NAG Toolbox for MATLAB

f11ga

1 Purpose

f11ga is a setup function, the first in a suite of three functions for the iterative solution of a symmetric system of simultaneous linear equations. f11ga must be called before f11gb, the iterative solver. The third function in the suite, f11gc, can be used to return additional information about the computation.

These three functions are suitable for the solution of large sparse symmetric systems of equations.

2 Syntax

```
[lwreq, ifail] = f11ga(method, precon, n, tol, maxitn, anorm, sigmax,
maxits, monit, 'sigcmp', sigcmp, 'norm_p', norm_p, 'weight', weight,
'iterm', iterm, 'sigtol', sigtol)
```

3 Description

The suite consisting of the functions f11ga, f11gb and f11gc is designed to solve the symmetric system of simultaneous linear equations $Ax = b$ of order n, where n is large and the matrix of the coefficients A is sparse.

f11ga is a setup function which must be called before f11gb, the iterative solver. The third function in the suite, f11gc can be used to return additional information about the computation. Either of two methods can be used.

(i) **Conjugate Gradient Method**

For this method (see Hestenes and Stiefel 1952, Golub and Van Loan 1996, Barrett *et al.* 1994 and Dias da Cunha and Hopkins 1994), the matrix A should ideally be positive-definite. The application of the conjugate gradient method to indefinite matrices may lead to failure or to lack of convergence.

(ii) **Lanczos Method (SYMMLQ)**

This method, based upon the algorithm SYMMLQ (see Paige and Saunders 1975 and Barrett *et al.* 1994), is suitable for both positive-definite and indefinite matrices. It is more robust than the conjugate gradient method but less efficient when A is positive-definite.

Both conjugate gradient and Lanczos (SYMMLQ) methods start from the residual $r_0 = b - Ax_0$, where x_0 is an initial estimate for the solution (often $x_0 = 0$), and generate an orthogonal basis for the Krylov subspace $\mathrm{span}\{A^k r_0\}$, for $k = 0, 1, \ldots$, by means of three-term recurrence relations (see Golub and Van Loan 1996). A sequence of symmetric tridiagonal matrices $\{T_k\}$ is also generated. Here and in the following, the index k denotes the iteration count. The resulting symmetric tridiagonal systems of equations are usually more easily solved than the original problem. A sequence of solution iterates $\{x_k\}$ is thus generated such that the sequence of the norms of the residuals $\{\|r_k\|\}$ converges to a required tolerance. Note that, in general, the convergence is not monotonic.

In exact arithmetic, after n iterations, this process is equivalent to an orthogonal reduction of A to symmetric tridiagonal form, $T_n = Q^{\mathrm{T}}AQ$; the solution x_n would thus achieve exact convergence. In finite-precision arithmetic, cancellation and round-off errors accumulate causing loss of orthogonality. These methods must therefore be viewed as genuinely iterative methods, able to converge to a solution **within a prescribed tolerance**.

The orthogonal basis is not formed explicitly in either method. The basic difference between the two methods lies in the method of solution of the resulting symmetric tridiagonal systems of equations: the conjugate gradient method is equivalent to carrying out an LDL^{T} (Cholesky) factorization whereas the Lanczos method (SYMMLQ) uses an LQ factorization.

Faster convergence can be achieved using a **preconditioner** (see Golub and Van Loan 1996 and Barrett *et al.* 1994). A preconditioner maps the original system of equations onto a different system, say

$$\bar{A}\bar{x} = \bar{b}, \tag{1}$$

with, hopefully, better characteristics with respect to its speed of convergence: for example, the condition number of the matrix of the coefficients can be improved or eigenvalues in its spectrum can be made to coalesce. An orthogonal basis for the Krylov subspace $\mathrm{span}\{\bar{A}^k\bar{r}_0\}$, for $k = 0, 1, \ldots$, is generated and the solution proceeds as outlined above. The algorithms used are such that the solution and residual iterates of the original system are produced, not their preconditioned counterparts. Note that an unsuitable preconditioner or no preconditioning at all may result in a very slow rate, or lack, of convergence. However, preconditioning involves a trade-off between the reduction in the number of iterations required for convergence and the additional computational costs per iteration. Also, setting up a preconditioner may involve non-negligible overheads.

