DEPARTMENT OF MECHANICAL ENGINEERING
COLLEGE OF ENGINEERING & TECHNOLOGY
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

CONVERGENCE ACCELERATION OF THE PROTEUS COMPUTER CODE WITH MULTIGRID METHODS

By

A. O. Demuren, Principal Investigator

and

S. O. Ibraheem, Research Associate

Final Report
For the period ended January 15, 1995

Prepared for
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, OH 44135

Under
Research Grant NAG-3-1329
J. R. Scott, Technical Monitor
Inlet, Duct, Nozzle Flow Physics Branch

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, VA 23518

January 1995

SUMMARY

This report presents the results of a study to implement convergence acceleration techniques based on the multigrid concept in the two–dimensional and three–dimensional versions of the Proteus computer code. The first section presents a review of the relevant literature on the implementation of the multigrid methods in computer codes fro compressible flow analysis. The next two sections present detailed stability analysis of numerical schemes for solving the Euler and Navier–Stokes equations, based on conventional von Neumann analysis and the bi–grid analysis, respectively. The next section presents details of the computational method used in the Proteus computer code. Finally, the multigrid implementation and applications to several two–dimensional and three–dimensional test problems are presented.

The results of the present study show that the multigrid method always leads to a reduction in the number of iterations (or time steps) required for convergence. However, there is an overhead associated with the use of multigrid acceleration. The overhead is higher in 2–D problems than in 3–D problems, thus overall multigrid savings in CPU time are in general better in the latter. Savings of about 40–50% are typical in 3–D problems, but they are about 20–30% in large 2–D problems. The present multigrid method is applicable to steady–state problems and is therefore ineffective in problems with inherently unstable solutions. (Updates for running the multigrid versions of the Proteus computer code and the stability analysis codes are contained in a supplement to this report.)

ACKNOWLEDGEMENT

This work is funded by NASA Lewis Research Center under Grant No. NAG–3–1329 with Dr. Jim Scott as Technical Monitor. Part of it was also conducted in residence at ICOMP, NASA Lewis Research Center. The authors are grateful for assistance received from Drs. Louis Povinelli, John Abbot and Charles Towne. Computations were performed on supercomputers at NASA Lewis and NASA Ames Research Centers.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

NOMENCLATURE

A, B, C	Cartesian inviscid Jacobians
CFL	Courant–Friedricks–Lewy number
c	sonic speed
c_p, c_v	specific heat constants
E, F, G	conserved inviscid fluxes in Cartesian or Curvilinear coordinates
E_v, F_v, G_v	viscous fluxes in Cartesian or Curvilinear coordinates
e_o	total specific internal energy
I	identity matrix
I	$\sqrt{-1}$
i, j, k	nodal points
k	thermal conductivity
L, N	generic operators
N, K, L	Fourier symbols
Pe	Peclet number
p	pressure
Q	solution vector
R	residuals
Re	Reynolds number
t	time
T	temperature
U	a constant vector
u, v, w	velocity components
X_A, X_B, X_C	eigenvectors of A, B and C
x, y, z	Cartesian coordinates

Greek Symbols

β	implicitness factor
γ	specific gas constant
Δ	incremental change
δ_{ij}	Kronecker delta
$\delta_x, \delta_y, \delta_z$	finite difference operators for first derivatives
$\varepsilon_e, \varepsilon_i$	constant coefficients of dissipation
$\Theta^1 = \theta_x, \theta_y, \theta_z$	modes in x, y, z directions
$\varkappa_2, \varkappa_4$	constant coefficient of dissipation
$\lambda_{\max}$	amplification factor
$\lambda_{\max\_bg}$	bi–grid amplification factor
$\lambda_\mu, \lambda_{\mu\_sg}$	smoothing factor
ν, λ	coefficients of viscosity
ϱ	density
ϱ_{mg}	multigrid asymptotic convergence rate

Superscripts

h, H	grid levels
–	non–positive eigenvalues
+	non–negative eigenvalues
n	time levels
^	Fourier symbols
*, **	intermediate solutions

Subscripts

i, j, k	grid points
∞	free–stream value
r	reference
x, y, z	indicates derivatives relative to these Cartesians directions

Chapter 1
INTRODUCTION

The field of Computational Fluid Dynamics (CFD) has been substantially developed to unravel the underlying physics of many complex flow phenomena that are difficult or even impossible to study experimentally. The success of CFD is directly linked with the rapid development of computers in the last two decades. A great number of numerical algorithms have been formulated to resolve the physics that characterize different aerodynamic fluid flow problems. The necessity to study the finely detailed models of physics in a steady or unsteady flow demands fine grid resolutions and a good choice of solution technique. For flows with engineering significance, the full Navier–Stokes equations, very often the Reynolds–averaged form, have been found to yield acceptable results for flow characteristics including heat transfer. However, even for this time–averaged approximation, the computational cost is often too expensive. To reduce this cost, acceleration techniques such as residual smoothing, local time stepping, enthalpy damping and multigrid are introduced. So far, multigrid is considered the most effective, especially when used to solve a strongly elliptic problem where only one or a few iterations are needed for convergence. Structurally, multigrid algorithms iterate on a hierarchy of consecutively coarser and coarser grids to accelerate convergence on the finest grid. However, the total computational work involved to capture real physical changes with multigrid is effectively less when compared to single grid computations.

1.1 Historical Review of Multigrid Methods

Multiple grids were first proposed in the form of two–grid level schemes to accelerate the convergence of iterative procedures by Southwell (1935), Stiefel (1952), Federenko (1961), amongst others. Full multiple grid methods were introduced for the Poisson equation by Federenko (1964) and the approach was generalized by Bakhalov (1966) to any second–order elliptic operator with continuous coefficients. According to Stuben and Trottenberg (1982), Hackbush (1976) also independently developed some fundamental elements of the multigrid method. Perhaps the most influential work on the application of multigrid methods to elliptic type problems is the paper by Brandt (1977) which also introduced the use of local mode analysis to determine the smoothing rates of multigrid schemes. Multigrid acceleration was also successfully applied to the transonic potential flow equation, which is of mixed elliptic–hyperbolic type, by South and Brandt (1976), Jameson (1979), McCarthy and Reyhner (1982), and a host of other researchers.

Most of the theory of the effectiveness of multigrid schemes pertained to problems with some measure of ellipticity. However, Ni (1981) proposed a distributed correction multigrid method based on an explicit scheme for solving the Euler equations in the steady state. Convergence acceleration due to the multigrid scheme was by at most a factor of five which was worse than typical speedup factors in applications to elliptic equations. Furthermore the scheme was only first–order accurate and was restricted to a CFL number of one. Jameson (1983) proposed an explicit four–stage time stepping multigrid algorithm for the steady–state Euler equations. The method was second–order accurate and the limiting CFL number for stability was 2.6—2.8. The mechanism for multigrid convergence acceleration to steady state in systems with little ellipticity is that larger time steps can be taken on coarser grids, while still maintaining the same CFL number, such that disturbances are more rapidly expelled through the boundaries. The interpolation of corrections from the coarse grid to the fine grid may introduce additional high frequency errors which must be rapidly damped if

the scheme is to be effective. Thus a requirement of any solution scheme to be used successfully in a multigrid procedure is that it rapidly dampens high frequency modes of the error.

Mulder (1989) presented a multigrid scheme to solve the two dimensional Euler equations with a finite–volume method which used van Leer's flux–vector splitting for upwind differencing and a symmetric Gauss–Siedel method as a relaxation scheme. Multigrid speedup factors were roughly nine and six for first–order and second–order accurate schemes, respectively. Anderson et al. (1988) also found similar multigrid convergence acceleration rates in the solution of the three–dimensional Euler equations with flux–vector splitting and three different approximate factorization schemes. Typically, 200—400 multigrid cycles were required for convergence to the level of the truncation errors. An interesting result was that although the three–factor spatially split factorization was stable only for CFL numbers below 20, it produced the fastest multigrid convergence of all the schemes. This was obtained at a CFL number of seven.

Jameson and Yoon (1986) presented finite–volume based multigrid methods for the 2–D Euler equations using an ADI scheme with approximate factorization. The differential operators were approximated with central differences with second and fourth–difference artificial dissipation terms added for stability and convergence. It was found that implicit fourth difference dissipation was required for efficient multigrid convergence. However, this required the solution of a block pentadiagonal system which was more expensive than the block–tridiagonal system resulting from the use of implicit second–difference dissipation. Compared with the single grid computation the multigrid speedup factor (based on residual reduction) was about eight in the former and four in the latter. Multigrid methods coupled with grid sequencing enabled quite rapid establishment of the solution fields, so that based on the buildup of the supersonic region, the speedup factors in the study above were twice as large. The problem with the ADI scheme as a baseline solver is that in

three–dimensions, a three–factor split is required and linearized stability analysis shows that this is only conditionally stable. To alleviate this problem Jameson and Yoon (1987) devised a multigrid method for 2–D Euler equations which used the lower–upper (LU) factorized implicit scheme of Jameson and Turkel (1981) as the baseline solver. Yokota and Caughey (1988) have developed a similar scheme for the calculation of three–dimensional transonic flow through rotating cascades. The scheme has only two factors and is unconditionally stable. It is indeed very similar to the flux–vector splitting method based on the eigenvalue–factored split investigated by Anderson et al. (1988). Their finding that although the three–factored split (similar to ADI) is only conditionally stable, it provided a better multigrid convergence rate than the unconditionally stable eigenvalue–split method (similar to LU), is noteworthy. However, one advantage of the LU scheme is that it requires cheaper block–bidiagonal inversions compared with block–tridiagonal or pentadiagonal inversions for an ADI scheme. The latter is necessary if implicit fourth–difference dissipation terms are used for better accuracy and convergence. Caughey (1988) demonstrated that block–pentadiagonal inversions in the ADI scheme could be reduced to scalar pentadiagonal ones by using a local similarity transformation to diagonalize the equations at each point. Thus, the computational work was reduced by a factor of four, and the decoupled system had similar convergence characteristics as the original one. Caughey and Iyer (1988) applied the scheme to solve the Euler equations for a supersonic inlet flow and found that the multigrid speedup factor was only 2.5, i.e., somewhat less than was found in transonic and subsonic flows. Yokota, Caughey and Chima (1988) also diagonalized the LU implicit multigrid scheme with no degradation in performance.

So far in this review, we have considered the application of multigrid methods to the Euler equations or potential flow equations. Several applications to the Navier–Stokes equations for incompressible fluid flow have been reported (Vanka (1986), Demuren (1989), Thompson and Ferziger (1989), Demuren (1992)). The relaxation schemes in all these applications are pressure–based in contrast to time–stepping schemes more common in

compressible flow applications. Multigrid speedup in the range of a few percent to factors of hundreds have been reported. It is likely that in the latter cases, the baseline relaxation scheme did not have good convergence properties for the particular applications. However, one of the attractions of the multigrid method is that a poor single–grid solver may actually have good high frequency smoothing properties and thus be an effective multigrid relaxation scheme. Rhie (1989) presented a pressure–based multigrid method for solving the Navier–Stokes equations over the range of flow speeds encompassing both the compressible and the incompressible fluid flow. Himansu and Rubin (1988) also presented a novel pressure–based multigrid method for the reduced Navier–Stokes equations for compressible and incompressible fluid flows. Apart from the obvious difficulties of the treatment of viscous terms and the implementation of a turbulence model, the solution of the Navier–Stokes equations usually requires the clustering of grids near walls in order to resolve the boundary layer, which often increases the stiffness of the system of equations and slows down the convergence rate of many iterative schemes. Multigrid convergence acceleration also tended to degrade with increase in Reynolds number. These difficulties fall under the category of problems with standard multigrid methods classified by Brandt (1977) as due to the alignment of coefficients of difference equations. He proposed that the problem be overcome by doing line relaxations in 2–D or plane relaxations in 3–D in the direction of alignment, or to perform only semi–coarsening of the grids in one of the directions instead of the more usual full coarsening, which should reduce the anisotropy of the coefficients. Himansu and Rubin (1988) implemented some aspect of both strategies with some success. Mulder (1989) considered the problem of alignment in somewhat more details and found that semi–coarsening in one direction was inadequate to cure it. Rather, it must be used in several directions at every grid level. Hence, in a 2–D problem two coarse grids are created for each finer grid, which implies that the total number of grid points and hence the operation count would be the same at each grid level. Such a scheme would negate one of the advantages of the multigrid method, namely, that all the computational work in performing

relaxations on coarse grids was cheaper than comparable work on the finest grid. So he devised a special procedure which ensured that on coarse grids, the total number of grids points was reduced and less computational work was done. The resulting scheme was shown to be efficient in resolving some flows with alignment, but it appears to be rather complicated to implement, and it is doubtful that it will find its way into a general purpose computer code anytime soon.

Implementation of the multigrid method in time–stepping solution schemes for the compressible Navier–Stokes equations appear to be a straightforward extension of that for the Euler equations. Although, for the reasons given above, worse performance may be expected. Chima, Turkel and Schaffer (1987) compared implementations of three types of multigrid methods in explicit time–stepping multistage solution methods for Euler and Navier–Stokes equations. They found the Full multigrid–Full approximation storage (FMG–FAS) method proposed by Brandt (1977) to be the most efficient producing speedup factors of about 8.5 in the solution of the Euler equations for selected problems, but only about 2.1 in the solution of the Navier–Stokes equations. Multigrid schemes which use explicit time–stepping algorithm to solve the 3–D, compressible Navier–Stokes equations have also been reported by Arnone and Swanson (1988), Radespiel et. al (1990) and Swanson and Radespiel (1991). These are mostly central–differencing approximation methods, and the choice of artificial dissipation was found to be very important for efficient convergence. Yokota (1989) extended the previous implementation for the Euler equations (Yokota et. al, (1988)) to the Reynolds–averaged, Navier–Stokes equations. The $k - \varepsilon$ turbulence model was used to approximate the Reynolds stresses. Application to the calculation of the three–dimensional flow through blade passages showed convergence rates similar to those for the Euler equations. The use of wall–functions meant that the boundary layer need not be fully resolved so that grids with very high aspect ratios could be avoided, and hence, the lack of performance degradation. A novel method for solving the

compressible, steady, Navier–Stokes equations was presented by Koren (1990). A first–order accurate upwind method with good smoothing properties was used for the discretization of the equations. Second–order accuracy was achieved through defect correction. The whole multigrid scheme exhibited good convergence characteristics in smooth flows, but somewhat poorer performance in non–smooth flows with shocks.

In the computation of flows in very complex geometries such as around multi–element airfoils or in complex inlet sections, two approaches are popular: unstructured grids or multiple blocks of structured grids. Multigrid acceleration has also been achieved in solutions of the Euler and Navier–Stokes equations with either approach. Mavripilis (1988, 1990) has demonstrated good multigrid convergence for the solution of the Euler equations on unstructured triangular meshes. Mavripilis and Jameson (1990) presented a similar implementation for the Navier–Stokes equations. Multigrid, multiblock methods were presented for the Euler equations by Yadlin and Caughey (1991) and for the Navier–Stokes equations by Baysal et. al. (1991) and Elmiligui (1992).

1.2 Time Integration

Accurate evolution of time–dependent fluid flow problems and the stability of numerical schemes are greatly dependent on the type of time integration employed. Time integration techniques that have been used to solve the Navier–Stokes equations can be broadly classified as either explicit or implicit schemes.

In explicit methods, a single set of unknown vectors that are required to be solved appears on the one side of the algebraic equations resulting from discretization. Solutions to these vectors at the present time are completely dependent on the solutions at previous times. Explicit methods are very easy to work with and need fewer operation counts, especially for unsteady problems. They are very efficient for unsteady flows with little variation in velocity and mesh density. However, they suffer from severe limitation on the time step due to

stability requirements. Where stability requirements dictate very small time steps, the temporal accuracy may be impaired and/or the computation time to drive the solution to steady state may become excessive. Also explicit techniques demand that each equation solved should have a time derivative term, but in incompressible flow this is absent in the continuity equation. In this case a special treatment (e.g. the introduction of artificial compressibility) may be necessary. This, of course, detracts from its advantages.

Implicit methods are desirable especially for stiff problems where disparate time scales are associated with the governing equations; e.g., in combustion processes. Implicit methods are unconditionally stable and thus allow for larger time steps, limited only by accuracy requirements, non–linearity and boundary treatment. Although they require larger operation counts when compared with explicit schemes, they may be optimum in time–dependent problems when the time scale of the unsteady phenomenon is much larger than the time step allowed by the Courant–Fredriechs–Lewis (CFL) condition (e.g., flow along an oscillating airfoil). The possibility of utilizing a larger time step than the CFL limit leads to a welcome gain in computational efficiency. Since a system of algebraic equations is solved either by direct or iterative methods at each time step, the implicit difference operator is constructed to guarantee diagonal dominance for convenient resolution of the equations. Sometimes in order to make the computation of the algebraic set of equations amenable to the tridiagonal matrix solution method, an implicit scheme can also be cast into a predictor–corrector form, where the implicit term is approximately factored into a set of smaller terms either over space or eigenvalue. The most popular methods include the Alternating Direction Implicit (ADI), the Lower and Upper (LU) decomposition and some upwind based factorization methods. In this work several kinds of approximate factorization schemes will be investigated for stability.

1.3 Stability Analysis

Although implicit numerical schemes allow for larger time steps for advancing the solution of the Euler and Navier–Stokes equations to steady state, approximate factorization (AF) is often introduced for efficiency. In the approximate factorization method, the complicated multi–dimensional matrix equation obtained at each time step is approximately factored into simpler one–dimensional terms which are easily invertible. Although this technique reduces the computational cost by taking advantage of the Tridiagonal Method (TDM), the approximation introduces errors that may place limitations on the CFL number and, thus, on the overall efficiency of the algorithm. As observed by Thomas et al. (1985), the approximately factored scheme has even greater stability restrictions in 3–D, and also an optimal convergence time step that is not known a priori. Therefore, to avoid the long and costly approach of trial and error of obtaining an optimal CFL number, it is highly desirable to carry out a stability analysis for any numerical scheme. Some researchers have found that analyzing scalar equations such as the convection or the diffusion equation can provide insight into the stability requirements for Euler and Navier–Stokes equations. Beam and Warming (1978) employed a combination of these scalar equations to approximate the restriction that were placed on their ADI methods for compressible Navier–Stokes equations. Jameson and Yoon (1986) and Caughey (1988), among others, used the scalar convection equation as a model problem for the Euler equations to investigate appropriate conditions for multigrid implementation. Rather than utilizing model equations, Jespersen and Pulliam (1983) developed a technique whereby Fourier analysis is extended to the actual coupled equations for the quasi–one–dimensional Euler equations. Jespersen (1983) further extended this technique to the 2–D Euler equations in order to find the best conditions at which to implement multigrid for a transonic flow. Thomas et al. (1985), von Lavante (1986) and Anderson et al. (1988) have also utilized a similar approach in the stability analysis of the Euler equations for certain approximate factorization and relaxation schemes. Finally,

Demuren and Ibraheem (1993, 1994) have also adopted this approach to investigate the stability of certain implicit solution techniques of the 3–D Euler and Navier–Stokes equations. Utilizing the frozen coefficients from actual supersonic and transonic flow fields of a quasi 1–D Euler equations, they further established the suitability of using uniform flow field in the stability analysis.

Substantial progress has been made to develop the multigrid method both theoretically and practically in all aspect of physics. However, the most influential work on the application of multigrid methods to elliptic type problems is, perhaps, that of Brandt (1977) who also proposed the use of local mode analysis to determine the smoothing rate of multigrid schemes.

In local mode analysis, the maximum eigenvalue (called the smoothing factor) of a particular relaxation technique computed over only the high–frequency modes is used as a measure of the relaxation's effectiveness in a multigrid scheme since, in this case, the role of relaxation is not to reduce the total error but to smoothen it out; i.e., remove the high–frequency components. It is assumed that the high–frequency modes have short wavelength that are spatially decoupled and that all high–frequency waves are completely "killed" on the fine grid and are not visible to the coarse grids. This, however, is not always the case since the inter–grid processes also influence the convergence rate. Brandt (1991) presented theoretical considerations for including the transfer processes in the local mode analysis in what is called the bi–grid method. Also, some theoretical background is given by Stuben and Trottenberg (1982) on how to compute a more realistic amplification factor for multigrid methods based on the bi–grid analysis, where some convergence norms were computed for the Poisson and Helmholz equations.

A number of works exist where the smoothing factor has been used to predict multigrid performance in practice. However, the bi–grid analysis is becoming more attractive because of its better accuracy and reliability. Van Asselt (1982) used the bi–grid analysis to determine

the proper amount of artificial viscosity to add at the different levels of coarse grids in a multigrid application. Mulder (1988, 1989) has also used the bi–grid method to construct an effective semi–coarsening in a multigrid method that can solve the problem of strong alignment which often occurs in convection problems. To select a relaxation scheme for a multigrid method suitable for a parallel solution of a time–dependent problem, Horton and Vandewall (1993) employed this technique using the heat equation as their model problem. The cause of the poor multigrid convergence rate that is experienced in high–Reynolds number flows (where the coarse grid corrections fail to approximate the fine grid problem well enough for certain components) has also been investigated by Brandt and Yavneh (1993) using the bi–grid method. In an effort to develop an effective multigrid algorithm for Navier–Stokes solutions on an unstructured grid with $O(N)$ complexity, Morano (1992), and Morano and Dervieux (1993) have used the bi–grid analysis on a 1–D model scalar convection equation with periodic boundary conditions. More recently, Ibraheem and Demuren (1994a) also presented some convergence norms for the Burger's equation based on bi–grid analysis.

Although implicit numerical schemes are becoming very popular, only few works exist to show the effectiveness of multigrid methods in these schemes especially when approximate factorization is introduced. Jameson and Yoon (1986) and Caughey (1988), for example, used the smoothing factor and scalar convection equation as a model for the Euler equations to investigate multigrid performance. Anderson et. al. (1988), and Demuren and Ibraheem (1993) have also computed the smoothing factors on the actual coupled Euler equations for some popular approximate factorizations. The latter work investigated the Navier–Stokes equations as well. In order to compare the predictive capability of smoothing factors with the spectral radius obtained from bi–grid analysis, Ibraheem and Demuren (1994b) considered the full Euler and Navier–Stokes Equations solved with different numerical schemes. In this work, amplification, smoothing and bi–grid factors for various implicit scheme solution methods for Euler and Navier–Stokes equations will be documented.

1.4 Objectives

The objectives of this work are as follows:

(1) To formulate von–Neumann type of stability analysis for the 1–D, 2–D and 3–D Euler and Navier–Stokes equations using various numerical schemes. Three upwind–difference based factorizations and two central–difference based factorizations will be selected for the Euler equations. In the upwind factorizations, two popular flux–vector splitting methods, one by Steger–Warming and the other by van Leer, will be used. The Lower and Upper (LU) factorization, and the Beam–Warming ADI methods will be assumed as the base–line algorithms for the central–difference schemes. Further, smoothing factors will be computed to establish the effectiveness of the selected schemes for multigrid application.

(2) To present a procedure for utilizing the bi–grid amplification factor as a more accurate tool for predicting practical multigrid performance in the above selected schemes. The predictive capability of the bi–grid method will be established using several model equations, including diffusion, convection and the Burger's equations; and several time–stepping methods, such as Euler forward explicit scheme, Runge–Kutta multistage scheme, a fully implicit scheme, and the semi–implicit scheme.

(3) To develop an efficient multigrid algorithm to solve steady state problems governed by the 2–D and 3–D Navier–Stokes equations based on the results from the above analyses for the Beam–Warming ADI method. The multigrid method is to be implemented in the two and three dimensional versions of the Proteus computer code (Towne et. al 1990, 1992), and the efficiency of the multigrid acceleration is to be demonstrated by application of the code to several test problems.

1.5 Report Outline

In Chap. 2, various approximate factorization methods are investigated for stability and their amplification factors and smoothing factors are computed. Detailed discussion is provided for the bi–grid analysis in Chap. 3. The bi–grid amplification factor for model problems as well as for Euler and Navier–Stokes equations are then computed under various numerical schemes. A brief description of the mathematical formulation of the Beam–Warming ADI method is presented in Chap. 4 for the Navier–Stokes equations, and a steady multigrid technique is introduced for this formulation to solve various steady state cases in Chap. 5. Finally, future research directions are pointed out in Chap. 6.

Chapter 2
SINGLE–GRID STABILITY ANALYSIS

In this chapter, the convergence characteristics of various approximate factorizations for the 3–D Euler and Navier–Stokes equations are examined using the von–Neumann stability analysis method. Three upwind–difference based factorizations and several central–difference based factorizations are considered for the Euler equations. In the upwind factorizations, both the flux–vector splitting methods of Steger–Warming and van Leer are considered. Analysis of the Navier–Stokes equations is performed only on the Beam–Warming central–difference scheme. In each case, the smoothing factor that is often used in predicting multigrid performance are also computed. Some issues central to stability analysis are further clarified.

2.1 Theory and Analysis

The Fourier analysis is adopted to study the stability analysis of the coupled Euler and Navier–Stokes equations. A discrete analog of these equations is formulated based on various approximate factorizations in this section. The Euler equations are first analyzed using upwind and LU factorizations. The ADI factorization is formulated for the Navier–Stokes equations with the Euler equations as a degenerate case.

2.1.1 Upwind Approximate Factorizations for Euler Equations

The conservation form of the 3–D Euler equations in Cartesian coordinates can be written as:

$$\frac{\partial Q}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = 0 \tag{2.1}$$

where Q is the solution vector and E, F, G are the conserved inviscid fluxes:

$$\begin{aligned} Q &= [\varrho,\ \varrho u,\ \varrho v,\ \varrho w,\ \varrho e_o]^T \\ E &= \left[\varrho u,\ \varrho u^2 + p,\ \varrho uv,\ \varrho uw,\ (\varrho e_o + p)u\right]^T \\ F &= \left[\varrho v,\ \varrho vu,\ \varrho v^2 + p,\ \varrho vw,\ (\varrho e_o + p)v\right]^T \\ G &= \left[\varrho w,\ \varrho wu,\ \varrho wv,\ \varrho w^2 + p,\ (\varrho e_o + p)w\right]^T \end{aligned} \tag{2.2}$$

If the Euler implicit scheme is used for time discretization, Eq. (2.1) can be written in the following form of the augmented Newton's method (Fletcher, 1991):

$$\left[\mathbf{I} + \Delta t(\partial_x A^n + \partial_y B^n + \partial_z C^n)\right]\Delta Q^n = -\Delta t\left(\partial_x E^n + \partial_y F^n + \partial_z G^n\right) \tag{2.3}$$

where the Jacobians A, B, C are $\partial E/\partial Q$, $\partial F/\partial Q$, $\partial G/\partial Q$, respectively. The expressions for A, B, C and are given in Appendix A. In upwind formulations, these fluxes are split to match the direction of the physical propagation of the solutions. Based on the direction of the characteristics at each grid point, the fluxes are split into their forward and backward contributions. Denoting the forward contribution with + and the backward with –, and forward and backward difference operators with δ^+ and δ^-, respectively, we can rewrite Eq. (2.3) as

$$\begin{aligned} &\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_x^+ A^-) + \Delta t(\delta_y^- B^+ + \delta_y^+ B^-) + \Delta t(\delta_z^- C^+ + \delta_z^+ C^-)\right]\Delta Q \\ &\quad = -\Delta t(\delta_x^- E^+ + \delta_x^+ E^- + \delta_y^- F^+ + \delta_y^+ F^- + \delta_z^- G^+ + \delta_z^+ G^-) = -\Delta t R^n \end{aligned} \tag{2.4}$$

The left hand side of the equation is usually approximated with first–order differences, but the right hand side uses second–order differences to improve the overall accuracy of the converged solution. However, even with first–order difference approximations of the

implicit terms, the equation is computationally expensive to solve. To reduce this cost, the implicit operator is factored into a sequence of easily invertible terms. Following Anderson et al. (1988) we will consider the following three factorizations:

$$\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_x^+ A^-)\right]\left[\mathbf{I} + \Delta t(\delta_y^- B^+ + \delta_y^+ B^-)\right]\left[\mathbf{I} + \Delta t(\delta_z^- C^+ + \delta_z^+ C^-)\right]\Delta Q = -\Delta t R^n \tag{2.5}$$

$$\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_y^- B^+ + \delta_z^- C^+)\right]\left[\mathbf{I} + \Delta t(\delta_x^+ A^- + \delta_y^+ B^- + \delta_z^+ C^-)\right]\Delta Q = -\Delta t R^n \tag{2.6}$$

$$\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_x^+ A^- + \delta_z^- C^+)\right]\left[\mathbf{I} + \Delta t(\delta_y^- B^+ + \delta_y^+ B^- + \delta_z^+ C^-)\right]\Delta Q = -\Delta t R^n \tag{2.7}$$

Equations (2.5), (2.6) and (2.7) shall be referred to as the spatial, eigenvalue and combination factorizations, respectively. There are different ways of obtaining the split fluxes expressed in the above equations but two popular methods viz.: Steger–Warming (1980) flux–vector splitting, and van Leer (1982) flux–vector splitting, are considered in this work. In the Steger–Warming case, the fluxes are obtained from the following transformation:

$$A^+ = X_A D_A^+ X_A^{-1}, \quad A^- = X_A D_A^- X_A^{-1}, \quad etc\,. \tag{2.8}$$

where D_A^+ and D_A^- are diagonal matrices whose elements are the positive and negative eigenvalues of A, respectively, and the columns of X_A are the eigenvectors of Jacobian A. The terms E^+ and E^- are obtained from $E^+ = A^+ Q$, $E^- = A^- Q$ etc. Equation (2.8) gives approximate values for A^+ and A^-. The exact (true) values are obtained from (see Jespersen and Pulliam, 1983):

$$A^+ = \frac{\partial E^+}{\partial Q}, \quad A^- = \frac{\partial E^-}{\partial Q}, \quad etc\,. \tag{2.9}$$

In order to resolve the singular nature of the Steger–Warming flux–vector splitting at the sonic speed, van Leer proposed the following splitting in Cartesian coordinates:

$$E^{\pm} = \pm \frac{\varrho(u+a)^2}{4a} \begin{bmatrix} 1 \\ \dfrac{(\gamma-1)u \pm 2a}{\gamma} \\ v \\ w \\ \dfrac{[(\gamma-1)u \pm 2a]^2}{2(\gamma^2-1)} + \dfrac{1}{2}(v^2+w^2) \end{bmatrix} \tag{2.10}$$

With similar forms for F^+, the Jacobians A^+, A^- etc. are obtained from Eq. (2.9). The analytical expressions for these can be obtained using a symbolic manipulator such as Mathematica. In these expressions, van Leer (1982) ensured continuous differentiability of the fluxes especially at the sonic transition and the stagnation point (Hirsh, 1990).

2.1.2 LU Approximate Factorization for the Euler Equations

This approach has become popular in recent times. It factors the implicit term of Eq. (2.3) into two components such that each component is strictly either a lower (L) or an upper (U) matrix as in the following equation:

$$\begin{aligned} \left[\mathbf{I} + \Delta t(\delta_x^- A_1 + \delta_y^- B_1 + \delta_z^- C_1)\right]\left[\mathbf{I} + \Delta t(\delta_x^+ A_2 + \delta_y^+ B_2 + \delta_z^+ C_2)\right]\Delta Q \\ = -\Delta t\left(\delta_x E + \delta_y F + \delta_z G\right) \end{aligned} \tag{2.11}$$

The Jacobian matrices are split to ensure diagonal dominance for each matrix inversion at each grid point. For our numerical computation we have adopted the flux–vector splitting devised by Jameson and Turkel (1981) viz.:

$$A_1 = \frac{(A + r_A\mathbf{I})}{2}, \quad A_2 = \frac{(A - r_A\mathbf{I})}{2}, \quad etc\,. \tag{2.12}$$

In the above, $r_A \geq \max(|\lambda_A|)$ etc. and λ_A are the eigenvalues of matrix A, viz.: $u+a$, $u-a$, u, u, u.

The explicit terms are central differenced and it is necessary to damp the associated high frequency waves and/or to correct the odd–even decouplings. In this study, the following combination of second– and fourth–order explicit linear dissipations, D_x^e, is employed. According to Caughey (1988), and Yokota and Caughey (1988), the former term is necessary for any spurious waves at the vicinity of shock while the latter ensures convergence to steady state.

$$D_x^e = \left(\varkappa_2 \Delta t \Delta x \delta_{xx} - \varkappa_4 \Delta t \Delta x^3 \delta_{xxxx}\right) Q \tag{2.13}$$

Noting that $\delta_{xx} = (1/\Delta x)(\delta_x^+ - \delta_x^-)$, the second–order term is split in a manner consistent with the differencing of the Jacobians and is implemented implicitly as often done in practice. Thus, with similar terms in the y and z directions, and their addition to Eq. (2.11) as diagonal matrix coefficients, we can write

$$\begin{aligned} &\left[\mathrm{I} + \Delta t(\delta_x^- A_1 + \delta_y^- B_1 + \delta_z^- C_1) + \varkappa_2 \Delta t(\delta_x^- + \delta_y^- + \delta_z^-)\right] \\ &\qquad \mathrm{X}\left[\mathrm{I} + \Delta t(\delta_x^+ A_2 + \delta_y^+ B_2 + \delta_z^+ C_2) - \varkappa_2 \Delta t(\delta_x^+ + \delta_y^+ + \delta_z^+)\right] \Delta Q \\ &\qquad = -\Delta t\left(\delta_x E + \delta_y F + \delta_z G\right) - \varkappa_4 \Delta t\left(\Delta x^3 \delta_{xxxx} + \Delta y^3 \delta_{yyyy} + \Delta z^3 \delta_{zzzz}\right) Q \end{aligned} \tag{2.14}$$

This factorization is similar to the eigenvalue factorization (see Eq. (2.6)) except that the explicit terms are centrally differenced rather than upwinded, thus, requiring the addition of dissipation. Also, the split fluxes of Jameson and Turkel which are less difficult to derive are used to achieve diagonal dominance in this case. $\varkappa_2$ and $\varkappa_4$ are second– and fourth–order coefficients of dissipation, and although $\varkappa_2$ is implemented implicitly, it is essentially an explicit dissipation coefficient.

2.1.3 ADI Factorizations for Euler and Navier–Stokes Equations

The 3–D Navier–Stokes equations in Cartesian coordinates can be written as

$$\frac{\partial Q}{\partial t} + \frac{\partial (E - E_v)}{\partial x} + \frac{\partial (F - F_v)}{\partial y} + \frac{\partial (G - G_v)}{\partial z} = 0 \tag{2.15}$$

where E, F, G are as defined earlier, and E_v, F_v, G_v are the viscous fluxes given by

$$\begin{aligned}
E_v &= \Big[0,\ \tfrac{2}{3}\mu(2u_x - v_y - w_z),\ \mu(u_y + v_x),\ \mu(u_z + w_x), \\
&\qquad \mu v(u_y + v_x) + \mu w(u_z + w_x) + \tfrac{2}{3}\mu u(2u_x - v_y - w_z) + kT_x\Big]^T \\
F_v &= \Big[0,\ \mu(u_y + v_x),\ \tfrac{2}{3}\mu(2v_y - u_x - w_z),\ \mu(v_z + w_y), \\
&\qquad \mu u(u_y + v_x) + \mu w(v_z + w_y) + \tfrac{2}{3}\mu v(2v_y - u_x - w_z) + kT_y\Big]^T \\
G_v &= \Big[0,\ \mu(w_x + u_z),\ \mu(v_z + w_y),\ \tfrac{2}{3}\mu(2w_z - v_y - u_x), \\
&\qquad \mu u(w_x + u_z) + \mu v(v_z + w_y) + \tfrac{2}{3}\mu w(2w_z - v_y - u_x) + kT_z\Big]^T
\end{aligned} \tag{2.16}$$

In Eq. (2.16), $T = p/[\varrho c_v(\gamma - 1)]$, and p is as defined in Appendix A. Also, Stokes hypothesis ($\lambda = -(2/3)\mu$) has been assumed. With E_v, F_v, G_v set to zero, we recover the Euler equations (2.1). Using the Beam and Warming (1978) scheme, the viscous fluxes are split directionally. Following the approach presented in Anderson et al. (1984) for 2–D Navier–Stokes equations, analysis yields the following ADI approximate factorization for the 3–D Navier–Stokes equations. Here, Euler time integration and constant fluid properties are assumed.

$$\begin{aligned}
&[\mathbf{I} + \Delta t(\delta_x A - \delta_{xx} R)]\big[\mathbf{I} + \Delta t(\delta_y B - \delta_{yy} S)\big][\mathbf{I} + \Delta t(\delta_z C - \delta_{zz} Y)]\Delta Q = \\
&\qquad -\Delta t\big[A\delta_x - R\delta_{xx} - R_1\delta_{yx} - R_2\delta_{zx} + B\delta_y - S_1\delta_{xy} - \\
&\qquad\qquad S\delta_{yy} - S_2\delta_{zy} + C\delta_z - Y_1\delta_{xz} - Y_2\delta_{yz} - Y\delta_{zz}\big]Q
\end{aligned} \tag{2.17}$$

The analytical expression for the various Jacobians (from the viscous fluxes) that appear in this equation are shown in Appendix B. The right–hand side resulted from linearization and from assuming the flux Jacobians to be locally constant. To damp the high–frequency waves that will arise due to central differencing, second–order implicit ($D_x^i = -\varepsilon_i \Delta t \Delta x \delta_{xx}$) and fourth–order explicit ($D_x^e = -\varepsilon_e \Delta t \Delta x^3 \delta_{xxxx}$) artificial dissipations are added as diagonal

matrix coefficients in the numerical examples. Thus, with similar dissipations added in the y and z directions Eq. (2.17) becomes

$$\begin{aligned}
&\left[\mathbf{I} + \Delta t(\delta_x A - \delta_{xx} R - \varepsilon_i \Delta x \delta_{xx})\right]\left[\mathbf{I} + \Delta t(\delta_y B - \delta_{yy} S - \varepsilon_i \Delta y \delta_{yy})\right] \\
&\mathrm{X}\left[\mathbf{I} + \Delta t(\delta_z C - \delta_{zz} Y - \varepsilon_i \Delta z \delta_{zz})\right] \Delta Q \\
&\quad = -\Delta t\Big[A\delta_x - R\delta_{xx} - R_1\delta_{yx} - R_2\delta_{zx} + B\delta_y - S_1\delta_{xy} - S\delta_{yy} - S_2\delta_{zy} \\
&\qquad + C\delta_z - Y_1\delta_{xz} - Y_2\delta_{yz} - Y\delta_{zz} + \varepsilon_e(\Delta x^3\delta_{xxxx} + \Delta y^3\delta_{yyyy} + \Delta z^3\delta_{zzzz})\Big] Q
\end{aligned} \tag{2.18}$$

The corresponding factorization for the Euler equations becomes transparent if the viscous flux Jacobians $R,\ R_1,\ R_2,\ S,\ S_1,\ S_2,\ Y,\ Y_1,\ Y_2$ are set to zero. ε_i and ε_e are second– and fourth–order coefficients of dissipation just as $\varkappa_2$ and $\varkappa_4$ except that ε_i is an implicit dissipation coefficient.

In the forgone analyses, different approximate factorizations that are widely used in practice have been formulated for the 3–D Euler and Navier–Stokes equations. The convergence characteristics of each of these are examined using the von–Neumann type Fourier analysis methods.

2.1.4 von–Neumann Stability Analysis

Each of Eqs. (2.5)–(2.7), (2.14) and (2.18) can be expressed as

$$N\Delta Q^n = -L = -\Delta t R^n \tag{2.19}$$

von–Neumann stability analysis is used on this system of linear Eq. (2.19) by letting the step by step solution be characterized by

$$Q^n = U_o \lambda^n e^{Ii\theta_x} e^{Ij\theta_y} e^{Ik\theta_z} \tag{2.20}$$

where λ is the amplification factor and $\{\theta_x, \theta_y, \theta_z\}$ represent the modes in the x–, y– and z–directions. Thus, Eq. (2.19) reduces to a complex generalized eigenvalue problem of the form (Andersen et al., 1988):

$$\hat{K}\mathbf{x} = \lambda\hat{N}\mathbf{x} \quad where \quad \hat{K} = \hat{N} - \hat{L} \tag{2.21}$$

The Fourier symbols are derived for each of the factorizations shown in Eqs. (2.5)–(2.7), (2.14) and (2.18). For example, for the Euler Equations based on the Beam–Warming scheme (from Eq. (2.18), these two Fourier symbols are expressed as follows:

$$\hat{N}(\Theta^1) = \left[\mathbf{I} + \frac{\Delta t}{\Delta x}\left(AI\sin(\theta_x) + 4\varepsilon_i \sin^2\frac{\theta_x}{2}\right)\right]\left[\mathbf{I} + \frac{\Delta t}{\Delta y}\left(BI\sin(\theta_y) + 4\varepsilon_i \sin^2\frac{\theta_y}{2}\right)\right] \mathrm{X}\left[\mathbf{I} + \frac{\Delta t}{\Delta z}\left(CI\sin(\theta_z) + 4\varepsilon_i \sin^2\frac{\theta_z}{2}\right)\right] \tag{2.22}$$

$$\hat{L}(\Theta^1) = \frac{\Delta t}{\Delta x} I\left(A\sin(\theta_x) + B\sin(\theta_y) + C\sin(\theta_z)\right) + \frac{16\Delta t\varepsilon_e}{\Delta x}\left(\sin^4\frac{\theta_x}{2} + \sin^4\frac{\theta_y}{2} + \sin^4\frac{\theta_z}{2}\right) \tag{2.23}$$

In the preceding equations, $\Theta^1 = \{\theta_x, \theta_y, \theta_z\}$. The Fourier symbols corresponding to the other approximate factorizations are documented in Demuren and Ibraheem (1992).

2.2 Solution Procedure

The convergence characteristics for solution algorithms based on each of the factorizations discussed are investigated by solving the generalized eigenvalue problem (2.21) over a fixed number of Fourier modes. Sixteen modes are selected, in the range $-\pi/2 \leq \Theta^1 \leq \pi/2$, and over these modes the maximum eigenvalue ($\lambda_{\max}$), the average eigenvalue (λ_{avg}) and the smoothing factor (λ_μ) are computed. The smoothing factor is computed to show the effectiveness of the selected scheme as a relaxation operator in a multigrid implementation.

This is calculated from $\lambda_\mu = \max(|\lambda|)$ for the high frequency modes in the range $\pi/4 \leq |\Theta^1| < \pi/2$. For the analyses, uniform flow is assumed with $M_\infty = 0.8$, zero yaw and angle of attack and $\gamma = 1.4$. Further, the grid spacing is assumed to be uniform in all directions. The time step and Reynolds number are calculated from:

$$\Delta t = \frac{CFL}{\left[\frac{|u|}{\Delta x} + \frac{|v|}{\Delta y} + \frac{|w|}{\Delta z} + c\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}}\right]} \tag{2.24}$$

$$\mathrm{Re} = \frac{\varrho |V| \left(\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}\right)}{\mu} \tag{2.25}$$

As a further test case, the quasi–one–dimensional Euler equations are solved with a similar formulation as the 3–D upwind spatial factorization, with uniform conditions of $M_\infty = 0.5$, zero yaw and angle of attack and $\varrho = 1.0$, chosen to enable comparison with Jespersen and Pulliam's results (1983). In this case, the computed parameters are the maximum eigenvalue ($\lambda_{\max}$), the L2–norm of the eigenvalue (l_2) and the eigenvalue at $\theta_x = \pi$, (λ_π). A batch file used to submit a typical 3–D test case is shown in Appendix D.

2.3 Results and Discussion

Computed values of the maximum eigenvalue ($\lambda_{\max}$), the average eigenvalue (λ_{avg}) and the smoothing factor (λ_μ) for the spatial, eigenvalue and combination factorizations based on the Steger and Warming flux–vector splitting are shown in Figs. 2.1(a), 2.1(b) and 2.1(c), respectively. Both the eigenvalue and the combination factorizations are unconditionally stable for all CFL numbers. The spatial factorization is stable only for CFL numbers below five. The maximum eigenvalue for each of the spatial, eigenvalue and combination factorizations is minimized at CFL numbers of three, eight and seven, respectively. Corresponding results obtained for 2–D case (not shown) indicate that the spatial and eigenvalue factorizations are unconditionally stable and have lower $\lambda_{\max}$ than the 3–D case,

for all CFL numbers. The corresponding minimum value of are minimized at a CFL numbers of eight and ten, respectively. The 1–D case is also stable for all CFL numbers with the maximum eigenvalue minimized at a CFL number of 11, for both spatial and eigenvalue factorizations (Table 2.1).

Figures 2.1(d–f) show the convergence characteristics of each of the factorizations based on the van Leer flux–vector splitting. These agree very well with that of Anderson et al. (1988). Except for the spatial factorization, all the schemes are unconditionally stable for all CFL numbers. The spatial factorization is stable only for CFL number below 14. The maximum eigenvalues for the spatial, eigenvalue and combination factorizations are minimized at CFL numbers of seven, four and seven, respectively. From the λ_μ curve, it appears that the spatial factorization with the Steger and Warming method has poorer smoothing properties compared with the van Leer spatial factorization. Based on linear analysis, there is also a smaller range of CFL numbers over which it is stable. The spatial factorization and the eigenvalue factorization of the 2–D case are found to be unconditionally stable with maximum eigenvalue minimized at CFL numbers of about nine and six, respectively. Results for the 1–D case are almost identical to those of the Steger and Warming analysis, with maximum eigenvalues minimized at CFL numbers of 11 and 19, respectively.

In the computations presented thus far, approximate Jacobians derived from a time linearization of the Euler equations have been employed in the Steger and Warming method on both the implicit and explicit sides. The effect of using the exact Jacobians in the stability analysis was investigated with the 1–D Euler equations using uniform conditions of $M_\infty = 0.5$ and $\varrho = 1.0$. The results are compared in Figs. 2.2(a) and 2.2(b), respectively. In both cases, first–order differencing was used on the implicit side (lhs) and second–order differencing on the explicit side (rhs), as in previous computations. From these figures, it can be observed that the results (as reflected by the variation of $\lambda_{\max}$, l_2, λ_π with CFL) are similar. This shows that the use of an approximate Jacobians does not place a

restriction on the stability. This is at variance with the conclusion of Jespersen and Pulliam (1983). Restriction on the stability will result if the Jacobians are "mixed" such that approximate Jacobians are used on the implicit side and the exact Jacobians on the explicit side. In this case, Fig. 2.2(c) shows that the stability is restricted to CFL numbers below unity. On the other hand, if the Jacobians are mixed in the reverse order (i.e., with exact Jacobians on the implicit side and approximate Jacobians on the explicit side), the results (see Fig. 2.2(d)) are not significantly affected. Further, from Figs. 2.3(a–d), where we have used second–order differencing on both sides, similar conclusions can be drawn.

All computations have been based on uniform flow conditions. To ascertain the suitability of using such uniform flow field assumptions in the stability analysis, computations were carried out on two non–uniform flow fields with the quasi–1–D Euler equations using local mode analysis. These correspond to supersonic and transonic flows in a diverging duct with steady–state solutions, shown in Figs. 2.4(a) and 2.4(b), respectively. The von–Neumann method is applied at each point in the flow field thereby accounting for the variation in flow properties. The stability results for the supersonic and transonic cases with first–order differencing on the implicit side and second–order differencing on the explicit side are shown in Figs. 2.4(c) and 2.4(d), respectively. These results follow a similar trend as those obtained for the 1–D Euler equations with uniform flow properties, except that instability is now predicted for lower CFL numbers. Boundary conditions were implemented explicitly and might have contributed to this instability. The use of local mode analysis here, is similar to the use of the total matrix method approach of Jespersen and Pulliam (1983), except that the former is easier to compute because it involves the solution of only a 3 X 3 eigenvalue problem.

Figures 2.5(a–c) show the convergence characteristics of the 3–D Euler equations using the LU approximate factorization with central difference approximations and various levels of second– and fourth–order artificial viscosities, $\varkappa_2$ and $\varkappa_4$. Without the addition of

second–order dissipation (i.e., $\varkappa_2 = 0$), the coefficient $\varkappa_4 = 0.4$ yields the optimal results (see Fig. 2.5(a)). Appropriate combinations of $\varkappa_2$ and $\varkappa_4$ (especially when $\varkappa_4 \geq \varkappa_2$) considerably reduce the amplification factor (see Fig. 2.5(b) as compared with Fig. 2.5(c)). The amplification factor is minimized in each case at a CFL of about five. Similar trends were observed in 1–D and 2–D cases.

