NAG Library Routine Document

F07AVF (ZGERFS)

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

F07AVF (ZGERFS) returns error bounds for the solution of a complex system of linear equations with multiple right-hand sides, $AX = B$, $A^{\mathrm{T}}X = B$ or $A^{\mathrm{H}}X = B$. It improves the solution by iterative refinement, in order to reduce the backward error as much as possible.

2 Specification

```
SUBROUTINE F07AVF (TRANS, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB, X, LDX,      &
                   FERR, BERR, WORK, RWORK, INFO)

INTEGER                 N, NRHS, LDA, LDAF, IPIV(*), LDB, LDX, INFO
REAL (KIND=nag_wp)      FERR(NRHS), BERR(NRHS), RWORK(N)
COMPLEX (KIND=nag_wp) A(LDA,*), AF(LDAF,*), B(LDB,*), X(LDX,*), WORK(2*N)
CHARACTER(1)            TRANS
```

The routine may be called by its LAPACK name ***zgerfs***.

3 Description

F07AVF (ZGERFS) returns the backward errors and estimated bounds on the forward errors for the solution of a complex system of linear equations with multiple right-hand sides $AX = B$, $A^{\mathrm{T}}X = B$ or $A^{\mathrm{H}}X = B$. The routine handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of F07AVF (ZGERFS) in terms of a single right-hand side b and solution x.

Given a computed solution x, the routine computes the *component-wise backward error* β. This is the size of the smallest relative perturbation in each element of A and b such that x is the exact solution of a perturbed system

$$(A + \delta A)x = b + \delta b$$
$$\left|\delta a_{ij}\right| \leq \beta\left|a_{ij}\right| \qquad \text{and} \qquad |\delta b_i| \leq \beta|b_i|.$$

Then the routine estimates a bound for the *component-wise forward error* in the computed solution, defined by:

$$\max_i |x_i - \hat{x}_i| / \max_i |x_i|$$

where $\hat{x}$ is the true solution.

For details of the method, see the F07 Chapter Introduction.

4 References

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

1: TRANS – CHARACTER(1) *Input*

On entry: indicates the form of the linear equations for which X is the computed solution as follows:

TRANS = 'N'
The linear equations are of the form $AX = B$.

TRANS = 'T'
The linear equations are of the form $A^{\mathrm{T}}X = B$.

TRANS = 'C'
The linear equations are of the form $A^{\mathrm{H}}X = B$.

Constraint: TRANS = 'N', 'T' or 'C'.

2: N – INTEGER *Input*

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

Constraint: N ≥ 0.

3: NRHS – INTEGER *Input*

On entry: r, the number of right-hand sides.

Constraint: NRHS ≥ 0.

4: A(LDA,$*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the second dimension of the array A must be at least $\max(1, \mathrm{N})$.

On entry: the n by n original matrix A as supplied to F07ARF (ZGETRF).

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F07AVF (ZGERFS) is called.

Constraint: LDA $\geq \max(1, \mathrm{N})$.

6: AF(LDAF,$*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the second dimension of the array AF must be at least $\max(1, \mathrm{N})$.

On entry: the LU factorization of A, as returned by F07ARF (ZGETRF).

7: LDAF – INTEGER *Input*

On entry: the first dimension of the array AF as declared in the (sub)program from which F07AVF (ZGERFS) is called.

Constraint: LDAF $\geq \max(1, \mathrm{N})$.

8: IPIV($*$) – INTEGER array *Input*

Note: the dimension of the array IPIV must be at least $\max(1, \mathrm{N})$.

On entry: the pivot indices, as returned by F07ARF (ZGETRF).

9: B(LDB,$*$) – COMPLEX (KIND=nag_wp) array *Input*

Note: the second dimension of the array B must be at least $\max(1, \mathrm{NRHS})$.

On entry: the n by r right-hand side matrix B.

10: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07AVF (ZGERFS) is called.

Constraint: LDB $\geq \max(1, \mathrm{N})$.

11: X(LDX,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array X must be at least $\max(1, \mathrm{NRHS})$.

On entry: the n by r solution matrix X, as returned by F07ASF (ZGETRS).

On exit: the improved solution matrix X.

12: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07AVF (ZGERFS) is called.

Constraint: LDX $\geq \max(1, \mathrm{N})$.

13: FERR(NRHS) – REAL (KIND=nag_wp) array *Output*

On exit: FERR(j) contains an estimated error bound for the jth solution vector, that is, the jth column of X, for $j = 1, 2, \ldots, r$.

14: BERR(NRHS) – REAL (KIND=nag_wp) array *Output*

On exit: BERR(j) contains the component-wise backward error bound β for the jth solution vector, that is, the jth column of X, for $j = 1, 2, \ldots, r$.

15: WORK($2 \times$ N) – COMPLEX (KIND=nag_wp) array *Workspace*

16: RWORK(N) – REAL (KIND=nag_wp) array *Workspace*

17: INFO – INTEGER *Output*

On exit: INFO $= 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, the ith parameter had an illegal value. An explanatory message is output, and execution of the program is terminated.

7 Accuracy

The bounds returned in FERR are not rigorous, because they are estimated, not computed exactly; but in practice they almost always overestimate the actual error.

8 Further Comments

For each right-hand side, computation of the backward error involves a minimum of $16n^2$ real floating point operations. Each step of iterative refinement involves an additional $24n^2$ real operations. At most five steps of iterative refinement are performed, but usually only one or two steps are required.

