

Tutorial for `goric()`: How to evaluate theory-based hypotheses using the GORIC and GORICA

Rebecca M. Kuiper

07 September 2023

Contents

Introduction	1
Analyses	1
Hypothesis specification	2
Preparation	3
Model Selection using GORIC(A)	3
GORIC & GORICA	3
GORICA	11
Standardized data	16
Defining parameters	17
Types of comparisons	19
Comparison = “none”	19
Comparison = “unconstrained”	20
Comparison = “complement”	21
Different types of output	23
If many restrictions: Another method to calculate the penalty	26
Note on not full row-rank	26

Introduction

Analyses

Below, you will find examples for the use of the `goric()` function in the `restriktor`-package.

Two types of analyses can be run:

- `type = "goric"` (the default), which can be applied to multivariate normal linear models;
- `type = "gorica"`, the approximation of the GORIC which can be applied to a broad range of models.

Furthermore, there is the argument `comparison =` for the following three types of comparisons:

- `"none"`
- `"unconstrained"` (default)
- `"complement"`

If you choose "none", only the hypotheses of interest are inspected. This can lead to choosing the best out of a set of weak hypotheses. Therefore, this is only recommended when the hypotheses of interest cover the full parameter space / cover the whole set of possible theories. If "unconstrained" is chosen, then the unconstrained / unrestricted / classical alternative hypothesis is included in the set. This safeguards from choosing a weak hypothesis, that is, a hypothesis not supported by the data, as the best one. If at least one of the theory-based / informative hypotheses is not weak, one can compare those to all other hypotheses in the set. Currently, "complement" only works for one hypothesis and not for a whole set. Then, the complement of the hypothesis of interest is evaluated and acts like a competing hypothesis. Since there is per definition no overlap between the hypothesis and its complement, this is more powerful than including the unconstrained hypothesis. Please note that in the example below, the default `comparison = "unconstrained"` is used. At the end of the file, there are three examples for the three types of comparisons.

The `goric()` function of `restriktor` takes different forms of input (for model parameter estimates and constraints).

- (1) Fitted unconstrained (lm or glm) object* + character constraints.
- (2) Fitted unconstrained (lm or glm) object + list with constraints matrix.
 - In case of within effects, the GORIC cannot be calculated.
- (3) Fitted `restriktor` object(s).
- (4) Numeric vector + character constraints.
- (5) Numeric vector + list with constraints matrix.

Please note that:

- The last two (i.e., using the numeric factor) can only be used when `type = "gorica"` (the first three can be used for both `goric` and `gorica`).
- Using the numeric factor can be especially helpful for objects other than (g)lm objects.
- The numeric vector contains parameter estimates, and these can be obtained from any type of model.
- The estimates are assumed to be normally distributed. So, for some models when sample size is low, this assumption may not hold. In that case, it is often not clear how well the GORICA performs. See Altinisik et al 2021, for some simulations regarding logistic regression and SEM models.

Hypothesis specification

A few remarks on how to specify hypotheses when working with the `goric()` function. When one uses `character constraints`:

- 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.
- One can also define hypothesis in terms of linear functions of parameters (For more details, see ‘Extra possibility specification hypotheses’ near the end of this document).

When one wish to use a list with constraints matrix (possibly together with `rhs` and `neq`), it should be done as follows: `list(constraints = xxx, rhs = xxx, neq = x)`, with `constraints` being a matrix, `rhs` a vector with the right hand side of the restrictions, where the length of the vector equals the number of rows of constraints matrix (when not all of them are zero), and `neq` a scalar denoting the number of equalities in the matrix (when not zero). By default, `rhs` and `neq` are set to 0. These arguments have to be specified such that: `+ constraints * parameters >= rhs`, + where the first `neq`-restrictions are equalities.

In the following examples, the above methods are illustrated.

