NASA Contractor Report 178038

ICASE REPORT NO. 85-59

NASA-CR-178038
19860009595

ICASE

ACCURACY OF SCHEMES FOR THE EULER EQUATIONS WITH NON-UNIFORM MESHES

E. Turkel

S. Yaniv

U. Landau

Contract Nos. NAS1-17070 and NAS1-18107

December 1985

INSTITUTE FOR COMPUTER APPLICATIONS IN SCIENCE AND ENGINEERING
NASA Langley Research Center, Hampton, Virginia 23665

Operated by the Universities Space Research Association

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

ACCURACY OF SCHEMES FOR THE EULER EQUATIONS WITH NON-UNIFORM MESHES

E. Turkel
Institute for Computer Applications in Science and Engineering
and
Tel-Aviv University

S. Yaniv and U. Landau
IAI, Israel

ABSTRACT

The effect of non-uniform grids on the solution of the Euler equations is analyzed. We consider a Runge-Kutta type scheme based on a finite volume formulation. We show that for arbitrary grids the scheme can be inconsistent even though it is second-order accurate for uniform grids. An improvement is suggested which leads to at least first-order accuracy for general grids. Test cases are presented in both two- and three-space dimensions. Applications to finite difference and implicit algorithms are also given.

Research for the first author was supported in part by the National Aeronautics and Space Administration under NASA Contract Nos. NAS1-17070 and NAS1-18107 while he was in residence at ICASE, NASA Langley Research Center, Hampton, VA 23665-5225.

1. INTRODUCTION

In recent years much progress has been made in the solution of the steady-state Euler equations in both two and three dimensions. For complex shapes these calculations are usually based on a body-fitted curvilinear grid. For general three-dimensional bodies it is very difficult to construct a body-fitted coordinate system, see e.g., [3]. In particular, the grid system used frequently is not smooth. In some cases there may exist discontinuities in the gradients of the grid while in other cases the gradients vary sharply but not discontinuously.

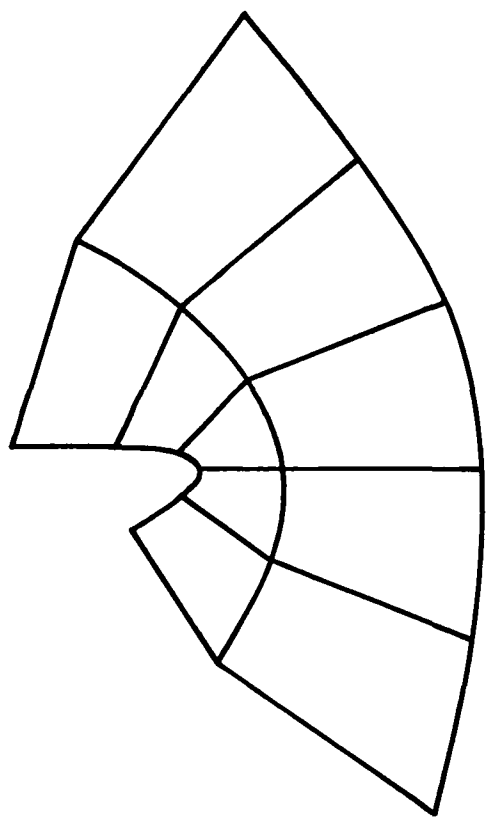

Figure 1

In Figure 1 we present a typical grid constructed by FL057 for a wing-alone configuration. This is an expanded view of the trailing edge for one slice in the spanwise direction. We see that even in this simple case there are large variations between neighboring cells.

2. SPACE DISCRETIZATION

We consider the Euler equations for an inviscid fluid written in conservation form. For simplicity of representation we shall consider two-dimensional flow, however, all the results generalize to three dimensions in a straightforward manner. We thus consider

$$w_t + f_x + g_y = 0. \tag{1}$$

where $w = (\rho, \rho u, \rho v, E)^t$. Rewriting (1) in integral form, we get

$$\frac{\partial}{\partial t} \iint_V w \, dv + \int_S F \cdot n \, ds = 0 \tag{2}$$

where $F = (f,g)^t$ and n is the outward normal. Letting $\bar{w}$ be the cell-averaged values of w, we can rewrite (2) as

$$V \frac{\partial \bar{w}}{\partial t} + \int_{AB} + \int_{BC} + \int_{CD} + \int_{DA} (f dy - g dx) = 0 \tag{3}$$

for the zone shown in Figure 2. At this stage, (3) is still exact. In order to solve (3), we now replace the integrals in (3) by an integration rule.

