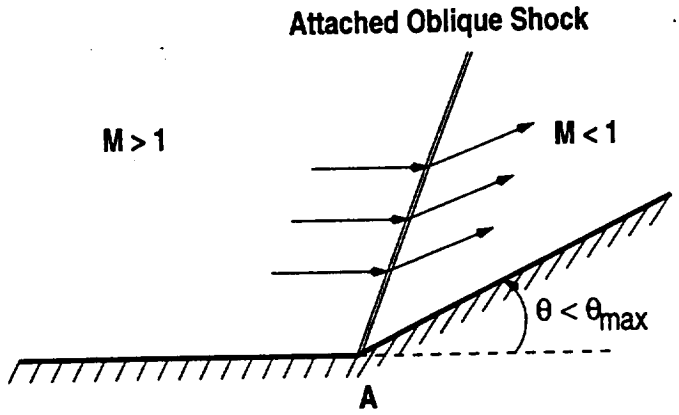

Figure 7.1 : Attached Oblique Shock for Supersonic Flow over a Ramp

7.1.1 Problem Physics

Consider a uniform supersonic flow bounded on one side by a wall as shown in Figure 7.1. At point A, the wall is inclined upwards into the flow by an angle θ. On account of the slip condition imposed on the wall boundary, flow will have to be parallel to the wall both upstream and downstream of point A. The result will be a sudden change in the direction of flow downstream of point A leading to the creation of an oblique shock as shown in Figure 7.1. This shock is attached to the wall at point A (the point of deflection) and hence referred to as an *attached oblique shock.*

It can be shown [4, 33], that if the angle of inclination of the wall, θ, is larger than an angle $\theta_{\max}$, which depends upon the freestream Mach number, then the resulting shock will no longer be attached and oblique as in Figure 7.1, but will have a curved shape such that it is normal to the wall at the point of contact. Also, the shock will no longer be attached to the wall at point A but will be attached somewhere upstream. Such a shock is called a *detached bow shock*, as shown in Figure 7.2.

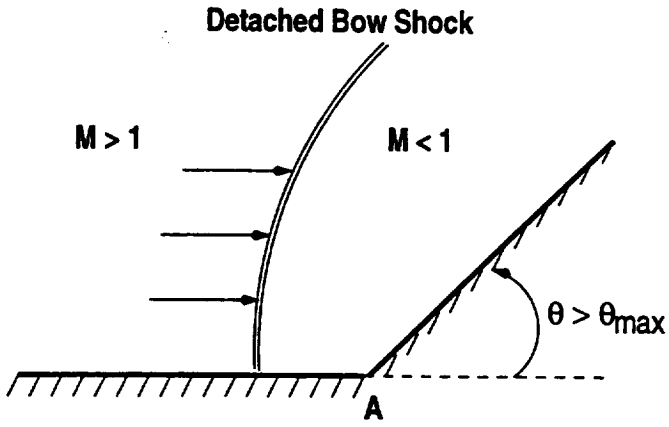

Figure 7.2 : Detached Bow Shock for Supersonic Flow over a Ramp

7.1.2 Motivation

The goal of this exercise is to see if the current approach to patched mesh computations could capture the bow shock correctly without any distortions near the individual mesh boundaries for supersonic steady state problems.

7.1.3 Modeling and Mesh Generation

For simplicity of mesh generation, the patched mesh for this simulation is composed of two algebraically interpolated meshes as shown in Figure 7.3. The angle of inclination is 45° and the freestream Mach number is 2.5. This is greater than the maximum angle θ_{max} for the prescribed Mach number and hence would result in the creation of a detached bow shock as mentioned before.

7.1.4 Results and Discussion

Figure 7.4 shows the pressure contours for the current simulation at steady state. The bow shock has been captured well and it can be seen that the pressure contours are smooth and continuous across the mesh discontinuity,

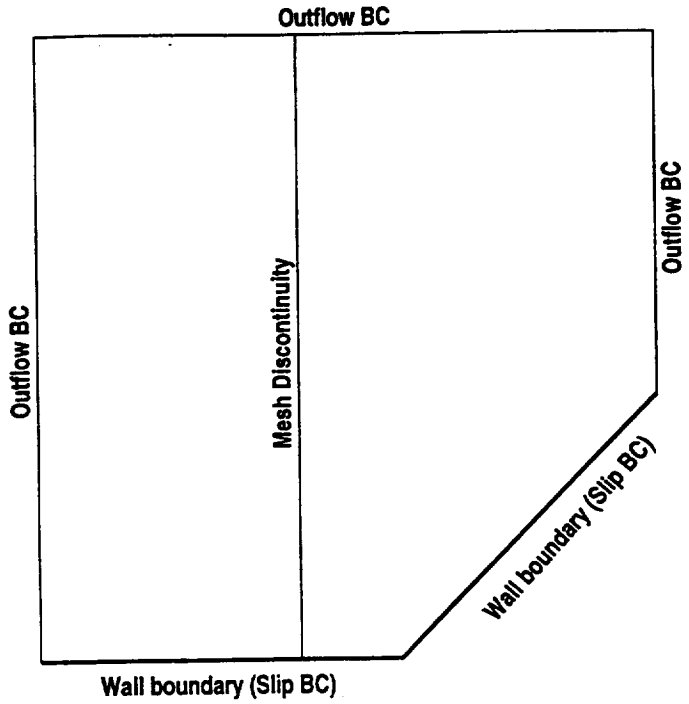

Figure 7.3 : Mesh Model for Supersonic Flow over a Ramp

indicating that the current scheme performs satisfactorily for steady-state problems.

7.2 Transonic Flow through a Channel with a Bump

To assess the ability of SUM to simulate transonic flows, it is tested on the case of transonic flow through a channel with a bump.

7.2.1 Problem Physics

Consider a uniform subsonic flow from left to right through a channel with a bump as seen in Figure 7.5. This situation is similar to that of transonic flow over an airfoil as described in [5, 33]. As the flow goes over the airfoil, it is accelerated as increase in the thickness of the airfoil leads to a decrease in the area of cross-section. It is seen that for uniform flows with Mach

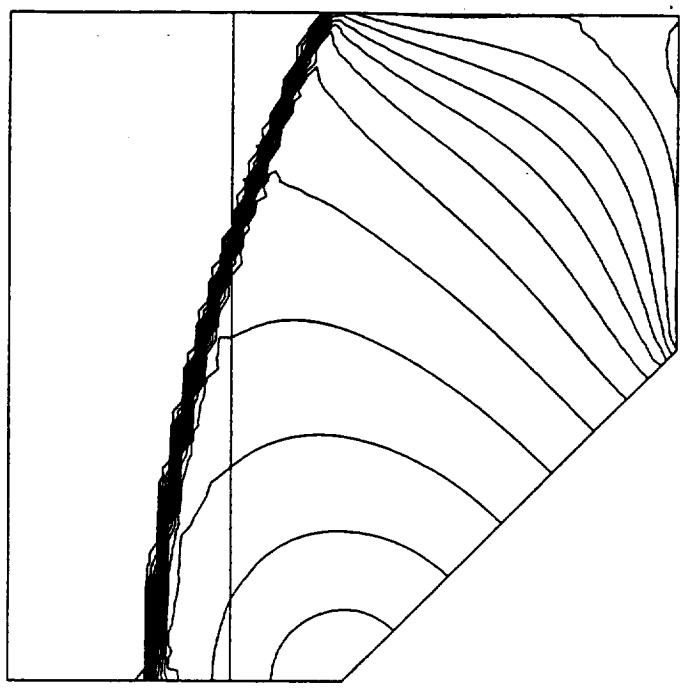

Figure 7.4 : Pressure Contours for Supersonic Flow Over a Ramp

numbers M_∞ less than a critical Mach number M_{cr}, the flow remains subsonic throughout. If $M_\infty \geq M_{cr}$, the flow gradually becomes supersonic and finally leads to the creation of a shock as seen in Figure 7.5.

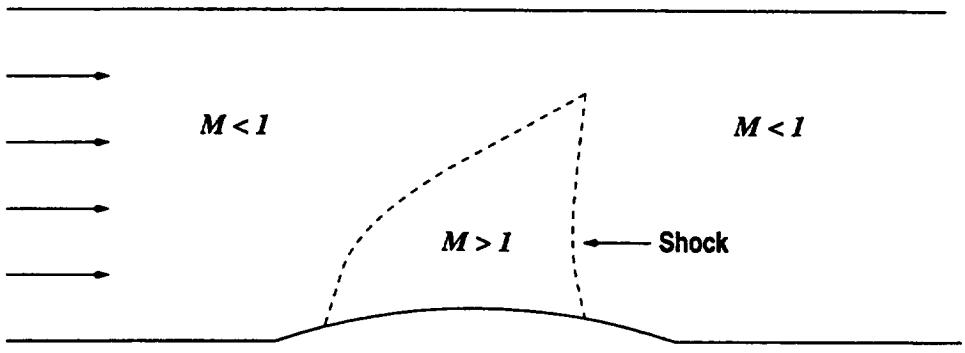

Figure 7.5 : Transonic Flow through a Channel with a Bump

7.2.2 Motivation

The primary motivation to analyze this problem is to test the ability of SUM to simulate transonic flows, irrespective of flow direction.

7.2.3 Modeling and Mesh Generation

A schematic representation of the model used for mesh generation is shown in Figure 7.6. Slipping boundary conditions are imposed on the top and bottom walls of the channel and inflow and outflow boundary conditions are specified on the left and right boundaries of the mesh respectively. Mesh generation was parameterized by the length of the arc l, and its maximum thickness t expressed as a fraction of l. In the simulations reported here, $l = 1$ and $t = 0.1$. Experiments were performed for two types of mesh discontinuities :

1. the mesh discontinuity is parallel to the direction of flow,
2. the mesh discontinuity is perpendicular to the direction of flow.

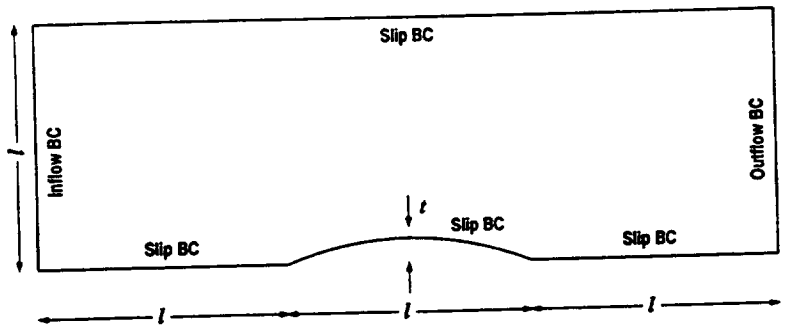

Figure 7.6 : Mesh Model for Transonic Flow through a Channel with a Bump

7.2.4 Results and Discussions

Figures 7.7 and 7.8 show the contours of the Mach number at steady state for flow going from left to right and right to left respectively for a single mesh computation.

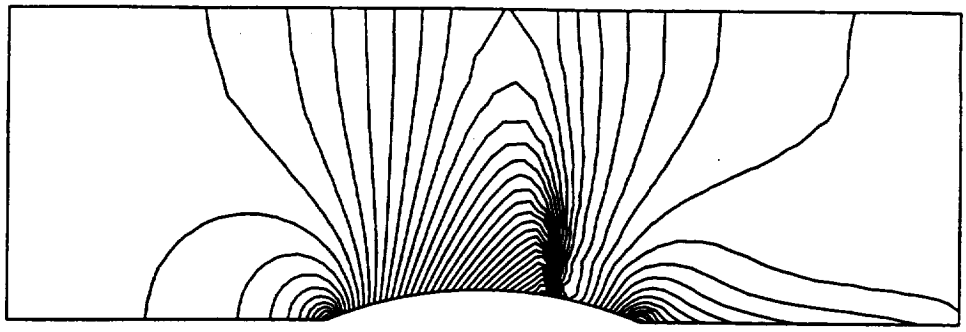

Figure 7.7 : Mach Number Contours for Transonic Flow through a Channel with a Bump — Flow from Left to Right using a Single Mesh

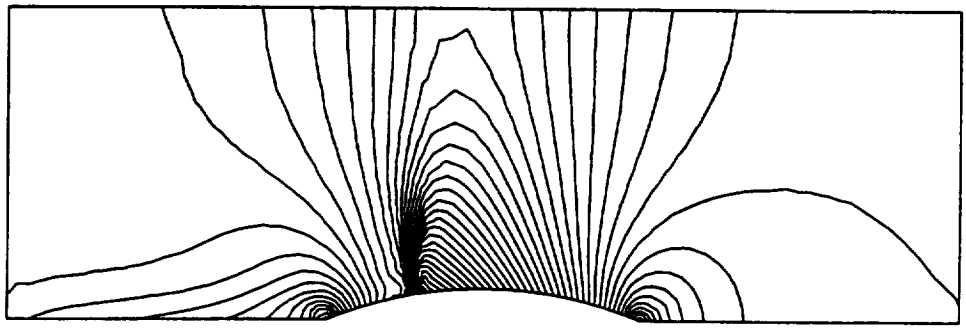

Figure 7.8 : Mach Number Contours for Transonic Flow through a Channel with a Bump — Flow from Right to Left using a Single Mesh

Figure 7.9 shows the Mach contours obtained from using two non-matching meshes with a vertical discontinuity for flow going from left to right, whereas Figure 7.10 indicates the same for flow from right to left. Figure 7.11 shows the Mach contours obtained on a mesh with a horizontal discontinuity, with flow from left to right.

In addition to the contour plots, the pressure coefficient C_p defined by

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho||U_\infty||^2}$$

is also plotted for points along the bottom wall of the channel for each case in Figures 7.12, 7.13 and 7.14 respectively.

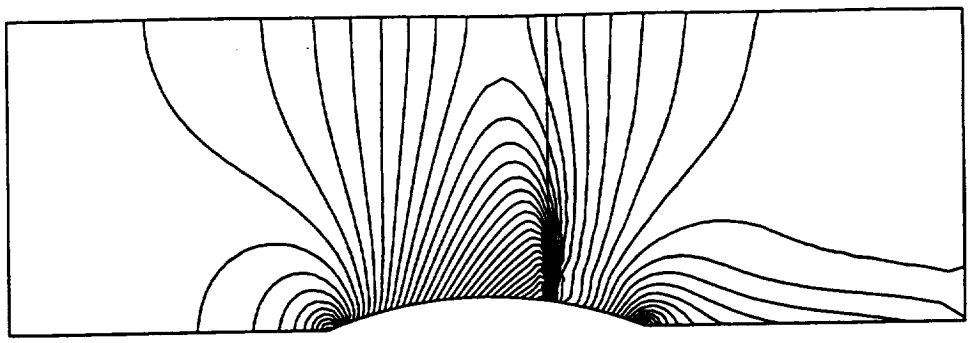

Figure 7.9 : Mach Number Contours for Transonic Flow through a Channel with a Bump — Flow from Left to Right using Two Non-matching Meshes with a Vertical Discontinuity

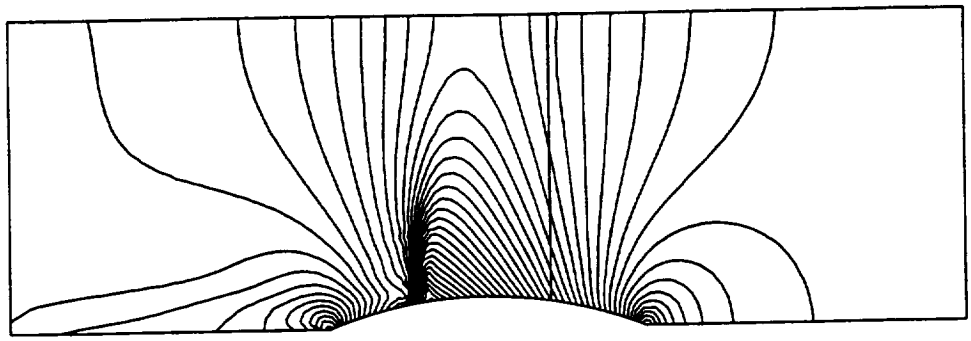

Figure 7.10 : Mach Number Contours for Transonic Flow through a Channel with a Bump — Flow from Right to Left using Two Non-matching Meshes with a Vertical Discontinuity

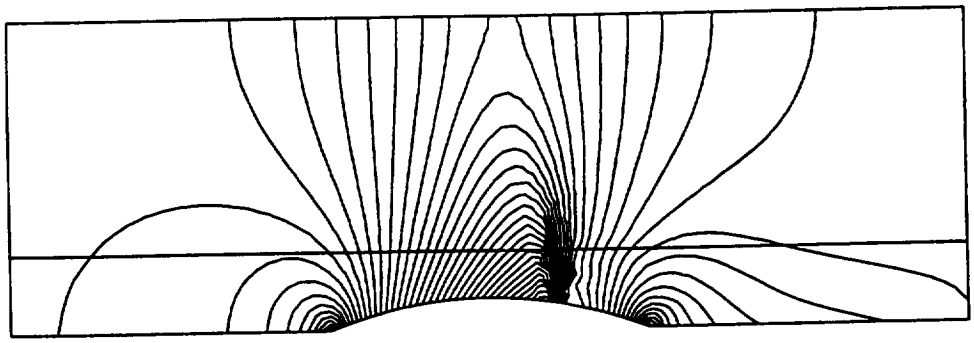

Figure 7.11 : Mach Number Contours for Transonic Flow through a Channel with a Bump — Flow from Left to Right using Two Non-matching Meshes with a Horizontal Discontinuity

From the figures, it can be concluded that results obtained from using non-matching meshes agree well with those obtained from single mesh calculations. The Mach contours are smooth and continuous across the mesh interfaces and both the magnitude and position of the shock are captured correctly in each case. The method developed is able to compute flows accurately, irrespective of the direction and subsonic/supersonic nature of flow.

