F11DCFP

NAG Parallel Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details. You are advised to enclose any calls to NAG Parallel Library routines between calls to Z01AAFP and Z01ABFP.

1 Description

Note: you should read the F11 Chapter Introduction before using this routine.

F11DCFP solves a real sparse (unsymmetric) system of linear equations:

$$Ax = b,$$

represented in coordinate storage format and distributed on a logical grid of processors in cyclic row block form. The routine is based on a restarted generalized minimal residual (RGMRES) (Saad and Schultz [1]) method enhanced by block Jacobi preconditioning.

F11DCFP uses, on each logical processor, the incomplete LU factorizations $M_k = P_k L_k D_k U_k Q_k$ of the local diagonal blocks A_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, of A (see Sections 2.6 and 2.7 of the F11 Chapter Introduction), as computed by F11DAFP, to define the block diagonal preconditioning matrix M. A call to F11DCFP must therefore always be preceeded by calls of F11ZAFP and F11DAFP.

F11DCFP also uses F11XBFP to efficiently compute the required matrix-vector multiplications. The routine F11XAFP must be called before F11DCFP to set up, in the array IAINFO, auxiliary information required by F11XBFP.

The matrix A and the preconditioning matrix M are represented in distributed coordinate storage format in the arrays A, IROW and ICOL and C, IROWC and ICOLC. The arrays A and C hold the local non-zero entries in the respective matrices, while IROW, ICOL, IROWC and ICOLC hold the corresponding local row and column indices.

F11DCFP is a black-box routine which calls F11BAFP, F11BBFP and F11BCFP. If you wish to use an alternative storage scheme, preconditioner, matrix-vector multiplication, termination criterion, or require additional diagnostic information, you should call these underlying routines directly.

2 Specification

```
 SUBROUTINE F11DCFP(ICNTXT, METHOD, N, NNZ, A, IROW, ICOL, NNZC, C,
1                   IROWC, ICOLC, IPIVP, IPIVQ, B, M, TOL, MAXITN,
2                   X, RNORM, ITN, IAINFO, WORK, LWORK, IFAIL)
 INTEGER            ICNTXT, N, NNZ, IROW(*), ICOL(*),
1                   NNZC, IROWC(*), ICOLC(*), IPIVP(*), IPIVQ(*),
2                   M, MAXITN, ITN, IAINFO(*), LWORK, IFAIL
 DOUBLE PRECISION   A(*), C(*), B(*), TOL, X(*), RNORM, WORK(LWORK)
 CHARACTER*(*)      METHOD
```

3 Data Distribution

3.1 Definitions

The following definitions are used in describing the data distribution within this document:

m_p – the number of rows in the logical processor grid.
n_p – the number of columns in the logical processor grid.
p – $m_p \times n_p$, the total number of processors.
M_b – the blocking factor for the distribution of the rows of the matrix.
m_l – the number of rows of the matrix assigned to the calling processor (= IAINFO(3), see IAINFO).
n_{int}^i – the number of internal interface indices for the calling processor (see Section 2.6 of the F11 Chapter Introduction) (= IAINFO(6), see IAINFO).
n_{int}^e – the number of external interface indices for the calling processor (see Section 2.6 of the F11 Chapter Introduction) (= IAINFO(7), see IAINFO).
n_{LB} – the number of row blocks assigned to the calling processor (= IAINFO(8), see IAINFO).

3.2 Global and Local Arguments

The input arguments METHOD, N, M, TOL, MAXITN and IFAIL are global and so must have the same value on entry to the routine on each processor. The output arguments RNORM, ITN and IFAIL are global and so will have the same value on exit from the routine on each processor. The remaining arguments are local.

3.3 Distribution Strategy

Blocks of M_b contiguous rows of the matrix A are stored in coordinate storage format on a logical grid of processors cyclically row by row (i.e., in the row major ordering of the grid) starting from the $\{0, 0\}$ logical processor.

The vectors b and x are distributed conformally to the matrix A, i.e., b and x are distributed across the logical processor grid in the same way as each of the columns of the matrix A is distributed. The pivot vectors IPIVP and IPIVQ are distributed in the same way.

These data distributions are described in more detail in Section 2.5 of the F11 Chapter Introduction.

This routine assumes that the data has already been correctly distributed, and if this is not the case will fail to produce correct results. However, the Library provides utility routines, F01YAFP and F01YEFP, which assists you in distributing the data correctly. Descriptions of these routines can be found in Chapter F01 of the NAG Parallel Library

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the BLACS context used by the communication mechanism, usually returned by a call to Z01AAFP.

