



## 1 SUMMARY

This routine uses **the Flexible Generalized Minimal Residual method with restarts every  $m$  iterations, **FGMRES**( $m$ ), to solve the  $n \times n$  unsymmetric linear system  $Ax = b$ , optionally using preconditioning. **FGMRES**( $m$ ) generalises the preconditioned **GMRES**( $m$ ) allowing the possibility of using a different right preconditioner  $P_R^{(i)}$  at each step in solving the preconditioned system**

$$\bar{A}x = \bar{b} \quad (1.1)$$

where  $\bar{A} = P_L A$  and  $\bar{b} = P_L b$  and  $P_L$  is a left preconditioner. Reverse communication is used for preconditioning operations and matrix-vector products of the form  $Az$ .

If  $P_R^{(i)}$  does not change at each step  $i$ , in exact arithmetic the algorithm computes the same solution as **GMRES**( $m$ ) applied to (1.1). **FGMRES**( $m$ ) needs more memory than **GMRES**( $m$ ) and, in particular, stores an additional  $n \times m$  real matrix. However, in floating point arithmetic, **FGMRES**( $m$ ) is numerically more stable than **GMRES**( $m$ ) method.

If Gaussian elimination with static pivoting option has been used to compute an approximate  $LU$  factorization of  $A$ , **FGMRES**( $m$ ) can be used to recover full backward error stability. In this case the left preconditioner should be chosen to be the identity and the right preconditioner should be chosen to be  $P_R^{(i)} = P_R = (LU)^{-1}$

**ATTRIBUTES — Version:** 1.3.0 (14 August 2013) **Types:** Real, Double. **Uses:** `_COPY`, `_DOT`, `_NRM2`, `_SCAL`, `_AXPY`, `_ROT`, `_ROTG`, `_TRSV`, `_GEMV` packages. **Date:** April 2007. **Origin:** M. Arioli, Rutherford Appleton Laboratory. **Language:** Fortran 77.

### 1.1 Argument lists and calling sequences

There are two entries:

- (a) MI15I/ID sets default values for control parameters. It should normally be called once prior to calls to MI15A/AD.
- (b) MI15A/AD uses the **FGMRES**( $m$ ) method to solve  $Ax = b$ , optionally using preconditioning. MI15A/AD uses reverse communication for preconditioning operations and matrix-vector products.

### 1.2 To set default values for the control parameters

The single precision version

```
CALL MI15I (ICNTL, CNTL, ISAVE, RSAVE, LSAVE)
```

The double precision version

```
CALL MI15ID (ICNTL, CNTL, ISAVE, RSAVE, LSAVE)
```

ICNTL is an INTEGER array of length 8 that need not be set by the user. On return it contains default values (see Section 1.4 for details).

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 4 that need not be set by the user. On return it contains a default value (see Section 1.4 for details).

ISAVE is an INTEGER array of length 17 used by MI15A/AD as private workspace and must not be altered by the user.

RSAVE is a REAL (DOUBLE PRECISION in the D version) array of length 9 used by MI15A/AD as private workspace and must not be altered by the user.

LSAVE is a LOGICAL array of length 4 used by MI15A/AD as private workspace and must not be altered by the user.

### 1.3 To solve $Ax = b$

The single precision version

```
CALL MI15A(IACT, N, M, W, LDW, Z, LDZ, LOCY, LOCZ, H, LDH, RESID, ICNTL, CNTL, INFO, ISAVE, RSAVE, LSAVE)
```

The double precision version

```
CALL MI15AD(IACT, N, M, W, LDW, Z, LDZ, LOCY, LOCZ, H, LDH, RESID, ICNTL, CNTL, INFO, ISAVE, RSAVE, LSAVE)
```

IACT is an INTEGER variable. Prior to the first call to MI15A/AD, IACT must be set by the user to 0. On each exit, IACT indicates the action required by the user. Possible values of IACT and the action required are as follows:

-1 An error has occurred and the user must terminate the computation (see INFO(1)).

1 If  $ICNTL(4) = 0$  (the default), convergence has been achieved and the user should terminate the computation. If  $ICNTL(4)$  is nonzero, the user may test for convergence. If the user does not wish to test for convergence (we do not recommend the user tests for convergence each time  $IACT=1$  is returned) or if convergence has not been achieved, the user must recall MI15A/AD without changing any of the arguments.

2 The user must perform the matrix-vector product

$$y = Az \tag{1.2}$$

and recall MI15A/AD. The vectors  $y$  and  $z$  are held in columns LOCY and LOCZ of array W, respectively.

