NAG Toolbox

nag_lapack_dsgesv (f07ac)

1 Purpose

nag_lapack_dsgesv (f07ac) computes the solution to a real system of linear equations

$$AX = B,$$

where A is an n by n matrix and X and B are n by r matrices.

2 Syntax

```
[a, ipiv, x, iter, info] = nag_lapack_dsgesv(a, b, 'n', n, 'nrhs_p', nrhs_p)
[a, ipiv, x, iter, info] = f07ac(a, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

nag_lapack_dsgesv (f07ac) first attempts to factorize the matrix in single precision and use this factorization within an iterative refinement procedure to produce a solution with full double precision accuracy. If the approach fails the method switches to a double precision factorization and solve.

The iterative refinement process is stopped if

$$\textbf{iter} > itermax,$$

where **iter** is the number of iterations carried out thus far and $itermax$ is the maximum number of iterations allowed, which is fixed at 30 iterations. The process is also stopped if for all right-hand sides we have

$$\|resid\| < \sqrt{\mathbf{n}}\|x\|\|A\|\epsilon,$$

where $\|resid\|$ is the ∞-norm of the residual, $\|x\|$ is the ∞-norm of the solution, $\|A\|$ is the ∞-operator-norm of the matrix A and ϵ is the ***machine precision*** returned by nag_machine_precision (x02aj).

The iterative refinement strategy used by nag_lapack_dsgesv (f07ac) can be more efficient than the corresponding direct full precision algorithm. Since this strategy must perform iterative refinement on each right-hand side, any efficiency gains will reduce as the number of right-hand sides increases. Conversely, as the matrix size increases the cost of these iterative refinements become less significant relative to the cost of factorization. Thus, any efficiency gains will be greatest for a very small number of right-hand sides and for large matrix sizes. The cut-off values for the number of right-hand sides and matrix size, for which the iterative refinement strategy performs better, depends on the relative performance of the reduced and full precision factorization and back-substitution. For now, nag_lapack_dsgesv (f07ac) always attempts the iterative refinement strategy first; you are advised to compare the performance of nag_lapack_dsgesv (f07ac) with that of its full precision counterpart nag_lapack_dgesv (f07aa) to determine whether this strategy is worthwhile for your particular problem dimensions.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

Buttari A, Dongarra J, Langou J, Langou J, Luszczek P and Kurzak J (2007) Mixed precision iterative refinement techniques for the solution of dense linear systems *International Journal of High Performance Computing Applications*

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

5.1 Compulsory Input Parameters

1: **a**$(lda,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **a** must be at least $\max(1,\mathbf{n})$.

The second dimension of the array **a** must be at least $\max(1,\mathbf{n})$.

The n by n coefficient matrix A.

2: **b**$(ldb,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **b** must be at least $\max(1,\mathbf{n})$.

The second dimension of the array **b** must be at least $\max(1,\mathbf{nrhs\_p})$.

The n by r right-hand side matrix B.

5.2 Optional Input Parameters

1: **n** – INTEGER

Default: the first dimension of the array **b** and the second dimension of the array **a**. (An error is raised if these dimensions are not equal.)

n, the number of linear equations, i.e., the order of the matrix A.

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

2: **nrhs_p** – INTEGER

Default: the second dimension of the array **b**.

r, the number of right-hand sides, i.e., the number of columns of the matrix B.

Constraint: $\mathbf{nrhs\_p} \geq 0$.

5.3 Output Parameters

1: **a**$(lda,:)$ – REAL (KIND=nag_wp) array

The first dimension of the array **a** will be $\max(1,\mathbf{n})$.

The second dimension of the array **a** will be $\max(1,\mathbf{n})$.

If iterative refinement has been successfully used (i.e., if $\mathbf{info} = 0$ and $\mathbf{iter} \geq 0$), then A is unchanged. If double precision factorization has been used (when $\mathbf{info} = 0$ and $\mathbf{iter} < 0$), A contains the factors L and U from the factorization $A = PLU$; the unit diagonal elements of L are not stored.

2: **ipiv**(**n**) – INTEGER array

If no constraints are violated, the pivot indices that define the permutation matrix P; at the ith step row i of the matrix was interchanged with row **ipiv**(i). **ipiv**$(i) = i$ indicates a row interchange was not required. **ipiv** corresponds either to the single precision factorization (if **info** $= 0$ and **iter** ≥ 0) or to the double precision factorization (if **info** $= 0$ and **iter** < 0).

3: **x**$(ldx, :)$ – REAL (KIND=nag_wp) array

The first dimension of the array **x** will be $\max(1, \mathbf{n})$.

The second dimension of the array **x** will be $\max(1, \mathbf{nrhs\_p})$.

If **info** $= 0$, the n by r solution matrix X.

4: **iter** – INTEGER

If **iter** > 0, iterative refinement has been successfully used and **iter** is the number of iterations carried out.

If **iter** < 0, iterative refinement has failed for one of the reasons given below and double precision factorization has been carried out instead.

iter $= -1$
Taking into account machine parameters, and the values of **n** and **nrhs_p**, it is not worth working in single precision.

iter $= -2$
Overflow of an entry occurred when moving from double to single precision.

iter $= -3$
An intermediate single precision factorization failed.

iter $= -31$
The maximum permitted number of iterations was exceeded.

5: **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*)

Element $\langle value \rangle$ of the diagonal is exactly zero. The factorization has been completed, but the factor U is exactly singular, so the solution could not be computed.

7 Accuracy

The computed solution for a single right-hand side, $\hat{x}$, satisfies the equation of the form

$$(A + E)\hat{x} = b,$$

where

$$\|E\|_1 = O(\epsilon)\|A\|_1$$

and ϵ is the ***machine precision***. An approximate error bound for the computed solution is given by

$$\frac{\|\hat{x}-x\|_1}{\|x\|_1} \leq \kappa(A)\frac{\|E\|_1}{\|A\|_1}$$

where $\kappa(A) = \left\|A^{-1}\right\|_1 \|A\|_1$, the condition number of A with respect to the solution of the linear equations. See Section 4.4 of Anderson *et al.* (1999) for further details.

8 Further Comments

The complex analogue of this function is nag_lapack_zcgesv (f07aq).

9 Example

This example solves the equations

$$Ax = b,$$

where A is the general matrix

$$A = \begin{pmatrix} 1.80 & 2.88 & 2.05 & -0.89 \\ 5.25 & -2.95 & -0.95 & -3.80 \\ 1.58 & -2.69 & -2.90 & -1.04 \\ -1.11 & -0.66 & -0.59 & 0.80 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 9.52 \\ 24.35 \\ 0.77 \\ -6.22 \end{pmatrix}.$$

9.1 Program Text

```
function f07ac_example

fprintf('f07ac example results\n\n');

a = [ 1.80,  2.88,  2.05, -0.89;
      5.25, -2.95, -0.95, -3.80;
      1.58, -2.69, -2.90, -1.04;
     -1.11, -0.66, -0.59,  0.80];
b = [ 9.52;
     24.35;
      0.77;
     -6.22];

[a, ipiv, x, iter, info] = f07ac( ...
                                  a, b);

disp('Solution');
disp(x');
disp('Pivot indices');
disp(double(ipiv'));
```

9.2 Program Results

```
f07ac example results

Solution
    1.0000   -1.0000    3.0000   -5.0000

Pivot indices
     2     2     3     4
```