NAG Toolbox for MATLAB

f12aq

1 Purpose

f12aq is a post-processing function in a suite of functions consisting of f12an, f12ap, f12aq, f12ar and f12as, that must be called following a final exit from f12aq.

2 Syntax

```
[nconv, d, z, v, comm, icomm, ifail] = f12aq(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, complex and nonsymmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, complex and nonsymmetric problems.

Following a call to f12ap, f12aq 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 complex nonsymmetric 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.

f12aq is based on the function **zneupd** from the ARPACK package, which uses the Implicitly Restarted Arnoldi 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 nonsymmetric matrices is provided in Lehoucq and Scott 1996. This suite of functions offers the same functionality as the ARPACK software for complex nonsymmetric 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.

f12aq is a post-processing function that must be called following a successful final exit from f12ap. f12aq uses data returned from f12ap and options set either by default or explicitly by calling f12ar, 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 – complex scalar**

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

2: **resid**$(*)$ **– complex array**

Note: the dimension of the array **resid** must be at least **n** (see f12an).

Must not be modified following a call to f12ap since it contains data required by f12aq.

3: **v**(**ldv**,$*$) **– complex 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 Arnoldi basis vectors for OP as constructed by f12ap.

4: **comm**$(*)$ **– complex array**

Note: the dimension of the array **comm** must be at least $\max(1, 3 \times \mathbf{n} + 3 \times \mathbf{ncv} \times \mathbf{ncv} + 5 \times \mathbf{ncv} + 60)$ (see f12an).

On initial entry: must remain unchanged from the prior call to f12an.

5: **icomm**$(*)$ **– int32 array**

Note: the dimension of the array **icomm** must be at least $\max(1, \mathbf{licomm})$ (see f12an).

On initial entry: must remain unchanged from the prior call to f12an.

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 f12ar.

2: **d**$(*)$ **– complex array**

Note: the dimension of the array **d** must be at least **ncv** (see f12an).

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

3: **z**$(\mathbf{n} \times \mathbf{ncv})$ **– complex 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 complex eigenvector associated with an eigenvalue is stored in the corresponding column of **z**.

4: **v**(**ldv**,$*$) **– complex 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 or 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**$(*)$ **– complex array**

Note: the dimension of the array **comm** must be at least $\max(1, 3 \times \mathbf{n} + 3 \times \mathbf{ncv} \times \mathbf{ncv} + 5 \times \mathbf{ncv} + 60)$ (see f12an).

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 f12an).

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 f12aq, as communicated through the parameter **icomm**, is zero.

ifail = 4

The number of converged eigenvalues as calculated by f12ap differ from the value passed to it through the parameter **icomm**.

ifail = 5

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

ifail = 6

Unexpected error: the computed Schur form could not be reordered by an internal call. Please contact NAG.

ifail = 7

Unexpected error in internal call while calculating eigenvectors. Please contact NAG.

ifail = 8

Either the solver function f12ap has not been called prior to the call of this function or a communication array has become corrupted.

ifail = 9

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

ifail $= 10$

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(20);

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

sigma = complex(0);


% Initialisation Step
[icomm, comm, ifail] = f12an(n, nev, ncv);

% Solve
while (irevcm ~= 5)
  [irevcm, resid, v, x, mx, nshift, comm, icomm, ifail] = ...
    f12ap(irevcm, resid, v, x, mx, comm, icomm);
  if (irevcm == 1 || irevcm == -1)
    x = f12_av(nx, x);
  elseif (irevcm == 4)
    [niter, nconv, ritz, rzest] = f12as(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] = f12aq(sigma, resid, v, comm,
icomm);
% sort to avoid difference in order showing as an error
```

```
Iteration 1, No. converged = 0, norm of estimates =        132.04155
Iteration 2, No. converged = 0, norm of estimates =         102.2167
Iteration 3, No. converged = 0, norm of estimates =         61.46018
Iteration 4, No. converged = 0, norm of estimates =        23.673492
Iteration 5, No. converged = 0, norm of estimates =        6.7565006
Iteration 6, No. converged = 0, norm of estimates =        1.8830144
Iteration 7, No. converged = 0, norm of estimates =       0.48683436
Iteration 8, No. converged = 0, norm of estimates =       0.11166714
Iteration 9, No. converged = 0, norm of estimates =      0.033214754
Iteration 10, No. converged = 0, norm of estimates =     0.0086903151
Iteration 11, No. converged = 0, norm of estimates =     0.0015400799
Iteration 12, No. converged = 0, norm of estimates =    0.00041464548
Iteration 13, No. converged = 0, norm of estimates =    0.00010864867
Iteration 14, No. converged = 0, norm of estimates =    2.6760132e-05
Iteration 15, No. converged = 0, norm of estimates =    6.1411233e-06
```

```
Iteration 16, No. converged = 0, norm of estimates =    1.1174096e-06
Iteration 17, No. converged = 0, norm of estimates =     3.047356e-07
Iteration 18, No. converged = 1, norm of estimates =    5.6326848e-08
Iteration 19, No. converged = 2, norm of estimates =    1.9947734e-08
Iteration 20, No. converged = 2, norm of estimates =     3.935598e-09
Iteration 21, No. converged = 2, norm of estimates =    7.6414746e-10
Iteration 22, No. converged = 2, norm of estimates =    1.3783855e-10
Iteration 23, No. converged = 2, norm of estimates =    1.4348987e-11
Iteration 24, No. converged = 3, norm of estimates =    2.2715713e-12
Iteration 25, No. converged = 3, norm of estimates =    3.9695554e-13
```