AAEC/E633

AUSTRALIAN ATOMIC ENERGY COMMISSION
RESEARCH ESTABLISHMENT
LUCAS HEIGHTS RESEARCH LABORATORIES

SUBROUTINE MLTGRD
A MULTIGRID ALGORITHM BASED ON MULTIPLICATIVE CORRECTION AND IMPLICIT NON-STATIONARY ITERATION

by

J.M. BARRY
J.P. POLLARD

NOVEMBER 1986

ISBN 0 642 59840 1

AUSTRALIAN ATOMIC ENERGY COMMISSION
RESEARCH ESTABLISHMENT

LUCAS HEIGHTS RESEARCH LABORATORIES

SUBROUTINE MLTGRD
A MULTIGRID ALGORITHM BASED ON MULTIPLICATIVE CORRECTION AND IMPLICIT NON-STATIONARY ITERATION

by

J.M. BARRY
J.P. POLLARD

ABSTRACT

A FORTRAN subroutine MLTGRD is provided to solve efficiently the large systems of linear equations arising from a five-point finite difference discretisation of some elliptic partial differential equations. MLTGRD is a multigrid algorithm which provides multiplicative correction to iterative solution estimates from successively reduced systems of linear equations. It uses the method of implicit non-stationary iteration for all grid levels.

National Library of Australia card number and ISBN 0 642 59840 1

The following descriptors have been selected from the INIS Thesaurus to describe the subject content of this report for information retrieval purposes. For further details please refer to IAEA-INIS-12 (INIS: Manual for Indexing) and IAEA-INIS-13 (INIS: Thesaurus) published in Vienna by the International Atomic Energy Agency.

ALGORITHMS; COMPUTER CALCULATIONS; ITERATIVE METHODS; M CODES; MATRICES; MESH GENERATION; MULTIGROUP THEORY; NEUTRON DIFFUSION EQUATION; PARTIAL DIFFERENTIAL EQUATIONS

CONTENTS

1. INTRODUCTION

There are various iterative schemes for solving the N × N linear system

$$Ax = b. \tag{1.1}$$

For many of these, the error measure defined at the $(n+1)^{th}$ iterations as

$$e^{(n+1)} = \frac{\left\| x^{(n+1)} - x^{(n)} \right\|}{\left\| x^{(n)} \right\|} , \tag{1.2}$$

where $\|...\|$ is a suitable vector norm, behaves as

$$e^{(n+1)} / e^{(n)} \ll e^{(n+m+1)} / e^{(n+m)}$$

for m sufficiently large. This slowing down in the rates of convergence applies generally for iterative procedures, including the method of implicit non-stationary iteration - MINI [Barry and Pollard 1977, 1978]. Speculation by Brandt [1977], Nakamura [1977] and Barry *et al.* [1983] suggested that the slowing down in the rate of convergence arises because iterative schemes have trouble in removing the low frequency components of the error vector

$$\Delta^{(n)} = x - x^{(n)} .$$

The ability of MINI to remove high frequency error components has been demonstrated [Barry *et al.* 1983]. Overall convergence would be improved by combining MINI with a technique for removing the low frequency components, such as coarse mesh problem redefinition. Hopefully, the low frequency components may be adequately represented by redefining the problem over a coarser geometric grid. On this grid, they may be removed with much less computational effort. The solution obtained over the coarse grid may then be used to modify the intermediate solution on the fine grid, or possibly even to provide an initial estimate for the fine grid.

The latter approach has been applied in neutron diffusion calculations for some time. Roe [1954] devised a scheme to improve the starting estimate for the solution of multigroup neutron diffusion equations by calculating an energy rebalance factor based on a physical principle of neutron balance. The technique was extended considerably to include spatial aspects [Wachspress 1966], where variational techniques were used to apply corrections to the initial trial solution. Developments in finite element techniques undoubtedly influenced the choice of correction function applied to the trial solution [Nakamura 1977].

Usually, the additive or multiplicative coarse mesh corrections were applied at the beginning of the iteration process to obtain an improved initial guess. The incorporation of the coarse mesh correction into the iteration process was a somewhat later development. In contrast, the multigrid approach of Nicolaides [1975], Brandt [1977] and others uses the idea of computing on a series of meshes as its central convergence mechanism. In the multigrid approach, iterative solution commences with the finest grid and iterations are performed until difficulties in convergence emerge; at this stage, a coarser grid is generated through a weighted residual approach and a solution attempted. If convergence on this grid is achieved rapidly, the corrections are made to the solution on the finer grid, otherwise a grid is generated again at a coarser level. The generation of new grids continues until convergence is obtained without difficulty. The iteration is then allowed to proceed by passing up and down various grid systems until convergence is accelerated.

In multigrid approaches, the coarse mesh matrix formulation is often obtained by deriving the reduced matrix from the physical geometry and the partial differential equation. In the present work, however, a return is made to the earlier approach of Wachspress [1966] in which the reduced linear system is derived directly from the matrix formulation.

Multigrid algorithms usually employ the same iterative scheme for all mesh grids, which is consistent with a rederivation of the matrix equation on each mesh. On the other hand, if the coarse mesh matrix is obtained directly from the fine mesh matrix in the manner proposed here, some of the matrix properties which guarantee convergence of relaxation and conjugate gradient schemes may be lost. In such circumstances, the MINI scheme will demonstrate empirically that convergence is a natural choice.

A subroutine implementation of the multigrid approach, MLTGRD, is currently restricted to solving N × N linear systems (see equation 1.1) in conjunction with the original MINI technique developed for finite difference discretisations of neutron diffusion problems. For this class of problem, A is sparse, real, irreducible, symmetric, positive definite (*i.e.* Stieltjes), and block triadiagonal with elements

$$
\begin{aligned}
&a_{ij} = a_{ji} \qquad i=1,2,\ldots,N;\ j=1,2,\ldots,N\ , \\
&a_{ij} \leq 0 \qquad i \neq j, \\
&a_{ii} \geq -\sum_{j \neq i} a_{ij} \quad \text{(with inequality for at least one value of i)}
\end{aligned}
\tag{1.3}
$$

and source elements which are non-negative (and not all zero),

$$b_i \geq 0\ . \tag{1.4}$$

Therefore A^{-1} has all positive elements and consequently all the elements of the unknown vector x are positive.

MINI is designed for such problems, however, the requirement on symmetry may be relaxed provided that at least column diagonal dominance is retained. Consequently, use of MLTGRD is currently restricted to problems for which MINI is appropriate. Although preliminary experimentation confirms the possibility that MINI can be used on problems with negative solutions, the applications of MLTGRD discussed in this report satisfy matrix properties 1.3 and 1.4.

For MLTGRD, the matrix A corresponds to that derived from a five-point finite difference approximation to the 2D elliptic problem, where L is a Laplacian operator.

$$L\ u(x,y) = f(x,y)\ .$$

The five-point finite difference stencil at the point (i,j) is shown in figure 1. The only non-zero, off-diagonal terms for the coefficient matrix A, derived from the stencil for the operator L over a rectangular ($N_x \times N_y$) grid, are

$a_{m\ m+N_x}$ = coupling (1) from (i,j) to (i,j+1),

$a_{m\ m-N_x}$ = coupling (3) from (i,j) to (i,j-1),

$a_{m\ m+1}$ = coupling (4) from (i,j) to (i+1,j),

$a_{m\ m-1}$ = coupling (2) from (i,j) to (i-1,j), and

$a_{m\ m}$ = diagonal term at (i,j). (1.5)

In the developments that follow, discrete forms of the unknown function, u(x,y), and the given driving function, f(x,y), will be taken to be the vectors x and b, respectively.

