NAG Toolbox for MATLAB

f11bd

1 Purpose

f11bd is a setup function, the first in a suite of three functions for the iterative solution of a real general (nonsymmetric) system of simultaneous linear equations. f11bd must be called before f11be, the iterative solver. The third function in the suite, f11bf, can be used to return additional information about the computation.

These three functions are suitable for the solution of large sparse general (nonsymmetric) systems of equations.

2 Syntax

```
[lwreq, work, ifail] = f11bd(method, precon, n, m, tol, maxitn, anorm, sigmax, monit, lwork, 'norm_p', norm_p, 'weight', weight, 'iterm', iterm)
```

3 Description

The suite consisting of the functions f11bd, f11be and f11bf is designed to solve the general (nonsymmetric) system of simultaneous linear equations $Ax = b$ of order n, where n is large and the coefficient matrix A is sparse.

f11bd is a setup function which must be called before f11be, the iterative solver. The third function in the suite, f11bf, can be used to return additional information about the computation. A choice of methods is available:

restarted generalized minimum residual method (RGMRES);

conjugate gradient squared method (CGS);

bi-conjugate gradient stabilized (ℓ) method (Bi-CGSTAB(ℓ));

transpose-free quasi-minimal residual method (TFQMR).

3.1 Restarted Generalized Minimum Residual Method (RGMRES)

The restarted generalized minimum residual method (RGMRES) (see Saad and Schultz 1986, Barrett *et al.* 1994 and Dias da Cunha and Hopkins 1994) starts from the residual $r_0 = b - Ax_0$, where x_0 is an initial estimate for the solution (often $x_0 = 0$). An orthogonal basis for the Krylov subspace span$\{A^k r_0\}$, for $k = 0, 1, 2, \ldots$, is generated explicitly: this is referred to as Arnoldi's method (see Arnoldi 1951). The solution is then expanded onto the orthogonal basis so as to minimize the residual norm $\|b - Ax\|_2$. The lack of symmetry of A implies that the orthogonal basis is generated by applying a 'long' recurrence relation, whose length increases linearly with the iteration count. For all but the most trivial problems, computational and storage costs can quickly become prohibitive as the iteration count increases. RGMRES limits these costs by employing a restart strategy: every m iterations at most, the Arnoldi process is restarted from $r_l = b - Ax_l$, where the subscript l denotes the last available iterate. Each group of m iterations is referred to as a 'super-iteration'. The value of m is chosen in advance and is fixed throughout the computation. Unfortunately, an optimum value of m cannot easily be predicted.

3.2 Conjugate Gradient Squared Method (CGS)

The conjugate gradient squared method (CGS) (see Sonneveld 1989, Barrett *et al.* 1994 and Dias da Cunha and Hopkins 1994) is a development of the bi-conjugate gradient method where the nonsymmetric Lanczos method is applied to reduce the coefficients matrix to real tridiagonal form: two bi-orthogonal sequences of vectors are generated starting from the residual $r_0 = b - Ax_0$, where x_0 is an initial estimate for the solution (often $x_0 = 0$) and from the *shadow residual* $\hat{r}_0$ corresponding to the arbitrary problem $A^{\mathrm{T}}\hat{x} = \hat{b}$,

where $\hat{b}$ can be any vector, but in practice is chosen so that $r_0 = \hat{r}_0$. In the course of the iteration, the residual and shadow residual $r_i = P_i(A)r_0$ and $\hat{r}_i = P_i\left(A^{\mathrm{T}}\right)\hat{r}_0$ are generated, where P_i is a polynomial of order i, and bi-orthogonality is exploited by computing the vector product $\rho_i = (\hat{r}_i, r_i) = \left(P_i\left(A^{\mathrm{T}}\right)\hat{r}_0, P_i(A)r_0\right) = \left(\hat{r}_0, P_i^2(A)r_0\right)$. Applying the 'contraction' operator $P_i(A)$ twice, the iteration coefficients can still be recovered without advancing the solution of the shadow problem, which is of no interest. The CGS method often provides fast convergence; however, there is no reason why the contraction operator should also reduce the once reduced vector $P_i(A)r_0$: this may well lead to a highly irregular convergence which may result in large cancellation errors.

