JOURNAL OF COMPUTATIONAL PHYSICS 48, 366–386 (1982)

Black Box Multigrid*†

J. E. DENDY, JR.

Los Alamos National Laboratory, Los Alamos, New Mexico 87545

Received January 7, 1982

One major problem with the multigrid method has been that each new grid configuration has required a major programming effort to develop a code that specifically handles that grid configuration. Such a penalty is not required for methods like SOR, ICCG, etc.; in these methods, one need only specify the matrix problem, no matter what the grid configuration. In this paper, we investigate such a situation for the multigrid method. The end result is a code, BOXMG, in which one need only specify the (logically rectangular, positive definite) matrix problem; BOXMG does everything else necessary to set up the auxilliary coarser problems to achieve a multigrid solution.

I. INTRODUCTION

In the multigrid method, one attempts to solve a discrete approximation

$$L^M U^M = F^M \tag{1}$$

to a continuous equation

$$LU = F. \tag{2}$$

To do this one constructs a sequence of grids $G^1,\ldots, G^M$ with corresponding mesh sizes $h_1 > \cdots > h_M$. In its simplest mode of operation, one does a fixed number, IM, of relaxation sweeps (Gauss–Seidel, for example) on Eq. (1) and then drops down to grid G^{M-1} and the equation

$$L^{M-1}V^{M-1} = f^{M-1} \equiv I_M^{M-1}(F^M - L^M v^M), \tag{3}$$

where V^{M-1} is to be the coarse grid approximation to $V^M \equiv U^M - u^M$, where $v^M = u^M$ is the last iterate on grid G^M, and where I_M^{M-1} is an interpolation operator from G^M to G^{M-1}. To solve Eq. (3) approximately, one resorts to recursion, taking ID relaxation sweeps on grid G^k before dropping down to grid G^{k-1}, $M-1 \geqslant k \geqslant 2$ and the equation

$$L^{k-1}V^{k-1} = f^{k-1} \equiv I_k^{k-1}(f^k - L^k v^k). \tag{4}$$

† This work was performed under the auspices of the U.S. Department of Energy.

0021-9991/82/120366-21$02.00/0

When grid G^1 is reached, the equation $L^1V^1 = f^1$ can be solved directly and $v^2 \leftarrow v^2 + I_1^2 v^1$ performed. Then one does IU relaxation sweeps on grid G^{k-1} before forming $v^k \leftarrow v^k + I_{k-1}^k v^{k-1}$, $3 \leqslant k \leqslant M$. (This description assumes $M \geqslant 3$, the cases $M = 1$ or 2 being trivial.)

One advantage of the multigrid method is that one obtains a fixed reduction of the error, significantly less than one, in the residual $F^M - L^M u^M$ per work performed per unknown on grid G^M. This is in sharp contrast to most iterative methods, for example, SOR, where the reduction increases as a function of the number of unknowns on grid G^M. Another advantage is that in many cases, multigrid achieves truncation error in work that is a small multiple of the number of unknowns. For further details, see [1, 2].

In most implementations of the multigrid method, the operators I_{k-1}^k have been grid dependent. In the simplest case, G^k and G^{k-1} are rectangular grids, the grid points of G^{k-1} are a subset of the grid points of G^k, the grid spacing h_{k-1} of G^{k-1} is twice the grid spacing h_k of G^k, and the interpolation I_{k-1}^k is bilinear. (See [1].) If there are always to be G^k grid points on the boundary, then there is a constraint on the number of x [y] grid points NXM [NYM] on G^M that $\mathrm{NXM} = (\mathrm{NXO} - 1)\,2^{M-1} + 1$ [$\mathrm{NYM} = (\mathrm{NYO} - 1)\,2^{M-1} + 1$], where NXO [NYO] is the number of x [y] grid points on G^1. Otherwise, interpolation near the boundary is a special case. The coding of interpolation is further complicated by whether the points on the boundary represent knowns (as in Dirichlet boundary conditions) or unknowns (as in Neumann boundary conditions).

Figure 1 shows two grids for a cell-centered approximation to an elliptic equation. (The x's represent G^k and the $\otimes$'s G^{k-1}.) Now the above constraint on unknowns does not help since the nearest G^k point to the boundary is $h_k/2$ from the boundary and the nearest G^{k-1} point to the boundary is $h_{k-1}/4$ from the boundary, where h_{k-1} and h_k are G^1 and G^2 mesh spacings, respectively. The incorporation of a Neumann boundary condition, for example, on grid G^k leads to frequencies which are not damped out by bilinear interpolation, and convergence is degraded. Again something special (either in the interpolation routine or the relaxation routine) must be done at the boundary. This is easy in principle—especially if Brandt is nearby to advise—but is a pain in practice. There are two possible solutions in this case. One is to let

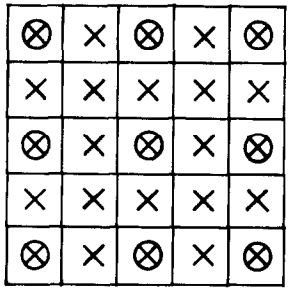

FIG. 1. Two grids for a cell-centered approximation to an elliptic equation (×), G^k; (⊗), G^{k-1}.

$h_{k-1} = 3h_k$, which again leads to the coding of a special interpolation. The other is to use G^{k-1} unknowns that are not a subset of G^k unknowns, as in Fig. 2. This latter solution was the one employed in [3]. Bilinear interpolation in this case involves special coding (for example, $a = \frac{1}{16}(9A + 3C + 3B + D)$ in Fig. 2), and there is again a constraint on the number of G^M unknowns to avoid special cases.

In addition to the grid structure, the actual difference equations cause programming difficulties. Consider, for example,

$$\begin{aligned} -\nabla \cdot (D(x,y)\,\nabla U(x,y)) + \sigma(x,y)\,U(x,y) &= f(x,y), && (x,y) \in \Omega, \\ \nu(x,y) \cdot D(x,y)\,\nabla U(x,y) + \gamma(x,y)\,U(x,y) &= 0, && (x,y) \in \partial\Omega, \end{aligned} \tag{5a}$$

where $\Omega = (0, A) \times (0, B)$ with boundary $\partial\Omega$, ν is the outward normal to $\partial\Omega$, D is positive, σ and γ are nonnegative, and D, σ, and f are allowed to be discontinuous across internal boundaries Γ of Ω; hence it is also assumed that

$$\begin{aligned} &U \text{ and } \mu \cdot (D\nabla U) \text{ are continuous at } (x,y) \text{ for almost every } (x,y) \in \Gamma \\ &(\text{where } \mu(x,y) \text{ is a fixed normal vector at } (x,y)). \end{aligned} \tag{5.b}$$

If the finite difference approximation of Eq. (5a) is a vertex centered one as in [2], then the *classic* multigrid method of [1] (I_{k-1}^k = bilinear interpolation, I_k^{k-1} = a fixed nine point weighting operator, and the coefficients of L^{k-1} a fixed weighting of the coefficients of L^k) performs well as long as the discontinuities in D are not too severe and as long as Γ does not consist of too many line segments; otherwise, it performs badly; indeed, it can even fail to converge in the fixed mode described above.

