Numerical multigrid software for elliptic PDEs
Routines: MGD1M and MGD5M

M. Nool, P.M. de Zeeuw

Department of Numerical Mathematics

Report NM-R9423 November 1994

CWI is the National Research Institute for Mathematics and Computer Science. CWI is part of the Stichting Mathematisch Centrum (SMC), the Dutch foundation for promotion of mathematics and computer science and their applications.
SMC is sponsored by the Netherlands Organization for Scientific Research (NWO). CWI is a member of ERCIM, the European Research Consortium for Informatics and Mathematics.

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

Numerical Multigrid Software for Elliptic PDEs

Routines: MGD1M and MGD5M

Margreet Nool
e-mail: greta@cwi.nl

Paul M. de Zeeuw
e-mail: pauldz@cwi.nl

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

Abstract

The Numerical Multigrid Software FORTRAN routines MGD1M and MGD5M, multitasked successors of MGD1V and MGD5V are described. General techniques to improve the (vector) performance on the CRAY Y-MP, which are also suitable for other high-performance machines have been applied. Parallelism has been introduced by applying auto- and macrotasking.
MGD1M solves 7-diagonal linear systems, that arise from 7-point discretizations of elliptic PDEs on a rectangle, using a multigrid technique with ILU relaxation as smoothing process. A straightforward ILU implementation leads to non-vectorizable recursive relations. By applying an alternative ordering of the grid points, the ILU relaxation vectorizes and the overall execution time is reduced considerably.
MGD5M solves 7-diagonal linear systems, that arise from 7-point discretizations of elliptic PDEs on a rectangle, using a multigrid technique with Incomplete Line LU(ILLU) relaxation as smoothing process. The ILLU relaxation has been partly parallelized on the finest grid by performing a twisted LDU-decomposition.

AMS Subject Classification (1991): 65N55, 65F10, 65Y05.
CR Subject Classification (1991): G.1.8, G.1.3, G.1.0.

Keywords & Phrases: Elliptic PDEs, Galerkin approximation, Multigrid methods, Numerical software, Sparse linear systems, ILU relaxation, twisted LDU-decompositions, ILLU relaxation.

Note: The implementations are available in auto-vectorizable ANSI FORTRAN 77. Variants, using macro- and autotasking tuned for the CRAY Y-MP, are available too.

Report NM-R9423
ISSN 0169-0388
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

MGD1M - Routine Document

1. Purpose

MGD1M solves 7-diagonal linear systems, that arise from 7-point discretizations of elliptic PDEs on a rectangle, using a multigrid technique.

2. Specification

```
      SUBROUTINE MGD1M( LEVELS, NXC, NYC, NXF, NYF, NM, A, U, RHS, UB, US, RESNO,
     +                  IOUT, ISTART, IPREP, MAXIT, TOL, IFAIL)
      INTEGER  LEVELS, NXC, NYC, NXF, NYF, NM, IOUT(5), ISTART, IPREP, MAXIT, IFAIL
      REAL     A(NM*7), U(NM), RHS(NM), UB(NXF*NYF), US(NM), RESNO, TOL
```

3. Description

MGD1M solves a 7-diagonal linear system, that arises from a 7-point discretization of an elliptic PDE on a rectangle. The system is written in the form

$$A \times U = RHS, \tag{1}$$

where A and RHS are the user-supplied matrix and right-hand side, respectively. Note, that only the 7 non-zero diagonals are stored. The approximate solution is found by means of the multigrid correction storage algorithm

with: smoothing by ILU relaxation, symmetric 7-point prolongation and restriction, Galerkin approximation of coarse-grid matrices; also on the coarsest grid ILU relaxation is used instead of an exact solver.

The relation between a coarse grid and a fine grid (two subsequent levels) is the following:

let $C = \{ (x_i, y_j) \mid x_i = (i-1)h + o_1,\ i = 1\ (1)\ NXC;$

$$y_j = (j-1)h + o_2,\ j = 1\ (1)\ NYC\ ) \}$$

be the coarse grid, then the next fine grid is

$$F = \{ (x_i, y_j) \mid x_i = (i-1)\frac{h}{2} + o_1,\ i = 1\ (1)\ (2\,NXC-1);$$

$$y_j = (j-1)\frac{h}{2} + o_2,\ j = 1\ (1)\ (2\,NYC-1)\ \}$$

The user needs no knowledge of the multigrid method, it suffices to comply with the specification of the parameters.

The user supplies a bound of the l_2-norm of the residual vector. The user may supply an initial approximation of the solution (alternatively, the zero solution is used as initial approximation).

4. References

[1] Hemker, P.W., Kettler, R., Wesseling, P., de Zeeuw, P.M., Multigrid Methods: Development of fast solvers, Appl. Math. Comp. 13, pp. 311-326 (1983).

[2] Hemker, P.W., Wesseling, P., de Zeeuw, P.M., A portable vector code for autonomous multigrid modules. In: PDE software: modules, interfaces and systems. (B. Engquist and T. Smedsaas eds.), pp. 29-40, Procs. IFIP WG 2.5 working conference, North-Holland, 1984.

[3] Hemker, P.W., de Zeeuw, P.M., Some implementations of multigrid linear system solvers. Lectures held at the University of Bristol, England, Sept.1983. In: D.J. Paddon and Holstein (eds.), Multigrid methods for integral and differential equations. The Institute of Mathematics and its Applications Conference Series, Oxford University Press, 1985, pp.85-116.

[4] Louter-Nool, M., MGD1M, a parallel multigrid code with a fast vectorized ILU-relaxation, CWI-Report NM-R9222.

[5] Numerical Algorithms Group, NAG FORTRAN library manual - mark 11, 1984.

[6] Sonneveld, P., Wesseling, P., de Zeeuw, P.M., Multigrid and conjugate gradient methods as convergence acceleration techniques. Lectures held at the University of Bristol, England, Sept.1983. In: D.J. Paddon and H. Holstein (eds.), Multigrid methods for integral and differential equations. The Institute of Mathematics and its Applications Conference Series, Oxford University Press, 1985, pp.117-167.

[7] Wesseling, P., A robust and efficient multigrid method. In: W. Hackbusch and U. Trottenberg (Eds.), Lecture Notes in Mathematics, Springer, Berlin, 1981, pp. 614-630.

[8] Zeeuw, P.M. de, NUMVEC FORTRAN library manual, Chapter: Elliptic PDEs, Routine: MGD1V and MGD5V, CWI-Report NM-R8624.

5. Parameters

LEVELS - INTEGER.
On entry, LEVELS must specify the number of levels in the multigrid method.
$1 \leq \text{LEVELS} \leq 12$.
Unchanged on exit.

NXC - INTEGER.
NYC - INTEGER.
On entry, NXC and NYC must specify the number of vertical and horizontal grid-lines, respectively, on the coarsest grid.
NXC, NYC ≥ 3.
Unchanged on exit.

NXF - INTEGER.
NYF - INTEGER.
On entry, NXF and NYF must specify the number of vertical and horizontal grid-lines, respectively, on the finest grid.
Unchanged on exit.

NM - INTEGER.
On entry, NM must specify the number of grid-points on all grids together.
Unchanged on exit.