3.3 Bi-Conjugate Gradient Stabilized (ℓ) Method (Bi-CGSTAB(ℓ))

The bi-conjugate gradient stabilized (ℓ) method (Bi-CGSTAB(ℓ)) (see Van der Vorst 1989, Sleijpen and Fokkema 1993 and Dias da Cunha and Hopkins 1994) is similar to the CGS method above. However, instead of generating the sequence $\left\{P_i^2(A)r_0\right\}$, it generates the sequence $\{Q_i(A)P_i(A)r_0\}$, where the $Q_i(A)$ are polynomials chosen to minimize the residual *after* the application of the contraction operator $P_i(A)$. Two main steps can be identified for each iteration: an OR (Orthogonal Residuals) step where a basis of order ℓ is generated by a Bi-CG iteration and an MR (Minimum Residuals) step where the residual is minimized over the basis generated, by a method akin to GMRES. For $\ell = 1$, the method corresponds to the Bi-CGSTAB method of Van der Vorst 1989. For $\ell > 1$, more information about complex eigenvalues of the iteration matrix can be taken into account, and this may lead to improved convergence and robustness. However, as ℓ increases, numerical instabilities may arise. For this reason, a maximum value of $\ell = 10$ is imposed, but probably $\ell = 4$ is sufficient in most cases.

3.4 Transpose-Free Quasi-Minimal Residual Method (TFQMR)

The transpose-free quasi-minimal residual method (TFQMR) (see Freund and Nachtigal 1991 and Freund 1993) is conceptually derived from the CGS method. The residual is minimized over the space of the residual vectors generated by the CGS iterations under the simplifying assumption that residuals are almost orthogonal. In practice, this is not the case but theoretical analysis has proved the validity of the method. This has the effect of remedying the rather irregular convergence behaviour with wild oscillations in the residual norm that can degrade the numerical performance and robustness of the CGS method. In general, the TFQMR method can be expected to converge at least as fast as the CGS method, in terms of number of iterations, although each iteration involves a higher operation count. When the CGS method exhibits irregular convergence, the TFQMR method can produce much smoother, almost monotonic convergence curves. However, the close relationship between the CGS and TFQMR method implies that the *overall* speed of convergence is similar for both methods. In some cases, the TFQMR method may converge faster than the CGS method.

3.5 General Considerations

For each method, a sequence of solution iterates $\{x_i\}$ is generated such that, hopefully, the sequence of the residual norms $\{\|r_i\|\}$ converges to a required tolerance. Note that, in general, convergence, when it occurs, is not monotonic.

In the RGMRES and Bi-CGSTAB(ℓ) methods above, your program must provide the **maximum** number of basis vectors used, m or ℓ, respectively; however, a **smaller** number of basis vectors may be generated and used when the stability of the solution process requires this (see Section 8).

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 coefficients matrix can be improved or eigenvalues in its spectrum can be made to coalesce. An orthogonal basis for the Krylov subspace span$\left\{\bar{A}^k\bar{r}_0\right\}$, 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 *left* preconditioner $\mathbf{m}^{-1}$ can be used by the RGMRES, CGS and TFQMR methods, such that $\bar{A} = \mathbf{m}^{-1}A \sim I_n$ in (1), where I_n is the identity matrix of order n; a *right* preconditioner $\mathbf{m}^{-1}$ can be used by the Bi-CGSTAB(ℓ) method, such that $\bar{A} = A\mathbf{m}^{-1} \sim I_n$. 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 $v = A^{\mathrm{T}}u$ (the latter only being required when an estimate of $\|A\|_1$ or $\|A\|_\infty$ is computed internally), and to solve the preconditioning equations $\mathbf{m}v = u$ are required, i.e., explicit information about $\mathbf{m}$, or its inverse is not required at any stage.

