

NAG Library Routine Document

F07FPF (ZPOSVX)

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

F07FPF (ZPOSVX) uses the Cholesky factorization

$$A = U^H U \quad \text{or} \quad A = LL^H$$

to compute the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n Hermitian positive definite matrix and X and B are n by r matrices. Error bounds on the solution and a condition estimate are also provided.

2 Specification

```
SUBROUTINE F07FPF (FACT, UPLO, N, NRHS, A, LDA, AF, LDAF, EQUED, S, B, LDB,      &
                  X, LDX, RCOND, FERR, BERR, WORK, RWORK, INFO)
```

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

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

3 Description

F07FPF (ZPOSVX) performs the following steps:

1. If FACT = 'E', real diagonal scaling factors, D_S , are computed to equilibrate the system:

$$(D_S A D_S)(D_S^{-1} X) = D_S B.$$

Whether or not the system will be equilibrated depends on the scaling of the matrix A , but if equilibration is used, A is overwritten by $D_S A D_S$ and B by $D_S B$.

2. If FACT = 'N' or 'E', the Cholesky decomposition is used to factor the matrix A (after equilibration if FACT = 'E') as $A = U^H U$ if UPLO = 'U' or $A = LL^H$ if UPLO = 'L', where U is an upper triangular matrix and L is a lower triangular matrix.
3. If the leading i by i principal minor of A is not positive definite, 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 = $N + 1$ is returned as a warning, but the routine still goes on to solve for X and compute error bounds as described below.
4. The system of equations is solved for X using the factored form of A .
5. Iterative refinement is applied to improve the computed solution matrix and to calculate error bounds and backward error estimates for it.
6. If equilibration was used, the matrix X is premultiplied by D_S 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 <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 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'
AF contains the factorized form of A . If EQUED = 'Y', the matrix A has been equilibrated with scaling factors given by S . A and AF will not be modified.
- FACT = 'N'
The matrix A will be copied to AF and factorized.
- FACT = 'E'
The matrix A will be equilibrated if necessary, then copied to AF and factorized.
- Constraint:* FACT = 'F', 'N' or 'E'.
- 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:* $N \geq 0$.
- 4: NRHS – INTEGER *Input*
- On entry:* r , the number of right-hand sides, i.e., the number of columns of the matrix B .
- Constraint:* NRHS ≥ 0 .
- 5: A(LDA,*) – COMPLEX (KIND=nag_wp) array *Input/Output*
- Note:** the second dimension of the array A must be at least $\max(1, N)$.
- On entry:* the n by n Hermitian matrix A .
- If FACT = 'F' and EQUED = 'Y', A must have been equilibrated by the scaling factor in S as $D_S A D_S$.
- If UPLO = 'U', the upper triangular part of A must be stored and the elements of the array below the diagonal are not referenced.
- If UPLO = 'L', the lower triangular part of A must be stored and the elements of the array above the diagonal are not referenced.
- On exit:* if FACT = 'F' or 'N', or if FACT = 'E' and EQUED = 'N', A is not modified.
If FACT = 'E' and EQUED = 'Y', A is overwritten by $D_S A D_S$.

- 6: LDA – INTEGER *Input*
On entry: the first dimension of the array A as declared in the (sub)program from which F07FPF (ZPOSVX) is called.
Constraint: $LDA \geq \max(1, N)$.
- 7: AF(LDAF,*) – COMPLEX (KIND=nag_wp) array *Input/Output*
Note: the second dimension of the array AF must be at least $\max(1, N)$.
On entry: if FACT = 'F', AF contains the triangular factor U or L from the Cholesky factorization $A = U^H U$ or $A = LL^H$, in the same storage format as A. If EQUED \neq 'N', AF is the factorized form of the equilibrated matrix $D_S A D_S$.
On exit: if FACT = 'N', AF returns the triangular factor U or L from the Cholesky factorization $A = U^H U$ or $A = LL^H$ of the original matrix A.
 If FACT = 'E', AF returns the triangular factor U or L from the Cholesky factorization $A = U^H U$ or $A = LL^H$ of the equilibrated matrix A (see the description of A for the form of the equilibrated matrix).
- 8: LDAF – INTEGER *Input*
On entry: the first dimension of the array AF as declared in the (sub)program from which F07FPF (ZPOSVX) is called.
Constraint: $LDAF \geq \max(1, N)$.
- 9: EQUED – CHARACTER(1) *Input/Output*
On entry: 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 = 'Y', equilibration was performed, i.e., A has been replaced by $D_S A D_S$.
On exit: if FACT = 'F', EQUED is unchanged from entry.
 Otherwise, if no constraints are violated, EQUED specifies the form of the equilibration that was performed as specified above.
Constraint: if FACT = 'F', EQUED = 'N' or 'Y'.
- 10: S(*) – REAL (KIND=nag_wp) array *Input/Output*
Note: the dimension of the array S must be at least $\max(1, N)$.
On entry: if FACT = 'N' or 'E', S need not be set.
 If FACT = 'F' and EQUED = 'Y', S must contain the scale factors, D_S , for A; each element of S must be positive.
On exit: if FACT = 'F', S is unchanged from entry.
 Otherwise, if no constraints are violated and EQUED = 'Y', S contains the scale factors, D_S , for A; each element of S is positive.
- 11: 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 EQUED = 'N', B is not modified.
 If EQUED = 'Y', B is overwritten by $D_S B$.