2: METHOD — CHARACTER*(*) *Global Input*

On entry: specifies the iterative method to be used. The only possible choice at this Release is:

'RGMRES' Restarted Generalized Minimal Residual Method (RGMRES) method.

Constraint: METHOD = 'RGMRES'.

3: N — INTEGER *Global Input*

On entry: the order of the matrix A, n. It must contain the same value as the parameter N used in prior calls of F11ZAFP, F11DAFP and F11XAFP in which the array IAINFO was initialised.

Constraint: N ≥ 1.

4: NNZ — INTEGER *Local Input*

On entry: the number of non-zero elements of matrix A stored on the calling processor. It must contain the same value as the parameter NNZ returned from a prior call of F11ZAFP in which the array IAINFO was initialised.

Constraint: NNZ ≥ 0.

5: A(∗) — DOUBLE PRECISION array *Local Input*

Note: the dimension of the array A must be at least max(1,NNZ).

On entry: the non-zero elements in the blocks of the matrix A assigned to the calling processor. The local non-zero elements must have been reordered by a prior call of F11ZAFP.

6: IROW(∗) — INTEGER array *Local Input*

7: ICOL(∗) — INTEGER array *Local Input*

Note: the dimension of the arrays IROW and ICOL must be at least max(1,NNZ).

On entry: the local row and column indices of the non-zero elements supplied in A. The contents of the arrays IROW and ICOL **must not** be changed between successive calls to library routines involving the matrix A.

8: NNZC — INTEGER array *Local Input*

On entry: the number of non-zero elements in the matrices C_k, $k = 1, 2, \ldots, n_{\mathrm{LB}}$, assigned to the calling processor as returned by a prior call of F11DAFP.

Constraint: NNZC ≥ 0.

9: C(∗) — DOUBLE PRECISION array *Local Input*

Note: the dimension of the array C must be at least max(1,NNZC).

On entry: the values returned in the array C by a prior call to F11DAFP.

10: IROWC(∗) — INTEGER array *Local Input*

11: ICOLC(∗) — INTEGER array *Local Input*

Note: the dimension of the arrays IROWC and ICOLC must be at least max(1,NNZC).

On entry: the local row and column indices of the non-zero elements supplied in C as returned by a prior call to F11DAFP.

12: IPIVP(∗) — INTEGER array *Local Input*

13: IPIVQ(∗) — INTEGER array *Local Input*

Note: the dimension of the arrays IPIVP and IPIVQ must be at least $\max(1, m_l)$.

On entry: the local indices of pivot elements values as returned by a prior call to F11DAFP.

14: B(∗) — DOUBLE PRECISION array *Local Input*

Note: the dimension of the array B must be at least $\max(1, m_l)$.

On entry: the local part of the vector b.

15: M — INTEGER *Global Input*

On entry: the size m of the RGMRES basis (see Section 2.2 of the F11 Chapter Introduction).

Constraint: $0 <$ M $\leq \min$(N,50).

16: TOL — DOUBLE PRECISION *Global Input*

On entry: the required tolerance. Let x_l denote the approximate solution at iteration l, and r_l the corresponding residual. The algorithm is considered to have converged at iteration l if:

$$\|r_l\|_\infty \leq \tau \times (\|b\|_\infty + \|A\|_\infty \|x_l\|_\infty).$$

If TOL ≤ 0.0, $\tau = \max(\sqrt{\epsilon}, \sqrt{n}\,\epsilon)$ is used, where ϵ is the ***machine precision***. Otherwise $\tau = \max(\mathrm{TOL}, 10\epsilon, \sqrt{n}\,\epsilon)$ is used.

Constraint: TOL < 1.0.

17: MAXITN — INTEGER *Global Input*

On entry: the maximum number of iterations allowed.

Constraint: MAXITN ≥ 1.

18: X(∗) — DOUBLE PRECISION array *Local Input/Local Output*

Note: the dimension of the array X must be at least $\max(1, m_l)$.

On entry: an initial approximation to the solution vector x.

On exit: an improved approximation to the solution vector x.

19: RNORM — DOUBLE PRECISION *Global Output*

On exit: the final value of the residual norm $\|r_l\|_\infty$, where l is the output value of ITN.

20: ITN — INTEGER *Global Output*

On exit: the number of iterations carried out.

21: IAINFO(∗) — INTEGER array *Local Input*

Note: the dimension of the array IAINFO must be at least max(2,IAINFO(2)).

