NAG Library Routine Document

F08KJF (DGESVJ)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F08KJF (DGESVJ) computes the one-sided Jacobi singular value decomposition (SVD) of a real m by n matrix A, $m \geq n$, with fast scaled rotations and de Rijk's pivoting, optionally computing the left and/or right singular vectors. For $m < n$, the routines F08KBF (DGESVD) or F08KDF (DGESDD) may be used.

2 Specification

```
SUBROUTINE F08KJF (JOBA, JOBU, JOBV, M, N, A, LDA, SVA, MV, V, LDV,        &
                   WORK, LWORK, INFO)

INTEGER             M, N, LDA, MV, LDV, LWORK, INFO
REAL (KIND=nag_wp) A(LDA,*), SVA(N), V(LDV,*), WORK(LWORK)
CHARACTER(1)        JOBA, JOBU, JOBV
```

The routine may be called by its LAPACK name ***dgesvj***.

3 Description

The SVD is written as

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

where Σ is an n by n diagonal matrix, U is an m by n orthonormal matrix, and V is an n by n orthogonal matrix. The diagonal elements of Σ are the singular values of A in descending order of magnitude. The columns of U and V are the left and the right singular vectors of A.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia http://www.netlib.org/lapack/lug

Drmac Z and Veselic K (2008a) New fast and accurate Jacobi SVD algorithm I *SIAM J. Matrix Anal. Appl.* **29 4**

Drmac Z and Veselic K (2008b) New fast and accurate Jacobi SVD algorithm II *SIAM J. Matrix Anal. Appl.* **29 4**

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

1: JOBA – CHARACTER(1) *Input*

On entry: specifies the structure of matrix A.

JOBA = 'L'
The input matrix A is lower triangular.

JOBA = 'U'
The input matrix A is upper triangular.

JOBA = 'G'
The input matrix A is a general m by n matrix, M $\geq$ N.

Constraint: JOBA = 'L', 'U' or 'G'.

2: JOBU – CHARACTER(1) *Input*

On entry: specifies whether to compute the left singular vectors and if so whether you want to control their numerical orthogonality threshold.

JOBU = 'U'
The left singular vectors corresponding to the nonzero singular values are computed and returned in the leading columns of A. See more details in the description of A. The numerical orthogonality threshold is set to approximately $tol = ctol \times \epsilon$, where ϵ is the ***machine precision*** and $ctol = \sqrt{m}$.

JOBU = 'C'
Analogous to JOBU = 'U', except that you can control the level of numerical orthogonality of the computed left singular vectors. The orthogonality threshold is set to $tol = ctol \times \epsilon$, where $ctol$ is given on input in WORK(1). The option JOBU = 'C' can be used if $m \times \epsilon$ is a satisfactory orthogonality of the computed left singular vectors, so $ctol$ = M could save a few sweeps of Jacobi rotations. See the descriptions of A and WORK(1).

JOBU = 'N'
The matrix U is not computed. However, see the description of A.

Constraint: JOBU = 'U', 'C' or 'N'.

3: JOBV – CHARACTER(1) *Input*

On entry: specifies whether and how to compute the right singular vectors.

JOBV = 'V'
The matrix V is computed and returned in the array V.

JOBV = 'A'
The Jacobi rotations are applied to the leading m_v by n part of the array V. In other words, the right singular vector matrix V is not computed explicitly, instead it is applied to an m_v by n matrix initially stored in the first MV rows of V.

JOBV = 'N'
The matrix V is not computed and the array V is not referenced.

Constraint: JOBV = 'V', 'A' or 'N'.

4: M – INTEGER *Input*

On entry: m, the number of rows of the matrix A.

Constraint: M $\geq$ 0.

5: N – INTEGER *Input*

On entry: n, the number of columns of the matrix A.

Constraint: M $\geq$ N $\geq$ 0.

6: A(LDA, $*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array A must be at least max(1, N).

On entry: the m by n matrix A.

On exit: the matrix U containing the left singular vectors of A.

If JOBU = 'U' or 'C'

if INFO = 0

rank(A) orthonormal columns of U are returned in the leading rank(A) columns of the array A. Here rank$(A) \leq$ N is the number of computed singular values of A that are above the safe range parameter, as returned by X02AMF. The singular vectors corresponding to underflowed or zero singular values are not computed. The value of rank(A) is returned by rounding WORK(2) to the nearest whole number. Also see the descriptions of SVA and WORK. The computed columns of U are mutually numerically orthogonal up to approximately $tol = \sqrt{m} \times \epsilon$; or $tol = ctol \times \epsilon$ (JOBU = 'C'), where ϵ is the ***machine precision*** and $ctol$ is supplied on entry in WORK(1), see the description of JOBU.

If INFO > 0

F08KJF (DGESVJ) did not converge in 30 iterations (sweeps). In this case, the computed columns of U may not be orthogonal up to tol. The output U (stored in A), Σ (given by the computed singular values in SVA) and V is still a decomposition of the input matrix A in the sense that the residual $\|A - \alpha \times U \times \Sigma \times V^{\mathrm{T}}\|_2 / \|A\|_2$ is small, where α is the value returned in WORK(1).