Note that the following relations should hold:

$$\text{NXF} = (\text{NXC}-1) \times 2^{\text{LEVELS}-1} + 1$$

$$\text{NYF} = (\text{NYC}-1) \times 2^{\text{LEVELS}-1} + 1$$

$$\text{NM} = \sum_{L=1}^{\text{LEVELS}} ((\text{NXC}-1) \times 2^{L-1}+1)((\text{NYC}-1) \times 2^{L-1}+1)$$

The program checks the consistency of these data

Examples:

LEVELS	2	3	4	5	6	7	8
NXC	5	5	5	5	5	5	5
NYC	5	5	5	5	5	5	5
NXF	9	17	33	65	129	257	513
NYF	9	17	33	65	129	257	513
NM	106	395	1484	5709	22350	88399	351568

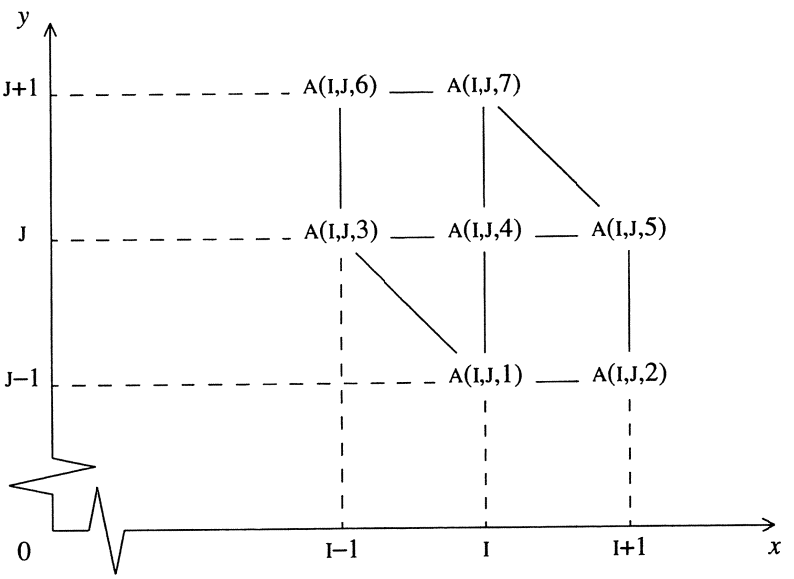
LEVELS	2	3	4	5	6	7	8
NXC	5	5	5	5	5	5	5
NYC	3	3	3	3	3	3	3
NXF	9	17	33	65	129	257	513
NYF	5	9	17	33	65	129	257
NM	60	213	774	2919	11304	44457	176298

A - REAL array of DIMENSION at least (NM*7).
Before entry, if IPREP=0 then the first NXF*NYF*7 elements of A must contain the 7-diagonal matrix of the linear system (1). (If IPREP=0 then the contents of A will be overwritten by the program.)
Before entry, if IPREP=1 then the first NM*7 elements of A must contain the unaltered contents of A after the last previous call of MGD1M. (If IPREP=1 those contents are assumed to be the decompositions of the matrix of (1) and its coarse-grid approximations.)

The following holds for the case that IPREP=0:

The easiest way for the user to fill the matrix A is writing a subroutine where the actual argument A is handled as an adjustable array with dimensions (NXF, NYF, 7).

The 7-point difference molecule at the point with subscripts (I, J) is positioned in the x,y -plane as follows:



Important: the user must provide the matrix A only on the finest grid. The coarse-grid matrices are computed internally by the routine by means of Galerkin approximation.

Important: the user must take care that parts of the molecules outside the domain are initialized to zero, otherwise wrong results are produced.

Remark: problems with topologically non-rectangular regions can also be solved using this routine by surrounding the region by a circumscribing topological rectangle. Let (ie,je) denote a nodal point external to the region of interest, then set A(ie,je,k)=0 for k=1,2,3,5,6,7 and A(ie,je,4)=eps with eps small, e.g. 10^{-9} (further set RHS(ie,je)=0).

On exit, the first NM*7 elements of A contain the incomplete Crout decompositions of the user-provided matrix A and its coarse-grid approximations.
Hence, the contents of A are altered on exit.

U - REAL array of DIMENSION at least (NM).
Before entry, but only if the routine is entered with ISTART = 1 or with ISTART = 2, the first NXF*NYF elements of U must contain an initial estimate for the iterative process. If ISTART = 0 no initialization of U is necessary (a zero initial estimate is assumed).

The easiest way for the user to fill the initial estimate is writing a subroutine where the actual argument U is handled as an adjustable array with dimensions (NXF, NYF).

On successful exit, the first NXF*NYF elements of U contain the (approximate) numerical solution.

RHS - REAL array of DIMENSION at least (NM).
Before entry, the first NXF*NYF elements of RHS must contain the right-hand side of the linear system (1), the other elements are used as workspace.

The easiest way for the user to fill the right-hand side is writing a subroutine where the actual argument RHS is handled as an adjustable array with dimensions (NXF, NYF).

Important: the user must provide the right-hand side of the discretized equation only on the finest grid.

The first NXF*NYF elements of RHS are unchanged on exit.

UB - REAL array of DIMENSION at least (NXF*NYF).
Before entry, but only if the routine is entered with ISTART = 2, the first NXF*NYF elements of UB must contain the residual RHS–A*U of the initial estimate U.
On successful exit, the first NXF*NYF elements of UB contain the residual of the approximate numerical solution U.

US - REAL array of DIMENSION at least (NM).

RESNO - REAL.
On exit, RESNO contains the l_2 -norm of the residual.

IOUT - INTEGER array of DIMENSION at least (5).
IOUT governs the amount of information about the solution process delivered to the user. Smaller IOUT-values mean less output. The user may select the unit-number on which this output is to appear by a call of X04ABF. Before entry, the first 5 elements of IOUT must contain one of the following values:

IOUT(1)	≥ 1	confirmation of input data
	≤ 0	none
IOUT(2)	≥ 2	matrices on all levels and right hand side on highest level
	= 1	matrix and right-hand side on highest level
	≤ 0	none

IOUT(3) ≥ 2	matrix-decompositions on all levels
= 1	matrix-decomposition on highest level
≤ 0	none
IOUT(4) ≥ 4	norms of residuals, reduction factors, final solution, final residual
= 3	norms of residuals, reduction factors, final residual
= 2	norms of residuals, reduction factors (i.e. monitoring the convergence-behavior)
= 1	final norm of residual, number of iterations
≤ 0	none
IOUT(5) ≥ 1	the time spent in various subroutines
≤ 0	none

The contents of IOUT are unchanged on exit.

ISTART - INTEGER.
On entry, ISTART must be set to 0, 1 or 2.

ISTART = 0	means that the initial estimate is zero, initialization of U is not necessary;
= 1	means that the user provides an initial estimate of the solution in U;
= 2	means that the user provides an initial estimate of the solution in U and the residual of U in UB.

Unchanged on exit.

IPREP - INTEGER.
On entry, IPREP must be set to 0 or 1.

IPREP = 1	if MGD1M has been called before and the information in A after that call has not been changed. (This is useful if the user wants to refine the numerical solution by re-entering the routine or if the user desires to solve a sequence of problems with identical matrices but different right-hand sides, this option then prevents a new setup of coarse-grid matrices and decompositions which have already been computed.)
= 0	if MGD1M has not been called before, or if after a call of MGD1M the information in A has been altered.

Unchanged on exit.

MAXIT - INTEGER.
TOL - REAL.
On entry, MAXIT must specify the maximum number of allowed multigrid iterations and TOL must specify the tolerance desired by the user, where TOL is a bound of the l_2-norm of the residual.
If during the multigrid process either MAXIT iterations have been performed or the tolerance has been reached, multigrid cycling is stopped.
For MAXIT a value of about 10 is adequate for many problems; clearly TOL should not be less than a reasonable multiple of NXF*NYF times the machine accuracy.
Both MAXIT and TOL are unchanged on exit.