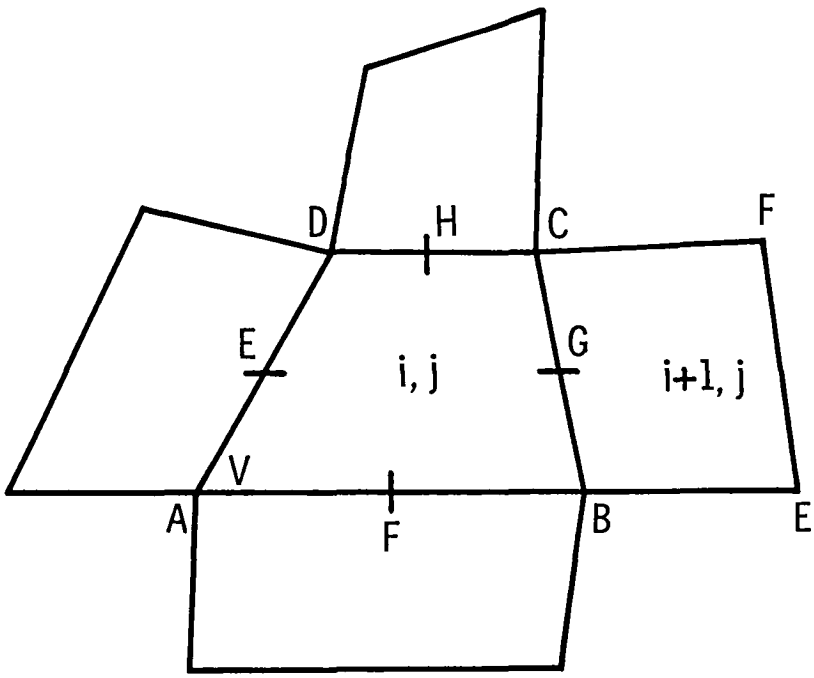

Figure 2

Midpoint Integration Rule

Using the formulation presented in [2] we assume that the metrics are defined at the mesh nodes while the dependent variables are cell averages or else are located in the center of the cell. Replacing the integrals in (3) by the midpoint rule we get

$$V \frac{\partial w_{i,j}}{\partial t} + Q_E + Q_F + Q_G + Q_H = 0 \qquad (4)$$

where $Q = f\Delta y - g\Delta x$. We examine this more carefully by considering the point $G = (i+\frac{1}{2}, j)$, then

$$Q_G = f_G(y_C - y_B) - g_G(x_C - x_B), \tag{5}$$

We are then left with the job of evaluating f_G, g_G. In the original formulation [2], w was calculated at G by

$$w_G = \frac{1}{2}\left(w_{i,j} + w_{i+1,j}\right), \tag{6}$$

and then $f_G = f(w_G)$, $g_G = g(w_G)$. Thus, both cells contribute equally to the value of w along the face independent of the geometry of the grid. Thus, we do not account for non-uniform volumes and stretchings and deformations in the various directions. Another possibility is to average f and g directly.

$$f_G = \frac{1}{2}\left(f_{i+1,j} + f_{i,j}\right). \tag{6a}$$

These two procedures are equivalent to within second-order accuracy. There are some minor differences with respect to shock resolution. In order to check the accuracy of (6) we first consider a one-dimensional example.

The one-dimensional equivalent of (4) - (5) (see Figure 3) is

$$\frac{d}{dt}\int_{x_{j-1/2}}^{x_{j+1/2}} w\,dx + f_{j+1/2} - f_{j-1/2} = 0. \tag{7}$$

