NAG Library Routine Document

F04ZCF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F04ZCF estimates the 1-norm of a complex matrix without accessing the matrix explicitly. It uses reverse communication for evaluating matrix-vector products. The routine may be used for estimating matrix condition numbers.

2 Specification

```
SUBROUTINE F04ZCF (ICASE, N, X, ESTNRM, WORK, IFAIL)
INTEGER                ICASE, N, IFAIL
REAL (KIND=nag_wp)     ESTNRM
COMPLEX (KIND=nag_wp) X(N), WORK(N)
```

3 Description

F04ZCF computes an estimate (a lower bound) for the 1-norm

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}| \tag{1}$$

of an n by n complex matrix $A = (a_{ij})$. The routine regards the matrix A as being defined by a user-supplied 'Black Box' which, given an input vector x, can return either of the matrix-vector products Ax or $A^{\mathrm{H}}x$, where A^{H} is the complex conjugate transpose. A reverse communication interface is used; thus control is returned to the calling program whenever a matrix-vector product is required.

Note: this routine is **not recommended** for use when the elements of A are known explicitly; it is then more efficient to compute the 1-norm directly from the formula (1) above.

The **main use** of the routine is for estimating $\|B^{-1}\|_1$, and hence the **condition number** $\kappa_1(B) = \|B\|_1 \|B^{-1}\|_1$, without forming B^{-1} explicitly ($A = B^{-1}$ above).

If, for example, an LU factorization of B is available, the matrix-vector products $B^{-1}x$ and $B^{-\mathrm{H}}x$ required by F04ZCF may be computed by back- and forward-substitutions, without computing B^{-1}.

The routine can also be used to estimate 1-norms of matrix products such as $A^{-1}B$ and ABC, without forming the products explicitly. Further applications are described in Higham (1988).

Since $\|A\|_\infty = \|A^{\mathrm{H}}\|_1$, F04ZCF can be used to estimate the ∞-norm of A by working with A^{H} instead of A.

The algorithm used is based on a method given in Hager (1984) and is described in Higham (1988). A comparison of several techniques for condition number estimation is given in Higham (1987).

Note: F04ZDF can also be used to estimate the norm of a real matrix. F04ZDF uses a more recent algorithm than F04ZCF and it is recommended that F04ZDF be used in place of F04ZCF.

4 References

Hager W W (1984) Condition estimates *SIAM J. Sci. Statist. Comput.* **5** 311–316

Higham N J (1987) A survey of condition number estimation for triangular matrices *SIAM Rev.* **29** 575–596

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. Software* **14** 381–396

5 Parameters

Note: this routine uses **reverse communication.** Its use involves an initial entry, intermediate exits and re-entries, and a final exit, as indicated by the **parameter ICASE**. Between intermediate exits and re-entries, **all parameters other than X must remain unchanged**.

1: ICASE – INTEGER *Input/Output*

On initial entry: must be set to 0.

On intermediate exit: ICASE $= 1$ or 2, and X(i), for $i = 1, 2, \dots, n$, contain the elements of a vector x. The calling program must

(a) evaluate Ax (if ICASE $= 1$) or $A^{\mathrm{H}}x$ (if ICASE $= 2$), where A^{H} is the complex conjugate transpose;

(b) place the result in X; and,

(c) call F04ZCF once again, with all the other parameters unchanged.

On final exit: ICASE $= 0$.

2: N – INTEGER *Input*

On initial entry: n, the order of the matrix A.

Constraint: N ≥ 1.

3: X(N) – COMPLEX (KIND=nag_wp) array *Input/Output*

On initial entry: need not be set.

On intermediate exit: contains the current vector x.

On intermediate re-entry: must contain Ax (if ICASE $= 1$) or $A^{\mathrm{H}}x$ (if ICASE $= 2$).

On final exit: the array is undefined.

4: ESTNRM – REAL (KIND=nag_wp) *Input/Output*

On initial entry: need not be set.

On intermediate exit: should not be changed.

On final exit: an estimate (a lower bound) for $\|A\|_1$.

5: WORK(N) – COMPLEX (KIND=nag_wp) array *Input/Output*

On initial entry: need not be set.

On final exit: contains a vector v such that $v = Aw$ where ESTNRM $= \|v\|_1 / \|w\|_1$ (w is not returned). If $A = B^{-1}$ and ESTNRM is large, then v is an approximate null vector for B.

6: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, $\mathrm{N} < 1$.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

In extensive tests on **random** matrices of size up to $n = 100$ the estimate ESTNRM has been found always to be within a factor eleven of $\|A\|_1$; often the estimate has many correct figures. However, matrices exist for which the estimate is smaller than $\|A\|_1$ by an arbitrary factor; such matrices are very unlikely to arise in practice. See Higham (1988) for further details.

8 Parallelism and Performance

F04ZCF is not threaded by NAG in any implementation.

F04ZCF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

9.1 Timing

The total time taken by F04ZCF is proportional to n. For most problems the time taken during calls to F04ZCF will be negligible compared with the time spent evaluating matrix-vector products between calls to F04ZCF.

The number of matrix-vector products required varies from 5 to 11 (or is 1 if $n = 1$). In most cases 5 products are required; it is rare for more than 7 to be needed.

9.2 Overflow

It is your responsibility to guard against potential overflows during evaluation of the matrix-vector products. In particular, when estimating $\|B^{-1}\|_1$ using a triangular factorization of B, F04ZCF should not be called if one of the factors is exactly singular – otherwise division by zero may occur in the substitutions.