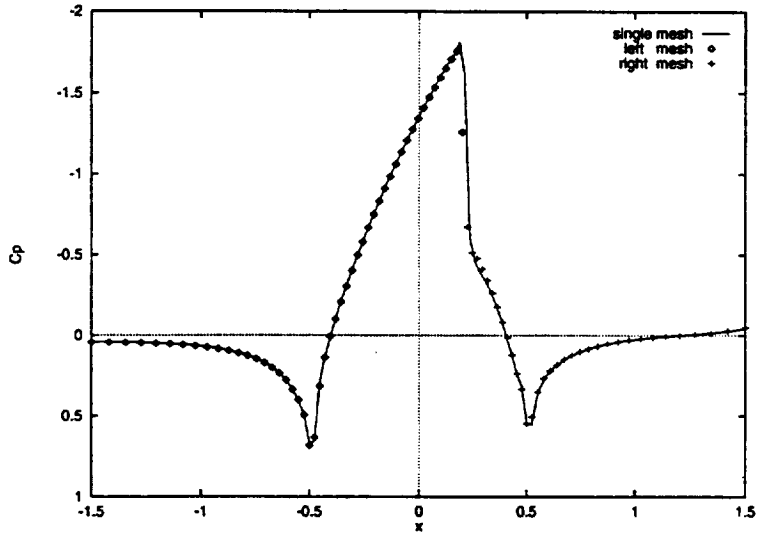

Figure 7.12 : Comparison of C_p Profiles on the Lower Wall of the Channel for Flow from Left to Right using Two Non-matching Meshes with a Vertical Discontinuity

7.3 The Shock Tube Problem

Once satisfactory results were obtained for the steady-state problems in Sections 7.1 and 7.2, SUM was tested on an classical unsteady problem to investigate the propagation of shocks and discontinuities across mesh boundaries with time. For this purpose, the shock tube problem proposed by Sod [58] was chosen.

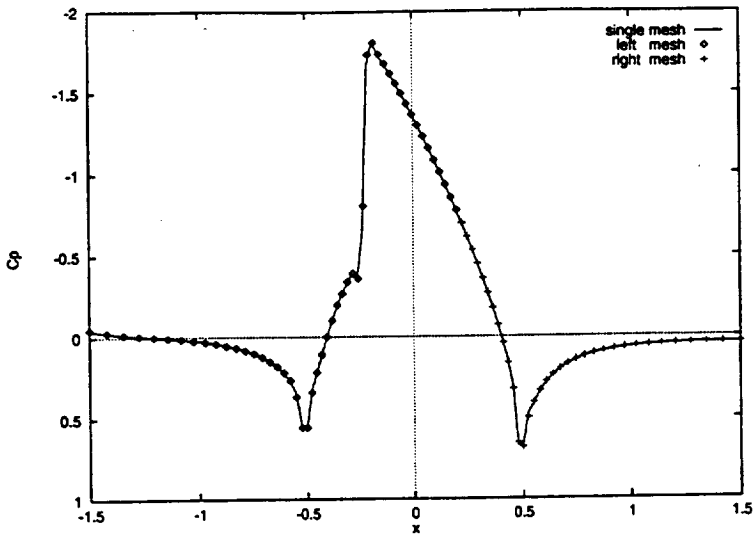

Figure 7.13 : Comparison of C_p Profiles on the Lower Wall of the Channel for Flow from Right to Left using Two Non-matching Meshes with a Vertical Discontinuity

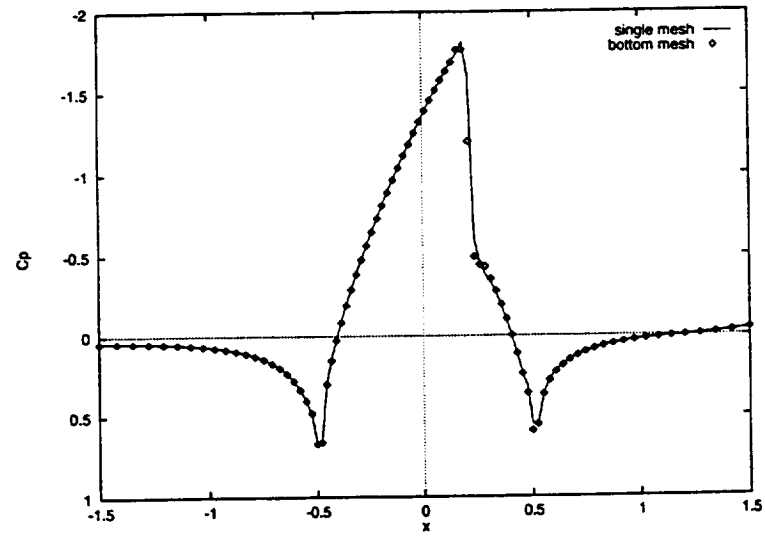

Figure 7.14 : Comparison of C_p Profiles on the Lower Wall of the Channel for Flow from Left to Right using Two Non-matching Meshes with a Horizontal Discontinuity

7.3.1 Problem Physics

The shock tube problem is designed to trace the development of shocks and other discontinuities from a contact discontinuity in the initial state for the Euler equations given by (4.2).

The shock tube is a 1×1 tube (in physical dimensions), closed at both ends with a diaphragm separating a region of high-pressure (p_4) gas on the left from a region of low-pressure (p_1) gas on the right. This setup and initial state is illustrated in Figure 7.15.

The diaphragm is broken. A shock wave then propagates into section 1 while an expansion wave propagates into section 4. This behavior is illustrated in Figure 7.16. As the normal shock wave propagates to the right, it increases the pressure behind it in region 2 and induces a mass motion in that region. The contact surface (interface between the region of high and low pressure) moves to the right with the same velocity as that of the mass motion in region 2. The expansion wave propagates to the left, smoothly and continuously decreasing the pressure in region 4 to the lower value p_3 behind the expansion wave.

For simulations reported here, the following initial conditions are used :

$$u_4 = u_1 = 0$$

$$p_4 = 1.0 \qquad p_1 = 0.1$$

$$\rho_4 = 1.0 \qquad \rho_1 = 0.125$$

An analytical solution is known in which the distribution of the pressure, density and velocity is known as the function of the initial pressure ratio (p_4/p_1) and the position of a point in the shock tube with respect to time [4].

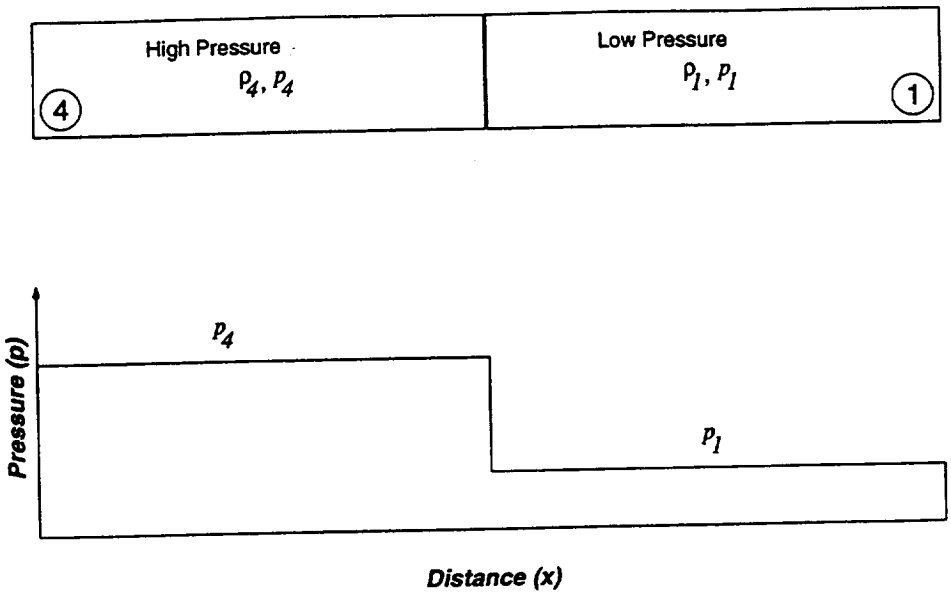

Figure 7.15 : Initial State for the Shock Tube Problem

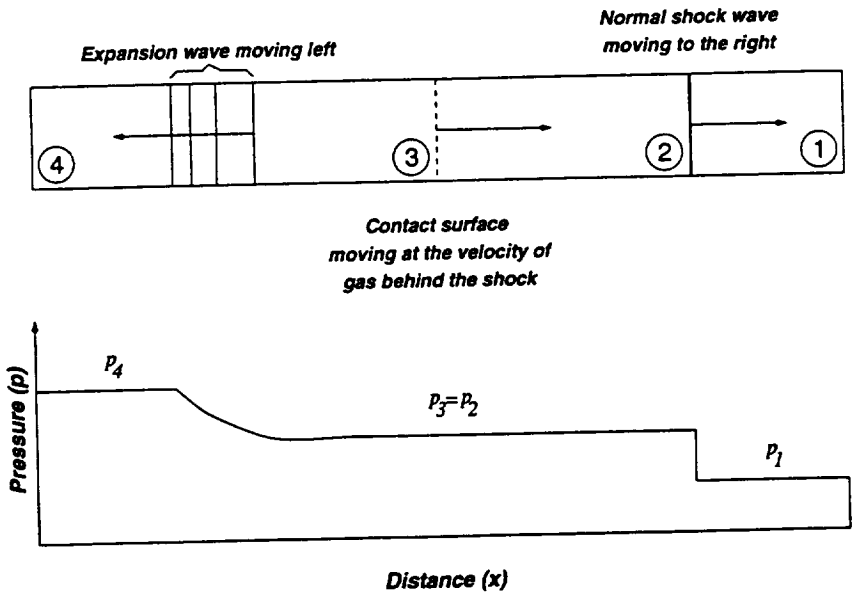

Figure 7.16 : Flow in the Shock Tube after the Diaphragm is Broken

7.3.2 Motivation

The shock tube problem presents an opportunity to investigate the simultaneous propagation of a shock, a contact discontinuity and an expansion fan in a simple computational setup. Because an analytical solution is available, it is easier to assess the accuracy of any method. Consequently, this problem is used often to test new techniques in computational fluid dynamics.

7.3.3 Modeling and Mesh Generation

The shock tube is modeled as 1×1 box in physical dimensions. The slip boundary condition is imposed at the wall boundary. The patched mesh in this case consists of 2 algebraically generated structured meshes with a discontinuity in the meshes at the center of the tube as seen in Figure 7.17.

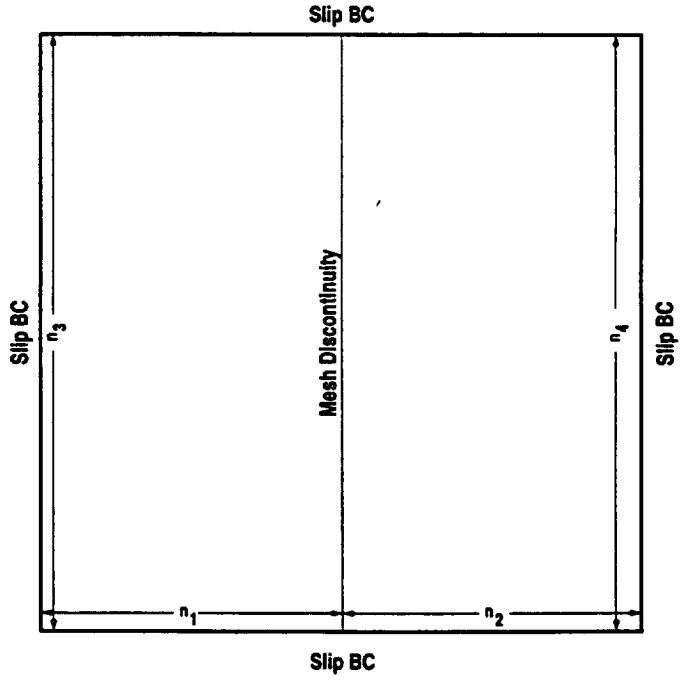

Figure 7.17 : Mesh Model for the Shock Tube Problem

Two configurations were tested : (a) One in which the pressure discontinuity is normal to the mesh discontinuity and, (b) one in which the pressure discontinuity is parallel to the mesh discontinuity. Case (a) tests the ability of the solver to propagate shocks and discontinuities smoothly while case (b) assesses the accuracy of the projection scheme.

7.3.4 Results and Discussion

Figure 7.18 shows the density distribution for a patched mesh computation, a single mesh computation and the analytical solution for case (a). From this it is evident that values for single and patched mesh computations match closely which in turn are in fair agreement with the exact solution. This illustrates the ability of the solver to capture shocks and discontinuities accurately.

Figures 7.19, 7.20 and 7.21 present results obtained for case (b). From Figures 7.19 and 7.20 it is seen that results on either side of the mesh discontinuity match well with the exact solution. Figure 7.21 verifies that the used of the mortar method results in accurate transfer of information from one mesh to another.

7.4 Idealized Rotor-Stator Calculation

The motivation for current research to develop a method for patched unstructured mesh computation was to enable it to handle slipping meshes as in the case of rotor-stator interaction. This section details the calculations performed on a simple, idealized stage comprising a rotor and a stator. These are similar to the simulations performed in [44] and [52].

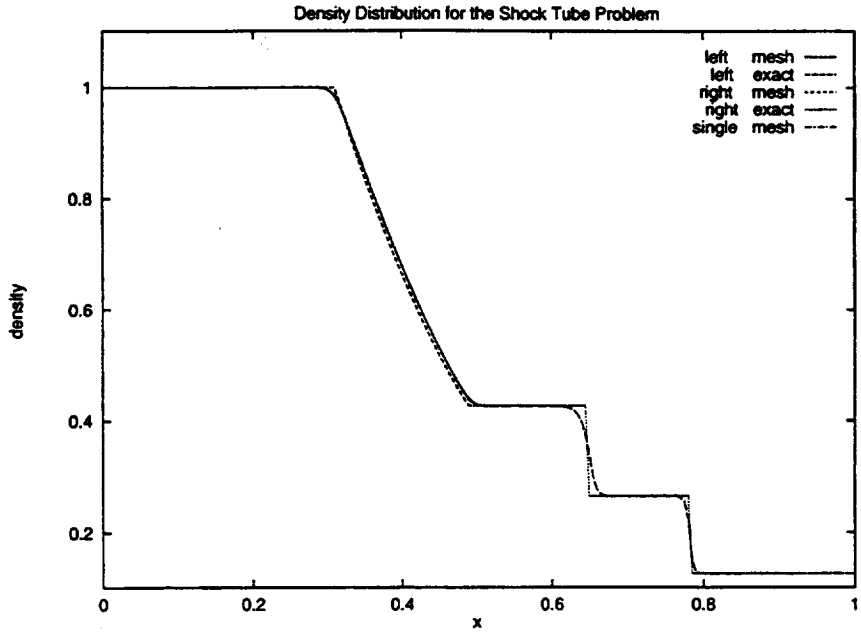

Figure 7.18 : Density Distribution for the Shock Tube Problem, Case (a)

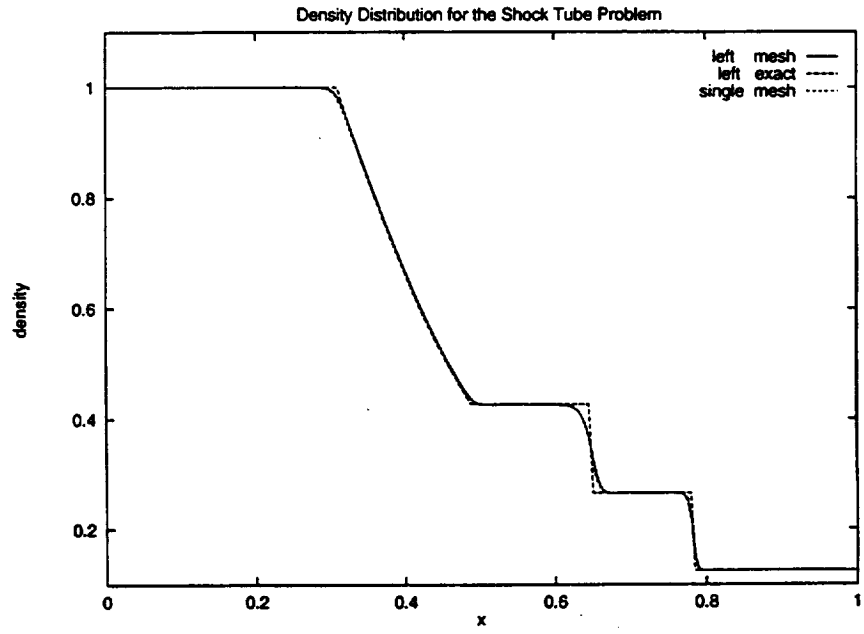

Figure 7.19 : Density Distribution for the Shock Tube Problem, Case (b), Comparison of Solutions for the Left Mesh

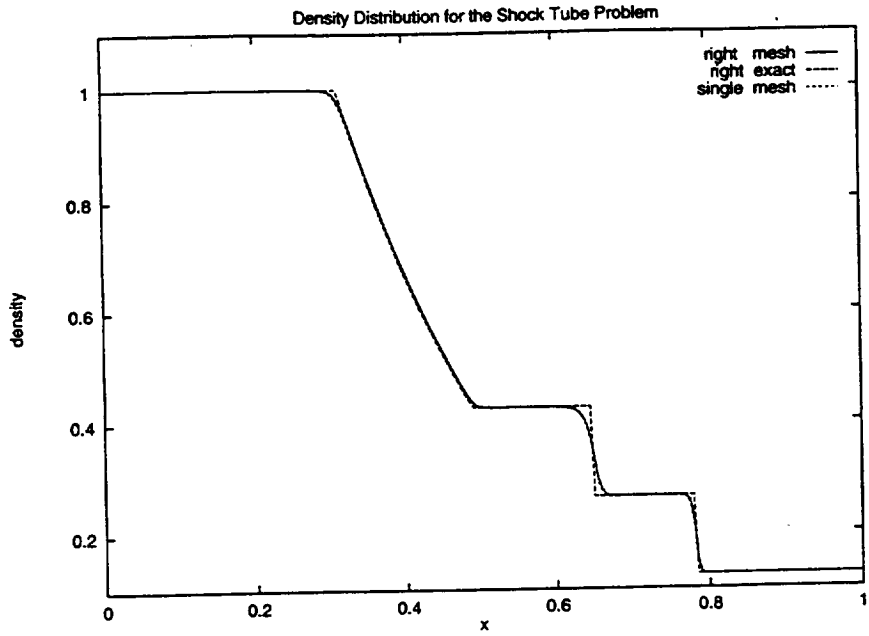

Figure 7.20 : Density Distribution for the Shock Tube Problem, Case (b), Comparison of Solutions for the Right Mesh

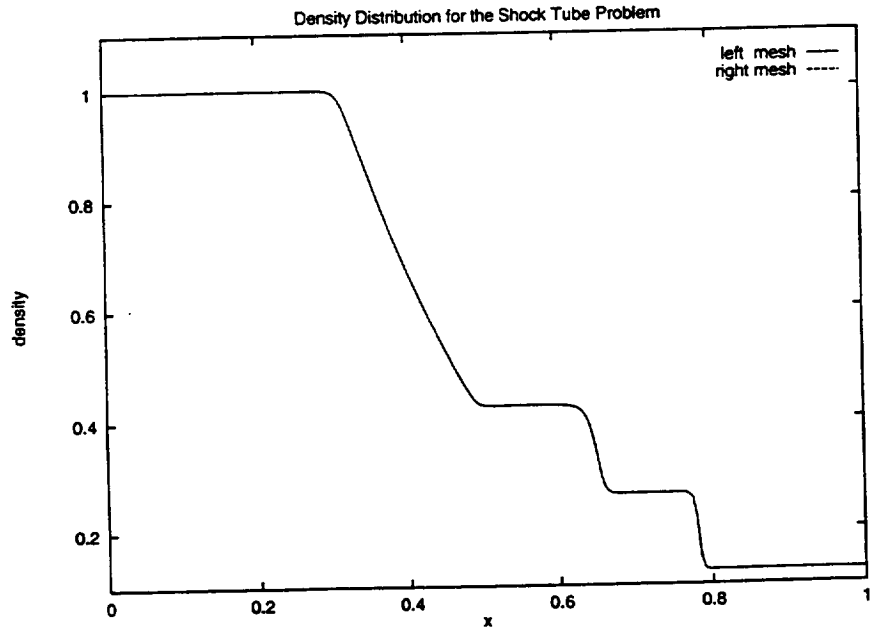

Figure 7.21 : Density Distribution for the Shock Tube Problem, Case (b), Comparison of Solutions on the Interface

7.4.1 Mesh Generation and Modeling

As shown in Figure 7.22, the rotor and stator blades were simplified and represented in two-dimensions as airfoils made of two circular arcs. The following parameters were used in mesh generation :

1. l_a : The chord length of the airfoil.
2. l_s : The spacing between the tail of the fore airfoil and the head of the aft airfoil.
3. t_a : Maximum airfoil thickness.
4. l_l and l_l : Extension of the fluid mesh upstream of the fore airfoil and downstream of the aft airfoil.
5. l_c : Distance of separation between two airfoils in the circumferential direction.

