AIAA 82-4012

Multigrid Code for Three-Dimensional Transonic Potential Flow about Inlets

D. R. McCarthy*
Indiana University—Purdue University, Fort Wayne, Ind.
and
T. A. Reyhner†
Boeing Commercial Airplane Company, Seattle, Wash.

In recent years, new mathematical techniques, known as multigrid methods, have been increasingly advocated for the acceleration of relaxation calculations. When successful, they have been extremely effective. However, it has been questioned whether, in view of their complexity, these methods could ever be fruitfully applied to practical problems with complex geometry in three dimensions. An existing code for axisymmetric inlets at angles of attack has been modified to utilize a four-level multigrid method. The mesh follows cylindrical coordinates in the physical domain and is not fitted to the body. Order-of-magnitude improvements in speed and accuracy over the unmodified code are reported. The procedure maintains an overall convergence rate of 0.9. Because this rate is independent of the mesh spacing, an important conclusion is that the method makes possible the use of meshes too fine to have been considered previously. It is also observed that the coding of the algorithm can, as expected, require considerable effort. Further, the adaptation of existing codes is found to be difficult if the structure of the original is not amenable to multigrid procedures. Nevertheless, the method is shown to be of enormous value, when carefully applied, in spite of nonlinearities and complicated geometry. We conclude that its application in production codes will ultimately be deemed practical.

Nomenclature

a	= speed of sound
F	= right-hand side of differential equations
G	= grid
h	= mesh step size
I	= interpolation/injection operator
K	= number of mesh levels
L	= differential operator
q_∞	= freestream velocity
r	= radial coordinate
R	= convergence rate
W	= work
z	= axial coordinate
Δz	= mesh spacing in z
γ	= ratio of specific heats
θ	= circumferential angle
$\Delta\theta$	= mesh spacing in θ
τ	= truncation error
ϕ	= potential function
$\Delta\phi$	= change in ϕ between successive relaxation sweeps
Φ	= exact solution for ϕ

Subscripts

k	= mesh level number
z,r,θ	= partial derivatives
∞	= freestream

Superscripts

k	= mesh level number
K	= number of mesh levels

Received Aug. 12, 1980; revision received July 27, 1981.

*Associate Professor, Department of Mathematical Sciences. Member AIAA.

†Senior Specialist Engineer, Propulsion Research Unit. Associate Fellow AIAA.

I. Introduction

LARGE-scale fluid dynamics computations have always been hindered by the slow asymptotic convergence rate of standard relaxation schemes. The situation is further aggravated by the mesh refinements necessary to resolve the flowfield accurately. Indeed, the computing time required for many realistic production codes is such as to seriously limit their practical value, especially for design applications which are by nature iterative. Consequently, there has been much attention given to methods for increasing the efficiency of the relaxation process. While gains are sometimes made, success is seldom dramatic and often relies on highly problem-dependent assumptions.

Recently, however, new mathematical techniques known as multigrid methods have been proposed and advocated, especially by Brandt.[1,2] Among other advantages, they are said to offer from one to several orders-of-magnitude improvement in execution time and to provide greatly increased accuracy as well. Indeed, Brandt has demonstrated remarkable success with two-dimensional elliptic and other more or less academic problems. The question has been raised, however, as to whether the method would be truly effective when applied to the large programs encountered with actual flows, especially the mixed hyperbolic-elliptic problems associated with the transonic regime. Initial success along these lines was demonstrated by South and Brandt,[3] who applied the multigrid method to the small-disturbance equation in two dimensions. The purpose of the current work is to demonstrate its effectiveness in a realistic production environment, namely, a three-dimensional full potential formulation involving complex geometry.

In order to minimize the programming effort, it was decided early to modify an existing code rather than to create a new one. In retrospect, it might be argued that this choice placed severe limitations on the future development of the code, and even on those aspects of multigrid which could be tested. Nevertheless, as the objective was to demonstrate feasibility, the choice was for that purpose appropriate.

