# NAG Toolbox

# nag_eigen_real_symm_sparse_arnoldi (f02fk)

## 1 Purpose

nag_eigen_real_symm_sparse_arnoldi (f02fk) computes selected eigenvalues and eigenvectors of a real sparse symmetric matrix.

## 2 Syntax

```
[nconv, w, v, resid, user, ifail] = nag_eigen_real_symm_sparse_arnoldi(n, a, irow, icol, nev, ncv, sigma, monit, option, 'nnz', nnz, 'user', user)
[nconv, w, v, resid, user, ifail] = f02fk(n, a, irow, icol, nev, ncv, sigma, monit, option, 'nnz', nnz, 'user', user)
```

## 3 Description

nag_eigen_real_symm_sparse_arnoldi (f02fk) computes selected eigenvalues and the corresponding right eigenvectors of a real sparse symmetric matrix $A$:

$$Av_i = \lambda_i v_i.$$

A specified number, $n_{ev}$, of eigenvalues $\lambda_i$, or the shifted inverses $\mu_i = 1/(\lambda_i - \sigma)$, may be selected either by largest or smallest modulus, largest or smallest value, or, largest and smallest values (both ends). Convergence is generally faster when selecting larger eigenvalues, smaller eigenvalues can always be selected by choosing a zero inverse shift ($\sigma = 0.0$). When eigenvalues closest to a given value are required then the shifted inverses of largest magnitude should be selected with shift equal to the required value.

The sparse matrix $A$ is stored in symmetric coordinate storage (SCS) format. See Section 2.1.2 in the F11 Chapter Introduction.

nag_eigen_real_symm_sparse_arnoldi (f02fk) uses an implicitly restarted Arnoldi (Lanczos) iterative method to converge approximations to a set of required eigenvalues and corresponding eigenvectors. Further algorithmic information is given in Section 9 while a fuller discussion is provided in the F12 Chapter Introduction. If shifts are to be performed then operations using shifted inverse matrices are performed using a direct sparse solver.

## 4 References

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

Lehoucq R B, Sorensen D C and Yang C (1998) *ARPACK Users' Guide: Solution of Large-scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods* SIAM, Philidelphia

## 5 Parameters

### 5.1 Compulsory Input Parameters

1: **n** – INTEGER

$n$, the order of the matrix $A$.

*Constraint*: $\mathbf{n} > 0$.

2: **a**(**nnz**) – REAL (KIND=nag_wp) array

The array of nonzero elements of the lower triangular part of the $n$ by $n$ symmetric matrix $A$.

3: **irow**(**nnz**) – INTEGER array

4: **icol**(**nnz**) – INTEGER array

The row and column indices of the elements supplied in array **a**.

If **irow**$(k) = i$ and **icol**$(k) = j$ then $A_{ij}$ is stored in **a**$(k)$. **irow** does not need to be ordered, an internal call to nag_sparse_real_symm_sort (f11zb) forces the correct ordering.

*Constraint*:

**irow** and **icol** must satisfy these constraints:$1 \leq$ **irow**$(i) \leq$ **n** and $1 \leq$ **icol**$(i) \leq$ **irow**$(i)$, for $i = 1, 2, \ldots,$ **nnz**.

5: **nev** – INTEGER

The number of eigenvalues to be computed.

*Constraint*: $0 <$ **nev** $<$ **n** $- 1$.

6: **ncv** – INTEGER

The dimension of the array **w**. the number of Arnoldi basis vectors to use during the computation.

At present there is no *a priori* analysis to guide the selection of **ncv** relative to **nev**. However, it is recommended that **ncv** $\geq 2 \times$ **nev** $+ 1$. If many problems of the same type are to be solved, you should experiment with increasing **ncv** while keeping **nev** fixed for a given test problem. This will usually decrease the required number of matrix-vector operations but it also increases the work and storage required to maintain the orthogonal basis vectors. The optimal 'cross-over' with respect to computation time is problem dependent and must be determined empirically.

