INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2005; **49**:211–231
Published online 8 June 2005 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.996

Effects of the Jacobian evaluation on Newton's solution of the Euler equations

O. Onur[§,¶] and S. Eyi[*,†,‡]

Department of Aerospace Engineering, Middle East Technical University, Ankara 06531, Turkey

SUMMARY

Newton's method is developed for solving the 2-D Euler equations. The Euler equations are discretized using a finite-volume method with upwind flux splitting schemes. Both analytical and numerical methods are used for Jacobian calculations. Although the numerical method has the advantage of keeping the Jacobian consistent with the numerical residual vector and avoiding extremely complex analytical differentiations, it may have accuracy problems and need longer execution time. In order to improve the accuracy of numerical Jacobians, detailed error analyses are performed. Results show that the finite-difference perturbation magnitude and computer precision are the most important parameters that affect the accuracy of numerical Jacobians. A method is developed for calculating an optimal perturbation magnitude that can minimize the error in numerical Jacobians. The accuracy of the numerical Jacobians is improved significantly by using the optimal perturbation magnitude. The effects of the accuracy of numerical Jacobians on the convergence of the flow solver are also investigated. In order to reduce the execution time for numerical Jacobian evaluation, flux vectors with perturbed flow variables are calculated only for neighbouring cells. A sparse matrix solver that is based on LU factorization is used. Effects of different flux splitting methods and higher-order discretizations on the performance of the solver are analysed. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: 2-D Euler equations; Newton's method; numerical Jacobians; analytical Jacobians; upwind flux splitting methods

1. INTRODUCTION

Newton'smethod is a widely used technique for finding the solution of a non-linear system of algebraic equations and providing quadratic convergence. Considering the flow domain as a whole makes the Newton's method stable; however, a good initial guess may still be required

*Correspondence to: Sinan Eyi, Department of Aerospace Engineering, Middle East Technical University, Ankara 06531, Turkey.
†E-mail: seyi@metu.edu.tr
‡Associate Professor.
§E-mail: omer@ae.metu.edu.tr
¶Graduate Research Assistant.

Received 15 January 2005
Revised 16 April 2005
Accepted 24 April 2005

Copyright © 2005 John Wiley & Sons, Ltd.

for convergence. Newton's method also requires the calculation of a Jacobian matrix that may be very large depending on the CFD model used. Although this method has been available for a long time, the high storage and factorization costs of large Jacobian matrices have limited the method's use. The recent development of very powerful computers and efficient matrix solvers has now made Newton's method more applicable for the solution of large systems of equations. Analytical or numerical methods can be used to calculate the Jacobian matrix. In the analytical method, the Jacobian matrix entries can be derived accurately. However, as the discretization of the fluid flow equations increases in complexity, this derivation becomes more difficult. Compared to the analytical method, numerical calculations of the Jacobians are much easier. Even for complex discretizations, numerical Jacobians can be easily evaluated. However, the numerical method may have accuracy problems and may need longer computation time. Thus, accurate and fast calculations of numerical Jacobians are very important for the performance of Newton's method.

In order to improve the solution of the Jacobian matrix, Wigton [1] described a technique called nested dissection node reordering. This technique significantly reduces storage requirements and factorization time. Similar techniques are included in today's advanced sparse matrix solvers like the one used in this study. To remove the difficulties in analytical differentiations, Wigton used the symbolic manipulation expert system MACSYMA. Venkatakrishnan [2] used nested dissection with advanced sparse matrix inversion routines, and introduced a diagonal term modification that improves convergence even from a poor initial guess. This modification is also applied in this study. Later, Venkatakrishnan [3] implemented the diagonal term modification for preconditioned conjugate gradient methods. Orkwis [4] compared the performance of several Newton's and quasi-Newton's method solvers and showed that quasi-Newton's methods could be more efficient than the exact Newton's method. Like Wigton, Orkwis used the symbolic manipulation system, MACSYMA, to calculate analytical Jacobian matrix entries. Orkwis noted that these Jacobians took approximately 40 000 lines of Fortran code to implement, some of which were not vectorizable. In order to improve the performance of quasi-Newton's methods, Orkwis employed diagonal term modification and Jacobian matrix freezing.

Whitfield and Taylor [5] presented one of the first implementations of numerical Jacobian calculation in a Newton-relaxation solver. In their study, a high-order flux difference splitting scheme was used. Since obtaining the Jacobian matrix analytically is impractical for such a discretization, they used a numerical method for Jacobian evaluation. Later, Vanden and Whitfield [6] applied direct and iterative methods to solve 3-D Euler equations. In the solution of block-tridiagonal systems, they used LU factorization followed by forward or backward substitution. Instead of a conventional matrix structure, a diagonal plane structure was presented to decrease the memory requirement. Orkwis and Vanden [7] compared numerical and analytical approaches for forming the Jacobian matrix and noted that the numerical approach is simpler and more practical than the analytical approach. They explained differentiation procedures for deriving the analytical and numerical Jacobians in detail. Aberle and Shumlak [8] have recently compared the accuracy, convergence, and computation time of the analytical and numerical methods. They have found that accuracy and convergence are nearly identical in the two methods, while analytical formulation requires less execution time.

One of the objectives of this study is to analyse the source of error in numerical Jacobians and to improve the performance of numerical Jacobian evaluation in terms of accuracy and execution time. The other objective is to investigate the effects of the accuracy of Jacobians on

Copyright © 2005 John Wiley & Sons, Ltd.

the convergence of Newton's solver. In this paper, Newton's method is first introduced; and then, the numerical and analytical calculations of the Jacobians are explained. After discussing the structure of the Jacobian matrix used in Newton's method, the solution strategies of the Jacobian matrix are presented. Next, errors in numerical Jacobians are analysed. The effects of finite-difference perturbation magnitude and computer precision on the accuracy of numerical Jacobians are investigated. To find an optimal finite-difference perturbation magnitude with minimum error, a method is developed and verified using a trial–error procedure. Finally, the convergence and CPU time performances of the developed flow solver are compared for different flux splitting schemes and higher-order discretizations.

2. FLOW MODEL

The steady, 2-D Euler equations in generalized coordinates, written in non-dimensional form are

$$\frac{\partial \hat{F}(\hat{W})}{\partial \xi} + \frac{\partial \hat{G}(\hat{W})}{\partial \eta} = 0 \tag{1}$$

Here, the conserved flow variable vector $\hat{W}$, the flux vectors $\hat{F}$ and $\hat{G}$ are

$$\hat{W} = J^{-1}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho e_t\end{bmatrix}, \quad \hat{F} = J^{-1}\begin{bmatrix}\rho U\\ \rho uU + \xi_x p\\ \rho vU + \xi_y p\\ (\rho e_t + p)U\end{bmatrix}, \quad \hat{G} = J^{-1}\begin{bmatrix}\rho V\\ \rho uV + \eta_x p\\ \rho vV + \eta_y p\\ (\rho e_t + p)V\end{bmatrix} \tag{2}$$

where ρ is the density, u and v are the components of the velocity vector, p is the pressure, e_t is the total energy per unit volume, and U and V are the contravariant velocity components. In Equation (2), J is the coordinate transformation Jacobian, ξ, and η are the curvilinear coordinates, and $\xi_x, \xi_y, \eta_x, \eta_y$ are the transformation metrics.

The differential form of the steady 2-D Euler equations given in Equation (1) can be discretized for an arbitrary quadrilateral control volume as

$$\frac{\delta_\xi \hat{F}}{\Delta \xi} + \frac{\delta_\eta \hat{G}}{\Delta \eta} = 0 \tag{3}$$

Writing the spatial derivatives of the flux vectors conservatively as flux balances across the cell, Equation (3) can then be written as

$$(\hat{F}_{i+1/2,j} - \hat{F}_{i-1/2,j}) + (\hat{G}_{i,j+1/2} - \hat{G}_{i,j-1/2}) = 0 \tag{4}$$

where $i \pm 1/2$ and $j \pm 1/2$ denote cell interfaces. The fluxes are calculated at the cell faces by using the flow variables interpolated from the cell centre values according the order of spatial discretization.