Alcouffe *et al.* [2] dealt with the situation in which D, σ, and f jump by orders of magnitude across Γ. They considered many possible choices of I_{k-1}^k, I_k^{k-1}, and L^{k-1}. Only one of these choices was found to be robust. This choice was $I_{k-1}^k = J_{k-1}^k$ (where J_{k-1}^k is defined below),

$$I_k^{k-1} = (I_{k-1}^k)^* \tag{6}$$

and

$$L^{k-1} = (I_{k-1}^k)^*\, L^k I_{k-1}^k. \tag{7}$$

The choices, Eqs. (6) and (7), are automatic in the finite element formulation of

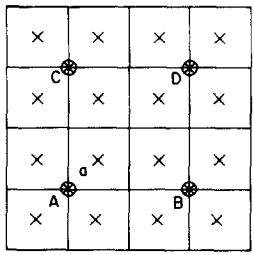

FIG. 2. Unknowns G^{k-1} that are not a subset of unknowns G^k.

multigrid [4]. Nicolaides [4] and Hackbush [5] both observed that $L^{k-1} = I_k^{k-1} L^k I_{k-1}^k$, with I_k^{k-1} not necessarily equal to $(I_{k-1}^k)^*$, is a good choice in that the residual of the corrected solution vanishes when transferred to the coarse grid. This can be shown to be a good feature if L^k is symmetric. In the finite element formulation of multigrid, I_{k-1}^k is also automatic. Indeed multigrid finite element with piecewise bilinear elements was one of the methods considered in [2] and found not to be robust.

The crucial choice, then, given Eqs. (6) and (7) is the choice of I_{k-1}^k. As discussed in [2], the first clue to the choice of I_{k-1}^k was that, because of Eq. (5b), I_{k-1}^k should approximately preserve the flux $\mu \cdot (D\nabla U)$ across Γ. In certain problems, however, when there were large jumps in both D and σ, it was discovered that on coarser grids where h^2 is large, the interfaces in σh^2 were as important as the interfaces in D. The obvious solution is to use the difference operator for the interpolation operator I_{k-1}^k. In one space dimension with three point difference operators, it is obvious how to do this. In two space dimensions, for the five point discrete Laplacian, it can also be done easily by the use of skewed five point discrete Laplacians; see [6]. This approach is doomed for Eq. (5) for two reasons. First, accurate skewed approximations are difficult if not impossible when interfaces are present. Second, even if L^M is a five point operator, the use of Eq. (7) generates nine point L^k's, $k < M$, making the above approach impossible. The solution arrived at in [2] is as follows: Suppose that at (IF, JF), L^k has the pointwise template

$$\begin{bmatrix} -T_{IF,JF+1}^k & -W_{IF,JF+1}^k & -R_{IF+1,JF+1}^k \\ Q_{IF,JF}^k & S_{IF,JF}^k & -Q_{IF+1,JF}^k \\ -R_{IF,JF}^k & -W_{IF,JF}^k & -T_{IF+1,JF}^k \end{bmatrix} \tag{8}$$

Form $\tilde{Q}_{IF+1,JF}^k = T_{IF+1,JF+1}^k + Q_{IF+1,JF}^k + R_{IF+1,JF}^k$; $\tilde{S}_{IF+1,JF}^k = -W_{IF+1,JF} + S_{IF+1,JF}^k - W_{IF+1,JF+1}^k$; and $\bar{Q}_{IF+2,JF}^k = T_{IF+2,JF}^k + Q_{IF+2,JF}^k + R_{IF+2,JF+1}^k$. Then, for horizontal lines embedded in the coarse grid, the interpolation J_{k-1}^k is given by

$$(J_{k-1}^k v^{k-1})_{IF+1,JF} = (\tilde{Q}_{IF+1,JF}^k v_{IC,JC}^{k-1} + \bar{Q}_{IF+2,JF} v_{IC+1,JC}^{k-1})/\tilde{S}_{IF+1,JF}^k. \tag{9}$$

(We have just summed Eq. (8) verically to average out its y-dependence.) A similar formula can be used for vertical lines embedded in the coarse grid squares. Then, at fine grid points centered in coarse grid squares, $v_{IF+1,JF+1}^k$ may be obtained from the difference formula; i.e.,

$$\begin{aligned} v_{IF+1,JF+1}^k = (&Q_{IF+1,JF+1}^k v_{IF,JF+1}^k + Q_{IF+2,JF+1}^k v_{IF+2,JF+1}^k \\ &+ W_{IF+1,JF+1}^k v_{IF+1,JF}^k + W_{IF+1,JF+2}^k v_{IF+1,JF+2}^k \\ &+ R_{IF+1,JF+1}^k v_{IF,JF}^k + R_{IF+2,JF+2}^k v_{IF+2,JF+2}^k \\ &+ T_{IF+1,JF+2}^k v_{IF,JF+2}^k + T_{IF+2,JF+1}^k v_{IF+2,JF}^k)/S_{IF+1,JF+1}^k. \end{aligned} \tag{10}$$

The vertical analogue of (8)–(10) completes the definition of J_{k-1}^k alluded to immediately preceding Eq. (6).

The near ultimate insult is a cell-centered difference approximation to Eq. (5) using $h_{k-1} = 3h_k$ or the grid structure of Fig. 2. The definition of I_{k-1}^k which approximately preserves flux across Γ in this case is not obvious, and the computation of $(I_{k-1}^k)^* L^k I_k^{k-1}$ is a disaster. Desperation being the mother of invention, one soon decides there has to be a better way.

II. Black Box Multigrid

The better way has already been described; one only needs to interpret it differently. The crucial observation is that once one has a pointwise template like Eq. (8) for L^M, then the definition of J_{k-1}^k and $(J_{k-1}^k)^* L^k J_{k-1}^k$ is independent of where this template came from. (We refer to this method as black box multigrid not because—as some would have it—multigrid is black magic, but because the code which implements the method acts as a block box for the user; he need only specify the difference equations on the finest grid since the code BOXMG generates the auxilliary coarse problems.)

The same artifice allows one to get rid of the restriction on the number of unknowns on the finest grid. For the situation depicted in Fig. 3, for example, one can imagine fictitious coarse grid points. The boundary conditions on the fine grid are incoporated into the operator, as in [2], so that for points (IF, JF) on the right boundary, for example, $R_{IF+1,JF+1}^k = Q_{IF+1,JF}^k = T_{IF+1,JF}^k = 0$ in Eq. (8). The boundary of the coarse grid does not coincide with the boundary of the fine grid, but the boundary conditions will be picked up by the formation of $(J_{k-1}^k)^* L^k J_{k-1}^k$.

