NAG Toolbox for MATLAB

f07bp

1 Purpose

f07bp uses the LU factorization to compute the solution to a complex system of linear equations

$$AX = B, \quad A^{\mathrm{T}}X = B \quad \text{or} \quad A^{\mathrm{H}}X = B,$$

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. Error bounds on the solution and a condition estimate are also provided.

2 Syntax

```
[ab, afb, ipiv, equed, r, c, b, x, rcond, ferr, berr, rwork, info] =
f07bp(fact, trans, kl, ku, ab, afb, ipiv, equed, r, c, b, 'n', n,
'nrhs_p', nrhs_p)
```

3 Description

f07bp performs the following steps:

1. **Equilibration**

   The linear system to be solved may be badly scaled. However, the system can be equilibrated as a first stage by setting **fact** = 'E'. In this case, real scaling factors are computed and these factors then determine whether the system is to be equilibrated. Equilibrated forms of the systems $AX = B$ and $A^{\mathrm{T}}X = B$ are

   $$(D_RAD_C)\left(D_C{}^{-1}X\right) = D_RB$$

   and

   $$(D_RAD_C)^{\mathrm{T}}\left(D_R{}^{-1}X\right) = D_CB,$$

   respectively, where D_R and D_C are diagonal matrices, with positive diagonal elements, formed from the computed scaling factors.

   When equilibration is used, A will be overwritten by D_RAD_C and B will be overwritten by D_RB (or D_CB when the solution of $A^{\mathrm{T}}X = B$ or $A^{\mathrm{H}}X = B$ is sought).

2. **Factorization**

   The matrix A, or its scaled form, is copied and factored using the LU decomposition

   $$A = PLU,$$

   where P is a permutation matrix, L is a unit lower triangular matrix, and U is upper triangular.

   This stage can be by-passed when a factored matrix (with scaled matrices and scaling factors) are supplied; for example, as provided by a previous call to f07bp with the same matrix A.

3. **Condition Number Estimation**

   The LU factorization of A determines whether a solution to the linear system exists. If some diagonal element of U is zero, then U is exactly singular, no solution exists and the function returns with a failure. Otherwise the factorized form of A is used to estimate the condition number of the matrix A. If the reciprocal of the condition number is less than ***machine precision*** then a warning code is returned on final exit.

4. **Solution**

   The (equilibrated) system is solved for X ($D_C{}^{-1}X$ or $D_R{}^{-1}X$) using the factored form of A (D_RAD_C).

5. **Iterative Refinement**

Iterative refinement is applied to improve the computed solution matrix and to calculate error bounds and backward error estimates for the computed solution.

6. **Construct Solution Matrix X**

   If equilibration was used, the matrix X is premultiplied by D_C (if **trans** $=$ 'N') or D_R (if **trans** $=$ 'T' or 'C') so that it solves the original system before equilibration.

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 URL: http://www.netlib.org/lapack/lug

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

Higham N J 2002 *Accuracy and Stability of Numerical Algorithms* (2nd Edition) SIAM, Philadelphia

5 Parameters

5.1 Compulsory Input Parameters

1: **fact – string**

Specifies whether or not the factorized form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factorized.

fact $=$ 'F'

> **afb** and **ipiv** contain the factorized form of A. If **equed** $\neq$ 'N', the matrix A has been equilibrated with scaling factors given by **r** and **c**. **ab**, **afb** and **ipiv** are not modified.

fact $=$ 'N'

> The matrix A will be copied to **afb** and factorized.

fact $=$ 'E'

> The matrix A will be equilibrated if necessary, then copied to **afb** and factorized.

Constraint: **fact** $=$ 'F', 'N' or 'E'.

2: **trans – string**

Specifies the form of the system of equations.

trans $=$ 'N'

> $AX = B$ (No transpose).

trans $=$ 'T'

> $A^{\mathrm{T}}X = B$ (Transpose).

trans $=$ 'C'

> $A^{\mathrm{H}}X = B$ (Conjugate transpose).

Constraint: **trans** $=$ 'N', 'T' or 'C'.

3: **kl – int32 scalar**

k_l, the number of subdiagonals within the band of the matrix A.

Constraint: **kl** ≥ 0.

4: **ku – int32 scalar**

k_u, the number of superdiagonals within the band of the matrix A.

Constraint: $\mathbf{ku} \geq 0$.

5: **ab(ldab,$*$) – complex array**

The first dimension of the array **ab** must be at least $\mathbf{kl} + \mathbf{ku} + 1$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The n by n coefficient matrix A.

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

$$\mathbf{ab}(k_u + 1 + i - j, j) \qquad \text{for } \max(1j - k_u) \leq i \leq \min(nj + k_l).$$

See Section 8 for further details.