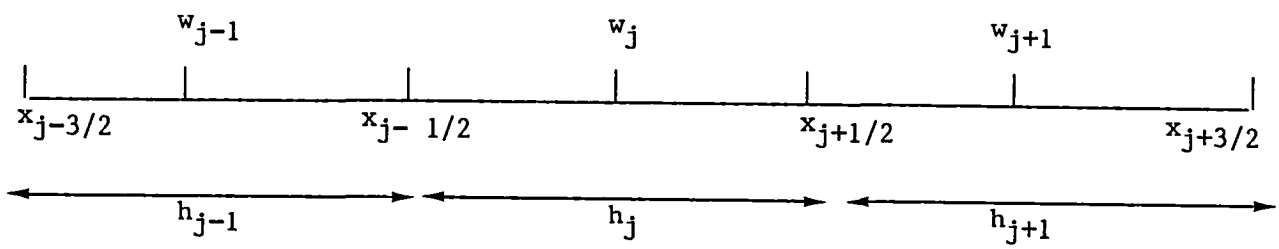

Figure 3

Using the averaging on f given by (6a) we get

$$\frac{dw_j}{dt} + \frac{f_{j+1} - f_{j-1}}{2h_j} = 0. \tag{8}$$

A straightforward Taylor-series expansion shows that (8) is <u>inconsistent</u> unless

$$\frac{h_{j+1} - 2h_j + h_{j-1}}{h_j} = 0(h) \tag{9}$$

where $h = \max_j(h_j)$. Let $r_j = h_{j+1}/h_j$. In [6] we define a grid as algebraic if $r_j = 1 + 0(h^p)$ and exponential if $r_{j+1}/r_j = 1 + 0(h)$. Thus, (8) is consistent only if the grid is algebraic, and is second-order accurate only if we also have $p = 2$.

We next replace (6) by

$$w_{j+1/2} = \frac{h_j\, w_{j+1} + h_{j+1}\, w_j}{h_j + h_{j+1}} \tag{10}$$

and (8) by

$$\frac{dw_j}{dt} + \frac{f(w_{j+1/2}) - f(w_{j-1/2})}{h_j} = 0. \tag{11}$$

This scheme is now first-order accurate and furthermore, it is second-order accurate for all algebraic grids. In [6] it is also shown how to construct a second-order finite difference version for exponentially stretched meshes.

Until now we have discussed the accuracy of the flux differences. In the Runge-Kutta schemes [2,5], it is also necessary to add an artificial

viscosity. This consists of two parts: one is an approximation to a second difference with a nonlinear coefficient that is of higher order in smooth regions. This essentially enforces an entropy-like condition and ensures the correct shock conditions. The second portion of the viscosity is an approximation to a fourth difference and is a linear term. This term is needed in smooth regions to prevent decoupling the odd and even points and to suppress the highest frequencies [5]. These viscosities are now also used in implicit A.D.I. algorithms [4] and also in Lax-Wendroff type methods [1]. Both these viscosities add third-order terms to the truncation error when the grid is sufficiently smooth. However, for rapidly varying grids these viscosities can reduce the order of the method. In particular, we have found the lift coefficient to be sensitive to the coefficients of these viscosity terms.

Implementation - Two-Dimensions

The formula given by (10) is a one-dimensional formula. In order to generalize to two dimensions we use this formula in each direction. Using the original mapping we calculate the position of the center of the cell and also the center of each face. Let (ξ,η) be a plane for which the grid is uniform and (x,y) be the physical plane. We assume that the mesh is given by a mapping T, $(x_i,y_j) = T(\xi_i,\eta_j)$. Then the center of a cell is defined by the image of $(\xi_{i+1/2}, \eta_{j+1/2})$ while the center of the faces are given by the image of $(\xi_{i\pm 1/2}, \eta_j)$ and $(\xi_i, \eta_{j\pm 1/2})$. We note that using the explicit mapping of [3] the mapping must be adjusted so that the center of the outermost cell is defined and is not at infinity. Having found these positions we calculate the distance from the center of the zone to the center of each face.

This gives the distance ℓ and m shown in Figure 4. Formula (10) is then replaced by

$$f_G = \frac{\ell f_{i+1,j} + m f_{i,j}}{\ell + m} \tag{12}$$

$$= \alpha f_{i+1,j} + (1-\alpha) f_{i,y}$$

with $\alpha = \ell/(\ell+m)$.

