NAG Toolbox

nag_sparseig_complex_band_solve (f12au)

1 Purpose

nag_sparseig_complex_band_solve (f12au) is the main solver function in a suite of functions consisting of nag_sparseig_complex_option (f12ar), nag_sparseig_complex_band_init (f12at) and nag_sparseig_complex_band_solve (f12au). It must be called following an initial call to nag_sparseig_complex_band_init (f12at) and following any calls to nag_sparseig_complex_option (f12ar).

nag_sparseig_complex_band_solve (f12au) returns approximations to selected eigenvalues, and (optionally) the corresponding eigenvectors, of a standard or generalized eigenvalue problem defined by complex banded non-Hermitian matrices. The banded matrix must be stored using the LAPACK storage format for complex banded non-Hermitian matrices.

2 Syntax

```
[nconv, d, z, resid, v, comm, icomm, ifail] = nag_sparseig_complex_band_solve
(kl, ku, ab, mb, sigma, resid, comm, icomm)
[nconv, d, z, resid, v, comm, icomm, ifail] = f12au(kl, ku, ab, mb, sigma,
resid, 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 banded, complex and non-Hermitian.

Following a call to the initialization function nag_sparseig_complex_band_init (f12at), nag_sparseig_complex_band_solve (f12au) returns the converged approximations to eigenvalues and (optionally) the corresponding approximate eigenvectors and/or a unitary 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 banded non-Hermitian matrices. There is negligible additional computational cost to obtain eigenvectors; a unitary basis is always computed, but there is an additional storage cost if both are requested.

The banded matrices A and B must be stored using the LAPACK column ordered storage format for banded non-Hermitian matrices; please refer to Section 3.2.4 in the F07 Chapter Introduction for details on this storage format.

nag_sparseig_complex_band_solve (f12au) is based on the banded driver functions **znbdr1** to **znbdr4** 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 non-Hermitian matrices is provided in Lehoucq and Scott (1996). This suite of functions offers the same functionality as the ARPACK banded driver software for complex non-Hermitian problems, but the interface design is quite different in order to make the option setting clearer and to combine the different drivers into a general purpose function.

nag_sparseig_complex_band_solve (f12au), is a general purpose function that must be called following initialization by nag_sparseig_complex_band_init (f12at). nag_sparseig_complex_band_solve (f12au) uses options, set either by default or explicitly by calling nag_sparseig_complex_option (f12ar), to return the converged approximations to selected eigenvalues and (optionally):

– the corresponding approximate eigenvectors;

– a unitary 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

Note: in the following description **n**, **nev** and **ncv** appears. In every case they should be interpretted as the value associated with the identically named argument in a prior call to nag_sparseig_complex_band_init (f12at).

5.1 Compulsory Input Parameters

1: **kl** – INTEGER

The number of subdiagonals of the matrices A and B.

Constraint: **kl** ≥ 0.

2: **ku** – INTEGER

The number of superdiagonals of the matrices A and B.

Constraint: **ku** ≥ 0.

3: **ab**$(ldab,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **ab** must be at least $2 \times \mathbf{kl} + \mathbf{ku} + 1$.

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

Must contain the matrix A in LAPACK banded storage format for non-Hermitian matrices (see Section 3.2.4 in the F07 Chapter Introduction).

4: **mb**$(ldmb,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension of the array **mb** must be at least $2 \times \mathbf{kl} + \mathbf{ku} + 1$.

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

Must contain the matrix B in LAPACK banded storage format for non-Hermitian matrices (see Section 3.2.4 in the F07 Chapter Introduction).

5: **sigma** – COMPLEX (KIND=nag_wp)

If the **Shifted Inverse** mode (see nag_sparseig_complex_option (f12ar)) has been selected then **sigma** must contain the shift used; otherwise **sigma** is not referenced. Section 4.2 in the F12 Chapter Introduction describes the use of shift and invert transformations.

6: **resid**$(\mathbf{n})$ – COMPLEX (KIND=nag_wp) array

Need not be set unless the option **Initial Residual** has been set in a prior call to nag_sparseig_complex_option (f12ar) in which case **resid** must contain an initial residual vector.

7: **comm**(**60**) – COMPLEX (KIND=nag_wp) array

Must remain unchanged from the prior call to nag_sparseig_complex_option (f12ar) and nag_sparseig_complex_band_init (f12at).

8: **icomm**(**140**) – INTEGER array

Must remain unchanged from the prior call to nag_sparseig_complex_option (f12ar) and nag_sparseig_complex_band_init (f12at).

5.2 Optional Input Parameters

None.

5.3 Output Parameters

1: **nconv** – INTEGER

The number of converged eigenvalues.

2: **d**(**nev**) – COMPLEX (KIND=nag_wp) array

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

3: **z**(ldz, :) – COMPLEX (KIND=nag_wp) array

The first dimension, ldz, of the array **z** will be

if the default option **Vectors** = Ritz has been selected, $ldz = \mathbf{n}$;
if the option **Vectors** = None or Schur has been selected, $ldz = 1$.

The second dimension of the array **z** will be $\mathbf{n} \times (\mathbf{nev} + 1)$.

If the default option **Vectors** = RITZ (see nag_sparseig_complex_option (f12ar)) has been selected then **z** contains the final set of eigenvectors corresponding to the eigenvalues held in **d**, otherwise **z** is not referenced. The complex eigenvector associated with an eigenvalue **d**(j) is stored in the corresponding array section of **z**, namely **z**(i, j), for $i = 1, 2, \ldots, \mathbf{n}$ and $j = 1, 2, \ldots, \mathbf{nconv}$.

4: **resid**(**n**) – COMPLEX (KIND=nag_wp) array

Contains the final residual vector. This can be used as the starting residual to improve convergence on the solution of a closely related eigenproblem. This has no relation to the error residual $Ax - \lambda x$ or $Ax - \lambda Bx$.

5: **v**(ldv, :) – COMPLEX (KIND=nag_wp) array

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

The second dimension of the array **v** will be $\max(1, \mathbf{n} \times \mathbf{ncv})$.

If the option **Vectors** = SCHUR or RITZ (see nag_sparseig_complex_option (f12ar)) 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.

The jth Schur vector is stored in the ith column of **v**.

6: **comm**(**60**) – COMPLEX (KIND=nag_wp) array

Communication array, used to store information between calls to nag_sparseig_complex_band_solve (f12au).

7: **icomm**(**140**) – INTEGER array

Communication array, used to store information between calls to nag_sparseig_complex_band_solve (f12au).

8: **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: **kl** ≥ 0.

ifail $= 2$

Constraint: **ku** ≥ 0.

ifail $= 3$

Constraint: $ldab \geq 2 \times \mathbf{kl} + \mathbf{ku} + 1$.

ifail $= 5$

The maximum number of iterations ≤ 0, the option **Iteration Limit** has been set.

ifail $= 6$

The options **Generalized** and **Regular** are incompatible.

ifail $= 7$

The option **Initial Residual** was selected but the starting vector held in **resid** is zero.

ifail $= 8$

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

ifail $= 9$

Constraint: $ldz \geq \mathbf{n}$.

ifail $= 10$

On entry, **Vectors** = Select, but this is not yet implemented.

ifail $= 11$

The number of eigenvalues found to sufficient accuracy is zero.

ifail $= 12$

Could not build an Arnoldi factorization.

ifail $= 13$

Error in internal call to compute eigenvalues and corresponding error bounds of the current upper Hessenberg matrix. Please contact NAG.

ifail $= 14$

During calculation of a Schur form, there was a failure to compute a number of eigenvalues Please contact NAG.

ifail $= 15$

The computed Schur form could not be reordered by an internal call. Please contact NAG.

ifail $= 16$

Error in internal call to compute eigenvectors. Please contact NAG.

ifail $= 17$

Failure during internal factorization of real banded matrix. Please contact NAG.

ifail $= 18$

Failure during internal solution of real banded matrix. Please contact NAG.

ifail $= 19$

Failure during internal factorization of complex banded matrix. Please contact NAG.

ifail $= 20$

Failure during internal solution of complex banded matrix. Please contact NAG.

ifail $= 21$

The maximum number of iterations has been reached.

ifail $= 22$

No shifts could be applied during a cycle of the implicitly restarted Arnoldi iteration.

ifail $= 23$

Overflow occurred during transformation of Ritz values to those of the original problem.

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, λ, is considered acceptable 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 constructs the matrices A and B using LAPACK band storage format and solves $Ax = \lambda Bx$ in shifted inverse mode using the complex shift σ.

9.1 Program Text

