NAG Toolbox

nag_lapack_dpftri (f07wj)

1 Purpose

nag_lapack_dpftri (f07wj) computes the inverse of a real symmetric positive definite matrix using the Cholesky factorization computed by nag_lapack_dpftrf (f07wd) stored in Rectangular Full Packed (RFP) format.

2 Syntax

```
[ar, info] = nag_lapack_dpftri(transr, uplo, n, ar)
[ar, info] = f07wj(transr, uplo, n, ar)
```

3 Description

nag_lapack_dpftri (f07wj) is used to compute the inverse of a real symmetric positive definite matrix A, stored in RFP format. The RFP storage format is described in Section 3.2.3 in the F07 Chapter Introduction. The function must be preceded by a call to nag_lapack_dpftrf (f07wd), which computes the Cholesky factorization of A.

If **uplo** = 'U', $A = U^{\mathrm{T}}U$ and A^{-1} is computed by first inverting U and then forming $(U^{-1})U^{-\mathrm{T}}$.

If **uplo** = 'L', $A = LL^{\mathrm{T}}$ and A^{-1} is computed by first inverting L and then forming $L^{-\mathrm{T}}(L^{-1})$.

4 References

Du Croz J J and Higham N J (1992) Stability of methods for matrix inversion *IMA J. Numer. Anal.* **12** 1–19

Gustavson F G, Waśniewski J, Dongarra J J and Langou J (2010) Rectangular full packed format for Cholesky's algorithm: factorization, solution, and inversion *ACM Trans. Math. Software* **37, 2**

5 Parameters

5.1 Compulsory Input Parameters

1: **transr** – CHARACTER(1)

Specifies whether the RFP representation of A is normal or transposed.

transr = 'N'
The matrix A is stored in normal RFP format.

transr = 'T'
The matrix A is stored in transposed RFP format.

Constraint: **transr** = 'N' or 'T'.

2: **uplo** – CHARACTER(1)

Specifies how A has been factorized.

uplo = 'U'
$A = U^{\mathrm{T}}U$, where U is upper triangular.

uplo = 'L'
$A = LL^{\mathrm{T}}$, where L is lower triangular.

Constraint: **uplo** = 'U' or 'L'.

3: **n** – INTEGER

n, the order of the matrix A.

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

4: **ar**$(\mathbf{n} \times (\mathbf{n}+\mathbf{1})/\mathbf{2})$ – REAL (KIND=nag_wp) array

The Cholesky factorization of A stored in RFP format, as returned by nag_lapack_dpftrf (f07wd).

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **ar**$(\mathbf{n} \times (\mathbf{n}+\mathbf{1})/\mathbf{2})$ – REAL (KIND=nag_wp) array

The factorization stores the n by n matrix A^{-1} stored in RFP format.

2: **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 (*warning*)

The leading minor of order ⟨*value*⟩ is not positive definite and the factorization could not be completed. Hence A itself is not positive definite. This may indicate an error in forming the matrix A. There is no function specifically designed to invert a symmetric matrix stored in RFP format which is not positive definite; the matrix must be treated as a full symmetric matrix, by calling nag_lapack_dsytri (f07mj).

7 Accuracy

The computed inverse X satisfies

$$\|XA - I\|_2 \leq c(n)\epsilon\kappa_2(A) \quad \text{and} \quad \|AX - I\|_2 \leq c(n)\epsilon\kappa_2(A),$$

where $c(n)$ is a modest function of n, ϵ is the ***machine precision*** and $\kappa_2(A)$ is the condition number of A defined by

$$\kappa_2(A) = \|A\|_2 \left\|A^{-1}\right\|_2.$$

8 Further Comments

The total number of floating-point operations is approximately $\frac{2}{3}n^3$.

The complex analogue of this function is nag_lapack_zpftri (f07ww).

9 Example

This example computes the inverse of the matrix A, where

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

Here A is symmetric positive definite, stored in RFP format, and must first be factorized by nag_lapack_dpftrf (f07wd).

9.1 Program Text

```
     function f07wj_example

fprintf('f07wj example results\n\n');

% Symmetric matrix in RFP format
transr = 'n';
uplo   = 'l';
ar = [ 0.76   0.34;
       4.16   1.18;
      -3.12   5.03;
       0.56  -0.83;
      -0.10   1.18];
n  = nag_int(4);
n2 = (n*(n+1))/2;
ar  = reshape(ar,[n2,1]);

% Factorize a
[ar, info] = f07wd(transr, uplo, n, ar);

if info == 0
  % Compute inverse of a
  [ar, info] = f07wj(transr, uplo, n, ar);
  % Convert inverse to full array form, and print it
  [a, info] = f01vg(transr, uplo, n, ar);
  fprintf('\n');
  [ifail] = x04ca(uplo, 'n', a, 'Inverse');
else
  fprintf('\na is not positive definite.\n');
end
```

9.2 Program Results

```
     f07wj example results

 Inverse
             1          2          3          4
 1      0.6995
 2      0.7769     1.4239
 3      0.7508     1.8255     4.0688
 4     -0.9340    -1.8841    -2.9342     3.4978
```