MI24

PACKAGE SPECIFICATION HSL

1 SUMMARY

This routine uses **the Generalized Minimal Residual method with restarts every m iterations, GMRES(m), to solve the $n \times n$ unsymmetric linear system $\mathbf{Ax} = \mathbf{b}$, optionally using preconditioning.** If $\mathbf{P}_L$, $\mathbf{P}_R$ are left and right preconditioning matrices, the routine actually solves the preconditioned system

$$\overline{\mathbf{A}}\overline{\mathbf{x}} = \overline{\mathbf{b}},$$

with $\overline{\mathbf{A}} = \mathbf{P}_L \mathbf{A} \mathbf{P}_R$ and $\overline{\mathbf{b}} = \mathbf{P}_L \mathbf{b}$. The solution may be recovered as $\mathbf{x} = \mathbf{P}_R \overline{\mathbf{x}}$. If $\mathbf{P}_L = \mathbf{I}$, preconditioning is said to be from the right, if $\mathbf{P}_R = \mathbf{I}$, it is said to be from the left, and otherwise it is from both sides. Reverse communication is used for preconditioning operations and matrix-vector products of the form $\mathbf{Az}$.

ATTRIBUTES — Version: 1.3.1. (2 December 2015) **Types:** Real (single, double). **Calls:** `_COPY`, `_DOT`, `_NRM2`, `_SCAL`, `_AXPY`, `_ROT`, `_ROTG`, `_TRSV`, `_GEMV`. **Language:** Fortran 77. **Original date:** March 2001. **Origin:** N.I.M. Gould and J.A. Scott, Rutherford Appleton Laboratory. **Remark:** This is a threadsafe version of `MI04A` and supersedes it.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequence

There are two entries:

(a) `MI24I/ID` sets default values for control parameters. It should normally be called once prior to calls to `MI24A/AD`.

(b) `MI24A/AD` uses the GMRES(m) method to solve $\mathbf{Ax} = \mathbf{b}$, optionally using preconditioning. `MI24A/AD` uses reverse communication for preconditioning operations and matrix-vector products.

2.1.1 To set default values for the control parameters

The single precision version

```
CALL MI24I(ICNTL,CNTL,ISAVE,RSAVE,LSAVE)
```

The double precision version

```
CALL MI24ID(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 2.2 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 2.2 for details).

`ISAVE` is an `INTEGER` array of length `17` used by `MI24` 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 `MI24` as private workspace and must not be altered by the user.

`LSAVE` is a `LOGICAL` array of length `4` used by `MI24` as private workspace and must not be altered by the user.

2.1.2 To solve $\mathbf{Ax}=\mathbf{b}$

The single precision version

```
CALL MI24A(IACT,N,M,W,LDW,LOCY,LOCZ,H,LDH,RESID,ICNTL,CNTL,INFO,ISAVE,RSAVE,LSAVE)
```

The double precision version

