NAG Toolbox for MATLAB

f12fc

1 Purpose

f12fc is a post-processing function in a suite of functions consisting of f12fa, f12fb, f12fc, f12fd and f12fe, that must be called following a final exit from f12fb.

2 Syntax

```
[nconv, d, z, v, comm, icomm, ifail] = f12fc(sigma, resid, v, comm,
icomm)
```

3 Description

The suite of functions is designed to calculate some of the eigenvalues, λ, (and optionally the corresponding eigenvectors, x) of a standard eigenvalue problem $Ax = \lambda x$, or of a generalized eigenvalue problem $Ax = \lambda Bx$ of order n, where n is large and the coefficient matrices A and B are sparse, real and symmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, real and symmetric problems.

Following a call to f12fb, f12fc returns the converged approximations to eigenvalues and (optionally) the corresponding approximate eigenvectors and/or an orthonormal basis for the associated approximate invariant subspace. The eigenvalues (and eigenvectors) are selected from those of a standard or generalized eigenvalue problem defined by real symmetric matrices. There is negligible additional cost to obtain eigenvectors; an orthonormal basis is always computed, but there is an additional storage cost if both are requested.

f12fc is based on the function **dseupd** from the ARPACK package, which uses the Implicitly Restarted Lanczos iteration method. The method is described in Lehoucq and Sorensen 1996 and Lehoucq 2001 while its use within the ARPACK software is described in great detail in Lehoucq *et al.* 1998. An evaluation of software for computing eigenvalues of sparse symmetric matrices is provided in Lehoucq and Scott 1996. This suite of functions offers the same functionality as the ARPACK software for real symmetric problems, but the interface design is quite different in order to make the option setting clearer to you and to simplify some of the interfaces.

f12fc, is a post-processing function that must be called following a successful final exit from f12fb. f12fc uses data returned from f12fb and options, set either by default or explicitly by calling f12fd, to return the converged approximations to selected eigenvalues and (optionally):

– the corresponding approximate eigenvectors;

– an orthonormal basis for the associated approximate invariant subspace;

– both.

4 References

Lehoucq R B 2001 Implicitly Restarted Arnoldi Methods and Subspace Iteration *SIAM Journal on Matrix Analysis and Applications* **23** 551–562

Lehoucq R B and Scott J A 1996 An evaluation of software for computing eigenvalues of sparse nonsymmetric matrices *Preprint MCS-P547-1195* Argonne National Laboratory

Lehoucq R B and Sorensen D C 1996 Deflation Techniques for an Implicitly Restarted Arnoldi Iteration *SIAM Journal on Matrix Analysis and Applications* **17** 789–821

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: **sigma – double scalar**

If one of the **Shifted** modes has been selected then **sigma** contains the real shift used; otherwise **sigma** is not referenced.

2: **resid**$(*)$ **– double array**

Note: the dimension of the array **resid** must be at least **n** (see f12fa).

Must not be modified following a call to f12fb since it contains data required by f12fc.

3: **v**(**ldv**,$*$) **– double array**

The first dimension of the array **v** must be at least n

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

The **ncv** columns of **v** contain the Lanczos basis vectors for OP as constructed by f12fb.

4: **comm**$(*)$ **– double array**

Note: the dimension of the array **comm** must be at least $\max(1, \mathbf{lcomm})$ (see f12fa).

On initial entry: must remain unchanged from the prior call to f12fa.

5: **icomm**$(*)$ **– int32 array**

Note: the dimension of the array **icomm** must be at least $\max(1, \mathbf{licomm})$ (see f12fa).

On initial entry: must remain unchanged from the prior call to f12fa.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

ldz, ldv

5.4 Output Parameters

1: **nconv – int32 scalar**

The number of converged eigenvalues as found by f12fb.

2: **d**$(*)$ **– double array**

Note: the dimension of the array **d** must be at least **ncv** (see f12ff).

The first **nconv** locations of the array **d** contain the converged approximate eigenvalues.

3: **z**$(\mathbf{n} \times (\mathbf{nev} + \mathbf{1}))$ **– double array**

If the default option **Vectors** = Ritz has been selected then **z** contains the final set of eigenvectors corresponding to the eigenvalues held in **d**. The real eigenvector associated with an eigenvalue is stored in the corresponding column of **z**.

4: **v**(**ldv**,$*$) **– double array**

The first dimension of the array **v** must be at least n

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

If the option **Vectors** = Schur has been set, or the option **Vectors** = Ritz has been set and a separate array **z** has been passed (i.e., **z** does not equal **v**), then the first **nconv** columns of **v** will contain approximate Schur vectors that span the desired invariant subspace.

5: **comm**($*$) – **double array**

Note: the dimension of the array **comm** must be at least $\max(1, \mathbf{lcomm})$ (see f12fa).

Contains data on the current state of the solution.

6: **icomm**($*$) – **int32 array**

Note: the dimension of the array **icomm** must be at least $\max(1, \mathbf{licomm})$ (see f12fa).

Contains data on the current state of the solution.

