INFORMATION TO USERS

The most advanced technology has been used to photograph and reproduce this manuscript from the microfilm master. UMI films the original text directly from the copy submitted. Thus, some dissertation copies are in typewriter face, while others may be from a computer printer.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyrighted material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is available as one exposure on a standard 35 mm slide or as a 17" × 23" black and white photographic print for an additional charge.

Photographs included in the original manuscript have been reproduced xerographically in this copy. 35 mm slides or 6" × 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA

Order Number 8809936

**A truncation error injection approach to viscous-inviscid
interaction**

Goble, Brian Dean, Ph.D.

The University of Arizona, 1988

U·M·I

300 N. Zeeb Rd.
Ann Arbor, MI 48106

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy.
Problems encountered with this document have been identified here with a check mark ✓.

1. Glossy photographs or pages _____
2. Colored illustrations, paper or print _____
3. Photographs with dark background _____
4. Illustrations are poor copy _____
5. Pages with black marks, not original copy ✓
6. Print shows through as there is text on both sides of page _____
7. Indistinct, broken or small print on several pages ✓
8. Print exceeds margin requirements _____
9. Tightly bound copy with print lost in spine _____
10. Computer printout pages with indistinct print _____
11. Page(s) _____ lacking when material received, and not available from school or author.
12. Page(s) _____ seem to be missing in numbering only as text follows.
13. Two pages numbered _____ . Text follows.
14. Curling and wrinkled pages _____
15. Dissertation contains pages with print at a slant, filmed as received _____
16. Other _____

U·M·I

**A TRUNCATION ERROR INJECTION APPROACH
TO VISCOUS-INVISCID INTERACTION**

by

Brian Dean Goble

**A Dissertation Submitted to the Faculty of the
DEPARTMENT OF AEROSPACE AND MECHANICAL ENGINEERING
In Partial Fulfillment of the Requirements
For the Degree of
DOCTOR OF PHILOSOPHY
With a Major in Aerospace Engineering
In the Graduate College
THE UNIVERSITY OF ARIZONA**

May 1988

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

2

As members of the Final Examination Committee, we certify that we have read
the dissertation prepared by Brian Dean Goble
entitled A Truncation Error Injection Approach
to Viscous-Inviscid Interaction

and recommend that it be accepted as fulfilling the dissertation requirement
for the Degree of Doctor of Philosophy.

K.-Y. Fung

Feb 9, 1988
Date

Hermann Fasel

2/9/1988
Date

Arne J. Pearlstein

2/12/88
Date

Date

Date

Final approval and acceptance of this dissertation is contingent upon the
candidate's submission of the final copy of the dissertation to the Graduate
College.

I hereby certify that I have read this dissertation prepared under my
direction and recommend that it be accepted as fulfilling the dissertation
requirement.

Dissertation Director K.-Y. Fung

Feb 12, 1988
Date

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Brian D. Goble

DEDICATION

This dissertation is dedicated to the memory of my father, Garrett Baker Goble. He did not live to see its completion, but his influence is there nonetheless.

ACKNOWLEDGMENTS

The author would like to thank his advisor Dr. K.-Y. Fung for his advice and guidance in performing this work, as well as his physical and numerical insight. The author would also like to thank his other committee members, Dr. Arne J. Pearlstein and Dr. Hermann Fasel, for their time and suggestions concerning the dissertation. Further thanks go to Dr. Tom Pulliam for his direction and time while the author was at NASA-Ames Research Center.

The author would like to thank his officemates Linda Kral, Jeff Schoen, David Dratler and Bruce Murray. Their friendship and support were greatly appreciated.

And finally, I would like to thank my family for all their support and for believing in me through it all. My wife Amy was always there when I needed her and gave me that extra encouragement I needed in order to finish this work.

TABLE OF CONTENTS

6

	Page
LIST OF ILLUSTRATIONS	8
LIST OF TABLES	11
ABSTRACT	13
1. INTRODUCTION	14
1.1 Transonic Flows	17
1.2 Multiple Mesh Schemes	19
1.3 Viscous-Inviscid Interaction	23
2. TRUNCATION ERROR INJECTION USED IN VISCOUS-INVISCID INTERACTION	26
2.1 Interpolation Scheme	29
2.2 Boundary Conditions	32
2.3 Implementation of the Scheme	36
3. DEVELOPMENT OF EQUATIONS	50
3.1 Generalized Curvilinear Coordinate Transformations	52
3.2 Thin-Layer Approximation	53
3.3 Numerical Method	55
3.4 Artificial Dissipation Model	58
4. VISCOUS-INVISCID INTERACTION RESULTS	60
4.1 Effects of Truncation Error Injection	61
4.2 Characteristics of Viscous-Inviscid Interaction Runs	66
4.3 Effects of Different Local Grid Boundary Conditions	72
4.4 Interaction Using Dissimilar Grids	75
4.5 Local Grid Used for Resolution Enhancement	79
5. CONCLUSION	115

APPENDIX A: SIMILARITY TRANSFORMATION MATRICES AND THE TIME-LINEARIZED FLUX JACOBIAN MATRICES	7 118
APPENDIX B: TEINJECT CODE: DESCRIPTION OF VARIABLES AND I/O FILES	122
APPENDIX C: SAMPLE JCL FILE FOR CODE TEINJECT: CASE VS37	145
REFERENCES	148

LIST OF ILLUSTRATIONS

8

Figure		Page
2.1	Mapping of generalized curvilinear coordinate system in physical space to computational space.	42
2.2	Two generalized curvilinear coordinate systems and bicubic patch for interpolation.	43
2.3	Local grid topologies. Boundary layer type grid and grid local to nose of airfoil.	44
2.4	Truncation error distribution for the conservation of mass equation. Case VRGA2A.	45
2.5	Convergence history. RMS Density residual vs. equivalent fine grid iterations. Case VTK37.	46
2.6a	Convergence history. Coefficient of Lift, C_L , vs. equivalent fine grid iterations. Case VTK37.	47
2.6b	Convergence history. Coefficient of Drag, C_D , vs. equivalent fine grid iterations. Case VTK37.	48
2.6c	Convergence history. Coefficient of Pitching Moment, C_M , vs. equivalent fine grid iterations. Case VTK37.	49
4.1	Pressure and Mach number contours for 257x65 fine grid case, VRGAA. Flow field 1.	85
4.2a	Negative logarithm of truncation error magnitude. Conservation of Mass equation. Case VRGA2A. Flow field 1.	86
4.2b	Negative logarithm of truncation error magnitude. Conservation of X-Momentum equation. Case VRGA2A. Flow field 1.	87
4.2c	Negative logarithm of truncation error magnitude. Conservation of Y-Momentum equation. Case VRGA2A. Flow field 1.	88
4.2d	Negative logarithm of truncation error magnitude. Conservation of Energy equation. Case VRGA2A. Flow field 1.	89

		9
4.3a	Negative logarithm of truncation error magnitude. Conservation of Mass equation. Case VRGA4A. Flow field 1.	90
4.3b	Negative logarithm of truncation error magnitude. Conservation of X-Momentum equation. Case VRGA4A. Flow field 1.	91
4.3c	Negative logarithm of truncation error magnitude. Conservation of Y-Momentum equation. Case VRGA4A. Flow field 1.	92
4.3d	Negative logarithm of truncation error magnitude. Conservation of Energy equation. Case VRGA4A. Flow field 1.	93
4.4	Boundary Layer profiles. Fine grid solution, VRGAA, and solution VK21. Straight Injection run. Flow field 1.	94
4.5	Grid RGA37 and grid RGA.	95
4.6	Pressure and Mach number contours for VS33 global solution and the 257x65 solution VRGAA. Flow field 1.	96
4.7	Pressure and Mach number contours for VS37 global solution and the 257x65 solution VRGAA. Flow field 1.	97
4.8	Boundary layer profiles. Fine grid solution, VRGAA, and solution VS21. Interaction run. Flow field 1.	98
4.9	Grids RGA and RGK.	99
4.10	Pressure and Mach number contours comparing global solutions for VTK37 with local boundary conditions CE and DD. Flow field 2.	100
4.11	Pressure and Mach number contours comparing local solutions for VTK37 with local boundary conditions CE and DD. Flow field 2.	101
4.12	Pressure and Mach number contours comparing global solution for VTK37 with local boundary conditions DD to VRGKA. Flow field 2.	102
4.13	Pressure and Mach number contours comparing global solutions for VTK37 and VTKM37. Flow field 2.	103

4.14	Pressure and Mach number contours comparing local solutions for VTK37 and VTKM37. Flow field 2.	10 104
4.15	Mach number contours. 257x65 single grid solution VNGAA. Flow field 3.	105
4.16	Mach number contours. 129x33 single grid solution VNGA2A. Flow field 3.	106
4.17	Mach number contours for case VNGA2G. 129x33 global grid interacting with 41x33 local grid. $\lambda = 2 \times 2$. Flow field 3.	107
4.18	Mach number contours for case VNGA2I. 129x33 global grid interacting with 81x65 local grid. $\lambda = 4 \times 4$. Flow field 3.	108
4.19	Mach number contours for case VNGA2J. 129x33 global grid interacting with 241x65 local grid. $\lambda = 12 \times 4$. Flow field 3.	109
4.20	Mach number contours for case VNGAF. 257x65 global grid interacting with 81x65 local grid. $\lambda = 2 \times 2$. Flow field 3.	110
4.21	Mach number contours for case VNGAG. 257x65 global grid interacting with 241x65 local grid. $\lambda = 6 \times 2$. Flow field 3.	111
4.22	Pressure and Mach number contours for VNGAA and fine grid interacting with 241x65 local grid, VNGAG. Flow field 3.	112
4.23	Vorticity magnitude contours for case VNGAG local solution. Flow field 3.	113
4.24	241x65 local grid NGAL5.	114

LIST OF TABLES

11

Table	Page
2.1 SAMPLE JCL FILE SEQUENCING INPUT - ORIGINAL	40
2.2 SAMPLE JCL FILE SEQUENCING INPUT - REVISED	41
4.1 STRAIGHT INJECTION RUNS - GRID RGA LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 1 ..	63
4.2 MAXIMUM TRUNCATION ERROR FOR EACH η -LINE	65
4.3 LOCAL GRIDS FROM GRID RGA	66
4.4 VISCOUS-INVISCID INTERACTION RUNS - GRID RGA LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 1 ..	67
4.5 VISCOUS-INVISCID INTERACTION RUNS - GRID RGA LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2 ..	69
4.6 LOCAL GRIDS FROM GRID RGK	70
4.7 VISCOUS-INVISCID INTERACTION RUNS - GRID RGK LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2 ..	72
4.8 VISCOUS-INVISCID INTERACTION RUNS - GRID RGK VARYING THE LOCAL GRID BOUNDARY CONDITIONS LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2 ..	74
4.9 INVISCID RUNS - GRIDS RGL AND RGM LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2 ..	76
4.10 LOCAL GRIDS FROM RGL AND RGM GRIDS	77
4.11 VISCOUS-INVISCID INTERACTION RUNS CASES USING DISSIMILAR GRIDS LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2 ..	78
4.12 VISCOUS-VISCOUS INTERACTION RUNS LOCAL PATCHED GRIDS LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 3 ..	81

4.13	VISCOUS-VISCOUS INTERACTION RUNS	12
	LOCAL PATCHED GRIDS GRID SIZES	
	AND RUNTIME INFORMATION - FLOW FIELD 3	82

A numerical procedure is presented which uses the truncation error injection methodology to efficiently achieve accurate approximations to complex problems having disparate length scales in the context of solving viscous, transonic flow over an airfoil. The truncation error distribution is estimated using the solution on a coarse grid. Local fine grids are formed which improve the resolution in regions of large truncation error. A fast fourth-order accurate scheme is presented for interpolating and relating the solutions between the generalized curvilinear coordinate systems of the local and global grids. It is shown that accurate solutions can be obtained on a global coarse grid with correction information obtained on local fine grids, which may or may not be topologically similar to the global grid as long as they are capable of resolving the local length scale. Dirichlet boundary conditions for the local grid yield the best results. The scheme also serves as the basis of a local refinement technique wherein a grid local to the nose of an airfoil is used to resolve a supersonic zone terminated by a shock and its interaction with a turbulent boundary layer. The solution on the local grid reveals details of the shock structure and a jet-like flow emanating from the root of the normal shock in the shock boundary layer interaction zone.

CHAPTER 1

INTRODUCTION

In the following a methodology will be described which allows the decoupling of a complex problem of multiple distinct length scales into problems of single length scale so that they can be solved more efficiently on a computer. The method is demonstrated in the context of viscous, transonic, lifting flow over an airfoil.

Computation of flows over aerodynamic bodies has reached a stage where solutions to most engineering problems can be found with some degree of accuracy. Solvers are available for equation sets ranging from the Laplace equation for subsonic or supersonic flow to the Reynolds-averaged compressible Navier-Stokes equations for transonic flow. The choice of equations to be solved to predict some given flow situation depends on such variables as degree of accuracy required, computer resources available, codes available, complexity of geometry, and complexity of flow situation. Ideally, the most accurate code available would be used on a grid with sufficient resolution to capture all of the relevant physics of the flow field. However, the computer resources, in terms of memory and CPU time required to implement the desired code on a suitable grid, are always limited.

The generation of such a grid is one of the main stumbling blocks in the solution process. One way to simplify the grid generation process and to increase the resolution for a given number of points is to break up a single global grid into several local grids that either overlap each other or interact through a single, coarse, global grid. The local grids could be generated about

individual components of the geometry or in regions where the characteristic length scale is much smaller than that of the surrounding flow field. Since the individual grids are only required to resolve one or two length scales, the task of generating them is greatly simplified, allowing points to be clustered where needed without wasting them elsewhere. Also, since the grids are separate, they do not have to be solved simultaneously, only requiring information pertaining to one grid to be resident in memory at a time. Although, in general, more points will be used overall, they will be used more efficiently and will give greater accuracy for a given amount of computing effort.

It is also possible to use different solvers on the different grids. Therefore, the solver that is most appropriate to the purpose of each grid can be chosen. For example, the full-potential equation could be solved on a coarse global grid while the Euler equations are solved around a shock to correctly predict the shock jumps and the Navier-Stokes equations are solved in the region near the body to resolve the viscous effects that are important there. In this way, a solver that is only as sophisticated as needed can be used. This approach is similar to the singular perturbation techniques in applied mathematics and also to viscous-inviscid interaction methods in aerodynamics.

Of course, using a multiple mesh scheme adds problems inherent to the approach. Information must be passed between the grids in such a way that the accuracy and stability of the scheme is not compromised. In addition, the bookkeeping required to keep track of the solution on several different grids is more involved and needs to be automated or it will become burdensome. There is also some memory and CPU time overhead due to

storing bookkeeping information and setting up the grid interaction algorithms. Another possible problem area is conservation. Most equation solvers used today in transonic aerodynamics applications are in conservative or divergence form. This form is required for the scheme to conserve such quantities as mass, momentum, and energy. In regions of the flow field where the solution is smooth, conservative form is not important, but when discontinuities, such as shocks, appear in the solution, conservation must be maintained in order to correctly predict their location. Even when the solvers themselves are conservative, if shocks cross a grid boundary that is not treated conservatively, the accuracy of the solution could suffer.

The method described here will interact two or more grids together in the solution of transonic, viscous flow over an airfoil. The method will support the use of a different solver for each grid, which will be interacted through a global coarse grid using the method proposed by Fung, Tripp, and Goble (1988). This approach treats the local grid solutions as more accurate approximations to the correct flow field than the global solution and uses them to correct the global solution. In this way, the global solution is influenced by each local grid and serves to transmit this data to each of the other local grids.

The next section presents a brief discussion on transonic flow, including its importance and characteristics. The rest of the chapter then provides some background material on other multiple mesh schemes in the literature, as well as a brief look at some current work in viscous-inviscid interaction. Chapter Two reviews the approach of Fung et al. (1988) and extends it to account for differences in operators, or formulations, as well as discussing the

interpolation routine and boundary conditions. Chapter Three presents the development of the solver used in this work and Chapter Four presents results using two interacting grids. Cases using both viscous-viscous and viscous-inviscid interaction are presented. Chapter Five follows with some conclusions and recommendations for future work.

1.1 Transonic Flows

The Navier-Stokes equations are the governing equations for fluid flow. These equations are a set of coupled, second-order, nonlinear, partial differential equations. In their complete form, they are indeed formidable to solve for practical aerodynamic problems. In many cases, however, these equations can be simplified by making some assumptions. Prandtl (1904) realized that the flow field could be effectively split into a viscous region and an inviscid region. He determined that viscous effects were only important in the region of the flow near a solid wall. He called this region the boundary layer. For high Reynolds number flows, where the boundary layer remained thin and attached to the body, the inviscid physics determined most of the qualities of the flow field and the boundary layer could be ignored. This deduction allowed the soon-to-be-born field of aerodynamics to grow and develop using strictly classical inviscid theory. However, as aircraft become more maneuverable and fly at higher angles of attack, an understanding of separated flow phenomena, which classical aerodynamics cannot provide, becomes more important. Viscous effects can no longer be assumed small and hence neglected.

To include viscous effects, one approach is to solve the viscous and the inviscid regions separately and match them in some way such as to have a

continuous solution. This approach is called viscous-inviscid interaction. Typically, the boundary layer equations are solved in the viscous region and matched with the solution of the potential equation for the inviscid part of the flow field. For mildly separated flows, where the boundary layer remains thin and the separated region has thickness on the order of that of the boundary layer, this procedure works very well and is very fast. For more strongly separated flows, where the separated region has thickness on the order of the chord, this approach breaks down, leaving the Navier-Stokes equations as the required equation set.

A flow field is transonic when both subsonic and supersonic flow regions exist in the flow field, typically separated by a sonic line and shock. Therefore, by its very nature, transonic flow is strongly nonlinear with many diverse length scales. The relevant scales range from the chord for the global aspects of the flow down to the thickness of the viscous sublayer at the wall and the thickness of a shock layer. This range of length scales spans several orders of magnitude, but any calculation must be able to capture them all in order to accurately predict the flow field. Another important phenomenon encountered in the transonic flow regime is shock wave-boundary layer interaction. The shock wave terminating the supersonic region impinges on the boundary layer, possibly causing separation and usually causing the boundary layer to thicken dramatically.

The ability to predict complex, transonic flow fields is becoming more important today. Most modern jetliners cruise in the transonic regime and military aircraft maneuver through it as well. As more rigorous demands are made on aircraft maneuverability and fuel efficiency, methods must be found

that can accurately predict separated flows with strong shocks in an efficient manner. Due to the nonlinearity of the equations, most studies of transonic flow have been done numerically, with the main exception being the hodograph methods for use in two dimensions, in which case the dependent and independent variables are interchanged. The transformation renders the resulting equation linear in the spatial variables at the expense of greatly complicating the boundary conditions. Nevertheless most of the theoretical work done in transonic flow has used the hodograph transformation.

The numerical approaches to inviscid, transonic flow began with the transonic small-disturbance equation, advanced to the full-potential equation, and finally to the Euler equations as computers became more powerful and understanding of transonic flows increased. Efforts at including viscous effects have included the aforementioned viscous-inviscid interaction techniques and solving the Navier-Stokes equations themselves. In order to capture all of the relevant physics and provide an accurate solution for the difficult problems found in transonic flow fields, the Navier-Stokes equations must be solved.

1.2 Multiple Mesh Schemes

The literature contains several different approaches to multiple mesh schemes. They can be grouped into two categories according to how the meshes contact each other. The first approach, known as patched gridding, abuts the meshes together in such a way that the bounding gridline for one mesh is the bounding gridline for the other mesh as well. The grids can be continuous or discontinuous across the interface and even if they are continuous, they can have discontinuous metrics. Rubbert and Lee (1982) discussed

two ways of generating patched grids, one which provides analytical continuity between the patches and one which generates grids without continuity but with improved control over mesh spacing and orientation at the boundaries. They claimed that a combination of the two approaches works best.

Rai (1984) used patches that were discontinuous across the boundaries. He chose the patched grid approach so that it would be easier to enforce conservation across the grid boundaries. In solving the Euler equations in 2-D with an explicit, first-order scheme, the fluxes are calculated across the boundaries to solve the boundary points for grid 1. The grid 1 fluxes are interpolated from grid 2 values using an interpolation scheme that forces the line integral of the fluxes in each coordinate system to be equivalent, thus enforcing conservation. Once the grid 1 boundary values are found, standard interpolation is used to get boundary values for grid 2. These values are used to solve for the interior points in grid 2. Rai presented results that show shocks passing through grid boundaries with ease, confirming that conservation is being satisfied. This approach was extended to second order implicit schemes in a later paper (Rai 1985) which also showed the time accuracy of the approach.

