Tutorial for updating hypotheses using the GORIC(A)

Rebecca M. Kuiper

09 februari 2024

Contents

This is a tutorial for updating hypotheses using the GORIC(A). It uses the goric() function of the restriktor package to calculate the GORIC and the GORICA.

Updating hypotheses, consists of the following three steps:

1. Conduct (exploratory) model selection to the first study.
2. Specify the (confirmatory/informative) hypotheses for the second/replication study using the results from Step 1, that is, update your hypotheses.
3. Conduct model selection to the second study.

Two examples will be used:

1. The first is based on Monin and Holubar, a replication study of Monin. Here, the study of Monin is used to update the exploratory hypotheses evaluated in Monin to theory-based, order-restricted hypotheses to be evaluated in Holubar. By examining these, one can check whether Holubar replicated the findings of Monin or not. Both these studies use an ANOVA model.
2. The second example is based on Altinisik et al.. Here, theory-based hypotheses are based on theories and on findings of previous studies. In this example, the GORICA is applied to the results of a logistic regression model.

At the end, there is code to automatically create all possible exploratory hypotheses, which can be helpful when there are a lot of parameters.

Example 1: ANOVA

In this example, the study of Monin will be used to render order-constrained hypotheses for the Holubar study, which is a (direct) replication study of Monin. For this, an exploratory analysis will be performed on the Monin data and inspect the sample means.

Step 1: Model selection 1st study

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("psych")) install.packages("psych")
if (!require("restriktor")) install.packages("restriktor")

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

# 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.

Notably, it is only possible to load the data if you are using the correct working directory (with both your R script and data file). The command `getwd()` shows you your current working directory. You can change the working directory to the one you prefer using the function `setwd()` by specifying the correct location between parentheses. Alternatively, in Rstudio, you can use the "Session" tab (on top) or you can use the "Files"-pane (on top of probably the right lower box of your Rstudio-screen, this pane is located next to the panes for "Plots", "Packages", "Help" and "Viewer").

If you open the data file `Data_Monin.txt` in a text editor, you can see that the variable labels have been inserted (using quotes; i.e., "...") in the first line of the file, which is called a header. Therefore, you have to specify 'header = TRUE' when loading the data:

```
# Load the data
Monin <- read.table("data/Data_Monin.txt", header = TRUE)
```

Since a .txt file was loaded, R does not know the measurement levels of the variables and assumes all of them to be continuous, meaning that they are of interval or ratio type. Hence, especially when there are more than two groups, one has to tell R that the variable `group` is a factor by using the `factor()` function on the `group` variable (i.e., a grouping / categorical / nominal variable):

```
# Make the variable group a factor
Monin$group <- factor(Monin$group)
```

To inspect the first 6 rows of the dataset, use the `head()` function:

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

```
      attract group
1  1.19560213     1
2  0.05852551     1
3 -1.10092218     1
4  1.78095312     1
5  1.86726943     1
6  1.12279975     1
```

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

```
descrstat <- describeBy(Monin$attract, Monin$group, mat = TRUE, digits = 3)
descrstat
```

```
    item group1 vars  n mean   sd median trimmed   mad    min   max range  skew kurtosis    se
X11    1      1    1 19 1.88 1.38  1.781   1.869 0.868 -1.101 5.050 6.151 0.201    0.135 0.317
X12    2      2    1 19 2.54 1.95  2.711   2.507 1.931 -1.099 6.746 7.845 0.024   -0.514 0.447
X13    3      3    1 29 0.02 2.38  0.438  -0.008 2.897 -3.911 4.916 8.828 0.028   -1.034 0.442
```

Preparation for GORIC(A)

ANOVA model: R-object First, an R-object with unconstrained estimates is needed, that is, in this example, the three group means and one residual variance. The linear regression model using `lm()` function is specified as follows:

```
lm_fit_Monin <- lm(attract ~ group - 1, data = Monin)
```

Note that:

1. `y ~ group - 1` instructs the function `lm` (linear model) to regress y on group.
2. The `- 1` instructs the function `lm` to drop the intercept and, therefore, estimate the means of each group, resulting here in five group means. On the other hand, if the intercept is not dropped, 'y ~ group' would estimate an intercept, representing the mean of the reference group, and the mean differences between the other (here, two) groups and the reference group.
3. The results are collected in, what is called, an R-object, named `lm_fit_Monin`.