An example of an extreme case of this artifice is the situation in which one wants to solve a Dirichlet problem on a given irregular region. One proceeds by embedding the region in a rectangle, writing down difference equations at points interior to the

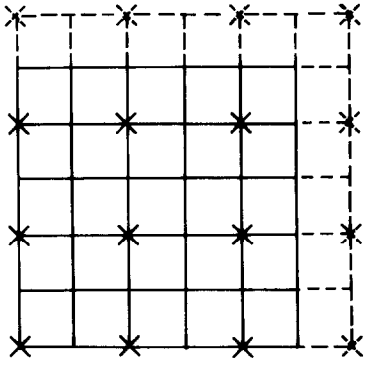

FIG. 3. Use of fictitious coarse grid points; the boundary of the coarse grid extends beyond the fine grid boundary.

region. These difference equations incorporate the Dirichlet data on the boundary of the region in such a way that there is no coupling between the interior points and the other points. At the other points, one writes down an equation $\sigma_{i,j}U_{i,j} = F_{i,j}$, where $\sigma_{i,j} \neq 0$ and $F_{i,j}$ are arbitrary. This artifice makes the problem logically rectangular. The solution to the difference equation is obtained at the interior points, and the solution $U_{i,j} = F_{i,j}/\sigma_{i,j}$ is obtained at the other points. On a serial machine, this process for solving irregular region problems may be inefficient for some regions, since the number of other points can be quite large. On a vector machine, however, the situation is unclear, since the embedding technique is immediately vectorizable and since other techniques may vectorize with difficulty.

One disadvantage to the black box method is storage. In the situation that the coefficients of the difference equations are easy to compute (for example, Laplace's equation on a rectangle), there is a storage penalty of at least five [seven] locations per fine grid point for the black box method for a five [nine] point operator; this assumes that the right-hand side is stored and that $1/S^M_{IF,JF}$ is computed and stored. If one is not going to restrict the number of unknowns on the finest frid, however, then not storing the coefficients means additional programming and checking for special cases. (If the checking involves an IF test in the inner loop of a double DO loop, the degradation in run time can be dramatic on a machine like a CDC 7600.) Moreover, we are more interested in problems like Eq. (5), where the coefficients of the difference equations are not easy to compute and have to be stored anyway.

If we assume that the finest grid coefficients are stored, then there is still a storage penalty for the black box method. First, even in the case that the operator on the finest grid is a nine point operator, nine point operators are generated on the coarser grids. If it is assumed that the given problem can be worked with five point operators on the coarser grids (an assumption which is not at all clear for Eq. (5)), then an extra two storage locations per coarse grid point are required, for a total of $2(1/4 + 1/16 + \cdots) = \frac{2}{3}$ locations per fine grid point. Second, the interpolation coefficients have to be stored, requiring four locations for Eq. (9) and its vertical analogue. (Unfortunately, since the coefficients in Eq. (9) do not necessarily sum to 1, both $\tilde{Q}^k_{IF+1,JF}/\tilde{S}_{IF+1,JF}$ and $\bar{Q}_{IF+2,JF}/\tilde{S}_{IF+1,JF}$ must be stored.) Equation (10) requires no additional storage but does require nine multiplies. By using Eq. (9) and its vertical analogue, Eq. (10) can be rewritten in terms of coefficients of $V^k_{IF,JF}$, $V^k_{IF+2,JF+2}$, $V^k_{IF,JF+2}$, and $V^k_{IF+2,JF}$; this reduces the computation of Eq. (10) to four multiplies but requires four storage locations per coarse grid point. Hence interpolation as currently implemented in BOXMG requires a total of $8(\frac{1}{4} + \frac{1}{16} + \cdots) = \frac{8}{3}$ locations per fine grid point.

One can also ask whether there is any disadvantage in execution time with the black box method. The worst case is that in which the operator on the finest grid is a five point operator; to be fair, let us assume that it is not the five point Laplacian, in order that advantage cannot be taken of the very simple form of the coefficients in the five point Laplacian. To be unfair to the black box method let us assume that I^{k-1}_k is injection in the *classic* multigrid method. Experimentally, for easy equations, BOXMG achieves a reduction of the error by a factor of 0.1 [0.05] per multigrid

cycle for IU = ID = 1 and IM = 2 [ID = 2, IU = 1, IM = 3]. This is, in contrast to figures of 0.25 and 0.125 for classic multigrid. If the total work for classic multigrid and black box multigrid is computed, including the work for I_{k-1}^{k} and I_{k}^{k-1} and if the comparison is expressed in terms of the convergence factor (convergence factor = reduction of error/work unit, where 1 work unit = 8 floating point operations, the amount of work for one Gauss–Seidel sweep on the finest grid), then the comparison is as in Table I. Thus there is no penalty in convergence factor for the black box method. There is a penalty, however, for the black box method in that the computation of I_{k-1}^{k} and L^{k-1} is not without cost; this is startup calculation time that does not have to be expended in the classic multigrid method. As soon as one considers the convergence factor for classic multigrid if nine point residual weighting is necessary (as it will be for all but the simplest problems), then the degradation of convergence factor makes it obvious that the black box method can pay for its overhead. Moreover, the black box method will work problems that classic multigrid cannot handle. If, however, one is solving just to trucation error, then the classic multigrid method is probably more efficient for problems with smooth coefficients. The extra expense of cubic interpolation the first time a grid is visited in the classic multigrid method is probably more than offset by the expense of computing L^{k}, $k < M$, in the black box method.

A relatively unimportant issue of implementation is whether $I_{k}^{k-1} = (J_{k-1}^{k})^{*}$ is necessary. In [2], a heuristic argument was made for this choice, but experiments seemed to indicate that the use of a fixed nine point weighting for I_{k}^{k-1} did not lead to any significant degradation of convergence factor, as long as $I_{k-1}^{k} = J_{k-1}^{k}$ and Eq. (7) is used to define L^{k-1}. (In some poblems, the fixed weighting even gave slightly better convergence.) A nine point fixed weighting for $I_{k}^{k-1} = (J_{k-1}^{k})^{*}$ is automatically correct at the boundary. Hence, since J_{k-1}^{k} is stored, it is easier to use $(J_{k-1}^{k})^{*}$ for I_{k}^{k-1}.

The multigrid algorithm described in Section I begins on the finest grid G^{M}. In the full multigrid algorithm described by Brandt [1, Sect. 6.3], one begins on the coarsest grid G^{1} instead and uses the coarser grids to generate a good initial guess. For three grids, for example, the pattern of grid transfer is $G^{1} \to G^{2} \to G^{1} \to G^{2} \to G^{3} \to G^{2} \to G^{1} \to G^{2} \to G^{3}$. In Brandt's scheme, when a grid is visited for the first time, cubic interpolation is used instead of bilinear interpolation, and when the finest frid (G^{3} in the example above) is visited for the second time, one has the solution to truncation

TABLE I

	Convergence factor, classic	Convergence factor, black box	Convergence factor, classic with residual weighting
IU = ID = 1, IM = 2	0.66	0.66	0.75
IU = 2, ID = 1, IM = 3	0.64	0.66	0.74

error. Indeed, for equations with smooth coefficients, not only are the pointwise values h^2 accurate, but the centered difference quotients approximate the first and second derivatives to h^2 accuracy.