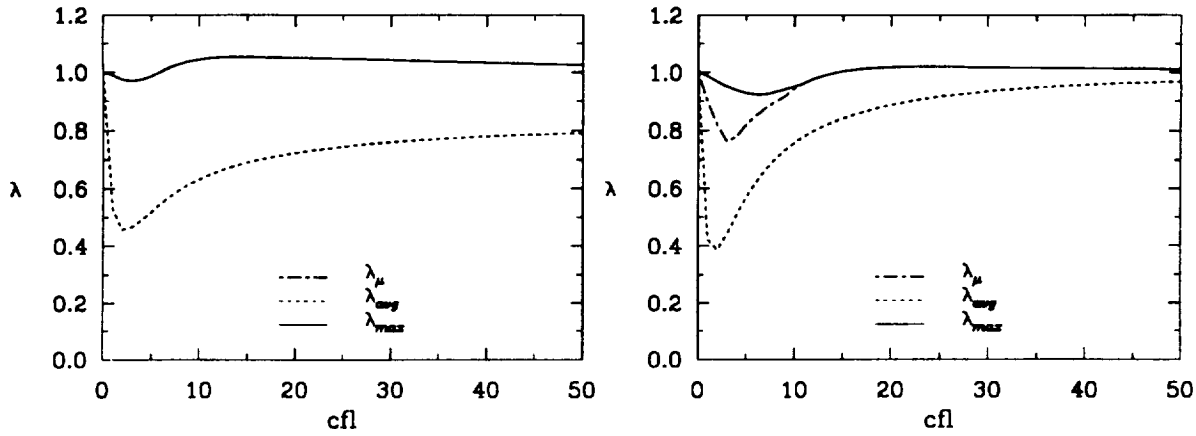
In Figs. 2.6(a–f), the convergence characteristics for the full 3–D Naiver–Stokes equations using the Beam and Warming (ADI) central difference scheme as the baseline solution algorithm are shown for different Reynolds numbers and levels of artificial dissipation. For the Reynolds number of 100 (Fig. 2.6(a)) and with no dissipation added, the scheme is stable for CFL number below 18. However, with artificial dissipation coefficients of $\varepsilon_e = 0.5$ and $\varepsilon_i = 1.0$ (Fig. 2.6(b)), the stability is restricted to a lower CFL number of 10, but with better smoothing properties. Optimal dissipation coefficients of $\varepsilon_i = 1.0$ and $\varepsilon_i = 2.0$ (Fig. 2.6(c)), are found to improve the stability to a CFL of about 18 while maintaining good smoothing properties. The maximum eigenvalue is minimized at a CFL number of about four for this optimal dissipation. Both 1–D and 2–D cases are unconditionally stable for all levels of dissipation. For $\varepsilon_i = 1.0$ and $\varepsilon_i = 2.0$, their maximum eigenvalues are both minimized at about CFL numbers of 24 and 11, respectively. The results are mostly similar at the higher Reynolds number, except for the case without artificial dissipation. Hence, the stability results are not significantly affected by Reynolds number. The stability results for the 3–D Euler equations with the Beam and Warming (ADI) central difference scheme are similar to those obtained for the full Navier–Stokes equations at the Reynolds number of 10^6. Generally, the addition of dissipation reduces the amplification factor and the smoothing factor at lower CFL numbers. Optimal smoothing is usually at a CFL number close to unity.

The above results are summarized in Table 2.1. In the Table, λ_m stands for the minimum amplification factor, CFL_m for the corresponding CFL number, CFL_l the maximum CFL number for stability and CFL_μ is the CFL number at which λ_μ is minimized.

Table 2.1: Stability Analysis results for various Factorizations

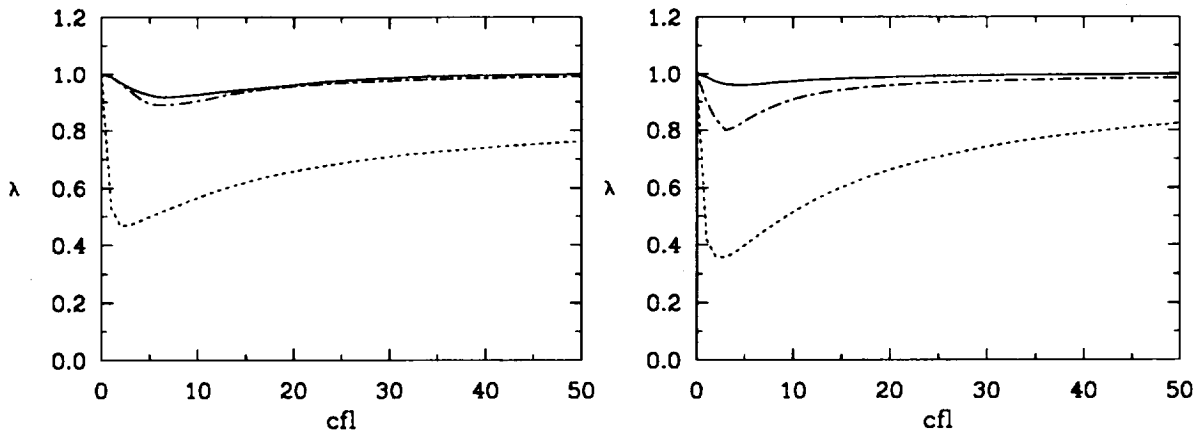
Factorization or Dissipation Coefficients		1-D				2-D				3-D			
		λ_m	CFL_m	CFL_I	CFL_μ	λ_m	CFL_m	CFL_I	CFL_μ	λ_m	CFL_m	CFL_I	CFL_μ
Spatial	Steger/Warming	.91	11	-	3	.90	8	-	7	.97	3	5	3
	van Leer	.91	11	-	3	.88	9	-	4	.92	7	14	3
Eigenvalue	Steger/Warming	.91	11	-	3	.88	10	-	7	.91	7	-	6
	van Leer	.83	19	-	7	.93	6	-	3	.96	4	-	3
Combination	Steger/Warming	-	-	-	-	-	-	-	-	90	8	-	7
	van Leer	-	-	-	-	-	-	-	-	.91	7	-	4
LU	κ_2=0, κ_4=.4	.98	5	-	1	.97	4	-	1	.97	3	-	1
	κ_2=2, κ_4=3	.93	4	-	2	.92	3	-	2	.91	3	-	2
	κ_2=3, κ_4=2	.96	4	-	1	.95	3	-	1	.95	3	-	1
ADI Navier-Stokes $Re=10^2$	ϵ_2=0, ϵ_4=0	-	-	-	-	.93	9	-	6	.95	7	18	5
	ϵ_2=1, ϵ_4=.5	.94	24	-	1	.90	13	-	1	.96	6	10	1
	ϵ_2=2, ϵ_4=1	.96	24	-	1	.93	11	-	1	.98	4	18	1
ADI* Navier-Stokes $Re=10^6$	ϵ_2=0, ϵ_4=0	-	-	-	-	.93	8	-	8	.97	8	16	1
	ϵ_2=1, ϵ_4=.5	.94	23	-	1	.91	13	-	1	.96	6	10	1
	ϵ_2=2, ϵ_4=1	.96	24	-	1	.93	12	-	1	.97	4	18	1

* Results for the ADI Euler Equations are identical.



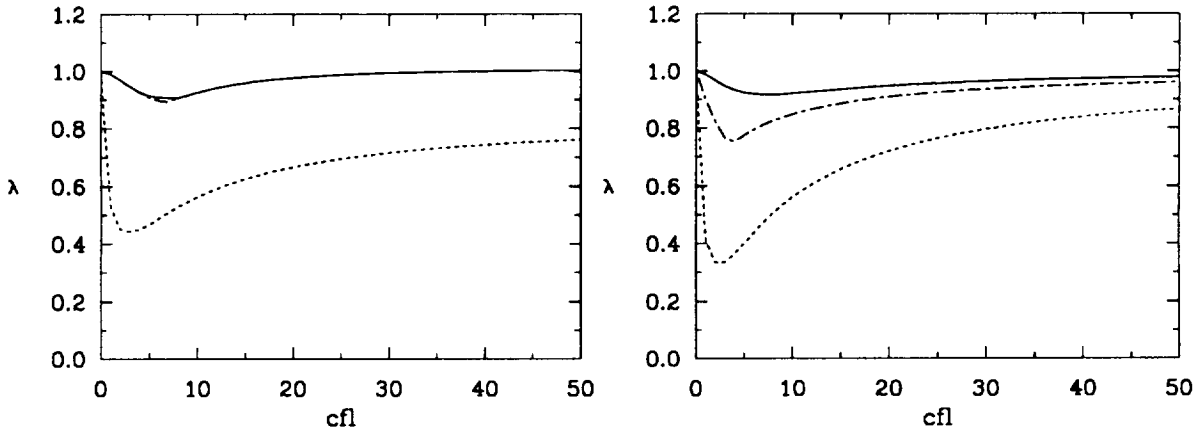
(a) Spatial Factorization

(d) Spatial Factorization



(b) Eigenvalue Factorization

(e) Eigenvalue Factorization



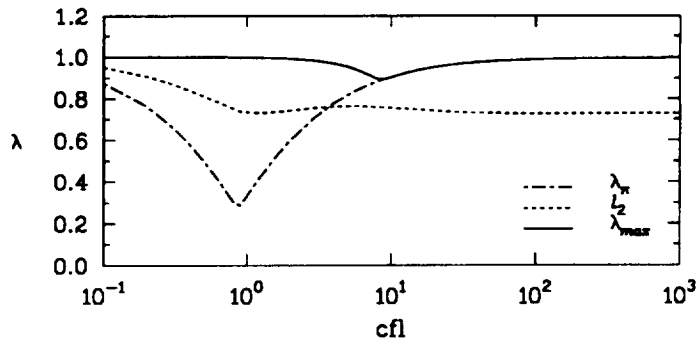
(c) Combination Factorization

(f) Combination Factorization

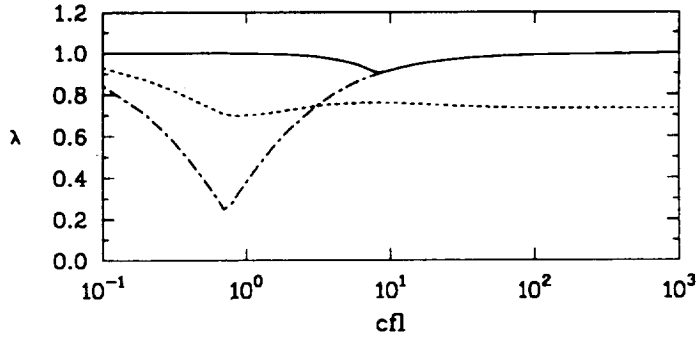
Steger–Warming

van Leer

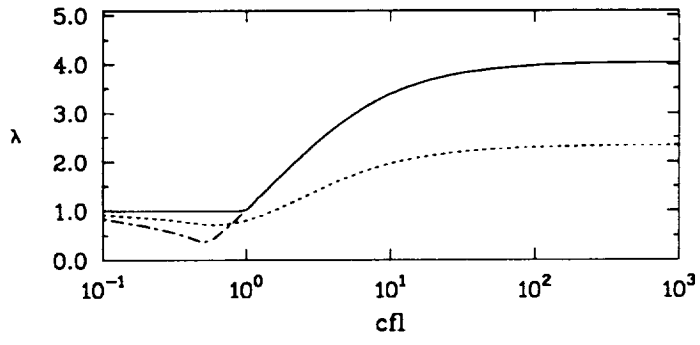
Fig. 2.1: 3–D Euler Equations using upwind schemes
(a)–(f) Convergence Characteristics



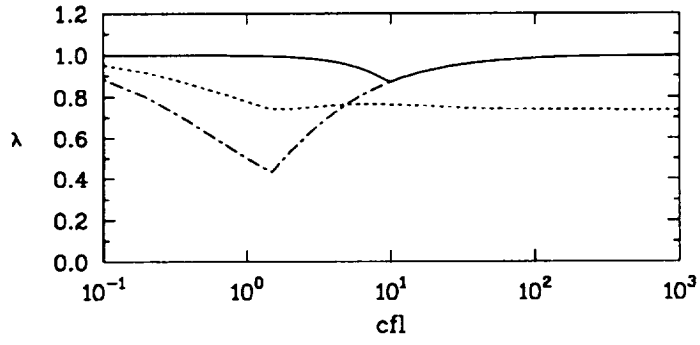
(a) Approx. Jacobians (lhs, rhs)



(b) Exact Jacobians (lhs, rhs)

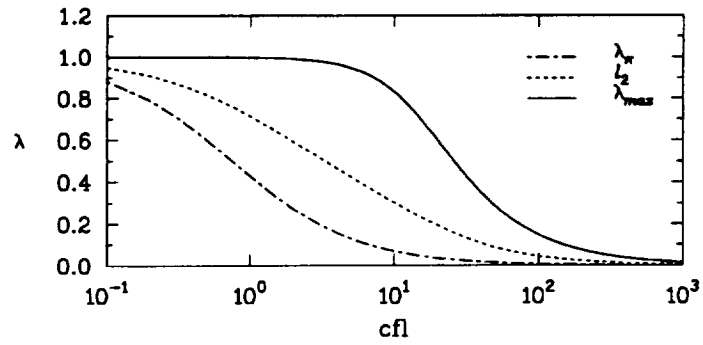


(c) Approx. Jacobians lhs; Exact Jacobians rhs

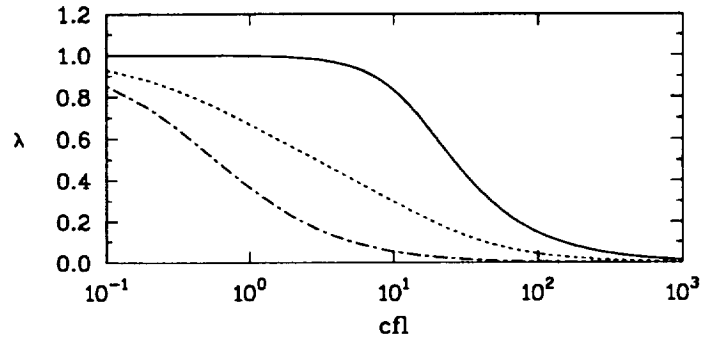


(d) Exact Jacobians lhs; Approx. Jacobians rhs

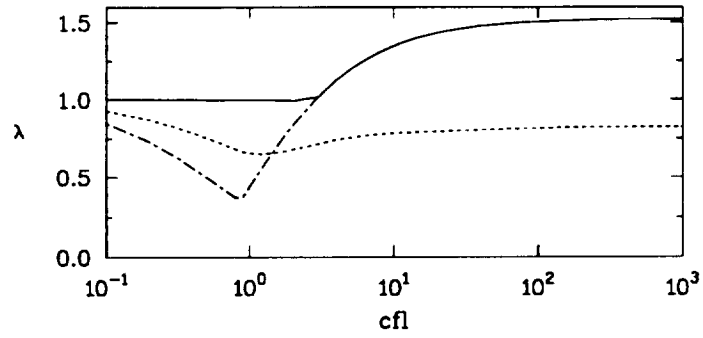
Fig. 2.2: 1–D Euler Equations using Steger–Warming schemes, first–order lhs, second–order rhs. (a)–(d) Convergence Characteristics.



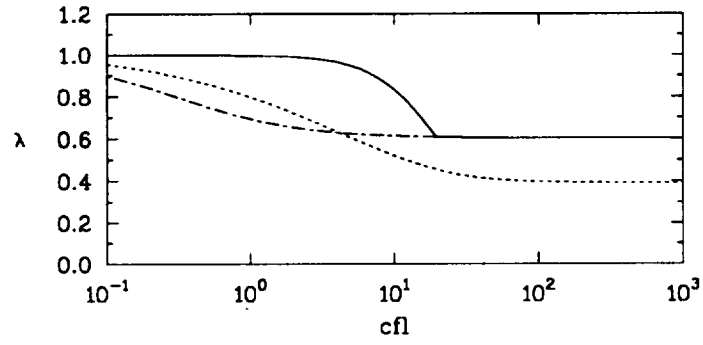
(a) Approx. Jacobians (lhs, rhs)



(b) Exact Jacobians (lhs, rhs)

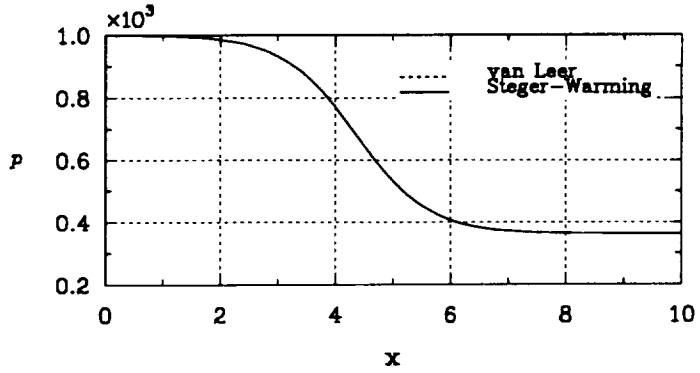


(c) Approx. Jacobians lhs; Exact Jacobians rhs

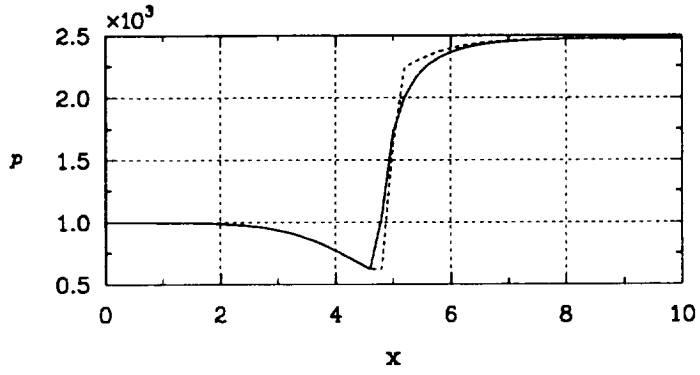


(d) Exact Jacobians lhs; Approx. Jacobians rhs

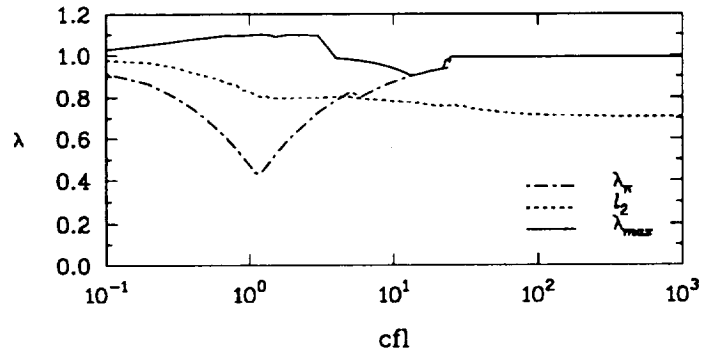
Fig. 2.3: 1−D Euler Equations using Steger−Warming schemes, second−order both sides. (a)−(d) Convergence Characteristics.



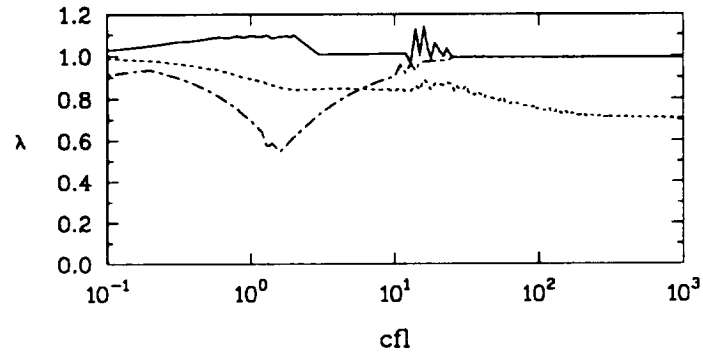
(a) Pressure distribution; Supersonic Case



(b) Pressure distribution; Transonic Case

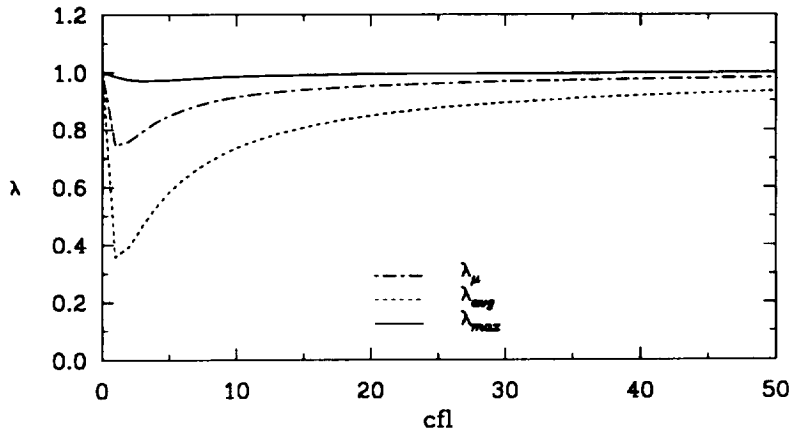


(c) Eigenvalues; Supersonic Case

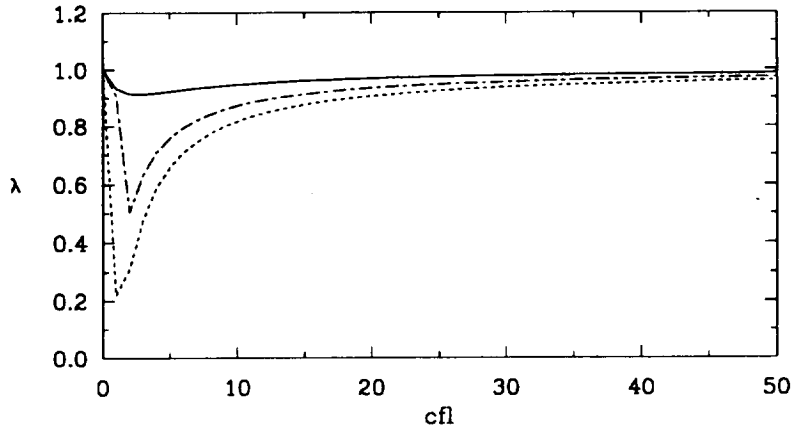


(d) Eigenvalues; Transonic Case

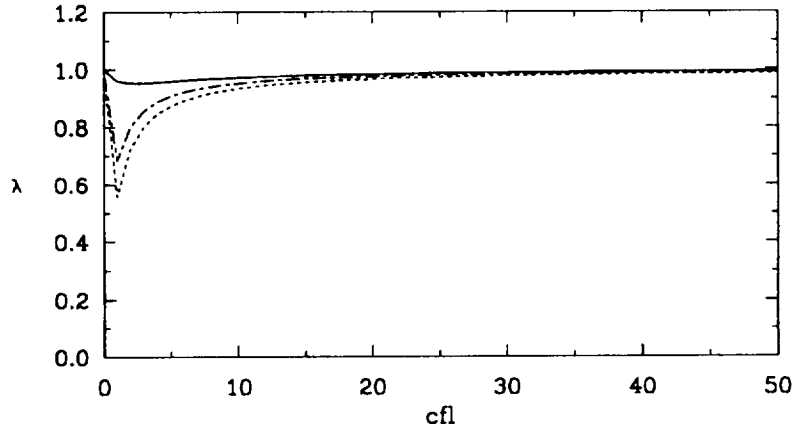
Fig. 2.4: Local mode analysis of Quasi−1−D Euler Equations M_{inlet}=1.5 (a)−(b) Pressure solutions (c)−(d) Convergence Characteristics.



(a) κ_2=0, κ_4=0.4



(b) κ_2=2, κ_4=3



(c) κ_2=3, κ_4=2

Fig. 2.5: 3−D Euler Equations using LU central schemes
(a)−(c) Convergence Characteristics

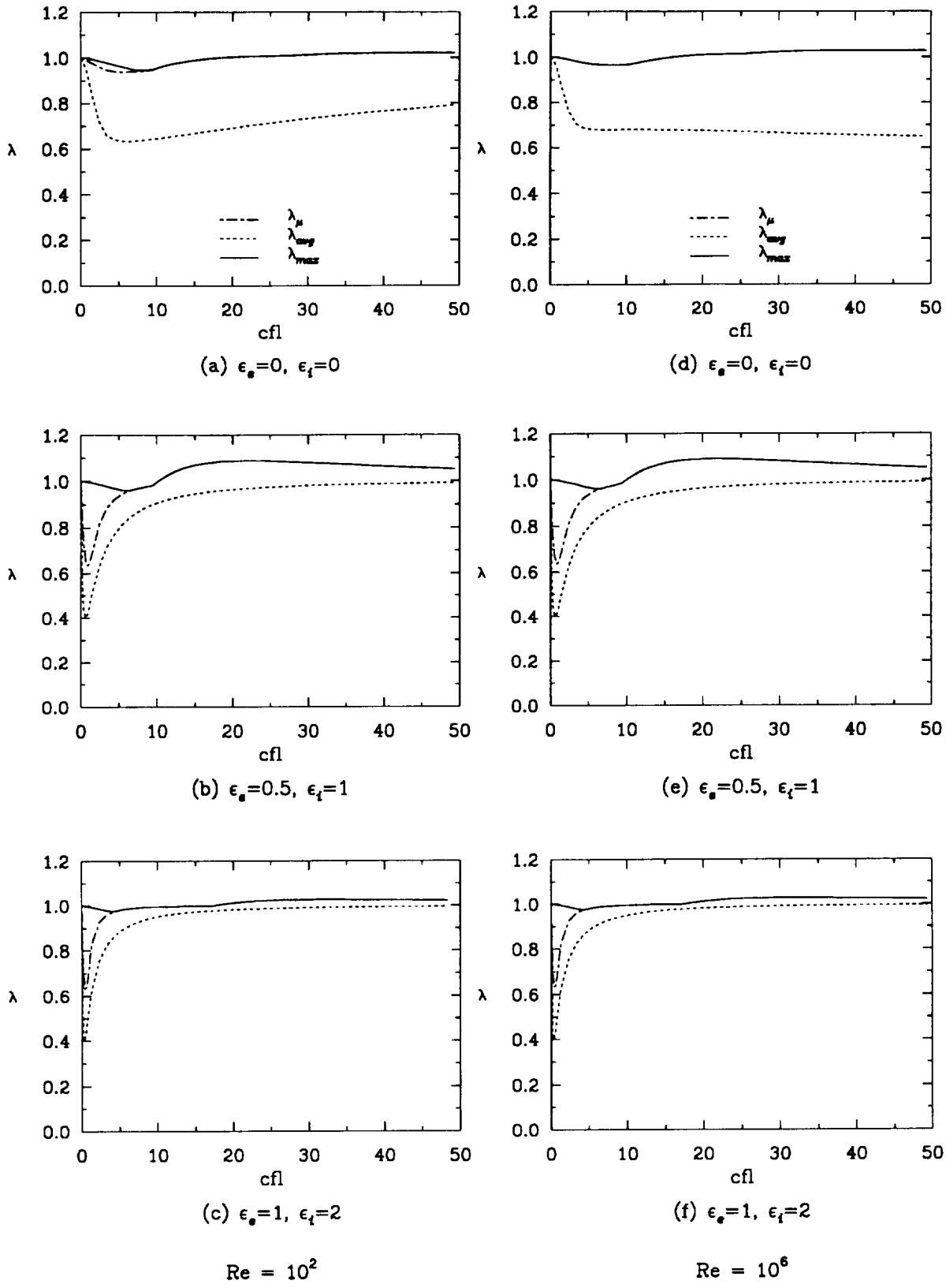


Fig. 2.6: 3-D Navier-Stokes Equations using ADI central schemes
(a)-(f) Convergence Characteristics

2.4 Concluding Remarks

The stability of some approximate factorization schemes for the solution of the 3–D Euler equations and Navier–Stokes equations have been studied. For the Euler equations, the Steger and Warming, and van Leer flux–vector splittings were used with three different upwind factorizations namely: spatial, eigenvalue and combination factorizations. For both flux–vector splittings, the eigenvalue and combination factorizations are unconditionally stable, but the spatial factorization is only conditionally stable for CFL numbers below five for the Steger and Warming scheme, and 14 for the van Leer scheme. Moreover, the amplification factor (λ_{max}) is minimized for the Steger and Warming scheme at CFL numbers of three, seven, and eight respectively, and for the van Leer scheme at seven, four, and seven, for spatial, eigenvalue and combination factorizations, respectively. Each of the approximate factorization methods has good smoothing properties for the van Leer flux–vector splitting, while for the Steger and Warming splitting, the smoothing factors are comparatively worse. Therefore, the van Leer splitting will be preferable for multigrid implementation. The Euler equations have also been analyzed for stability using the LU approximate factorization with central differences and various levels of artificial dissipation. It was found to be unconditionally stable in all dimensions with the maximum eigenvalue minimized at a CFL number of about three. Contrary to the conclusion drawn by Jespersen and Pulliam (1983) that the use of approximate Jacobians places restriction on the stability, it is shown, after careful investigation, that if they are used on both the implicit and the explicit sides, the stability results are comparable to the case where the exact Jacobians are used. The von–Neumann analysis method was also employed in performing local mode analysis for actual (supersonic and transonic) flow fields of a quasi 1–D problem to show the suitability of using uniform flow field in the stability analysis. Stability results for the 3–D Euler and Navier–Stokes equations solved with the Beam and Warming (ADI) central scheme with various levels of artificial dissipation (and at different Reynolds number for the latter) have

been presented. It was observed that the stability is not significantly affected by Reynolds numbers and that addition of dissipation reduces the amplification factor and the smoothing factor at lower CFL numbers.

Chapter 3
BI–GRID STABILITY ANALYSIS

The objective of this chapter is to present a procedure for utilizing the bi–grid amplification factor as a more effective tool for predicting practical multigrid performance in a range of numerical methods. Bi–grid analysis, based on the von–Neumann type method, is first presented for the 1–D convection and diffusion model problems, and the linearized Burger's equation. Numerical results from practical multigrid solution of these problems are compared to both predictions from bi–grid analysis and smoothing factors derived from the more usual single grid analysis. Both analyses and practical computations are based on the following different time–stepping methods: the Euler forward explicit scheme, the Runge–Kutta multistage scheme, a fully implicit scheme, and the semi–implicit scheme. The influence of the Peclet number on the convergence characteristics of the different schemes is investigated using the Burger's equation. Finally, for more practical situations, multigrid performance of various approximate factorizations for the 3–D Euler and Navier–Stokes equations are examined using the bi–grid stability analysis. For the Euler equations, bi–grid analysis is presented for three upwind difference–based factorizations and several central difference–based factorizations. In the upwind factorizations, both the flux–vector splitting methods of Steger–Warming and van Leer are considered. The central–differenced schemes include the Lower and Upper (LU) and ADI factorizations. The time–stepping algorithm for the Navier–Stokes equations is based on the Beam–Warming central difference scheme only. Finally, effects of grid aspect ratio and flow skewness are examined.

3.1 Bi–grid Analysis

Consider a given differential problem which can be written as

$$L\{u(x)\} = f(x) , \qquad \text{for } x \ \ in \ \ \Omega \tag{3.1}$$

where L is a linear operator. A typical 2–level multigrid cycle solution to this problem will involve the following steps:

(1) pre–relaxation on a fine grid using any technique S_1, ν^1 times

(2) computation of the defect R

(3) restriction of the defect to the coarser grid

(4) exact solution of the error equation on the coarse grid

(5) prolongation of the error onto and the correction on the fine grid

(6) post–relaxation on the fine grid using any technique S_2, ν^2 times

These can be represented for any intermediate solution *w*, by using the usual operators as follows:

$$\begin{aligned}
&(1) \quad w^{n+\frac{1}{2}} = S_1^{\nu^1} w^n \\
&(2) \quad R = f - L_h w^{n+\frac{1}{2}} \\
&(3) \quad I_h^H R \\
&(4) \quad v_H = L_H^{-1}(I_h^H R) \\
&(5) \quad I_H^h v_H + w^{n+\frac{1}{2}} \\
&(6) \quad w^{n+1} = S_2^{\nu^2}(I_H^h v_H + w^{n+\frac{1}{2}})
\end{aligned} \tag{3.2}$$

Combining these steps, we can write

$$w^{n+1} = S_2^{\nu^2}[I_H^h L_H^{-1} I_h^H (f - L_h S_1^{\nu^1} w^n) + S_1^{\nu^1} w^n] \tag{3.3}$$

The steady–state solution (u) is not changed by the coarse grid correction scheme; thus

$$u^{n+1} = S_2^{\nu^2}[I_H^h L_H^{-1} I_h^H (f - L_h S_1^{\nu^1} u^n) + S_1^{\nu^1} u^n] \tag{3.4}$$

Subtracting (2) from (1) and noting that $e^{n+1} = u^{n+1} - w^{n+1}$ gives

$$
\begin{aligned}
e^{n+1} &= S_2^{\nu^2}(\mathrm{I} - I_H^h L_H^{-1} I_h^H L_h) S_1^{\nu^1} e^n \\
&= S_2^{\nu^2}\, K\, S_1^{\nu^1} e^n \\
&= M e^n
\end{aligned}
\tag{3.5}
$$

where $K = \mathrm{I} - I_H^h L_H^{-1} I_h^H L_h$

$$
M = S_2^{\nu^2}(\mathrm{I} - I_H^h L_H^{-1} I_h^H L_h) S_1^{\nu^1} \tag{3.6}
$$

M is the bi–grid amplification matrix and K is the coarse grid correction matrix. It can be shown (Stuben and Trottenberg, 1982) that when linear operators are used for the restriction, I_h^H, and the prolongation, I_H^h, transfer processes, the coarse grid correction matrix is not a convergent iteration matrix. i.e.,

$$
\varrho(K) = \varrho(\mathrm{I} - I_H^h L_H^{-1} I_h^H L_h) \geq 1 \tag{3.7}
$$

Hence, the fine grid smoothing steps S_1, and S_2 are important for a convergent scheme. The spectral radius of the bi–grid amplification matrix (λ_{max_bg}) and its l_2 norm can be used to predict the performance of a multigrid method. While the spectral radius measures the asymptotic convergence rate of the multigrid method, the l_2 norm measures the actual error reduction per iteration. λ_{max_bg} is defined as follows:

$$
\lambda_{max\_bg} = \max\{\varrho[\hat{M}(\Theta)]\} \tag{3.8}
$$

$\hat{M}(\Theta)$ is the Fourier representation of the matrix M. A brief comment about Θ will be in order. Due to the aliasing process, low–frequency modes will couple with the coarse grid Fourier modes and, thus, for any $\Theta^1 = \{\theta_x, \theta_y, \theta_z\}$ such that $-\pi/2 \leq \theta_x, \theta_y, \theta_z \leq \pi/2$, there exists a corresponding set of harmonics up to an integer multiple of 2π. For 1–D, 2–D and 3–D problems, we define Θ as the following set :

$$
\begin{aligned}
&\text{1–D} \qquad \Theta = \{(\theta_x), (\theta_x \pm \pi)\} \\
&\text{2–D} \qquad \Theta = \{(\theta_x, \theta_y), (\theta_x, \theta_y \pm \pi), (\theta_x \pm \pi, \theta_y), (\theta_x \pm \pi, \theta_y \pm \pi)\} \\
&\text{3–D} \qquad \Theta = \{(\theta_x, \theta_y, \theta_z), (\theta_x, \theta_y, \theta_z \pm \pi), (\theta_x, \theta_y \pm \pi, \theta_z), (\theta_x, \theta_y \pm \pi, \theta_z \pm \pi), \\
&\qquad\qquad (\theta_x \pm \pi, \theta_y, \theta_z), (\theta_x \pm \pi, \theta_y, \theta_z \pm \pi), (\theta_x \pm \pi, \theta_y \pm \pi, \theta_z), \\
&\qquad\qquad (\theta_x \pm \pi, \theta_y \pm \pi, \theta_z \pm \pi)\}
\end{aligned} \tag{3.9}
$$

Or more generally,

$$
\text{d–D} \qquad \Theta = \{\Theta^1, \Theta^2, \Theta^3, \ldots, \Theta^{2^d}\} \tag{3.10}
$$

(where d is the dimensionality of the space, and $\Theta^1, \Theta^2, \ldots . \Theta^{2^d}$ are permuted in a similar manner with the $\pm$ signs chosen such that the harmonics lie in the high–frequency range). Hence, based on the Θ components and on the number of degrees of freedom of the problem, q, $\hat{M}(\Theta)$ is a $2^d q \; x \; 2^d q$ matrix. Thus, it is a 2x2 matrix for a 1–D scalar problem and 40x40 matrix for the Euler or Navier–Stokes equations in 3–D. The Fourier representation for the corresponding operators viz.: smoothing factor, fine grid problem, interpolation, restriction and the coarse grid problem can be constructed as follows (Brandt, 1991):

$$
\begin{aligned}
&\hat{S} = (\hat{S}_2^{\nu_2}, \; \hat{S}_1^{\nu_1}) = diag\left[\hat{S}(\Theta^1), \; \hat{S}(\Theta^2), \ldots . \; \hat{S}(\Theta^{2^d})\right] && 2^d q \; X \; 2^d q \\
&\hat{L}_h = diag\left[\hat{L}(\Theta^1), \; \hat{L}(\Theta^2), \ldots .. \; \hat{L}(\Theta^{2^d})\right] && 2^d q \; X \; 2^d q \\
&\hat{I}_H^h = \left[\hat{I}_H^h(\Theta^1), \; \hat{I}_H^h(\Theta^2), \ldots\ldots . \; \hat{I}_H^h(\Theta^{2^d})\right] && 2^d q \; X \; q \\
&\hat{I}_h^H = \left[\hat{I}_h^H(\Theta^1), \; \hat{I}_h^H(\Theta^2), \ldots . \ldots . \; \hat{I}_h^H(\Theta^{2^d})\right] && q \; X \; 2^d q \\
&\hat{L}_H = \hat{L}(2\Theta^1) && q \; X \; q
\end{aligned} \tag{3.11}
$$

The difference operator, $\hat{L}_H(2\Theta^1)$, on the coarse grid is only qxq since the coarse grid problem is solved exactly.

$\hat{S}$ and $\hat{L}$ depend on the choice of the smoother and the governing equations, respectively. The transfer processes, however, are less problem–dependent. Following Brandt (1991), the Fourier symbol of the prolongation operator based on an I^{th}–order polynomial is given by

$$\hat{I}_H^h(\Theta^m)_{kl} = \delta_{kl} \prod_{i=1}^{d} \psi_I(\cos\Theta_i^m) \quad m = 1, 2^d \tag{3.12}$$

where $\psi_2(\xi) = (1 + \xi)/2$, $\psi_4(\xi) = (2 + 3\xi - \xi^2)/4$, etc. are the 2nd and 4th order interpolation functions, and δ_{kl} is the Kronecker delta. We restricted our analysis to the 2nd order case, since it is more commonly used. The restriction operator is expressed as

$$2^d \hat{I}_h^H(\Theta^m) = [\hat{I}_H^h(\Theta^m)]^{T*} \tag{3.13}$$

$T*$ in the above equation represents the conjugate transpose. The restriction operator is often the adjoint of the prolongation operator in practice. In this study, the corresponding full weighting is used for the restriction operation for the Euler and Navier–Stokes equations, while simple injection is employed for the model problems. In the latter case, the Fourier symbol for the restriction operator is simply unity. A description of how the Fourier representation $\hat{M}(\Theta)$ can be constructed is given later for certain problems.

3.2 Model Equations

The model equations used in the present study are the conservation equations for the convection of a scalar, the diffusion of a scalar, and the linearized Burger's equation which is essentially a convection–diffusion equation. Each of these equations is integrated in time using (i) the Euler forward–explicit scheme, (ii) a Runge–Kutta multistage scheme, (iii) a fully implicit scheme and (iv) a semi–implicit scheme.

The model equations for convection, diffusion, and the linear Burger's equation can be expressed as:

	convection:	$u_t + cu_x = 0$
	diffusion:	$u_t - \nu u_{xx} = 0$
(convection–diffusion)	Burger's:	$u_t^* + u_o u_x^* = \nu u_{xx}^*$

(3.14)

In the Burger's equation, $u_o = constant$ is assumed in our analysis. Thus, it can be put in the following non–dimensional form:

$$u_t + u_x = \frac{1}{Pe} u_{xx} \tag{3.15}$$

where Pe in the above equation is the Peclet number defined as follows:

$$Pe = \frac{u_o D}{\nu} \tag{3.16}$$

(D is an appropriate length scale)

(i) Euler forward–explicit scheme

The Euler explicit method can be applied to the above equations to yield the following general discrete form:

$$u_i^{n+1} = u_i^n - \Delta t R^n \tag{3.17}$$

where R^n represents the residual expressed as follows:

$$\begin{aligned} &\text{convection:} && R^n = \frac{c}{\Delta x}(u_i^n - u_{i-1}^n) \\ &\text{diffusion:} && R^n = -\frac{\nu}{\Delta x^2}(u_{i+1}^n - 2u_i^n + u_{i-1}^n) \\ &\text{Burger's:} && R^n = \frac{1}{\Delta x}(u_i^n - u_{i-1}^n) - \frac{1}{\Delta x^2 Pe}(u_{i+1}^n - 2u_i^n + u_{i-1}^n) \end{aligned} \tag{3.18}$$

Spatial discretization in the above formulations is based on first–order upwind differences for convection, second–order central differences for diffusion, and the corresponding combination in the Burger's equation. First–order upwind differencing of the convective flux introduces inaccuracy due to too much numerical diffusion which may be of the same order of the natural diffusion in the Burger's equation. If second–order central differencing is used for the convective flux, a second–order accurate scheme can be obtained, but with severe limitations on the Peclet number due to dispersion errors. Although the addition of artificial viscosity could dampen the high–frequency oscillations at high Peclet numbers, it is highly problem dependent. A better approach to achieve a second–order accuracy while

sustaining a smooth solution at the vicinity of a shock or high gradients is to discretize the convective flux using higher–order upwind schemes, preferably in conjunction with some type of flux limiter. Hence, with a third–order discretization of the convective flux, a second–order accurate scheme for the Burger's equation can be obtained with R^n given by

$$R^n = \frac{1}{2\Delta x}(u_{i+1}^n - u_{i-1}^n) - \frac{1}{6\Delta x}(u_{i+1}^n - 3u_i^n + 3u_{i-1}^n - u_{i-2}^n) - \frac{1}{\Delta x^2 Pe}(u_{i+1}^n - 2u_i^n + u_{i-1}^n) \tag{3.19}$$

(ii) Runge–Kutta Multistage scheme

With each of the above schemes integrated in time using the Euler forward explicit method, the time step was limited to a small range by stability considerations, thus making it inefficient for steady–state computations. A Runge–Kutta (RK) method was introduced by Jameson *et. al.* (1981) to permit larger time steps to be taken. For an m–stage scheme, the time integration can be written as follows:

$$\begin{aligned} u_i^o &= u_i^n \\ u_i^k &= u_i^o - \alpha_k \Delta t R^{k-1} \qquad k = 1, m \\ u_i^{n+1} &= u_i^m \end{aligned} \tag{3.20}$$

Note that with $m = 1$, the RK scheme reduces to the Euler forward explicit scheme and hence is sometime called RK1. Coefficients α_k are optimized such that larger time steps can be used for faster convergence.

Three different sets of coefficients for a 4–stage Runge–Kutta scheme are investigated in this study, in line with the earlier work of Morano (1992). These are the standard coefficients (RK4–S, $\alpha_1 = .25, \alpha_2 = .3333, \alpha_3 = .5, \alpha_4 = 1$), and the optimized coefficients of Lallemand (RK4–L, $\alpha_1 = .11, \alpha_2 = .2766, \alpha_3 = .5, \alpha_4 = 1$) and van Leer (RK4–VL, $\alpha_1 = .0833, \alpha_2 = .2069, \alpha_3 = .4265, \alpha_4 = 1$).

(iii) Implicit scheme

An implicit time integration scheme in delta form can easily be formulated for each of our model problems. For example, the corresponding implicit formulation for the Burger's equation with first–order accuracy is written as follows:

$$\left[-\beta\left(\frac{\Delta t}{\Delta x}+\frac{\Delta t}{Pe\Delta x^2}\right)\right]\Delta u_{i-1}^n+\left[1+\beta\left(\frac{\Delta t}{\Delta x}+\frac{2\Delta t}{Pe\Delta x^2}\right)\right]\Delta u_i^n$$
$$-\beta\left(\frac{\Delta t}{Pe\Delta x^2}\right)\Delta u_{i+1}^n=-\Delta t R^n \qquad (3.21)$$

$$R^n=\frac{1}{\Delta x}\left(u_i^n-u_{i-1}^n\right)-\frac{1}{\Delta x^2 Pe}\left(u_{i+1}^n-2u_i^n+u_{i-1}^n\right)$$
$$\Delta u_i^n=u_i^{n+1}-u_i^n$$

The quantity β in the preceding formulation is called the implicitness factor. $\beta = 1.0$ gives a fully implicit scheme.

(iv) Semi–implicit scheme

If $\beta = 0.5$ in equation (3.21) above we have a semi–implicit scheme. This reduces to the Crank–Nicolson scheme if the overall spatial differencing is second–order accurate.

3.2.1 Fourier Symbols

For illustration, the bi–grid amplification matrix $\hat{M}(\Theta)$ is constructed for the convection problem using the Euler–forward explicit scheme for the relaxation.

Consider the discrete form of the operator L and let the step–by–step solution be characterized by Fourier modes (with periodic boundary conditions) as

$$u^n = U_o\lambda^n e^{\theta_x Ii} \qquad (3.22)$$

Then each of the operators that forms the matrix $\hat{M}(\Theta)$ becomes:

$$
\begin{aligned}
\hat{S}(\Theta^m) &= \left(1 - \frac{c\Delta t}{\Delta x}\right) + \frac{c\Delta t}{\Delta x}[\cos(\Theta^m) - I\sin(\Theta^m)] \\
\hat{L}_h(\Theta^m) &= -\frac{1}{\Delta x}[1 - \cos(\Theta^m) + I\sin(\Theta^m)] \\
\hat{I}_H^h(\Theta^m) &= \frac{1}{2}[1 + \cos(\Theta^m)] \qquad\qquad m = 1, 2 \\
\hat{I}_h^H(\Theta^m) &= 1 \qquad \text{for injection} \\
\hat{L}_H &= -\frac{1}{2\Delta x}\left[1 - \cos(2\Theta^1) + I\sin(2\Theta^1)\right]
\end{aligned}
\tag{3.23}
$$

where $\Theta^1 = \theta_x$ and $\Theta^2 = \theta_x + \pi$

Thus, from Eq. (3.8), $\hat{M}(\Theta)$ can be expressed as:

$$
\begin{aligned}
\hat{M}(\Theta) &= \begin{bmatrix} \hat{S}(\Theta^1) & 0 \\ 0 & \hat{S}(\Theta^2) \end{bmatrix}^{\nu_1} \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} \hat{S}(\Theta^1) & 0 \\ 0 & \hat{S}(\Theta^2) \end{bmatrix}^{\nu_2} \\
K_{11} &= 1 - \hat{I}_H^h(\Theta^1)\hat{I}_h^H(\Theta^1)\hat{L}_h(\Theta^1)/\hat{L}_H \\
K_{12} &= -\hat{I}_H^h(\Theta^1)\hat{I}_h^H(\Theta^2)\hat{L}_h(\Theta^2)/\hat{L}_H \\
K_{21} &= -\hat{I}_H^h(\Theta^2)\hat{I}_h^H(\Theta^1)\hat{L}_h(\Theta^1)/\hat{L}_H \\
K_{22} &= 1 - \hat{I}_H^h(\Theta^2)\hat{I}_h^H(\Theta^2)\hat{L}_h(\Theta^2)/\hat{L}_H
\end{aligned}
\tag{3.24}
$$

Note that $\hat{L}_H$ is evaluated only at the fundamental frequency $\{2\theta_x, 2\theta_y, 2\theta_z\}$, hence it is 1x1. The result obtained above is similar to that derived by Morano (1992), although our presentation is more general and is more easily extended to multi–dimensions.