Preparation

Before starting the analyses, it is necessary to install and load the packages. Note that one needs to install the package only once, but has to load the package each time the R script is run.

```
# 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
#
# If from CRAN:
if (!require("restriktor")) install.packages("restriktor")
library(restriktor) # for goric function
```

Next, data is generated as example data to illustrate the functionality of the `goric()` function. If you want to obtain the same results as obtained here, it is necessary to set the seed at a fixed number.

```
set.seed(123)

n <- 10          # sample size
x1 <- rnorm(n)   # predictor 1, where n random draws are taken from a standard
                  # normal distributed population
x2 <- rnorm(n)   # predictor 2
x3 <- rnorm(n)   # predictor 3
e <- rnorm(n)    # error term
y <- 1 + x1 + x2 + x3 + e # outcome, obtained by filling in the regression
                        # equation with an intercept of 1 and
                        # regression coefficients of 1.
```

Then, a regression model can be fit using the `lm()` function:

```
fit.lm <- lm(y ~ x1 + x2 + x3)
```

Below, the GORIC and GORICA are applied to this model.

To apply the GORIC(A), hypotheses need to be specified. Normally, these would be based on theory. However, since (hypothetical) data is generated, quasi-randomly two hypotheses are specified.

Notes on specifying hypotheses:

- Hypotheses in R are in terms of the labeling used for the estimates (instead of the names for the population parameters, say, β_0, \dots, β_3). Thus, here: `.Intercept.`, `x1`, `x2`, and/or `x3`.
- In case estimates of continuous predictor variables are compared (e.g., `"x1 < x2"`), one should standardize the data (this is shown in more detail below) or use the standardized parameter estimates.

Model Selection using GORIC(A)

GORIC & GORICA

The upcoming analyses can be performed by means of both the GORIC and the GORICA. By default, the `goric()` function calculates the **GORIC** values (which then uses: `type = "goric"`). To calculate the **GORICA** values, the argument `type` has to be set to `gorica` (i.e., `type = "gorica"`) within the `goric()` function.

Note on GORIC and GORICA weights:

- The GORICA weights (asymptotically) equal the GORIC weights. The differences are minor and often not notable with 2 decimals. Because of these minor differences, the relative weights (i.e., ratio of weights) can differ. These differences in relative weights can even be large, when dividing a very large number by a very small number with minor differences in these values.

Fitted unconstrained (g)lm object + character constraints

Next, the hypotheses need to be specified. Note that the ; separates the restrictions within one hypothesis.

```
H1 <- "x1 > 0"
H2 <- "x1 > 0; x2 > 0"
```

Now, the GORIC values and weights can be calculated.

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

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	0.333	0.284	0.284
2	H2	-3.295	3.847	14.284	0.333	0.545	0.545
3	unconstrained	-3.295	5.000	16.589	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORIC-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:
x1 > 0

H2:

```
x1 > 0; x2 > 0
```

To obtain the GORICA values and weights instead:

```
set.seed(123) # Set seed value
out_gorica <- goric(fit.lm, hypotheses = list(H1, H2), type = "gorica")
summary(out_gorica)
```

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.500	-0.244	0.333	0.284	0.284
2	H2	3.622	2.847	-1.549	0.333	0.545	0.545
3	unconstrained	3.622	4.000	0.756	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:

x1 > 0

H2:

x1 > 0; x2 > 0

Fitted unconstrained (g)lm object + list with constraints matrix

In the example above, `rhs` and `neq` are assumed to be 0, which is also the assumed default. It means that the left hand side of (in)equalities are set to 0's, and that the number of equality restrictions is equal to 0.

```
H1 <- list(constraints = c(0,1,0,0))
# "x1 > 0"
H2 <- list(constraints = rbind(c(0,1,0,0), c(0,0,1,0)))
# "x1 > 0; x2 > 0"
```

The GORIC values and weights are obtained via:

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

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	0.333	0.284	0.284
2	H2	-3.295	3.847	14.284	0.333	0.545	0.545
3	unconstrained	-3.295	5.000	16.589	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORIC-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

and the GORICA values and weights via:

```
set.seed(123) # Set seed value
out_gorica <- goric(fit.lm, hypotheses = list(H1, H2), type = "gorica")
summary(out_gorica)
```

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.500	-0.244	0.333	0.284	0.284

