NAG Library Routine Document

G02DKF

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

G02DKF calculates the estimates of the arguments of a general linear regression model for given constraints from the singular value decomposition results.

2 Specification

```
SUBROUTINE G02DKF (IP, ICONST, P, C, LDC, B, RSS, IDF, SE, COV, WK,      &
                   IFAIL)

INTEGER           IP, ICONST, LDC, IDF, IFAIL
REAL (KIND=nag_wp) P(IP*IP+2*IP), C(LDC,ICONST), B(IP), RSS, SE(IP),     &
                   COV(IP*(IP+1)/2),                                     &
                   WK(2*IP*IP+IP*ICONST+2*ICONST*ICONST+4*ICONST)
```

3 Description

G02DKF computes the estimates given a set of linear constraints for a general linear regression model which is not of full rank. It is intended for use after a call to G02DAF or G02DDF.

In the case of a model not of full rank the routines use a singular value decomposition (SVD) to find the parameter estimates, $\hat{\beta}_{\text{svd}}$, and their variance-covariance matrix. Details of the SVD are made available in the form of the matrix P^*:

$$P^* = \begin{pmatrix} D^{-1}P_1^{\text{T}} \\ P_0^{\text{T}} \end{pmatrix},$$

as described by G02DAF and G02DDF.

Alternative solutions can be formed by imposing constraints on the arguments. If there are p arguments and the rank of the model is k, then $n_c = p - k$ constraints will have to be imposed to obtain a unique solution.

Let C be a p by n_c matrix of constraints, such that

$$C^{\text{T}}\beta = 0$$

then the new parameter estimates $\hat{\beta}_c$ are given by

$$\hat{\beta}_c \qquad\qquad = A\hat{\beta}_{\text{svd}};$$

$$= \left(I - P_0(C^{\text{T}}P_0)^{-1}\right)\hat{\beta}_{\text{svd}},$$

where I is the identity matrix, and the variance-covariance matrix is given by

$$AP_1 D^{-2}P_1^{\text{T}} A^{\text{T}},$$

provided $(C^{\text{T}}P_0)^{-1}$ exists.

4 References

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

Hammarling S (1985) The singular value decomposition in multivariate statistics *SIGNUM Newsl.* **20(3)** 2–25

Searle S R (1971) *Linear Models* Wiley

5 Arguments

1:     IP – INTEGER   *Input*

On entry: p, the number of terms in the linear model.

Constraint: $IP \geq 1$.

2:     ICONST – INTEGER   *Input*

On entry: the number of constraints to be imposed on the arguments, n_c.

Constraint: $0 < ICONST < IP$.

3:     $P(IP \times IP + 2 \times IP)$ – REAL (KIND=nag_wp) array   *Input*

On entry: as returned by G02DAF and G02DDF.

4:     $C(LDC, ICONST)$ – REAL (KIND=nag_wp) array   *Input*

On entry: the ICONST constraints stored by column, i.e., the ith constraint is stored in the ith column of C.

5:     LDC – INTEGER   *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which G02DKF is called.

Constraint: $LDC \geq IP$.

6:     $B(IP)$ – REAL (KIND=nag_wp) array   *Input/Output*

On entry: the parameter estimates computed by using the singular value decomposition, $\hat{\beta}_{svd}$.

On exit: the parameter estimates of the arguments with the constraints imposed, $\hat{\beta}_c$.

7:     RSS – REAL (KIND=nag_wp)   *Input*

On entry: the residual sum of squares as returned by G02DAF or G02DDF.

Constraint: $RSS > 0.0$.

8:     IDF – INTEGER   *Input*

On entry: the degrees of freedom associated with the residual sum of squares as returned by G02DAF or G02DDF.

Constraint: $IDF > 0$.

9:     $SE(IP)$ – REAL (KIND=nag_wp) array   *Output*

On exit: the standard error of the parameter estimates in B.

10:     COV(IP × (IP + 1)/2) – REAL (KIND=nag_wp) array   *Output*

On exit: the upper triangular part of the variance-covariance matrix of the IP parameter estimates given in B. They are stored packed by column, i.e., the covariance between the parameter estimate given in $B(i)$ and the parameter estimate given in $B(j)$, $j \geq i$, is stored in $COV((j \times (j-1)/2 + i))$.

11:     WK(2 × IP × IP + IP × ICONST + 2 × ICONST × ICONST + 4 × ICONST)
        – REAL (KIND=nag_wp) array   *Workspace*

Note that a simple upper bound for the size of the workspace is $5 \times IP \times IP$.

12:     IFAIL – INTEGER   *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

6   Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

    On entry, IP < 1,
    or   ICONST ≤ 0,
    or   ICONST ≥ IP,
    or   LDC < IP,
    or   RSS ≤ 0.0,
    or   IDF ≤ 0.

IFAIL = 2

    C does not give a model of full rank.

IFAIL = -99

    An unexpected error has been triggered by this routine. Please contact NAG.

    See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -399

    Your licence key may have expired or may not have been installed correctly.

    See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -999

    Dynamic memory allocation failed.

    See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7   Accuracy

It should be noted that due to rounding errors an argument that should be zero when the constraints have been imposed may be returned as a value of order ***machine precision***.

8   Parallelism and Performance

G02DKF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

G02DKF 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

G02DKF is intended for use in situations in which dummy $(0-1)$ variables have been used such as in the analysis of designed experiments when you do not wish to change the arguments of the model to give a full rank model. The routine is not intended for situations in which the relationships between the independent variables are only approximate.