2. MULTIGRID APPROACH

One form of multigrid approach for the solution of a linear system of equations A x = b of order N is shown in figure 2. The algorithm is appropriate to a five-point discretisation of a partial differential equation over a rectangular grid of size $N_x \times N_y = N$. Solution commences with the MINI iterative scheme on the finest grid of order N_k, where k denotes the level of the grid.

For the finest grid, k = K and $N_K = N$, where

$$K = min\left\{ \left\lfloor \left(log\,(N_x) \right)_{NDIVX} \right\rfloor + 1 \left\lfloor \left(log\,(N_y) \right)_{NDIVY} \right\rfloor + 1 \right\}, \tag{2.2}$$

lx = floor of x, *i.e.* nearest integer not greater than x, and NDIVX and NDIVY are user-specified (integer) dividing factors, usually given as 2. For the coarsest grid, k = 1.

The estimate of x_k at any level of grid k is $x_k^{(n_k)}$, where n_k is the iteration count for level k. A minimum number of iterations is performed at each level before a test for difficulty in convergence is applied. Should convergence be occurring but be slow, *i.e.*

$$e_k^{(n_k+1)} / e_k^{(n_k)} > \delta\ , \tag{2.3}$$

and

$$e_k^{(n_k+1)} < e_k^{(n_k)}\ ,$$

where δ is typically 0.8, and instead of the vector norm 1.2 we use

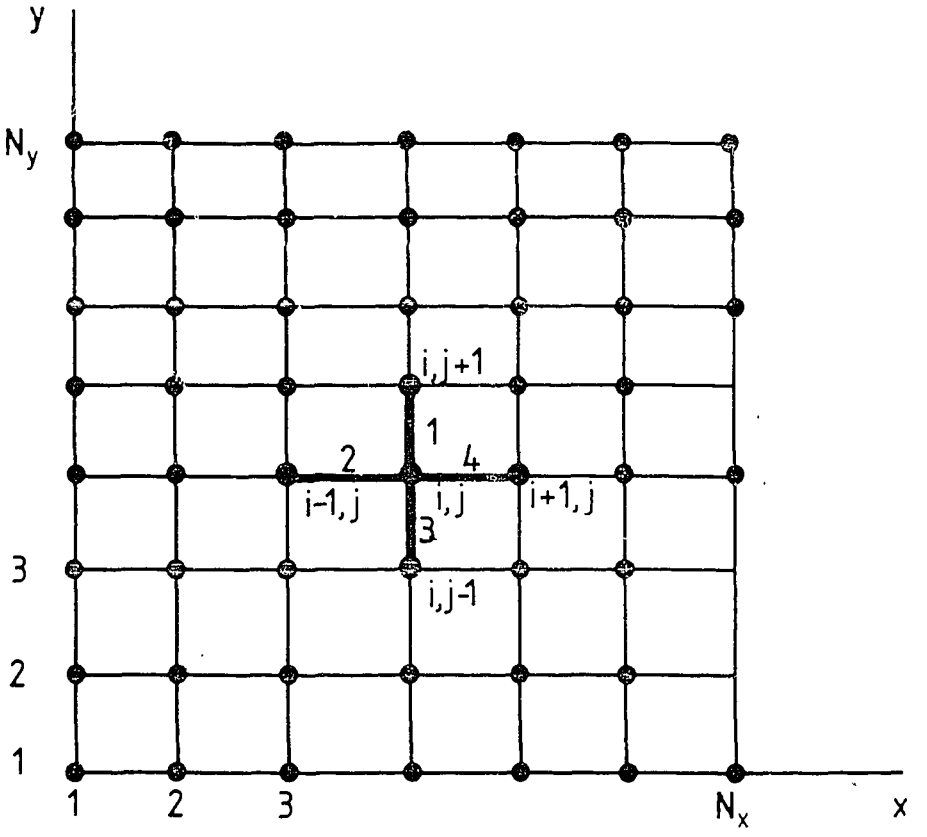

Figure 1 Five point finite difference stencil

$$e_k^{(n_k)} = \sum_{m=1}^{N_k} \left\{ \frac{\left| {}_m x_k^{(n_k)} - {}_m x_k^{(n_k-1)} \right|}{\left| {}_m x_k^{(n_k-1)} \right|} / N_k \right\}, \tag{2.4}$$

and ${}_m x_k^{(n_k)}$ denotes the mth elements of the vector $x_k^{(n_k)}$, grid reduction is applied. Equation 2.4 could be replaced by an absolute error criterion should non-positive solutions be admitted.

The matrix reduction used in the basic algorithm (figure 2) corresponds to combining the grid points along the (x,y) geometry (figure 1) in groups of NDIVX and NDIVY corresponding to the (k+1)th level systems, $A_{k+1}\, x_{k+1} = b_{k+1}$. The reduced system

$$A_k x_k = b_k$$

is produced by the coarse mesh method

$$A_k \longleftarrow CM\ (A_{k+1},\ x_{k+1}^{(n_{k+1})})$$

$$b_k \longleftarrow CM\ (b_{k+1}) \quad ,$$

described in section 3. The reduced matrix A_k is of the order $N_k = N_{xk}\, N_{yk}$, where

$$N_{xk} = \lfloor (N_x/(NDIVX)^{K-k}) \text{ and } N_{yk} = \lfloor (N_y/(NDIVY)^{K-k}) \tag{2.5}$$

and the notation is as for equation 2.2. For example, say,

$N_x = 17$, $N_y = 15$, NDIVX = 2 and NDIVY = 2, then K = 4 and $N_4 = 17 \times 15$,

$N_3 = 8 \times 7$, $N_2 = 4 \times 3$ and $N_1 = 2 \times 1$.