7: **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 = 1

On entry, **ldz** $<\max(1, \mathbf{n})$ or **ldz** < 1 when no vectors are required.

ifail = 2

On entry, the option **Vectors** = Select was selected, but this is not yet implemented.

ifail = 3

The number of eigenvalues found to sufficient accuracy prior to calling f12fc, as communicated through the parameter **icomm**, is zero.

ifail = 4

The number of converged eigenvalues as calculated by f12fb differ from the value passed to it through the parameter **icomm**.

ifail = 5

Unexpected error during calculation of a tridiagonal form: there was a failure to compute all the converged eigenvalues. Please contact NAG.

ifail = 6

The function was unable to dynamically allocate sufficient internal workspace. Please contact NAG.

ifail = 7

An unexpected error has occurred. Please contact NAG.

7 Accuracy

The relative accuracy of a Ritz value, λ, is considered acceptable if its Ritz estimate $\leq$ **Tolerance** $\times |\lambda|$. The default **Tolerance** used is the ***machine precision*** given by x02aj.

8 Further Comments

None.

9 Example

```
n = int32(100);
nx = int32(10);
nev = int32(4);
ncv = int32(10);

irevcm = int32(0);
resid = zeros(100,1);
v = zeros(100,20);
x = zeros(100,1);
mx = zeros(100,1);

sigma = 0;


% Initialisation Step
[icomm, comm, ifail] = f12fa(n, nev, ncv);

% Set Optional Parameters
[icomm, comm, ifail] = f12fd('SMALLEST MAGNITUDE', icomm, comm);

% Solve
while (irevcm ~= 5)
  [irevcm, resid, v, x, mx, nshift, comm, icomm, ifail] = ...
    f12fb(irevcm, resid, v, x, mx, comm, icomm);
  if (irevcm == 1 || irevcm == -1)
    x = f12f_av(nx, x);
  elseif (irevcm == 4)
    [niter, nconv, ritz, rzest] = f12fe(icomm, comm);
      fprintf('Iteration %d, No. converged = %d, norm of estimates =
%16.8g\n', niter, nconv, norm(rzest(1:nev),2));
  end
end

% Post-process to compute eigenvalues/vectors
[nconv, d, z, v, comm, icomm, ifail] = f12fc(sigma, resid, v, comm,
icomm)
```

```
Iteration 1, No. converged = 0, norm of estimates =       81.010211
Iteration 2, No. converged = 0, norm of estimates =       45.634095
Iteration 3, No. converged = 0, norm of estimates =       42.747772
Iteration 4, No. converged = 0, norm of estimates =       8.6106757
Iteration 5, No. converged = 0, norm of estimates =      0.71330195
Iteration 6, No. converged = 0, norm of estimates =      0.15050738
Iteration 7, No. converged = 0, norm of estimates =     0.015776765
Iteration 8, No. converged = 0, norm of estimates =    0.0038996544
Iteration 9, No. converged = 0, norm of estimates =    0.0004324447
Iteration 10, No. converged = 0, norm of estimates =    0.00011026365
Iteration 11, No. converged = 0, norm of estimates =    1.2358564e-05
Iteration 12, No. converged = 0, norm of estimates =    3.1712516e-06
Iteration 13, No. converged = 1, norm of estimates =    3.5633917e-07
Iteration 14, No. converged = 1, norm of estimates =    4.1370551e-08
Iteration 15, No. converged = 2, norm of estimates =    5.3820859e-09
Iteration 16, No. converged = 1, norm of estimates =     7.557502e-10
Iteration 17, No. converged = 1, norm of estimates =    4.0268792e-11
Iteration 18, No. converged = 2, norm of estimates =    1.4048495e-11
Iteration 19, No. converged = 2, norm of estimates =     1.774664e-10
Iteration 20, No. converged = 2, norm of estimates =    1.1754284e-09
Iteration 21, No. converged = 2, norm of estimates =     7.782271e-09
Iteration 22, No. converged = 2, norm of estimates =    5.2551985e-10
Iteration 23, No. converged = 2, norm of estimates =    1.8326785e-10
Iteration 24, No. converged = 2, norm of estimates =    7.8753868e-11
Iteration 25, No. converged = 2, norm of estimates =    2.2420618e-11
Iteration 26, No. converged = 2, norm of estimates =    8.5727233e-13
Iteration 27, No. converged = 2, norm of estimates =    3.4872081e-13
Iteration 28, No. converged = 2, norm of estimates =    2.6274361e-14
Iteration 29, No. converged = 2, norm of estimates =    2.1845559e-14
```

```
Iteration 30, No. converged = 3, norm of estimates =     8.670653e-16
Iteration 31, No. converged = 3, norm of estimates =    2.5548816e-16
Iteration 32, No. converged = 3, norm of estimates =    4.9338532e-18
nconv =
           4
d =
   19.6054
   48.2193
   48.2193
   76.8333
         0
         0
         0
         0
         0
         0
z =
     array elided
v =
     array elided
comm =
     array elided
icomm =
     array elided
ifail =
           0
```