2	H2	3.622	2.847	-1.549	0.333	0.545	0.545
3	unconstrained	3.622	4.000	0.756	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Fitted unconstrained (g)lm object + list with constraints matrix (rhs and neq non-zero)

In this example, `rhs` and `neq` are not assumed to be 0. Please note that equality restrictions should be stated first in the constraints' list.

```
H1 <- list(constraints = c(0,1,0,0), rhs = 2)
# "x1 > 2"
H2 <- list(constraints = rbind(c(0,1,0,0), c(0,0,1,0)), rhs = c(1,0), neq = 1)
# "x1 = 1; x2 > 0"
```

The GORIC values and weights are obtained via:

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

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-8.212	4.500	25.425	0.007	0.231	0.010
2	H2	-6.197	3.500	19.394	0.052	0.629	0.196
3	unconstrained	-3.295	5.000	16.589	0.941	0.140	0.795

Ratio GORIC-weights:

	vs. H1	vs. H2	vs. unconstrained
--	--------	--------	-------------------

H1	1.000	0.049	0.012
H2	20.399	1.000	0.246
unconstrained	82.896	4.064	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.133	0.007
H2	7.504	1.000	0.055
unconstrained	136.673	18.213	1.000

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.368	1.649
H2	2.718	1.000	4.482
unconstrained	0.607	0.223	1.000

and the GORICA values and weights via:

```
set.seed(123) # Set seed value
out_gorica <- gorica(fit.lm, hypotheses = list(H1, H2), type = "gorica")
summary(out_gorica)
```

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	-1.400	3.500	9.799	0.006	0.231	0.008
2	H2	1.262	2.500	2.477	0.086	0.629	0.295
3	unconstrained	3.622	4.000	0.756	0.908	0.140	0.697

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.026	0.011
H2	38.906	1.000	0.423
unconstrained	91.977	2.364	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.070	0.007
H2	14.313	1.000	0.094
unconstrained	151.645	10.595	1.000

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.368	1.649
H2	2.718	1.000	4.482
unconstrained	0.607	0.223	1.000

Fitted restriktor object(s)

After specifying the hypotheses:

```
H1 <- "x1 > 0"
H2 <- "x1 > 0; x2 > 0"
```

the restriktor objects can be fitted as follows:

```
fit1.restr <- restriktor(fit.lm, constraints = H1)
fit2.restr <- restriktor(fit.lm, constraints = H2)
```

Now, use the GORIC for model selection:

```
set.seed(123) # Set seed value
out <- goric(fit1.restr, fit2.restr)
summary(out)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	0.333	0.284	0.284
2	H2	-3.295	3.847	14.284	0.333	0.545	0.545
3	unconstrained	-3.295	5.000	16.589	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORIC-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:
x1 > 0

H2:
x1 > 0; x2 > 0

or the GORICA:

```
set.seed(123) # Set seed value
out_gorica <- gorica(fit1.restr, fit2.restr, type = "gorica")
summary(out)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	0.333	0.284	0.284
2	H2	-3.295	3.847	14.284	0.333	0.545	0.545
3	unconstrained	-3.295	5.000	16.589	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORIC-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:
x1 > 0

H2:
x1 > 0; x2 > 0

GORICA

The upcoming analyses can only be performed by means of the *GORICA*. This is because, the *GORICA* only needs the parameter estimates or only those of interest (that is, those addressed in the hypotheses) and their covariance matrix. Because of this, it is applicable for a wider range of models (e.g., logistic regression and SEM models).

Numeric vector + character constraints

One needs to extract the estimates (of interest), as well as their variance-covariance matrix. Both can be extracted from the `fit.lm` object (also from non-(g)lm objects).

```
est <- coef(fit.lm)
VCOV <- vcov(fit.lm)
```

Next, the hypotheses need to be specified using the labels of 'est' (which can also be re-labelled as is done in the next subsection).

```
H1 <- "x1 > 0"
H2 <- "x1 > 0; x2 > 0"
```

Then, use the `goric()` function to select the best model.

