NAG Toolbox

nag_linsys_real_gen_sparse_lsqsol (f04qa)

1 Purpose

nag_linsys_real_gen_sparse_lsqsol (f04qa) solves sparse nonsymmetric equations, sparse linear least squares problems and sparse damped linear least squares problems, using a Lanczos algorithm.

2 Syntax

```
[b, x, se, itnlim, itn, anorm, acond, rnorm, arnorm, xnorm, user, inform,
ifail] = nag_linsys_real_gen_sparse_lsqsol(n, b, aprod, damp, atol, btol,
conlim, itnlim, msglvl, 'm', m, 'user', user)

[b, x, se, itnlim, itn, anorm, acond, rnorm, arnorm, xnorm, user, inform,
ifail] = f04qa(n, b, aprod, damp, atol, btol, conlim, itnlim, msglvl, 'm', m,
'user', user)
```

3 Description

nag_linsys_real_gen_sparse_lsqsol (f04qa) can be used to solve a system of linear equations

$$Ax = b \tag{1}$$

where A is an n by n sparse nonsymmetric matrix, or can be used to solve linear least squares problems, so that nag_linsys_real_gen_sparse_lsqsol (f04qa) minimizes the value ρ given by

$$\rho = \|r\|, \quad r = b - Ax \tag{2}$$

where A is an m by n sparse matrix and $\|r\|$ denotes the Euclidean length of r so that $\|r\|^2 = r^{\mathrm{T}}r$. A damping argument, λ, may be included in the least squares problem in which case nag_linsys_real_gen_sparse_lsqsol (f04qa) minimizes the value ρ given by

$$\rho^2 = \|r\|^2 + \lambda^2\|x\|^2. \tag{3}$$

λ is supplied as the argument **damp** and should of course be zero if the solution to problems (1) or (2) is required. Minimizing ρ in (3) is often called ridge regression.

nag_linsys_real_gen_sparse_lsqsol (f04qa) is based upon algorithm LSQR (see Paige and Saunders (1982a) and Paige and Saunders (1982b)) and solves the problems by an algorithm based upon the Lanczos process. The function does not require A explicitly, but A is specified via **aprod** which must perform the operations $(y + Ax)$ and $(x + A^{\mathrm{T}}y)$ for a given n-element vector x and m element vector y. A argument to **aprod** specifies which of the two operations is required on a given entry.

The function also returns estimates of the standard errors of the sample regression coefficients (x_i, for $i = 1, 2, \ldots, n$) given by the diagonal elements of the estimated variance-covariance matrix V. When problem (2) is being solved and A is of full rank, then V is given by

$$V = s^2\left(A^{\mathrm{T}}A\right)^{-1}, \quad s^2 = \rho^2/(m-n), \quad m > n$$

and when problem (3) is being solved then V is given by

$$V = s^2\left(A^{\mathrm{T}}A + \lambda^2 I\right)^{-1}, \quad s^2 = \rho^2/m, \quad \lambda \neq 0.$$

Let $\bar{A}$ denote the matrix

$$\bar{A} = A, \quad \lambda = 0; \quad \bar{A} = \begin{pmatrix} A \\ \lambda I \end{pmatrix}, \quad \lambda \neq 0, \tag{4}$$

let $\bar{r}$ denote the residual vector

$$\bar{r} = r, \quad \lambda = 0; \quad \bar{r} = \begin{pmatrix} b \\ 0 \end{pmatrix} - \bar{A}x, \quad \lambda \neq 0 \tag{5}$$

corresponding to an iterate x, so that $\rho = \|\bar{r}\|$ is the function being minimized, and let $\|A\|$ denote the Frobenius (Euclidean) norm of A. Then the function accepts x as a solution if it is estimated that one of the following two conditions is satisfied:

$$\rho \leq tol_1 \|\bar{A}\| . \|x\| + tol_2 \|b\| \tag{6}$$

$$\|\bar{A}^{\mathrm{T}} \bar{r}\| \leq tol_1 \|\bar{A}\| \rho \tag{7}$$

where tol_1 and tol_2 are user-supplied tolerances which estimate the relative errors in A and b respectively. Condition (6) is appropriate for compatible problems where, in theory, we expect the residual to be zero and will be satisfied by an acceptable solution x to a compatible problem. Condition (7) is appropriate for incompatible systems where we do not expect the residual to be zero and is based on the observation that, in theory,

$$\bar{A}^{\mathrm{T}} \bar{r} = 0$$

when x is a solution to the least squares problem, and so (7) will be satisfied by an acceptable solution x to a linear least squares problem.