*Constraint*: **nev** $<$ **ncv** $\leq$ **n**.

7: **sigma** – REAL (KIND=nag_wp)

If the **Shifted Inverse** mode has been selected then **sigma** contains the real shift used; otherwise **sigma** is not referenced. This mode can be selected by setting the appropriate options in the user-supplied function **option**.

8: **monit** – SUBROUTINE, supplied by the NAG Library or the user.

**monit** is used to monitor the progress of nag_eigen_real_symm_sparse_arnoldi (f02fk). **monit** may be the dummy function nag_eigen_arnoldi_monit_symm (f02fkz) if no monitoring is actually required. (nag_eigen_arnoldi_monit_symm (f02fkz) is included in the NAG Toolbox.) **monit** is called after the solution of each eigenvalue sub-problem and also just prior to return from nag_eigen_real_symm_sparse_arnoldi (f02fk).

```
[istat, user] = monit(ncv, niter, nconv, w, rzest, istat, user)
```

**Input Parameters**

1: **ncv** – INTEGER

The dimension of the arrays **w** and **rzest**. The number of Arnoldi basis vectors used during the computation.

2: **niter** – INTEGER

The number of the current Arnoldi iteration.

3: **nconv** – INTEGER

The number of converged eigenvalues so far.

4: **w**(**ncv**) – REAL (KIND=nag_wp) array

The first **nconv** elements of **w** contain the converged approximate eigenvalues.

5: **rzest**(**ncv**) – REAL (KIND=nag_wp) array

The first **nconv** elements of **rzest** contain the Ritz estimates (error bounds) on the converged approximate eigenvalues.

6: **istat** – INTEGER

Set to zero.

7: **user** – INTEGER array

**monit** is called from nag_eigen_real_symm_sparse_arnoldi (f02fk) with the object supplied to nag_eigen_real_symm_sparse_arnoldi (f02fk).

**Output Parameters**

1: **istat** – INTEGER

If set to a nonzero value nag_eigen_real_symm_sparse_arnoldi (f02fk) returns immediately with **ifail** = 9.

2: **user** – INTEGER array

9: **option** – SUBROUTINE, supplied by the NAG Library or the user.

You can supply non-default options to the Arnoldi eigensolver by repeated calls to nag_sparseig_real_symm_option (f12fd) from within **option**. (Please note that it is only necessary to call nag_sparseig_real_symm_option (f12fd); no call to nag_sparseig_real_symm_init (f12fa) is required from within **option**.) For example, you can set the mode to **Shifted Inverse**, you can increase the **Iteration Limit** beyond its default and you can print varying levels of detail on the iterative process using **Print Level**.

If only the default options (including that the eigenvalues of largest magnitude are sought) are to be used then **option** may be the dummy function nag_eigen_arnoldi_option (f02eky) (nag_eigen_arnoldi_option (f02eky) is included in the NAG Toolbox). See Section 10 for an example of using **option** to set some non-default options.

```
[icomm, comm, istat, user] = option(icomm, comm, istat, user)
```

**Input Parameters**

1: **icomm**(:) – INTEGER array

Contains details of the default option set. This array must be passed as argument **icomm** in any call to nag_sparseig_real_symm_option (f12fd).

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

Contains details of the default option set. This array must be passed as argument **comm** in any call to nag_sparseig_real_symm_option (f12fd).

3: **istat** – INTEGER

Set to zero.

4: **user** – INTEGER array

**option** is called from nag_eigen_real_symm_sparse_arnoldi (f02fk) with the object supplied to nag_eigen_real_symm_sparse_arnoldi (f02fk).

**Output Parameters**

1: **icomm**(:) – INTEGER array

Contains data on the current options set which may be altered from the default set via calls to nag_sparseig_real_symm_option (f12fd).

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

Contains data on the current options set which may be altered from the default set via calls to nag_sparseig_real_symm_option (f12fd).

