NAG Toolbox

nag_sparseig_real_monit (f12ae)

1 Purpose

nag_sparseig_real_monit (f12ae) can be used to return additional monitoring information during computation. It is in a suite of functions consisting of nag_sparseig_real_init (f12aa), nag_sparseig_real_iter (f12ab), nag_sparseig_real_proc (f12ac), nag_sparseig_real_option (f12ad) and nag_sparseig_real_monit (f12ae).

2 Syntax

```
[niter, nconv, ritzr, ritzi, rzest] = nag_sparseig_real_monit(icomm, comm)
[niter, nconv, ritzr, ritzi, rzest] = f12ae(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 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 nonsymmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, real and nonsymmetric problems.

On an intermediate exit from nag_sparseig_real_iter (f12ab) with **irevcm** $= 4$, nag_sparseig_real_monit (f12ae) may be called to return monitoring information on the progress of the Arnoldi iterative process. The information returned by nag_sparseig_real_monit (f12ae) is:

– the number of the current Arnoldi iteration;

– the number of converged eigenvalues at this point;

– the real and imaginary parts of the converged eigenvalues;

– the error bounds on the converged eigenvalues.

nag_sparseig_real_monit (f12ae) 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 argument value (see Lehoucq *et al.* (1998) for details of ARPACK routines). nag_sparseig_real_monit (f12ae) should not be called at any time other than immediately following an **irevcm** $= 4$ return from nag_sparseig_real_iter (f12ab).

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**(:) – INTEGER array

The dimension of the array **icomm** must be at least $\max(1, \mathbf{licomm})$, where **licomm** is passed to the setup function (see nag_sparseig_real_init (f12aa))

The array **icomm** output by the preceding call to nag_sparseig_real_iter (f12ab).

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

The dimension of the array **comm** must be at least $\max(1, 3 \times \mathbf{n} + 3 \times \mathbf{ncv} \times \mathbf{ncv} + 6 \times \mathbf{ncv} + 60)$ (see nag_sparseig_real_init (f12aa))

The array **comm** output by the preceding call to nag_sparseig_real_iter (f12ab).

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **niter** – INTEGER

The number of the current Arnoldi iteration.

2: **nconv** – INTEGER

The number of converged eigenvalues so far.

3: **ritzr**(:) – REAL (KIND=nag_wp) array

The dimension of the array **ritzr** will be **ncv** (see nag_sparseig_real_init (f12aa))

The first **nconv** locations of the array **ritzr** contain the real parts of the converged approximate eigenvalues.

4: **ritzi**(:) – REAL (KIND=nag_wp) array

The dimension of the array **ritzi** will be **ncv** (see nag_sparseig_real_init (f12aa))

The first **nconv** locations of the array **ritzi** contain the imaginary parts of the converged approximate eigenvalues.

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

The dimension of the array **rzest** will be **ncv** (see nag_sparseig_real_init (f12aa))

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

6 Error Indicators and Warnings

None.

7 Accuracy

A Ritz value, λ, is deemed to have converged if its Ritz estimate $\leq$ **Tolerance** $\times |\lambda|$. The default **Tolerance** used is the ***machine precision*** given by nag_machine_precision (x02aj).

8 Further Comments

None.

9 Example

This example solves $Ax = \lambda Bx$ in shifted-real mode, where A is the tridiagonal matrix with 2 on the diagonal, -2 on the subdiagonal and 3 on the superdiagonal. The matrix B is the tridiagonal matrix with 4 on the diagonal and 1 on the off-diagonals. The shift sigma, σ, is a complex number, and the operator used in the shifted-real iterative process is $\text{OP} = \text{real}\big((A - \sigma B)_{-1} B\big)$.

9.1 Program Text

```
     function f12ae_example

fprintf('f12ae example results\n\n');

n   = nag_int(100);
nev = nag_int(4);
ncv = nag_int(20);
sigmar = 0.4;
sigmai = 0.6;
sigma = sigmar + i*sigmai;

% A diagonals
dd = 2;
dl = -2;
du = 3;

irevcm = nag_int(0);
resid = zeros(n,1);
v = zeros(n, ncv);
x = zeros(n, 1);
mx = zeros(n);
y = zeros(n,1);

[icomm, comm, ifail] = f12aa( ...
                              n, nev, ncv);
[icomm, comm, ifail] = f12ad( ...
                              'Regular Inverse', icomm, comm);
[icomm, comm, ifail] = f12ad( ...
                              'Generalized', icomm, comm);