- 12: LDB – INTEGER *Input*
On entry: the first dimension of the array B as declared in the (sub)program from which F07FPF (ZPOSVX) is called.
Constraint: $LDB \geq \max(1, N)$.
- 13: X(LDX,*) – COMPLEX (KIND=nag_wp) array *Output*
Note: the second dimension of the array X must be at least $\max(1, NRHS)$.
On exit: if INFO = 0 or 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 = 'Y', and the solution to the equilibrated system is $D_S^{-1}X$.
- 14: LDX – INTEGER *Input*
On entry: the first dimension of the array X as declared in the (sub)program from which F07FPF (ZPOSVX) is called.
Constraint: $LDX \geq \max(1, N)$.
- 15: RCOND – REAL (KIND=nag_wp) *Output*
On exit: if no constraints are violated, an estimate of the reciprocal condition number of the matrix A (after equilibration if that is performed), computed as $RCOND = 1.0 / (\|A\|_1 \|A^{-1}\|_1)$.
- 16: FERR(NRHS) – REAL (KIND=nag_wp) array *Output*
On exit: if INFO = 0 or N + 1, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq FERR(j)$ where \hat{x}_j is the j th 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.
- 17: BERR(NRHS) – REAL (KIND=nag_wp) array *Output*
On exit: if INFO = 0 or 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).
- 18: WORK(2 × N) – COMPLEX (KIND=nag_wp) array *Workspace*
- 19: RWORK(N) – REAL (KIND=nag_wp) array *Workspace*
- 20: 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 i th argument had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0 and INFO ≤ N

If INFO = i and $i \leq N$, the leading minor of order i of A is not positive definite, so the factorization could not be completed, and the solution has not been computed. RCOND = 0.0 is returned.

INFO = N + 1

The triangular matrix U (or L) 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 x is the exact solution of a perturbed system of equations $(A + E)x = b$, where

$$\text{if UPLO} = \text{'U'}, |E| \leq c(n)\epsilon|U^H||U|;$$

$$\text{if UPLO} = \text{'L'}, |E| \leq c(n)\epsilon|L||L^H|,$$

$c(n)$ is a modest linear function of n , and ϵ is the *machine precision*. See Section 10.1 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

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

where $\text{cond}(A, \hat{x}, b) = \frac{\| |A^{-1}|(|A||\hat{x}| + |b|) \|_\infty}{\|\hat{x}\|_\infty} \leq \text{cond}(A) = \| |A^{-1}| |A| \|_\infty \leq \kappa_\infty(A)$. If \hat{x} is the j th 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 $\frac{4}{3}n^3$ floating point operations.

For each right-hand side, computation of the backward error involves a minimum of $16n^2$ floating point operations. Each step of iterative refinement involves an additional $24n^2$ 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 equations of the form $Ax = b$; the number is usually 4 or 5 and never more than 11. Each solution involves approximately $8n^2$ operations.

The real analogue of this routine is F07FBF (DPOSVX).

