NAG Fortran Library Routine Document

F08ZPF (ZGGGLM)

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

F08ZPF (ZGGGLM) solves a complex general Gauss–Markov linear (least-squares) model problem.

2 Specification

```
SUBROUTINE F08ZPF (M, N, P, A, LDA, B, LDB, D, X, Y, WORK, LWORK, INFO)
INTEGER            M, N, P, LDA, LDB, LWORK, INFO
complex*16         A(LDA,*), B(LDB,*), D(*), X(*), Y(*), WORK(*)
```

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

3 Description

F08ZPF (ZGGGLM) solves the complex general Gauss–Markov linear model (GLM) problem

$$\underset{x}{\text{minimize}}\ \|y\|_2 \quad \text{subject to} \quad d = Ax + By$$

where A is an m by n matrix, B is an m by p matrix and d is an m element vector. It is assumed that $n \leq m \leq n + p$, $\text{rank}(A) = n$ and $\text{rank}(E) = m$, where $E = (\,A \quad B\,)$. Under these assumptions, the problem has a unique solution x and a minimal 2-norm solution y, which is obtained using a generalized QR factorization of the matrices A and B.

In particular, if the matrix B is square and nonsingular, then the GLM problem is equivalent to the weighted linear least-squares problem

$$\underset{x}{\text{minimize}}\ \left\|B^{-1}(d - Ax)\right\|_2.$$

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

Anderson E, Bai Z and Dongarra J (1991) Generalized *QR* factorization and its applications *LAPACK Working Note No. 31* University of Tennessee, Knoxville

5 Parameters

1: M – INTEGER *Input*

On entry: m, the number of rows of the matrices A and B.

Constraint: $\text{M} \geq 0$.

2: N – INTEGER *Input*

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

Constraint: $0 \leq \text{N} \leq \text{M}$.

3: P – INTEGER *Input*

On entry: p, the number of columns of the matrix B.

Constraint: $\mathrm{P} \geq \mathrm{M} - \mathrm{N}$.

4: A(LDA,$*$) – ***complex*16*** array *Input/Output*

Note: the second dimension of the array A must be at least $\max(1, \mathrm{N})$.

On entry: the m by n matrix A.

On exit: A is overwritten.

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08ZPF (ZGGGLM) is called.

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

6: B(LDB,$*$) – ***complex*16*** array *Input/Output*

Note: the second dimension of the array B must be at least $\max(1, \mathrm{P})$.

On entry: the m by p matrix B.

On exit: B is overwritten.

7: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08ZPF (ZGGGLM) is called.

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

8: D($*$) – ***complex*16*** array *Input/Output*

Note: the dimension of the array D must be at least $\max(1, \mathrm{M})$.

On entry: the left-hand side vector d of the GLM equation.

On exit: D is overwritten.

9: X($*$) – ***complex*16*** array *Output*

Note: the dimension of the array X must be at least $\max(1, \mathrm{N})$.

On exit: the solution vector x of the GLM problem.

10: Y($*$) – ***complex*16*** array *Output*

Note: the dimension of the array Y must be at least $\max(1, \mathrm{P})$.

On exit: the solution vector y of the GLM problem.

11: WORK($*$) – ***complex*16*** array *Workspace*

Note: the dimension of the array WORK must be at least $\max(1, \mathrm{LWORK})$.

On exit: if $\mathrm{IFAIL} = 0$, $\mathrm{WORK}(1)$ contains the minimum value of LWORK required for optimum performance.

12: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the subprogram from which F08ZPF (ZGGGLM) is called unless $\mathrm{LWORK} = -1$, in which case a workspace query is assumed and the routine only calculates the optimal dimension of WORK (using the formula given below).

Suggested value: for optimum performance LWORK should be at least $\mathrm{N} + \min(\mathrm{M}, \mathrm{P}) + \max(\mathrm{M}, \mathrm{P}) \times nb$, where nb is the ***block size***.

Constraint: $\mathrm{LWORK} \geq \max(1, \mathrm{M} + \mathrm{N} + \mathrm{P})$ or $\mathrm{LWORK} = -1$.

13: INFO – INTEGER *Output*

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

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

$\mathrm{INFO} < 0$

If $\mathrm{INFO} = -i$, the ith argument had an illegal value.

7 Accuracy

For an error analysis, see Anderson *et al.* (1991). See also Section 4.6 of Anderson *et al.* (1999).

8 Further Comments

When $p = m \geq n$, the total number of real floating-point operations is approximately $\frac{8}{3}(2m^3 - n^3) + 16nm^2$; when $p = m = n$, the total number of real floating-point operations is approximately $\frac{56}{3}m^3$.