```
CALL MI24AD(IACT,N,M,W,LDW,LOCY,LOCZ,H,LDH,RESID,ICNTL,CNTL,INFO,ISAVE,RSAVE,LSAVE)
```

`IACT` is an `INTEGER` variable. Prior to the first call to `MI24A/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 `MI24A/AD` without changing any of the arguments.

2 The user must perform the matrix-vector product

$$\mathbf{y} := \mathbf{Az} \tag{2.1}$$

and recall `MI24A/AD`. The vectors $\mathbf{y}$ and $\mathbf{z}$ are held in columns `LOCY` and `LOCZ` of array `W`, respectively.

3 The user must perform the preconditioning operation

$$\mathbf{y} := \mathbf{P}_L\mathbf{z}, \tag{2.2}$$

where $\mathbf{P}_L$ is the left preconditioning matrix, and recall `MI24A/AD`. The vectors $\mathbf{y}$ and $\mathbf{z}$ 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

$$\mathbf{y} := \mathbf{P}_R\mathbf{z}, \tag{2.3}$$

where $\mathbf{P}_R$ is the right preconditioning matrix, and recall `MI24A/AD`. The vectors $\mathbf{y}$ and $\mathbf{z}$ 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 $\mathbf{A}$. This variable must be preserved by the user between calls to `MI24A/AD`. This argument is not altered by the routine. **Restriction**: `N` ≥ `1`.

`M` is an `INTEGER` variable that must be set by the user to m, the maximum number of iterations performed by GMRES(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 `MI24A/AD`. This argument is not altered by the routine. **Restriction**: `M` ≥ `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 $\mathbf{b}$ and, if `ICNTL(5)` is nonzero, the first `N` entries of column 2 must be set to the initial estimate of the solution vector $\mathbf{x}$. On exit with `IACT` = `1`, the first `N` entries of column 1 hold the current residual vector $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$, and the current estimate of the solution $\mathbf{x}$ is held in the first `N` entries of column 2. On exit with `IACT` > `1`, the user is required to form $\mathbf{y}$ in column `LOCY` of `W` (see argument `IACT`). The remaining columns of `W` must not be altered by the user between calls to MI24A/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 $\geq$ 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 MI24A/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 MI24A/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 $\|\mathbf{b}-\mathbf{Ax}\|_2$, where $\mathbf{x}$ 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 MI24I/ID. Details of the control parameters are given in Section 2.2. 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 MI24I/ID. Details of the control parameter is given in Section 2.2. 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 MI24A/AD, a value for INFO(1) of zero indicates that the subroutine has performed successfully. For nonzero values, see Section 2.3. 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 MI24 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 MI24 as private workspace and must not be altered by the user.

LSAVE is a LOGICAL array of length 4 used by MI24 as private workspace and must not be altered by the user.

2.2 Control parameters

The elements of the arrays ICNTL and CNTL control the action of MI24A/AD. Default values may be set by calling MI24I/ID.

ICNTL(1) is the stream number for error messages and has the default value 6. Printing of error messages is suppressed if 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 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 MI24A/AD is to be used. It has default value 0 and in this case the computed solution $\mathbf{x}$ is accepted if $\|\mathbf{b}-\mathbf{Ax}\|_2$ is less than or equal to max($\|\mathbf{b}-\mathbf{Ax}^{(0)}\|_2$ *CNTL(1), CNTL(2)), where $\mathbf{x}^{(0)}$ 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 $\mathbf{x} = (0, 0, ..., 0)^T$ 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 MI24A/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.

ICNTRL(7) and ICNTRL(8) are set to zero by MI24I/ID but not currently used by MI24A/AD.

CNTL(1) and CNTL(2) are the convergence tolerances (see Section 4). CNTL(1) has default value $\sqrt{u}$, where u is the relative machine precision (that is, the smallest machine number such that $1 + u > 1$), while CNTL(2) has default value zero. If ICNTL(4) is nonzero, CNTL(1) and CNTL(2) are not accessed by MI24A/AD.

CNTRL(3) and CNTRL(4) are set to zero by MI24I/ID but not currently used by MI24A/AD.

2.3 Error diagnostics

If MI24A/AD returns with a negative value of INFO(1), an error has occurred; if MI24A/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 (u,1.0), where u is the relative machine precision. CNTL(1) is reset to the default convergence tolerance u.

2.4 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.

3 GENERAL INFORMATION

Use of common: None.

Other routines called directly: MI24A/AD calls the BLAS kernels SNRM2/DNRM2, SCOPY/DCOPY, SAXPY/DAXPY, SSCAL/DSCAL, SDOT/DDOT, SROT/DROT, SROTG/DROTG, STRSV/DTRSV, SGEMV/DGEMV.

Input/output: Error messages are printed on unit ICNTL(1) and warnings on unit ICNTL(2); see Section 2.3.

Restrictions:

N ≥ 1, M ≥ 1, LDW ≥ N, LDH ≥ M+1.

4 METHOD

The Generalized Mimimal Residual method is due to Saad and Schultz (1986). The method is described in detail by J. J. Dongarra, I. S. Duff, D. C. Sorensen and H. A. van der Vorst (1991), Section 7.1.5. The salient features of the algorithm used by `MI24A/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 $\mathbf{x}^{(0)}$ be the initial guess supplied by the user
else
    set $\mathbf{x}^{(0)} = (0, 0, ..., 0)^{\mathrm{T}}$
end if
if `ICNTL(6)` is greater then zero
    `ITMAX` = `ICNTL(6)`
else
    `ITMAX` = `N`
end if
set the iteration count $k = 0$
do $j = 1, 2, ...,$ `ITMAX`
    Return to the user with `IACT` $= 2$ to obtain $\mathbf{Ax}^{(k)}$.
    Compute the residual $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{Ax}^{(k)}$·
    Return to the user with `IACT` $= 3$ to obtain $\mathbf{r}{=}\mathbf{P}_L \mathbf{r}^{(k)}$·
    if `ICNTL(4)` is zero,
        if $\|\mathbf{r}^{(k)}\|_2 \le \max(\|\mathbf{r}^{(0)}\|_2 *$`CNTL(1)`,`CNTL(2)`) convergence has been achieved. Return with `IACT` $= 1$.
    else
        Return with `IACT` $= 1$ to allow the user to check for convergence.
    end if
    Compute $\mathbf{v}_1 {=} \mathbf{r} / \|\mathbf{r}\|_2$
    Set $\mathbf{s}_1 {=} \mathbf{r} / \|\mathbf{r}\|_2$
    set $l = k$
    do $i = 1, 2, ...,$ m
        set $l = l{+}1$
        if $l >$ `ITMAX`, set `INFO(1)` and return with `IACT` $= -1$.
        Return to the user with `IACT` $= 4$ to obtain $\mathbf{z} = \mathbf{P}_R \mathbf{v}_i$·
        Return to the user with `IACT` $= 2$ to obtain $\mathbf{u} = \mathbf{Az}$.
        Return to the user with `IACT` $= 3$ to obtain $\mathbf{w} = \mathbf{P}_L \mathbf{u}$.
        Orthogonalize $\mathbf{w}$ against $\mathbf{v}_1 ... \mathbf{v}_i$ using the modified Gram-Schmidt process.
        Compute $\mathbf{v}_{i+1} = \mathbf{w} / \|\mathbf{w}\|$.
        Form a Trapezoidal matrix $\mathbf{H}$ which is a basis for the Krylov subspace spanned by $\mathbf{v}_1 ... \mathbf{v}_{i+1}$.
        Use $\mathbf{H}$ to calculate the residual of the vector $\mathbf{x}^{(k)} + \mathbf{y}_i$, where $\mathbf{y}_i$ lies in the Krylov subspace and is selected to minimize $\|\mathbf{P}_L(\mathbf{b} - \mathbf{A}(\mathbf{x}^{(k)} + \mathbf{y}))\|$·
        If this residual is small, exit the i loop
    end do
    Set $\mathbf{x}^{(l)} = \mathbf{x}^{(k)} + \mathbf{y}_i$ and $k = k + i$
end do

References

Y. Saad and M. H. Schultz (1986). GMRES: a generalized minimal residual method for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput., **7**, 856-869.

J. J. Dongarra, I. S. Duff, D. C. Sorensen and H. A. van der Vorst (1991). Solving Linear Systems on Vector and Shared Memory Computers. SIAM, Philadelphia, USA.

5 EXAMPLE OF USE

The following program illustrates the calling sequence for MI24.