The code chosen was the Reyhner code for three-dimensional transonic potential flow around axisymmetric

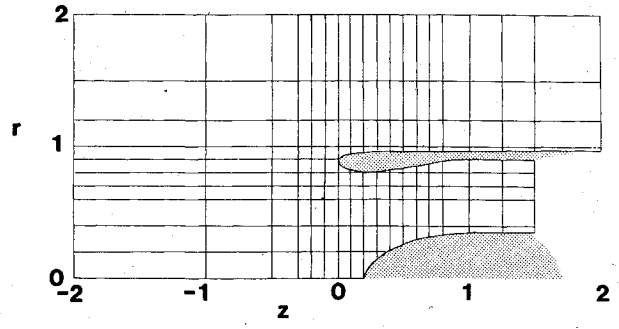

Fig. 1 Cross section of the 1.26 contraction ratio inlet (NASA TM X-2937) showing part of the z-r grid for $k=1$.

inlets at angle of attack.[4] It is a finite-difference nonconservative SLOR scheme which operates in cylindrical coordinates in the physical domain. These coordinates are not body fitted; the coordinate lines are permitted to intersect the surface as shown in Fig. 1. The principal advantage to such an approach is its ease of generalization to asymmetric three-dimensional cases; the main disadvantage is that it necessitates the construction of a wide variety of procedures for handling the numerous types of mesh nodes at and near intersections of coordinate lines with the surface.

This approach circumvented the need to write a new relaxation scheme for a complex situation; it posed the difficulty that the original code was not designed to easily accommodate multigrid. This code had, however, been designed to work with two grids, solving first on a coarse one and then interpolating to a fine one and solving again. Thus it already possessed some refinement capability; multigrid, however, requires fine-to-coarse as well as coarse-to-fine transfers. Additionally, the absence of transformations promised the advantage of more transparent interpretation of the effects of grid manipulation, as well as simpler extension to the asymmetric case.

II. Algorithmic Structure

For completeness, we present here a brief outline of the multigrid algorithm as particularly implemented in this work. For a detailed discussion of the general method, the reader is referred to Ref. 2 which contains a full exposition.

Our objective is to solve the potential flow equation in cylindrical coordinates r, θ, and z:

$$L(\phi)=0 \tag{1}$$

where

$$L(\phi)=(a^2-\phi_r^2)\phi_{rr}+\left(a^2-\frac{\phi_\theta^2}{r^2}\right)\frac{\phi_{\theta\theta}}{r^2}+(a^2-\phi_z^2)\phi_{zz}$$
$$-2\frac{\phi_r\phi_\theta}{r^2}\phi_{r\theta}-2\phi_r\phi_z\phi_{rz}-2\frac{\phi_\theta\phi_z}{r^2}\phi_{\theta z}+\left(a^2+\frac{\phi_\theta^2}{r^2}\right)\frac{\phi_r}{r} \tag{2}$$

and a, the local speed of sound, is obtained from

$$a^2=a_\infty^2-\frac{\gamma-1}{2}\left(\phi_r^2+\frac{\phi_\theta^2}{r^2}+\phi_z^2-q_\infty^2\right) \tag{3}$$

The finite-difference multigrid method replaces Eq. (1) with a collection of discretization equations

$$L^k(\phi^k)=F^k \tag{4}$$

where $k=1,\ldots,K$. Here L^k represents a discretized version of the operator L, and ϕ^k and F^k represent scalar fields on a grid G^k which is one of a hierarchy of grids of varying coarseness, $k=1,\ldots,K$. In general, G^{k+1} is obtained from G^k by inserting one new mesh line midway between each pair of G^k mesh lines. On the finest level, the field F^k is identically zero, i.e., represents the discretization of the right-hand side of Eq. (1); on coarser levels, F^k represents an adjusted right-hand side calculated by the full approximation storage (FAS) procedure.[2] This procedure is required to accommodate the nonlinearity of Eq. (1).

Table 1 Mesh chart

Level	z	r	θ	Nodes in one z plane	Total number of nodes
1	21	11	5	55	1,155
2	41	21	5	105	4,305
3	81	41	5	205	16,605
4	161	81	5	405	65,205
5	321	161	5	805	258,405

Our system operates on a hierarchy of four meshes ($K=4$). Table 1 gives the number of z, r, and θ mesh lines used for each level. For reasons to be explained (Sec. III), results reported in this paper maintain 5 θ mesh at all levels. Also, level 5 was never attempted because of storage limitations. Because of regularities in θ, special difference quotients can be used in the θ direction to keep the number of θ mesh required small.[4] These are distributed only from 0 to 180 deg, as the plane with $\theta=0$ and 180 deg is a plane of symmetry. Levels 3 and 4 utilize extensive disk storage. The mesh spacing in θ is uniform. G^1 is not uniform in r and z. Refinements are formed by bisection of intervals in r and z.

