1 SUMMARY

This routine uses **the Conjugate Gradient method to solve the $n \times n$ symmetric positive definite linear system $\mathbf{Ax} = \mathbf{b}$, optionally using preconditioning.** If $\mathbf{PP}^T$ is the preconditioning matrix, the routine actually solves the preconditioned system

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

with $\overline{\mathbf{A}} = \mathbf{PAP}^T$ and $\overline{\mathbf{b}} = \mathbf{Pb}$ and recovers the solution $\mathbf{x} = \mathbf{P}^T\overline{\mathbf{x}}$. Reverse communication is used for preconditioning operations and matrix-vector products of the form $\mathbf{Az}$.

ATTRIBUTES — Version: 1.3.0. (26 March 2013) **Types:** Real (single, double). **Calls:** `_COPY`, `_DOT`, `_NRM2`, `_SCAL`, `_AXPY`. **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 `MI01A` and supersedes it.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequence

There are two entries:

(a) `MI21I/ID` sets default values for control parameters. It should normally be called once prior to calls to `MI21A/AD`.

(b) `MI21A/AD` uses the Conjugate Gradient method to solve $\mathbf{Ax} = \mathbf{b}$, optionally using preconditioning. `MI21A/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 MI21I(ICNTL,CNTL,ISAVE,RSAVE)
```

The double precision version

```
CALL MI21ID(ICNTL,CNTL,ISAVE,RSAVE)
```

`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 `5` that need not be set by the user. On return it contains default values (see Section 2.2 for details).

`ISAVE` is an `INTEGER` array of length `10` used by `MI21` as private workspace and must not be altered by the user.

`RSAVE` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `6` used by `MI21` 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 MI21A(IACT,N,W,LDW,LOCY,LOCZ,RESID,ICNTL,CNTL,INFO,ISAVE,RSAVE)
```

The double precision version

```
CALL MI21AD(IACT,N,W,LDW,LOCY,LOCZ,RESID,ICNTL,CNTL,INFO,ISAVE,RSAVE)
```

`IACT` is an `INTEGER` variable. Prior to the first call to `MI21A/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 MI21A/AD without changing any of the arguments.

2 The user must perform the matrix-vector product

$$\mathbf{y} := \mathbf{A}\mathbf{z} \qquad (2.1)$$

and recall MI21A/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

$$\mathbf{y} := \mathbf{P}\mathbf{P}^T\mathbf{z}, \qquad (2.2)$$

where $\mathbf{P}\mathbf{P}^T$ is the preconditioning matrix, and recall MI21A/AD. The vectors **y** and **z** are held in the first N entries of columns LOCY and LOCZ of array W, respectively.

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 MI21A/AD. This argument is not altered by the routine. **Restriction**: N ≥ 1.

W is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LDW, 4). 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 $\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}$, 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 MI21A/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.

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 MI21A/AD.

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 $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$, 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 MI21I/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 5 that contains control parameters and must be set by the user. Default values for the components may be set by a call to MI21I/ID. Details of the control parameters are 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 MI21A/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 10 used by MI21 as private workspace and must not be altered by the user.

RSAVE is a REAL (DOUBLE PRECISION in the D version) array of length 6 used by MI21 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 MI21A/AD. Default values may be set by calling MI21I/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.

ICNTL(4) controls whether the convergence test offered by MI21A/AD is to be used. It has default value 0 and in this case the computed solution $\mathbf{x}$ is accepted if $\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2$ is less than or equal to max($\|\mathbf{b}-\mathbf{A}\mathbf{x}^{(0)}\|_2$*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 $\mathbf{x}$. 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 MI21A/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 n. If the user does not want the maximum to be n, 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) determines whether the normalized curvature is used when testing for breakdown (see Section 4). The normalized curvature is only used if ICNTL(7) = 1. The default value is 0. (see Section 4).

ICNTL(8) is set to zero by MI21I/ID but not currently used by MI21A/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 MI21A/AD.

CNTL(3) is the smallest allowable curvature (see Section 4). It has default value –1.0, and in this case it is reset by MI21A/AD to the nu, where u is the relative machine precision. If the user wishes to change the smallest allowable curvature, CNTL(3) should be set to the required positive after calling MI21I/ID. Values of CNTL(3) which are less than or equal to zero are treated as if they were the default –1.0.

CNTL(4) and CNTL(5) are set to zero by MI21I/ID but not currently used by MI21A/AD.

2.3 Error diagnostics

If MI21A/AD returns with a negative value of INFO(1), an error has occurred; if MI21A/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 LDW < N. Immediate return with input parameters unchanged.

–3 The curvature encountered is too small and the algorithm has broken down (see Section 4).

–4 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 $\sqrt{u}$.

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: MI21A/AD also calls the BLAS kernels SNRM2/DNRM2, SCOPY/DCOPY, SAXPY/DAXPY, SSCAL/DSCAL, SDOT/DDOT.

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

Restrictions:

N ≥ 1,
LDW ≥ N.

