NAG Toolbox for MATLAB

g03ba

1 Purpose

g03ba computes orthogonal rotations for a matrix of loadings using a generalized orthomax criterion.

2 Syntax

```
[fl, flr, r, iter, ifail] = g03ba(stand, g, nvar, fl, 'k', k, 'acc',
acc, 'maxit', maxit)
```

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:

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

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 standardized, 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 1971, 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 1971, but instead of the eigenvalue decomposition, the algorithm has been adapted to incorporate a singular value decomposition.

4 References

Cooley W C and Lohnes P R 1971 *Multivariate Data Analysis* Wiley

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

5 Parameters

5.1 Compulsory Input Parameters

1: **stand – string**

Indicates if the matrix of loadings is to be row standardized before rotation.

stand = 'S'

The loadings are row standardized.

stand = 'U'

The loadings are left unstandardized.

Constraint: **stand** = 'S' or 'U'.

2: **g – double scalar**

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

Constraint: $\mathbf{g} \geq 0.0$.

3: **nvar – int32 scalar**

p, the number of original variables.

Constraint: $\mathbf{nvar} \geq \mathbf{k}$.

4: **fl(ldfl,k) – double array**

ldfl, the first dimension of the array, must be at least **nvar**.

Λ, the matrix of loadings. **fl**(i,j) must contain the loading for the ith variable on the jth factor, for $i = 1,2,\ldots,p$ and $j = 1,2,\ldots,k$.

5.2 Optional Input Parameters

1: **k – int32 scalar**

Default: The second dimension of the arrays **fl**, **flr**, **r**. (An error is raised if these dimensions are not equal.)

k, the number of derived variates or factors.

Constraint: $\mathbf{k} \geq 2$.

2: **acc – double scalar**

Indicates the accuracy required. The iterative procedure of Cooley and Lohnes 1971 will be stopped and the final refinement computed when the change in V is less than $\mathbf{acc} \times \max(1.0, V)$. 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.

Default: 0.00001

Constraint: $\mathbf{acc} \geq 0.0$.

3: **maxit – int32 scalar**

The maximum number of iterations.

Constraint: $\mathbf{maxit} \geq 1$.

5.3 Input Parameters Omitted from the MATLAB Interface

ldfl, ldr, wk

5.4 Output Parameters

1: **fl(ldfl,k) – double array**

If **stand** = 'S', the elements of **fl** are standardized 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.

2: **flr(ldfl,k) – double array**

The rotated matrix of loadings, Λ^*. **flr**(i,j) will contain the rotated loading for the ith variable on the jth factor, for $i = 1,2,\dots,p$ and $j = 1,2,\dots,k$.

3: **r(ldr,k) – double array**

The matrix of rotations, **r**.

4: **iter – int32 scalar**

The number of iterations performed.

5: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **k** < 2,
or **nvar** $<$ **k**,
or **g** < 0.0,
or **ldfl** $<$ **nvar**,
or **ldr** $<$ **k**,
or **acc** < 0.0,
or **maxit** ≤ 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. You 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

None.

9 Example

```
stand = 'U';
g = 1;
nvar = int32(10);
fl = [0.788, -0.152, -0.352;
      0.874, 0.381, 0.041;
      0.8139999999999999, -0.043, -0.213;
      0.798, -0.17, -0.204;
      0.641, 0.07000000000000001, -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, 0.07099999999999999];
[flOut, flr, r, iter, ifail] = g03ba(stand, g, nvar, fl)
```

```
flOut =
    0.7880   -0.1520   -0.3520
    0.8740    0.3810    0.0410
    0.8140   -0.0430   -0.2130
    0.7980   -0.1700   -0.2040
    0.6410    0.0700   -0.0420
    0.7550   -0.2980    0.0670
    0.7820   -0.2210    0.0280
    0.7670   -0.0910    0.3580
    0.7330   -0.3840    0.2290
    0.7710   -0.1010    0.0710
flr =
    0.3293   -0.2888   -0.7590
    0.8488   -0.2735   -0.3397
    0.4500   -0.3266   -0.6330
    0.3450   -0.3965   -0.6566
    0.4526   -0.2758   -0.3696
    0.2628   -0.6154   -0.4642
    0.3322   -0.5614   -0.4854
    0.4725   -0.6841   -0.1832
    0.2088   -0.7537   -0.3543
    0.4229   -0.5135   -0.4089
r =
    0.6335   -0.5337   -0.5603
    0.7580    0.5733    0.3109
    0.1553   -0.6217    0.7677
iter =
           7
ifail =
           0
```