The second approach to multiple mesh schemes is known as overlaid gridding or component gridding. Here it will be referred to as overlaid gridding. There has been considerably more work done using this approach than the patched grid approach, probably due to its flexibility. In overlaid gridding, grids are generated about a component such as an external store or nacelle, or about regions of the flow field that will require more resolution than the rest of the flow field. These grids are generated independently of each other with

the only stipulation being that either they intersect each other in such a way that the entire domain is covered, or that all the local grids lie on top of a single, coarse, global grid which has the purpose of passing information between the grids. Rai (1984) pointed out two advantages of this approach over the former as being that the grids are easier to generate and that grids with movement relative to each other will be easier to handle. Two disadvantages are that interpolation of the same dimension as the problem must be used and that conservation will be more difficult to achieve.

Atta (1981) interacted two dissimilar grids together using the full potential equation to solve flow over an airfoil. The inner grid was a body conforming O-grid using a generalized curvilinear coordinate system, while the outer grid was Cartesian with a square cutout in the middle for the airfoil. The grids overlapped by two to four cells. Boundary conditions were interpolated using second order 2-D Lagrangian interpolation without any attempt being made to ensure conservation at grid boundaries. Results are presented for a supercritical case with the shock crossing the grid boundary. A supercritical flow is defined as a flow in which the local Mach number exceeds 1.0 somewhere in the flow field. Atta claimed that the maximum relative error in the sonic line position did not exceed 1.5%. The computing times and accuracy compared favorably with single grid results using TAIR, a full-potential code by Holst (1979). Atta and Vadyak (1982) extended this method to 3-D and studied flow about a wing-pylon-nacelle configuration.

Mastin and McConnaughey (1984) discussed both the patched and overlaid grid approaches. They pointed out that when patching grids together, special cells can result that have more than four sides, or nodes that have

more than eight neighbors. These points require special treatment, which is discussed in their paper. The corresponding problem in the overlaid grid approach is the choice of interpolation scheme. Several schemes are presented, including first and second order Taylor schemes, first and second order bivariate schemes and finite-element-like triangular interpolation. The second order bivariate scheme appeared to give the best results in the examples shown. The conservation problem is not considered in this paper.

Benek, Steger and Dougherty (1983), Steger, Dougherty, and Benek (1985), and Dougherty, Benek and Steger (1985) presented their overlaid grid scheme, CHIMERA, which interacts one or more component grids with a global grid via interpolated boundary conditions. The grids are all generated independently, without even the restriction that they do not intersect any bodies. A hierarchy is set up such that grids on one level are completely contained within grids of the next level up. Points in the higher grids that are covered by a lower grid or the body are "blanked" out and effectively eliminated from the calculation. The resulting "holes" form interior boundaries for the higher grids. Boundary conditions for the boundaries of these holes and the outer boundaries of the component grids are found by interpolation using first order Taylor interpolation. For an unsteady case, these grids are allowed to move with respect to each other, so the holes and boundary points are found at every step. The code handled complex boundaries well and gave good results for a plunging airfoil. The interpolation routines are not conservative, giving inaccurate results when a shock crosses a grid boundary.

Holst et al. (1985) presented a zonal method that uses both of the above approaches. A 3-D flow field is divided up into four blocks in three

levels, with each block being fully enclosed within the block above and fully enclosing the block below it. Blocks 3 and 4 are the exceptions; they are at the same level and are both fully contained within block 2. The interface between grids 1 and 2 is an overlaid grid boundary with a one cell overlap. The last three interfaces, between grids 2 and 3, between grids 2 and 4, and between grids 3 and 4, are patched grid boundaries with values being injected from one grid to the other to ensure conservation. These inner boundaries are the only boundaries which are conservative. The Euler equations are solved on grids 1 and 2 and the thin-layer Navier-Stokes equations are solved on grids 3 and 4. Results are presented that show good agreement with experimental results for lifting and nonlifting supercritical cases, as well as a case with massive separation. The results also show that the shocks cross the conservative interfaces without noticeable distortion.

1.3 Viscous-Inviscid Interaction

In the study of 2-D transonic flow over an airfoil, the inclusion of viscous effects is important in order to obtain accurate predictions of the lift and drag forces, as well as the position of the shock and separation points. These viscous effects can be incorporated into the solution in several ways. The two most common ways are to solve the Navier-Stokes equations in the entire flow field or to interact a solution of the boundary layer equations with some inviscid solver. In the Navier-Stokes approach, the equations are usually solved throughout the entire flow field on a single grid (e.g. Briley 1971, Pulliam 1984, Pulliam, Jespersen and Barth 1985, Barton and Pulliam 1986). Recently zonal methods utilizing the Navier-Stokes equations have appeared in the literature (Holst et al. 1985, Kaynak, Holst and Cantwell 1985).

The second approach, often called interacting boundary layer theory or viscous-inviscid interaction, does not have the same problem with boundary conditions as does the Navier-Stokes approach. Also, boundary layer codes are usually quite fast and give excellent resolution of the boundary layer. Their main disadvantage is that they cannot compute cases with separation that extends beyond the thin region near the body in which the equations are valid. However, these codes can calculate flow with local separation and reattachment. Catherall and Mangler (1966) showed that by integrating the boundary layer equations in inverse mode, with displacement thickness prescribed instead of the pressure gradient, the singularity associated with the separation point was eliminated. Since that time, several improved codes have been developed that can do the calculations in either direct or inverse mode (Carter 1974, 1978, Klineberg and Steger 1974, Van Dalsem and Steger 1983). Carter (1979, 1981), Van Dalsem and Steger (1983), Van Dalsem (1984), and Steger and Van Dalsem (1985) have interacted their codes with full-potential codes to achieve complete viscous-inviscid interaction. Steger and Van Dalsem (1985) also interacted their boundary layer code with a vector-potential code allowing rotational flow and convection of entropy, which improved the shock capturing capabilities of the scheme. These different viscous-inviscid interaction codes have been applied to a wide variety of transonic flow problems, including cases with shock induced separation (Carter 1981).

The conventional matching procedures used in most viscous-inviscid interaction schemes are very efficient and lead to rapid convergence. However, in most cases, they do not allow convection of vorticity and entropy

into the inviscid region of the flow field. Also, the interaction schemes are very complex and require some engineering judgement in applying them to non-standard problems. In an effort to generalize and simplify the interaction scheme, Steger and Van Dalsem (1985) used a forcing function approach to interact a boundary layer code with a vector-potential code. The boundary layer code generated values for vorticity, entropy, and changes of stagnation enthalpy which are input to the vector-potential solver in the entire viscous region of the flow field. Although the vector-potential formulation cannot generate vorticity or entropy, it adequately convects these quantities.

CHAPTER 2

TRUNCATION ERROR INJECTION USED IN VISCOUS-INVISCID INTERACTION

The scheme to be described here combines elements of overlaid grid-
ding and interacting boundary layer theory, as mentioned in Chapter One.
Several grids can be used, as necessary, to resolve a viscous, transonic flow
field in a computationally efficient manner. One possible local grid would be
a boundary layer type grid encompassing the region immediately adjacent to
the airfoil where viscous effects are important. Another possibility would be
to generate a grid about each shock that appears in the flow field. These
grids, and any others deemed necessary, would be interacted together through
a single, global, coarse grid which covered the entire domain of interest. The
global grid would have the purpose of passing information between the grids
and the far field using the truncation error injection scheme proposed by
Fung et al. (1988). This scheme was used by Liang and Fung (1987) to
study inviscid, steady, transonic flow past a wedge using the transonic small-
disturbance equation. In that study, the truncation error injection scheme
was used with only one solver. The effect of the scheme was to improve a
coarse grid solution using a series of fine grids. Fung and Fu (1987) used
this method in the study of a pitching airfoil to fix the solution at the mean
position. Their study used the unsteady transonic small-disturbance equation
and produced a speedup factor of 16 when comparing a solution achieved on
a coarse grid with truncation error injection to a fine grid solution of
equivalent accuracy. These two previous works each dealt with only one

equation set. In the scheme proposed here, truncation error injection will not only be transmitting information due to differences in mesh size, but also due to differences in solvers.

Truncation error occurs when a differential equation is discretized on a grid with finite mesh spacing. To illustrate what is meant by truncation error injection, a simple symbolic approach will be used. The Navier-Stokes equations can be written as

$$LQ = F(Q, Re) \quad (2.1)$$

where $F(Q, Re)$ represents the terms due to viscous effects and L is the inviscid operator for the Euler equations. The function F will be nonzero only in a small region of the flow field where viscous effects are important. This usually occurs only in the region adjacent to a solid boundary where the boundary layer forms due to the retarding effect of the boundary. Eqn. (2.1) can now be discretized on some grid, G_f , which encompasses this part of the domain and is fine enough to resolve the boundary layer adequately. The equation can be solved to find Q_f . This fine grid solution is next operated on once with the Euler operator discretized on a coarse, global grid G_c to form a new function

$$F'_c(Q_f, Re) = L_c Q_f \quad (2.2)$$

which is a measure of both the viscous terms and the difference in the grid resolutions. The grid G_c should be fine enough to resolve the inviscid aspects of the flow field but not so fine as to resolve the viscous effects. Now, accounting for the fact that Q_f was found on G_f and is zero outside of this region, the new function can be represented as

$$F'_c(Q_f, Re) = \begin{cases} L_c Q_f, & G_c \cap G_f \\ 0, & \text{otherwise} \end{cases} \quad (2.3)$$

Now solve the Euler equations on G_c using F'_c as defined in Eqn. (2.3) as a forcing function. The Euler solution will be influenced by the viscous effects represented in the forcing function.

The development shown above used the Euler and Navier-Stokes equations to illustrate the idea behind the interaction strategy, but the approach is solver-independent. The viscous solution could actually be found using one of the boundary layer solvers mentioned in Chapter One and any other appropriate inviscid solver could be used instead of an Euler solver.

In order to form the forcing function F'_c , the fine grid solution must be interpolated to the coarse grid. Ideally, this process would be done in a conservation-preserving manner in order to allow shocks to cross grid boundaries. As pointed out by Rai (1984), conservation is difficult to enforce in an overlaid grid approach. Therefore, another approach must be found to ensure that shocks are not smeared in the final solution. Two steps are being taken in this work to solve this problem. First, the supersonic region and shock location must be fully enclosed within a local grid. In this work, where the flow is only slightly supercritical, this is achieved by extending the boundary layer grid a sufficient distance into the free stream to fully surround the supersonic zone. Ideally, and definitely for flow fields with large supersonic zones, a separate grid would be generated for this purpose as mentioned above. Either way, the shock will not be required to cross a grid boundary.

Second, a two-dimensional bicubic Lagrangian interpolation scheme will be used to reduce the interpolation error between the grids. This

interpolation scheme is described in the next section: Section 2.2 then presents the far-field boundary conditions used by the code. Section 2.3 presents some details of applying the method to a representative problem.

2.1 Interpolation Scheme

The interpolation scheme has been set up to interpolate a function from one generalized curvilinear coordinate system to another. Figure 2.1 shows an example of how one physical coordinate system, (x,y) , is mapped into the computational plane, (ξ,η) . Figure 2.2 shows parts of two grids used here as an example of how the interpolation is set up. Since the mapping between the two coordinate systems is, in general, unknown and nonlinear, a Newton iteration scheme is used to find the correct coefficients for bicubic patches to approximately map one system to the other. Bicubic patches have been used in order to retain sufficient accuracy in the interpolation process. Brandt (1980a) recommended using an interpolation scheme of order $m+p$ in his multigrid approach, where m is the order of the highest derivative in the equations and p is the order of the approximation to the equations. Applying this rule in the present work indicates using bicubic patches. The scheme has been tested and optimized and should provide fourth-order accuracy with minimal cost in computer time and memory.

The approach is as follows. Suppose that some function Q_1 is known as $Q_1(X_1(\xi,\eta), Y_1(\xi,\eta))$, where (X_1, Y_1) are the physical coordinates of grid 1 and (ξ, η) are the coordinates in the corresponding computational plane. First of all, each point in grid 2 is located in grid 1 with a nearest neighbor search routine. The mappings $X_1(\xi, \eta)$, $Y_1(\xi, \eta)$, and $Q_1(\xi, \eta)$ are found approximately using a local bicubic surface to represent the evenly spaced data.

Figure 2.2 shows an example of two generalized coordinate systems and a bicubic patch imposed on grid 1. Grid 1 is shown using dotted lines except where the bicubic patch is imposed in solid lines. The origin for the patch, O , and the unknown point, P , in grid 2 are indicated. The point in grid 1 found in the nearest neighbor search is used as the origin for the bicubic surface and as an initial guess to a Newton type zero finding routine. Let (ξ, η) be the coordinates of the nearest neighbor point and $(\ddot{\xi}, \ddot{\eta})$ the new estimates relative to (ξ, η) for the location of the grid 2 point in the computational plane. Then $X_1(\xi, \eta)$ and $Y_1(\xi, \eta)$ are found, as well as the partial derivatives with respect to ξ and η at that point. This new data is used in the Newton routine to drive the following functions to zero,

$$f(\ddot{\xi}, \ddot{\eta}) = x' - X_1(\ddot{\xi}, \ddot{\eta}) \quad g(\ddot{\xi}, \ddot{\eta}) = y' - Y_1(\ddot{\xi}, \ddot{\eta}) \quad (2.4)$$

where (x', y') are the physical coordinates of the current point in grid 2. After each step, the $(\ddot{\xi}, \ddot{\eta})$ are updated. At convergence, the coordinates $(\ddot{\xi}, \ddot{\eta})$ of the current grid 2 point in the computational domain are known. Using the mapping $Q_1(\xi, \eta)$ found earlier, the value $Q_2(\ddot{\xi}, \ddot{\eta})$ can now be found.

Several different nearest neighbor routines were tried in the course of this work. For interpolation accuracy, a routine was needed which ensured that the grid 2 point was inside the quadrilateral with the nearest neighbor as one of the vertices. Grid skewness and curvature prevented a minimum physical distance routine from dependably providing the correct quadrilateral. Oftentimes, the nearest neighbor was actually across several grid lines, reducing the accuracy of the interpolation routine. An attempt at estimating the distances in the computational plane met with the same obstacle. Finally, a

modified version of the above interpolation scheme was used with bilinear patches instead of bicubic patches. The compact support of the bilinear patches provided positive data as to the location of the grid 2 point in relation to the local origin of the patch. An initial guess was made for the origin and local coordinates were set up to the grid 2 point. If the point was not within the four sides of the patch, then the guess for the origin was moved in the direction of the grid 2 point and the process was repeated until the point was within the confines of the quadrilateral. This scheme works very well and has the added benefit that the coding is the same as for the bicubic patches used in the interpolation. Once the nearest neighbor is found such that it is the lower left vertex of the quadrilateral containing the grid 2 point, the bicubic interpolation can proceed using that point as its origin. The local patch is set up such that the grid 2 point is always in the middle cell of the patch, except at the boundaries.

In the present work, the grids do not move relative to each other. Therefore, the nearest neighbor search routine and the Newton iterations only need be performed during code initialization. For each node in each grid, (ξ, η) and (ξ, η) are found and saved. Then, whenever the interpolation process needs to be performed, the new coefficients for the mapping $Q_1(\xi, \eta)$ are found, based on (ξ, η) , and the interpolation is executed using (ξ, η) as the local coordinate value. Note that the mappings are local and, therefore, are independent of each other. The process of inverting the 16x16 matrix to find the coefficients for each mapping can be vectorized over the number of points in the streamwise direction. Also, since the mappings are based on local coordinate systems with (ξ, η) as their origin, the matrix is the same for

all points. Only the right-hand sides change. Therefore, the matrix is factored once and stored. The coefficients are found using this factorization, which improves the speed of the process.

A test case was run to determine the overhead for the interpolation routines. A 129x33 global grid was interacted with a 239x37 local grid. The setup routines required 1.1 seconds of CPU time on the NASA-Ames CRAY X-MP/48. Each subsequent call to the interpolation routines for interpolating up or down required an average of 0.056 CPU seconds. Forming the forcing function required an additional 0.042 CPU seconds per call. Totaling over the length of the calculation gives a total overhead of about 2.6 CPU seconds or approximately 0.25% of the total CPU time for the job. Even if a much shorter job was being run, the total overhead due to the interpolation process amounts to less than 1.0% of the total CPU time.

2.2 Boundary Conditions

The boundary conditions being used in the code are very similar to those used in the original code (Pulliam 1984). The wall and wake conditions are the same. The far field characteristic-like boundary conditions are slightly modified so they may also be used for the local grid. As used by Pulliam (1984), the locally one-dimensional Riemann invariants are

$$R_1 = V_n - \frac{2a}{\gamma-1} \quad R_2 = V_n + \frac{2a}{\gamma-1} \quad (2.5)$$

where V_n is the velocity normal to the far field boundary. For a subsonic boundary where $\eta = \eta_{\max}$, R_1 is an incoming characteristic evaluated from free stream conditions and R_2 is an outgoing characteristic evaluated using conditions within the domain. For an inflow boundary, where $V_n < 0$, R_1 , the

tangential velocity, V_t , and an approximation to the entropy, $S = P/\rho^\gamma$, are all specified from free stream conditions, while R_2 is extrapolated from the interior. For outflow, where $V_n > 0$, R_2 , V_t and S are extrapolated from the interior, while R_1 is then specified from free stream conditions. From these quantities, the four conservative variables ρ , ρu , ρv , and e can be found.

In the present work, this approach has been modified slightly. The Riemann invariants, as shown above, are found by assuming a locally 1-D flow and neglecting the derivatives perpendicular to this direction. Though these terms may be small if the outer boundary is far enough away from the body, they will not be small for a boundary that is near the body, as is the case in this study. However, these terms are tedious to evaluate. In order to better account for these cross terms without having to evaluate them, the governing equations are linearized about the previous time step and dealt with in perturbation form.

Starting with the equations of motion of unsteady flow in two dimensions

$$\frac{\partial \rho}{\partial t} + u \frac{\partial \rho}{\partial x} + \rho \frac{\partial u}{\partial x} + v \frac{\partial \rho}{\partial y} + \rho \frac{\partial v}{\partial y} = 0 \quad (2.6a)$$

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial x} = 0 \quad (2.6b)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial y} = 0 \quad (2.6c)$$

$$\frac{\partial S}{\partial t} + u \frac{\partial S}{\partial x} + v \frac{\partial S}{\partial y} = 0 \quad (2.6d)$$

with the speed of sound defined by

$$a^2 = \left[\frac{\partial P}{\partial \rho} \right]_s \quad (2.7)$$

and assuming isentropic flow of a perfect gas, with locally 1-D Riemann invariants defined as

$$R_1 = u - \frac{2a}{\gamma-1} \quad R_2 = u + \frac{2a}{\gamma-1} \quad (2.8)$$

the above equations yield

$$\frac{\partial R_1}{\partial t} + (u-a) \frac{\partial R_1}{\partial x} = -v \frac{\partial u}{\partial y} + \frac{a}{\rho} \frac{\partial \rho v}{\partial y} \quad (2.9a)$$

$$\frac{\partial R_2}{\partial t} + (u+a) \frac{\partial R_2}{\partial x} = -v \frac{\partial u}{\partial y} - \frac{a}{\rho} \frac{\partial \rho v}{\partial y} \quad (2.9b)$$

Instead of neglecting the right hand sides (RHS) of these two equations to get the original boundary conditions, in Eqn. (2.9b), for example, set $R_2 = \bar{R}_2 + \Delta R_2$, which gives

$$\begin{aligned} \frac{\partial \bar{R}_2}{\partial t} + \frac{\partial \Delta R_2}{\partial t} + (u+a) \left[\frac{\partial \bar{R}_2}{\partial x} + \frac{\partial \Delta R_2}{\partial x} \right] \\ = -v \frac{\partial u}{\partial y} - \frac{a}{\rho} \frac{\partial \rho v}{\partial y} \end{aligned} \quad (2.10)$$

Assume that $\bar{R}_2$ satisfies the original equation, this gives

$$\frac{\partial \Delta R_2}{\partial t} + (u+a) \frac{\partial \Delta R_2}{\partial x} = 0 \quad (2.11)$$

from which it follows that

$$\Delta R_2 = \Delta u + \frac{2\Delta a}{\gamma-1} \quad \text{along} \quad \frac{dx}{dt} = u+a \quad (2.12a)$$

and similarly

$$\Delta R_1 = \Delta u - \frac{2\Delta a}{\gamma-1} \quad \text{along} \quad \frac{dx}{dt} = u-a \quad (2.12b)$$

are constants. For an outer boundary, exchanging Δu for ΔV_n , ΔR_1 is

always evaluated using information from outside the boundary and ΔR_2 is evaluated one gridline in from the boundary. Solving for ΔV_n and Δa gives

$$\Delta V_n = \frac{\Delta R_1 + \Delta R_2}{2} \quad \Delta a = \frac{\gamma-1}{2} \frac{\Delta R_2 - \Delta R_1}{2} \quad (2.13)$$

under general conditions. V_t and s are calculated from

$$V_t^N = V_t^{N-1} + \Delta V_t \quad S^N = S^{N-1} + \Delta S \quad (2.14)$$

where, for example,

$$\Delta V_t = V_t^N - V_t^{N-1} \quad (2.15)$$

is evaluated according to the following conditions. For inflow, $V_n < 0$, so ΔV_t and ΔS are evaluated using information from outside the boundary. For outflow, $V_n > 0$, so ΔV_t and ΔS are evaluated one gridline in from the boundary. Once ΔV_t and ΔS are found, they are used in the following expressions to find $\Delta \rho$ and ΔP .

$$\Delta \rho = \frac{1}{\gamma-1} \left[\frac{a^2 S}{\gamma} \right]^{\frac{1}{\gamma-1}-1} \left[\frac{a^2 \Delta S}{\gamma} + \frac{2a \Delta a S}{\gamma} \right] \quad (2.16a)$$

$$\Delta P = \frac{a^2 \Delta \rho}{\gamma} + \frac{2a \Delta a \rho}{\gamma} \quad (2.16b)$$

The new values for P and ρ can now be found using equations analogous to Eqn. (2.14) and the other conservative variables can be formed from ρ , V_n , V_t , and P . Note that when the conditions outside the boundary do not change, $\Delta R_1 = 0$ and V_t and S do not change for inflow boundaries.

