Apply GORICA to `EffectLiteR` estimates

Axel Mayer, Leonard Vanbrabant, and Rebecca M. Kuiper

27 August 2024

Contents

This tutorial shows how one can apply the GORICA – an AIC-type information criterion that can evaluate order-restricted, theory-based hypotheses – to the estimates of the `EffectLiteR` package.

The `EffectLiteR` package uses structural equation modeling to estimate average and conditional effects of a treatment variable on an outcome variable, taking into account multiple continuous and categorical covariates. It automatically generates `lavaan` (Rosseel, 2012) syntax for a multi-group structural equation model, runs the model in `lavaan`, and extracts various average and conditional effects of interest. The main function of the package is `effectLite`.

In this tutorial, it will be shown how you can evaluate order-restricted, theory-based hypotheses regarding average and/or conditional effects, using the `restriktor` and `EffectLiteR` packages in R.

For more information regarding `EffectLiteR`, see:
Mayer, A., Dietzfelbinger, L., Rosseel, Y. & Steyer, R. (2016). The EffectLiteR approach for analyzing average and conditional effects. *Multivariate Behavioral Research*, 51, 374-391.

For (more) information regarding interpreting the GORIC(A) output, see 'Guidelines_output_GORIC' (https://github.com/rebeccakuiper/Tutorials).

Data preparation

First, load the required libraries (after they have been installed). These libraries contain functions, such as `goric`, that will be used in the R code below. Each time you reopen R, you will have to load the required libraries.

```
## First, install the packages, if you have not done this already:
if (!require("EffectLiteR")) install.packages("EffectLiteR")
if (!require("restriktor")) install.packages("restriktor")
if (!require("psych")) install.packages("psych")
```

```
## Then, load the packages:
library(EffectLiteR)
library(restriktor) # for the goric function
library(psych) # for the function describeBy

# If you want to use restriktor from github:
#if (!require("devtools")) install.packages("devtools")
#library(devtools)
#install_github("LeonardV/restriktor")
#library(restriktor) # for goric function
```

Second, it is necessary to load the data.

For this Tutorial, you can use the data available from the `EffectLiteR` package. Several datasets are available by loading the package and here the following two datasets are used:

- example01 (for an adjusted means example)
- nonortho (for an average effects example)

To use the data accordingly, it is recommended to save it in a separate object.

```
# Load the data
data_AdjMeans <- example01
data_AvEffects <- nonortho
```

Example 1 - Adjusted Means

Inspect the data

To inspect the dataset, use the `head` function:

```
head(data_AdjMeans) # Look at first (6) rows of the data
```

```
          x     k1 kateg2          z1         z2          z3         dv
9    treat2   male      2  0.57664715 -0.4946821  1.51831377  1.5595743
6    treat2 female      1 -0.05489584 -0.9371025  1.54194766 -0.4078295
7   control   male      2 -1.88284217 -2.1620611  0.01940434  0.1354053
11   treat1 female      2  0.68738815  1.0884689  1.01883391  0.2818120
10  control female      2 -0.18567233  0.6432084 -0.42386395 -0.3440413
6.1  treat2 female      1 -0.84563945 -1.1052582 -0.22462067  0.5631160
```

To see a more detailed overview of the data via descriptive statistics split by `group` variable, use the `describeBy` function with `data_AdjMeans$x` set to be a grouping variable:

```
descrstat <- describeBy(data_AdjMeans$dv, data_AdjMeans$x, mat = TRUE, digits = 3)
descrstat
```

```
    item  group1 vars   n  mean    sd median trimmed   mad    min   max range   skew kurtosis    se
X11    1 control    1 686 0.018 0.965  0.034   0.021 0.980 -2.746 2.473 5.219 -0.045   -0.297 0.037
X12    2  treat1    1 615 0.008 1.009  0.012  -0.004 1.015 -2.664 4.183 6.846  0.195    0.264 0.041
X13    3  treat2    1 699 0.050 1.015  0.020   0.041 1.081 -2.960 3.283 6.243  0.110   -0.194 0.038
```

Preparation for GORIC(A)

Use EffectLiteR

To use the estimates for the adjusted means with GORICA, an object with the main function `effectLite` needs to be created:

```
model_AdjMeans <- effectLite(y="dv", x="x", z=c("z1","z2"), data=data_AdjMeans, method="sem")
```

For input in `effectLite`:

y: Dependent variable (character string). Can be the name of a manifest variable or of a latent variable.

x: Treatment variable (character string) treated as categorical variable.

z: Vector of continuous covariates (character vector). Names of both manifest and latent variables are allowed.

data: A data frame containing the data above.

method: Can be one of c("sem","lm") and indicates which function is used to fit the model.