Copyright © 2005 John Wiley & Sons, Ltd.

2.1. Flux splitting

The Euler equations have convective fluxes that represent the inviscid phenomena. Although central difference schemes are easy to implement to calculate the fluxes, they require artificial dissipation. Another approach is to use the upwind schemes, which require no explicit artificial dissipation. In this study, upwind flux splitting schemes are used for the spatial discretization of the flux vector.

In Newton's flow solver, any flux scheme could be employed in the discretized residual and Jacobian calculations. Steger and Warming [9] showed that, using wave splitting, convective flux vectors can be split into two parts according to the sign of the eigenvalues of the flux Jacobian. Implementing this splitting procedure, the upwind discretized form of the steady, 2-D Euler equations can be written as

$$[\hat{F}^+(\hat{W}^-_{i+1/2,j}) + \hat{F}^-(\hat{W}^+_{i+1/2,j})] - [\hat{F}^+(\hat{W}^-_{i-1/2,j}) + \hat{F}^-(\hat{W}^+_{i-1/2,j})]$$
$$+[\hat{G}^+(\hat{W}^-_{i,j+1/2}) + \hat{G}^-(\hat{W}^+_{i,j+1/2})] - [\hat{G}^+(\hat{W}^-_{i,j-1/2}) + \hat{G}^-(\hat{W}^+_{i,j-1/2})] = 0 \tag{5}$$

Van-Leer [10] introduced a different method for splitting fluxes, in which the flux vector is a function of the contravariant Mach number. Roe [11] extended the flux splitting idea by considering the Riemann problem of discontinuous flow variables, and used constant Jacobian matrices to define the flux vectors as functions of Roe-averaged flow variables.

2.2. Higher-order schemes with limiters

The interface fluxes given in Equation (5) require the flow variables on cell faces, $\hat{W}^{\pm}_{i\pm1/2,j}$ or $\hat{W}^{\pm}_{i,j\pm1/2}$. A zeroth-order interpolation can be realized easily as follows:

$$\hat{W}^-_{i+1/2} = \hat{W}_i \quad \hat{W}^+_{i+1/2} = \hat{W}_{i+1} \tag{6}$$

For higher-order spatial discretizations, these conserved variables are determined from an upwind-biased interpolation of the primitive variables at cell centres. This is called MUSCL (monotonic upstream-centred scheme for conservation laws) [12] and its general form can be written as

$$\begin{aligned} \hat{W}^-_{i+1/2} &= \hat{W}_i + \left\{ \frac{\phi}{4}[(1-\kappa)\nabla + (1+\kappa)\Delta] \right\}_i \\ \hat{W}^+_{i+1/2} &= \hat{W}_{i+1} - \left\{ \frac{\phi}{4}[(1+\kappa)\nabla + (1-\kappa)\Delta] \right\}_{i+1} \end{aligned} \tag{7}$$

where the difference operators are

$$\Delta_i = \hat{W}_{i+1} - \hat{W}_i \quad \nabla_i = \hat{W}_i - \hat{W}_{i-1} \tag{8}$$

The order of discretization and the type of differencing are determined by assigning different values to ϕ and κ. In higher-order spatial discretizations, numerical oscillations are expected where large flow gradients occur. In order to control and reduce the order of stencil in these regions, flux limiters can be used. The flux limiter ϕ can be written as a function of the

Copyright © 2005 John Wiley & Sons, Ltd.

difference ratio r, which can be defined as

$$r_i = \frac{\Delta_i + \in}{\nabla_i + \in} \tag{9}$$

where $\in$ is a small number to prevent division by zero in the zero gradient flow regions. Then, Equation (7) can be rewritten as

$$\begin{aligned} \hat{W}^-_{i+1/2} &= \hat{W}_i + \left\{ \frac{\phi(r)}{4}[(1-\kappa)\nabla + (1+\kappa)\Delta] \right\}_i \\ \hat{W}^+_{i+1/2} &= \hat{W}_{i+1} - \left\{ \frac{\phi(1/r)}{4}[(1+\kappa)\nabla + (1-\kappa)\Delta] \right\}_{i+1} \end{aligned} \tag{10}$$

In this study, a Van Albada limiter is used and the limiter function is defined as

$$\phi(r) = \frac{r + r^2}{1 + r^2} \tag{11}$$

In the regions of small gradient flows, the value of ϕ approaches one and the algorithm actually uses no limiter. On the contrary, in the regions of very large gradient flows, its value approaches zero and the algorithm reduces the interpolation to the first order.

3. SOLUTION METHOD

The system of non-linear equations of discretized governing equations can be written in the form

$$\hat{R}(\hat{W}) = 0 \tag{12}$$

where $\hat{R}$ is the residual vector of the system. Then, Newton's method can be formulated as

$$\left(\frac{\partial \hat{R}}{\partial \hat{W}} \right)^n \Delta \hat{W}^n = -R(\hat{W}^n) \tag{13}$$

The increment $\Delta \hat{W}$ at the nth iteration is found by solving the above system. The new values of the flow variable vector $\hat{W}$ at the $(n+1)$th iteration can be calculated as

$$\hat{W}^{n+1} = \hat{W}^n + \Delta \hat{W}^n \tag{14}$$

In Newton's solver, the Jacobian matrix has to be evaluated. The Jacobian matrix elements are the derivatives of the residual vector with respect to the flow variables vector. To investigate the accuracy of the numerical Jacobians, both the numerical and analytical Jacobians are calculated. In the study, analytical Jacobians are derived only for the first-order Steger–Warming discretization. The accuracy of numerical Jacobians is investigated by comparing the numerical and analytical Jacobians with the same discretization.

Copyright © 2005 John Wiley & Sons, Ltd.

3.1. Analytical Jacobian derivation

It is obvious that the calculation of flux and residual Jacobians by manual differentiation is time-consuming and likely to be erroneous. However, for Euler fluxes with the first-order Steger–Warming flux splitting discretization, this procedure becomes somewhat feasible and can be calculated by hand [13]. The residual Jacobians simply represent the combinations of flux Jacobians. Using Equation (5), the residual vector can be defined as

$$\begin{aligned}\hat{R}_{i,j} = &[\hat{F}^+(\hat{W}_{i,j}) + \hat{F}^-(\hat{W}_{i+1,j})] - [\hat{F}^+(\hat{W}_{i-1,j}) + \hat{F}^-(\hat{W}_{i,j})] \\ &+ [\hat{G}^+(\hat{W}_{i,j}) + \hat{G}^-(\hat{W}_{i,j+1})] - [\hat{G}^+(\hat{W}_{i,j-1}) + \hat{G}^-(\hat{W}_{i,j})]\end{aligned} \tag{15}$$

From the above equation, the residual Jacobians can be represented as follows:

$$\begin{gathered}\frac{\partial \hat{R}_{i,j}}{\partial \hat{W}_{i,j}} = \hat{A}^+_{i,j} - \hat{A}^-_{i,j} + \hat{B}^+_{i,j} - \hat{B}^-_{i,j} \\ \frac{\partial \hat{R}_{i,j}}{\partial \hat{W}_{i+1,j}} = \hat{A}^-_{i+1,j} \quad \frac{\partial \hat{R}_{i,j}}{\partial \hat{W}_{i,j+1}} = \hat{B}^-_{i,j+1} \quad \frac{\partial \hat{R}_{i,j}}{\partial \hat{W}_{i-1,j}} = -\hat{A}^+_{i-1,j} \quad \frac{\partial \hat{R}_{i,j}}{\partial \hat{W}_{i,j-1}} = -\hat{B}^+_{i,j-1}\end{gathered} \tag{16}$$

where $\hat{A}$ represents the Jacobian matrices of ξ-directional flux vector $\hat{F}$, and $\hat{B}$ represents the Jacobian matrices of η-directional flux vector $\hat{G}$.

