NAG Library Routine Document

F07BNF (ZGBSV)

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

F07BNF (ZGBSV) computes the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n band matrix, with k_l subdiagonals and k_u superdiagonals, and X and B are n by r matrices.

2 Specification

```
SUBROUTINE F07BNF (N, KL, KU, NRHS, AB, LDAB, IPIV, B, LDB, INFO)
INTEGER                 N, KL, KU, NRHS, LDAB, IPIV(N), LDB, INFO
COMPLEX (KIND=nag_wp) AB(LDAB,*), B(LDB,*)
```

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

3 Description

F07BNF (ZGBSV) uses the LU decomposition with partial pivoting and row interchanges to factor A as $A = PLU$, where P is a permutation matrix, L is a product of permutation and unit lower triangular matrices with k_l subdiagonals, and U is upper triangular with $(k_l + k_u)$ superdiagonals. The factored form of A is then used to solve the system of equations $AX = B$.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

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

5 Arguments

1: N – INTEGER *Input*

On entry: n, the number of linear equations, i.e., the order of the matrix A.

Constraint: N ≥ 0.

2: KL – INTEGER *Input*

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

Constraint: KL ≥ 0.

3: KU – INTEGER *Input*

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

Constraint: KU ≥ 0.

4: NRHS – INTEGER *Input*

On entry: r, the number of right-hand sides, i.e., the number of columns of the matrix B.

Constraint: NRHS ≥ 0.

5: AB(LDAB, $*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array AB must be at least max(1, N).

On entry: the n by n coefficient matrix A.

The matrix is stored in rows $k_l + 1$ to $2k_l + k_u + 1$; the first k_l rows need not be set, more precisely, the element A_{ij} must be stored in

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

See Section 9 for further details.

On exit: if INFO ≥ 0, AB is overwritten by details of the factorization.

The upper triangular band matrix U, with $k_l + k_u$ superdiagonals, is stored in rows 1 to $k_l + k_u + 1$ of the array, and the multipliers used to form the matrix L are stored in rows $k_l + k_u + 2$ to $2k_l + k_u + 1$.

6: LDAB – INTEGER *Input*

On entry: the first dimension of the array AB as declared in the (sub)program from which F07BNF (ZGBSV) is called.

Constraint: LDAB $\geq 2 \times$ KL + KU + 1.

7: IPIV(N) – INTEGER array *Output*

On exit: if no constraints are violated, the pivot indices that define the permutation matrix P; at the ith step row i of the matrix was interchanged with row IPIV(i). IPIV(i) $= i$ indicates a row interchange was not required.

8: B(LDB, $*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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.

On exit: if INFO $= 0$, the n by r solution matrix X.

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07BNF (ZGBSV) is called.

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

10: 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.

INFO > 0

Element $\langle value \rangle$ of the diagonal is exactly zero. The factorization has been completed, but the factor U is exactly singular, so the solution could not be computed.

7 Accuracy

The computed solution for a single right-hand side, $\hat{x}$, satisfies an equation of the form

$$(A + E)\hat{x} = b,$$

where

$$\|E\|_1 = O(\epsilon)\|A\|_1$$

and ϵ is the ***machine precision***. An approximate error bound for the computed solution is given by

$$\frac{\|\hat{x} - x\|_1}{\|x\|_1} \leq \kappa(A)\frac{\|E\|_1}{\|A\|_1},$$

where $\kappa(A) = \left\|A^{-1}\right\|_1 \|A\|_1$, the condition number of A with respect to the solution of the linear equations. See Section 4.4 of Anderson *et al.* (1999) for further details.

Following the use of F07BNF (ZGBSV), F07BUF (ZGBCON) can be used to estimate the condition number of A and F07BVF (ZGBRFS) can be used to obtain approximate error bounds. Alternatives to F07BNF (ZGBSV), which return condition and error estimates directly are F04CBF and F07BPF (ZGBSVX).

8 Parallelism and Performance

F07BNF (ZGBSV) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F07BNF (ZGBSV) 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

The band storage scheme for the array AB is illustrated by the following example, when $n = 6$, $k_l = 1$, and $k_u = 2$. Storage of the band matrix A in the array AB:

$$\begin{array}{cccccc} * & * & * & + & + & + \\ * & * & a_{13} & a_{24} & a_{35} & a_{46} \\ * & a_{12} & a_{23} & a_{34} & a_{45} & a_{56} \\ a_{11} & a_{22} & a_{33} & a_{44} & a_{55} & a_{66} \\ a_{21} & a_{32} & a_{43} & a_{54} & a_{65} & * \end{array}$$

Array elements marked $*$ need not be set and are not referenced by the routine. Array elements marked $+$ need not be set, but are defined on exit from the routine and contain the elements u_{14}, u_{25} and u_{36}.