It is easiest to think of the solution procedure as beginning on G^4, the finest level. We set F^4 identically zero. Beginning with an initial estimate ϕ^4, we relax Eq. (4), using radial-line relaxation. The objective of multigrid is to utilize only the first few relaxation sweeps on any level, before the convergence rate deteriorates to its asymptotic value. When this "stalling" begins, computations are transferred to the next coarser level on the assumption that the wavelengths of the major error components are more visible to the coarser grid. Because of the reduced number of nodes, relaxation there is both less expensive and more effective. When convergence is attained at any level, calculations are returned to the next finer level, with the process terminating when convergence is achieved on level 4.

Normally, one would like to monitor convergence on any grid G^k by watching the residuals $[F^k-L^k(\phi^k)]$. Here this is difficult because of variations in the scaling of the various formulas used at different types of mesh nodes and because, owing to the structure of the original code, their calculation would require an extra sweep of the field after each relaxation sweep, effectively doubling the computational work. Therefore, we monitor $\Delta\phi^k$, the change in the solution from sweep to sweep, which is readily available. We monitor both the average and maximum values (over the flowfield) of $\Delta\phi^k$. The magnitude of $\Delta\phi^k$ provides an easily computed, if not ideal, measure of the degree of convergence. In practice, it suffices as a control parameter within the code; in theory, it is far less desirable than the residual for the sake of comparisons. The ratio of successive $\Delta\phi^k$, on the other hand, provides an acceptable estimate of the instantaneous convergence rate.

During the first few sweeps on any level, convergence rates remain well below 1; typically, they are between 0.5 and 0.7. The two rates quickly deteriorate; when they both exceed preset limits, computations are moved to the next coarser level. The limits used here were 0.90 for the rate based on the average $\Delta\phi$ and 0.95 for that based on the maximum $\Delta\phi$.

Transfer of the G^k problem to G^{k-1} cannot be accomplished by naively applying the injection operator I_k^{k-1}, which "selects" every other value from any field on G^k, to all quantities involved. For, just as the exact continuous solution

Φ will not satisfy the G^k equations, neither will the injected exact G^k solution $I_k^{k-1}\Phi^k$ satisfy the "selected" G^{k-1} equations. This is because L^{k-1} is only an approximation to L^k. The injected G^k solution would satisfy instead

$$L^{k-1}(I_k^{k-1}\Phi^k) = I_k^{k-1}F^k + \tau_k^{k-1} \tag{5}$$

where

$$\tau_k^{k-1} = L^{k-1}(I_k^{k-1}\Phi^k) - I_k^{k-1}L^k(\Phi^k) \tag{6}$$

since the last term on the right is $I_k^{k-1}F^k$. Here τ_k^{k-1} is called the truncation error at G^{k-1} relative to G^k. Its insertion enables the behavior of Φ^k to be simulated on G^{k-1}. Thus, following Eq. (5), when we reach G^{k-1}, we relax instead the equation

$$L^{k-1}(\phi^{k-1}) = F^{k-1} \tag{7}$$

where, by definition,

$$F^{k-1} = I_k^{k-1}F^k + \tau_k^{k-1} \tag{8}$$

Here we use $I_k^{k-1}\phi^k$ as an initial estimate, where ϕ^k denotes the last iterate from G^k. There is one further detail: τ_k^{k-1} is not known because the exact solution Φ^k is not known. Thus τ_k^{k-1} must be estimated from ϕ^k via Eq. (6). Thus each right-hand side F^{k-1}, $k=2,3,4$, is defined recursively from F^k upon grid coarsening. Because τ_k^{k-1} must be estimated from ϕ^k, F^{k-1} may be different upon different visits to G^{k-1}, $k=2,3,4$. One may also regard the equation for F^{k-1} as guaranteeing that the exact solution Φ^k will not be changed by a multigrid cycle. As an aside, this requirement can be simulated in a computer code by setting $L(\phi^k)$ to zero and forcing the code to cycle. Such procedures are very useful for uncovering coding errors.

