

NAG Library Routine Document

F08KQF (ZGELSD)

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

F08KQF (ZGELSD) computes the minimum norm solution to a complex linear least squares problem

$$\min_x \|b - Ax\|_2.$$

2 Specification

```
SUBROUTINE F08KQF (M, N, NRHS, A, LDA, B, LDB, S, RCOND, RANK, WORK, LWORK, &
                  RWORK, IWORK, INFO)
```

```
INTEGER          M, N, NRHS, LDA, LDB, RANK, LWORK, IWORK(*), INFO
```

```
REAL (KIND=nag_wp) S(*), RCOND, RWORK(*)
```

```
COMPLEX (KIND=nag_wp) A(LDA,*), B(LDB,*), WORK(max(1,LWORK))
```

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

3 Description

F08KQF (ZGELSD) uses the singular value decomposition (SVD) of A , where A is a complex m by n matrix which may be rank-deficient.

Several right-hand side vectors b and solution vectors x can be handled in a single call; they are stored as the columns of the m by r right-hand side matrix B and the n by r solution matrix X .

The problem is solved in three steps:

1. reduce the coefficient matrix A to bidiagonal form with Householder transformations, reducing the original problem into a 'bidiagonal least squares problem' (BLS);
2. solve the BLS using a divide-and-conquer approach;
3. apply back all the Householder transformations to solve the original least squares problem.

The effective rank of A is determined by treating as zero those singular values which are less than RCOND times the largest singular value.

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>

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

5 Parameters

1: M – INTEGER *Input*

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

Constraint: $M \geq 0$.

- 2: N – INTEGER *Input*
On entry: n , the number of columns of the matrix A .
Constraint: $N \geq 0$.
- 3: NRHS – INTEGER *Input*
On entry: r , the number of right-hand sides, i.e., the number of columns of the matrices B and X .
Constraint: $NRHS \geq 0$.
- 4: A(LDA,*) – COMPLEX (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 coefficient matrix A .
On exit: the contents of A are destroyed.
- 5: LDA – INTEGER *Input*
On entry: the first dimension of the array A as declared in the (sub)program from which F08KQF (ZGELSD) is called.
Constraint: $LDA \geq \max(1, M)$.
- 6: B(LDB,*) – COMPLEX (KIND=nag_wp) array *Input/Output*
Note: the second dimension of the array B must be at least $\max(1, NRHS)$.
On entry: the m by r right-hand side matrix B .
On exit: B is overwritten by the n by r solution matrix X . If $m \geq n$ and $RANK = n$, the residual sum of squares for the solution in the i th column is given by the sum of squares of the modulus of elements $n + 1, \dots, m$ in that column.
- 7: LDB – INTEGER *Input*
On entry: the first dimension of the array B as declared in the (sub)program from which F08KQF (ZGELSD) is called.
Constraint: $LDB \geq \max(1, M, N)$.
- 8: S(*) – REAL (KIND=nag_wp) array *Output*
Note: the dimension of the array S must be at least $\max(1, \min(M, N))$.
On exit: the singular values of A in decreasing order.
- 9: RCOND – REAL (KIND=nag_wp) *Input*
On entry: used to determine the effective rank of A . Singular values $S(i) \leq RCOND \times S(1)$ are treated as zero. If $RCOND < 0$, *machine precision* is used instead.
- 10: RANK – INTEGER *Output*
On exit: the effective rank of A , i.e., the number of singular values which are greater than $RCOND \times S(1)$.
- 11: WORK(max(1, LWORK)) – COMPLEX (KIND=nag_wp) array *Workspace*
On exit: if $INFO = 0$, the real part of $WORK(1)$ contains the minimum value of $LWORK$ required for optimal performance.
- 12: LWORK – INTEGER *Input*
On entry: the dimension of the array $WORK$ as declared in the (sub)program from which F08KQF (ZGELSD) is called.