Extract information from the EffectLiteR object

Note that `goric` can use `lm` or `glm` objects (even most `lavaan` objects) as input. When using the GORICA, `goric` has another input option as well (that is why the GORICA can easily be applied to a broad range of models): One can enter the (structural) parameter estimates and their covariance matrix. Thus, these must be extracted from the `model_AdjMeans` object.

For this example, we are interested in the adjusted means for the dv and the examined covariates and the covariance between them:

```
parnames <- c("adjmean0","adjmean1","adjmean2") # to have a better overview
est_AdjMeans <- model_AdjMeans@results@est[parnames]
VCOV_AdjMeans <- model_AdjMeans@results@vcov.def[parnames,parnames]

round(est_AdjMeans, 3)
```

```
adjmean0 adjmean1 adjmean2
   0.017    0.006    0.051
```

```
round(VCOV_AdjMeans, 2)
```

```
         adjmean0 adjmean1 adjmean2
adjmean0        0        0        0
adjmean1        0        0        0
adjmean2        0        0        0
```

Set of hypotheses

Hypotheses

In the example, the following could be hypothesized regarding the adjusted means (before seeing the data):

```
H1 <- "adjmean1 < adjmean0 < adjmean2"
```

This hypothesis will be inspected versus its complement, meaning all other possible orderings / hypotheses (thus, excluding the one of interest).

Extra: How to specify restrictions

To evaluate the hypotheses of interest, it is necessary to specify the restrictions in these hypotheses correctly:

- Within the `goric` function, it is possible to use the following operators: `>`, `<`, `=`, `<=`, `>=`, `==` (where the last three denote the same constraint as the first three).
- The `goric` function can deal with:
  - pairwise restrictions separated by a semicolon `;` (e.g., *"beta1 > beta2; beta2 > beta3"*), a comma `;`, or a and-sign `&`.

– combined restrictions consisting of more than one operator (e.g., *"beta1 > beta2 > beta3"*).

Note that, in the code, one should use the labels of the parameter estimates (so, here: adjmean0 to adjmean2); while you should use population parameters (often denoted by Greek letters) in an article.

Extra: Failsafe / Safeguard hypothesis

To prevent from selecting a weak hypothesis, that is, a hypothesis not supported by the data, one should include a failsafe/safeguard hypothesis. This can be:

- the unconstrained hypothesis (which includes all possible hypotheses, thus including the one(s) of interest);
- the complement (which includes all other possible hypotheses, thus excluding the one(s) of interest).

The unconstrained is used to prevent from choosing a weak hypothesis. When at least one of the theory-based hypotheses has more support than the unconstrained then it is not weak. In that case, the non-weak hypotheses can be compared to all the other theory-based hypotheses in the set.
The complement is also a failsafe hypothesis, but acts as a competing hypothesis, that is, theory-based hypotheses can/will be compared to the complement (as well).
The first option is the default; since, currently, the complement can only be used for one hypothesis of interest. However, the relative support of two informative hypotheses is independent from the choice of failsafe hypothesis. So, if the goal is to compare informative hypotheses, one could just as well use the unconstrained.

Apply GORICA

GORICA

To apply the GORICA to the EffectLiteR adjusted means estimates, use:

```
set.seed(123) # More information below
gorica_AdjMeans <- goric(est_AdjMeans, VCOV=VCOV_AdjMeans, hypotheses=list(H1=H1), comparison = 'comple
#summary(gorica_AdjMeans)
gorica_AdjMeans
```

```
restriktor (0.5-90): generalized order-restricted information criterion approximation:

Results:
        model  loglik  penalty   gorica  loglik.weights  penalty.weights  gorica.weights
1          H1   7.005    1.846  -10.318           0.505            0.692           0.696
2  complement   6.985    2.654   -8.663           0.495            0.308           0.304
---
The order-restricted hypothesis 'H1' has 2.29 times more support than its complement.
```

From the output, one can conclude that the order-restricted hypothesis H_1 has 2.3 times more support than its complement. However, also note that the fit (i.e., loglik) part of H_1 and its complement are close. This indicates that there could be support for a boundary; where one or more inequalities are set to equalities. For more information on this and also on how to proceed, see the 'Guidelines_output_GORIC.html' on https://github.com/rebeccakuiper/Tutorials.