When a converged solution ϕ^{k-1} has been obtained on G^{k-1}, it must be interpolated by some process I_{k-1}^k to fill in missing values on the finer grid G^k. (A quadratic interpolator was used here, principally because it was provided in the original program.) Now $I_{k-1}^k\phi^{k-1}$ would be a poor solution on G^k, precisely because it would lack the high-frequency components which were lost from ϕ^k (the last iterate on G^k) when the grid was coarsened. If, however, ϕ^k has been saved (since the last visit to G^k), then these are easily recovered. For if ϕ^k is injected onto G^{k-1} and then smoothly interpolated back to G^k, the result is a "smoothed" version of ϕ^k. Subtracting this from ϕ^k leaves only the high frequencies. Thus we achieve a corrected guess ϕ_{new}^k on G^k by

$$\phi_{\text{new}}^k = I_{k-1}^k\phi^{k-1} + (\phi^k - I_{k-1}^k I_k^{k-1}\phi^k) \tag{9}$$

Then relaxation continues on G^k. (Note that $I_{k-1}^k I_k^{k-1}$ is not the identity.) Again, this procedure guarantees that an exact solution Φ^k is a stationary point of the cycling process.

The entire process terminates when convergence has been achieved on level 4. We require the average $\Delta\phi^4$ to be less than 10^{-7} at convergence. For $k<4$, criteria are set internal to the program to require about an order-of-magnitude improvement before convergence is declared.

In order to stabilize the field before changing grids, at least five sweeps were required during each visit to any level, regardless of other stalling and convergence criteria. On level 1, where no coarser grid is available, relaxation is continued until $\Delta\phi$ is less than 10^{-8} regardless of the rate. Sweeping on level 1, however, is relatively inexpensive because of the small number of mesh nodes. Further, the asymptotic relaxation convergence rate is typically $1-O(h^2)$, where h is the mesh step size. Thus this rate is fastest on level 1. In addition, over-relaxation and extrapolation are used (on level 1 only) to accelerate convergence.

In practice, the logical structure of the code augments that described here as follows. The initial field on G^4 is obtained by interpolation from a converged solution on G^3, which has itself been obtained from a multigrid process in which G^3 plays the role of the finest grid. The initial field for this process is obtained similarly from G^2, and so on. Thus each grid in turn functions temporarily as the finest grid. Many structures are possible; this one has the advantage of yielding useful intermediary results. The initial field on G^1 is uniform flow at freestream conditions.

III. Refinements in θ

Before discussing the general results, we address a problem which arose concerning refinement of the mesh in the θ direction. In level changes which involve such refinements, it is common to observe the following phenomenon. At first the behavior of the multigrid cycle appears normal; however, as the error decreases an uncharacteristic cycle appears. For example, level 3 with, say, 5 θ mesh, experiences relaxation stalling, but when computation is transferred to level 2, with only 3 θ mesh, level 2 finds itself immediately converged. As a result, computation is returned to level 3 with little change and the cycle then repeats. The difficulty was ultimately found to lie not in the mesh-switching procedure, but in the relaxation scheme itself. An examination of the residuals on level 3 reveals a large unliquidated error component with a wavelength in the θ direction which is definitely comparable to the mesh size. This indicates that the rate at which these components are smoothed is quite slow, well above even the conservative 0.9 at which stalling is declared. As a local smoothing rate analysis[2] will show, this is largely attributable to the fact that the aspect ratio $(r\Delta\theta)/\Delta z$ is large, especially for large r. Aside from extreme coarsening of the z mesh or extreme refinement of the θ mesh, repair is impossible without extensive modification of the relaxation scheme, which is precisely what we sought to avoid.

Thus it is important to note that in any multigrid scheme, the relaxation procedure must be a good *smoother*. Relaxation procedures which are effective in *reducing* the error may not be effective in smoothing it. Large aspect ratios contribute substantially to this problem; thus it may be intrinsic to the use of cylindrical (or any) coordinate systems in which aspect ratios vary widely throughout the field. Other experience with the aspect ratio problem and a related local smoothing rate analysis are reported in Ref. 3. It has been suggested, there and elsewhere, that alternating direction methods are the proper fix. This would require major coding changes and has not been attempted here.

In order to circumvent the difficulty, we present here only results in which the number of θ mesh is held fixed at 5. This limitation is not exceedingly restrictive, however, because the flow is normally quite regular in θ and thus increasing the number of θ mesh produces little change in (converged) results. Also, the number of θ mesh is small enough not to be a problem with convergence rate and computer time requirements. All work units in the tables have been adjusted to reflect the constant θ situation; thus, multigrid with refinement in θ (and a smoothing scheme at least as good in the θ direction as this one is in z and r) should exhibit convergence rates even better than those given here.