In BOXMG, we have not implemented cubic interpolation nor the generalization of it ([2, Eq. (5.12)]) for equations with discontinuous coefficients; the reason is that numerical experiments indicated no advantage for either versus J^k_{k-1} in equations with discontinuous coefficients. This issue is discussed further in Section IV.

One final issue, discovered by Brandt, is that of the use of the right-hand side in interpolation. Generally, the use of the right-hand side provides an $O(h^2)$ correction to interpolation and is not worthwhile. In black box multigrid, however, the right-hand side at the boundary can contain boundary data, and in such cases, not using the right-hand side can lead to O(1) interpolation errors at the boundary, and consequently destroy all hope of solving to truncation error in one or two cycles. Thus, to the right-hand side of Eq. (9) we add $r^k_{IF+1,JF}/\tilde{S}^k_{IF+1,JF}$, and to the right-hand side of Eq. (10) we add $r^k_{IF+1,JF+1}/S^k_{IF+1,JF+1}$, where r^k is the residual; when a grid is visited for the first time $r^k = F^k$ (if a zero initial guess is used).

III. The Parameters of BOXMG

In this section, we discuss the parameters the user must specify to use BOXMG. These are actually discussed in the comments of BOXMG following the reading of the parameters, but we provide a little more detail here. We hope this description and the examples of Section IV will make the use of BOXMG clear. We had originally intended to rewrite BOXMG in perfect, portable Fortran. Ignoring for the moment whether such a beast exists, we discovered that we were phychologically incapable of the quest. Nevertheless, we still hope that BOXMG will prove useful and that its coding is clear enough to be changed by others for their devious ends.

The grid in BOXMG is always logically rectangular. The parameters NXM and NYM specify the number of unknowns in the x and y coordinates respectively. HXM and HYM specify the x and y spacing respectively on the finest grid; these parameters are only used in computing the discrete L^2 norm of the residual, since the user specifies the equations on the finest grid. Indeed, since the equations on the finest grid can be written on a Lagrangian grid, HXM and HYM may have little meaning in some cases.

The tolerance is denoted TOL. In one mode of BOXMG, iteration is continued until the discrete L^2 error of the residual is less than TOL or until the accumulated number of multigrid cycles NCYC is equal to |ISTRT|. (BOXMG assumes the equations to be in undivided form for this purpose; if they are not, the user must make the appropriate adjustment in TOL.)

We denote by IFD an indicator for the scheme on the finest grid. If IFD = 1, a five point scheme is assumed; otherwise, a nine point scheme is assumed. We have already discussed IU, ID, and IM in Section I. The recommended choices are IU = ID = 1 and IM = 2 or IU = 1, ID = 2, and IM = 3. For problems with smooth

coefficients, the latter choice is slightly better; for problems with rough coefficients the first choice is better. If ISTRT < 0, BOXMG will begin iterating on the finest grid. If ISTRT > 0, BOXMG will begin on the coarsest grid and will bootstrap itself up to the finest grid, as discussed in Section II, and then continue cycling. (The first full multigrid cycle in this case, where the finest grid is visited twice, is just counted as one cycle in the computation of NCYC.)

IRELAX is an indicator for the type of relaxation. IRELAX $= 1$ means point relaxation. IRELAX $= 2$ means line relaxation by lines in x. IRELAX $= 3$ means line relaxation by lines in y. IRELAX $= 4$ means line relaxation by lines in x followed by line relaxation by lines in y. These options are included for flexibility. For equations like $\varepsilon u_{xx} + u_{yy} = f$, $\varepsilon \ll 1$, (or for $\Delta u = f$, where $\Delta x \gg \Delta y$ on the finest grid) line relaxation by lines in y is needed for a good smoothing rate [2]. For $u_{xx} + \varepsilon u_{yy} = f$, line relaxation by lines in x is needed for a good smoothing rate. In some cases, both are needed.

ITAU is an indicator for computing and printing an estimate of the truncation error. If ITAU $= 0$, then the discrete L^2 norm of

$$J_M^{M-1} F^M - L_M^{M-1} \tilde{I}_M^{M-1} U^M, \tag{11}$$

where $\tilde{I}_M^{M-1}$ is injection, is computed and printed. If ITAU $\neq 0$, then Eq. (11) is not computed and printed. A discussion of this feature is given in Section IV.

ICOEF determines when $(J_{k-1}^k)^* L^k J_{k-1}^k$ will be computed. If ICOEF $= 0$, then when M, the number of grids is computed, ICOEF will be set equal to M, and $(J_{k-1}^k)^* L^k J_{k-1}^k$ will be computed for $k \leqslant M$. If ICOEF $= 1$, then L^k must be specified for every grid, G^k, $1 \leqslant k \leqslant M$ since $(J_{k-1}^k)^* L^k J_{k-1}^k$ will not be computed for any grid. This feature allows the user to run something like classic multigrid except that J_{k-1}^k will still be computed by the code and may or may not be bilinear interpolation; hence, the use of this option (ICOEF $= 1$) may lead to divergence if discontinuities of order of magnitude exist in the coefficients.

IVW, MCYCL, ALPHL, and ALPHM are cycling parameters. The type of cycle to be performed is determined by IVW. If IVW $= 1$, the usual V-cycle will be performed. If IVW $= 2$, W-cycles will be performed. Larger values of IVW give more exotic patterns. If ISTRT > 0, grid G^k, $k < M$ will be visited MCYCL times before grid G^{k+1} is visited unless $\|s^k\|$, the discrete L^2 norm of $J_k^{k-1} f^k - L^{k-1} \tilde{I}_k^{k-1} U^k$ is less than ALPHL (ALPHM if $k = M$) times $\|r^k\|$, the discrete L^2 norm of the residual on G^k. Likewise, grid G^M will be visited |ISTRT| times (ISTRT + 1 times if ISTRT > 0) unless $\|s^M\|$ is less than ALPHM $\|r^M\|$. The usual value of MCYCL is 1. The theoretical value of ALPHM to achieve truncation error is 2^{-2d+1}, where d is the space dimension ($d = 2$ in this paper). This value assumes the equations are in undivided form and that the G^k residuals are computed dynamically (and thus roughly twice the size of the static G^k residuals). The theoretical value for ALPHL—under the same assumptions—is $2^{-d-p+1}(2^d - 1)^{-1}$, where p is the order of approximation; this value is derived as in [9, Sect. 5.2] and gives the point at which work on grid G^{k+1} becomes as efficient as additional work on grid G^k. (For

the examples in this paper, the value of ALPHL is immaterial, since we always take MCYCL = 1.) Finally, if one is solving in the mode where the discrete L^2 norm of the residual is to be reduced to less than TOL, then ALPHM = 0 should be used. The flexibility provided by IVW, MCYCL, ALPHL, and ALPHM is awesome.