The first termination criterion

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

is also available for all three methods. In (2), $p = 1$, ∞ or 2 and τ denotes a user-specified tolerance subject to $\max\big(10, \sqrt{n}\big)$, $\epsilon \leq \tau < 1$, where ϵ is the ***machine precision***. Facilities are provided for the estimation of the norm of the coefficients matrix $\|A\|_1$ or $\|A\|_\infty$, when this is not known in advance, 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 are not used because any such criteria would require information about the condition number $\kappa(A)$ which is not easily obtainable.

The second termination criterion

$$\|\bar{r}_k\|_2 \leq \tau\big(\|\bar{r}_0\|_2 + \sigma_1(\bar{A}), \|\Delta\bar{x}_k\|_2\big) \tag{3}$$

is available for all methods except TFQMR. In (3), $\sigma_1(\bar{A}) = \big\|\bar{A}\big\|_2$ 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) for the Bi-CGSTAB(ℓ) method with $\ell > 1$. Only the RGMRES method provides facilities to estimate $\sigma_1(\bar{A})$ internally, when this is not supplied (see Section 8).

Termination criterion (2) is the recommended choice, despite its additional costs per iteration when using the Bi-CGSTAB(ℓ) method with $\ell > 1$. 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 f11bd must be called, followed by the solver f11be. f11bf can be called either when f11be is carrying out a monitoring step or after f11be has completed its tasks. Incorrect sequencing will raise an error condition.

In general, it is not possible to recommend one method in preference to another. RGMRES is often used in the solution of systems arising from PDEs. On the other hand, it can easily stagnate when the size m of the orthogonal basis is too small, or the preconditioner is not good enough. CGS can be the fastest method, but the computed residuals can exhibit instability which may greatly affect the convergence and quality of the solution. Bi-CGSTAB(ℓ) seems robust and reliable, but it can be slower than the other methods: if a preconditioner is used and $\ell > 1$, Bi-CGSTAB(ℓ) computes the solution of the preconditioned system $\bar{x}_k = \mathbf{m}x_k$: the preconditioning equations must be solved to obtain the required solution. The algorithm employed limits to 10% or less, when no intermediate monitoring is requested, the number of times the preconditioner has to be thus applied compared with the total number of applications of the preconditioner. TFQMR can be viewed as a more robust variant of the CGS method: it shares the CGS method speed but avoids the CGS fluctuations in the residual, which may give rise to instability. Also, when the termination criterion (2) is used, the CGS, Bi-CGSTAB(ℓ) and TFQMR methods will restart the iteration automatically when necessary in order to solve the given problem.

4 References

Arnoldi W 1951 The principle of minimized iterations in the solution of the matrix eigenvalue problem *Quart. Appl. Math.* **9** 17–29

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

Freund R W 1993 A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems *SIAM J. Sci. Comput.* **14** 470–482

Freund R W and Nachtigal N 1991 QMR: a Quasi-Minimal Residual Method for Non-Hermitian Linear Systems *Numer. Math.* **60** 315–339

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

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

Saad Y and Schultz M 1986 GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **7** 856–869

Sleijpen G L G and Fokkema D R 1993 BiCGSTAB(ℓ) for linear equations involving matrices with complex spectrum *ETNA* **1** 11–32

Sonneveld P 1989 CGS, a fast Lanczos-type solver for nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **10** 36–52

Van der Vorst H 1989 Bi-CGSTAB, a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems *SIAM J. Sci. Statist. Comput.* **13** 631–644

5 Parameters

5.1 Compulsory Input Parameters

1: **method – string**

The iterative method to be used.

method = 'RGMRES'

Restarted generalized minimum residual method.

method = 'CGS'

Conjugate gradient squared method.

method = 'BICGSTAB'

Bi-conjugate gradient stabilized (ℓ) method.

method = 'TFQMR'

Transpose-free quasi-minimal residual method.