If **fact** = 'F' and **equed** $\neq$ 'N', AB must have been equilibrated by the scaling factors in **r** and/or **c**.

6: **afb(ldafb,$*$) – complex array**

The first dimension of the array **afb** must be at least $2 \times \mathbf{kl} + \mathbf{ku} + 1$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

If **fact** = 'N' or 'E', **afb** need not be set.

If **fact** = 'F', details of the LU factorization of the n by n band matrix A, as computed by f07br.

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

If **equed** $\neq$ 'N', **afb** is the factorized form of the equilibrated matrix A.

7: **ipiv($*$) – int32 array**

Note: the dimension of the array **ipiv** must be at least $\max(1, \mathbf{n})$.

If **fact** = 'N' or 'E', **ipiv** need not be set.

If **fact** = 'F', **ipiv** contains the pivot indices from the factorization $A = LU$, as computed by f07bd; row i of the matrix was interchanged with row **ipiv**(i).

8: **equed – string**

If **fact** = 'N' or 'E', **equed** need not be set.

If **fact** = 'F', **equed** must specify the form of the equilibration that was performed as follows:

if **equed** = 'N', no equilibration;
if **equed** = 'R', row equilibration, i.e., A has been premultiplied by D_R;
if **equed** = 'C', column equilibration, i.e., A has been postmultiplied by D_C;
if **equed** = 'B', both row and column equilibration, i.e., A has been replaced by $D_R A D_C$.

Constraint: if **fact** = 'F', **equed** = 'N', 'R', 'C' or 'B'.

9: **r($*$) – double array**

Note: the dimension of the array **r** must be at least $\max(1, \mathbf{n})$.

If **fact** = 'N' or 'E', **r** need not be set.

If **fact** = 'F' and **equed** = 'R' or 'B', **r** must contain the row scale factors for A, D_R; each element of **r** must be positive.

10: **c**$(*)$ **– double array**

Note: the dimension of the array **c** must be at least $\max(1, \mathbf{n})$.

If **fact** = 'N' or 'E', **c** need not be set.

If **fact** = 'F' or **equed** = 'C' or 'B', **c** must contain the column scale factors for A, D_C; each element of **c** must be positive.

11: **b**(**ldb**,$*$) **– complex array**

The first dimension of the array **b** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{nrhs\_p})$

The n by r right-hand side matrix B.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The second dimension of the array **ab** The second dimension of the array **afb** The dimension of the array **ipiv** The dimension of the array **r** The dimension of the array **c**.

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

Constraint: $\mathbf{n} \geq 0$.

2: **nrhs_p – int32 scalar**

Default: The second dimension of the array **b**.

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

Constraint: $\mathbf{nrhs\_p} \geq 0$.

5.3 Input Parameters Omitted from the MATLAB Interface

ldab, ldafb, ldb, ldx, work

5.4 Output Parameters

1: **ab**(**ldab**,$*$) **– complex array**

The first dimension of the array **ab** must be at least $\mathbf{kl} + \mathbf{ku} + 1$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

If **fact** = 'F' or 'N', or if **fact** = 'E' and **equed** = 'N', **ab** is not modified.

If **equed** $\neq$ 'N' then, if **info** ≥ 0, A is scaled as follows:

if **equed** = 'R', $A = D_rA$;
if **equed** = 'C', $A = AD_c$;
if **equed** = 'B', $A = D_rAD_c$.

2: **afb**(**ldafb**,$*$) **– complex array**

The first dimension of the array **afb** must be at least $2 \times \mathbf{kl} + \mathbf{ku} + 1$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

If **fact** = 'F', **afb** is unchanged from entry.

Otherwise, if **info** ≥ 0, then if **fact** = 'N', **afb** returns details of the LU factorization of the band matrix A, and if **fact** = 'E', **afb** returns details of the LU factorization of the equilibrated band matrix A (see the description of **ab** for the form of the equilibrated matrix).

3: **ipiv**$(*)$ **– int32 array**

Note: the dimension of the array **ipiv** must be at least $\max(1,\mathbf{n})$.

If **fact** $=$ 'F', **ipiv** is unchanged from entry.

Otherwise, if **info** ≥ 0, **ipiv** contains 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.

If **fact** $=$ 'N', the pivot indices are those corresponding to the factorization $A = LU$ of the original matrix A.

If **fact** $=$ 'E', the pivot indices are those corresponding to the factorization of $A = LU$ of the equilibrated matrix A.

4: **equed – string**

If **fact** $=$ 'F', **equed** is unchanged from entry.

Otherwise, if **info** ≥ 0, **equed** specifies the form of equilibration that was performed as specified above.

5: **r**$(*)$ **– double array**

Note: the dimension of the array **r** must be at least $\max(1,\mathbf{n})$.