3.2.2 Multigrid Implementation

A simple two–level multigrid (V cycle) method was implemented to test the relative accuracy of the bi–grid amplification factor and the smoothing factor in predicting multigrid performance. The two–level algorithm consists of the steps given in Sec. 2 and is recursively expressible as follows:

$$
\begin{aligned}
&\textit{Proc Multigrid} \quad (u^n, u^{n+1}, R^n, k) \\
&\quad \textit{(if } (k = 1) \\
&\qquad \textit{either} \quad u^{n+1} = L_H^{-1} R^n \\
&\qquad \textit{or} \quad\;\; u^{n+1} = S^{\infty} u^n \\
&\quad \textit{else} \\
&\qquad u^{n+1} \leftarrow S^1 u^n \\
&\qquad R^n \leftarrow I_h^H (R^n - L u^n) \\
&\qquad \textit{Multigrid} \;\; (0, u_H, R^n, k-1) \\
&\qquad u^{n+1} \leftarrow u^{n+1} + I_H^h u_H \\
&\quad \textit{endif)}
\end{aligned}
\tag{3.25}
$$

In the Eq. (3.25), L and S stand for the discrete operator and relaxation scheme corresponding to each of the model equations and numerical schemes discussed in previous sections. For this two–level V cycle multigrid implementation, the exact solution of the residual equation is employed. Only one pre–relaxation with no post–relaxation is performed on the fine grid.

3.2.3 Local Relaxation

Bi–grid analysis is exact for problems with periodic boundary conditions since it is based on the Fourier method. However, the asymptotic convergence rate for certain multigrid solutions deteriorates from the bi–grid prediction due to singularities such as a discontinuity in the material and /or solutions, and also due to the type and coefficients of the boundary conditions. Poor multigrid performance results since such singularities lead to too large a correction from the coarse grids in the localized region. To improve the performance of a multigrid solution, further relaxation can be performed on the fine grid in the region of the singularities after applying the coarse grid correction. This local relaxation is, in fact, an extra post–relaxation, but is confined to only certain nodal points and is carried out only a few number of times. The extra computational work is negligible if only a few partial sweeps is involved. The convection dominated problems subject to Dirichlet boundary conditions that are considered here undergo high changes in the gradients in order to satisfy the exit boundary conditions. Therefore, multigrid performance in these problems deviates from the results predicted by the bi–grid analysis. However, a few passes on the fine grid over the

boundary conditions and over the interior equation in some small neighborhood of the boundary (about three nodal points at the exit) is found sufficient to improve multigrid performance to the exact value predicted by bi–grid analysis.

3.2.4 Numerical Experiments

The bi–grid amplification factor (λ_{max_bg}), the smoothing factor (λ_{μ_sg}) and the practical asymptotic convergence rate (ϱ_{mg}) of the multigrid scheme were obtained for the following test problems:

(1) The convection problem with periodic boundary conditions, viz.:

$$u(0,t) = u(1,t) \quad ; \quad u(x,0) = \sin 2\pi x \tag{3.26}$$

(2) The convection problem with Dirichlet boundary conditions, viz.:

$$u(0,t) = 1 \quad , \quad u(1,t) = 0 \quad for\ t>0 \quad ; \quad u(x,0) = \sin 2\pi x \tag{3.27}$$

(3) The diffusion problem with similar Dirichlet boundary conditions as in (2) above

(4) The Burger's equation with similar Dirichlet boundary conditions as in (2) above.

The bi–grid amplification factor is obtained from Eq. (3.8) and the smoothing factor is obtained from the usual single grid amplification factor over the high frequency range $\pi/2 \le \Theta^1 \le \pi$ as $\lambda_{\mu\_sg} = \max\left\{\varrho[\hat{S}(\Theta^1)]\right\}$. In each case, sixteen Fourier modes are selected, and the associated eigenvalues are solved for using linear algebra routines such as found in the IMSL library. The asymptotic convergence rate of the multigrid experiments, on the other hand, is computed from

$$\varrho_{mg} = \left(\frac{\| R^{n2} \|}{\| R^{n1} \|}\right)^{\frac{1}{n2-n1}} \tag{3.28}$$

where $\| R^{n1} \|$ and $\| R^{n2} \|$ are the l_2 norm of the residuals at time levels $n1$ and $n2$, respectively.

The pseudotime Δt to advance the convection and the diffusion problems to steady state is computed from $CFL = \frac{\Delta t}{\Delta x}$ and $d = \frac{\Delta t}{Pe\Delta x^2}$, respectively. The CFL number is the Courant–Friedrichs–Lewy number, and d is the diffusion number. For the Burger's equation, Δt is computed from:

$$\Delta t = \min(\sigma\Delta x \ , \ \sigma\Delta x^2 Pe) \tag{3.29}$$

where σ is an appropriate parameter chosen to reduce to the diffusion number d at low Pe numbers and to reduced to the CFL number at high Pe numbers. This choice ensures that the appropriate time step is used in each flow regime. Δx is computed from D/20. Preliminary tests showed that the same results are obtained with 40 or 80 points.

The exact steady–state solution for the Burger's equation, subject to the boundary condition type discussed above, is given by

$$u = u(0,t)\left[\frac{1 - \exp[Pe\ (\frac{x}{D} - 1)]}{1 - \exp(-Pe)}\right] \tag{3.30}$$

It is valid for all range of Pe considered in this study.

3.2.5 Results for the Model Equations

Figures 3.1 and 3.2 show results of the analyses of the 1–D convection equation using the Euler forward explicit scheme. The model problem of Fig. 3.1 has periodic boundary conditions, whereas that of Fig. 3.2 has Dirichlet boundary conditions. The bi–grid analysis gives perfect prediction of practical multigrid performance in the former, whereas the smoothing factors from the single grid analysis are much too high. Both methods of analysis ignore boundary effects, so the same predictions are obtained in Figs. 3.1 and 3.2, and the analyses predictions are strictly correct only for problems with periodic boundary conditions. This is confirmed in Fig. 3.2(b) where the asymptotic multigrid convergence rate is now much worse than predicted by the bi–grid analysis. The reason for the degradation of the multigrid performance is the singularity which appears near the exit in Fig. 3.2(a). This

degradation in performance could be cured with a few local relaxation sweeps (Brandt and Yavneh, 1993), as shown in Fig. 3.2(c). Each sweep had marginal computational cost and five sweeps were sufficient to bring the multigrid performance for the Dirichlet problem in line with that with periodic boundary conditions and the prediction of the bi–grid analysis. Clearly the Euler forward explicit scheme does not have good convergence properties except for CFL numbers close to 0.5, and it is divergent for CFL numbers greater than 1. Better convergence properties are achieved with Runge–Kutta (RK) schemes. Three 4–stage RK schemes were analyzed, and the results are shown in Fig. 3.3 for the 1–D convection problem with periodic boundary conditions. With optimized coefficients Fig. 3.3(c), convergence could be obtained for CFL numbers up to three. Further, bi–grid amplification factors below 0.4 are obtained for the range of CFL numbers from 0.5 to 2.5. There is also perfect agreement between the results of the bi–grid analysis and the practical multigrid convergence rates. Similar multigrid results were obtained by Morano (1992). Figure 3.4 shows the result for the Dirichlet boundary conditions. In this case the multigrid convergence rates at higher CFL numbers are much better than predicted by either method. Clearly, the boundary effects are stronger with the RK scheme and there is no simple way to account for them in the analyses. Figure 3.5 shows results for a fully implicit scheme and for the semi–implicit Crank–Nicolson scheme, for the 1–D convection equation. Although both schemes are stable for the whole range of CFL numbers, the Crank–Nicolson scheme suffers from very poor convergence rate at high CFL numbers.

Results for the 1–D diffusion equation are presented in Figs. 3.6–3.8. Dirichlet boundary conditions are applied throughout, and the steady state–solution is shown in Fig. 3.6(a). In each case, the bi–grid analysis gives perfect agreement with the multigrid convergence rate whereas the smoothing rate obtained from the single grid analysis is consistently too optimistic. On the whole, the predicted convergence rates for each method are similar to the corresponding one obtained from the convection equation, if the diffusion number, d, is replaced by the CFL number in the latter. Clearly, if the goal is to achieve rapid convergence

to the steady state, the fully implicit scheme with high d or CFL number is the obvious choice.

The linearized Burger's equation represents a mixed convection–diffusion problem. The whole range of model type, from pure diffusion to pure convection, can be obtained simply by varying the Peclet number from a very small value to a very large value. Computed results for four values of Pe (10^{-4}, 20, 100, 10^{6}) are presented in Figs. 3.9–3.12, for the various discretization schemes considered here. The exact solution at the steady state is shown in Fig. 3.9(a), for the Dirichlet boundary conditions u(0,t) = 1, u(1,t) = 0. For high values of Pe, there is a singularity near x=1. As explained previously in Sec. 3.2.3 local relaxation is performed to reduce the adverse effect of this singularity on the overall multigrid convergence rate. The results for the first– and second–order Euler time explicit schemes are presented in Figs. 3.9 and 3.10. In each case the bi–grid analysis gives quite good prediction of the multigrid convergence rate. On the other hand, single–grid analysis gives too optimistic estimates at low Pe and too pessimistic estimates at high Pe. The second–order scheme shows much poorer convergence rates, especially at high Pe. The results for the fully–implicit and semi–implicit schemes are presented in Figs. 3.11 and 3.12. The superiority of the fully–implicit scheme is confirmed, especially for high Pe flows. For σ (or CFL number) greater than ten, it is close to a direct solver with $\lambda \rightarrow 0$. In these cases too, the bi–grid analysis agrees quite well with the practical multigrid convergence rate, except near $\sigma = 1$ in the semi–implicit scheme at high Pe. Because of the limited range of σ where the convergence rate is much less that unity, the semi–implicit Crank–Nicolson scheme is not a viable method for obtaining steady solutions for the model problem. If the main interest is rapid convergence to steady state, then the fully–implicit scheme at high values of σ (or CFL number) will be optimum.

3.3 Euler and Navier–Stokes Equations

Presently bi–grid stability analysis has been presented for typical explicit and implicit solution methods for model problems which range from the diffusion equation to the convection equation and including the convection–diffusion equation at different Peclet numbers. For large scale practical computations, interest is really in solving the system of Euler or Navier–Stokes equations. In the following sections, the bi–grid stability analysis of fully–implicit schemes for the Euler and Navier–Stokes equations are examined under various approximate factorization methods.

As formulated previously in Chap. 2, the coupled Euler and Navier–Stokes equations based on the different time–stepping approximate factorizations are

$$\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_x^+ A^-)\right]\left[\mathbf{I} + \Delta t(\delta_y^- B^+ + \delta_y^+ B^-)\right]\left[\mathbf{I} + \Delta t(\delta_z^- C^+ + \delta_z^+ C^-)\right]\Delta Q = -\Delta t R^n \tag{3.31}$$

$$\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_y^- B^+ + \delta_z^- C^+)\right]\left[\mathbf{I} + \Delta t(\delta_x^+ A^- + \delta_y^+ B^- + \delta_z^+ C^-)\right]\Delta Q = -\Delta t R^n \tag{3.32}$$

$$\left[\mathbf{I} + \Delta t(\delta_x^- A^+ + \delta_x^+ A^- + \delta_z^- C^+)\right]\left[\mathbf{I} + \Delta t(\delta_y^- B^+ + \delta_y^+ B^- + \delta_z^+ C^-)\right]\Delta Q = -\Delta t R^n \tag{3.33}$$

where

$$R^n = \delta_x^- E^+ + \delta_x^+ E^- + \delta_y^- F^+ + \delta_y^+ F^- + \delta_z^- G^+ + \delta_z^+ G^- \tag{3.34}$$

$$\begin{aligned}&\left[\mathbf{I} + \Delta t(\delta_x^- A_1 + \delta_y^- B_1 + \delta_z^- C_1) + \varkappa_2 \Delta t(\delta_x^- + \delta_y^- + \delta_z^-)\right] \\ &\quad \mathrm{X}\ \left[\mathbf{I} + \Delta t(\delta_x^+ A_2 + \delta_y^+ B_2 + \delta_z^+ C_2) - \varkappa_2 \Delta t(\delta_x^+ + \delta_y^+ + \delta_z^+)\right]\Delta Q \\ &\quad = -\Delta t\left(\delta_x E + \delta_y F + \delta_z G\right) - \varkappa_4 \Delta t\left(\Delta x^3 \delta_{xxxx} + \Delta y^3 \delta_{yyyy} + \Delta z^3 \delta_{zzzz}\right) Q\end{aligned} \tag{3.35}$$

$$\begin{aligned}&\left[\mathbf{I} + \Delta t(\delta_x A - \delta_{xx} R - \varepsilon_i \Delta x \delta_{xx})\right]\left[\mathbf{I} + \Delta t(\delta_y B - \delta_{yy} S - \varepsilon_i \Delta y \delta_{yy})\right] \\ &\mathrm{X}\ \left[\mathbf{I} + \Delta t(\delta_z C - \delta_{zz} Y - \varepsilon_i \Delta z \delta_{zz})\right]\Delta Q \\ &= -\Delta t\left[A\delta_x - R\delta_{xx} - R_1\delta_{yx} - R_2\delta_{zx} + B\delta_y - S_1\delta_{xy} - S\delta_{yy} - S_2\delta_{zy}\right. \\ &\qquad \left. + C\delta_z - Y_1\delta_{xz} - Y_2\delta_{yz} - Y\delta_{zz} + \varepsilon_e(\Delta x^3 \delta_{xxxx} + \Delta y^3 \delta_{yyyy} + \Delta z^3 \delta_{zzzz})\right] Q\end{aligned} \tag{3.36}$$

Equations (3.31),(3.32) and (3.33) are the upwind schemes that are referred to as the spatial, eigenvalue and combination factorizations, respectively, in Chap. 2. The flux–vector splitting methods of Steger–Warming (1980) and van Leer (1982) are also assumed. Equation (3.35) is the Lower and Upper (LU) approximate factorization. Here, the fluxes devised by Jameson and Turkel (1981), viz.: $A_1 = (A + |A|)/2$ and $A_2 = (A - |A|)/2$, are used to achieve diagonal dominance. The operators δ^+ and δ^- denote forward and backward difference operators, respectively. The terms $\varkappa_2$ and $\varkappa_4$, and ε_i and ε_e are the artificial dissipation coefficients for the LU decomposition and the ADI schemes, respectively. Equation (3.36) is the Beam–Warming ADI scheme for the Navier–Stokes equations, which degenerate to the Euler equations when the viscous flux Jacobians $R,\ R_1,\ R_2,\ S,\ S_1,\ S_2,\ Y,\ Y_1,\ Y_2$ are set to zero.

3.3.1 Fourier Symbols

The bi–grid amplification matrix $\hat{M}(\Theta)$ is constructed from $M = S_2^{\nu_2}(\mathrm{I} - I_H^h L_H^{-1} I_h^H L_h) S_1^{\nu_1}$. For ease of presentation, the Euler equations alone are selected for illustration, with the ADI central scheme used as the smoother. In this case, viscous fluxes $R,\ R_1,\ R_2,\ S,\ S_1,\ S_2,\ Y,\ Y_1,\ Y_2$ are set to zero. The components operators of matrix $\hat{M}(\Theta)$ are expressed as follows:

(i) The fine/coarse grid Operator $\hat{L}$

The Euler equivalent form of Eq. (2.15) is

$$\frac{\partial Q}{\partial t} = -\left(\frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z}\right) + \text{dissipation} \tag{3.37}$$

where dissipation is added to damp oscillations. Thus, in quasi–linear form:

$$L(Q) = -\left(A\frac{\partial Q}{\partial x} + B\frac{\partial Q}{\partial y} + C\frac{\partial Q}{\partial z}\right) + \varepsilon_i\left(\Delta x\frac{\partial^2 Q}{\partial x^2} + \Delta y\frac{\partial^2 Q}{\partial y^2} + \Delta z\frac{\partial^2 Q}{\partial z^2}\right) - \varepsilon_e\left(\Delta x^3\frac{\partial^4 Q}{\partial x^4} + \Delta y^3\frac{\partial^4 Q}{\partial y^4} + \Delta z^3\frac{\partial^4 Q}{\partial z^4}\right) \tag{3.38}$$

Holding A, B, C locally constant and employing second–order central differencing, the Fourier symbol of the fine grid problem assuming equal mesh spacing in all directions becomes

$$\begin{aligned}\hat{L}_h(\Theta^m) = & -\frac{I}{\Delta x}\left[A\sin(\Theta_1^m) + B\sin(\Theta_2^m) + C\sin(\Theta_3^m)\right] \\ & + \frac{2\varepsilon_i}{\Delta x}\left[\cos(\Theta_1^m) + \cos(\Theta_2^m) + \cos(\Theta_3^m) - 3\right] \qquad \text{m=1,8} \\ & - \frac{16\varepsilon_e}{\Delta x}\left[\sin^4\left(\frac{\Theta_1^m}{2}\right) + \sin^4\left(\frac{\Theta_2^m}{2}\right) + \sin^4\left(\frac{\Theta_3^m}{2}\right)\right]\end{aligned} \tag{3.39}$$

Note that Θ_k^m represent the k^{th} element of the Θ^m component (see Eq. (3.9–3.11)).

For any arbitrary mode, Eq. (3.39) is a 40 X 40 matrix since each Jacobian is a 5x5 matrix and there are 8 harmonics including the fundamental mode. The coarse grid problem is assumed to be a version of the original problem on the fine grid and the coarse grid is formed simply by deleting every other fine grid point. Thus, the mesh size and Fourier modes are $\{2\Delta x, 2\Theta^1\}$ and its Fourier signature can be written as:

$$\begin{aligned}\hat{L}_H(2\Theta^1) = & -\frac{I}{2\Delta x}\left[A\sin(2\theta_x) + B\sin(2\theta_y) + C\sin(2\theta_z)\right] + \frac{\varepsilon_i}{\Delta x}\left[\cos(2\theta_x) + \cos(2\theta_y)\right. \\ & \left. + \cos(2\theta_z) - 3\right] - \frac{8\varepsilon_e}{\Delta x}\left(\sin^4\theta_x + \sin^4\theta_y + \sin^4\theta_z\right)\end{aligned} \tag{3.40}$$

In the above equation, only the fundamental mode, $\Theta^1 = \{\theta_x, \theta_y, \theta_z\}$, is employed since the coarse grid problem is assumed to be solved exactly. Hence, this is only a 5 X 5 matrix.

(ii) The relaxation Operator $\hat{S}$

Each of the equations (3.31)–(3.33), (3.35) and (3.36) can be expressed as

$$N\Delta Q^n = -L = -\Delta t R^n \tag{3.41}$$

von Neumann stability analysis is used on this system of linear equations by letting the step–by–step solution be characterized by

$$Q^n = U_o \lambda^n e^{Ii\theta_x} e^{Ij\theta_y} e^{Ik\theta_z} \tag{3.42}$$

where λ is the single grid amplification factor. Thus, Eq. (3.41) reduces to a complex generalized eigenvalue problem of the form

$$\hat{K}\mathbf{x} = \lambda \hat{N}\mathbf{x} \quad where \quad \hat{K} = \hat{N} - \hat{L} \tag{3.43}$$

The Fourier symbols of $\hat{N}$ and $\hat{L}$, for our particular example, can easily be shown to be

$$\hat{N}(\Theta^m) = \left[\mathbf{I} + \frac{\Delta t}{\Delta x}\left(AI\sin(\Theta_1^m) + 4\varepsilon_i \sin^2\frac{\Theta_1^m}{2}\right)\right]\left[\mathbf{I} + \frac{\Delta t}{\Delta y}\left(BI\sin(\Theta_2^m) + 4\varepsilon_i \sin^2\frac{\Theta_2^m}{2}\right)\right] \times \left[\mathbf{I} + \frac{\Delta t}{\Delta z}\left(CI\sin(\Theta_3^m) + 4\varepsilon_i \sin^2\frac{\Theta_3^m}{2}\right)\right] \tag{3.44}$$

$$\hat{L}(\Theta^m) = \frac{\Delta t}{\Delta x} I\left(A\sin(\Theta_1^m) + B\sin(\Theta_2^m) + C\sin(\Theta_3^m)\right) + \frac{16\Delta t \varepsilon_e}{\Delta x}\left(\sin^4\frac{\Theta_1^m}{2} + \sin^4\frac{\Theta_2^m}{2} + \sin^4\frac{\Theta_3^m}{2}\right) \tag{3.45}$$

The Fourier symbols corresponding to the other approximate factorizations are documented in Demuren and Ibraheem (1992). For each harmonic, Θ^m $(m = 1, 8)$, Eq. (3.43) is solved to give five eigenvalues from which the elements of $\hat{S}(\Theta)$ are constructed. For example, if the eigenvalues corresponding to the mode $\Theta^1 = \{\theta_x, \theta_y, \theta_z\}$ are $\Lambda = \{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\}$, then, from Eq. (3.11), $\hat{S}(\Theta^1) = \Lambda\mathbf{I}$. The effective fine grid smoothing operation is obtained by raising the smoothing matrices to the power of ν^1 and ν^2, the pre– and post–smoothing counts, respectively.

(iii) The Transfer Operators $\hat{I}_H^h$ and $\hat{I}_h^H$

For a second–order interpolation, the Fourier symbol of the prolongation operator, from Eq. (3.12), is expressed as

$$\hat{I}_H^h(\Theta^m) = \frac{1}{8}\left[1 + \cos(\Theta_1^m)\right]\left[1 + \cos(\Theta_2^m)\right]\left[1 + \cos(\Theta_3^m)\right] \tag{3.46}$$

The restriction operator, $\hat{I}_h^H$, is computed from this equation and Eq. (3.13) assuming full–weighing.

Based on the above operators, $\hat{M}(\Theta)$ is assembled from $M = S_2^{\nu^2}(\mathrm{I} - I_H^h L_H^{-1} I_h^H L_h) S_1^{\nu^1}$. A symbolic form is given in Appendix C. It is an 8x8 block matrix of which each elemental block is a 5x5 matrix.

3.3.2 Solution Procedure

The eigenvalues for the bi–grid matrix $\hat{M}(\Theta)$ are computed from Eq. (3.8) over fixed Fourier modes to obtain the amplification factor. Sixteen modes are selected, in the range $-\pi/2 \le \Theta^1 \le \pi/2$. The smoothing factor is also computed from the generalized eigenvalue problem (3.43) over only the high–frequency modes $\pi/4 \le |\Theta^1| < \pi/2$ as $\lambda_{\mu\_sg} = \max(|\lambda|)$. In each case, the eigenvalues are solved for using the linear algebra routines such as found in the IMSL library. Uniform flow is assumed with $M_\infty = 0.8$, zero yaw (α_y) and angle of attack (α_a), and $\gamma = 1.4$. Further, the grid spacing is assumed to be uniform in all directions. Effects of aspect ratio and flow skewness are also investigated. The time–step and Reynolds number are calculated from

$$\Delta t = \frac{CFL}{\left[\frac{|u|}{\Delta x} + \frac{|v|}{\Delta y} + \frac{|w|}{\Delta z} + c\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}}\right]} \tag{3.47}$$

$$\mathrm{Re} = \frac{\varrho |V| \left(\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}\right)}{\mu} \tag{3.48}$$

Some other pertinent definitions used are as follows:

$$|V| = \sqrt{u^2 + v^2 + w^2} \quad , \quad M_\infty = \frac{|V|}{a} \quad , \quad v = u\tan(\alpha_y) \quad , \quad w = u\tan(\alpha_a) \tag{3.49}$$

A batch file used to submit a typical 3–D test case is shown in Appendix D.

3.3.3 Convergence Rates

In previous sections, we have thoroughly assessed the capability of bi–grid analysis to predict more accurately the performance of multigrid methods using scalar model equations. In order to completely rely on its results to guide us in correctly implementing multigrid procedures in future chapters, it is equally very important to know how bi–grid will predict multigrid performance in complicated practical problems. Rather than implementing multigrid procedures for each of the schemes discussed above, as we have done for the scalar model problems, we base our comparisons on the actual multigrid solutions obtained by Anderson et. al. (1988) for the three upwind based factorizations using van–Leer flux–vector splitting. The multigrid solutions were obtained for the ONERA M6 wing at transonic conditions: a Mach number of 0.84, an angle of attack of 3.06^{o} and mesh size 97X17X17. From Figs. 3.13(d)–3.13(f) the prediction (from both the single grid analysis and bi–grid analysis) rates multigrid performance for these schemes in this order: spatial, combination and eigenvalue factorizations, which also agrees with the results of Anderson et al. However, they experimentally found that practical multigrid solutions required an optimal CFL number of about seven for each of the schemes. This is the exact result predicted by the present bi–grid analysis, and is much greater than the CFL of about three predicted by the single grid analysis. Anderson et al. also computed the convergence rates for the best scheme, namely spatial factorization, and the worst scheme, namely the eigenvalue factorization. Their results are compared with the values predicted by the bi–grid and smoothing factors in Table 3.1. From this table, the superiority of bi–grid analysis over single grid analysis is further demonstrated. Although the eigenvalue factorization has the worst multigrid convergence rate of 0.93, Anderson et al. found that it represents a good improvement over a corresponding single grid computation with a convergence rate of 0.98. This latter value also coincides with $\lambda_{\max}$ computed in Chap. 2 for this scheme.

Table 3.1: Convergence Characteristics of Transonic Flow on ONERA M8 Wing

Optimal	Spatial	Eigenvalue	Combination	Comment
$(CFL, \lambda)_{\mu - sg}$	3, 0.76	3, 0.80	4, 0.75	single grid analysis
$(CFL, \lambda)_{\max - bg}$	7, 0.89	7, 0.91	7, 0.89	bi–grid analysis
$(CFL, \varrho)_{mg}$	7, 0.90	7, 0.93	7, –	From Andersen et al. (1988)

3.3.4 Results for the Euler and Navier–Stokes Equations

Figure 3.13 shows the convergence results for the 3–D Euler equations using the upwind schemes. The computed values for the smoothing factor(λ_{μ_sg}) and bi–grid amplification factor ($\lambda_{\max\_bg}$) for the spatial, eigenvalue, and combination factorizations based on the Steger–Warming flux–vector splitting are shown in Figs. 3.13(a)–(c), respectively. Both factors predict instability for the spatial–split scheme, especially for a CFL number beyond five. In the eigenvalue and combination factorizations, better convergence characteristics are observed, although the smoothing factor's prediction is slightly more optimistic. For these two factorizations, bi–grid analysis predicts near instability at CFL number above 25, whereas the smoothing factor predicts unconditional stability for all CFL numbers. Figs 3.13(d)–(f) show predictions for multigrid performance of each factorization using the van Leer flux–vector splitting. Except for the spatial factorization, all the schemes are predicted unconditionally stable for all CFL numbers by both bi–grid and smoothing factors. The spatial factorization is stable only for CFL numbers below 12 and possesses better convergence characteristics at CFL number below 8 than the other two factorizations. From both analyses, i.e. from (λ_{μ_sg}) and ($\lambda_{\max\_bg}$), van Leer flux–vector splitting gives better convergence characteristics than the Steger–Warming method for multigrid procedures. It is observed that the present results of the smoothing factors for the van Leer method are similar to those presented by Anderson et. al. (1988), and Demuren and Ibraheem (1994).

Results for the 3–D Euler equations using the LU approximate factorization with central difference approximations and various levels of second– and fourth–order artificial viscosities, $\varkappa_2$ and $\varkappa_4$, are shown in Figs. 3.14(a)–(c). Without the addition of second–order dissipations, i.e. $\varkappa_2 = 0$, the coefficient $\varkappa_4 = 0.3$ yields the optimal results (see Fig. 3.14(a)). From Figs. 3.14(b) and 3.14(c), bi–grid and single grid analyses predict that an appropriate combination of $\varkappa_2$ and $\varkappa_4$ (especially when $\varkappa_4 \geq \varkappa_2$) can significantly improve the performance of the LU scheme when used as a relaxation scheme for multigrid. Also for

all levels of dissipation, the smoothing factors estimates are more optimistic than the bi–grid results, especially at lower CFL numbers.

The convergence characteristics for the 3–D Euler and Navier–Stokes equations for different levels of artificial dissipation and Reynolds numbers are shown in Figs. 3.14(d)–(f) and Fig. 3.15, using the Beam–Warming (ADI) central difference scheme as the base solution algorithm. With no dissipation added to the Euler equations (Fig. 3.14(d)), the bi–grid analysis predicts instability for all CFL numbers, while the smoothing factor predicts stability for CFL numbers below 15. From Figs. 3.14(e) and 3.14(f), optimal multigrid performance is predicted by the bi–grid analysis for dissipation levels of $\varepsilon_e = 0.5$ and $\varepsilon_i = 1.0$. These results are similar to those obtained for the Navier–Stokes equations at Re=10^6 (see Figs. 3.15(d)–3.15(f)). With Reynolds number of 100 and no dissipation, both bi–grid and smoothing factors predict stability for certain range of CFL numbers although the latter is more optimistic. Also at this Reynolds number, the optimal dissipation levels are $\varepsilon_e = 0.5$ and $\varepsilon_i = 1.0$.

All computations have been based on zero yaw and angle of attack, and also on uniform grid spacing in all directions. Sensitivities of convergence characteristics to flow skewness and aspect ratio are studied using the ADI central–difference scheme at Reynolds number of 100, and dissipation levels of $\varepsilon_e = 0.5$ and $\varepsilon_i = 1.0$. The results are shown in Figs. 3.16 and 3.17. Generally, convergence characteristics are improved with increases in yaw angle at zero angle of attack, although the range of stable CFL numbers becomes smaller (Figs. 3.16(a)–(c)). From Figs. 3.16(d)–(f), no significant difference is observed in the convergence results when the yaw and angle of attack are set equal to each other. However, from Fig. 3.17, the convergence characteristics become worse with increases in grid aspect ratio.

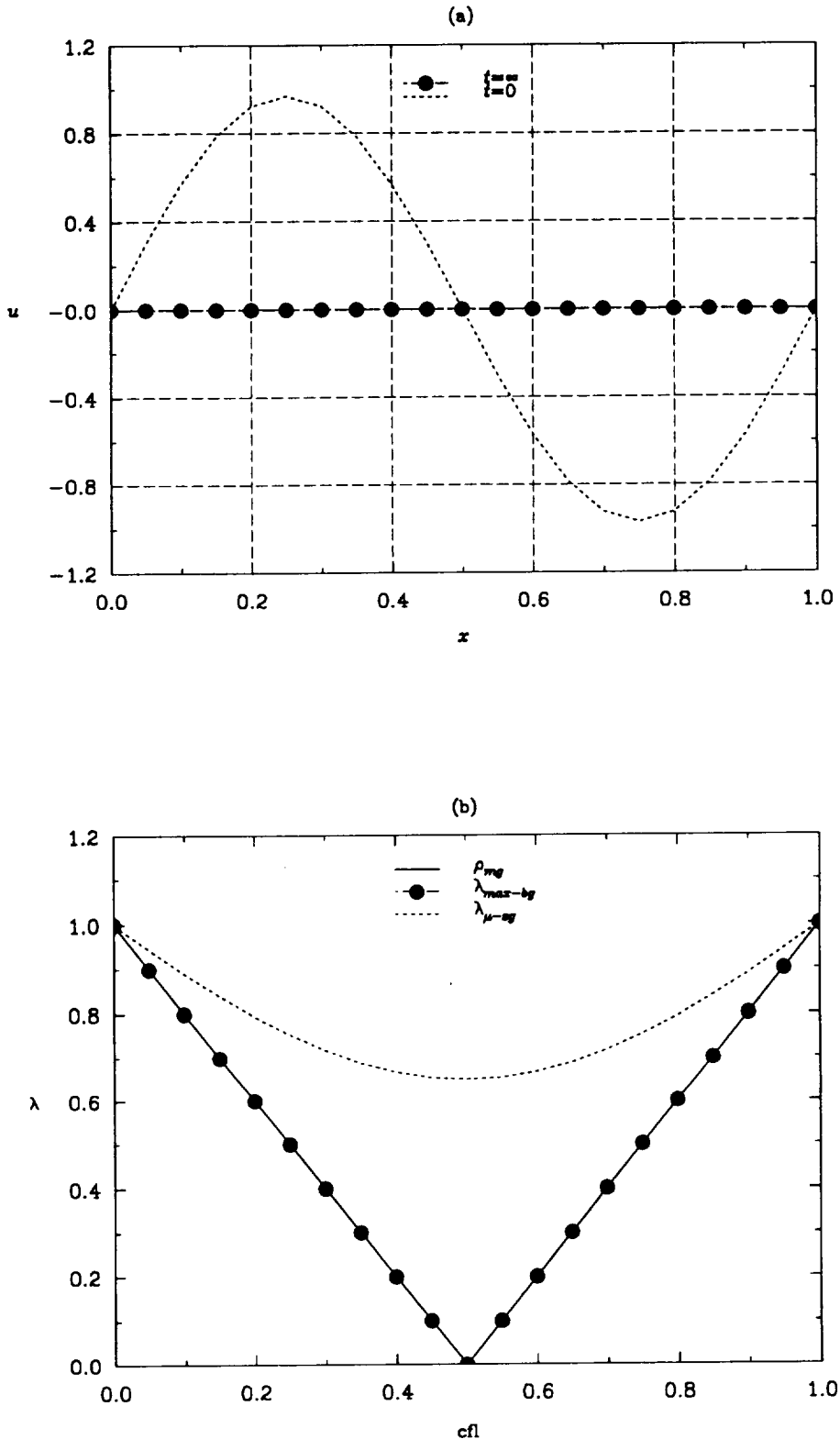
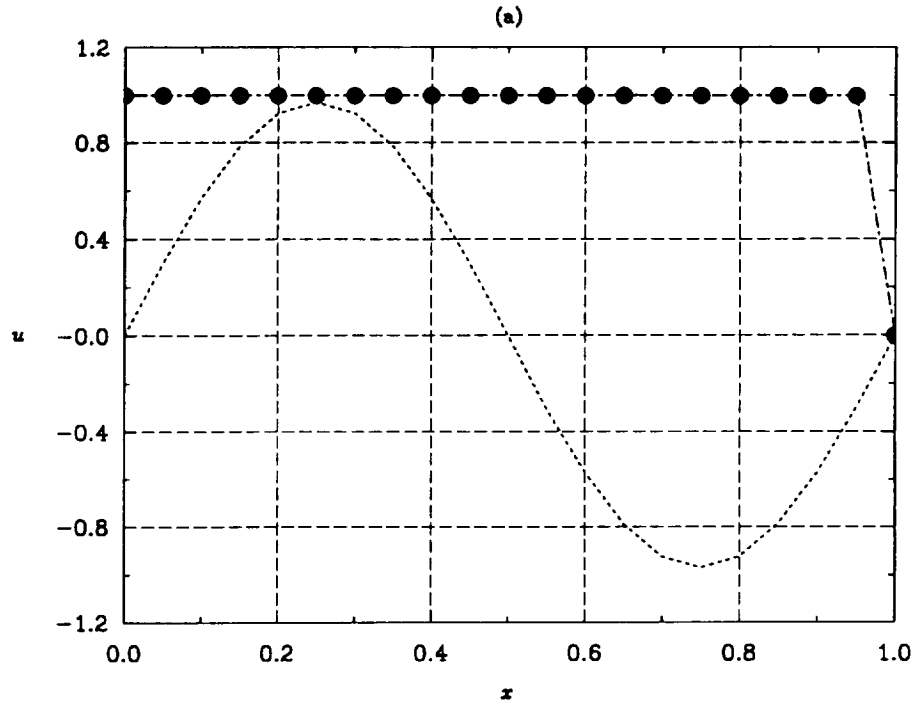


Fig. 3.1: 1–D Convection Equation (a) Steady solution (b) Convergence Characteristics (Euler forward explicit; Periodic B.C's).



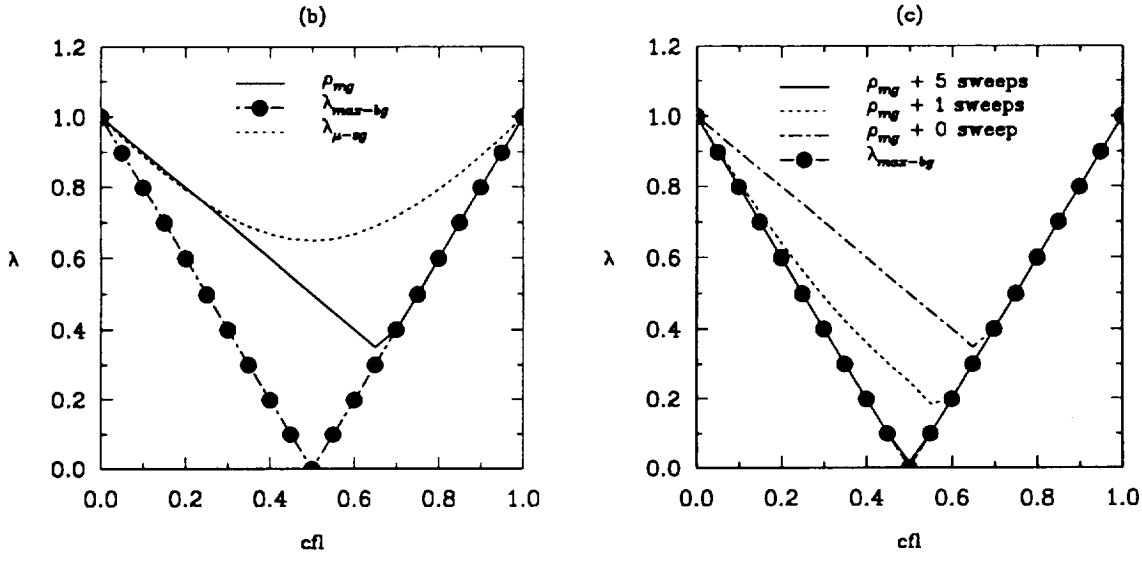


Fig. 3.2: 1−D Convection Equation (a) Steady solution (b) Convergence Characteristics without local relaxation (c) Convergence Characteristics with local relaxation(Euler forward explicit; Dirichlet B.C's).

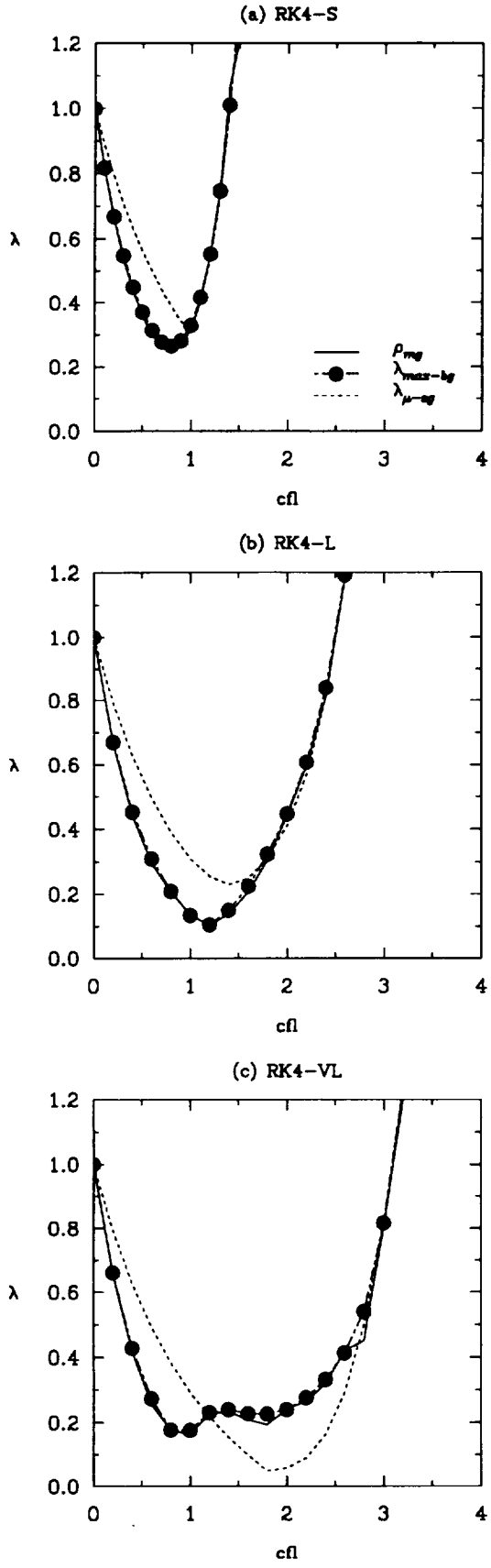


Fig. 3.3: Convergence Characteristics for 1−D Convection Equation (a)Standard (b) Lallemand (c) van Leer Coefficients (4−Stage Runge Kutta; Dirichlet B.C's)

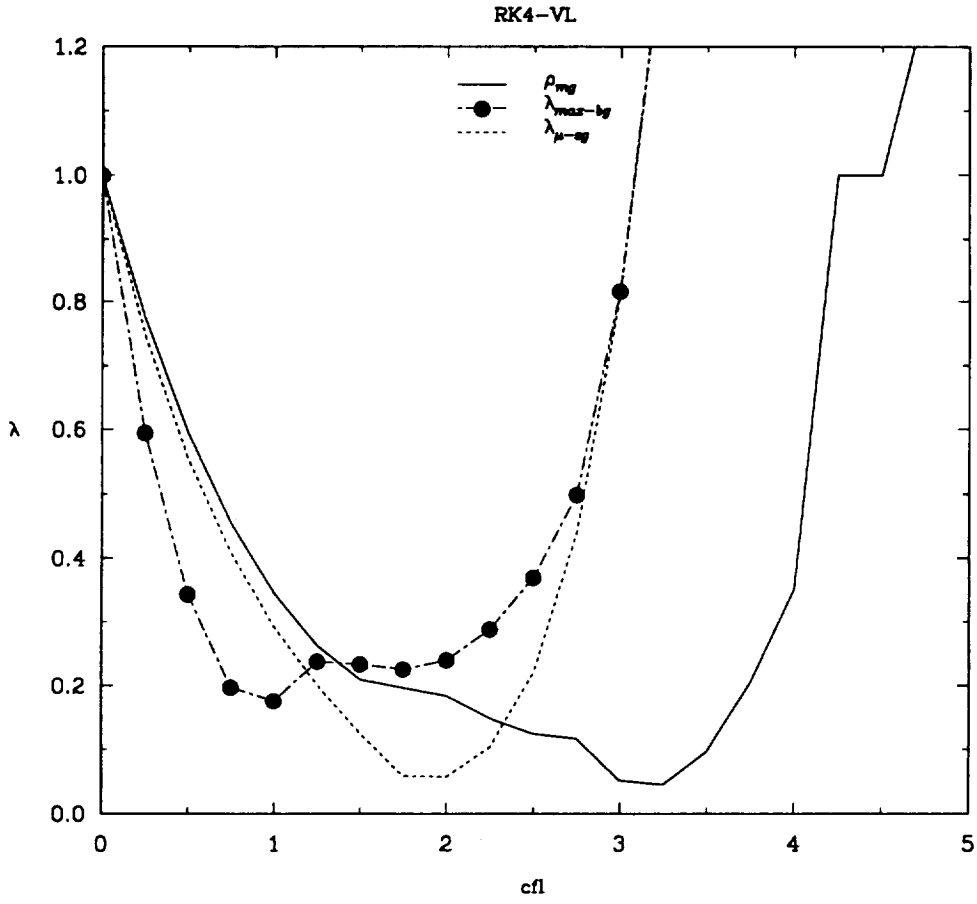


Fig. 3.4: Convergence Characteristics for 1−D Convection Equation (4−Stage Runge Kutta; Dirichlet B.C's; van Leer coefficients).

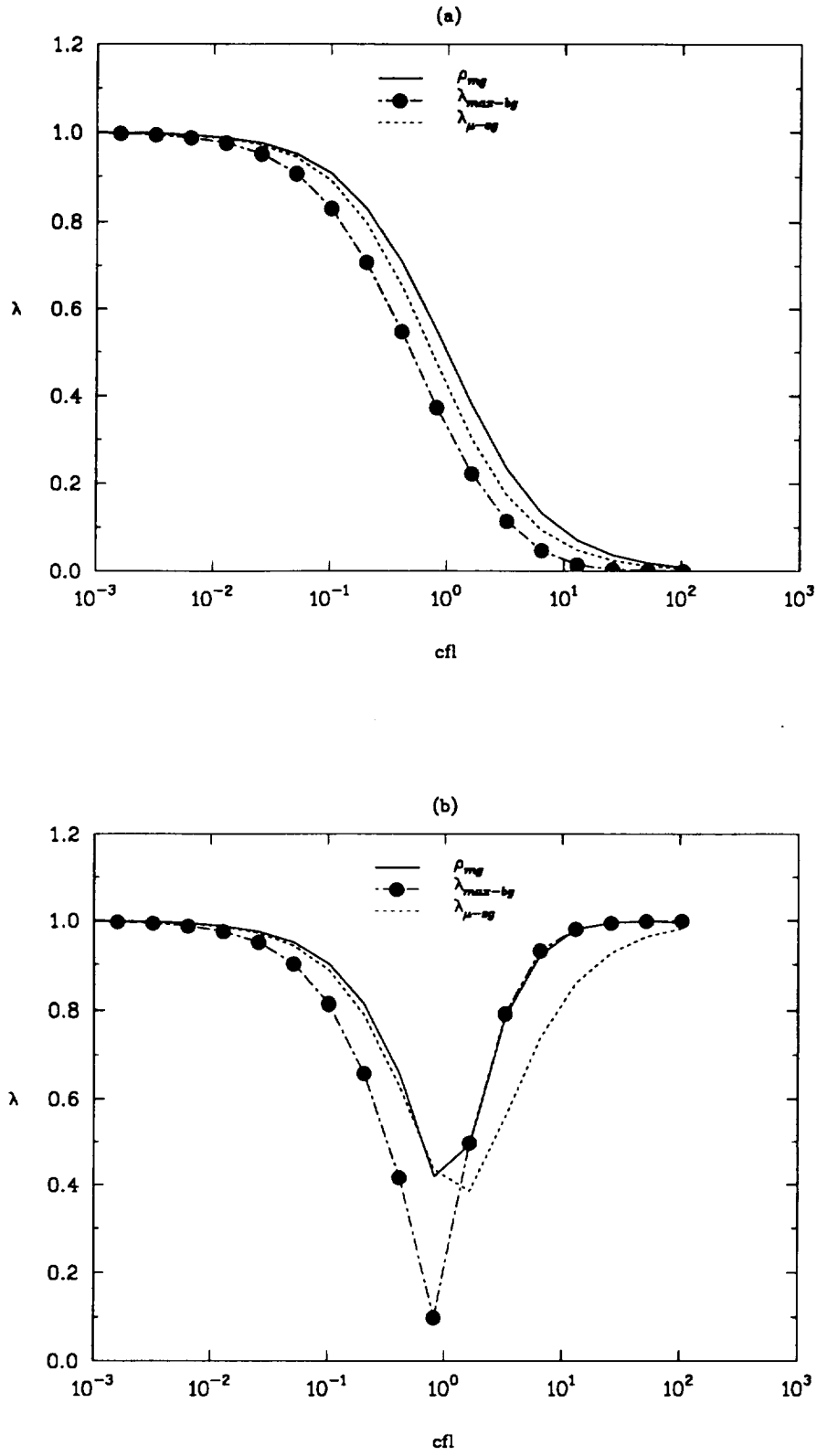


Fig. 3.5: Convergence Characteristics for 1−D Convection Equation (a) Implicit (b) Semi−implicit time integrations (Dirichlet B.C's).