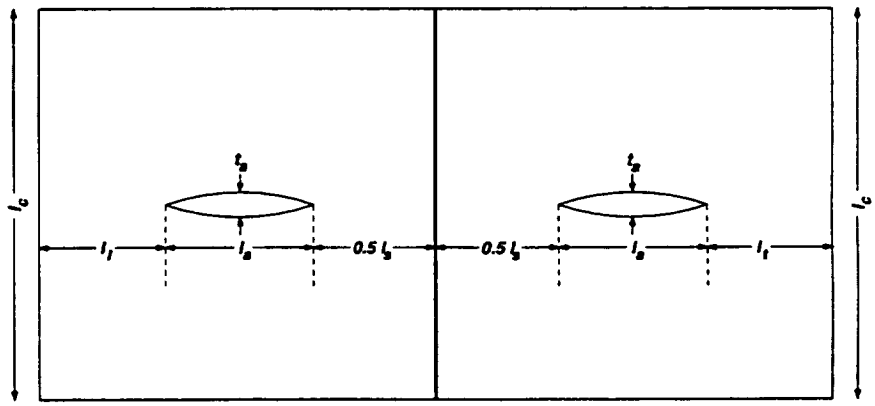

Figure 7.22 : Mesh Generation Parameters for the Idealized Rotor-Stator Calculation

In order to simulate interaction between a rotor and a stator, the fore airfoil is moved downward with a prescribed velocity.

Two models were used in the calculations :

1. **Case (a)** : This corresponds approximately to the simulation per-

formed in [52] and has the following parameters :

$$l_a = l_c = 1.0$$

$$l_l = l_t = 0.25$$

$$l_s = 0.50$$

$$t_a = 0.06$$

2. **Case (b)** : This is similar to the calculation performed in [44] and has the following parameters :

$$l_a = l_c = 1.0$$

$$l_l = l_t = 0.25$$

$$l_s = 0.20$$

$$t_a = 0.06$$

Thus, the only difference between case (a) and case (b) is the reduced spacing between the airfoils in the axial direction. This would highlight the pronounced effect this spacing has on overall engine performance.

In each case, solutions were obtained with both first- and second-order accuracy in space. A simulation consisted of two parts : (1) obtaining the steady state solution for flow around the stationary airfoil from an uniform Mach flow of $M = 1.5$, and (2) computation of the unsteady response by the downward motion of the fore airfoil till the desired end of computations. In both cases, the airfoil is moved with a downward velocity of 0.1M with respect to the uniform flow.

7.4.2 Results and Discussion

Results with first order accuracy for cases (a) and (b) are compared with those in [52] and [44], while those with second order accuracy are presented to highlight benefits from increased accuracy.

The following general trends can be observed.

1. An intricate pattern of shocks and expansion fans is observed at steady state. Oblique shocks are attached to the leading and trailing edges of both the fore and aft airfoils and these are slightly weakened by the expansion waves that emanate from the surfaces of the airfoils.

   Figure 7.23 shows the pressure contours at steady state for case (a) and these can be compared with Figures 7.24 and 7.41 for results obtained with first and second order accuracy respectively. No results are available at steady state for case (b), however, Figures 7.50 and 7.68 show the pressure contours at steady state by the current method with first and second order accuracy respectively.

2. After the initial transients subside, the flow pattern becomes periodic in time as evidenced by the pressure history at midchord on the lower and upper surfaces of the aft airfoils in each case.

   Figure 7.25 shows the pressure history at midchord on the lower surface of the aft airfoil from [52] for case (a). These results can be compared with Figures 7.26 and 7.42 which are obtained with first and second order accuracy, respectively.

   Figure 7.27 shows the pressure history at midchord on the upper surface of the aft airfoil from [52] for case (a). These results can be compared with Figures 7.28 and 7.43 which are obtained with first and second order accuracy, respectively.

In either case, there is a phase shift between the pressure history at midchord on the lower and upper surfaces of the airfoil. Also, the mean pressure value on the upper surface is higher than at steady state while that on the lower surface is lower than the steady state pressure.

3. The downward motion of the forward airfoil results in an effective angle of attack which in turn results in the creation of an attached oblique shock at the leading edge on the lower side and a weak expansion fan on the upper side at the leading edge of the first airfoil. Another attached oblique shock is also evident at the trailing edge of the fore airfoil. At the start of a cycle, the shock associated with the leading edge of the second airfoil is detached, and is seen impinging on the surface of the adjacent airfoil.

There is an area of interaction between the trailing edge shock of the first airfoil and the leading edge shock of the second. This area of interaction moves downward as the first airfoil moves downward. This causes the leading edge shock of the aft airfoil to attach and detach itself periodically as is seen in the sequence of contour plots in the subsequent pages. In the first three plots, each plot corresponding to a fifth of a cycle, the process of attachment becomes evident. The shock then starts to detach itself and finally returns to its initial state of detachment at the end of the cycle.

Figures 7.29, 7.31, 7.33, 7.35, 7.37 and 7.39 show the pressure contours at different stages of a cycle for case (a) from [52]. These can be compared with Figures 7.30, 7.32, 7.34, 7.36, 7.38 and 7.40 which are obtained with first order accuracy and Figures 7.44, 7.45, 7.46, 7.47, 7.48 and 7.49 which are obtained with second order accuracy.

Figures 7.54, 7.56, 7.58, 7.60, 7.62 and 7.64 show the pressure contours at different stages of a cycle for case (b) from [44]. These can be compared with Figures 7.55, 7.57, 7.59, 7.61, 7.63 and 7.65 which are obtained with first order accuracy and Figures 7.69, 7.70, 7.71, 7.72, 7.73 and 7.74 which are obtained with second order accuracy.

7.4.2.1 Conclusions

The key observation to be made from these plots is that the contour lines across the mesh discontinuity are smooth and continuous. This validates the ability of SUM to handle discontinuities correctly. In general, results obtained with SUM match well with those available in literature. Also significant is the fact that contour plots at the end of a cycle are almost exactly identical to those at the end of a cycle, indicating SUM's temporal accuracy.

Calculations performed with second order accuracy show a wider variation in midchord pressures both on the upper and lower surfaces of the aft airfoil. Shocks captured in this case are sharper, and more cycles are needed in order to subside the transients.

The effects of interaction between the fore and the aft airfoil become more pronounced as the gap between these two airfoils is reduced. This is corroborated by examining the corresponding contour plots at each stage of the cycle for cases (a) and (b).

7.5 Summary

1. SUM was tested on several benchmark problems to assess its ability to accurately simulate flows having strong gradients with shocks and discontinuities.

2. It is observed that SUM allows the smooth and undisturbed passage of shocks and discontinuities across mesh boundaries, for both transonic and supersonic flow, irrepsective of flow direction. Both the position and the strength of shocks are captured satisfactorily.
3. Results obtained with SUM for the case of rotor-stator interaction which involves simulation of flows with slipping meshes highlight its ability to handle complex flows on moving, non-matching meshes.

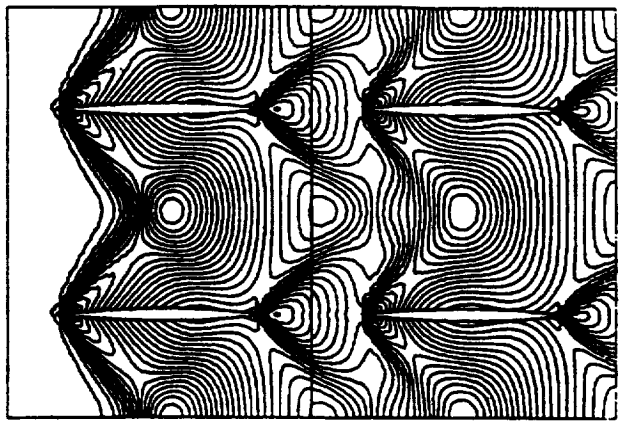

Figure 7.23 : Pressure Contours at Steady-State for Case (a) from [52]

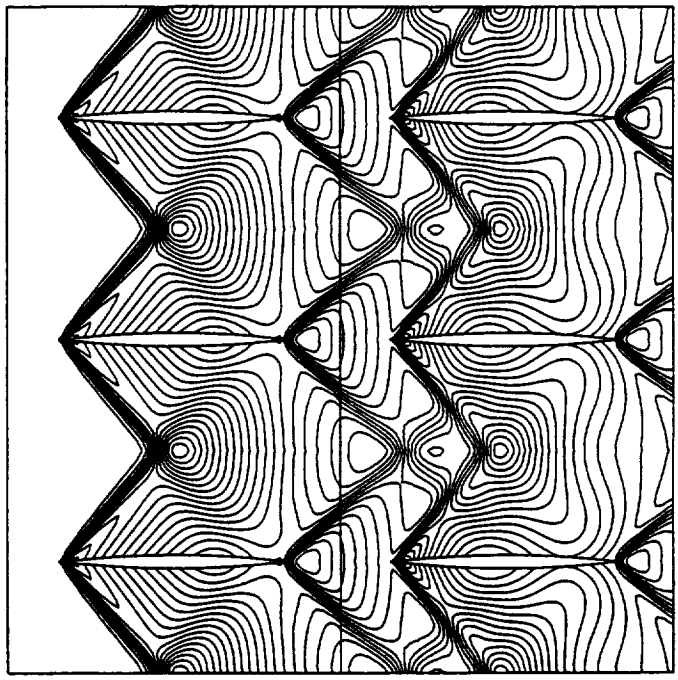

Figure 7.24 : Pressure Contours at Steady-State for Case (a) with First Order Accuracy

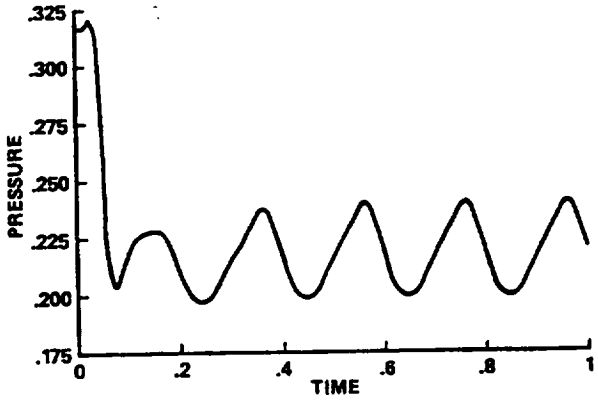

Figure 7.25 : Pressure History at Midchord on the Lower Surface of the Aft Airfoil for Case (a) from [52]

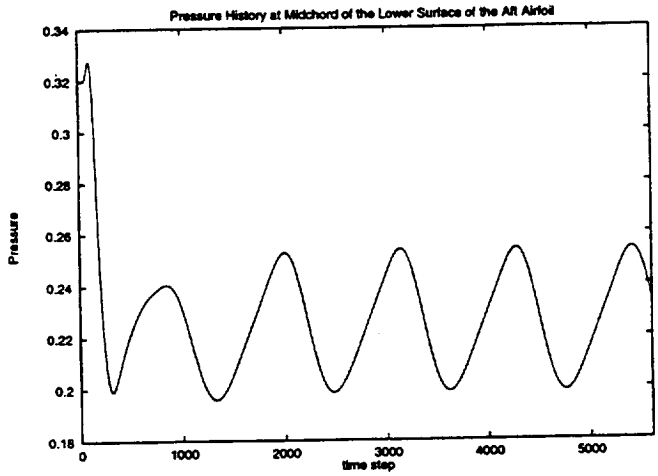

Figure 7.26 : Pressure History at Midchord on the Lower Surface of the Aft Airfoil for Case (a) with First Order Accuracy

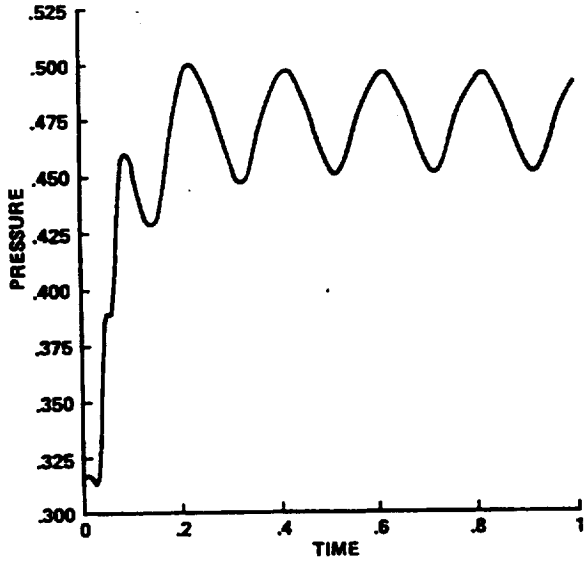

Figure 7.27 : Pressure History at Midchord on the Upper Surface of the Aft Airfoil for Case (a) from [52]

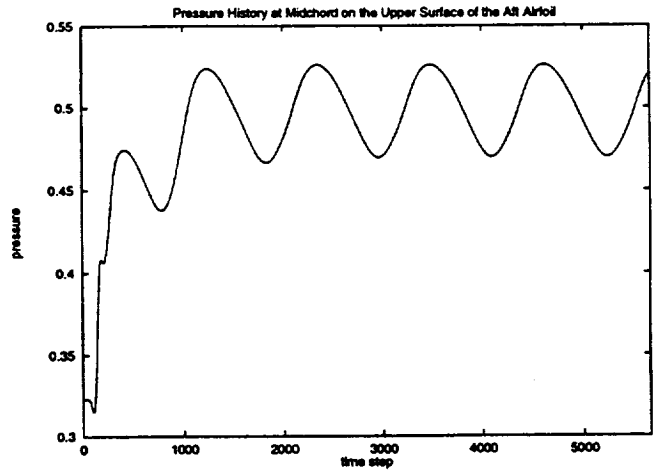

Figure 7.28 : Pressure History at Midchord on the Upper Surface of the Aft Airfoil for Case (a) with First Order Accuracy

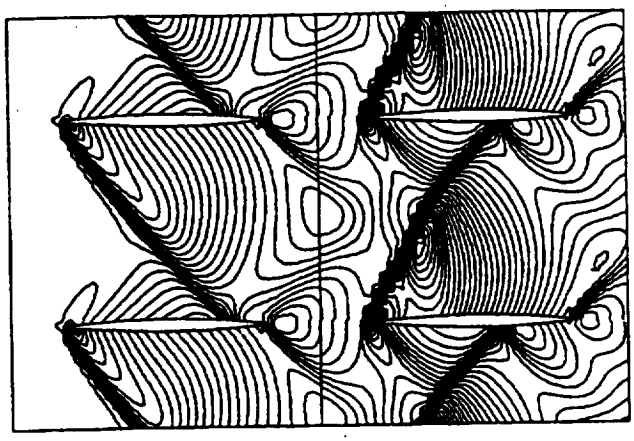

Figure 7.29 : Pressure Contours for Case (a) and the end of 4.0 cycles from [52]

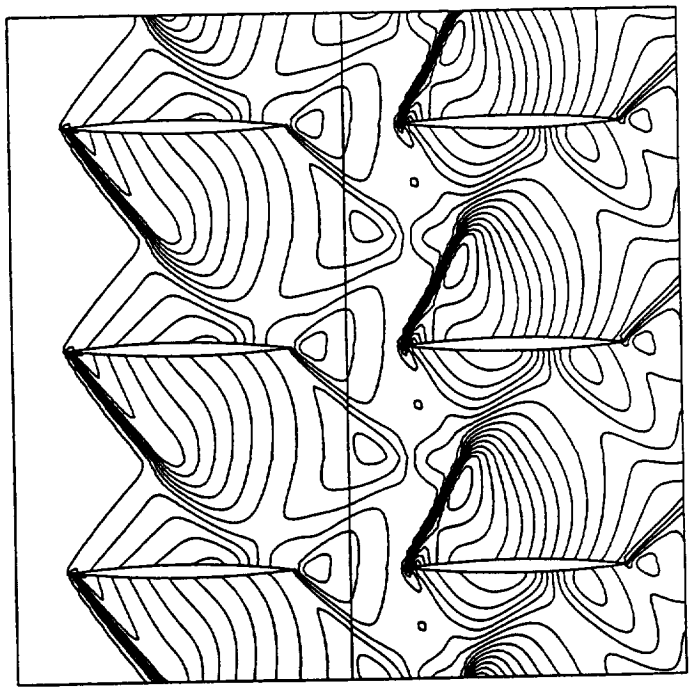

Figure 7.30 : Pressure Contours for Case (a) and the end of 4.0 cycles with First Order Accuracy