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

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.500	-0.244	0.333	0.284	0.284
2	H2	3.622	2.847	-1.549	0.333	0.545	0.545
3	unconstrained	3.622	4.000	0.756	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:

$x_1 > 0$

H2:

$x_1 > 0; x_2 > 0$

However, one could also use the structural parameters only, that is, the parameters used in all hypotheses in the set. In this case, the following code can be used:

```
H1 <- "x1 > 0"
H2 <- "x1 > 0; x2 > 0"

est <- coef(fit.lm)[2:3]
VCOV <- vcov(fit.lm)[2:3, 2:3]

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

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	1.614	1.500	-0.227	0.333	0.284	0.284
2	H2	1.614	0.847	-1.532	0.333	0.545	0.545
3	unconstrained	1.614	2.000	0.773	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:

$x_1 > 0$

H2:

$x_1 > 0; x_2 > 0$

Now, the exact same weights are obtained. However, the penalties are two points lower than before, which is due to the fact that two unconstrained/free parameters are left out. The log-likelihoods also change. But, when comparing the support for the hypotheses, these differences cancel out.

Relabeled numeric vector + character constraints

By default, the names/labels of the parameter estimates are based on variable names in the model (and in case of a factor, it is also based on the different levels it can take on). Using other names/labels often makes more sense. Then, one can specify the hypotheses in terms of those labels. For example, one can do the following:

```
est <- coef(fit.lm)
names(est) <- c("beta0", "beta1", "beta2", "beta3")
VCOV <- vcov(fit.lm)

H1 <- "beta1 > 0"
H2 <- "beta1 > 0; beta2 > 0"

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

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.500	-0.244	0.333	0.284	0.284
2	H2	3.622	2.847	-1.549	0.333	0.545	0.545
3	unconstrained	3.622	4.000	0.756	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:

beta1 > 0

H2:

beta1 > 0; beta2 > 0

‘Manual’ numeric vector + character constraints

Sometimes, one cannot extract the coefficients and variance-covariance matrix directly from an R-object; for example, due to the fact that previous analyses were performed by means of other software, or since it stems from another article. In these situations, it is also possible to manually insert the regression coefficients and the variance-covariance matrix.

```
est <- c(1.2479164, 1.4067410, 0.8260098, 0.8113972)

names(est) <- c("beta0", "beta1", "beta2", "beta3")

VCOV <- matrix(c(0.023513875, 0.002584345, 0.000504821, 0.011663312,
                 0.002584345, 0.035044625, -0.016490653, 0.004143737,
                 0.000504821, -0.016490653, 0.036437680, 0.016195388,
                 0.011663312, 0.004143737, 0.016195388, 0.036158130),
               byrow = TRUE, ncol = 4)

H1 <- "beta1 > 0"
H2 <- "beta1 > 0; beta2 > 0"

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

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.500	-0.244	0.333	0.284	0.284
2	H2	3.622	2.847	-1.549	0.333	0.545	0.545
3	unconstrained	3.622	4.000	0.756	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:

$\beta_1 > 0$

H2:

$\beta_1 > 0; \beta_2 > 0$

#Numeric vector + list with constraints matrix (rhs, neq)

When one only wants to use a list with a constraints matrix, the following code can be used:

```
est <- coef(fit.lm)
VCOV <- vcov(fit.lm)

H1 <- list(constraints = c(0,1,0,0))
# "x1 > 0"
H2 <- list(constraints = rbind(c(0,1,0,0), c(0,0,1,0)))
# "x1 > 0; x2 > 0"

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

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.500	-0.244	0.333	0.284	0.284
2	H2	3.622	2.847	-1.549	0.333	0.545	0.545
3	unconstrained	3.622	4.000	0.756	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORICA-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Standardized data