3: **istat** – INTEGER

If set to a nonzero value nag_eigen_real_symm_sparse_arnoldi (f02fk) returns immediately with **ifail** $= 10$.

4: **user** – INTEGER array

## 5.2 Optional Input Parameters

1: **nz** – INTEGER

*Default*: the dimension of the arrays **irow**, **icol** and the dimension of the array **a**. (An error is raised if these dimensions are not equal.)

The dimension of the array **a**.the number of nonzero elements in the lower triangular part of the matrix $A$.

*Constraint*: $1 \leq \mathbf{nnz} \leq \mathbf{n} \times (\mathbf{n}+1)/2$.

2: **user** – INTEGER array

**user** is not used by nag_eigen_real_symm_sparse_arnoldi (f02fk), but is passed to **monit** and **option**. 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: **nconv** – INTEGER

The number of converged approximations to the selected eigenvalues. On successful exit, this will normally be **nev**.

2: **w**(**ncv**) – REAL (KIND=nag_wp) array

The first **nconv** elements contain the converged approximations to the selected eigenvalues.

3: **v**($ldv$,:) – REAL (KIND=nag_wp) array

The first dimension of the array **v** will be **n**.

The second dimension of the array **v** will be **ncv**.

Contains the eigenvectors associated with the eigenvalue $\lambda_i$, for $i = 1, 2, \ldots,$ **nconv** (stored in **w**). For eigenvalue, $\lambda_j$, the corresponding eigenvector is stored in **v**$(i, j)$, for $i = 1, 2, \ldots, n$.

4: **resid**(**nev**) – REAL (KIND=nag_wp) array

The residual $\|Aw_i - \lambda_i w_i\|_2$ for the estimates to the eigenpair $\lambda_i$ and $w_i$ is returned in **resid**$(i)$, for $i = 1, 2, \ldots,$ **nconv**.

5: **user** – INTEGER array

6: **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** $= 1$

Constraint: **n** $> 0$.

**ifail** $= 2$

Constraint: **nnz** $> 0$.

Constraint: **nnz** $\leq$ **n** $\times$ (**n** $+ 1)/2$.

**ifail** $= 4$

Constraint: $1 \leq$ **irow**$(i) \leq$ **n**.

**ifail** $= 5$

Constraint: $1 \leq$ **icol**$(i) \leq$ **irow**$(i)$.

**ifail** $= 6$

Constraint: **nev** $<$ (**n** $- 1$).

Constraint: **nev** $> 0$.

**ifail** $= 7$

Constraint: **ncv** $>$ **nev**.

Constraint: **ncv** $\leq$ **n**.

**ifail** $= 8$

On entry, the matrix $(A - \sigma I)$ is numerically singular and could not be inverted. Try perturbing the value of $\sigma$.

**ifail** $= 9$

User requested termination in **monit**.

**ifail** $= 10$

User requested termination in **option**.

**ifail** $= 14$

Constraint: $ldv \geq$ **n**.

**ifail** $= 20$

The maximum number of iterations, through the optional parameter **Iteration Limit**, has been set to a non-positive value.

**ifail** $= 21$

The option **Both Ends** has been set but only 1 eigenvalue is requested.

**ifail** $= 22$

The maximum number of iterations has been reached.

**ifail** $= 30$

A serious error, code ⟨*value*⟩, has occurred in an internal call to ⟨*value*⟩. Check all function calls and array sizes. If the call is correct then please contact NAG for assistance.

**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

The relative accuracy of a Ritz value (eigenvalue approximation), $\lambda$, is considered acceptable if its Ritz estimate $\leq$ **Tolerance** $\times \lambda$. The default value for **Tolerance** is the ***machine precision*** given by nag_machine_precision (x02aj). The Ritz estimates are available via the **monit** function at each iteration in the Arnoldi process, or can be printed by setting option **Print Level** to a positive value.