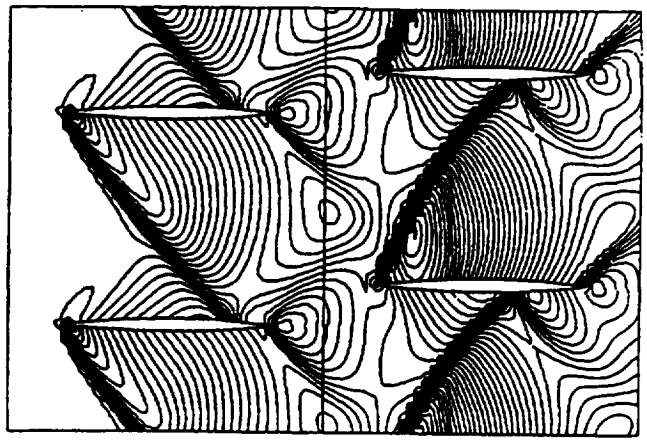

Figure 7.31 : Pressure Contours for Case (a) and the end of 4.2 cycles from [52]

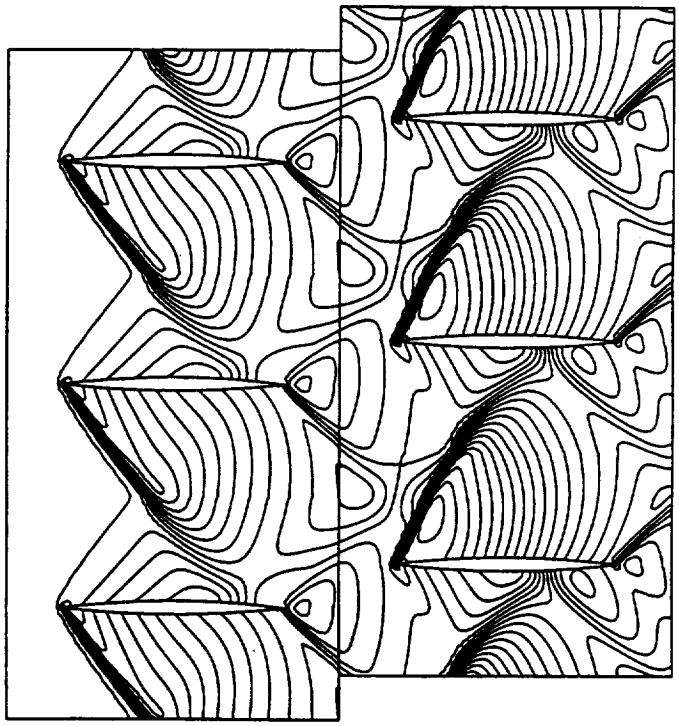

Figure 7.32 : Pressure Contours for Case (a) and the end of 4.2 cycles with First Order Accuracy

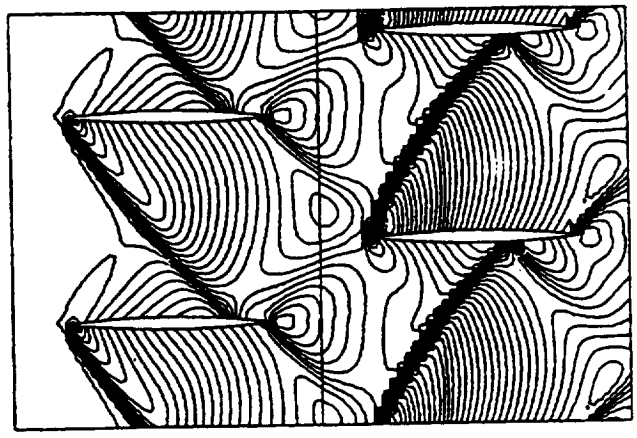

Figure 7.33 : Pressure Contours for Case (a) and the end of 4.4 cycles from [52]

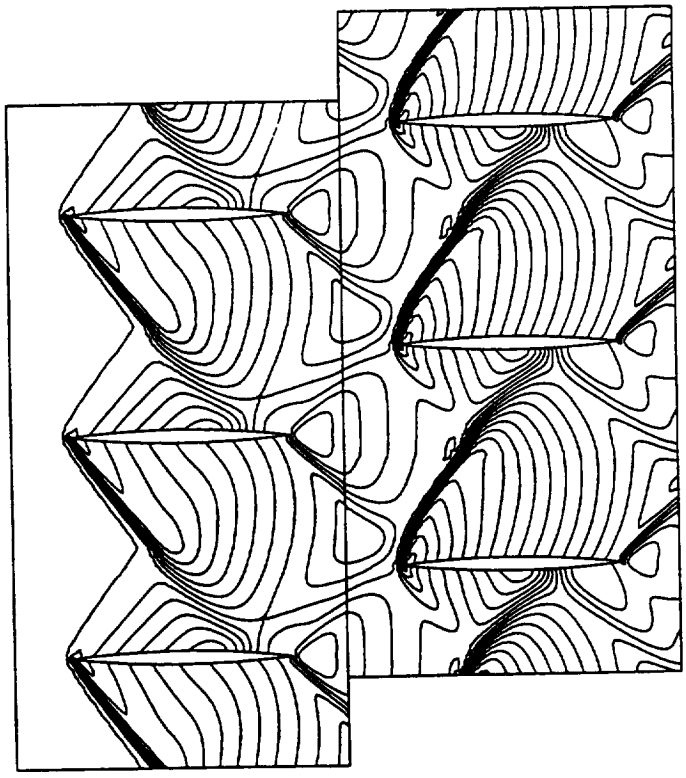

Figure 7.34 : Pressure Contours for Case (a) and the end of 4.4 cycles with First Order Accuracy

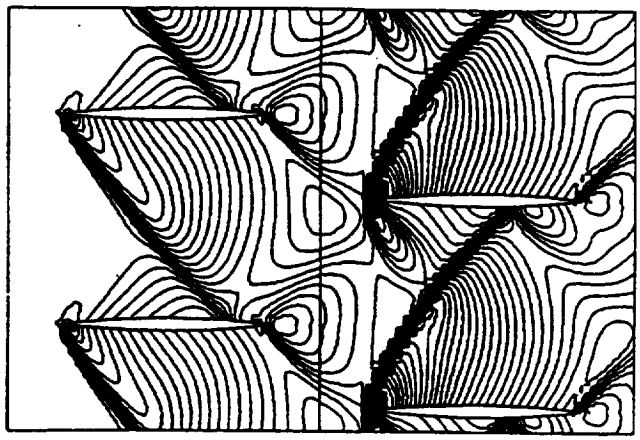

Figure 7.35 : Pressure Contours for Case (a) and the end of 4.6 cycles from [52]

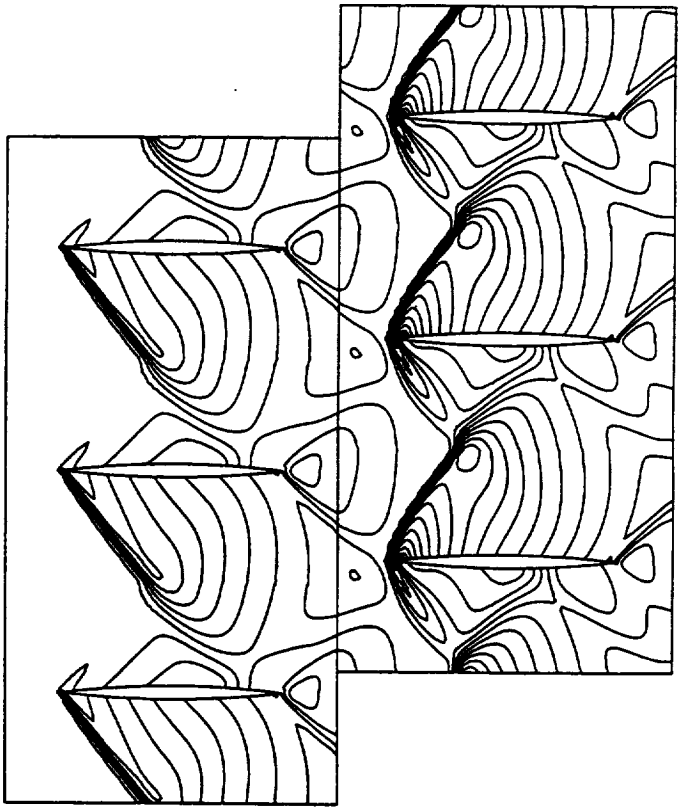

Figure 7.36 : Pressure Contours for Case (a) and the end of 4.6 cycles with First Order Accuracy

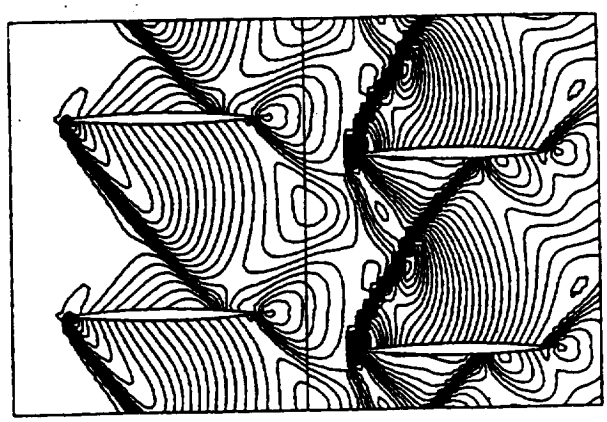

Figure 7.37 : Pressure Contours for Case (a) and the end of 4.8 cycles from [52]

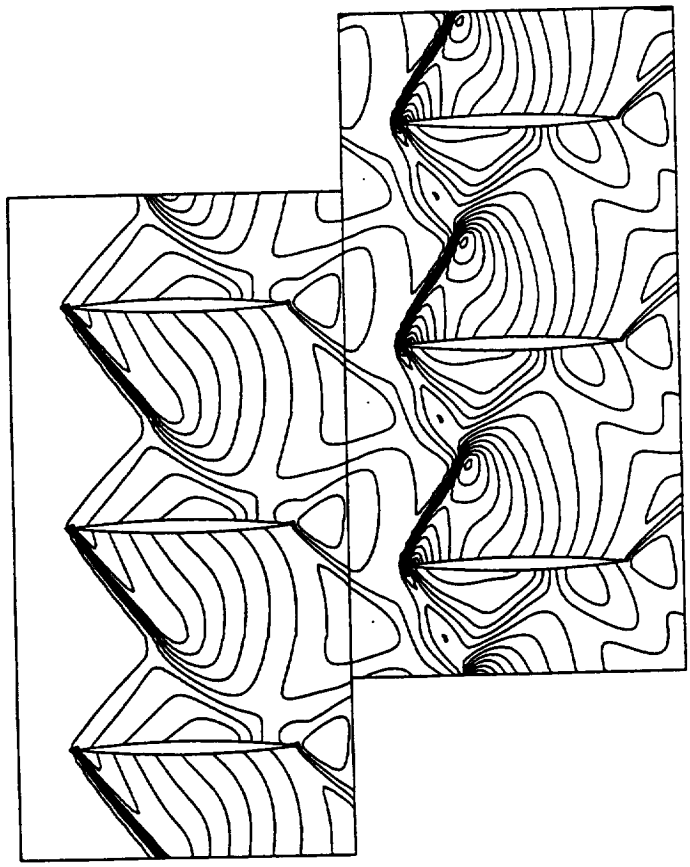

Figure 7.38 : Pressure Contours for Case (a) and the end of 4.8 cycles with First Order Accuracy

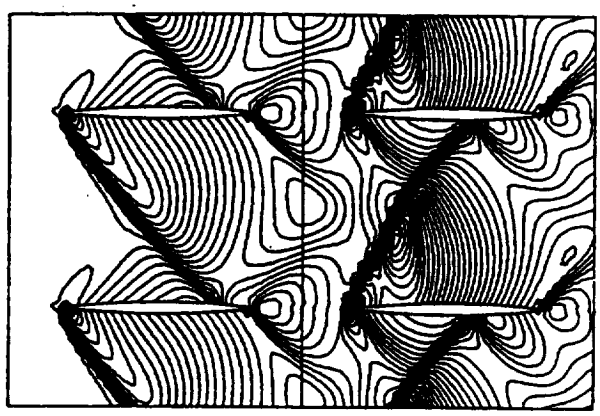

Figure 7.39 : Pressure Contours for Case (a) and the end of 5.0 cycles from [52]

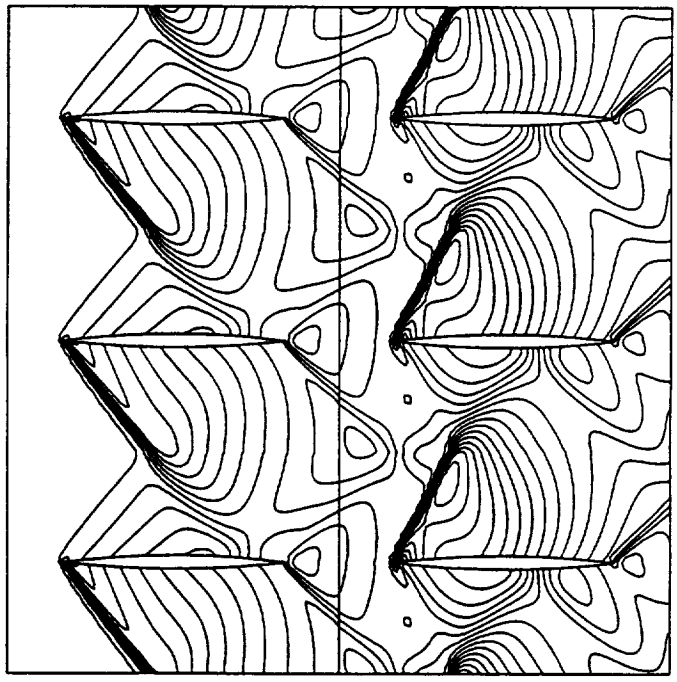

Figure 7.40 : Pressure Contours for Case (a) and the end of 5.0 cycles with First Order Accuracy

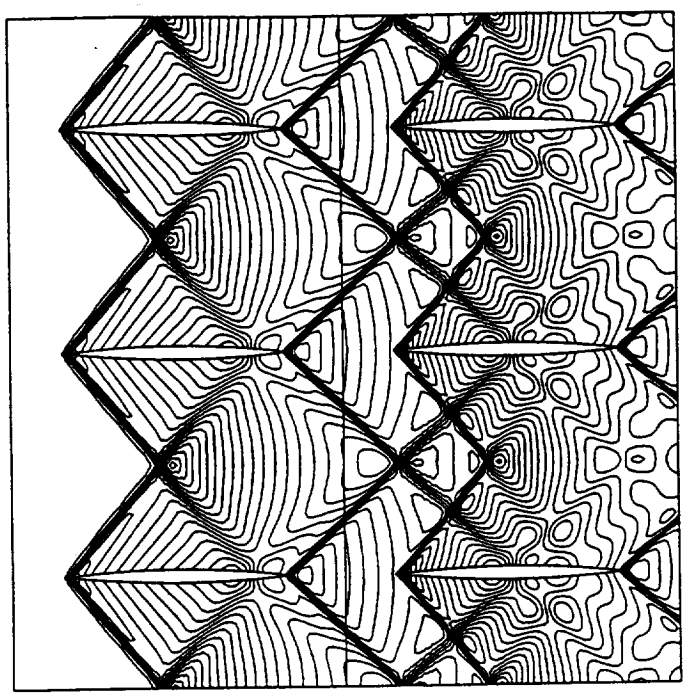

Figure 7.41 : Pressure Contours at Steady-State for Case (a) with Second Order Accuracy

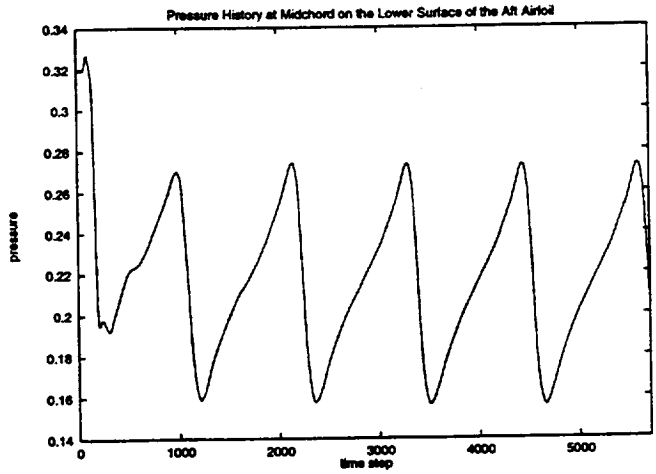

Figure 7.42 : Pressure History at Midchord on the Lower Surface of the Aft Airfoil for Case (a) with Second Order Accuracy

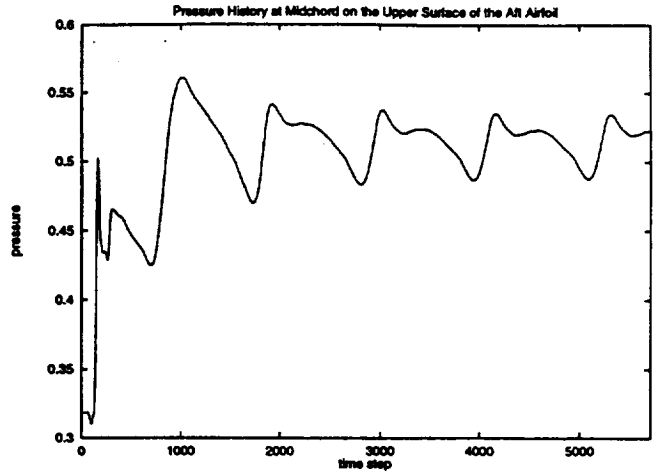

Figure 7.43 : Pressure History at Midchord on the Upper Surface of the Aft Airfoil for Case (a) with Second Order Accuracy