If estimates of continuous predictor variables are compared (e.g., “ $x_1 < x_2$ ”), then one should standardize the data. To demonstrate this, new data will be generated. In the data, the contribution of the variables to the dependent variable will differ. Below, the vector of ratios of these contributions is set to $= c(1, 1.5, 2)$. This, then reflects (the ratios of) the population regression coefficient values. It means that the contribution to the outcome variable y , increases from x_1 to x_3 , where that of x_2 is 1.5 and that of x_3 is 2 times as large as that of x_1 .

```
ratio <- c(1, 1.5, 2) # ratio between population effects of the predictors

n <- 30 # sample size

x1 <- rnorm(n) # predictor1:
               # n draws from a standard normal distributed population
x2 <- rnorm(n) # predictor2
x3 <- rnorm(n) # predictor3
e <- rnorm(n)

data <- cbind(x1, x2, x3)
data_s <- as.data.frame(scale(data))

y <- ratio[1] * data_s$x1 + ratio[2] * data_s$x2 + ratio[3] * data_s$x3 + e
```

Note. Since there is only one outcome, the outcome does not need to be standardized.

Now, fit the regression model, and specify a hypothesis by means of character constraints.

```
fit.lm_s <- lm(y ~ -1 + x1 + x2 + x3, data = data_s)

H1 <- "x1 < x2"
```

Then, calculate the GORIC values and weights:


```
set.seed(123) # Set seed value
out_s <- goric(fit.lm, hypotheses = list(H1), comparison = "complement")
summary(out_s, brief = FALSE)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-5.447	4.500	19.895	0.104	0.500	0.104
2	complement	-3.295	4.500	15.589	0.896	0.500	0.896

The order-restricted hypothesis 'H1' has 0.116 times more support than its complement.

Ratio GORIC-weights:

	vs. H1	vs. complement
H1	1.000	0.116
complement	8.607	1.000

Ratio loglik-weights:

	vs. H1	vs. complement
H1	1.000	0.116
complement	8.607	1.000

Ratio penalty-weights:

	vs. H1	vs. complement
H1	1	1
complement	1	1

Order-restricted coefficients:

	(Intercept)	x1	x2	x3
H1	1.236	1.120	1.120	0.878
complement	1.248	1.407	0.826	0.811

Restriction matrices:

```
$H1
  (Intercept) x1 x2 x3   op rhs active
1:           0 -1  1  0  >=  0    yes
```

```
$complement
[1] not H1
```

Defining parameters

In `restriktor`, it is also possible to specify hypotheses in terms of linear functions of the parameters. For instance, in case of adjusted means, you can specify the restrictions in two ways:

1. You can write them out yourself (using the parameter/variable names from an R-object, like `lm()`).

2. You can first specify the adjusted means (below called "m1" to "m4") and specify the hypotheses in terms of those:

```
H1 <- ' m1 := .Intercept.
      m2 := .Intercept. + facialBurns
      m3 := .Intercept. + gender
      m4 := .Intercept. + facialBurns + gender + gender:facialBurns
      m1 < m4
      m2 < m4
      m3 < m4 '
```

For more details, please see Vanbrabant, Van Loey, and Kuiper 2020). However, please note that this code cannot be run now, since the variables and, more importantly, the parameters are labeled differently. One can run:

```
H1 <- ' m1 := .Intercept.
      m2 := .Intercept. + x1
      m3 := .Intercept. + x2
      m4 := .Intercept. + x3
      m1 < m4
      m2 < m4
      m3 < m4 '

set.seed(123) # Set seed value
out <- goric(fit.lm, hypotheses = list(H1))
summary(out)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-6.610	3.879	20.978	0.035	0.754	0.100
2	unconstrained	-3.295	5.000	16.589	0.965	0.246	0.900

Ratio GORIC-weights:

	vs. H1	vs. unconstrained
H1	1.000	0.111
unconstrained	8.972	1.000

Ratio loglik-weights:

	vs. H1	vs. unconstrained
H1	1.000	0.036
unconstrained	27.532	1.000

Ratio penalty-weights:

	vs. H1	vs. unconstrained
H1	1.000	3.069
unconstrained	0.326	1.000

order-restricted hypotheses:

H1:

```

m1 := .Intercept.
m2 := .Intercept. + x1
m3 := .Intercept. + x2
m4 := .Intercept. + x3
m1 < m4
m2 < m4
m3 < m4

```

Types of comparisons

There are three types of comparisons: "none", "unconstrained", and "complement". Subsequently, it is shown, for the last type, what output can be asked for, which of course holds for all examples.