The exact minimum amount of workspace needed depends on M , N and $NRHS$. As long as $LWORK$ is at least

$$\max(1, M + N + r, 2r + r \times NRHS),$$

where $r = \min(M, N)$, the code will execute correctly.

If $LWORK = -1$, a workspace query is assumed; the routine only calculates the optimal size of the $WORK$ array and the minimum size of the $IWORK$ array, and returns these values as the first entries of the $WORK$ and $IWORK$ arrays, and no error message related to $LWORK$ is issued.

Suggested value: for optimal performance, $LWORK$ should generally be larger than the required minimum. Consider increasing $LWORK$ by at least $nb \times \min(M, N)$, where nb is the optimal **block size**.

Constraint: $LWORK$ must be at least $\max(1, M + N + r, 2r + r \times NRHS)$ or $LWORK = -1$.

13: $RWORK(*)$ – REAL (KIND=nag_wp) array *Workspace*

Note: the dimension of the array $RWORK$ must be at least $\max(1, lrwork)$, where $lrwork$ is at least

$$10 \times N + 2 \times N \times smlsiz + 8 \times N \times nvl + 3 \times smlsiz \times NRHS + (smlsiz + 1)^2, \quad \text{if } M \geq N$$

or

$$10 \times M + 2 \times M \times smlsiz + 8 \times M \times nvl + 3 \times smlsiz \times NRHS + (smlsiz + 1)^2, \quad \text{if } M < N$$

where $smlsiz$ is equal to the maximum size of the subproblems at the bottom of the computation tree (usually about 25), and $nvl = \max(0, \text{int}(\log_2(\min(M, N)/(smlsiz + 1))) + 1)$, the code will execute correctly.

On exit: if $INFO = 0$, $RWORK(1)$ contains the required minimal size of $lrwork$.

14: $IWORK(*)$ – INTEGER array *Workspace*

Note: the dimension of the array $IWORK$ must be at least $\max(1, liwork)$, where $liwork$ is at least $\max(1, 3 \times \min(M, N) \times nvl + 11 \times \min(M, N))$.

On exit: if $INFO = 0$, $IWORK(1)$ returns the minimum $liwork$.

15: $INFO$ – INTEGER *Output*

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

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

$INFO < 0$

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

$INFO > 0$

The algorithm for computing the SVD failed to converge; if $INFO = i$, i off-diagonal elements of an intermediate bidiagonal form did not converge to zero.

7 Accuracy

See Section 4.5 of Anderson *et al.* (1999) for details.

8 Further Comments

The real analogue of this routine is F08KCF (DGELSD).

9 Example

This example solves the linear least squares problem

$$\min_x \|b - Ax\|_2$$

for the solution, x , of minimum norm, where

$$A = \begin{pmatrix} 0.47 - 0.34i & -0.32 - 0.23i & 0.35 - 0.60i & 0.89 + 0.71i & -0.19 + 0.06i \\ -0.40 + 0.54i & -0.05 + 0.20i & -0.52 - 0.34i & -0.45 - 0.45i & 0.11 - 0.85i \\ 0.60 + 0.01i & -0.26 - 0.44i & 0.87 - 0.11i & -0.02 - 0.57i & 1.44 + 0.80i \\ 0.80 - 1.02i & -0.43 + 0.17i & -0.34 - 0.09i & 1.14 - 0.78i & 0.07 + 1.14i \end{pmatrix}$$

and

$$b = \begin{pmatrix} 2.15 - 0.20i \\ -2.24 + 1.82i \\ 4.45 - 4.28i \\ 5.70 - 6.25i \end{pmatrix}.$$

A tolerance of 0.01 is used to determine the effective rank of A .

Note that the block size (NB) of 64 assumed in this example is not realistic for such a small problem, but should be suitable for large problems.