IFAIL - INTEGER.
For this routine, the normal use of IFAIL is extended to control the printing of error and warning messages as well as specifying hard or soft failure (see [5, chapter P01]). On entry IFAIL must be set to a value with the decimal expansion cba, where each of the decimal digits c, b and a must have the value 0 or 1.

a = 0	specifies hard failure, otherwise soft failure;
b = 0	suppresses error messages, otherwise error messages will be printed (see section 6);

c = 0 suppresses warning messages, otherwise warning messages will be printed (see section 6).

For users not familiar with this parameter the recommended value is 110 (i.e. hard failure with all messages printed).
Unless the routine detects an error (see section 6), IFAIL contains 0 on exit.

6. Error indicators and warnings

Errors detected by the routine:-

For some errors the routine outputs an explanatory message on the current error message unit (see routine X04AAF), unless suppressed by the value of IFAIL on entry.

IFAIL = 1
On entry, LEVELS is out of bounds.

IFAIL = 2
On entry, NXC or NYC is less than 3.

IFAIL = 3
On entry, no consistency among LEVELS, NXC and NXF.

IFAIL = 4
On entry, no consistency among LEVELS, NYC and NYF.

IFAIL = 5
On entry, NM is wrong.

IFAIL = 6
divergence, maybe due to this routine or to user discretization.

IFAIL = 7
poor convergence, maybe due to this routine or to user discretization or to reaching round off level.

IFAIL = 8
convergence, but MAXIT iterations performed without reaching TOL. This could be due to a value of MAXIT which is too small or to reaching round off level. The user is advised to monitor the convergence behavior by choosing IOUT(4) ≥ 2.

7. Auxiliary routines

This routine calls the following routines:
CPUTIM, CTUMV, CTUPF, CYCLES, DECOMP, GALERK, ILUDEC, OUTMAT, OUTVEC, PREPAR, PROLON, P01AAF, RAP, RESTRI, SOLVE, TIMING, X04AAF and X04ABF,
and the following BLAS library routines:
SCOPY, SNRM2.

8. Timing

The time taken per multigrid iteration is proportional to NXF × NYF.
However, the vector performance (mflop-rate) increases with increasing NXF, therefore it is advised to take the longest side of the rectangular grid along the x-axis assuming that for many problems there is not much orientation-sensitivity in the algorithm used.
According to multigrid theory it is plausible that the convergence rate is independent of the chosen meshwidth.

9. Storage

Internally declared arrays contain 6 REAL and 36 INTEGER elements.

10. Accuracy

If the process converges, and MAXIT is large enough, the l_2 -norm of the residual becomes less than TOL.

11. Further comments

Labeled common blocks CPU and POI are used by the routine and must therefore be avoided by users.
The user is strongly recommended to set IFAIL to obtain self-explanatory error- and advisory-messages. The user may select the unit numbers on which this output is to appear by calls of X04AAF (for error messages) or X04ABF (for advisory messages) - see section 13 for an example. Otherwise the default unit numbers will be used.

11.1. Vectorization and parallelization information

The ILU-decomposition and relaxation of the original MGD1V[8] are responsible for more than 65% of the total CPU-time. The coupling of the elements leads to recurrence relations, when rows are computed sequentially. The use of highly efficient routines as FOLR and FOLR2 of the Cray SCIentific LIBrary, which solve first order linear recurrences, results in an acceptable performance gain of 17%. However, an alternative ordering of the grid points, the so-called knight's move ordering of Ashcraft and Grimes, leads to a scheme which is no longer recursive and consequently, a much higher performance is obtained. On the finest grid the MFlop rate increases from 54 Mflops (for the FOLR-implementation) to 155 Mflops for the knight's move ordering.

The prolongation and restriction operators as well as the Galerkin approximations turn out to be highly parallel: a speedup of nearly four is achieved on a Cray Y-MP with four processors. Although a considerable reduction in CPU-time can be achieved by vectorization the relaxation still remains the most time-consuming part, because it can not be parallelized without an important loss of vector performance. The new grid lines are too short to exploit parallelism efficiently, whereas the strict order of updating of the new lines does not allow concurrent processing. For more information on vectorization and parallelization see Louter-Nool[4].

The program has been written in ANSI FORTRAN 77, except for the auxiliary routine DECOMP. A distinction can be made between the parallel and the vector implementation; the latter does not apply macrotasking. Actually, there are two versions of the routine DECOMP: one implementation of DECOMP calls the macrotasking CRAY routines TSKSTART and TSKWAIT in order to perform the ILU-decomposition of different grids in parallel, the other one is meant for running on a single processor. For the parallel case the compiler option -Zp is recommended, otherwise the compiler option -Zv is recommended. To get an idea how the wall-clock time can be reduced by parallel execution, the wall-clock times for different phases in the multigrid process are listed in Table 1. The results are achieved for the Poisson equation, discussed in section 13, using 8 levels with a coarsest grid of 5×5 grid points. Moreover, the timings are compared to those achieved for MGD1V[8].

	MGD1V	MGD1M					
	Vector	Vector	Speedup	$p = 1$	$p = 2$	$p = 3$	$p = 4$
Galerkin	.131	.043		.042	.021	.014	.013
S_p, $p = 2, 3, 4$			*3.05*		*2.00*	*2.96*	*3.16*
Decompose	.169	.045		.046	.032	.032	.032
S_p, $p = 2, 3, 4$			*3.77*		*1.41*	*1.41*	*1.41*
MG-cycles	.774	.380		.380	.295	.268	.255
S_p, $p = 2, 3, 4$			*2.03*		*1.29*	*1.42*	*1.49*
Total	1.074	.468		.468	.348	.314	.300
S_p, $p = 2, 3, 4$			*2.29*		*1.34*	*1.49*	*1.56*

TABLE 1. Wall-Clock time in seconds for 8 Levels and Speedups.

12. Keywords

Elliptic PDEs,
Galerkin approximation,
Multigrid methods,
Sparse linear systems,
ILU relaxation.

13. Example

We solve the Poisson equation on the unit square with Dirichlet boundary conditions and the right-hand side constructed according to the exact solution $x\,(1-x) + y\,(1-y)$. In this example the boundary conditions are eliminated.

First we try 3 iterations with a zero initial approximation and if no divergence is found (using the soft fail option) we try to refine the solution until the residual norm becomes lesser than 10^{-9} using the former approximation (computations are performed with 14 decimal digits).

Note the calls to X04AAF and X04ABF prior to the call of MGD1M.