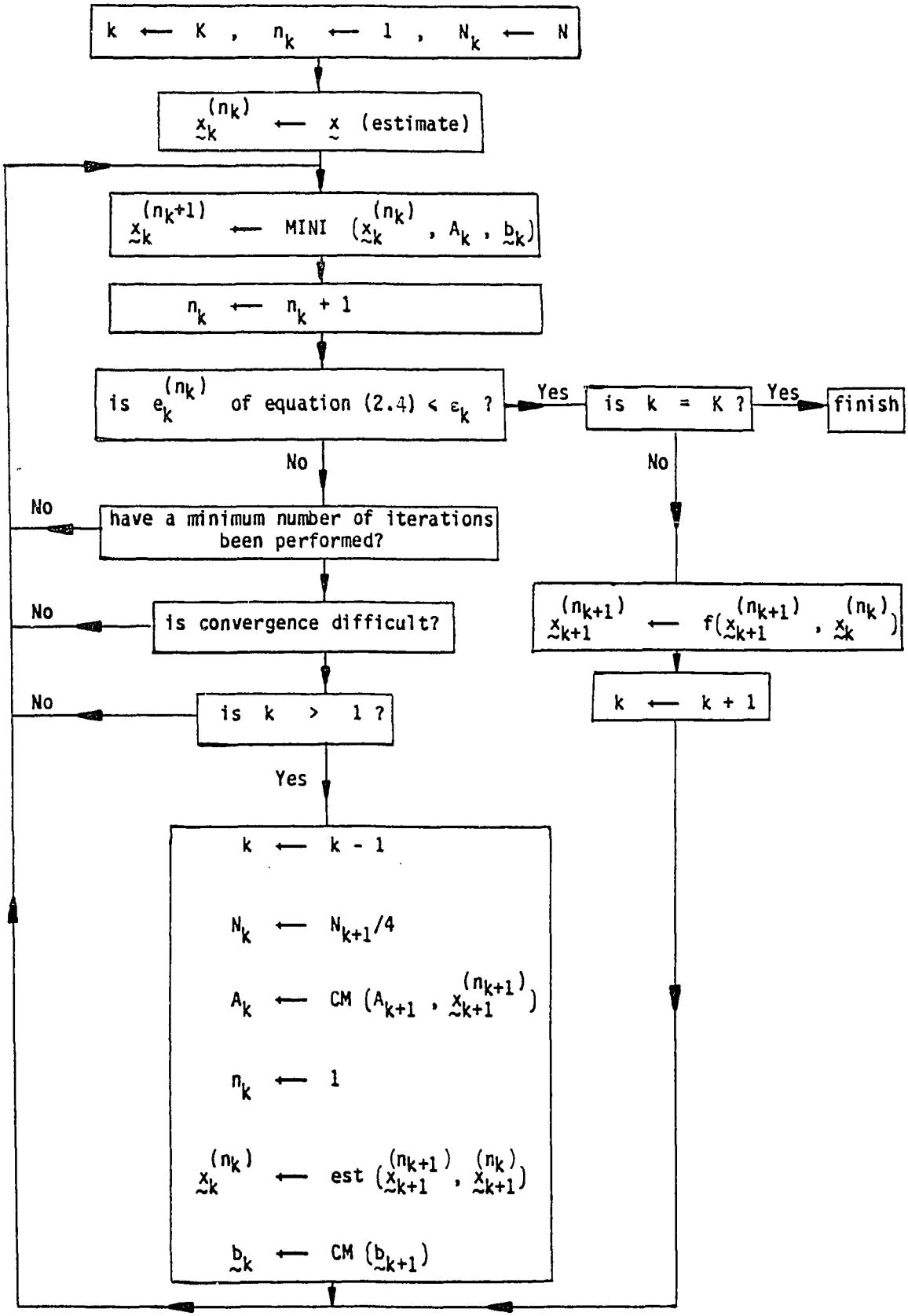

Figure 2 Basic multigrid algorithm with NDIVX = 2 and NDIVY = 2

In the flow diagram (figure 2) equation 2.5 is taken as the simplified form, for ease of presentation:

$$N_k = N_{k+1/4} \tag{2.6}$$

as it would be if N_x and N_y are both two raised to some power with NDIVX = 2 and NDIVY = 2.

In the multigrid algorithm, x_k is not a coarse grid approximation to x_{k+1} ; rather it is used to correct the n^{th}_{k+1} iterate $x_k^{(n_{k+1})}$ through a multiplicative relationship

$$x_{k+1}^{(n_{k+1})} \longleftarrow f\ (x_{k+1}^{(n_{k+1})}\ ,\ x_k^{(n_k)}\ )\ .$$

For all grid levels k < K, the initial estimate $x_{k+1}^{(1)}$ is given by

$$x_{k+1}^{(1)} = est\ (x_k^{n_k+1}\ ,\ x_k^{n_k}\ )\ .$$

The function *est* uses averaged relative differences

$$_j x_{k+1}^{(1)} = 1 + [\ \sum_{m-1}^{N_k}\ (\ _m x_k^{n_k+1} - {}_m x_k^{n_k}\ )\ /\ {}_m x_k^{n_k}\ /N_k\ ]\ ,\ j=1, 2, \cdots, N_{k+1}\ .$$

3. COARSE MESH REBALANCING

In MLTGRD, a form of multiplicative disjoint regions rebalancing is used to improve the solution (*i.e.* remove) low frequency error components from grids at all levels. The n^{th}_{k+1} estimate of the solution to the matrix problem

$$A_{k+1}\ x_{k+1} = b_{k+1}\ , \tag{3.1}$$

for the grid of level $k+1$, and order N_{k+1} is denoted by $x_{k+1}^{(n_{k+1})}$. From system 3.1 a reduced system of size N_k,

$$A_k x_k = b_k\ , \tag{3.2}$$

is produced by a set of selection operators $P_{k+1} = {}_m P_{k+1}$ (m = 1,2,...,N_k) such that

$$A_k \longleftarrow CM\ (A_{k+1},\ P_{k+1},\ x_{k+1}^{(n_{k+1})}\ )\ ,\ \text{and}$$

$$b_k \longleftarrow CM\ (P_{k+1},\ b_{k+1})\ .$$

From solution of the reduced system 3.2, a new estimate $x_{k+1}^{(n_{k+1})}$ for x_{k+1} of 3.1 ,

$$x_{k+1}^{(n_{k+1})} = f\ (x_{k+1}^{(n_{k+1})}\ ,\ P_{k+1}\ ,\ c_{k+1}\ )\ , \tag{3.3}$$

is obtained by determining the N_k elements of c_{k+1}, yet to be defined, so as to minimise the residual:

$$r_{k+1} = A_{k+1}\ x_{k+1}^{(n_{k+1})} - b_{k+1} \tag{3.4}$$

after appropriate weighing vectors are selected. The P_{k+1} operators form disjunctive partitioning over the fine grid or any subsequent idealisation of it. The new estimate or corrected vector of (3.3) is given by

$$x_{k+1}^{(n_{k+1})} \longleftarrow \sum_{m=1}^{N_k}\ (_m c_{k+1}\ {}_m P_{k+1}\ )\ x_{k+1}^{(n_{k+1})}\ , \tag{3.5}$$

where the $N_k \times N_k$ diagonal matrices P_{k+1} satisfy

(i) diag. $[_m P_{k+1}]$ $\begin{cases} = 1 & \text{for all points in the } m^{th} \\ & \text{coarse mesh partition at level } k+1 \\ = 0 & \text{otherwise, and} \end{cases}$

(ii) $\sum_{m=1}^{N_k}\ {}_m P_{k+1} = I\ .$

The N_k coarse mesh partitions appropriate to the grid of level $k+1$ are defined in terms of the original two-dimensional geometry or some lower level abstraction of it. A typical partitioning is shown in figure 3.

The $(N_{xk}-1)(N_{yk}-1)$ partitions are formed by effectively halving the fine mesh points in each direction. The users of MLTGRD can control the number of mesh points in each direction collapsed onto the coarse grid. When an exact subdivision is impossible, the right hand side (or top) of the equation is treated as shown in figure 3. The unknown term $_m c_k$. (m = 1 ,2, ...,N_k) of equation 3.5 serve as disjunctive multiplicative correction factors over each coarse mesh partition. They are determined by the weighted residual method

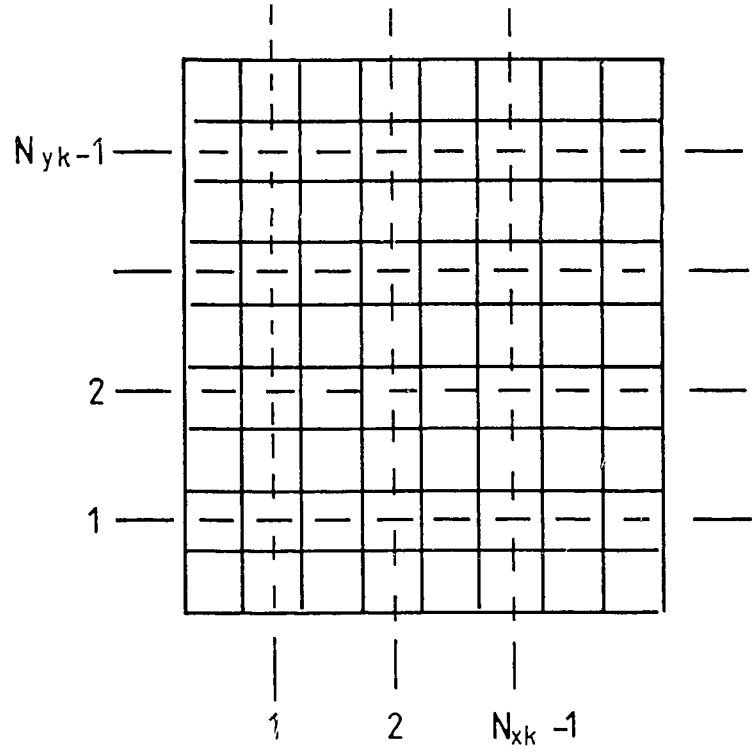

Figure 3 Coarse grid for 2 × 2 disjunctive partitioning

$$< {}_m w_{k+1} , \; r_{k+1} > \; = 0 \quad (m = 1, 2, \ldots, N_k) , \tag{3.6}$$