Comparison = "none"

If `comparison = "none"`, then only the hypotheses of interest are inspected. This can lead to choosing the best hypothesis out of a set of weak hypotheses. Hence, one should only use this if the hypotheses cover the whole parameter space, that is, cover all possible theories/hypotheses.

```

H1 <- "x1 > 0"
H2 <- "x1 > 0; x2 > 0"

set.seed(123) # Set seed value
out <- goric(fit.lm, hypotheses = list(H1, H2), comparison = "none")
summary(out)

```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	0.500	0.342	0.342
2	H2	-3.295	3.847	14.284	0.500	0.658	0.658

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORIC-weights:

	vs. H1	vs. H2
H1	1.000	0.521
H2	1.921	1.000

Ratio loglik-weights:

	vs. H1	vs. H2
H1	1	1
H2	1	1

Ratio penalty-weights:

	vs. H1	vs. H2
H1	1.000	0.521
H2	1.921	1.000

order-restricted hypotheses:

H1:

$x_1 > 0$

H2:

$x_1 > 0; x_2 > 0$

Comparison = “unconstrained”

If `comparison = "unconstrained"`, then the unconstrained / unrestricted / classical alternative hypothesis is included in the set. This safeguards from choosing a weak hypothesis as the best one.

```
H1 <- "x1 > 0"
H2 <- "x1 > 0; x2 > 0"

set.seed(123) # Set seed value
out <- goric(fit.lm, hypotheses = list(H1, H2), comparison = "unconstrained")
# or:
# out <- goric(fit.lm, H1, H2, type = "gorica", comparison = "unconstrained")
summary(out)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	0.333	0.284	0.284
2	H2	-3.295	3.847	14.284	0.333	0.545	0.545
3	unconstrained	-3.295	5.000	16.589	0.333	0.172	0.172

Note: If log-likelihood values are equal or close to each other, the goric ratio weights are determined

Ratio GORIC-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

Ratio loglik-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1	1	1
H2	1	1	1
unconstrained	1	1	1

Ratio penalty-weights:

	vs. H1	vs. H2	vs. unconstrained
H1	1.000	0.521	1.649
H2	1.921	1.000	3.167
unconstrained	0.607	0.316	1.000

order-restricted hypotheses:

H1:

$x_1 > 0$

H2:

$x_1 > 0; x_2 > 0$

Since at least one of, even both, H1 and H2 are not weak, one can compare the support for H1 versus H2 (where H2 is 1.921 times more supported).

Comparison = “complement”

Currently, `comparison = "complement"` only works for one hypothesis and not for a whole set of hypotheses. Then, the complement of the hypothesis of interest is evaluated and acts like a competing hypothesis. Since there is per definition no overlap between the hypothesis and its complement, this is more powerful than including the unconstrained.

H1 vs its complement

```
H1 <- "x1 > 0"

set.seed(123) # Set seed value
out <- goric(fit.lm, hypotheses = list(H1), comparison = "complement")
summary(out, brief = FALSE)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	1.000	0.500	1.000
2	complement	-15.009	4.500	39.018	0.000	0.500	0.000

The order-restricted hypothesis 'H1' has 122335.376 times more support than its complement.

Ratio GORIC-weights:

	vs. H1	vs. complement
H1	1.00e+00	1.22e+05
complement	0.00e+00	1.00e+00

Ratio loglik-weights:

	vs. H1	vs. complement
H1	1.00e+00	1.22e+05
complement	0.00e+00	1.00e+00

Ratio penalty-weights:

	vs. H1	vs. complement
H1	1	1

```
complement 1      1
---
```

Order-restricted coefficients:

```
      (Intercept)  x1    x2    x3
H1          1.248    1.407 0.826 0.811
complement  1.144    0.000 1.488 0.645
---
```

Restriction matrices:

```
$H1
      (Intercept) x1 x2 x3    op rhs active
1:              0  1  0  0   >=   0     no
```

```
$complement
[1] not H1
```

`summary(out)`

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	4.500	15.589	1.000	0.500	1.000
2	complement	-15.009	4.500	39.018	0.000	0.500	0.000

```
---
```

The order-restricted hypothesis 'H1' has 122335.376 times more support than its complement.

