NAG Library Routine Document

F01ABF

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

F01ABF calculates the accurate inverse of a real symmetric positive definite matrix, using a Cholesky factorization and iterative refinement.

2 Specification

```
SUBROUTINE F01ABF (A, LDA, N, B, LDB, Z, IFAIL)
INTEGER            LDA, N, LDB, IFAIL
REAL (KIND=nag_wp) A(LDA,N), B(LDB,N), Z(N)
```

3 Description

To compute the inverse X of a real symmetric positive definite matrix A, F01ABF first computes a Cholesky factorization of A as $A = LL^{\mathrm{T}}$, where L is lower triangular. An approximation to X is found by computing L^{-1} and then the product $L^{-\mathrm{T}}L^{-1}$. The residual matrix $R = I - AX$ is calculated using ***additional precision***, and a correction D to X is found by solving $LL^{\mathrm{T}}D = R$. X is replaced by $X + D$, and this iterative refinement of the inverse is repeated until full machine accuracy has been obtained.

4 References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation II, Linear Algebra* Springer–Verlag

5 Arguments

1: A(LDA, N) – REAL (KIND=nag_wp) array *Input/Output*

On entry: the upper triangle of the n by n positive definite symmetric matrix A. The elements of the array below the diagonal need not be set.

On exit: the lower triangle of the inverse matrix X is stored in the elements of the array below the diagonal, in rows 2 to $n + 1$; x_{ij} is stored in A$(i + 1, j)$ for $i \geq j$. The upper triangle of the original matrix is unchanged.

2: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F01ABF is called.

Constraint: LDA $\geq$ N + 1.

3: N – INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: N $\geq$ 1.

4: B(LDB, N) – REAL (KIND=nag_wp) array *Output*

On exit: the lower triangle of the inverse matrix X, with x_{ij} stored in B(i, j), for $i \geq j$.

5: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F01ABF is called.

Constraint: LDB $\geq$ N.

6: Z(N) – REAL (KIND=nag_wp) array *Workspace*

7: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL $= 1$

The matrix A is not positive definite, possibly due to rounding errors.

IFAIL $= 2$

The refinement process fails to converge, i.e., the matrix A is ill-conditioned.

IFAIL $= 3$

N < 1, or LDA $<$ N $+ 1$, or LDB $<$ N.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The computed inverse should be correct to full machine accuracy. For a detailed error analysis see page 40 of Wilkinson and Reinsch (1971).

8 Parallelism and Performance

F01ABF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F01ABF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The time taken by F01ABF is approximately proportional to n^3.

10 Example

This example finds the inverse of the 4 by 4 matrix:

$$\begin{pmatrix} 5 & 7 & 6 & 5 \\ 7 & 10 & 8 & 7 \\ 6 & 8 & 10 & 9 \\ 5 & 7 & 9 & 10 \end{pmatrix}.$$

10.1 Program Text

```
    Program f01abfe

!     F01ABF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f01abf, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, lda, ldb, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), b(:,:), z(:)
!     .. Executable Statements ..
      Write (nout,*) 'F01ABF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      lda = n + 1
      ldb = n
      Allocate (a(lda,n),b(ldb,n),z(n))
      Read (nin,*)(a(i,1:n),i=1,n)

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f01abf(a,lda,n,b,ldb,z,ifail)

!     Print the result matrix B
      Call x04caf('L','N',ldb,n,b,ldb,'Lower triangle of inverse',ifail)

    End Program f01abfe
```

10.2 Program Data

```
F01ABF Example Program Data
   4                    : n
   5.   7.   6.   5.
   7.  10.   8.   7.
   6.   8.  10.   9.
   5.   7.   9.  10.    : a
```

10.3 Program Results

```
 F01ABF Example Program Results

 Lower triangle of inverse
             1          2          3          4
 1     68.0000
 2    -41.0000    25.0000
 3    -17.0000    10.0000     5.0000
 4     10.0000    -6.0000    -3.0000     2.0000
```