where the N_k disjunctive weighting vectors are

$${}_m w_{k+1} \; = \; {}_m P_{k+1} \quad 1 \; ,$$

$< \; , \; >$ denotes the inner product of two vectors, and 1 is a vector of length N_{k+1} with unit components. Substitution of equations 3.4 and 3.5 in 3.6 leads to the following linear system of order N_k:

$$\sum_{m'=1}^{N_k} < {}_m w_{k+1}, \; A_{k+1} \; {}_{m'} P_{k+1} \; x_{k+1}^{(n_{k+1})} >_{m'} c_{k+1} = < {}_m w_{k+1}, \; b_{k+1} > \quad (m = 1, 2, \ldots, N_k) \tag{3.7}$$

or, in matrix rotation,

$$A_k \;\; c_{k+1} \; = \; \hat{b}_{k+1} \, , \tag{3.8}$$

where $\hat{b}_{k+1}$ denotes the right hand side of equation 3.7, and A_k is the reduced matrix arising from the collection of coefficients of the unknowns ${}_{m'} c_{k+1}$ on the left hand side of the same equation.

4. MATRIX PROPERTIES

It is desirable that the multigrid approach use the same iterative scheme (and code) at all grid levels, with the possible exception (although not in this case) of the coarsest grid where a direct technique may be more appropriate. Consequently, it is important that the coarse mesh generation algorithm retain sufficient properties of the original matrix system (1.3 and 1.4) to ensure convergence.

For the initial fine mesh system 1.1, only symmetry and now diagonal dominance are lost for the first reduced matrix A_{K-1} [Barry 1982]. MINI and Gauss-Seidel techniques are appropriate at this level; however, those which rely on symmetry may fail, or at least optimal extrapolation parameters may be unobtainable.

Further coarse mesh reduction of A_{K-1} retains column diagonal dominance as indicated by the summation over an arbitrary column m of A_{k+1}. From equation 3.7, summing over the rows of A_k gives

$$\sum_{m=1}^{N_k} < {}_m w_{k+1},\ A_{k+1}\ {}_m P_{k+1}\ x_{k+1}^{(n_k+1)} > \ ,$$

$$= < \sum_{m=1}^{N_k} diag\left\{ {}_m P_{k+1} \right\}\ ,\ A_{k+1}\ {}_m P_{k+1}\ x_{k+1}^{(n_k+1)} > \ ,$$

$$= < 1\ ,\ A_{k+1}\ {}_m P_{k+1}\ x_{k+1}^{(n_k+1)} >$$

$$= \sum_{s=1}^{N_k} \sum_{t \in D_m} a_{st}\ {}_t x_{k+1}$$

(dropping extraneous notation and a_{st} denotes elements of A_{k+1}) ,

$$= \sum_{t \in D_m} {}_t x_{k+1} \sum_{s=1}^{N_k} a_{st} \quad (\textit{i.e.}\ \text{a weighted } \Sigma \text{ over the rows of } A_{k+1}),$$

where D_m denotes the set of all indices corresponding to non-zero diagonal components of ${}_m P_{k+1}$. Column diagonal dominance of A_k follows immediately from column diagonal dominance of A_{k+1} and $x_{k+1} > 0$. Consequently, MINI is appropriate for all levels of grid refinement.

5. PERFORMANCE OF MULTIGRID METHOD

MLTGRD has been tested on several realistic problems. The version tested conforms to the basic algorithms of figure 2. The test for convergence, however, was relaxed for each coarse mesh level. At the finest level, $\varepsilon = 0.0001$ was selected, but for each coarse grid such a stringent test is inappropriate because the quantity being computed is merely a multiplicative correction to the higher level. Consequently, for coarse grids ($k < K$), $\varepsilon_{k-1} = 0.1 \times e_k$, where e_k is the average relative error in x as given in equation 2.4.

The generation of each coarse grid and the subsequent multiplicative correction is not without cost. The numbers of floating point instructions for both phases are given in table 1, where they are compared with the number necessary for the iterative schemes MINI, SLOR and ICCG. The cost of the rebalance operation is approximately the same as a Gauss-Seidel iteration but significantly less than all other iterative schemes.

TABLE 1
COST OF COARSE MESH GENERATION AND CORRECTION PLUS OVERHEADS FOR SEVERAL ITERATIVE METHODS

Method	Floating Point Operations	
	* ÷	+−
Coarse mesh	6N	5N
Gauss-Seidel	5N	4N
MINI	12N	7N
SLOR	10N	6N
ICCG	37N	32N

The results for six problems solved by the MLTGRD are reported.

Problem 1

$$-\nabla^2 u(x,y) = \sin x \sin y$$

for $0 \leq x \leq \pi$, $0 \leq y \leq \pi$, with $u(x,y) = 0$ on all boundaries. An estimate of $u_o(x,y) = 1$ was taken and grids of equal increments were selected.

Problems 2,3,4

These problems are taken from Wachspress [1966]:

$$-\nabla . D \nabla u(x,y) = 1$$

for a square region of side length 20 cm, as shown in figure 4. The different problems are identified below:

Problem	D_1	D_2	D_3	D_4
2	1	2	4	8
3	1	5	25	125
4	1	10	100	1000

An initial estimate of $u_o(x,y) = 1$ was used.

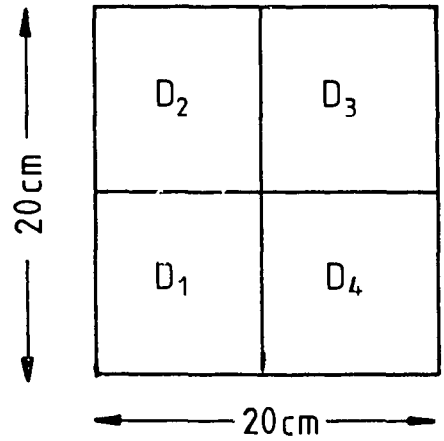

Figure 4 Geometry for problems 2, 3, 4

Problem 5

Problem 5 is the same as problem 3, except uneven mesh intervals of 0.5, 0.5, 0.5, 0.01, 0.49, 1.0, 2.0, 1.0, 3.0, and 1.0 are used repeatedly until the whole mesh is filled.

Problem 6

Problem 6 is based on a two-dimensional study with a control insulating region [Wachspress 1966], as given in figure 5. Even mesh intervals are used. The void region makes resolution of this problem extremely difficult. An initial estimate of $u_o(x,y) = 1$ was used.

