NAG Library Function Document

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[], const 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, nag_real_cholesky_solve_mult_rhs (f04agc) must be preceded by a call to nag_real_cholesky (f03aec) which computes a Cholesky factorization of A as $A = LL^{\mathrm{T}}$, where L is lower triangular. The columns x of the solution X are found by forward and backward substitution in $Ly = b$ and $L^{\mathrm{T}}x = y$, where b is a column of the right-hand sides.

4 References

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

5 Arguments

1: **n** – Integer *Input*

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

Constraint: $\mathbf{n} \geq 1$.

2: **nrhs** – Integer *Input*

On entry: r, the number of right-hand sides.

Constraint: $\mathbf{nrhs} \geq 1$.

3: $\mathbf{a}[\mathbf{n} \times \mathbf{tda}]$ – double *Input*

Note: the (i, j)th element of the matrix A is stored in $\mathbf{a}[(i-1) \times \mathbf{tda} + j - 1]$.

On entry: 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).

4: **tda** – Integer *Input*

On entry: the stride separating matrix column elements in the array **a**.

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

5: **p**[**n**] – double *Input*

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

6: **b**[**n** × **tdb**] – const double *Input*

Note: the (i,j)th element of the matrix B is stored in **b**$[(i-1) \times$ **tdb** $+ j - 1]$.

On entry: the n by r right-hand side matrix B. See also Section 9.

7: **tdb** – Integer *Input*

On entry: the stride separating matrix column elements in the array **b**.

Constraint: **tdb** $\geq$ **nrhs**.

8: **x**[**n** × **tdx**] – double *Output*

Note: the (i,j)th element of the matrix X is stored in **x**$[(i-1) \times$ **tdx** $+ j - 1]$.

On exit: the n by r solution matrix X. See also Section 9.

9: **tdx** – Integer *Input*

On entry: the stride separating matrix column elements in the array **x**.

Constraint: **tdx** $\geq$ **nrhs**.

10: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_2_INT_ARG_LT

On entry, **tda** $= \langle$*value*$\rangle$ while **n** $= \langle$*value*$\rangle$. These arguments must satisfy **tda** $\geq$ **n**.

On entry, **tdb** $= \langle$*value*$\rangle$ while **nrhs** $= \langle$*value*$\rangle$. These arguments must satisfy **tdb** $\geq$ **nrhs**.

On entry, **tdx** $= \langle$*value*$\rangle$ while **nrhs** $= \langle$*value*$\rangle$. These arguments must satisfy **tdx** $\geq$ **nrhs**.

NE_INT_ARG_LT

On entry, **n** $= \langle$*value*$\rangle$.
Constraint: **n** ≥ 1.

On entry, **nrhs** $= \langle$*value*$\rangle$.
Constraint: **nrhs** ≥ 1.

7 Accuracy

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

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by nag_real_cholesky_solve_mult_rhs (f04agc) is approximately proportional to n^2r.

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

10 Example

This example solves the set of linear equations $AX = B$ where

$$A = \begin{pmatrix} 5 & 7 & 6 & 5 \\ 7 & 10 & 8 & 7 \\ 6 & 8 & 10 & 9 \\ 5 & 7 & 9 & 10 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 23 \\ 32 \\ 33 \\ 31 \end{pmatrix}.$$

10.1 Program Text

```
/* nag_real_cholesky_solve_mult_rhs (f04agc) Example Program.
 *
 * Copyright 1996 Numerical Algorithms Group.
 *
 * Mark 4, 1996.
 * Mark 8 revised, 2004.
 */

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

#define A(I, J) a[(I) *tda + J]
#define B(I, J) b[(I) *tdb + J]
#define X(I, J) x[(I) *tdx + J]

int main(void)
{
  Integer  exit_status = 0, i, id, j, n, nrhs, tda, tdb, tdx;
  NagError fail;
  double   *a = 0, *b = 0, d1, *p = 0, *x = 0;

  INIT_FAIL(fail);

  printf("nag_real_cholesky_solve_mult_rhs (f04agc) Example Program"
          " Results\n");
  /* Skip heading in data file */
  scanf("%*[^\n]");
  scanf("%ld", &n);
  nrhs = 1;
  if (n >= 1)
    {
      if (!(a = NAG_ALLOC(n*n, double)) ||
          !(b = NAG_ALLOC(n*nrhs, double)) ||
          !(p = NAG_ALLOC(n, double)) ||
          !(x = NAG_ALLOC(n*nrhs, double)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
      tda = n;
      tdb = nrhs;
      tdx = nrhs;
    }
  else
    {
      printf("Invalid n.\n");
      exit_status = 1;
      return exit_status;
    }
  for (i = 0; i < n; ++i)
```

```
    for (j = 0; j < n; ++j)
      scanf("%lf", &A(i, j));
  for (i = 0; i < n; ++i)
    for (j = 0; j < nrhs; ++j)
      scanf("%lf", &B(i, j));

  /* Cholesky decomposition */
  /* nag_real_cholesky (f03aec).
   * LL^T factorization and determinant of real symmetric
   * positive-definite matrix
   */
  nag_real_cholesky(n, a, tda, p, &d1, &id, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_real_cholesky (f03aec).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

  /* Approximate solution of linear equations */
  /* nag_real_cholesky_solve_mult_rhs (f04agc).
   * Approximate solution of real symmetric positive-definite
   * simultaneous linear equations (coefficient matrix already
   * factorized by nag_real_cholesky (f03aec))
   */
  nag_real_cholesky_solve_mult_rhs(n, nrhs, a, tda, p, b, tdb, x, tdx, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf(
              "Error from nag_real_cholesky_solve_mult_rhs (f04agc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }
  printf("\n Solution\n");
  for (i = 0; i < n; ++i)
    {
      for (j = 0; j < nrhs; ++j)
        printf("%9.4f", X(i, j));
      printf("\n");
    }
 END:
  NAG_FREE(a);
  NAG_FREE(b);
  NAG_FREE(p);
  NAG_FREE(x);
  return exit_status;
}
```

10.2 Program Data

```
nag_real_cholesky_solve_mult_rhs (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
```

10.3 Program Results

```
nag_real_cholesky_solve_mult_rhs (f04agc) Example Program Results

 Solution
   1.0000
   1.0000
   1.0000
```

```
   1.0000
```