NAG Fortran Library Routine Document

F07PBF (DSPSVX)

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

F07PBF (DSPSVX) uses the diagonal pivoting factorization

$$A = UDU^T \quad \text{or} \quad A = LDL^T$$

to compute the solution to a real system of linear equations

$$AX = B,$$

where A is an n by n symmetric matrix stored in packed format and X and B are n by r matrices. Error bounds on the solution and a condition estimate are also provided.

2 Specification

```
      SUBROUTINE F07PBF (FACT, UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X, LDX,
     1                   RCOND, FERR, BERR, WORK, IWORK, INFO)
      INTEGER            N, NRHS, IPIV(*), LDB, LDX, IWORK(*), INFO
      double precision   AP(*), AFP(*), B(LDB,*), X(LDX,*), RCOND, FERR(*),
     1                   BERR(*), WORK(*)
      CHARACTER*1        FACT, UPLO
```

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

3 Description

The following steps are performed:

1. If FACT = 'N', the diagonal pivoting method is used to factor A as $A = UDU^T$, if UPLO = 'U' or $A = LDL^T$, if UPLO = 'L', where U (or L) is a product of permutation and unit upper (lower) triangular matrices and D is symmetric and block diagonal with 1 by 1 and 2 by 2 diagonal blocks.
2. If some $d_{ii} = 0$, so that D is exactly singular, then the routine returns with INFO $= i$. Otherwise, the factored 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***, INFO $= \mathrm{N} + 1$ is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.
3. The system of equations is solved for X using the factored form of A.
4. Iterative refinement is applied to improve the computed solution matrix and to calculate error bounds and backward error estimates for it.

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

1: FACT – CHARACTER*1 *Input*

On entry: specifies whether or not the factored form of A has been supplied on entry:

if FACT = 'F' on entry, AFP and IPIV contain the factored form of A. AP, AFP and IPIV will not be modified;
if FACT = 'N', the matrix A will be copied to AFP and factored.

Constraint: FACT = 'F' or 'N'.

2: UPLO – CHARACTER*1 *Input*

On entry: if UPLO = 'U', the upper triangle of A is stored.

If UPLO = 'L', the lower triangle of A is stored.

Constraint: UPLO = 'U' or 'L'.

3: N – INTEGER *Input*

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

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

4: NRHS – INTEGER *Input*

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

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

5: AP($*$) – ***double precision*** array *Input*

Note: the dimension of the array AP must be at least $\max(\mathrm{N} \times (\mathrm{N}+1)/2)$.

On entry: the upper or lower triangle of the symmetric matrix A, packed columnwise in a linear array. The jth column of A is stored in the array AP as follows:

if UPLO = 'U', $\mathrm{AP}(i + (j-1) \times j/2) = a_{ij}$ for $1 \leq i \leq j$;
if UPLO = 'L', $\mathrm{AP}(i + (j-1) \times (2 \times n - j)/2) = a_{ij}$ for $j \leq i \leq n$.

6: AFP($*$) – ***double precision*** array *Input/Output*

Note: the dimension of the array AFP must be at least $\max(\mathrm{N} \times (\mathrm{N}+1)/2)$.

On entry: if FACT = 'F', AFP contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = UDU^T$ or $A = LDL^T$ as computed by F07PDF (DSPTRF), stored as a packed triangular matrix in the same storage format as A.

On exit: if FACT = 'N', AFP contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = UDU^T$ or $A = LDL^T$ as computed by F07PDF (DSPTRF), stored as a packed triangular matrix in the same storage format as A.