Extra: Specify type of method

The `goric` function calculates the *GORIC* value by default (type = "goric"). To calculate the *GORICA* values, the argument `type` has to be set to `gorica` (type = "gorica"). However, when the input is the estimates with their covariance matrix (like here), then `restriktor` will recognize that the GORICA should be used.

Extra: Seed values

In the calculation of the GORIC, an iterative process is used to calculate the penalty / complexity part. Therefore, one needs to set a seed value using the `set.seed`. This has two advantages:

1. Using the same seed value leads to the same penalty value every time this code is run (also helpful for reproducibility).
2. Using different seed values, allows for sensitivity check on the penalty value. If it is sensitive, then increase number of iterations used in calculation of the penalty.

Example 2: Average Effects

When investigating average effects, almost the same steps need to be performed. For more details, see the description above.

Inspect the data

To inspect the dataset, use the `head` function:

```
head(data_AvEffects) # Look at first (6) rows of the data
```

```
          y z x
1 130.98478 2 2
2 139.44400 2 2
3  91.26276 1 2
4 106.05693 1 2
5 119.74821 1 0
6  91.58844 0 2
```

Here, it can be seen that there are only 3 variables with one continuous dependent variable (y) and two categorical variables, where x is the independent variable and z the categorical covariate.

To see a more detailed overview of the data via descriptive statistics split by `group` variable, use the `describeBy` function with `data_AvEffects$x` set to be a grouping variable:

```
descrstat <- describeBy(data_AvEffects$y, data_AvEffects$x, mat = TRUE, digits = 3)
descrstat
```

```
    item group1 vars   n    mean     sd  median trimmed    mad    min     max   range   skew kurtosis
X11    1      0    1 179 110.638 25.604 112.538 111.346 21.500 32.659 176.557 143.898 -0.267    0.586 1
X12    2      1    1 176  99.418 20.498  98.627  99.148 20.406 40.746 146.339 105.592  0.071   -0.376 1
X13    3      2    1 145 119.208 29.991 120.077 119.423 33.784 49.264 183.064 133.799 -0.098   -0.554 2
```

Preparation for GORIC(A)

Use EffectLiteR

To use the estimates for the average effects with GORICA, an object with the main function `effectLite` needs to be created:

```
model_AvEffects <- effectLite(y="y", x="x", k="z", data=data_AvEffects, method="sem")
```

Note that now the argument `z` is no longer used, but the argument `k` is:

k: Vector of manifest variables treated as categorical covariates (character vector).

Extract information from the EffectLiteR object

For this example, we are interested in the average effect between the two categorical variables:

```
parnames <- c("Eg1","Eg2")
est_AvEffects <- model_AvEffects@results@est[parnames]
VCOV_AvEffects <- model_AvEffects@results@vcov.def[parnames,parnames]

est_AvEffects
```

```
      Eg1       Eg2
0.8802244 6.0841473
```

```
VCOV_AvEffects
```

```
         Eg1       Eg2
Eg1 6.784013  4.846823
Eg2 4.846823 10.614931
```

Set of hypotheses

In the example, the following could be hypothesized regarding the average effects (before seeing the data):

```
H1 <- "Eg1 < Eg2"
# Here, we say that the first average effect is smaller than the second average effect
```

This hypothesis will be inspected versus its complement, meaning all other possible orderings / hypotheses (thus, excluding the one of interest).

Apply GORICA

To apply the GORICA to the EffectLiteR average effect estimates, use:

```
set.seed(123)
gorica_AvEffects <- goric(est_AvEffects, VCOV=VCOV_AvEffects, hypotheses=list(H1=H1), comparison = 'com
#summary(gorica_AvEffects)
gorica_AvEffects
```

```
restriktor (0.5-90): generalized order-restricted information criterion approximation:

Results:
        model  loglik  penalty  gorica  loglik.weights  penalty.weights  gorica.weights
1          H1  -3.779    1.500  10.558           0.853            0.500           0.853
2  complement  -5.536    1.500  14.072           0.147            0.500           0.147
---
The order-restricted hypothesis 'H1' has 5.80 times more support than its complement.
```

From the output, one can conclude that the order-restricted hypothesis H_1 has 5.8 times more support than its complement. Notably, since the fit parts clearly differ, there is no indication for support for the boundary (see the 'Guidelines_output_GORIC.html' on https://github.com/rebeccakuiper/Tutorials).