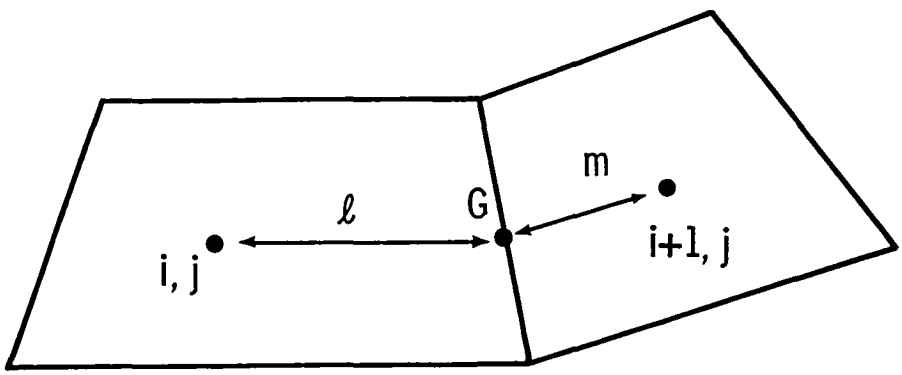

Figure 4

We see from (12) that only the ratio $\ell/(\ell+m)$ need be known for each zone. Hence, once the cell centers and surface centers are found, then only one value need be stored per cell per direction. Thus, in three dimensions we need to store three three-dimensional vectors and then the positions of the cell and face centers can be discarded.

In the codes used, FL052, FL053, FL057, FL059 the metrics are given explicitly by analytic formulae. Hence, it is easy to find the cell and face

centers. For other mesh generation techniques it is not as easy to explicitly find these other positions. One possibility is to construct a grid with twice as many grid lines. Then the nodal positions are found by keeping every other point. The other grid points on the fine grid give the cell and face centers on the final grid. Since we do not need to store all this information, this is not a large cost. A more approximate procedure is to calculate the cell and face centers by simple averaging from the nodal positions. When the grid is highly stretched in the outer field one must be careful calculating the averages in the far field.

We note that the formulas given by (12) are one-dimensional formulas given along each coordinate line given by the mapping and not along x and y coordinate lines. Hence, this improvement accounts for non-uniform stretchings but does not necessarily account for shearing effects between cells. Furthermore, if we wish a constant solution to be preserved in the far field, it is necessary to average w by (12) and then calculate f. Averaging f will destroy constant solutions since the metrics will be averaged differently in each direction.

Trapezoidal Integration Rule

In the previous sections we replaced each line integral by a midpoint rule. Thus, (see Figure 2)

$$\int_B^C f\,dy - g\,dx \approx (f\Delta y - g\Delta x)_G. \tag{13}$$

For a general mesh this formula is only first- order accurate since G is

defined as the image of $(\xi_{i+1/2}, \eta_j)$ and is not necessarily the center of the line BC. Choosing G to be the center of the line BC would cause trouble with the interpolation formulas, e.g., (12). An alternative is to replace the midpoint rule (13) by the trapezoidal rule

$$\int_B^C f\,dy - g\,dx \simeq \frac{1}{2}\Big[\Delta y\big(f(B) + f(C)\big) - \Delta x\big(g(B) + g(C)\big)\Big]. \tag{14}$$

This formula is now second-order accurate for non-uniform meshes. In order to evaluate f at node, e.g., B, we use bilinear interpolation. This interpolation is done in a standard square. Thus, we know the value of w at the cell centers $(x_{i,j})$, $(x_{i+1,j}, y_{i+1,y})$, $(x_{i,j+1}, y_{i,j+1})$ and $(x_{i+1,j+1}, y_{i+1,j+1})$. We now use an isoparametric bilinear mapping to map the quadrilateral given by these points into the unit square in (ξ,η) space. Hence,

$$x = A_1 + A_2\,\xi + A_3\,\eta + A_4\,\xi\eta \tag{15a}$$

$$y = B_1 + B_2\,\xi + B_3\,\eta + B_4\,\xi\eta. \tag{15b}$$

We then assume that w satisfies the same mapping, so

$$w = C_1 + C_2\,\xi + C_3\,\eta + C_4\,\xi\eta. \tag{15c}$$

Knowing x, y, w at the four cell centers corresponding to $\xi,\eta = 0,1$ we can evaluate the coefficients A_i, B_i, C_i. We next calculate the value of ξ and η that gives the values of x and y for node B. Given that value of

ξ and η we calculate w at the node B by (15c). As is standard in finite element theory these mappings preserve the continuity of x, y and w between zones and the interpolation is first-order accurate.

This method is more accurate than the midpoint rule previously discussed but is also more time consuming. Further, the bilinear transformation needs to be modified near boundaries.