Constraint: **method** = 'RGMRES', 'CGS', 'BICGSTAB' or 'TFQMR'.

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: $\mathbf{n} > 0$.

4: **m – int32 scalar**

If **method** = 'RGMRES', **m** is the dimension m of the restart subspace.

If **method** = 'BICGSTAB', **m** is the order ℓ of the polynomial Bi-CGSTAB method.

Otherwise, **m** is not referenced.

Constraints:

if **method** = 'RGMRES', $0 < \mathbf{m} \leq \min(\mathbf{n}, 50)$;
if **method** = 'BICGSTAB', $0 < \mathbf{m} \leq \min(\mathbf{n}, 10)$.

5: **tol – double scalar**

The tolerance τ for the termination criterion. If $\mathbf{tol} \leq 0.0, \tau = \max\left(\sqrt{\epsilon}, \sqrt{n}\epsilon\right)$ is used, where ϵ is the ***machine precision***. Otherwise $\tau = \max\left(\mathbf{tol}, 10\epsilon, \sqrt{n}\epsilon\right)$ is used.

Constraint: $\mathbf{tol} < 1.0$.

6: **maxitn – int32 scalar**

The maximum number of iterations.

Constraint: $\mathbf{maxitn} > 0$.

7: **anorm – double scalar**

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

If $\mathbf{anorm} \leq 0.0$, $\mathbf{iterm} = 1$ and **norm_p** = '1' or 'I', then $\|A\|_1 = \|A\|_\infty$ is estimated internally by f11be.

If $\mathbf{iterm} = 2$, **anorm** is not referenced.

Constraint: if $\mathbf{iterm} = 1$ and $\mathbf{norm\_p} = 2$, $\mathbf{anorm} > 0.0$.

8: **sigmax – double scalar**

If $\mathbf{iterm} = 2$, the largest singular value σ_1 of the preconditioned iteration matrix; otherwise, **sigmax** is not referenced.

If $\mathbf{sigmax} \leq 0.0$, $\mathbf{iterm} = 2$ and **method** = 'RGMRES', then the value of σ_1 will be estimated internally.

Constraint: if **method** = 'CGS' or 'BICGSTAB' and $\mathbf{iterm} = 2$, $\mathbf{sigmax} > 0.0$.

9: **monit – int32 scalar**

If $\mathbf{monit} > 0$, the frequency at which a monitoring step is executed by f11be: if **method** = 'CGS' or 'TFQMR', a monitoring step is executed every **monit** iterations; otherwise, a monitoring step is executed every **monit** super-iterations (groups of up to m or ℓ iterations for RGMRES or Bi-CGSTAB(ℓ), respectively).

There are some additional computational costs involved in monitoring the solution and residual vectors when the Bi-CGSTAB(ℓ) method is used with $\ell > 1$.

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

10: **lwork – int32 scalar**

Constraint: **lwork** ≥ 100.

Note: although the minimum value of **lwork** ensures the correct functioning of f11bd, a larger value is required by the other functions in the suite, namely f11be and f11bf. The required value is as follows:

Method	**Requirements**
RGMRES	**lwork** $= 100 + n(m+3) + m(m+5) + 1$, where m is the dimension of the basis.
CGS	**lwork** $= 100 + 7n$.
Bi-CGSTAB(ℓ)	**lwork** $= 100 + (2n + \ell)(\ell + 2) + p$, where ℓ is the order of the method.
TFQMR	**lwork** $= 100 + 10n$,

where

$p = 2n$ if $\ell > 1$ and **iterm** $= 2$ was supplied.

$p = n$ if $\ell > 1$ and a preconditioner is used or **iterm** $= 2$ was supplied.

$p = 0$ otherwise.

5.2 Optional Input Parameters

1: **norm_p – string**

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

norm_p = '1'

l_1 norm.

norm_p = 'I'

l_∞ norm.

norm_p = '2'

l_2 norm.