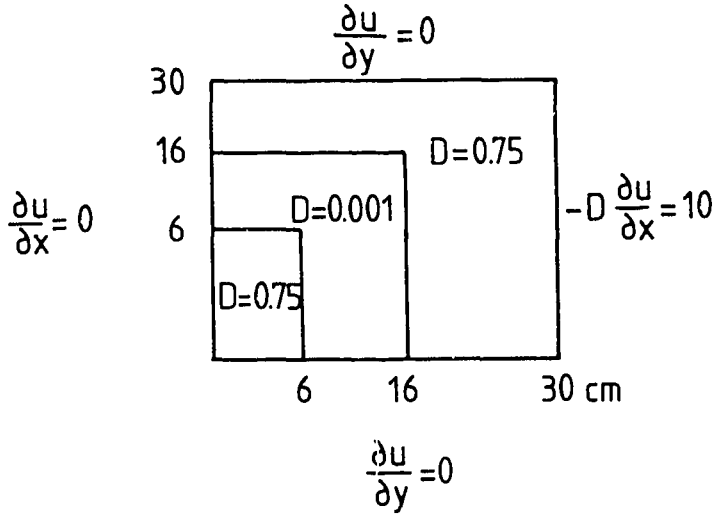

Figure 5 Problem 6

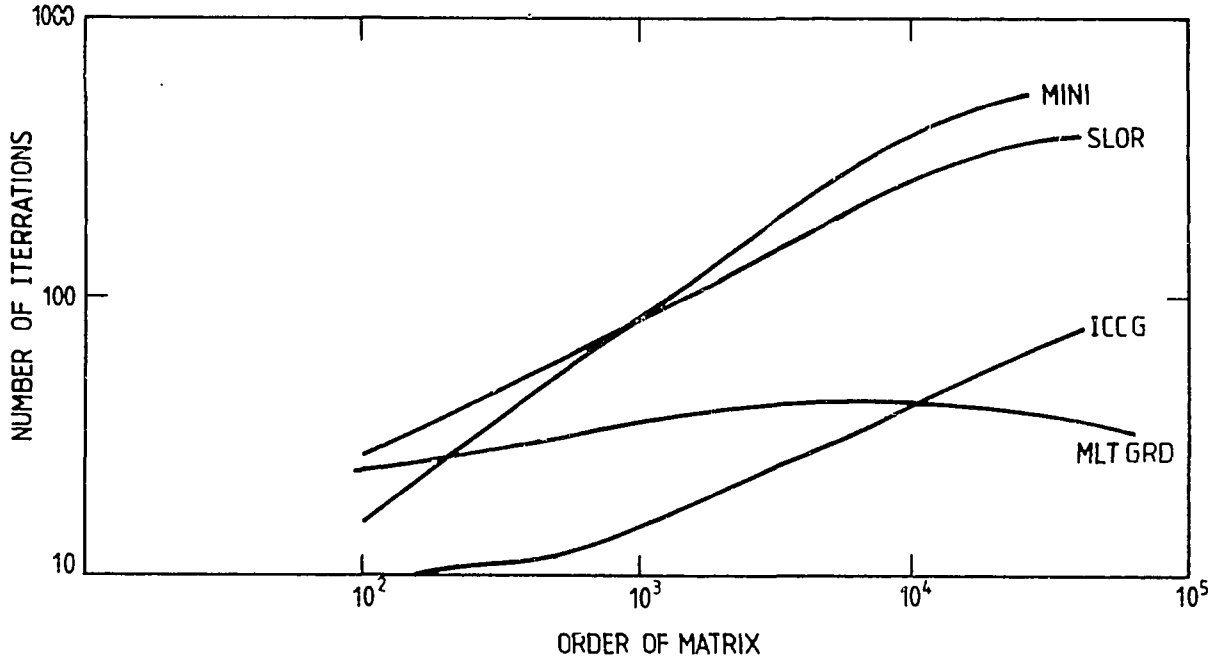

Figure 6 Number of iterations as a function of the order of the matrix for problem 1

The number of iterations necessary for convergence with MLTGRD, MINI, SLOR and ICCG for a typical system (problem 1) are shown in figure 6 as a function of the number of unknowns on a log-log scale. For MLTGRD results, a quantity known as the number of equivalent fine mesh iterations is computed for the comparison. This measures the total work required for the multigrid systems in terms of equivalent numbers of fine mesh iterations. It is determined by discounting all the coarse mesh iterations to equivalent fine mesh iterations and counting each coarse mesh generation and subsequent rebalance as an equivalent iteration on the finer grid. As the order of the matrix increases, MINI, SLOR and ICCG require more iterations. Because the computational times became excessive, further calculation was terminated. The multigrid system starts out a little more expensively but does not exhibit the fast growth rate and quickly levels off, appearing to be much less dependent upon the number of unknowns over the range of problems so far studied. The multigrid greatly expands the size of problem which may be undertaken, however, the log scale on the graph disguises its advantage.

In terms of number of iterations, MLTGRD starts to perform better than ICCG when the order of the matrix is about 10^4. On performance times, the advantage of MLTGRD becomes apparent for matrices of much lower order.

The effect of coarse mesh rebalance on the error equation 2.4 is two-fold. First, it brings about a drop in the error on the next iteration. Second, and perhaps even more importantly, the errors on subsequent iterations drop away more rapidly after it is applied. This is shown for the error on the finest level grid (250×250) of problem 2 in figure 7. Sometimes application of rebalance causes the error to grow but this is more than compensated for by a steep decrease in the error reduction curves as the iterations continue. Analysis of the effect of the rebalance must be investigated over more than one iteration. In this respect, it is like MINI where, in a proof of convergence, error over more than one iteration would have to be considered.

Results for the six problems are presented in tables 2-7. Missing entries indicate that results were not obtained (or that a solution was not even attempted) because of timing considerations. The number of iterations required and central processing unit (CPU) times are shown. The times reported are from an IBM 3033S with double precision arithmetic and code compiled with the IBM VS FORTRAN compiler at the highest optimisation (3). The times given should serve only as a guide because the multigrid code is new and has not been subjected to the same rigorous level of coding refinement as the code for the three iterative methods. Performance is reported of the following methods:

(i) MLTGRD with at least three fine mesh iterations before rebalance;
(ii) MLTGRD with rebalance before iterations commence;
(iii) MINI;
(iv) coarse mesh rebalance before first iterations only;

(v) coarse mesh rebalancing repeatedly applied at the finest level only when convergence becomes slow;
(vi) ICCG ; and
(vii) SLOR.

The result for multigrid algorithms are given in numbers of equivalent fine mesh iterations.