On entry: the first IAINFO(2) elements of IAINFO contain auxiliary information about the matrix A. The array IAINFO must be initialised by a prior call of F11ZAFP, and additional information must be stored in IAINFO by prior calls of F11DAFP and F11XAFP. The first IAINFO(2) elements of IAINFO must not be changed between successive calls to library routines involving the matrix A.

22: WORK(LWORK) — DOUBLE PRECISION *Local Workspace*

23: LWORK — INTEGER *Local Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F11DCFP is called.

Constraint: LWORK $\geq (m+3)(m_l+1) + m(m+4) + 2p + m_l + \max(n_{int}^i, n_{int}^e) + 3 + q$, where $q = 0$ in this release.

24: IFAIL — INTEGER *Global Input/Global Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in the Essential Introduction) the recommended values are:

IFAIL = 0, if multigridding is **not** employed;
IFAIL = −1, if multigridding is employed.

On exit: IFAIL = 0 unless the routine detects an error (see Section 5).

5 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output from the root processor (or processor {0,0} when the root processor is not available) on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = −2000

The routine has been called with an invalid value of ICNTXT on one or more processors.

IFAIL = −1000

The logical processor grid and library mechanism (Library Grid) have not been correctly defined, see Z01AAFP.

IFAIL = $-i$

On entry, the ith argument had an invalid value. For global arguments, this may also be caused by an argument not having the same value on all logical processors. An explanatory message distinguishes between these two cases.

IFAIL = 1

IAINFO was not set up by prior calls of F11ZAFP, F11DAFP and F11XAFP.

IFAIL = 2

On entry, the data stored in the arguments N, NNZ, IROW, ICOL and IAINFO is inconsistent. This indicates that, after the array IAINFO was set up by calls of F11ZAFP and F11XAFP, at least one of these arguments was changed between successive calls to library routines.

IFAIL = 3

On entry, the data stored in the arguments N, NNZC, IROWC, ICOLC, IPIVP, IPIVQ and IAINFO is inconsistent. This indicates that, after the array IAINFO was set up by calls of F11ZAFP and F11DAFP, at least one of these arguments was changed between successive calls to library routines.

IFAIL = 4

The required accuracy could not be obtained. However, a reasonable accuracy may have been obtained, and further iterations could not improve the result. You should check the output value of RNORM for acceptability. This error code usually implies that your problem has been fully and satisfactorily solved to within or close to the accuracy available on your system. Further iterations are unlikely to improve on this situation.

IFAIL = 5

The required accuracy could not be obtained in MAXITN iterations.

IFAIL = 6

A serious error has occurred in an internal call to an auxiliary routine. Check all subroutine calls and array sizes. Seek expert help.

6 Further Comments

6.1 Accuracy

On successful termination, the final residual $r_l = b - Ax_l$, where l = ITN, satisfies the termination criterion

$$\|r_l\|_\infty \leq \tau \times (\|b\|_\infty + \|A\|_\infty \|x_l\|_\infty).$$

The value of the final residual norm is returned in RNORM.

6.2 Computational costs

The number of arithmetic operation performed by each processor in each iteration is roughly proportional to the value of NNZC. The number of communication operations depends on the sparsity pattern of the matrix A and the particular row block distribution used.

The number of iterations required to achieve a prescribed accuracy cannot be easily determined *a priori*, as it can depend dramatically on the conditioning and spectrum of the preconditioned coefficient matrix $\bar{A} = M^{-1}A$.

7 References

[1] Saad Y and Schultz M (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **7** 856–869

8 Example

To solve the linear system of equations $Ax = b$ of order $n = n_x^2$, where n_x is a user-specified integer. The symmetric sparse coefficient matrix A is given by the block matrix

$$A = \begin{pmatrix} D & E & 0 & \cdots & & \cdots & 0 \\ F & D & E & \ddots & & & \vdots \\ 0 & F & D & \ddots & & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & & \ddots & D & E & 0 \\ \vdots & & & \ddots & E & D & E \\ 0 & \cdots & & \cdots & 0 & F & D \end{pmatrix},$$

where the matrix blocks D, E and F of order n_x are defined in terms of the quantity $h := (n_x + 1)^{-1}$ as follows:

$$D = \frac{1}{h^2}\begin{pmatrix} 4 & -h-1 & 0 & \cdots & & \cdots & 0 \\ h-1 & 4 & -h-1 & \ddots & & & \vdots \\ 0 & h-1 & 4 & \ddots & & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & & \ddots & 4 & -h-1 & 0 \\ \vdots & & & \ddots & h-1 & 4 & -h-1 \\ 0 & \cdots & & \cdots & 0 & h-1 & 4 \end{pmatrix},$$

$$E = -\frac{h+1}{h^2}\, I$$

and

$$F = \frac{h-1}{h^2}\, I,$$

where I is the identity matrix. The right-hand side vector is given by $b = 10^2(1, 1, \cdots, 1)^T$.