Suggested value:

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

Default:

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

Constraints:

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

2: **weight – string**

Specifies whether a vector w of user-supplied weights is to be used in the computation of the vector norms required in 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 f11be.

weight = 'N'

All weights are implicitly set equal to one. Weights do not need to be supplied on initial entry to f11be.

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

Default: 'N'

Constraints:

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

3: **iterm – int32 scalar**

Defines the termination criterion to be used.

iterm = 1

Use the termination criterion defined in (2).

iterm = 2

Use the termination criterion defined in (3).

Suggested value: **iterm** = 1.

Default: 1

Constraints:

if **method** = 'RGMRES', 'CGS' or 'BICGSTAB', **iterm** = 1 or 2;
if **method** = 'TFQMR', **iterm** = 1.

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 f11be. (See also Section 5 of the document for f11be.)

2: **work**(**lwork**) **– double array**

The workspace **work** is initialized by f11bd. It must **not** be modified before calling the next function in the suite, namely f11be.

3: **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: **norm_p**, 4: **weight**, 5: **iterm**, 6: **n**, 7: **m**, 8: **tol**, 9: **maxitn**, 10: **anorm**, 11: **sigmax**, 12: **monit**, 13: **lwreq**, 14: **work**, 15: **lwork**, 16: **ifail**.

ifail $= 1$

f11bd has been called out of sequence.

7 Accuracy

Not applicable.

8 Further Comments

RGMRES can estimate internally the maximum singular value σ_1 of the iteration matrix, using $\sigma_1 \sim \|T\|_1$, where T is the upper triangular matrix obtained by QR factorization of the upper Hessenberg matrix generated by the Arnoldi process. The computational costs of this computation are negligible when compared to the overall costs.

Loss of orthogonality in the RGMRES method, or of bi-orthogonality in the Bi-CGSTAB(ℓ) method may degrade the solution and speed of convergence. For both methods, the algorithms employed include checks on the basis vectors so that the number of basis vectors used for a given super-iteration may be less than the value specified in the input parameter **m**. Also, the CGS, Bi-CGSTAB(ℓ) and TFQMR methods will restart automatically the computation from the last available iterates, when the stability of the solution process requires it.

Termination criterion (3), when available, involves only the residual (or norm of the residual) produced directly by the iteration process: this may differ from the norm of the true residual $\tilde{r}_k = b - Ax_k$, particularly when the norm of the residual is very small. Also, if the norm of the initial estimate of the solution is much larger than the norm of the exact solution, convergence can be achieved despite very large errors in the solution. On the other hand, termination criterion (3) is cheaper to use and inspects the progress of the actual iteration. Termination criterion (2) should be preferred in most cases, despite its slightly larger costs.