```
      PROGRAM MAIN
C
C  Solve the linear system A x = b, where A is tridiagonal with
C  superdiagonal values 1, subdiagonals -1 and diagonals 2, and where
C  the square roots of the inverse of the diagonal of A are used to
C  precondition from both sides
C
      INTEGER N, LDW, M, LDH, I, IACT, LOCY, LOCZ
      PARAMETER ( N = 10, M = 5, LDW = N, LDH = M + 1 )
      REAL TWO, ONE, THREE, SQRTWO, RESID
      PARAMETER ( TWO = 2.0E+0, ONE = 1.0E+0, THREE = 3.0E+0 )
      INTEGER ICNTL( 8 ), INFO( 4 )
      REAL CNTL( 4 ), W( LDW, M + 7 ), H( LDH, M + 2 )
      INTEGER ISAVE(17)
      REAL RSAVE(9)
      LOGICAL LSAVE(4)
      EXTERNAL MI24A, MI24I
      SQRTWO = SQRT( TWO )
      CALL MI24I( ICNTL, CNTL, ISAVE, RSAVE, LSAVE )
C
C  Precondition from both sides
C
      ICNTL( 3 ) = 3
      ICNTL( 6 ) = 100
C
C  Set right hand side, b
C
      W( 1, 1 ) = THREE
      DO 10 I = 2, N - 1
         W( I, 1 ) = TWO
   10 CONTINUE
      W( N, 1 ) = ONE
C
C  Perform an iteration of the GMRES(m) method
C
      IACT = 0
   20 CONTINUE
      CALL MI24A( IACT, N, M, W, LDW, LOCY, LOCZ, H, LDH, RESID,
     *             ICNTL, CNTL, INFO, ISAVE, RSAVE, LSAVE )
      IF ( IACT .LT. 0 ) THEN
         WRITE( 6, 2020 ) INFO( 1 )
         GO TO 60
      END IF
C
C  Perform the matrix-vector product
C
      IF ( IACT .EQ. 2 ) THEN
         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 )
          GO TO 20
       END IF
C
C  Perform the left preconditioning operation
C
       IF ( IACT .EQ. 3 ) THEN
          DO 40 I = 1, N
             W( I, LOCY ) = W( I, LOCZ ) / SQRTWO
   40     CONTINUE
          GO TO 20
       END IF
C
C  Perform the right preconditioning operation
C
       IF ( IACT .EQ. 4 ) THEN
          DO 50 I = 1, N
             W( I, LOCY ) = W( I, LOCZ ) / SQRTWO
   50     CONTINUE
          GO TO 20
       END IF
C
C  Solution found
C
       WRITE( 6, 2000 ) INFO( 2 ), ( W( I, 2 ), I = 1, N )
       IF ( INFO( 1 ) .GT. 0 ) WRITE( 6, 2010 ) INFO( 1 )
   60 CONTINUE
       STOP
 2000 FORMAT( I6,' iterations required by MI24 ', // ' Solution = ',
     *        / ( 1P, 5E10.2 ) )
 2010 FORMAT( ' Warning: INFO( 1  ) = ',I2,' on exit ' )
 2020 FORMAT( ' Error return: INFO( 1  ) = ',I2,' on exit ' )
       END
```

This produces the following output:

```
    23 iterations required by MI24

 Solution =
  1.00D+00  1.00D+00  1.00D+00  1.00D+00  1.00D+00
  1.00D+00  1.00D+00  1.00D+00  1.00D+00  1.00D+00
```