The main advantage of the analytical method is that the residual Jacobians can be calculated accurately. The order of error in the analytical method can be as small as the round-off error. Although the analytical method requires code development, running an analytical code is very fast. However, as the complexity of discretized residual equations increases, the derivation of the analytical Jacobian becomes more difficult and therefore, the numerical method should be considered for Jacobian calculation.

3.2. Numerical Jacobian calculation

The best alternative for analytical Jacobian evaluation is to compute the Jacobians numerically as accurately as possible. Using a small number as the finite-difference perturbation magnitude ε, the numerical Jacobians can be calculated by one-sided derivatives as follows [14]:

$$\frac{\partial \hat{R}_i}{\partial \hat{W}_j} = \frac{R_i(\hat{W} + \varepsilon e_j) - R_i(\hat{W})}{\varepsilon} \tag{17}$$

where e_j is the jth unit vector. In this vector, the value of the jth component is one, and the values of all other components are zero.

The value of ε does not have to be positive. It is obvious that by employing a positive ε, the derivative will be forward differenced, while with a negative ε, a backward differenced derivative can be obtained. The choice of the sign of the finite-difference perturbation magnitude becomes very important in some cases. For example, in the Steger–Warming scheme, the flux vectors are non-differentiable where the eigenvalues of flux Jacobians change sign. The perturbation of the flow variables may change the sign of the eigenvalues, and as a result, large differences between analytical and numerical Jacobians may be observed. In order to

Copyright © 2005 John Wiley & Sons, Ltd.

prevent this, the effect of perturbation on the sign of the eigenvalues must be checked, and the backward or forward differencing should be implemented accordingly.

In the numerical method, Jacobian evaluation does not require the large coding effort needed in the analytical method. The same residual discretization is used for both the original and perturbed flow variables. This reuse of the same code is one of the big advantages of the numerical approach. Moreover, for cases in which the analytical Jacobians are too difficult to obtain, such as in higher-order discretizations, numerical Jacobians can be obtained without any difficulty. Two main disadvantages of the numerical method are the accuracy problem and long computation time. The accuracy of the numerical Jacobians is related to the finite-difference perturbation magnitude. Using an optimal perturbation magnitude that produces the minimum error may improve the accuracy of numerical Jacobians. The main reason for having a long computation time is the need for calculating the residual vector with each perturbed flow variable in the whole domain. For a given cell, the residual is only a function of flow variables in that cell and the neighbouring cells according to the discretization used. In order to reduce computation time, the perturbed residual is computed only with flow variables in these cells. For first-order discretizations, in addition to the given cell, four neighbouring cells are used. Considering four flow variables in each cell, 20 perturbed residual vector evaluations are required in the numerical Jacobian calculation for the given cell. In second-order discretizations, using eight neighbouring cells in addition to the given cell, 36 perturbed residual vector evaluations are required. In this way, the numerical Jacobian evaluation method may become faster although the speed of the analytical method may not be reached.

3.3. Matrix structure

The Jacobian matrix requires partial derivatives of every residual equation with respect to every flow variable. Fortunately, most of these derivatives are zero because of the fact that the discretized residual equations only depend on local flow variables. Although there is no need to compute and store the zero elements of the Jacobian matrix, full matrix solvers require the whole matrix to be constructed. The storage and solution costs of this type of matrix structure are prohibitively expensive for large problems. Thus, storing only the non-zero elements of the Jacobian matrix and employing a sparse matrix solver are very important for improving the solver's efficiency.

The Jacobian matrix of the complete system is square, with dimensions equal to the total number of flow variables in the system. Considering the test case with a 65×49 grid size, the total number of variables exceeds ten thousand and the Jacobian matrix has hundreds of millions of elements. For first-order upwind discretizations, a five-point stencil is required, which produces a block diagonal matrix made up of five 4×4-blocks. In second-order discretizations, a nine-point stencil is employed, which produces a block diagonal matrix made up of nine 4×4-block bands. Thus, all elements of the Jacobian matrix, except for these block bands and the boundary entries, are zero.

3.4. Matrix solution strategies

In this study, the UMFPACK (unsymmetric-pattern multifrontal package) sparse matrix solver package [15] is used in order to solve the linear system of equations. In this method, the full matrix is converted into sparse storage mode and then factorized using a sequence of small

Copyright © 2005 John Wiley & Sons, Ltd.

dense frontal matrices by LU factorization. The usage of a sparse matrix solver increases the efficiency of the flow solver significantly.

In Newton's method, the Jacobian matrix has to be recomputed and refactored at each iteration, since the matrix is a non-linear function of the changing flow variables. However, great gains in efficiency can be achieved by only computing and factoring the Jacobian matrix once, and using this frozen Jacobian matrix for all subsequent iterations. In the early stages of iterations, Newton's method has a tendency to diverge, and freezing the Jacobian matrix in these stages may not be a good choice. Once the high convergence rate is achieved, the Jacobian matrix can be frozen. Even though freezing the Jacobian matrix may increase the number of iterations, a great reduction in CPU time may be achieved.

Newton's method requires a good initial guess for convergence. This is a considerable drawback. In this study, flow variables are initialized with their free-stream values, although it may be a poor guess. Several ideas are available to modify Newton's method to improve stability. For example, a time-like term can be added to the diagonal of the Jacobian matrix to make it more diagonally dominant [2]. Increased diagonal dominance leads to a more stable linear solution. With the addition of a time-like term, the modified Newton's method becomes

$$\left(\frac{1}{\Delta t}[I] + \frac{\partial \hat{R}}{\partial \hat{W}}\right)^n \Delta \hat{W}^n = -R(\hat{W}^n) \tag{18}$$

As $\Delta t \to \infty$, the original Newton's method can be constructed. In the modified Newton's method, a small initial value Δt^0 is chosen and a new value of Δt can be obtained using L_2-norm of the residuals as

$$\Delta t^n = \Delta t^0 \frac{\|R(\hat{W}^0)\|_2}{\|R(\hat{W}^n)\|_2} \tag{19}$$

In this study, to improve the convergence from free-stream initial conditions, the modified Newton's method is used. For all calculations Δt^0 is taken as one, and the L_2-norm is computed including all the residuals in the domain. The convergence of this method is slow until Δt gets very large. However, the modification in Newton's method is useful in the early stages of iterations. Later, the solution becomes more accurate, and the diagonal term addition may not be needed. The withdrawal of the diagonal term from the matrix at the proper convergence level significantly reduces the number of iterations and CPU time.

4. ACCURACY OF NUMERICAL JACOBIANS

4.1. Error analysis

In numerical Jacobian calculation, mainly two types of errors occur. These are truncation and condition errors [16]. While truncation error is due to neglected terms in the Taylor's series expansion, condition error is associated with numerical noise and caused by loss of computer precision.

Copyright © 2005 John Wiley & Sons, Ltd.

The first derivative of any function $f(x)$ can be approximated as a forward difference for some perturbation magnitude ε:

$$\frac{\Delta f(x)}{\Delta x} = \frac{f(x+\varepsilon) - f(x)}{\varepsilon} \tag{20}$$

In this approximation, the truncation error due to the neglected terms in the Taylor series expansion can be written as

$$E_T(\varepsilon) = \frac{\partial^2 f(\zeta)}{\partial x^2} \frac{\varepsilon}{2} \tag{21}$$

where $\zeta = [x, x+\varepsilon]$.