7: IPIV($*$) – INTEGER array *Input/Output*

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

On entry: if FACT = 'F', IPIV contains details of the interchanges and the block structure of D, as determined by F07PDF (DSPTRF). If $\mathrm{IPIV}(k) > 0$, then rows and columns k and $\mathrm{IPIV}(k)$ were interchanged and $D(k,k)$ is a 1 by 1 diagonal block. If UPLO = 'U' and $\mathrm{IPIV}(k) = \mathrm{IPIV}(k-1) < 0$, then rows and columns $k-1$ and $-\mathrm{IPIV}(k)$ were interchanged and $D(k-1:k, k-1:k)$ is a 2 by 2 diagonal block. If UPLO = 'L' and $\mathrm{IPIV}(k) = \mathrm{IPIV}(k+1) < 0$, then rows and columns $k+1$ and $-\mathrm{IPIV}(k)$ were interchanged and $D(k:k+1, k:k+1)$ is a 2 by 2 diagonal block.

On exit: if FACT = 'N', IPIV contains details of the interchanges and the block structure of D, as determined by F07PDF (DSPTRF).

8: B(LDB,$*$) – ***double precision*** 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.

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07PBF (DSPSVX) is called.

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

10: X(LDX,$*$) – ***double precision*** array *Output*

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

On exit: if $\mathrm{INFO} = 0$ or $\mathrm{INFO} = \mathrm{N} + 1$, the n by r solution matrix X.

11: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07PBF (DSPSVX) is called.

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

12: RCOND – ***double precision*** *Output*

On exit: the estimate of the reciprocal condition number of the matrix A. If RCOND is less than the ***machine precision*** (in particular, if $\mathrm{RCOND} = 0$), the matrix is singular to working precision. This condition is indicated by a return code of $\mathrm{INFO} > 0$.

13: FERR($*$) – ***double precision*** array *Output*

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

On exit: if $\mathrm{INFO} = 0$ or $\mathrm{INFO} = \mathrm{N} + 1$, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \mathrm{FERR}(j)$ 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.

14: BERR($*$) – ***double precision*** array *Output*

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

On exit: if $\mathrm{INFO} = 0$ or $\mathrm{INFO} = \mathrm{N} + 1$, an estimate of the componentwise 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).

15: WORK($*$) – ***double precision*** array *Workspace*

Note: the dimension of the array WORK must be at least $\max(1, 3 \times \mathrm{N})$.

16: IWORK($*$) – INTEGER array *Workspace*

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

17: INFO – INTEGER *Output*

On exit: $\mathrm{INFO} = 0$ 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 argument had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0

If INFO $= i$ and $i \leq$ N, d_{ii} is exactly zero. The factorization has been completed but the factor D is exactly singular, so the solution and error bounds could not be computed. RCOND $= 0$ is returned.

If INFO $= i$ and $i = \mathrm{N} + 1$, D 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

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

where ϵ is the ***machine precision***. See Chapter 11 of Higham (2002) for further details.

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

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

where $\operatorname{cond}(A, \hat{x}, b) = \left\| |A^{-1}|(|A||\hat{x}| + |b|) \right\|_\infty / \|\hat{x}\|_\infty \leq \operatorname{cond}(A) = \left\| |A^{-1}||A| \right\|_\infty \leq \kappa_\infty(A)$. 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 factorization of A requires approximately $\frac{1}{3}n^3$ floating point operations.

For each right-hand side, computation of the backward error involves a minimum of $4n^2$ floating point operations. Each step of iterative refinement involves an additional $6n^2$ operations. At most 5 steps of iterative refinement are performed, but usually only 1 or 2 steps are required. Estimating the forward error involves solving a number of systems of equations of the form $AX = B$; the number is usually 4 or 5 and never more than 11. Each solution involves approximately $2n^2$ operations.

The complex analogues of this routine are F07PPF (ZHPSVX) for Hermitian matrices, and F07QPF (ZSPSVX) for symmetric matrices.