Estimating the forward error involves solving a number of systems of linear equations of the form $Ax = b$ or $A^{\mathrm{H}}x = b$; the number is usually 5 and never more than 11. Each solution involves approximately $8n^2$ real operations.

The real analogue of this routine is F07AHF (DGERFS).

9 Example

This example solves the system of equations $AX = B$ using iterative refinement and to compute the forward and backward error bounds, where

$$A = \begin{pmatrix} -1.34 + 2.55i & 0.28 + 3.17i & -6.39 - 2.20i & 0.72 - 0.92i \\ -0.17 - 1.41i & 3.31 - 0.15i & -0.15 + 1.34i & 1.29 + 1.38i \\ -3.29 - 2.39i & -1.91 + 4.42i & -0.14 - 1.35i & 1.72 + 1.35i \\ 2.41 + 0.39i & -0.56 + 1.47i & -0.83 - 0.69i & -1.96 + 0.67i \end{pmatrix}$$

and

$$B = \begin{pmatrix} 26.26 + 51.78i & 31.32 - 6.70i \\ 6.43 - 8.68i & 15.86 - 1.42i \\ -5.75 + 25.31i & -2.15 + 30.19i \\ 1.16 + 2.57i & -2.56 + 7.55i \end{pmatrix}.$$

Here A is nonsymmetric and must first be factorized by F07ARF (ZGETRF).

9.1 Program Text

```
    Program f07avfe

!     F07AVF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dbf, zgerfs, zgetrf, zgetrs
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
      Character (1), Parameter          :: trans = 'N'
!     .. Local Scalars ..
      Integer                           :: i, ifail, info, lda, ldaf, ldb, ldx, &
                                           n, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), af(:,:), b(:,:), work(:),  &
                                            x(:,:)
      Real (Kind=nag_wp), Allocatable   :: berr(:), ferr(:), rwork(:)
      Integer, Allocatable              :: ipiv(:)
      Character (1)                     :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F07AVF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, nrhs
      lda = n
      ldaf = n
      ldb = n
      ldx = n
      Allocate (a(lda,n),af(ldaf,n),b(ldb,nrhs),work(2*n),x(ldx,n),berr(nrhs), &
        ferr(nrhs),rwork(n),ipiv(n))

!     Read A and B from data file, and copy A to AF and B to X

      Read (nin,*)(a(i,1:n),i=1,n)
      Read (nin,*)(b(i,1:nrhs),i=1,n)

      af(1:n,1:n) = a(1:n,1:n)
      x(1:n,1:nrhs) = b(1:n,1:nrhs)

!     Factorize A in the array AF
!     The NAG name equivalent of zgetrf is f07arf
```

```
      Call zgetrf(n,n,af,ldaf,ipiv,info)

      Write (nout,*)
      Flush (nout)
      If (info==0) Then

!       Compute solution in the array X

!       The NAG name equivalent of zgetrs is f07asf
        Call zgetrs(trans,n,nrhs,af,ldaf,ipiv,x,ldx,info)

!       Improve solution, and compute backward errors and
!       estimated bounds on the forward errors

!       The NAG name equivalent of zgerfs is f07avf
        Call zgerfs(trans,n,nrhs,a,lda,af,ldaf,ipiv,b,ldb,x,ldx,ferr,berr, &
          work,rwork,info)

!       Print solution

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04dbf('General',' ',n,nrhs,x,ldx,'Bracketed','F7.4', &
          'Solution(s)','Integer',rlabs,'Integer',clabs,80,0,ifail)

        Write (nout,*)
        Write (nout,*) 'Backward errors (machine-dependent)'
        Write (nout,99999) berr(1:nrhs)
        Write (nout,*) 'Estimated forward error bounds (machine-dependent)'
        Write (nout,99999) ferr(1:nrhs)
      Else
        Write (nout,*) 'The factor U is singular'
      End If

99999 Format ((5X,1P,4(E11.1,7X)))
    End Program f07avfe
```

9.2 Program Data

```
F07AVF Example Program Data
  4  2                                                    :Values of N and NRHS
 (-1.34, 2.55) ( 0.28, 3.17) (-6.39,-2.20) ( 0.72,-0.92)
 (-0.17,-1.41) ( 3.31,-0.15) (-0.15, 1.34) ( 1.29, 1.38)
 (-3.29,-2.39) (-1.91, 4.42) (-0.14,-1.35) ( 1.72, 1.35)
 ( 2.41, 0.39) (-0.56, 1.47) (-0.83,-0.69) (-1.96, 0.67)  :End of matrix A
 (26.26, 51.78) (31.32, -6.70)
 ( 6.43, -8.68) (15.86, -1.42)
 (-5.75, 25.31) (-2.15, 30.19)
 ( 1.16,  2.57) (-2.56,  7.55)                            :End of matrix B
```

9.3 Program Results

```
 F07AVF Example Program Results

 Solution(s)
                     1                  2
 1  ( 1.0000, 1.0000) (-1.0000,-2.0000)
 2  ( 2.0000,-3.0000) ( 5.0000, 1.0000)
 3  (-4.0000,-5.0000) (-3.0000, 4.0000)
 4  ( 0.0000, 6.0000) ( 2.0000,-3.0000)

 Backward errors (machine-dependent)
         4.1E-17           8.7E-17
 Estimated forward error bounds (machine-dependent)
         5.8E-14           7.8E-14
```