nag_real_cholesky_solve_mult_rhs (f04agc)

1. Purpose

nag_real_cholesky_solve_mult_rhs (f04agc) calculates the approximate solution of a set of real symmetric positive-definite linear equations with multiple right-hand sides, $AX = B$, where A has been factorized by nag_real_cholesky (f03aec).

2. Specification

```
#include <nag.h>
#include <nagf04.h>

void nag_real_cholesky_solve_mult_rhs(Integer n, Integer nrhs, double a[],
     Integer tda, double p[], double b[], Integer tdb, double x[], Integer tdx,
     NagError *fail)
```

3. Description

To solve a set of real linear equations $AX = B$ where A is symmetric positive-definite, the function must be preceded by a call to nag_real_cholesky (f03aec) which computes a Cholesky factorization of A as $A = LL^T$, where L is lower triangular. The columns x of the solution X are found by forward and backward substitution in $Ly = b$ and $L^T x = y$, where b is a column of the right-hand sides.

4. Parameters

n

Input: n, the order of the matrix A.
Constraint: **n** ≥ 1.

nrhs

Input: r, the number of right-hand sides.
Constraint: **nrhs** ≥ 1.

a[n][tda]

Input: the upper triangle of the n by n positive-definite symmetric matrix A, and the sub-diagonal elements of its Cholesky factor L, as returned by nag_real_cholesky (f03aec).

tda

Input: the second dimension of the array **a** as declared in the function from which nag_real_cholesky_solve_mult_rhs is called.
Constraint: **tda** $\geq$ **n**.

p[n]

Input: the reciprocals of the diagonal elements of L, as returned by nag_real_cholesky (f03aec).

b[n][tdb]

Input: the n by r right-hand side matrix B. See also Section 6.

tdb

Input: the second dimension of the array **b** as declared in the function from which nag_real_cholesky_solve_mult_rhs is called.
Constraint: **tdb** $\geq$ **nrhs**.

x[n][tdx]

Output: the n by r solution matrix X. See also Section 6.

tdx

Input: the second dimension of the array **x** as declared in the function from which nag_real_cholesky_solve_mult_rhs is called.
Constraint: **tdx** $\geq$ **nrhs**.

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

NE_INT_ARG_LT

On entry, **n** must not be less than 1: **n** = ⟨*value*⟩.
On entry, **nrhs** must not be less than 1: **nrhs** = ⟨*value*⟩.

NE_2_INT_ARG_LT

On entry, **tda** = ⟨*value*⟩ while **n** = ⟨*value*⟩. These parameters must satisfy **tda** ≥ **n**.
On entry, **tdb** = ⟨*value*⟩while **nrhs** = ⟨*value*⟩. These parameters must satisfy **tdb** ≥ **nrhs**.
On entry, **tdx** = ⟨*value*⟩while **nrhs** = ⟨*value*⟩. These parameters must satisfy **tdx** ≥ **nrhs**.

6. Further Comments

The time taken is approximately proportional to n^2r.

The function may be called with the same actual array supplied for parameters **b** and **x**, in which case the solution vectors will overwrite the right-hand sides.

6.1. Accuracy

The accuracy of the computed solutions depends on the conditioning of the original matrix. For a detailed error analysis see Wilkinson and Reinsch (1971) p 39.

6.2. References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation (Vol II, Linear Algebra)* Springer-Verlag pp 31–44.

7. See Also

nag_real_cholesky (f03aec)

8. Example

To solve the set of linear equations $AX = B$ where

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

and

$$B = \begin{pmatrix} 23 \\ 32 \\ 33 \\ 31 \end{pmatrix}.$$

8.1. Program Text

```
/* nag_real_cholesky_solve_mult_rhs(f04agc) Example Program
 *
 * Copyright 1996 Numerical Algorithms Group.
 *
 * Mark 4, 1996.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagf03.h>
#include <nagf04.h>

#define NMAX 8
#define NRHS 1
#define TDA NMAX
```

```
#define TDB NRHS
#define TDX NRHS

main()
{
  double d1;
  Integer i, id, j, n;
  double a[NMAX][TDA], b[NMAX][TDB], p[NMAX], x[NMAX][TDX];
  static NagError fail;

  Vprintf("f04agc Example Program Results\n");
  /* Skip heading in data file */
  Vscanf("%*[^\n]");
  Vscanf("%ld", &n);
  if (n<1 || n>NMAX)
    {
      Vfprintf(stderr,"\nn is out of range: n = %ld\n", n);
      exit(EXIT_FAILURE);
    }
  for (i=0; i<n; ++i)
    for (j=0; j<n; ++j)
      Vscanf("%lf", &a[i][j]);
  for (i=0; i<n; ++i)
    for (j=0; j<NRHS; ++j)
      Vscanf("%lf", &b[i][j]);
  fail.print = TRUE;

  /* Cholesky decomposition */
  f03aec(n, (double *)a, (Integer)TDA, p, &d1, &id, &fail);
  if (fail.code != NE_NOERROR)
    exit(EXIT_FAILURE);

  /* Approximate solution of linear equations */
  f04agc(n, (Integer)NRHS, (double *)a, (Integer)TDA, p, (double *)b,
         (Integer)TDB, (double *)x, (Integer)TDX, &fail);
  if (fail.code != NE_NOERROR)
    exit(EXIT_FAILURE);
  Vprintf("\n Solution\n");
  for (i=0; i<n; ++i)
    {
      for (j=0; j<NRHS; ++j)
        Vprintf("%9.4f", x[i][j]);
      Vprintf("\n");
    }
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

```
f04agc Example Program Data
  4
    5    7    6    5
    7   10    8    7
    6    8   10    9
    5    7    9   10
   23   32   33   31
```

8.3. Program Results

```
f04agc Example Program Results

 Solution
   1.0000
   1.0000
   1.0000
   1.0000
```