## 8 Further Comments

nag_eigen_real_symm_sparse_arnoldi (f02fk) calls functions based on the ARPACK suite in Chapter F12. These functions use an implicitly restarted Lanczos iterative method to converge to approximations to a set of required eigenvalues (see the F12 Chapter Introduction).

In the default **Regular** mode, only matrix-vector multiplications are performed using the sparse matrix $A$ during the Lanczos process; nag_sparse_real_symm_matvec (f11xe) can be used to perform this task. Each iteration is therefore cheap computationally, relative to the alternative, **Shifted Inverse**, mode described below. It is most efficient (i.e., the total number of iterations required is small) when the eigenvalues of largest magnitude are sought and these are distinct.

Although there is an option for returning the smallest eigenvalues using this mode (see **Smallest Magnitude** option), the number of iterations required for convergence will be far greater or the method may not converge at all. However, where convergence is achieved, **Regular** mode may still prove to be the most efficient since no inversions are required. Where smallest eigenvalues are sought and **Regular** mode is not suitable, or eigenvalues close to a given real value are sought, the **Shifted Inverse** mode should be used.

If the **Shifted Inverse** mode is used (via a call to nag_sparseig_real_symm_option (f12fd) in **option**) then the matrix $A - \sigma I$ is used in linear system solves by the Lanczos process. This is first factorized internally using a direct sparse $LDL^{\mathrm{T}}$ factorization under the assumption that the matrix is indefinite. If the factorization determines that the matrix is numerically singular then the function exits with an error. In this situation it is normally sufficient to perturb $\sigma$ by a small amount and call nag_eigen_real_symm_sparse_arnoldi (f02fk) again. After successful factorization, subsequent solves are performed by backsubstitution using the sparse factorization.

Finally, nag_eigen_real_symm_sparse_arnoldi (f02fk) transforms the eigenvectors. Each eigenvector $w$ is normalized so that $\|w\|_2 = 1$.

The monitoring function **monit** provides some basic information on the convergence of the Lanczos iterations. Much greater levels of detail on the Lanczos process are available via option **Print Level**. If

this is set to a positive value then information will be printed, by default, to standard output. The destination of monitoring information can be changed using the **Monitoring** option.

# 9 Example

This example solves $Ax = \lambda x$ in **Shifted Inverse** mode, where $A$ is obtained from the standard central difference discretization of the one-dimensional Laplacian operator $\frac{\partial^2 u}{\partial x^2}$ on $[0, 1]$, with zero Dirichlet boundary conditions.

