Package: GO (via r-universe)

August 25, 2024

Title Gaussian Ordination and Community Simulation

Version 0.4-0

Description Functions used to produce a manuscript on Unconstrained Gaussian Ordination.

Imports vegan, parallel, coenocliner (>= 0.2-0), stats, graphics

SystemRequirements tikz

Encoding UTF-8

License GPL-2

Repository https://jarioksa.r-universe.dev

RemoteUrl https://github.com/jarioksa/GO

RemoteRef HEAD

RemoteSha 25de8e42da373cc8eb6e0d92d3393f3286385192

Contents

`GO1` *Unconstrained Gaussian Maximum Likelihood Ordination*

Description

The functions fit unconstrained maximum likelihood ordination with unit-width Gaussian response models.

Usage

```
GO1(comm, tot = max(comm), freqlim = 5, parallel = 1, trace = TRUE, ...)

GO(comm, k = 1, tot = max(comm), freqlim = 5, family = c("poisson",
  "binomial"), far = 10, init, trace = TRUE, ...)

metaGO(comm, k = 1, trymax = 3, firstOK = TRUE, trace = FALSE, ...)

## S3 method for class 'GO'
plot(x, choices = 1, label = FALSE, marginal = FALSE,
  cex = 0.7, col = 1:6, ...)

## S3 method for class 'GO'
anova(object, ...)

spanodev(mod1, mod2 = NULL, ...)

## S3 method for class 'GO'
predict(object, newdata, type = c("response", "link"), ...)

## S3 method for class 'GO'
calibrate(object, newdata, ...)
```

Arguments

`comm`	Community data frame.
`tot`	Total abundance used in Binomial models. This can be either a single value for all data, or a vector with value for each row of `comm`. The default is to use the maximum value in matrix.
`freqlim`	Minimum number of occurrence for analysed species.
`parallel`	Number of parallel processes.
`trace`	logical; print information on the progress of the analysis.
`k`	Number of estimated gradients (axes).
`family`	Error distribution. Can be either `"poisson"` for quasi-Poisson or `"binomial"` for quasi-Binomial (and must be quoted).
`far`	Threshold distance for species optima regarded as alien and frozen in fitting.
`init`	Initial configuration to start the iterations. This should be a matrix, and number of rows should match the community data, and number coluns the number of gradients (`k`). The default is to use scores from `decorana`.
`trymax`	Maximum number of random starts.
`firstOK`	Do not launch random starts if default start succeeds.
`x`	Fitted model.
`choices`	The axis or axes plotted.
`label`	Label species responses.

`marginal`	Plot marginal responses or slice through origin of other dimensions.
`cex`	Character size for `labels`.
`col`	Colours of response curves.
`object`	Ordination result object.
`mod1, mod2`	Compared result objects
`newdata`	New gradient locations to `predict` species responses or new community data to `calibrate` gradient locations.
`type`	Predictions in the scale of responses or in the scale of link function.
`...`	Other parameters passed to functions. In `GO` these are passed to `nlm` and can include, e.g., `iterlim` (which often must be set to higher value than the default 100).

Details

Function is under development and unreleased. It will be released under different name in **vegan**. The current version is only provided for review purposes. The function and its support functions require **vegan**, although this requirements is not explicitly made. The optimization is based on `nlm` function, and passes arguments to this function.

Function `anova` can analyse two nested models or a single model against null model of flat responses using parametric tests based on quasi-Likelihood. Function `spanodev` performs similar test split by species. Function `predict` returns estimated response curves, and `newdata` can be gradient locations. Function `calibrate` returns estimated gradient locations, and `newdata` can be community data.

The `plot` function displays fitted respose curves against one ordination axis. In principle, the ordination can be rotated using **vegan** function `MDSrotate`, but this requires a version that agrees to analyse `GO` results. Traditional ordination plots of SU scores and species optima can be displayed with `ordiplot` (**vegan** package). The function is written so that several other **vegan** and standard R functions can handle results.

Functions

- `GO1`: Alternating estimation of species parameters and gradient locations in one dimension.
- `GO`: Simultaneous estimation of species parameters and gradient locations.
- `metaGO`: Start GO several times from random configurations if default start fails or optionally always
- `spanodev`: Comparison of goodness of fit for individual species.

Author(s)

Jari Oksanen

See Also

`cgo` in **VGAM** package.

Examples

```
library(vegan) ## *must* be given before using the function
data(varespec)
mod <- GO(varespec, k=2, far=5, tot=100, family="binomial", iterlim=1000)
plot(mod, label=TRUE)
ordiplot(mod, type="t")
ordiplot(mod, dis="si", type="t")
anova(mod)
mod1 <- update(mod, k=1)
anova(mod1, mod)
spanodev(mod1)
spanodev(mod1, mod)
```

GradLocs *Utility Functions for 'coenocliner' Package*

Description

Functions to automated simulation routines using **coenocliner** package.

Usage

```
GradLocs(n, xrange, yrange)

GradMul(xy, xmul, ymul)

BinomGaussPar(nsp, xrange, yrange, buffer = 2, tsd = 0.1)

Gauss2betaPar(gausspar, shape = c(0.5, 6.5), cover = 0.95)

DropMissingSpec(comm)

coenorun1(sim, tot = 1, family = "binomial", far = 4, trace = TRUE)
```

Arguments

`n`	Number of SUs.
`xrange, yrange`	Desired range of gradients.
`xy`	Gradient locations in two dimensions.
`xmul, ymul`	Multipliers for each gradient
`nsp`	Number of species.
`buffer`	Width of buffer zone for optima surrounding ranges.
`tsd`	Standard deviation of tolerance in log-Normal distribution, in log scale
`gausspar`	Gaussian response parameters for species as returned by `BinomGaussPar`.

`shape`	Random log-uniform range of shape parameters $alpha$ and $gamma$ of response function
`cover`	Find range of beta response so that the same span covers the same proportion of 1-dim integral as the Gaussian response function.
`comm`	Community data.
`sim`	One simulated community.
`tot`	Binomial total in `sim`.
`family`	Error family passed to `GO`.
`far`	Weirdness limit passed to `GO`.
`trace`	Print tracing information. If `FALSE` or `0`, work as silently as possible, and higher values print more.

Functions

- `GradLocs`: Gradient Locations
- `GradMul`: Multiply input gradient which presumably is a unit square
- `BinomGaussPar`: Gaussian Parameters for Binomial Response.
- `Gauss2betaPar`: Translate Gaussian parameters into corresponding beta response parameters.
- `DropMissingSpec`: Drop missing species from the data.
- `coenorun1`: Takes one simulated community for ordination with GO, NMDS, CA and DCA and returns average Procrustes precision

Author(s)

Jari Oksanen

Examples

```
require(coenocliner) || stop("examples need 'coenocliner' package")
## small simulation
nsim <- 10
npoints <- 50
## generate a set of species parameters over the maximum extent
sp <- replicate(nsim, BinomGaussPar(800, 8, 4))
## sample much narrower proportion of the space
xy <- replicate(nsim, GradLocs(npoints, 3, 2))
## Simulations: these can be easily parallelized using mclapply
## (Linux, Mac) or parSapply (all).
sapply(seq_len(nsim), function(i)
     coenorun1(coenocline(xy[,,i], "gaussian", sp[,i],
         countModel="bernoulli")))
```

Index