These boundary conditions give results virtually identical to the original versions when used for far field conditions in a single grid calculation. However, their formulation allows them to be used for the outer boundaries

of local grids as well where the initial boundary conditions are interpolated from the global solution and ΔR_1 is always zero. Extrapolation conditions are used for the downstream outflow boundaries.

These far field boundary conditions are designed only for subsonic boundaries and are used for both the global and the local grids. The same wall conditions are also used on both grids. For viscous runs, the velocity is set to zero at the body to enforce no-slip and no-penetration conditions. The pressure and density are each extrapolated from one point above the boundary using zeroth order extrapolation. Though it may seem to be overspecifying the problem to use viscous boundary conditions on the global grid when solving the Euler equations, the conditions are consistent with the final solution. As of yet, no problems have been encountered due to using these viscous conditions. However, some effort in the future should be spent on replacing them with something more consistent with the equations being solved.

2.3 Implementation of the Scheme

This section describes some aspects of applying the method to a problem once it has been defined. Since this method is independent of the solvers being used, the following description will be discussed generally. Two grids will be used, the global or coarse grid, and the local or fine grid. A set of equations will be solved on each grid and interacted together. These equations will not necessarily be the same. In order to solve these sets of equations, an appropriate solver must be used. The solver for the equations being solved on the global grid is designated S1 and the solver for the local grid equations is designated S2.

Chapter Four presents results of two variations of the above arrangement. The first results presented constitute a global, coarse grid and a fine grid which is local to the wall and near wake region. The Navier-Stokes equations are solved on the boundary layer type fine grid and the Euler equations are solved on the global grid. The second situation consists of a global grid and a refined grid local to the nose of the airfoil. Since the local grid is not covering the full viscous region, the global equations must also be the Navier-Stokes equations. In this case, the local grid is serving to provide extra resolution in a particular region of the flow field. Figure 2.3 shows the grid topology for each of the two cases along with the global grid. In each plot, the local grid stands out from the global grid.

Using some type of grid generator, one can generate the global grid so that it will give some insight as to the general nature of the flow field, but will not necessarily have the resolution required to give an accurate solution. The solution can then be found on this grid using S2. Then, a half grid is generated containing every other point in the coarse grid in both directions. This grid is then used to estimate the truncation error distribution in the field in the following way.

The coarse grid solution is interpolated to the half grid using the interpolation scheme described in Section 2.1. Then, the solution is operated on once by the discretized S1 operator. The resulting residual is an estimate of the truncation error incurred in the coarse grid solution due to grid resolution and/or differences in the equation sets being solved. This residual can be plotted and inspected. Figure 2.4 shows an example of the truncation error distribution found for one of the cases discussed in Chapter Four. The

negative logarithm of the magnitude of the truncation error is plotted. Once this distribution is known, a local fine grid can be generated which will cover the regions which contain error greater than some predetermined tolerance. Again, this grid can be generated by any means. Optionally, one could use pattern recognition schemes from the computer science field and have a routine generate the grid required. Berger (1982) has taken this approach, but it is beyond the scope of this dissertation.

The calculation now proceeds using both the fine and coarse grids. Computations alternate between the two grids. The global grid solution proceeds first using solver S1 in order to initialize the flow field. After sufficient initialization, the solution is interpolated down to the fine grid. Computation now proceeds on the fine grid using solver S2 and boundary conditions from the global grid as discussed in Section 2.2. After some number of iterations on the fine grid, the calculation is stopped and the fine grid solution is interpolated up to the global grid. The solution in the regions of the global grid not covered by the fine grid is not modified in any way by the interpolation process.

This interpolated solution is then operated on once by the operator for solver S1 to generate the forcing function. This function is zeroed outside of the region common to both grids and then stored. Calculation now proceeds on the global grid, again using solver S1. However, at each step, the forcing function is added to the RHS of the calculation so that the global solution is influenced by the fine grid solution.

The computation proceeds in this manner for several up and down cycles until convergence is reached or an acceptably converged solution is

achieved. The number of iterations for each grid and the number of cycles to use was studied to determine the optimal values. The more cycles there are, with fewer iterations per cycle, the faster the convergence of the two solutions to a common solution. However, each interpolation process introduces error into the solution which must be damped out before the real solution can actually proceed with convergence. It took approximately 50 iterations to damp out this error. Figure 2.5 shows a convergence history for a typical calculation. The root mean square (RMS) residual for the conservation of mass equation is plotted versus equivalent fine grid iterations where the root mean square of some function f is defined as

$$RMS(f) = \left[\frac{1}{N} \sum_{n=1}^N f_n^2 \right]^{\frac{1}{2}} \quad (2.17)$$

Each of the upward spikes is an interpolation process. The narrow regions between the spikes are the residual for the coarse grid. The broader regions between the spikes are the residual for the fine grid.

Most of the results presented later were run with the sequencing input shown in Table 2.1. In this table, the first two columns indicate the size of the grid on which iteration will be conducted, the third column indicates how many iterations to perform for each grid, and the fourth column contains the scaling factor for the spatially variable time step. The fifth column contains values for LQ, the grid index for the grid on which iteration was last performed. LQTO is the grid index for the grid to be iterated on this time. If LQ and LQTO are equivalent, then no interpolation takes place. Otherwise, interpolation occurs from grid LQ to grid LQTO. The global grid is always grid 1.

TABLE 2.1 SAMPLE JCL FILE SEQUENCING INPUT - ORIGINAL

SAMPLE JOB SEQUENCING INPUT - ORIGINAL					
JMXI	KMXI	IENDS	DTISEQ	LQ	LQTO
129	33	400	1.0	1	1
245	37	400	1.0	1	2
129	33	400	3.0	2	1
245	37	400	3.0	1	2
129	33	400	5.0	2	1
245	37	400	5.0	1	2
129	33	400	5.0	2	1
245	37	400	5.0	1	2
129	33	600	5.0	2	1
245	37	600	5.0	1	2
129	33	800	5.0	2	1
245	37	800	5.0	1	2
129	33	1000	5.0	2	1
245	37	1000	5.0	1	2
129	33	1000	5.0	2	1

Figure 2.6 shows that the solution did not change very much in the later cycles. The case shown here is a viscous-inviscid interaction run using a 129x33 global grid and a 245x37 local grid. The free stream Mach number is 0.676 and the Reynolds number is 5.7×10^6 . The angle of attack is 1.93 degrees, producing a small supersonic zone near the nose of the airfoil. Figures 2.6a-2.6c shows plots of C_L , C_D , and C_M versus equivalent fine grid iterations. The cyclic differences in each plot in the later iterations is in accordance with which grid the calculations are proceeding on. Note that each of these cycles does not change as the calculation proceeds. Therefore, sequencings similar to that in Table 2.2 were used in the end. The cycles with 200 iterations per grid served to synchronize the solutions and the final cycle with 800 iterations per grid served to converge the solutions to reasonable levels.

TABLE 2.2 SAMPLE JCL FILE SEQUENCING INPUT - REVISED

SAMPLE JOB SEQUENCING INPUT - REVISED					
JMXI	KMXI	IENDS	DTISEQ	LQ	LQTO
129	33	200	1.0	1	1
245	37	200	1.0	1	2
129	33	200	3.0	2	1
245	37	400	3.0	1	2
129	33	400	5.0	2	1
245	37	400	5.0	1	2
129	33	400	5.0	2	1
245	37	400	5.0	1	2
129	33	400	5.0	2	1
245	37	800	5.0	1	2
129	33	800	5.0	2	1

In order to save CPU time and memory, the solver handles data for only one grid at a time. The data for the other grids must be saved when it is not being used. Also, the interpolation routines require some storage overhead. Most of this overhead could be absorbed into existing scratch space in the solver since the interpolation and solution routines will not be running at the same time. In all, extra memory equivalent to 25 times the maximum number of points used in any one of the grids is used. This is rather excessive and could be reduced significantly by saving the data on disk instead of in core, as is done now. Since the grids are exchanged only about 20 times during a normal calculation, the extra I/O time would not be excessive.

This chapter has served to present the building blocks for the truncation error injection scheme. The interpolation scheme, boundary conditions, and injection methodology are put together with the appropriate solvers to build the full scheme. The next chapter serves to introduce the solver used in this work.

Figure 2.1 Mapping of generalized curvilinear coordinate system in physical space to computational space.

Figure 2.2 Two generalized curvilinear coordinate systems and bicubic patch for interpolation.

Figure 2.3 Local grid topologies. Boundary layer type grid and grid local to nose of airfoil.

Figure 2.4 Truncation error distribution for the conservation of mass equation. Case VRGA2A. Flow field 1.

RMS DENSITY RESIDUAL

VTK37C - BOUNDARY - 16

Figure 2.5 Convergence history. RMS Density residual vs. equivalent fine grid iterations. Case VTK37.

Figure 2.6a Convergence history. Coefficient of Lift, C_L , vs. equivalent fine grid iterations. Case VTK37.

COEFFICIENT OF DRAG

VTK37C - BOUNDARY - 16

Figure 2.6b Convergence history. Coefficient of Drag, C_D , vs. equivalent fine grid iterations. Case VTK37.

COEFFICIENT OF PITCHING MOMENT

VTK37C - BOUNDARY - 16

Figure 2.6c Convergence history. Coefficient of Pitching Moment, C_M , vs. equivalent fine grid iterations. Case VTK37.

CHAPTER 3

DEVELOPMENT OF EQUATIONS

The governing equations used in this work are the two-dimensional Navier-Stokes equations in strong conservation form. Strong conservation form is chosen to allow the scheme to capture shocks as opposed to fitting them. The development of the equations in this chapter follows that of Pulliam (1984).

The equations can be written in nondimensional form as

$$\partial_t Q + \partial_x E + \partial_y F = \text{Re}^{-1} (\partial_x E_v + \partial_y F_v) \quad (3.1)$$

where

$$\begin{aligned} Q &= \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix} & E &= \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(e+p) \end{bmatrix} & F &= \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(e+p) \end{bmatrix} \\ E_v &= \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ f_4 \end{bmatrix} & F_v &= \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ g_4 \end{bmatrix} \end{aligned} \quad (3.2)$$

with

$$\begin{aligned} \tau_{xx} &= \mu (4u_x - 2v_y) / 3 \\ \tau_{xy} &= \mu (u_y + v_x) \\ \tau_{yy} &= \mu (-2u_x + 4v_y) / 3 \\ f_4 &= u \tau_{xx} + v \tau_{xy} + \mu Pr^{-1} (\gamma - 1)^{-1} \partial_x a^2 \\ g_4 &= u \tau_{xy} + v \tau_{yy} + \mu Pr^{-1} (\gamma - 1)^{-1} \partial_y a^2 \end{aligned} \quad (3.3)$$

Pressure is related to the conservative flow variables, Q , by the equation of state

$$p = (\gamma - 1) \left[e - \frac{1}{2} \rho (u^2 + v^2) \right] \quad (3.4)$$

where γ is the ratio of specific heats, generally taken as 1.4. The speed of sound, a , for ideal fluids is $a^2 = \gamma p / \rho$. The dynamic viscosity, μ , is typically made up of a constant plus a computed turbulent eddy viscosity. Also, Stokes' hypothesis has been assumed in the above equations, Re is the Reynolds number and Pr is the Prandtl number.

The variables are nondimensionalized as follows

$$\tilde{\rho} = \frac{\rho}{\rho_\infty}, \quad \tilde{u} = \frac{u}{a_\infty}, \quad \tilde{v} = \frac{v}{a_\infty}, \quad \tilde{e} = \frac{e}{\rho_\infty a_\infty^2} \quad (3.5)$$

where ∞ refers to free stream conditions. The reference length used here is the chord of the airfoil and the velocities have been scaled by the free stream speed of sound instead of the free stream velocity. The viscous parameters are scaled as

$$\tilde{\mu} = \frac{\mu}{\mu_\infty}, \quad Re = \frac{\rho_\infty l a_\infty}{\mu_\infty} \quad (3.6)$$

Note that the Reynolds number is scaled by the free stream speed of sound instead of the free stream velocity. The traditional Reynolds number can be recovered by multiplying by the free stream Mach number, $M_\infty = u_\infty / a_\infty$. For convenience, the tildes will be dropped through the rest of the development. Also, note that the Euler equations result when $Re \rightarrow \infty$ and the right-hand side of Eqn. (3.1) goes to zero.

3.1 Generalized Curvilinear Coordinate Transformations

The Navier-Stokes equations are now transformed from Cartesian coordinates to a general curvilinear coordinate system. This transformation eases the application to complex geometries, especially with regard to wall boundary conditions. Also, it is easier to cluster points where they are needed using the curvilinear coordinate system. In the transformation used here, the computational space is Cartesian with unit cell size, and has a rectangular domain regardless of the shape of the physical (x,y) domain. The transformation can be written as

$$\begin{aligned}\tau &= t \\ \xi &= \xi(x,y) \\ \eta &= \eta(x,y)\end{aligned}\tag{3.7}$$

These expressions are used to transform the above equations. The resulting transformed equations remain in strong conservation form and can be written as

$$\partial_\tau \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} = \text{Re}^{-1} (\partial_\xi \hat{E}_v + \partial_\eta \hat{F}_v)\tag{3.8}$$

where

$$\begin{aligned}\hat{Q} &= J^{-1} \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, & \hat{E} &= J^{-1} \begin{bmatrix} \rho U \\ \rho u U + \xi_x p \\ \rho v U + \xi_y p \\ U(e+p) \end{bmatrix}, \\ & & \hat{F} &= J^{-1} \begin{bmatrix} \rho V \\ \rho u V + \eta_x p \\ \rho v V + \eta_y p \\ V(e+p) \end{bmatrix}\end{aligned}\tag{3.9}$$

with

$$U = \xi_x u + \xi_y v, \quad V = \eta_x u + \eta_y v, \quad J = \xi_x \eta_y - \xi_y \eta_x. \quad (3.10)$$

Here, U and V are the contravariant velocities perpendicular to the η and ξ coordinate lines, respectively, and J is the Jacobian of the transformation.

The viscous flux terms can be written as

$$\hat{E}_v = J^{-1} (\xi_x E_v + \xi_y F_v), \quad \hat{F}_v = J^{-1} (\eta_x E_v + \eta_y F_v) \quad (3.11)$$

with the transformed stress terms

$$\begin{aligned} \tau_{xx} &= \mu (4(\xi_x u_\xi + \eta_x u_\eta) - 2(\xi_y v_\xi + \eta_y v_\eta)) / 3 \\ \tau_{xy} &= \mu (\xi_y u_\xi + \eta_y u_\eta + \xi_x v_\xi + \eta_x v_\eta) \\ \tau_{yy} &= \mu (-2(\xi_x u_\xi + \eta_x u_\eta) + 4(\xi_y v_\xi + \eta_y v_\eta)) / 3 \\ f_4 &= u \tau_{xx} + v \tau_{xy} + \mu Pr^{-1} (\gamma - 1)^{-1} (\xi_x \partial_\xi a^2 + \eta_x \partial_\eta a^2) \\ g_4 &= u \tau_{xy} + v \tau_{yy} + \mu Pr^{-1} (\gamma - 1)^{-1} (\xi_y \partial_\xi a^2 + \eta_y \partial_\eta a^2) \end{aligned} \quad (3.12)$$

where terms such as u_x have been expanded by the chain rule.

3.2 Thin-Layer Approximation

In high Reynolds number flows, the effects of viscosity are important only near wall boundaries. Prandtl, recognizing this, developed his famous boundary layer equations. However, Prandtl neglected all but one term in the normal component of the momentum equation, setting pressure constant across the boundary layer. Consequently, the equations are singular at a separation point, i.e. where the local wall shear stress goes to zero. When scaling the normal component of the momentum equation, terms of at least three different orders appear. Prandtl eliminated all but the highest order term, the normal derivative of the pressure, therefore setting it equal to zero. We can back off one step and keep the terms of the two highest orders. The result is the thin-layer Navier-Stokes equations which can be written as

$$\partial_\tau \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} = \text{Re}^{-1} \partial_\eta \hat{S}_v \quad (3.13)$$

with the viscous flux term being

$$\hat{S}_v = J^{-1} \begin{bmatrix} 0 \\ \eta_x m_1 + \eta_y m_2 \\ \eta_x m_2 + \eta_y m_3 \\ \eta_x (um_1 + vm_2 + m_4) + \eta_y (um_2 + vm_3 + m_5) \end{bmatrix} \quad (3.14)$$

and

$$\begin{aligned} m_1 &= \mu (4\eta_x u_\eta - 2\eta_y v_\eta) / 3 \\ m_2 &= \mu (\eta_y u_\eta + \eta_x v_\eta) \\ m_3 &= \mu (-2\eta_x u_\eta + 4\eta_y v_\eta) / 3 \\ m_4 &= \mu Pr^{-1} (\gamma - 1)^{-1} \eta_x \partial_\eta a^2 \\ m_5 &= \mu Pr^{-1} (\gamma - 1)^{-1} \eta_y \partial_\eta a^2 \end{aligned} \quad (3.15)$$

As can be seen, the thin-layer version of the Navier-Stokes equations retains the normal momentum equation. The terms that are eliminated from the full equations are all the viscous terms in the streamwise direction.

In a typical calculation, the grid is heavily clustered normal to all solid surfaces in order to resolve the boundary layer, but the grid lines perpendicular to the surface are widely spaced. Even if the full Navier-Stokes equations were programmed in, the streamwise viscous terms would not be accurately resolved. Also, these terms are negligible in attached and mildly separated flows. Pulliam(1984) lists three requirements for using the thin-layer approximation:

1. All body surfaces must be mapped onto $\eta = \text{constant}$ surfaces.
2. Grid spacing is clustered at the body surfaces such that sufficient resolution for a particular Reynolds number is obtained. (At least one or

two grid points in the sublayer).

3. All the viscous derivatives in the ξ -direction are neglected, while the terms in the η -direction are retained. All of the inviscid terms are used.

The thin-layer approximation can break down for low Reynolds number flow and for flow with massive separation regions.

3.3 Numerical Method

The finite difference scheme used to solve Eqn. (3.13) is the diagonal implicit algorithm of Pulliam and Chaussee (1981). This scheme is an extension of the implicit approximate factorization algorithm of Beam and Warming (1976) in delta form. An implicit scheme is used to avoid the excessive time step restrictions inherent in explicit formulations. In computing transonic, viscous flows, highly clustered grids are used to resolve the boundary layer and shocks in the flow field. These fine grids result in very stiff problems requiring small time steps for stability using an explicit scheme. An implicit scheme lessens or eliminates these restrictions.

As implemented here, the Beam-Warming scheme uses first-order accurate Euler implicit time differencing and second-order accurate spatial differencing. The scheme is an approximate factorization dividing the operator into a ξ -inversion and an η -inversion. Eqn. (3.16) shows the scheme in delta form

$$\begin{aligned} & \left[I + h\delta_{\xi}\hat{A}^n \right] \left[I + h\delta_{\eta}\hat{B}^n - hRe^{-1}\delta_{\eta}J^{-1}\hat{M}^n \right] \Delta\hat{Q}^n \\ & = -h \left[\delta_{\xi}\hat{E}^n + \delta_{\eta}\hat{F}^n - Re^{-1}\delta_{\eta}\hat{S}^n \right] = \hat{R}^n \end{aligned} \quad (3.16)$$

where h is the time step, $\hat{R}^n$ is the residual or RHS, and the central

difference operator, δ , is defined as

$$\delta_{\xi}\hat{Q} = \frac{1}{2\Delta\xi} \left[\hat{Q}(\xi+\Delta\xi,\eta) - \hat{Q}(\xi-\Delta\xi,\eta) \right] \quad (3.17)$$

The term "delta form" comes from the manner in which the solution vector, $\hat{Q}^n$, shows up as $\Delta\hat{Q}^n = \hat{Q}^{n+1} - \hat{Q}^n$. The Jacobian matrices $\hat{A}^n$ and $\hat{B}^n$ are found by linearizing the flux vectors $\hat{E}^{n+1}$ and $\hat{F}^{n+1}$ in time such that

$$\hat{E}^{n+1} = \hat{E}^n + \hat{A}^n \Delta\hat{Q}^n + O(\Delta\tau^2) \quad (3.18a)$$

$$\hat{F}^{n+1} = \hat{F}^n + \hat{B}^n \Delta\hat{Q}^n + O(\Delta\tau^2) \quad (3.18b)$$

where

$$\hat{A} = \frac{\partial \hat{E}}{\partial \hat{Q}} \quad \text{and} \quad \hat{B} = \frac{\partial \hat{F}}{\partial \hat{Q}} \quad (3.19)$$

are the flux Jacobian matrices. These flux Jacobians and the viscous coefficient matrix, $\hat{M}$, which comes from the time linearization of the viscous vector $\hat{S}^{n+1}$, are documented in Appendix A.