Because of computer precision, the exact value of the function $f(x)$ and its computed value $\tilde{f}(x)$ can be different due to round-off error $E(x)$:

$$\begin{aligned} \tilde{f}(x) &= f(x) + E(x) \\ \tilde{f}(x+\varepsilon) &= f(x+\varepsilon) + E(x+\varepsilon) \end{aligned} \tag{22}$$

The first derivative with computed function $\tilde{f}(x)$ can be calculated as

$$\begin{aligned} \frac{\Delta \tilde{f}(x)}{\Delta x} &= \frac{\tilde{f}(x+\varepsilon) - \tilde{f}(x)}{\varepsilon} \\ \frac{\Delta \tilde{f}(x)}{\Delta x} &= \frac{f(x+\varepsilon) - f(x)}{\varepsilon} + \frac{E(x+\varepsilon) - E(x)}{\varepsilon} \\ \frac{\Delta \tilde{f}(x)}{\Delta x} &= \frac{\Delta f(x)}{\Delta x} + E_C(\varepsilon) \end{aligned} \tag{23}$$

where $E_C(\varepsilon)$ is the condition error. Considering a bound of round-off error $E_R = \max\{|E(x)|, |E(x+\Delta x)|\}$, the maximum of the condition error can be written as

$$E_C(\varepsilon) = \frac{2E_R}{\varepsilon} \tag{24}$$

This bound of round-off error can be considered the precision error, which depends on the computer processor and compiler. The precision is defined according to the machine epsilon Σ_M, which is the smallest number that the computer can recognize. A reasonable estimate of Σ_M can be given as follows:

$$\Sigma_M = \frac{1}{2^m} \quad \text{such that } 1 + \Sigma_M > 1 \tag{25}$$

where m is the number of possible highest bits in the binary representation of the mantissa.

All the calculations of this study are realized on a 1.5 GHz Pentium IV dual processor with a compiler with single and double precision options. The machine epsilon Σ_M values of this compiler-computer configuration are found according to Equation (25). For single precision $\Sigma_M \cong 3.0 \times 10^{-8}$, and for double precision $\Sigma_M \cong 5.6 \times 10^{-17}$ values are reached.

Copyright © 2005 John Wiley & Sons, Ltd.

4.2. Optimal perturbation magnitude analysis

An optimization method is performed to find a perturbation magnitude with minimum error. The total error in the numerical calculation of the first derivative of function $f(x)$ is simply the sum of the truncation and condition errors:

$$E_{\mathrm{TOTAL}}(\varepsilon) = \frac{2E_{\mathrm{R}}}{\varepsilon} + \left|\frac{\partial^2 f(\zeta)}{\partial x^2}\right| \frac{\varepsilon}{2} \tag{26}$$

The total error is highly dependent on perturbation magnitude ε. If the perturbation magnitude is too small, the condition error dominates, and if the perturbation magnitude is too large, the truncation error becomes more important. So, there must be an optimal value for the perturbation magnitude that gives a minimum error.

The optimal perturbation magnitude can be estimated as the ε value that makes the derivative of Equation (26) zero:

$$\frac{\partial E_{\mathrm{TOTAL}}(\varepsilon)}{\partial \varepsilon} = -\frac{2E_{\mathrm{R}}}{\varepsilon^2} + \frac{1}{2}\left|\frac{\partial^2 f(\zeta)}{\partial x^2}\right| = 0 \tag{27}$$

$$\varepsilon_{\mathrm{OPT}} = 2\sqrt{\frac{E_{\mathrm{R}}}{\left|\dfrac{\partial^2 f(\zeta)}{\partial x^2}\right|}} \tag{28}$$

The above equation requires the calculation of a second derivative and the bound of round-off error. In many cases, for the second derivative, a finite-difference relation has to be employed, which also brings errors. The order of the second derivatives can be estimated as the order of function values. Since the normalized equations are solved, the order of the flow variables is one. Actually, when the second derivatives of the residuals are calculated for the whole flow field using a finite-difference relation in double precision, the L_1-norm of the second derivative is found to be around three. Hence, assuming the order of the second derivative to be one may not introduce large errors. The bound of round-off error is the precision error and can be taken as the machine epsilon for both single and double precision cases. Taking the second derivative as one, Equation (28) becomes

$$\varepsilon_{\mathrm{OPT}} = 2\sqrt{\Sigma_{\mathrm{M}}} \tag{29}$$

In this study, the optimal value of the finite-difference perturbation magnitude is calculated using Equation (29). In the past, similar methods have been proposed to calculate the appropriate perturbation size for numerical Jacobian calculations [14, 17]. However, the studies on the performance and the reliability of these methods are limited. One of the objectives of this study is to evaluate the performance and the reliability of the present optimization method. The performance of the optimization method is studied in Section 5. In this section, a trial–error-like procedure is performed to validate the reliability of the optimization method. The error in the numerical residual Jacobians is analysed for several finite-difference perturbation magnitudes. Both forward and backward difference schemes are used in numerical Jacobian calculations. Computations are performed with both single and double precisions. The maximum and the average errors are obtained using L_∞- and L_1-norms of the errors between

Copyright © 2005 John Wiley & Sons, Ltd.

numerical and analytical residual Jacobians as

$$E_{\mathrm{MAX}} = \frac{\left\| \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^n) \right)_{\mathrm{an}} - \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^n) \right)_{\mathrm{num}} \right\|_\infty}{\left\| \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^0) \right)_{\mathrm{an}} - \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^0) \right)_{\mathrm{num}} \right\|_\infty}$$

$$E_{\mathrm{AVG}} = \frac{\left\| \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^n) \right)_{\mathrm{an}} - \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^n) \right)_{\mathrm{num}} \right\|_1}{\left\| \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^0) \right)_{\mathrm{an}} - \left(\frac{\partial \hat{R}}{\partial \hat{W}}(\hat{W}^0) \right)_{\mathrm{num}} \right\|_1} \tag{30}$$

4.3. Test case results

The accuracy of numerical Jacobians is studied for a flow field around a 15° ramp geometry. The free-stream Mach number, M_∞ is 2. This geometry and flow condition produce shock waves which will be very suitable for studying the effects of high-flow-gradients on the errors in numerical Jacobians. A 65×49 grid is used, as shown in Figure 1. The free-stream condition and geometry produce a supersonic flow at the inlet and outlet. Thus, at the inlet, flow variables are initialized with their free-stream values, while at the outlet they are interpolated from the inner values. Wall and symmetry boundary conditions are employed at the lower and upper sides of the boundary. In this study, numerical and analytical Jacobians are calculated and compared for the first-order Steger–Warming scheme. Both Jacobians are obtained from an already converged solution whose Mach number contours are given in Figure 2.

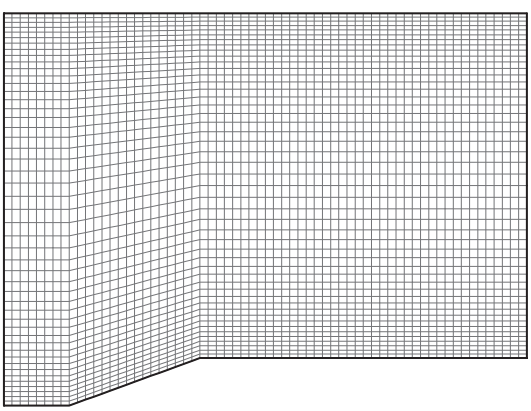

Figure 1. 65×49 grid for ramp geometry.

Copyright © 2005 John Wiley & Sons, Ltd.

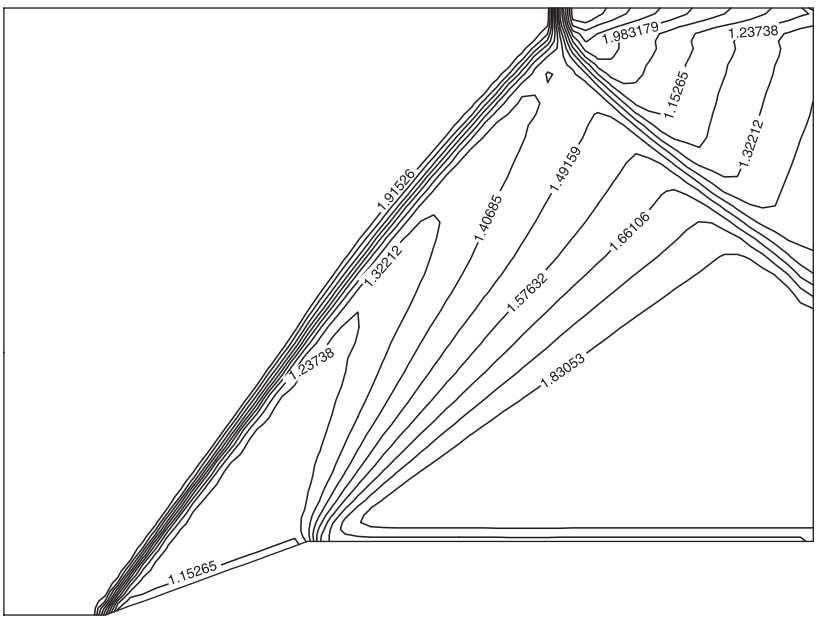

Figure 2. Mach contours of converged solution.