The total number of floating-point operations required to solve the equations $AX = B$ depends upon the pivoting required, but if $n \gg k_l + k_u$ then it is approximately bounded by $O(nk_l(k_l + k_u))$ for the factorization and $O(n(2k_l + k_u)r)$ for the solution following the factorization.

The real analogue of this routine is F07BAF (DGBSV).

10 Example

This example solves the equations

$$Ax = b,$$

where A is the band matrix

$$A = \begin{pmatrix} -1.65+2.26i & -2.05-0.85i & 0.97-2.84i & 0 \\ 6.30i & -1.48-1.75i & -3.99+4.01i & 0.59-0.48i \\ 0 & -0.77+2.83i & -1.06+1.94i & 3.33-1.04i \\ 0 & 0 & 4.48-1.09i & -0.46-1.72i \end{pmatrix}$$

and

$$b = \begin{pmatrix} -1.06+21.50i \\ -22.72-53.90i \\ 28.24-38.60i \\ -34.56+16.73i \end{pmatrix}.$$

Details of the LU factorization of A are also output.

10.1 Program Text

```
    Program f07bnfe

!     F07BNF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dff, zgbsv
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, j, k, kl, ku, ldab,  &
                                          n
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: ab(:,:), b(:)
      Integer, Allocatable             :: ipiv(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, min
!     .. Executable Statements ..
      Write (nout,*) 'F07BNF Example Program Results'
      Write (nout,*)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, kl, ku
      ldab = 2*kl + ku + 1
      Allocate (ab(ldab,n),b(n),ipiv(n))

!     Read the band matrix A and the right hand side b from data file

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

!     Solve the equations Ax = b for x
!     The NAG name equivalent of zgbsv is f07bnf
      Call zgbsv(n,kl,ku,1,ab,ldab,ipiv,b,n,info)

      If (info==0) Then

!       Print solution

        Write (nout,*) 'Solution'
```

```
        Write (nout,99999) b(1:n)

!       Print details of the factorization

        Write (nout,*)
        Flush (nout)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04dff(n,n,kl,kl+ku,ab,ldab,'Bracketed','F7.4',                  &
          'Details of factorization','None',rlabs,'None',clabs,80,0,ifail)

!       Print pivot indices'

        Write (nout,*)
        Write (nout,*) 'Pivot indices'
        Write (nout,99998) ipiv(1:n)

      Else
        Write (nout,99997) 'The (', info, ',', info, ')',                     &
          ' element of the factor U is zero'
      End If

99999 Format ((3X,4(' (',F7.4,',',F7.4,')',:)))
99998 Format (1X,I5,3I18)
99997 Format (1X,A,I3,A,I3,A,A)
    End Program f07bnfe
```

10.2 Program Data

```
F07BNF Example Program Data
  4  1  2                                                 :Values of N, KL and KU
 (-1.65,  2.26) ( -2.05, -0.85) (  0.97, -2.84)
 ( 0.00,  6.30) ( -1.48, -1.75) ( -3.99,  4.01) (  0.59, -0.48)
                ( -0.77,  2.83) ( -1.06,  1.94) (  3.33, -1.04)
                                (  4.48, -1.09) ( -0.46, -1.72) :End of matrix A
 (-1.06, 21.50) (-22.72,-53.90) ( 28.24,-38.60) (-34.56, 16.73) :End of vector B
```

10.3 Program Results

```
 F07BNF Example Program Results

 Solution
    (-3.0000, 2.0000) ( 1.0000,-7.0000) (-5.0000, 4.0000) ( 6.0000,-8.0000)

 Details of factorization
   ( 0.0000, 6.3000) (-1.4800,-1.7500) (-3.9900, 4.0100) ( 0.5900,-0.4800)
   ( 0.3587, 0.2619) (-0.7700, 2.8300) (-1.0600, 1.9400) ( 3.3300,-1.0400)
                     ( 0.2314, 0.6358) ( 4.9303,-3.0086) (-1.7692,-1.8587)
                                       ( 0.7604, 0.2429) ( 0.4338, 0.1233)

 Pivot indices
     2               3               3               4
```