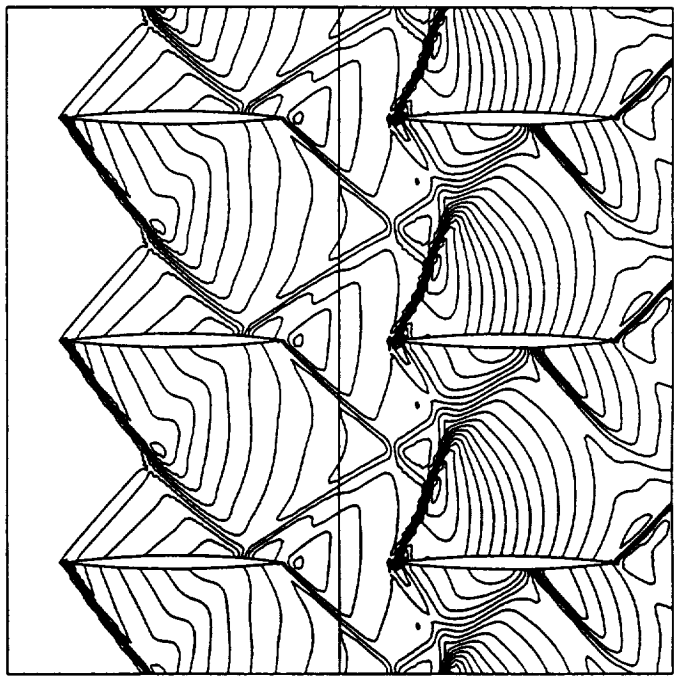

Figure 7.44 : Pressure Contours for Case (a) and the end of 5.0 cycles with Second Order Accuracy

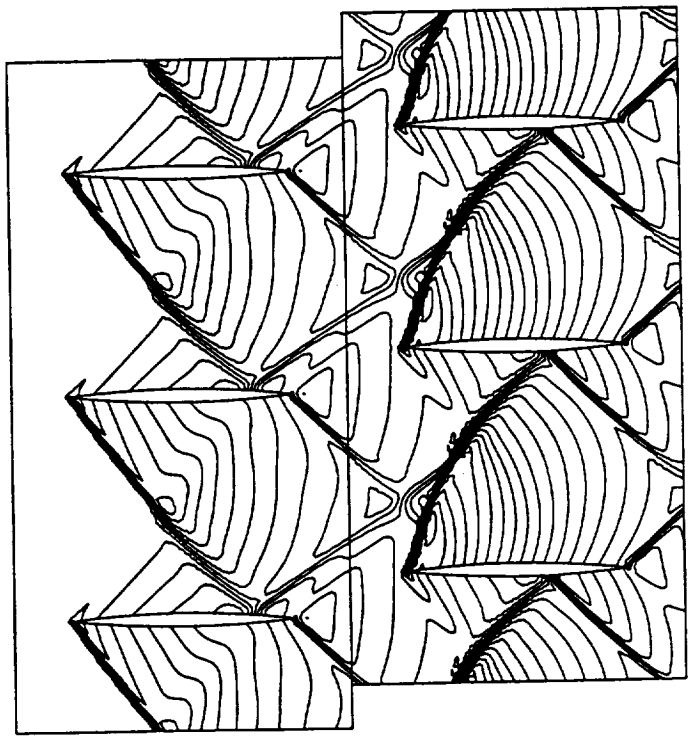

Figure 7.45 : Pressure Contours for Case (a) and the end of 5.2 cycles with Second Order Accuracy

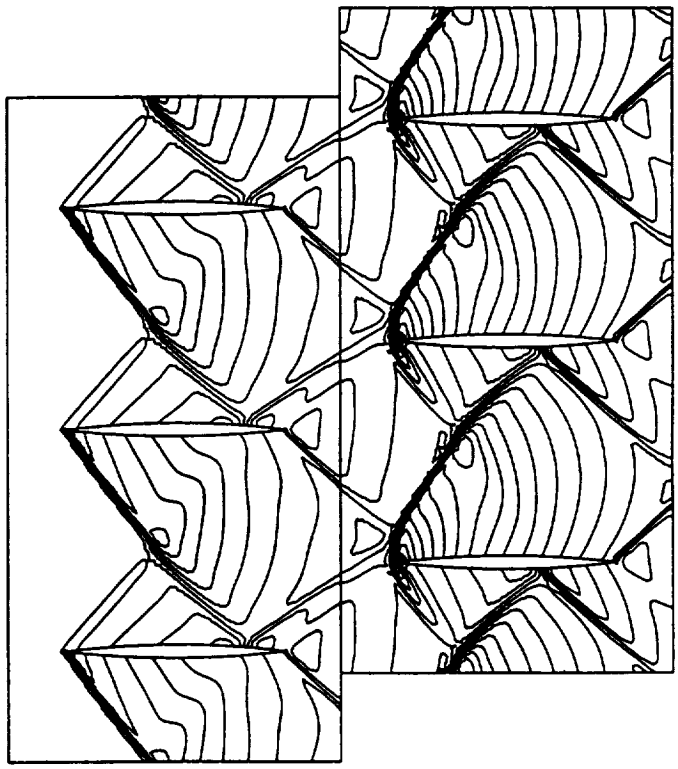

Figure 7.46 : Pressure Contours for Case (a) and the end of 5.4 cycles with Second Order Accuracy

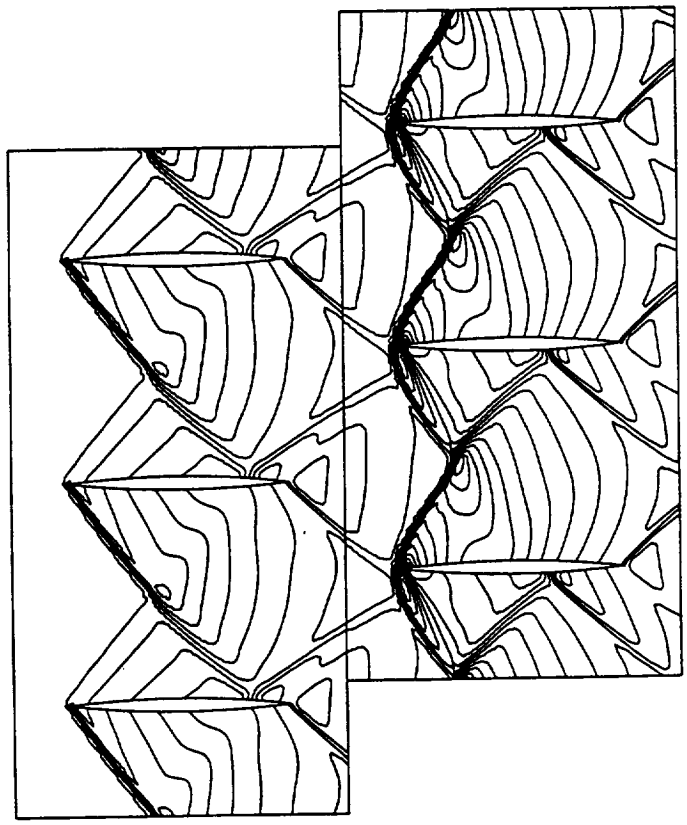

Figure 7.47 : Pressure Contours for Case (a) and the end of 5.6 cycles with Second Order Accuracy

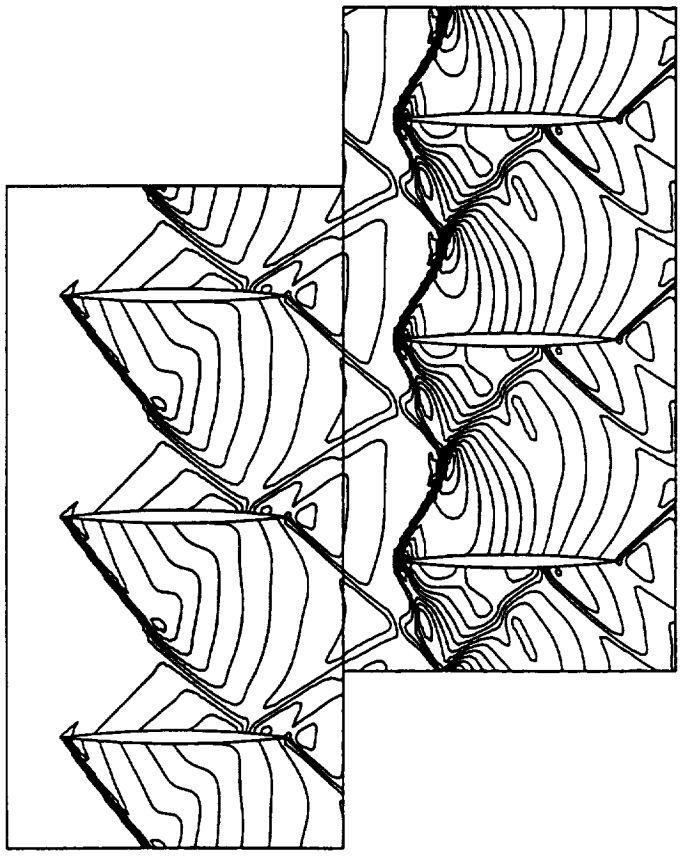

Figure 7.48 : Pressure Contours for Case (a) and the end of 5.8 cycles with Second Order Accuracy

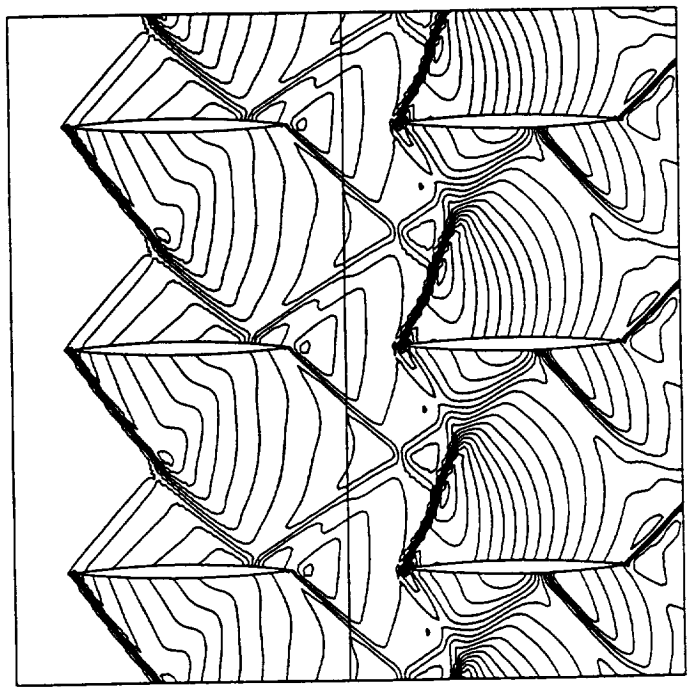

Figure 7.49 : Pressure Contours for Case (a) and the end of 6.0 cycles with Second Order Accuracy

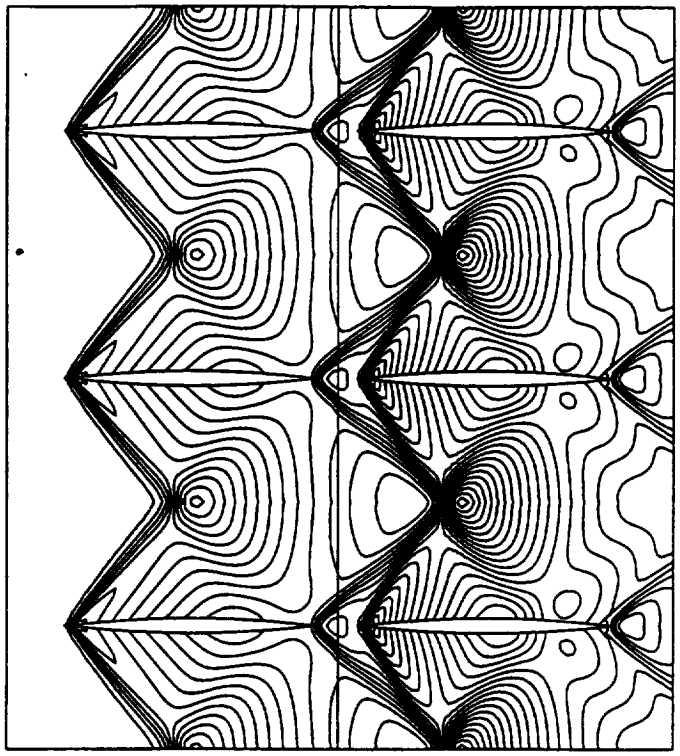

Figure 7.50 : Pressure Contours at Steady-State for Case (b) with First Order Accuracy

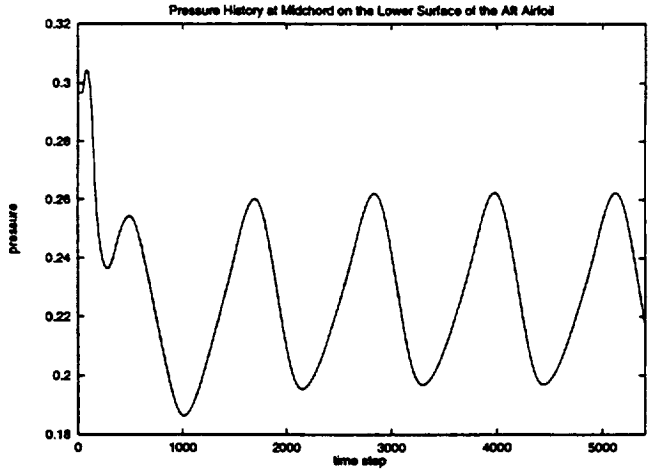

Figure 7.51 : Pressure History at Midchord on the Lower Surface of the Aft Airfoil for Case (b) with First Order Accuracy

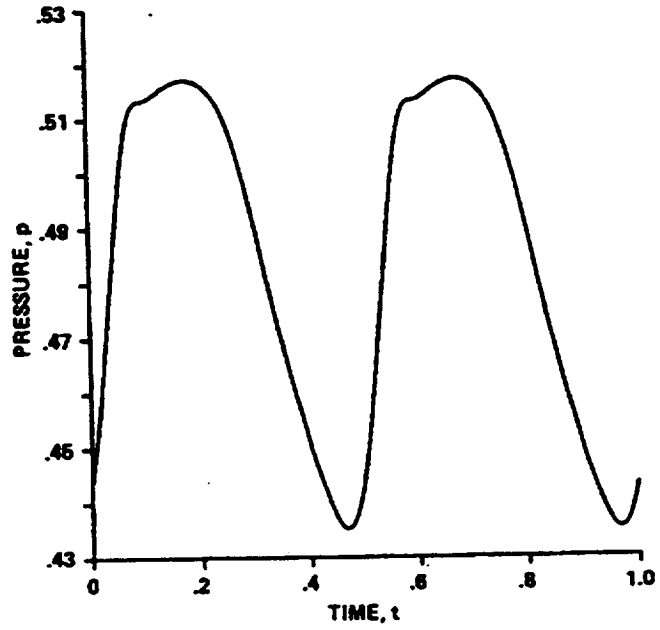

Figure 7.52 : Pressure History at Midchord on the Upper Surface of the Aft Airfoil for Case (b) from [44]

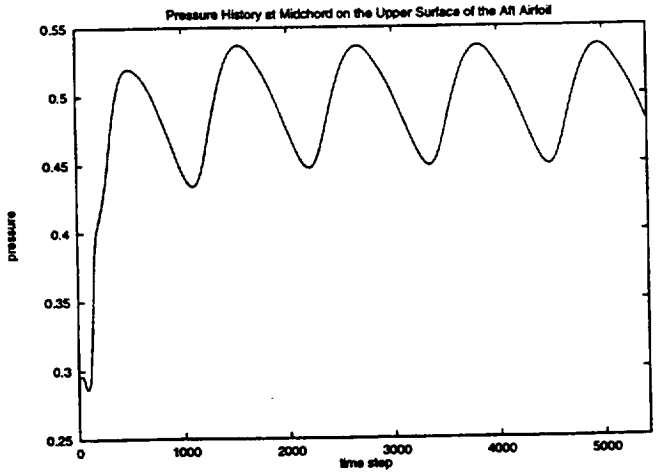

Figure 7.53 : Pressure History at Midchord on the Upper Surface of the Aft Airfoil for Case (b) with First Order Accuracy

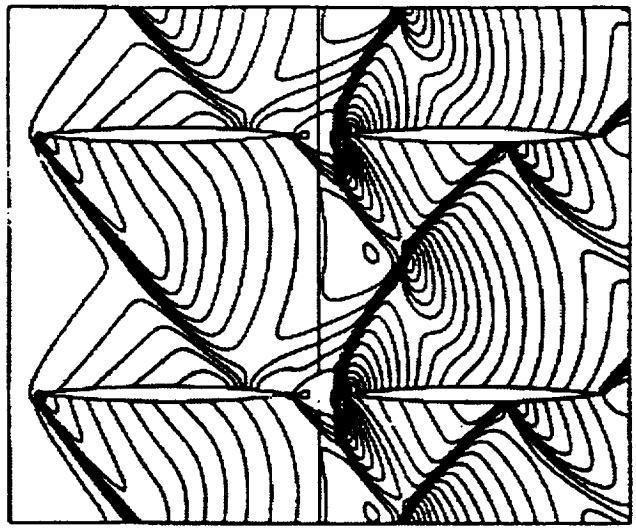

Figure 7.54 : Pressure Contours for Case (b) and the end of 4.0 cycles from [44]

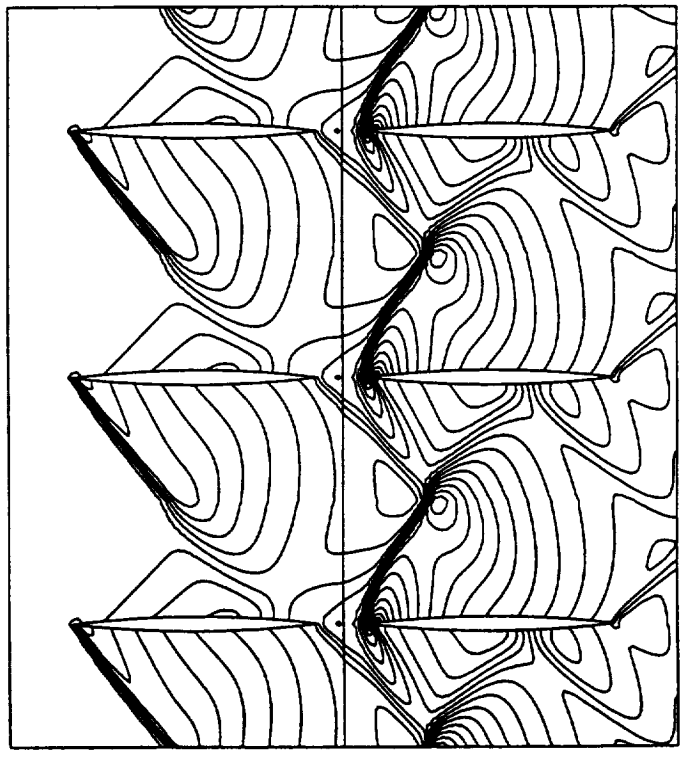

Figure 7.55 : Pressure Contours for Case (b) and the end of 5.0 cycles with First Order Accuracy