A preconditioner must be **symmetric and positive-definite**, i.e., representable by $M = EE^{\mathrm{T}}$, where M is nonsingular, and such that $\bar{A} = E^{-1}AE^{-\mathrm{T}} \sim I_n$ in (1), where I_n is the identity matrix of order n. Also, we can define $\bar{r} = E^{-1}r$ and $\bar{x} = E^{\mathrm{T}}x$. These are formal definitions, used only in the design of the algorithms; in practice, only the means to compute the matrix-vector products $v = Au$ and to solve the preconditioning equations $Mv = u$ are required; that is, explicit information about M, E or their inverses is not required at any stage.

The first termination criterion

$$\|r_k\|_p \leq \tau\left(\|b\|_p + \|A\|_p,\, \|x_k\|_p\right) \tag{2}$$

is available for both conjugate gradient and Lanczos (SYMMLQ) methods. In (2), $p = 1, \infty$ or 2 and τ denotes a user-specified tolerance subject to $\max(10, \sqrt{n})\epsilon \leq \tau < 1$, where ϵ is the ***machine precision***. Facilities are provided for the estimation of the norm of the matrix of the coefficients $\|A\|_1 = \|A\|_\infty$, when this is not known in advance, used in (2), by applying Higham's method (see Higham 1988). Note that $\|A\|_2$ cannot be estimated internally. This criterion uses an error bound derived from **backward** error analysis to ensure that the computed solution is the exact solution of a problem as close to the original as the termination tolerance requires. Termination criteria employing bounds derived from **forward** error analysis could be used, but any such criteria would require information about the condition number $\kappa(A)$ which is not easily obtainable.

The second termination criterion

$$\|\bar{r}_k\|_2 \leq \tau \max\left(1.0, \|b\|_2/\|r_0\|_2\right)\left(\|\bar{r}_0\|_2 + \sigma_1(\bar{A}), \|\Delta\bar{x}_k\|_2\right) \tag{3}$$

is available only for the Lanczos method (SYMMLQ). In (3), $\sigma_1(\bar{A}) = \|\bar{A}\|_2$ is the largest singular value of the (preconditioned) iteration matrix $\bar{A}$. This termination criterion monitors the progress of the solution of the preconditioned system of equations and is less expensive to apply than criterion (2). When $\sigma_1(\bar{A})$ is not supplied, facilities are provided for its estimation by $\sigma_1(\bar{A}) \sim \max_k \sigma_1(T_k)$. The interlacing property $\sigma_1(T_{k-1}) \leq \sigma_1(T_k)$ and Gerschgorin's theorem provide lower and upper bounds from which $\sigma_1(T_k)$ can be easily computed by bisection. Alternatively, the less expensive estimate $\sigma_1(\bar{A}) \sim \max_k \|T_k\|_1$ can be used, where $\sigma_1(\bar{A}) \leq \|T_k\|_1$ by Gerschgorin's theorem. Note that only order of magnitude estimates are required by the termination criterion.

Termination criterion (2) is the recommended choice, despite its (small) additional costs per iteration when using the Lanczos method (SYMMLQ). Also, if the norm of the initial estimate is much larger than the norm of the solution, that is, if $\|x_0\| \gg \|x\|$, a dramatic loss of significant digits could result in complete lack of convergence. The use of criterion (2) will enable the detection of such a situation, and the iteration will be restarted at a suitable point. No such restart facilities are provided for criterion (3).

Optionally, a vector w of user-specified weights can be used in the computation of the vector norms in termination criterion (2), i.e., $\|v\|_p^{(w)} = \left\|v^{(w)}\right\|_p$, where $\left(v^{(w)}\right)_i = w_i v_i$, for $i = 1, 2, \ldots, n$. Note that the use of weights increases the computational costs.

The sequence of calls to the functions comprising the suite is enforced: first, the setup function f11ga must be called, followed by the solver f11gb. f11gc can be called either when f11gb is carrying out a monitoring step or after f11gb has completed its tasks. Incorrect sequencing will raise an error condition.

4 References

Barrett R, Berry M, Chan T F, Demmel J, Donato J, Dongarra J, Eijkhout V, Pozo R, Romine C and Van der Vorst H 1994 *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* SIAM, Philadelphia

Dias da Cunha R and Hopkins T 1994 PIM 1.1 — the parallel iterative method package for systems of linear equations user's guide — Fortran 77 version *Technical Report* Computing Laboratory, University of Kent at Canterbury, Kent, UK

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

Hestenes M and Stiefel E 1952 Methods of conjugate gradients for solving linear systems *J. Res. Nat. Bur. Stand.* **49** 409–436

Higham N J 1988 FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation *ACM Trans. Math. Software* **14** 381–396

Paige C C and Saunders M A 1975 Solution of sparse indefinite systems of linear equations *SIAM J. Numer. Anal.* **12** 617–629

5 Parameters

5.1 Compulsory Input Parameters

1: **method – string**

The iterative method to be used.

method = 'CG'

Conjugate gradient method.

method = 'SYMMLQ'

Lanczos method (SYMMLQ).

