Package 'gllvm'

July 9, 2026

Type Package

Title Generalized Linear Latent Variable Models

Version 2.0.13

Date 2026-07-08

Maintainer Jenni Niku <jenni.m.e.niku@jyu.fi>

Description Analysis of multivariate data using generalized linear latent variable models (gllvm). Estimation is performed using either the Laplace method, variational approximations, or extended variational approximations, implemented via TMB (Kristensen et al. (2016), <doi:10.18637/jss.v070.i05>).

License GPL-2

Imports MASS, Matrix, fishMod, mgcv, alabama, nloptr, methods

Depends R (>= 3.5.0), TMB

Encoding UTF-8

LazyData true

LinkingTo TMB, Rcpp, RcppEigen

RoxygenNote 7.3.3

NeedsCompilation yes

URL https://jenniniku.github.io/gllvm/, https://github.com/JenniNiku/gllvm

BugReports https://github.com/JenniNiku/gllvm/issues

Suggests knitr, rmarkdown, testthat, gclus, corrplot, lattice, mvabund, ape, parallel

VignetteBuilder knitr

Author Jenni Niku [aut, cre], Wesley Brooks [aut], Riki Herliansyah [aut], Francis K.C. Hui [aut], Pekka Korhonen [aut], Sara Taskinen [aut], Bert van der Veen [aut], David I. Warton [aut]

Repository CRAN

Date/Publication 2026-07-09 07:50:02 UTC

Contents

AICc . . . 3
anova.gllvm . . . 3
beetle . . . 4
coefplot.gllvm . . . 6
confint.gllvm . . . 8
ecoCoefs . . . 9
eSpider . . . 9
fungi . . . 11
getEnvironCov.gllvm . . . 12
getLoadings.gllvm . . . 15
getLV.gllvm . . . 15
getPredictErr.gllvm . . . 16
getResidualCor.gllvm . . . 17
getResidualCov.gllvm . . . 18
gllvm . . . 21
glmmVA . . . 35
goodnessOfFit . . . 37
kelpforest . . . 39
logLik.gllvm . . . 40
microbialdata . . . 41
ordiplot.gllvm . . . 42
phyloplot.gllvm . . . 45
plot.gllvm . . . 48
predict.gllvm . . . 49
predict.glmmVA . . . 52
predictLVs.gllvm . . . 53
predictPairwise.gllvm . . . 54
predictSR.gllvm . . . 55
randomCoefplot.gllvm . . . 57
ranef . . . 59
ranef.glmmVA . . . 59
residuals.gllvm . . . 60
residuals.predictSR.gllvm . . . 61
se.gllvm . . . 62
simulate.gllvm . . . 63
Skabbholmen . . . 64
summary.gllvm . . . 65
update.gllvm . . . 67
vcov.gllvm . . . 68
VP.gllvm . . . 68

Index **71**

AICc *Corrected Akaike information criterion and number of observations*

Description

Calculates corrected Akaike information criterion for small sample sizes, and extracts number of observations.

Usage

```
## S3 method for class 'gllvm'
AICc(object, ...)

## S3 method for class 'gllvm'
nobs(object, ...)
```

Arguments

object	an object of class 'gllvm'.
...	Not used.

Author(s)

Jenni Niku, Bert van der Veen

anova.gllvm *Analysis Of Deviance for gllvm*

Description

Computes an analysis of deviance table for two generalized linear latent variable model fits.

Usage

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

Arguments

object	an object of class 'gllvm'.
...	one or more objects of class 'gllvm'

Details

Computes likelihood-ratio test for two or more gllvm models. Test results makes sense only for nested models. Notice also that this test is not designed for testing models which have degrees of freedom difference larger than 20. For such models the P-value should be treated as very approximate.

Author(s)

Jenni Niku

Examples

```
## Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- antTraits$abund
X <- antTraits$env
TR <- antTraits$traits
# Fit gllvm model
fit1 <- gllvm(y, X, TR, formula = ~ Bare.ground + Shrub.cover, family = poisson())
fit2 <- gllvm(y, X, TR, formula = ~ Bare.ground + Shrub.cover +
             (Bare.ground + Shrub.cover) : Webers.length, family = poisson())
# Test if the model with fourth corner interaction terms is significantly
# better using likelihood-ratio test:
anova(fit1, fit2)
```

---

beetle *ground beetle assemblages*

Description

These data describe the abundance of ground beetle assemblages from the Scottish landscape alongside with the environmental data and species traits. The data includes abundances of 68 species of ground beetle species.

Usage

```
data(beetle)
```

Format

Y A data frame of species composition of the ground-beetle assemblages.

