G03BAF – NAG Fortran Library Routine Document

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

G03BAF computes orthogonal rotations for a matrix of loadings using a generalized orthomax criterion.

2 Specification

```
 SUBROUTINE G03BAF(STAND, G, NVAR, K, FL, LDF, FLR, R, LDR, ACC,
1                  MAXIT, ITER, WK, IFAIL)
 INTEGER           NVAR, K, LDF, LDR, MAXIT, ITER, IFAIL
 real              G, FL(LDF,K), FLR(LDF,K), R(LDR,K), ACC,
1                  WK(2*NVAR+K*K+5*(K-1))
 CHARACTER*1       STAND
```

3 Description

Let Λ be the p by k matrix of loadings from a variable-directed multivariate method, e.g., canonical variate analysis or factor analysis. This matrix represents the relationship between the original p variables and the k orthogonal linear combinations of these variables, the canonical variates or factors. The latter are only unique up to a rotation in the k-dimensional space they define. A rotation can then be found that simplifies the structure of the matrix of loadings, and hence the relationship between the original and the derived variables. That is the elements, λ^*_{ij}, of the rotated matrix, Λ^*, are either relatively large or small. The rotations may be found by minimizing the criterion:

$$V = \sum_{j=1}^{k} \sum_{i=1}^{p} (\lambda^*_{ij})^4 - \frac{\gamma}{p} \sum_{j=1}^{k} \left[\sum_{i=1}^{p} (\lambda^*_{ij})^2 \right]^2$$

where the constant γ gives a family of rotations with $\gamma = 1$ giving varimax rotations and $\gamma = 0$ giving quartimax rotations.

It is generally advised that factor loadings should be standardised, so that the sum of squared elements for each row is one, before computing the rotations.

The matrix of rotations, R, such that $\Lambda^* = \Lambda R$, is computed using first an algorithm based on that described by Cooley and Lohnes [1], which involves the pairwise rotation of the factors. Then a final refinement is made using a method similar to that described by Lawley and Maxwell [2], but instead of the eigenvalue decomposition the algorithm has been adapted to incorporate a singular value decomposition.

4 References

[1] Cooley W C and Lohnes P R (1971) *Multivariate Data Analysis* Wiley

[2] Lawley D N and Maxwell A E (1971) *Factor Analysis as a Statistical Method* Butterworths (2nd Edition)

5 Parameters

1: STAND — CHARACTER*1 *Input*

On entry: indicates if the matrix of loadings is to be row standardised before rotation.

If STAND = 'S' the loadings are row standardised.
If STAND = 'U' the loadings are left unstandardised.

Constraint: STAND = 'S' or 'U'.

2: G — ***real*** *Input*

On entry: the criterion constant, γ, with $\gamma = 1.0$ giving varimax rotations and $\gamma = 0.0$ giving quartimax rotations.

Constraint: G $\geq$ 0.0.

3: NVAR — INTEGER *Input*

On entry: the number of original variables, p.

Constraint: NVAR $\geq$ K.

4: K — INTEGER *Input*

On entry: the number of derived variates or factors, k.

Constraint: K $\geq$ 2.

5: FL(LDF,K) — ***real*** array *Input/Output*

On entry: the matrix of loadings, Λ. FL(i,j) must contain the loading for the ith variable on the jth factor, for $i = 1,2,\ldots,p$; $j = 1,2,\ldots,k$.

On exit: if STAND = 'S' the elements of FL are standardised so that the sum of squared elements for each row is 1.0 and then after the computation of the rotations are rescaled; this may lead to slight differences between the input and output values of FL. If STAND = 'U' FL will be unchanged on exit.

6: LDF — INTEGER *Input*

On entry: the first dimension of the arrays FL and FLR as declared in the (sub)program from which G03BAF is called.

Constraint: LDF $\geq$ NVAR.

7: FLR(LDF,K) — ***real*** array *Output*

On exit: the rotated matrix of loadings, Λ^*. FLR(i,j) will contain the rotated loading for the ith variable on the jth factor, for $i = 1,2,\ldots,p$; $j = 1,2,\ldots,k$.

8: R(LDR,K) — ***real*** array *Output*

On exit: the matrix of rotations, R.

9: LDR — INTEGER *Input*

On entry: the first dimension of the array R as declared in the (sub)program from which G03BAF is called.

Constraint: LDR $\geq$ K.