Note: the listing of the Example Program presented below does not give a full pathname for the data file being opened, but in general the user must give the full pathname in this and any other OPEN statement.

8.1 Example Text

```
*     F11DCFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MLMAX, NLBMAX
      PARAMETER        (MLMAX=1000,NLBMAX=10)
      INTEGER          LA, LC
      PARAMETER        (LA=10*MLMAX,LC=2*LA)
      INTEGER          LIA, LWORK
      PARAMETER        (LIA=2*LA,LWORK=20*MLMAX)
*     .. Local Scalars ..
      DOUBLE PRECISION RNORM, TOL
      INTEGER          I, ICNTXT, IFAIL, ITN, K, M, MAXITN, MB, ML, MP,
     +                 MYCOL, MYROW, N, NLB, NNZ, NNZC, NP, NX
      LOGICAL          ROOT
      CHARACTER        DUP, OPTIM, SYMM, ZERO
      CHARACTER*10     METHOD
      CHARACTER*80     FORMAT
*     .. Local Arrays ..
```

```
      DOUBLE PRECISION A(LA), B(MLMAX), C(LC), DTOL(NLBMAX),
     +                 WORK(LWORK), X(MLMAX)
      INTEGER          CA(1), IAINFO(LIA), ICOL(LA), ICOLC(LC), IERR(2),
     +                 IPIVP(MLMAX), IPIVQ(MLMAX), IROW(LA), IROWC(LC),
     +                 LFILL(NLBMAX), NPIVM(NLBMAX), RA(1)
      CHARACTER        MILU(NLBMAX), PSTRAT(NLBMAX)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         BLACS_GRIDINFO, F01YAFP, F01YEFP, F11DAFP,
     +                 F11DCFP, F11XAFP, F11ZAFP, GMAT, GVEC, PRINTI,
     +                 X04YAFP, Z01AAFP, Z01ABFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'F11DCFP Example Program Results'
      END IF
*
*     Open the input file on all processors
*
      OPEN (NIN,FILE='f11dcfpe.d')
*
*     Skip heading in data file
*
      READ (NIN,*)
      READ (NIN,*) MP, NP
*
*     Create the processes and initialize
*
      IFAIL = 0
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
      CALL BLACS_GRIDINFO(ICNTXT,MP,NP,MYROW,MYCOL)
*
*     Initialize some global data
*
      DUP = 'F'
      ZERO = 'R'
      SYMM = 'S'
      OPTIM = 'S'
*
*     Read the problem parameters
*
      READ (NIN,*) NX
      N = NX**2
*
*     Read the algorithmic parameters
*
      READ (NIN,*) METHOD
      READ (NIN,*) M, MB
      READ (NIN,*) TOL, MAXITN
      READ (NIN,*) FORMAT
*
*     Close the input file
*
      CLOSE (NIN)
*
*     Generate the matrix of the coefficients
```

```
*
      CALL F01YAFP(ICNTXT,GMAT,N,MB,NNZ,A,LA,IROW,ICOL,IFAIL)
*
*     Set up auxiliary data for subsequent operations
*
      CALL F11ZAFP(ICNTXT,N,MB,NNZ,A,IROW,ICOL,DUP,ZERO,IAINFO,LIA,
     +             IFAIL)
*
      ML = IAINFO(3)
      NLB = IAINFO(8)
*
*     Check whether number of rows and number of row blocks are
*     less than the corresponding maximum possible values defined by
*     MLMAX and NLBMAX
*
      IERR(1) = 0
      IERR(2) = 0
      IF (ML.GT.MLMAX) IERR(1) = 1
      IF (NLB.GT.NLBMAX) IERR(2) = 1
      CALL IGAMX2D(ICNTXT,'All',' ',1,1,IERR,2,RA,CA,1,-1,-1)
      IF (IERR(1).NE.0) THEN
         IF (ROOT) WRITE (NOUT,FMT=99997)
         GO TO 60
      ELSE IF (IERR(2).NE.0) THEN
         IF (ROOT) WRITE (NOUT,FMT=99996)
         GO TO 60
      END IF
*
*     Generate the right-hand side vector
*
      CALL F01YEFP(ICNTXT,GVEC,N,B,IAINFO,IFAIL)
*
*     Set up auxiliary data for matrix-vector multiplication
*
      CALL F11XAFP(ICNTXT,N,NNZ,A,IROW,ICOL,SYMM,OPTIM,IAINFO,LIA,IFAIL)
*
*     Set up block Jacobi preconditioner
*
      DO 20 K = 1, NLB
         LFILL(K) = -1
         DTOL(K) = 1.D-3
         PSTRAT(K) = 'C'
         MILU(K) = 'M'
   20 CONTINUE

      CALL F11DAFP(ICNTXT,N,NNZ,A,IROW,ICOL,LFILL,DTOL,PSTRAT,MILU,
     +             IPIVP,IPIVQ,NNZC,C,LC,IROWC,ICOLC,NPIVM,IAINFO,LIA,
     +             IFAIL)
*
*     Print a summary of input parameters and options
*
      IF (ROOT) CALL PRINTI(NOUT,METHOD,N,MAXITN,TOL,M)
*
*     Set initial approximation to solution
*
      DO 40 I = 1, ML
         X(I) = 0.D0
   40 CONTINUE
```

```
*
*     Solve the equations
*
      CALL F11DCFP(ICNTXT,METHOD,N,NNZ,A,IROW,ICOL,NNZC,C,IROWC,ICOLC,
     +             IPIVP,IPIVQ,B,M,TOL,MAXITN,X,RNORM,ITN,IAINFO,WORK,
     +             LWORK,IFAIL)
*
*     Produce a report
*
      IF (ROOT) THEN
         WRITE (NOUT,'(/1X,''Summary of results''/1X,18(''-'')/)')
         WRITE (NOUT,99999)
     +     'Number of iterations carried out                    -', ITN
         WRITE (NOUT,99998)
     +     'Residual norm                                       -',