13.1. Program text

```
      PROGRAM EXAMPL
C
      INTEGER       LEVELS, NXC, NYC, NXF, NYF, NM
      PARAMETER( LEVELS = 7, NXC = 5, NYC = 5,
     +           NXF = 257, NYF = 257, NM = 88399)
C
C     .. Scalar Arguments ..
      INTEGER       IFAIL, IPREP, ISTART, MAXIT, NOUT
      REAL          RESNO, TOL
C     ..
C     .. Array Arguments ..
      INTEGER       IOUT( 5 )
      REAL          RHS( NM ), A( 7 * NM ), VS( NM ),
     +              V( NM ), VB( NXF * NYF )
C     ..
C     .. External Subroutines ..
```

```
      EXTERNAL   CPUTIM, MATRHS, MGD1M, TIMING, X04AAF, X04ABF
C     ..
C     .. Timing Variables ..
      REAL       CPB, CPE, WCB, WCE
C     ..
C     .. Data Statements ..
      DATA       IOUT /1, 0, 0, 2, 1/
      DATA       NOUT /6/
C     ..
C     .. Executable Statements ..
C
      CALL CPUTIM( CPB )
      CALL TIMING( WCB )
C
      OPEN( UNIT = NOUT, FILE = 'OUTPUT' )
      CALL X04AAF( 1, NOUT )
      CALL X04ABF( 1, NOUT )
C
      WRITE( NOUT, 9999 )
C
C     Problem set up.
C     Subroutine MATRHS is an example of a subroutine which fills the
C     matrix and the right-hand side.
C
      CALL MATRHS( A, RHS, NXF, NYF, NOUT )
C
C     Approximate the solution of the linear system by performing 3
C     MGD1M iterations, using the soft fail option.
C
      ISTART= 0
      IPREP  = 0
      MAXIT = 3
      TOL    = 0.0E+0
      IFAIL  = 111
C
      CALL MGD1M( LEVELS, NXC, NYC, NXF, NYF, NM,
     +            A, V, RHS, VB, VS, RESNO,
     +            IOUT, ISTART, IPREP, MAXIT, TOL, IFAIL)
C
C     Possible refinement until residual norm .LT. 1.0E-9
C
      IF( IFAIL.GE.7 ) THEN
        ISTART= 2
        IPREP  = 1
        MAXIT = 100
        TOL    = 1.0E-9
        IFAIL  = 111
C
        CALL MGD1M( LEVELS, NXC, NYC, NXF, NYF, NM,
     +              A, V, RHS, VB, VS, RESNO,
     +              IOUT, ISTART, IPREP, MAXIT, TOL, IFAIL)
      END IF
C
```

```
      CALL TIMING( WCE )
      CALL CPUTIM( CPE )
C
      WRITE (NOUT, 9998) WCE - WCB, CPE - CPB
C
C     .. Format Statements ..
C
 9998 FORMAT( ' Total Wall clock time Example of Use : ', t45, F8.3, /,
     +        ' Total CPU time      Example of Use : ', t45, F8.3 )
 9999 FORMAT( ' 1 MGD1M Example program results. ' )
C
      STOP
C
C     End of Program EXAMPL
C
      END
      SUBROUTINE MATRHS( A, RHS, NXF, NYF, NOUT )
C
C     .. Scalar Arguments ..
      INTEGER       NXF, NYF, NOUT
C     ..
C     .. Array Arguments ..
      REAL          A( NXF, NYF, 7 ), RHS( NXF, NYF )
C     ..
C
C-----------------------------------------------------------------------
C     MATRHS is a subroutine which fills the matrix and the right-hand
C     side. It is part of the example program.
C
C     The example is the Poisson equation on the unit square with
C     Dirichlet boundary conditions and the exact solution is:
C     X * (XSIZE - X) + Y * (YSIZE - Y).
C     In this example the boundary conditions are eliminated.
C-----------------------------------------------------------------------
C     ..
C     .. Local Variables ..
      INTEGER       I, J
      REAL          B, X, XH, XSIZE, XY, Y, YH, YSIZE, YX
      REAL          TWOYX, FOURXY
C     ..
C     .. Parameters ..
      REAL          ZERO, ONE, TWO, FOUR
      PARAMETER     ( ZERO = 0.0E+0, ONE  = 1.0E+0,
     +              TWO  = 2.0E+0, FOUR = 4.0E+0 )
C
C     .. Intrinsic Functions ..
      INTRINSIC     REAL
C     ..
C     .. Executable Statements ..
C     ..
C
      XSIZE = ONE
      YSIZE = ONE
```

```
      XH = XSIZE / REAL( NXF + 1 )
      YH = YSIZE / REAL( NYF + 1 )
C
C     /(... + 1 ) Because of elimination of boundary conditions
C
      WRITE( NOUT, 9999 ) XSIZE, YSIZE, XH, YH
C
      XY = XH / YH
      YX = YH / XH
C
C     Initial filling of the matrix and the right-hand side neglecting
C     the boundaries.
C
      DO 10 J = 1, NYF
        DO 10 I = 1, NXF
          A( I, J, 1 ) = -XY
          A( I, J, 2 ) = ZERO
          A( I, J, 3 ) = -YX
          A( I, J, 4 ) = TWOYX
          A( I, J, 5 ) = -YX
          A( I, J, 6 ) = ZERO
          A( I, J, 7 ) = -XY
          RHS( I, J ) = FOURXY
   10 CONTINUE
C
C     Correction for the Dirichlet boundary conditions corresponding
C     to the exact solution, X * (XSIZE - X) + Y * (YSIZE - Y)
C
C     Note, that after this correction-process all parts of the
C     difference-molecules outside the domain are initialized to zero!
C
C-----------------------------------------------------------------------
C     Lower and Upper Boundary
C-----------------------------------------------------------------------
      X = ZERO
      DO 20 I = 1, NXF
        X = X + XH
        B = X * ( XSIZE - X )
        RHS( I, 1 ) = RHS( I, 1 ) - A( I, 1 , 1 ) * B
        RHS( I, NYF ) = RHS( I, NYF ) - A( I, NYF, 7 ) * B
        A( I, 1 , 1 ) = ZERO
        A( I, NYF, 7 ) = ZERO
   20 CONTINUE
C-----------------------------------------------------------------------
C     Left and Right-hand Boundary
C-----------------------------------------------------------------------
      Y = ZERO
      DO 30 J = 1, NYF
        Y = Y + YH
        B = Y * ( YSIZE - Y )
        RHS( 1 , J ) = RHS( 1 , J ) - A( 1 , J, 3 ) * B
        RHS( NXF, J ) = RHS( NXF, J ) - A( NXF, J, 5 ) * B
        A( 1 , J, 3 ) = ZERO
```

```
            A( NXF, J, 5 ) = ZERO
   30     CONTINUE
C
C         .. Format Statements ..
C
 9999     FORMAT( ' Poisson Problem: '//' XSIZE  = ',
     +           1PE13.6/' YSIZE  = ',
     +           E13.6/' XH, YH = ', 2E13.6 )
C
          RETURN
C
C         End of Subroutine MATRHS
C
          END
```

13.2. Program data

None.

13.3. Program results

```
 MGD1M Example program results.
 Poisson Problem:

 XSIZE   =   1.000000E+00
 YSIZE   =   1.000000E+00
 XH, YH  =   3.875969E-03 3.875969E-03

 Multigrid Program MGD1M, version 9 June 1993

 LEVELS     NXC     NYC     NXF     NYF        NM
      7       5       5     257     257     88399
  MAXIT         TOL
      3         0.000000E+00
                IOUT  ISTART  IPREP  IFAIL
      1  0   0  2  1       0      0    111

 L2-Norm of initial residual =     0.587E+01

 MGD1M, Iteration Number  =         1
 L2-Norm of Residual      =     0.731E-03
 Current Reduction Factor =     0.125E-03
 Average Reduction Factor =     0.125E-03

 MGD1M, Iteration Number  =         2
 L2-Norm of Residual      =     0.445E-04
 Current Reduction Factor =     0.609E-01
 Average Reduction Factor =     0.276E-02

 MGD1M, Iteration Number  =         3
 L2-Norm of Residual      =     0.310E-05
 Current Reduction Factor =     0.696E-01
```

```
Average Reduction Factor =    0.808E-02

Maxit Iterations performed without reaching TOL.

Good Convergence Rate, so V and VB are valuable,
at this point one may restart MGD1M with ISTART = 2 and IPREP = 1

Error detected by Library routine MGD1M    - IFAIL =  8


CP-time and Wall Clock used ( sec. )

Galerkin     0.010344       0.010339

Decompose    0.014962       0.014957

MG-cycles    0.058977       0.058972

Multigrid Program MGD1M, version 9 June 1993

LEVELS    NXC    NYC     NXF     NYF        NM
     7      5      5     257     257     88399
 MAXIT         TOL
   100         0.100000E-08
               IOUT  ISTART  IPREP  IFAIL
     1  0  0  2  1       2      1    111

L2-Norm of initial residual =    0.310E-05

MGD1M, Iteration Number  =        1
L2-Norm of Residual      =    0.205E-06
Current Reduction Factor =    0.660E-01
Average Reduction Factor =    0.660E-01

MGD1M, Iteration Number  =        2
L2-Norm of Residual      =    0.141E-07
Current Reduction Factor =    0.688E-01
Average Reduction Factor =    0.674E-01

MGD1M, Iteration Number  =        3
L2-Norm of Residual      =    0.936E-09
Current Reduction Factor =    0.665E-01
Average Reduction Factor =    0.671E-01


CP-time and Wall Clock used ( sec. )

Galerkin     0.000000       0.000000

Decompose    0.000000       0.000000

MG-cycles    0.058586       0.058581
Total Wall clock time Example of Use :        0.151
```

```
Total CPU time        Example of Use :        0.150
```