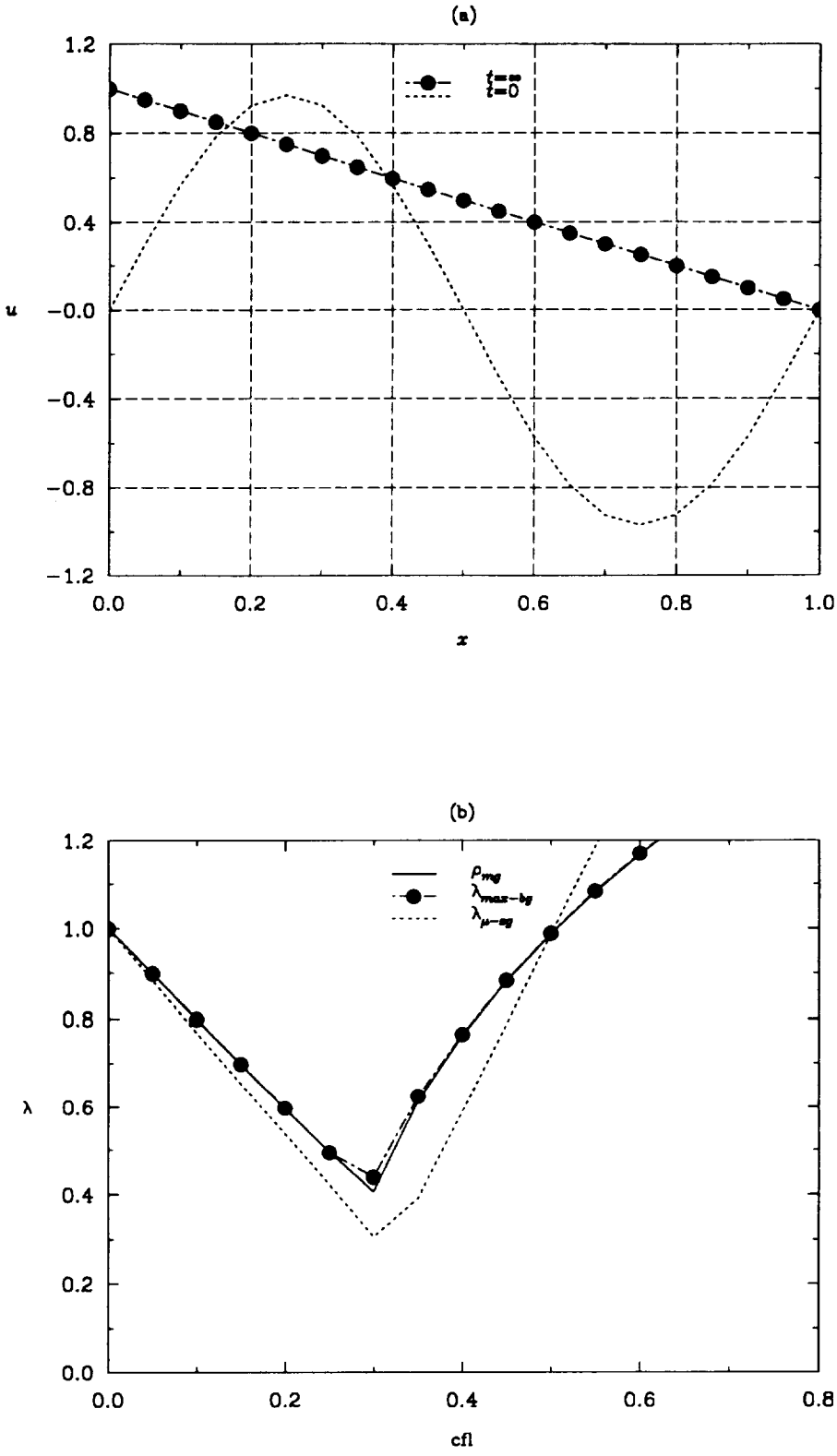


Fig. 3.6: 1−D Diffusion Equation (a) Steady solution (b) Convergence Characteristics (Euler forward explicit; Dirichlet B.C's).

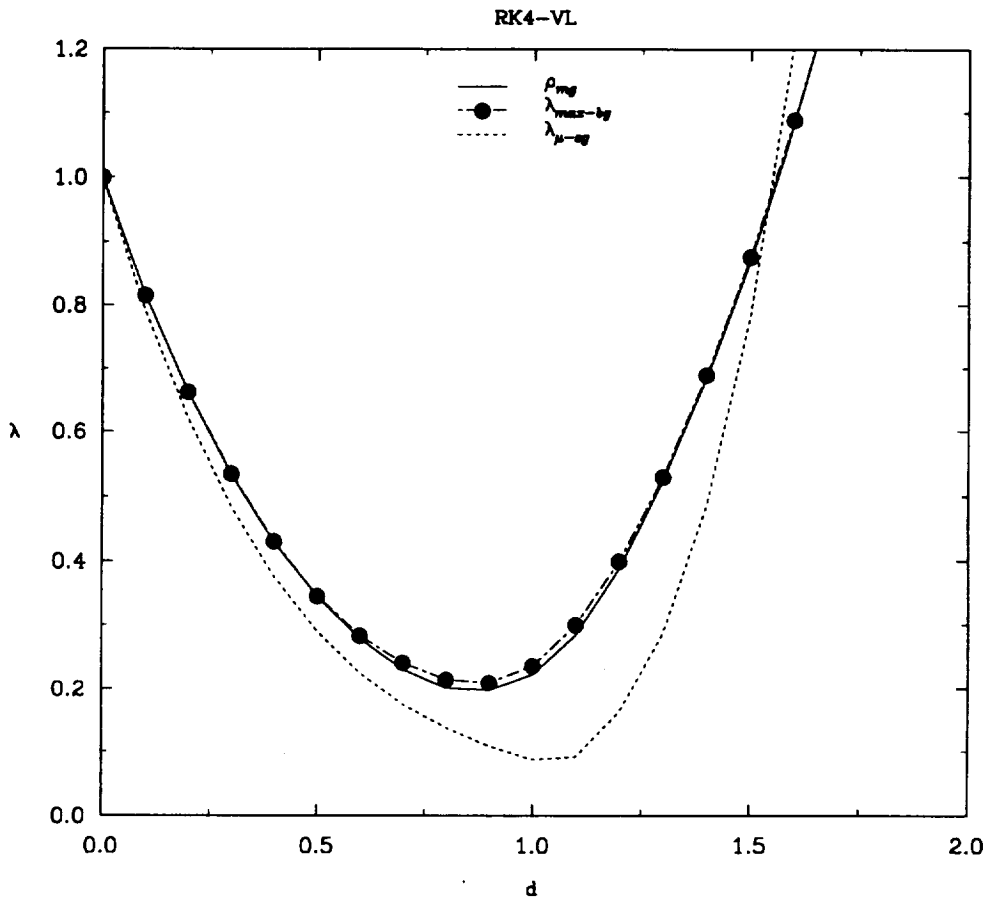


Fig. 3.7: Convergence Characteristics for 1–D Diffusion Equation (4–Stage Runge Kutta; Dirichlet B.C's; van Leer coefficients).

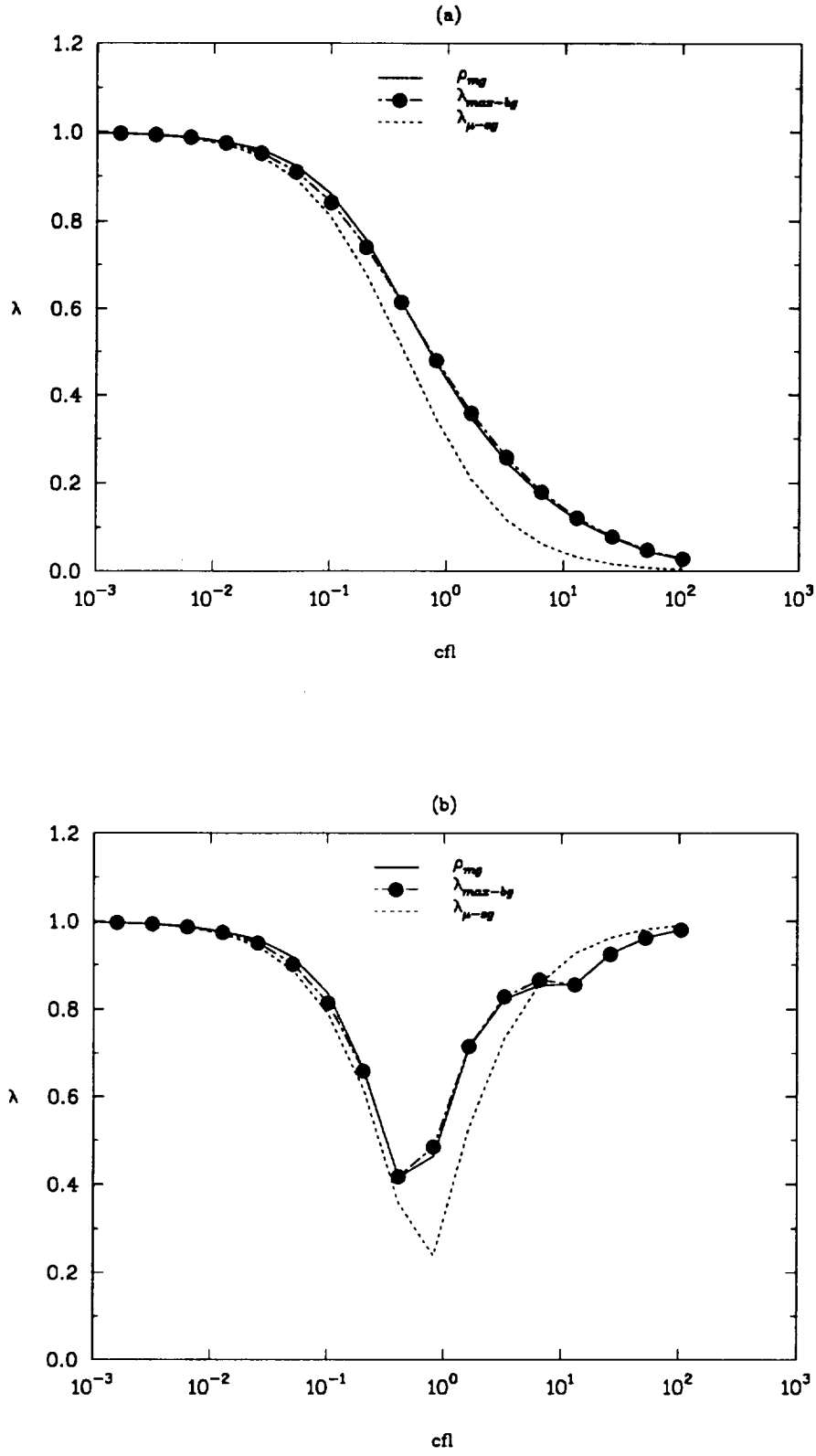
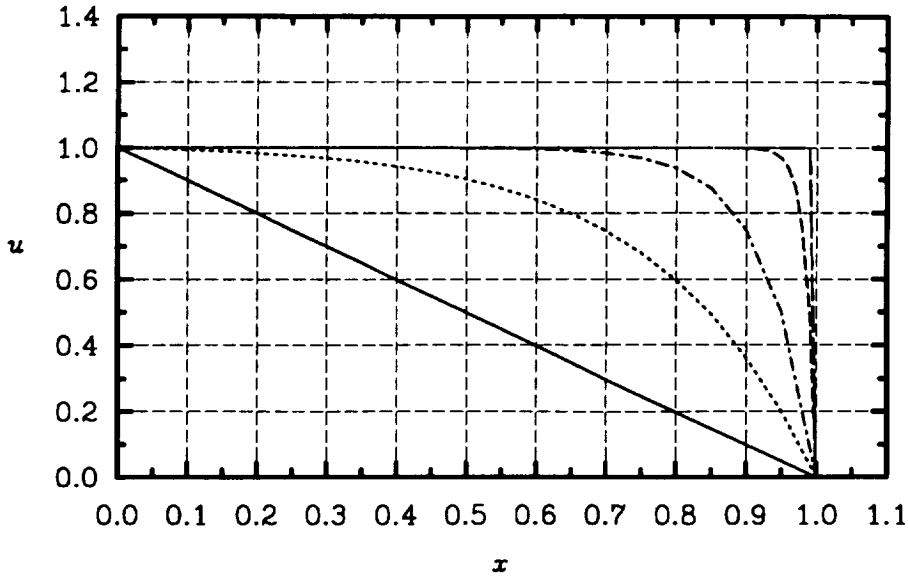
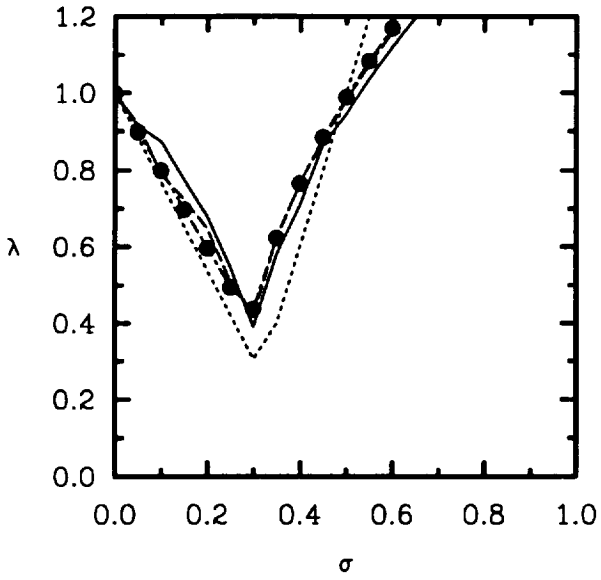
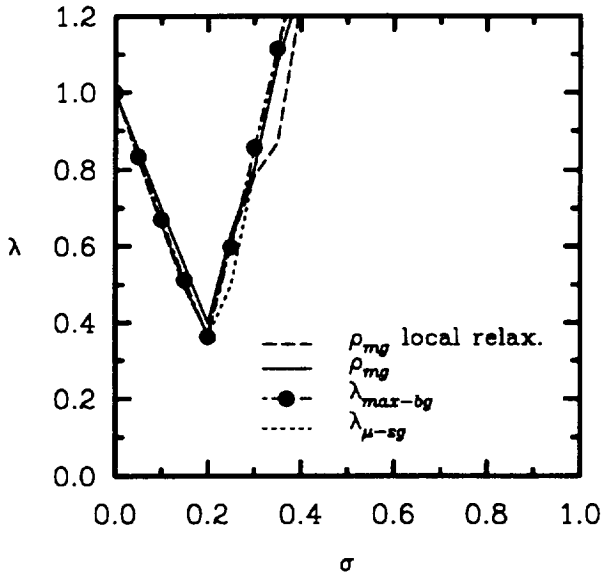
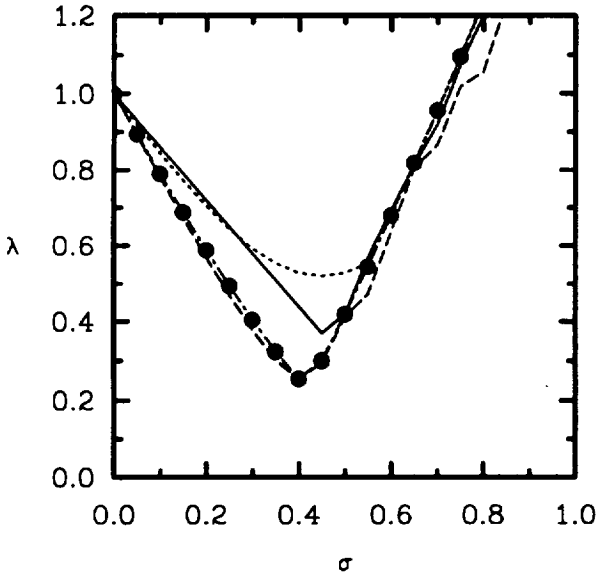


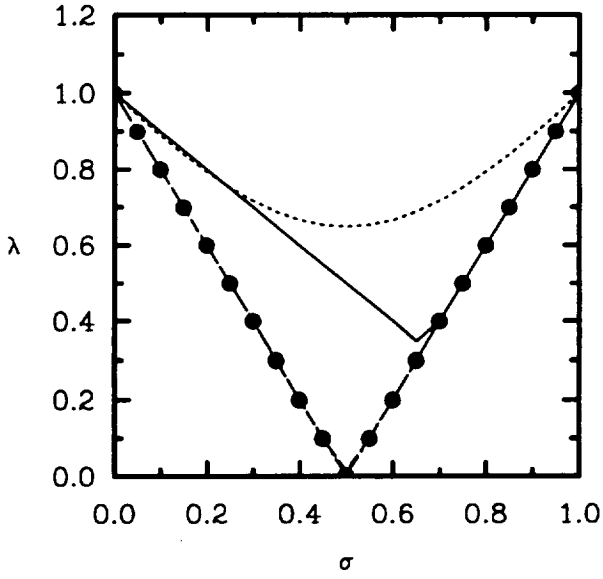
Fig. 3.8: Convergence Characteristics for 1−D Diffusion Equation (a) Implicit (b) Semi−implicit time integrations (Dirichlet B.C's).


1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
u
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1
x


1.2
1.0
0.8
0.6
0.4
0.2
0.0
λ
0.0 0.2 0.4 0.6 0.8 1.0
σ


1.2
1.0
0.8
0.6
0.4
0.2
0.0
λ
0.0 0.2 0.4 0.6 0.8 1.0
σ
ρmg local relax.
ρmg
λmax-bg
λμ-sg


1.2
1.0
0.8
0.6
0.4
0.2
0.0
λ
0.0 0.2 0.4 0.6 0.8 1.0
σ


1.2
1.0
0.8
0.6
0.4
0.2
0.0
λ
0.0 0.2 0.4 0.6 0.8 1.0
σ

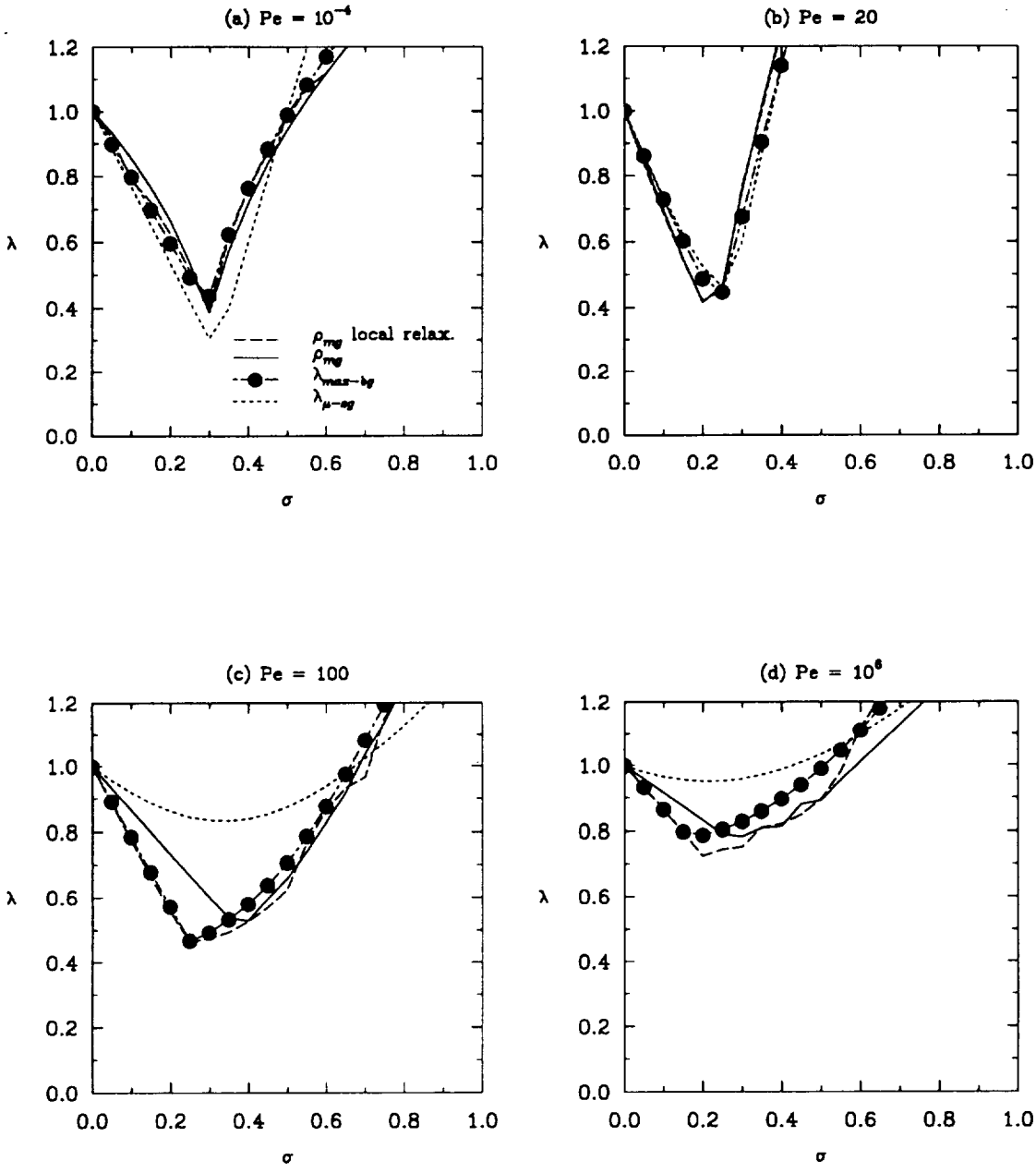


Fig. 3.10: 1−D Burger's Equation (a)−(d) Convergence Characteristics (Euler forward explicit; 2nd O accurate).

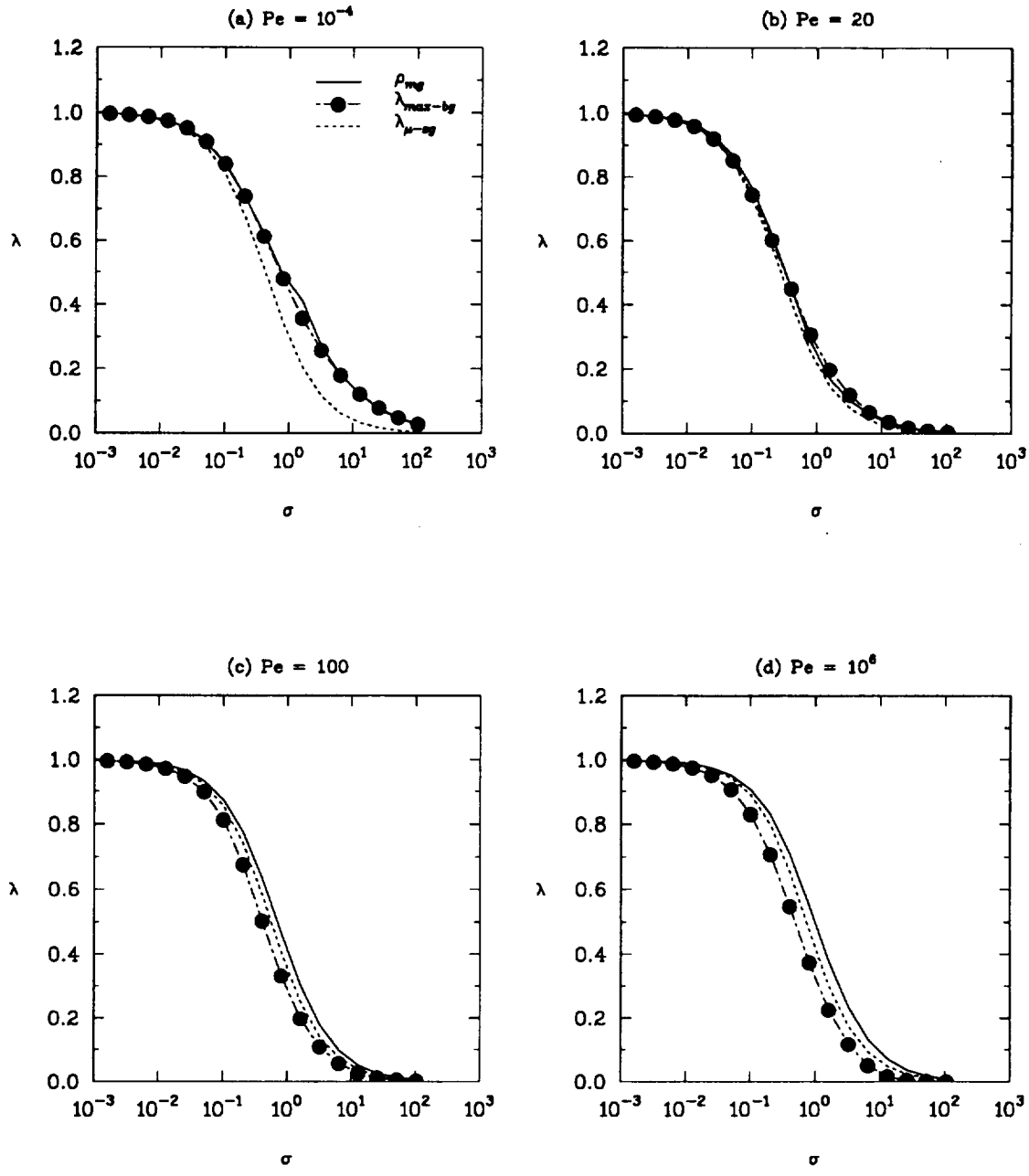


Fig. 3.11: 1−D Burger's Equation (a)−(d) Convergence Characteristics (Implicit time integration).

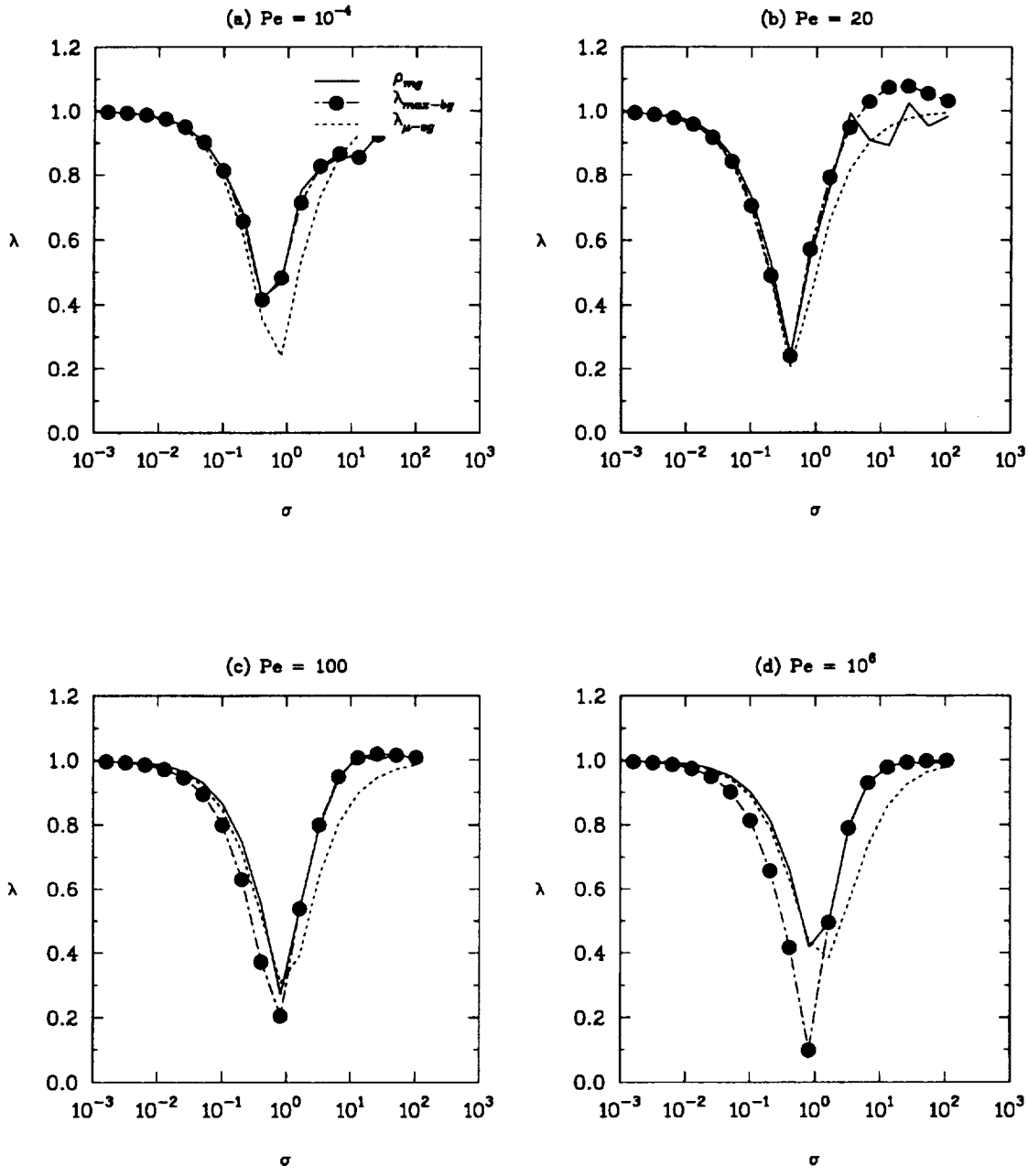
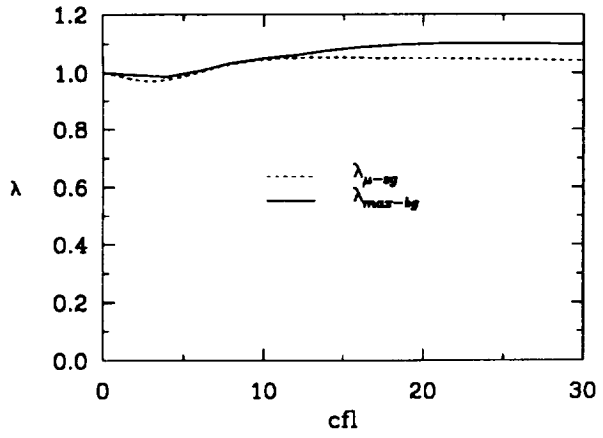
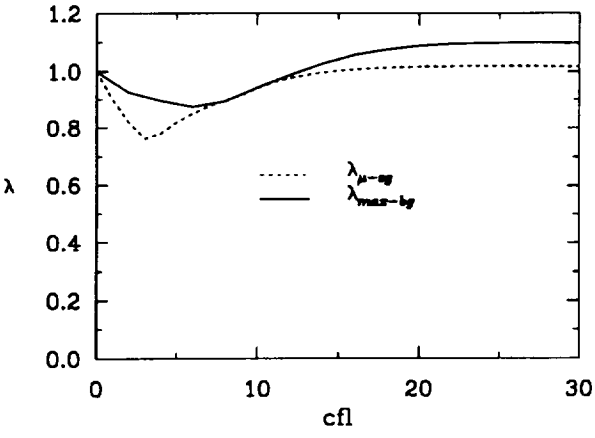


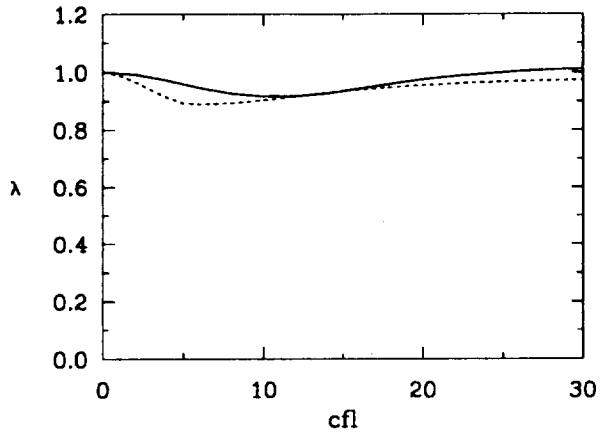
Fig. 3.12: 1–D Burger's Equation (a)–(d) Convergence Characteristics (Semi–implicit time integration).



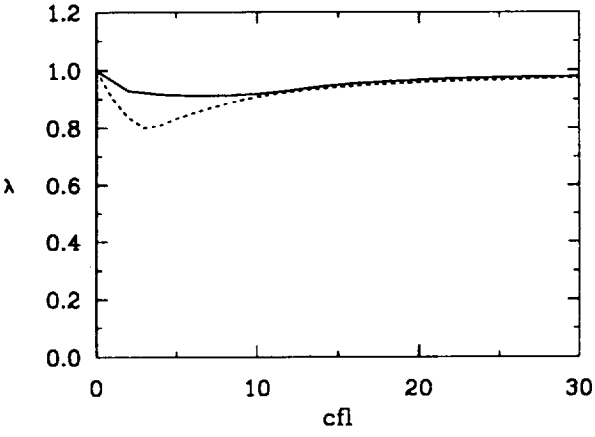
(a) Spatial Factorization



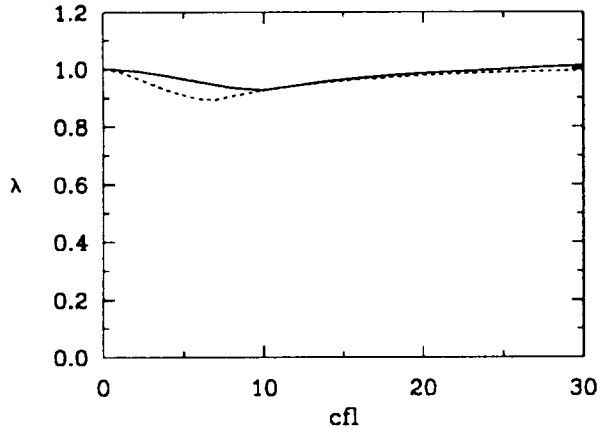
(d) Spatial Factorization



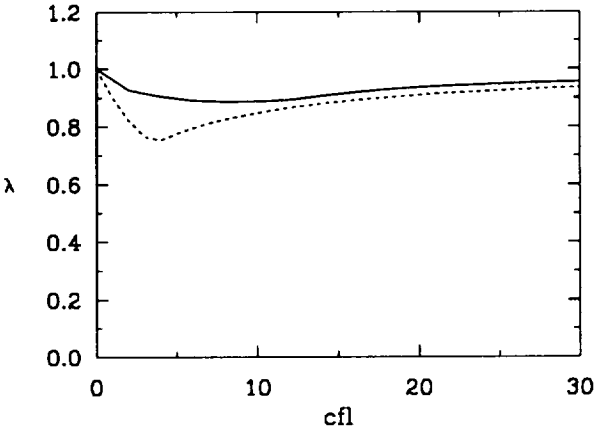
(b) Eigenvalue Factorization



(e) Eigenvalue Factorization



(c) Combination Factorization



(f) Combination Factorization

Steger & Warming

van Leer

Fig. 3.13: 3–D Euler Equations using upwind schemes (a)–(f)
Convergence Characteristics (ν^1=1; ν^2=0)

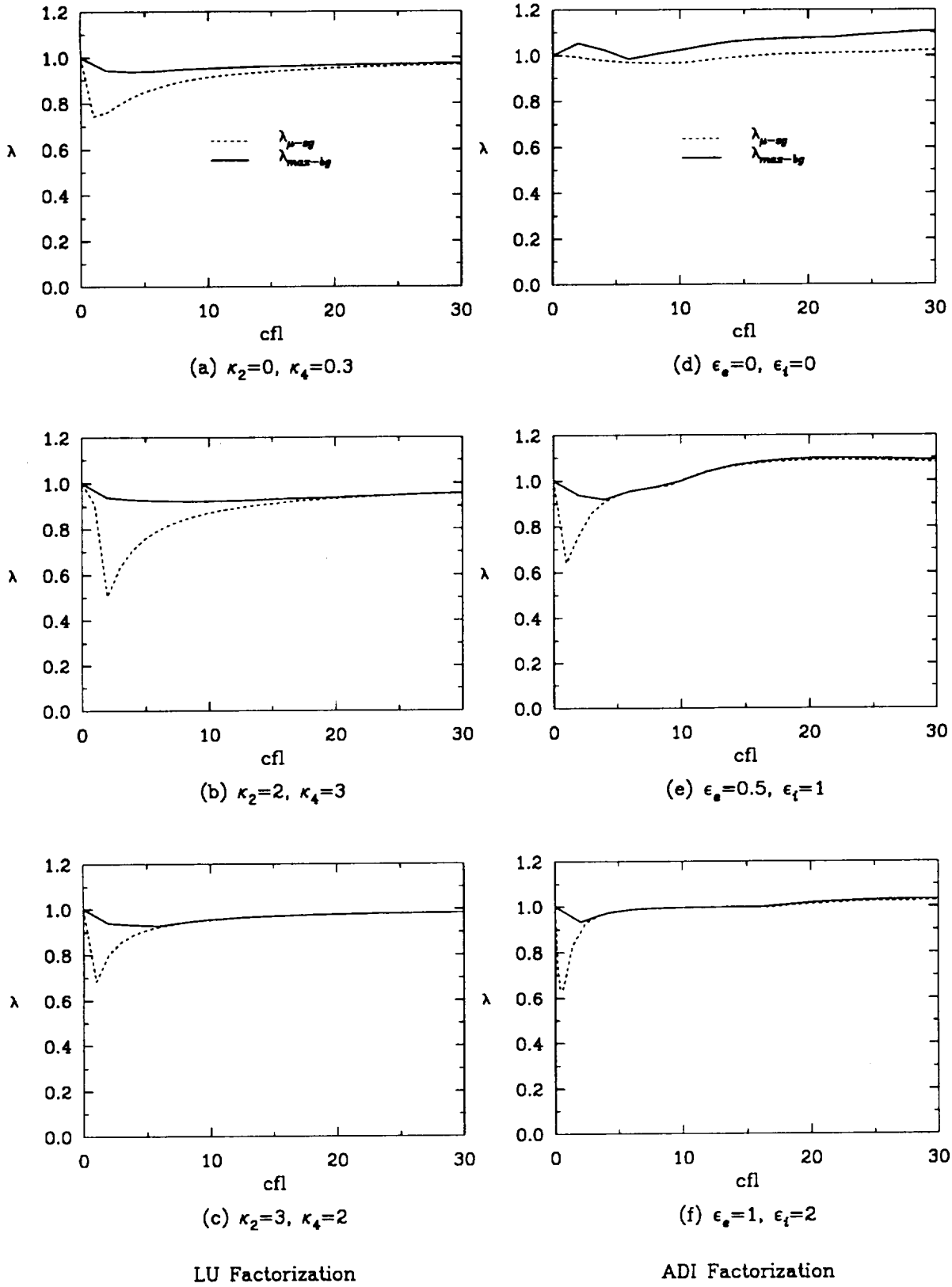


Fig. 3.14: 3−D Euler Equations using central schemes (a)−(f)
Convergence Characteristics (ν^1=1; ν^2=0)

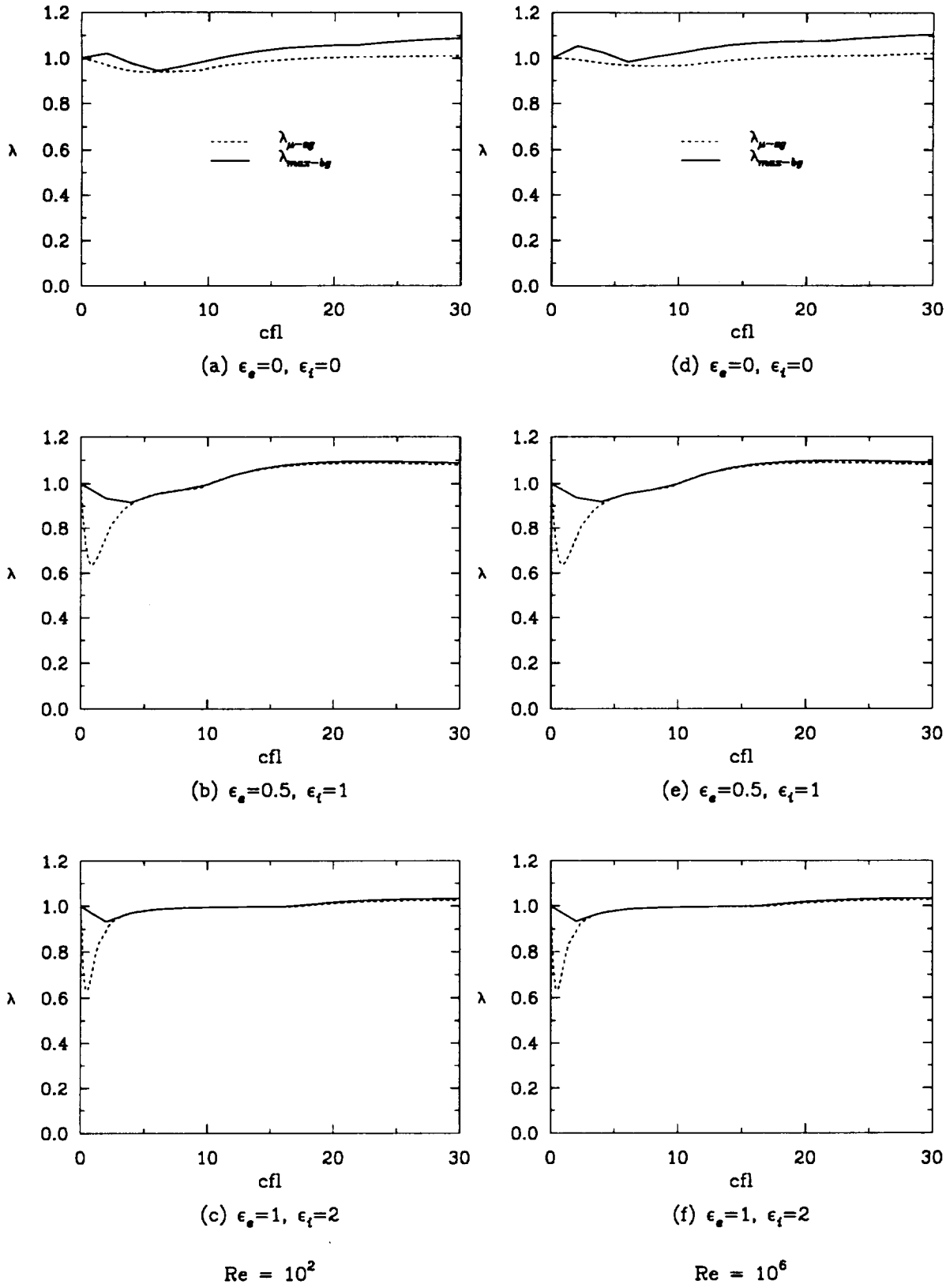


Fig. 3.15: 3–D Navier–Stokes Equations using central schemes (a)–(f) Convergence Characteristics (ν^1=1; ν^2=0)

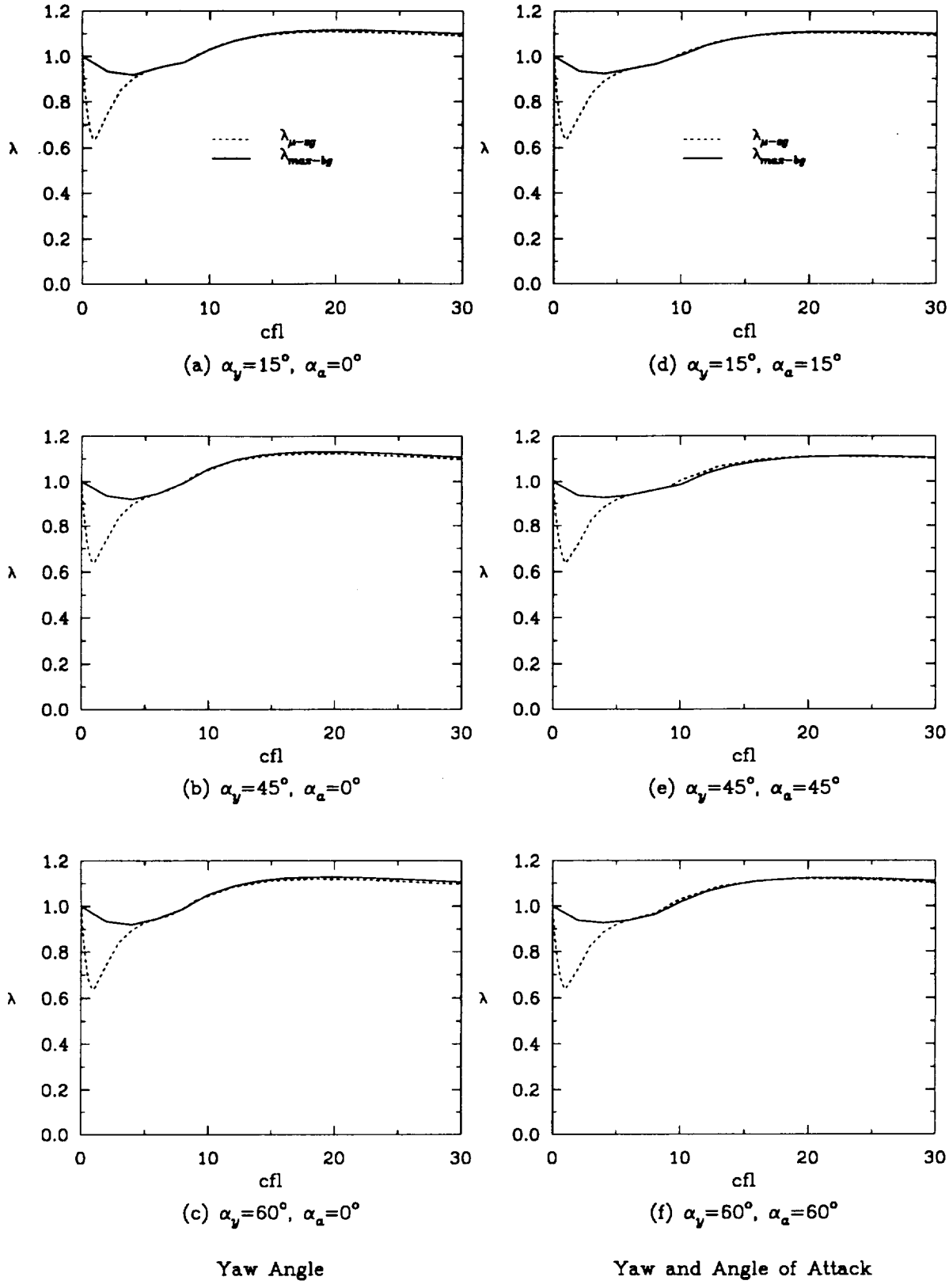


Fig. 3.16: 3–D Navier–Stokes Equations using central schemes (a)–(f) Convergence Characteristics; Flow Skewness (Re=100, ϵ_e=0.5, ϵ_i=1, ν^1=1; ν^2=0)

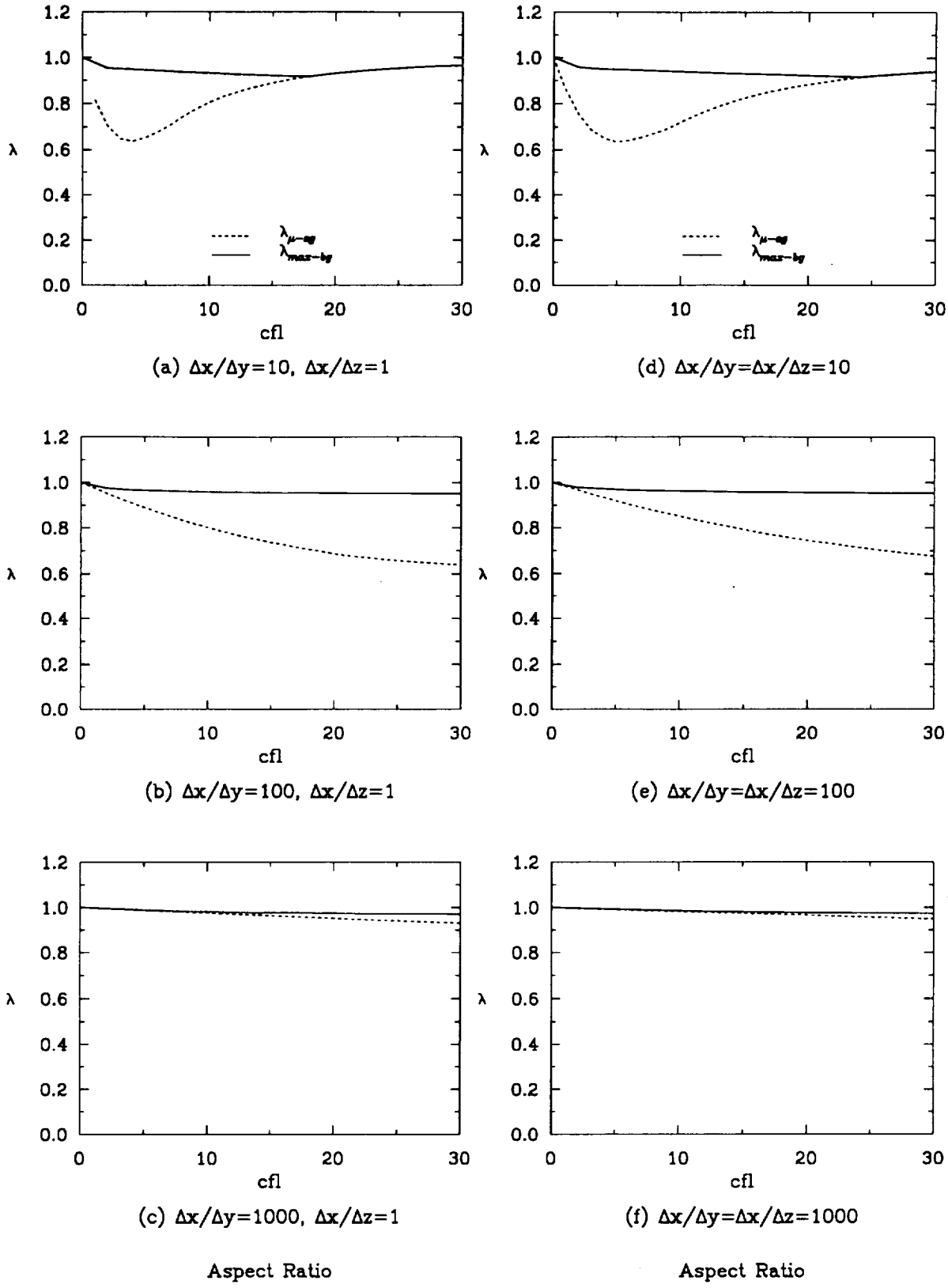


Fig. 3.17: 3–D Navier–Stokes Equations using central schemes (a)–(f) Convergence Characteristics; Aspect Ratio (Re=100, ϵ_e=0.5, ϵ_i=1, ν^1=1; ν^2=0)

3.4 Concluding Remarks

Bi–grid stability analysis has been presented for typical explicit and implicit solution methods for model problems which range from the diffusion equation to the convection equation and including the convection–diffusion equation at different Peclet numbers. Bi–grid amplification factors were compared with smoothing factors and multigrid convergence rates. The predicted bi–grid amplification factors agree quite well with the asymptotic convergence rate of the multigrid method. The smoothing rate of the relaxation scheme obtained from a local mode analysis on a single grid is not an accurate predictor of the multigrid convergence rate. For multigrid performance in large scale practical computations, bi–grid amplification factors and smoothing factors were computed from the system of 3–D Euler and Navier–Stokes equations; various approximate factorization methods that are popular in practice have been considered. The bi–grid results also compared better with the convergence rate of a typical multigrid solution of the 3–D transonic flow than did predictions of the smoothing factor approach. Armed with this versatile tool, the multigrid procedure can now be developed in subsequent chapters for both 2–D and 3–D steady flows.