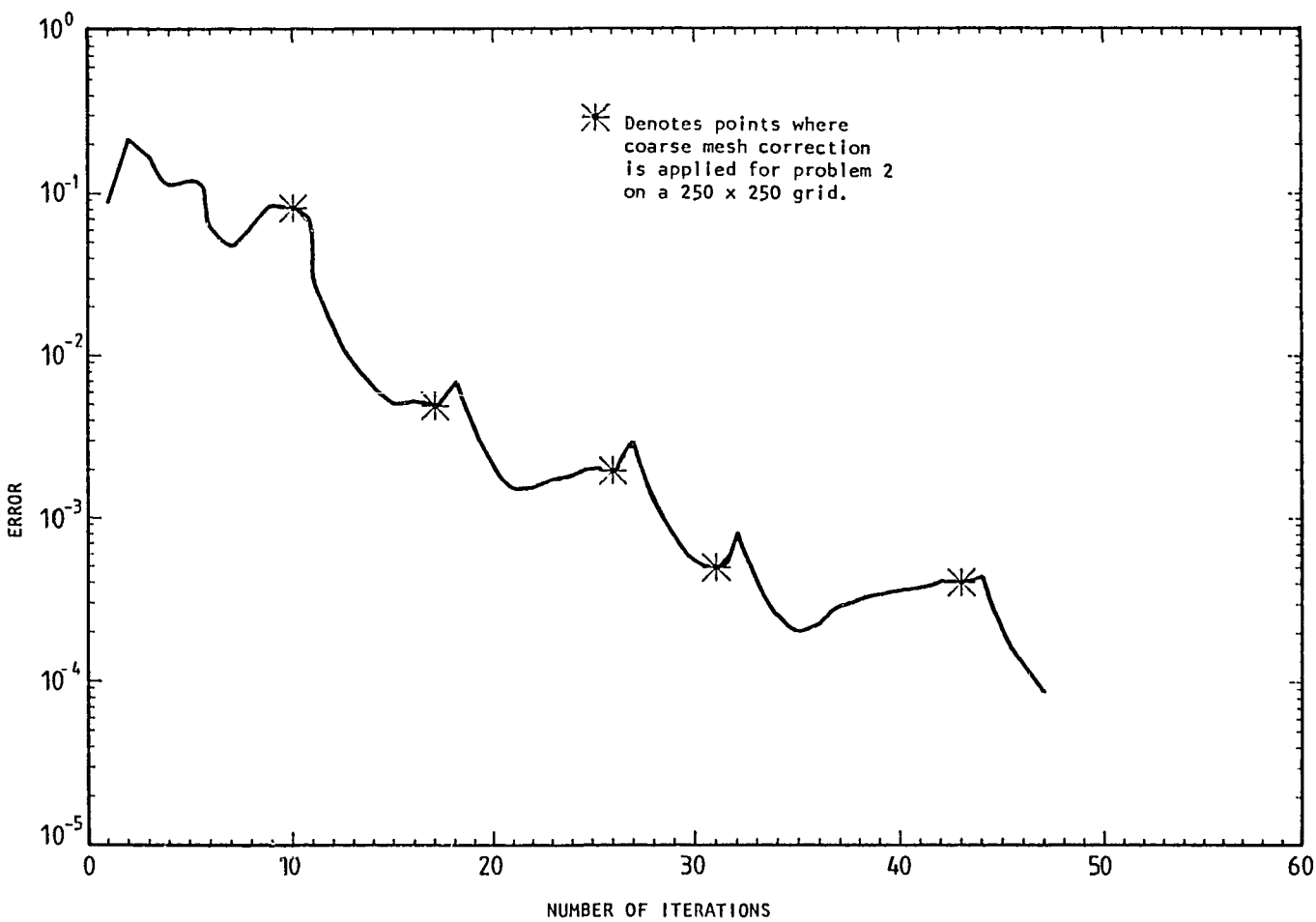

Figure 7 Reduction of error with multigrid refinement for finest mesh

TABLE 2
RESULTS FOR PROBLEM 1
SHOWING NUMBER OF ITERATIONS AND TIME (min)

Mesh Size	Multigrid	Multigrid Starting at First Iteration	Mini	Coarse Mesh Rebalance	Multiple Coarse Mesh Rebalance	ICCG	SLOR
10 × 10	19 (0.005)	24 (0.006)	17 (0.002)	18 (0.004)	19 (0.004)	9 (0.002)	28 (0.002)
20 × 20	32 (0.02)	30 (0.02)	40 (0.02)	41 (0.02)	32 (0.02)	13 (0.02)	43 (0.02)
40 × 40	35 (0.09)	43 (0.11)	118 (0.25)	119 (0.25)	52 (0.12)	22 (0.09)	97 (0.16)
80 × 80	39 (0.41)	47 (0.48)	332 (2.8)	345 (2.8)	69 (0.60)	40 (0.65)	244 (1.57)
100 × 100	42 (0.72)	49 (0.81)	455 (6.0)	478 (6.2)	80 (1.1)	44 (1.1)	325 (3.3)
120 × 120	46 (1.1)	46 (1.1)	557 (11.3)	612 (11.5)	73 (1.44)	52 (1.9)	412 (6.3)
160 × 160	46 (2.0)	45 (1.9)			113 (3.9)	68 (4.4)	
200 × 200	51 (3.5)	42 (3.0)			154 (8.4)	84 (8.6)	
220 × 220	54 (4.5)	44 (3.8)			168 (11.1)	92 (11.4)	
250 × 250	60 (6.4)	37 (4.3)				104 (16.4)	

TABLE 3
RESULTS FOR PROBLEM 2
SHOWING NUMBER OF ITERATIONS AND TIME (min)

Mesh Size	Multigrid	Multigrid Starting at First Iteration	Mini	Coarse Mesh Rebalance	Multiple Coarse Mesh Rebalance	ICCG	SLOR
10 × 10	18 (0.005)	20 (0.004)	16 (0.002)	18 (0.005)	18 (0.004)	8 (0.002)	21 (0.002)
20 × 20	32 (0.02)	29 (0.02)	39 (0.02)	32 (0.02)	33 (0.02)	14 (0.05)	40 (0.02)
40 × 40	45 (0.11)	42 (0.10)	100 (0.20)	107 (0.22)	46 (0.11)	26 (0.11)	75 (0.12)
80 × 80	50 (0.52)	44 (0.46)	176 (1.4)	297 (2.4)	68 (0.62)	47 (0.97)	126 (0.82)
100 × 100	56 (0.92)	45 (0.74)	335 (4.4)	412 (5.3)	61 (0.87)	57 (1.4)	136 (1.42)
120 × 120	52 (1.2)	48 (1.1)	300 (5.6)	531 (10.0)	85 (1.7)	68 (2.5)	137 (2.07)
160 × 160	54 (2.3)	50 (2.1)	413 (14.1)		155 (5.4)	88 (5.7)	211 (5.78)
200 × 200	51 (3.6)	52 (3.5)			204 (11.1)	109 (11.0)	
220 × 220	56 (4.7)	51 (4.2)				119 (14.5)	
250 × 250	74 (7.9)	47 (5.1)					

TABLE 4
RESULTS FOR PROBLEM 3
SHOWING NUMBER OF ITERATIONS AND TIME (min)

Mesh Size	Multigrid	Multigrid Starting at First Iteration	Mini	Coarse Mesh Rebalance	Multiple Coarse Mesh Rebalance	ICCG	SLOR
10 × 10	17 (0.004)	20 (0.005)	17 (0.002)	16 (0.004)	17 (0.004)	10 (0.005)	28 (0.003)
20 × 20	28 (0.02)	28 (0.02)	35 (0.02)	31 (0.02)	31 (0.02)	16 (0.02)	53 (0.02)
40 × 40	47 (0.12)	41 (0.11)	107 (0.21)	85 (0.18)	47 (0.11)	29 (0.12)	99 (0.16)
80 × 80	48 (0.49)	44 (0.46)	317 (2.6)	266 (2.2)	67 (0.61)	55 (0.90)	164 (1.1)
100 × 100	53 (0.89)	44 (0.74)	437 (5.6)	364 (4.7)	89 (1.23)	67 (1.70)	193 (1.9)
120 × 120	50 (1.2)	45 (1.1)	584 (10.9)		98 (1.94)	77 (2.8)	201 (3.0)
160 × 160	54 (2.4)	56 (2.4)			151 (5.13)	101 (6.5)	223 (6.1)
200 × 200	57 (3.8)	53 (3.5)			187 (10.90)	124 (12.4)	
220 × 220	60 (4.8)	56 (4.5)					
250 × 250	60 (6.3)	54 (5.8)					

TABLE 5
RESULTS FOR PROBLEM 4 SHOWING
NUMBER OF ITERATIONS AND TIME (min)

Mesh Size	Multigrid	Multigrid Starting at First Iteration	Mini	Coarse Mesh Rebalance	Multiple Coarse Mesh Rebalance	ICCG	SLOR
10 × 10	19 (0.005)	19 (0.005)	16 (0.002)	16 (0.004)	19 (0.005)	10 (0.003)	30 (0.003)
20 × 20	28 (0.02)	28 (0.02)	34 (0.02)	30 (0.02)	29 (0.02)	18 (0.02)	49 (0.02)
40 × 40	41 (0.11)	41 (0.115)	94 (0.18)	83 (0.17)	44 (0.10)	33 (0.13)	103 (0.16)
80 × 80	51 (0.54)	42 (0.47)	380 (3.1)	238 (1.9)	76 (0.68)	58 (0.94)	238 (1.54)
100 × 100	57 (0.94)	48 (0.84)	577 (7.5)	335 (4.4)	94 (1.3)	70 (1.8)	268 (2.8)
120 × 120	53 (1.2)	46 (1.1)		454 (8.5)	120 (2.3)	84 (3.1)	277 (4.2)
160 × 160	55 (2.3)	49 (2.1)			199 (6.9)	110 (7.1)	
200 × 200	60 (4.0)	52 (3.6)				137 (13.6)	
220 × 220	55 (4.4)	54 (4.3)					
250 × 250	53 (5.6)	62 (6.5)					