9 Example

```
method = 'BICGSTAB';
precon = 'P';
n = int32(8);
m = int32(1);
nnz = int32(24);
a = zeros(10000, 1);
a(1:24) = [2; -1; 1; 4; -3; 2; -7; 2; 3; -4; 5; 5; -1; 8; -3; -6; 5; 2; -
5; -1; 6; -1; 2; 3];
irow = zeros(10000, 1, 'int32');
irow(1:24) = [int32(1);
        int32(1);
        int32(1);
        int32(2);
        int32(2);
        int32(2);
        int32(3);
        int32(3);
        int32(4);
        int32(4);
        int32(4);
        int32(4);
        int32(5);
        int32(5);
        int32(5);
        int32(6);
        int32(6);
        int32(6);
        int32(7);
        int32(7);
        int32(7);
```

```
        int32(8);
        int32(8);
        int32(8)];
icol = zeros(10000, 1, 'int32');
icol(1:24) = [int32(1);
        int32(4);
        int32(8);
        int32(1);
        int32(2);
        int32(5);
        int32(3);
        int32(6);
        int32(1);
        int32(3);
        int32(4);
        int32(7);
        int32(2);
        int32(5);
        int32(7);
        int32(1);
        int32(3);
        int32(6);
        int32(3);
        int32(5);
        int32(7);
        int32(2);
        int32(6);
        int32(8)];
lfill = int32(0);
ipivp = zeros(8, 1, 'int32');
ipivq = zeros(8, 1, 'int32');
dtol = 0;
milu = 'N';
tol = 1e-08;
maxitn = int32(20);
anorm = 0;
sigmax = 0;
monit = int32(1);
lwork = int32(6000);
irevcm = int32(0);
u = zeros(8, 1);
v = [6;
     8;
     -9;
     46;
     17;
     21;
     22;
     34];
wgt = zeros(8, 1);
[a, irow, icol, ipivp, ipivq, istr, idiag, nnzc, npivm, ifail] = ...
    f11da(nnz, a, irow, icol, lfill, dtol, milu, ipivp, ipivq);
[lwreq, work, ifail] = ...
       f11bd(method, precon, n, m, tol, maxitn, anorm, sigmax, monit,
lwork, 'norm_p', '1');
while (irevcm ~= 4)
  [irevcm, u, v, work, ifail] = f11be(irevcm, u, v, wgt, work);

  if (irevcm == -1)
    [v, ifail] = f11xa('T', a(1:nnz), irow(1:nnz), icol(1:nnz), 'N', u);
  elseif (irevcm == 1)
    [v, ifail] = f11xa('N', a(1:nnz), irow(1:nnz), icol(1:nnz), 'N', u);
  elseif (irevcm == 2)
     [v, ifail] = f11db('N', a, irow, icol, ipivp, ipivq, istr, idiag,
'N', u);
    if (ifail ~= 0)
      irevcm = 6;
    end
  elseif (irevcm == 3)
    [itn, stplhs, stprhs, anorm, sigmax, ifail] = f11bf(work);
```

```
        fprintf('\nMonitoring  at  iteration  number  %d\nresidual  norm:
%14.4e\n', itn, stplhs);
    fprintf('\n   Solution Vector  Residual Vector\n');
    for i = 1:n
      fprintf('%+16.4e %+16.4e\n', u(i), v(i));
    end
  end
end
% Get information about the computation
[itn, stplhs, stprhs, anorm, sigmax, ifail] = f11bf(work);
fprintf('\nNumber 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('i-norm of matrix a:                      %14.4e\n', anorm);
fprintf('\n   Solution Vector  Residual Vector\n');
for i = 1:n
  fprintf('%+16.4e %+16.4e\n', u(i), v(i));
end
```

```
Monitoring at iteration number 1
residual norm:     1.4059e+02

   Solution Vector  Residual Vector
     -4.5858e+00      +1.5256e+01
     +1.0154e+00      +2.6624e+01
     -2.2234e+00      -8.7498e+00
     +6.0097e+00      +1.8602e+01
     +1.3827e+00      +8.2821e+00
     -7.9070e+00      +2.0416e+01
     +4.4270e-01      +9.6094e+00
     +5.9248e+00      +3.3055e+01

Monitoring at iteration number 2
residual norm:     3.2742e+01

   Solution Vector  Residual Vector
     +4.1642e+00      -2.9585e+00
     +4.9370e+00      -5.5523e+00
     +4.8101e+00      +8.2070e-01
     +5.4324e+00      -1.6828e+01
     +5.8531e+00      +5.5975e-01
     +1.1925e+01      -1.9150e+00
     +8.4826e+00      +1.0081e+00
     +6.0625e+00      -3.1004e+00
Number of iterations for convergence: 3
Residual norm:                                1.0474e-08
Right-hand side of termination criteria:      5.5900e-06
i-norm of matrix a:                           1.1000e+01
   Solution Vector  Residual Vector
+1.0000e+00      -1.3686e-09
     +2.0000e+00      -2.6364e-09
     +3.0000e+00      +2.2691e-10
     +4.0000e+00      -3.2517e-09
     +5.0000e+00      +6.3660e-10
     +6.0000e+00      -5.2943e-10
     +7.0000e+00      +9.6704e-10
     +8.0000e+00      -8.5717e-10
```