Solving Eqn. (3.16) requires multiple inversions of a 4x4 block tridiagonal-matrix, which constitutes the major fraction of the work in the Beam-Warming algorithm. The diagonal version of this algorithm due to Pulliam and Chaussee (1981) eliminates the need for block tridiagonal inversions by using similarity transformations to diagonalize the Jacobian matrices $\hat{A}^n$ and $\hat{B}^n$. The result is scalar pentadiagonal matrices which are computationally more efficient.

Each Jacobian matrix has a set of eigenvalues and a complete set of distinct eigenvectors. Therefore, similarity transformations can be used to diagonalize $\hat{A}$ and $\hat{B}$ as,

$$\Lambda_{\xi} = T_{\xi}^{-1} \hat{A} T_{\xi} \quad \text{and} \quad \Lambda_{\eta} = T_{\eta}^{-1} \hat{B} T_{\eta} \quad (3.20)$$

where Λ_ξ , Λ_η , T_ξ and T_η are also given in Appendix A. Substituting these transformations into Eqn. (3.16) results in

$$\begin{aligned} & \left[T_\xi T_\xi^{-1} + h \delta_\xi (T_\xi \Lambda_\xi T_\xi^{-1}) \right] \left[T_\eta T_\eta^{-1} + h \delta_\eta (T_\eta \Lambda_\eta T_\eta^{-1}) \right] \Delta \hat{Q}^n \\ & = \hat{R}^n \end{aligned} \quad (3.21)$$

which is still equivalent to Eqn. (3.16), except for the implicit viscous term. The viscous flux vector, $\hat{M}^n$, does not diagonalize along with the convective flux vectors $\hat{A}$ and $\hat{B}$. Therefore, in the method being used here, the implicit viscous terms are discarded completely. Although this approach seems a little drastic, Pulliam (1984) has shown that it results in a fast, robust code. Pulliam also discusses some other alternatives to discarding the implicit viscous terms. Note that the explicit viscous terms are still used, so the converged solution will be the same as that in Eqn. (3.16).

Now, returning to Eqn. (3.21), factoring the eigenvector matrices outside the spatial operators yields

$$T_\xi \left[I + h \delta_\xi \Lambda_\xi \right] \hat{N} \left[I + h \delta_\eta \Lambda_\eta \right] T_\eta^{-1} \Delta \hat{Q}^n = \hat{R}^n \quad (3.22)$$

where $\hat{N} = T_\xi^{-1} T_\eta$, (see Appendix A). Note that the eigenvector matrices are functions of space and therefore the accuracy of the scheme is reduced to, at best, first order in time. However, the RHS of Eqn. (3.22) is the same as in Eqn. (3.16) so the converged solution is still second-order accurate in space. Eqn. (3.22) is the final version of the scheme. This diagonal algorithm results in an overall savings in computational work of as high as 40%.

A linear stability analysis for the implicit scheme (Beam and Warming 1976) shows that it is unconditionally stable, but there are actually stability bounds due to nonlinear effects. This is especially true in strongly nonlinear

cases, such as flows with shocks.

3.4 Artificial Dissipation Model

When flow situations, such as shocks or high Reynolds number viscous behavior, are numerically calculated, scales of motion are generated which are too fine for a grid to resolve. These scales are generated by the nonlinear interactions in the convection terms of the momentum equations. In these interactions, high wavenumber components are generated which can cause problems. Physically, they represent shocks or viscous dissipation of the very high wavenumbers. Numerically, they can cause problems by either aliasing onto the lower wavenumber components, which can be resolved, or by continually building up at the upper reaches of the grid's resolution capability. Either way, serious problems with accuracy and stability can occur.

The most common way of dealing with these high wavenumber components is to add some form of numerical dissipation to the algorithm which introduces an error level that does not compromise the physical viscous dissipation. In the Pulliam-Chaussee version of the code being used here, this dissipation takes the form of a nonlinear artificial dissipation model by Jameson, Schmidt, and Turkel (1981). This model is a combination of second- and fourth-order dissipation terms and, for the ξ -inversion step, can be written as

$$\nabla_{\xi} \left[\sigma_{j+1,k} J_{j+1,k}^{-1} + \sigma_{j,k} J_{j,k}^{-1} \right] \left[\epsilon_{j,k}^{(2)} \Delta_{\xi} - \epsilon_{j,k}^{(4)} \Delta_{\xi} \nabla_{\xi} \Delta_{\xi} \right] Q_{j,k} \quad (3.23)$$

with

$$\epsilon_{j,k}^{(2)} = \kappa_2 \Delta t \max(\chi_{j+2,k}, \chi_{j+1,k}, \chi_{j,k}, \chi_{j-1,k}) \quad (3.24a)$$

$$\chi_{j,k} = \frac{|p_{j+1,k} - 2p_{j,k} + p_{j-1,k}|}{|p_{j+1,k} + 2p_{j,k} + p_{j-1,k}|} \quad (3.24b)$$

$$\epsilon_{j,k}^{(4)} = \max(0, \kappa_4 \Delta t - \epsilon_{j,k}^{(2)}) \quad (3.24c)$$

$$\sigma_{j,k} = |U| + a \sqrt{\xi_x^2 + \xi_y^2} \quad (3.24d)$$

where typical values for the constants are $\kappa_2 = 1/4$ and $\kappa_4 = 1/256$. The term $\sigma_{j,k}$ is the spectral radius of $\hat{A}$. Similiar terms are used in the η -direction.

The first term in Eqn. (3.23) is a second-order dissipation term with a coefficient which varies with the pressure gradient and is used to increase the dissipation around shocks. The second term is a fourth-order dissipation term with a constant coefficient. This term is turned off when the coefficient for the second-order dissipation is larger than that for the fourth-order dissipation. This occurs in the vicinity of a shock where the pressure gradient term, $\chi_{j,k}$, is large. Pulliam (1984) presented results that demonstrate the effectiveness of this nonlinear artificial dissipation scheme in reducing non-physical oscillations around a shock.

CHAPTER 4

VISCOUS-INVISCID INTERACTION RESULTS

Now that the building blocks for the truncation error injection scheme have been presented, the code can be applied to some interaction cases. In this chapter, two types of problems will be presented. They both consist of steady flow of air over an airfoil. The first problem consists of viscous-inviscid interaction in subcritical and slightly supercritical flow fields. In these cases, the Euler equations are solved on the global grid and the thin-layer Navier-Stokes equations are solved on the local grid. This local grid completely surrounds the airfoil and extends into the near wake region. Therefore, the entire region where viscous effects are important is covered by the local grid.

Two different flow fields will be used in studying this problem. Flow field 1 is the flow around a RAE2822 airfoil at 1.93 degrees angle of attack. The free stream Mach number is 0.64 and the Reynolds number is 5.7×10^6 . Turbulence modeling is turned on at 11% of chord using the algebraic turbulence model due to Baldwin and Lomax (1978). This flow field is fully subcritical, i.e., the local Mach number is everywhere less than 1.0. Flow field 2 is similar to flow field 1 except that the free stream Mach number is 0.676. This flow field is slightly supercritical, i.e., somewhere within the flow field the local Mach number is greater than 1.0.

The second problem consists of a relatively fine global grid and a refined local grid which only covers the near nose region of a NACA 0012 airfoil at a high angle of attack. This local grid is used to resolve the rapidly

expanding flow occurring as the flow rounds the nose of the airfoil. In this problem, the thin-layer Navier-Stokes equations are solved on both grids. Flow field 3 consists of steady flow around a NACA 0012 airfoil at 13.5 degrees angle of attack. The free stream Mach number is 0.301 and the Reynolds number is 3.91×10^6 . Turbulence modeling is turned on at the leading edge. This flow field is also slightly supercritical. A small supersonic bubble exists just behind the nose.

4.1 Effects of Truncation Error Injection

In order to efficiently use truncation error injection, here referred to as TEI, one must have some idea of where refinement is needed in the flow field in order to improve the solution. One could find the fine grid solution, interpolate it to a coarser grid, and then form the truncation error (or TE), but once the fine grid solution is known, there is no reason to continue. The goal is to ascertain which part of the domain can be effectively resolved by the coarse grid and which part of the grid requires refinement before the solution is found. One must be able to estimate the spatial distribution of the TE easily and then be able to interpret the distribution in a way that will indicate where the grid needs refinement.

Define the term $T.E._k^h$ to be the truncation error found on a grid of characteristic mesh size h using the solution found on a grid of characteristic mesh size k . The TE found using the solution from a grid with characteristic mesh size h on a half grid with characteristic mesh spacing $2h$ is designated as $T.E._h^{2h}$. This TE can be estimated by performing the desired calculation on a coarse grid and then forming the TE using an even coarser grid, i.e., forming $T.E._{2h}^{4h}$. There is a direct relationship between $T.E._k^{2k}$ and $T.E._h^{2h}$. In other

words, one can find the solution on a coarse grid with mesh spacing h , form the truncation error on a grid with mesh spacing $2h$ and relate it directly to the TE found on a grid with mesh spacing of $2k$ using a solution found on a grid with mesh spacing k . If one does a simple analysis using central differences on a cartesian grid with constant mesh spacing, one finds the following relation to first order,

$$T.E._h^{2h} = \frac{h^2}{k^2} T.E._k^{2k}, \quad (4.1)$$

confirming the relationship.

Two runs were made to test the validity of the relationship. First, a solution was found on a 257×65 fine grid for flow field 1. Pressure and Mach contours for this case are shown in Figure 4.1. Once the solution was found on the fine grid, it was interpolated to a 129×33 grid consisting of every other point from the fine grid. Then the TE distribution was found. The distribution of the TE for each of the four governing equations is shown in Figure 4.2. The plots show contours of the negative logarithm of the magnitude of the TE, so the more negative the number the larger the TE. There are no negative contours labeled in this plot because they are too close to the body. Note how the magnitude of the terms increases moving toward the body and that, although far from smooth, the distribution is somewhat monotonic.

The process was repeated using the 129×33 grid as the fine grid and a 65×17 coarse grid. Again, the coarse grid was formed by taking every other point from the fine grid. The TE distribution from this run is shown in Figure 4.3. Aside from the differences due to the differing densities of the grids, the two distributions look sufficiently similar to accept the coarse grid results

as an estimate of those on the fine grid. In fact, the coarse grid TE, $T.E. \frac{4h}{2h}$, appears to overpredict the size of the TE at any given radius from the airfoil.

Once a method was found for estimating the TE on the fine grid, the effects of injecting the error had to be determined. The forcing function $T.E. \frac{2h}{h}$ is used in solving the Euler equations on the coarse grid with only some subset of the forcing function injected. The subsets were formed by turning off the truncation error beyond some specified $\eta = \text{constant}$ line. The calculation was run until the RMS density residual dropped below 10^{-14} . Table 4.1 shows the lift, drag, and moment coefficients for the various runs as compared with the coarse grid viscous solution, VSSTRT, and the fine grid viscous solution, VRGAA.

TABLE 4.1 STRAIGHT INJECTION RUNS - GRID RGA
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 1

RESULTS - VARYING THE INJECTION REGION SIZE INJECTED COARSE GRID SOLUTIONS - FLOW FIELD 1							
CASE	K	S-S PTS	C_L	C_{DP}	C_{DF}	C_M	% LIFT ERROR
Fine and Coarse Grid Solutions							
VRGAA		0	.54648	.00376	.00646	-.08144	0.00
VSSTRT		0	.53020	.00545	.00572	-.07967	2.98
Injected Coarse Grid Solutions							
VK21	8	3	.61312	.00410	.00746	-.09540	12.19
VK23	9	1	.59831	.00441	.00710	-.09247	9.48
VK25	10	0	.58097	.01183		-.08964	6.31
VK27	11	0	.56545	.01197		-.08676	3.47
VK29	12	0	.55644	.00468	.00701	-.08413	1.82
VK31	13	0	.55112	.00458	.00698	-.08302	0.85
VK35	15	0	.54807	.00456	.00697	-.08243	0.29
VK37	16	0	.54769	.00446	.00697	-.08229	0.22
VK43	19	0	.54719	.00422	.00695	-.08189	0.13

The flow field in this example is fully subcritical. Note, however, that injecting TE only inside the eighth or ninth K-line caused a very small supersonic zone to appear in the coarse grid solution. The lift and moment coefficients are also quite different from those of the fine grid solution. Figure 4.4 compares the boundary layer profiles for the fine grid solution and for solution VK21. The longer arrows in the profile are from VK21. The line parallel to the wall is the limit of injection for this case. It is obvious that the injection region does not cover the full viscous region. Therefore, not all of the viscous effects are accounted for by the forcing function. The result is the boundary layer profile in Figure 4.4 which is too full. The flow around the leading edge of the airfoil is not damped enough by the inadequate viscous forcing terms and consequently accelerates enough to reach a supercritical state.

The solutions VK25 and VK27 diverged after only a few hundred iterations. It is not presently understood why they diverged, while VK23 and VK21 converged.

The rest of the solutions provided fairly accurate approximations to the fine grid solution with the accuracy improving as the size of the injection region was increased. Note how the lift monotonically converges to the correct value with increasing injection region size. The highest relative errors occurred for the coefficient of drag due to pressure, C_{DP} , although the highest absolute errors occurred in the lift. The injection regions for VK29 through VK35 fully covered the boundary layer on the forward part of the airfoil, but did not cover the boundary layer on the aft 10% of the airfoil nor all of the near wake region. The solutions VK37 and VK43 took into account all of the

TABLE 4.2 MAXIMUM TRUNCATION ERROR FOR EACH η -LINE

RESIDUAL FORCING FUNCTIONS FINE GRID TO COARSE GRID VISCOUS-INVISCID INTERACTION - FLOW FIELD 1			
η	MAXIMUM TRUNCATION ERROR	J	N
2	0.62×10^3	63	4
3	0.17×10^3	64	4
4	0.31×10^2	65	4
5	0.17×10^2	63	4
6	0.13×10^2	62	4
7	0.12×10^2	63	4
8	0.11×10^2	63	4
9	0.87×10^1	63	4
10	0.60×10^1	63	4
11	0.34×10^1	63	4
12	0.19×10^1	64	4
13	0.11×10^1	120	4
14	0.16×10^1	119	4
15	0.12×10^1	120	4
16	0.87×10^0	120	4
17	0.68×10^0	121	4
18	0.29×10^0	121	2
19	0.32×10^0	122	4

boundary layer and the near wake region as well.

The above results confirm that in order to get a good solution, the full boundary layer and near wake region must be included in the injection zone. From these cases, what can be deduced about the size of the truncation error required in order to get a good solution? Table 4.2 shows the maximum truncation error for each η =constant line in the coarse grid. The table also shows the J location for the maximum value and for which equation the maximum value occurred. One can see that the maximum TE almost always occurred for the energy equation. This trend has been observed in all other runs to date as well. If it is assumed that the solutions beyond VK27 provided

adequate approximations to the fine grid solution, all being within about 1% of the fine grid value for lift, then it can be deduced that TE larger than $O(1)$ must be included in the forcing function. Later runs tend to support this conclusion as well.

4.2 Characteristics of Viscous-Inviscid Interaction Runs

Now that some guidelines have been set for determining the required size of the injection region, the effect of injection in interaction runs needs to be determined. For these interaction runs, local grids were chosen based on the fine grid TE distribution, TE_h^{2h} . Table 4.3 shows the sizes of the 6 grids that were chosen and the size of the TE which they cover. Grids RGA31, RGA33 and RGA37 were added to find a lower limit on the acceptable local grid size. All these grids were taken from the 257x65 fine grid by eliminating points outside of the required region. Figure 4.5 shows grid RGA37 as compared to the 257x65 fine grid, RGA.

TABLE 4.3 LOCAL GRIDS FROM GRID RGA

MAGNITUDE OF TRUNCATION ERROR COVERED BY EACH LOCAL GRID VISCOUS - INVISCID INTERACTION - FLOW FIELD 1		
GRID	GRID SIZE	COVERS TRUNCATION ERROR LARGER THAN
RGA	257x65	10^{-4}
RGA55	257x55	10^{-3}
RGA49	253x49	10^{-2}
RGA43	245x43	10^{-1}
RGA37	243x37	10^0
RGA35	233x35	10^0
RGA33	233x33	10^{+1}
RGA31	233x31	10^{+1}
RGA21	209x21	10^{+1}

Calculations were made using each of the local grids and the results compared to the single grid result on the 257x65 grid. The lift, drag, and moment coefficients are shown in Table 4.4 along with the number of supersonic points and the residual achieved on the grid. The coarse grid solution is included for completeness. All the interaction runs executed the same number of iterations, 4000 for the fine grid and 5000 for the coarse grid, unless the residual for any part of a cycle dropped below some fixed tolerance. When the residual for the current grid calculation reached 10^{-14} , the calculation was ended and control was moved to the next grid. Appendix C shows a sample JCL file for case VS37.

TABLE 4.4 VISCOUS-INVISCID INTERACTION RUNS - GRID RGA
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 1

RESULTS - VARYING THE LOCAL GRID DOMAIN VISCOUS-INVISCID INTERACTION - FLOW FIELD 1						
CASE	RESID	C_L	C_{DP}	C_{DF}	C_M	% LIFT ERROR
Fine and Coarse Grid Solutions						
VRGAA	10^{-14}	.54648	.00376	.00646	-.08144	0.00
VSSTRT	10^{-09}	.53020	.00545	.00572	-.07967	2.98
Local Fine Grid Solutions						
VS33	10^{-10}	.52194	.00368	.00641	-.07667	4.49
VS35	10^{-14}	.54685	.00407	.00647	-.08180	0.07
VS37	10^{-11}	.54527	.00433	.00648	-.08166	0.22
VS43	10^{-14}	.54654	.00407	.00647	-.08173	0.01
Injected Coarse Grid Solutions						
VS33	10^{-11}	.52177	.00422	.00691	-.07688	4.52
VS35	10^{-11}	.54631	.00452	.00696	-.08208	0.03
VS37	10^{-12}	.54499	.00452	.00696	-.08187	0.27
VS43	10^{-13}	.54638	.00435	.00695	-.08192	0.02

As can be seen by comparing the coefficients in the table, the solution accuracy drops for local grids smaller than RGA35. In fact VS33 gave worse

results than using the coarse grid solution alone. Figure 4.6 shows the pressure contours and Mach number contours for VS33 as compared to the fine grid solution. Figure 4.7 shows the same comparison but for VS37. Both solutions are fairly close, but in VS33, the C_p contours oscillate in the wake. This is due to the boundary conditions (BC's) on the local grid and will be discussed later. Note that VS37 is much better in this regard. The line surrounding the body and extending off the plots to the right represents the edge of the injection region. One can see the big improvement in the solution achieved by adding just four lines to the fine grid. However, the difference between VS37 and VS43 is not nearly as large. Note also that the interacting solutions compare favorably with equivalent straight injection solutions shown in Table 4.2 although the accuracy does not improve monotonically here. The lack of monotonicity is due to the failure of the local grid in case VS37 to converge to as low a residual as in the neighboring cases. This was due to problems in the turbulence model which prevented convergence beyond some point, and is not a consequence of the current method.

Cases using grids smaller than RGA33 diverged. This result is in agreement with the straight injection runs in Section 4.1, which diverged with TE injection regions of similar size. As Figure 4.4 shows, the boundary layer in case VK21 was not adequately accounted for by the forcing function. In that run, the injected coarse grid solution was found by injecting the correct fine grid TE and then iterating to convergence. Figure 4.8 shows the boundary layer from the equivalent interaction run, VS21, just before it diverged. There are three parallel lines running the length of the plot. The bottom one is the airfoil, the middle one is the outer limit of the injection region, and the

outer one is the outer limit of the region in which the Navier-Stokes equations are solved. There is a six point buffer zone between the edge of the injection region and the end of the viscous grid. Nevertheless, the extent of the zone is not sufficient to provide an accurate representation of the viscous effects and, apparently was too restrictive on the solver. All of the runs on grids smaller than RGA33 diverged due to instabilities on the local grid, most likely due to not covering the full viscous region.