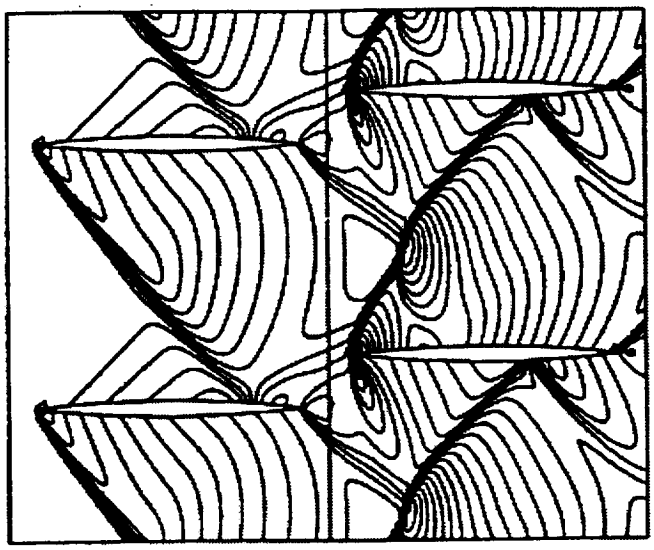

Figure 7.56 : Pressure Contours for Case (b) and the end of 4.2 cycles from [44]

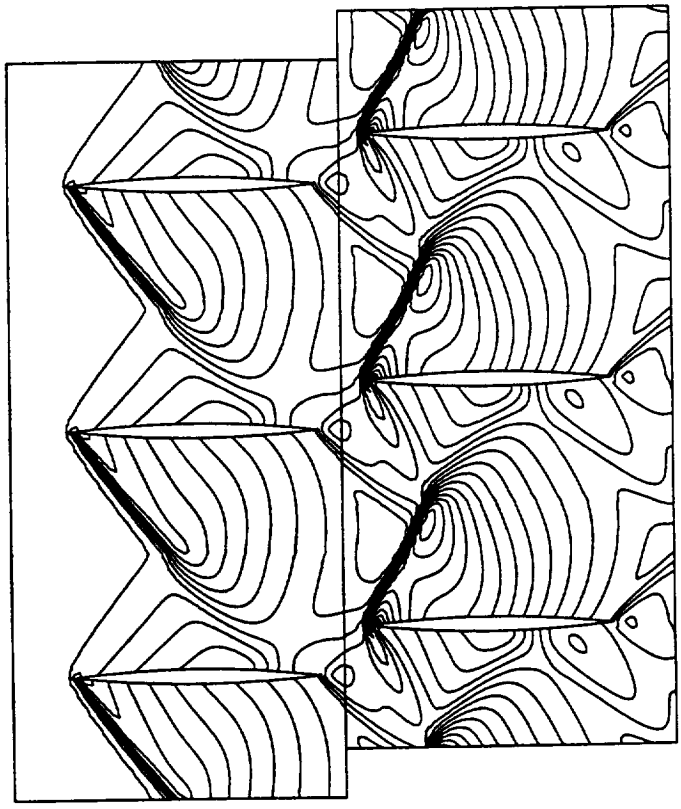

Figure 7.57 : Pressure Contours for Case (b) and the end of 5.2 cycles with First Order Accuracy

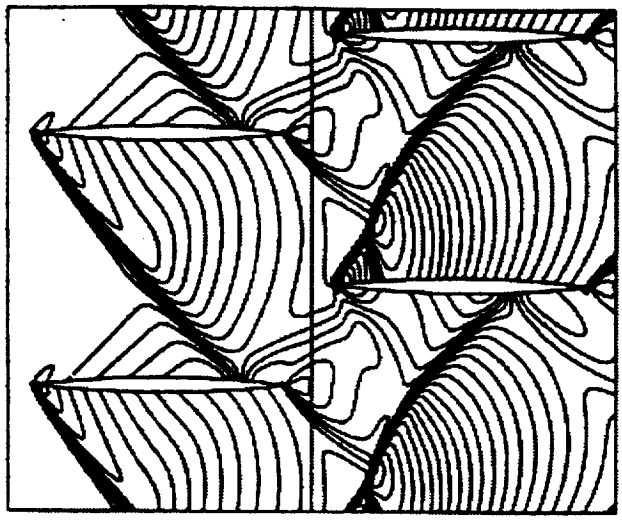

Figure 7.58 : Pressure Contours for Case (b) and the end of 4.4 cycles from [44]

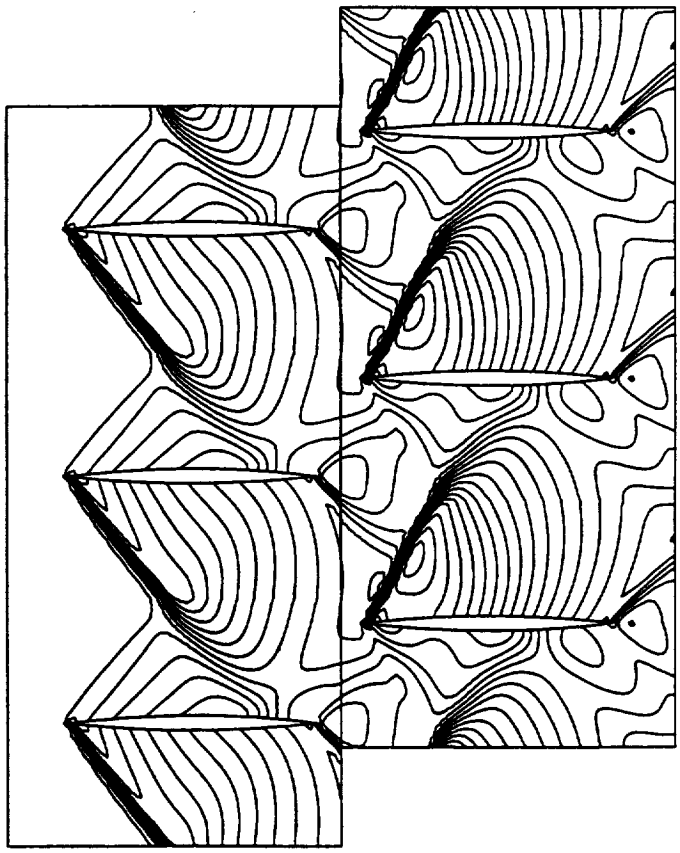

Figure 7.59 : Pressure Contours for Case (b) and the end of 5.4 cycles with First Order Accuracy

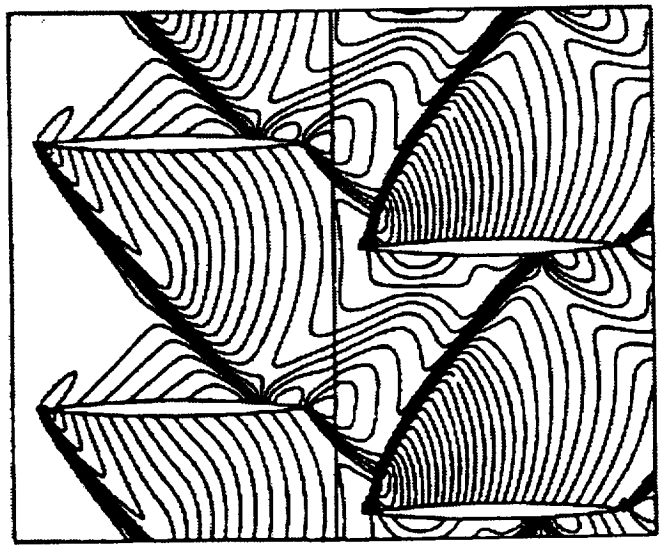

Figure 7.60 : Pressure Contours for Case (b) and the end of 4.6 cycles from [44]

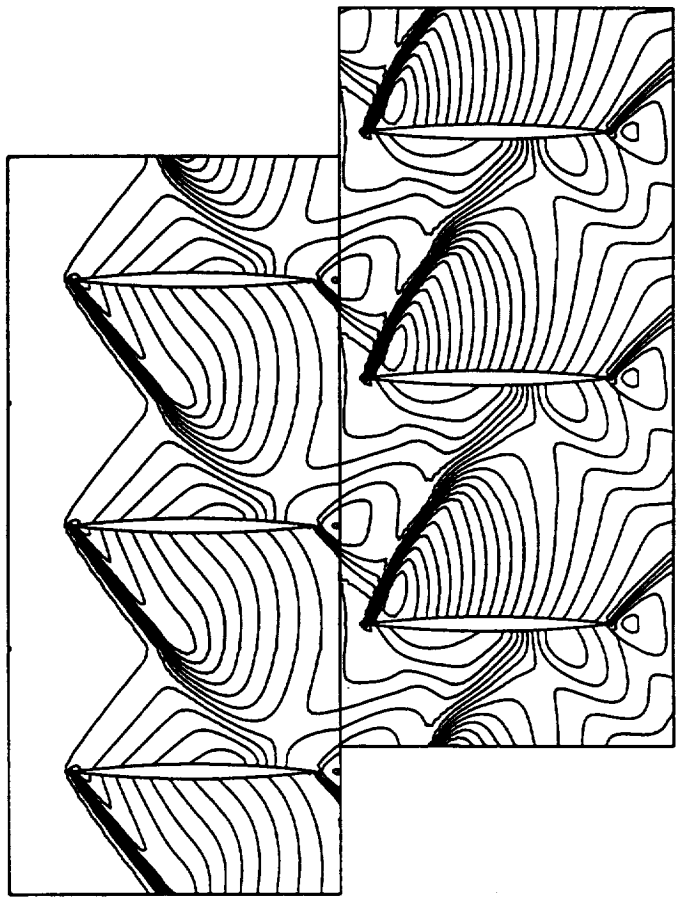

Figure 7.61 : Pressure Contours for Case (b) and the end of 5.6 cycles with First Order Accuracy

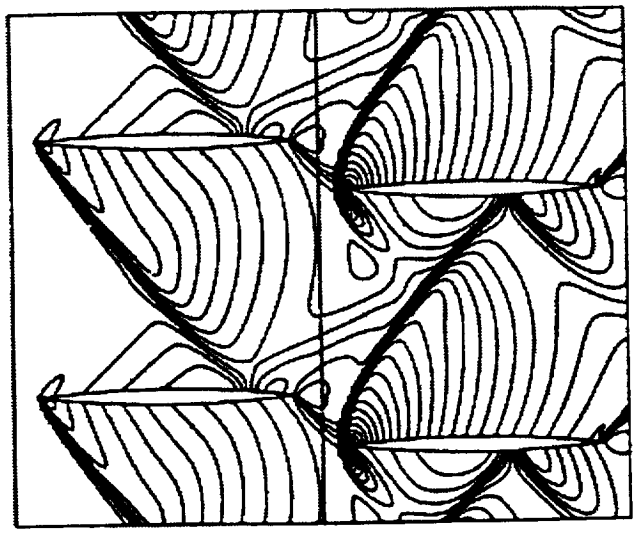

Figure 7.62 : Pressure Contours for Case (b) and the end of 4.8 cycles from [44]

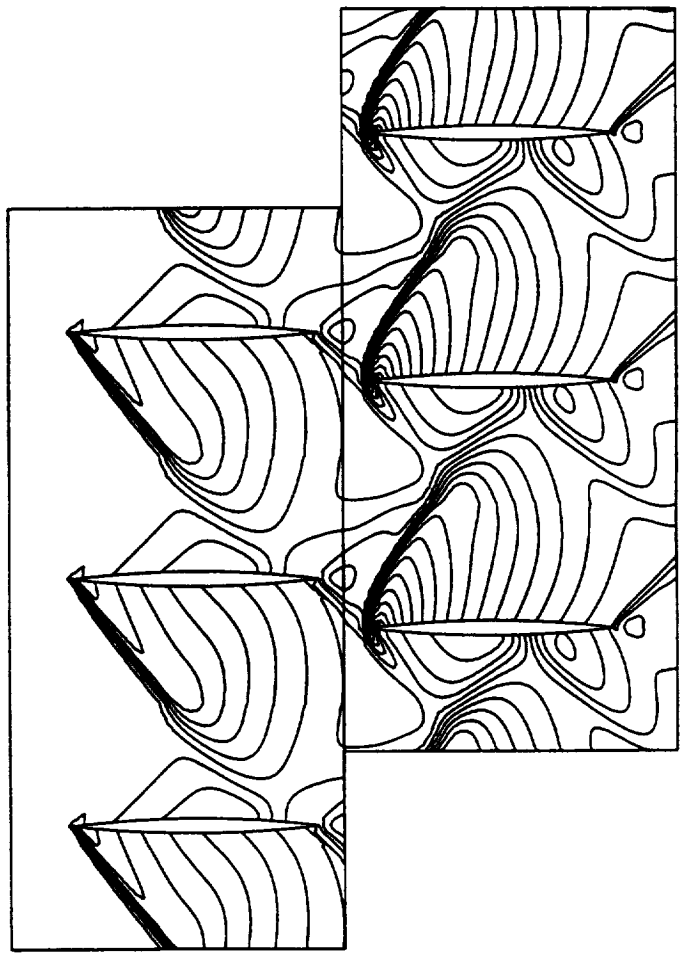

Figure 7.63 : Pressure Contours for Case (b) and the end of 5.8 cycles with First Order Accuracy

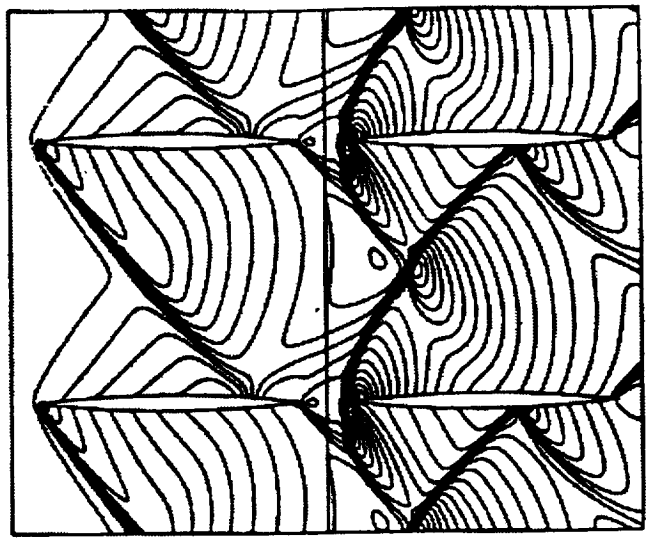

Figure 7.64 : Pressure Contours for Case (b) and the end of 5.0 cycles from [44]

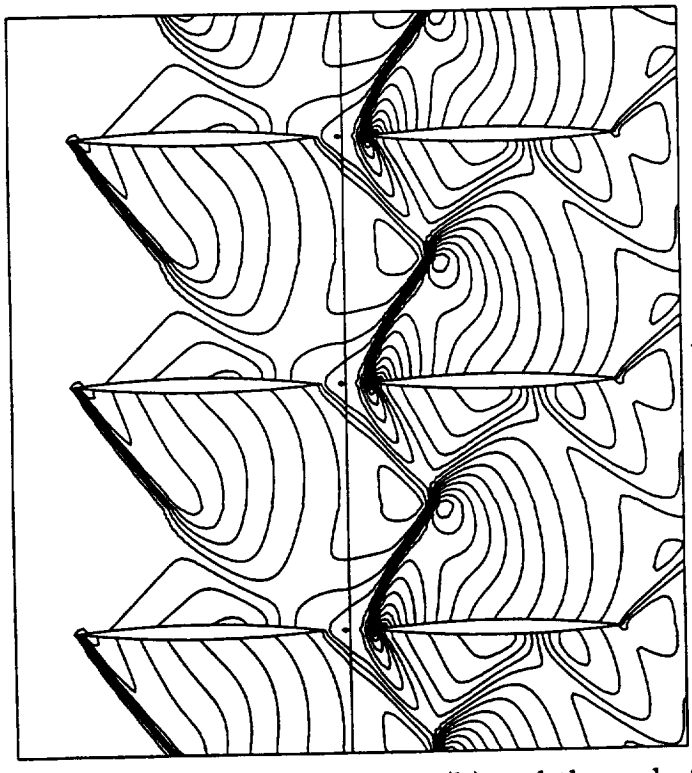

Figure 7.65 : Pressure Contours for Case (b) and the end of 6.0 cycles with First Order Accuracy

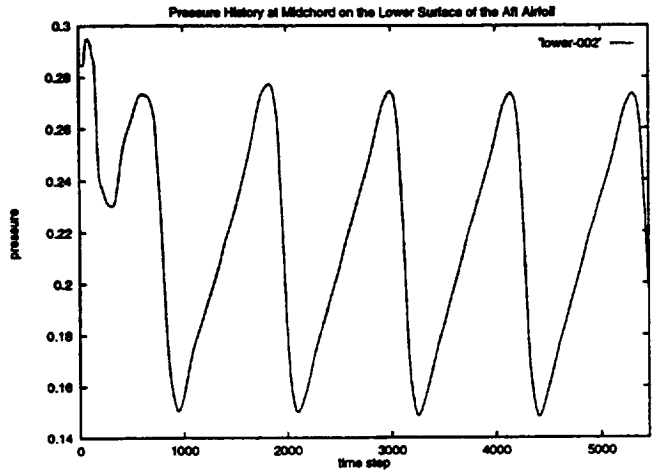

Figure 7.66 : Pressure History at Midchord on the Lower Surface of the Aft Airfoil for Case (b) with Second Order Accuracy

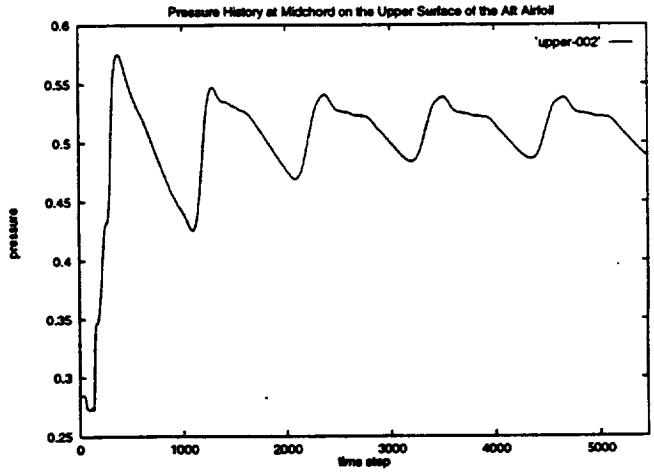

Figure 7.67 : Pressure History at Midchord on the Upper Surface of the Aft Airfoil for Case (b) with Second Order Accuracy

