NAG Toolbox for MATLAB

f07jh

1 Purpose

f07jh computes error bounds and refines the solution to a real system of linear equations $AX = B$, where A is an n by n symmetric positive-definite tridiagonal matrix and X and B are n by r matrices, using the modified Cholesky factorization returned by f07jd and an initial solution returned by f07je. Iterative refinement is used to reduce the backward error as much as possible.

2 Syntax

```
[x, ferr, berr, info] = f07jh(d, e, df, ef, b, x, 'n', n, 'nrhs_p',
nrhs_p)
```

3 Description

f07jh should normally be preceded by calls to f07jd and f07je. f07jd computes a modified Cholesky factorization of the matrix A as

$$A = LDL^{\mathrm{T}},$$

where L is a unit lower bidiagonal matrix and D is a diagonal matrix, with positive diagonal elements. f07je then utilizes the factorization to compute a solution, $\hat{X}$, to the required equations. Letting $\hat{x}$ denote a column of $\hat{X}$, f07jh computes a *component-wise backward error*, β, the smallest relative perturbation in each element of A and b such that $\hat{x}$ is the exact solution of a perturbed system

$$(A + E)\hat{x} = b + f, \quad \text{with} \quad |e_{ij}| \leq \beta|a_{ij}|, \quad \text{and} \quad |f_j| \leq \beta|b_j|.$$

The function also estimates a bound for the *component-wise forward error* in the computed solution defined by $\max|x_i - \hat{x}_i|/\max|\hat{x}_i|$, where x is the corresponding column of the exact solution, X.

Note that the modified Cholesky factorization of A can also be expressed as

$$A = U^{\mathrm{T}}DU,$$

where U is unit upper bidiagonal.

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 URL: http://www.netlib.org/lapack/lug

5 Parameters

5.1 Compulsory Input Parameters

1: **d**($*$) – **double array**

Note: the dimension of the array **d** must be at least $\max(1, \mathbf{n})$.

Must contain the n diagonal elements of the matrix of A.

2: **e**($*$) – **double array**

Note: the dimension of the array **e** must be at least $\max(1, \mathbf{n} - 1)$.

Must contain the $(n - 1)$ subdiagonal elements of the matrix A.

3: **df**$(*)$ **– double array**

Note: the dimension of the array **df** must be at least $\max(1, \mathbf{n})$.

Must contain the n diagonal elements of the diagonal matrix D from the LDL^{T} factorization of A.

4: **ef**$(*)$ **– double array**

Note: the dimension of the array **ef** must be at least $\max(1, \mathbf{n})$.

Must contain the $(n-1)$ subdiagonal elements of the unit bidiagonal matrix L from the LDL^{T} factorization of A.

5: **b**(**ldb**,$*$) **– double array**

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

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

The n by r matrix of right-hand sides B.

6: **x**(**ldx**,$*$) **– double array**

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

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

The n by r initial solution matrix X.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the array **d** The dimension of the array **df** The dimension of the array **ef**.

n, the order of the matrix A.

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

2: **nrhs_p – int32 scalar**

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

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 Input Parameters Omitted from the MATLAB Interface

ldb, ldx, work

5.4 Output Parameters

1: **x**(**ldx**,$*$) **– double array**

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

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

The n by r refined solution matrix X.

2: **ferr**$(*)$ **– double array**

Note: the dimension of the array **ferr** must be at least $\max(1, \mathbf{nrhs\_p})$.

Estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \mathbf{ferr}(j)$, where $\hat{x}_j$ is the jth column of the computed solution returned in the array **x** and x_j is the corresponding column of the exact solution X. The estimate is almost always a slight overestimate of the true error.

3: **berr**($*$) – **double array**

Note: the dimension of the array **berr** must be at least $\max(1, \textbf{nrhs\_p})$.

Estimate of the component-wise relative backward error of each computed solution vector $\hat{x}_j$ (i.e., the smallest relative change in any element of A or B that makes $\hat{x}_j$ an exact solution).

4: **info – int32 scalar**

info $= 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

info $= -i$

If **info** $= -i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **n**, 2: **nrhs_p**, 3: **d**, 4: **e**, 5: **df**, 6: **ef**, 7: **b**, 8: **ldb**, 9: **x**, 10: **ldx**, 11: **ferr**, 12: **berr**, 13: **work**, 14: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

7 Accuracy

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

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

where

$$\|E\|_\infty = O(\epsilon)\|A\|_\infty$$

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

$$\frac{\|\hat{x} - x\|_\infty}{\|x\|_\infty} \leq \kappa(A)\frac{\|E\|_\infty}{\|A\|_\infty},$$

where $\kappa(A) = \left\|A^{-1}\right\|_\infty \|A\|_\infty$, 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. Function f07jg can be used to compute the condition number of A.

8 Further Comments

The total number of floating-point operations required to solve the equations $AX = B$ is proportional to nr. At most five steps of iterative refinement are performed, but usually only one or two steps are required.

The complex analogue of this function is f07jv.

9 Example

```
d = [4;
     10;
     29;
     25;
     5];
```

```
e = [-2;
     -6;
     15;
     8];
df = [4;
     9;
     25;
     16;
     1];
ef = [-0.5;
     -0.6666666666666666;
     0.6;
     0.5;
     0];
b = [6, 10;
     9, 4;
     2, 9;
     14, 65;
     7, 23];
x = [2.5, 2;
     2, -0.9999999999999999;
     1, -3;
     -1, 6;
     3, -5];
[xOut, ferr, berr, info] = f07jh(d, e, df, ef, b, x)
```

```
xOut =
    2.5000    2.0000
    2.0000   -1.0000
    1.0000   -3.0000
   -1.0000    6.0000
    3.0000   -5.0000
ferr =
   1.0e-13 *
    0.2425
    0.4663
berr =
   1.0e-16 *
         0
    0.1110
info =
           0
```