Chapter 4
COMPUTATIONAL METHOD

The 2–D and 3–D Proteus computer codes were developed at NASA Lewis Research Center by Towne et al. (1990, 1992). The codes solve the Reynolds averaged, unsteady compressible Navier–Stokes equations in strong conservation law form. The governing equations, derived from the basic principles of conservation of mass, momentum and energy, are written in Cartesian coordinates and transformed into generalized nonorthogonal body fitted coordinates. They are solved by marching in time using a fully coupled alternating–direction–implicit (ADI) solution procedure with generalized first– or second–order time accuracy. Turbulence effects are accounted for using either an algebraic or two–equation eddy viscosity model. A brief summary of the mathematical formulation for the 3–D version follows.

4.1 Governing Equations

The basic governing equations are the three–dimensional compressible Navier–Stokes equations. In generalized curvilinear coordinates, the three–dimensional planar equations can be written in strong conservation law form using vector notation as:

$$\frac{\partial Q}{\partial t} + \frac{\partial (E - E_v)}{\partial \xi} + \frac{\partial (F - F_v)}{\partial \eta} + \frac{\partial (G - G_v)}{\partial \zeta} = 0 \tag{4.1}$$

where the conserved variables of vector Q are defined as:

$$Q = \frac{1}{J}[\varrho,\ \varrho u,\ \varrho v,\ \varrho w,\ \varrho e_o]^T \tag{4.2}$$

The inviscid flux vectors E, F and G are

$$E = \frac{1}{J}\begin{bmatrix} \varrho u\xi_x + \varrho v\xi_y + \varrho w\xi_z \\ (\varrho u^2 + p)\xi_x + \varrho uv\xi_y + \varrho uw\xi_z \\ \varrho uv\xi_x + (\varrho v^2 + p)\xi_y + \varrho vw\xi_z \\ \varrho uw\xi_x + \varrho vw\xi_y + (\varrho w^2 + p)\xi_z \\ (\varrho e_o + p)u\xi_x + (\varrho e_o + p)v\xi_y + (\varrho e_o + p)w\xi_z \end{bmatrix} \tag{4.3}$$

$$F = \frac{1}{J}\begin{bmatrix} \varrho u\eta_x + \varrho v\eta_y + \varrho w\eta_z \\ (\varrho u^2 + p)\eta_x + \varrho uv\eta_y + \varrho uw\eta_z \\ \varrho uv\eta_x + (\varrho v^2 + p)\eta_y + \varrho vw\eta_z \\ \varrho uw\eta_x + \varrho vw\eta_y + (\varrho w^2 + p)\eta_z \\ (\varrho e_o + p)u\eta_x + (\varrho e_o + p)v\eta_y + (\varrho e_o + p)w\eta_z \end{bmatrix} \tag{4.4}$$

$$G = \frac{1}{J}\begin{bmatrix} \varrho u\zeta_x + \varrho v\zeta_y + \varrho w\zeta_z \\ (\varrho u^2 + p)\zeta_x + \varrho uv\zeta_y + \varrho uw\zeta_z \\ \varrho uv\zeta_x + (\varrho v^2 + p)\zeta_y + \varrho vw\zeta_z \\ \varrho uw\zeta_x + \varrho vw\zeta_y + (\varrho w^2 + p)\zeta_z \\ (\varrho e_o + p)u\zeta_x + (\varrho e_o + p)v\zeta_y + (\varrho e_o + p)w\zeta_z \end{bmatrix} \tag{4.5}$$

The viscous flux vectors E_V, F_V and G_V are

$$E_V = \frac{1}{J}\frac{1}{\mathrm{Re}_r}\begin{bmatrix} 0 \\ \tau_{xx}\xi_x + \tau_{xy}\xi_y + \tau_{xz}\xi_z \\ \tau_{xy}\xi_x + \tau_{yy}\xi_y + \tau_{yz}\xi_z \\ \tau_{xz}\xi_x + \tau_{yz}\xi_y + \tau_{zz}\xi_z \\ \beta_x\xi_x + \beta_y\xi_y + \beta_z\xi_z \end{bmatrix} \tag{4.6}$$

$$G_V = \frac{1}{J}\frac{1}{\mathrm{Re}_r}\begin{bmatrix} 0 \\ \tau_{xx}\zeta_x + \tau_{xy}\zeta_y + \tau_{xz}\zeta_z \\ \tau_{xy}\zeta_x + \tau_{yy}\zeta_y + \tau_{yz}\zeta_z \\ \tau_{xz}\zeta_x + \tau_{yz}\zeta_y + \tau_{zz}\zeta_z \\ \beta_x\zeta_x + \beta_y\zeta_y + \beta_z\zeta_z \end{bmatrix} \tag{4.7}$$

$$F_V = \frac{1}{J}\frac{1}{\mathrm{Re}_r}\begin{bmatrix} 0 \\ \tau_{xx}\eta_x + \tau_{xy}\eta_y + \tau_{xz}\eta_z \\ \tau_{xy}\eta_x + \tau_{yy}\eta_y + \tau_{yz}\eta_z \\ \tau_{xz}\eta_x + \tau_{yz}\eta_y + \tau_{zz}\eta_z \\ \beta_x\eta_x + \beta_y\eta_y + \beta_z\eta_z \end{bmatrix} \tag{4.8}$$

where

$$\begin{aligned} \beta_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - \frac{1}{\mathrm{Pr}_r}q_x \\ \beta_y &= u\tau_{xy} + v\tau_{yy} + w\tau_{yz} - \frac{1}{\mathrm{Pr}_r}q_y \\ \beta_y &= u\tau_{xz} + v\tau_{yz} + w\tau_{zz} - \frac{1}{\mathrm{Pr}_r}q_z \end{aligned} \tag{4.9}$$

and

$$\begin{aligned} \tau_{xx} &= 2\mu\frac{\partial u}{\partial x} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) \\ \tau_{yy} &= 2\mu\frac{\partial v}{\partial y} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) \\ \tau_{zz} &= 2\mu\frac{\partial w}{\partial z} + \lambda\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right) \\ \tau_{xy} &= \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \quad \tau_{xz} = \mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right), \quad \tau_{yz} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \\ q_x &= -k\frac{\partial T}{\partial x} \\ q_y &= -k\frac{\partial T}{\partial y} \\ q_y &= -k\frac{\partial T}{\partial z} \end{aligned} \tag{4.10}$$

The governing equations as expressed above have been nondimensionalized using reference conditions u_r, l_r, ϱ_r and T_r. The derivatives in the shear stresses and heat fluxes are in Cartesians coordinates and must be evaluated in terms of the generalized coordinates using the chain rule. For example,

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial \xi}\xi_x + \frac{\partial u}{\partial \eta}\eta_x + \frac{\partial u}{\partial \zeta}\zeta_x \tag{4.11}$$

Note that the conserved variables and all the fluxes are defined in curvilinear coordinates and not in Cartesian coordinates as in Chaps. 3 and 4. In addition to the above equations, an equation of state is required to link the pressure to the dependent variables. The ideal gas law is chosen which for a calorically perfect gases can be written as

$$p = (\gamma - 1)\left[\varrho e_o - \frac{1}{2}\varrho\left(u^2 + v^2 + w^2\right)\right] \tag{4.12}$$

4.2 Time Differencing

The governing equations are solved by marching in time from some known set of initial conditions using a finite difference technique. The time differencing used is the generalized scheme of Beam and Warming (1978), where the time derivative term in Eq. (4.1) is written as

$$\frac{\partial Q}{\partial t} = \frac{\theta_1}{1+\theta_2}\frac{\partial(\Delta Q^n)}{\partial t} + \frac{1}{1+\theta_2}\frac{\partial Q^n}{\partial t} + \frac{\theta_2}{1+\theta_2}\frac{\Delta Q^{(n-1)}}{\Delta t} + O\left[\left(\theta_1 - \frac{1}{2} - \theta_2\right)\Delta t, (\Delta t)_2\right] \tag{4.13}$$

or,

$$\Delta Q^n = \frac{\theta_1 \Delta t}{1+\theta_2}\frac{\partial(\Delta Q^n)}{\partial t} + \frac{\Delta t}{1+\theta_2}\frac{\partial Q^n}{\partial t} + \frac{\theta_2}{1+\theta_2}\Delta Q^{(n-1)} + O\left[\left(\theta_1 - \frac{1}{2} - \theta_2\right)\Delta t, (\Delta t)_2\right] \tag{4.14}$$

where $\Delta Q^n = Q^{n+1} - Q^n$.

The parameters θ_1 and θ_2 determine the type of time differencing scheme used. Some of the methods available with the above formula are given in Table 4.1.

Table 4.1: Different Types of Time–Stepping Methods

θ_1	θ_2	Method	Truncation Error
0	0	Euler explicit	$O(\Delta t)$
0	–1/2	Leapfrog explicit	$O(\Delta t)^2$
1	0	Euler implicit	$O(\Delta t)$
1/2	0	Tapezoidal implicit	$O(\Delta t)^2$
1	1/2	3–point backward implicit	$O(\Delta t)^2$

Solving Eq. (4.1) for $\partial Q/\partial t$ and substituting the results into Eq. (4.14) for $\partial(\Delta Q^n)/\partial t$ and $\partial Q^n/\partial t$ yields

$$\begin{aligned}\Delta Q^n = &-\frac{\theta_1 \Delta t}{1+\theta_2}\left(\frac{\partial(\Delta E^n)}{\partial \xi}+\frac{\partial(\Delta F^n)}{\partial \eta}+\frac{\partial(\Delta G^n)}{\partial \zeta}\right)-\frac{\Delta t}{1+\theta_2}\left(\frac{\partial E^n}{\partial \xi}+\frac{\partial F^n}{\partial \eta}+\frac{\partial G^n}{\partial \zeta}\right)\\ &+\frac{\theta_1 \Delta t}{1+\theta_2}\left(\frac{\partial(\Delta E_V^n)}{\partial \xi}+\frac{\partial(\Delta F_V^n)}{\partial \eta}+\frac{\partial(\Delta G_V^n)}{\partial \zeta}\right)\\ &+\frac{\Delta t}{1+\theta_2}\left(\frac{\partial E_V^n}{\partial \xi}+\frac{\partial F_V^n}{\partial \eta}+\frac{\partial G_V^n}{\partial \zeta}\right)+\frac{\theta_2}{1+\theta_2}\Delta Q^{(n-1)}+O\left[\left(\theta_1-\frac{1}{2}-\theta_2\right)\Delta t,(\Delta t)_2\right]\end{aligned} \tag{4.15}$$

4.3 Linearization Procedure

Equation (4.15) is nonlinear, since, for example, $\Delta E^n = E^{n+1} - E^n$ and the unknown E^{n+1} is a nonlinear function of the dependent variables and the metric coefficients resulting from the generalized grid transformation. The equations are linearized in order to use the finite

difference technique. For any nonlinear expression G, a Taylor series expansion about a known time level n can be written as

$$G^{n+1} = G^n + \frac{\partial G^n}{\partial t}\Delta t + O(\Delta t)^2 \tag{4.16}$$

This linearization procedure, when applied to the entire inviscid fluxes ΔE^n, ΔF^n and ΔG^n terms in the Eq. (4.15), can be expressed as

$$\begin{aligned}
\Delta E^n &= \left(\frac{\partial E^n}{\partial Q}\right)^n \Delta Q^n + O(\Delta t)^2 \\
\Delta F^n &= \left(\frac{\partial F^n}{\partial Q}\right)^n \Delta Q^n + O(\Delta t)^2 \\
\Delta G^n &= \left(\frac{\partial G^n}{\partial Q}\right)^n \Delta Q^n + O(\Delta t)^2
\end{aligned} \tag{4.17}$$

$(\partial E/\partial Q)^n$, $(\partial F/\partial Q)^n$ and $(\partial G/\partial Q)^n$ are the flux Jacobian matrices.

The viscous fluxes ΔE_V^n, ΔF_V^n and ΔG_V^n are also linearized, although in a slightly different manner. The mixed, or cross, derivative terms in these fluxes would lead to considerable complications in the implicit numerical solution algorithm. In order to avoid this problem, the fluxes are split directionally as

$$\begin{aligned}
E_V &= E_{V_1} + E_{V_2} \\
F_V &= F_{V_1} + F_{V_2} \\
G_V &= G_{V_1} + G_{V_2}
\end{aligned} \tag{4.18}$$

Thus, E_{V_1}, F_{V_1} and G_{V_1} only contain derivatives in the ξ and η directions, respectively, and E_{V_2} F_{V_2} and G_{V_2} contain derivatives in the other direction. The linearization for the E_{V_1} F_{V_1} and G_{V_1} fluxes is carried out in a similar manner as in the inviscid case while the cross derivatives terms are simply lagged as follows (Beam and Warming, 1978)

$$\Delta E^n_{V_2} = \Delta E^{n-1}_{V_2} + O(\Delta t)^2$$
$$\Delta F^n_{V_2} = \Delta F^{n-1}_{V_2} + O(\Delta t)^2 \qquad (4.19)$$
$$\Delta G^n_{V_2} = \Delta G^{n-1}_{V_2} + O(\Delta t)^2$$

With these equations, the linearized form of Eq. (4.15) can now be written as

$$\begin{aligned}
&\Delta Q^n + \frac{\theta_1 \Delta t}{1+\theta_2}\left\{\frac{\partial}{\partial\xi}\left[\left(\frac{\partial E}{\partial Q}\right)^n \Delta Q^n\right] + \frac{\partial}{\partial\eta}\left[\left(\frac{\partial F}{\partial Q}\right)^n \Delta Q^n\right] + \frac{\partial}{\partial\zeta}\left[\left(\frac{\partial G}{\partial Q}\right)^n \Delta Q^n\right]\right\} \\
&\quad - \frac{\theta_1 \Delta t}{1+\theta_2}\left\{\frac{\partial}{\partial\xi}\left[\left(\frac{\partial E_{V_1}}{\partial Q}\right)^n \Delta Q^n\right] + \frac{\partial}{\partial\eta}\left[\left(\frac{\partial F_{V_1}}{\partial Q}\right)^n \Delta Q^n\right] + \frac{\partial}{\partial\zeta}\left[\left(\frac{\partial G_{V_1}}{\partial Q}\right)^n \Delta Q^n\right]\right\} = \\
&\quad - \frac{\Delta t}{1+\theta_2}\left(\frac{\partial E}{\partial\xi} + \frac{\partial F}{\partial\eta} + \frac{\partial G}{\partial\zeta}\right)^n + \frac{\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_1}}{\partial\xi} + \frac{\partial F_{V_1}}{\partial\eta} + \frac{\partial G_{V_1}}{\partial\zeta}\right)^n \\
&\quad + \frac{(1+\theta_3)\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi} + \frac{\partial F_{V_2}}{\partial\eta} + \frac{\partial G_{V_2}}{\partial\zeta}\right)^n - \frac{\theta_3 \Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi} + \frac{\partial F_{V_2}}{\partial\eta} + \frac{\partial G_{V_2}}{\partial\zeta}\right)^n \\
&\quad + \frac{\theta_2}{1+\theta_2}\Delta Q^{(n-1)} + O\left[\left(\theta_1 - \tfrac{1}{2} - \theta_2\right)(\Delta t)^2, (\theta_3 - \theta_1)(\Delta t)^2, (\Delta t)^2\right]
\end{aligned} \qquad (4.20)$$

θ_3 is introduced to replace θ_1 in the coefficients of the cross derivative viscous terms. θ_3 is set equal to θ_1 for second–order time accuracy. For first–order time differencing, however, θ_3 can be set equal to zero without losing accuracy.

Equation (4.20) can be further put in the following form:

$$\begin{aligned}
&\left\{\mathbf{I} + \frac{\theta_1 \Delta t}{1+\theta_2}\left[\frac{\partial}{\partial\xi}\left(\frac{\partial E}{\partial Q} - \frac{\partial E_{V_1}}{\partial Q}\right) + \frac{\partial}{\partial\eta}\left(\frac{\partial F}{\partial Q} - \frac{\partial F_{V_1}}{\partial Q}\right) + \frac{\partial}{\partial\zeta}\left(\frac{\partial G}{\partial Q} - \frac{\partial G_{V_1}}{\partial Q}\right)\right]\right\}^n \Delta Q^n = \\
&\quad - \frac{\Delta t}{1+\theta_2}\left(\frac{\partial E}{\partial\xi} + \frac{\partial F}{\partial\eta} + \frac{\partial G}{\partial\zeta}\right)^n + \frac{\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_1}}{\partial\xi} + \frac{\partial F_{V_1}}{\partial\eta} + \frac{\partial G_{V_1}}{\partial\zeta}\right)^n \\
&\quad + \frac{(1+\theta_3)\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi} + \frac{\partial F_{V_2}}{\partial\eta} + \frac{\partial G_{V_2}}{\partial\zeta}\right)^n - \frac{\theta_3 \Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi} + \frac{\partial F_{V_2}}{\partial\eta} + \frac{\partial G_{V_2}}{\partial\zeta}\right)^n \\
&\quad + \frac{\theta_2}{1+\theta_2}\Delta Q^{(n-1)} + O\left[\left(\theta_1 - \tfrac{1}{2} - \theta_2\right)(\Delta t)^2, (\theta_3 - \theta_1)(\Delta t)^2, (\Delta t)^2\right]
\end{aligned} \qquad (4.21)$$

where $\partial/\partial\xi$ term as an example is meant to imply

C-2.

$$\left[\frac{\partial}{\partial\xi}\left(\frac{\partial E}{\partial Q}\right)^n\right]\Delta Q^n = \frac{\partial}{\partial\xi}\left[\left(\frac{\partial E}{\partial Q}\right)^n \Delta Q^n\right] \tag{4.22}$$

4.4 Solution Procedure

The governing Eq. (4.21), presented in linearized matrix form result in a system of algebraic equations. The coefficient matrix is banded and the band width depends on the grid size and choice of spatial differencing method. The left hand side requires an inversion of a very large matrix. The exact inversions of the matrix is very costly due to the large number of operations and computer memory required. To reduce this computational expense, an approximate factorization (AF) method is introduced which factors the implicit operator into a sequence of easily invertible matrices. The splitting is based on the alternating direction implicit (ADI) method of Beam and Warming, where splitting is along the spatial directions as follows:

$$\begin{aligned}\text{LHS}(4.21) = &\left[\mathbf{I} + \frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\xi}\left(\frac{\partial E}{\partial Q} - \frac{\partial E_{V_1}}{\partial Q}\right)\right]^n\left[\mathbf{I} + \frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\eta}\left(\frac{\partial F}{\partial Q} - \frac{\partial F_{V_1}}{\partial Q}\right)\right]^n \\ &\left[\mathbf{I} + \frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\zeta}\left(\frac{\partial G}{\partial Q} - \frac{\partial G_{V_1}}{\partial Q}\right)\right]^n \Delta Q^n + h.o.t\end{aligned} \tag{4.23}$$

The higher–order–term (h.o.t) represents the splitting error which can be neglected without affecting the overall time accuracy of the algorithm, even when second–order time differencing is assumed. However, this approximate factorization error may place a restriction on the choice of time step.

Equation (4.21) can thus be rewritten in a spatially–factored form, which, neglecting the temporal truncation and splitting error terms, becomes

$$\left[\mathbf{I}+\frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\xi}\left(\frac{\partial E}{\partial Q}-\frac{\partial E_{V_1}}{\partial Q}\right)\right]^n\left[\mathbf{I}+\frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\eta}\left(\frac{\partial F}{\partial Q}-\frac{\partial F_{V_1}}{\partial Q}\right)\right]^n$$
$$\left[\mathbf{I}+\frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\zeta}\left(\frac{\partial G}{\partial Q}-\frac{\partial G_{V_1}}{\partial Q}\right)\right]^n\Delta Q^n=$$
$$-\frac{\Delta t}{1+\theta_2}\left(\frac{\partial E}{\partial\xi}+\frac{\partial F}{\partial\eta}+\frac{\partial G}{\partial\zeta}\right)^n+\frac{\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_1}}{\partial\xi}+\frac{\partial F_{V_1}}{\partial\eta}+\frac{\partial G_{V_1}}{\partial\zeta}\right)^n \tag{4.24}$$
$$+\frac{(1+\theta_3)\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi}+\frac{\partial F_{V_2}}{\partial\eta}+\frac{\partial G_{V_2}}{\partial\zeta}\right)^n-\frac{\theta_3\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi}+\frac{\partial F_{V_2}}{\partial\eta}+\frac{\partial G_{V_2}}{\partial\zeta}\right)^n+\frac{\theta_2}{1+\theta_2}\Delta Q^{(n-1)}$$

The equations in the approximate factorization form presented above are split into the following three–sweep sequence:

Sweep 1 (ξ direction)

$$\left[\mathbf{I}+\frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\xi}\left(\frac{\partial E}{\partial Q}-\frac{\partial E_{V_1}}{\partial Q}\right)\right]^n\Delta Q^*=$$
$$-\frac{\Delta t}{1+\theta_2}\left(\frac{\partial E}{\partial\xi}+\frac{\partial F}{\partial\eta}+\frac{\partial G}{\partial\zeta}\right)^n+\frac{\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_1}}{\partial\xi}+\frac{\partial F_{V_1}}{\partial\eta}+\frac{\partial G_{V_1}}{\partial\zeta}\right)^n \tag{4.25}$$
$$\frac{(1+\theta_3)\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi}+\frac{\partial F_{V_2}}{\partial\eta}+\frac{\partial G_{V_2}}{\partial\zeta}\right)^n-\frac{\theta_3\Delta t}{1+\theta_2}\left(\frac{\partial E_{V_2}}{\partial\xi}+\frac{\partial F_{V_2}}{\partial\eta}+\frac{\partial G_{V_2}}{\partial\zeta}\right)^n+\frac{\theta_2}{1+\theta_2}\Delta Q^{(n-1)}$$

Sweep 2 (η direction)

$$\left[\mathbf{I}+\frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\eta}\left(\frac{\partial F}{\partial Q}-\frac{\partial F_{V_1}}{\partial Q}\right)\right]^n\Delta Q^{**}=\Delta Q^* \tag{4.26}$$

Sweep 3 (ζ direction)

$$\left[\mathbf{I}+\frac{\theta_1\Delta t}{1+\theta_2}\frac{\partial}{\partial\zeta}\left(\frac{\partial G}{\partial Q}-\frac{\partial G_{V_1}}{\partial Q}\right)\right]^n\Delta Q^n=\Delta Q^{**} \tag{4.27}$$

In the above equations, Q^*, Q^{**} represent intermediate solutions to the governing equation. Each sweep requires the solution of a series of 5x5 block–tridiagonal systems and can be efficiently solved using the Thomas algorithm.

4.5 Space Differencing

To solve the governing equations, an evenly spaced grid is defined in the computational (ξ, η) coordinate system. The spatial derivatives in Eqs. (4.25) to (4.27) are then approximated by second–order finite difference formulas. First derivatives in the ξ direction are, for example, approximated using the following central difference formula:

$$\left(\frac{\partial f}{\partial \xi}\right)_{i,j} \approx \delta_\xi f_{i,j} = \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta\xi} \tag{4.28}$$

The subscripts i and j represent grid point indices in the ξ and η directions, respectively. The non–cross derivative viscous terms and the cross derivative viscous terms in the ξ direction all have the form $\frac{\partial}{\partial \xi}\left[f\frac{\partial}{\partial \xi}(g\Delta Q)\right]_{i,j}$ and $\frac{\partial}{\partial \xi}\left(f\frac{\partial g}{\partial \eta}\right)_{i,j}$ respectively. It can be shown that their second–order central difference approximation can be written as

$$\begin{aligned}\frac{\partial}{\partial \xi}\left[f\frac{\partial}{\partial \xi}(g\Delta Q)\right]_{i,j} &\approx \delta_\xi\left[f\delta_\xi(g\Delta Q)\right]_{i,j} \\ &= \frac{1}{2(\Delta\xi)^2}\Big\{\left(f_{i-1,j} + f_{i,j}\right)(g\Delta Q)_{i-1,j} - \left(f_{i-1,j} + 2f_{i,j} + f_{i+1,j}\right)(g\Delta Q)_{i,j} \\ &\qquad + \left(f_{i,j} + f_{i+1,j}\right)(g\Delta Q)_{i+1,j}\Big\}\end{aligned} \tag{4.29}$$

$$\begin{aligned}\frac{\partial}{\partial \xi}\left(f\frac{\partial g}{\partial \eta}\right)_{i,j} &\approx \delta_\xi\left(f\delta_\eta g\right)_{i,j} \\ &= \frac{1}{4\Delta\xi\Delta\eta}\left[f_{i+1,j}\left(g_{i+1,j+1} - g_{i+1,j-1}\right) - f_{i-1,j}\left(g_{i-1,j+1} - g_{i-1,j-1}\right)\right]\end{aligned} \tag{4.30}$$

When first derivatives are needed normal to a computational boundary, such as for Neumann boundary conditions, either first– or second–order one–sided differencing is used.

4.6 Artificial Viscosity

With the central difference formulation presented above, high frequency instabilities can appear as the solution develops. For example, in high Reynolds number flows oscillations can result from the odd–even decouplings inherent in the use of second–order central differencing. In addition, physical phenomena such as shock waves can cause instabilities when they are captured by a finite difference algorithm. Artificial viscosity, or dissipation, is normally added to the solution algorithm to suppress these high frequency instabilities. Two artificial viscosity models are considered in this study: a constant coefficient model used by Steger (1978), and the nonlinear coefficients model of Jameson, Schmidt and Turkel (1981). The constant coefficient model uses a combination of explicit and implicit artificial viscosity. The explicit artificial viscosity is further a combination of fourth– and second–order differences. As stated by Caughey (1988), the second–difference terms dissipate spurious waves in the shock region and fourth–difference terms are employed for steady state convergence. The implicit artificial viscosity is sometimes necessary to extend the linear stability bound of the fourth–order explicit dissipation.

The explicit artificial viscosity is implemented in the numerical algorithm by adding the following terms to the source terms in Eq. (4.25):

$$\frac{\varepsilon_e^2 \Delta t}{J}\left(\nabla_\xi \Delta_\xi Q + \nabla_\eta \Delta_\eta Q + \nabla_\zeta \Delta_\zeta Q\right) - \frac{\varepsilon_e^4 \Delta t}{J}\left[\left(\nabla_\xi \Delta_\xi\right)^2 Q + \left(\nabla_\eta \Delta_\eta\right)^2 Q + \left(\nabla_\zeta \Delta_\zeta\right)^2 Q\right] \quad (4.31)$$

The implicit artificial viscosity is implemented by adding the following terms to the implicit terms of Eqs. (4.25) to (4.27), respectively

$$-\frac{\varepsilon_i \Delta t}{J}\left[\nabla_\xi \Delta_\xi (J \Delta Q^*)\right] \tag{4.32a}$$

$$-\frac{\varepsilon_i \Delta t}{J}\left[\nabla_\eta \Delta_\eta (J \Delta Q^n)\right] \tag{4.32b}$$

$$-\frac{\varepsilon_i \Delta t}{J}\left[\nabla_\zeta \Delta_\zeta (J \Delta Q^n)\right] \tag{4.32c}$$

In Eq. (4.31), ε_e^2 and ε_e^4 are the second– and fourth–order explicit artificial viscosity coefficients, and in Eq. (4.32), ε_i is the implicit artificial viscosity coefficient.

The nonlinear coefficient artificial viscosity model is strictly explicit. Following the approach of Pulliam (1986), the following terms are added to the source term of Eq. (4.25).

$$\begin{aligned}
&\nabla_\xi \left\{ \left[\left(\frac{\phi}{J}\right)_{i+1} + \left(\frac{\phi}{J}\right)_i \right] \left(\varepsilon_\xi^2 \Delta_\xi Q - \varepsilon_\xi^4 \Delta_\xi \nabla_\xi \Delta_\xi Q \right)_i \right\} \\
&+ \nabla_\eta \left\{ \left[\left(\frac{\phi}{J}\right)_{j+1} + \left(\frac{\phi}{J}\right)_j \right] \left(\varepsilon_\eta^2 \Delta_\eta Q - \varepsilon_\eta^4 \Delta_\eta \nabla_\eta \Delta_\eta Q \right)_i \right\} \\
&+ \nabla_\zeta \left\{ \left[\left(\frac{\phi}{J}\right)_{k+1} + \left(\frac{\phi}{J}\right)_k \right] \left(\varepsilon_\zeta^2 \Delta_\zeta Q - \varepsilon_\zeta^4 \Delta_\zeta \nabla_\zeta \Delta_\zeta Q \right)_k \right\}
\end{aligned} \tag{4.33}$$

where ϕ is define as

$$\phi = \phi_x + \phi_y + \phi_z \tag{4.34}$$

and ϕ_x, ϕ_y and ϕ_z are spectral radii defined by

$$\begin{aligned}
\phi_x &= \frac{|\xi_x u + \xi_y v + \xi_z w| + a\sqrt{\xi_x^2 + \xi_y^2 + \xi_z^2}}{\Delta \xi} \\
\phi_y &= \frac{|\eta_x u + \eta_y v + \eta_z w| + a\sqrt{\eta_x^2 + \eta_y^2 + \eta_z^2}}{\Delta \eta} \\
\phi_z &= \frac{|\zeta_x u + \zeta_y v + \zeta_z w| + a\sqrt{\zeta_x^2 + \zeta_y^2 + \zeta_z^2}}{\Delta \zeta}
\end{aligned} \tag{4.35}$$

The parameters ε^2 and ε^4 are the second– and fourth–order artificial viscosity coefficients. Instead of being specified directly by the user, as they are in the constant coefficient model, in the nonlinear coefficient model they are a function of the pressure field. For coefficients of the ξ direction differences,

$$\begin{aligned}\left(\varepsilon_\xi^2\right)_i &= \varkappa_2 \Delta t \max(\sigma_{i+1}, \sigma_i, \sigma_{i-1}) \\ \left(\varepsilon_\xi^4\right)_i &= \max\left(0, \varkappa_4 \Delta t - \left(\varepsilon_\xi^2\right)_i\right)\end{aligned} \tag{4.36}$$

where

$$\sigma_i = \left| \frac{p_{i+1} - 2p_i + p_{i-1}}{p_{i+1} - 2p_i + p_{i-1}} \right| \tag{4.37}$$

Similar formulas are used for the coefficients of the η direction differences.

The parameter σ is a pressure–gradient scaling parameter that increases the amount of second–order smoothing relative to the fourth–order smoothing near shock waves. The logic used to compute ε^4 serves to switche off the fourth–order smoothing when the second–order smoothing term is large. The parameters $\varkappa_2$ and $\varkappa_4$ are user–specified constants, and the optimum values are problem dependent.

4.7 Turbulence Models

The Navier–Stokes equations Eq. (4.1) solved in this study are time–averaged; i.e., they are the Reynolds–averaged Navier–Stokes equations. Therefore, they do not contain enough information for turbulence to form a closed set of equations. To remedy this problem, the Reynolds stress and turbulent heat flux terms are modeled using the Boussinesq approach. The turbulent Reynolds stress resulting from time averaging is assumed proportional to the laminar stress tensor with the coefficients of proportionality defined as eddy viscosity, μ_t. Similarly, the turbulent Reynolds heat flux is assumed to be proportional to the laminar heat

flux with the coefficient of proportionality defined as turbulent thermal conductivity, k_t. An effective viscosity is thus defined as $\mu = \mu_l + \mu_t$, and an effective thermal conductivity coefficient is defined as $k = k_l + k_t$. μ_l is the laminar (or molecular) viscosity coefficient, and k_l is the molecular thermal conductivity coefficient. These turbulence coefficients are computed in this study using either a generalized version of the Baldwin and Lomax algebraic eddy viscosity model (1978), or the Chien (1982) model. The former is a two–layer algebraic model while the latter is a two–equation $k - \varepsilon$ model. The Chien formulation for the $k - \varepsilon$ model is chosen in particular because it is numerically stable and approximates the near wall region reasonably well. A detailed analysis of these two turbulence modelling methods is presented by Towne et al. (1990).

4.8 Boundary Conditions

Improper treatment of boundary and initial conditions can lead to serious errors and perhaps instability in the numerical solution. For the above solution algorithm, the boundary conditions are treated implicitly. Several types of boundaries that can be encountered in real problems are adequately provided for. Such boundaries can be real or artificial. Real boundaries include simple solid or porous surfaces and artificial boundaries can be far field boundaries or symmetry planes. Since the equations are solved by marching in time, a set of initial conditions (throughout the flow field) is also required to start the time marching procedure. For unsteady flows, they should represent a real flow field, and a converged steady state solution from a previous run or experimental results may be a good choice. For steady flows, the ideal initial conditions would represent a real flow field that is close to the expected final solution.

4.9 Concluding Remarks

The above formulations have been made for the 3–D planar form of Navier–Stokes equations in curvilinear coordinate system. Detailed analysis including formulation for axisymmetry forms can be found in Towne et al. (1992). Complete formulation for the 2–D version can also be found in Towne et al. (1990).

Chapter 5
MULTIGRID METHOD AND RESULTS

A multigrid procedure has been developed to accelerate the convergence of the Beam–Warming ADI numerical scheme formulated in previous chapter. The multigrid algorithm adopted is the Full Approximation Storage Full Multigrid method (FAS–FMG) which is applicable to nonlinear systems of equations. Test problems with different geometries and flow conditions are selected to validate the implementation. Practical convergence rates are compared with predicted rates from both the bi–grid and the single grid analyses for a 2–D problem.

5.1 Full Approximate Storage Full Multigrid (FAS–FMG)

Consider the problem

$$L^h(U^h) = f^h \tag{5.1}$$

where L^h is a non–linear operator on a grid, g^h, with spacing h. The forcing function, f, is known and U^h is the solution to the problem on the grid with spacing h. Taking u^h as an approximation to U^h with an error

$$V^h = U^h - u^h \tag{5.2}$$

Equation (5.1) can be expressed as

$$L^h(u^h + V^h) = f^h \tag{5.3}$$

$L^h u^h$ is subtracted from both sides of Eq. (5.3) to give

$$L^h(u^h + V^h) - L^h(u^h) = f^h - L^h(u^h) \tag{5.4}$$

If the terms are smooth, they can be represented on a coarser grid, g^{2h} with spacing $2h$. The grid g^{2h} is formed by deleting every other point in g^h; therefore, g^{2h} is a subset of g^h. Points are eliminated from g^{2h} to form g^{4h}, and so forth, to form g^{8h}, g^{16h} etc. Each subsequent grid is a subset of the previous grid, which places compatibility constraints on the number of grid points in each direction. On the coarse–grid, g^{2h}, Eq. (5.4) becomes

$$L^{2h}(I_h^{2h}u^h + V^{2h}) - L^{2h}(I_h^{2h}u^h) = I_h^{2h}(f^h - L^h(u^h)) \tag{5.5}$$

or

$$L^{2h}(U^{2h}) = f^{2h} \tag{5.6}$$

where

$$f^{2h} = I_h^{2h}(f^h - L^h(u^h)) + L^{2h}(I_h^{2h}u^h) \tag{5.7}$$

where I_h^{2h} is the restriction operator.

Since Eq. (5.6) is on a coarser grid than Eq. (5.1), the numerical solution for u^{2h} is much less expensive to obtain because fewer points are involved. Note that the operator used on the coarse–grid has the same form as the fine–grid operator, the grid spacing (h and $2h$) being the only difference. Once the values of u^{2h} are obtained, the fine–grid iterative solution is updated using the following equation:

$$(u^h)_{New} = (u^h)_{Old} + I_{2h}^h\left[u^{2h} - I_h^{2h}(u^h)_{Old}\right] \tag{5.8}$$

and I_{2h}^h is the prolongation operator.

A grid with spacing $4h$ can then be used to find corrections to the "solution" of the problem on the grid with spacing $2h$. Successively coarser grids may be used until a grid is reached which is so coarse that a direct solution may be used (or a nearly exact solution with only a small number of iteration sweeps). The correction from the coarsest grid is then used to correct the

correction on the next finer grid; and this is continued through successively finer grids until the finest level is reached and the approximate solution is updated.

The usefulness of corrections obtained on a coarser grid is dependent on the smoothness of the fine–grid error passed to the coarse–grid. Hence, it is absolutely necessary that the high–frequency components of the error on the fine–grid be minimized, if not completely eliminated. It is the responsibility of the smoother to dampen out the high frequency components of the error. The removal of the low–frequency components of the error is unimportant for all but the coarsest grid since these frequencies can be resolved on the coarser grids where they become high frequencies. If the high frequencies are not damped, then the restriction operator will pass aliased information to the coarser grid and the entire multigrid scheme will cease to converge. Obviously, the choice of the smoother is critical to the proper functioning of multigrid. Some smoothers are naturally effective and some have to be modified. For instance Elmiligui (1992) has developed coefficients for Runge–Kutta multistage time–stepping scheme such that good high frequency damping can be achieved at relatively high CFL numbers. In the present work more accurate analytical tools, presented in Chaps. 2 and 3, have provided insight into the effectiveness of the smoother used.

The cycle of work performed, starting on the finest grid, successively using the coarser grids, and then returning to the finest grid is called one multigrid cycle. The cycles are repeated until sufficient convergence is obtained on the finest grid. Examples of popular multigrid cycles are sketched in Appendix E.

The restriction operator has two forms. One form is used to restrict the dependent variables, $I_h^{2h}(u^h)$; i.e., the flow field quantities $\varrho, \varrho u, \varrho v,$ and ϱe_o and the other form is used for the restriction of residuals, $I_h^{2h}\left[L^h(u^h)\right]$. In this study, the injection method is used in the former and the full weighting technique is used in the latter. The prolongation operation, I_{2h}^h, used in the current work is a bi–linear interpolation for 2–D, and tri–linear interpolation for 3–D.

With this general discussion of the FAS–FMG algorithm in focus, its application to the formulated ADI scheme becomes clearer.

5.2 Multigrid Application to the ADI scheme

The 3–D ADI scheme can be written in the operator form:

$$L_1 L_2 L_3 \Delta Q = -\Delta t R \tag{5.9}$$

where L_1, L_2 and L_3 are operators to represent the implicit ADI factors, and ΔQ, Δt and R are, as usual, flow field corrections, time step and residual. A simple multigrid cycle is performed as follows:

(1) Solve Eq. (5.9) on the finest grid g^h, i.e.,

$$L_1^h L_2^h L_3^h \Delta Q^h = -\Delta t R^h \tag{5.10}$$

(2) Compute the flow field at the present time step.

$$(Q^{n+1})^h = (Q^n)^h + (\Delta W^n)^h \tag{5.11}$$

(3) Compute the residual R^h from the right–hand side of Eq. (4.22),

(4) Transfer the flow variables Q^h and residual R^h to grid g^{2h} and compute the forcing terms P^{2h} as follows:

$$P^{2h} = I_h^{2h} R^h - R^{2h}(Q^{2h}) \tag{5.12}$$

(5) Then, the coarse grid g^{2h} problem, driven by the forcing terms, becomes:

$$L_1^{2h} L_2^{2h} L_3^{2h} \Delta Q^{2h} = -\Delta t\left(R^{2h}(Q^{2h}) + I_h^{2h} R^h - R^{2h}(I_h^{2h} Q^h)\right) \tag{5.13}$$

(6) Steps (3)–(5) is repeated recursively on g^{4h}, g^{8h}, etc. to solve for the corrections, and the cumulative corrections are interpolated on to the fine grid and added to the solution, until the finest grid g^h, for example:

$$Q^{2h} \leftarrow Q^{2h} + I_{4h}^{2h}(Q^{4h} - I_{2h}^{4h}Q^{2h}) \tag{5.14}$$

$$Q^{h} \leftarrow Q^{h} + I_{2h}^{h}(Q^{2h} - I_{h}^{2h}Q^{h}) \tag{5.15}$$

The above steps are repeated on the fine grid until $R^h = 0$ or a predetermined tolerance level, since we are interested only in steady state solutions. Depending on the schedule for the grids and the order in which they are visited, the multigrid structure may take the form of a simple V–Cycle or more complicated ones like the full multigrid V–Cycle or the W–cycle. See Appendix E for some samples. In the present study, the full multigrid V–Cycle is utilized throughout.

In general one iteration is performed on the finest grid, one on intermediate grids and four on the coarsest grid. But on coarser grids, the computational work is successively reduced by a factor of 4 for 2–D and by a factor of 8 for 3–D. This implies that the computational work units (relative to a single grid iteration on the finest grid) for a three–level multigrid cycle is $1\frac{3}{4}$ for 2–D and $1\frac{5}{16}$ for 3–D. Additional overheads for intergrid transfers, generation of grids and calculation of metrics on coarser grids, calculation of residuals, etc., raise the total work units to about $2\frac{5}{8}$ and 2 respectively, for 2–D and 3–D multigrid procedures. That is, the multigrid convergence rate must be faster by at least these factors for there to be a saving in CPU time. One method for reducing the relative overhead of the multigrid procedure is to perform more fine grid iterations per multigrid (MG) cycle, e.g., three iterations on the finest grid instead on one. Then the effective total work units per MG cycle are reduced to 2 and $1\frac{5}{8}$, respectively. Preliminary tests showed that this latter approach is better for 2–D

computations whereas, the former approach is better for the 3–D computations. In the 2–D case, the slower convergence rate resulting from performing a MG cycle only every third fine–grid iteration is more than compensated for by the cheaper overall MG overhead, whereas in the 3–D case, where the MG overhead costs are lower, the faster convergence rate of performing one MG cycle per fine–grid iteration is dominant. These approaches are mostly used in the results presented in subsequent sections.

5.3 2–D Multigrid Solutions

5.3.1 Test Problems

Various test cases shown in Table 5.1 were investigated to validate the implementation of the multigrid cycles in the 2–D version of the Proteus code.

The geometry for the various cases are shown in Fig. 5.1. The first two test cases are the steady, inviscid and viscous flow past a two–dimensional circular cylinder. Inviscid and viscous solutions were obtained at different reference Mach numbers ranging from 0.05 to 0.6. A reference Reynolds number of 20 is assumed for the viscous case. For the Euler inviscid flow, freestream conditions are prescribed at the far field which are then used to start the computation. The exact potential flow solution is used to start the viscous flow computations. The next two test cases are turbulent flow over a flat plate at zero pressure gradient with a freestream Mach number of 0.3. The Baldwin–Lomax turbulence model was used in the first case while the Chien $k - \varepsilon$ model was used in the second case.

Table 5.1: Description of Test Cases for 2–D

Test Case	Flow Problem		Coarse grid	Fine grid
1	Inviscid	Cylinder	25X49	49X97
2	Viscous		49X49	97X97
3	B/L	Flat Plate	81X53	161X105
4	Chien		81X53	161X105
5	Non–Linear	Transonic	81X51	–
6	Constant		81X51	–

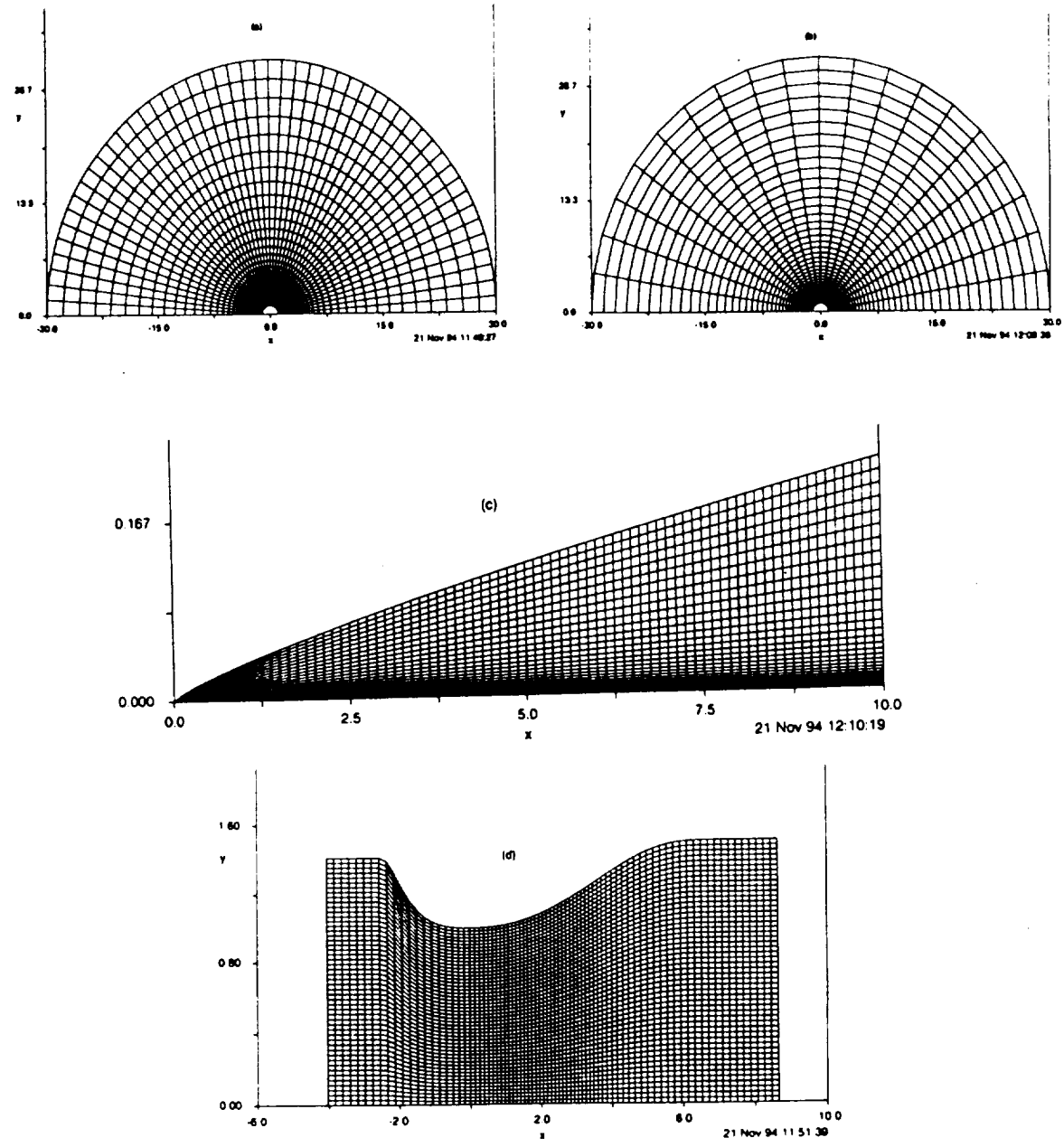


Fig. 5.1: Computational mesh for the test cases (a) Viscous flow around a cylinder (b) Euler flow around a cylinder (c) turbulent flow over a flat plate (d) Sajben transonic flow.