In the case of a uniform grid the midpoint rule and the trapezoidal rule give rise to different schemes for advancing w in time. For simplicity of notation, we assume that the fluxes are averaged rather than w. For a uniform grid the midpoint rule reduces to

$$\frac{\partial w_{ij}}{\partial t} + \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta x} + \frac{g_{i,j+1} - g_{i,j-1}}{2\Delta y} = 0. \tag{16}$$

For the uniform grid the trapezoidal rule reduces to

$$\begin{aligned} \frac{\partial w_y}{\partial t} + \frac{1}{8\Delta x}\Big[& f_{i+1,j+1} + 2f_{i+1,j} + f_{i+1,j-1} \\ & - \left(f_{i-1,j+1} + 2f_{i-1,j} + f_{i-1,j-1}\right)\Big] \\ & + \frac{1}{8\Delta y}\Big[g_{i+1,j+1} + 2g_{i,j+1} + g_{i-1,j+1} \\ & - \left(g_{i+1,j-1} + 2g_{i,j-1} + g_{i-1,j-1}\right)\Big] = 0. \end{aligned} \tag{17}$$

We now analyze the stability of both (16) and (17). Linearizing the equation and freezing coefficients, we obtain

$$w_t + Aw_x + Bw_y = 0. \tag{18}$$

Using a Leapfrog or Runge-Kutta scheme the stability depends on the eigenvalues of the matrix

$$D = \alpha A + \beta B \tag{19}$$

where

$$\alpha = \sin\theta,\ \beta = \sin\phi \qquad \text{for (16),}$$

$$\alpha = \sin\theta\left(\frac{1+\cos\phi}{2}\right),\ \beta = \sin\phi\left(\frac{1+\cos\theta}{2}\right) \qquad \text{(for (17).}$$

We consider a general coordinate system (ξ,η) and denote the Cartesian coordinates by (x,y). To simplify the analysis we symmetrize D. We then find (see [5]) that

$$D_0 = TDT^{-1} = \begin{pmatrix} w & ac & bc & 0 \\ ac & w & 0 & 0 \\ bc & 0 & w & 0 \\ 0 & 0 & 0 & w \end{pmatrix}, \tag{20}$$

where c is the speed of sound and

$$a = \alpha y_\eta - \beta y_\xi,\ b = \beta x_\xi - \alpha x_\eta,\ w = \alpha q + \beta r$$

with (21)

$$q = y_\eta\, u - x_\eta\, v \quad \text{and} \quad r = x_\xi\, v - y_\xi\, u.$$

Hence, the eigenvalues of D are given by

$$d = W,\ W \pm \sqrt{a^2 + b^2}\ c\ . \tag{22}$$

It follows that the time step restriction for (16) is of the form

$$\frac{\Delta t}{V} < \frac{K}{|q| + |r| + L_1\ c} \tag{23}$$

where K is a scheme-dependent constant, and

$$L_1 = x_\xi^2 + x_\eta^2 + y_\xi^2 + y_\eta^2 + \sqrt{(x_\xi^2 + y_\xi^2 - x_\eta^2 - y_\eta^2)^2 + f(x_\xi\ x_\eta + y_\xi\ y_\eta)}.$$

For an orthogonal mesh, L reduces to

$$L_1 = 2 \cdot \max(x_\xi^2 + y_\xi^2,\ x_\eta^2 + y_\eta^2).$$

The time restriction for the trapezoidal rule (17) is of the form

$$\frac{\Delta t}{V} < \frac{K}{\frac{|q| + |r| + \max(|q|, |r|)}{2} + L_2\ c}\ . \tag{24}$$

and

$$L_2 = \frac{27}{32}\ L_1.$$

Thus, the time restriction for the trapezoidal rule is slightly less stringent than that for the midpoint rule. The coupling of modes between mesh points also differs between the midpoint and trapezoidal rules.

3. COMPUTATIONAL RESULTS

We first consider the two-dimensional code FL052. This uses a Runge-Kutta algorithm for an O mesh with acceleration by local time step, enthalpy damping, and residual smoothing [2,5] but without any multigrid techniques. All calculations were done on a coarse 64×16 mesh. As the mesh is refined the accuracy of all the methods gets better.

The first case that we consider is flow about a NACA 0012 airfoil with $M_\infty = 0.3$ and $\alpha = 10^o$. The resultant flow is subsonic everywhere and so the solution could be computed using the full potential equation. The lift coefficient is then found to be about 1.27. In Table 1 we present some results for this case. In the first column we indicate which case is being presented. In the second column we present the coefficient of the fourth-order viscosity. In the third and fourth columns we present C_L and C_D respectively. In the last column we give the RMS of the density residual after 1000 time steps. All calculations were done with a four-step Runge-Kutta method with $\alpha_1 = 1/4$, $\alpha_2 = 1/3$, $\alpha_3 = 1/2$, $\alpha_4 = 1$. This gives fourth-order accuracy in time when the coefficients of the differential equation are independent of time. Since the Euler equations are nonlinear there is only second-order time accuracy. In any case, the time accuracy is not of importance since we are considering only steady flows.

