NAG Toolbox

nag_lapack_dppsvx (f07gb)

1 Purpose

nag_lapack_dppsvx (f07gb) uses the Cholesky factorization

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

to compute the solution to a real system of linear equations

$$AX = B,$$

where A is an n by n symmetric positive definite 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 Syntax

```
[ap, afp, equed, s, b, x, rcond, ferr, berr, info] = nag_lapack_dppsvx(fact,
uplo, ap, afp, equed, s, b, 'n', n, 'nrhs_p', nrhs_p)
```

```
[ap, afp, equed, s, b, x, rcond, ferr, berr, info] = f07gb(fact, uplo, ap, afp,
equed, s, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

nag_lapack_dppsvx (f07gb) performs the following steps:

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

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

   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 $A = U^{\mathrm{T}}U$ if **uplo** = 'U' or $A = LL^{\mathrm{T}}$ 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 function 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** $\geq$ **n** $+ 1$ is returned as a warning, but the function 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

5.1 Compulsory Input Parameters

1: **fact** – CHARACTER(1)

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'
: **afp** contains the factorized form of A. If **equed** = 'Y', the matrix A has been equilibrated with scaling factors given by **s**. **ap** and **afp** will not be modified.

fact = 'N'
: The matrix A will be copied to **afp** and factorized.

fact = 'E'
: The matrix A will be equilibrated if necessary, then copied to **afp** and factorized.

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

2: **uplo** – CHARACTER(1)

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: **ap**(:) – REAL (KIND=nag_wp) array

The dimension of the array **ap** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$

If **fact** = 'F' and **equed** = 'Y', **ap** must contain the equilibrated matrix $D_S A D_S$; otherwise, **ap** must contain the n by n symmetric matrix A, packed by columns.

More precisely,

if **uplo** = 'U', the upper triangle of A must be stored with element A_{ij} in $\mathbf{ap}(i + j(j-1)/2)$ for $i \leq j$;

if **uplo** = 'L', the lower triangle of A must be stored with element A_{ij} in $\mathbf{ap}(i + (2n-j)(j-1)/2)$ for $i \geq j$.

4: **afp**(:) – REAL (KIND=nag_wp) array

The dimension of the array **afp** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$

If **fact** = 'F', **afp** contains the triangular factor U or L from the Cholesky factorization $A = U^{\mathrm{T}}U$ or $A = LL^{\mathrm{T}}$, in the same storage format as **ap**. If **equed** = 'Y', **afp** is the factorized form of the equilibrated matrix $D_S A D_S$.

5: **equed** – CHARACTER(1)

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 $D_S A D_S$.

Constraint: if **fact** = 'F', **equed** = 'N' or 'Y'.

6: **s**(:) – REAL (KIND=nag_wp) array

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

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.

7: **b**(ldb, :) – REAL (KIND=nag_wp) array

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

The second dimension of the array **b** 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** – INTEGER

Default: the first dimension of the array **b** and the dimension of the array **s**.

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

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

2: **nrhs_p** – INTEGER

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 Output Parameters

1: **ap**(:) – REAL (KIND=nag_wp) array

The dimension of the array **ap** will be $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$

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

If **fact** = 'E' and **equed** = 'Y', **ap** stores $D_S A D_S$.

2: **afp**(:) – REAL (KIND=nag_wp) array

The dimension of the array **afp** will be $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$

If **fact** = 'N' or if **fact** = 'E' and **equed** = 'N', **afp** returns the triangular factor U or L from the Cholesky factorization $A = U^{\mathrm{T}}U$ or $A = LL^{\mathrm{T}}$ of the original matrix A.

If **fact** = 'E' and **equed** = 'Y', **afp** returns the triangular factor U or L from the Cholesky factorization $A = U^{\mathrm{T}}U$ or $A = LL^{\mathrm{T}}$ of the equilibrated matrix A (see the description of **ap** for the form of the equilibrated matrix).

3: **equed** – CHARACTER(1)

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.

4: **s**(:) – REAL (KIND=nag_wp) array

The dimension of the array **s** will be $\max(1, \mathbf{n})$

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.

5: **b**(ldb, :) – REAL (KIND=nag_wp) array

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

The second dimension of the array **b** will be $\max(1, \mathbf{nrhs\_p})$.

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

If **equed** = 'Y', **b** stores $D_S B$.

6: **x**(ldx, :) – REAL (KIND=nag_wp) array

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

The second dimension of the array **x** will be $\max(1, \mathbf{nrhs\_p})$.

If **info** $= 0$ or $\mathbf{n} + 1$, 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 $D_S^{-1}X$.

7: **rcond** – REAL (KIND=nag_wp)

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** $= 1.0/\left(\|A\|_1\|A^{-1}\|_1\right)$.

8: **ferr**(**nrhs_p**) – REAL (KIND=nag_wp) array

If **info** $= 0$ or $\mathbf{n} + 1$, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \mathbf{ferr}(j)$ 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.

9: **berr**(**nrhs_p**) – REAL (KIND=nag_wp) array

If **info** $= 0$ or $\mathbf{n} + 1$, 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).

10: **info** – INTEGER

info $= 0$ unless the function 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 and **info** $\leq$ **n**

The leading minor of order ⟨*value*⟩ 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 $= \mathbf{n} + 1$ (*warning*)

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

if **uplo** = 'U', $|E| \leq c(n)\epsilon|U^{\mathrm{T}}||U|$;

if **uplo** = 'L', $|E| \leq c(n)\epsilon|L||L^{\mathrm{T}}|$,

$c(n)$ 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

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

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 $\mathbf{berr}(j)$ and a bound on $\|x - \hat{x}\|_\infty / \|\hat{x}\|_\infty$ is returned in $\mathbf{ferr}(j)$. 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 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 $2n^2$ operations.

The complex analogue of this function is nag_lapack_zppsvx (f07gp).

9 Example

This example solves the equations

$$AX = B,$$

where A is the symmetric positive definite matrix

$$A = \begin{pmatrix} 4.16 & -3.12 & 0.56 & -0.10 \\ -3.12 & 5.03 & -0.83 & 1.18 \\ 0.56 & -0.83 & 0.76 & 0.34 \\ -0.10 & 1.18 & 0.34 & 1.18 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 8.70 & 8.30 \\ -13.35 & 2.13 \\ 1.89 & 1.61 \\ -4.14 & 5.00 \end{pmatrix}.$$

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