The function also includes a test to prevent convergence to solutions, x, with unacceptably large elements. This can happen if A is nearly singular or is nearly rank deficient. If we let the singular values of $\bar{A}$ be

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$$

then the condition number of $\bar{A}$ is defined as

$$\mathrm{cond}(\bar{A}) = \sigma_1 / \sigma_k$$

where σ_k is the smallest nonzero singular value of $\bar{A}$ and hence k is the rank of $\bar{A}$. When $k < n$, then $\bar{A}$ is rank deficient, the least squares solution is not unique and nag_linsys_real_gen_sparse_lsqsol (f04qa) will normally converge to the minimal length solution. In practice $\bar{A}$ will not have exactly zero singular values, but may instead have small singular values that we wish to regard as zero.

The function provides for this possibility by terminating if

$$\mathrm{cond}(\bar{A}) \geq c_{\mathrm{lim}} \tag{8}$$

where c_{lim} is a user-supplied limit on the condition number of $\bar{A}$. For problem (1) termination with this condition indicates that A is nearly singular and for problem (2) indicates that A is nearly rank deficient and so has near linear dependencies in its columns. In this case inspection of $\|r\|$, $\|A^{\mathrm{T}} r\|$ and $\|x\|$, which are all returned by the function, will indicate whether or not an acceptable solution has been found. Condition (8), perhaps in conjunction with $\lambda \neq 0$, can be used to try and 'regularize' least squares solutions. A full discussion of the stopping criteria is given in Section 6 of Paige and Saunders (1982a).

Introduction of a nonzero damping argument λ tends to reduce the size of the computed solution and to make its components less sensitive to changes in the data, and nag_linsys_real_gen_sparse_lsqsol (f04qa) is applicable when a value of λ is known *a priori*. To have an effect, λ should normally be at least $\sqrt{\epsilon}\|A\|$ where ϵ is the ***machine precision***. For further discussion see Paige and Saunders (1982b) and the references given there.

Whenever possible the matrix A should be scaled so that the relative errors in the elements of A are all of comparable size. Such a scaling helps to prevent the least squares problem from being unnecessarily sensitive to data errors and will normally reduce the number of iterations required. At the very least, in the absence of better information, the columns of A should be scaled to have roughly equal column length.

4 References

Paige C C and Saunders M A (1982a) LSQR: An algorithm for sparse linear equations and sparse least squares *ACM Trans. Math. Software* **8** 43–71

Paige C C and Saunders M A (1982b) Algorithm 583 LSQR: Sparse linear equations and least squares problems *ACM Trans. Math. Software* **8** 195–209

5 Parameters

5.1 Compulsory Input Parameters

1: **n** – INTEGER

n, the number of columns of the matrix A.

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

2: **b**(**m**) – REAL (KIND=nag_wp) array

The right-hand side vector b.

3: **aprod** – SUBROUTINE, supplied by the user.

aprod must perform the operations $y := y + Ax$ and $x := x + A^{\mathrm{T}}y$ for given vectors x and y.

```
[mode, x, y, user] = aprod(mode, m, n, x, y, user, lruser, liuser)
```

Input Parameters

1: **mode** – INTEGER

Specifies which operation is to be performed.

mode $= 1$
aprod must compute $y + Ax$.

mode $= 2$
aprod must compute $x + A^{\mathrm{T}}y$.

2: **m** – INTEGER

m, the number of rows of A.

3: **n** – INTEGER

n, the number of columns of A.