9 Example

To solve the weighted least-squares problem

$$\underset{x}{\text{minimize}} \left\| B^{-1}(d - Ax) \right\|_2,$$

where

$$B = \begin{pmatrix} 0.5 - 1.0i & 0.0 + 0 & 0.0 + 0 & 0.0 + 0 \\ 0.0 + 0 & 1.0 - 2.0i & 0.0 + 0 & 0.0 + 0 \\ 0.0 + 0 & 0.0 + 0 & 2.0 - 3.0i & 0.0 + 0 \\ 0.0 + 0 & 0.0 + 0 & 0.0 + 0 & 5.0 - 4.0i \end{pmatrix},$$

$$d = \begin{pmatrix} 6.00 - 0.40i \\ -5.27 + 0.90i \\ 2.72 - 2.13i \\ -1.30 - 2.80i \end{pmatrix}$$

and

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.03 + 0.96i & -0.91 + 2.06i \\ -0.98 + 1.98i & -1.20 + 0.19i & -0.66 + 0.42i \\ 0.62 - 0.46i & 1.01 + 0.02i & 0.63 - 0.17i \\ 1.08 - 0.28i & 0.20 - 0.12i & -0.07 + 1.23i \end{pmatrix}.$$

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

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F08ZPF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, NB, NMAX, PMAX
      PARAMETER        (MMAX=10,NB=64,NMAX=10,PMAX=10)
      INTEGER          LDA, LDB, LWORK
      PARAMETER        (LDA=NMAX,LDB=NMAX,LWORK=MMAX+NMAX+NB*(NMAX+PMAX)
     +                 )
*     .. Local Scalars ..
      DOUBLE PRECISION RNORM
      INTEGER          I, INFO, J, M, N, P
*     .. Local Arrays ..
      COMPLEX *16      A(LDA,MMAX), B(LDB,PMAX), D(NMAX), WORK(LWORK),
     +                 X(MMAX), Y(PMAX)
*     .. External Functions ..
      DOUBLE PRECISION DZNRM2
      EXTERNAL         DZNRM2
*     .. External Subroutines ..
      EXTERNAL         ZGGGLM
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F08ZPF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, M, P
      IF (M.LE.MMAX .AND. N.LE.NMAX .AND. P.LE.PMAX) THEN
*
*        Read A, B and D from data file
*
         READ (NIN,*) ((A(I,J),J=1,M),I=1,N)
         READ (NIN,*) ((B(I,J),J=1,P),I=1,N)
         READ (NIN,*) (D(I),I=1,N)
*
*        Solve the weighted least-squares problem
*
*        minimize ||inv(B)*(d - A*x)|| (in the 2-norm)
*
         CALL ZGGGLM(N,M,P,A,LDA,B,LDB,D,X,Y,WORK,LWORK,INFO)
*
*        Print least-squares solution
*
         WRITE (NOUT,*) 'Weighted least-squares solution'
         WRITE (NOUT,99999) (X(I),I=1,M)
*
*        Print residual vector y = inv(B)*(d - A*x)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Residual vector'
         WRITE (NOUT,99998) (Y(I),I=1,P)
*
*        Compute and print the square root of the residual sum of
*        squares
*
         RNORM = DZNRM2(P,Y,1)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Square root of the residual sum of squares'
         WRITE (NOUT,99997) RNORM
      ELSE
         WRITE (NOUT,*)
     +     'One or more of MMAX, NMAX and PMAX is too small'
      END IF
      STOP
```

```
*
99999 FORMAT (3(' (',F9.4,',',F9.4,')',:))
99998 FORMAT (3(' (',1P,E9.2,',',1P,E9.2,')',:))
99997 FORMAT (1X,1P,E10.2)
      END
```

9.2 Program Data

```
F08ZPF Example Program Data

  4             3             4                         :Values of M, N and P

( 0.96,-0.81) (-0.03, 0.96) (-0.91, 2.06)
(-0.98, 1.98) (-1.20, 0.19) (-0.66, 0.42)
( 0.62,-0.46) ( 1.01, 0.02) ( 0.63,-0.17)
( 1.08,-0.28) ( 0.20,-0.12) (-0.07, 1.23)               :End of matrix A

( 0.50,-1.00) ( 0.00, 0.00) ( 0.00, 0.00) ( 0.00, 0.00)
( 0.00, 0.00) ( 1.00,-2.00) ( 0.00, 0.00) ( 0.00, 0.00)
( 0.00, 0.00) ( 0.00, 0.00) ( 2.00,-3.00) ( 0.00, 0.00)
( 0.00, 0.00) ( 0.00, 0.00) ( 0.00, 0.00) ( 5.00,-4.00) :End of matrix B

( 6.00,-0.40)
(-5.27, 0.90)
( 2.72,-2.13)
(-1.30,-2.80)                                           :End of vector d
```

9.3 Program Results

```
 F08ZPF Example Program Results

 Weighted least-squares solution
 (  -0.9846,   1.9950) (   3.9929,  -4.9748) (  -3.0026,   0.9994)

 Residual vector
 ( 1.26E-04,-4.66E-04) ( 1.11E-03,-8.61E-04) ( 3.84E-03,-1.82E-03)
 ( 2.03E-03, 3.02E-03)

 Square root of the residual sum of squares
   5.79E-03
```