3 The user must perform the preconditioning operation

$$y = P_L z \tag{1.3}$$

where  $P_L$  is the left preconditioning matrix, and recall MI15A/AD. The vectors  $y$  and  $z$  are held in the first  $N$  entries of columns LOCY and LOCZ of array W, respectively. This return is only possible if the user has requested preconditioning from the left ( $ICNTL(3)=1$ ) or from both sides ( $ICNTL(3)=3$ ) (see ICNTL).

4 The user must perform the preconditioning operation

$$y = P_R^{(i)} z \tag{1.4}$$

where  $P_R^{(i)}$  is the right preconditioning matrix, and recall MI15A/AD. The vectors  $y$  and  $z$  are held in the first  $N$  entries of columns LOCY and LOCZ of array W, respectively. This return is only possible if the user has requested preconditioning from the right ( $ICNTL(3)=2$ ) or from both sides ( $ICNTL(3)=3$ ) (see ICNTL).

N is an INTEGER variable that must be set by the user to  $n$ , the order of the matrix  $A$ . This variable must be preserved by the user between calls to MI15A/AD. This argument is not altered by the routine. **Restriction:**  $N \geq 1$ .

M is an INTEGER variable that must be set by the user to  $m$ , the maximum number of iterations performed by **FGMRES(m)** between restarts. A compromise between a large value of  $m$  which reduces the overall number of iterations and a small value which limits the storage required must be sought. Unfortunately, it is hard to make firm recommendations about a suitable value as a good value is problem dependent. This variable must be preserved by the user between calls to MI15A/AD. This argument is not altered by the routine. **Restriction:**  $M \geq 1$ .

$W$  is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LDW, M+7). Prior to the first call, the first N entries of column 1 must be set to hold the right-hand side vector  $b$  and, if ICNTL(5) is nonzero, the first N entries of column 2 must be set to the initial estimate of the solution vector  $x$ . On exit with IACT= 1, the first N entries of column 1 hold the current residual vector  $r = b - Ax$ , and the current estimate of the solution  $x$  is held in the first N entries of column 2. On exit with IACT> 1, the user is required to form  $y$  in column LOCY of  $W$  (see argument IACT). The remaining columns of  $W$  must not be altered by the user between calls to MI15A/AD.

LDW is an INTEGER variable that must be set by the user to the first dimension of the array  $W$ . This argument is not altered by the routine. **Restriction:** LDW>N.

$Z$  is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LDZ,M). The user need not set  $Z$  and the columns of  $Z$  must not be altered by the user between calls to MI15A/AD.

LDZ is an INTEGER variable that must be set by the user to the first dimension of the array  $W$ . This argument is not altered by the routine. **Restriction:** LDZ>N.

LOCY, LOCZ are INTEGER variables that need not be set by the user. On exit with IACT > 1, they indicate which columns of  $W$  contain the vectors  $y$  and  $z$  (see argument IACT). These arguments must not be altered by the user between calls to MI15A/AD.

$H$  is a REAL (DOUBLE PRECISION in the D version) two-dimensional workspace array with dimensions (LDH, M + 2). The user need not set  $H$  and it must not be altered by the user between calls to MI15A/AD.

LDH is an INTEGER variable that must be set by the user to the first dimension of the array  $H$ . This argument is not altered by the routine. **Restriction:** LDH $\geq$ M+1.

RESID is a REAL (DOUBLE PRECISION in the D version) variable that need not be set by the user. On exit with IACT = 1, RESID holds the 2-norm of the current residual vector  $\|b - Ax\|_2$ , where  $x$  is the current estimate of the solution.

ICNTL is an INTEGER array of length 8 that contains control parameters. Default values for the components may be set by a call to MI15I/ID. Details of the control parameters are given in Section 1.4. This argument is not altered by the routine.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 4 that contains control parameters and must be set by the user. Default values for the components may be set by a call to MI15I/ID. Details of the control parameter is given in Section 1.4. This argument is not altered by the routine.

INFO is an INTEGER array of length 4 that need not be set by the user. It is used to hold information about the execution of the subroutine. On exit from MI15A/AD, a value for INFO(1) of zero indicates that the subroutine has performed successfully. For nonzero values, see Section 1.5. INFO(2) holds the number of iterations performed. INFO(3) and INFO(4) are not currently used.

ISAVE is an INTEGER array of length 17 used by MI15A/AD as private workspace and must not be altered by the user.