MGD5M - Routine Document

1. Purpose

MGD5M solves 7-diagonal linear systems, that arise from 7-point discretizations of elliptic PDEs on a rectangle, using a multigrid technique.

2. Specification

```
      SUBROUTINE MGD5M( LEVELS, NXC, NYC, NXF, NYF, NM, A, U, RHS, UB, WORK, LDU, RESNO,
     +                  IOUT, ISTART, IPREP, MAXIT, TOL, IFAIL)
      INTEGER  LEVELS, NXC, NYC, NXF, NYF, NM, IOUT(5), ISTART, IPREP, MAXIT, IFAIL
      REAL     A(NM*7), U(NM), RHS(NM), UB(NM), WORK((NXF+(NXF-1)/2)*9),
     +         LDU(NM*3), RESNO, TOL
```

3. Description

MGD5M solves a 7-diagonal linear system, that arises from a 7-point discretization of an elliptic PDE on a rectangle. The system is written in the form

$$A \times U = RHS, \tag{1}$$

where A and RHS are the user-supplied matrix and right-hand side, respectively. Note, that only the 7 non-zero diagonals are stored. The approximate solution is found by means of the multigrid correction storage algorithm

with: smoothing by ILLU relaxation, symmetric 7-point prolongation and restriction, Galerkin approximation of coarse-grid matrices; also on the coarsest grid ILLU relaxation is used instead of an exact solver. On the finest grid a twisted ILLU relaxation is performed on two processors simultaneously.

The relation between a coarse grid and a fine grid (two subsequent levels) is the following:

let $C = \{ (x_i, y_j) \mid x_i = (i-1)h + o_1,\ i = 1\ (1)\ NXC;$

$$y_j = (j-1)h + o_2,\ j = 1\ (1)\ NYC\ ) \}$$

be the coarse grid, then the next fine grid is

$$F = \{ (x_i, y_j) \mid x_i = (i-1)\frac{h}{2} + o_1,\ i = 1\ (1)\ (2\ NXC-1);$$

$$y_j = (j-1)\frac{h}{2} + o_2,\ j = 1\ (1)\ (2\ NYC-1)\ \}$$

The user needs no knowledge of the multigrid method, it suffices to comply with the specification of the parameters.

The user supplies an absolute residual tolerance in the form of a bound of its l_2 -norm. The user may also supply an initial approximation (alternatively, the zero solution is used as initial approximation).

4. References

[1] Hemker, P.W., de Zeeuw, P.M., Some implementations of multigrid linear system solvers. Lectures held at the University of Bristol, England, Sept.1983. In: D.J. Paddon and Holstein (eds.), Multigrid methods for integral and differential equations. The Institute of Mathematics and its Applications Conference Series, Oxford University Press, 1985, pp.85-116.

[2] Louter-Nool, M., Numerical Multigrid Software: MGD5M, a parallel multigrid code with a twisted ILLU-relaxation, CWI-Report NM-R9226.

[3] Meijerink, J.A., Iterative methods for the solution of linear equations based on incomplete factorization of the matrix. Publication 643, Shell Research B.V., Kon. Shell Expl. and Prod. Lab., Rijswijk, The Netherlands, July 1983.

[4] Numerical Algorithms Group, NAG FORTRAN library manual - mark 11, 1984.

[5] Sonneveld, P., Wesseling, P., de Zeeuw, P.M., Multigrid and conjugate gradient methods as convergence acceleration techniques. Lectures held at the University of Bristol, England, Sept.1983. In: D.J. Paddon and H. Holstein (eds.), Multigrid methods for integral and differential equations. The Institute of Mathematics and its Applications Conference Series, Oxford University Press, 1985, pp.117-167.

[6] Zeeuw, P.M. de, NUMVEC FORTRAN library manual, Chapter: Elliptic PDEs, Routine: MGD1V and MGD5V, CWI-Report NM-R8624.

5. Parameters

LEVELS - INTEGER.
On entry, LEVELS must specify the number of levels in the multigrid method.
$1 \leq$ LEVELS ≤ 12.
Unchanged on exit.

NXC - INTEGER.
NYC - INTEGER.
On entry, NXC and NYC must specify the number of vertical and horizontal grid-lines, respectively, on the coarsest grid.
NXC, NYC ≥ 3.
Unchanged on exit.

NXF - INTEGER.
NYF - INTEGER.
On entry, NXF and NYF must specify the number of vertical and horizontal grid-lines, respectively, on the finest grid.
Unchanged on exit.

NM - INTEGER.
On entry, NM must specify the number of grid-points on all grids together.
Unchanged on exit.

Note that the following relations should hold:

$$\text{NXF} = (\text{NXC} - 1) \times 2^{\text{LEVELS}-1} + 1$$

$$\text{NYF} = (\text{NYC} - 1) \times 2^{\text{LEVELS}-1} + 1$$

$$\text{NM} = \sum_{L=1}^{\text{LEVELS}} ((\text{NXC} - 1) \times 2^{L-1} + 1)((\text{NYC} - 1) \times 2^{L-1} + 1)$$

The program checks the consistency of these data

Examples:

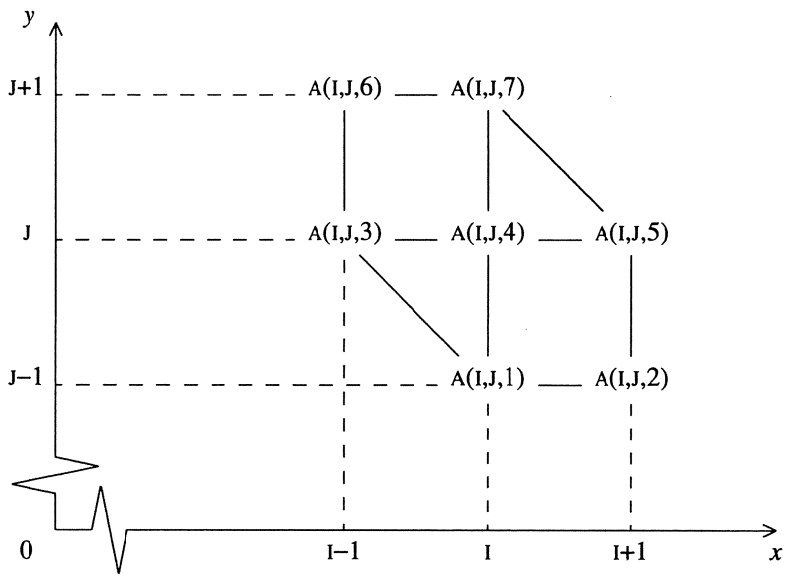
LEVELS	2	3	4	5	6	7	8
NXC	5	5	5	5	5	5	5
NYC	5	5	5	5	5	5	5
NXF	9	17	33	65	129	257	513
NYF	9	17	33	65	129	257	513
NM	106	395	1484	5709	22350	88399	351568

LEVELS	2	3	4	5	6	7	8
NXC	5	5	5	5	5	5	5
NYC	3	3	3	3	3	3	3
NXF	9	17	33	65	129	257	513
NYF	5	9	17	33	65	129	257
NM	60	213	774	2919	11304	44457	176298

A - REAL array of DIMENSION at least (NM*7).
Before entry, the first NXF*NYF*7 elements of A must contain the 7-diagonal matrix of the linear system (1).