Aside from specifying these parameters, the user must provide the subroutine PUTF, which specifies the difference equations on the finest grid. (As remarked above, certain values of ICOEF would require PUTF to make sense for coarser grids as well.) An example of a PUTF is given in the listing of BOXMG in [7]. PUTF has one argument K, the number of the grid, and a call to KEY in it, CALL KEY(K, JST, II, JJ, HX, HY), which fetches the storage for the arrays. For IFD = 1, the user must specify the arrays FR, FA, SO, and QF. For IFD $\neq$ 1, he must specify FSW and FNW as well. The logical grid is assumed to be (I, J); $I = 1,..., II$; $J = 1,..., JJ$. The sets $\{(1, J): J = 1,..., JJ\}$, $\{(II, J): J = 1,..., JJ\}$, $\{(I, 1): I = 1,..., II\}$, and $\{(I, JJ): I = 1,..., II\}$ are fictitious points, assumed for ease of programming. For IFD = 1, the template

$$\begin{bmatrix} & -\mathrm{FA}(JP+I) & \\ -\mathrm{FR}(JO+I) & \mathrm{SO}(JO+I) & -\mathrm{FR}(JO+I+1) \\ & -\mathrm{FA}(JO+I) & \end{bmatrix},$$

is assumed at gridpoint (I, J) of grid k, where $JO = JST(J)$ and $JP = JST(J+1)$. For IFD $\neq$ 1, the template

$$\begin{bmatrix} -\mathrm{FNW}(JP+I) & -\mathrm{FA}(JP+I) & -\mathrm{FNW}(JP+I+1) \\ -\mathrm{FR}(JO+I) & \mathrm{SO}(JO+I) & -\mathrm{FR}(JO+I+1) \\ -\mathrm{FSW}(JO+I) & -\mathrm{FA}(JO+I) & -\mathrm{FNW}(JO+I+1) \end{bmatrix}$$

is assumed. In both cases, QF($JO + I$) should be the right hand side, $I = 2,..., I1 \equiv II - 1$; $J = 2,..., J1 \equiv JJ - 1$ in PUTF. The boundary conditions should be incorporated into the operator, so that all coefficients referring to fictitious points should be zero. For example, FR($JO + 2$), FSW($JO + 2$), and FNW($JP + 2$) should all be zero, $J = 1,..., JJ$.

The program BOXMG automatically determines the number of grids M from the input parameters NXM and NYM. It does this by bisecting the given logical grid until it arrives at a grid which cannot be practically bisected any further, i.e., when the number of x or y unknowns is three or four. (For NXM $\gg$ NYM or NXM $\ll$ NYM, this may lead to the situation of its being profitable to bisect the coarsest grid only in the x or y direction, but this feature is not provided in BOXMG.) Once the number of levels is determined, BOXMG computes how much storage must be allowed for the various arrays. If insufficient storage has been declared, a message is printed and the code terminates. Further details on storage are given in the documentation in BOXMG.

IV. Examples

All examples in this section assume the difference equations to be in undivided form; thus the $\tau_{i,j}^{M-1}$ of (11) has a factor of h^2 multiplying it which is not present in the usual divided form of this quantity.

The first example is for Eq. (5) for $\Omega = (0, 24) \times (0, 24)$. The boundary conditions are

$$\frac{\partial u}{\partial v} = -u/2D, \qquad \text{on} \quad y = 24 \quad \text{or} \quad x = 24,$$

$$= 0, \qquad \text{otherwise,}$$

and D is given by

$$D(x, y) = \begin{cases} 1, & \text{if} \quad (x, y) \in \{[0, 12) \times [0, 12)\} \cup \{(12, 24] \times (12, 24]\} \\ 1000, & \text{otherwise.} \end{cases}$$

We take $\sigma = 1/3D$, and $f = 0$ when $D = 1$ and $f = 1$ when $D = 1000$. The results are

TABLE II

Number of unknowns	CPU time in seconds on CDC 7600	$\max_{i,j} \lvert\tau_{i,j}^{M-1}$ and estimates of r	Reduction in discrete L^2 norm of residual in last cycle and number of cycles
13×13 ALPHM = 0.125	0.017	2.06	0.09; NCYC = 2
25×25 ALPHM = 0.125	0.052	6.00, −1; $r = 1.61$	0.09; NCYC = 2
49×49 ALPHM = 0.125	0.170	1.84, −1; $r = 1.171$	0.07; NCYC = 2
13×13 TOL = 10^{-6}	0.036	2.06	0.13; NCYC = 7
25×25 TOL = 10^{-6}	0.126	6.00, −1; $r = 1.61$	0.34; NCYC = 9
49×49 TOL = 10^{-6}	0.430	1.84, −1; $r = 171$	0.46; NCYC = 13
49×49 TOL = 10^{-6}, IVW = 2	0.554	1.84, −1	0.23; NCYC = 8
48×48 ALPHM = 0.125	0.156	2.74, −1	0.07; NCYC = 2

summarized in Table II. In this table, 1.64, −1, for example, is used for 1.64×10^{-1}. Also $\tau_{i,j}^{M-1}$ is the quantity in (11); the number r is the exponent in the asymptotic expansion of the τ's; it is computed by the formula

$$\log(\tau^{2h}/\tau^{h})/\log 2.$$

The first three rows show the results of running with ALPHM = 0.125; the next four rows continue from there until the discrete L^2 norm of the error is less than 10^{-6}. In the last row, HXM and HYM are still 0.5, so that the region is really $(0., 23.) \times (0., 23.)$; this example illustrates the situation in Fig. 3 for the transition from G^k to G^{k-1}, $k = 5, 4, 3, 2$. Since $(0., 23.) \times (0., 23.)$ is a small perturbation of $(0., 24.) \times (0., 24.)$, one would expect comparable results for the two cases. As an example of parameters, for the last row NXM = NYM = 48, HXM = HYM = 0.5, TOL = whatever, IFD = 1, IU = 1, ID = 1, IM = 2, ISTRT = 20, IRELAX = 1, ITAU = 0, ICOEF = 0, IVW = 1, MCYCL = 1, ALPHL = 1/24, ALPHM = 0.125.

The second example is the same as the first, except that cell-centered differencing is employed. For the runs made, the interface comes midway between cell centers. In one dimension, if an interface is located at ih and $D(x) = D_+$ if $x > ih$ and $D(x) = D_-$ if $x < ih$, then the difference equation at $(i - 1/2)h$ is

$$-D_- u_{-i-3/2} + \left(D_- + \frac{1}{1/2(D_+^{-1} + D_-^{-1})}\right) u_{i-1/2} - \frac{1}{1/2(D_+^{-1} + D_-^{-1})} u_{i+1/2};$$