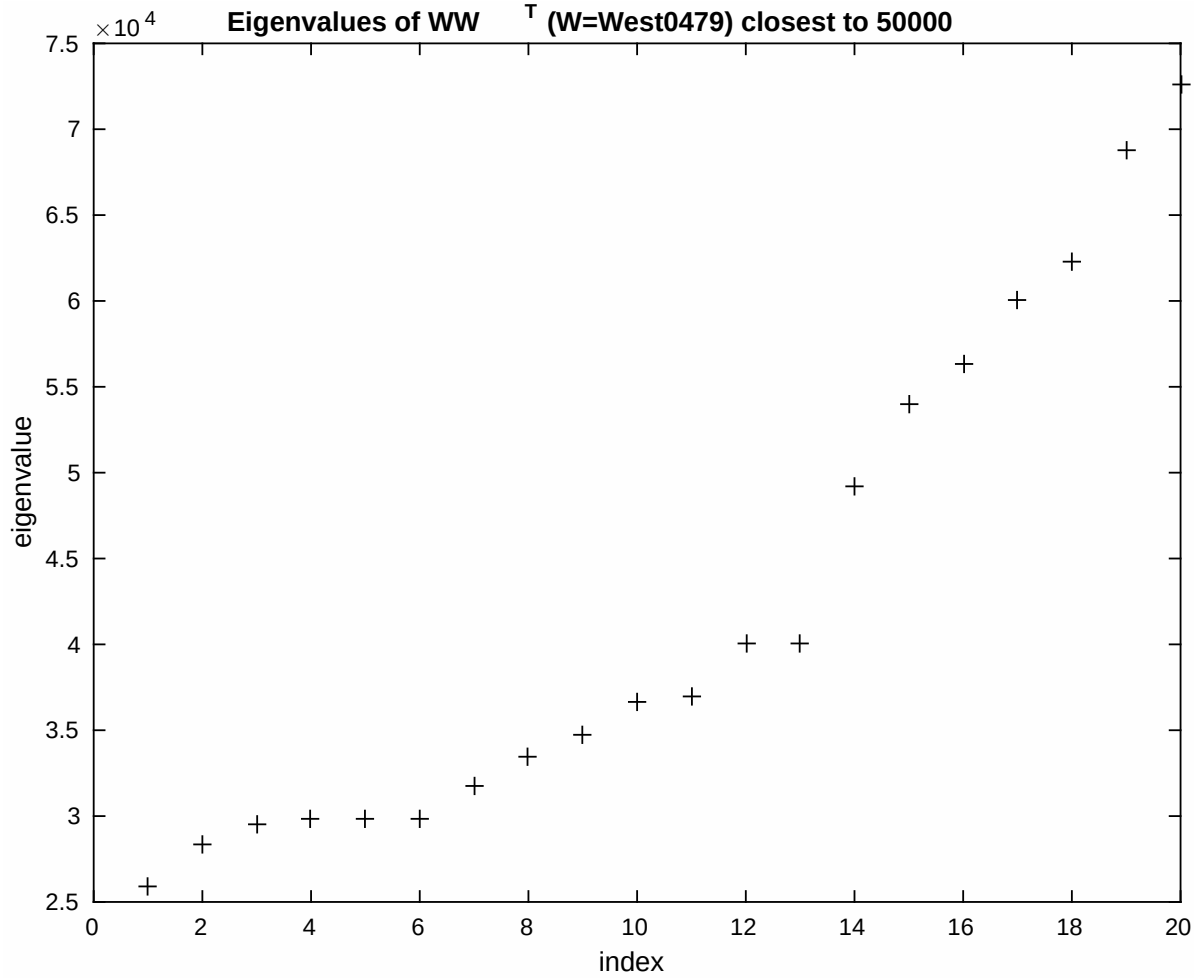
## 9.1 Program Text

```
     function f02fk_example

fprintf('f02fk example results\n\n');

% This example demonstrates the use of f02fk to evaluate a number of
% eigenvalues of a sparse symmetric matrix closest to a given value.

% Use existing sparse matrix to generate symmetric one.
load('west0479.mat')
W = west0479;
S = W * W';

%' Extract details of sparse matrix S.
[irow,icol,a] = find(S);
n   = nag_int(size(S,1));
nnz = nag_int(size(irow,1));

% zero out strictly upper triangle
for i = 1 : nnz
  if (icol(i)>irow(i));
     a(i) = 0.0;
  end
end

% Use f11za to remove zero entries.
irow = nag_int(irow);
icol = nag_int(icol);
dup  = 'R';
zero = 'R';
[nnz, a, icol, irow, icolzp, ifail] = ...
  f11za(...
        n, nnz, a, icol, irow, dup, zero);

% Evaluate nev eigenvalues (w) closest to sigma.
nev   = nag_int(20);
ncv   = nag_int(60);
sigma = 50000;

[nconv, w, v, resid, user, ifail] = ...
    f02fk(...
          n, a, irow, icol, nev, ncv, sigma, 'f02fkz', @option);

fprintf('\n The %d Ritz values closest to %13.5e are:\n', nconv, sigma);

disp(w(1:nconv));

fig1 = figure;
plot(w(1:nconv),'k+');
title('Eigenvalues of WW^T (W=West0479) closest to 50000');
xlabel('index');
ylabel('eigenvalue');

function [istat, user] = monit(ncv, niter, nconv, w, rzest, istat, user);
```

```
  istat = nag_int(0);

function [icomm, comm, istat, user] = option(icomm, comm, istat, user);
    [icomm, comm, ifail] = f12fd('Shifted Inverse', icomm, comm);
    istat = ifail;
```