9 Example

This example solves the equations

$$AX = B,$$

where A is the Hermitian positive definite matrix

$$A = \begin{pmatrix} 3.23 & 1.51 - 1.92i & 1.90 + 0.84i & 0.42 + 2.50i \\ 1.51 + 1.92i & 3.58 & -0.23 + 1.11i & -1.18 + 1.37i \\ 1.90 - 0.84i & -0.23 - 1.11i & 4.09 & 2.33 - 0.14i \\ 0.42 - 2.50i & -1.18 - 1.37i & 2.33 + 0.14i & 4.29 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 3.93 - 6.14i & 1.48 + 6.58i \\ 6.17 + 9.42i & 4.64 - 4.75i \\ -7.17 - 21.83i & -4.91 + 2.29i \\ 1.99 - 14.38i & 7.64 - 10.79i \end{pmatrix}.$$

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

9.1 Program Text

Program f07fpfe

```

!      F07FPF Example Program Text
!
!      Mark 24 Release. NAG Copyright 2012.
!
!      .. Use Statements ..
!      Use nag_library, Only: nag_wp, x04dbf, zposvx
!      .. Implicit None Statement ..
!      Implicit None
!      .. Parameters ..
!      Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
!      Real (Kind=nag_wp)         :: rcond
!      Integer                    :: i, ifail, info, lda, ldaf, ldb, ldx, &
!                                n, nrhs
!      Character (1)              :: equed
!      .. Local Arrays ..
!      Complex (Kind=nag_wp), Allocatable :: a(:,,:), af(:,:), b(:,:), work(:), &
!                                x(:,:)
!      Real (Kind=nag_wp), Allocatable :: berr(:), ferr(:), rwork(:), s(:)
!      Character (1)                  :: clabs(1), rlabs(1)
!      .. Executable Statements ..
!      Write (nout,*) 'F07FPF Example Program Results'
!      Write (nout,*)
!      Flush (nout)
!      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,nrhs), &
!                berr(nrhs),ferr(nrhs),rwork(n),s(n))
!
!      Read the upper triangular part of A from data file
!
!      Read (nin,*)(a(i,i:n),i=1,n)
!
!      Read B from data file
!
!      Read (nin,*)(b(i,1:nrhs),i=1,n)
!
!      Solve the equations AX = B for X
!      The NAG name equivalent of zposvx is f07fpf
!      Call zposvx('Equilibration','Upper',n,nrhs,a,lda,af,ldaf,equed,s,b,ldb, &
!                x,ldx,rcond,ferr,berr,work,rwork,info)
!
!      If ((info==0) .Or. (info==n+1)) Then
!
!      Print solution, error bounds, condition number and the form
!      of equilibration
!
!      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,*)
!      Write (nout,*) 'Estimated forward error bounds (machine-dependent)'
!      Write (nout,99999) ferr(1:nrhs)
!      Write (nout,*)
!      Write (nout,*) 'Estimate of reciprocal condition number'
!      Write (nout,99999) rcond

```

```

Write (nout,*)
If (equed=='N') Then
  Write (nout,*) 'A has not been equilibrated'
Else If (equed=='Y') Then
  Write (nout,*) &
  'A has been row and column scaled as diag(S)*A*diag(S)'
End If

If (info==n+1) Then
  Write (nout,*)
  Write (nout,*) 'The matrix A is singular to working precision'
End If
Else
  Write (nout,99998) 'The leading minor of order ', info, &
  ' is not positive definite'
End If

99999 Format ((3X,1P,7E11.1))
99998 Format (1X,A,I3,A)
End Program f07fpfe

```

9.2 Program Data

F07FPF Example Program Data

```

  4      2                                     :Values of N and NRHS
( 3.23,  0.00) ( 1.51, -1.92) ( 1.90,  0.84) ( 0.42,  2.50)
              ( 3.58,  0.00) (-0.23,  1.11) (-1.18,  1.37)
              ( 4.09,  0.00) ( 2.33, -0.14)
              ( 4.29,  0.00) :End of matrix A

( 3.93, -6.14) ( 1.48,  6.58)
( 6.17,  9.42) ( 4.65, -4.75)
(-7.17,-21.83) (-4.91,  2.29)
( 1.99,-14.38) ( 7.64,-10.79)                                     :End of matrix B

```

9.3 Program Results

F07FPF Example Program Results

Solution(s)

```

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

```

Backward errors (machine-dependent)

```

5.9E-17    4.8E-17

```

Estimated forward error bounds (machine-dependent)

```

6.0E-14    7.2E-14

```

Estimate of reciprocal condition number

```

6.6E-03

```

A has not been equilibrated