9.3 Use in Conjunction with NAG Library Routines

To estimate the 1-norm of the inverse of a matrix A, the following skeleton code can normally be used:

```
          ...  code to factorize A ...
          IF (A is not singular) THEN
             ICASE = 0
    10       CALL F04ZCF (ICASE,N,X,ESTNRM,WORK,IFAIL)
             IF (ICASE.NE.0) THEN
                IF (ICASE.EQ.1) THEN
                   ...  code to compute A(-1)x ...
                ELSE
                   ...  code to compute (A(-1)(H)) x ...
                END IF
                GO TO 10
             END IF
          END IF
```

To compute $A^{-1}x$ or $A^{-\mathrm{H}}x$, solve the equation $Ay = x$ or $A^{\mathrm{H}}y = x$ for y, overwriting y on x. The code will vary, depending on the type of the matrix A, and the NAG routine used to factorize A.

Note that if A is any type of **Hermitian** matrix, then $A = A^{\mathrm{H}}$, and the code following the call of F04ZCF can be reduced to:

```
    IF (ICASE.NE.0) THEN
       ...  code to compute A(-1)x ...
       GO TO 10
    END IF
```

The example program in Section 10 illustrates how F04ZCF can be used in conjunction with NAG Library routines for complex band matrices (factorized by F07BRF (ZGBTRF)).

It is also straightforward to use F04ZCF for Hermitian positive definite matrices, using F06TFF, F07FRF (ZPOTRF) and F07FSF (ZPOTRS) for factorization and solution.

For upper or lower triangular matrices, no factorization routine is needed: $A^{-1}x$ and $A^{-\mathrm{H}}x$ may be computed by calls to F06SJF (ZTRSV) (or F06SKF (ZTBSV) if the matrix is banded, or F06SLF (ZTPSV) if the matrix is stored in packed form).

10 Example

This example estimates the condition number $\|A\|_1\|A^{-1}\|_1$ of the order 5 matrix

$$A = \begin{pmatrix} 1+i & 2+i & 1+2i & 0 & 0 \\ 2i & 3+5i & 1+3i & 2+i & 0 \\ 0 & -2+6i & 5+7i & 6i & 1-i \\ 0 & 0 & 3+9i & 4i & 4-3i \\ 0 & 0 & 0 & -1+8i & 10-3i \end{pmatrix}$$

where A is a band matrix stored in the packed format required by F07BRF (ZGBTRF) and F07BSF (ZGBTRS).

Further examples of the technique for condition number estimation in the case of real matrices can be seen in the example program section of F04YCF.

10.1 Program Text

```
    Program f04zcfe

!     F04ZCF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: f04zcf, f06ubf, nag_wp, zgbtrf, zgbtrs
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: anorm, cond, estnrm
      Integer                          :: i, icase, ifail, info, j, k, kl, ku, &
                                          lda, ldx, n, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), work(:), x(:,:)
      Real (Kind=nag_wp)               :: rwork(1)
      Integer, Allocatable             :: ipiv(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, min
!     .. Executable Statements ..
      Write (nout,*) 'F04ZCF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, kl, ku, nrhs
      lda = 2*kl + ku + 1
      ldx = n
      Allocate (a(lda,n),work(n),x(ldx,nrhs),ipiv(n))
      k = kl + ku + 1
      Read (nin,*)((a(k+i-j,j),j=max(i-kl,1),min(i+ku,n)),i=1,n)

!     First compute the 1-norm of A.
      anorm = f06ubf('1-norm',n,kl,ku,a(kl+1,1),lda,rwork)
      Write (nout,*)
      Write (nout,99999) 'Computed norm of A =', anorm

!     Next estimate the 1-norm of inverse(A).
!     Factorise A into P*L*U.
!     The NAG name equivalent of zgbtrf is f07brf
      Call zgbtrf(n,n,kl,ku,a,lda,ipiv,info)

      icase = 0

loop: Do
!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call f04zcf(icase,n,x,estnrm,work,ifail)

        If (icase/=0) Then
!         The NAG name equivalent of the backsolve routine zgbtrs is f07bsf
          If (icase==1) Then
!           Return X := inv(A)*X by solving A*Y = X, overwriting
!           Y on X.
            Call zgbtrs('No transpose',n,kl,ku,nrhs,a,lda,ipiv,x,ldx,info)
          Else If (icase==2) Then
!           Return X := conjg(inv(A)')*X by solving conjg(A')*Y
!           = X, overwriting Y on X.
            Call zgbtrs('Conjugate transpose',n,kl,ku,nrhs,a,lda,ipiv,x,ldx, &
              info)
          End If
!       Continue until icase is returned as 0.
        Else
          Write (nout,99999) 'Estimated norm of inverse(A) =', estnrm
          cond = anorm*estnrm
          Write (nout,99998) 'Estimated condition number of A =', cond
          Exit loop
        End If
      End Do loop

99999 Format (1X,A,F8.4)
99998 Format (1X,A,F6.1)
    End Program f04zcfe
```

10.2 Program Data

```
F04ZCF Example Program Data
 5  1  2  1                                                  : n, kl, ku, nrhs
 ( 1.0, 1.0) ( 2.0, 1.0) ( 1.0, 2.0)
 ( 0.0, 2.0) ( 3.0, 5.0) ( 1.0, 3.0) ( 2.0, 1.0)
             (-2.0, 6.0) ( 5.0, 7.0) ( 0.0, 6.0) ( 1.0,-1.0)
                         ( 3.0, 9.0) ( 0.0, 4.0) ( 4.0,-3.0)
                                     (-1.0, 8.0) (10.0,-3.0) : matrix A
```

10.3 Program Results

```
 F04ZCF Example Program Results

 Computed norm of A = 23.4875
 Estimated norm of inverse(A) = 37.0391
 Estimated condition number of A = 870.0
```