4: **x**(**n**) – REAL (KIND=nag_wp) array

The vector x.

5: **y**(**m**) – REAL (KIND=nag_wp) array

The vector y.

6: **user** – REAL (KIND=nag_wp) array
7: **lruser** – INTEGER
8: **liuser** – INTEGER

aprod is called from nag_linsys_real_gen_sparse_lsqsol (f04qa) with the object supplied to nag_linsys_real_gen_sparse_lsqsol (f04qa).

Output Parameters

1: **mode** – INTEGER

May be used as a flag to indicate a failure in the computation of $y + Ax$ or $x + A^{\mathrm{T}}y$. If **mode** is negative on exit from **aprod**, nag_linsys_real_gen_sparse_lsqsol (f04qa) will exit immediately with **ifail** set to **mode**.

2: **x**(**n**) – REAL (KIND=nag_wp) array

If **mode** $= 1$, **x** must be unchanged.

If **mode** $= 2$, **x** must contain $x + A^{\mathrm{T}}y$.

3: **y**(**m**) – REAL (KIND=nag_wp) array

If **mode** $= 1$, **y** must contain $y + Ax$.

If **mode** $= 2$, **y** must be unchanged.

4: **user** – REAL (KIND=nag_wp) array

4: **damp** – REAL (KIND=nag_wp)

The value λ. If either problem (1) or problem (2) is to be solved, then **damp** must be supplied as zero.

5: **atol** – REAL (KIND=nag_wp)

The tolerance, tol_1, of the convergence criteria (6) and (7); it should be an estimate of the largest relative error in the elements of A. For example, if the elements of A are correct to about 4 significant figures, then **atol** should be set to about 5×10^{-4}. If **atol** is supplied as less than ϵ, where ϵ is the ***machine precision***, then the value ϵ is used instead of **atol**.

6: **btol** – REAL (KIND=nag_wp)

The tolerance, tol_2, of the convergence criterion (6); it should be an estimate of the largest relative error in the elements of B. For example, if the elements of B are correct to about 4 significant figures, then **btol** should be set to about 5×10^{-4}. If **btol** is supplied as less than ϵ, then the value ϵ is used instead of **btol**.

7: **conlim** – REAL (KIND=nag_wp)

The value c_{lim} of equation (8); it should be an upper limit on the condition number of $\bar{A}$. **conlim** should not normally be chosen much larger than $1.0/$**atol**. If **conlim** is supplied as zero, then the value $1.0/\epsilon$ is used instead of **conlim**.

8: **itnlim** – INTEGER

An upper limit on the number of iterations. If **itnlim** ≤ 0, then the value **n** is used in place of **itnlim**, but for ill-conditioned problems a higher value of **itnlim** is likely to be necessary.

9: **msglvl** – INTEGER

The level of printing from nag_linsys_real_gen_sparse_lsqsol (f04qa). If **msglvl** ≤ 0, then no printing occurs, but otherwise messages will be output on the advisory message channel (see nag_file_set_unit_advisory (x04ab)). A description of the printed output is given in Section 9.1. The level of printing is determined as follows:

msglvl ≤ 0
No printing.

msglvl $= 1$
A brief summary is printed just prior to return from nag_linsys_real_gen_sparse_lsqsol (f04qa).

msglvl ≥ 2
A summary line is printed periodically to monitor the progress of nag_linsys_real_gen_sparse_lsqsol (f04qa), together with a brief summary just prior to return from nag_linsys_real_gen_sparse_lsqsol (f04qa).

5.2 Optional Input Parameters

1: **m** – INTEGER

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

m, the number of rows of the matrix A.

Constraint: **m** ≥ 1.

2: **user** – REAL (KIND=nag_wp) array

user is not used by nag_linsys_real_gen_sparse_lsqsol (f04qa), but is passed to **aprod**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

5.3 Output Parameters

1: **b**(**m**) – REAL (KIND=nag_wp) array

2: **x**(**n**) – REAL (KIND=nag_wp) array

The solution vector x.