The Blasius solution for a laminar boundary layer over a flat plate was used to set the initial conditions for the turbulent flow computations with the Baldwin–Lomax model. To start the computations in the case with the Chien $k - \varepsilon$ model, the converged solutions from the Baldwin–Lomax model were used to compute the turbulent quantities required to march the $k - \varepsilon$ equations. The last test case is a transonic turbulent flow in a converging–diverging duct. The flow entered the duct subsonically, was accelerated through the throat to supersonic speed, then decelerated through a normal shock and exited the duct subsonically. This is a popular laboratory test case which can mimic real inlet flows; e.g., see Sajben et. al. (1984). The geometry is obtained from the following equations:

$$y = \begin{cases} 1.4144 & \text{for } -4.04 \le x \le -2.598 \\ \alpha \cosh\xi/(\alpha - 1 + \cosh\xi) & \text{for } -2.598 < x < 7.216 \\ 1.5 & \text{for } 7.216 \le x \le 8.65 \end{cases} \tag{5.16}$$

where the parameter ξ is defined as

$$\xi = \frac{C_1(x/x_l)\left[1 + C_2 x/x_l\right]^{C_3}}{(1 - x/x_l)^{C_4}} \tag{5.17}$$

The various constants used in the formula for the top wall height in the converging $-2.598 \le x \le 0$ and diverging $(0 \le x \le 7.216)$ parts of the duct are given in Table 5.2.

In all cases, appropriate boundary conditions were set so as not to impair the convergence acceleration. The steady–state solutions of the various test cases described above are shown in Fig. 5.2.

Table 5.2: Constants for Computational Coordinates calculations

Constant	*Converging*	*Diverging*
α	1 . 4114	1 . 500
x_l	− 2 . 5985	7 . 216
C_1	0 . 8100	2 . 250
C_2	1 . 0000	0 . 000
C_3	0 . 5000	0 . 000
C_4	0 . 6000	0 . 600

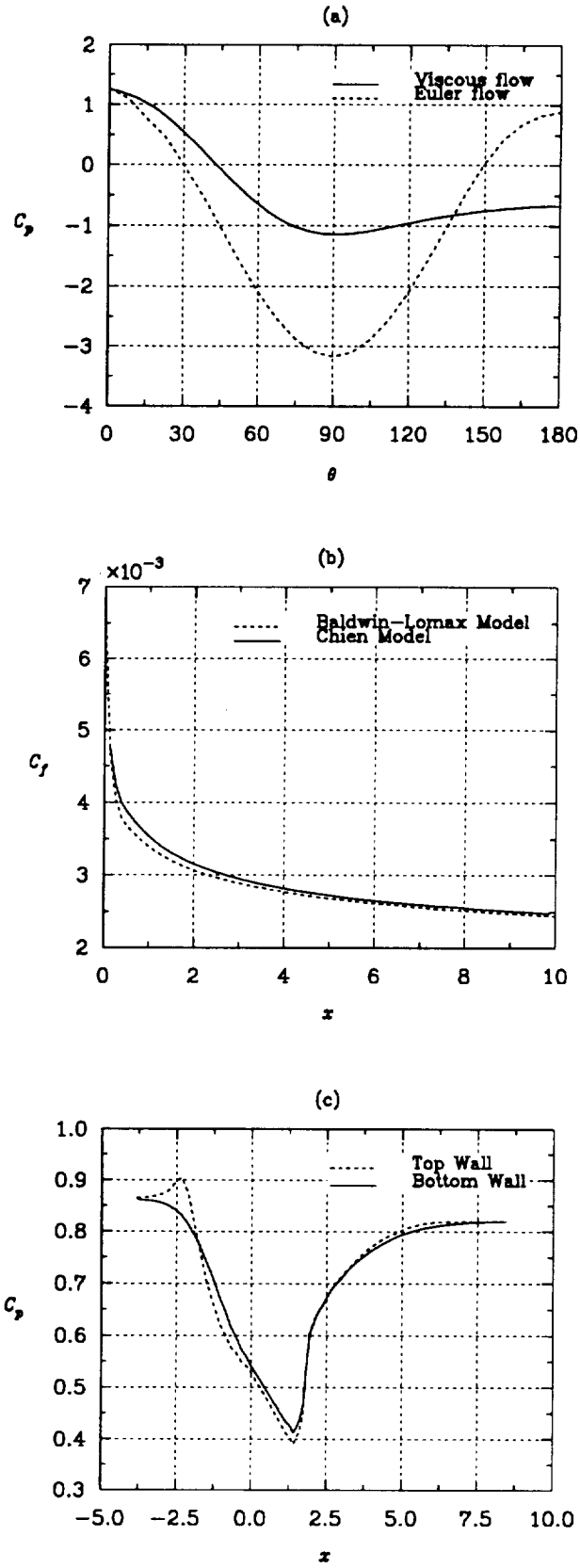


Fig. 5.2: Steady solutions for test cases (a)–(b) C_p for Euler and viscous flow past a cylinder (c) C_f for turbulent flow over a flat plate (d) C_p for Sajben transonic case.

5.3.2 Multigrid Performance

In order to evaluate the multigrid performance implemented in this study, the l_2 norm of the residual for the continuity and momentum equations are plotted against the number of iterations. For each of the coarse grids and the fine grid, three calculations are performed; single grid, two–level multigrid (MG) and 3–level MG. Figs. 5.3 and 5.4 show the result of the Euler flow at $M_\infty = 0.2$ where MG reduced the number of iterations by a factor of about two. However, no gain is observed in the CPU saving (see Table 5.3). The additional work unit required in this case by MG cycles offset the gain in convergence. Viscous flow results are shown in Figs. 5.5–5.10 for different freestream Mach numbers. At $M_\infty = 0.2$, similar conclusions drawn for the Euler above are observed. At $M_\infty = 0.05$ and $M_\infty = 0.6$, however, faster convergence and substantial savings in CPU time are obtained especially on the fine grid. For instance, at $M_\infty = 0.6$, three–level MG at a CFL number of four reduced the number of iterations by a factor of about five on the fine grid, at a measured savings of about 62% of the CPU time. In this case also, multigrid successfully relaxes the stiffness per time–step since the single grid computation failed if the CFL number is greater than two, while the MG can go up to four. These results are summarized in Table 5.4.

The influence of turbulence models on the performance of multigrid is studied in cases three and four. Figs. 5.11–5.12 show the convergence history of the three calculations mentioned above for turbulent flow over a flat plate using the Baldwin–Lomax model to compute the turbulence quantities. Convergence is accelerated by more than a factor of two both for the coarse grid and fine grid. The measured saving in CPU time in this case is about 20% for the fine grid and only about 10% for the coarse grid. The superiority of MG is finally demonstrated in Figs. 5.13–5.14 where the Chien $k - \varepsilon$ model was used for turbulence closure. In fact the single grid failed to converge even after a very large number of iterations (10,000 for the coarse grid and about 9,000 for the fine grid), whereas the MG converged in about 4,000 cycles. Further, the single grid computation was limited to a CFL number below

Table 5.3: Multigrid Performance of 2–D Inviscid Test Problems

Grid	Level	CFL	Iter	CPU Time (s)	Speedup Factor	Remark
	1	10	260	7.9		Optimal cfl
Coarse	2	8	160	9.9	0.80	Optimal cfl
	3	6	170	13.1	0.60	
	1	10	420	28.6		
Fine	2	10	230	29.6	0.97	
	3	10	240	35.9	0.80	

Table 5.4: Multigrid Performance of 2–D Viscous Test Problems

Case	Grid	Level	CFL	Iter	CPU Time (s)	Speedup Factor
M=0.2	Coarse	1	10	340	17.5	
		2	10	230	23.0	0.76
		3	10	240	28.2	0.62
	Fine	1	10	710	103.7	
		2	10	390	104.8	.99
		3	10	330	97.6	1.06
M=0.6	Coarse	1	2	1460	72.3	
		2	2	720	69.7	1.03
		3	2	780	86.2	0.83
		2	3	510	49.6	1.45
		3	4	750	83.3	0.87
	Fine	1	2	3440	492.8	
		2	2	1710	448.7	1.09
		3	2	1430	403.1	1.22
		2	3	1140	299.0	1.65
		3	4	680	194.0	2.54
M=0.05	Coarse	1	10	1270	62.6	
		2	10	760	73.5	0.85
		3	10	780	86.5	0.72
	Fine	1	10	2510	361.3	
		2	10	1280	336.7	1.07
		3	10	1100	312.5	1.16

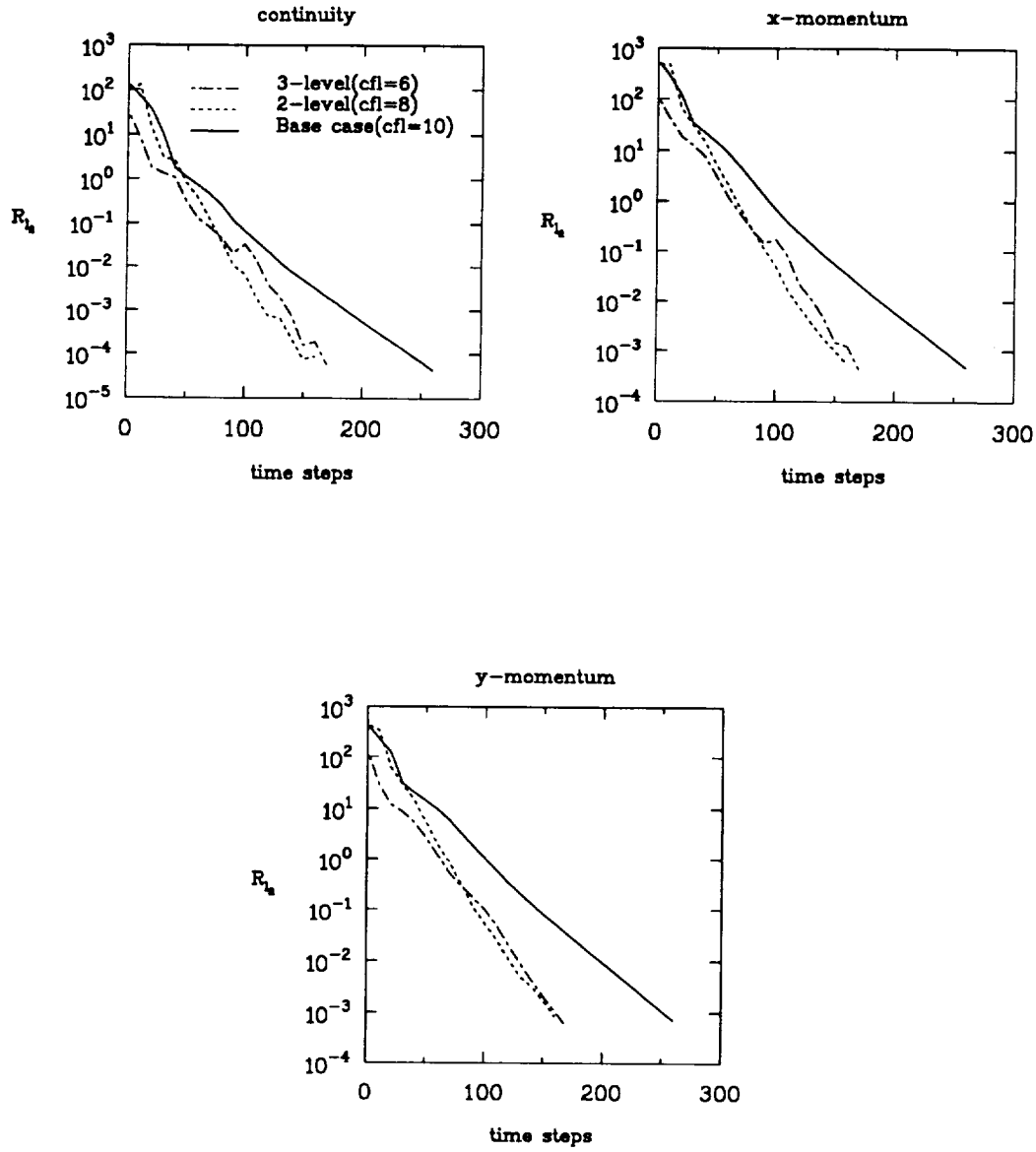


Fig. 5.3: R_{l_2} Convergence History for Euler flow past a circular cylinder at M_∞ = 0.2 and CFL = 10; 25X49 coarse case grid

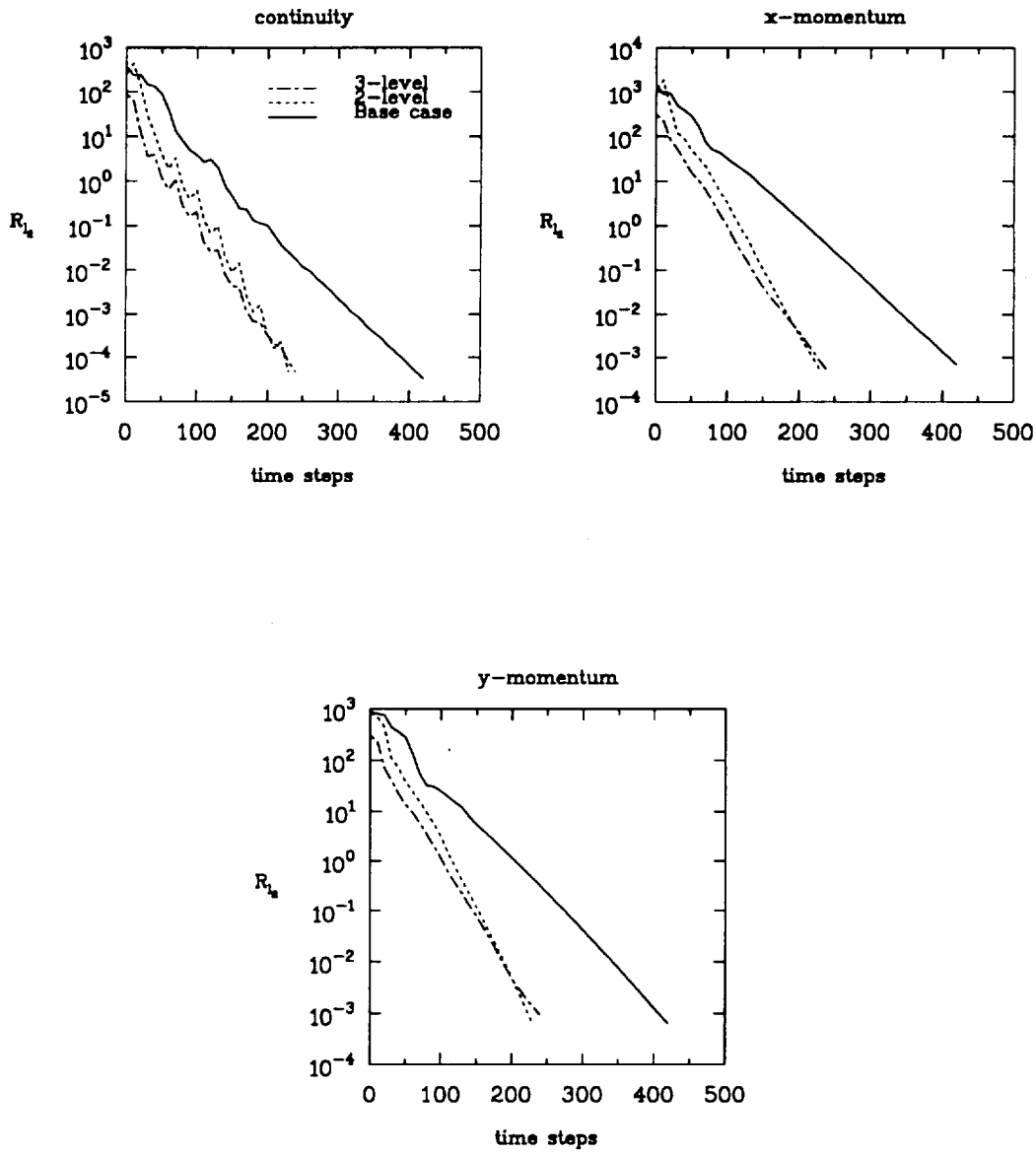


Fig. 5.4: R_{l_2} Convergence History for Euler flow past a circular cylinder at M_∞ = 0.2 and CFL = 10; 49X97 fine grid.

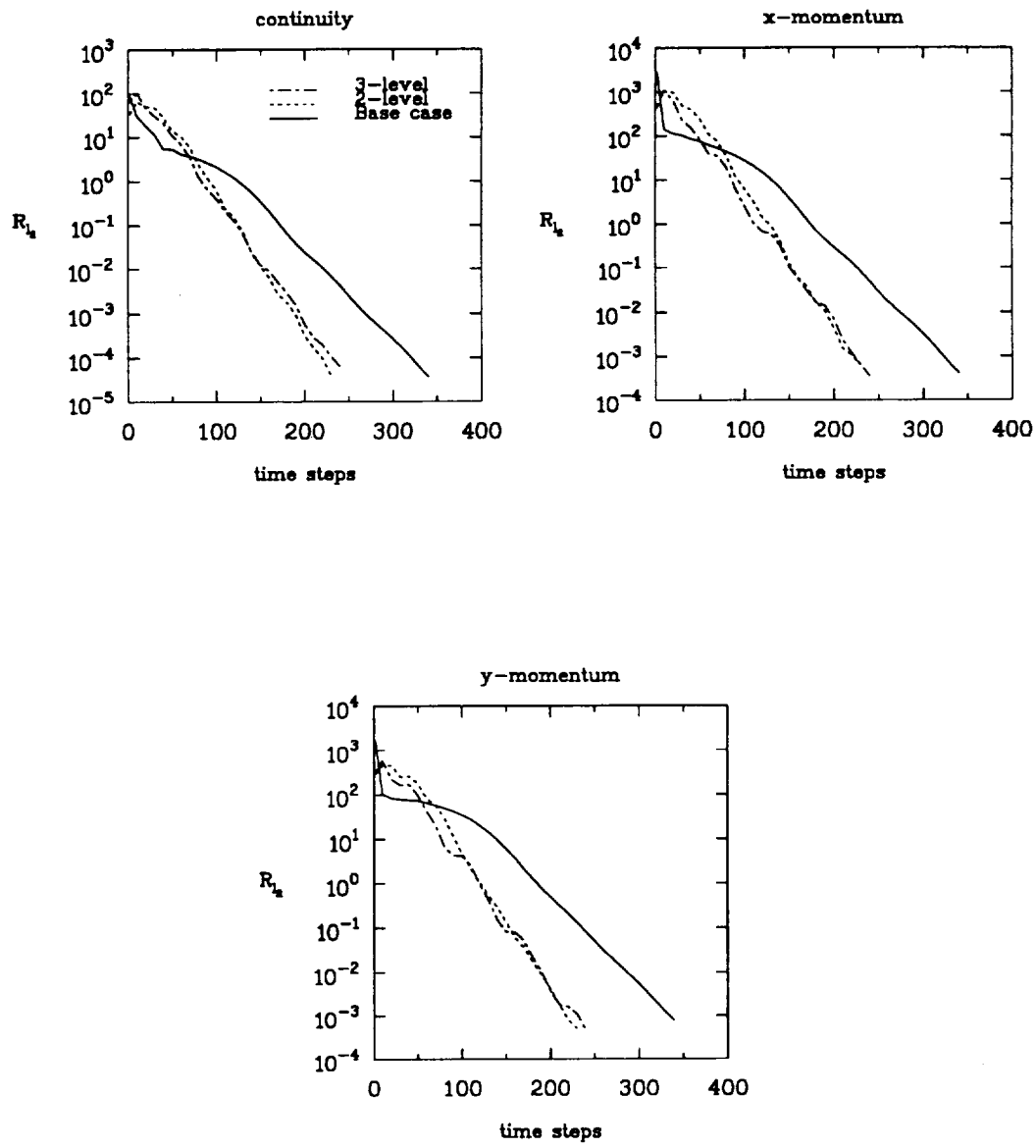


Fig. 5.5: R_{l_2} Convergence History for viscous flow past a circular cylinder at M_∞ = 0.2 and CFL = 10; 49X49 coarse case grid.

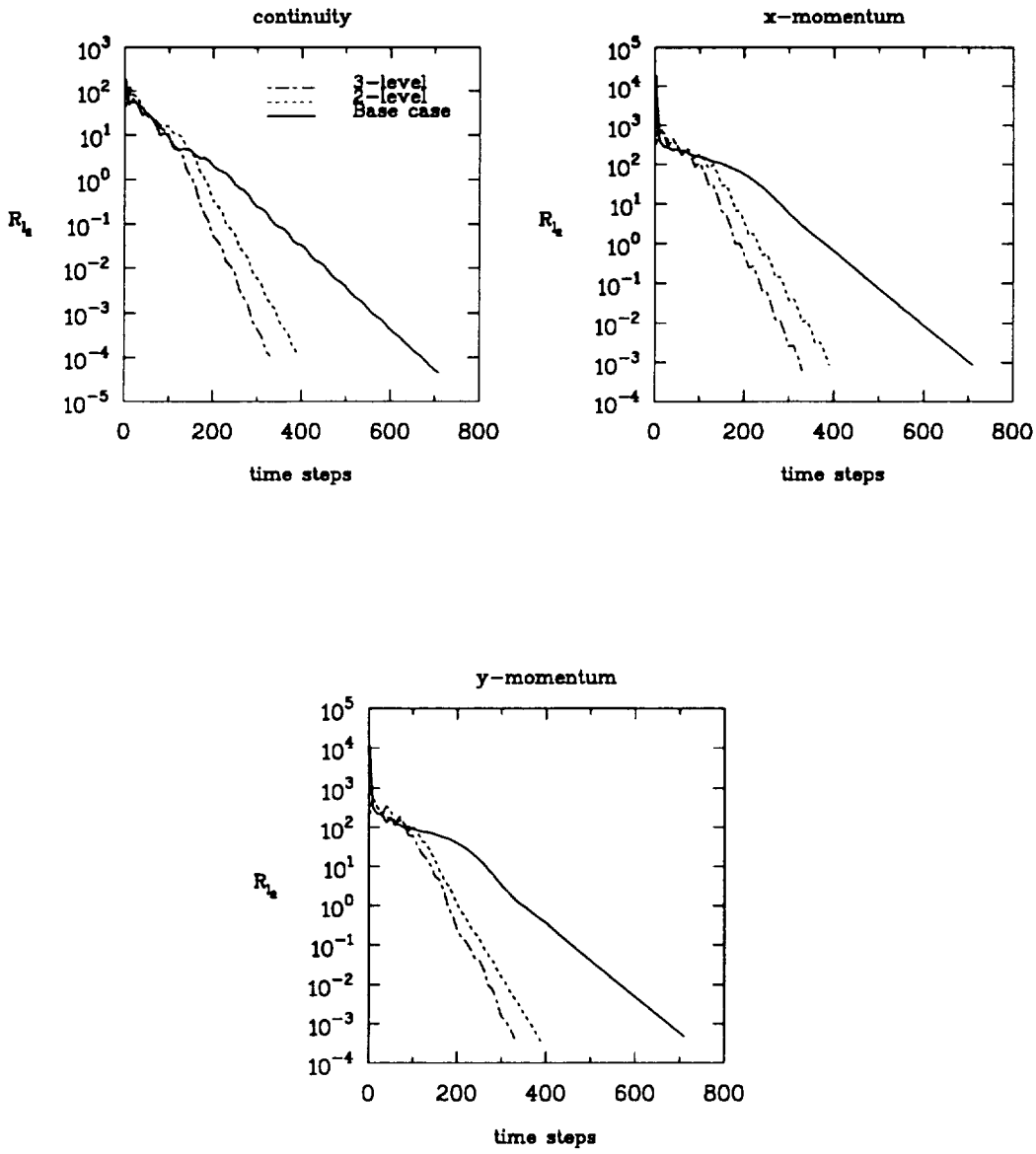


Fig. 5.6: R_{l_2} Convergence History for viscous flow past a circular cylinder at M_∞ = 0.2 and CFL = 10; 97X97 fine grid.

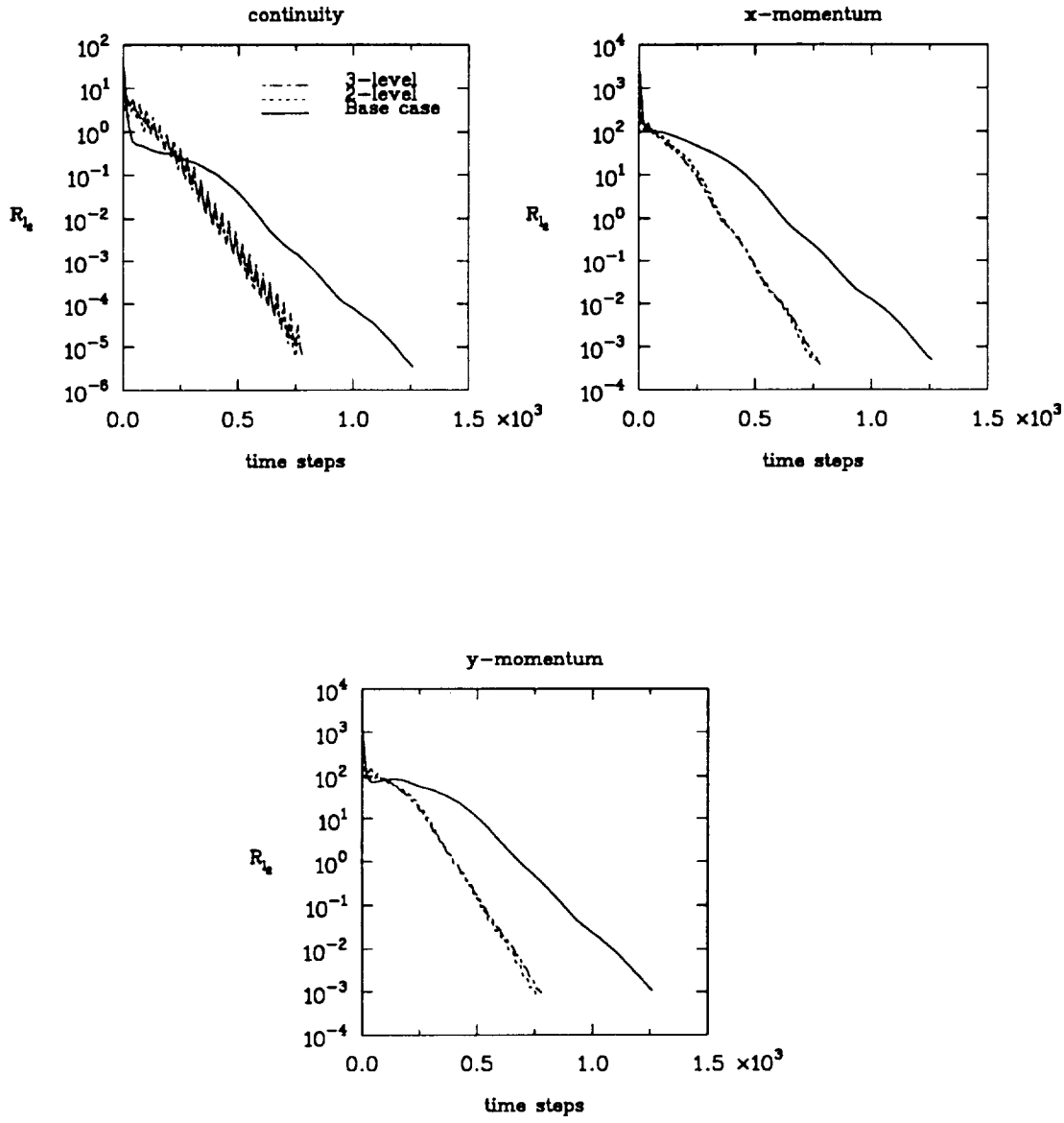


Fig. 5.7: R_{l_2} Convergence History for viscous flow past a circular cylinder at M_∞ = 0.05 and CFL = 10; 49X49 coarse case grid.

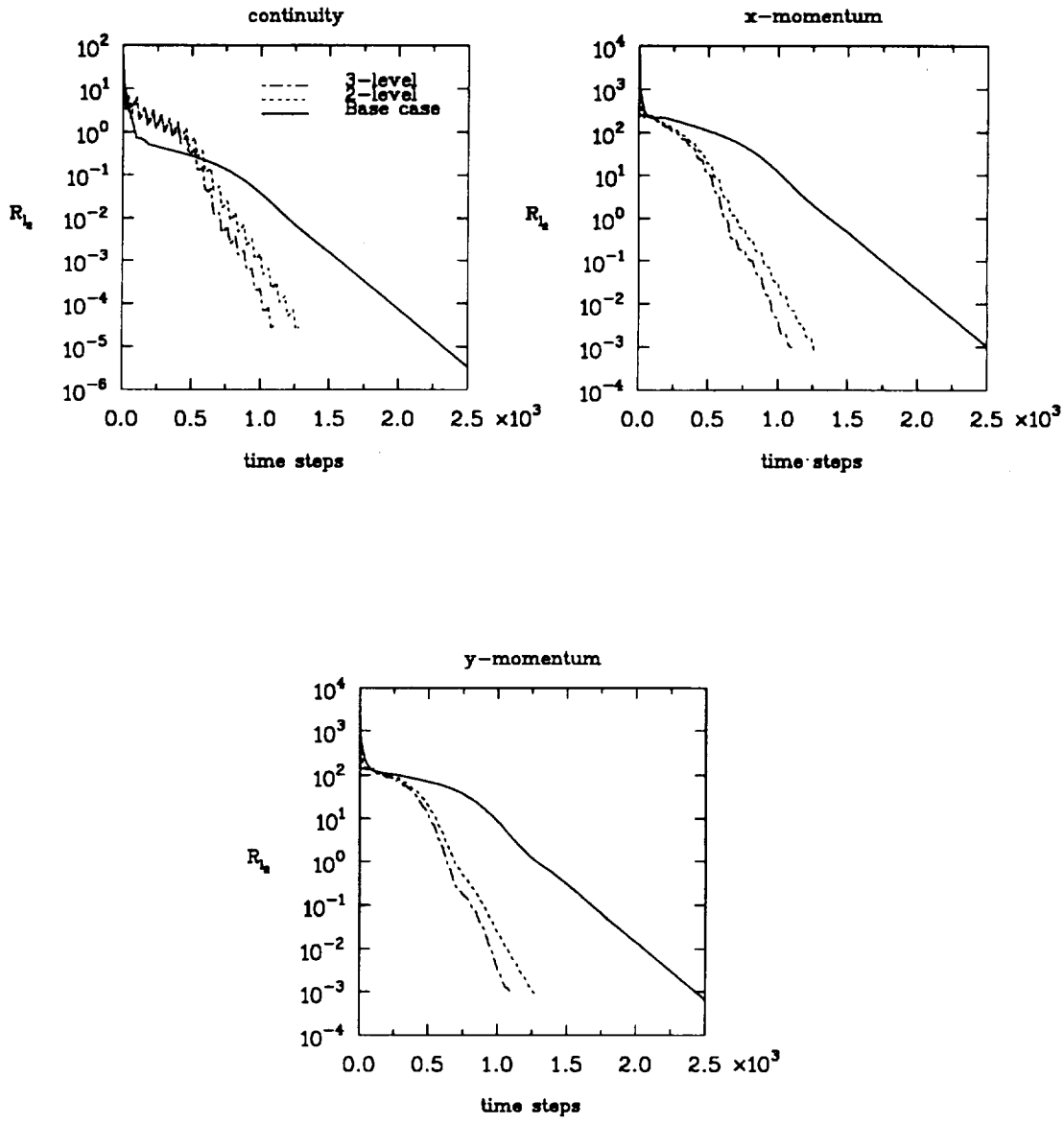


Fig. 5.8: R_{l_2} Convergence History for viscous flow past a circular cylinder at M_∞ = 0.05 and CFL = 10; 97X97 fine grid.

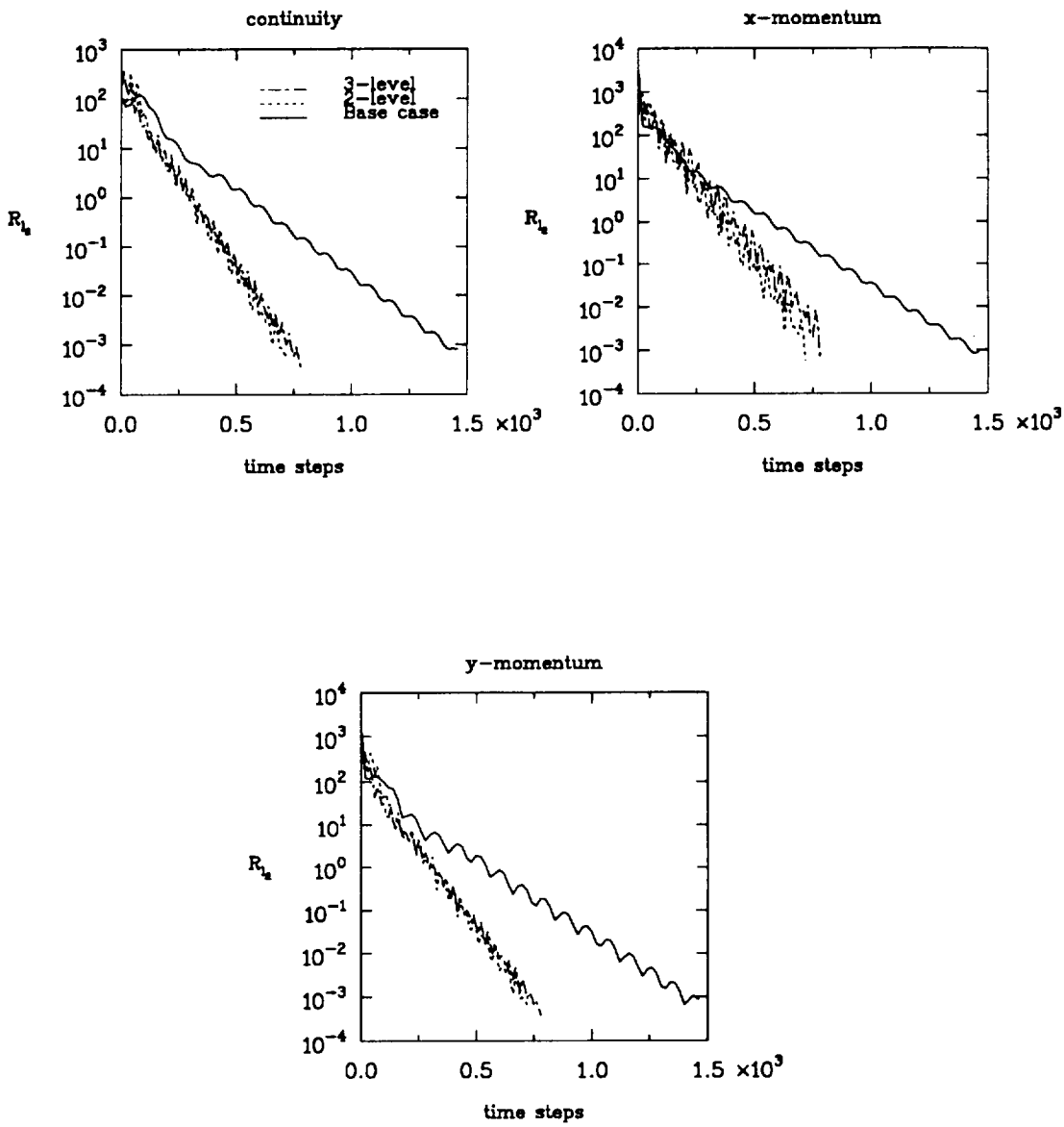


Fig. 5.9: R_{l_2} Convergence History for viscous flow past a circular cylinder at M_∞ = 0.6 and CFL = 10; 49X49 coarse case grid.

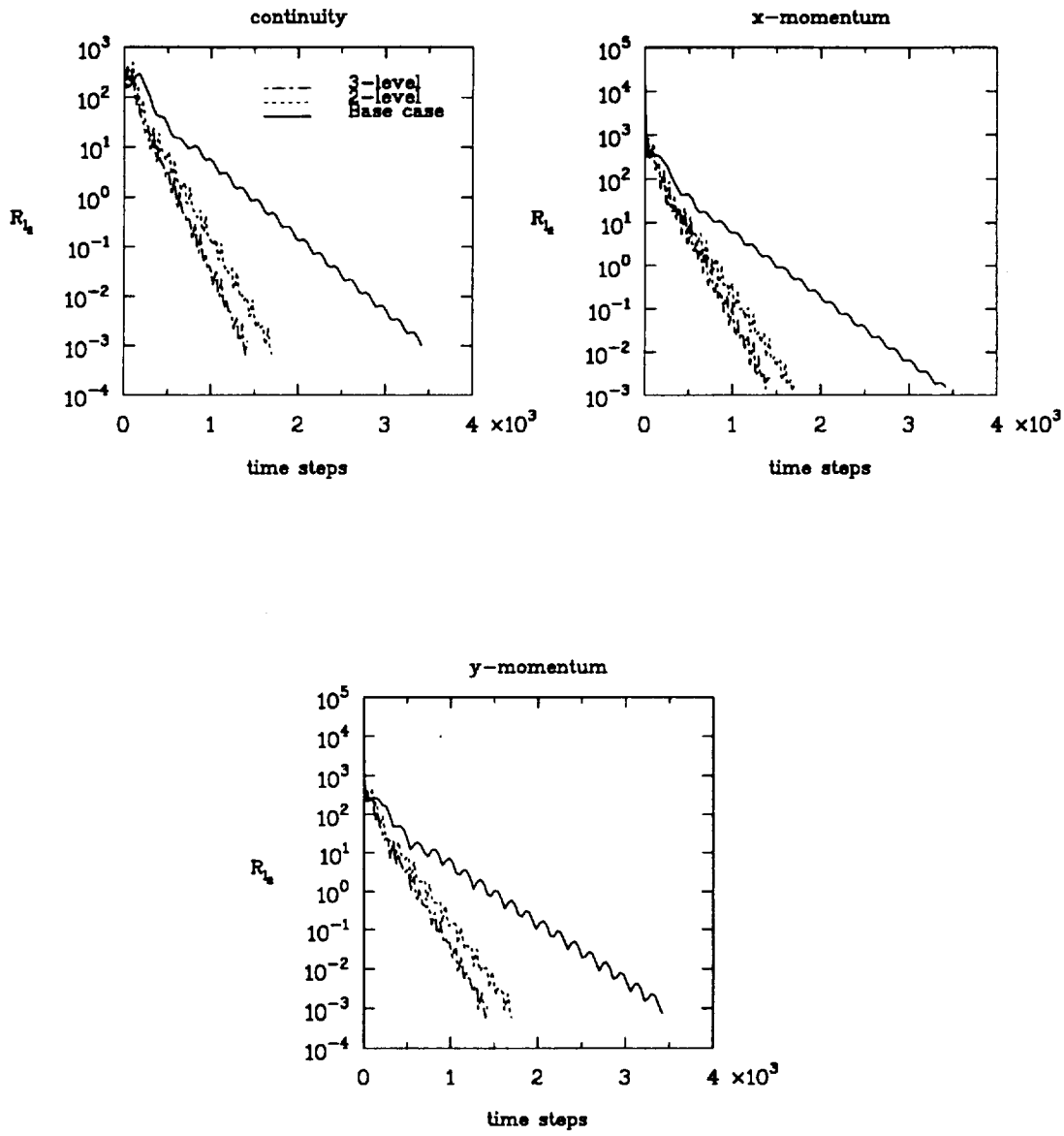


Fig. 5.10: R_{l_2} Convergence History for viscous flow past a circular cylinder at M_∞ = 0.6 and CFL = 10; 97X97 fine grid.

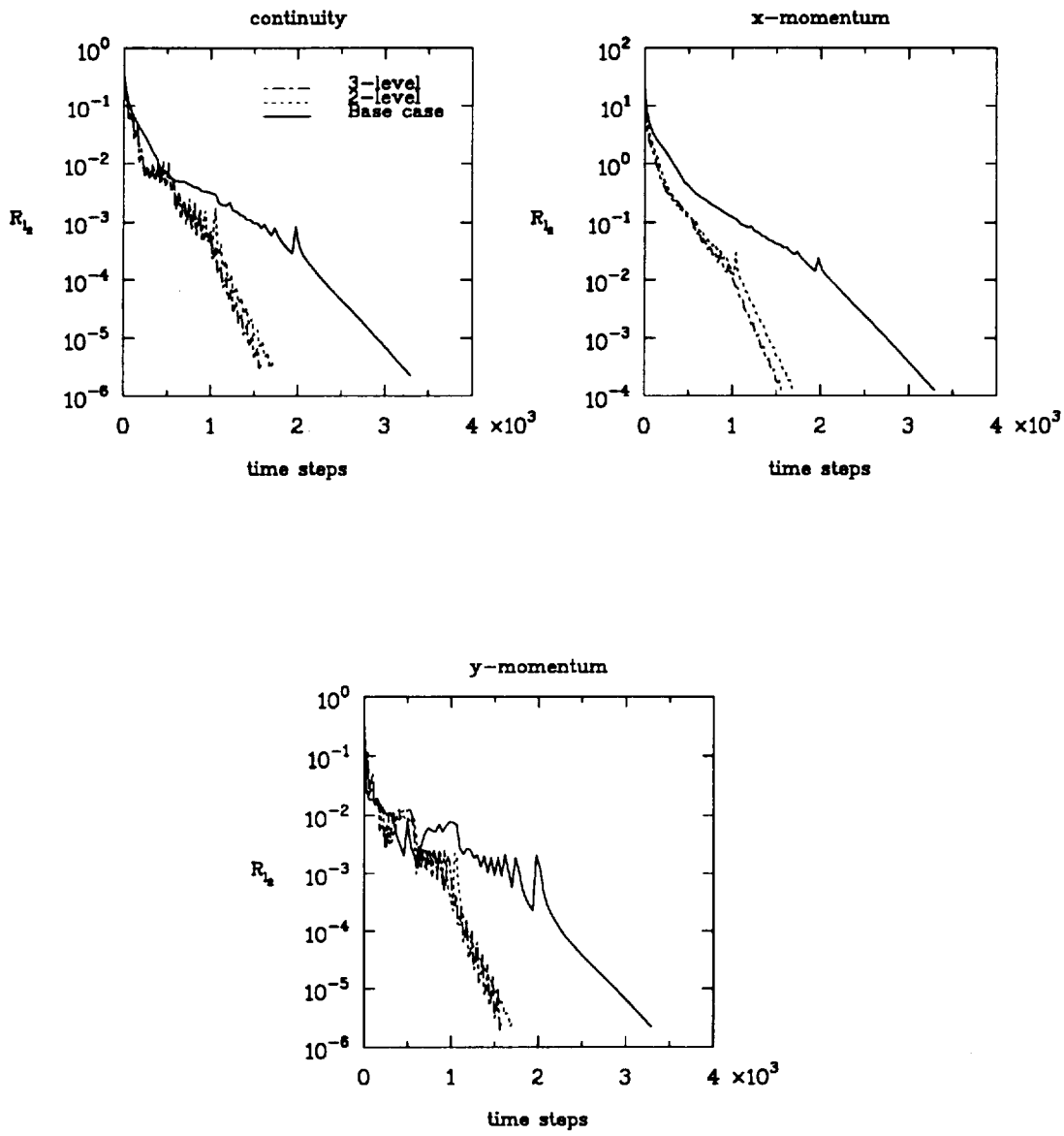


Fig. 5.11: R_{l_2} Convergence History for flat plate turbulent flow with Baldwin–Lomax model and CFL = 20; 81X53 coarse case grid.

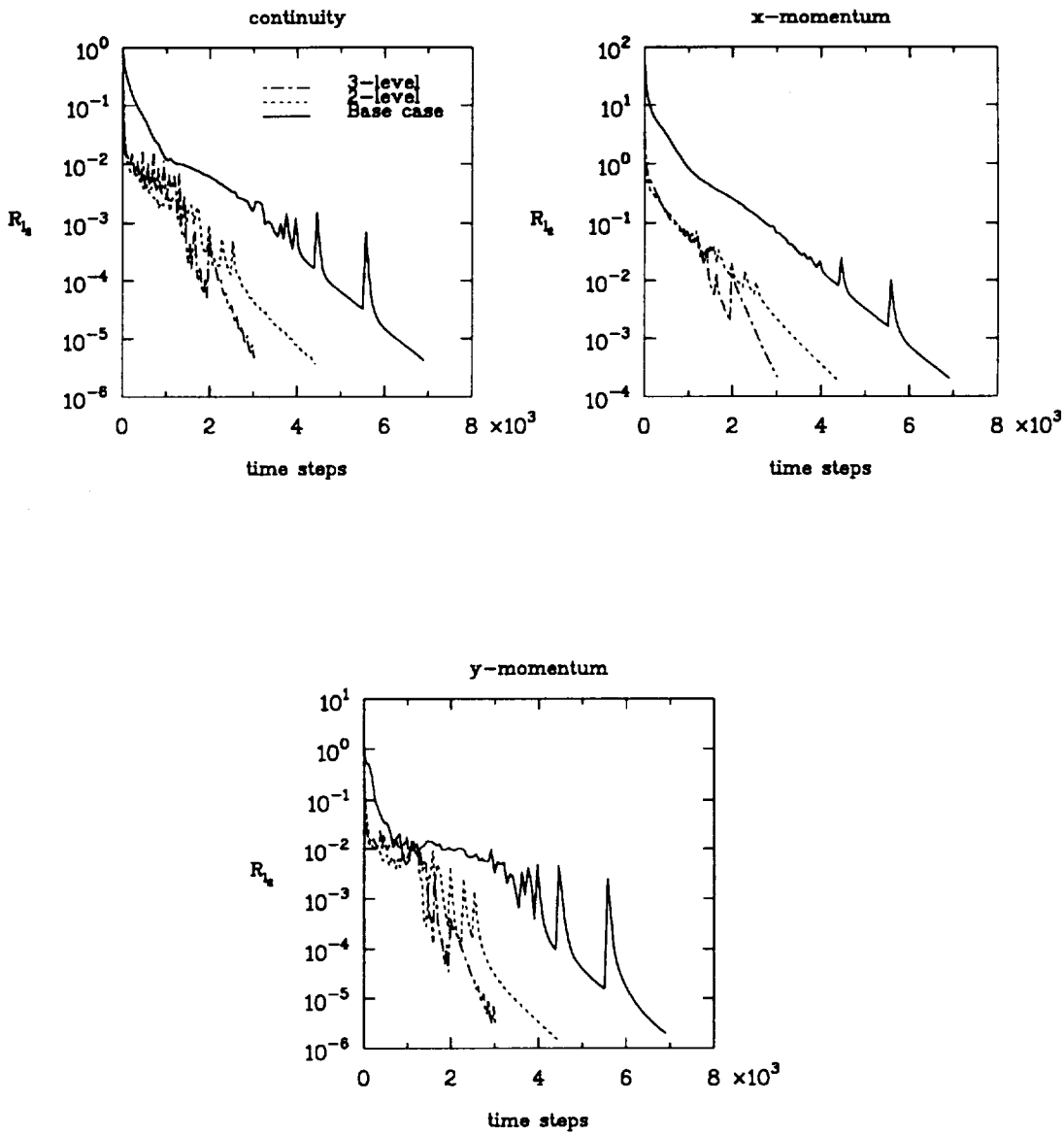


Fig. 5.12: R_{l_2} Convergence History for flat plate turbulent flow with Baldwin–Lomax model and CFL = 20; 161X105 fine grid.