Constraint: **method** = 'CG' or 'SYMMLQ'.

2: **precon – string**

Determines whether preconditioning is used.

precon = 'N'

No preconditioning.

precon = 'P'

Preconditioning.

Constraint: **precon** = 'N' or 'P'.

3: **n – int32 scalar**

n, the order of the matrix A.

Constraint: **n** > 0.

4: **tol – double scalar**

The tolerance τ for the termination criterion.

If **tol** ≤ 0.0, $\tau = \max\left(\sqrt{\epsilon}, \sqrt{n}\epsilon\right)$ is used, where ϵ is the ***machine precision***.

Otherwise $\tau = \max\left(\mathbf{tol}, 10\epsilon, \sqrt{n}\epsilon\right)$ is used.

Constraint: **tol** < 1.0.

5: **maxitn – int32 scalar**

The maximum number of iterations.

Constraint: **maxitn** > 0.

6: **anorm – double scalar**

If **anorm** > 0.0, the value of $\|A\|_p$ to be used in the termination criterion (2) (**iterm** $= 1$).

If **anorm** ≤ 0.0, **iterm** $= 1$ and **norm_p** = '1' or 'I', then $\|A\|_1 = \|A\|_\infty$ is estimated internally by f11gb.

If **iterm** $= 2$, then **anorm** is not referenced.

Constraint: if **iterm** $= 1$ and **norm_p** $= 2$, **anorm** > 0.0.

7: **sigmax – double scalar**

If **sigmax** > 0.0, the value of $\sigma_1(\bar{A}) = \|E^{-1}AE^{-\mathrm{T}}\|_2$.

If **sigmax** ≤ 0.0, $\sigma_1(\bar{A})$ is estimated by f11gb when either **sigcmp** = 'S' or termination criterion (3) (**iterm** $= 2$) is employed, though it will be used only in the latter case.

Otherwise, **sigmax** is not referenced.

8: **maxits – int32 scalar**

The maximum iteration number $k =$ **maxits** for which $\sigma_1(T_k)$ is computed by bisection (see also Section 3). If **sigcmp** = 'N' or **sigmax** > 0.0, then **maxits** is not referenced.

Suggested value: **maxits** $= \min(10, n)$ when **sigtol** is of the order of its default value (0.01).

Constraint: if **sigcmp** = 'S' and **sigmax** ≤ 0.0, $1 \leq$ **maxits** $\leq$ **maxitn**.

9: **monit – int32 scalar**

If **monit** > 0, the frequency at which a monitoring step is executed by f11gb: the current solution and residual iterates will be returned by f11gb and a call to f11gc made possible every **monit** iterations, starting from the MONIT(th). Otherwise, no monitoring takes place. There are some additional computational costs involved in monitoring the solution and residual vectors when the Lanczos method (SYMMLQ) is used.

Constraint: **monit** $\leq$ **maxitn**.

5.2 Optional Input Parameters

1: **sigcmp – string**

Determines whether an estimate of $\sigma_1(\bar{A}) = \|E^{-1}AE^{-\mathrm{T}}\|_2$, the largest singular value of the preconditioned matrix of the coefficients, is to be computed using the bisection method on the sequence of tridiagonal matrices $\{T_k\}$ generated during the iteration. Note that $\bar{A} = A$ when a preconditioner is not used.

If **sigmax** > 0.0 (see below), i.e., when $\sigma_1(\bar{A})$ is supplied, the value of **sigcmp** is ignored.

sigcmp = 'S'

$\sigma_1(\bar{A})$ is to be computed using the bisection method.

sigcmp = 'N'

The bisection method is not used.

If the termination criterion (3) is used, requiring $\sigma_1(\bar{A})$, an inexpensive estimate is computed and used (see Section 3).

Suggested value: **sigcmp** = 'N'.

Default: 'N'

Constraint: **sigcmp** = 'S' or 'N'.

2: **norm_p – string**

Defines the matrix and vector norm to be used in the termination criteria.

norm_p = '1'

Use the l_1 norm.

precon = 'I'

Use the l_∞ norm.

precon = '2'

Use the l_2 norm.

Suggested value:

if **iterm** = 1, **norm_p** = 'I';;
if **iterm** = 2, **norm_p** = '2' (see below)..