Since the flow is subsonic the second-order viscosity does not play an important role. We see from Table 1 that accounting for the non-uniform mesh increases the accuracy of both the lift and drag coefficients. We also see that for this coarse mesh that the value of VIS4 also greatly affects the accuracy. All these calculations were done with a distribution of points in the normal direction as given in the original version of FL052. Other runs

not shown were done with an exponential-like stretching in the normal direction. This provides a smoother mesh and less severe stretching in the far field. As expected the use of (10) does not improve things as much when the mesh is smoother. In fact, for a uniform mesh (6) and (10) coincide. In all cases computed the use of (10) improved the accuracy of the solution.

The next case that we consider is flow about a NACA 0012 airfoil with $M = 0.85$ and $\alpha = 1^{\circ}$. In this case there is a strong shock. Comparisons with other codes indicates that the $C_L \sim 0.36 - 0.40$ and $C_D \sim 0.55$. As before, the lift coefficient is underpredicted by the coarse mesh. Increasing VIS4 increases C_L and also slightly increases C_D. As before we find that the use of the weighting formula (10) increases the accuracy of both C_L and C_D. In this case, we see that the weighted formula also substantially increases the convergence rate in some cases. As before, using a smoother stretching in the normal direction increases the accuracy of all the cases, though using (10) still marginally increases the accuracy. In this case, we again see that the value of VIS4 affects accuracy while other computations show that VIS2 can also have a strong effect on the accuracy of the code.

As the final two-dimensional inviscid calculation we consider flow about an RAE 2822 airfoil with $M_\infty = 0.75$ and $\alpha = 3^{\circ}$. This is also a transonic case which needs good shock resolution. Runs with finer meshes indicate that $C_L \sim 1.10$ and $C_D \sim .042$. In this case the C_L is again underpredicted by 10-20% and the use of the weighting formula (10) again increases the accuracy as does increasing VIS4. It is to be noted that the fine mesh calculations are still done with the Euler equations and we are not considering viscous effects. In this case the use of a smoother exponential-like stretching in the normal direction again dramatically increases the accuracy but at the

expense of a slower convergence rate.

We next tested the weighting formulas in three dimensions using FL057 for a wing-alone configuration. We found that the code still converged for extreme cases as $M = 0.60$ and $\alpha = 24^{o}$. The code was run with a coarse mesh of $64\times10\times10$ and a finer (though still coarse) mesh of $96\times16\times20$. The grid generation used the standard geometry routines contained in FL057. Typical results are shown in Table 4. The use of small value for VIS4, less than 0.5, introduced oscillations in the solution and sometimes the method would not even converge. The use of (10) in each coordinate direction allowed for faster convergence and fewer oscillations in the converged solution. As VIS4 was increased to 1.0 the oscillations became less important. Also, as the mesh was refined the oscillations diminished. For the case $M = 0.60$, $\alpha = 24$ and the $96\times16\times20$ mesh, the standard method would not converge unless we set VIS2 = 1.0. Using the weighting formula (10) the code coverged with VIS2 = 0.5. Recalculating the midpoints of the cells and cell surfaces at each time step adds about 30% to the running time. If the distance ratios in each direction are stored then the increase in running time is negligible but three additional three-dimensional vectors need to be stored.

4. CONCLUSIONS

Using a Taylor-series expansion one can show that the standard weighting of the fluxes (6) at interfaces for a finite volume scheme can be even inconsistent for general meshes. A new weighting formula (10) guarantees that the scheme is at least first-order accurate for all meshes. As expected, computations demonstrate that for smooth meshes the differences between (6)

and (10) are minimal. When the stretchings are more severe then (10) can offer significant advantages, especially on coarser meshes. In addition, in three dimensions we found that the weighting (10) can also improve the convergence properties of the scheme. We also found that the constants multiplying the artificial viscosity can have a large effect on the accuracy of the solution, especially on the lift coefficient. Weighting formulas for nodal schemes and MacCormack schemes are given in [6].