Table 2 Run comparisons

Level		Unmodified, standard mesh	Unmodified, dense mesh	Multigrid, dense mesh
3	Sweeps[a]	150	500	28
	Work[b]	32	75	15
	$\Delta\phi$	5.4 E-6	7.7 E-6	4.9 E-7
4	Sweeps[a]	—	400	22
	Work[b]	—	475	44
	$\Delta\phi$	—	2.4 E-5	9.8 E-8

[a] Does not count work at lower levels. [b] Counts work at lower levels.

IV. Results

It is natural to ask for a comparison of the speeds of the modified and unmodified programs. This, however, requires some care in order to avoid misleading conclusions and to comprehend the full strength of the method.

In Table 2, we report results for the test case of the NASA TM X-2937 1.26 contraction ratio inlet at a 30 deg angle of attack. Figure 1 shows a $\theta = \text{const}$ cut through the inlet, together with a portion of the coarsest grid. (The grid extends out of the figure to the left.) This mesh was used to test the multigrid analysis. For accurate prediction of the inlet flowfield it is desirable to use a mesh extending further out radially. The freestream is at 45 m/s and the average Mach number at the throat is 0.64. This case is largely subsonic with a supersonic bubble inside the inlet. It is a case which is particularly slow to converge using the unmodified code. Results for this case in comparison with experiment are shown in Fig. 7 of Ref. 4.

Normally, the Reyhner code is run with two meshes whose density corresponds roughly to levels 2 and 3 of the multigrid program. The level 3 results of such a run are given in column 1 of Table 2; results for the multigrid run are given in column 3. Comparison is hindered by the lack of results for the unmodified code at mesh densities corresponding to level 4. Column 2 therefore reports the results of a run of the unmodified code in which the two levels are made to correspond to levels 3 and 4. The runs in columns 1 and 3 were made on a Cyber 175; actual CPU times were 138 and 203 s, respectively. The run in column 2 was, unfortunately, made on a CDC 6600. For that reason, the CPU times are not given in the table. One may estimate comparative timings by noting that the time to execute a single relaxation sweep of level 3 is just under 1 s on the Cyber 175; for level 4 it is just under 4 s; thus the run in column 2, if executed on the Cyber 175, should require about 1.5 h of CPU time. (This was regarded as prohibitively expensive to rerun.) These times do not include disk reads and writes.

The modified code possesses multigrid ability in all three coordinates; however, because of the aspect ratio problems cited earlier, the results reported in the table are for 5 θ mesh at all levels, except for column 2 which has only 3 θ mesh at level 3. For the purpose of estimating computational work, it is useful and customary to take as one work unit the cost of a single relaxation sweep of the finest grid, level 4. The work units reported have all been adjusted to this standard by assuming that the relaxation work performed on lower levels is proportional to the number of nodes processed. Thus a sweep of level 3 costs approximately one-quarter of a work unit (or 3/20 in column 2, since level 3 there has only 3 θ mesh).

While relaxation constitutes virtually all of the work in the unmodified program, there is a somewhat larger overhead in multigrid, owing to the mesh-switching procedures, which is not reflected in these work units. The amount of this overhead may be estimated in various ways. For example, it is known that a sweep of level 4 requires about 3.9 s; the 44 work units of column 3 therefore require about 171 of the 203 s CPU. This suggests overhead of around 20%, which is consistent with the observations of Brandt.[2] No attempt has been made here to control overhead or storage requirements. In fact, disk files were used freely, and execution wall clock time is lengthened considerably by the relatively slow disk reads and writes.

V. Fixed-Grid Comparison

A most natural comparison to make is to pose the problem of solving on level 4, say, and then to examine the relative efficiency of the modified and unmodified programs. For this purpose, we compare columns 2 and 3 of Table 2 at level 4. It can be seen immediately that the multigrid code requires less than one-tenth the total relaxation work. More importantly, however, the level of convergence (as measured by $\Delta\phi$) appears better for multigrid by approximately two orders of magnitude. No more specific comparison should be made. For one thing, the standard code employs an over-relaxation parameter of 1.85, while multigrid uses none. Thus the $\Delta\phi$ of column 2 is artificially inflated by at least this factor and (owing to immediate replacement of values) possibly more. This could of course be avoided by the use of residuals as a convergence measure (see Sec. VIII).