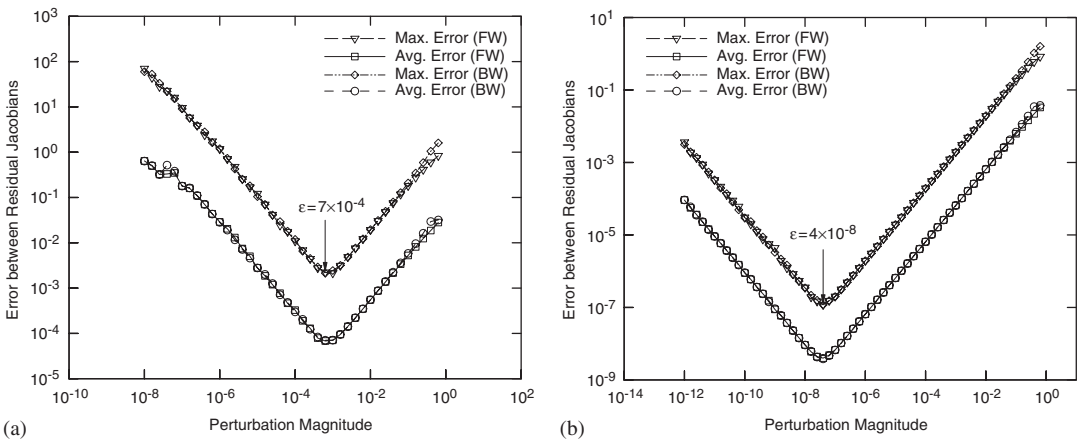

Figure 3. Effect of perturbation magnitude on errors in numerical Jacobians: (a) single precision; and (b) double precision.

With different finite-difference perturbation magnitudes, the change of maximum and average errors in numerical Jacobians is analysed. The numerical Jacobians are calculated using both forward and backward differencing to investigate their effects. In order to understand the effects of computer precision on the accuracy of numerical Jacobians, computations are performed with both single and double precisions. Figure 3 shows the change of maximum

Copyright © 2005 John Wiley & Sons, Ltd.

Table I. Optimization method results for single and double precision.

Precision	Optimization method (Σ_M)		
	Precision error (Σ_M)	Second derivative	ε_{opt} (Σ_M)
Single	3.0×10^{-8}	1.0	3.5×10^{-4}
Double	5.6×10^{-17}	1.0	1.5×10^{-8}

Table II. Comparison of trial–error procedure and the optimization method for single and double precisions.

Precision	Trial–error procedure		Optimization method
	ε_{opt} (max. error)	ε_{opt} (avg. error)	ε_{opt} (Σ_M)
Single	9.8×10^{-4}	6.4×10^{-4}	3.5×10^{-4}
Double	4.0×10^{-8}	4.0×10^{-8}	1.5×10^{-8}

and average errors with perturbation magnitude. It is observed that errors produced in numerical Jacobians with forward and backward difference schemes are in the same order of magnitude. Computations with double precision improve the accuracy of numerical Jacobians. With double precision, the average error can be reduced to the order of 10^{-8}. However, with single precision the average error can be reduced to only an order of 10^{-4}. Figure 3 shows that error in numerical Jacobians is highly dependent on finite-difference perturbation magnitude, and there is an optimal perturbation value that gives the minimum error. As explained in Section 3.2, if the perturbation magnitude is too small, the condition error increases. The value of optimal perturbation can be read as around 4×10^{-8} for double precision and 7×10^{-4} for single precision.

The accuracy of the optimization method for finding a perturbation magnitude with minimum error is verified. In the method, the values of the second derivatives are taken as one, and Equation (29) is used to find the optimal perturbation magnitude. Computations are performed with both single and double precisions, and the results are given in Table I. The optimal values are calculated in the order of 10^{-4} and 10^{-8} for single and double precisions, respectively. The optimal perturbation magnitudes, taken from Figure 3 and calculated from the optimization method, are compared in Table II. Results show that the optimization method can accurately predict the optimal perturbation magnitude. Using Equation (26) with the perturbation magnitudes found by using the optimization method, the minimum errors are calculated in the order of 10^{-4} and 10^{-8} for single and double precisions. These values totally agree with the corresponding error values read from Figure 3.

The change of optimal perturbation magnitude with grid size is also studied. Figure 4 shows the change of error with perturbation magnitude for three different grid sizes: 33×25, 65×49 and 129×97. In numerical Jacobian calculations, forward differencing and double precision are used. Results show that an error in numerical Jacobians slightly decreases as the grid gets finer. However, as the grid size varies, no change in optimal perturbation magnitude is observed. The CPU times to evaluate analytical and numerical Jacobians are also compared for different grid sizes. The ratios of CPU time for numerical Jacobian evaluations to CPU

Copyright © 2005 John Wiley & Sons, Ltd.

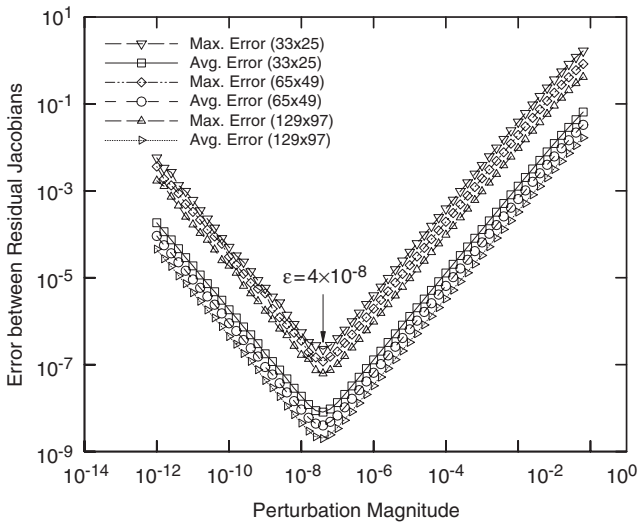

Figure 4. Effect of perturbation magnitude on errors in numerical Jacobians for different grid size.

time for analytical Jacobian evaluations are approximately 2.5, 5 and 10 for $33 \times 25, 65 \times 49$, and 129×97 grids, respectively.

5. SOLVER PERFORMANCE

5.1. Performance analysis

Newton's method requires the solution of a large linear system of equations at each iteration. Considering the test case with a 65×49 grid and four flow variables at each cell, the total number of variables is 13 200 and the Jacobian matrix has 174×10^6 elements. Keeping this huge matrix requires around 3 GB memory, and the solution of such a matrix at each iteration takes an extremely long computer time. An efficient solution of a linear system of equations will improve the performance of the solver significantly. In this study, the UMFPACK sparse matrix solver package is used to minimize the storage and factorization costs. In the first-order upwind discretizations, a five-point stencil produces five 4×4-block bands. A full matrix solver requires the storage and solution of the whole matrix, including zeros; with UMFPACK the performance of the solver can be improved significantly. After eliminating the unnecessary zero entries, the size of the matrix drops to 2–3 MB and a large reduction in CPU time is achieved.