## 9.2 Program Results

```
     f02fk example results

 The 20 Ritz values closest to   5.00000e+04 are:
   1.0e+04 *

    2.5904
    2.8321
    2.9565
    2.9806
    2.9806
    2.9865
    3.1759
    3.3510
    3.4773
    3.6629
    3.6940
    4.0007
    4.0008
    4.9172
    5.3954
    5.6332
    6.0003
    6.2267
    6.8829
    7.2555
```




## 10 Optional Parameters

Internally nag_eigen_real_symm_sparse_arnoldi (f02fk) calls functions from the suite nag_sparseig_real_symm_init (f12fa), nag_sparseig_real_symm_iter (f12fb), nag_sparseig_real_symm_proc (f12fc), nag_sparseig_real_symm_option (f12fd) and nag_sparseig_real_symm_monit (f12fe). Several optional parameters for these computational functions define choices in the problem specification or the algorithm logic. In order to reduce the number of formal arguments of nag_eigen_real_symm_sparse_arnoldi (f02fk) these optional parameters are also used here and have associated *default values* that are usually appropriate. Therefore, you need only specify those optional parameters whose values are to be different from their default values.

Optional parameters may be specified via the user-supplied function **option** in the call to nag_eigen_real_symm_sparse_arnoldi (f02fk). **option** must be coded such that one call to nag_sparseig_real_symm_option (f12fd) is necessary to set each optional parameter. All optional parameters you do not specify are set to their default values.

The remainder of this section can be skipped if you wish to use the default values for all optional parameters.

The following is a list of the optional parameters available. A full description of each optional parameter is provided in Section 11.1.

**Advisory**
**Both Ends**
**Defaults**
**Iteration Limit**

**Largest Algebraic**
**Largest Magnitude**
**List**
**Monitoring**
**Nolist**
**Print Level**
**Regular**
**Regular Inverse**
**Shifted Inverse**
**Smallest Algebraic**
**Smallest Magnitude**
**Tolerance**

## 10.1 Description of the Optional Parameters

For each option, we give a summary line, a description of the optional parameter and details of constraints.

The summary line contains:

the keywords, where the minimum abbreviation of each keyword is underlined;

a parameter value, where the letters $a$, $i$ and $r$ denote options that take character, integer and real values respectively;

the default value, where the symbol $\epsilon$ is a generic notation for ***machine precision*** (see nag_machine_precision (x02aj)).

Keywords and character values are case and white space insensitive.

**Advisory** $i$ Default $=$ the value returned by nag_file_set_unit_advisory (x04ab)

If the optional parameter **List** is set then optional parameter specifications are listed in a **List** ***file*** by setting the option to a file identification (unit) number associated with **Advisory** messages (see nag_file_set_unit_advisory (x04ab) and nag_file_open (x04ac)).

**Defaults**

This special keyword may be used to reset all optional parameters to their default values.

**Iteration Limit** $i$ Default $= 300$

The limit on the number of Lanczos iterations that can be performed before nag_sparseig_real_symm_iter (f12fb) exits. If not all requested eigenvalues have converged to within **Tolerance** and the number of Lanczos iterations has reached this limit then nag_sparseig_real_symm_iter (f12fb) exits with an error; nag_sparseig_real_symm_proc (f12fc) can still be called subsequently to return the number of converged eigenvalues, the converged eigenvalues and, if requested, the corresponding eigenvectors.