In any case, such comparisons are misleading. To see the difficulty note that, because of the exponential relationship between work and $\Delta\phi$, it is tempting but meaningless to multiply the two factors together and declare that multigrid is three orders of magnitude superior. It would be at least preferable to require both programs to achieve the same $\Delta\phi$ and compare the work required, or to allow both programs to do the same amount of work and compare the resulting $\Delta\phi$. Such an analysis may be based on average convergence rates. The convergence rate R may be defined as the ratio of successive values of $\Delta\phi$ which are separated by one unit of work. If work W is required to reduce $\Delta\phi$ by one order of magnitude, then

$$R^W = 1/10 \tag{10}$$

or

$$W(R) = -1/\log R \tag{11}$$

For the results cited here, R is about 0.9 for multigrid and about 0.995 (the asymptotic rate for level 4 relaxation) for the unmodified program. The numbers are

$$W(0.9) = 22 \tag{12}$$

and

$$W(0.995) = 459 \tag{13}$$

Even assuming that the 0.995 rate were maintained, an additional 900 sweeps would be required to match the convergence level of the multigrid procedure by sweeping level 4 alone, bringing the total to about 1.5 h of CPU time, as compared to about 3.5 min for multigrid. To make an equal work comparison, on the other hand, if 475 work units were expended on multigrid, the $\Delta\phi$ would be reduced by (theoretically) 20 orders of magnitude, a clearly absurd undertaking. The point is that such comparisons favor multigrid exponentially as the amount of work is increased and are thus highly problem dependent. Therefore, quantitative comparisons of the type made above must be recognized as unique to the situation at hand. It is possible to infer, however, that the relative advantage which multigrid enjoys becomes more evident as the desired accuracy is increased.

VI. Grid-to-Grid Comparison

The foregoing analysis will hold for any procedure which improves convergence rates on a fixed grid, such as a modification of the relaxation procedure itself. It does not address the relative behavior of the modified and unmodified programs as the grids are refined.

It is well known that asymptotic convergence rates for many standard relaxation schemes are of the form $1 - O(h^2)$ (or, at best, $1 - O(h)$ for optimal over-relaxation), where h is the mesh size. That is, halving of h results in a factor of 4 (or at best 2) deterioration in the convergence rate. Thus, for a fixed desired accuracy, the work per grid point increases exponentially with grid refinement. It is for this reason that level 4 is essentially impractical for the unmodified code: the level of convergence achieved by it was insufficient for engineering purposes. (Note that grid refinement affects the $\Delta\phi$ required for convergence in two ways: first, it is sometimes argued that better convergence must be achieved on finer grids because, during subsequent velocity calculations, division by smaller step sizes will amplify errors; but more

Table 3 Convergence history

K	k	Number of sweeps	$\Delta\phi$ upon entering	$\Delta\phi$ upon leaving	Conv. declared at	Work units (cum.)	Current conv. rate	Cycle conv. rate
1	1	92	9.08E-03	3.28E-09	1.00E-08	1.63	0.850	
2	2	9	2.27E-04	5.43E-05	2.50E-05	2.22	0.836	
	1	81	8.34E-04	2.20E-09	1.00E-08	3.66	0.852	
	2	5	3.19E-05	5.66E-06	2.50E-05	3.99	0.649	0.919
3	3	13	3.02E-05	5.44E-06	5.00E-07	7.30	0.867	
	2	9	1.98E-05	1.74E-05	2.72E-06	7.89	0.984	
	1	78	2.89E-04	1.38E-09	1.00E-08	9.28	0.853	
	2	6	1.44E-05	1.77E-06	2.72E-06	9.67	0.658	0.919
	3	10	2.21E-06	6.26E-07	5.00E-07	12.22	0.869	0.894
	2	6	1.78E-06	1.24E-06	3.13E-07	12.61	0.930	
	1	59	1.79E-05	9.38E-09	1.00E-08	13.66	0.878	
	2	5	1.15E-06	2.66E-07	3.13E-07	13.99	0.693	0.929
	3	5	3.54E-07	2.34E-07	5.00E-07	15.26	0.901	0.921
4	4	9	4.22E-06	8.08E-07	1.00E-07	24.26	0.813	
	3	10	2.67E-06	1.67E-06	4.04E-07	26.80	0.949	
	2	9	6.08E-06	4.58E-06	4.18E-07	27.40	0.965	
	1	66	5.95E-05	2.11E-09	1.00E-08	28.56	0.854	
	2	17	5.16E-06	4.10E-07	4.18E-07	29.70	0.854	0.933
	3	5	9.43E-07	3.80E-07	4.04E-07	30.96	0.797	0.913
	4	13	4.31E-07	9.78E-08	1.00E-07	43.96	0.884	0.898

Table 4 Summary of Table 3

Level	Total sweeps	Total visits	Total work	Percentage of work
1	376	5	6.66	15.15
2	66	8	4.36	9.92
3	43	5	10.94	24.88
4	22	2	22.00	50.05

importantly $\Delta\phi$ is a mesh-dependent measure which, for the same error in ϕ, is proportional to h^2 and must be held smaller on finer grids.)