Ratio GORIC-weights:

	vs. H1	vs. complement
H1	1.00e+00	1.22e+05
complement	0.00e+00	1.00e+00

```
---
```

Ratio loglik-weights:

	vs. H1	vs. complement
H1	1.00e+00	1.22e+05
complement	0.00e+00	1.00e+00

```
---
```

Ratio penalty-weights:

	vs. H1	vs. complement
H1	1	1
complement	1	1

```
---
```

order-restricted hypotheses:

```
H1:
x1 > 0
```

H2 vs its complement

```
H2 <- "x1 > 0; x2 > 0"
```

```
set.seed(123) # Set seed value
```

```
out <- goric(fit.lm, hypotheses = list(H2), comparison = "complement")
summary(out)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	3.847	14.284	0.999	0.691	1.000
2	complement	-10.375	4.653	30.055	0.001	0.309	0.000

The order-restricted hypothesis 'H1' has 2658.773 times more support than its complement.

Ratio GORIC-weights:

	vs. H1	vs. complement
H1	1	2659
complement	0	1

Ratio loglik-weights:

	vs. H1	vs. complement
H1	1.000	1188.275
complement	0.001	1.000

Ratio penalty-weights:

	vs. H1	vs. complement
H1	1.000	2.238
complement	0.447	1.000

order-restricted hypotheses:

H1:

x1 > 0; x2 > 0

Note on comparisons

Please note that the obtained weights for the two hypotheses cannot be compared. If this is the goal of the analysis, one should evaluate them simultaneously in one set, as done earlier.

Different types of output

The command `out` gives the default results (i.e., per model / hypothesis the log-likelihood, penalty, GORIC(A), and GORIC(A)-weights).

```
out
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	3.847	14.284	0.999	0.691	1.000
2	complement	-10.375	4.653	30.055	0.001	0.309	0.000

The order-restricted hypothesis 'H1' has 2658.773 times more support than its complement.

The command `summary(out)` gives, besides the results, also the relative GORIC(A)-weights matrix.

```
summary(out)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	3.847	14.284	0.999	0.691	1.000
2	complement	-10.375	4.653	30.055	0.001	0.309	0.000

The order-restricted hypothesis 'H1' has 2658.773 times more support than its complement.

Ratio GORIC-weights:

	vs. H1	vs. complement
H1	1	2659
complement	0	1

Ratio loglik-weights:

	vs. H1	vs. complement
H1	1.000	1188.275
complement	0.001	1.000

Ratio penalty-weights:

	vs. H1	vs. complement
H1	1.000	2.238
complement	0.447	1.000

order-restricted hypotheses:

H1:

$x_1 > 0$; $x_2 > 0$

The command `summary(out, brief = FALSE)` gives also the order / inequality restricted coefficients (order-restricted mle's) and constraint matrices.

```
summary(out, brief = FALSE)
```

restriktor (0.5-20): generalized order-restricted information criterion:

Results:

	model	loglik	penalty	goric	loglik.weights	penalty.weights	goric.weights
1	H1	-3.295	3.847	14.284	0.999	0.691	1.000
2	complement	-10.375	4.653	30.055	0.001	0.309	0.000

The order-restricted hypothesis 'H1' has 2658.773 times more support than its complement.

Ratio GORIC-weights:

	vs. H1	vs. complement
H1	1	2659
complement	0	1

Ratio loglik-weights:

	vs. H1	vs. complement
H1	1.000	1188.275
complement	0.001	1.000

Ratio penalty-weights:

	vs. H1	vs. complement
H1	1.000	2.238
complement	0.447	1.000

Order-restricted coefficients:

	(Intercept)	x1	x2	x3
H1	1.248	1.407	0.826	0.811
complement	1.236	1.781	0.000	0.444

Restriction matrices:

```
$H1
  (Intercept) x1 x2 x3   op rhs active
1:           0  1  0  0  >=  0     no
2:           0  0  1  0  >=  0     no
```

\$complement

[1] not H1

Furthermore, one can ask for different elements of the output by means of the following commands:

- `out$result` - gives results (i.e., per model / hypothesis the log-likelihood, penalty, GORIC(A), and GORIC(A)-weights).
- `out$ratio.gw` - gives the relative GORIC(A)-weights matrix, which gives the relative support of an hypothesis versus another.
- `out$ormle` - gives the order / inequality restricted coefficients (order-restricted mle's).
- `coef(out)` - also gives the order / inequality restricted coefficients (order-restricted mle's).
- `out$type` - states what type of analysis is used (GORIC or GORICA).
- `out$comparison` - states what type of comparison is used (none, unconstrained, complement).
- `out$constraints` - gives the constraints matrices.
- `out$rhs` - gives the right hand side (rhs) of the constraints.

- `out$neq` - gives the number of equalities (neq) in the constraints, where the first neq are then equalities.

If many restrictions: Another method to calculate the penalty

Working with a lot of restrictions and a lot of parameters calculations increases the computation time, which has to do with the calculation of the level probability needed for the penalty term. There are two methods that can be used in calculating the penalty. The default method is often much faster (if the number of parameters is not too high) and needs less input specification. It can, however, not deal with hypotheses that are not of full row-rank (which easily happens if you specify many restrictions). In that case, **restriktor** uses automatically the other (bootstrap) method. In this bootstrap method, one can also more easily change the number of iterations on which the penalty is based (`'mix.bootstrap'`). The computation time of this bootstrap method can be reduced by using multiple cores. For a windows device, you then have to use `'parallel = "snow"'` (see the goric tutorial for more options). It is worth noting that setting up a local cluster can take some time if one is not provided. Therefore, in the example given below, using no parallel computing may actually be faster. However, the advantage of using parallel computing becomes apparent when dealing with a much larger number of bootstrap samples.

To use this bootstrap method (on a windows machine), use:

```
if (!require("parallel")) install.packages("parallel")
library(parallel)
nrCPUcores <- detectCores(all.tests = FALSE, logical = TRUE)

set.seed(123) # Set seed value
goric(est, VCOV = VCOV, hypotheses = list(H1), comparison = "complement",
      mix.weights = "boot", parallel = "no", ncpus = nrCPUcores, mix.bootstrap = 9999)
```

restriktor (0.5-20): generalized order-restricted information criterion approximation:

Level probabilities:

Number of requested bootstrap draws 9999
 Number of successful bootstrap draws for H1: 9999

Results:

	model	loglik	penalty	gorica	loglik.weights	penalty.weights	gorica.weights
1	H1	3.622	3.499	-0.246	1.000	0.501	1.000
2	complement	-24.612	3.501	56.227	0.000	0.499	0.000

The order-restricted hypothesis 'H1' has 1832665052464.442 times more support than its complement.

In case of bootstrapping, the following additional arguments are used:

- `mix.weights` - method for calculating the level probabilities (options: "none", "pmvnorm", "boot").
- `mix.bootstrap` - number of bootstrap samples.
- `parallel` - use of parallel processing. Windows: `parallel = 'snow'`; unix: `parallel = 'multicore'`.
- `ncpus` - number of processors in your PC.

This can also be helpful in case the restriction matrix is not full row-rank, as discussed next.

Note on not full row-rank

If the restriction matrix is not of full row-rank, this means one of the following:

1. Redundant restriction, either:
 - a) leave the redundant one out (`goric()` function does this for you), or:

- b) use another (more time-consuming) way of obtaining the level probabilities for the penalty term:
Bootstrapping, as discussed above (goric function does this by default).
- 2. Range restrictions (e.g., $-2 < \text{group1} < 2$), which can be evaluated, but there is a sensitivity (like with a prior in a Bayes factor), which currently cannot be checked for.
- 3. Conflicting restrictions (e.g., $2 < \text{group1} < -2$), which can evidently never hold and is thus impossible to evaluate. In this case, one should delete the incorrect restriction.