```
function f12au_example

fprintf('f12au example results\n\n');

nx = nag_int(10);
n = nx*nx;
nev = nag_int(4);
ncv = nag_int(10);
sigma = 0.4 + 0.6i;

% Construct the matrix A in banded form and store in AB.
% KU, KL are number of superdiagonals and subdiagonals within
% the band of matrices A and M.
kl = nx;
ku = nx;

% Construct ab and mb
ab = zeros(2*kl+ku+1,n);
mb = zeros(2*kl+ku+1,n);

% Main diagonal of A.
idiag = kl + ku + 1;
ab(idiag,1:n) = 4;
mb(idiag,1:n) = ab(idiag,1);

% First subdiagonal and superdiagonal of A.
isup = kl + ku;
isub = kl + ku + 2;
rho = 100;
h = 1/double(nx+1);
for i=1:nx
  lo = (i-1)*nx;
  for j=lo+1:lo+nx-1
    ab(isub,j+1) = -1 + 0.5*h*rho;
    ab(isup,j) = -1 - 0.5*h*rho;
  end
end
mb(isub,2:n)   = 1;
mb(isup,1:n-1) = 1;

% kl-th subdiagonal and ku-th super-diagonal.
isup = kl + 1;
isub = 2*kl + ku + 1;
for i = 1:nx - 1
  lo = (i-1)*nx;
  for j = lo + 1:lo + nx
    ab(isup,nx+j) = -1;
    ab(isub,j) = -1;
  end
end

resid = complex(zeros(100,1));

[icomm, comm, ifail] = f12at( ...
                              n, nev, ncv);
[icomm, comm, ifail] = f12ar( ...
                              'SHIFTED INVERSE', icomm, comm);
[icomm, comm, ifail] = f12ar( ...
                              'GENERALIZED', icomm, comm);

% Find eigenvalues closest in value to SIGMA and corresponding eigenvectors
[nconv, d, z, resid, v, comm, icomm, ifail] = ...
```

```
    f12au( ...
           kl, ku, complex(ab), complex(mb), sigma, resid, comm, icomm);

fprintf('\nRitz values closest to sigma:\n');
disp(d);
```

9.2 Program Results

```
     f12au example results

Ritz values closest to sigma:
   0.3610 + 0.7223i
   0.4598 + 0.7199i
   0.2868 + 0.7241i
   0.2410 + 0.7257i
```