TABLE 6
RESULTS FOR PROBLEM 5 SHOWING
NUMBER OF ITERATIONS AND TIME (min)

Mesh Size	Multigrid	Multigrid Starting at First Iteration	Mini	Coarse Mesh Rebalance	Multiple Coarse Mesh Rebalancing	ICCG	SLOR
20 × 20	41 (0.03)	45 (0.03)	59 (0.03)	65 (0.04)	45 (0.03)	22 (0.02)	54 (0.02)
40 × 40	58 (0.15)	50 (0.14)	186 (0.38)	189 (0.39)	73 (0.18)	43 (0.18)	168 (0.27)
80 × 80	64 (0.66)	60 (0.64)	663 (5.1)	567 (4.6)	149 (1.27)	76 (1.26)	359 (2.37)
100 × 100	88 (1.3)	59 (0.96)	942 (12.1)	822 (10.5)	216 (2.9)	93 (2.3)	234 (2.4)
120 × 120	68 (1.6)	67 (1.6)			277 (5.2)	113 (4.1)	
160 × 160	66 (2.7)	56 (2.4)				139 (8.9)	
200 × 200	72 (4.9)	62 (4.2)				168 (16.7)	
220 × 220	75 (5.9)	51 (4.2)					
250 × 250	71 (6.8)	52 (5.0)					

TABLE 7
RESULTS FOR PROBLEM 6
SHOWING NUMBER OF ITERATIONS AND TIME (min)

Mesh Size	Multigrid	Multigrid Starting at First Iteration	Mini	Coarse Mesh Rebalance	Multiple Coarse Mesh Rebalancing	ICCG	SLOR
10 × 10	27 (0.005)	39 (0.008)	24 (0.003)	34 (0.007)	27 (0.005)	15 (0.003)	215 (0.02)
20 × 20	37 (0.025)	34 (0.023)	354 (0.09)	73 (0.04)	49 (0.03)	26 (0.03)	698 (0.27)
40 × 40	64 (0.16)	51 (0.12)	2716 (6.0)	1265 (2.6)	229 (0.48)	47 (0.20)	2473 (3.9)
80 × 80	85 (0.85)	43 (0.45)		680 (5.5)	958 (7.8)	90 (1.44)	1838 (11.8)
100 × 100	76 (1.2)	47 (0.77)				111 (2.8)	
120 × 120	97 (2.2)	57 (1.4)				133 (4.8)	
160 × 160	74 (3.2)	54 (2.3)				173 (11.1)	
200 × 200	95 (6.2)	48 (3.2)					
220 × 200	118 (8.9)	65 (5.2)					
250 × 250	96 (9.6)	65 (6.7)					

6. SUMMARY OF RESULTS

The results show that the number of iterations required with MINI, SLOR and ICCG grows as the order of the matrix increases. In this respect, the ICCG technique performs much better than the other two schemes for the class of problems considered. The case for selecting ICCG as the best of three iterative algorithms is not as strong as it might appear. For medium to large three-dimensional reactor physics calculations [Barry 1982], MINI is the recommended choice for the code POW3D [Barry and Pollard 1986].

For many reactor physics calculations, some form of coarse mesh rebalance scheme is desirable to accelerate convergence of the iterative scheme. For most problems, simple coarse mesh rebalance for MINI (column 4 of tables 2-7) offers only a limited advantage over MINI (other than problem 1 where it causes a deterioration). Multiple coarse mesh rebalance (column 5 of tables 2-7) has more to offer and supports the contention that frequent application is worthwhile.

Both forms of multigrid algorithms (columns 1 and 2 of tables 2-7) are very effective as the order increases on all problems. For problems with voids (such as problem 6) or uneven grids, this is not the case. When compared on CPU time instead of number of equivalent iterations, the same pattern of relative efficiency is observed.

For very small mesh problems, there is nothing to suggest that the multigrid scheme offers any advantage, however, should it be the default option in any code, the overheads would appear reasonable. For larger scale problems the advantages are enormous.

The multigrid approach succeeds so well because by improving the solution estimate, much of the computational effort is transferred from the fine to the coarse grid. In this transfer, the arithmetic and storage overheads are reduced by a factor of four (or thereabouts) for every new level. For problem 5 with a 240 × 240 mesh the number of iterations on the finest grid is 40 per cent of the total (table 8).

For the finest grid on an IBM machine, about 3700 kbytes of memory is required for the data. On currently available computer equipment at medium size scientific installations, this amount of real storage would not be available during normal shifts. Consequently, page swapping would be necessary in the virtual computing environment. By transferring the large number of iterations to the lower level grids, smaller regions are required and all data may be held in real memory, even during prime shifts. It is expected that even the 920 kbytes required by the first multigrid level would remain in real memory for the duration of the iterative procedure on the Lucas Heights main computer. For all the more reduced matrix levels, better use of cache memory is also possible with multigrids.

Numerical iterative schemes for large systems of linear equations are notoriously bad users of virtual memory. Most computer algorithms function so that the most recently used page is retained in real memory, and the earliest used page is dispatched. For the iterative schemes, as the line (or block) progresses across the grid, the earliest page used is the one which is required next. Consequently, many page exceptions are generated for each iteration pass when storage is in heavy demand by others. By reducing the size of the problem, the multigrid algorithm offers this advantage of secondary performance as well as the mathematical improvement that allows removal of low frequency error components in the solution estimate.

TABLE 8
REDISTRIBUTION OF COMPUTATIONAL EFFORT
FOR PROBLEM 5 WITH 240 × 240 GRID

Grid Level	Mesh Size	Storage (kbytes)	Number of Iterations	Number of Times Level Initiated
8	240 × 240	3700	21	1
7	120 × 120	920	68	4
6	60 × 60	240	85	9
5	30 × 30	57	161	14
4	15 × 15	14	183	20
3	7 × 7	3	417	19
2	3 × 3	0.5	523	33
1	1 × 1	0.1	56	28
		Total = 4935	Equivalent Iterations = 52	

7. CONCLUSIONS

The multigrid approach used here differs from the conventional method and is an extremely powerful tool for solving large systems of linear equations. On CPU time, its performance may best be described as extraordinarily powerful, when compared with single level iterative schemes.

Like all powerful tools, MLTGRD should not be used indiscriminantly in a virtual computing environment. Even though large architecture machines such as the IBM XA mainframe computers will allow problems of any conceivable size to be attempted, the user must consider the real storage limitations. An inappropriate orientation of the problem may well cause the computer to 'thrash' the pages. Sufficient real storage must be available at least to hold all data associated with the line along which MINI solves.

8. REFERENCES

Barry, J.M. [1982] - PhD Thesis, Wollongong University.

Barry, J.M., Jenkinson, J.H., Pollard,J.P. [1983] - A discrete Fourier analysis of coarse mesh rebalancing and some associated iterative methods. *J. Aust. Math. Soc., Ser. B- Appl. Math.*, 25:190-216.

Barry, J.M., Pollard, J.P. [1977] - Method of implicit non-stationary iteration for solving neutron diffusion linear problems. *Ann. Nucl. Energy*, 4:485-493.

Barry, J.M., Pollard, J.P. [1978] - Application of the method of implicit non-stationary iteration (MINI) to 3D neutron diffusion problems. *Ann. Nucl. Energy*, 6:121-131.

Barry J.M., Pollard, J.P. [1986]- AUS diffusion neutronics Model POW3D : a Mathematical description. AAEC/E (in press).

Brandt, A. [1977] - Multi level adaptive solutions to boundary-value problems. *Math. of Comput.*, 31(138)333-390.

Nakamura, S. [1977] - Computational Methods in Engineering and Science with Applications to Fluid Dynamics and Nuclear Systems. John Wiley, New York.