3: **se**(**n**) – REAL (KIND=nag_wp) array

The estimates of the standard errors of the components of x. Thus **se**(i) contains an estimate of $\sqrt{\nu_{ii}}$, where ν_{ii} is the ith diagonal element of the estimated variance-covariance matrix V. The estimates returned in **se** will be the lower bounds on the actual estimated standard errors, but will usually be correct to at least one significant figure.

4: **itnlim** – INTEGER

Unchanged unless **itnlim** ≤ 0 on entry, in which case it is set to **n**.

5: **itn** – INTEGER

The number of iterations performed.

6: **anorm** – REAL (KIND=nag_wp)

An estimate of $\left\|\bar{A}\right\|$ for the matrix $\bar{A}$ of (4).

7: **acond** – REAL (KIND=nag_wp)

An estimate of cond$\left(\bar{A}\right)$ which is a lower bound.

8: **rnorm** – REAL (KIND=nag_wp)

An estimate of $\|\bar{r}\|$ for the residual, $\bar{r}$, of (5) corresponding to the solution x returned in **x**. Note that $\|\bar{r}\|$ is the function being minimized.

9: **arnorm** – REAL (KIND=nag_wp)

An estimate of the $\left\|\bar{A}^{\mathrm{T}}\bar{r}\right\|$ corresponding to the solution x returned in **x**.

10: **xnorm** – REAL (KIND=nag_wp)

An estimate of $\|x\|$ for the solution x returned in **x**.

11: **user** – REAL (KIND=nag_wp) array

12: **inform** – INTEGER

The reason for termination of nag_linsys_real_gen_sparse_lsqsol (f04qa).

inform $= 0$
The exact solution is $x = 0$. No iterations are performed in this case.

inform $= 1$
The termination criterion of (6) has been satisfied with tol_1 and tol_2 as the values supplied in **atol** and **btol** respectively.

inform $= 2$
The termination criterion of (7) has been satisfied with tol_1 as the value supplied in **atol**.

inform $= 3$
The termination criterion of (6) has been satisfied with tol_1 and/or tol_2 as the value ϵ, where ϵ is the ***machine precision***. One or both of the values supplied in **atol** and **btol** must have been less than ϵ and was too small for this machine.

inform $= 4$
The termination criterion of (7) has been satisfied with tol_1 as the value ϵ. The value supplied in **atol** must have been less than ϵ and was too small for this machine.

The values **inform** $= 5$, 6 and 7 correspond to failure with **ifail** $= 2$, 3 and 4 respectively (see Section 6) and when **ifail** is negative **inform** will be set to the same negative value.

13: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail < 0 (*warning*)

A negative value of **ifail** indicates an exit from nag_linsys_real_gen_sparse_lsqsol (f04qa) because you have set **mode** negative in **aprod**. The value of **ifail** will be the same as your setting of **mode**.

ifail $= 1$

On entry, **m** < 1,
or **n** < 1,
or $lruser < 1$,
or $liuser < 1$.

ifail $= 2$

The condition of (8) has been satisfied for the value of c_{lim} supplied in **conlim**. If this failure is unexpected you should check that **aprod** is working correctly. Although conditions (6) or (7) have not been satisfied, the values returned in **rnorm**, **arnorm** and **xnorm** may nevertheless indicate that an acceptable solution has been reached.

ifail $= 3$

The condition of (8) has been satisfied for the value $c_{\text{lim}} = 1.0/\epsilon$, where ϵ is the ***machine precision***. The matrix $\bar{A}$ is nearly singular or rank deficient and the problem is too ill-conditioned for this machine. If this failure is unexpected, you should check that **aprod** is working correctly.

ifail $= 4$

The limit on the number of iterations has been reached. The number of iterations required by nag_linsys_real_gen_sparse_lsqsol (f04qa) and the condition of the matrix $\bar{A}$ can depend strongly on the scaling of the problem. Poor scaling of the rows and columns of A should be avoided whenever possible.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

When the problem is compatible, the computed solution x will satisfy the equation

$$r = b - Ax,$$

where an estimate of $\|r\|$ is returned in the argument **rnorm**. When the problem is incompatible, the computed solution x will satisfy the equation

$$\bar{A}^{\mathrm{T}}\bar{r} = e,$$

where an estimate of $\|e\|$ is returned in the argument **arnorm**. See also Section 6.2 of Paige and Saunders (1982b).