10: ACC — ***real*** *Input*

On entry: indicates the accuracy required. The iterative procedure of Cooley and Lohnes [1] will be stopped and the final refinement computed when the change in V is less than ACC $\times$ max$(1.0, V)$. If ACC is greater than or equal to 0.0 but less than ***machine precision*** or if ACC is greater than 1.0, then ***machine precision*** will be used instead.

Suggested value: 0.00001.

Constraint: ACC $\geq$ 0.0.

11: MAXIT — INTEGER *Input*

On entry: the maximum number of iterations.

Suggested value: 30.

Constraint: MAXIT $\geq$ 1.

12: ITER — INTEGER *Output*

On exit: the number of iterations performed.

13: WK(2∗NVAR+K∗K+5∗(K−1)) — ***real*** array *Workspace*

14: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (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 detected by the routine:

IFAIL = 1

On entry, K < 2,
or NVAR < K,
or G < 0.0,
or LDF < NVAR,
or LDR < K,
or ACC < 0.0,
or MAXIT $\leq$ 0,
or STAND $\neq$ 'S' or 'U'.

IFAIL = 2

The singular value decomposition has failed to converge. This is an unlikely error exit.

IFAIL = 3

The algorithm to find R has failed to reach the required accuracy in the given number of iterations. The user should try increasing ACC or increasing MAXIT. The returned solution should be a reasonable approximation.

7 Accuracy

The accuracy is determined by the value of ACC.

8 Further Comments

If the results of a principal component analysis as carried out by G03AAF are to be rotated, the loadings as returned in the array P by G03AAF can be supplied via the parameter FL to G03BAF>. The resulting rotation matrix can then be used to rotate the principal component scores as returned in the array V by G03AAF. The routine F06YAF may be used for this matrix multiplication.

9 Example

The example is taken from Lawley and Maxwell [2] (page 75). The results from a factor analysis of ten variables using three factors are input and rotated using varimax rotations without standardising rows.

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.

```
*     G03BAF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX
      PARAMETER        (NMAX=10,MMAX=3)
*     .. Local Scalars ..
      real             ACC, G
      INTEGER          I, IFAIL, ITER, J, K, MAXIT, NVAR
      CHARACTER        STAND
*     .. Local Arrays ..
      real             FL(NMAX,MMAX), FLR(NMAX,MMAX), R(MMAX,MMAX),
     +                 WK(2*NMAX+MMAX*MMAX+5*(MMAX-1))
*     .. External Subroutines ..
      EXTERNAL         G03BAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G03BAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NVAR, K, G, STAND, ACC, MAXIT
      IF (NVAR.LE.NMAX .AND. K.LE.MMAX) THEN
         DO 20 I = 1, NVAR
            READ (NIN,*) (FL(I,J),J=1,K)
   20    CONTINUE
         IFAIL = 0
*
         CALL G03BAF(STAND,G,NVAR,K,FL,NMAX,FLR,R,MMAX,ACC,MAXIT,ITER,
     +               WK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) '    Rotated factor loadings'
         WRITE (NOUT,*)
         DO 40 I = 1, NVAR
            WRITE (NOUT,99999) (FLR(I,J),J=1,K)
   40    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) '    Rotation matrix'
         WRITE (NOUT,*)
         DO 60 I = 1, K
            WRITE (NOUT,99999) (R(I,J),J=1,K)
   60    CONTINUE
      END IF
      STOP
*
99999 FORMAT (4(2X,6F8.3))
      END
```

9.2 Program Data

```
 G03BAF Example Program Data
   10 3 1.0 'U' 0.00001 20
  0.788 -0.152 -0.352
  0.874  0.381  0.041
  0.814 -0.043 -0.213
  0.798 -0.170 -0.204
  0.641  0.070 -0.042
  0.755 -0.298  0.067
  0.782 -0.221  0.028
  0.767 -0.091  0.358
  0.733 -0.384  0.229
  0.771 -0.101  0.071
```

9.3 Program Results

```
 G03BAF Example Program Results

     Rotated factor loadings

     0.329  -0.289  -0.759
     0.849  -0.273  -0.340
     0.450  -0.327  -0.633
     0.345  -0.397  -0.657
     0.453  -0.276  -0.370
     0.263  -0.615  -0.464
     0.332  -0.561  -0.485
     0.472  -0.684  -0.183
     0.209  -0.754  -0.354
     0.423  -0.514  -0.409

     Rotation matrix

     0.633  -0.534  -0.560
     0.758   0.573   0.311
     0.155  -0.622   0.768
```