a similar formula holds in two dimensions. The results are summarized in Table III. Sample parameters for this problem are those in the last row, where NXM = NYM = 48, HXM = HYM = 0.5, TOL = 10^{-6}, IFD = 1, IU = 1, ID = 1, IM = 2, ISTRT = 50, IRELAX = 1, ITAU = 0, ICOEF = 0, IVW = 1, MCYCL = 1, ALPHL = 1/24, ALPHM = 0. $\mathrm{Max}_{i,j} |\tau_{i,j}^{M-1}|$ is assumed next to (24., 12.), near the interface and right boundary. Away from the interfaces, the $\tau_{i,j}^{M-1}$'s are well behaved. Let us examine the answers at (24., 12.) and compare them with those obtained from the vertex-centered scheme. By using the approximation to the boundary condition for a horizontal averaging and conservation of flux for vertical averaging, we can get approximations to the solution at (24., 12.) for the cell-centered scheme; call them $\bar{u}_{cc}^3$, $\bar{u}_{cc}^4$, $\bar{u}_{cc}^5$ (for finest grid 12×12, 24×24, and 48×48, respectively, and tolerance 10^{-6}). Let u_{vc}^3, u_{vc}^4, u_{vc}^5 be the answers from the vertex-centered scheme at (24., 12.) (for finest grid 13×13, 25×25, and 49×49, respectively, and tolerance 10^{-6}). Compute $\bar{u}_{cc}^{RE} = 4/3\bar{u}_{cc}^5 - 1/3\bar{u}_{cc}^4$ and $u_{vc}^{RE} = 4/3u_{vc}^5 - 1/3u_{vc}^4$. (RE ∈ Richardson extrapolation.) Then $\bar{u}_{cc}^3 - \bar{u}_{cc}^{RE} = 0.801$, $\bar{u}_{cc}^4 - u_{cc}^{RE} = 0.226$, and $\bar{u}_{cc}^5 - u_{cc}^{RE} = 0.0506$; and $u_{vc}^3 - u_{vc}^{RE} = 0.766$, $u_{vc}^4 - u_{vc}^{RE} = 0.224$, and $u_{vc}^5 - u_{vc}^{RE} = 0.056$. Thus the assumption of asymptotic error of Ch^2 at (24., 12.) for both schemes is justified, and—at least for this example—there is no reason from considerations of accuracy to prefer the vertex-centered scheme to the cell-centered scheme. (We have also checked points away from the interfaces and boundaries, and the same conclusion—less interesting in these cases—is valid.)

TABLE III

Number of unknowns	CPU time in Seconds on CDC 7600	$\max_{i,j} \lvert\tau_{i,j}^{M-1}\rvert$ and estimate of r	Reduction in discrete L^2 norm of residual in last cycle and number of cycles
12×12 ALPHM = 0.125	0.014	4.82	0.13; NCYC = 2
24×24 ALPHM = 0.125	0.045	3.73; $r = 0.37$	0.10; NCYC = 2
48×48 ALPHM = 0.125	0.116	2.30; $r = 0.70$	0.08; NCYC = 1
12×12 TOL = 10^{-6}	0.038	4.87	0.14; NCYC = 9
24×24 TOL = 10^{-6}	0.103	3.75; $r = 0.38$	0.12; NCYC = 8
48×48 TOL = 10^{-6}	0.354	2.26; $r = 0.73$	0.10; NCYC = 8

The third example is

$$-\Delta U = F \quad \text{on} \quad \Omega = (0, 1) \times (0, 1),$$
$$U = 0 \quad \text{on} \quad \partial\Omega,$$

where F is chosen so the solution is $U(x, y) = 3e^{x-y}xy(1-x)(1-y)$. The only way one can handle such a Dirichlet problem with BOXMG is to incorporate the boundary data into the right-hand side of the finest grid. Thus the difference operator along the $x = 0$ boundary away from the corners is

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ 0 & 4 & -1 \\ & -1 & \end{bmatrix},$$

i.e., the boundary is not treated as part of the grid at all. To have the boundary treated as part of the grid in this case would have required numerous special cases in BOXMG; hence, we decided not to implement this option.

The results are summarized in Table IV. In this table,

$$D^0 U_{i,j} = (u_{i+i,j} - u_{i-i,j})/2h.$$

In this example, two cycles appear to be sufficient to solve nearly to truncation error in both the function values and their derivatives, even though cubic interpolation is not employed. An example of parameters for this problem is for the fourth row,

TABLE IV

Number of unknowns	CPU time in seconds on CDC 7600	$\max_{i,j} \lvert u_{i,j} - U_{i,j} \rvert$ and experimental p	$\max_{i,j} \left\lvert D^0 u_{i,j} - \frac{\partial U}{\partial x} \right\rvert_{i,j} \rvert$ and experimental q	$\max_{i,j} \lvert \tau_{i,j}^{M-1} \rvert$ and experimental r	Reduction in discrete L^2 norm of residual in last cycle and number of cycles
9×9 ALPHM = 0.125	0.011	1.01, −3	1.13, −2	6.26, −3	0.05; NCYC = 2
19×19 ALPHM = 0.125	0.032	2.35, −4; $p = 2.00$	3.15, −3; $q = 1.84$	8.17, −4; $r = 2.93$	0.03; NCYC = 2
39×39 ALPHM = 0.125	0.109	6.37, −5; $p = 1.99$	8.37, −4; $q = 1.91$	7.28, −5; $r = 3.48$	0.02; NCYC = 2
9×9 TOL = 10^{-6}	0.018	1.02, −3	1.12, −2	6.27, −3	0.09; NCYC = 4
19×19 TOL = 10^{-6}	0.052	2.56, −4; $p = 1.99$	3.15, −3; $q = 1.83$	8.16, −4; $r = 2.99$	0.11; NCYC = 5
39×39 TOL = 10^{-6}	0.174	6.42, −5; $p = 2.00$	8.37, −4; $q = 1.91$	7.24, −5; $r = 3.49$	0.11; NCYC = 5
39×39 ISTRT = 1	0.082	7.48, −5; $p = 1.76$	9.82, −4; $q = 1.76$	6.99, −5; $r = 3.54$	0.01; NCYC = 1

where NXM = NYM = 9, HXM = HYM = 0.1, TOL = 10^{-6}, IFD = 1, IU = 1, ID = 1, IM = 2, ISTRT = 50, IRELAX = 1, ITAU = 0, ICOEF = 0, IVW = 1, MCYCL = 1, ALPHL = 1/24, ALPHM = 0.

The fourth example is

$$\begin{aligned} -\Delta U &= F \qquad \text{on} \quad \Omega = \text{circle of diameter 1. centered at (0., 0.)}, \\ U(x,y) &= g(x,y) = 3e^x e^{-y} xy(1-x)(1-y) \qquad \text{if} \quad (x,y) \in \partial\Omega, \end{aligned} \tag{12}$$

where F is chosen so that the solution is $U(x,y) = 3e^x e^{-y} xy(1-x)(1-y)$. This example illustrates the technique of embedding. We embed Ω in $\Omega' = (-0.5,\ 0.5) \times (-0.5, 0.5)$. At points in $\Omega' \backslash \bar{\Omega}$, we write down the equation $u = 0.$; at points in Ω whose north, south, east, and west neighbors are in $\bar{\Omega}$, we use the usual five point Laplacian. For simplicity, we use the simplest treatment of points that do not fall into either of the above sets. Consider, for example, a point $U^M_{i,j}$ in Ω whose neighbor $U^M_{i+i,j}$ is not in $\bar{\Omega}$, and let the distance from $U^M_{i,j}$ to $\partial\Omega$ be θh. Approximating $-h^2 U_{xx}$ by $(-U^M_{i-i,j} + 2U^M_{i,j} - U^M_{i+i,j})$ and using the relation $g((i+\theta)\,h, h) \cong (1-\theta)\,U^M_{i,j} + \theta U^M_{i+i,j}$ to solve for $U^M_{i+i,j}$ gives the following difference equation at (ih, jh):

$$-U^M_{i-i,j} + (3 + (1/\theta))\,U^M_{i,j} - U^M_{i,j-1} - U^M_{i,j+1} = h^2 F(ih, jh) + (1/\theta)\,g((i+\theta)\,h, jh); \tag{13}$$

note that there is no coupling between (ih, jh) and $((i+1)\,h, jh)$. The results are summarized in Table V.

