NAG Toolbox for MATLAB

f12as

1 Purpose

f12as can be used to return additional monitoring information during computation. It is in a suite of functions consisting of f12an, f12ap, f12aq, f12ar and f12as.

2 Syntax

```
[niter, nconv, ritz, rzest] = f12as(icomm, comm)
```

3 Description

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

On an intermediate exit from f12ap with **irevcm** $= 4$, f12as may be called to return monitoring information on the progress of the Arnoldi iterative process. The information returned by f12as is:

– the number of the current Arnoldi iteration;

– the number of converged eigenvalues at this point;

– the converged eigenvalues;

– the error bounds on the converged eigenvalues.

f12as does not have an equivalent function from the ARPACK package which prints various levels of detail of monitoring information through an output channel controlled via a parameter value (see Lehoucq *et al.* 1998 for details of ARPACK routines). f12as should not be called at any time other than immediately following an **irevcm** $= 4$ return from f12ap.

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: **icomm**$(*)$ – **int32 array**

Note: the dimension of the array **icomm** must be at least $\max(1, \textbf{licomm})$, where **licomm** is passed to the setup function f12an (see f12an).

The array **icomm** output by the preceding call to f12ap.

2: **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).

The array **comm** output by the preceding call to f12ap.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **niter – int32 scalar**

The number of the current Arnoldi iteration.

2: **nconv – int32 scalar**

The number of converged eigenvalues so far.

3: **ritz**$(*)$ **– complex array**

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

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

4: **rzest**$(*)$ **– complex array**

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

The first **nconv** locations of the array **rzest** contain the complex Ritz estimates on the converged approximate eigenvalues.

6 Error Indicators and Warnings

None.

7 Accuracy

A Ritz value, λ, is deemed to have converged if the magnitude of 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
```