```
function f07gb_example

fprintf('f07gb example results\n\n');

% Symmetric matrix A, upper triangular part packed in ap
uplo = 'U';
n = 4;
ap = [4.16  ...
     -3.12  5.03  ...
      0.56 -0.83  0.76  ...
     -0.10  1.18  0.34  1.18];

% Rhs
b = [  8.70, 8.30;
     -13.35, 2.13;
       1.89, 1.61;
      -4.14, 5.00];

% Input parameters
fact  = 'Equilibration';
uplo  = 'U';
afp   = ap;
equed = ' ';
s      = zeros(n,1);

% Solve
[ap, afp, equed, s, b, x, rcond, ferr, berr, info] = ...
  f07gb( ...
         fact, uplo, ap, afp, equed, s, b);

disp('Solution(s)');
disp(x);
disp('Backward errors (machine-dependent)');
fprintf('%10.1e',berr);
fprintf('\n');
disp('Estimated forward error bounds (machine-dependent)');
fprintf('%10.1e',ferr);
fprintf('\n\n');
disp('Estimate of reciprocal condition number');
fprintf('%10.1e\n\n',rcond);

if equed=='N'
  fprintf('A has not been equilibrated\n');
else
  fprintf('A has been equilibrated\n');
end
```

9.2 Program Results

```
f07gb example results

Solution(s)
    1.0000    4.0000
   -1.0000    3.0000
    2.0000    2.0000
   -3.0000    1.0000

Backward errors (machine-dependent)
   6.7e-17   8.1e-17
Estimated forward error bounds (machine-dependent)
   2.3e-14   2.3e-14

Estimate of reciprocal condition number
   1.0e-02

A has not been equilibrated
```