Default:

if **iterm** = 1, 'I';
'2' otherwise.

Constraints:

if **iterm** = 1, **norm_p** = '1', 'I' or '2';
if **iterm** = 2, **norm_p** = '2'.

3: **weight – string**

Specifies whether a vector w of user-supplied weights is to be used in the vector norms used in the computation of termination criterion (2) (**iterm** = 1): $\|v\|_p^{(w)} = \left\|v^{(w)}\right\|_p$, where $v_i^{(w)} = w_i v_i$, for $i = 1, 2, \ldots, n$. The suffix $p = 1, 2, \infty$ denotes the vector norm used, as specified by the parameter **norm_p**. Note that weights cannot be used when **iterm** = 2, i.e., when criterion (3) is used.

weight = 'W'

User-supplied weights are to be used and must be supplied on initial entry to f11gb.

weight = 'N'

All weights are implicitly set equal to one. Weights do not need to be supplied on initial entry to f11gb.

Suggested value: **weight** = 'N'.

Default: 'N'

Constraints:

if **iterm** = 1, **weight** = 'W' or 'N';
if **iterm** = 2, **weight** = 'N'.

4: **iterm – int32 scalar**

Defines the termination criterion to be used.

iterm $= 1$

Use the termination criterion defined in (2) (both conjugate gradient and Lanczos (SYMMLQ) methods).

iterm $= 2$

Use the termination criterion defined in (3) (Lanczos method (SYMMLQ) only).

Suggested value: **iterm** $= 1$.

Default: 1

Constraints:

if **method** = 'CG', **iterm** $= 1$;
if **method** = 'SYMMLQ', **iterm** $= 1$.

5: **sigtol – double scalar**

The tolerance used in assessing the convergence of the estimate of $\sigma_1(\bar{A}) = \|\bar{A}\|_2$ when the bisection method is used.

If **sigtol** ≤ 0.0, the default value **sigtol** $= 0.01$ is used. The actual value used is max(**sigtol**, ϵ).

If **sigcmp** = 'N' or **sigmax** > 0.0, then **sigtol** is not referenced.

Suggested value: **sigtol** $= 0.01$ should be sufficient in most cases.

Default: 0.01

Constraint: if **sigcmp** = 'S' and **sigmax** ≤ 0.0, **sigtol** < 1.0.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **lwreq – int32 scalar**

The minimum amount of workspace required by f11gb. (See also Section 5 of the document for f11gb.)

2: **ifail – int32 scalar**

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= -i$

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

1: **method**, 2: **precon**, 3: **sigcmp**, 4: **norm_p**, 5: **weight**, 6: **iterm**, 7: **n**, 8: **tol**, 9: **maxitn**, 10: **anorm**, 11: **sigmax**, 12: **sigtol**, 13: **maxits**, 14: **monit**, 15: **lwreq**, 16: **ifail**.

ifail $= 1$

f11ga has been called out of sequence.

7 Accuracy

Not applicable.

8 Further Comments

When $\sigma_1(\bar{A})$ is not supplied (**sigmax** ≤ 0.0) but it is required, it is estimated by f11gb using either of the two methods described in Section 3, as specified by the parameter **sigcmp**. In particular, if **sigcmp** = 'S', then the computation of $\sigma_1(\bar{A})$ is deemed to have converged when the differences between three successive values of $\sigma_1(T_k)$ differ, in a relative sense, by less than the tolerance **sigtol**, i.e., when

$$\max\left(\frac{\left|\sigma_1^{(k)}-\sigma_1^{(k-1)}\right|}{\sigma_1^{(k)}},\frac{\left|\sigma_1^{(k)}-\sigma_1^{(k-2)}\right|}{\sigma_1^{(k)}}\right)\leq \textbf{sigtol}.$$

The computation of $\sigma_1(\bar{A})$ is also terminated when the iteration count exceeds the maximum value allowed, i.e., $k \geq$ **maxits**.

Bisection is increasingly expensive with increasing iteration count. A reasonably large value of **sigtol**, of the order of the suggested value, is recommended and an excessive value of **maxits** should be avoided. Under these conditions, $\sigma_1(\bar{A})$ usually converges within very few iterations.