9 Example

To solve the equations

$$Ax = b,$$

where A is the symmetric matrix

$$A = \begin{pmatrix} -1.81 & 2.06 & 0.63 & -1.15 \\ 2.06 & 1.15 & 1.87 & 4.20 \\ 0.63 & 1.87 & -0.21 & 3.87 \\ -1.15 & 4.20 & 3.87 & 2.07 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 0.96 & 3.93 \\ 6.07 & 19.25 \\ 8.38 & 9.90 \\ 9.50 & 27.85 \end{pmatrix}.$$

Error estimates for the solutions, and an estimate of the reciprocal of the condition number of the matrix A are also output.

9.1 Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F07PBF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=8)
      INTEGER          LDB, LDX, NRHSMX
      PARAMETER        (LDB=NMAX,LDX=NMAX,NRHSMX=NMAX)
      CHARACTER        UPLO
      PARAMETER        (UPLO='U')
*     .. Local Scalars ..
      DOUBLE PRECISION RCOND
      INTEGER          I, IFAIL, INFO, J, N, NRHS
*     .. Local Arrays ..
      DOUBLE PRECISION AFP((NMAX*(NMAX+1))/2), AP((NMAX*(NMAX+1))/2),
     +                 B(LDB,NRHSMX), BERR(NRHSMX), FERR(NRHSMX),
     +                 WORK(3*NMAX), X(LDX,NRHSMX)
      INTEGER          IPIV(NMAX), IWORK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         DSPSVX, X04CAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F07PBF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NRHS
      IF (N.LE.NMAX .AND. NRHS.LE.NRHSMX) THEN
*
*        Read the upper or lower triangular part of the matrix A from
*        data file
*
         IF (UPLO.EQ.'U') THEN
            READ (NIN,*) ((AP(I+(J*(J-1))/2),J=I,N),I=1,N)
         ELSE IF (UPLO.EQ.'L') THEN
            READ (NIN,*) ((AP(I+((2*N-J)*(J-1))/2),J=1,I),I=1,N)
         END IF
*
*        Read B from data file
*
         READ (NIN,*) ((B(I,J),J=1,NRHS),I=1,N)
*
*        Solve the equations AX = B for X
*
         CALL DSPSVX('Not factored',UPLO,N,NRHS,AP,AFP,IPIV,B,LDB,X,LDX,
     +               RCOND,FERR,BERR,WORK,IWORK,INFO)
*
         IF ((INFO.EQ.0) .OR. (INFO.EQ.N+1)) THEN
*
*           Print solution, error bounds and condition number
*
            IFAIL = 0
            CALL X04CAF('General',' ',N,NRHS,X,LDX,'Solution(s)',IFAIL)
*
```

```
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Backward errors (machine-dependent)'
            WRITE (NOUT,99999) (BERR(J),J=1,NRHS)
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +        'Estimated forward error bounds (machine-dependent)'
            WRITE (NOUT,99999) (FERR(J),J=1,NRHS)
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Estimate of reciprocal condition number'
            WRITE (NOUT,99999) RCOND
            WRITE (NOUT,*)
*
            IF (INFO.EQ.N+1) THEN
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +           'The matrix A is singular to working precision'
            END IF
         ELSE
            WRITE (NOUT,99998) 'The diagonal block ', INFO,
     +        ' of D is zero'
         END IF
      ELSE
         WRITE (NOUT,*) 'NMAX and/or NRHSMX too small'
      END IF
      STOP
*
99999 FORMAT ((3X,1P,7E11.1))
99998 FORMAT (1X,A,I3,A)
      END
```

9.2 Program Data

```
F07PBF Example Program Data
  4     2                  :Values of N and NRHS
 -1.81   2.06   0.63  -1.15
         1.15   1.87   4.20
               -0.21   3.87
                       2.07 :End of matrix A
  0.96   3.93
  6.07  19.25
  8.38   9.90
  9.50  27.85              :End of matrix B
```

9.3 Program Results

```
 F07PBF Example Program Results

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

 Backward errors (machine-dependent)
       6.7E-17    5.2E-17

 Estimated forward error bounds (machine-dependent)
       2.4E-14    3.2E-14

 Estimate of reciprocal condition number
       1.3E-02
```