     +     RNORM
         WRITE (NOUT,'(/1X,''Solution vector''/1X,15(''-'')/)')
      END IF
      CALL X04YAFP(ICNTXT,NOUT,N,X,FORMAT,IAINFO,WORK,IFAIL)
*
*     Completion
*
   60 CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
*     End of example program
*
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,A,3X,1P,D9.2)
99997 FORMAT (1X,'** ERROR: Number of rows per processor too large')
99996 FORMAT (1X,'** ERROR: Number of row blocks per processor too ',
     +       'large')
      END
*
************************************************************************
************************************************************************
*
      SUBROUTINE PRINTI(NOUT,METHOD,N,MAXITN,TOL,M)
*
*     Prints a summary of the input parameters and options.
*
*
*     .. Scalar Arguments ..
      DOUBLE PRECISION  TOL
      INTEGER           M, MAXITN, N, NOUT
      CHARACTER*10      METHOD
*     .. Executable Statements ..
      WRITE (NOUT,99999)
      WRITE (NOUT,99998)
     +  'Method used (METHOD)                               -',
     +  METHOD
      WRITE (NOUT,99997)
     +  'Order of the system of equations (N)                -', N
      WRITE (NOUT,99996)
     +  'Tolerance (TOL)                                    -', TOL
      WRITE (NOUT,99997)
     +  'Maximum number of iterations allowed (MAXITN)      -',
```

```
     +  MAXITN
      WRITE (NOUT,99997)
     +  'Dimension of RGMRES orthogonal basis (M)              -', M
*
*     End of subroutine PRINTI
*
      RETURN
*
*
99999 FORMAT (/1X,'Summary of input parameters and options',/1X,39('-'),
     +       /)
99998 FORMAT (1X,A,4X,A)
99997 FORMAT (1X,A,I5)
99996 FORMAT (1X,A,3X,1P,D9.2)
      END
*
************************************************************************
************************************************************************
*
      SUBROUTINE GMAT(I1,I2,N,NNZL,AL,LAL,IROWL,ICOLL)
*
*
*     .. Scalar Arguments ..
      INTEGER          I1, I2, LAL, N, NNZL
*     .. Array Arguments ..
      DOUBLE PRECISION AL(LAL)
      INTEGER          ICOLL(LAL), IROWL(LAL)
*     .. Local Scalars ..
      DOUBLE PRECISION H, H2, H24, HM1, HP1
      INTEGER          I, INZL, IX, IY, J, NX
      LOGICAL          BOUNDX, BOUNDY
*     .. Intrinsic Functions ..
      INTRINSIC        DBLE, DSQRT, MOD, NINT
*     .. Executable Statements ..
      NNZL = 0
      NX = NINT(DSQRT(DBLE(N)))
      H = 1/DBLE(NX+1)
      H2 = DBLE((NX+1)**2)
      H24 = 4*H2
      HM1 = H2*(H-1.D+0)
      HP1 = -H2*(H+1.D+0)
      DO 40 I = I1, I2
*
         INZL = NNZL
*
*     Calculate number of non-zero elements in I-th row
*
         IX = 1 + MOD(I-1,NX)
         IY = 1 + (I-1)/NX
         BOUNDX = (IX.EQ.1) .OR. (IX.EQ.NX)
         BOUNDY = (IY.EQ.1) .OR. (IY.EQ.NX)
         IF (BOUNDX .AND. BOUNDY) THEN
            NNZL = NNZL + 3
         ELSE IF ( .NOT. (BOUNDX .OR. BOUNDY)) THEN
            NNZL = NNZL + 5
         ELSE
            NNZL = NNZL + 4
         END IF
```

```
*
*     Check whether there is sufficient storage space
*
         IF (NNZL.GT.LAL) GO TO 40
*
*     Set non-zero elements in I-th row
*
         DO 20 J = INZL + 1, NNZL
            IROWL(J) = I
   20    CONTINUE
*
         INZL = INZL + 1
         ICOLL(INZL) = I
         AL(INZL) = H24
*
         IF (IY.GT.1) THEN
            INZL = INZL + 1
            ICOLL(INZL) = I - NX
            AL(INZL) = HM1
         END IF
*
         IF (IX.GT.1) THEN
            INZL = INZL + 1
            ICOLL(INZL) = I - 1
            AL(INZL) = HM1
         END IF
*
         IF (IX.LT.NX) THEN
            INZL = INZL + 1
            ICOLL(INZL) = I + 1
            AL(INZL) = HP1
         END IF
*
         IF (IY.LT.NX) THEN
            INZL = INZL + 1
            ICOLL(INZL) = I + NX
            AL(INZL) = HP1
         END IF
*
         NNZL = INZL
*
   40 CONTINUE
*
*     End of subroutine GMAT
*
      RETURN
      END
*
************************************************************************
************************************************************************
*
      SUBROUTINE GVEC(I1,I2,X)
*
*
*     .. Scalar Arguments ..
      INTEGER         I1, I2
*     .. Array Arguments ..
      DOUBLE PRECISION X(*)
```

```
*     .. Local Scalars ..
      INTEGER         I
*     .. Executable Statements ..
      DO 20 I = I1, I2
         X(I-I1+1) = 1.D+2
   20 CONTINUE
*
*     End of subroutine GVEC
*
      RETURN
      END
```