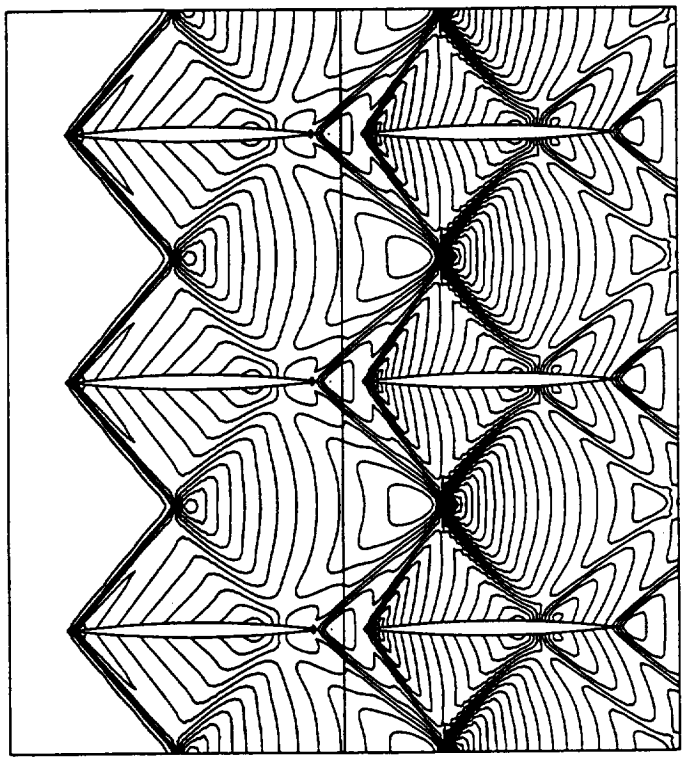

Figure 7.68 : Pressure Contours at Steady-State for Case (b) with Second Order Accuracy

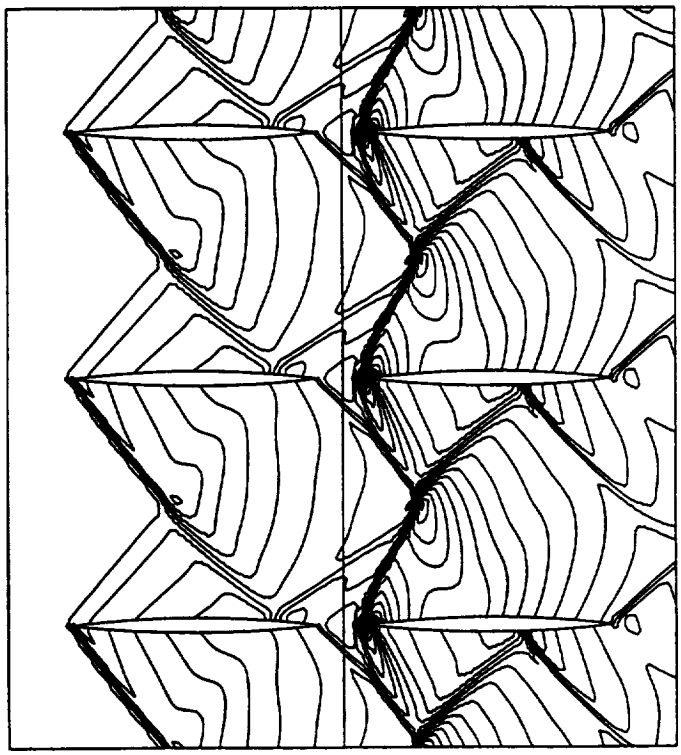

Figure 7.69 : Pressure Contours for Case (b) and the end of 6.0 cycles with Second Order Accuracy

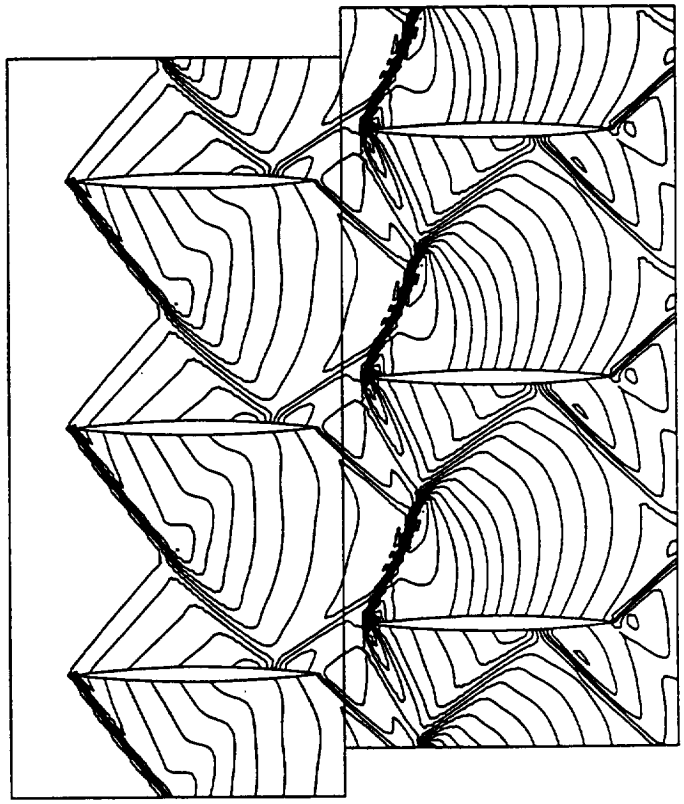

Figure 7.70 : Pressure Contours for Case (b) and the end of 6.2 cycles with Second Order Accuracy

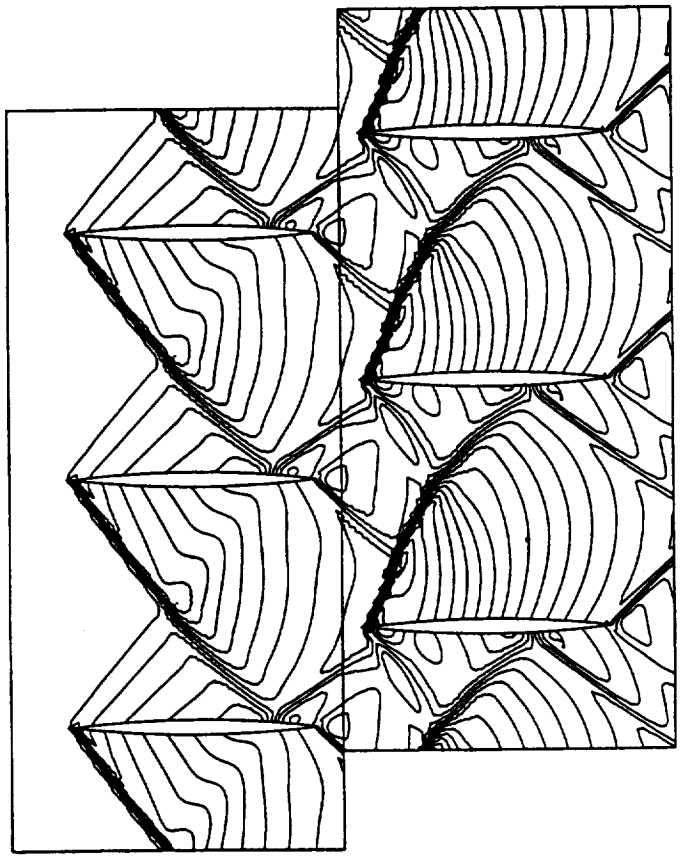

Figure 7.71 : Pressure Contours for Case (b) and the end of 6.4 cycles with Second Order Accuracy

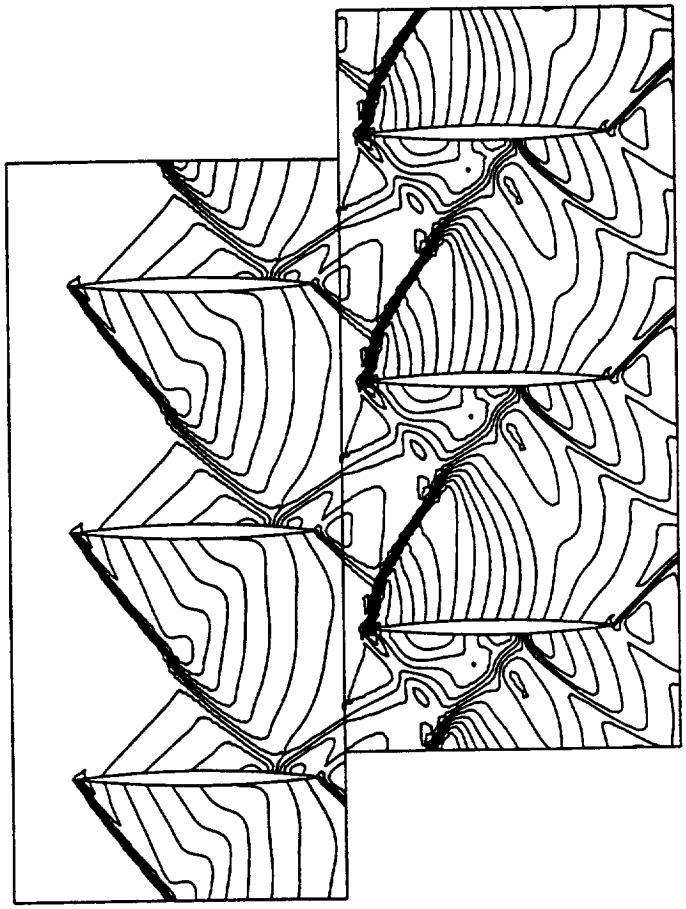

Figure 7.72 : Pressure Contours for Case (b) and the end of 6.6 cycles with Second Order Accuracy

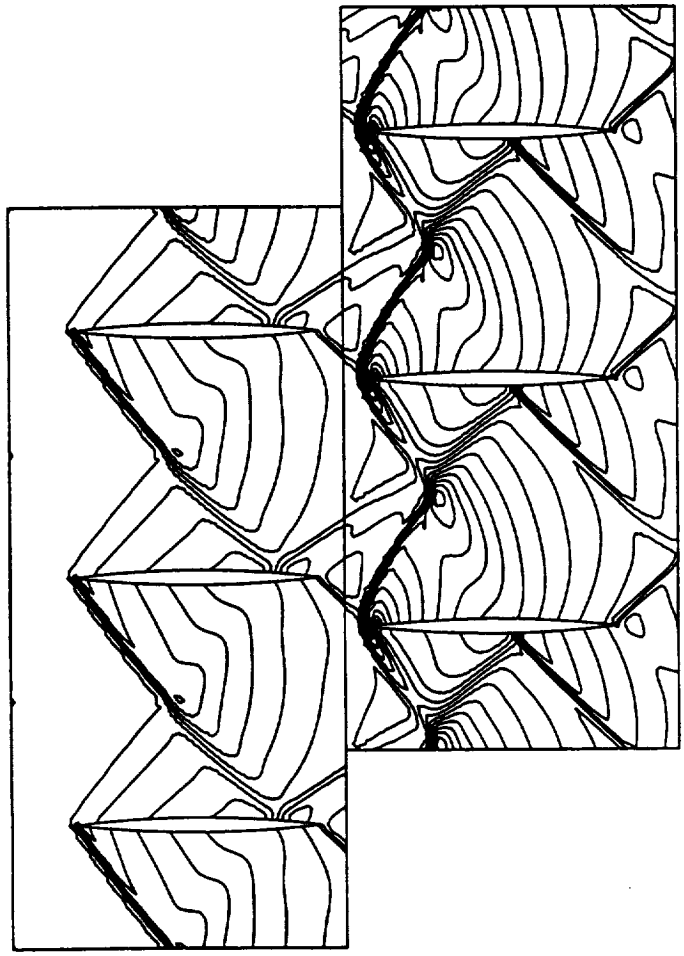

Figure 7.73 : Pressure Contours for Case (b) and the end of 6.8 cycles with Second Order Accuracy

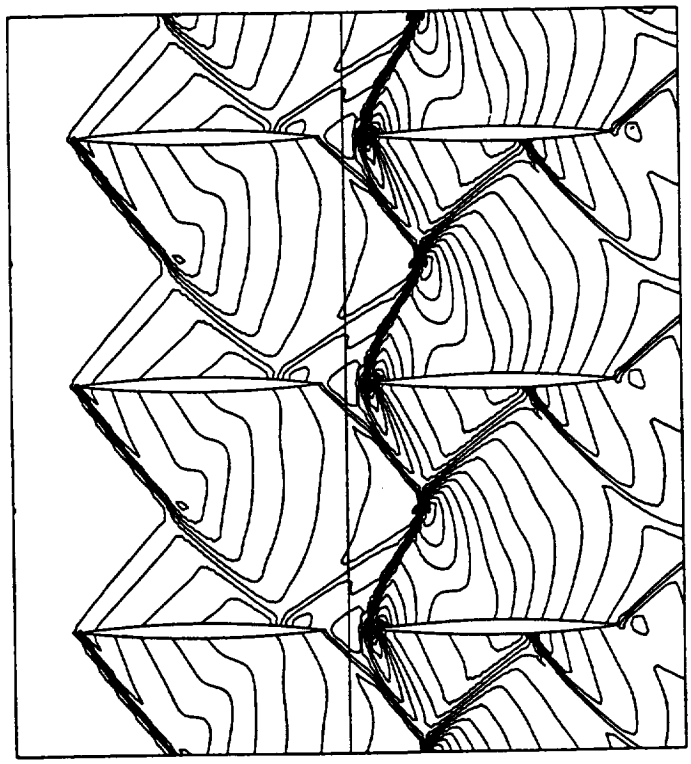

Figure 7.74 : Pressure Contours for Case (b) and the end of 7.0 cycles with Second Order Accuracy

Chapter 8

Conclusion

The final and concluding chapter of this dissertation reviews current work, evaluates it by comparing and contrasting it with other related published research and makes recommendations for future work.

8.1 Discussion and Review of Present Work

The fundamental motivation for the present research was the massively parallel three-dimensional aeroelastic analysis of aircraft engines using unstructured meshes for the fluid component and staggered methods. The realization of this ambitious project was found to require additional work in several modeling and methodology ingredients. One of these was selected for detailed investigation, namely flow simulations on non-matching unstructured meshes, with particular application to the analysis of rotor-stator interaction phenomena in turbomachinery.

Several important factors need to be considered while developing such a method. The two most significant ones are flux conservation and continuity of physical variables. Flux conservation is essential to ensure that any method converges to the single physically correct solution of a conservation law, especially as regards the correct transmission and positioning of shocks

and discontinuities. On the other hand, continuity at the mesh interface is required so that artificial discontinuities do not arise at the interface on account of the numerical approximation.

A number of flux conservative methods have been proposed and implemented in the context of both rotor-stator interaction and domain decomposition. However, these have been largely developed for structured meshes with regular geometric patterns. Furthermore, these methods are usually computationally intensive, requiring a considerable amount of geometric computations to calculate volume weights which are needed to determine the manner in which fluxes are transferred at the mesh interface. It has also been mentioned in the literature [49], that in some cases, attempts to ensure flux conservation give rise to some forms of instability.

In addition to flux conservative methods, non-conservative methods have been quite popular, particularly in the field of domain decomposition with the use of overset or Chimera grids. The main drawback of these methods that is repeatedly brought out is their inability to capture shocks and discontinuities correctly and failure to converge to the correct solutions. While these claims are true to some extent, it is felt that in many cases in which these results were reported, proper precaution was not exercised and faults lay more in the way in which these methods were used than in the way in which they were designed. Also, some authors [13] have suggested that discrete local conservation with continuity is sufficient and complete global conservation is not essential to ensure the success or validity of a method.

In the present work, both non-overlapping and overlapping methods were investigated. It was observed that non-overlapping methods gave incorrect results on account of their being inconsistent with the adopted spatial discretization. This mandates the use of an overlapping scheme.

Flux conserving overlapping schemes generally involve the determination of volume weights which are required for the exchange of information between meshes. For the spatial discretization that the current fluid solver uses, determination of these volume weights would have meant either calculating the areas of intersection of finite-volume cells of the projecting mesh and the overlapped mesh or the intersections between the edges of these cells. Each of these procedures requires intricate programming efforts and if implemented, would be computationally expensive because finite-volume cells are, in general, irregular polygons with the number of edges and vertices changing from cell to cell. Therefore, a non-conservative approach was followed in which greater emphasis was laid on the satisfaction of the continuity requirements and the transfer of variables from one mesh to another so that discrete local conservation could be achieved.

The method developed, identified by the acronym SUM (standing for Slipping Unstructured Meshes), uses optimally overlapping meshes to exchange information and compute fluxes. Two projection schemes were implemented in SUM, namely linear interpolation (SUM/LI) and the mortar method (SUM/MP). It was felt that the way in which meshes were overlapped and the manner in which information was exchanged would result in the creation of a robust method for use in applications of current interest.

8.2 Evaluation

The applicability of SUM has been demonstrated on several test problems in Chapter 7. The numerical experiments indicate that this method is successful in capturing and locating shocks and discontinuities correctly for both supersonic and transonic steady-state and unsteady flow simulations. It is

also applied to a simplified rotor-stator configuration. The results match fairly well with those from published sources.

From the results presented in Chapter 7, it can be inferred that SUM can be used in confidence to analyze flows which have strong gradients, or discontinuities or both. In Chapter 6, it is seen that while exact global conservation is not imposed, discrete conservation is achieved locally on each mesh and the imbalances in fluxes reported globally are within an acceptable range of tolerance. An interesting observation made is that the variant of SUM with linear interpolation, SUM/LI achieves almost the same level of flux conservation as its variant with mortar projection, SUM/MP.

The major drawback SUM is its reliance on mesh overlapping. Care must be taken to ensure that this is properly done. However, in most cases, this requirement merely stipulates that the mesh overlap should occur in such a way that the mesh refinement in the region of overlap is compatible with the mesh refinement in the non-overlapping region and that the individual meshes, though not matched, are refined to approximately equal degrees. This requirement is not overly restrictive as non-matching of meshes usually occurs at critical locations (such as the interface between a rotor and a stator) and meshes there would have to be sufficiently refined to adequately capture flow properties.