The easiest way for the user to fill the matrix A is writing a subroutine where the actual argument A is handled as an adjustable array with dimensions (NXF, NYF, 7).

The 7-point difference molecule at the point with subscripts (I, J) is positioned in the x,y -plane as follows:



Important: the user must provide the matrix A only on the finest grid. The coarse-grid matrices are computed internally by the routine by means of Galerkin approximation.

Important: the user must take care that parts of the molecules outside the domain are initialized to zero, otherwise wrong results are produced.

Remark: problems with topologically non-rectangular regions can also be solved using this routine by surrounding the region by a circumscribing topological rectangle. Let (ie,je) denote a nodal point external to the region of interest, then set A(ie,je,k)=0 for k=1,2,3,5,6,7 and A(ie,je,4)=eps with eps small, e.g. 10^{-9} (further set RHS(ie,je)=0).

The first NXF*NYF*7 elements of A are unchanged on exit.

U - REAL array of DIMENSION at least (NM).

Before entry, but only if the routine is entered with ISTART = 1 or with ISTART = 2, the first NXF*NYF elements of U must contain an initial estimate for the iterative process. If ISTART = 0 no initialization of U is necessary (a zero initial estimate is assumed).

The easiest way for the user to fill the initial estimate is writing a subroutine where the actual argument U is handled as an adjustable array with dimensions (NXF, NYF).

On successful exit, the first NXF*NYF elements of U contain the (approximate) numerical solution.

RHS - REAL array of DIMENSION at least (NM).

Before entry, the first NXF*NYF elements of RHS must contain the right-hand side of the linear system (1), the other elements are used as workspace.

The easiest way for the user to fill the right-hand side is writing a subroutine where the actual argument RHS is handled as an adjustable array with dimensions (NXF, NYF).

Important: The user must provide the right-hand side of the discretized equation only on the finest grid.

The first NXF*NYF elements of RHS are unchanged on exit.

UB - REAL array of DIMENSION at least (NM).

Before entry, but only if the routine is entered with ISTART = 2, the first NXF*NYF elements of UB must contain the residual RHS–A*U of the initial estimate U.

On successful exit, the first NXF*NYF elements of UB contain the residual of the approximate numerical solution U.

WORK - REAL array of DIMENSION at least (NXF+(NXF–1)/2)*9.

Used as workspace.

LDU - REAL array of DIMENSION at least (NM*3).

Before entry, if IPREP = 0, the elements of LDU need not to be initialized.

Before entry, if IPREP = 1, then the first NM*3 elements of LDU must contain the unaltered contents of LDU after the last previous call of MGD5M. (If IPREP = 1 those contents are assumed to be the ILLU-decompositions of the matrix of (1) and its coarse-grid approximations.)

On exit, if IPREP = 0, the first NM*3 elements of LDU contain the ILLU-decompositions of the user-provided matrix A and its coarse-grid approximations. Only for the finest grid a twisted decomposition is given.

If IPREP = 1 then the contents of LDU are unchanged on exit.

RESNO - REAL.

On exit, RESNO contains the l_2 -norm of the residual.

IOUT - INTEGER array of DIMENSION at least (5).

IOUT governs the amount of information about the solution process delivered to the user. Smaller IOUT-values mean less output. The user may select the unit-number on which this output is to appear by a call of X04ABF. Before entry, the first 5 elements of IOUT must contain one of the following values:

IOUT(1) ≥ 1	confirmation of input data
≤ 0	none
IOUT(2) ≥ 2	matrices on all levels and right-hand side on highest level
= 1	matrix and right-hand side on highest level
≤ 0	none
IOUT(3) ≥ 2	matrix-decompositions on all levels
= 1	matrix-decomposition on highest level
≤ 0	none
IOUT(4) ≥ 4	norms of residuals, reduction factors, final solution, final residual
= 3	norms of residuals, reduction factors, final residual
= 2	norms of residuals, reduction factors (i.e. monitoring the convergence-behavior)
= 1	final norm of residual, number of iterations
≤ 0	none
IOUT(5) ≥ 1	the time spent in various subroutines
≤ 0	none

The contents of IOUT are unchanged on exit.

ISTART - INTEGER.

On entry, ISTART must be set to 0, 1 or 2.

ISTART = 0	means that the initial estimate is zero, initialization of U is not necessary;
= 1	means that the user provides an initial estimate of the solution in U;
= 2	means that the user provides an initial estimate of the solution in U and the residual of U in UB.

Unchanged on exit.

IPREP - INTEGER.

On entry, IPREP must be set to 0 or 1.

IPREP = 1	if MGD5M has been called before and the information in LDU after that call has not been changed. (This is useful if the user wants to refine the numerical solution by re-entering the routine or if the user desires to solve a sequence of problems with identical matrices but different right-hand sides, this option then prevents a new set-up of coarse-grid matrices and decompositions which have already been computed.)
= 0	if MGD5M has not been called before, or if after a call of MGD5M the information in LDU has been altered.

Unchanged on exit.

MAXIT - INTEGER.

TOL - REAL.

On entry, MAXIT must specify the maximum number of allowed multigrid iterations and TOL must specify the tolerance desired by the user, where TOL is a bound of the l_2 -norm of the residual.

If during the multigrid process either MAXIT iterations have been performed or the tolerance has been reached, multigrid cycling is stopped.

For MAXIT a value of about 10 is adequate for many problems; clearly TOL should not be less than a reasonable multiple of NXF*NYF times the machine accuracy.

Both MAXIT and TOL are unchanged on exit.

IFAIL - INTEGER.

For this routine, the normal use of IFAIL is extended to control the printing of error and warning messages as well as specifying hard or soft failure (see [4, chapter P01]). On entry IFAIL must be set to a value with the decimal expansion cba, where each of the decimal digits c, b and a must have the value

0 or 1.

a = 0 specifies hard failure, otherwise soft failure;
b = 0 suppresses error messages, otherwise error messages will be printed (see section 6);
c = 0 suppresses warning messages, otherwise warning messages will be printed (see section 6).

For users not familiar with this parameter the recommended value is 110 (i.e. hard failure with all messages printed).
Unless the routine detects an error (see section 6), IFAIL contains 0 on exit.

6. Error indicators and warnings

Errors detected by the routine:-

For some errors the routine outputs an explanatory message on the current error message unit (see routine X04AAF), unless suppressed by the value of IFAIL on entry.

IFAIL = 1
On entry, LEVELS is out of bounds.

IFAIL = 2
On entry, NXC or NYC is less than 3.

IFAIL = 3
On entry, no consistency among LEVELS, NXC and NXF.

IFAIL = 4
On entry, no consistency among LEVELS, NYC and NYF.

IFAIL = 5
On entry, NM is wrong.

IFAIL = 6
divergence, maybe due to this routine or to user discretization.

IFAIL = 7
poor convergence, maybe due to this routine or to user discretization or to reaching round off level.

