NAG Toolbox

nag_eigen_real_gen_partialsvd (f02wg)

1 Purpose

nag_eigen_real_gen_partialsvd (f02wg) returns leading terms in the singular value decomposition (SVD) of a real general matrix and computes the corresponding left and right singular vectors.

2 Syntax

```
[nconv, sigma, u, v, resid, user, ifail] = nag_eigen_real_gen_partialsvd(m, n, k, ncv, av, 'user', user)
[nconv, sigma, u, v, resid, user, ifail] = f02wg(m, n, k, ncv, av, 'user', user)
```

3 Description

nag_eigen_real_gen_partialsvd (f02wg) computes a few, k, of the largest singular values and corresponding vectors of an m by n matrix A. The value of k should be small relative to m and n, for example $k \sim O(\min(m,n))$. The full singular value decomposition (SVD) of an m by n matrix A is given by

$$A = U\Sigma V^{\mathrm{T}},$$

where U and V are orthogonal and Σ is an m by n diagonal matrix with real diagonal elements, σ_i, such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min(m,n)} \geq 0.$$

The σ_i are the *singular values* of A and the first $\min(m,n)$ columns of U and V are the *left* and *right singular vectors* of A.

If U_k, V_k denote the leading k columns of U and V respectively, and if Σ_k denotes the leading principal submatrix of Σ, then

$$A_k \equiv U_k \Sigma_k {V^{\mathrm{T}}}_k$$

is the best rank-k approximation to A in both the 2-norm and the Frobenius norm.

The singular values and singular vectors satisfy

$$Av_i = \sigma_i u_i \quad \text{and} \quad A^{\mathrm{T}}u_i = \sigma_i v_i \quad \text{so that} \quad A^{\mathrm{T}}A\nu_i = \sigma_i^2 \nu_i \text{ and } AA^{\mathrm{T}}u_i = \sigma_i^2 u_i,$$

where u_i and v_i are the ith columns of U_k and V_k respectively.

Thus, for $m \geq n$, the largest singular values and corresponding right singular vectors are computed by finding eigenvalues and eigenvectors for the symmetric matrix $A^{\mathrm{T}}A$. For $m < n$, the largest singular values and corresponding left singular vectors are computed by finding eigenvalues and eigenvectors for the symmetric matrix AA^{T}. These eigenvalues and eigenvectors are found using functions from Chapter F12. You should read the F12 Chapter Introduction for full details of the method used here.

The real matrix A is not explicitly supplied to nag_eigen_real_gen_partialsvd (f02wg). Instead, you are required to supply a function, **av**, that must calculate one of the requested matrix-vector products Ax or $A^{\mathrm{T}}x$ for a given real vector x (of length n or m respectively).

4 References

Wilkinson J H (1978) Singular Value Decomposition – Basic Aspects *Numerical Software – Needs and Availability* (ed D A H Jacobs) Academic Press

5 Parameters

5.1 Compulsory Input Parameters

1: **m** – INTEGER

m, the number of rows of the matrix A.

Constraint: $\mathbf{m} \geq 0$.

If $\mathbf{m} = 0$, an immediate return is effected.

2: **n** – INTEGER

n, the number of columns of the matrix A.

Constraint: $\mathbf{n} \geq 0$.

If $\mathbf{n} = 0$, an immediate return is effected.

3: **k** – INTEGER

k, the number of singular values to be computed.

Constraint: $0 < \mathbf{k} < \min(\mathbf{m}, \mathbf{n}) - 1$.

4: **ncv** – INTEGER

The dimension of the arrays **sigma** and **resid** and the second dimension of the arrays **u** and **v**. this is the number of Lanczos basis vectors to use during the computation of the largest eigenvalues of $A^{\mathrm{T}}A$ ($m \geq n$) or AA^{T} ($m < n$).

At present there is no *a priori* analysis to guide the selection of **ncv** relative to **k**. However, it is recommended that $\mathbf{ncv} \geq 2 \times \mathbf{k} + 1$. If many problems of the same type are to be solved, you should experiment with varying **ncv** while keeping **k** fixed for a given test problem. This will usually decrease the required number of matrix-vector operations but it also increases the internal storage required to maintain the orthogonal basis vectors. The optimal 'cross-over' with respect to CPU time is problem dependent and must be determined empirically.

Constraint: $\mathbf{k} < \mathbf{ncv} \leq \min(\mathbf{m}, \mathbf{n})$.

5: **av** – SUBROUTINE, supplied by the user.

av must return the vector result of the matrix-vector product Ax or $A^{\mathrm{T}}x$, as indicated by the input value of **iflag**, for the given vector x.

av is called from nag_eigen_real_gen_partialsvd (f02wg) with the argument as supplied to nag_eigen_real_gen_partialsvd (f02wg). You are free to use these arrays to supply information to **av**.

```
[iflag, ax, user] = av(iflag, m, n, x, user)
```

Input Parameters

1: **iflag** – INTEGER

If $\mathbf{iflag} = 1$, **ax** must return the m-vector result of the matrix-vector product Ax.

If $\mathbf{iflag} = 2$, **ax** must return the n-vector result of the matrix-vector product $A^{\mathrm{T}}x$.

2: **m** – INTEGER

The number of rows of the matrix A.

3: **n** – INTEGER

The number of columns of the matrix A.

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

The vector to be pre-multiplied by the matrix A or A^{T}.