REFERENCES

[1] Davis, R. L., Ni, R. H., and Carter, J. E., "Cascade Viscous Flow Analysis Using the Navier-Stokes Equations," AIAA Paper 86-0033, 1986.

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

[3] Jameson, A. and Baker, T. J., "Solution of the Euler Equations for Complex Configurations," AIAA 6th Comput. Fluid Dynamics Conference, Paper 83-1919, 1983.

[4] Pulliam, T. H., "Artificial Dissipation Models for the Euler Equations," AIAA Paper 85-0438, 1985.

[5] Turkel, E., "Acceleration to a Steady State for the Euler Equations," Numerical Equations for the Euler Equations," Numerical Methods for the Euler Equations of Fluid Dynamics, SIAM, 1985, pp. 281-311.

[6] Turkel, E., "Accuracy of Schemes with Non-uniform Meshes for Compressible Fluid Flows," to appear J. Numerical Mathematics.

[7] Woan, C. J. and Bonner, E., "An Investigation of Gridding on the Accuracy of FL057MM and FL057MG Euler Wing Solutions, AIAA Paper 85-4090, 1985.

TABLE 1: Flow about NACA 0012 with $M_\infty = 0.3$, $\alpha = 10^o$.
O mesh with 64 × 16 grid.

Weighting	VIS4	C_L	C_D	Residual
(6)	0.2	1.227	.008	9×10^{-9}
	0.5	1.240	.007	6×10^{-9}
(10)	0.2	1.264	.002	2×10^{-8}
	0.5	1.288	-.001	1×10^{-8}
finer mesh		~1.274	0	

TABLE 2: Flow about NACA 0012 with $M_\infty = 0.85$, $\alpha = 1^o$
O mesh with 64 × 16 grid.

Weighting	VIS4	C_L	C_D	Residual
(6)	0.2	.2631	.0487	1×10^{-9}
	0.5	.2738	.0499	1×10^{-5}
(10)	0.2	.2772	.0492	1×10^{-10}
	0.5	.3006	.0513	2×10^{-8}
finer mesh		~.36-.40	~.054	

TABLE 3: Flow about RAE 2822 with $M_\infty = 0.75$, $\alpha = 3^\circ$.

Weighting	VIS4	C_L	C_D	Residual
(6)	0.2	.9586	.0395	1×10^{-11}
	0.5	.9910	.0407	5×10^{-12}
(10)	0.2	1.003	.0414	3×10^{-11}
	0.5	1.059	.0431	5×10^{-12}
finer mesh		~1.10	~.042	

TABLE 4: Three-Dimensional Case with $M_\infty = 0.60$, $\alpha = 24^\circ$.

Weighting	VIS4	C_L	C_D	No. Iterations	Residual
(6)	0.25	1.0248	.4170	300	8×10^{-5}
	1.00	1.1359	.4611	150	7×10^{-6}
(10)	0.25	1.0155	.4132	300	4×10^{-5}
	1.00	1.1265	.4571	150	7×10^{-6}

1. Report No. NASA CR-178038 ICASE Report No. 85-59	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle	5. Report Date
ACCURACY OF SCHEMES FOR THE EULER EQUATIONS WITH NON-UNIFORM MESHES	December 1985
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
E. Turkel, S. Yaniv and U. Landau	85-59
	10. Work Unit No.
9. Performing Organization Name and Address	11. Contract or Grant No.
Institute for Computer Applications in Science and Engineering Mail Stop 132C, NASA Langley Research Center Hampton, VA 23665-5225	NAS1-17070; NAS1-18107
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contractor Report
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code
	505-31-83-01

15. Supplementary Notes

Langley Technical Monitor:
C. South Jr.
Final Report

Submitted to SIAM J. Sci. Stat. J. Comput.

16. Abstract

The effect of non-uniform grids on the solution of the Euler equations is analyzed. We consider a Runge-Kutta type scheme based on a finite volume formulation. We show that for arbitrary grids the scheme can be inconsistent even though it is second-order accurate for uniform grids. An improvement is suggested which leads to at least first-order accuracy for general grids. Test cases are presented in both two- and three-space dimensions. Applications to finite difference and implicit algorithms are also given.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
accuracy Euler equations Runge-Kutta scheme non-uniform grids	64 - Numerical Analysis 02 - Aerodynamics Unclassified - Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	21	A02

For sale by the National Technical Information Service, Springfield, Virginia 22161