IFAIL = 8
convergence, but MAXIT iterations performed without reaching TOL. This could be due to a value of MAXIT which is too small or to reaching round off level. The user is advised to monitor the convergence behavior by choosing IOUT(4) ≥ 2

7. Auxiliary routines

The parallel implementation of MGD5M calls the following routines:
BLOCKS, BLOCKT, CPUTIM, CYCLES, DECOMP, GALERK, ILLUDC, ILLUDT, OUTMAT, OUTVEC, PQDEC, PREPAR, PROLON, P01AAF, RAP, RESIDU, RESTRI, SMOOTH, SOLVE, SOLVET, TIMING, TRIDEC, X04AAF and X04ABF.
The vector implementation of MGD5M calls:
BLOCKS, CPUTIM, CYCLES, DECOMP, GALERK, ILLUDC, OUTMAT, OUTVEC, PREPAR, PROLON, P01AAF, RAP, RESIDU, RESTRI, SMOOTH, SOLVE, TIMING, TRIDEC, X04AAF and X04ABF.
Both implementations call the following BLAS routines:
SCOPY, SNRM2.
Two sets of routines are offered to the user, one for parallel and one for non-parallel execution. See also section 11.1 for more information about compilation and linking.

8. Timing

The time taken per multigrid iteration is proportional to NXF × NYF.
However, the vector performance (mflop-rate) increases with increasing NXF, therefore it is advised to take the longest side of the rectangular grid along the x-axis assuming that for many problems there is not much orientation-sensitivity in the algorithm used.
According to multigrid theory it is plausible that the convergence rate is independent of the chosen meshwidth.

9. Storage

Internally declared arrays contain 6 REAL and 36 INTEGER elements.

10. Accuracy

If the process converges, and MAXIT is large enough, the l_2 -norm of the residual becomes less than TOL.

11. Further comments

Labeled common blocks CPU and POI are used by the routine and must therefore be avoided by users.
The user is strongly recommended to set IFAIL to obtain self-explanatory error- and advisory-messages. The user may select the unit numbers on which this output is to appear by calls of X04AAF (for error messages) or X04ABF (for advisory messages) - see section 13 for an example. Otherwise the default unit numbers will be used.

11.1. Vectorization and parallelization information

The multigrid method MGD5M uses Incomplete Line LU-relaxation(ILLU) as smoothing process. However, this method hardly permits vectorization and parallelization. A new technique to decrease the wall-clock time is to perform the underlying tridiagonal decomposition in a twisted form allowing parallelism. For each grid line a tridiagonal system must be solved. For the ILLU-decomposition and corresponding solution method on a finest grid of N * N grid points, the maximum vector length equals N, which is rather small compared to a maximum vector length of N * N for the ILU-decomposition. Therefore, the twisted form has only be applied at the highest level. Moreover, particular attention has been paid to the influence of parallel overhead; it has been reduced considerably by combining computations on consecutive grid lines.

The prolongation and restriction operators as well as the Galerkin approximations are the same as for MGD1M and, thus, highly parallel. Again, the decomposition of different grids performed in the preparational phase, can be done in parallel. A higher speedup is achieved compared to MGD1M due to the parallel decomposition of the finest grid. For more information on vectorization and parallelization see Louter-Nool[2].

The program has been written in ANSI FORTRAN 77, except for the auxiliary routine DECOMP. A distinction can be made between the parallel and the vector implementation; the latter does not apply macrotasking. Actually, there are two versions of the routine DECOMP: one implementation of DECOMP calls the macrotasking CRAY routines TSKSTART and TSKWAIT in order to perform the ILLU-decomposition of different grids in parallel, the other one is meant for running on a single processor. To get an idea how the wall-clock time can be reduced by parallel execution, the wall-clock times for different phases in the multigrid process are listed in Table 1. For the parallel case the compiler option -Zp is recommended, otherwise the compiler option -Zv is recommended.

	MGD5V	MGD5M					
	Vector	Vector	Speedup	$p = 1$	$p = 2$	$p = 3$	$p = 4$
Galerkin	.130	.037		.037	.019	.013	.010
S_p, $p = 2, 3, 4$			*3.51*		*1.95*	*2.85*	*3.70*
Decompose	.239	.221		.227	.153	.102	.101
S_p, $p = 2, 3, 4$			*1.08*		*1.44*	*2.17*	*2.19*
MG-cycles	1.133	1.045		1.099	.689	.660	.647
S_p, $p = 2, 3, 4$			*1.08*		*1.52*	*1.58*	*1.62*
Total	1.502	1.303		1.357	.861	.775	.758
S_p, $p = 2, 3, 4$			*1.15*		*1.51*	*1.68*	*1.72*

TABLE 1. Wall-Clock time in seconds and speedups for 8 Levels.

12. Keywords

El.iptic PDEs,
Galerkin approximation,
Multigrid methods,
Sparse linear systems,
Twisted LDU-decomposition,
ILLU relaxation.

13. Example

We solve the Poisson equation on the unit square with Dirichlet boundary conditions and the right-hand side constructed according to the exact solution $x\,(1-x) + y\,(1-y)$. In this example the boundary conditions are eliminated.

First we try 3 iterations with a zero initial approximation and if no divergence is found (using the soft fail option) we try to refine the solution until the residual norm becomes lesser than 10^{-9} using the former approximation (computations are performed with 14 decimal digits).

Note the calls to X04AAF and X04ABF prior to the call of MGD5M.