5: **user** – INTEGER array

av is called from nag_eigen_real_gen_partialsvd (f02wg) with the object supplied to nag_eigen_real_gen_partialsvd (f02wg).

Output Parameters

1: **iflag** – INTEGER

May be used as a flag to indicate a failure in the computation of Ax or $A^{\mathrm{T}}x$. If **iflag** is negative on exit from **av**, nag_eigen_real_gen_partialsvd (f02wg) will exit immediately with **ifail** set to **iflag**.

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

If **iflag** $= 1$, contains the m-vector result of the matrix-vector product Ax.

If **iflag** $= 2$, contains the n-vector result of the matrix-vector product $A^{\mathrm{T}}x$.

3: **user** – INTEGER array

5.2 Optional Input Parameters

1: **user** – INTEGER array

user is not used by nag_eigen_real_gen_partialsvd (f02wg), but is passed to **av**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

5.3 Output Parameters

1: **nconv** – INTEGER

The number of converged singular values found.

2: **sigma**(**ncv**) – REAL (KIND=nag_wp) array

The **nconv** converged singular values are stored in the first **nconv** elements of **sigma**.

3: **u**(ldu, **ncv**) – REAL (KIND=nag_wp) array

The left singular vectors corresponding to the singular values stored in **sigma**.

The ith element of the jth left singular vector u_j is stored in **u**(i,j), for $i = 1,2,\ldots,m$ and $j = 1,2,\ldots,$ **nconv**.

4: **v**(ldv, **ncv**) – REAL (KIND=nag_wp) array

The right singular vectors corresponding to the singular values stored in **sigma**.

The ith element of the jth right singular vector v_j is stored in **v**(i,j), for $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,$ **nconv**.

5: **resid**(**ncv**) – REAL (KIND=nag_wp) array

The residual $\left\|Av_j - \sigma_j u_j\right\|$, for $m \geq n$, or $\left\|A^{\mathrm{T}}u_j - \sigma_j v_j\right\|$, for $m < n$, for each of the converged singular values σ_j and corresponding left and right singular vectors u_j and v_j.

6: **user** – INTEGER array

7: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

nag_eigen_real_gen_partialsvd (f02wg) returns with **ifail** $= 0$ if at least k singular values have converged and the corresponding left and right singular vectors have been computed.

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

Constraint: **m** ≥ 0.

ifail $= 2$

Constraint: **n** ≥ 0.

ifail $= 3$

Constraint: **k** > 0.

ifail $= 4$

Constraint: **k** $<$ **ncv** $\leq \min(\mathbf{m}, \mathbf{n})$.

ifail $= 5$

Constraint: $ldu \geq$ **m**.

ifail $= 6$

Constraint: $ldv \geq$ **n**.

ifail $= 8$

The maximum number of iterations has been reached.

ifail $= 9$

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

ifail $= 10$

Could not build a full Lanczos factorization.

ifail $= 11$

The number of eigenvalues found to sufficient accuracy is zero.

ifail $= 20$

An error occurred during an internal call. Consider increasing the size of **ncv** relative to **k**.

ifail < 0

On output from user-defined function **av**, **iflag** was set to a negative value, .

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

See Section 2.14.2 in the F08 Chapter Introduction.

8 Further Comments

None.

9 Example

This example finds the four largest singular values (σ) and corresponding right and left singular vectors for the matrix A, where A is the m by n real matrix derived from the simplest finite difference discretization of the two-dimensional kernal $k(s,t)dt$ where

$$k(s,t) = \begin{cases} s(t-1) & \text{if } 0 \leq s \leq t \leq 1 \\ t(s-1) & \text{if } 0 \leq t < s \leq 1 \end{cases}.$$

9.1 Program Text

```
function f02wg_example

fprintf('f02wg example results\n\n');

m = nag_int(100);
n = nag_int(500);
k = nag_int(4);
ncv = nag_int(10);

[nconv, sigma, u, v, resid, user, ifail] = ...
f02wg(m, n, k, ncv, @av);

res = [sigma(1:nconv) resid(1:nconv)];
fprintf('  Singular Value   Residual\n');
fprintf('   %10.5f  %12.2e\n',res');

function [iflag, ax, user] = av(iflag, m, n, x, user)
  if (iflag == 1)
    ax = zeros(m, 1);
  else
    ax = zeros(n, 1);
  end

  % Computes  w <- A*x or w <- Trans(A)*x.
  h = 1/(double(m)+1);
  k = 1/(double(n)+1);
  if (iflag == 1)
    t = 0;
    for j = 1:n
      t = t + k;
      s = 0;
      ktx = k*(t-1)*x(j);
      for i = 1:min(j,m)
        s = s + h;
        ax(i) = ax(i) + ktx*s;
```

```
      end
      ktx = k*t*x(j);
      for i = j+1:m
        s = s + h;
        ax(i) = ax(i) + ktx*(s-1);
      end
    end
  else
    t = 0;
    for j = 1:n
      t = t + k;
      s = 0;
      for i = 1:min(j,m)
        s = s + h;
        ax(j) = ax(j) + k*s*(t-1)*x(i);
      end
      for i = j+1:m
        s = s + h;
        ax(j) = ax(j) + k*t*(s-1)*x(i);
      end
    end
  end
```

9.2 Program Results

```
    f02wg example results

  Singular Value   Residual
      0.00830      1.82e-18
      0.01223      3.95e-18
      0.02381      1.95e-17
      0.11274      2.53e-17
```