1 SUMMARY

This subroutine **estimates the 1-norm** ($\max_j \sum_{i=1}^{n} |a_{ij}|$) of an $n \times n$ complex matrix **A** given the ability to multiply a vector by both the matrix and its conjugate transpose. Because the explicit form of **A** is not required, the subroutine can be used for estimating the norm of matrix functions such as the inverse. Additionally this subroutine is potentially useful for estimating condition numbers of a matrix when the matrix is sparse or not available explicitly.

ATTRIBUTES — **Version:** 1.1.0. (3 December 2009) **Types:** Real (single, double). **Original date:** June 2001. **Remark:** `MF71` is a threadsafe version of `MF41`. **Origin:** M. Arioli, I.S. Duff, Rutherford Appleton Laboratory.

2 HOW TO USE THE PACKAGE

'Reverse communication' is used to provide the subroutine with the values of $\mathbf{A}\,\mathbf{x}$ or $\mathbf{A}^H\mathbf{x}$ for a given vector $\mathbf{x}$ ($\mathbf{A}^H$ denotes the complex conjugate transpose of the matrix **A**). When the subroutine requires the values of these products, it sets **x** in the array `X` and returns to the user's program with a flag, called `KASE`, set to `1` if $\mathbf{A}\,\mathbf{x}$ is required or set to `2` if $\mathbf{A}^H\mathbf{x}$ is required. The user's program must set `X` to the value of the matrix-vector product required by the subroutine and must not alter any of the other arguments of the subroutine. Initially the user must set the value of `KASE` to `0`; the subroutine sets `KASE` to `0` at the end of the computation. An example of use is shown in Section 5.

2.1 Argument list

The single precision version:

```
CALL MF71A(N,KASE,X,EST,W,KEEP)
```

The double precision version:

```
CALL MF71AD(N,KASE,X,EST,W,KEEP)
```

`N` is an `INTEGER` variable that must be set by the user to the order of the matrix. This argument is not altered by the subroutine. **Restriction:** `N` >0.

`KASE` is an `INTEGER` variable that must be set by the user to `0` on the initial call. The subroutine sets `KASE` to `1` or `2` in the intermediate returns and to `0` for the final return. `KASE` $=-1$ indicates an error condition (see §2.5).
When `KASE` $=1$ the user must supply the product of `X` by the matrix **A**.
When `KASE` $=2$ the user must supply the product of `X` by $\mathbf{A}^H$, the complex conjugate transpose of the matrix **A**.

`X` is a `COMPLEX` (`COMPLEX*16` in the D version) array of length `N` that need not be set by the user initially. On intermediate returns, the user must overwrite `X` by the product of **A** with `X` (`KASE` $=1$) or by the product of $\mathbf{A}^H$ with `X` (`KASE` $=2$).

`EST` is a `REAL` (`DOUBLE PRECISION` in the D version) variable. It is set by the subroutine to contain a lower bound estimate for the 1-norm of the matrix.

`W` is a `COMPLEX` (`COMPLEX*16` in the D version) array of length `N` that is used as workspace.

`KEEP` is an `INTEGER` array of length 5 which is used by the routine as private workspace and must not be altered by the user.

2.2 Finding the ∞–norm

Since $\|\mathbf{B}\|_\infty = \|\mathbf{B}^H\|_1$, the subroutine can be used to estimate the ∞–norm of $\mathbf{B}$ by working with $\mathbf{A} = \mathbf{B}^H$.

2.3 Finding condition numbers

The subroutine can be used to estimate the classical condition number $\kappa(\mathbf{B}) = \|\mathbf{B}\|\,\|\mathbf{B}^{-1}\|$ (for either 1- or ∞–norm) of a matrix $\mathbf{B}$. To compute the products of $\mathbf{A} = \mathbf{B}^{-1}$ and $\mathbf{A} = \mathbf{B}^{-H}$ times a given vector $\mathbf{x}$, the user must be able to solve $\mathbf{B}\,\mathbf{y} = \mathbf{x}$ and $\mathbf{B}^H\mathbf{y} = \mathbf{x}$.

2.4 Nonsquare matrices

It is also possible to use `MF71` for computing the 1-norm or the ∞–norm of an $m \times n$ matrix $\mathbf{B}$. This can be easily calculated by computing the norm of the square matrix obtained by bordering the matrix $\mathbf{B}$ with $(m-n)$ zero columns if $m > n$ or with $(n-m)$ zero rows if $m < n$.

2.5 Errors and diagnostic messages

The value -1 for `KASE` indicates that `N` ≤ 0.

3 GENERAL INFORMATION

Workspace: Provided by user, see argument `W`.

Use of common: None.

Other routines called directly: None.

Input/output: None.

Restrictions: `N` > 0.