8.2 Example Data

```
F11DCFP Example Program Data
    2   2                        :  MP, NP
    8                            :  NX
  'RGMRES'                       :  METHOD
  10   16                        :  M, MB
   1.0D-06  100                  :  TOL, MAXITN
   '(8F8.4)'                     :  FORMAT
```

8.3 Example Results

```
F11DCFP Example Program Results

Summary of input parameters and options
---------------------------------------

Method used (METHOD)                                    -    RGMRES
Order of the system of equations (N)                    -   64
Tolerance (TOL)                                         -    1.00D-06
Maximum number of iterations allowed (MAXITN)           -  100
Dimension of RGMRES orthogonal basis (M)                -   10

Summary of results
------------------

Number of iterations carried out                    -   13
Residual norm                                       -    1.70D-03

Solution vector
---------------

 2.0231  3.0860  3.5805  3.6951  3.5227  3.1002  2.4225  1.4364
 3.0860  4.7996  5.6149  5.8043  5.5143  4.8090  3.6933  2.1219
 3.5805  5.6149  6.5941  6.8219  6.4696  5.6165  4.2777  2.4241
 3.6951  5.8042  6.8219  7.0595  6.6941  5.8076  4.4167  2.4959
 3.5227  5.5142  6.4696  6.6941  6.3566  5.5311  4.2250  2.4014
 3.1002  4.8090  5.6165  5.8076  5.5311  4.8443  3.7394  2.1571
 2.4225  3.6933  4.2777  4.4167  4.2250  3.7394  2.9382  1.7417
 1.4364  2.1219  2.4241  2.4959  2.4014  2.1571  1.7417  1.0827
```