It can be helpful to check the names used in this model, because these are needed when specifying the hypotheses:

```
names(coef(lm_fit_Monin))
```

```
[1] "group1" "group2" "group3"
```

Set of (exploratory) hypotheses On the Monin data set, an exploratory analysis will be performed, which means evaluating all combinations with equalities (and no restrictions). To specify hypotheses, remember the following:

- Within the `restriktor()` and `goric()` functions, it is possible to use the following operators: `>`, `<`, `=`, `<=`, `>=`, `==` (where the last three denote the same constraint as the first three).
- The `goric()` and the `restriktor()` functions can deal with:
  - pairwise restrictions separated by a semicolon `;` (e.g., *"beta1 > beta2; beta2 = beta3"*).
  - combined restrictions consisting of more than one operator (e.g., *"beta1 > beta2 = beta3"*).

  Note that one should use the labels of the parameter estimates (in the example above: group1-group3).
- One can also define hypothesis in terms of linear functions of parameters (For more details, see 'Extra possibility specification hypotheses' near the end of the goric() tutorial called 'Tutorial_GORIC_restriktor_General').

All possible exploratory hypotheses are denoted by:

```
H00 <- "group1 = group2 = group3"
H01 <- "group1 = group2"
H02 <- "group1 = group3"
H03 <- "group2 = group3"
```

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),

where the first option is the default. Notably, currently, the complement can only be used for one hypothesis of interest.

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.
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 (see below).

Model selection using GORIC(A)

```
set.seed(123)  # Set seed value
goric(lm_fit_Monin, hypotheses = list(H00 = H00, H01 = H01, H02 = H02, H03 = H03))
```

Calculate GORIC values and weights

```
restriktor (0.5-50): generalized order-restricted information criterion:

Results:
           model    loglik  penalty    goric  loglik.weights  penalty.weights  goric.weights
1            H00  -149.907    2.000  303.815           0.000            0.447          0.000
2            H01  -141.191    3.000  288.383           0.369            0.164          0.610
3            H02  -145.404    3.000  296.809           0.005            0.164          0.009
4            H03  -148.907    3.000  303.815           0.000            0.164          0.000
5  unconstrained  -140.665    4.000  289.330           0.625            0.060          0.380
```

Calculate GORICA values and weights In case you want to use the GORICA values and weights instead, use: type = "gorica". Thus:

```
set.seed(123)  # Set seed value
goric(lm_fit_Monin, hypotheses = list(H00 = H00, H01 = H01, H02 = H02, H03 = H03),
    type = "gorica")
```

Step 2: Update hypotheses

It can be seen that $H_0 1$ ($\mu_1 = \mu_2, \mu_3$) receives the most support. From the sample means (see `descrstat`), it is concluded that μ_1 and μ_2 are both larger than μ_3. Therefore, the following hypothesis will be evaluated in the Holubar data:

$$H_1 : \mu_1 = \mu_2 > \mu_3.$$

Since H_u obtained some support as well, one could specify (using the sample means of the Monin data) the following competing hypothesis:

$$H_2 : \mu_2 > \mu_1 > \mu_3.$$

Step 3: Model selection 2nd study

Data preparation

First, read in the Holubar dataset (the replication study of Monin), and tell R that the variable `gr` (group) is a factor instead of a continuous variable (although it is not necessary because it consists of only two groups).

```
Holubar <- read.table("data/Data_Holubar.txt", header = TRUE)  # load the data
Holubar$gr <- factor(Holubar$gr)  # tell R that gr is a factor
```

If you want a more detailed overview of the data, also by means of descriptive statistics splitted by `group`, use

```
head(Holubar)
```

```
         at gr
1 0.5549239  1
2 3.6167880  1
3 0.8071903  1
4 1.2733173  1
5 2.3898220  1
6 0.1910118  1
```

```
descrstat <- describeBy(Holubar$at, Holubar$gr, mat = TRUE, digits = 3)
descrstat
```

```
    item group1 vars  n mean   sd median trimmed   mad    min   max range   skew kurtosis    se
X11    1      1    1 20 0.98 1.20  1.177   0.930 1.118 -0.719 3.617 4.335  0.190   -0.778 0.268
X12    2      2    1 27 0.02 1.88  0.169   0.111 1.951 -4.961 2.921 7.883 -0.517   -0.104 0.362
X13    3      3    1 28 0.27 1.72 -0.099   0.203 2.435 -2.276 3.796 6.073  0.292   -1.025 0.325
```

Preparation for GORIC(A)