The fifth example is the same as the fourth, except that in this case we use mapping to solve it. That is, we map the boundary of Ω onto the boundary of $\Omega'' = (0,\ 1) \times (0,\ 1)$, giving x and y as a function of ξ and η on $\partial\Omega''$ and solving approximately the problem

$$\begin{aligned} \Delta x &= 0, \qquad (\xi,\eta) \in \Omega'', \qquad x = x(\xi,\eta), \qquad (\xi,\eta) \in \partial\Omega'', \\ \Delta y &= 0, \qquad (\xi,\eta) \in \Omega'', \qquad y = y(\xi,\eta), \qquad (\xi,\eta) \in \partial\Omega''. \end{aligned} \tag{14}$$

We do this by discretizing Ω'', approximating Eq. (14) by five point Laplacians and specifying

$$\begin{aligned} x(ih,\ 1) &= \tfrac{1}{2}\cos((3\pi/4) - ih(\pi/2)), & y(ih,\ 1) &= \tfrac{1}{2}\sin((3\pi/4) - ih(\pi/2)), \\ x(ih,\ 0) &= \tfrac{1}{2}\cos((5\pi/4) + ih(\pi/2)), & y(ih,\ 0) &= \tfrac{1}{2}\sin((5\pi/4) + ih(\pi/2)), \\ x(0,\ ih) &= \tfrac{1}{2}\cos((5\pi/4) - ih(\pi/2)), & y(0,\ ih) &= \tfrac{1}{2}\sin((5\pi/4) - ih(\pi/2)), \\ x(1,\ ih) &= \tfrac{1}{2}\cos((-\pi/4) + ih(\pi/2)), & y(1,\ ih) &= \tfrac{1}{2}\sin((-\pi/4) + ih(\pi/2)). \end{aligned}$$

Ideally, Eq. (14) should be solved by multigrid, but for simplicity in this example, we used SOR. Equation (13) transforms to the following in the $\xi - \eta$ coordinate system:

$$\begin{aligned} (G_1 y_\eta - G_2 x_\eta)_\xi + (G_1 x_\xi - G_2 y_\xi)_\eta &= FJ, \qquad (\xi,\eta) \in \Omega'', \\ u(\xi,\eta) &= g(\xi,\eta), \qquad (\xi,\eta) \in \partial\Omega'', \end{aligned} \tag{15}$$

TABLE V

number of unknowns	CPU time in seconds on CDC 7600	$\max_{i,j} \lvert u_{i,j} - U_{i,j} \rvert$ and experimental p	$\max_{i,j} \left\lvert D^0 U_{i,j} - \frac{\partial U}{\partial x} \right\rvert_{i,j}$ and experimental q	$\max_{i,j} \lvert \tau_{i,j}^{M-1} \rvert$ and experimental r	Discrete L^2 norm of error and estimate of p	Reduction in discrete L^2 norm of residual in last cycle and number of cycles
9×9 ALPHM = 0.125	0.011	3.21, −3	2.53, −2	3.15, −2	5.37, −4	0.04, NCYC = 2
19×19 ALPHM = 0.125	0.032	9.31, −4; $p = 1.78$	8.11, −3; $q = 1.64$	1.71, −1; $r = 2.44$	1.72, −4; $p = 1.64$	0.07, NCYC = 2
39×39 ALPHM = 1.25	0.109	7.24, −4; $p = 0.36$	3.77, −3; $q = 1.11$	1.62, −1; $r = 0.08$	1.91, −4; $p = -0.15$	0.10, NCYC = 2
9×9 TOL = 10^{-6}	0.018	3.21, −3	2.53, −2	3.15, −2	5.37, −4	0.04, NCYC = 5
19×19 TOL = 10^{-6}	0.073	9.14, −4; $p = 1.81$	7.94, −3; $q = 1.67$	1.71, −1; $r = -2.44$	1.66, −4; $p = 1.69$	0.10, NCYC = 8
39×39 TOL = 10^{-6}	0.269	2.50, −4; $p = 1.87$	2.77, −3; $q = 1.51$	1.61, −1; $r = 0.09$	4.82, −5; $p = 1.78$	0.13, NCYC = 9
39 ALPHM = 0.05	0.134	2.77, −4; $p = 1.74$	2.80, −3; $q = 1.53$	1.61, −1; $r = 0.09$	6.04, −5; $p = 1.51$	0.11, NCYC = 3

TABLE

Number of unknowns	CPU time in seconds on CDC 7600	$\max_{i,j} \lvert u_{i,j} - U_{i,j} \rvert$ and experimental q	Maximum derivative error and estimate of q	Discrete L^2 norm of error and estimate of p
10×10 ALPHM $= 0.125$	0.012	1.66, −2	3.35, −1	4.45, −3
20×20 ALPHM $= 0.125$	0.037	7.35, −3; $p = 1.17$	3.95, −1; $q = -0.24$	1.76, −3; $p = 1.33$
40×40 ALPHM $= 0.125$	0.140	4.89, −3; $p = 0.58$	4.19, −1; $q = -0.09$	8.89, −4; $p = 0.99$
10×10 TOL $= 10^{-6}$	0.025	1.66, −2	3.35, −1	4.45, −3
20×20 TOL $= 10^{-6}$	0.078	7.40, −3; $p = 1.17$	3.96, −1; $q = -0.24$	1.77, −3; $p = 1.33$
40×40 TOL $= 10^{-6}$	0.327	4.86, −3; $p = 0.61$	4.22, −1; $q = -0.09$	8.95, −4; $p = 0.98$

where $G_1 = (1/J)(u_\xi y_\eta - u_\eta y_\xi)$, $G_2 = (1/J)(u_\eta x_\xi - u_\xi x_\eta)$, and $J = x_\xi y_\eta - x_\eta y_\xi$. Equation (15) is differenced in cell-centered form, and IRELAX $= 1$ was used. The results are summarized in Table VI.

Since J is singular at the corners of Ω'', it is not surprising that the error in the approximation to the x derivative (the finite difference version of $(1/J)(y_\eta u_\xi - y_\xi u_\eta)$) grows larger as the mesh is refined. For the fixed point $(0.1, 0.1)$—the interior point nearest $(0, 0)$ on the 10×10 $\xi - \eta$ grid—this error decreases; nevertheless the maximum error in the approximation to the x derivative grows and is always assumed at a point nearest one of the corners of Ω''.