SUM is computationally efficient and requires only the determination of triangle segment intersections in the region of overlap. This search can be optimized by choosing the mesh overlaps to satisfy certain constraints which would depend upon the type of application. The transfer or exchange of information between meshes imposes little overhead in computational costs and results are obtained with almost the same efficiency as in case of single mesh computations.

SUM is fairly general and can be tailored to suit a wide range of applications such as the motion of control surfaces on an aircraft wing and store separation operations. In addition, it is also a candidate for non-conforming domain decompositions in CFD. The latter would allow separate generation of meshes with varying degree of refinement. This would be specially beneficial for turbulence computations where narrow, flat cells are required near the wall boundaries whereas coarser meshes are acceptable away from the wall. It could also be used to 'glue' solutions obtained on structured and unstructured meshes, which is again of interest in turbulence computations. To carry the same idea further, a method similar to the one developed can be used to link flow solutions obtained with methods of varying fidelity on different portions of the physical domain, i.e., the simultaneous use of potential, Euler, Navier-Stokes and turbulence flow solvers depending upon the degree of refinement required in the solution, see Section 2.1.1.

To summarize, the following are the major accomplishments of the present work :

1. First fully three dimensional aeroelastic simulation of an entire stage of an aircraft engine.
2. First application of the mortar element method for Euler flows on unstructured meshes with a finite-volume discretization.
3. First instance of flow computations being carried out on non-matching finite-volume unstructured meshes with second order spatial accuracy.

Item (3) above is the most significant achievement of this research.

8.3 Recommendations for Future Research

The in-depth research work has focused on a particular aspect of a prototype of a more complete computational setup designed to analyze complicated flows through realistic turbomachinery models. The following are the desired additions and/or modifications that need to be made in order to take the current method to that level of capability :

1. In some cases, it becomes essential to consider turbulent flows both in turbomachinery applications such as rotor-stator interaction and also other applications of interest like the motion of control surfaces and store separation operations. Thus, the task of foremost importance would be the extension of the current method to include viscous and turbulence effects.
2. Interlinked with use of turbulence models would be the need to allow for implicit time-stepping given the need to use highly resolved meshes to capture turbulence effects. Of particular relevance to the current method would be to perform the mortar projection in an implicit manner.
3. The driving motivation for current work was the desire to perform aeroelastic analysis of turbomachinery components. Steps that need to be taken in this direction include the use of a mass-spring model to move the mesh and the exchange of pressures and displacements to and from the structural components, such as those mentioned in Section 3.4.
4. It would also be of interest to extend the current method to allow mesh interfaces to be oriented arbitrarily and the presence of multiple mesh interfaces. Although this would not be necessary in rotor-stator

interaction computations, it is useful in the linkage of solutions obtained from various models discussed earlier.

5. SUM lends itself well to parallelization and that would be another area where further efforts are needed. In the simplest scenario, two processors could be used in rotor-stator simulations, one for each stage. As complexity in modeling increases more processors could be assigned. In particular, the issues that need to be addressed are the way in which the geometrical computations involved in the region of overlap could be optimized and the parallelization of the mortar method.
6. Lastly, the current method with all the additions and modifications mentioned above needs to be extended to 3-dimensions.

On a closing note, it is felt that this research provides a starting block for further investigation and design of methods of this type used to merge solutions obtained on separate computational domains. Applications of these are many and in particular, it is hoped that this work is a step in the right direction towards the unified multidisciplinary analysis of complete aircraft engines.

Bibliography

[1] G. Abdoulaev, Y. Achdou, J.-C. Hontand, Y. A. Kuznetsov, O. Pironneau, and C. Prud'homme. Nonmatching Grids for Fluids. In *Proceedings of the Tenth International Conference on Domain Decomposition*, 1990.

[2] J. J. Adamczyk. Model Equation for Simulating Flows in Multistage Turbomachinery. ASME Paper 85-GT-226, 1985.

[3] G. Anagnostou, Y. Maday, C. Mavriplis, and A. Patera. The Mortar Element Method : Generalization and Implementation. In *Proceedings of the Third International Conference on Domain Decomposition Methods for Partial Differential Equations*. SIAM, 1990.

[4] J. D. Anderson, Jr. *Modern Compressible Flow with Historical Perspective*. McGraw-Hill Publishing Company, second edition, 1990.

[5] J. D. Anderson, Jr. *Fundamentals of Aerodynamics*. McGraw-Hill, Inc., second edition, 1991.

[6] M. A. Bakhle, T. S. R. Reddy, and T. G. Keith, Jr. Time Domain Flutter Analysis of Cascades Using a Full-Potential Solver. *AIAA Journal*, 30(1):163–170, January 1992.

[7] J. T. Batina. Unsteady Euler Airfoil Solutions using Unstructured Dynamic Meshes. AIAA Paper 89-0115.

[8] O. O. Bendiksen. Aeroelastic Problems in Turbomachines. AIAA Paper AIAA-90-1157-CP.

[9] O. O. Bendiksen. Role of Shocks in Transonic/Supersonic Compressor Rotor Flutter. *AIAA Journal*, 24:1179–1186, July 1986.

[10] M. J. Berger. On Conservation at Grid Interfaces. *SIAM Journal on Numerical Analysis*, 24(5):967–984, October 1987.

[11] C. Bernardi, Y. Maday, and A. T. Patera. A New Nonconforming Approach to Domain Decomposition : The Mortar Element Method. Technical report, Universite Pierre er Marie Curie, Paris, France, January 1990.

[12] X.-C. Cai, M. Dryja, and M. Sarkis. Overlapping non-matching grid mortar element methods for elliptic problems. *SIAM Journal for Numerical Analysis*, Submitted for Publication.

[13] G. Chesshire and W. D. Henshaw. A Scheme for Conservative Interpolation on Overlapping Grids. *SIAM Journal on Scientific Computing*, 15(4):819–845, July 1994.

[14] R. V. Chima. Development of an Explicit Multigrid Algorithm for Quasi Three-Dimensional Flows in Turbomachinery. NASA TM-87128, January 1986.

[15] R. V. Chima and J. W. Yokota. Numerical Analysis of Three-Dimensional Viscous Internal Flows. *AIAA Journal*, 28(5):798–806, May 1990.

[16] J. Conlon. Striving for Peak Design. NASA InSights newsletter, May 1997.

[17] J. Donea. Arbitrary Lagrangian-Eulerian Finite Element Methods. In T. Belytschko and T. J. R. Hughes, editors, *Computational Methods for Transient Analysis*, volume 1 of *Computational Methods in Mechanics*, chapter 10, pages 473–516. North-Holland, 1983.

[18] R. P. Dring, H. D. Joslyn, L. W. Hardin, and J. H. Wagner. Turbine Rotor-Stator Interaction. *Journal of Engineering for Power*, 104:729–742, October 1982.

[19] J. I. Erdos, E. Alzner, and W. McNally. Numerical Solution of Periodic Transonic Flow through a Fan Stage. *AIAA Journal*, 15(11):1559–1568, November 1977.

[20] C. Farhat, L. Crivelli, and F. X. Roux. A Transient FETI Methodology for Large-Scale Parallel Implicit Computations in Structural Mechanics. *International Journal of Numerical Methods in Engineering*, 37:1945–1975, 1994.

[21] C. Farhat, L. Crivelli, and F. X. Roux. Extending Substructure Based Iterative Solvers to Multiple Load and Repeated Analyses. *Computer Methods in Applied Mechanics and Engineering*, 1994:195–209, 1994.

[22] C. Farhat and S. Lanteri. Simulation of Compressible Viscous Flows on a variety of MPPs : Computational Algorithms for Unstructured Dynamic Meshes and Performance Results. *Computer Methods in Applied Mechanics and Engineering*, 119:35–60, 1994.

[23] C. Farhat, S. Lanteri, and H. D. Simon. TOP/DOMDEC — a Software Tool for Mesh Partitioning and Parallel Processing. *Computing Systems in Engineering*, 6(1):13–26, February 1995.

[24] S. Fleeter. Aeroelasticity Research for Turbomachine Applications. *Journal of Aircraft*, 16(5):320–326, May 1979.

[25] Y. C. Fung. *An Introduction to the Theory of Aeroelasticity.* John Wiley & Sons, Inc., 1955.

[26] M. Geradin and D. Rixen. *Mechanical Vibrations, Theory and Applications to Structural Dynamics.* Wiley, 1994.

[27] G. A. Gerolymos. Numerical Integration of the Blade-to-Blade Surface Euler Equations in Vibrating Cascades. *AIAA Journal*, 26(12):1483–1492, December 1988.

[28] G. A. Gerolymos. Advances in the Numerical Integration of Three-Dimensional Euler Equations in Vibrating Cascades. *Journal of Turbomachinery*, 115:781–790, October 1993.

[29] M. B. Giles. Stator/Rotor Interaction in a Transonic Turbine. *Journal of Propulsion and Power*, 6:621–627, Sept.–Oct. 1990.

[30] U. A. Gumaste, C. A. Felippa, and C. Farhat. Massively Parallel 3D Aeroelastic Analysis of Jet Engines. In *Computational Aerosciences Meeting.* NASA, 1996.

[31] S. L. Keeling, R. W. Tramel, and J. A. Benek. A Theoretical Framework for Chimera Domain Decomposition . In *Sixth International Symposium on CFD*, pages 1–6, September 1995.

[32] M. Koya and S. Kotake. Numerical Analysis of Fully Three-Dimensional Periodic Flows through a Turbine Stage. *Journal of Engineering for Gas Turbines and Power*, 107:945–952, October 1985.

[33] A. M. Kuethe and C.-Y. Chow. *Foundations of Aerodynamics - Bases of Aerodynamic Design.* John Wiley & Sons, fourth edition, 1986.

[34] C. Lacour and Y. Maday. Two Different Approaches for Matching Non-conforming Grids : The Mortar Element Method and the FETI Method. *BIT*, 37(3):13–33, October 1996.

[35] B. Lakshminarayana. An Assessment of Computational Fluid Dynamic Techniques in the Analysis and Design of Turbomachinery – The 1990 Freeman Scholar Lecture. *Jounral of Fluids Engineering*, 113:315–352, September 1991.

[36] F. Lane. Supersonic Flow Past an Oscillating Cascade with Supersonic Leading-Edge Locus. *Journal of the Aeronautical Sciences*, 24:65–66, January 1957.

[37] S. Lanteri. Parallel Solutions of Three-Dimensional Compressible Flows. Rapport de Recherche 2594, INRIA Sophia-Antipolis, June 1995.

[38] S. Lanteri and C. Farhat. Viscous Flow Computations on MPP systems : Implementational Issues and Performance Results for Unstructured Grids. In R. F. Sincovec and *et al*, editors, *Parallel Processing for Scientific Computing*, pages 65–70. SIAM, 1993.

[39] P. D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves.* Society of Industrial and Applied Mathematics, 1973.

[40] A. Leissa. Vibrational Aspects of Rotating Turbomachinery Blades. *Applied Mechanics Reviews*, 34(5):629–635, May 1981.

[41] M. Lesoinne. *Mathematical Analysis of Three-Field Numerical Methods for Aeroelastic Problems.* PhD thesis, University of Colorado, Boulder, 1994.

[42] M. Lesoinne and C. Farhat. Geometric Conservation Laws for Aerolelastic Computations Using Unstructured Dynamic Meshes. AIAA Paper 95-1709, 1995.

[43] R. J. LeVeque. *Numerical Methods for Conservation Laws.* Lectures in Mathematics – ETH Zürich. Birkhäuser Verlag, second edition, 1992.

[44] N. K. Madavan and M. M. Rai. Computational Analysis of Rotor-Stator Interaction in Turbomachinery Using Zonal Techniques. In P. A. Henne, editor, *Applied Computational Aerodynamics*, volume 125 of *Progress in Astronautics and Aeronautics*, chapter 13, pages 481–532. American Institute of Aeronautics and Astronautics, 1990.

[45] N. Maman and C. Farhat. Matching Fluid and Structure Meshes for Aeroelastic Computations : A Parallel Approach. *Computers and Structures*, 1994.

[46] S. R. Mathur, N. K. Madavan, and R. G. Rajagopalan. Solution-Adaptive Structured-Unstructured Grid Method for Unsteady Turbomachinery Analysis, Part I : Methodology. *Journal of Propulsion and Power*, 10(4):576–584, July–Aug. 1994.

[47] S. R. Mathur, N. K. Madavan, and R. G. Rajagopalan. Solution-Adaptive Structured-Unstructured Grid Method for Unsteady Turbomachinery Analysis, Part II : Results. *Journal of Propulsion and Power*, 10(4):585–592, July–Aug. 1994.

[48] K. C. Park and C. A. Felippa. Partitioned Analysis of Coupled Systems. In T. Belytschko and T. J. R. Hughes, editors, *Computational Methods for Transient Analysis*, volume 1 of *Computational Methods in Mechanics*, chapter 10, pages 157–219. North-Holland, 1983.

[49] E. Pärt-Enander and B. Sjögreen. Conservative and Non-Conservative Interpolation between Overlapping Grids for Finite Volume Solutions of Hyperbolic Problems. *Computers & Fluids*, 23(3):551–574, 1994.

[50] S. Piperno, C. Farhat, and B. Larrouturou. Partitioned procedures for the transient solution of coupled aeroelastic problems Part I : Model

problem, theory and two-dimensional application. *Computer Methods in Applied Mechanics and Engineering*, 124:79–112, 1995.

[51] M. M. Rai. A Conservative Treatment of Zonal Boundaries for Euler Equation Calculations. *Journal of Computational Physics*, 62:472–503, Feb. 1986.

[52] M. M. Rai. A Relaxation Approach to Patched-Grid Calculations with the Euler Equations. *Journal of Computational Physics*, 66(1):99–131, September 1986.

[53] M. M. Rai. Navier-Stokes Simulations in Rotor/Stator Interaction using Patched and Overlaid Grids. *Journal of Propulsion and Power*, 3(5):387–396, Sept.-Oct. 1987.

[54] M. M. Rai and N. K. Madavan. Multi-Airfoil Navier-Stokes Simulations of Turbine Rotor-Stator Interaction. *Journal of Turbomachinery*, 112:377–384, July 1989.

[55] E. S. Reddy and C. C. Chamis. BLASIM : A Computational Tool to Assess Ice Impact on Engine Blades. AIAA Paper 93-1638.

[56] T. S. R. Reddy, M. A. Bakhle, R. Srivastava, O. Mehmed, and G. L. Stefko. A Review of Recent Aeroelastic Analysis Methods for Propulsion at NASA Lewis Research Center. NASA TP-3406, December 1993.

[57] P. L. Roe. Approximate Riemann Solvers, Parameter Vectors and Difference Schemes. *Journal of Computational Physics*, 43:357–371, 1981.

[58] G. A. Sod. A Survey of Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws. *Journal of Computational Physics*, 1978.

[59] R. Srivastava, L. N. Sankar, T. S. R. Reddy, and D. L. Huff. Application of an Efficient Hybrid Scheme for Aeroelastic Analysis of Advanced Propellers. *Journal of Propulsion*, 7(5):767–775, Sept–Oct 1991.

[60] J. L. Steger and R. F. Warming. Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods. *Journal of Computational Physics*, 40:263–293, 1981.

[61] M. Stewart. Axisymmetric Aerodynamic Numerical Analysis of a Turbofan Engine. ASME Paper 95-GT-338, 1995.

[62] E. Sutjahjo and C. C. Chamis. Three-Dimensional Multidisciplinary Finite Elements for Coupled Analysis involving Fluid Mechanics, Heat Transfer and Solid Mechanics. AIAA Paper 96-1372.

[63] T. Tezduyar, M. Behr, and J. Liou. A New Strategy for Finite Element Computations involving Moving Boundaries and Interfaces — The Deforming Spatial Domain/Space-Time Procedure : I. The Concept and the Preliminary Numerical Tests. *Computer Methods in Applied Mechanics and Engineering*, 94:339–351, 1992.

[64] P. D. Thomas and C. K. Lombard. Geometric Conservation Law and Its Application to Flow Computations on Moving Grids. *AIAA Journal*, 17(10):1030–1037, October 1979.

[65] G. D. van Albada, B. van Leer, and W. W. Roberts. A Comparitive Study of Computational Methods in Cosmic Gas Dynamics. *Astronomy and Astrophysics*, 108:76–84, 1982.

[66] B. van Leer. Towards the Ultimate Conservative Difference Scheme V : a Second-Order Sequel to Godunov's Method. *Journal of Computational Physics*, 32:361–370, 1979.

[67] Z. J. Wang. A Fully Conservative Interface Algorithm for Overlapped Grids. *Journal of Computational Physics*, 122:96–106, 1995.

[68] C.-H. Wu. A General Theory of Three-Dimensional Flow in Subsonic and Supersonic Turbomachines of Axial-, Radial-, and Mixed-Flow Types. NACA TN-2604, 1952.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1999	Final Contractor Report

4. TITLE AND SUBTITLE

Euler Flow Computations on Non-Matching Unstructured Meshes

5. FUNDING NUMBERS

WU–523–22–13–00
NAG3–1425

6. AUTHOR(S)

Udayan Gumaste

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Colorado
College of Engineering
Boulder, Colorado 80309–0001

8. PERFORMING ORGANIZATION REPORT NUMBER

E–11702

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
John H. Glenn Research Center at Lewis Field
Cleveland, Ohio 44135–3191

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA CR—1999-209155

11. SUPPLEMENTARY NOTES

Project Manager, C.C. Chamis, Research and Technology Directorate, NASA Glenn Research Center, organization code 5000, (216) 433–3252.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category: 39 Distribution: Standard

This publication is available from the NASA Center for AeroSpace Information, (301) 621–0390.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Advanced fluid solvers to predict aerodynamic performance-coupled treatment of multiple fields are described. The interaction between the fluid and structural components in the bladed regions of the engine is investigated with respect to known blade failures caused by either flutter or forced vibrations. Methods are developed to describe aeroelastic phenomena for internal flows in turbomachinery by accounting for the increased geometric complexity, mutual interaction between adjacent structural components and presence of thermal and geometric loading. The computer code developed solves the full three dimensional aeroelastic problem of stage. The results obtained show that flow computations can be performed on non-matching finite-volume unstructured meshes with second order spatial accuracy.

14. SUBJECT TERMS

Forced vibrations; Fluid/structure interaction; Finite volume; Turbomachinery

15. NUMBER OF PAGES

205

16. PRICE CODE

A10

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	