TABLE 4.5 VISCOUS-INVISCID INTERACTION RUNS - GRID RGA
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2

RESULTS - VARYING THE LOCAL GRID DOMAIN VISCOUS-INVISCID INTERACTION - FLOW FIELD 2						
CASE	S-S PTS	C_L	C_{DP}	C_{DF}	C_M	% LIFT ERROR
Fine and Coarse Grid Solutions						
VRGAB	91	.57580	.00420	.00638	-.08456	0.00
VTSTRT	2	.55673	.00608	.00572	-.08263	3.31
VRGHA	376	.58182	.00379	.00615	-.08516	1.05
VRGDA	235	.57618	.00404	.00640	-.08453	0.07
Local Fine Grid Solutions						
VT31	9	.51360	.00442	.00624	-.07146	10.80
VT33	28	.54069	.00431	.00632	-.07823	6.10
VT35	74	.57309	.00466	.00638	-.08449	0.47
VT37	76	.57369	.00489	.00639	-.08479	0.37
VT43	85	.57566	.00458	.00639	-.08493	0.02
Injected Coarse Grid Solutions						
VT31	2	.51529	.00472	.00674	-.07288	10.51
VT33	5	.54105	.00483	.00683	-.07844	6.04
VT35	19	.57253	.00510	.00689	-.08476	0.57
VT37	20	.57337	.00509	.00690	-.08502	0.42
VT43	21	.57541	.00489	.00688	-.08517	0.07

The runs mentioned so far were all performed using subcritical conditions. Similar runs were made using flow field 2 as the test case to ascertain the effects of a supersonic zone. This case has been run by Pulliam (1984)

and others, so there are solutions available for comparison. The truncation error for this case was found to have the same basic distribution as above, so the same grids can be used. Table 4.5 lists the results from these runs as compared to the 129x33 coarse grid solution, VTSTRT, and the 257x65 fine grid solution, VRGBA. A 385x97 case, VRGHA, and a 385x65 solution, VRGDA, are included as well. The accuracy of the cases using grids larger than RGA33 is apparent. However, the most notable point in these results is that the size of the supersonic zone decreased as the extent of the local grid decreased. The supersonic zone as found in the fine grid solutions is small enough to be covered by all of the local grids larger than RGA21. The injection region is also large enough to cover the supersonic zone for all cases shown. Although it is not known exactly why the size went down, the effect is very small. The supersonic bubble is mainly along the wall where the grid is highly clustered. A difference of 17 supersonic points is not physically significant.

TABLE 4.6 LOCAL GRIDS FROM GRID RGK

MAGNITUDE OF TRUNCATION ERROR COVERED BY EACH LOCAL GRID VISCOUS - INVISCID INTERACTION - FLOW FIELD 2		
CASE	GRID SIZE	COVERS TRUNCATION ERROR LARGER THAN
VTK65	257x65	10^{-4}
VTK49	257x49	10^{-2}
VTK37	245x37	10^{-1}
VTK31	221x31	10^0
VTK21	205x21	10^{+1}

The above supercritical flow situation was also run using grids based on another 257x65 fine grid, RGK. The only difference between RGA and

RGK is the slope of the wake gridline as it comes off the trailing edge of the airfoil. Figure 4.9 shows grids RGA and RGK for comparison. Since grid RGK better resolves the wake for this case, it should provide a better solution than did grid RGA. Table 4.6 shows the grid sizes that were indicated using TE_h^{2h} for VRGKA. Note that the grids required using RGK are smaller in the η direction for the larger magnitude TE than those required using the RGA grid as shown in Table 4.3. Table 4.7 shows the results from the runs made on these grids. Note the difference between the fine grid solution here, VRGKA, and the previous fine grid solution, VRGAB. Less circulation is set up around the airfoil using this grid resulting in a 1.4% difference in the coefficient of lift. The interaction runs provided similar accuracy as in the previous case. Solution VTK21 did not converge, but the results are shown here from just before it diverged. These results differ from the previous ones in that including the $O(1)$ TE did not drastically improve the solution, e.g., case VTK31. This is explained by noting that the boundary of grid RGK31 still does not cover the full viscous region on the aft part of the airfoil.

The solution VTAK37 interacts the 245x37 local grid RGK37 with the 129x33 global grid RGA2 as the global grid. One can see that the solution tends towards the solution favored by the local grid. Although the results are not as accurate as VTK37, they are still quite acceptable.

TABLE 4.7 VISCOUS-INVISCID INTERACTION RUNS - GRID RGK
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2

RESULTS - VARYING THE LOCAL GRID DOMAIN VISCOUS-INVISCID INTERACTION - FLOW FIELD 2						
CASE	S-S PTS	C_L	C_{DP}	C_{DF}	C_M	% LIFT ERROR
Fine and Coarse Grid Solutions						
VRGKA	80	.56796	.00415	.00638	-.08289	0.00
VRGK2A	0	.54379	.00594	.00572	-.07982	4.26
VRGOA	342	.57584	.00376	.00615	-.08389	1.39
Local Fine Grid Solutions						
VTK21	150	.54362	.04442	.00446	-.07750	4.29
VTK31	1	.50339	.00527	.00628	-.07241	11.37
VTK37	68	.56769	.00479	.00640	-.08332	0.05
VTAK37	56	.56341	.00492	.00639	-.08305	0.80
VTK49	80	.56818	.00422	.00638	-.08303	0.04
VTK65	80	.56827	.00413	.00638	-.08289	0.05
Injected Coarse Grid Solutions						
VTK21	16	.59420	.01046	.00423	-.09564	4.62
VTK31	0	.50560	.00478	.00677	-.07206	10.98
VTK37	18	.56773	.00515	.00691	-.08375	0.04
VTAK37	14	.56228	.00480	.00688	-.08269	1.00
VTK49	21	.56814	.00441	.00688	-.08311	0.03
VTK65	21	.56807	.00432	.00688	-.08292	0.02

4.3 Effects of Different Local Grid Boundary Conditions

The previous section studied the effect of the size of the local grid domain on the resulting interactive solution. In that section, the local and global grids used the same boundary conditions. Viscous boundary conditions were used at the wall, characteristic boundary conditions on the constant- η boundary and extrapolation boundary conditions along the constant- ξ boundaries at the rear of the domain. In those runs, the local and global grids were topologically similar. For both grids the outer constant- η boundary was outside the viscous boundary layer and the constant- ξ boundaries were far aft in the wake region. The other local grid topology studied here is a grid local

to the nose of the airfoil. The global grid BC's would not be as effective in this case and other types of boundary conditions must be studied.

In the interaction runs discussed so far, the global solution is being computed inviscidly. Therefore, one could possibly use inviscid boundary conditions for the global grid while retaining viscous BC's for the local grid. Obviously, inviscid BC's would not work for the local, viscous calculation. However, Dirichlet conditions would be a possibility for the local grid. They would use quite a bit less CPU time than the extrapolation and characteristic BC's and might provide faster convergence. Also, they would probably be more suitable for the airfoil nose local grid mentioned above than the global grid BC's are.

The case VTK37 from above was chosen to study the effects of boundary conditions on the final solution. Three cases were run using different combinations of boundary conditions for the local grid. The wall BC's were always the viscous wall conditions, even on the global grid. The global grid always used characteristic BC's on the constant- η far-field boundary and extrapolation BC's on the constant- ξ far-field boundaries. The local grid boundary conditions had to be suited to a viscous calculation. Therefore the constant- η outer boundary BC's were either Dirichlet conditions specified from the global grid or characteristic BC's. The constant- ξ boundary BC's were either Dirichlet conditions or extrapolation conditions. Table 4.8 shows the results from the three runs. The first column of the table indicates which type of boundary condition was used on that grid. For example, a listing of CE indicates that characteristic conditions were used on the constant- η outer boundary and extrapolation conditions were used on the constant- ξ boundaries. A listing of

DE indicates that Dirichlet conditions were used on the constant- η outer boundary and extrapolation conditions were used on the constant- ξ boundaries. A listing of DD indicates that Dirichlet conditions were used on both the constant- ξ and constant- η far field boundaries.

**TABLE 4.8 VISCOUS-INVISCID INTERACTION RUNS - GRID RGK
VARYING THE LOCAL GRID BOUNDARY CONDITIONS
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2**

RESULTS - VARYING THE LOCAL GRID BOUNDARY CONDITIONS VISCOUS-INVISCID INTERACTION - FLOW FIELD 2						
BC TYPE	RESID	S-S PTS	C_L	C_D	C_M	CPU TIME
Local Fine Grid Solutions						
CE	1.7×10^{-10}	68	.56769	.01119	-.08332	1074
DD	1.6×10^{-10}	68	.56796	.01118	-.08333	1030
DE	1.6×10^{-10}	68	.56796	.01118	-.08333	1037
Injected Coarse Grid Solutions						
CE	3.1×10^{-11}	18	.56773	.01206	-.08375	1133
DD	3.1×10^{-11}	18	.56800	.01207	-.08377	1087
DE	3.1×10^{-11}	18	.56800	.01207	-.08377	1095

As the results in Table 4.8 indicate, the choice of boundary conditions is flexible for the local grid. Figure 4.10 compares the global solutions for the first two cases. They give essentially the same global solution. However, comparing the local solutions in Figure 4.11 shows that the DD solution reduces the tendency for wiggles in the solution towards the aft end of the airfoil and in the wake region. Figure 4.12 compares the DD local grid solution with the fine grid solution. They agree quite well. Assuming that no instabilities show up in future runs, the Dirichlet conditions are recommended for the local grid. They produced results as accurate as the original boundary conditions, while saving some computation time and memory.

4.4 Interaction Using Dissimilar Grids

In the cases presented so far, the global coarse grid has been a subset of the same grid from which the local, fine grids were cut. The coarse grid has been found by eliminating every other point from the global fine grid. Therefore, the global grid has essentially been a coarse viscous grid in that the clustering of points near the wall has been very high. In the viscous-inviscid interaction method being presented here, the global grid must adequately resolve the inviscid part of the flow field. The grid must only be clustered enough at the wall to accept the truncation error injection from the fine grid. In this light, we form a new 257x65 grid, RGL, similar to RGA but with a minimum wall spacing of 1.0×10^{-4} instead of 2.0×10^{-5} as in RGA. A 129x33 coarse grid, RGL2, is formed by taking every other point from RGL. This grid and grid RGM2, corresponding to RGM and RGK, are used as the global grid in two more viscous-inviscid interaction runs.

First, the Euler equations were solved on each of the four grids and, using inviscid-inviscid interaction, on the two grid sets. Table 4.9 shows the results from these six runs. IRGLB and IRGMB are each fine grid runs and IRGL2B and IRGM2B are each coarse grid runs. IRGL2C and IRGM2C interact the 257x65 grids as local grids with their respective 129x33 grids as global grids. The coarse grid solutions approximate the fine grid lift coefficients to within 1.9% and 3.9% respectively. The size of the supersonic zones are only predicted to within about 40% for each grid. However, when the coarse and fine grids are interacted together, essentially the fine grid solution is returned.

**TABLE 4.9 INVISCID RUNS - GRIDS RGL AND RGM
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2**

RESULTS - VARYING THE LOCAL GRID DOMAIN INVISCID RUNS - FLOW FIELD 2					
CASE	S-S PTS	C_L	C_D	C_M	% LIFT ERROR
Fine and Coarse Grid Solutions					
IRGLB	596	.70121	.00238	-.10835	0.00
IRGL2B	103	.68787	.00438	-.10746	1.90
IRGMB	500	.67299	.00220	-.10238	0.00
IRGM2B	71	.64675	.00399	-.09864	3.90
Local Fine Grid Solutions					
IRGL2C	596	.70119	.00239	-.10837	0.00
IRGM2C	501	.67297	.00221	-.10240	0.00
Injected Coarse Grid Solutions					
IRGL2C	149	.70087	.00259	-.10844	0.05
IRGM2C	127	.67258	.00241	-.10241	0.06

Inviscid lift prediction to within 2% and 4% is considered adequate for this case, so the two coarse grids can be used as global grids in the viscous-inviscid interaction runs. The required size of the local fine grids was found using both methods mentioned in the beginning of the chapter for comparison purposes. The two approaches yielded slightly different results. Table 4.10 shows the grids indicated by using the fine grid truncation error, $T.E._h^{2h}$, and the coarse grid truncation error, $T.E._h^{4h}$. In the earlier runs, the coarse grid truncation error provided estimates that tended to be more conservative than the actual TE as measured on the fine grid, so the 245x49 grid was chosen to cover $O(10^{-2})$ TE and the 237x35 grid was chosen to cover the $O(10^{-1})$ TE for the RGM grid. For the RGL grid, the 245x47 grid was chosen to cover the $O(10^{-1})$ TE. Table 4.11 presents the results from the runs using these grids.

TABLE 4.10 LOCAL GRIDS FROM RGL AND RGM GRIDS

MAGNITUDE OF TRUNCATION ERROR COVERED BY EACH LOCAL GRID VISCOUS - INVISCID INTERACTION - FLOW FIELD 2			
TE USED	COVERS TRUNCATION ERROR LARGER THAN	GRID SIZE	
		RGL GRID	RGM GRID
$T.E._h^{2h}$	10^{-2}	253x51	245x49
	10^{-1}	245x47	229x37
	10^0	237x35	221x31
$T.E._{2h}^{4h}$	10^{-2}	245x45	249x43
	10^{-1}	233x41	237x35
	10^0	233x25	229x23

As can be seen by comparison to the results in Table 4.5, these runs produced predictions to the viscous fine grid solution which were as accurate as the previous ones. However, upon comparing the solutions graphically, one can see that these new solutions are better than the old. Figure 4.13 compares the global solutions for VTK37 and VTKM37 and Figure 4.14 compares the local solutions. These two cases used the same local grid RGK37, but different global grids. Note the wiggles in the pressure contours in the wake region for solution VTK37. They are not present for solution VTKM37, possibly due to the wider spacing of the global grid points in grid RGM2. Both solutions used characteristic and extrapolation boundary conditions (CE) on the local grid, so the wiggles can apparently be eliminated by using the inviscid grid with its wider mesh spacing or by using Dirichlet conditions on the local grid as shown in Section 4.3.

The difference between the A and B runs for VTAL43 and VTKM37 was the number of iterations performed. The A runs used 2000 iterations on the global grid and 1800 iterations on the local grid. The B series ran 5000 iterations on the global grid and 4000 iterations on the local grid. The

**TABLE 4.11 VISCOUS-INVISCID INTERACTION RUNS
CASES USING DISSIMILAR GRIDS
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 2**

RESULTS - VARYING THE LOCAL GRID SIZE AND SEQUENCING VISCOUS-INVISCID INTERACTION - FLOW FIELD 2						
CASE	S-S PTS	C_L	C_{DP}	C_{DF}	C_M	% LIFT ERROR
Fine Grid Solution - Grid RGA						
VRGAB	91	.57580	.00420	.00638	-.08456	0.00
Local Fine Grid Solutions - RGA Based						
VTAL43A	89	.57672	.00443	.00638	-.08486	0.16
VTAL43B	89	.57653	.00443	.00638	-.08483	0.13
VTAL47	91	.57627	.00429	.00638	-.08470	0.08
Injected Coarse Grid Solutions - Grid RGL2						
VTAL43A	19	.57677	.00475	.00336	-.08516	0.17
VTAL43B	19	.57656	.00475	.00336	-.08515	0.13
VTAL47	20	.57643	.00461	.00336	-.08496	0.11
Fine Grid Solution - Grid RGK						
VRGKA	80	.56796	.00415	.00638	-.08289	0.00
Local Fine Grid Solutions - Grid RGK Based						
VTKM35	75	.56928	.00442	.00639	-.08329	0.23
VTKM37A	75	.56898	.00453	.00639	-.08331	0.18
VTKM37B	75	.56926	.00450	.00639	-.08335	0.22
VTKM49	80	.56866	.00416	.00638	-.08297	0.12
Injected Coarse Grid Solutions - Grid RGM2						
VTKM35	18	.56992	.00492	.00336	-.08387	0.34
VTKM37A	17	.56875	.00493	.00336	-.08366	0.14
VTKM37B	16	.56903	.00495	.00336	-.08375	0.19
VTKM49	16	.56894	.00445	.00336	-.08321	0.17

difference between the two solutions in each case was minimal, indicating that the later iterations serve to reduce the residual, but do not change the solution very much. Case VTKM35 converged slowly using the A series sequencing scheme so it was run using 400 iterations per grid per cycle instead of 200 as in the A series. Overall, it ran 2800 iterations on the global grid and 2400 iterations on the local grid. Case VTKM49 was run with the same sequencing.

However, case VTAL47 used the A sequencing and converged well. It is possible that the convergence problems showed up only with VTKM35 since it was a marginal run anyway, in terms of covering the viscous region. Nevertheless, the sequencing used in that run is recommended for other runs as well.

4.5 Local Grid used for Resolution Enhancement

The next flow field studied was flow field 3. The flow field is largely subsonic, but with a small supersonic zone where the flow must accelerate to round the nose due to the high angle of attack. Figure 4.15 shows the Mach number contours as calculated on a 257x65 fine grid. Note the supersonic zone appearing just behind the nose in the second plot. The sonic line is the bolder line in the middle of the other contour lines. This supersonic zone should be terminated by a shock but there is not enough resolution to capture it. Figure 4.16 shows the same case as calculated on a 129x33 coarse grid consisting of every other point of the fine grid. The supersonic zone is even less developed here than in the fine grid case. In order to resolve the rapid expansion zone terminated by the shock, as well as the generation of most of the vorticity in the flow field, a local, fine grid is chosen which will cover the nose of the airfoil from behind the stagnation point on the underside around to after the shock region on the top side of the airfoil.

Unlike the previous runs, the local grid used here does not cover the entire near wall and wake region. Therefore, in order to account for the viscous effects everywhere, the Navier-Stokes equations must be solved on the global grid as well as on the local, fine grid. Also, in the previous runs, the local grid was chosen to cover some magnitude of the estimated TE.

Here, the local grid is chosen to provide added resolution in one particular region of the flow field in order to help understand the physics occurring in that region.

The calculation was run using a patch which covered the region of interest, yet had the exact grid spacing of the global grid. In other words, the patch was a subset of the global grid. This patch was interacted with the global grid to see if the interaction scheme itself would affect the overall solution. Ideally, the same solution would be found as in the single grid case. Several different size patches were used, all producing exactly the same solution as the single grid case. It can be concluded that the interaction scheme itself does not bias the solution to any noticeable degree.

Now, the interaction can proceed using a patch which has been refined. The patches chosen here are formed from the fine grid using a program that will refine a given grid in the computational plane using the interpolation routine described in Section 2.1. For example, in this case here, a 41x33 patch is removed from the fine grid. In the computational plane, this grid can be refined by a factor of two in each direction to form an 81x65 local grid with the same domain. Every other point in the new grid will lie on a node from the original grid. The remaining nodes will be distributed in accordance with the distribution function as used to generate the original grid. The process was repeated to form a 241x65 grid, with total refinement factor of six in the ξ -direction and two in the η -direction.

Several runs were made using each of the above three local grids with the 129x33 global grid and with the 257x65 global grid. Table 4.12 shows the results from these runs. Table 4.13 shows the global and patch grid sizes for

**TABLE 4.12 VISCOUS-VISCOUS INTERACTION RUNS
LOCAL PATCHED GRIDS
LIFT, DRAG, AND MOMENT RESULTS - FLOW FIELD 3**

RESULTS - VARYING THE LOCAL GRID DOMAIN VISCOUS-VISCOUS INTERACTION - FLOW FIELD 3						
CASE	S-S PTS	C_L	C_{DP}	C_{DF}	C_M	% LIFT ERROR
Fine and Coarse Grid Solutions						
VNGAA	135	1.42893	.02728	.00548	.02542	0.00
VNGA2A	15	1.32333	.04402	.00439	.02393	7.39
Local Fine Grid Solutions						
VNGA2F	122	.32218	-.18290	.00070	.06260	-
VNGA2H	127	.65270	-.14708	.00143	.11856	-
VNGA2G	121	.32206	-.18309	.00071	.06256	-
VNGA2I	528	.33180	-.19118	.00047	.06435	-
VNGA2J	1661	.33695	-.19463	.00030	.06531	-
VNGAF	563	.33623	-.19492	.00040	.06515	-
VNGAG	1685	.33818	-.19567	.00027	.06553	-
Injected Coarse Grid Solutions						
VNGA2F	30	1.41851	.03073	.00537	.02365	0.73
VNGA2H	30	1.42091	.02985	.00544	.02344	0.56
VNGA2G	29	1.41032	.03219	.00546	.02376	1.30
VNGA2I	32	1.45580	.02741	.00568	.02213	1.88
VNGA2J	35	1.45666	.02720	.00579	.02212	1.94
VNGAF	142	1.44621	.02515	.00563	.02485	1.21
VNGAG	144	1.45014	.02481	.00551	.02479	1.48

each case, the refinement ratio, λ , the total number of iterations, N , the total CPU time and the final RMS residual. Note how each of the first three cases in Table 4.12 using the 129x33 grid as the global grid gave lift coefficients within about 2% of the value as found on the fine grid alone. The local grids for each of these cases were taken directly from the fine grid. Figure 4.17 shows the Mach number contours for case VNGA2G. The solution compares quite favorably with the fine grid solution, underpredicting the maximum Mach number and minimum pressure by only 3-4% as opposed to 16-17% for the coarse grid solution.