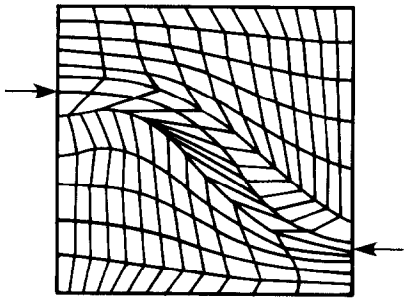

FIG. 4. Mesh, used in a Rayleigh–Taylor calculation [3], which presents a challenge to the black box approach.

VI

discrete L^2 norm of x derivative error and estimate of q	x derivative at (0.1, 0.1) and estimate of q	x derivative error at (0.5, 0.5) and estimate of q	$\max_{i,j} \lvert\tau_{i,j}^{M-1}\rvert$ and estimate of r	Reduction in discrete L^2 norm of residual in last cycle and number of cycles
4.63, −2	3.34, −1	1.97, −2	2.52, −1	0.04, NCYC = 2
3.85, −2; $q = 0.26$	1.86, −1; $q = 0.04$	5.48, −3; $q = 1.85$	4.47, −1; $r = -0.83$	0.04, NCYC = 2
2.75, −2; $q = 0.48$	9.22, −2; $q = 1.01$	1.53, −3; $q = 1.84$	8.03, −1; $r = -0.85$	0.05, NCYC = 2
4.63, −2	3.35, −1	1.93, −2	2.52, −1	0.07, NCYC = 5
3.85, −2; $q = 0.26$	1.86, −1; $q = 0.84$	5.58, −3; $q = 1.80$	4.46, −1; $r = -0.82$	0.10, NCYC = 7
2.75, −2; $q = 0.49$	9.33, −2; $q = 1.00$	1.56, −3; $q = 1.84$	8.03, −1; $r = -0.85$	0.13, NCYC = 9

The sixth example uses the mesh in Fig. 4, which was the mesh used in a Rayleigh–Taylor calculation in [3]. We include it since it is rather distorted (in fact, as commented in [3], a *bowtie* forms on the next time step) and represents a challenge to the black box approach. We use the same differencing as employed for Eq. (15). For this example, it is not clear what continuous system is being approximated. If, however, we use Dirichlet data identically equal to 1 and $F \equiv 0$, then the solution to the difference equations is identically 1. The results are summarized in Table VII.

V. Caveats and Extensions

The examples of Section IV all exhibit good behavior. We begin this section with three examples which do not. The first is

$$-\Delta u - \varepsilon u = F \qquad \text{on} \quad \Omega = (0.1) \times (0.1),$$

$$\frac{\partial u}{\partial v} = 0 \qquad \text{on} \quad \partial\Omega.$$

One can solve this problem with BOXMG until ε becomes too large; ε is too large when relaxation sweeps on grid G^2 magnify, instead of reduce the error. The remedy would be to change BOXMG to allow the coarsest grid G^1 to be finer; with this

TABLE VII

Number of unknowns	CPU time in in seconds on CDC 7600	$\max_{i,j} \lvert u_{i,j} - U_{i,j} \rvert$	Maximum error in x derivative	L^2 norm of error	L^2 norm of error in x derivative	$\max_{i,j} \lvert \tau_{i,j}^{M-1} \rvert$	Reduction in discrete L^2 norm of residual in last cycle and number of cycles
12×12 ALPHM = 0.125 IRELAX = 1	0.016	1.46, −1	1.29, −2	1.95, 0	1.50, −1	2.98, −1	0.42, NCYC = 2
12×12 TOL = 10^{-6} IRELAX = 1	0.082	8.55, −7	1.45, −5	1.05, −5	1.48, −6	3.56, 1	0.52, NCYC = 20
12×12 ALPHM = 0.125 IRELAX = 4	0.017	1.14, −2	8.96, −3	1.82, 0	1.31, −1	3.22, 1	0.16, NCYC = 1
12×12 IRELAX = 4 TOL = 10^{-6}	0.079	2.35, −7	1.85, −8	3.88, −6	2.76, −7	3.57, 1	0.23, NCYC = 10

remedy, BOXMG could be extended to handle some nondefinite symmetric problems. See the discussion in [1, Sect. 4.1].

Another example of poor behavior is for a difference operator with a template like

$$\begin{bmatrix} -\varepsilon & -\varepsilon & -1 \\ -\varepsilon & 2+6\varepsilon & -\varepsilon \\ -1 & -\varepsilon & -\varepsilon \end{bmatrix}, \tag{16}$$

where $\varepsilon \ll 1$. None of the relaxation options in BOXMG provides good smoothing on the finest grid for such an operator. The remedy is to write a block relaxation routine which relaxes the strongly coupled one dimensional sets as blocks; in this case they are the southwest to northeast diagonals. Such a template as Eq. (16) can arise in physically meaningful problems; see [8], for example. (In [8], however, situations like Eq. (16) would arise so infrequently as not to be worth the effort of special treatment.)

A final example of poor behavior is when the difference operator on the finest grid is close to the skewed Laplacian (or any operator with strong connections like the skewed Laplacian),

$$\begin{bmatrix} -1 & -\varepsilon & -1 \\ -\varepsilon & 4(1+\varepsilon) & -\varepsilon \\ -1 & -\varepsilon & -1 \end{bmatrix},$$

where $\varepsilon \ll 1$. This situation was discussed in [3]. The BOXMG program can be easily modified to handle this situation using the ideas in [10].

Several extensions of BOXMG are possible and are under investigation. Two, fairly straightforward, are to symmetric systems and three-dimensional problems. The third, more difficult, is to nonsymmetric equations. The fourth is to handle equations on arbitrary regions without resorting to embedding. The fifth is local mesh refinement—both fixed and adaptive. For all except the first two extensions, it is not clear at this time how much of the black box philosophy can be retained, and in the third and fourth extensions, it is not clear if there is a uniform strategy for both serial and vector machines.

VI. Code Listing

A listing of BOXMG is contained in [7], which may be obtained by requesting it from: J. E. Dendy, Jr., T-7, MS-610, Los Alamos National Laboratory, Los Alamos, NM 87545.

Acknowledgment

We thank Achi Brandt for advice that improved this paper.

REFERENCES

1. A. BRANDT, *Math. Comput.* **31** (1977), 333.
2. R. E. ALCOUFFE, A. BRANDT, J. E. DENDY JR., AND J. W. PAINTER, *SIAM J. SISSC* **2** (4) (1981), 430.
3. A. BRANDT, J. E. DENDY, JR., AND H. RUPPEL, *J. Comput. Phys.* **34** (3) (1980), 348.
4. R. A. NICOLAIDES, *J. Comput. Phys.* **19** (1975), 418.
5. W. HACKBUSH, *Comput.* **20** (1978), 291.
6. J. M. HYMAN *J. Comput. Phys.* **23** (2) (1977), 124.
7. J. E. DENDY, "Black Box Multigrid," Los Alamos report No. LA-UR-81-2337.
8. D. HEWETT, *J. Comput. Phys.* **38** (3) (1980), 378.
9. A. BRANDT, "Stages in Developing Multigrid Solutions," to appear.
Applied Math., Weizmann Institute, 1981.