9 Example

```
method = ’CG’;
precon = ’P’;
n = int32(7);
tol = 1e-06;
maxitn = int32(20);
anorm = 0;
sigmax = 0;
maxits = int32(7);
monit = int32(2);
lfill = int32(0);
dtol = 0;
mic = ’N’;
dscale = 0;
pstrat = ’M’;
ipiv = zeros(n, 1, ’int32’);
nnz = int32(16);
a = zeros(100, 1);
a(1:nnz) = [4, 1, 5, 2, 2, 3, -1, 1, 4, 1, -2, 3, 2, -1, -2, 5];
irow = zeros(100, 1, ’int32’);
irow(1:nnz) = [int32(1), int32(2), int32(2), int32(3), int32(4), ...
               int32(4), int32(5), int32(5), int32(5), int32(6), ...
                      int32(6),  int32(6),  int32(7),  int32(7),  int32(7),
int32(7)];
icol = zeros(100, 1, ’int32’);
icol(1:nnz) = [int32(1), int32(1), int32(2), int32(3), int32(2), ...
               int32(4), int32(1), int32(4), int32(5), int32(2), ...
                      int32(5),  int32(6),  int32(1),  int32(2),  int32(3),
int32(7)];
v = [15, 18, -8, 21, 11, 10, 29];
u = zeros(7, 1);
irevcm = int32(0);

% Calculate incomplete Cholesky factorisation
[a, irow, icol, ipiv, istr, nnzc, npivm, ifail] = ...
    f11ja(nnz, a, irow, icol, lfill, dtol, mic, dscale, ipiv);

% Initialise the solver
[lwreq, ifail] = ...
     f11ga(method, precon, n, tol, maxitn, anorm, sigmax, maxits, monit,
...
            ’norm_p’, ’1’, ’sigcmp’, ’s’);

work = zeros(lwreq, 1);

while (irevcm ~= 4)
```

```
  [irevcm, u, v, work, ifail] = f11gb(irevcm, u, v, work);

  if (irevcm == 1)
    % Calculate v = Au
    [v, ifail] = f11xe(a, irow, icol, ’No Checking’, u, ’nnz’, nnz);
    if (ifail ~= 0)
      irevcm = int32(6);
    end
  elseif (irevcm == 2)
    % Solve Mv = u
    [v, ifail] = f11jb(a, irow, icol, ipiv, istr, ’No Checking’, u);
    if (ifail ~= 0)
      irevcm = int32(6);
    end
  elseif (irevcm == 3)
    % Display monitoring information
    [itn, stplhs, stprhs, anorm, sigmax, its, sigerr, ifail] = f11gc();
    if (ifail == 0)
      fprintf(’\nMonitoring at iteration no. %d\n’, itn);
      fprintf(’residual norm: %14.4f\n’, stplhs);
      fprintf(’  Solution Vector  Residual Vector\n’);
      for i=1:n
        fprintf(’%16.4f %16.4f\n’, u(i), v(i));
      end
    end
  end
end

% Obtain information about the computation
[itn, stplhs, stprhs, anorm, sigmax, its, sigerr, ifail] = f11gc();

% Print the output data
fprintf(’\nFinal results\n’);
fprintf(’Number of iterations for convergence:        %d\n’, itn);
fprintf(’Residual norm:                           %14.4e\n’, stplhs);
fprintf(’Right-hand side of termination criteria: %14.4e\n’, stprhs);
fprintf(’1-norm of matrix A:                      %14.4e\n’, anorm);
fprintf(’Largest singular value of A_bar:         %14.4e\n’, sigmax);
fprintf(’  Solution Vector  Residual Vector\n’);
for i=1:n
  fprintf(’%16.4f %16.4f\n’, u(i), v(i));
end
```

```
Monitoring at iteration no. 2
residual norm:         1.9938
  Solution Vector  Residual Vector
          0.9632          -0.2296
          1.9934           0.2225
          3.0583           0.0958
          4.1453          -0.2515
          4.8289          -0.1716
          5.6630           0.6753
          7.1062          -0.3474

Monitoring at iteration no. 4
residual norm:         0.0067
  Solution Vector  Residual Vector
          0.9994          -0.0011
          2.0011          -0.0025
          3.0008          -0.0000
          3.9996           0.0000
          4.9991           0.0021
          5.9993          -0.0009
          7.0007           0.0001
Final results
Number of iterations for convergence:        5
Residual norm:                               2.0428e-14
Right-hand side of termination criteria:     3.9200e-04
1-norm of matrix A:                          1.0000e+01
Largest singular value of A_bar:             1.3596e+00
```

```
Solution Vector  Residual Vector
1.0000           0.0000
          2.0000          0.0000
          3.0000         -0.0000
          4.0000         -0.0000
          5.0000         -0.0000
          6.0000          0.0000
          7.0000          0.0000
```