In contrast, the overall convergence rate achieved by multigrid is approximately the smoothing rate, i.e., that observed during the first few sweeps. This rate depends only on the relaxation scheme and not on the mesh size. Thus mesh refinement increases work only as rapidly as it increases the number of points; the work *per point* remains fixed. It is this feature of multigrid which makes level 4 practical. Thus the most valuable feature of the method may not be the speed with which it calculates already available results, but that it renders accessible results which were previously out of reach.

VII. Convergence History

For reference, we show in Table 3 the convergence history for the multigrid run in column 3 of Table 2. K denotes the current finest level and k denotes the level being swept. Table 4 is a summary of Table 3.

VIII. Error and Convergence Rate Measures

Results for theoretical studies of model problems are often reported in terms of actual error, since for many such studies problems are chosen in which exact mathematical results are known. In the current case, such results are unavailable, so that error must be judged indirectly. A widely used error measure in such cases is the residual $F^k - L^k(\phi^k)$, the amount by which the difference equations fail to be satisfied. A strong argument, based on the premise that the algebraic systems resulting from the discretization of partial differential equations are largely well conditioned, can be mounted in favor of the contention that this residual is proportional to the actual error. For this reason and because the residual is not mesh dependent, its use as a measure of error is highly recommended.

In the present situation, the field of residuals (or, more precisely, some norm thereof) is not readily available for two reasons. One of these is that the original code did not require residuals; thus separate routines are required to construct them, as they are needed during grid coarsening. However, this calculation requires work equivalent to a relaxation sweep. Hence it is done only when necessary, that is, when the need to switch grids has already been detected by some other means. The additional work would be justified in a theoretical study, of course; however, there is a second difficulty, less easily handled. It results from variations in the scaling of the equations at the various mesh nodes. In the original code, there was no need to keep such scaling uniform and the algebraic formulation was therefore manipulated freely. However, the size of the residual clearly depends on the form in which the equations are written; that is, if an equation should be, say, multiplied by a scalar α, the solutions would be unaffected but the residual would be multiplied by α. Because the equations at different types of nodes were written differently, one cannot meaningfully compare residuals at different spatial locations. This prohibits not only the calculation of residual norms, but also the use of weighted injections during grid coarsening in which the right-hand side of Eq. (7) at a coarse grid point is based on a weighted average of residuals at all nearby fine grid points. Such injection is recommended for nonlinear problems.[2]

IX. Conclusions and Observations

It is easy to conclude that the method is of enormous power and value. The work cited here is evidence in favor of the claim that neither nonlinearity nor irregularly shaped domains pose any serious impediment to its application. In addition, the basic philosophy of the method admits numerous extensions and refinements, providing the framework, for example, for embedded mesh refinements which cover only a portion of the field. There are, however, some cautions to be mentioned and a few observations to make.

Agreement with Experiment

It is not claimed that the results obtained here will necessarily be in any better agreement with experiment than those available previously. Approximations and in-

consistencies in the mathematical model, e.g., the neglect of viscous effects, may well be of more importance. However, it is claimed that once the mathematical problem has been posed, multigrid methods enable its solution with greater accuracy in two senses: first, by making the use of finer grids possible, discretization error is reduced; and second, by accelerating convergence, multigrid permits more accurate solution of the discretized problem.

Programming Effort

The most significant difficulty by far is that of coding the algorithm. Considerable thought should be given to the data structure in order to minimize the problems associated with grid manipulation. It is instructive to gain experience by experimentation with a simple problem before undertaking a large project. While one may profit from a study of proposed universal data structures,[5] the routines which already exist to implement them (GRIDPACK) are incomplete and for a variety of reasons probably inapplicable in problems of the type considered here.

Modification of Existing Programs

The success demonstrated here is obvious evidence that modification of existing programs to incorporate multigrid is possible. However, one must then live within the data structure originally provided (or an augmentation of it). This structure may not be convenient for multigrid. A more natural operation is to insert an existing relaxation scheme into a general multigrid program. The relaxation scheme is of concern as well, since it must effectively smooth the error, which it may not have been designed to do. Some analysis of the smoothing rate is desirable, but direct calculation of this rate is impractical in complicated situations. A package program (SMORATE) is available for these calculations.[5] SMORATE is two-dimensional and requires local estimation of coefficients, but can nevertheless be of some use.