8 Further Comments

The time taken by nag_linsys_real_gen_sparse_lsqsol (f04qa) is likely to be principally determined by the time taken in **aprod**, which is called twice on each iteration, once with **mode** $= 1$ and once with **mode** $= 2$. The time taken per iteration by the remaining operations in nag_linsys_real_gen_sparse_lsqsol (f04qa) is approximately proportional to $\max(m, n)$.

The Lanczos process will usually converge more quickly if A is pre-conditioned by a nonsingular matrix M that approximates A in some sense and is also chosen so that equations of the form $My = c$ can efficiently be solved for y. For a discussion of pre-conditioning, see the F11 Chapter Introduction. In the context of nag_linsys_real_gen_sparse_lsqsol (f04qa), problem (1) is equivalent to

$$\left(AM^{-1}\right)y = b, \quad Mx = y$$

and problem (2) is equivalent to minimizing

$$\rho = \|r\|, \quad r = b - \left(AM^{-1}\right)y, \quad Mx = y.$$

Note that the normal matrix $(AM^{-1})^{\mathrm{T}}(AM^{-1}) = M^{-\mathrm{T}}(A^{\mathrm{T}}A)M^{-1}$ so that the pre-conditioning AM^{-1} is equivalent to the pre-conditioning $M^{-\mathrm{T}}(A^{\mathrm{T}}A)M^{-1}$ of the normal matrix $A^{\mathrm{T}}A$.

Pre-conditioning can be incorporated into nag_linsys_real_gen_sparse_lsqsol (f04qa) simply by coding **aprod** to compute $y + AM^{-1}x$ and $x + M^{-\mathrm{T}}A^{\mathrm{T}}y$ in place of $y + Ax$ and $x + A^{\mathrm{T}}y$ respectively, and then solving the equations $Mx = y$ for x on return from nag_linsys_real_gen_sparse_lsqsol (f04qa). The

quantity $y + AM^{-1}x$ should be computed by solving $Mz = x$ for z and then computing $y + Az$, and $x + M^{-\mathrm{T}}A^{\mathrm{T}}y$ should be computed by solving $M^{\mathrm{T}}z = A^{\mathrm{T}}y$ for z and then forming $x + z$.

8.1 Description of the Printed Output

When **msglvl** > 0, then nag_linsys_real_gen_sparse_lsqsol (f04qa) will produce output (except in the case where the function fails with **ifail** $= 1$) on the advisory message channel (see nag_file_set_unit_advisory (x04ab)).

When **msglvl** ≥ 2 then a summary line is printed periodically giving the following information:

Output	**Meaning**
`ITN`	Iteration number, k.
`X(1)`	The first element of the current iterate x_k.
`FUNCTION`	The current value of the function, ρ, being minimized.
`COMPAT`	An estimate of $\|\bar{r}_k\|/\|b\|$, where $\bar{r}_k$ is the residual corresponding to x_k. This value should converge to zero (in theory) if and only if the problem is compatible. `COMPAT` decreases monotonically.
`INCOMPAT`	An estimate of $\|\bar{A}^{\mathrm{T}}\bar{r}_k\|/(\|\bar{A}\|\|\bar{r}_k\|)$ which should converge to zero if and only if at the solution ρ is nonzero. `INCOMPAT` is not usually monotonic.
`NRM(ABAR)`	A monotonically increasing estimate of $\|\bar{A}\|$.
`COND(ABAR)`	A monotonically increasing estimate of the condition number $\mathrm{cond}(\bar{A})$.