4 METHOD

The method used is based on that developed by Hager (1984) and incorporates the modifications suggested by Higham (1987). Because $\|\mathbf{A}\|_1$ is the global maximum of the function $f(\mathbf{x}) = \|\mathbf{A}\,\mathbf{x}\|_1$ over the set $S = \{\,\mathbf{x} : \|\mathbf{x}\|_1 \leq 1\,\}$, Hager (1984) introduces an iterative method which, at each step, moves from a point in S to another one where the value of $f(\mathbf{x})$ increases. In a finite number of iterations ($\leq n$) the algorithm guarantees the convergence to a local maximum of $f(\mathbf{x})$. In practice, 2 or 3 iterations are sufficient to obtain a local maximum. In `MF71A/AD`, the number of iterations is limited to 5. Each iteration requires one matrix-vector product of the form $\mathbf{Ax}$ and one of the form $\mathbf{A}^H\mathbf{x}$ so the maximum number of calls to `MF71A/AD` that the user need make is 12.

`MF71A/AD` does not require the user to supply the matrix $\mathbf{A}$ but does require the user to compute products $\mathbf{Ax}$ and $\mathbf{A}^H\mathbf{x}$. Hence, it is possible to use `MF71A/AD` to compute the norm of a matrix which is a function of matrices without computing the resulting matrix explicitly. For example, the norm of the inverse of a matrix $\mathbf{B}$ may be estimated if the systems $\mathbf{B}\,\mathbf{y} = \mathbf{x}$ and $\mathbf{B}^H\mathbf{y} = \mathbf{x}$ can be solved. Arioli, Demmel, and Duff (1988) use this method for estimating different kinds of condition numbers of a sparse matrix.

References

Arioli, M., Demmel J. W., and Duff, I. S. (1989). Solving sparse systems with sparse backward error. *SIAM J. Matrix Anal. and Applics.* **10**, 165-190.

Hager, W. W. (1984). Condition Estimates. *SIAM J. Sci. Stat. Comput.* **5**, 311-316.

Higham, N. J. (1988). Fortran codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Trans. Math. Softw.* **14**, 381-396.

5 EXAMPLE OF USE

We wish to use MF71A to compute an estimate of the 1-norm of the matrix

$$\mathbf{A} = \begin{pmatrix} i & -2 & 1 & 3-4i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & i & 0 \\ 1 & 0 & 0 & 3+4i \end{pmatrix}.$$

The following program is used.

```
C     .. Local Scalars ..
      REAL EST
      INTEGER I,ITER,J,KASE,LA,N
C     ..
C     .. Local Arrays ..
      COMPLEX A(10,10),V(10),W(10),X(10)
      INTEGER KEEP(5)
C     ..
C     .. External Subroutines ..
      EXTERNAL MAPX,MF71A
C     ..
C Read in matrix
      READ (5,FMT=*) N
      READ (5,FMT=*) ((A(I,J),J=1,N),I=1,N)
C
      LA = 10
C
C Compute 1-norm
      KASE = 0
      DO 10 ITER = 1,12
         CALL MF71A(N,KASE,X,EST,V,KEEP)
C Trap fatal errors
         IF (KASE.LT.0) GO TO 30
C Jump if converged
         IF (KASE.EQ.0) GO TO 20
         CALL MAPX(LA,N,A,X,W,KASE)
   10 CONTINUE
   20 WRITE (6,FMT=9010) EST
      GO TO 40
C
   30 WRITE (6,FMT=9000)
   40 STOP
C
 9000 FORMAT (' MF71A- N .LE. 0 ')
 9010 FORMAT (' Estimate of the 1-norm of A        ',D11.4)
      END
C
      SUBROUTINE MAPX(LA,N,A,X,W,KASE)
C Subroutine to compute matrix-vector products
C
C     .. Parameters ..
      COMPLEX ZERO
      PARAMETER (ZERO= (0.0,0.0))
C     ..
C     .. Scalar Arguments ..
      INTEGER KASE,LA,N
C     ..
C     .. Array Arguments ..
      COMPLEX A(LA,LA),W(N),X(N)
C     ..
```

```
C     .. Local Scalars ..
      INTEGER I,J
C     ..
C     .. Intrinsic Functions ..
      INTRINSIC CONJG
C     ..
      IF (KASE.EQ.1) THEN
C Compute AX
         DO 20 I = 1,N
            W(I) = ZERO
            DO 10 J = 1,N
               W(I) = W(I) + A(I,J)*X(J)
   10       CONTINUE
   20    CONTINUE
      ELSE IF (KASE.EQ.2) THEN
C Compute A(H)X  (A(H) is complex conjugate transpose)
         DO 40 I = 1,N
            W(I) = ZERO
            DO 30 J = 1,N
               W(I) = W(I) + CONJG(A(J,I))*X(J)
   30       CONTINUE
   40    CONTINUE
      END IF
      DO 50 I = 1,N
         X(I) = W(I)
   50 CONTINUE
      RETURN
      END
```

The input data used for this problem is:

```
 4
(0.0, 1.0) (-2.0, 0.0)  (1.0, 0.0)  (3.0,-4.0)
(0.0, 0.0)  (1.0, 0.0)  (0.0, 0.0)  (0.0, 0.0)
(0.0, 0.0)  (0.0, 0.0)  (0.0, 1.0)  (0.0, 0.0)
(1.0, 0.0)  (0.0, 0.0)  (0.0, 0.0)  (3.0, 4.0)
```

and this produces the output:

```
 Estimate of the 1-norm of A         0.1000D+02
```