Convergence Measures

For the various reasons outlined in Sec. VIII, residuals should be used to monitor convergence and convergence rates instead of the field changes used here. We suggest that two procedures be followed:

1) The residuals should be obtained "dynamically," i.e., while the field is being swept. This requires writing the relaxation scheme in a form such that the residuals are explicitly calculated as a natural part of the relaxation process.

2) Some absolute scale for the residuals should be established. If a fluid flow equation(s) is properly scaled, then it should be possible to force the residual to represent a rate of production of mass, momentum, energy, or some combination of these quantities per unit volume per unit time. All equations, including boundary conditions, should be written in this form so that residuals at different points may be compared, and to facilitate weighted injection.

Detecting Success

It is important to have some idea of what constitutes a successful run. In our experience there is significant danger that an improperly working program, containing fairly gross coding errors, can run to convergence. Because such convergence is obtained following a sequence of normal relaxation sweeps, that is, as a result of calculations which would have been done in the absence of multigrid, the solution obtained will be the expected one. Thus such errors appear only in the form of slow, or at best moderately improved, convergence rates, giving the impression that the multigrid method functions less well than might have been expected. It is therefore necessary to have some a priori estimate of what convergence rate to expect. Of course, one would hope to come near the smoothing rate of the relaxations as an overall convergence rate, and usually the stalling rate is set somewhere near this number. (Declaring stalling at too large a rate simply causes unnecessary work; setting it too low can cause oscillation between grids in which no progress results.) The overall effect of each multigrid cycle is monitored via the cycle convergence rate, which may be employed at any level. This rate R is defined as the rate which, if maintained on the current level only, would, in the same amount of work, reduce the residual (or in our case $\Delta\phi$) by the same factor. Let ΔW denote the number of multigrid work units between two successive instances of stalling (or between stalling and convergence) on level k. Let W_k denote the work of sweeping G^k once, so that $\Delta W/W_k$ is the number of sweeps of level k alone which could have been obtained for the same computational work. Then

$$R^{\Delta W/W_k} = \left[\frac{\Delta\phi_{\text{new}}}{\Delta\phi_{\text{old}}}\right] \tag{14}$$

or,

$$R = \left[\frac{\Delta\phi_{\text{new}}}{\Delta\phi_{\text{old}}}\right]^{W_k/\Delta W} \tag{15}$$

We show this rate in the last column of Table 3; note the favorable comparison with the stall criterion of 0.9.

Note that one should not, immediately upon return to any higher level, expect to see an improvement in the degree of convergence, but rather an improvement in the convergence rate. This is why the rate calculated above should be monitored over a full cycle.

Experience

We conclude by noting that success depends on experience with the method; we doubt that many first attempts at applying the method in a complicated situation will meet with immediate success. Ours certainly did not. However, we believe that once some experience has been gained, the method will be found to be most promising and worthwhile.

Acknowledgments

This work was supported in part by Grant NSG-1635 from NASA Langley Research Center. We wish also to acknowledge the support of the Boeing Commercial Airplane Company. The work of Gary E. Shurtleff of Boeing Computer Services, Inc., in programming the analysis is gratefully acknowledged.

References

[1] Brandt, A., "Multilevel Adaptive Computations in Fluid Dynamics," *AIAA Journal*, Vol. 18, Oct. 1980, pp. 1165-1172.

[2] Brandt, A., "Multi-Level Adaptive Solutions to Boundary Value Problems," *Mathematics of Computation*, Vol. 31, 1977, pp. 333-390.

[3] South, J. C. Jr. and Brandt, A., "Application of a Multi-Level Grid Method to Transonic Flow Calculations," ICASE Rept. 76-8, NASA, 1976.

[4] Reyhner, T. A., "Transonic Potential Flow Around Axisymmetric Inlets and Bodies of Angle of Attack," *AIAA Journal*, Vol. 15, Sept. 1977, pp. 1299-1306.

[5] Brandt, A., "Multi-Level Adaptive Techniques (MLAT) for Partial Differential Equations: Ideas and Software," *Mathematical Software*, Vol. III, Academic Press, New York, 1977, pp. 277-318.