nag_real_lu_solve_mult_rhs (f04ajc)

1.   Purpose

nag_real_lu_solve_mult_rhs (f04ajc) calculates the approximate solution of a set of real linear equations with multiple right-hand sides, $AX = B$, where A has been factorized by nag_real_lu (f03afc).

2.   Specification

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

void nag_real_lu_solve_mult_rhs(Integer n, Integer nrhs, double a[],
      Integer tda, Integer pivot[], double b[], Integer tdb, NagError *fail)
```

3.   Description

To solve a set of real linear equations $AX = B$, this function must be preceded by a call to nag_real_lu (f03afc) which computes an LU factorization of A with partial pivoting, $PA = LU$, where P is a permutation matrix, L is lower triangular and U is unit upper triangular. The columns x of the solution X are found by forward and backward substitution in $Ly = Pb$ and $Ux = y$, where b is a column of the right-hand sides.

4.   Parameters

n
>   Input: n, the order of the matrix A.
>   Constraint: $\mathbf{n} \geq 1$.

nrhs
>   Input: r, the number of right-hand sides.
>   Constraint: $\mathbf{nrhs} \geq 1$.

a[n][tda]
>   Input: details of the LU factorization, as returned by nag_real_lu (f03afc).

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

pivot[n]
>   Input: details of the row interchanges as returned by nag_real_lu (f03afc).

b[n][tdb]
>   Input: the n by r right-hand side matrix B.
>   Output: B is overwritten by the solution matrix X.

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

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: $\mathbf{n} = \langle value \rangle$.
>   On entry, **nrhs** must not be less than 1: $\mathbf{nrhs} = \langle value \rangle$.

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**.

6.   Further Comments

The time taken by the function is approximately proportional to $n^2 r$.

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 106.

6.2.   References

Wilkinson J H and Reinsch C (1971) *Handbook for Automatic Computation (Vol II, Linear Algebra)* Springer-Verlag pp 93–110.

7.   See Also

nag_real_lu (f03afc)
nag_real_lin_eqn (f04arc)

8.   Example

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

$$A = \begin{pmatrix} 33 & 16 & 72 \\ -24 & -10 & -57 \\ -8 & -4 & -17 \end{pmatrix}$$

and

$$B = \begin{pmatrix} -359 \\ 281 \\ 85 \end{pmatrix}.$$

8.1.   Program Text

```
/* nag_real_lu_solve_mult_rhs(f04ajc) Example Program
 *
 * Copyright 1990 Numerical Algorithms Group.
 *
 * Mark 1, 1990.
 */

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


#define TDB 3
#define NMAX 8
#define TDA NMAX

main()
{
  double detf;
  Integer i, dete, j, n, nrhs = 1;
  double a[NMAX][TDA], b[NMAX][TDB];
  Integer pivot[NMAX];
  static NagError fail;
```

```
          Vprintf("f04ajc Example Program Results\n");
          /* Skip heading in data file */
          Vscanf("%*[^\n]");
          Vscanf("%ld",&n);
          if (n>0 && n<=NMAX)
            {
              for (i=0; i<n; i++)
                for (j=0; j<n; j++)
                  Vscanf("%lf",&a[i][j]);
              /* Crout decomposition  */
              fail.print = TRUE;
              f03afc(n,(double *)a,(Integer)TDA,pivot,&detf,&dete,&fail);
              if (fail.code!=NE_NOERROR)
                exit(EXIT_FAILURE);
              else
                {
                  for (i=0; i<n; i++)
                    for (j=0; j<nrhs; j++)
                      Vscanf("%lf",&b[i][j]);
                  /*
                   * Approximate solution of linear equations
                   */
                  f04ajc(n,nrhs,(double *)a,(Integer)TDA,pivot,(double *)b,
                         (Integer)TDB,&fail);
                  if (fail.code!=NE_NOERROR)
                    exit(EXIT_FAILURE);
                  Vprintf("Solution\n");
                  for (i=0; i<n; i++)
                    for (j=0; j<nrhs; j++)
                      Vprintf("%9.4f\n",b[i][j]);
                }
            }
          else
            {
              Vfprintf(stderr, "N is out of range: N = %5ld\n", n);
              exit(EXIT_FAILURE);
            }
          exit(EXIT_SUCCESS);
        }
```

8.2. Program Data

```
f04ajc Example Program Data
  3
   33   16   72
  -24  -10  -57
   -8   -4  -17
 -359  281   85
```

8.3. Program Results

```
f04ajc Example Program Results
Solution
   1.0000
  -2.0000
  -5.0000
```