Using the two finer local grids with the 129x33 global grid produced lift values higher than produced in the 257x65 single grid case. Figure 4.18 shows the Mach number contours for VNGA2I, using the 81x65 local grid, and Figure 4.19 shows the Mach number contours for VNGA2J, using the 241x65 local grid. Although case VNGA2G showed no evidence of a shock at all, case VNGA2I shows the beginnings of a shock forming. The resolution of the grid is still insufficient to capture the shock. As Figure 4.19 shows, case VNGA2J adequately resolves the supersonic zone and shock. The thickening of the boundary layer due to the shock impingement is also evident in the plot. The minimum pressure is about 5% lower for this solution than for VNGAA, further indication that the expansion has been more accurately resolved.

TABLE 4.13 VISCOUS-VISCOUS INTERACTION RUNS
LOCAL PATCHED GRIDS
GRID SIZES AND RUNTIME INFORMATION - FLOW FIELD 3

RESULTS - VARYING THE LOCAL GRID DOMAIN VISCOUS-VISCOUS INTERACTION - FLOW FIELD 3						
CASE	GRID SIZE		λ	N	CPU TIME	RESID
	GLOBAL	LOCAL				
Coarse and Fine Grid Solutions						
VNGA2A	129x33			5600	564	10^{-14}
VNGAA	257x65			5950	2067	10^{-9}
Interacting Solutions						
VNGA2F	129x33	41x45	2x2	3400	276	10^{-12}
VNGA2H	129x33	61x45	2x2	3400	302	10^{-12}
VNGA2G	129x33	41x33	2x2	3400	263	10^{-12}
VNGA2I	129x33	81x65	4x4	3400	365	10^{-9}
VNGA2J	129x33	241x65	12x4	8400	1640	10^{-13}
VNGAF	257x65	81x65	2x2	4800	1251	10^{-9}
VNGAG	257x65	241x65	6x2	8400	2772	10^{-10}

Switching to the 257x65 global grid produced results similar to the two previous cases. Figure 4.20 shows the Mach contours for case VNGAF, using the 81x65 local grid, and Figure 4.21 shows the Mach contours for case VNGAG, using the 241x65 local grid. Again, the increasing resolution of the shock can be seen as the density of the local grid is increased. Figure 4.21 shows the best resolution of the shock achieved here. Figure 4.22 compares the global solution for VNGAG to the 257x65 single grid solution. The solutions agree very closely on a global level. They differ only in the nose region where the local grid is located in case VNGAG, showing that the local grid is needed to help resolve the details of the flow field. However, it is conceivable that if the interaction between the shock and the boundary layer becomes stronger and the local grid is fine enough to capture the physics of separation, the global grid alone might give erroneous results due to lack of local resolution.

In Figure 4.21, note the delineation of the sonic zone by the bolder line in the second plot. There appears to be a jetlike behavior where the shock meets the boundary layer. Note the contour lines jutting out downstream. Figure 4.23 shows the vorticity magnitude contours for the same case. By comparing the two plots, one can see that the jetlike region is approximately at the edge of the boundary layer. Each contour in Figure 4.23 represents an order of magnitude of the vorticity. Therefore, the flow in the jet region is only slightly rotational. Figure 4.24 shows the 241x65 local grid NGAL5 in the same scale in which the fine grid results have been presented.

It should be mentioned that the Reynolds number at the foot of the shock is only about 3.91×10^4 . However, throughout the boundary layer the

same model of turbulence viscosity is used. The shock structure, the sonic zone, and the behavior of the boundary layer are highly interdependent. It is conceivable that a different turbulence model may yield a completely different local flow field. Such studies are beyond the scope of this work, which at this point only provides a better tool.

Figure 4.1 Pressure and Mach number contours for 257x65 fine grid case, VRGAA. Flow field 1.

DENSITY
VRGA2AF - FORCING FUNCTION

Figure 4.2a Negative logarithm of truncation error magnitude. Conservation of Mass equation. Case VRGA2A. Flow field 1.

X MOMENTUM
VRGA2AF - FORCING FUNCTION

Figure 4.2b Negative logarithm of truncation error magnitude. Conservation of X-Momentum equation. Case VRGA2A. Flow field 1.

Y MOMENTUM.
VRGA2AF - FORCING FUNCTION

Figure 4.2c Negative logarithm of truncation error magnitude. Conservation of Y-Momentum equation. Case VRGA2A. Flow field 1.

STAGNATION ENERGY PER UNIT VOLUME
VRGA2AF - FORCING FUNCTION

Figure 4.2d Negative logarithm of truncation error magnitude. Conservation of Energy equation. Case VRGA2A. Flow field 1.

DENSITY
VRGA4AF - FORCING FUNCTION

Figure 4.3a Negative logarithm of truncation error magnitude. Conservation of Mass equation. Case VRGA4A. Flow field 1.

X MOMENTUM
VRGA4AF - FORCING FUNCTION

Figure 4.3b Negative logarithm of truncation error magnitude. Conservation of X-Momentum equation. Case VRGA4A. Flow field 1.

Y MOMENTUM
VRGA4AF - FORCING FUNCTION

Figure 4.3c Negative logarithm of truncation error magnitude. Conservation of Y-Momentum equation. Case VRGA4A. Flow field 1.

STAGNATION ENERGY PER UNIT VOLUME
VRGA4AF - FORCING FUNCTION

Figure 4.3d Negative logarithm of truncation error magnitude. Conservation of Energy equation. Case VRGA4A. Flow field 1.

Figure 4.4 Boundary Layer profiles. Fine grid solution, VRGAA, and solution VK21. Straight Injection run. Flow field 1.

Figure 4.5 Grid RGA37 and grid RGA.

Figure 4.6 Pressure and Mach number contours for VS33 global solution and the 257x65 solution VRGAA. Flow field 1.

Figure 4.7 Pressure and Mach number contours for VS37 global solution and the 257x65 solution VRGAA. Flow field 1.

Figure 4.8 Boundary layer profiles. Fine grid solution, VRGAA, and solution VS21. Interaction run. Flow field 1.

Figure 4.9 Grids RGA and RGK.

Figure 4.10 Pressure and Mach number contours comparing global solutions for VTK37 with local boundary conditions CE and DD. Flow field 2.

Figure 4.11 Pressure and Mach number contours comparing local solutions for VTK37 with local boundary conditions CE and DD. Flow field 2.

Figure 4.12 Pressure and Mach number contours comparing global solution for VTK37 with local boundary conditions DD to VRGKA. Flow field 2.

Figure 4.13 Pressure and Mach number contours comparing global solutions for VTK37 and VTKM37. Flow field 2.

Figure 4.14 Pressure and Mach number contours comparing local solutions for VTK37 and VTKM37. Flow field 2.

Figure 4.15 Mach number contours. 257x65 single grid solution VNGAA. Flow field 3.

Figure 4.17 Mach number contours for case VNGA2G. 129x33 global grid interacting with 41x33 local grid. $\lambda = 2 \times 2$. Flow field 3.

Figure 4.20 Mach number contours for case VNGAF. 257x65 global grid interacting with 81x65 local grid. $\lambda = 2 \times 2$. Flow field 3.

Figure 4.21 Mach number contours for case VNGAG. 257x65 global grid interacting with 241x65 local grid. $\lambda = 6 \times 2$. Flow field 3.

Figure 4.22 Pressure and Mach number contours for VNGAA and fine grid interacting with 241x65 local grid, VNGAG. Flow field 3.

Figure 4.23 Vorticity Magnitude contours for case VNGAG local solution.
Flow field 3.

Figure 4.24 Local grid NGAL5 - 241x65.

CHAPTER 5

CONCLUSION

An effective method for splitting a problem with multiple length scales into multiple problems with single length scales has been presented in the context of viscous, transonic flow over an airfoil. It has been shown that the viscous effects in the flow field can be effectively decoupled from the inviscid solution and resolved by a fine grid local to the body and near wake region. This grid has been interacted with a coarser global grid and provides approximations to a global fine grid solution to within 1%. The effects of injecting different magnitudes of truncation error have been ascertained and it has been concluded that truncation error of order $O(1)$ must be included in order to produce a solution with lift coefficient accurate to within about 1%. However, there is the added requirement that the local grid fully enclose the viscous region towards the aft end of the airfoil and in the near wake region. The presence of a supersonic zone did not affect the utility of the method, as long as the zone is fully enclosed within the local grid and does not cross any grid boundaries.

Best results are achieved using Dirichlet conditions on the fine grid and a global grid clustered only enough to accept the injected truncation error. Otherwise, although the integrated coefficients may be accurate, the solution can have waviness in the pressure coefficient field.

The method has also been applied to a grid local to the nose region only. This case served to demonstrate the usefulness of the method in resolving very fine scales in one region without refining the entire grid. In

this case, the shock terminating the supersonic bubble at the nose of an airfoil at a high angle of attack but at low free stream Mach number was resolved. Details of the shock boundary layer interaction were captured and the existence of a jet emanating from the base of the shock has been indicated. The results were found assuming fully turbulent flow using an algebraic turbulence model.

An interpolation routine capable of interpolating between two generalized coordinate systems has also been presented. This routine has been vectorized and optimized such that its memory and CPU time requirements are almost negligible compared to the work of the solver. It is felt that in three dimensions, the requirements will be relatively less burdensome. Therefore, this routine could replace existing trilinear or tetrahedron based schemes in common use now, greatly increasing the accuracy of the interpolation process without greatly increasing computational work.

It is felt that future work should focus on several points. First, the wall boundary conditions for the global grid should be improved for viscous-inviscid interaction runs. Although the use of viscous wall conditions in solving the Euler equations did not appear to cause problems in this work, the possibility always exists and should be dealt with at some later time.

Second, it is recommended that a grid which expands laterally with downstream distance be used. The large size of the viscous zone at the aft end of the airfoil is the limiting factor in reducing the size of the local grid. An expanding grid could cover this viscous region without having to use more points than necessary. The boundary layer is thickening and the grid can thicken with it without losing resolution. The grids used in this work had

unnecessary points towards the nose of the airfoil. In several cases, a smaller local grid with boundary layer type growth in the streamwise direction could have been used and still covered all of the required injection region. A different grid generator or a modified version of the one used here should be employed in future work to generate more efficient grids.

APPENDIX A

SIMILARITY TRANSFORMATION MATRICES AND
THE TIME-LINEARIZED FLUX JACOBIAN MATRICES

In two dimensions the Jacobian matrices $\hat{A}$ or $\hat{B}$ =

$$\left[\begin{array}{cc} \kappa_t & \kappa_x \\ \kappa_x \phi^2 - u\theta & \kappa_t + \theta - \kappa_x(\gamma-2)u \\ \kappa_y \phi^2 - v\theta & \kappa_x v - \kappa_y(\gamma-1)u \\ -\theta(\gamma e/\rho - 2\phi^2) & \kappa_x(\gamma e/\rho - \phi^2) - (\gamma-1)u\theta \end{array} \right] \quad (A.1)$$

$$\left[\begin{array}{cc} \kappa_y & 0 \\ \kappa_y u - \kappa_x(\gamma-1)v & \kappa_x(\gamma-1) \\ \kappa_t + \theta - \kappa_y(\gamma-2)v & \kappa_y(\gamma-1) \\ \kappa_y(\gamma e/\rho - \phi^2) - (\gamma-1)v\theta & \kappa_t + \gamma\theta \end{array} \right]$$

where

$$\theta = \kappa_x u + \kappa_y v \quad \phi^2 = (\gamma-1) \left[\frac{u^2 + v^2}{2} \right] \quad (A.2)$$

with $\kappa = \xi$ or η for $\hat{A}$ or $\hat{B}$ respectively.

The viscous flux Jacobian is

$$\hat{M} = \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \\ m_{21} & \alpha_1 \partial_\eta(\rho^{-1}) & \alpha_2 \partial_\eta(\rho^{-1}) & 0 \\ m_{31} & \alpha_2 \partial_\eta(\rho^{-1}) & \alpha_3 \partial_\eta(\rho^{-1}) & 0 \\ m_{41} & m_{42} & m_{43} & m_{44} \end{array} \right] J \quad (A.3a)$$

where

$$\begin{aligned}
m_{21} &= -\alpha_1 \partial_\eta (u/\rho) - \alpha_2 \partial_\eta (v/\rho) \\
m_{31} &= -\alpha_2 \partial_\eta (u/\rho) - \alpha_3 \partial_\eta (v/\rho) \\
m_{41} &= \alpha_4 \partial_\eta [-(e/\rho^2) + (u^2 + v^2)/\rho] \\
&\quad - \alpha_1 \partial_\eta (u^2/\rho) - 2\alpha_2 \partial_\eta (uv/\rho) - \alpha_3 \partial_\eta (v^2/\rho) \\
m_{42} &= -\alpha_4 \partial_\eta (u/\rho) - m_{21} \\
m_{43} &= -\alpha_4 \partial_\eta (v/\rho) - m_{31} \\
m_{44} &= \alpha_4 \partial_\eta (\rho^{-1}) \\
\alpha_1 &= \mu [(4/3)\eta_x^2 - \eta_y^2] \\
\alpha_2 &= \mu (1/3)\eta_x \eta_y \\
\alpha_3 &= \mu [\eta_x^2 + (4/3)\eta_y^2] \\
\alpha_4 &= \mu \gamma Pr^{-1} (\eta_x^2 + \eta_y^2)
\end{aligned} \tag{A.3b}$$

The flux Jacobian matrices of Eqn. (3.16) have real eigenvalues and a complete set of eigenvectors. The similarity transforms are

$$\hat{A} = T_\xi \Lambda_\xi T_\xi^{-1} \quad \text{and} \quad \hat{B} = T_\eta \Lambda_\eta T_\eta^{-1} \tag{A.4}$$

where

$$\Lambda_\xi = \begin{bmatrix} U & 0 & 0 & 0 \\ 0 & U & 0 & 0 \\ 0 & 0 & U + a\sqrt{\xi_x^2 + \xi_y^2} & 0 \\ 0 & 0 & 0 & U - a\sqrt{\xi_x^2 + \xi_y^2} \end{bmatrix} \tag{A.5a}$$

$$\Lambda_\eta = \begin{bmatrix} V & 0 & 0 & 0 \\ 0 & V & 0 & 0 \\ 0 & 0 & V + a\sqrt{\eta_x^2 + \eta_y^2} & 0 \\ 0 & 0 & 0 & V - a\sqrt{\eta_x^2 + \eta_y^2} \end{bmatrix} \tag{A.5b}$$

with

$$T_{\kappa} = \begin{bmatrix} 1 & 0 \\ u & \kappa_y \rho \\ v & -\kappa_x \rho \\ \frac{\phi^2}{(\gamma-1)} & \rho(\kappa_y u - \kappa_x v) \end{bmatrix}$$

$$\begin{bmatrix} \alpha & \alpha \\ \alpha(u + \kappa_x a) & \alpha(u - \kappa_x a) \\ \alpha(v + \kappa_y a) & \alpha(v - \kappa_y a) \\ \alpha \left[\frac{\phi^2 + a^2}{(\gamma-1)} + a\bar{\theta} \right] & \alpha \left[\frac{\phi^2 + a^2}{(\gamma-1)} - a\bar{\theta} \right] \end{bmatrix} \quad (\text{A.6})$$

$$T_{\kappa}^{-1} = \begin{bmatrix} (1 - \phi^2/a^2) & (\gamma-1)u/a^2 \\ -(\kappa_y u - \kappa_x v)/\rho & \kappa_y/\rho \\ \beta(\phi^2 - a\bar{\theta}) & \beta[\kappa_x a - (\gamma-1)u] \\ \beta(\phi^2 + a\bar{\theta}) & -\beta[\kappa_x a + (\gamma-1)u] \end{bmatrix}$$

$$\begin{bmatrix} (\gamma-1)v/a^2 & -(\gamma-1)/a^2 \\ -\kappa_x/\rho & 0 \\ \beta[\kappa_y a - (\gamma-1)v] & \beta(\gamma-1) \\ -\beta[\kappa_y a + (\gamma-1)v] & \beta(\gamma-1) \end{bmatrix} \quad (\text{A.7})$$

and $\alpha = \rho/(\sqrt{2}a)$, $\beta = 1/(\sqrt{2}\rho a)$, $\bar{\theta} = \kappa_x u + \kappa_y v$, and, for example, $\kappa_x = \kappa_x/\sqrt{\kappa_x^2 + \kappa_y^2}$. κ is equal to either ξ or η as required. ϕ^2 is defined as before.

Relations exist between T_{ξ} and T_{η} of the form

$$\hat{N} = T_{\xi}^{-1} T_{\eta} \quad \text{and} \quad \hat{N}^{-1} = T_{\eta}^{-1} T_{\xi} \quad (\text{A.8})$$

where

$$\hat{N} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & m_1 & -\mu m_2 & \mu m_2 \\ 0 & \mu m_2 & \mu^2(1+m_1) & \mu^2(1-m_1) \\ 0 & -\mu m_2 & \mu^2(1-m_1) & \mu^2(1+m_1) \end{bmatrix} \quad (\text{A.9a})$$

$$\hat{N}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & m_1 & \mu m_2 & -\mu m_2 \\ 0 & -\mu m_2 & \mu^2(1+m_1) & \mu^2(1-m_1) \\ 0 & \mu m_2 & \mu^2(1-m_1) & \mu^2(1+m_1) \end{bmatrix} \quad (\text{A.9b})$$

with $m_1 = (\xi_x \eta_x + \xi_y \eta_y)$, $m_2 = (\xi_x \eta_y - \xi_y \eta_x)$ and $\mu = 1/\sqrt{2}$.

Note that the matrix $\hat{N}$ is a function of the metrics only and not of the flow variables.

APPENDIX B

TEINJECT CODE: DESCRIPTION OF VARIABLES AND I/O FILES

This material was taken in part from subroutine COMMENT in Tom Pulliam's ARC2D code (1984).

I/O Files

- TAPE3: Binary restart data, read in IOALL(4).
- TAPE4: Binary restart data, written in IOALL(10).
- TAPE5: Standard input, data at bottom of .jcl file.
- TAPE6: Standard output, data returned with .cpr file.
- TAPE7: Output of FINEITS, DELQ, DELQMX, RESID, RESIDMX, CL, CD, and CM for plotting using program RESID on the VAX. Written in IOALL(8).
- TAPE8: Binary or formatted grid input, accessed in IOALL(3).
- TAPE11: Grid output, unformatted from IOALL(16).
- TAPE13: Store a binary solution file for plotting on IRIS workstation using PLOT3D. Written in IOALL(13).
- TAPE20: Store the forcing function. This is the most recent forcing function calculated by the program. Written in IOALL(20).
- TAPE23: Output of CPBOD vs time in IOALL(12) for IRIS plotting.
- TAPE31: Read in the geometry for the first local grid. Read in IOALL(21).
- TAPE32: Read in the binary restart file for the first local grid. Read in IOALL(22).
- TAPE33: Read in the geometry for the second local grid. Read in IOALL(23). Not currently used.

TAPE34: Read in the binary restart file for the second local grid. Read in IOALL(24). Not currently used.

TAPE35: Store a binary restart file for the local grid. Written in IOALL(19).

TAPE36, TAPE37, TAPE38, TAPE39:

Same as TAPE13 but split into four files for ease of transfer back to UofA. Written in IOALL(13).

Sequencing Procedure

Time step sequencing and mesh sequencing is available. The last few lines of the input control the mesh sizes and the time steps used. The format is:

```

READ(*,*)      : A comment line
READ(*,*) ISEQUAL
READ(*,*)      : A comment line
DO II = 1, ISEQUAL
  READ(*,*) JMXI(II),KMXI(II),IENDS(II),DTISEQ(II),
>          LQS(II),LQTOS(II)
ENDDO

```

You can use this to get mesh sequencing from a coarse mesh to the target fine mesh. For example if:

JMAX = 193, KMAX = 33

then you can sequence to the fine mesh using

```

ISEQUAL
3
JMXI   KMXI   IENDS   DTISEQ   LQ   LQTO
  49     11     100     5.0     1     1
  97     18     100     5.0     1     1
 193     33     100     5.0     1     1

```

or if different time steps are required on the same grid you could use

```

ISEQUAL
5
JMXI   KMXI   IENDS   DTISEQ   LQ   LQTO

```

49	11	100	5.0	1	1
97	18	100	3.0	1	1
97	18	100	5.0	1	1
193	33	100	2.0	1	1
193	33	100	5.0	1	1

The variables LQ and LQTO are only used in an interaction run. A typical sequencing input for an interaction run follows. In this case, the above 193x33 grid is used as a global grid interacting with a 65x45 local grid. Note that LQ and LQTO are both 1 for the first step. If this were a restart run, then LQ would be set equal to the value of LQTO at the end of the previous run.