RSAVE is a REAL (DOUBLE PRECISION in the D version) array of length 9 used by MI15A/AD as private workspace and must not be altered by the user.

LSAVE is a LOGICAL array of length 4 used by MI15A/AD as private workspace and must not be altered by the user.

## 1.4 Control parameters

The elements of the arrays ICNTL and CNTL control the action of MI15A/AD. Default values may be set by calling MI15I/ID.

ICNTL(1) is the stream number for error messages and has the default value 6. Printing of error messages is suppressed if  $\text{ICNTL}(1) \leq 0$ .

ICNTL(2) is the stream number for warning messages and has the default value 6. Printing of warning messages is suppressed if  $\text{ICNTL}(2) \leq 0$ .

ICNTL(3) controls whether the user wishes to use preconditioning. It has default value 0 and in this case no preconditioning is used. If ICNTL(3) is nonzero, the user will be expected to perform preconditioning. Possible values are

- 1 Preconditioning from the left.
- 2 Preconditioning from the right.
- 3 Preconditioning from the both sides.

ICNTL(4) controls whether the convergence test offered by MI15A/AD is to be used. It has default value 0 and in this case the computed solution  $x$  is accepted if  $\|b - Ax\|_2$  is less than or equal to  $\max(\|b - Ax^{(0)}\|_2 * \text{CNTL}(1), \text{CNTL}(2))$ , where  $x^{(0)}$  is the initial estimate of the solution. If the user does not want to use this test for convergence, ICNTL(4) should be nonzero. In this case, the user may test for convergence when IACT = 1 is returned.

ICNTL(5) controls whether the user wishes to supply an initial estimate of the solution vector  $x$ . It has default value 0 and in this case  $x = (0, 0, \dots, 0)^T$  is used as the initial estimate. If the user wishes to supply an initial estimate, ICNTL(5) should be nonzero and the initial estimate placed in the first N entries of column 2 of the array W prior to the first call to MI15A/AD.

ICNTL(6) determines the maximum number of iterations allowed. It has default value -1 and in this case the maximum number of iterations allowed is  $2n$ . If the user does not want the maximum to be  $2n$ , ICNTL(6) should be set to the maximum number of iterations the user does wish to allow. Values of ICNTL(6) which are less than or equal to zero are treated as if they were the default -1.

ICNTL(7) and ICNTL(8) are set to zero by MI15I/ID but not currently used by MI15A/AD.

CNTL(1) and CNTL(2) are the convergence tolerances (see Section 3). CNTL(1) has default value  $\sqrt{\epsilon}$ , where  $\epsilon$  is the relative machine precision (that is, the smallest machine number such that  $1 + \epsilon > 1$ , while CNTL(2) has default value zero. If ICNTL(4) is nonzero, CNTL(1) and CNTL(2) are not accessed by MI15A/AD.

CNTL(3) and CNTL(4) are set to zero by MI15I/ID but not currently used by MI15A/AD.

## 1.5 Error diagnostics

If MI15A/AD returns with a negative value of INFO(1), an error has occurred; if MI15A/AD returns with a positive value of INFO(1), a warning has been issued. Error messages are output on unit ICNTL(1) and warnings on unit ICNTL(2). Possible non-zero values of INFO(1) are given below.

-1  $N < 1$ . Immediate return with input parameters unchanged.

-2  $M < 1$ . Immediate return with input parameters unchanged.

- 3  $LDW < N$ . Immediate return with input parameters unchanged.
- 4  $LDH < M + 1$ . Immediate return with input parameters unchanged.
- 5 The maximum number of iterations determined by the control parameter  $ICNTL(6)$  has been exceeded.
- +1 The user-supplied convergence tolerance  $CNTL(1)$  lies outside the interval  $(\epsilon, 1.0)$ , where  $\epsilon$  is the relative machine precision.  $CNTL(1)$  is reset to the default convergence tolerance  $\epsilon$ .

### 1.6 Underflows

The nature of the calculations performed in this subroutine means that underflows are likely to occur. It is quite safe to set numbers that underflow to zero, and action by the user may be required to ensure that this is done efficiently by the computing system in use.

## 2 GENERAL INFORMATION

**Other routines called directly:** MI15A/AD calls the BLAS kernels `_NRM2`, `_COPY`, `_AXPY`, `_SCAL`, `_DOT`, `_ROT`, `_ROTG`, `_TRSV`, `_GEMV`.