ANOVA model: R-object Then, fit an ANOVA-model by means of the `lm()` function (linear model) and directly check the names that are used in this model:

```
lm_fit_Holubar <- lm(at ~ gr - 1, data = Holubar)
names(coef(lm_fit_Holubar))
```

```
[1] "gr1" "gr2" "gr3"
```

Set of hypotheses Based on the results and sample means of Monin, the following two competing hypotheses were created:

```
H1 <- "gr1 = gr2 > gr3"
H2 <- "gr2 > gr1 > gr3"
```

Model selection using GORIC(A)

```
set.seed(123)  # Set seed value
output_repl <- goric(lm_fit_Holubar, hypotheses = list(H1, H2))
summary(output_repl)
```

Calculate GORIC values and weights

```
restriktor (0.5-50): generalized order-restricted information criterion:

Results:
           model    loglik  penalty    goric  loglik.weights  penalty.weights  goric.weights
1             H1  -144.981    2.500  294.962           0.111            0.510          0.303
2             H2  -144.981    2.803  295.569           0.111            0.376          0.224
3  unconstrained  -143.038    4.000  294.076           0.777            0.114          0.473
---

Ratio GORIC-weights:
```

```
               vs. H1  vs. H2  vs. unconstrained
H1              1.000   1.355              0.642
H2              0.738   1.000              0.474
unconstrained   1.557   2.109              1.000
---

Ratio loglik-weights:
               vs. H1  vs. H2  vs. unconstrained
H1              1.000   1.000              0.143
H2              1.000   1.000              0.143
unconstrained   6.978   6.978              1.000
---

Ratio penalty-weights:
               vs. H1  vs. H2  vs. unconstrained
H1              1.000   1.355              4.482
H2              0.738   1.000              3.309
unconstrained   0.223   0.302              1.000
---

order-restricted hypotheses:

H1:
gr1 = gr2 > gr3

H2:
gr2 > gr1 > gr3
```

Since the support for H_1 and H_2 is lower than for H_u, it can be concluded that both are weak hypotheses. Hence, the study of Holubar did not replicate the findings of Monin.

Alternative set In case you are only interested in the 'main hypothesis' H_1 found in Monin, you could also evaluate this against its complement:

```
set.seed(123)  # Set seed value
goric(lm_fit_Holubar, hypotheses = list(H1), comparison = "complement")
```

```
restriktor (0.5-50): generalized order-restricted information criterion:

Results:
        model    loglik  penalty    goric  loglik.weights  penalty.weights  goric.weights
1          H1  -144.981    2.500  294.962           0.125            0.731          0.280
2  complement  -143.038    3.500  293.076           0.875            0.269          0.720
---
The order-restricted hypothesis 'H1' has 0.390 times more support than its complement.
```

Since H_1 has only 0.39 (lower than 1) times more support than its complement, it is a weak hypothesis. Hence, the study of Holubar did not replicate the findings of Monin.

Calculate GORICA values and weights Notably, in case you want to use the GORICA, you can use the following commands:

```
set.seed(123)  # Set seed value
goric(lm_fit_Holubar, hypotheses = list(H1, H2), type = "gorica")
```

When you want to calculate the GORICA for H_1 and its complement, use:

```
set.seed(123)  # Set seed value
goric(lm_fit_Holubar, hypotheses = list(H1), type = "gorica", comparison = "complement")
```

Example 2: Logistic regression example

In this example, the order-constrained hypotheses for a replication data set are based on the results of the original study and based on theories from two previous studies. More details can be found in Altinisik et al. (2021). Since the data cannot be shared, the estimates of the parameters of interest and their covariance matrix are used, which can serve as input for the GORICA.

Load libraries

Start with loading 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 package, if you have not done this already:
if (!require("restriktor")) install.packages("restriktor")

## Then, load the packages:
library(restriktor)  # for the goric function

# 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
```

Load data

The estimates of structural parameters (i.e., the parameters of interest, that is, the parameters used in the hypotheses) are

```
est <- c(0.4765484, 0.2759022, 0.8669835, -0.443003, 0.8066336, -0.246261)
names(est) <- c("RS_Gr1", "RSES_Gr1", "RS_Gr2", "RSES_Gr2", "RS_Gr3", "RSES_Gr3")
```

Notably, the parameters need to be labelled such that those can be used in specifying the hypotheses.

Their covariance matrix is:

```
VCOV <- matrix(c(0.1736427, -0.1736427, 0, 0, 0, 0, -0.1736427, 0.6325835, 0, 0,
    0, 0, 0, 0, 0.2094106, -0.2094106, 0, 0, 0, 0, -0.2094106, 0.3740883, 0, 0, 0,
    0, 0, 0, 0.1351709, -0.1351709, 0, 0, 0, 0, -0.1351709, 0.1829996), byrow = TRUE,
    ncol = 6)
```

Notably, the covariance matrix does not need to be labelled.

Set of (updated) hypotheses

As discussed in Altinisik et al. (2021), the set of hypotheses is based on theories which are updated by findings from previous studies. Note that the same names/labels as the ones above are used.