If **fact** $=$ 'F', **r** is unchanged from entry.

Otherwise, if **info** ≥ 0 and **equed** $=$ 'R' or 'B', **r** contains the row scale factors for A, D_R, such that A is multiplied on the left by D_R; each element of **r** is positive.

6: **c**$(*)$ **– double array**

Note: the dimension of the array **c** must be at least $\max(1,\mathbf{n})$.

If **fact** $=$ 'F', **c** is unchanged from entry.

Otherwise, if **info** ≥ 0 and **equed** $=$ 'C' or 'B', **c** contains the row scale factors for A, D_C; each element of **c** is positive.

7: **b**(**ldb**,$*$) **– complex array**

The first dimension of the array **b** must be at least $\max(1,\mathbf{n})$

The second dimension of the array must be at least $\max(1,\mathbf{nrhs\_p})$

If **equed** $=$ 'N', **b** is not modified.

If **trans** $=$ 'N' and **equed** $=$ 'R' or 'B', **b** contains $D_R B$.

If **trans** $=$ 'T' or 'C' and **equed** $=$ 'C' or 'B', **b** contains $D_C B$.

8: **x**(**ldx**,$*$) **– complex array**

The first dimension of the array **x** must be at least $\max(1,\mathbf{n})$

The second dimension of the array must be at least $\max(1,\mathbf{nrhs\_p})$

If **info** $= 0$ or **info** $\geq N + 1$, the n by r solution matrix X to the original system of equations. Note that the arrays A and B are modified on exit if **equed** $\neq$ 'N', and the solution to the equilibrated system is $D_C^{-1}X$ if **trans** $=$ 'N' and **equed** $=$ 'C' or 'B', or $D_R^{-1}X$ if **trans** $=$ 'T' or 'C' and **equed** $=$ 'R' or 'B'.

9: **rcond – double scalar**

If **info** ≥ 0, an estimate of the reciprocal condition number of the matrix A (after equilibration if that is performed), computed as **rcond** $= 1/\left(\|A\|_1\|A^{-1}\|_1\right)$.

10: **ferr**$(*)$ – **double array**

Note: the dimension of the array **ferr** must be at least $\max(1, \textbf{nrhs\_p})$.

If **info** $= 0$ or **info** $\geq N + 1$, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \textbf{ferr}(j)$ where $\hat{x}_j$ is the jth column of the computed solution returned in the array **x** and x_j is the corresponding column of the exact solution X. The estimate is as reliable as the estimate for **rcond**, and is almost always a slight overestimate of the true error.

11: **berr**$(*)$ – **double array**

Note: the dimension of the array **berr** must be at least $\max(1, \textbf{nrhs\_p})$.

If **info** $= 0$ or **info** $\geq N + 1$, an estimate of the component-wise relative backward error of each computed solution vector $\hat{x}_j$ (i.e., the smallest relative change in any element of A or B that makes $\hat{x}_j$ an exact solution).

12: **rwork**$(*)$ – **double array**

Note: the dimension of the array **rwork** must be at least $\max(1, \mathbf{n})$.

If **info** $= 0$, **rwork**(1) contains the reciprocal pivot growth factor $\max|a_{ij}|/\max|u_{ij}|$. If **rwork**$(1)$ is much less than 1, then the stability of the LU factorization of the (equilibrated) matrix A could be poor. This also means that the solution X, condition estimator **rcond**, and forward error bound **ferr** could be unreliable. If the factorization fails with **info** $>$ 0leqN, **rwork**(1) contains the reciprocal pivot growth factor for the leading **info** columns of A.

13: **info** – **int32 scalar**

info $= 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

info $= -i$

If **info** $= -i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **fact**, 2: **trans**, 3: **n**, 4: **kl**, 5: **ku**, 6: **nrhs_p**, 7: **ab**, 8: **ldab**, 9: **afb**, 10: **ldafb**, 11: **ipiv**, 12: **equed**, 13: **r**, 14: **c**, 15: **b**, 16: **ldb**, 17: **x**, 18: **ldx**, 19: **rcond**, 20: **ferr**, 21: **berr**, 22: **work**, 23: **rwork**, 24: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

info > 0 and **info** $\leq N$

If **info** $= i$, u_{ii} is exactly zero. The factorization has been completed, but the factor U is exactly singular, so the solution and error bounds could not be computed. **rcond** $= 0$ is returned.

info $= N + 1$

U is nonsingular, but **rcond** is less than ***machine precision***, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of **rcond** would suggest.

7 Accuracy

For each right-hand side vector b, the computed solution $\hat{x}$ is the exact solution of a perturbed system of equations $(A + E)\hat{x} = b$, where

$$|E| \leq c(n)\epsilon P|L||U|,$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***. See Section 9.3 of Higham 2002 for further details.