The use of a diagonal term addition and Jacobian matrix freezing is investigated to improve the efficiency of the solver. Deciding when to remove the diagonal term and to freeze the Jacobian matrix is very important to obtain the best performance of the solver. The performance of the solver is measured as the CPU time to obtain a converged solution with a specified maximum density residual tolerance: 1.0×10^{-5} for single precision and 1.0×10^{-13}

Copyright © 2005 John Wiley & Sons, Ltd.

for double precision. The time-like term is removed when the Δt value given by Equation (19) reaches Δt_{rm}, and the Jacobian matrix is frozen when the average density residual reaches to R_{frz}. In order to get the best performance in each run, a trial–error study is performed to determine the values of Δt_{rm} and R_{frz}. This study shows that there is no single value for Δt_{rm}, but the value of R_{frz} can be fixed as 1.0×10^{-2} for single precision, and 1.0×10^{-4} for double precision. Although the number of iterations may increase with the use of Jacobian freezing, the CPU time spent for the converged solution decreases when skipping the Jacobian calculation. With these trial–error procedures, the best combinations of parameters for removing the diagonal terms and freezing the Jacobians are found, and these parameters are used in the performance analysis of Newton's solver. In this section, the performance of Newton's method is demonstrated for the same geometry and flow condition used in Section 4. In the solution of Newton's method, flow variables are initialized with the free-stream solution.

5.2. Test case results

In this part of the study, the accuracy of numerical Jacobians on the convergence of Newton's method is studied. The convergence histories with analytical and numerical Jacobians are compared for the first-order Steger–Warming scheme. Numerical Jacobians are obtained with different perturbation magnitudes. The convergences with optimal and other perturbation magnitudes are compared. Forward and backward difference schemes are used for numerical Jacobian calculations, and the convergences with these schemes are compared. Computations are performed with both single and double precisions in order to see the effect of computer precision on the convergence of the solver. In Table III and Figure 5, the performances of the solver with analytical and numerical Jacobians are compared. The following observations have been made in relation to the performance of the solver with numerical Jacobians. In computations with single precision, the perturbation magnitude significantly affects the solver's convergence. The convergence of Newton's method with double precision is less sensitive to the perturbation magnitude. Although the speed of the analytical method may not be reached, the converged solution with the least CPU time is always obtained with optimal perturbation magnitudes, which corresponds to 7×10^{-4} for single precision and 4×10^{-8} for double precision. With the optimal perturbation magnitude, the convergences with analytical and numerical Jacobians are almost identical. Using a forward or backward difference scheme in numerical Jacobian calculations does not significantly affect the convergence of the solver.

Table III. Effects of perturbation magnitude on convergence history with Steger–Warming discretization.

Precision	Jacobian	CPU time (s)		Number of iterations		Δt_{rm}	R_{frz}
		Forward	Backward	Forward	Backward		
Single	Analytical	642.98		24			
	Num. $\varepsilon = 7 \times 10^{-3}$	Diverged	Diverged	Diverged	Diverged		
	Num. $\varepsilon = 7 \times 10^{-4}$	2793.12	2098.23	24	24	3	10^{-2}
	Num. $\varepsilon = 7 \times 10^{-5}$	Diverged	Diverged	Diverged	Diverged		
Double	Analytical	469.76		26			
	Num. $\varepsilon = 4 \times 10^{-4}$	3916.30	3998.25	27	28		
	Num. $\varepsilon = 4 \times 10^{-8}$	3868.71	3919.83	26	26	3	10^{-4}
	Num. $\varepsilon = 4 \times 10^{-12}$	3856.80	4059.56	27	28		

Copyright © 2005 John Wiley & Sons, Ltd.

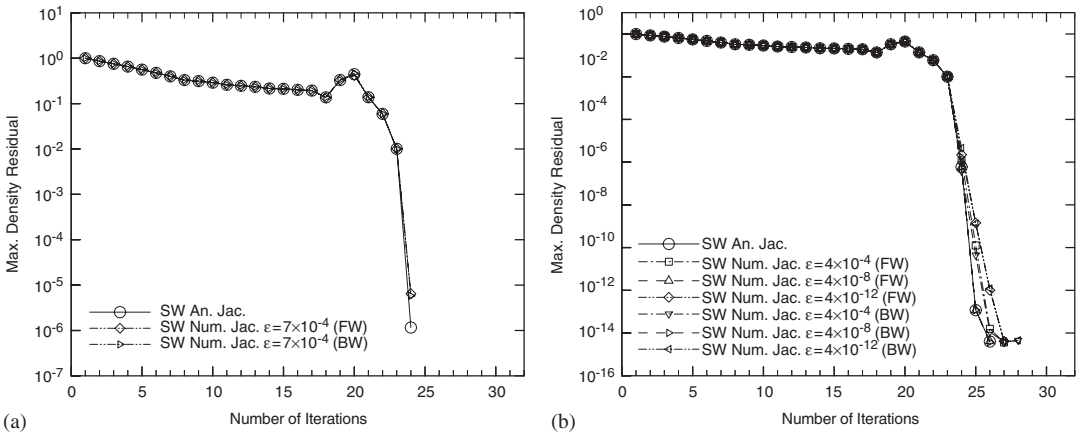

Figure 5. Effects of perturbation magnitude on convergence history using Steger–Warming discretization: (a) single precision; and (b) double precision.

Table IV. Effects of grid size on convergence history with Steger–Warming discretization.

Grid size	Jacobian	CPU time (s)	Number of iterations	Δt_{rm}	R_{frz}
33×25	Analytical	12.33	11		
	Num. $\varepsilon = 4 \times 10^{-8}$	25.24	11	1.5	10^{-4}
65×49	Analytical	469.76	26		
	Num. $\varepsilon = 4 \times 10^{-8}$	3868.71	26	3	10^{-4}
129×97	Analytical	63201.34	218		
	Num. $\varepsilon = 4 \times 10^{-8}$	535129.62	218	6	10^{-4}

Results show that with double precision the maximum density residual can be reduced to the order of 10^{-15}, while with single precision the same residual converges to only an order of 10^{-6}. The CPU times of the runs with single and double precisions are in the same order of magnitude. Thus, the computations with double precision significantly improve the order of convergence without affecting CPU time considerably. A trial–error study using the analytical Jacobians shows that removing the time-like term after Δt value reaches three gives the best performance.

In order to determine the effects of grid size on the performance of the solver, the previous parametric study is realized for different grid sizes. The convergences of the solver with the analytical and numerical Jacobians are compared with three different grid sizes of $33 \times 25, 65 \times 49$, and 129×97. Numerical Jacobians are obtained with forward differencing using the optimal perturbation magnitude value of 4×10^{-8}. The computations are performed with double precision. Table IV and Figure 6 show that identical convergence histories are obtained with the analytical and numerical Jacobians. By using the different number of grid points, effects of the size of linear systems on the performance of the solver are also analysed. It is observed that CPU time increases significantly with grid size. The increase in CPU time is mainly related to the increase in the number of iterations. It is observed that change in

Copyright © 2005 John Wiley & Sons, Ltd.

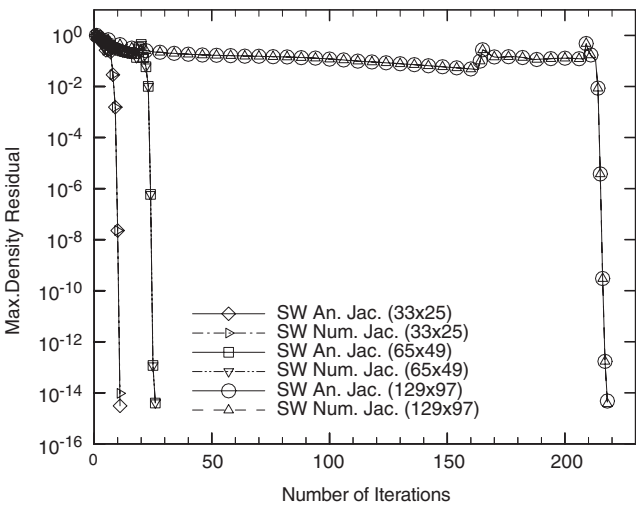

Figure 6. Effects of grid size on convergence history using Steger–Warming discretization.

grid size does not affect the memory requirement significantly. With the present size of the linear system, the CPU and memory requirements are in an acceptable range. However, for much larger systems, especially in 3-D problems, the direct solution of sparse matrices may not be very efficient. In this case, iterative solvers, such as Krylov methods, may be more appropriate [18]. As given in Table IV, although the value of Δt_{rm} increases with the grid size, the value of R_{frz} remains the same.

