NAG Library Routine Document

F07BHF (DGBRFS)

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

F07BHF (DGBRFS) returns error bounds for the solution of a real band system of linear equations with multiple right-hand sides, $AX = B$ or $A^{\mathrm{T}}X = B$. It improves the solution by iterative refinement, in order to reduce the backward error as much as possible.

2 Specification

```
SUBROUTINE F07BHF (TRANS, N, KL, KU, NRHS, AB, LDAB, AFB, LDAFB, IPIV,      &
                   B, LDB, X, LDX, FERR, BERR, WORK, IWORK, INFO)
INTEGER            N, KL, KU, NRHS, LDAB, LDAFB, IPIV(*), LDB, LDX,         &
                   IWORK(N), INFO
REAL (KIND=nag_wp) AB(LDAB,*), AFB(LDAFB,*), B(LDB,*), X(LDX,*),            &
                   FERR(NRHS), BERR(NRHS), WORK(3*N)
CHARACTER(1)       TRANS
```

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

3 Description

F07BHF (DGBRFS) returns the backward errors and estimated bounds on the forward errors for the solution of a real band system of linear equations with multiple right-hand sides $AX = B$ or $A^{\mathrm{T}}X = B$. The routine handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of F07BHF (DGBRFS) 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

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

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

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

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 Arguments

1: TRANS – CHARACTER(1) *Input*

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

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

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

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

2: N – INTEGER *Input*

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

Constraint: $\mathrm{N} \geq 0$.

3: KL – INTEGER *Input*

On entry: k_l, the number of subdiagonals within the band of the matrix A.

Constraint: $\mathrm{KL} \geq 0$.

4: KU – INTEGER *Input*

On entry: k_u, the number of superdiagonals within the band of the matrix A.

Constraint: $\mathrm{KU} \geq 0$.

5: NRHS – INTEGER *Input*

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

Constraint: $\mathrm{NRHS} \geq 0$.

6: $\mathrm{AB}(\mathrm{LDAB}, *)$ – REAL (KIND=nag_wp) array *Input*

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

On entry: the original n by n band matrix A as supplied to F07BDF (DGBTRF).

The matrix is stored in rows 1 to $k_l + k_u + 1$, more precisely, the element A_{ij} must be stored in

$$\mathrm{AB}(k_u + 1 + i - j, j) \quad \text{for } \max(1, j - k_u) \leq i \leq \min(n, j + k_l).$$

See Section 9 in F07BAF (DGBSV) for further details.

7: LDAB – INTEGER *Input*

On entry: the first dimension of the array AB as declared in the (sub)program from which F07BHF (DGBRFS) is called.

Constraint: $\mathrm{LDAB} \geq \mathrm{KL} + \mathrm{KU} + 1$.

8: $\mathrm{AFB}(\mathrm{LDAFB}, *)$ – REAL (KIND=nag_wp) array *Input*

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

On entry: the LU factorization of A, as returned by F07BDF (DGBTRF).

9: LDAFB – INTEGER *Input*

On entry: the first dimension of the array AFB as declared in the (sub)program from which F07BHF (DGBRFS) is called.

Constraint: $\mathrm{LDAFB} \geq 2 \times \mathrm{KL} + \mathrm{KU} + 1$.

10: IPIV($*$) – INTEGER array *Input*

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

On entry: the pivot indices, as returned by F07BDF (DGBTRF).

11: B(LDB, $*$) – REAL (KIND=nag_wp) array *Input*

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

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

12: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07BHF (DGBRFS) is called.

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

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

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

On entry: the n by r solution matrix X, as returned by F07BEF (DGBTRS).

On exit: the improved solution matrix X.

14: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07BHF (DGBRFS) is called.

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

15: 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 $j = 1, 2, \ldots, r$.

16: 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 $j = 1, 2, \ldots, r$.

17: WORK($3 \times N$) – REAL (KIND=nag_wp) array *Workspace*

18: IWORK(N) – INTEGER array *Workspace*

19: INFO – INTEGER *Output*

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

6 Error Indicators and Warnings

$INFO < 0$

If $INFO = -i$, argument i 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 Parallelism and Performance

F07BHF (DGBRFS) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F07BHF (DGBRFS) 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

For each right-hand side, computation of the backward error involves a minimum of $4n(k_l + k_u)$ floating-point operations. Each step of iterative refinement involves an additional $2n(4k_l + 3k_u)$ operations. This assumes $n \gg k_l$ and $n \gg k_u$. 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 $A^{\mathrm{T}}x = b$; the number is usually 4 or 5 and never more than 11. Each solution involves approximately $2n(2k_l + k_u)$ operations.