4 METHOD

The Conjugate Gradient method is due to Hestenes and Stiefel (1952). The method is described in detail by J. J. Dongarra, I. S. Duff, D. C. Sorensen and H. A. van der Vorst (1991), Section 7.1.1. The algorithm used by MI21A/AD proceeds 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
Return, if necessary, with IACT = 2 for the user to compute $\mathbf{Ax}^{(0)}$ and then obtain the initial residual $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{Ax}^{(0)}$.
if ICNTL(6) is greater then zero
    ITMAX = ICNTL(6)
else
    ITMAX = N
end if
do $i = 1, 2, ...,$ ITMAX

    if ICNTL(3) is nonzero
        Return with IACT = 3 for the user to compute $\mathbf{z} = \mathbf{PP}^T\mathbf{r}^{(i-1)}$.
    else
        $\mathbf{z} = \mathbf{r}^{(i-1)}$
    end if
    $\rho_{i-1} = \mathbf{z}^T\mathbf{r}^{(i-1)}$
    if $i = 1$
        $\mathbf{p}^{(1)} = \mathbf{z}^{(1)}$

else

$\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$

$\mathbf{p}^{(i)} = \mathbf{z} + \beta_{i-1}\mathbf{p}^{(i-1)}$

end if

Return with `IACT` = `2` for the user to compute $\mathbf{q} = \mathbf{Ap}$.

$\kappa_{i-1} = \mathbf{q}^T\mathbf{p}^{(i)}$ (if `ICNTL(7)` = `1`, $\kappa_{i-1} = \mathbf{q}^T\mathbf{p}^{(i)}/\mathbf{p}^{(i)^T}\mathbf{p}^{(i)}$)

if $\kappa_{i-1} <$ `CNTL(3)` the method has broken down. Set `INFO(1)` and return with `IACT` = `−1`.

$\alpha_i = \rho_{i-1}/\kappa_{i-1}$

$\mathbf{x}^{(i)} = \mathbf{x}^{(i-1)} + \alpha_i\mathbf{p}^{(i)}$

$\mathbf{r}^{(i)} = \mathbf{r}^{(i-1)} - \alpha_i\mathbf{q}$

if `ICNTL(4)` is zero,

if $\|\mathbf{r}^{(i)}\|_2 \le \max(\|\mathbf{r}^{(0)}\|_2 *$ `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

end do

References

M. R. Hestenes and E. Stiefel (1952). Methods of conjugate gradients for solving linear systems. J. Res. N. B. S., **49**, 409-436.

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 `MI21`.

```
      PROGRAM MAIN
C
C  Solve the linear system A x = b, where A is tridiagonal with
C  off-diagonal values 1 and diagonals 2, and where
C  the inverse of the diagonal of A is used as a preconditioner
C
      INTEGER N, LDW
      PARAMETER ( N = 10, LDW = N )
      DOUBLE PRECISION ZERO, ONE, TWO, THREE, FOUR
      PARAMETER ( TWO   = 2.0D+0, ZERO = 0.0D+0, ONE = 1.0D+0 )
      PARAMETER ( THREE = 3.0D+0, FOUR = 4.0D+0)
      DOUBLE PRECISION RESID
      INTEGER I, IACT, LOCY, LOCZ
      DOUBLE PRECISION CNTL( 5 ), W( LDW, 4 )
      INTEGER ICNTL( 8 ),INFO( 4 )
      INTEGER ISAVE(10)
      DOUBLE PRECISION RSAVE(6)
      EXTERNAL MI21AD, MI21ID
      CALL MI21ID( ICNTL, CNTL, ISAVE, RSAVE )
      ICNTL( 3 ) = 1
C
C  Set right hand side, b
C
      W( 1, 1 ) = THREE
      DO 10 I = 2, N - 1
         W( I, 1 ) = FOUR
   10 CONTINUE
```

```
      W( N, 1 ) = THREE
C
C  Perform an iteration of the Conjugate Gradient method
C
      IACT = 0
   20 CONTINUE
      CALL MI21AD( IACT, N, W, LDW, LOCY, LOCZ, RESID,
     *             ICNTL, CNTL, INFO, ISAVE, RSAVE )
      IF ( IACT .LT. 0 ) THEN
         WRITE( 6, 2020 ) INFO( 1 )
         GO TO 50
      END IF
      IF ( IACT .EQ. 2 ) THEN
C
C  Perform the matrix-vector product
C
         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
      IF ( IACT .EQ. 3 ) THEN
C
C  Perform the preconditioning operation
C
         DO 40 I = 1, N
            W( I, LOCY ) = W( I, LOCZ ) / TWO
   40    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 )
   50 CONTINUE
      STOP
 2000 FORMAT( I6, ' iterations required by MI21 ', // ' Solution = ',
     *        / ( 1P, 5D10.2 ) )
 2010 FORMAT( ' Warning: INFO( 1 ) = ', I2, ' on exit ')
 2020 FORMAT( ' Error return: INFO( 1 ) = ', I2, ' on exit ')
      END
```

This produces the following output:

```
     5 iterations required by MI21

 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
```