9 Example

This example solves the linear least squares problem

$$\min \rho = \|r\|, \quad r = b - Ax$$

where A is the 13 by 12 matrix and b is the 13 element vector given by

$$A = \begin{pmatrix} 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}, \quad b = -h^2 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ -h^{-3} \end{pmatrix}$$

with $h = 0.1$.

Such a problem can arise by considering the Neumann problem on a rectangle

$$\frac{\delta u}{\delta n} = 0$$

$$\frac{\delta u}{\delta n} = 0 \quad \boxed{\nabla^2 u = g(x, y)} \quad \frac{\delta u}{\delta n} = 0 \quad \int_c u = 1$$

$$\frac{\delta u}{\delta n} = 0$$

where C is the boundary of the rectangle, and discretizing as illustrated below with the square mesh

The 12 by 12 symmetric part of A represents the difference equations and the final row comes from the normalizing condition. The example program has $g(x, y) = 1$ at all the internal mesh points, but apart from this is written in a general manner so that the number of rows (NROWS) and columns (NCOLS) in the grid can readily be altered.

9.1 Program Text

```
     function f04qa_example

fprintf('f04qa example results\n\n');

m = nag_int(13);
n = nag_int(12);
a = [  1,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0;
       0,  1,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0;
       0,  0,  1, -1,  0,  0,  0,  0,  0,  0,  0,  0;
      -1,  0, -1,  4, -1,  0,  0, -1,  0,  0,  0,  0;
       0, -1,  0, -1,  4, -1,  0,  0, -1,  0,  0,  0;
       0,  0,  0,  0, -1,  1,  0,  0,  0,  0,  0,  0;
       0,  0,  0,  0,  0,  0,  1, -1,  0,  0,  0,  0;
       0,  0,  0, -1,  0,  0, -1,  4, -1,  0, -1,  0;
       0,  0,  0,  0, -1,  0,  0, -1,  4, -1,  0, -1;
       0,  0,  0,  0,  0,  0,  0,  0, -1,  1,  0,  0;
       0,  0,  0,  0,  0,  0,  0, -1,  0,  0,  1,  0;
       0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,  1;
       1,  1,  1,  0,  0,  1,  1,  0,  0,  1,  1,  1];
b = zeros(m,1);
b(4:5) = -0.01;
b(8:9) = -0.01;
b(m)   = 10;

damp   = 0;
atol   = 1e-05;
btol   = 1e-04;
conlim = 10^6 - 10^-11;
itnlim = nag_int(100);
msglvl = nag_int(1);

[b, x, se, itnlim, itn, anorm, acond, rnorm, arnorm, xnorm, ...
 ruser, inform, ifail] = ...
f04qa( ...
       n, b, @aprod, damp, atol, btol, conlim, itnlim, msglvl, 'user', a);
fprintf('\n\nSolution is x:\n');
fprintf('%9.3f%9.3f%9.3f%9.3f%9.3f%9.3f\n',x);
fprintf('\n\nNorm of the residual = %12.2e\n', rnorm);

function [mode, x, y, user] = aprod(mode, m, n, x, y, user)

  % a is passed as user

  if (mode == 1)
    y = y + user*x;
  else
    x = x + transpose(user)*y;
  end
```

9.2 Program Results

```
     f04qa example results

 Output from sparse linear least squares solver.

 Least squares solution of  A*x = b

 The matrix A has     13 rows and     12 cols
 The damping parameter is  damp =      0.00E+00

 atol =     1.00E-05     conlim =      1.00E+06
 btol =     1.00E-04     itnlim =    100
```

```
 No. of iterations  =      2
 stopping condition =      2
 ( The least squares solution is good enough, given atol )

 Actual        norm(rbar), norm(x)            1.15E-02    4.33E+00
    Norm(transpose(Abar)*rbar)                6.98E-15
 Estimates of  norm(Abar), cond(Abar)         4.12E+00    2.45E+00

Solution is x:
    1.250    1.250    1.250    1.247    1.247    1.250
    1.250    1.247    1.247    1.250    1.250    1.250

Norm of the residual =     1.15e-02
```