**Largest Magnitude** Default
**Both Ends**
**Largest Algebraic**
**Smallest Algebraic**
**Smallest Magnitude**

The Lanczos iterative method converges on a number of eigenvalues with given properties. The default is for nag_sparseig_real_symm_iter (f12fb) to compute the eigenvalues of largest magnitude using **Largest Magnitude**. Alternatively, eigenvalues may be chosen which have **Largest Algebraic** part, **Smallest Magnitude**, or **Smallest Algebraic** part; or eigenvalues which are from **Both Ends** of the algebraic spectrum.

**Nolist** Default
**List**

Normally each optional parameter specification is not listed as it is supplied. This behaviour can be changed using the **List** and **Nolist** options.

**Monitoring** $i$ Default $= -1$

If $i > 0$, monitoring information is output to channel number $i$ during the solution of each problem; this may be the same as the **Advisory** channel number. The type of information produced is dependent on the value of **Print Level**, see the description of the optional parameter **Print Level** for details of the information produced. Please see nag_file_open (x04ac) to associate a file with a given channel number.

**Print Level** $i$ Default $= 0$

This controls the amount of printing produced by nag_eigen_real_symm_sparse_arnoldi (f02fk) as follows.

$= 0$ No output except error messages. If you want to suppress all output, set **Print Level** $= 0$.

$> 0$ The set of selected options.

$= 2$ Problem and timing statistics on final exit from f12fb.

$\geq 5$ A single line of summary output at each Lanczos iteration.

$\geq 10$ If **Monitoring** is set, then at each iteration, the length and additional steps of the current Lanczos factorization and the number of converged Ritz values; during re-orthogonalization, the norm of initial/restarted starting vector; on a final Lanczos iteration, the number of update iterations taken, the number of converged eigenvalues, the converged eigenvalues and their Ritz estimates.

$\geq 20$ Problem and timing statistics on final exit from f12fb. If **Monitoring** $> 0$, **Monitoring** is set, then at each iteration, the number of shifts being applied, the eigenvalues and estimates of the symmetric tridiagonal matrix $H$, the size of the Lanczos basis, the wanted Ritz values and associated Ritz estimates and the shifts applied; vector norms prior to and following re-orthogonalization.

$\geq 30$ If **Monitoring** $> 0$, **Monitoring** is set, then on final iteration, the norm of the residual; when computing the Schur form, the eigenvalues and Ritz estimates both before and after sorting; for each iteration, the norm of residual for compressed factorization and the symmetric tridiagonal matrix $H$; during re-orthogonalization, the initial/restarted starting vector; during the Lanczos iteration loop, a restart is flagged and the number of the residual requiring iterative refinement; while applying shifts, some indices.

$\geq 40$ If **Monitoring** $> 0$, **Monitoring** is set, then during the Lanczos iteration loop, the Lanczos vector number and norm of the current residual; while applying shifts, key measures of progress and the order of $H$; while computing eigenvalues of $H$, the last rows of the Schur and eigenvector matrices; when computing implicit shifts, the eigenvalues and Ritz estimates of $H$.

$\geq 50$ If **Monitoring** is set, then during Lanczos iteration loop: norms of key components and the active column of $H$, norms of residuals during iterative refinement, the final symmetric tridiagonal matrix $H$; while applying shifts: number of shifts, shift values, block indices, updated tridiagonal matrix $H$; while computing eigenvalues of $H$: the diagonals of $H$, the computed eigenvalues and Ritz estimates.

Note that setting **Print Level** $\geq 30$ can result in very lengthy **Monitoring** output.

**Regular** Default
**Regular Inverse**
**Shifted Inverse**

These options define the computational mode which in turn defines the form of operation OP$(x)$ to be performed.

| | |
|---|---|
| **Regular** | OP $= A$ |
| **Shifted Inverse** | OP $= (A - \sigma I)^{-1}$ where $\sigma$ is real |
| **Regular Inverse** | OP $= A^{-1}$ |

**Tolerance** $r$ Default $= \epsilon$

An approximate eigenvalue has deemed to have converged when the corresponding Ritz estimate is within **Tolerance** relative to the magnitude of the eigenvalue.

---