If JOBU = 'N'

if INFO = 0

Note that the left singular vectors are 'for free' in the one-sided Jacobi SVD algorithm. However, if only the singular values are needed, the level of numerical orthogonality of U is not an issue and iterations are stopped when the columns of the iterated matrix are numerically orthogonal up to approximately $m \times \epsilon$. Thus, on exit, A contains the columns of U scaled with the corresponding singular values.

If INFO > 0

F08KJF (DGESVJ) did not converge in 30 iterations (sweeps).

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08KJF (DGESVJ) is called.

Constraint: LDA $\geq \max(1, \mathrm{M})$.

8: SVA(N) – REAL (KIND=nag_wp) array *Output*

On exit: the, possibly scaled, singular values of A.

If INFO = 0

The singular values of A are $\sigma_i = \alpha \mathrm{SVA}(i)$, for $i = 1, 2, \ldots, n$, where α is the scale factor stored in WORK(1). Normally $\alpha = 1$, however, if some of the singular values of A might underflow or overflow, then $\alpha \neq 1$ and the scale factor needs to be applied to obtain the singular values.

If INFO > 0

F08KJF (DGESVJ) did not converge in 30 iterations and $\alpha \times$ SVA may not be accurate.

9: MV – INTEGER *Input*

On entry: if JOBV = 'A', the product of Jacobi rotations is applied to the first m_v rows of V.

If JOBV $\neq$ 'A', MV is ignored. See the description of JOBV.

10: V(LDV, $*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array V must be at least max(1, N) if JOBV = 'V' or 'A', and at least 1 otherwise.

On entry: if JOBV = 'A', V must contain an m_v by n matrix to be premultiplied by the matrix V of right singular vectors.

On exit: the right singular vectors of A.

If JOBV = 'V', V contains the n by n matrix of the right singular vectors.

If JOBV = 'A', V contains the product of the computed right singular vector matrix and the initial matrix in the array V.

If JOBV = 'N', V is not referenced.

11: LDV – INTEGER *Input*

On entry: the first dimension of the array V as declared in the (sub)program from which F08KJF (DGESVJ) is called.

Constraints:

if JOBV = 'V', LDV $\geq$ max(1, N);
if JOBV = 'A', LDV $\geq$ max(1, MV);
otherwise LDV $\geq$ 1.

12: WORK(LWORK) – REAL (KIND=nag_wp) array *Communication Array*

On entry: if JOBU = 'C', WORK(1) = $ctol$, where $ctol$ defines the threshold for convergence. The process stops if all columns of A are mutually orthogonal up to $ctol \times \epsilon$. It is required that $ctol \geq 1$, i.e., it is not possible to force the routine to obtain orthogonality below ϵ. $ctol$ greater than $1/\epsilon$ is meaningless, where ϵ is the ***machine precision***.

On exit: contains information about the completed job.

WORK(1)
the scaling factor, α, such that $\sigma_i = \alpha \mathrm{SVA}(i)$, for $i = 1, 2, \ldots, n$ are the computed singular values of A. (See description of SVA.)

WORK(2)
nint(WORK(2))gives the number of the computed nonzero singular values.

WORK(3)
nint(WORK(3)) gives the number of the computed singular values that are larger than the underflow threshold.

WORK(4)
nint(WORK(4)) gives the number of iterations (sweeps of Jacobi rotations) needed for numerical convergence.

WORK(5)
$\max_{i \neq j} |\cos(A(:, i), A(:, j))|$ in the last iteration (sweep). This is useful information in cases when F08KJF (DGESVJ) did not converge, as it can be used to estimate whether the output is still useful and for subsequent analysis.

WORK(6)
The largest absolute value over all sines of the Jacobi rotation angles in the last sweep. It can be useful for subsequent analysis.

Constraint: if JOBU = 'C', WORK(1) $\geq$ 1.0.

13: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08KJF (DGESVJ) is called.

Constraint: LWORK $\geq \max(6, \mathrm{M} + \mathrm{N})$.

14: INFO – INTEGER *Output*

On exit: INFO $= 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

INFO < 0

If INFO $= -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0

F08KJF (DGESVJ) did not converge in the allowed number of iterations (30), but its output might still be useful.

7 Accuracy

The computed singular value decomposition is nearly the exact singular value decomposition for a nearby matrix $(A + E)$, where

$$\|E\|_2 = O(\epsilon)\|A\|_2,$$

and ϵ is the ***machine precision***. In addition, the computed singular vectors are nearly orthogonal to working precision. See Section 4.9 of Anderson *et al.* (1999) for further details.

See Section 6 of Drmac and Veselic (2008a) for a detailed discussion of the accuracy of the computed SVD.

8 Parallelism and Performance

F08KJF (DGESVJ) is not threaded by NAG in any implementation.

F08KJF (DGESVJ) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

This SVD algorithm is numerically superior to the bidiagonalization based QR algorithm implemented by F08KBF (DGESVD) and the divide and conquer algorithm implemented by F08KDF (DGESDD) algorithms and is considerably faster than previous implementations of the (equally accurate) Jacobi SVD method. Moreover, this algorithm can compute the SVD faster than F08KBF (DGESVD) and not much slower than F08KDF (DGESDD). See Section 3.3 of Drmac and Veselic (2008b) for the details.