10   Example

Data from an experiment with four treatments and three observations per treatment are read in. A model, including the mean term, is fitted by G02DAF and the results printed. The constraint that the sum of treatment effect is zero is then read in and the parameter estimates with this constraint imposed are computed by G02DKF and printed.

10.1 Program Text

```
    Program g02dkfe

!     G02DKF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g02daf, g02dkf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: rss, tol
      Integer                          :: i, iconst, idf, ifail, ip, irank,    &
                                          ldc, ldq, ldx, lwk, lwt, m, n, tlwk
      Logical                          :: svd
      Character (1)                    :: mean, weight
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: b(:), c(:,:), cov(:), h(:), p(:),    &
                                          q(:,:), res(:), se(:), wk(:), wt(:), &
                                          x(:,:), y(:)
      Integer, Allocatable             :: isx(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: count
!     .. Executable Statements ..
      Write (nout,*) 'G02DKF Example Program Results'
      Write (nout,*)

!     Skip heading in data file
```

```
      Read (nin,*)
      Read (nin,*) n, m, weight, mean

!     Read in data
      If (weight=='W' .Or. weight=='w') Then
        lwt = n
      Else
        lwt = 0
      End If
      ldx = n
      Allocate (x(ldx,m),isx(m),y(n),wt(lwt))

      If (lwt>0) Then
        Read (nin,*)(x(i,1:m),y(i),wt(i),i=1,n)
      Else
        Read (nin,*)(x(i,1:m),y(i),i=1,n)
      End If

!     Read in variable inclusion flags
      Read (nin,*) isx(1:m)

!     Calculate IP
      ip = count(isx>0)
      If (mean=='M' .Or. mean=='m') Then
        ip = ip + 1
      End If

      lwk = 5*(ip-1) + ip*ip
      ldq = n
      ldc = ip
      Allocate (b(ip),se(ip),cov(ip*(ip+1)/2),res(n),h(n),q(ldq,ip+1),p(2*ip+  &
        ip*ip),wk(lwk),c(ldc,ip))

!     Use suggested value for tolerance
      tol = 0.000001E0_nag_wp

!     Fit general linear regression model
      ifail = 0
      Call g02daf(mean,weight,n,x,ldx,m,isx,ip,y,wt,rss,idf,b,se,cov,res,h,q,  &
        ldq,svd,irank,p,tol,wk,ifail)

!     Display the initial estimates
      Write (nout,*) 'Estimates from G02DAF'
      Write (nout,*)
      Write (nout,99999) 'Residual sum of squares = ', rss
      Write (nout,99998) 'Degrees of freedom = ', idf
      Write (nout,*)
      Write (nout,*) 'Variable    Parameter estimate   Standard error'
      Write (nout,*)
      Write (nout,99997)(i,b(i),se(i),i=1,ip)

!     Read in constraints
      iconst = ip - irank
      Read (nin,*)(c(i,1:iconst),i=1,ip)

      tlwk = 2*ip*ip + ip*iconst + 2*iconst*iconst + 4*iconst
      If (tlwk>lwk) Then
!       Reallocate workspace
        Deallocate (wk)
        Allocate (wk(tlwk))
      End If

!     Re-estimate the parameters given the constraints
      ifail = 0
      Call g02dkf(ip,iconst,p,c,ldc,b,rss,idf,se,cov,wk,ifail)

!     Display the results
      Write (nout,*)
      Write (nout,*) 'Estimates from G02DKF using constraints'
      Write (nout,*)
      Write (nout,*) 'Variable    Parameter estimate   Standard error'
```

```
      Write (nout,*)
      Write (nout,99997)(i,b(i),se(i),i=1,ip)

99999 Format (1X,A,E13.3)
99998 Format (1X,A,I4)
99997 Format (1X,I6,2E20.3)
    End Program g02dkfe
```

10.2 Program Data

```
G02DKF Example Program Data
 12 4 'U' 'M'              :: N, M, MEAN, WEIGHT
1.0 0.0 0.0 0.0 33.63
0.0 0.0 0.0 1.0 39.62
0.0 1.0 0.0 0.0 38.18
0.0 0.0 1.0 0.0 41.46
0.0 0.0 0.0 1.0 38.02
0.0 1.0 0.0 0.0 35.83
0.0 0.0 0.0 1.0 35.99
1.0 0.0 0.0 0.0 36.58
0.0 0.0 1.0 0.0 42.92
1.0 0.0 0.0 0.0 37.80
0.0 0.0 1.0 0.0 40.43
0.0 1.0 0.0 0.0 37.89   :: End of X, Y
 1   1   1   1          :: ISX
0.0
1.0
1.0
1.0
1.0                     :: C (constraints)
```

10.3 Program Results

```
 G02DKF Example Program Results

 Estimates from G02DAF

 Residual sum of squares =     0.222E+02
 Degrees of freedom =    8

 Variable    Parameter estimate    Standard error

      1            0.306E+02             0.385E+00
      2            0.545E+01             0.839E+00
      3            0.674E+01             0.839E+00
      4            0.110E+02             0.839E+00
      5            0.732E+01             0.839E+00

 Estimates from G02DKF using constraints

 Variable    Parameter estimate    Standard error

      1            0.382E+02             0.481E+00
      2           -0.219E+01             0.833E+00
      3           -0.896E+00             0.833E+00
      4            0.341E+01             0.833E+00
      5           -0.319E+00             0.833E+00
```