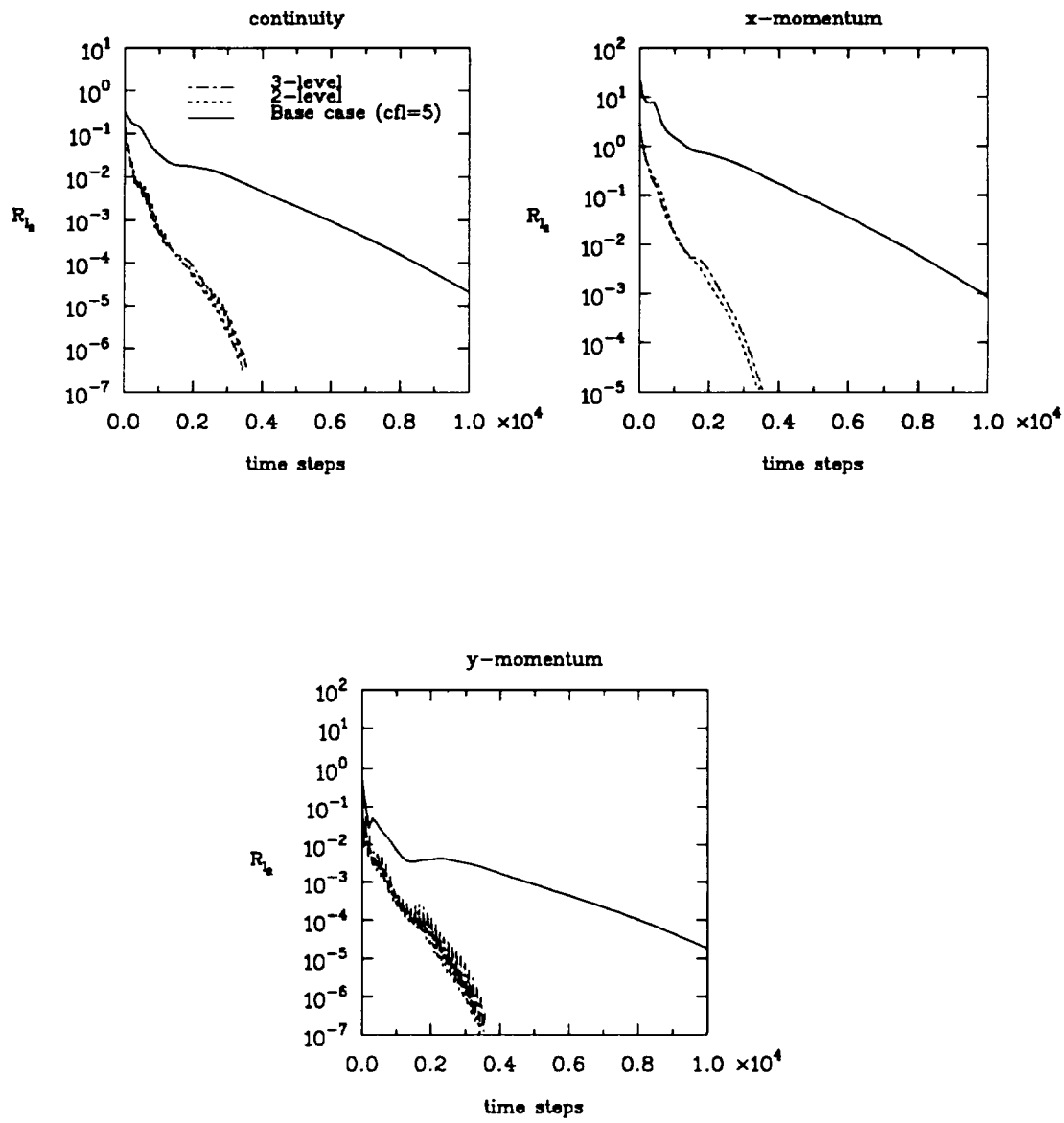


Fig. 5.13: R_{l_2} Convergence History for flat plate turbulent flow with Chien k–ε model and CFL = 20; 81X53 coarse case grid.

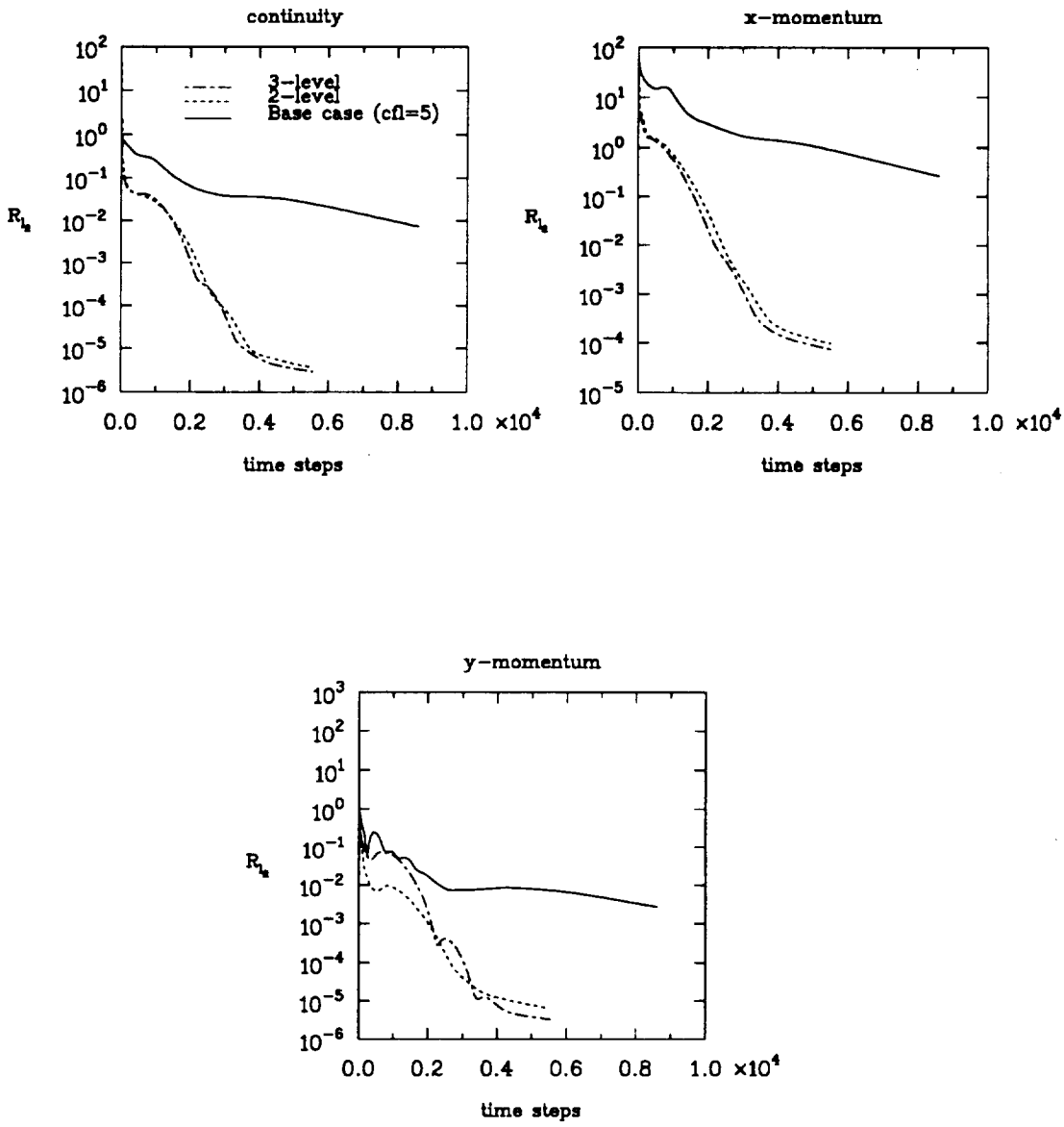


Fig. 5.14: R_{l_2} Convergence History for flat plate turbulent flow with Chien k−ϵ model and CFL = 20; 161X105 fine grid.

five, while the MG was successful up to a CFL number of 20. Although MG acceleration is clearly demonstrated in each of the models, it performed much better with the Chien model. The additional stiffness in the latter introduced by solving the $k - \varepsilon$ equations is by far offset by its better predictions of the Reynolds stresses. Tables 5.5 summarizes these resutls.

From Fig. 5.15, the trace of the skin friction at a point in the flow domain shows that the Sajben transonic flow is inherently unsteady. This is further confirmed in Figs. 5.16–5.17 where the convergence history oscillates for both the single grid and two–level MG calculations. In Fig. 5.16, the Jameson type of non–linear dissipation has been used whereas in Fig. 5.17, constant dissipation was assumed. Similar results (not shown) were observed on the finer grid. This inherent unsteadiness has also been observed from experiment by Sajben (1984) and therefore cannot be solved with the multigrid technique developed here. Multigrid methods for unsteady problems need to be implemented (see Ibraheem, 1994).

5.3.3 Convergence Rates

In the above computations, the results of both the single grid and the bi–grid stability analyses are continuously used as guidelines. For instance, the asymptotic convergence rates (using Eq. (3.28)) that are computed from practical multigrid solutions of the test cases for the inviscid and viscous flows past a circular cylinder are compared with the predictions from analysis in Fig. 5.18. Rather than evaluating the corresponding bi–grid and smoothing factors from uniform flow conditions, however, as performed in the analysis presented in Chaps. 2 and 3, they are computed at each point in the flow field, thereby accounting for the variation in flow properties. Figures 5.18(a) and 5.18(b) show estimates from both analyses based on the computed frozen coefficients of the inviscid and viscous flows, respectively. These results are also summarized in Table 5.6, and are compared with the asymptotic convergence rate measured from the practical multigrid computations. For both flow problems, the smoothing factor deviates more from the practical solution than does the bi–grid factor.

Table 5.5:Multigrid Performance of 2–D Flat Plate Test Problems

Case	Grid	Level	CFL	Iter	CPU Time (s)	Speedup Factor	Remarks
B/L	Coarse	1	20	3320	283.2		
		2	20	1730	268.8	1.05	
		3	20	1590	271.2	1.04	
	Fine	1	20	6950	1827.0		
		2	20	3690	1725.0	1.05	
		3	20	3060	1575.0	1.16	
Chien	Coarse	1	5	>10000	>1180.0		Max. cfl
		2	20	3470	650.0	>1.81	
		3	20	3600	732.0	>1.61	
	Fine	1	5	>8680	>3422.0		Max. cfl
		2	20	5540	3298.0	>1.03	
		3	20	5580	3398.0	>1.03	

Table 5.6: Convergence Characteristics of 2–D Euler and Viscous Flows around a Cylinder

CFL	Euler			Viscous flow		
	λ_{μ_sg}	$\lambda_{\max\_bg}$	ϱ_{mg}	λ_{μ_sg}	$\lambda_{\max\_bg}$	ϱ_{mg}
0.5	0.88	0.94	0.99	0.95	0.96	0.99
1.0	0.80	0.92	0.98	0.91	0.94	0.98
2.0	0.76	0.91	0.96	0.85	0.93	0.96
4.0	0.81	0.90	0.93	0.77	0.92	0.94
6.0	0.84	0.90	0.92	0.76	0.91	0.93
8.0	0.87	0.90	0.92	0.81	0.91	0.92
10.0	0.89	0.90	0.94	0.84	0.91	0.92
12.0	0.91	0.90	0.95	0.87	0.91	0.92

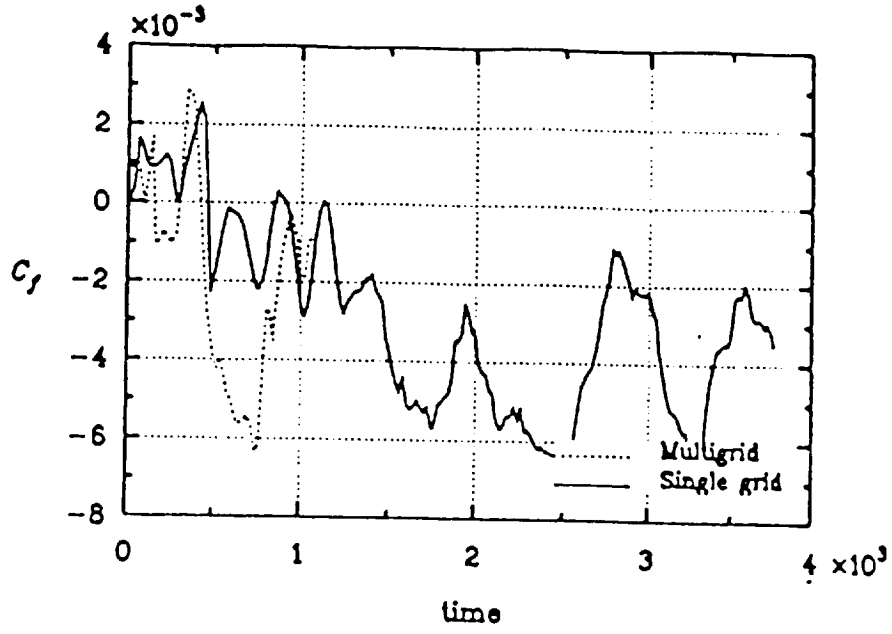


Fig. 5.15: Trace of skin friction coefficients for Sajben Transonic flow; midway on lower wall

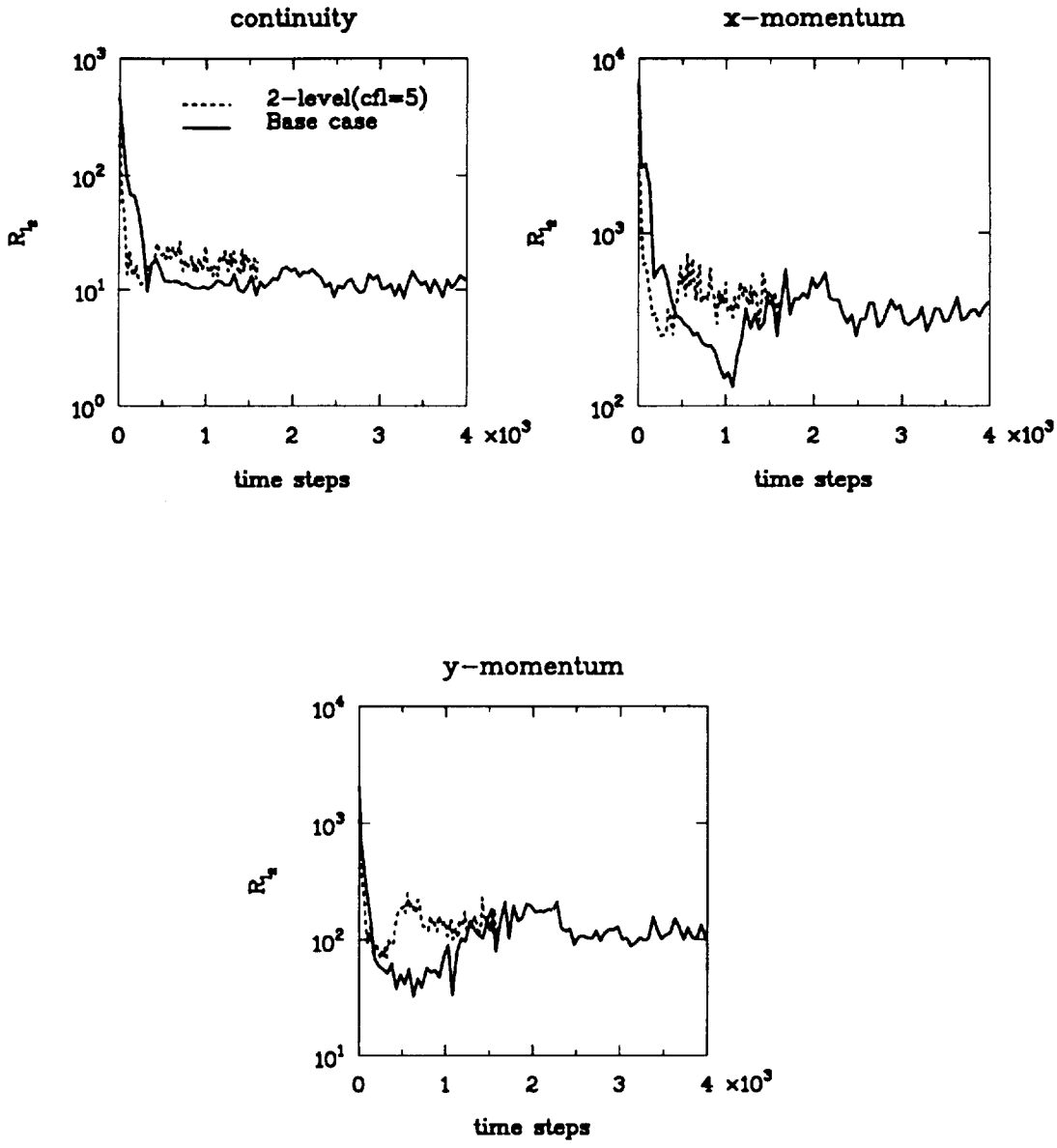


Fig. 5.16: R_{l_2} Convergence History for Sajben Transonic case non−linear dissipation and CFL = 5; 81X51 coarse case grid.

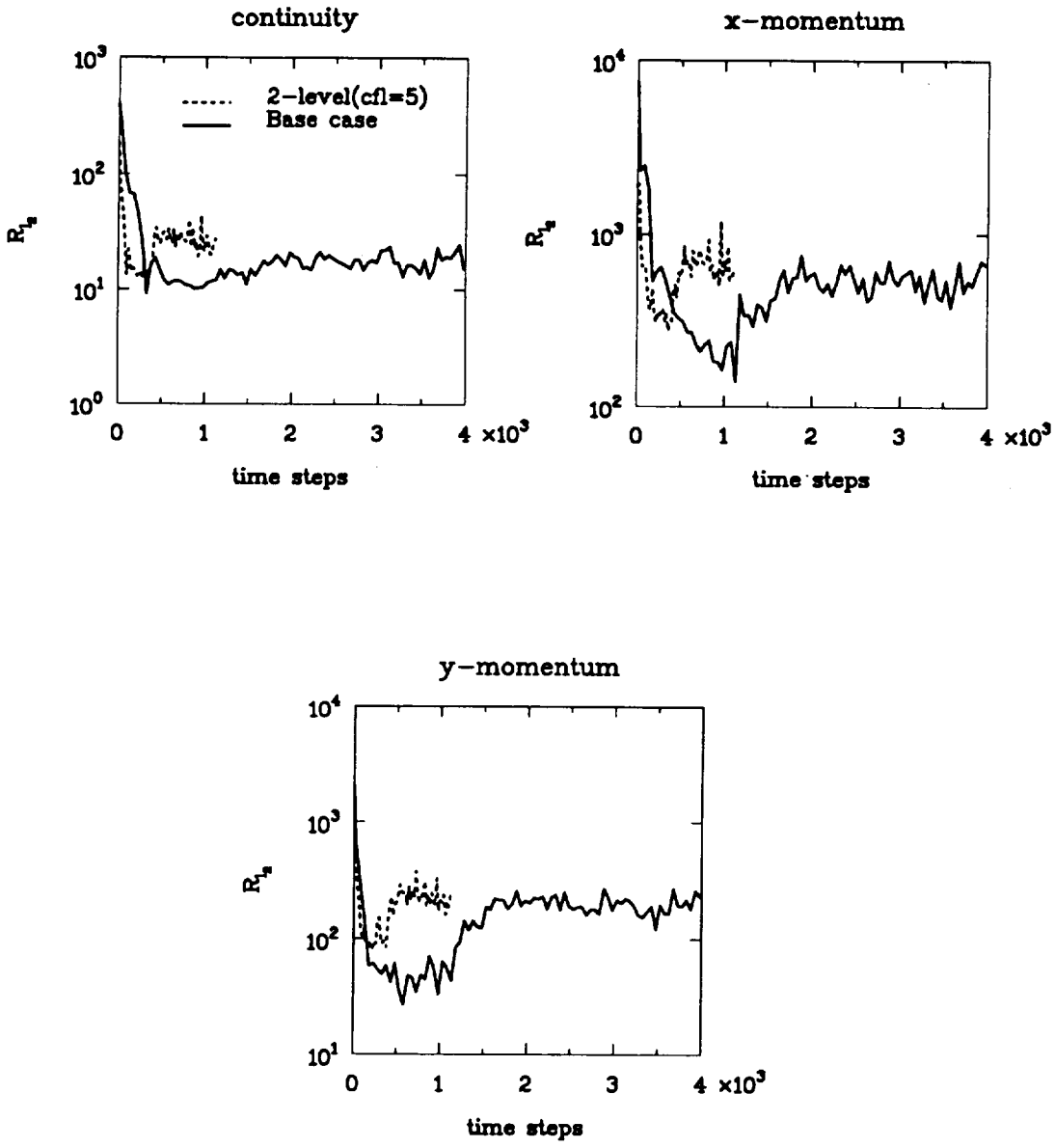


Fig. 5.17: R_{l_2} Convergence History for Sajben Transonic case with constant dissipation and CFL = 5; 81X51 coarse case grid.

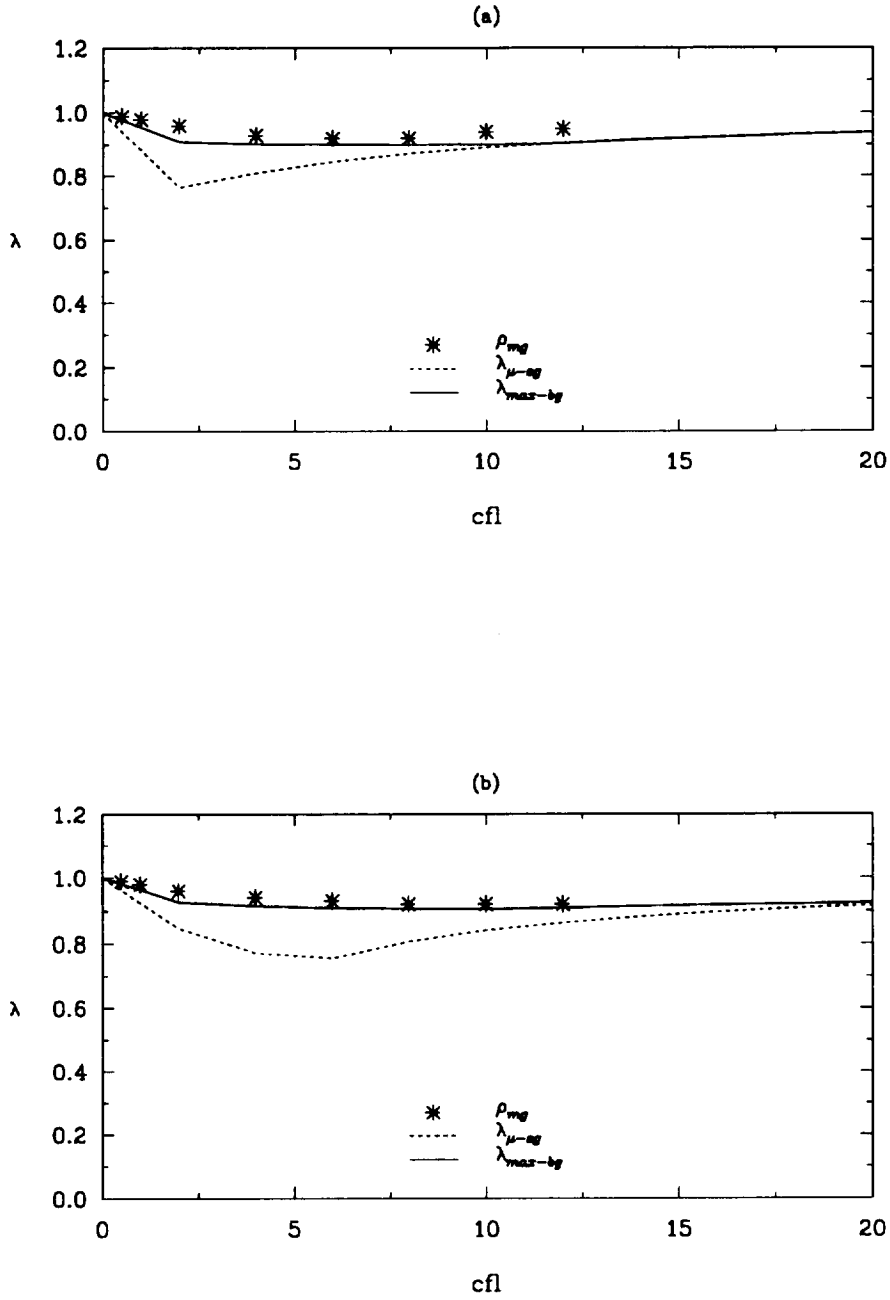


Fig. 5.18: 2–D Euler and Navier–Stokes flows around a circular cylinder using ADI central schemes (a) Inviscid Flow (b) Viscous Flow (Re=100, ϵ_e=0.5, ϵ_i=1, ν^1=1; ν^2=0)

5.4 3–D Multigrid Solutions

5.4.1 Test Problems

The test cases for the 3–D multigrid procedure are the two problems chosen to illustrate the original 3–D Proteus computer code (Towne et. al, 1992), namely the developing laminar flow (M=0.1, Re=60) in a rectangular duct (Aspect Ratio = 5:1) and the turbulent flow (M=0.2, Re=40,000) in an S–duct. In the latter, computations were performed separately with either the algebraic Baldwin–Lomax turbulence model or the two–equation $k-\varepsilon$ turbulence model with low Reynolds number extensions proposed by Chien (1982). In each case, computations were performed on the standard grid and on a coarser grid. Table 5.7 summarizes the test cases. The standard CFL number of ten is used on the finest grid in each case.

Table 5.7: Description of Test Cases for 3–D

Test Case	Flow Problem		Coarse grid	Fine grid
1	Laminar	Rectangular duct	41X21X21	101X21X41
3	B/L	S–duct	–	81X33X65
4	Chien		41X17X33	81X33X65

5.4.2 Multigrid Performance

The convergence rates of the single grid and the multigrid solutions for the developing duct flow are compared in Figs. 5.19 to 5.22. Coarse grid results in forms of the L2 norm and the average residuals of the continuity, x–, y–, z–momentum equations are presented in Figs. 5.19 and 5.20, respectively. It is clear that the MG solutions converged faster than the single grid one. The initial multigrid error is lower, because the full multigrid procedure provided better initial guesses on the fine grid by initially performing about 200 iterations on the

coarsest grid. In this case the MG cycle was performed every third fine grid iteration, so that the effective work units for each MG cycle is about 1.6. Thus, for a five order of magnitude reduction in x–momentum residuals, there is about a 20% saving in total CPU time with the 3–level MG procedure. The fine grid results presented in Figs. 5.21 and 5.22 show even better savings. In this case, there is an MG cycle for every fine–grid iteration, so that the effective work units for each cycle is two. But for a five–order reduction in residuals the MG method (2 or 3 level) requires about 400 iterations whereas the single grid method requires about 1700 iterations, i.e., more than 50% reduction in CPU time for the MG solutions.

The residuals for the calculations of the turbulent S–duct flow with the Baldwin–Lomax model are presented in Figs. 5.23 and 5.24. The convergence rates are much slower in this case in comparison to the laminar flow. Nevertheless, the 3–level MG procedures showed much faster convergence, with a reduction of about 50% in CPU time to reach the same residual level. Surprisingly, the $k - \varepsilon$ model computations showed much faster convergence rates. Coarse grid computations are presented in Figs. 5.25 and 5.26. In this case only two level MG cycles could be used. The effect of the CFL number used on the coarser grid is shown; the higher CFL number of five leads to faster convergence and is thus preferable, but it has been found to lead to instability and divergence in some problems. The computational saving in this case is about 30%, with a residual reduction of five orders in magnitude achieved in about 1,700 iteration with two level MG procedure compared with 4,800 iterations with the single grid method. Corresponding fine grid results are presented in Figs. 5.27 and 5.28. However, due to high computational costs, only residual reductions of two orders of magnitude with the MG procedure are presented. The single grid computation produces only one order of magnitude reduction in residuals in the same number of iterations. By extrapolation, it is estimated that the savings in CPU time for the MG solution is about 40–50% in this case.

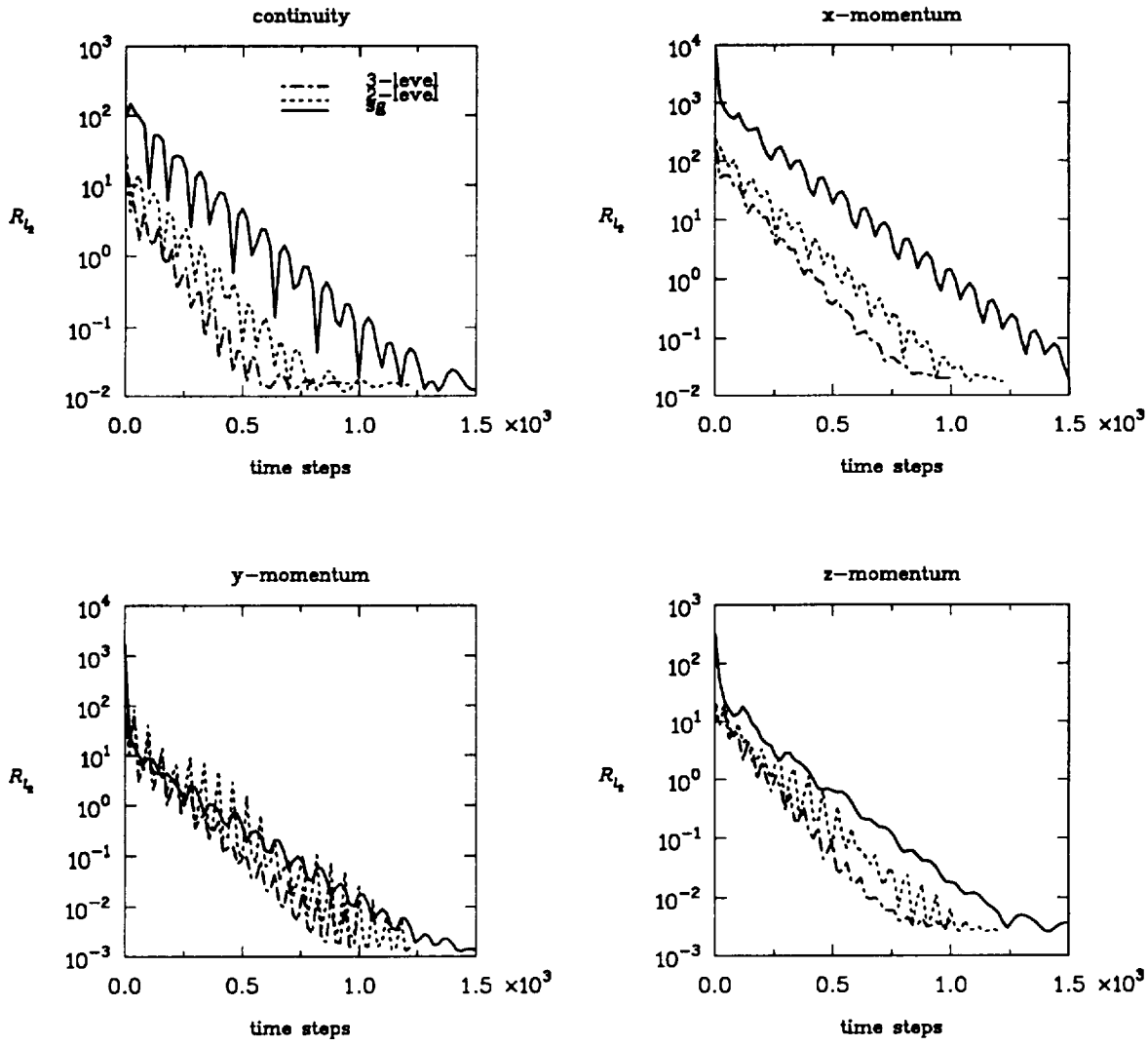


Fig. 5.19: R_{l_2} Convergence History for 3-D Duct flow; 2nd O b.c, 41X21X21 coarse grid

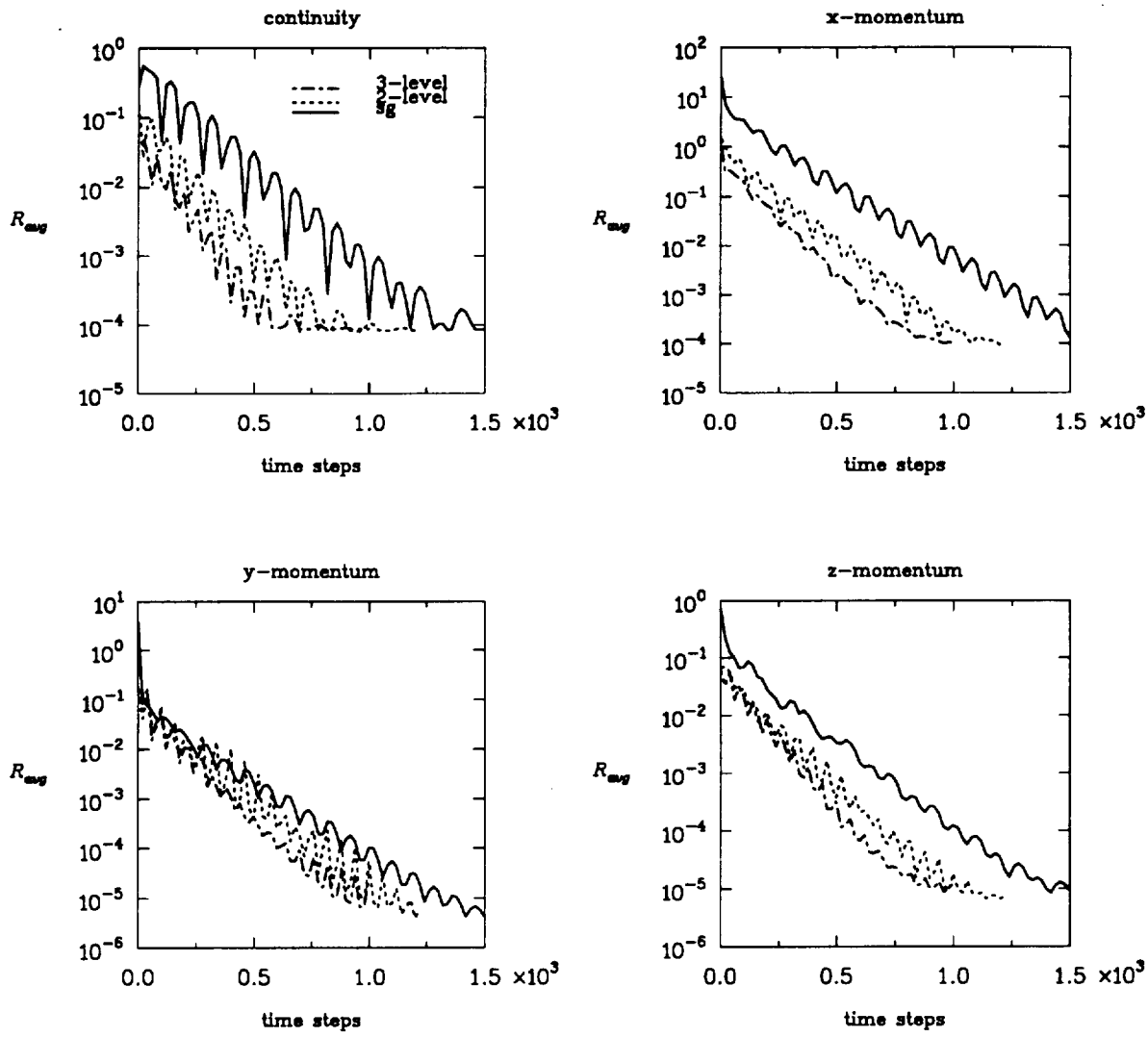


Fig. 5.20: R_{avg} Convergence History for 3-D Duct flow; 2nd O b.c, 41X21X21 coarse grid

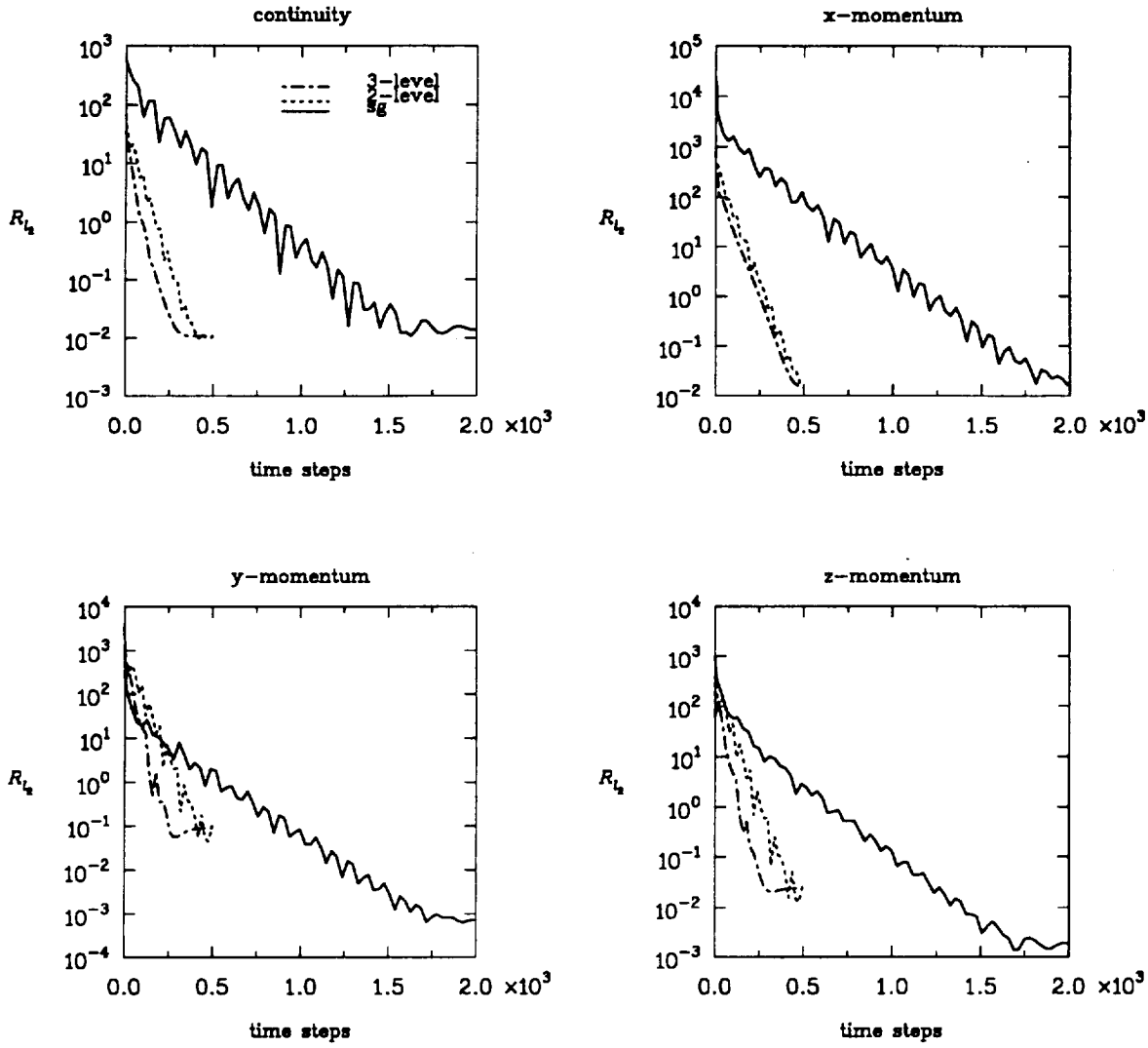


Fig. 5.21: R_{l_2} Convergence History for 3-D Duct flow; 2nd O b.c, 101X21X41 fine grid

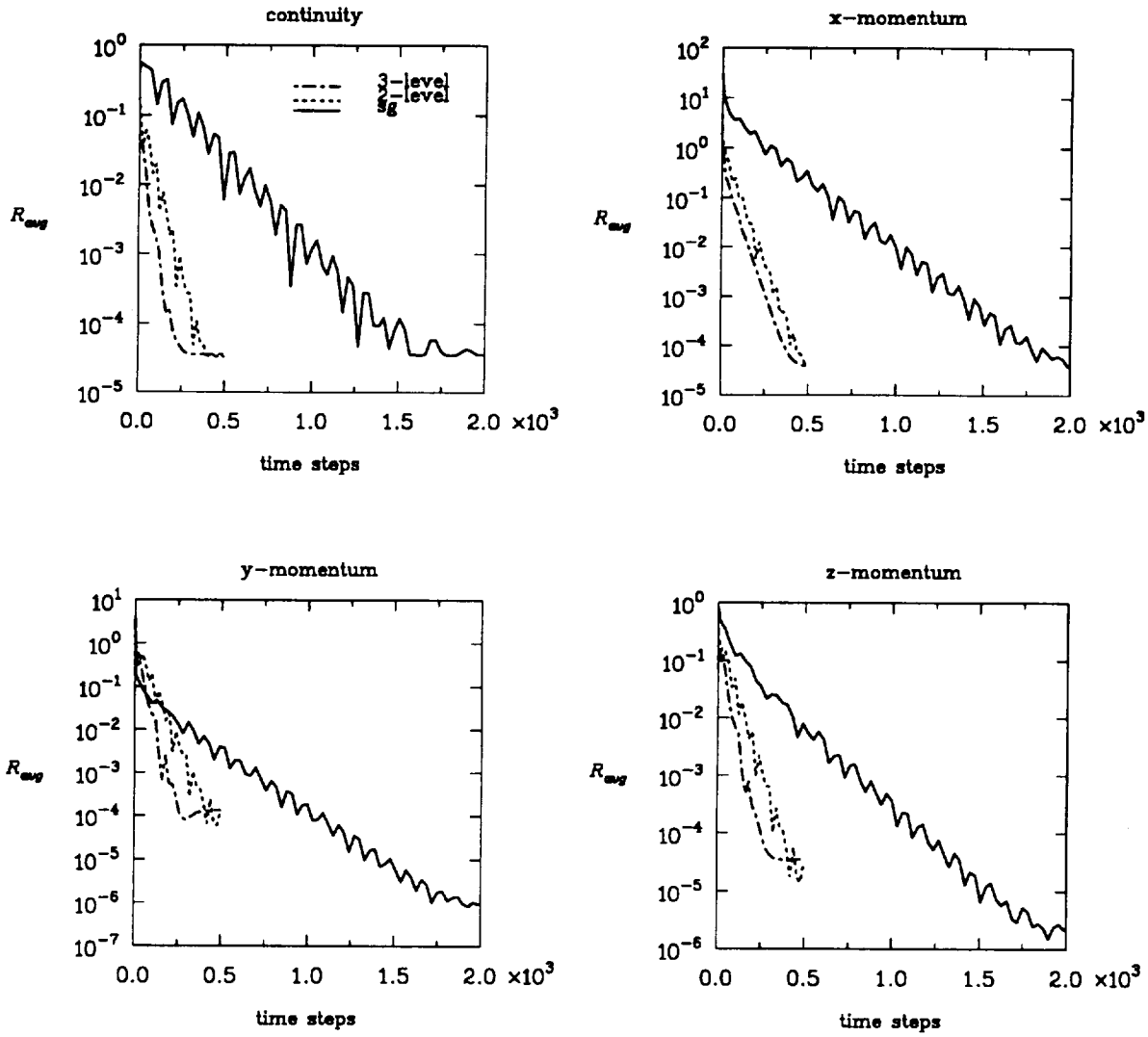


Fig. 5.22: R_{avg} Convergence History for 3–D Duct flow; 2nd O b.c, 101X21X41 fine grid

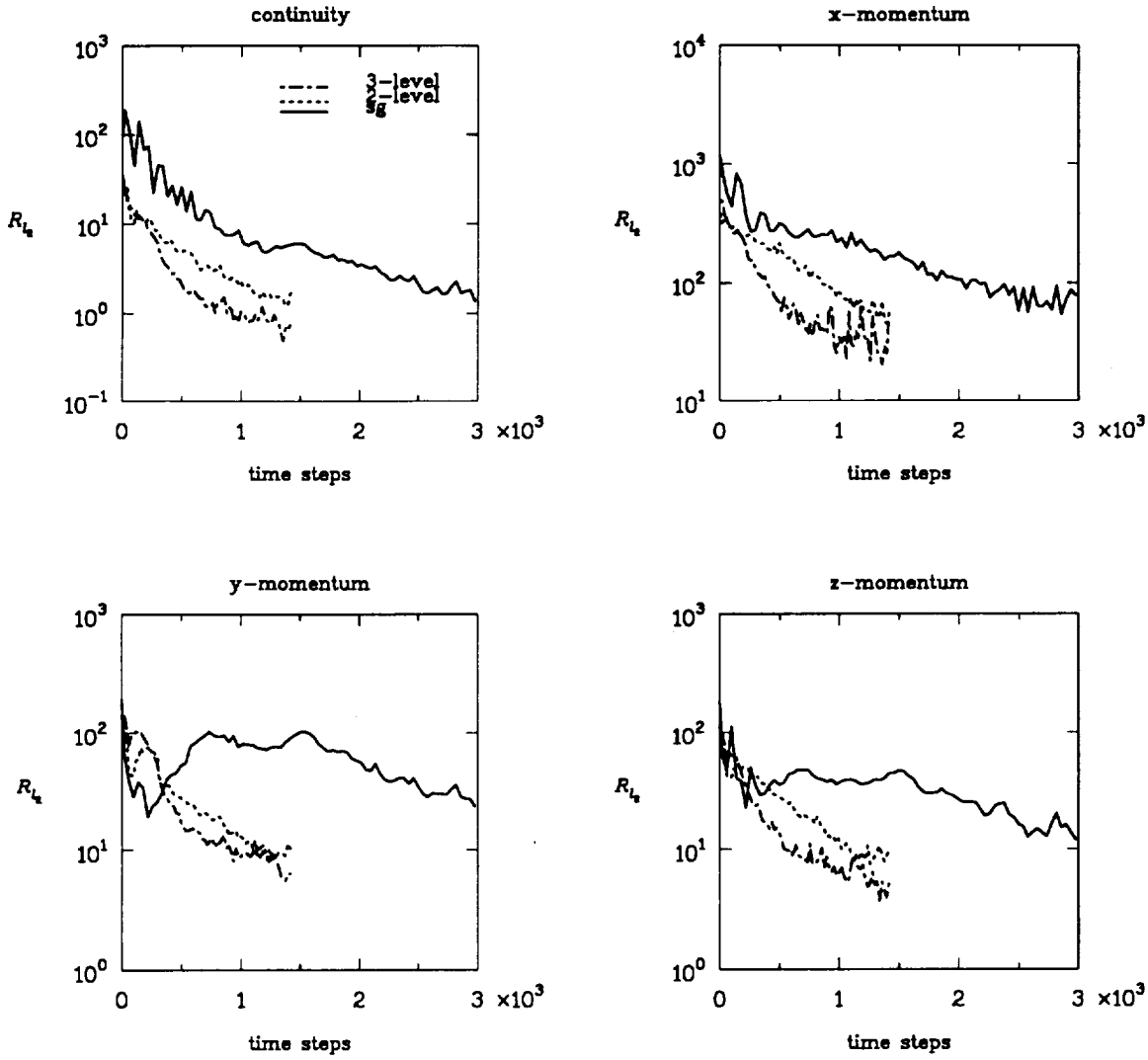


Fig. 5.23: R_{l_2} Convergence History for 3-D S-Duct flow;
Baldwin-Lomax, 2nd O b.c, 81X33X65 grid