ISEQUAL

7

JMXI	KMXI	IENDS	DTISEQ	LQ	LQTO
193	33	400	1.0	1	1
65	45	400	1.0	1	2
193	33	400	3.0	2	1
65	45	400	3.0	1	2
193	33	400	5.0	2	1
65	45	400	5.0	1	2
193	33	400	5.0	2	1

In the case of a C mesh, JTAIL1 and JTAIL2 must be set such that all sub meshes have a point at the physical trailing edge of the airfoil. That is:

(JTAIL1-1) must be divisible by 2 to a whole integer to produce a sub mesh JTAIL1 at the trailing edge.

ISEQUAL:

The number of mesh sequences or time step sequences.

JMXI(I): The J dimension on sub mesh I. (May be JMAX). Must be JMAX or $(JMAX-1)/(JMXI(I)-1)$ must be an integer. JMXI(ISEQUAL) must be JMAX.

KMXI(I): The K dimension on sub mesh I. (May be KMAX). Must be $\leq$ KMAX. KMXI(ISEQUAL) must be KMAX.

DTISEQ(I):

The time step used on mesh I.

IENDS(I): The number of time steps used on mesh I.

LQ: Grid level previously used or calculated on. Used in MAIN for interactive runs only.

LQTO: Grid level about to be used. $LQ(I) = LQTO(I-1)$. Used in MAIN for interactive runs only. Also used in BCFOIL and BCLOCAL.

Dimension Parameters

MAXJ, MAXK, MAXJK:

Used as original dimensions of the permanent arrays. A grid must have a maximum J size less than or equal to MAXJ and a maximum K size less than or equal to MAXK. The maximum total size must be less than or equal to MAXJK. Typically $MAXJK = MAXJ \times MAXK$.

MAXM, MAXN, MAXMN:

Exactly analogous to MAXJ, MAXK, MAXJK but only used in arrays which are used for interaction runs. If a single grid run is being made, these can all be set to 1. If an interaction run is being made, they must be equal to their counterparts in the JK case.

MAXQ: Number of grids being used.

MAXLQ: Number of local grids being used. Equal to MAXQ-1.

MESDIM: Dimension for number of sequences which are allowed.

JJDIM, KKDIM:

Set equal to MAXJ and MAXK respectively. These dimensions are used in the arrays which save old values for the characteristic boundary conditions. The dimensions are specified this way because all the arrays are in common instead of being passed as arguments.

Variables

Q: Conservative variables.

Q(I,J,1)	Density	/ determinant of metric Jacobian
Q(I,J,2)	Density×U	/ determinant of metric Jacobian
Q(I,J,3)	Density×V	/ determinant of metric Jacobian
Q(I,J,4)	Energy	/ determinant of metric Jacobian

PRESS: Pressure = GAMI×(Q4 – 0.5×(Q2**2 + Q3**2)/Q1) NOTE: Scaled by the Jacobian.

SNDSP: A = SQRT(GAMMA×PRESS/Q1) NOTE: No Jacobian scaling.

S: Used for explicit right hand side in RHS, altered in VISRHS, smoothed in SMOOTH or FILTER, holds intermediate quantities in the AF scheme in STEP, etc., and ultimately holds the change in Q.

XY: Metric transformations.

XY(I,J,1)	$\partial\xi/\partial x$
XY(I,J,2)	$\partial\xi/\partial y$
XY(I,J,3)	$\partial\eta/\partial x$
XY(I,J,4)	$\partial\eta/\partial y$

XYJ: Jacobians of metric transformations.

$XYJ = (\partial\xi/\partial x \times \partial\eta/\partial y) - (\partial\xi/\partial y \times \partial\eta/\partial x)$

XIT: $\partial\xi/\partial t$

ETT: $\partial\eta/\partial t$

DS: Spatially variable time step, scaled on metric Jacobian, set in SETUP, used in STEP, STEPF2D, EIGEN, FLMATX2, FLMATY2.

X, Y: Cartesian X and Y coordinates of the grid. ξ, η are the coordinates in the computational plane.

FMU: Laminar molecular viscosity from Sutherland's Law.

TURMU: Turbulent Eddy viscosity.

- VORT:** Vorticity. Holds total vorticity. NOTE: The way it is computed, routine VORTDXI must be called first.
- A, B, C:** Lower diagonal, diagonal, and upper diagonal elements, respectively.
- D:** Jacobians from FILLMAT, intermediate results in BTRIX, BTRIY. See note in MAIN about the use of D for temporaries.
- E:** Temporary space for periodic block solver BTRIPX.

Storage Arrays

- QSAVE:** Stores the grid and solution fields in an interaction run. Used in MAIN.
- FORCE:** Array containing the residual after the RHS is calculated using the solution interpolated from the local grid. This is the forcing function. Found in FITOCO, used in INTEGRAT.

Interpolation Arrays

JPTS, KPTS:

Stores the J, K indices for the node in one grid which is closest to a particular node in the other grid. In XYDOWN, (JPTS(M,N), KPTS(M,N)) are the indices for the global grid point which is closest to the point (X(M,N), Y(M,N)) in the local grid. In XYUP, (JPTS(M,N), KPTS(M,N)) are the indices for the local grid point which is closest to the point (X(M,N), Y(M,N)) in the global grid. Used in QDOWN and QUP.

XNEWT, YNEWT:

(XNEWT(M,N), YNEWT(M,N)) stores the local coordinates for the point (M, N) in the bivariate patch with origin at (JPTS(M,N), KPTS(M,N)). These coordinates, and the interpolation they are used for, are in the computational plane. See interpolation notes. Found in XYDOWN and XYUP. Used in QDOWN and QUP.

- FFLAG:** Flag array set to zero for points in the global grid outside of the local grid domain. The value is set to 1.0E-9 if the point is to be

interpolated to but not included in the forcing function. This will happen for global grid points that are in the fringe of the local grid. The size of this fringe zone is determined by JSHIFTU and KSHIFTU. The value of FFLAG is set to 1.0 if the point is to be interpolated to and is to be included in the forcing function, i.e. for all global grid points which are in the interior of the local grid domain. Found in XYUP, used in QUP and FITOCO.

Characteristic Boundary Condition Storage Arrays

QNFOLDJ, QTFOLDJ, CSFOLDJ:

Arrays which hold the freestream values for normal velocity, tangential velocity, and sound speed along the $\eta = \text{constant}$ far-field boundary for the previous time step.

QNFOLDK, QTFOLDK, CSFOLDK:

Arrays which hold the freestream values for normal velocity, tangential velocity, and sound speed along the $\xi = \text{constant}$ far-field boundary for the previous time step.

QNXOLDJ, QTXOLDJ, CSXOLDJ, ENXOLDJ:

Arrays which hold the extrapolated values for normal velocity, tangential velocity, sound speed, and entropy along the $\eta = \text{constant}$ far-field boundary for the previous time step.

QNXOLDK, QTXOLDK, CSXOLDK, ENXOLDK:

Arrays which hold the extrapolated values for normal velocity, tangential velocity, sound speed, and entropy along the $\xi = \text{constant}$ far-field boundary for the previous time step.

Physical Parameters

These are the parameters that govern the physics of the flow field being simulated. Unless otherwise specified, they are read in or computed in INPUT.

ALPHA: Angle of attack of the airfoil. (DEFAULT=0.0)

FSMACH:

Free stream Mach number. (DEFAULT=0.397)

RE: Reynolds number based on free stream velocity. The code scales this to RE based on AINF in INITIA. Referenced in VISRHS, VISMAT, MUTUR. (DEFAULT=0.0)

WTRAT: Ratio of fixed wall temperature (Rankine) to TINF. Used in BCFOIL.

WTRAT = 0.0:

Adiabatic wall conditions. (DEFAULT)

WTRAT $\neq$ 0.0:

Isothermal wall conditions. Temperature at the wall is then $WTRAT \times TINF$.

GAMMA: Gas constant. (1.4)

GAMI: GAMMA - 1

PI: $4.0 \times ATAN(1.0)$

RHOINF, UINF, VIN, EINF, PINF:

Freestream values for RHO, U, V, E, AND PRESS.

TRANSLO, TRANSUP:

Transition point in terms of % chord for lower and upper surface respectively. Read in in LINPUT and GINPUT. Used in LINITIA and GINITIA. Note: TRANSLO = 0.0 implies transition at leading edge. These two variables are not grid dependent but the next two are. (DEFAULTS=0.0)

JTRANLO, JTRANUP:

J index for transition on lower and upper surface respectively. Found in LINITIA for the local grid and GINITIA for the global grid. Used in MUTUR. $TURMU(J,K) = 0.0$ for $J = JTRANLO, JTRANUP$.

Global Logical Variables

These logical variables are read in in INPUT and affect all grids which are used in the calculation.

STORE: Write restart data on TAPE4. Used in MAIN.

STORE = .FALSE.:
No write of data. (DEFAULT)

STORE = .TRUE.:
Write restart data.

PATCH: Logical variable determining whether Unver Kaynak's Mach number scaling is used in FILTERX and FILTERY. Used in FOF-MACH.

PATCH = .TRUE.:
Mach number scaling is used.

PATCH = .FALSE.:
Mach number scaling is not used. (DEFAULT)

FORCER: Logical variable determining whether the calculation is an interaction run or a single grid run. Used in MAIN and INTEGRAT.

FORCER = .TRUE.:
Interaction run. Two grids are assumed to be used. Interpolation logic is set up and used.

FORCER = .FALSE.:
Single grid calculation. One grid is assumed. No interpolation set up is performed. (DEFAULT)

VISXI, VISETA, VISCROSS:

Logicals to turn on various explicit viscous terms. VISETA = .TRUE., VISXI = VISCROSS = .FALSE. is the thin-layer approximation, which is the standard option. CAUTION: Implicit viscous terms are only programmed for the thin layer approximation. In general this is the case which should be used. These variables are used in VISRHS.

VISXI = .FALSE.:
No explicit viscous terms in ξ . (DEFAULT)

VISXI = .TRUE.:
Explicit viscous terms in ξ . Note!! Eddy viscosity turned off for VISXI = .TRUE.

VISETA = .FALSE.

No explicit viscous terms in η .

VISETA = .TRUE.

Explicit viscous terms in η . (DEFAULT)

VISCROSS = .FALSE.:

No explicit viscous cross terms. (DEFAULT)

VISCROSS = .TRUE.:

Explicit viscous cross terms.

CHARBC: Determines whether switch for characteristic far-field boundary conditions or extrapolation conditions depends on VISCBC or VISCOUS. Used in BCFOIL and BCLOCAL.

CHARBC = .TRUE.:

Switch depends on VISCOUS.

CHARBC = .FALSE.:

Switch depends on VISCBC. (DEFAULT)

DRCHLT1, DRCHLT2:

Determines whether Dirichlet conditions are used on constant-eta and constant-xi boundaries respectively for the local grid. Used in BCLOCAL.

DRCHLT1, DRCHLT2 = .TRUE.:

Dirichlet conditions are used on the boundaries.

DRCHLT1, DRCHLT2 = .FALSE.:

Characteristic or Extrapolation boundary conditions are used. (DEFAULT)

Local Logical Variables

These logical variables are read in for each grid, in GINPUT for the global grid and in LINPUT for the local grid.

RESTART:

Logical variable used to indicate a restart run. Expects data on TAPE3. Note that one solution can be a restart without the other

being a restart. Used in BCFOIL to determine if scaling occurs in startup. Used in LINITIA and GINITIA.

RESTART = .FALSE.:

No read of restart data. Scaling used in BCFOIL.
(DEFAULT)

RESTART = .TRUE.:

Read restart data. No scaling used in BCFOIL.

PERIODIC:

Switch for nonperiodic vs periodic coding.

PERIODIC = .FALSE.:

Non-periodic option. (DEFAULT)

PERIODIC = .TRUE.:

Periodic option.

BCAIRF: Option for different boundary conditions.

BCAIRF = .TRUE.:

BCFOIL called for airfoil conditions.

BCAIRF = .FALSE.:

BCLOCAL called for local grid conditions. (DEFAULT)

SHARP, CUSP:

Special logic is used at the trailing edge of an airfoil when it is sharp or cusped and an 'O' type mesh is being used. In that case sometimes the metrics and Jacobian are ill behaved and errors occur. This special logic changes the metrics and Jacobian (see XYMETTS) and modifies the boundary conditions (see BC). NOTE: These only have meaning for periodic meshes.

SHARP = .FALSE.:

Nothing special done. (DEFAULT)

SHARP = .TRUE.:

Special logic.

CUSP = .FALSE.:

Nothing special done. (DEFAULT)

CUSP = .TRUE.:
Special logic.

CIRCUL: Switch for circulation model at farfield boundary. Sets circulation at farfield boundary based on body value. Circulation computed from CL.

CIRCUL = .FALSE.:
For no model. (DEFAULT)

CIRCUL = .TRUE.:
For model.

VISCBC: Logical variable determining whether wall boundary conditions are for viscous flow or inviscid flow.

VISCBC = .TRUE.:
Viscous boundary conditions are used on the body.

VISCBC = .FALSE.:
Inviscid boundary conditions are used on the body.
(DEFAULT)

VISCOUS:
Determines whether flow is treated viscously or inviscidly. Read in in INPUT.

VISCOUS = .FALSE.:
Inviscid flow. (DEFAULT)

VISCOUS = .TRUE.:
Viscous flow.

TURBULNT:
Switch for turbulence model. Only used for **VISCOUS = .TRUE.**

TURBULNT = .FALSE.:
Laminar flow. (DEFAULT)

TURBULNT = .TRUE.:
Turbulent flow.

Algorithm Options

The following parameters set method options. Unless otherwise specified, they are all read in in INPUT.

METH: Determines which method is used.

METH = -1:

Standard block pentadiagonal implicit scheme with constant coefficient dissipation.

METH = 1:

Standard block tridiagonal implicit scheme with constant coefficient dissipation.

METH = 2:

Block pentadiagonal scheme with 2nd - 4th order nonlinear artificial dissipation.

METH = 3:

Pentadiagonal implicit scheme with 2nd - 4th order nonlinear implicit smoothing. (DEFAULT)

METH = 4:

Pentadiagonal implicit scheme with 4th - 4th order implicit/explicit constant dissipation.

Use block implicit scheme for time accurate computation. The present version uses constant coefficient artificial dissipation. For steady state computations use pentadiagonal scheme with nonlinear artificial dissipation.

IVIS: Determines implicit treatment of viscous terms. For METH = 3, 4.

IVIS = 1: Uses pentadiagonal for ξ and block for η with implicit viscous eigenvalues included.

IVIS = 2: Inviscid implicit operator in ξ and η viscous explicit operator. (DEFAULT)

JACDT: Switch to turn on variable time step. For unsteady problems, use a constant time step. For steady - state use a variable time step.

JACDT = 0:
Constant time step.

JACDT = 1:
Time step scaled by Jacobian. (DEFAULT)

JACDT = 2:
Time step based on minimum $1/\text{eigenvalues}$.

JACDT = 3:
Time step based on constant CFL at each point.

DTRATE: A scaling factor for the variable time step based on Jacobian. Allows for larger time steps when Jacobian gets large, for instance near the body. Only used for JACDT = 1. The formula for variable dt is shown below. (DEFAULT=1.0)

$$\text{dt(used)} = \frac{DT(1 + \sqrt{J})}{1 + \sqrt{J}/DTRATE}$$

A class of three step first or second order time accurate schemes can be employed. For steady state, Euler Implicit performs the best. For unsteady, 3pt Implicit is a good choice.

PHIDT, THETADT:
Time integration parameters.

PHIDT = 0	THETADT = 1	EULER IMPLICIT	(DEFAULT)
PHIDT = 0	THETADT = 1/2	TRAP. IMPLICIT	
PHIDT = 1/2	THETADT = 1	3PT IMPLICIT	

ORDERXY:
Order of operations for implicit integration.

ORDERXY = .TRUE.:
XI - ETA order. Works for all methods. (DEFAULT)

ORDERXY = .FALSE.:
ETA - XI order. See INPUT for exceptions.

Dissipation Parameters

These parameters are used to determine the character of the artificial viscosity being used in the scheme. Unless otherwise specified, they are read in in INPUT.

SMU: Smoothing parameter in explicit right hand side. Used for METH = 1,4,-1 in SMOOTH. Rescaled by DT. (DEFAULT=1.0)

SMUIM: Smoothing parameter in implicit left hand side. Used for METH = 1,4,-1 in FLMATX2, FLMATY2. Rescaled by DT. (DEFAULT=3.0)

DIS2X, DIS2Y:

Second order X and Y coefficients for FILTER. Used for METH = 2,3 in FILTERX, FILTERY, STEPF2DX, STEPF2DY. (DEFAULT=0.0)

DIS4X, DIS4Y:

Fourth order X and Y coefficient for FILTER. Used for METH = 2,3 in FILTERX, FILTERY, STEPF2DX, STEPF2DY. (DEFAULT=0.64)

ISPEC: Switch for spectral radius scaling in artificial dissipation.

ISPEC = 0 Form SPECTX and SPECTY the same.
For METH = 1, 4, -1.
SPECTX = SPECTY = 1./XYJ

ISPEC = 1 Form SPECTX and SPECTY independently.
For METH = 2, 3. (DEFAULT)
SPECTX = (ABS(UU) + CCX)/XYJ
SPECTY = (ABS(VV) + CCY)/XYJ

ISPEC = 4 Form SPECTX and SPECTY independently.
SPECTX = (ABS(ξ_x) + ABS(ξ_y))/XYJ
SPECTY = (ABS(η_x) + ABS(η_y))/XYJ

ISPEC = -1 Form SPECTX and SPECTY the same.
SPECTX = SPECTY = 64./XYJ

SPECTX, SPECTY:

The spectral radius scaling in the ξ and η directions, respectively. Formed in SPECT from INTEGRAT and FORCE. Used in

FILTERX and FILTERY.

UU, VV: The contravariant velocities in the ξ and η directions, respectively. Formed in EIGVAL from INTEGRAT and FORCE. Used to form SPECTX and SPECTY.

CCX, CCY:

The Eigenvalues in the ξ and η directions, respectively. Formed in EIGVAL from INTEGRAT and FORCE. Used to form SPECTX and SPECTY.

COEFX, COEFY:

The pressure coefficients in the ξ and η directions, respectively. Formed in GRADCOEF from INTEGRAT and FORCE. Used in FILTERX and FILTERY respectively.

FMACH: Function suggested by Unver Kaynak to decrease the artificial dissipation near the wall in viscous runs. Varys as FSMACH. Found in FOFMACH from INTEGRAT and FORCE. Used in FILTERX and FILTERY.

J, K Indices

The following variables are all grid dependent. They are read in in the appropriate input routine for the grid or calculated in SETUP for each grid. The local grid input routine is LINPUT and the global grid input routine is GINPUT.

JDIMEN, KDIMEN:

The size of the current grid in an interaction run or the final grid in a single grid run. Equal to JMAXOLD, KMAX in most cases. Sent as arguments to all the subroutines that bring in permanent arrays as arguments.

JMAX: Total number of points in XI direction. Equal to JMAXOLD-1 for a periodic grid with overlap. (DEFAULT=193)

KMAX: Total number of points in ETA direction. (DEFAULT=33)

JMAXOLD:

Inputted value of JMAX. If overlap, then set JMAXOLD = JMAX and JMAX = JMAX-1 in INITIA. For cases with overlap, the

output routines (especially for PLOT3D) use JMAXOLD so that the plots show a closed geometry and solution domain.

JM: JMAX - 1. Set in SETUP.

KM: KMAX - 1. Set in SETUP.

JBEGIN, JEND:

Absolute lower and upper limits on J, typically 1 and JMAX. Set in SETUP.

KBEGIN, KEND:

Absolute lower and upper limits on K, typically 1 and KMAX. Set in SETUP.

JLOW: Lower do loop limit in J direction. Found in SETUP.

JLOW = 2 for PERIODIC = .FALSE.

JLOW = 1 for PERIODIC = .TRUE.

JUP: Upper do loop limit in J direction. Found in SETUP.

JUP = JMAX - 1 for PERIODIC = .FALSE.

JUP = JMAX for PERIODIC = .TRUE.

KLOW: Lower do loop limit in K direction, typically KLOW = 2. Found in SETUP.

KUP: Upper do loop limit in K direction, typically KUP = KMAX - 1. Found in SETUP.

JTAIL1: First solid body point. Must be ≥ 2 for nonperiodic grid. Set = 1 in LINITIA or GINITIA if PERIODIC (O-grid).

JTAIL2: Last solid body point. Must be $\leq$ JMAX - 1 for nonperiodic grid. Set = JMAX in LINITIA or GINITIA if PERIODIC (O-grid).

JPLUS: Index array for J differencing. Found in SETUP.

JPLUS(J) = J + 1

JPLUS(JMAX) = JMAX for PERIODIC = .FALSE.

JPLUS(JMAX) = 1 for PERIODIC = .TRUE.