9.1 Program Text

```

Program f08kqfe

!      F08KQF Example Program Text

!      Mark 24 Release. NAG Copyright 2012.

!      .. Use Statements ..
Use nag_library, Only: nag_wp, zgelsd
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
Real (Kind=nag_wp)         :: rcond
Integer                    :: i, info, lda, liwork, lrwork, lwork, &
                           m, n, rank
!      .. Local Arrays ..
Complex (Kind=nag_wp), Allocatable :: a(:,,:), b(:), work(:)
Complex (Kind=nag_wp)           :: lw(1)
Real (Kind=nag_wp)              :: lrw(1)
Real (Kind=nag_wp), Allocatable :: rwork(:), s(:)
Integer, Allocatable            :: iwork(:)
Integer                        :: liw(1)
!      .. Intrinsic Procedures ..
Intrinsic                      :: nint, real
!      .. Executable Statements ..
Write (nout,*) 'F08KQF Example Program Results'
Write (nout,*)
!      Skip heading in data file
Read (nin,*)
Read (nin,*) m, n
lda = m
Allocate (a(lda,n),b(n),s(m))

!      Read A and B from data file

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

!      Choose RCOND to reflect the relative accuracy of the input
!      data

```

```

      rcond = 0.01E0_nag_wp

!      Call f08kqf/zgelsd in workspace query mode.
      lwork = -1
      Call zgelsd(m,n,l,a,lda,b,n,s,rcond,rank,lw,lwork,lrw,liw,info)
      lwork = nint(real(lw(1)))
      lrwork = nint(lrw(1))
      liwork = liw(1)
      Allocate (work(lwork),rwork(lrwork),iwork(liwork))

!      Solve the least squares problem min( norm2(b - Ax) ) for the
!      x of minimum norm.

!      The NAG name equivalent of zgelsd is f08kqf
      Call zgelsd(m,n,l,a,lda,b,n,s,rcond,rank,work,lwork,rwork,iwork,info)

      If (info==0) Then

!          Print solution

          Write (nout,*) 'Least squares solution'
          Write (nout,99999) b(1:n)

!          Print the effective rank of A

          Write (nout,*)
          Write (nout,*) 'Tolerance used to estimate the rank of A'
          Write (nout,99998) rcond
          Write (nout,*) 'Estimated rank of A'
          Write (nout,99997) rank

!          Print singular values of A

          Write (nout,*)
          Write (nout,*) 'Singular values of A'
          Write (nout,99996) s(1:m)
      Else If (info>0) Then
          Write (nout,*) 'The SVD algorithm failed to converge'
      End If

99999 Format (4(' (',F7.4,',',F7.4,')':))
99998 Format (3X,1P,E11.2)
99997 Format (1X,I6)
99996 Format (1X,7F11.4)
      End Program f08kqfe

```

9.2 Program Data

F08KQF Example Program Data

```

      4              5              :Values of M and N

( 0.47,-0.34) (-0.32,-0.23) ( 0.35,-0.60) ( 0.89, 0.71) (-0.19, 0.06)
(-0.40, 0.54) (-0.05, 0.20) (-0.52,-0.34) (-0.45,-0.45) ( 0.11,-0.85)
( 0.60, 0.01) (-0.26,-0.44) ( 0.87,-0.11) (-0.02,-0.57) ( 1.44, 0.80)
( 0.80,-1.02) (-0.43, 0.17) (-0.34,-0.09) ( 1.14,-0.78) ( 0.07, 1.14) :End of A

( 2.15,-0.20)
(-2.24, 1.82)
( 4.45,-4.28)
( 5.70,-6.25)              :End of vector b

```

9.3 Program Results

F08KQF Example Program Results

```

Least squares solution
( 3.9747,-1.8377) (-0.9186, 0.8253) (-0.3105, 0.1477) ( 1.0050, 0.8626)
(-0.2256,-1.9425)

```

Tolerance used to estimate the rank of A

1.00E-02

Estimated rank of A

3

Singular values of A

2.9979 1.9983 1.0044 0.0064