**Input/output:** Error messages are printed on unit  $ICNTL(1)$  and warnings on unit  $ICNTL(2)$ ; see Section 1.5.

**Restrictions:**  $N \geq 1, M \geq 1, LDW \geq N, LDH \geq M + 1$ .

## 3 METHOD

The Flexible Generalized Minimal Residual method is due to Saad (1993). The method is described in detail by J. J. Dongarra, I. S. Duff, D. C. Sorensen and H. A. van der Vorst (1998), Section 8.2.7. An roundoff error analysis of **FGMRES** with right preconditioning can be found in M. Arioli, I. S. Duff, S. Gratton, and S. Pralet (2006). The salient features of the algorithm used by MI15A/AD as far as the user is concerned are as follows:

Check the input data for errors. Set  $INFO(1)$  and return with  $IACT = -1$  if a fatal error is encountered.

**if**  $ICNTL(5)$  **is nonzero**

let  $x^{(0)}$  be the initial guess supplied by the user

**else**

set  $x^{(0)} = (0, \dots, 0)^T$

**end if**

**if**  $ICNTL(6) > 0$

$ITMAX = ICNTL(6)$

**else**

$ITMAX = N$

**end if**

set the iteration count  $k = 0$

**do**  $j = 1, 2, \dots, ITMAX$

Return to the user with  $IACT = 2$  to obtain  $Ax^{(k)}$ .

Compute the residual  $r^{(k)} = b - Ax^{(k)}$ .

Return to the user with IACT= 3 to obtain  $r = P_L r^{(k)}$ .

**if** ICNTL(4) is zero **then**,

**if**  $\|r^{(k)}\|_2 \leq \max(\|r^{(0)}\|_2 * \text{CNTL}(1), \text{CNTL}(2))$  convergence has been achieved. Return with IACT= 1.

**else**

Return with IACT= 1 to allow the user to check for convergence.

The recommended criteria for the right preconditioned version of **FGMRES** is

$\|r^{(k)}\|_2 \leq \epsilon(\|A\|_2 \|x^{(k)}\|_2 + \|b\|_2)$

**end if**

Compute  $v_1 = r / \|r\|_2$

Set  $s_1 = r / \|r\|_2$

set  $l = k$

**do**  $i = 1, 2, \dots, m$

set  $l = l + 1$

**if**  $l > \text{ITMAX}$ ,

set INFO(1) and return with IACT= -1.

**end if**

Return to the user with IACT= 4 to obtain  $z = P_R^{(i)} v_i$ .

Return to the user with IACT= 2 to obtain  $u = Az$ .

Return to the user with IACT= 3 to obtain  $w = P_L u$ .

Orthogonalize  $w$  against  $v_1, \dots, v_i$  using the modified Gram-Schmidt process.

Compute  $v_{i+1} = w / \|w\|_2$ .

Form a trapezoidal matrix  $H$  that is a basis for the Krylov subspace spanned by  $v_i, \dots, v_{i=1}$ .

Use  $H$  to calculate the residual of the vector

$x^{(k)} + y_i$ , where  $y_i$  lies in the Krylov subspace generated by

$Z = (z_1, \dots, z_i)$  and is selected to minimize

$\|P_L(b - A(x^{(k)} + y))\|$ .

If this residual is small, exit the  $i$ -loop

**end do**

Set  $x^{(1)} = x^{(k)} + Zy_i$  and  $k = k + 1$ .

**end do**

## References

- M. Arioli, I. S. Duff, S. Gratton, and S. Pralet (2006). A note on GMRES preconditioned by a perturbed  $LDL^T$  decomposition with static pivoting. RAL-TR-2006-007.
- Y. Saad (1993). A flexible inner-outer preconditioned GMRES algorithm. SIAM J. Sci. Stat. Comput., 14, 461-469.
- J. J. Dongarra, I. S. Duff, D. C. Sorensen and H. A. van der Vorst (1998). Numerical Linear Algebra for High-Performance Computers. SIAM, Philadelphia, USA.