% Form factorise complex tridiagonal C = A - sigma*B
cl(1:n-1,1) = complex(dl) - sigma;
cd(1:n,1)   = complex(dd) - 4*sigma;
cu(1:n-1,1) = complex(du) - sigma;
[cl, cd, cu, cu2, ipiv, info] = f07cr( ...
                                       cl, cd, cu);

while (irevcm ~= 5)
  [irevcm, resid, v, x, mx, nshift, comm, icomm, ifail] = ...
    f12ab( ...
           irevcm, resid, v, x, mx, comm, icomm);
  if (irevcm == -1)
    % Solve (A-sigB)y = Bx, Bx not stored
    x = f12ae_Bx(n,x);
    z = reshape(complex(x),[n,1]);
    [z, info] = f07cs( ...
                       'N', cl, cd, cu, cu2, ipiv, z);
    x = real(z);
  elseif (irevcm == 1)
    % Solve (A-sigB)y = Bx, Bx stored in mx
    z = complex(mx);
    [z, info] = f07cs( ...
                       'N', cl, cd, cu, cu2, ipiv, z);
    x = real(z);
  elseif (irevcm == 2)
```

```
    % y = Bx
    x = f12ae_Bx(n,x);
  elseif (irevcm == 4)
    [niter, nconv, ritzr, ritzi, rzest] = f12ae(icomm, comm);
    if (niter == 1)
      fprintf('\n');
    end
    fprintf('Iteration %2d No. converged = %d Norm of estimates = %10.2e\n', ...
            niter, nconv, norm(rzest));
  end
end

[nconv, dr, di, z, v, comm, icomm, ifail] = ...
f12ac( ...
       sigmar, sigmai, resid, v, comm, icomm);

% Recover eigenvalues of original problem
eig = f12ae_recover(n,nconv,di,v);

fprintf('\nThe %4d Ritz values closest to %7.2f%+7.2fi are:\n\n', ...
        nconv, sigmar, sigmai);
disp(eig');

function [y] = f12ae_Bx(n,x)
  y(1) = 4*x(1) + x(2);
  for j=2:n-1
    y(j) = x(j-1) + 4*x(j) + x(j+1);
  end
  y(n) = x(n-1) + 4*x(n);

function [y] = f12ae_Ax(n,x)
  y(1) = 2*x(1) + 3*x(2);
  for j=2:n-1
    y(j) = -2*x(j-1) + 2*x(j) + 3*x(j+1);
  end
  y(n) = -2*x(n-1) + 2*x(n);

function [eig] = f12ae_recover(n,nconv,di,v)
  first = true;
  for j = 1:nconv
    % Use Rayleigh Quotient to recover eigenvalues of the original problem
    if di(j)==0
      % Ritz value is real. x = v(:,j); eig = x'Ax/x'Bx.
      x = v(:,j);
      ax = f12ae_Ax(n,x);
      bx = f12ae_Bx(n,x);
      eig(j) = dot(x,ax)/dot(x,bx) + 0i;
    elseif (first)
      % Ritz value is complex: x = v(:,j) - i v(:,j+1).
      xr = v(:,j);
      xi = v(:,j+1);
      z = xr + i*xi;
      % Compute x'(Ax):
      az = f12ae_Ax(n,xr) + i*f12ae_Ax(n,xi);
      num = dot(z,az);
      % Compute x'(Bx):
      az = f12ae_Bx(n,xr) + i*f12ae_Bx(n,xi);
      den = dot(z,az);
      eig(j) = num/den;
      first = false;
    else
      % Second of complex conjugate pair.
      eig(j) = conj(eig(j-1));
      first = true;
    end
  end
```

9.2 Program Results

```
     f12ae example results

Iteration  1 No. converged = 0 Norm of estimates =   3.05e+00
Iteration  2 No. converged = 0 Norm of estimates =   2.89e+00
Iteration  3 No. converged = 0 Norm of estimates =   3.74e+00
Iteration  4 No. converged = 0 Norm of estimates =   3.11e+00
Iteration  5 No. converged = 0 Norm of estimates =   3.93e+00
Iteration  6 No. converged = 0 Norm of estimates =   3.21e+00

The    4 Ritz values closest to    0.40  +0.60i are:

   0.5000 - 0.5958i
   0.5000 + 0.5958i
   0.5000 - 0.6331i
   0.5000 + 0.6331i
```