JMINUS: Index array for J differencing. Found in SETUP.

```
JMINUS(J) = J - 1
JMINUS(1) = 1      for PERIODIC = .FALSE.
JMINUS(1) = JMAX  for PERIODIC = .TRUE.
```

KPLUS: Index array for K differencing. Found in SETUP.

```
KPLUS(K)      = K + 1
KPLUS(KMAX) = KMAX
```

KMINUS: Index array for K differencing. Found in SETUP.

```
KMINUS(J) = K - 1
KMINUS(1) = 1
```

NOTE: Code has the option to use an overlapping or nonoverlapping periodic grid. In the overlap case the 1 and JMAX point are overlapped in the grid and special logic in the above parameters is used to account for this. See SETUP.

Other Global Parameters

These are all read in in INPUT.

STRITIT: The number of iterations over which the slow (rather than impulsive) start is spread. Used in BCFOIL. (DEFAULT=12)

IPRINT: Output switch for frequency of residual history output on TAPE6. Used in MAIN. (DEFAULT=1)

NP: Frequency of calls to IOALL(15) to show RHO, U, V, E, P, Cp, MACH, X, Y on TAPE6. (DEFAULT=10000)

NPCP: Frequency of calls to IOALL(12) to output CP vs time for plotting on IRIS or other devices. Writes to TAPE23. (DEFAULT=10000)

Some Parameters Used in MAIN

ISTART: The starting iteration, read in from the restart data file.

NUMITER:

Iteration counter for main steps.

DT: Time step. Either a true dt for time accurate computation or a scaling factor for the spatially variable time step. In a steady state calculation DT is order(1.0). (DEFAULT=5.0)

TOTIME: $DT \times NUMITER \times FSMACH$

CPUTIME:

CPU time.

ISTEPS: Actual number of iterations executed by the solver.

IEND: Ending value for NUMITER in the iteration do loop.

SCALIT: Determines ratio of coarse grid size to fine grid size.

FINEITS: $SCALIT \times NUMITER$

IBAIL: Flag in MAIN used to show that a set of iterations has converged.

IRATE: Parameter in MAIN which determines the number of iterations over which the convergence rate is calculated.

IP: Used to determine which function in IOALL is executed when IOALL is called.

IIG: Sequence level in MAIN.

NSUPER: Number of supersonic points. Computed in IOALL(5).

RESID: L2-Norm of residual. Computed in RESIDL2. Output in IOALL(8) onto TAPE7. Output in IOALL(5) to TAPE6. Zeroed in INITIA.

RESIDMX:

Max-Norm of residual. Computed in RESIDL2. Output in IOALL(8) onto TAPE7. Zeroed in INITIA.

MAXRES(2), MAXDQ(2):

The J and K locations of the maximum residuals and the change in Q, over the 4 conservative variables.

CLFINE: The lift on the airfoil as computed on the fine grid in an interaction run. Found in IOALL(5) and used in BCFOIL.

IIGRID: Sequence level in MAIN. Used in BCFOIL.

Grid Parameters

These are all read in in INPUT.

IREAD: Determines how the grid is brought into the program. Used in IOALL.

IREAD = 0:

Create grid for cylinder. (DEFAULT)

IREAD = 1:

Formatted read of grid from TAPE8 in IOALL(3).
JMAX, KMAX (2I5) X, Y (5E14.7).

IREAD = 2:

Unformatted read of grid from TAPE8 in IOALL(3).

DSWALL:

Minimum spacing at body for cylinder grid. Passed as parameter to CYLGRID from INITIA. Referenced only in CYLGRID. (DEFAULT=-0.005)

DSWALL = 0.0:

Produces uniform circle grid.

DSWALL > 0.0:

Produces stretched grid with DSWALL minimum spacing in η and cosine clustering to front and back.

DSWALL < 0.0:

Produces uniform grid in ξ and stretched in η .

SOBMAX:

Upper dimension in diameters ($D=1$) for cylinder grid. Passed as parameter to CYLGRID from INITIA. (DEFAULT=12.0)

CL vs ALPHA Logic**CLALPHA:**

Option to compute ALPHA for a given CL.

CLALPHA = .FALSE.

Logic not used. (DEFAULT)

CLALPHA = .TRUE.

Logic turned on.

CLINPUT:

Target coefficient of lift.

ICLSTRT: Start ALPHA modification after ICLSTRT iterations.

ICLFREQ:

Update ALPHA every ICLFREQ steps using the formula:

DELTA ALPHA = -RELAXCL × DELTA CL

DELTA CL = CLINPUT - CL(CALCULATED)

RELAXCL:

Relaxation parameter for DELTA ALPHA. Should use values of order 1.0.

CLTOL: Update of ALPHA stopped when ABS(DELTA CL) reaches tolerance, CLTOL. Order (1.0E-4).

ISW: Dummy switch for print out.

Forcing Parameters

LQMAX: Number of grids to be used in the calculation. Read in in INPUT Used in MAIN.

LQPAR: Grid which is the parent of the current local grid. Used in MAIN.

INTDOWN:

Switch used to determine whether down interpolation actually takes place. Down interpolation is not desirable if a coarser grid is only being used to estimate the size of the forcing function.

ICOUNT, IPOINT, IND:

Quantities used to calculate, store and use pointers needed in MAIN for the indirect indexing of JPTS, KPTS, XNEWT, and YNEWT.

JPLOW, JPUP, KPLOW, KPUP:

Approximate boundaries of the local grid in the global grid. Initial values read in in LINPUT, recalculated in XYDOWN.

JSHIFTD, KSHIFTD:

Determines integer distance which bicubic patch centers are displaced away from boundaries for the down interpolation. Read in in INPUT. Used in XYDOWN. (DEFAULTS=2)

JSHIFTU, KSHIFTU:

Determines integer distance which bicubic patch centers are displaced away from boundaries for the up interpolation. Read in in INPUT. Used in XYUP. (DEFAULTS=2)

NMIN, NMAX:

The smallest and largest values of N in the global grid for which up interpolation takes place. Used in XYUP and QUP.

MMIN, MMAX:

The smallest and largest values of M in the global grid for each N for which up interpolation takes place. Used in XYUP and QUP.

M, N: The indices for the points in the grid being interpolated TO in XYDOWN and XYUP. (J, K) are the indices for the points in the grid being interpolated FROM.

SUMX, SUMY, SUM1, SUM2, SUM3, SUM4:

Scratch arrays used in finding the coefficients for the bivariate patches. Used in XYDOWN, XYUP, QDOWN, and QUP.

RX, RY, RQ1, RQ2, RQ3, RQ4:

Initially hold the RHS's in the matrix equation for the coefficients of the bivariate patches, then holds the coefficients themselves. Used in XYDOWN, XYUP, QDOWN, and QUP.

NUMX, NUMY:

Temporaries holding the numerators in the Newton zero-finding algorithm. Used in XYDOWN and XYUP.

FX, FXX, FXY:

The function value, x-derivative, and y-derivative for the X(M,N) bivariate patch, evaluated at (XNEWT, YNEWT). Used in XYDOWN and XYUP.

FY, FYX, FYY:

The function value, x-derivative, and y-derivative for the Y(M,N) bivariate patch, evaluated at (XNEWT, YNEWT). Used in XYDOWN and XYUP.

IFIRST: Switch set in MAIN. Determines whether down interpolation is performed for the entire local grid or just for the boundary points. Interpolation is performed for the entire grid when IFIRST > 0, implying that the local grid is being initialized by the global grid. Otherwise, only the boundary points are interpolated to, so that the local grid solution does not have to redevelop after every down interpolation.

QTEMP: Temporary array containing the interpolated values in XYUP and QUP. Needed in the case of the local grid crossing the cut at the trailing edge of the airfoil.

MCUT: Value of M at the cut of the airfoil in XYUP and QUP.

APPENDIX C

SAMPLE .JCL FILE FOR CODE TEINJECT

CASE VS37

```

JOB,JN= VS37,T= 1800,MFL= 1900000.
ACCOUNT,AC= ,US= ,UPW= .
*.
*. LINK IN THE LIBRARIES.
AMESLIB.
IMSLIB.
*.
*. ACCESS THE RESTART FILES.
ACCESS,DN= REST1,PDN= VS37Q1 ,ID= STTBDG,NA.
ACCESS,DN= REST2,PDN= VS37Q2 ,ID= STTBDG,NA.
ASSIGN,DN= REST1,A= FT03.
ASSIGN,DN= REST2,A= FT32.
*.
*. ACCESS THE INPUT GRIDS.
ACCESS,DN= GRID1,PDN= RGA2 ,ID= STTBDG.
ACCESS,DN= GRID2,PDN= RGA37 ,ID= STTBDG,NA.
ASSIGN,DN= GRID1,A= FT08.
ASSIGN,DN= GRID2,A= FT31.
*.
ASSIGN,DN= SOLN1,A= FT04.
ASSIGN,DN= SOLN2,A= FT35.
ASSIGN,DN= FORC,A= FT20.
ASSIGN,DN= SOLNA,A= FT36.
ASSIGN,DN= SOLNB,A= FT37.
ASSIGN,DN= SOLNC,A= FT38.
ASSIGN,DN= SOLND,A= FT39.
ASSIGN,DN= CLCD,A= FT07.
*.
*. HELPING, INTEGRATOR, INPUT AND BC ROUTINES.
FETCH,DN= CM,MF= EA,TEXT= 'STT1:< GOBLE.WORK.TCOD> COMDECK.F;3'.
*.FETCH,DN= HP,MF= EA,TEXT= 'STT1:< GOBLE.WORK.TCOD> HLPDECK.F;3'.
*.FETCH,DN= IG,MF= EA,TEXT= 'STT1:< GOBLE.WORK.TCOD> IGRDECK.F13'.
*.FETCH,DN= BC,MF= EA,TEXT= 'STT1:< GOBLE.WORK.TCOD> BCNDECK.F28'.
*.
*. MAIN PROGRAM AND LIBRARY ROUTINES.
FETCH,DN= MP,MF= EA,TEXT= 'STT1:< GOBLE.WORK.TCOD> MAINA.FOR'.
*.FETCH,DN= LB,MF= EA,TEXT= 'STT1:< GOBLE.WORK.TCOD> ML150.FOR'.
ACCESS,DN= ARCLIB,PDN= ML150FOR6,ID= STTBDG,ED= 3.
*.
*. UPDATE, CFT AND RUN THE CODE.
UPDATE,P= 0,I= MP:CM,C= COMP,N= 0,L,E,S= 0,DW,DC= OFF,F.

```

```

CFT,I= COMP,L= 0,OPT= FULLIFCON,MAXBLOCK= 4000,ON= Z.
LDR,MAP= OFF,LIB= IMSL:ARCLIB.
*.
*. SAVE THE RESTARTS AND FORCING FUNCTION FOR LATER USE.
*.SAVE,DN= SOLN1,PDN= VS37Q1 ,ID= STTBDG,NA.
*.SAVE,DN= SOLN2,PDN= VS37Q2 ,ID= STTBDG,NA.
*.SAVE,DN= FORC,PDN= VS37F ,ID= STTBDG,NA.
*.SAVE,DN= CLCD,PDN= VS37C ,ID= STTBDG,NA.
*.
*. USE SENDWKS.
*.
*.ACCESS,DN= SENDWKS,PDN= SENDWKS,ID= STTRDM,OWN= RFTRDM.
*.SENDWKS,DN= FORC ,VDN= '[GOBLE.WORK.SCRR]VS37F.BIN',NOREC.
*.
EXIT.
DUMPJOB.
DEBUG,NOTSYMS= Q:XY:XIT:ETT:PRESS:XYJ:X:Y:QSAVE:FORCE:TURMU:VORT:
SNDSP:DS:S:SPECTX:SPECTY:XNEWT:YNEWT:FFLAG,MAXDIM= 1:1:1,BLOCKS= BASE.
*.
/EOF
$INPUTS FSMACH= 0.64, ALPHA= 1.93, RE= 5.7E6,
DIS4X= 0.64, DIS4Y= 0.64, DIS2X= 1.0, DIS2Y= 1.0,
IPRINT= 50, IREAD= 2,
FORCER= TRUE, PATCH= FALSE, STORE= TRUE,
CHARBC= FALSE, DRCHLT1= FALSE, DRCHLT2= FALSE,
LQMAX= 2 $

$LINPUTS JMAX= 243, KMAX= 37, JTAIL1= 26, JTAIL2= 218,
JPLOW= 2, JPUP= 128, KPLOW= 1, KPUP= 19,
TRANSLO= 0.11, TRANSUP= 0.11, LQPARIN= 1,
RESTART= FALSE,
BCAIRF= FALSE, CIRCUL= FALSE, PERIODIC= FALSE,
VISCOUS= TRUE, VISCBC= TRUE, TURBULNT= TRUE $

$GINPUTS JMAX= 129, KMAX= 33, JTAIL1= 17, JTAIL2= 113,
TRANSLO= 0.0, TRANSUP= 0.0,
RESTART= FALSE,
BCAIRF= TRUE, CIRCUL= FALSE, PERIODIC= FALSE,
VISCOUS= FALSE, VISCBC= TRUE, TURBULNT= FALSE $

ISEQUAL IF ISEQUAL GT 1 THEN JMXI,KMXI,IENDS,DTISEQ : II= 1,ISEQUAL
13
JMXI, KMXI, IENDS, DTISEQ, LQ, LQTO FOR II= 1,ISEQUAL
129, 33, 400, 1.0, 1, 1
239, 37, 400, 1.0, 1, 2
129, 33, 400, 3.0, 2, 1
239, 37, 400, 3.0, 1, 2
129, 33, 400, 5.0, 2, 1
239, 37, 400, 5.0, 1, 2
129, 33, 400, 5.0, 2, 1

```


239,	37,	400,	5.0,	1,	2
129,	33,	600,	5.0,	2,	1
239,	37,	600,	5.0,	1,	2
129,	33,	800,	5.0,	2,	1
239,	37,	800,	5.0,	1,	2
129,	33,	1000,	5.0,	2,	1
239,	37,	1000,	5.0,	1,	2
129,	33,	1000,	5.0,	2,	1

/EOF

REFERENCES

- Atta, Essam H. (1981) "Component-Adaptive Grid Interfacing," AIAA Paper 81-0382.
- Atta, Essam H. and Vadyak, J. (1982) "A Grid Interfacing Zonal Algorithm for Three Dimensional Transonic Flows About Aircraft Configurations," AIAA Paper 82-1017.
- Baldwin, B.S. and Lomax, H. (1978) "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-0257.
- Barton, John T. and Pulliam, Thomas H. (1986) "Airfoil Computation at High Angles of Attack, Inviscid and Viscous Phenomena," *AIAA Journal*, Vol. 24, No. 5, pp. 705-712.
- Beam, Richard and Warming, Robert F. (1976) "An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation Law form," *Journal of Computational Physics*, Vol. 22, pp. 87-110.
- Benek, John A., Steger, Joseph L., and Dougherty, F. Carroll (1983) "A Flexible Grid Embedding Technique with Applications to the Euler Equations," AIAA Paper 83-1944.
- Berger, Marsha (1982) "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," Ph.D. Dissertation, Stanford University.
- Brandt, Achi (1980) "Multi-Level Adaptive Techniques (MLAT) for Singular-Perturbation Problems," NASA-TM-80966, N80-22016.
- Brandt, Achi (1980) "Multilevel Adaptive Computations in Fluid Dynamics," *AIAA Journal*, Vol. 18, No. 10, pp. 1165-1172.
- Briley, R.W. (1971) "A Numerical Study of Laminar Separation Bubbles using the Navier-Stokes Equations," *Journal of Fluid Mechanics*, Vol. 47, Part 4, pp. 713-736.
- Carter, James E. (1974) "Solutions for Laminar Boundary Layers with Separation and Reattachment," AIAA Paper 74-583.
- Carter, James E. (1978) "Inverse Boundary-Layer Theory and Comparison with Experiment," NASA TP 1208.

- Carter, James E. (1979) "A New Boundary-Layer Inviscid Iteration Technique for Separated Flow," AIAA Paper 79-1450.
- Carter, James E. (1981) "Viscous-Inviscid Interaction Analysis of Transonic Turbulent Separated Flow," AIAA Paper 81-1241.
- Catherall, D. and Mangler, K.W. (1966) "The Integration of the Two-Dimensional Laminar Boundary-Layer Equations Past the Point of Vanishing Skin Friction," *Journal of Fluid Mechanics*, Vol. 26, Part 1, pp. 163-182.
- Dougherty, F. Carroll, Benek, John A., and Steger, Joseph L. (1985) "On Applications of Chimera Grid Schemes To Store Separation," NASA TM 88193.
- Fung, K.-Y. and Fu, J.-K. (1987) "Computation of Unsteady Transonic Aerodynamics with Truncation Error Injection," *AIAA Journal*, Vol. 25, No. 7, pp. 1018-1020.
- Fung, K.-Y., Tripp, J. and Goble B. (1988) "Adaptive Refinement with Truncation Error Injection," *Computer Methods in Applied Mechanics and Engineering*, Vol. 66, No. 1, pp. 1-16.
- Holst, T.L. (1979) "Implicit Algorithm for the Conservative Transonic Full Potential Equation Using an Arbitrary Mesh," *AIAA Journal*, Vol. 17, No. 10, pp. 1038-1045.
- Holst, T.L., Gundy, K.L., Flores, J., Chaderjian, N.M., Kaynak, U., and Thomas, S.D. (1985) "Numerical Solution of Transonic Wing Flows Using an Euler/Navier-Stokes Zonal Approach," AIAA Paper 85-1640.
- Jameson, Antony, Schmidt, W. and Turkel, Eli (1981) "Numerical Solutions of the Euler Equations by Finite Volume Methods using Runge-Kutta Time-Stepping Schemes," AIAA Paper 81-1259.
- Kaynak, Unver, Holst, Terry L., and Cantwell, Brian J. (1985) "Computation of Transonic Separated Wing Flows Using an Euler/Navier-Stokes Zonal Approach," NASA TM 88311.
- Klineberg, J.M. and Steger, Joseph L. (1974) "The Numerical Calculation of Laminar Boundary-Layer Separation," NASA TN D-7732.
- Liang, Shen-Min and Fung, K.-Y. (1987) "Refined Numerical Solution of the Transonic Flow Past a Wedge," *AIAA Journal*, Vol. 25, No. 9, pp. 1171-1175.

- Mastin, C. Wayne and McConnaughey, H.V. (1984) "Computational Problems on Composite Grids," AIAA Paper 84-1611.
- Prandtl, Ludwig (1904) "Über Flüssigkeitsbewegung bei sehr kleiner Reibung," *Proc. 3rd Int. Math. Congr.*, Heidelberg.
- Pulliam, Thomas H. and Chaussee, D.S. (1981) "A Diagonal Form of an Implicit Approximate-Factorization Algorithm," *Journal of Computational Physics*, Vol. 39, No. 2, P. 347.
- Pulliam, Thomas H. (1984) "Euler and Thin Layer Navier-Stokes Codes: ARC2D, ARC3D, Notes for Computational Fluid Dynamics User's Workshop," The University of Tennessee Space Institute, Tullahoma, Tenn., UTSI E02-4005-023-84.
- Pulliam, Thomas H., Jespersen, Dennis C. and Barth, Timothy J. (1985) "Navier-Stokes Computations for Circulation Controlled Airfoils," AIAA Paper 85-1587.
- Pulliam, Thomas H. and Steger, Joseph L. (1985) "Recent Improvements in Efficiency, Accuracy, and Convergence of Implicit Approximate Factorization Algorithms," AIAA Paper 85-0360.
- Rai, Man Mohan (1984) "A Conservative treatment of Zonal Boundaries for Euler Equation Calculations," AIAA Paper 84-0164.
- Rai, Man Mohan (1985) "An Implicit, Conservative, Zonal-Boundary Scheme for Euler Equation Calculations," AIAA Paper 85-0488.
- Rubbert, P.E. and Lee, K.D. (1982) "Patched Coordinate Systems," *Numerical Grid Generation*, (J.F. Thompson, Ed.), Elsevier/North-Holland, New York.
- Steger, Joseph L. (1978) "Implicit Finite Difference Simulation of Flow about Arbitrary Two Dimensional Geometries," *AIAA Journal*, Vol. 16, No. 7, pp. 679-686.
- Steger, Joseph L., Dougherty, F. Carroll, and Benek, John A. (1983) "A Chimera Grid Scheme," ASME Mini-Symposium on Advances in Grid Generation, Houston, TX, June. "

- Steger, Joseph L. and Van Dalsem, William R. (1985) "Developments in the Simulation of Separated Flows Using Finite Difference Methods," *Proceedings of the Third Symposium on Numerical and Physical Aspects of Aerodynamic Flows*, California State University, Long Beach, CA, Jan. 21-24.
- Van Dalsem, William R. (1984) "Simulation of Separated Transonic Airfoil Flow by Finite-Difference Viscous-Inviscid Interaction," Ph.D. Dissertation, Stanford University.
- Van Dalsem, William R. and Steger, Joseph L. (1983) "Finite-Difference Simulation of Transonic Separated Flow Using a Full Potential-Boundary Layer Interaction Approach," AIAA Paper 83-1689.