#### 4 EXAMPLE OF USE

The following program illustrates the calling sequence for MI15 in order to solve the linear system  $Ax = b$ , where

$$A = \begin{bmatrix} 2 & 1 & & & & & & & & & \\ -1 & 2 & 1 & & & & & & & & \\ & -1 & 2 & 1 & & & & & & & \\ & & -1 & 2 & 1 & & & & & & \\ & & & -1 & 2 & 1 & & & & & \\ & & & & -1 & 2 & 1 & & & & \\ & & & & & -1 & 2 & 1 & & & \\ & & & & & & -1 & 2 & 1 & & \\ & & & & & & & -1 & 2 & 1 & \\ & & & & & & & & -1 & 2 & \\ & & & & & & & & & -1 & 2 \end{bmatrix} \quad b = \begin{bmatrix} 3 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

The inverse of the diagonal of  $A$  is used to left precondition and 5-steps of Jacobi method are used to right precondition.

```

PROGRAM MAIN
  INTEGER N, LDW, M, LDH, I, J, K, IACT, LOCY, LOCZ, LDZ
  PARAMETER ( N = 10, M = 5, LDW = N, LDZ = N, LDH = M + 1 )
  DOUBLE PRECISION ZERO, TWO, ONE, THREE, RESID
  PARAMETER ( ZERO=0.0D0, TWO=2.0D+0, ONE=1.0D+0, THREE=3.0D+0 )
  INTEGER ICNTL( 8 ), INFO( 4 )
  DOUBLE PRECISION CNTL( 4 ), W( LDW, M + 7 ), H( LDH, M + 2 )
  DOUBLE PRECISION Z( LDZ, M)
  INTEGER ISAVE(17)
  DOUBLE PRECISION RSAVE(9)
  LOGICAL LSAVE(4)
  EXTERNAL MI15AD, MI15ID

C
  CALL MI15ID( ICNTL, CNTL, ISAVE, RSAVE, LSAVE )
C Precondition from both sides
  ICNTL( 3 ) = 3
  ICNTL( 6 ) = 100
C Set right hand side, b
  W( 1, 1 ) = THREE
  DO 10 I = 2, N - 1
    W( I, 1 ) = TWO
  10 CONTINUE
  W( N, 1 ) = ONE
C Solve the system using the Flex-GMRES(m) method
  IACT = 0
  DO 80 K = 1, 2*N
    CALL MI15AD( IACT, N, M, W, LDW, Z, LDZ, LOCY, LOCZ, H, LDH,
  *             RESID, ICNTL, CNTL, INFO, ISAVE, RSAVE, LSAVE )
C
  IF ( IACT .LT. 0 ) THEN
C ERROR
  WRITE( 6, '(A,I2,A)' ) ' Error return: INFO( 1 ) = ',
  *                       INFO( 1 ), ' on exit '
  STOP

```

```

C
      ELSEIF ( IACT .EQ. 1 ) THEN
C Solution found
      WRITE( 6, '(I6,A, // A, / (5F8.3))' ) INFO( 2 ),
*           ' iterations required by MI15 ',
*           ' Solution = ', ( W( I, 2 ), I = 1, N )
      IF ( INFO( 1 ) .GT. 0 ) THEN
        WRITE( 6, '(A,I2,A)' ) ' Warning: INFO( 1 ) = ',
*                               INFO( 1 ), ' on exit '
      ENDIF
      STOP
C
      ELSEIF ( IACT .EQ. 2 ) THEN
C Perform the matrix-vector product
      W( 1, LOCY ) = TWO * W( 1, LOCZ ) + W( 2, LOCZ )
      DO 30 I = 2, N - 1
        W( I, LOCY ) = - W( I - 1, LOCZ ) + TWO * W( I, LOCZ ) +
*                       W( I + 1, LOCZ )
      30 CONTINUE
      W( N, LOCY ) = - W( N - 1, LOCZ ) + TWO * W( N, LOCZ )
C
      ELSEIF ( IACT .EQ. 3 ) THEN
C Perform the left preconditioning operation
      DO 40 I = 1, N
        W( I, LOCY ) = W( I, LOCZ ) / TWO
      40 CONTINUE
C
      ELSEIF ( IACT .EQ. 4 ) THEN
C Perform the right preconditioning operation
      DO 70 I = 1, N
        W( I, LOCY ) = ZERO
      70 CONTINUE
      DO 60 J = 1, 5
        W( 1, LOCY ) = ( W( 1, LOCZ ) - W( 2, LOCY ) ) / TWO
        DO 50 I = 2, N-1
          W( I, LOCY ) = ( W( I, LOCZ ) + W( I-1, LOCY ) -
*                       W( I+1, LOCY ) ) / TWO
        50 CONTINUE
        W( N, LOCY ) = ( W( N, LOCZ ) + W( N-1, LOCY ) ) / TWO
      60 CONTINUE
C
      END IF
      80 CONTINUE
      END

```

This produces the following output:

```

5 iterations required by MI15

Solution =

```



1.000 1.000 1.000 1.000 1.000  
1.000 1.000 1.000 1.000 1.000