13.1. Program text

```
      PROGRAM EXAMPL
C
C     .. Parameters ..
      INTEGER          WORK, LEVELS, NXC, NYC, NXF, NYF, NM
      PARAMETER( LEVELS = 7, NXC =  5, NYC =  5,
     +                 NXF = 257, NYF = 257, NM = 88399, IWORK = 9*(NXF+(NXF-1)/2)
C
C     .. Scalar Arguments ..
      INTEGER          IFAIL, IPREP, ISTART, MAXIT, NOUT
      REAL             RESNO, TOL
C     ..
C     .. Array Arguments ..
      INTEGER          IOUT( 5 )
      REAL             A, V, RHS, VB,  WORK, LDU
C     ..
```

```
C       .. External Subroutines ..
        EXTERNAL    CPUTIM, MATRHS, MGD5M, TIMING, X04AAF, X04ABF
C       ..
C       .. Common Blocks ..
        COMMON      /ARRAYS/ A( NM*7 ), V( NM ),
       +            RHS( NM ), VB( NM ), WORK( IWORK ),
       +            LDU( NM*3 )
C       ..
C       .. Timing Variables ..
        REAL        CPB, CPE, WCB, WCE
C       ..
C       .. Data Statements ..
        DATA        IOUT /1, 0, 0, 2, 1/
        DATA        NOUT /7/
C       ..
C       .. Executable Statements ..
C
        CALL CPUTIM( CPB )
        CALL TIMING( WCB )
C
        OPEN( UNIT = NOUT ,FILE = 'OUTPUT' )
        CALL X04AAF( 1, NOUT )
        CALL X04ABF( 1, NOUT )
C
        WRITE( NOUT, 9999 )
C
C       Problem set up.
C       Subroutine MATRHS is an example of a subroutine which fills the
C       matrix and the right-hand side.
C
        CALL MATRHS( A, RHS, NXF, NYF, NOUT )
C
C       Approximate the solution of the linear system by performing 3
C       MGD5M iterations, using the soft fail option.
C
        ISTART = 0
        IPREP  = 0
        MAXIT  = 3
        TOL    = 0.0
        IFAIL  = 111
C
        CALL MGD5M( LEVELS, NXC, NYC, NXF, NYF, NM,
       +            A, V, RHS, VB, WORK, LDU, RESNO,
       +            IOUT, ISTART, IPREP, MAXIT, TOL, IFAIL )
C
C       Possible refinement until residual norm .LT. 1.0E-9
C
        IF( IFAIL.GE.7 ) THEN
          ISTART= 2
          IPREP  = 1
          MAXIT= 100
          TOL    = 1.0E-9
          IFAIL  = 111
```

```
C
         CALL MGD5M( LEVELS, NXC, NYC, NXF, NYF, NM,
     +               A, V, RHS, VB, WORK, LDU, RESNO,
     +               IOUT, ISTART, IPREP, MAXIT, TOL, IFAIL )
         END IF
C
         CALL TIMING( WCE )
         CALL CPUTIM( CPE )
C
         WRITE (NOUT, 9998) WCE - WCB, CPE - CPB
C
C        .. Format Statements ..
C
 9999    FORMAT( ' 1 MGD5M Example program results. ' )
 9998    FORMAT( ' Total Wall clock time Example of Use : ', t45, F8.3, /,
     +           ' Total CPU time      Example of Use : ', t45, F8.3 )
C
         STOP
C
C        End of Program EXAMPL
C
         END

         SUBROUTINE MATRHS( A, RHS, NXF, NYF, NOUT )
C
C        .. Scalar Arguments ..
         INTEGER      NXF, NYF, NOUT
C        ..
C        .. Array Arguments ..
         REAL         A( NXF, NYF, 7 ), RHS( NXF, NYF )
C        ..
C-------------------------------------------------------------------
C        MATRHS is a subroutine which fills the matrix and the right-hand
C        side. It is part of the example program.
C
C        The example is the Poisson equation on the unit square with
C        Dirichlet boundary conditions and the exact solution is:
C        X * (XSIZE - X) + Y * (YSIZE - Y).
C        In this example the boundary conditions are eliminated.
C-------------------------------------------------------------------
C        ..
C        .. Local Variables ..
         INTEGER      I, J
         REAL         B, X, XH, XSIZE, XY, Y, YH, YSIZE, YX
         REAL         TWOYX, FOURXY
C        ..
C        .. Parameters ..
         REAL         ZERO, ONE, TWO, FOUR
         PARAMETER    ( ZERO = 0.0E+0, ONE  = 1.0E+0,
     +                  TWO  = 2.0E+0, FOUR = 4.0E+0 )
C
C        .. Intrinsic Functions ..
         INTRINSIC    REAL
```

```
C       ..
C       .. Executable Statements ..
C       ..
C
        XSIZE = ONE
        YSIZE = ONE
        XH = XSIZE / REAL( NXF + 1 )
        YH = YSIZE / REAL( NYF + 1 )
C
C       /(...+1) Because of elimination of boundary conditions
C
        WRITE( NOUT, 9999 ) XSIZE, YSIZE, XH, YH
C
        XY = XH / YH
        YX = YH / XH
C
C       Initial filling of the matrix and the right-hand side neglecting
C       the boundaries.
C
        DO 10 J = 1, NYF
          DO 10 I = 1, NXF
            A( I, J, 1 ) = -XY
            A( I, J, 2 ) = ZERO
            A( I, J, 3 ) = -YX
            A( I, J, 4 ) = TWOYX
            A( I, J, 5 ) = -YX
            A( I, J, 6 ) = ZERO
            A( I, J, 7 ) = -XY
            RHS( I, J ) = FOURXY
   10   CONTINUE
C
C       Correction for the Dirichlet boundary conditions corresponding
C       to the exact solution, X * (XSIZE - X) + Y * (YSIZE - Y)
C
C       Note, that after this correction-process all parts of the
C       difference-molecules outside the domain are initialized to zero!
C
C-------------------------------------------------------------------------
C       Lower and Upper Boundary
C-------------------------------------------------------------------------
        X = ZERO
        DO 20 I = 1, NXF
          X = X + XH
          B = X * ( XSIZE - X )
          RHS( I, 1 ) = RHS( I, 1 ) - A( I, 1 , 1 ) * B
          RHS( I, NYF ) = RHS( I, NYF ) - A( I, NYF, 7 ) * B
          A( I, 1 , 1 ) = ZERO
          A( I, NYF, 7 ) = ZERO
   20   CONTINUE
C-------------------------------------------------------------------------
C       Left and Right-hand Boundary
C-------------------------------------------------------------------------
        Y = ZERO
```

```
            DO 30 J = 1, NYF
              Y = Y + YH
              B = Y * ( YSIZE - Y )
              RHS( 1  , J ) = RHS( 1  , J ) - A( 1  , J, 3 ) * B
              RHS( NXF, J ) = RHS( NXF, J ) - A( NXF, J, 5 ) * B
              A( 1  , J, 3 ) = ZERO
              A( NXF, J, 5 ) = ZERO
   30       CONTINUE
C
C           .. Format Statements ..
C
 9999       FORMAT( ' Poisson Problem: '//' XSIZE  = ',
     +       1PE13.6/' YSIZE  = ',
     +       E13.6/' XH, YH = ', 2E13.6 )
C
            RETURN
C
C           End of Subroutine MATRHS
C
            END
```

13.2. Program data

None.

13.3. Program results

```
 MGD5M Example program results.
 Poisson Problem:

 XSIZE   =   1.000000E+00
 YSIZE   =   1.000000E+00
 XH, YH  =   3.875969E-03 3.875969E-03

 Multigrid program MGD5M, version  9 June 1993

 LEVELS     NXC     NYC     NXF     NYF         NM
      7       5       5     257     257      88399
  MAXIT         TOL
      3         0.000000E+00
                IOUT  ISTART  IPREP  IFAIL
      1  0  0  2  1       0      0    111

 L2-Norm of initial residual =   0.587E+01

 MGD5M, Iteration Number  =       1
 L2-Norm of Residual      =   0.303E-03
 Current Reduction Factor =   0.516E-04
 Average Reduction Factor =   0.516E-04

 MGD5M, Iteration Number  =       2
 L2-Norm of Residual      =   0.125E-04
```

```
Current Reduction Factor =   0.411E-01
Average Reduction Factor =   0.146E-02

MGD5M, Iteration Number  =       3
L2-Norm of Residual      =   0.446E-06
Current Reduction Factor =   0.359E-01
Average Reduction Factor =   0.424E-02

Maxit Iterations performed without reaching TOL.

Good Convergence Rate, so V and VB are valuable,
at this point one may restart MGD5M with ISTART = 2 and IPREP = 1

Error detected by Library routine MGD5M    - IFAIL =  8


CP-time and Wall Clock used ( sec. )

Galerkin        0.010          0.010

Decompose       0.059          0.066

MG-cycles       0.189          0.185

Multigrid program MGD5M, version  9 June 1993

LEVELS    NXC    NYC     NXF     NYF        NM
     7      5      5     257     257     88399
 MAXIT        TOL
   100        0.100000E-08
              IOUT  ISTART  IPREP  IFAIL
     1  0   0  2  1       2      1    111

L2-Norm of initial residual =   0.446E-06

MGD5M, Iteration Number  =       1
L2-Norm of Residual      =   0.178E-07
Current Reduction Factor =   0.400E-01
Average Reduction Factor =   0.400E-01

MGD5M, Iteration Number  =       2
L2-Norm of Residual      =   0.739E-09
Current Reduction Factor =   0.414E-01
Average Reduction Factor =   0.407E-01


CP-time and Wall Clock used ( sec. )

Galerkin        0.000          0.000

Decompose       0.000          0.000

MG-cycles       0.126          0.123
```

```
Total Wall clock time Example of Use :        0.387
Total CPU time        Example of Use :        0.398
```