If x is the true solution, then the computed solution $\hat{x}$ satisfies a forward error bound of the form

$$\frac{\|x-\hat{x}\|_\infty}{\|\hat{x}\|_\infty} \leq w_c\,\mathrm{cond}(A,\hat{x},b)$$

where $\mathrm{cond}(A,\hat{x},b) = \left\||A^{-1}|(|A||\hat{x}|+|b|)\right\|_\infty/\|\hat{x}\|_\infty \leq \mathrm{cond}(A) = \left\||A^{-1}||A|\right\|_\infty \leq \kappa_\infty(A)$. If $\hat{x}$ is the jth column of X, then w_c is returned in **berr**(j) and a bound on $\|x-\hat{x}\|_\infty/\|\hat{x}\|_\infty$ is returned in **ferr**(j). See Section 4.4 of Anderson *et al.* 1999 for further details.

8 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**:

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

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 condition number estimation typically requires between four and five solves and never more than eleven solves, following the factorization. The solution is then refined, and the errors estimated, using iterative refinement; see f07bv for information on the floating-point operations required.

In practice the condition number estimator is very reliable, but it can underestimate the true condition number; see Section 15.3 of Higham 2002 for further details.

The real analogue of this function is f07bb.

9 Example

```
fact = 'Equilibration';
trans = 'No transpose';
kl = int32(1);
ku = int32(2);
ab = [complex(0, 0), complex(0, 0), complex(0.97, -2.84), complex(0.59, -
0.48);
          complex(0,  0), complex(-2.05,  -0.85),  complex(-3.99,  +4.01),
complex(3.33, -1.04);
      complex(-1.65, +2.26), complex(-1.48, -1.75), complex(-1.06, +1.94),
complex(-0.46, -1.72);
         complex(0,  +6.3),  complex(-0.77,  +2.83),  complex(4.48,  -1.09),
complex(0,0)];
afb = complex(zeros(5, 4) );
ipiv = [int32(0);
     int32(0);
     int32(0);
     int32(0)];
equed = ' ';
r = [0;
     0;
     0;
     0];
c = [0;
     0;
     0;
     0];
b = [complex(-1.06, +21.5), complex(12.85, +2.84);
     complex(-22.72, -53.9), complex(-70.22, +21.57);
     complex(28.24, -38.6), complex(-20.73, -1.23);
     complex(-34.56, +16.73), complex(26.01, +31.97)];
[abOut, afbOut, ipivOut, equedOut, rOut, cOut, bOut, x, ...
 rcond, ferr, berr, rwork, info] = ...
    f07bp(fact, trans, kl, ku, ab, afb, ipiv, equed, r, c, b)
```

```
abOut =
         0                    0              0.9700 - 2.8400i   0.5900 -
0.4800i
         0              -2.0500 - 0.8500i  -3.9900 + 4.0100i   3.3300 -
1.0400i
   -1.6500 + 2.2600i   -1.4800 - 1.7500i   -1.0600 + 1.9400i   -0.4600 -
1.7200i
        0 + 6.3000i  -0.7700 + 2.8300i   4.4800 - 1.0900i       0
afbOut =
         0                   0                    0              0.5900 -
0.4800i
         0                   0              -3.9900 + 4.0100i   3.3300 -
1.0400i
         0              -1.4800 - 1.7500i   -1.0600 + 1.9400i   -1.7692 -
1.8587i
         0 + 6.3000i   -0.7700 + 2.8300i    4.9303 - 3.0086i    0.4338 +
0.1233i
   0.3587 + 0.2619i   0.2314 + 0.6358i   0.7604 + 0.2429i       0
ipivOut =
           2
           3
           3
           4
equedOut =
N
rOut =
    0.2558
    0.1250
    0.2288
    0.1795
cOut =
    1.0000
    1.2139
    1.0000
    1.0000
bOut =
  -1.0600 +21.5000i  12.8500 + 2.8400i
 -22.7200 -53.9000i -70.2200 +21.5700i
  28.2400 -38.6000i -20.7300 - 1.2300i
 -34.5600 +16.7300i  26.0100 +31.9700i
x =
  -3.0000 + 2.0000i   1.0000 + 6.0000i
   1.0000 - 7.0000i  -7.0000 - 4.0000i
  -5.0000 + 4.0000i   3.0000 + 5.0000i
   6.0000 - 8.0000i  -8.0000 + 2.0000i
rcond =
    0.0096
ferr =
   1.0e-13 *
    0.3607
    0.4384
berr =
   1.0e-16 *
    0.4459
    0.8246
rwork =
    1.0000
    0.0000
    0.0000
    0.0000
```

```
info =
           0
```