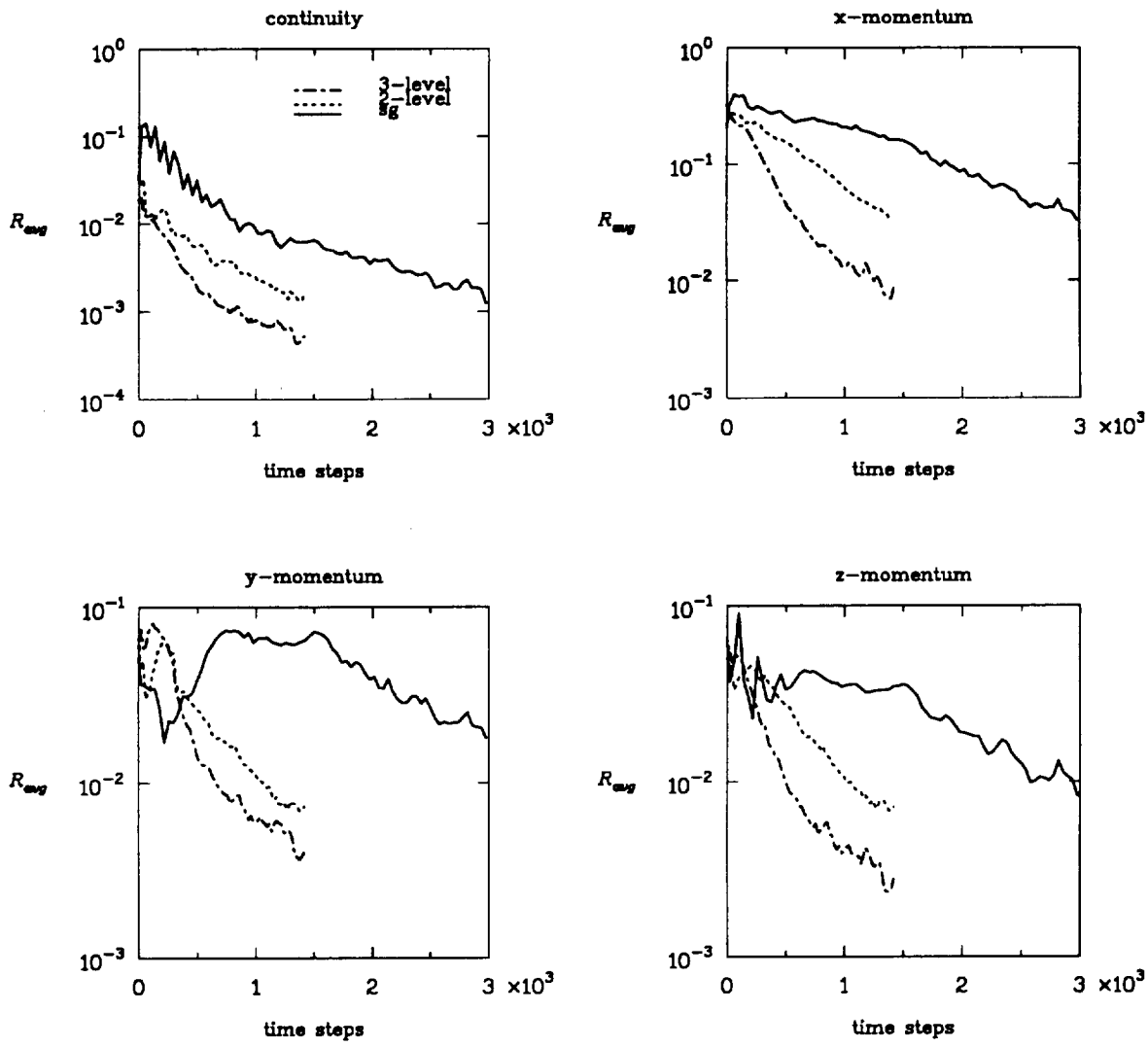


Fig. 5.24: R_{avg} Convergence History for 3-D S-Duct flow;
Baldwin-Lomax, 2nd O b.c, 81X33X65 grid

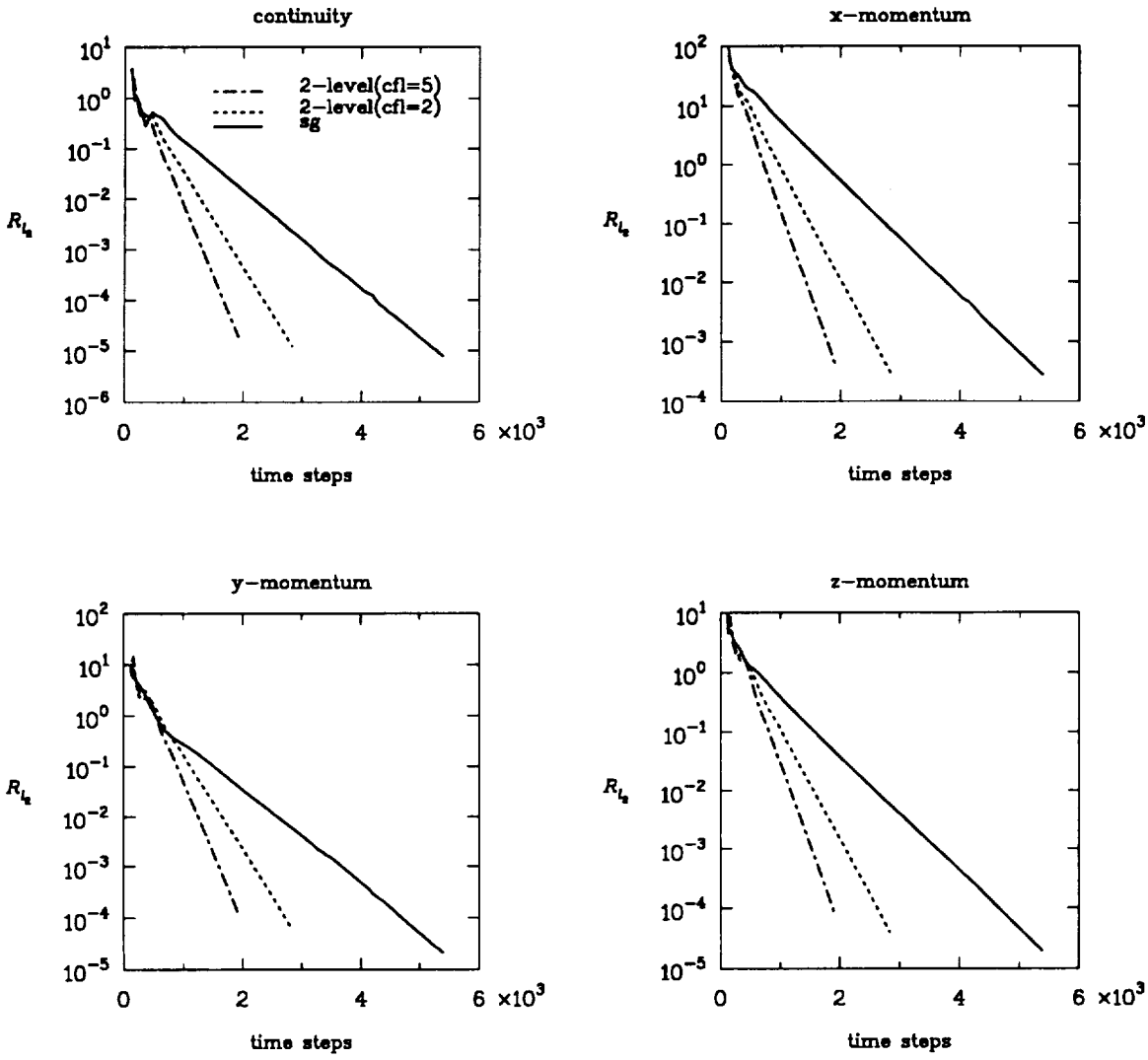


Fig. 5.25: R_{l_2} Convergence History for 3−D S−Duct flow;
k−ϵ Chien, 41X17X33 coarse grid

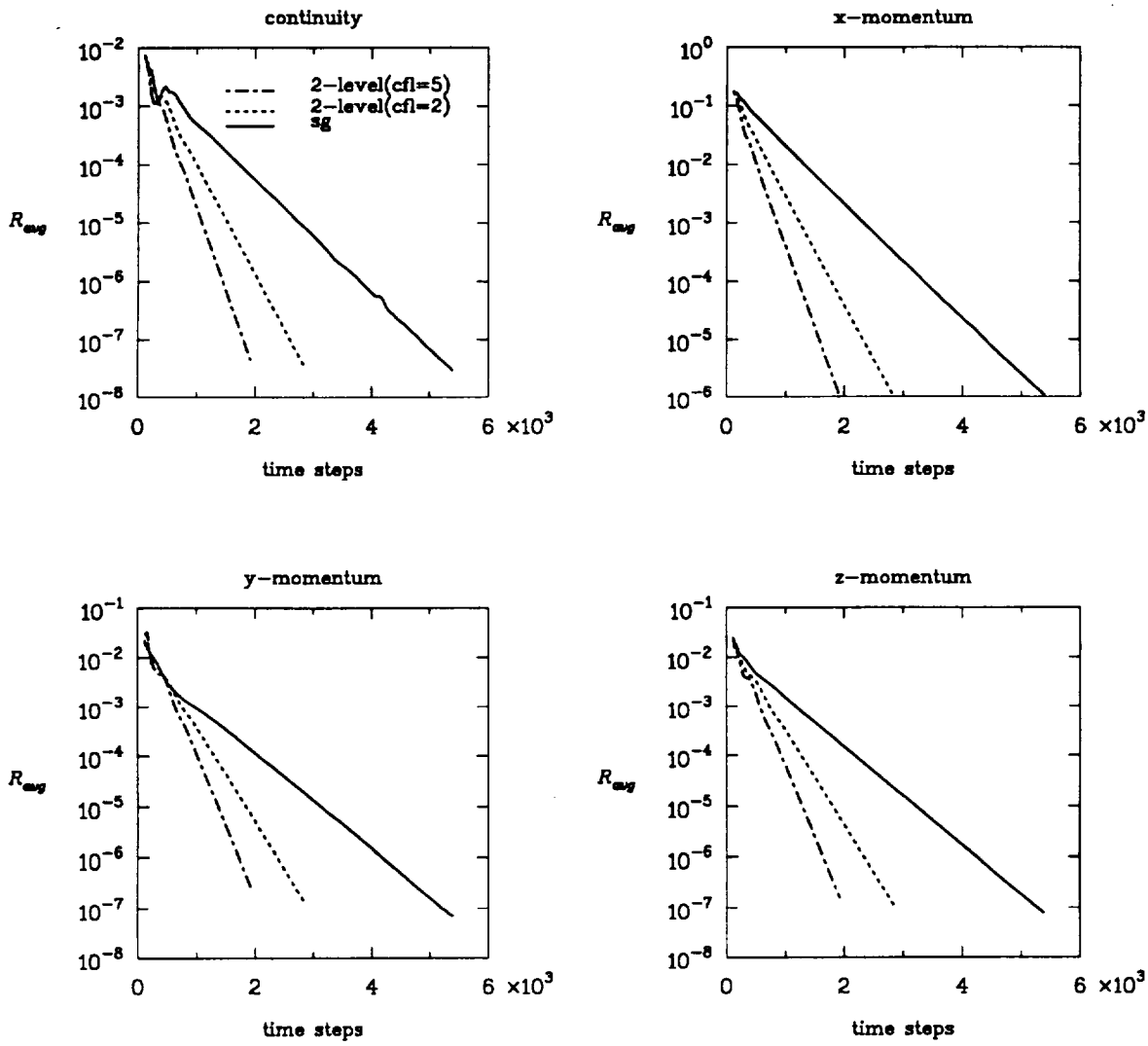


Fig. 5.26: R_{avg} Convergence History for 3−D S−Duct flow;
k−ϵ Chien, 41X17X33 coarse grid

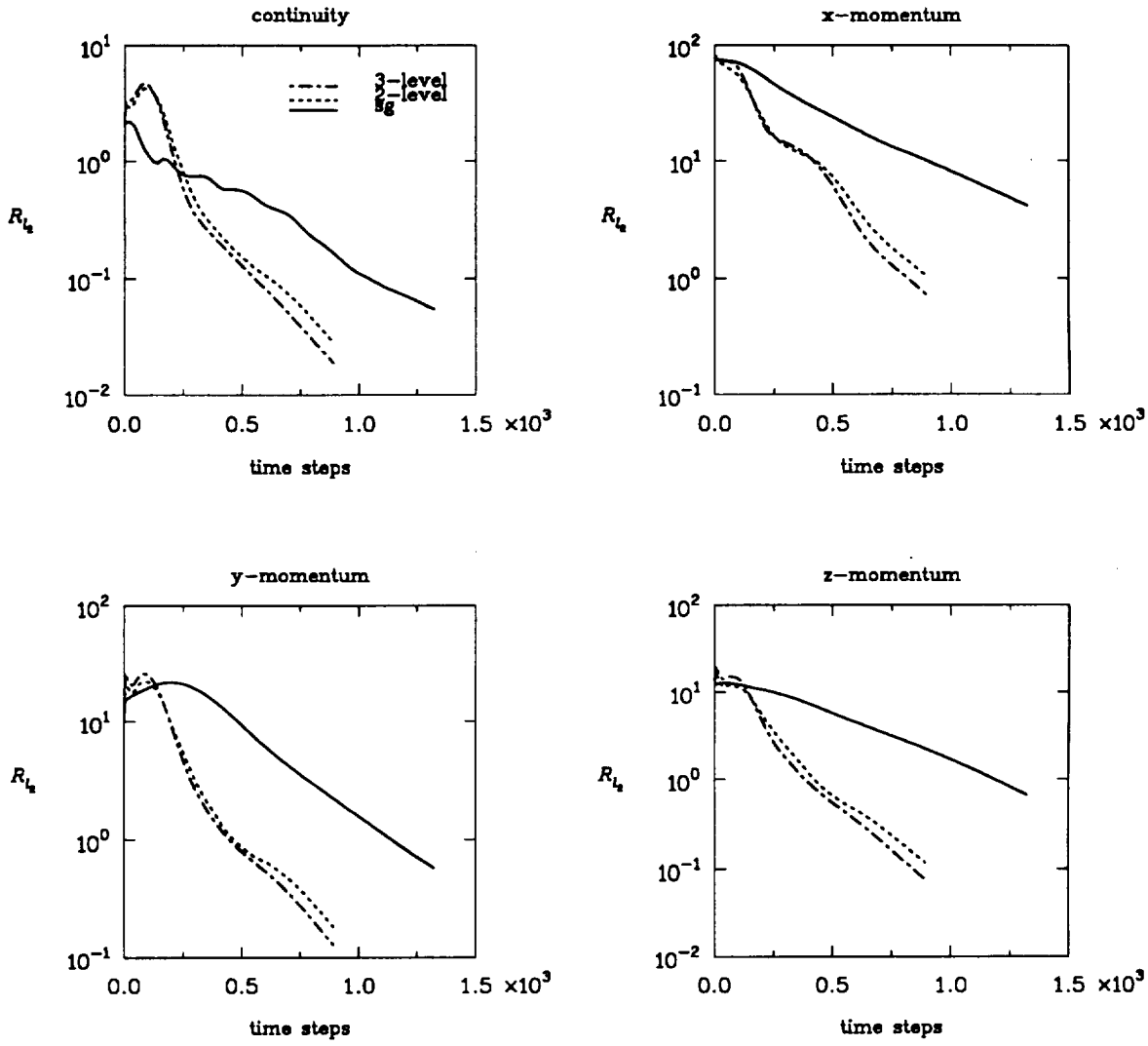


Fig. 5.27: R_{l_2} Convergence History for 3-D S-Duct flow;
k-ϵ Chien, 81X33X65 fine grid

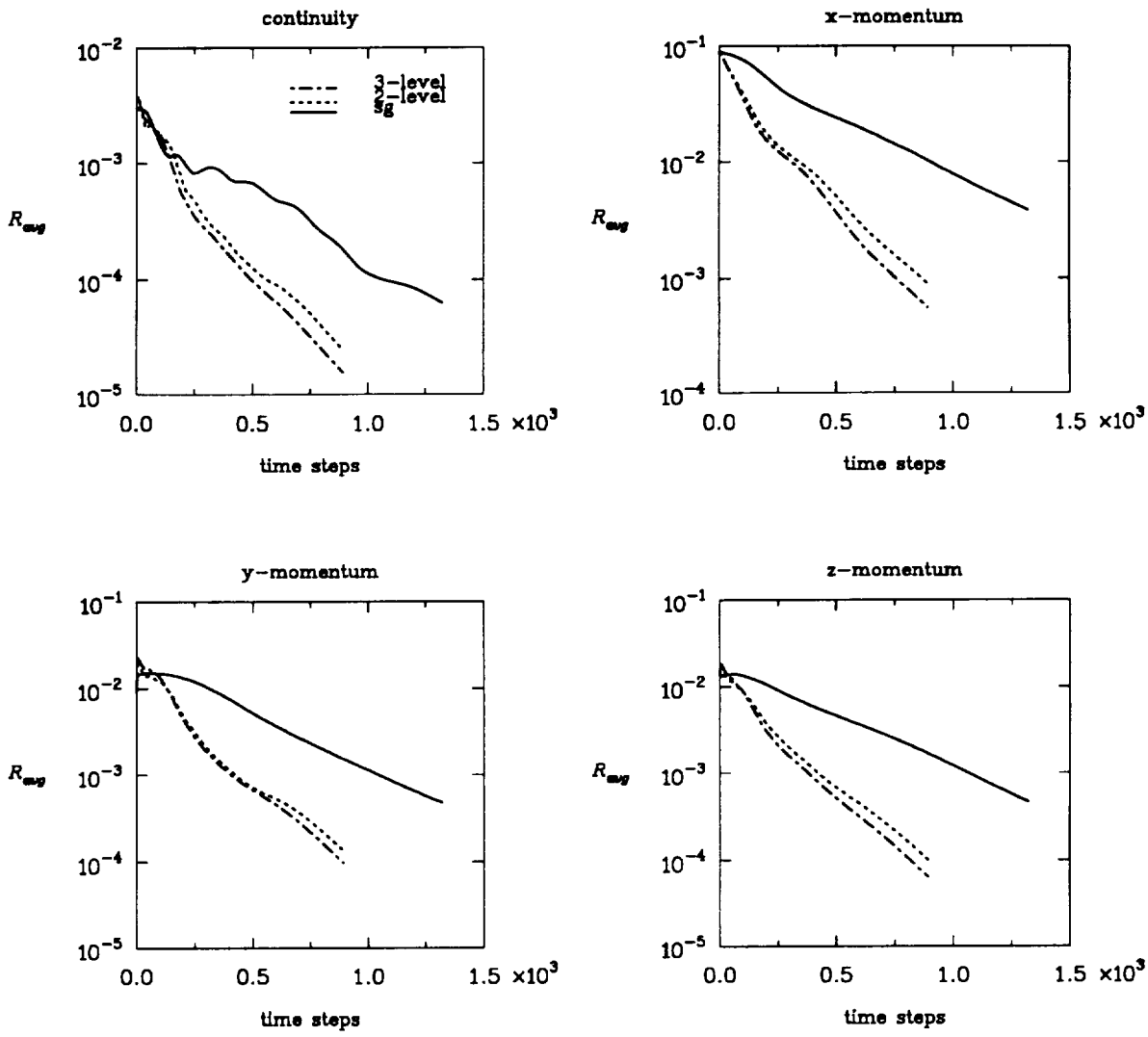


Fig. 5.28: R_{avg} Convergence History for 3−D S−Duct flow;
k−ϵ Chien, 81X33X65 fine grid

Chapter 6
CONCLUSIONS AND RECOMMENDATIONS

For a numerical scheme it is often the practice that a suitable time step is chosen heuristically or is obtained based on the results of stability studies of model scalar equations. For complicated multidimensional problems, however, this approach is not only inaccurate but also expensive. In this work the stability analysis of the full, coupled 3–D Euler and Navier–Stokes equations has been investigated for various numerical schemes. These schemes include three upwind difference based factorizations, namely Spatial, Eigenvalue and Combination splits, and two central difference based factorizations, namely the LU and ADI methods. In the former, both the Steger–Warming and van Leer flux–vector splitting methods are considered. The range of CFL numbers over which each scheme is stable and the optimum CFL numbers are presented.

In the process of computing the convergence characteristics of the above schemes, a measure of multigrid performance, namely the smoothing factor, is also evaluated. Computation of the smoothing factor is, however, restricted to the high frequency range only and does not incorporate the transfer processes that are fundamental to multigrid methods. The bi–grid procedure which was mathematically formulated to account for aliasing effects and the transfer operations has, therefore, been utilized to assess the performance of different numerical schemes for model problems using the diffusion, convection and linearized burger's equations. Compared to bi–grid results, it is observed that the smoothing factor predicts poorly the performance of certain numerical schemes as smoothers for the multigrid procedure. Motivated by these results, the bi–grid method is further used to assess the

performance of multigrid computations of Euler and Navier–Stokes solutions using the selected numerical schemes. The schemes suitable for multigrid techniques are identified.

The established results from these predictions served as a guide in the implementation of the multigrid method for numerical computations using the Navier–Stokes equations. Convergence acceleration to steady state of various 2–D and 3–D test cases ranging from inviscid to viscous turbulent flows are investigated for different geometries. In general, multigrid acceleration is found to be more effective in 3–D problems than in 2–D problems. Saving of up to 60% CPU time are obtained in several of the test cases. Furthermore, it was observed that for flows that are inherently unsteady, e.g. flows with flapping shocks, the multigrid technique as implemented failed to accelerate convergence. For such problems, multigrid implementation has to be modified to accommodate their inherent unsteadiness.

This study cannot be complete without pointing out the directions in which further research should be performed. Some time–stepping algorithms have been investigated for multigrid smoothers. It would be highly desirable to compile the smoothing properties of many more of the popularly used smoothers. In the present work, the bi–grid results have been compared to either the ideal multigrid sequence in the case of the model Burger's equation, or the Full Multigrid method in the case of the upwind and Beam–Warming ADI methods. Effort should be made to measure the deviation of some other multigrid cycles from the bi–grid predictions. Finally, time–dependent multigrid algorithm can be used to solve those problems that posses inherent unsteadiness.

REFERENCES

Anderson D. A., Tannehill J. C. and Pletcher R. H., 1984, *Computational Fluid Mechanics and Heat Transfer*, McGraw–Hill, New York, pp. 489.

Anderson W. K., Thomas J. L. and Whitfield D. L., 1988, "Three–Dimensional Multigrid Algorithms for the Flux–Split Euler Equations," *NASA TP–2829.*

Arnone A. and Swanson R. C., 1988, "A Navier–Stokes Solver for Cascade Flows," *NASA–CR 181682.*

Bakhalov N. S., 1966, "On the convergence of a Relaxation Method with Natural Constraints on an Elliptic Operator," *Z. Vycisl, Mat. Mat. Fiz.*, Vol. 6, pp. 861.

Baldwin B. S. and Lomax H., 1978, "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," *AIAA Paper 78–257.*

Baysal O., Fouladi R. and Lessard V. R., 1991, "Multigrid and Upwind Viscous Flow Solver on Three–Dimensional Overlapped and Embedded Grids," *AIAA J.* Vol. 29 pp. 903.

Beam R. M. and Warming R. F., 1978, "An Implicit Scheme for the Compressible Navier–Stokes Equations," *AIAA J.*, Vol. 16, pp. 393.

Brandt A., 1977, "Multi–level Adaptive Solutions to Boundary–value Problems," *Math. Comp.*, Vol. 31, pp. 138.

Brandt A., 1991, "Rigorous Quantitative Analysis of Multigrid," Preliminary Report Prepared for Air–Force Office of Scientific Research, United States Air Force.

Brandt A. and Yavneh I., 1993, "Accelerated Multigrid Convergence and High–Reynolds Recirculating Flows," *SIAM J. Sci. Computing*, Vol. 14, pp. 607.

Caughey D. A., 1988, "Diagonal Implicit Multigrid Algorithm for the Euler Equations," *AIAA J.*, Vol. 26, pp. 841.

Caughey D. A. and Iyer, 1989, "Diagonal Implicit Multigrid Calculation of Inlet Flowfield," *AIAA J.*, Vol. 27, pp. 110.

Chien K. Y., 1982, "Prediction of Channel and Boundary–Layer Flows with a Low–Reynolds–Number Turbulence Model," *AIAA J.*, Vol. 20, pp. 33.

Chima E., Turkel E. and Schaffer S., 1987, "Comparison of Three Explicit Multigrid Methods for the Euler and Navier–Stokes Equations," *NASA TM 88878.*

Demuren A. O., 1989, "Application of Multigrid Methods for Solving the Navier–Stokes Equations," *J. Mech. Eng. Sci.*, Vol. 203. Also *NASA TM 102359.*

Demuren A. O., 1992, "Multigrid Acceleration and Turbulence Models for Computations of 3–D Turbulent Jets in Crossflow," *Int. J Heat and Mass Transfer*, Vol. 35, pp. 2783.

Demuren A. O. and Ibraheem S. O., 1992, "Convergence Acceleration of the Proteus Computer Code with Multigrid Methods," Interim Report Prepared for the Internal Fluid Mechanics Division, NASA Lewis Research Center.

Demuren A. O. and Ibraheem S. O., 1993, "On the Stability Analysis of Approximate Factorization Methods for 3–D Euler and Navier–Stokes Equations," *NASA TM–106314*.

Demuren A. O. and Ibraheem S. O., 1994, "On the Stability Analysis of Approximate Factorization Methods for 3–D Euler and Navier–Stokes Equations," *Numerical Heat Transfer, Part B*, Vol. 25, pp. 97.

Elmiligui A. A., 1992, "Explicit Multistage schemes for the solution of the Three–Dimensional Compressible Euler/Navier–Stokes Equations," *PhD Thesis*, Old Dominion University, pp. 70.

Federenko R. P., 1961, "A Relaxation Method for Solving Elliptic Difference Equations," *Z. Vycisl, Mat. Mat. Fiz.*, Vol. 1, pp. 922.

Federenko R. P., 1964, "The speed of convergence of one iterative process," *Z. Vycisl, Mat. Mat. Fiz.*, Vol. 4, pp. 559.

Fletcher C. A. J., 1991, "Computational Techniques for Fluid Dynamics 1," Springer Verlag, New York, pp. 164 and pp. 210.

Hackbusch W., 1976, "Ein iteratives verfahren zur schnellen Auflosung elliptisher Randevertprobleme," Report 76–12, Institut fur Angewandte Mathematika, Universitat Koln.

Himansu and Rubin, 1988, "Multigrid Acceleration of a Relaxation Procedure for the Reduced Navier–Stokes Equations," *AIAA J.*, Vol. 26, pp. 1044.

Hirsch C., 1990, *Numerical Computation of Internal and External Flows 2*, John Wiley & Sons, New York, pp. 408.

Horton G. and Vandewalle S., 1993, "A Space–Time Multigrid Method For Parabolic PDES," submitted to *SIAM J. Sci. Computing*.

Ibraheem S. O, 1994, "Multigrid Acceleration of Time–Dependent Solutions of Navier–Stokes Equations," *PhD Thesis*, Old Dominion University.

Ibraheem S. O. and Demuren A. O., 1994a, "Bi–grid Stability Analysis Method for Multigrid Prediction in Burger's Equation," in Proceedings, SIAM 18th Annual Meeting, Southeastern–Atlantic section, Winston–Salem, North Carolina, pp. 20.

Ibraheem S. O. and Demuren A. O., 1994b, "On Bi–grid Local Mode Analysis of Solution Techniques for 3–D Euler and Navier–Stokes Equations," *Virginia J. of Sci.*, Vol. 45, pp. 40. Also *NASA TM–106749* and submitted to *J. of Comp. Physics*.

Jameson A., 1979, "Acceleration of Transonic Potential Flow Calculations on Arbitrary Meshes by the Multiple grid Method," *AIAA Paper 79–1458*.

Jameson A., 1983, "Solution of the Euler Equations for Two–Dimensional Transonic Flow by a Multigrid Method," *Appl. Math. Comp.*, Vol. 13, pp. 327.

Jameson A., Schmidt W., Turkel E., 1981, "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge–Kutta Time–Stepping Schemes," *AIAA Paper 81–1259.*

Jameson A. and Turkel E., 1981, "Implicit Schemes and LU Decompositions," *Math. of Comp.*, Vol. 37, pp. 385.

Jameson A. and Yoon S., 1986, "Multigrid Solution of the Euler Equations Using Implicit Schemes," *AIAA J.*, Vol. 24, pp. 1737.

Jameson A. and Yoon S., 1987, "Lower–Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA J.*, Vol.25, pp. 929.

Jespersen D. C., 1983, "Design and Implementation of a Multigrid Code for the Euler Equations," *Appl. Math. Comp.*, Vol. 13, pp. 357.

Jespersen D. C. and Pulliam T. H., 1983, "Flux Vector Splitting and Approximate Newton Methods," AIAA Sixth CFD Conference.

Koren B., 1990, "Multigrid and Defect Correction for the Steady Navier–Stokes Equations," *J. Comp. Phy.*, Vol. 87, pp. 25.

Mavripilis D. J., 1988, "Multigrid Solution of the Two–Dimensional Euler Equations on Unstructured Triangular Meshes," *AIAA J.*, Vol. 26, pp. 824.

Mavripilis D. J., 1990, "Accurate Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes," *AIAA J.*, Vol. 28, pp. 213.

Mavripilis D. J. and Jameson A., 1990, "Multigrid Solution of the Navier–Stokes Equations on Triangular Meshes," *AIAA J.*, Vol. 28, pp. 1415.

Mccarthy D. R., and Reyhner T. A., 1982, "Multigrid Code for Three Dimensional Transonic Flows," *AIAA J.*, Vol. 20, pp. 45.

Morano E., 1992, "Resolution des equation d'Euler par une methode multigrille stationaire," *PhD Thesis*, University de Nice Sophia–Antipolis, pp. 50.

Morano E. and Dervieux A., 1993, "Steady Relaxation Methods for Unstructured Multigrid Euler and Navier Stokes Equations," submitted to *J. of Computational Fluid Dynamics.*

Mulder W. A., 1988, "Analysis of a multigrid Method for the Euler Equations of Gas Dynamics in Two Dimensions," in Multigrid Methods, Theory, Applications and Supercomputing, Lecture Notes in Pure and Applied Mathematics, edited by S. F. McCormick, Marcel Dekker Inc., New York, pp. 467.

Mulder W. A., 1989, "A New Multigrid Approach to Convection Problems," *J. Comp. Phy.*, Vol. 83, pp. 303.

Ni R. H., 1981, "A Multiple Grid Scheme for Solving the Euler Equations," *AIAA J.*, Vol. 20, pp. 1565.

Pulliam T. H., 1986, "Artificial Dissipation Models for the Euler Equations," *AIAA J.*, Vol. 24, pp. 1931.

Radespiel R., Rossow C. and Swanson R. C., 1990, "Efficient Cell–Vertex Multigrid Scheme for Three–Dimensional Navier–Stokes Equations," *AIAA J.*, Vol 28, pp. 1464.

Rhie C. M., 1989, "Pressure–Based Navier–Stokes Solver Using Multigrid Method," *AIAA J.*, Vol. 27, pp. 1017.

Sajben M., Bogar T. J. and Kroutil J. C., 1984, "Forced Oscillation Experiments in Supercritical Diffuser Flows," *AIAA J.*, Vol 22, pp. 465.

South J. C. and Brandt A., 1976, "The Multi–Grid Method : Fast Relaxation for Transonic Flows," *Adv. Eng. Sci.*, Vol 4, pp. 1359.

Southwell R. V., 1935, Stress–Calculation in Frameworks by the Method of "Systematic Relaxation of Constraints", Proc. R. Soc. London, Vol. 151A, No. 872, pp. 56.

Stiefel E. L., 1952, "Uber einige Methodern der Relaxationsrechnung," *Z.A.M.P*, Vol. 3, pp. 1.

Steger J. L., 1978, "Implicit Finite–Difference Simulation of Flow about Arbitrary Two–Dimensional Geometries," *AIAA J.*, Vol. 16, pp. 679.

Steger J. L. and Warming R. F., 1980, "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite–Difference Methods," *J. Comp. Phys.*, Vol. 40, pp. 263.

Stuben K. and Trottenberg U., 1982, "Multigrid methods: Fundamental Algorithms, Model Problems Analysis and Applications," in Lecture notes in Mathematics, Multigrid Methods, *960,* Springer–Verlag, New York, pp. 1.

Swanson R. C. and R. Radespiel, 1991, "Cell Centered and Cell Vertex Multigrid Schemes for Navier–Stokes Equations," *AIAA J.*, Vol. 29, pp. 697.

Thomas J. L. , van Leer B. and Walters R. W., 1985, "Implicit Flux–Split Scheme For the Euler Equations," *AIAA paper 85–1680.*

Thompson M. C. and Ferziger J. H., 1989, "An Adaptive Multigrid Technique for the Incompressible Navier–Stokes Equations," *J. Comp. Phy.*, Vol. 82, pp. 94.

Towne C. E., Schwab J. R., Benson T. J. and Suresh A., 1990, "PROTEUS Two–Dimensional Navier–Stokes Computer Code – Version 1.0, Volumes 1–3," *NASA TM–102551–3.*

Towne C. E., Schwab J. R., Benson T. J. and Suresh A., 1992, "PROTEUS Three–Dimensional Navier–Stokes Computer Code – Version 1.0, Volumes 1–3," *NASA TM.*

van Asselt E. J., 1982, "The Multigrid Method and Artificial viscosity," in Lecture Notes in Mathematics, Multigrid Methods 960, Springer–Verlag, NY, pp. 313.

van Leer B., 1982, "Flux–Vector Splitting for the Euler Equations," *ICASE 82–30.*

Vanka, S. P., 1986, "Block–Implicit Multigrid Solution of Navier–Stokes Equations in Primitive Variables," *J. Comp. Phy.*, Vol. 65, pp. 138.

von Lavante E., Claes D. and Anderson W. K., 1986, "The Effects of various Implicit operators on a Flux Vector Splitting Method," *AIAA paper 86—0273.*

Yadlin Y. and Caughey D. A., 1991, "Block Multigrid Implicit Solution of the Euler Equations of Compressible Fluid Flow," *AIAA J.*, Vol. 29, pp. 712.

Yokota J. W., 1987, "An LU Implicit Multigrid Algorithm to solve the Euler Equations for Transonic flow in Rotating Turbomachinery Passages," *PhD Thesis*, Cornell University, pp. 83.

Yokota J. W., 1989, "A Diagonally Inverted LU Implicit Multigrid Scheme for the 3–D Navier–Stokes Equations and a Two Equation Model of Turbulence," *AIAA Paper No. AIAA–89–0.*

Yokota J. W. and Caughey D. A., 1988, "LU Implicit Multigrid Algorithm for the Three–Dimensional Euler Equations," *AIAA J.*, Vol. 26, pp. 1061.

Yokota, J. W., Caughey D. A. and Chima R. V., 1988, "A Diagonally Inverted LU Implicit Multigrid Scheme," *NASA TM 100911.*

APPENDIX A

INVISCID FLUX JACOBIANS

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -u^2 + \frac{\gamma-1}{2}q^2 & (3-\gamma)u & -(\gamma-1)v & -(\gamma-1)w & (\gamma-1) \\ -uv & v & u & 0 & 0 \\ -uw & w & 0 & u & 0 \\ -u\left[\gamma e_o - (\gamma-1)q^2\right] & \gamma e_o - \frac{\gamma-1}{2}(q^2+2u^2) & -(\gamma-1)uv & -(\gamma-1)uw & \gamma u \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -uv & v & u & 0 & 0 \\ -v^2 + \frac{\gamma-1}{2}q^2 & -(\gamma-1)u & (3-\gamma)v & -(\gamma-1)w & (\gamma-1) \\ -uw & 0 & w & u & 0 \\ -v\left[\gamma e_o - (\gamma-1)q^2\right] & -(\gamma-1)uv & \gamma e_o - \frac{(\gamma-1)}{2}(q^2+2v^2) & -(\gamma-1)vw & \gamma v \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ -uw & w & 0 & u & 0 \\ -vw & 0 & w & v & 0 \\ -w^2 + \frac{\gamma-1}{2}q^2 & -(\gamma-1)u & -(\gamma-1)v & (3-\gamma)w & (\gamma-1) \\ -w\left[\gamma e_o - (\gamma-1)q^2\right] & -(\gamma-1)uw & -(\gamma-1)uw & \gamma e_o - \frac{(\gamma-1)}{2}(q^2+2w^2)u & \gamma w \end{bmatrix}$$

where $p = (\gamma-1)(\rho e_o - 0.5\rho q^2)$ and $q^2 = u^2 + v^2 + w^2$

APPENDIX B

VISCOUS FLUX JACOBIANS

$$R = \frac{\partial E_{v,x}}{\partial Q_x} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\frac{4}{3}u & \frac{4}{3} & 0 & 0 & 0 \\ -v & 0 & 1 & 0 & 0 \\ -w & 0 & 0 & 1 & 0 \\ \frac{\gamma}{Pr}(q^2-e_o)-(\frac{4}{3}u^2-v^2-w^2) & u(\frac{4}{3}-\frac{\gamma}{Pr}) & v(1-\frac{\gamma}{Pr}) & w(1-\frac{\gamma}{Pr}) & \frac{\gamma}{Pr} \end{bmatrix}$$

$$S = \frac{\partial F_{v,y}}{\partial Q_y} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -u & 1 & 0 & 0 & 0 \\ -\frac{4}{3}v & 0 & \frac{4}{3} & 0 & 0 \\ -w & 0 & 0 & 1 & 0 \\ \frac{\gamma}{Pr}(q^2-e_o)-(\frac{4}{3}v^2-u^2-w^2) & u(1-\frac{\gamma}{Pr}) & v(\frac{4}{3}-\frac{\gamma}{Pr}) & w(1-\frac{\gamma}{Pr}) & \frac{\gamma}{Pr} \end{bmatrix}$$

$$Y = \frac{\partial G_{v,z}}{\partial Q_z} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -u & 1 & 0 & 0 & 0 \\ -v & 0 & 1 & 0 & 0 \\ -\frac{4}{3}w & 0 & 0 & \frac{4}{3} & 0 \\ \frac{\gamma}{Pr}(q^2-e_o)-(\frac{4}{3}w^2-u^2-v^2) & u(1-\frac{\gamma}{Pr}) & v(1-\frac{\gamma}{Pr}) & w(\frac{4}{3}-\frac{\gamma}{Pr}) & \frac{\gamma}{Pr} \end{bmatrix}$$

$$R_1 = \frac{\partial E_{v,y}}{\partial Q_y} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\frac{2}{3}v & 0 & -\frac{2}{3} & 0 & 0 \\ -u & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{3}uv & v & -\frac{2}{3}u & 0 & 0 \end{bmatrix}, R_2 = \frac{\partial E_{v,z}}{\partial Q_z} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}w & 0 & 0 & -\frac{2}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -u & 1 & 0 & 0 & 0 \\ -\frac{1}{3}uw & w & 0 & -\frac{2}{3}u & 0 \end{bmatrix}$$

$$S_1 = \frac{\partial F_{v,x}}{\partial Q_x} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -v & 0 & 1 & 0 & 0 \\ -\frac{2}{3}u & -\frac{2}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{3}uv & -\frac{2}{3}v & u & 0 & 0 \end{bmatrix}, S_2 = \frac{\partial F_{v,z}}{\partial Q_z} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}w & 0 & 0 & -\frac{2}{3} & 0 \\ -v & 0 & 1 & 0 & 0 \\ -\frac{1}{3}uv & 0 & w & -\frac{2}{3}v & 0 \end{bmatrix}$$

$$Y_1 = \frac{\partial G_{v,x}}{\partial Q_x} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -w & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}u & -\frac{2}{3} & 0 & 0 & 0 \\ -\frac{1}{3}uw & -\frac{2}{3}w & 0 & u & 0 \end{bmatrix}, Y_2 = \frac{\partial G_{v,y}}{\partial Q_y} = \frac{\mu}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -w & 0 & 0 & 1 & 0 \\ \frac{2}{3}v & 0 & -\frac{2}{3} & 0 & 0 \\ -\frac{1}{3}vw & 0 & -\frac{2}{3}w & v & 0 \end{bmatrix}$$

where $Pr = \frac{\mu c_p}{k} = \frac{\mu \gamma c_v}{k}$.

APPENDIX C

THE BI-GRID AMPLIFICATION MATRIX $\hat{M}(\Theta)$

$$\hat{M}(\Theta) = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} & M_{15} & M_{16} & M_{17} & M_{18} \\ M_{21} & M_{22} & M_{23} & M_{24} & M_{25} & M_{26} & M_{27} & M_{28} \\ M_{31} & M_{32} & M_{33} & M_{34} & M_{35} & M_{36} & M_{37} & M_{38} \\ M_{41} & M_{42} & M_{43} & M_{44} & M_{45} & M_{46} & M_{47} & M_{48} \\ M_{51} & M_{52} & M_{53} & M_{54} & M_{55} & M_{56} & M_{57} & M_{58} \\ M_{61} & M_{62} & M_{63} & M_{64} & M_{65} & M_{66} & M_{67} & M_{68} \\ M_{71} & M_{72} & M_{73} & M_{74} & M_{75} & M_{76} & M_{77} & M_{78} \\ M_{81} & M_{82} & M_{83} & M_{84} & M_{85} & M_{86} & M_{87} & M_{88} \end{bmatrix}$$

The diagonal elements are:

$$\begin{aligned} M_{11} &= \mathbf{I} - \hat{I}_H^h(\Theta^1)\hat{I}_h^H(\Theta^1)\hat{L}_h(\Theta^1)\hat{S}_1^{\nu_1}(\Theta^1)\hat{S}_2^{\nu_2}(\Theta^1)\hat{L}_H^{-1} \\ M_{22} &= \mathbf{I} - \hat{I}_H^h(\Theta^2)\hat{I}_h^H(\Theta^2)\hat{L}_h(\Theta^2)\hat{S}_1^{\nu_1}(\Theta^2)\hat{S}_2^{\nu_2}(\Theta^2)\hat{L}_H^{-1} \\ M_{33} &= \mathbf{I} - \hat{I}_H^h(\Theta^3)\hat{I}_h^H(\Theta^3)\hat{L}_h(\Theta^3)\hat{S}_1^{\nu_1}(\Theta^3)\hat{S}_2^{\nu_2}(\Theta^3)\hat{L}_H^{-1} \\ &\vdots \\ M_{88} &= \mathbf{I} - \hat{I}_H^h(\Theta^8)\hat{I}_h^H(\Theta^8)\hat{L}_h(\Theta^8)\hat{S}_1^{\nu_1}(\Theta^8)\hat{S}_2^{\nu_2}(\Theta^8)\hat{L}_H^{-1} \end{aligned}$$

and the off-diagonal elements are:

$$M_{nm} = -\hat{I}_H^h(\Theta^n)\hat{I}_h^H(\Theta^m)\hat{L}_h(\Theta^m)\hat{S}_1^{\nu_1}(\Theta^n)\hat{S}_2^{\nu_2}(\Theta^{\hat{m}})L_H^{-1}$$

where, for example,

$$\begin{aligned} M_{21} &= -\hat{I}_H^h(\Theta^2)\hat{I}_h^H(\Theta^1)\hat{L}_h(\Theta^1)\hat{S}_1^{\nu_1}(\Theta^2)\hat{S}_2^{\nu_2}(\Theta^1)\hat{L}_H^{-1} \\ M_{32} &= -\hat{I}_H^h(\Theta^3)\hat{I}_h^H(\Theta^2)\hat{L}_h(\Theta^2)\hat{S}_1^{\nu_1}(\Theta^3)\hat{S}_2^{\nu_2}(\Theta^2)\hat{L}_H^{-1} \\ M_{84} &= -\hat{I}_H^h(\Theta^8)\hat{I}_h^H(\Theta^4)\hat{L}_h(\Theta^4)\hat{S}_1^{\nu_1}(\Theta^8)\hat{S}_2^{\nu_2}(\Theta^4)\hat{L}_H^{-1} \\ M_{76} &= -\hat{I}_H^h(\Theta^7)\hat{I}_h^H(\Theta^6)\hat{L}_h(\Theta^6)\hat{S}_1^{\nu_1}(\Theta^7)\hat{S}_2^{\nu_2}(\Theta^6)\hat{L}_H^{-1} \end{aligned}$$

Each element is a 5x5 matrix corresponding to the 5 dependent variables.

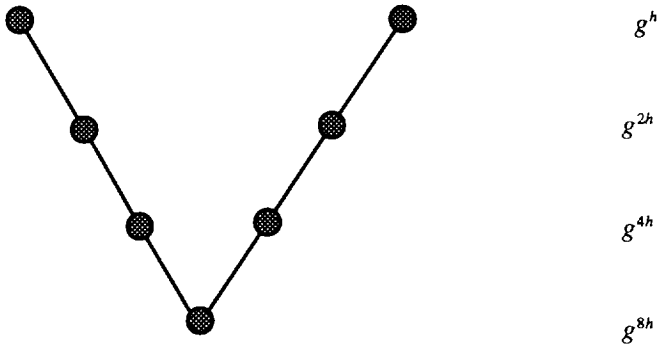
APPENDIX D

JCL FOR STABILITY CODES

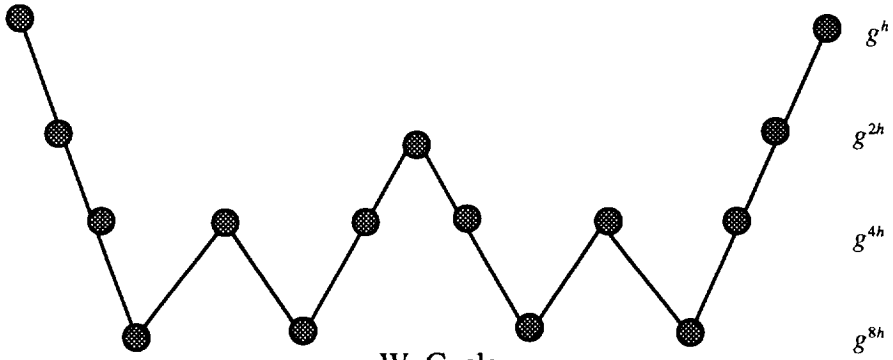
```
#QSUB -lM 1.0mW -lT 300
#QSUB -mb -me
#QSUB -r stab
#QSUB -s /bin/sh

set -x
ja

#Setup and link necessary input/output files

touch stab3d_sg.out
ln    stab3d_sg.out   fort.22

cf77 -o temp.ex stab3d_sg.f -limsl                    for single grid stability analysis
or cf77 -o temp.ex stab3d_bg.f -limsl                 for bi-grid stability analysis

#Run code

temp.ex<<EOD
Single Grid Convergence characteristics for Navier-Stokes equations @ Re = 100
&io
 nout = 22,                          fortran output file
&end

&num
 cfls=0,              Starting cfl
 cfle=30,             Stopping cfl
 delcfl=2,            cfl interval
 avs4e=1,             amount of 4th O explicit artificial viscosity
 avs2e=0,             amount of 2nd O explicit artificial viscosity, always zero
 avs2i=2,             amount of implicit artificial viscosity
&end

&flow
 ieuler=0,                           Viscous or inviscid flow
 fy=10, fz=10,                       Aspect Ratios
 a_attack=45, a_yaw=45,              Angle of Attack, Angle of Yaw
 machr=0.5                           Reference Mach number
 rer=800,                            Reference Reynolds number
&end
EOD
```

APPENDIX E

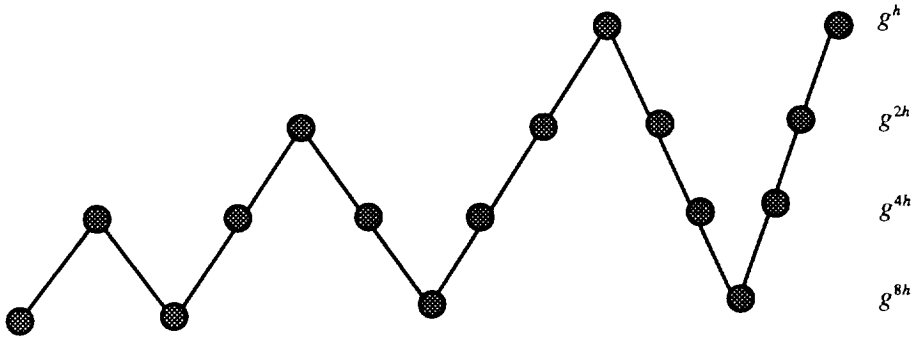
DIFFERENT MULTIGRID CYCLES



V–Cycle



W–Cycle



Full Multigrid V–Cycle