Nicolaides, R.A. [1975] - On multiple grid and related techniques for solving discrete elliptic systems. *J. Comput. Phys.*, 19:418-431.

Roe, G. M. [1954] - Adaptation of multigroup methods to cylindrical geometries. KAPL950.

Wachspress, E.L. [1966] - Iterative Solution of Elliptic Systems and Applications to the Neutron Diffusion Equations of Reactor Physics. Prentice-Hall, Englewood Cliffs, New Jersey.

APPENDIX A
CALLING MLTGRD

MLTGRD is invoked by

CALL MLTGRD(CF,B,X,R,ISUB,ITCNT,L1,L2,NX,NY,ICFMSH,LNX,LNY,IMESH,WPMINI,WQMINI,WRMINI,ITMIN, JMETH,FACTOR,ACCFI,ACC,NDIVX,NDIVY,RETAC,LINE)

MLTGRD is available only in a double precision form for 32 bit computers. Our experience indicates that, for very large systems of linear equations, this is desirable if not essential even for the coefficient matrix.

CF is used to hold the non-zero elements of the original sparse matrix A of system 1.1. In addition to holding the original term, it should also be of sufficient size to accommodate all the coarse mesh matrices generated for every level grid. Its size should be

$L1 = (\Sigma_{k=1}^{K} N_k) * 5$.

The non-zero coefficients of A (which satisfy conditions 1.2) are to be stored in CF as if it were dimensioned CF (NX,NY,5). The way in which the coefficients are ordered is easily understood from equation 1.4 and figure 1, *i.e.*

$CF(1,I,J) = a_{k,k+NX}$,

$CF(2,I,J) = a_{k,k-1}$,

$CF(3,I,J) = a_{k,k-NX}$,

$CF(4,I,J) = a_{k,k+1}$,

$CF(5,I,J) = a_{k,k}$,

where $k = I + (J-1) * NX$. The array CF has some 'holes' corresponding to fictitious external grid points when the five-point stencil of figure 1 is positioned at the boundary. These elements of CF are never used by MLTGRD.

B is a vector whose first NX*NY locations contain the right hand side of equation 1.1. B must be dimensioned of length L2 ($= \sum_{k=1}^{K} N_k$) and satisfy condition 1.3.

X is a vector whose first NX*NY elements contain the initial estimate on entry and the solution on return. In addition, the vector is used for intermediate coarse mesh systems and its total length must be L2.

R is a work vector of length L2.

ISUB is an integer vector whose size is at least the number of coarse meshes +1. On return the $(i+1)^{th}$ element contains the number of times the i^{th} coarse mesh was used.

ITCNT is an integer vector whose size is at least the number of coarse meshes + 1. On return the i^{th} element contains the total number of iterations used for all passes through that level grid (i = 1 corresponds to the original fine grid).

L1 is the size of the coefficient matrix.

L2 is the size of the various vectors.

NX is the number of grid points along the x axis of the original geometry.

NY is the number of grid points along the y axis of the original geometry.

ICFMSH, LNX, LNY, IMESH, ACC, RETAC — are work arrays whose lengths are the same as ITCNT. The first four are of the integer type and ACC and RETAC are real*8.

WPMINI, WQMINI, WRMINI — are real*8 working arrays of size max (NX,NY). Alternatively, they may be dimensioned to the grid size appropriate to the chosen line direction (LINE).

ITMIN is the minimum number of iterations which must be performed before a coarser mesh is considered after the start or re-commencement of an iterative scheme at any level (three are usually sufficient).

JMETH = 0 The multigrid procedure is used to refine the initial estimate before any fine mesh iterations are performed.

= 1 At least ITMIN iterations on the fine mesh are performed at the start before the multigrid system is enabled.

From experience gained so far JMETH = 0 is the recommended choice. Results for both options have been given in the tables.

FACTOR is the factor S of equation 2.2 which is used to determine whether convergence is becoming difficult (0.8 is suggested).

ACCFI is the relative accuracy required of the fine mesh for convergence:

$$\sum_{m=1}^{N_k} \left\{ \frac{\left| {}_m x_k^{(n_k)} - {}_m x_k^{(n_k^{-1})} \right|}{\left| {}_m x_k(n_k) \right|} \right\} / N_k \leq ACCFI \ .$$

NDIVX, NDIVY are the number of fine mesh points in the x and y directions to be gathered into successive coarse meshes. The gathering operation commences from the origin, as indicated in figure 3.

LINE The iterative scheme MINI used in MLTGRD is of a block form. The block is based on a single line in the x or y direction based on the geometry. In figure A1, four possible line specifications are shown. The best selection may be made by balancing the following criteria (listed in decreasing importance):

(i) passing the line in the direction of the smaller grid step sizes;

(ii) passing the line along the axis with the maximum number of grid points ; and

(iii) passing the line so that solutions first passes through points associated with the most significant values of the source b.

For LINE > 2, selection is automatic based on the third criterion alone. To minimise paging for larger problems, | LINE | = 2 is advantageous. Users should set up such problems accordingly, otherwise truly intolerable overheads will eventuate on the main computer.

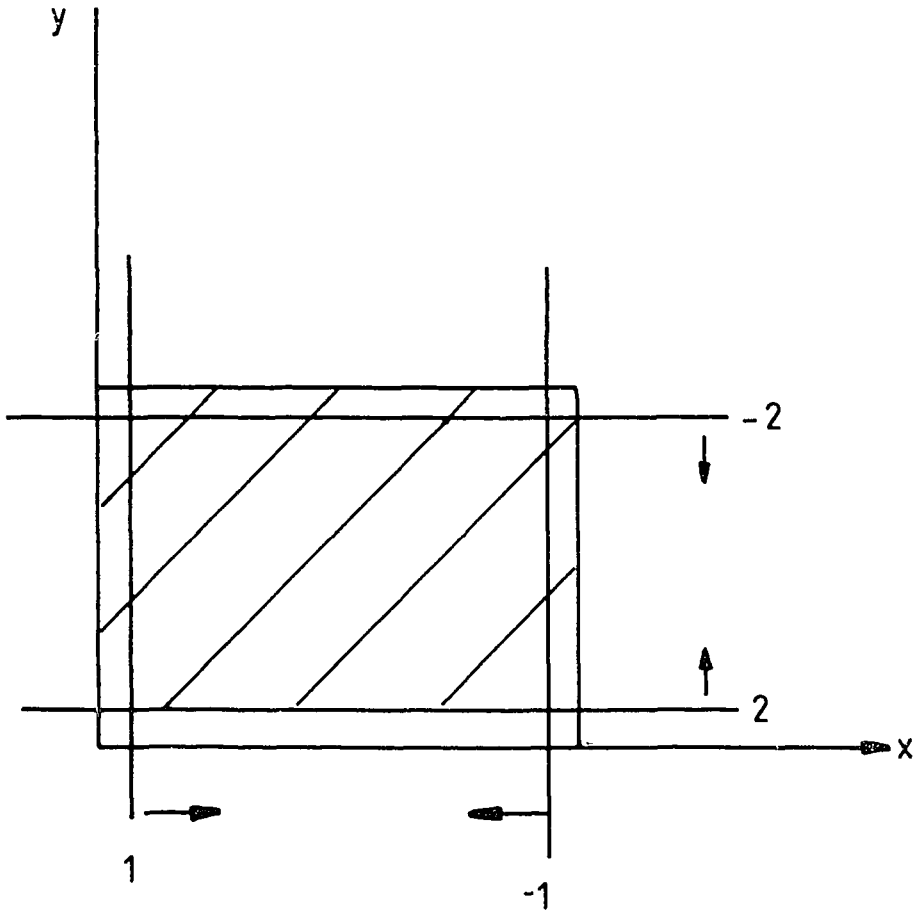

Figure A1 Line orientation and direction of travel for MINI