The complex analogue of this routine is F07BVF (ZGBRFS).

10 Example

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

$$A = \begin{pmatrix} -0.23 & 2.54 & -3.66 & 0.00 \\ -6.98 & 2.46 & -2.73 & -2.13 \\ 0.00 & 2.56 & 2.46 & 4.07 \\ 0.00 & 0.00 & -4.78 & -3.82 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 4.42 & -36.01 \\ 27.13 & -31.67 \\ -6.14 & -1.16 \\ 10.50 & -25.82 \end{pmatrix}.$$

Here A is nonsymmetric and is treated as a band matrix, which must first be factorized by F07BDF (DGBTRF).

10.1 Program Text

```
    Program f07bhfe

!     F07BHF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: dgbrfs, dgbtrf, dgbtrs, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: zero = 0.0E0_nag_wp
      Integer, Parameter               :: nin = 5, nout = 6
      Character (1), Parameter         :: trans = 'N'
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, j, k, kl, ku, ldab,  &
                                          ldafb, ldb, ldx, n, nrhs
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: ab(:,:), afb(:,:), b(:,:), berr(:),  &
                                          ferr(:), work(:), x(:,:)
      Integer, Allocatable             :: ipiv(:), iwork(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, min
!     .. Executable Statements ..
```

```
      Write (nout,*) 'F07BHF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, nrhs, kl, ku
      ldab = kl + ku + 1
      ldafb = 2*kl + ku + 1
      ldb = n
      ldx = n
      Allocate (ab(ldab,n),afb(ldafb,n),b(ldb,nrhs),berr(nrhs),ferr(nrhs),     &
        work(3*n),x(ldx,n),ipiv(n),iwork(n))

!     Set A to zero to avoid referencing uninitialized elements

      ab(1:kl+ku+1,1:n) = zero

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

      k = ku + 1
      Read (nin,*)((ab(k+i-j,j),j=max(i-kl,1),min(i+ku,n)),i=1,n)
      Read (nin,*)(b(i,1:nrhs),i=1,n)

      afb(kl+1:2*kl+ku+1,1:n) = ab(1:kl+ku+1,1:n)
      x(1:n,1:nrhs) = b(1:n,1:nrhs)

!     Factorize A in the array AFB
!     The NAG name equivalent of dgbtrf is f07bdf
      Call dgbtrf(n,n,kl,ku,afb,ldafb,ipiv,info)

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

!       Compute solution in the array X
!       The NAG name equivalent of dgbtrs is f07bef
        Call dgbtrs(trans,n,kl,ku,nrhs,afb,ldafb,ipiv,x,ldx,info)

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

!       The NAG name equivalent of dgbrfs is f07bhf
        Call dgbrfs(trans,n,kl,ku,nrhs,ab,ldab,afb,ldafb,ipiv,b,ldb,x,ldx,     &
          ferr,berr,work,iwork,info)

!       Print solution

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04caf('General',' ',n,nrhs,x,ldx,'Solution(s)',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 ((3X,1P,7E11.1))
    End Program f07bhfe
```

10.2 Program Data

```
F07BHF Example Program Data
  4  2  1  2                  :Values of N, NRHS, KL and KU
 -0.23   2.54  -3.66
 -6.98   2.46  -2.73  -2.13
         2.56   2.46   4.07
               -4.78  -3.82   :End of matrix A
  4.42 -36.01
 27.13 -31.67
 -6.14  -1.16
 10.50 -25.82                 :End of matrix B
```

10.3 Program Results

```
 F07BHF Example Program Results

 Solution(s)
             1          2
 1     -2.0000     1.0000
 2      3.0000    -4.0000
 3      1.0000     7.0000
 4     -4.0000    -2.0000

 Backward errors (machine-dependent)
       1.1E-16    9.9E-17
 Estimated forward error bounds (machine-dependent)
       1.6E-14    1.9E-14
```