```
H1 <- "RS_Gr1 + RSES_Gr1 = 0; RS_Gr1 > 0;
       RS_Gr2 + RSES_Gr2 = 0; RS_Gr2 > 0;
       RSES_Gr3 = 0; RS_Gr3 > 0"
H2 <- "RS_Gr1 + RSES_Gr1 = 0; RS_Gr1 > 0;
```

```
       RSES_Gr2 = 0; RS_Gr2 = 0;
       RSES_Gr3 = 0; RS_Gr3 > 0"
```

Model selection using GORICA

Next, the GORICA values and weights are calculated:

```
set.seed(123)
output_gorica_Altinisik <- goric(est, hypotheses = list(H1, H2), VCOV = VCOV, type = "gorica")
summary(output_gorica_Altinisik)
```

```
restriktor (0.5-50): generalized order-restricted information criterion approximation:

Results:
           model  loglik  penalty  gorica  loglik.weights  penalty.weights  gorica.weights
1             H1  -1.373    1.500   5.746           0.202            0.376           0.760
2             H2  -3.168    1.000   8.335           0.034            0.620           0.208
3  unconstrained  -0.045    6.000  12.089           0.764            0.004           0.032
---

Ratio GORICA-weights:
               vs. H1  vs. H2  vs. unconstrained
H1              1.000   3.650             23.847
H2              0.274   1.000              6.534
unconstrained   0.042   0.153              1.000
---

Ratio loglik-weights:
               vs. H1  vs. H2  vs. unconstrained
H1              1.000   6.018              0.265
H2              0.166   1.000              0.044
unconstrained   3.775  22.715              1.000
---

Ratio penalty-weights:
               vs. H1  vs. H2  vs. unconstrained
H1              1.000   0.607             90.017
H2              1.649   1.000            148.413
unconstrained   0.011   0.007              1.000
---

order-restricted hypotheses:

H1:
RS_Gr1 + RSES_Gr1 = 0; RS_Gr1 > 0;
RS_Gr2 + RSES_Gr2 = 0; RS_Gr2 > 0;
RSES_Gr3 = 0; RS_Gr3 > 0

H2:
RS_Gr1 + RSES_Gr1 = 0; RS_Gr1 > 0;
RSES_Gr2 = 0; RS_Gr2 = 0;
RSES_Gr3 = 0; RS_Gr3 > 0
```

It can be seen that both H_1 and H_2 are not weak hypotheses and that H_1 is the preferred one: It is 3.65 times more supported than H_2. Hence, the support for these theories is also found in this replication and,

based on this replication study, it is concluded that H_1 is more supported.

Extra: Generating all exploratory (=) restrictions

```
# make columns per number of variables in hypothesis
h <- paste(names(coef(lm_fit_Monin)), " = 0")
col <- NULL
for (i in 1:length(h)) {
    col <- c(col, combn(h, i, FUN = function(x) {
        return(paste(x, collapse = "; "))
    }))
}
hypotheses <- data.frame(col)
# make dataframe without the missings and split in two columns
if (!require("tidyr")) install.packages("tidyr")
```

```
Loading required package: tidyr
```

```
library(tidyr)
hypotheses <- gather(hypotheses, na.rm = TRUE, factor_key = TRUE)
# all hypothesis in text, by R
hypotheses$value
```

```
[1] "group1  = 0"                          "group2  = 0"                          "group3  = 0"
```

```
# number of hypotheses
nrhypos <- sum(!is.na(hypotheses$value))
# create all hypotheses H01 to H0nrhypos
for (i in 1:nrhypos) {
    assign(paste("H0", i, sep = ""), hypotheses$value[i])
}
# Now all the exploratory (=) hypotheses are specified

# In case you want to see them all, paste the following result in the R
# console:
cat(paste0("H0", 1:nrhypos, ";"))
```

```
H01; H02; H03; H04; H05; H06; H07;
```

```
# and past result in R console and you see them.

# paste result from following into goric function
cat(paste0("H0", 1:nrhypos, ","))
```

```
H01, H02, H03, H04, H05, H06, H07,
```