The performance of Newton's method is evaluated for different flux splitting schemes. The first-order Steger–Warming, Van Leer and Roe schemes are used for flux calculations. The benefits of using the same flux calculation scheme for both Jacobian and residual calculation are analysed. The convergences of the flow solver with analytical Steger–Warming Jacobians, and Van Leer and Roe discretized residuals are compared. Again, the appropriate values of parameters for removing the diagonal terms and freezing the Jacobian matrix are used to reflect the best performance of the solver. The numerical Jacobians are calculated using forward differencing, and the calculations are realized with double precision. The performance analysis with the Steger–Warming scheme has already been studied in Table IV and Figure 5(b). The effects of using Van Leer and Roe flux splitting schemes on the convergence of Newton's method are analysed in Table V and Figure 7. It was observed that with their own numerical Jacobians, the Steger–Warming scheme has the fastest convergence in terms of both iterations and CPU times, and the Roe scheme is the slowest one. Figure 7 shows that using different schemes in Jacobian and residual calculations makes the solver performance worse. Van Leer or Roe residuals with Steger–Warming analytical Jacobians converge much more slowly than Van Leer or Roe residuals with their own numerical Jacobians. The Roe residual with its numerical Jacobians converges in 55 iterations, while with the use of Steger–Warming analytical Jacobians the convergence is possible after 250 iterations. However, considering Table V, the convergence with analytical Jacobians still requires less CPU time with respect to numerical Jacobians. In the solutions with Van Leer and Roe schemes, the number of iterations and

Copyright © 2005 John Wiley & Sons, Ltd.

Table V. Effects of perturbation magnitude on convergence history with Van Leer and Roe discretizations.

Scheme	Jacobian	CPU time (s)	Number of iterations	Δt_{rm}	R_{frz}
Van Leer	SW analytical	803.33	56	3	10^{-4}
	Num. $\varepsilon = 4 \times 10^{-4}$	4607.94	32		
	Num. $\varepsilon = 4 \times 10^{-8}$	4559.52	32		
	Num. $\varepsilon = 4 \times 10^{-12}$	4593.69	33		
Roe	SW analytical	3549.57	257	5	10^{-4}
	Num. $\varepsilon = 4 \times 10^{-4}$	8296.57	56		
	Num. $\varepsilon = 4 \times 10^{-8}$	8202.54	55		
	Num. $\varepsilon = 4 \times 10^{-12}$	8268.18	58		

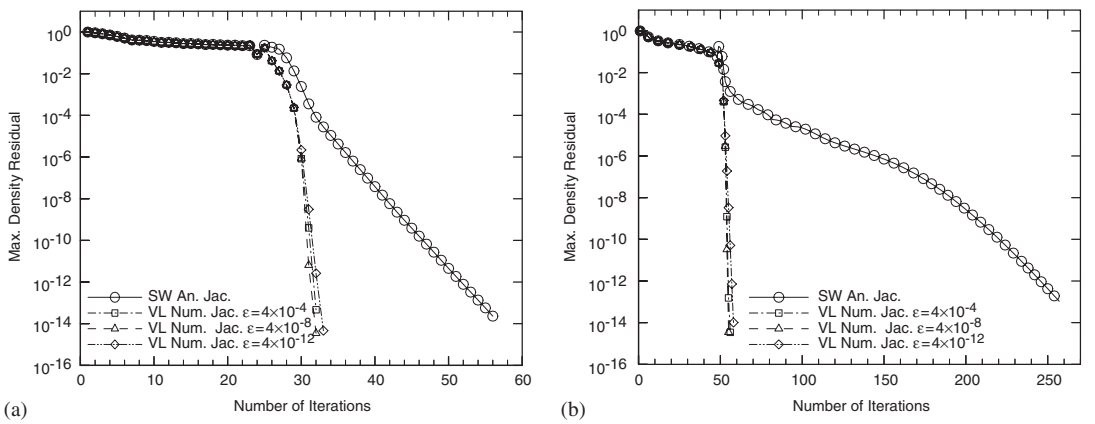

Figure 7. Effects of perturbation magnitude on convergence history using Van Leer and Roe discretizations: (a) Van-Leer scheme; and (b) Roe scheme.

CPU times to obtain converged solutions are expected to be much smaller if the analytical Jacobian and residual calculations use the same schemes. Results also show that the value of Δt_{rm} changes with the flux calculation scheme, but the value of R_{frz} remains the same.

The effects of higher-order schemes on the performance of Newton's solver are studied. The second-order Steger–Warming, Van Leer and Roe schemes are used for flux calculations. The Van Albada limiter is implemented in the extrapolation of the flow variables. Again, the numerical Jacobians are calculated using forward differencing and double precision. The convergences of the solver with analytical and numerical Jacobians are compared. In the solutions with analytical Jacobians, Jacobians are based on the first-order Steger–Warming scheme, and discretized residuals are based on the second-order Steger–Warming, Van Leer and Roe schemes. In the solutions with numerical Jacobians, the second-order Steger–Warming, Van Leer and Roe schemes are used in both Jacobian and discretized residual calculations. Numerical Jacobians are calculated with different perturbation magnitudes. To get a better convergence, the appropriate values of parameters for diagonal term addition and Jacobian matrix freezing are used. Figure 8 shows the convergence histories of the solver with different second-order schemes. Results show that the numerical Jacobians with optimal

Copyright © 2005 John Wiley & Sons, Ltd.

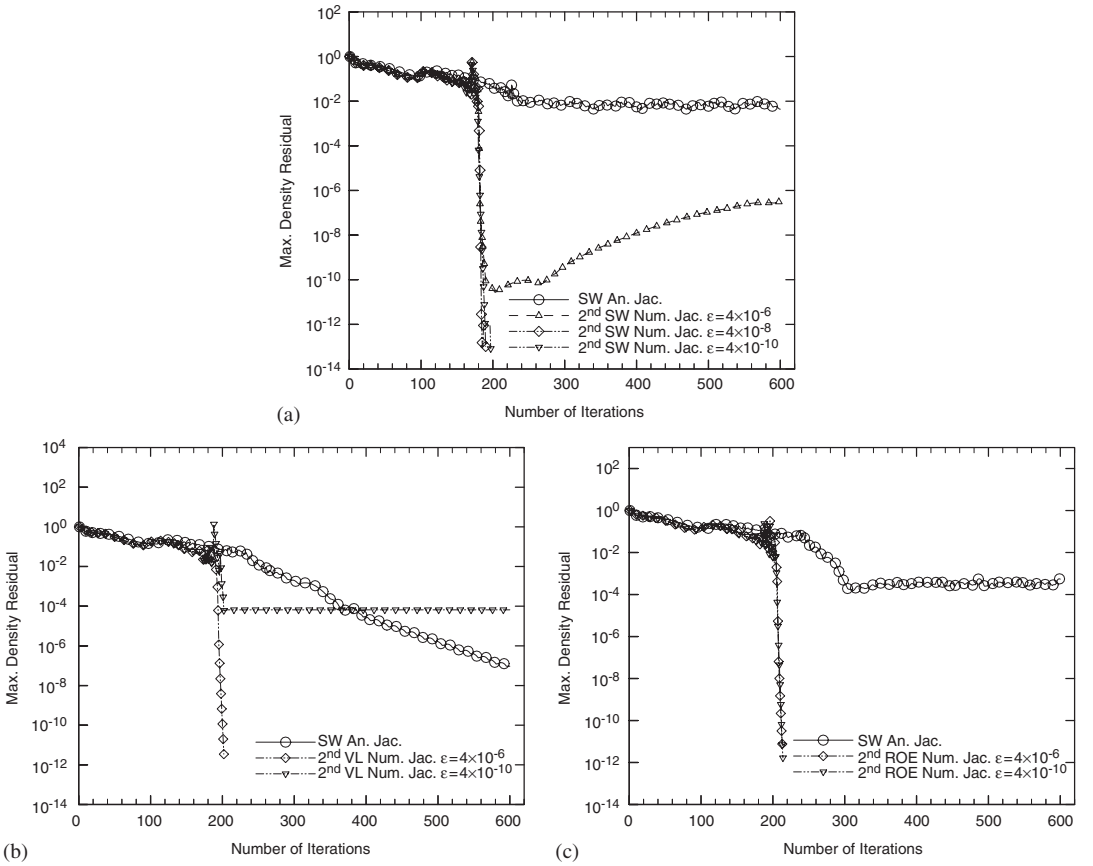

Figure 8. Effects of perturbation magnitude on convergence history using 2nd order discretizations: (a) Steger–Warming scheme; (b) Van-Leer scheme; and (c) Roe scheme.