10 Example

This example finds the singular values and left and right singular vectors of the 6 by 4 matrix

$$A = \begin{pmatrix} 2.27 & -1.54 & 1.15 & -1.94 \\ 0.28 & -1.67 & 0.94 & -0.78 \\ -0.48 & -3.09 & 0.99 & -0.21 \\ 1.07 & 1.22 & 0.79 & 0.63 \\ -2.35 & 2.93 & -1.45 & 2.30 \\ 0.62 & -7.39 & 1.03 & -2.57 \end{pmatrix},$$

together with approximate error bounds for the computed singular values and vectors.

10.1 Program Text

```
    Program f08kjfe

!     F08KJF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: ddisna, dgesvj, nag_wp, x02ajf, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: eps, serrbd
      Integer                          :: i, ifail, info, j, lda, ldv, lwork,  &
                                          m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), rcondu(:), rcondv(:), s(:),  &
                                          v(:,:), work(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: abs
!     .. Executable Statements ..
      Write (nout,*) 'F08KJF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n
      lda = m
      ldv = n
      lwork = n + m
      Allocate (a(lda,n),rcondu(m),rcondv(m),s(n),v(ldv,n),work(lwork))

!     Read the m by n matrix A from data file

      Read (nin,*)((a(i,j),j=1,n),i=1,m)

!     Compute the singular values and left and right singular vectors
!     of A (A = U*S*V, m.ge.n)

!     The NAG name equivalent of dgesvj is f08kjf
      Call dgesvj('G','U','V',m,n,a,lda,s,0,v,ldv,work,lwork,info)

      If (info==0) Then

!       Compute the approximate error bound for the computed singular values
!       using the 2-norm, s(1) = norm(A), and machine precision, eps.
        eps = x02ajf()
        serrbd = eps*s(1)

!       Print solution
        Write (nout,*) 'Singular values'
        Write (nout,99999)(s(j),j=1,n)
```

```
        If (abs(work(1)-1.0_nag_wp)>eps) Then
          Write (nout,99996) 'Values need scaling by factor = ', work(1)
        End If
        Flush (nout)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04caf('General',' ',m,n,a,lda,'Left singular vectors',ifail)

        Write (nout,*)
        Flush (nout)

        ifail = 0
        Call x04caf('General',' ',n,n,v,ldv,'Right singular vectors',ifail)


!       Call DDISNA (F08FLF) to estimate reciprocal condition
!       numbers for the singular vectors
        Call ddisna('Left',m,n,s,rcondu,info)
        Call ddisna('Right',m,n,s,rcondv,info)

!       Print the approximate error bounds for the singular values
!       and vectors
        Write (nout,'(/1X,A)') 'Error estimate for the singular values'
        Write (nout,99998) serrbd
        Write (nout,'(/1X,A)') 'Error estimates for left singular vectors'
        Write (nout,99998)(serrbd/rcondu(i),i=1,n)
        Write (nout,'(/1X,A)') 'Error estimates for right singular vectors'
        Write (nout,99998)(serrbd/rcondv(i),i=1,n)
      Else
        Write (nout,99997) 'Failure in DGESVJ. INFO =', info
      End If


99999 Format (3X,(8F8.4))
99998 Format (4X,1P,6E11.1)
99997 Format (1X,A,I4)
99996 Format (/1X,A,1P,E13.5)
    End Program f08kjfe
```

10.2 Program Data

```
F08KJF Example Program Data

   6      4                   :Values of M and N

   2.27  -1.54   1.15  -1.94
   0.28  -1.67   0.94  -0.78
  -0.48  -3.09   0.99  -0.21
   1.07   1.22   0.79   0.63
  -2.35   2.93  -1.45   2.30
   0.62  -7.39   1.03  -2.57  :End of matrix A
```

10.3 Program Results

```
 F08KJF Example Program Results

 Singular values
     9.9966  3.6831  1.3569  0.5000
 Left singular vectors
          1       2       3       4
 1  -0.2774  0.6003 -0.1277  0.1323
 2  -0.2020  0.0301  0.2805  0.7034
 3  -0.2918 -0.3348  0.6453  0.1906
 4   0.0938  0.3699  0.6781 -0.5399
 5   0.4213 -0.5266  0.0413 -0.0575
 6  -0.7816 -0.3353 -0.1645 -0.3957

 Right singular vectors
```

```
          1       2       3       4
 1  -0.1921  0.8030  0.0041 -0.5642
 2   0.8794  0.3926 -0.0752  0.2587
 3  -0.2140  0.2980  0.7827  0.5027
 4   0.3795 -0.3351  0.6178 -0.6017

 Error estimate for the singular values
        1.1E-15

 Error estimates for left singular vectors
        1.8E-16    4.8E-16    1.3E-15    2.2E-15

 Error estimates for right singular vectors
        1.8E-16    4.8E-16    1.3E-15    1.3E-15
```