perturbation magnitudes always give the best convergence. In all three flux calculations, the maximum residual reduction can be achieved with numerical Jacobians and optimal perturbation magnitudes. Compared to the first-order schemes, the second-order schemes are more sensitive to perturbation magnitude. Results also show the advantages of using the same flux calculation scheme for both Jacobian and residual calculations. When the first-order analytical Steger–Warming Jacobians are used with the second-order Steger–Warming, Van Leer and Roe discretized residuals, convergence becomes very slow. It is expected that if the analytical Jacobians are evaluated by using the same second-order schemes used in residual calculations, the CPU times and the number of iterations to obtain converged solutions will decrease significantly. However, obtaining analytical Jacobians using second-order schemes may not be easy. So, numerical methods offer advantages for calculating Jacobians that may be difficult or impossible to generate analytically. In general, a convergence problem is observed in the second-order schemes, which may be due to the nature of the flux limiter employed in the flow solver. The performances of each run with the second-order discretization are analysed in

Copyright © 2005 John Wiley & Sons, Ltd.

Table VI. Effects of perturbation magnitude on convergence history with 2nd order discretizations.

Scheme	Jacobian	CPU time (s)	Number of iterations	Δt_{rm}	R_{frz}
SW	SW analytical	11814.16	600+		
	Num. $\varepsilon = 4 \times 10^{-6}$	42874.08	600+		
	Num. $\varepsilon = 4 \times 10^{-8}$	32931.64	190	64	10^{-4}
	Num. $\varepsilon = 4 \times 10^{-10}$	32949.47	197		
Van Leer	SW analytical	9590.41	600+		
	Num. $\varepsilon = 4 \times 10^{-6}$	Diverged	Diverged		
	Num. $\varepsilon = 4 \times 10^{-8}$	34368.53	202	128	10^{-4}
	Num. $\varepsilon = 4 \times 10^{-10}$	43040.22	600+		
Roe	SW analytical	11710.3	600+		
	Num. $\varepsilon = 4 \times 10^{-6}$	Diverged	Diverged		
	Num. $\varepsilon = 4 \times 10^{-8}$	37770.02	220	192	10^{-4}
	Num. $\varepsilon = 4 \times 10^{-10}$	37479.44	214		

Table VI. In all three flux calculations, the CPU times of the runs with optimal perturbations are nearly the same. However, a convergence in the order of 10^{-14} can be achieved only using the Steger–Warming scheme.

6. CONCLUSIONS

A Newton's method was developed for 2-D Euler equations. Upwind flux splitting schemes were used in the finite-volume discretization. The Jacobian matrix was calculated using both analytical and numerical methods. The sources of error in numerical Jacobians were analysed. An optimization method was developed to minimize the error in numerical Jacobian calculations. The performance of Newton's method with numerical Jacobians was analysed. The convergence histories and CPU times of Newton's method with analytical and numerical Jacobians were compared. The effects of evaluating numerical Jacobians with different perturbation magnitudes on the convergence of Newton's solver were studied. The convergence histories of the solver with different flux calculation schemes were analysed.

The finite-difference perturbation magnitude is one of the most important parameters that affects the accuracy of numerical Jacobians. As a function of computer precision, there is an optimal perturbation magnitude at which numerical Jacobians can be calculated with minimum error. The optimization method developed in this study is easy to use and it can accurately predict the optimal perturbation magnitude for numerical Jacobian evaluations. Numerical Jacobians obtained with optimal perturbation improve the convergence of Newton's method. Computer precision is another important parameter that affects the accuracy of numerical Jacobians. Computations with double precision significantly improve the accuracy of numerical Jacobians. Evaluating Jacobians with the numerical method may require longer execution time. In order to reduce this execution time, flux vectors with perturbed flow variables can be calculated for only related neighbouring cells. Using the same flux scheme for both Jacobian and residual calculations improves the convergence of the solver. Calculation of the Jacobian numerically keeps the Jacobian consistent with the residual vector. Therefore,

Copyright © 2005 John Wiley & Sons, Ltd.

numerical Jacobians have advantages, especially in higher-order discretizations in which obtaining analytical Jacobians is more difficult. In the case of starting from a poor initial condition, the time-like term addition to the matrix diagonal improves the convergence of the solver in early iterations. Jacobian freezing may be another useful strategy for decreasing the execution time. In the solution of large linear systems of equations, the sparse matrix solver, UMFPACK, significantly reduces memory requirements and CPU time.

REFERENCES

1. Wigton LB. Application of MACSYMA and sparse matrix technology to multi-element airfoil calculations. *AIAA Paper 87-1142*, 1987.
2. Venkatakrishnan V. Newton solution of inviscid and viscous problems. *AIAA Journal* 1989; **27**:885–891.
3. Venkatakrishnan V. Preconditioned conjugate gradient methods for the compressible Navier–Stokes equations. *AIAA Journal* 1991; **29**:1092–1100.
4. Orkwis PD. Comparison of Newton's and quasi-Newton's method solvers for the Navier–Stokes equations. *AIAA Journal* 1993; **31**:832–836.
5. Whitfield DL, Taylor LK. Discretized Newton-relaxation solution of high resolution flux-difference split schemes. *AIAA Paper 91-1539*, 1991.
6. Vanden KJ, Whitfield DL. Direct and iterative algorithms for the three-dimensional Euler equations. *AIAA Paper 93-3378*, 1993.
7. Orkwis PD, Vanden KJ. On the accuracy of numerical versus analytical Jacobians. *AIAA Paper 94-0176*, 1994.
8. Aberle C, Schumlak U. Application of analytical methods to computing numerical flux Jacobians. *AIAA Paper 01-31093*, 2001.
9. Steger JL, Warming RF. Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods. *Journal of Computational Physics* 1981; **40**:263–293.
10. Van Leer B. Flux vector splitting for the Euler equations. *ICASE Report 82-30*, 1982.
11. Roe PL. Characteristics-based schemes for the Euler equations. *Annual Review of Fluid Mechanics* 1986; **18**:337–365.
12. Hirsch C. *Numerical Computation of Internal and External Flows*, vols. 1–2. Wiley: Chichester, 1990.
13. Onur O. Effect of the Jacobian evaluation on direct solutions of the Euler equations. *M.S. Thesis*, Middle East Technical University, Ankara, Turkey, 2003.
14. Dennis JE, Schnabel RB. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice Hall: Englewood Cliffs, NJ, 1983.
15. Davis TA. *UMFPACK Version 4.1 User Manual*. University of Florida: Florida, 2003.
16. Gill PE, Murray W, Wright MH. *Practical Optimization*. Academic Press: London, 1992.
17. Kelley CT. *Iterative Methods for Linear and Nonlinear Equations*. Society for Industrial and Applied Mathematics: Philadelphia, PA, 1995.
18. van der Vorst HA. *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press: Cambridge, MA, 2003.

Copyright © 2005 John Wiley & Sons, Ltd.