X A data frame of study design variables and environmental data.

SiteCode Unique id for sample.

Landuse Land use type.

Grid Grid where sample were collected.

Area sampling area in Scotland.

Samplingyear Sampling year

Texture 1, peat; 2, peaty loam; 3, loamy sand; 4, sandy loam; 5, sandy clay loam; 6, sandy silt loam; 7, silty clay

Org organic content (% loss of organic content on ignition), log10 transformed

pH soil pH

AvailP available P (mg/L), log10 transformed

AvailK available K (mg/L)

Moist percentage moisture content

Bare percentage cover estimate of bare ground in 11 1-m2 quadrats, arcsine transformed

Litter percentage cover estimate of litter cover in 11 1-m2 quadrats, log10 transformed

Bryophyte percentage cover estimate of bryophytes in 11 1-m2 quadrats, arcsine transformed

Plants.m2 number of reproductive stems (flowering or fruiting) in 11 1-m2 quadrats

Canopy height canopy height (cm) in 11 1-m2 quadrats

Stemdensity number of stems (ramets) in 100 cm2

Biom_l5 dry mass (g) of biomass 0–5 cm from soil surface in 400 cm2

Biom_m5 dry mass (g) of biomass .5 cm from soil surface in 400 cm2, log10 transformed

Reprobiom biomass of reproductive parts (flowers and fruits) in 100 cm2, log10 transformed

Elevation elevation in m a.s.l.

Management management intensity index (see Materials and Methods: Environmental data)

TR A data frame of the species names and species morphological and life trait characteristics.

SPECIES Species names.

CODE Species codes corresponding species names in abundance matrix

LYW diameter of the eye, measured from above

LAL length of the antenna

LPW maximum width of the pronotum

LPH maximum depth ("vaulting") of the pronotum

LEW maximum width of the elytra

LFL length of the metafemur (with the articulation segments), from the coxa to the apex

LTR length of the metatrochanter

LRL length of the metatarsi

LFW maximum width of the metafemur

LTL total length (length of the pronotum in the medial line plus length of the elytra, from the medial ridge of the scutellum to the apex)

CLG color of the legs (1, pale; 2, black; 3, metallic)

CLB color of the body (1, pale; 2, black; 3, metallic)

WIN wing development (1, apterous or brachypterous; 2, dimorphic; 3, macropterous)

PRS shape of the pronotum (1, oval; 2, cordiform; 3, trapezoidal)

OVE overwintering (1, only adults; 2, adults and larvae or only larvae)

FOA food of the adult (1, mostly Collembola; 2, generalist predator; 3, mostly plant material)

DAY daily activity (1, only diurnal; 2, nocturnal)

BRE breeding season (1, spring; 2, summer; 3, autumn or winter)

EME main period of emergence of the adults (1, spring; 2, summer; 3, autumn)

ACT main period of adult activity (1, autumn; 2, summer only)

Details

Beetles were sampled with two parallel rows of nine pitfall traps (diameter 7.5 cm, 2 m apart) at each site, starting in early May.

Detailed description of the data available in the reference and in the Ecological Archives E082-012.

References

Ignacio Ribera, Sylvain Dolédec, Iain S. Downie, and Garth N. Foster. 2001. Effect of land disturbance and stress on species traits of ground beetle assemblages. Ecology 82:1112-1129.

Examples

```
## Not run:
data(beetle)
# Abundance matrix
Y <- beetle$Y
# Environmental data
X <- beetle$X
# Species traits
TR <- beetle$TR

## End(Not run)
```

`coefplot.gllvm` *Plot covariate coefficients and confidence intervals*

Description

Plots covariate coefficients and their confidence intervals.

Usage

```
## S3 method for class 'gllvm'
coefplot(
  object,
  y.label = TRUE,
  which.Xcoef = NULL,
  order = TRUE,
  cex.ylab = 0.5,
  cex.xlab = 1.3,
  mfrow = NULL,
  mar = c(4, 6, 2, 1),
  xlim.list = NULL,
  ind.spp = NULL,
  ...
)
```

Arguments

object	an object of class 'gllvm'.
y.label	logical, if TRUE (default) colnames of y with respect to coefficients are added to plot.
which.Xcoef	vector indicating which covariate coefficients will be plotted. Can be vector of covariate names or numbers. Default is NULL when all covariate coefficients are plotted.
order	logical, whether or not coefficients are ordered, defaults to TRUE.
cex.ylab	the magnification to be used for axis annotation relative to the current setting of cex.
cex.xlab	the magnification to be used for axis annotation.
mfrow	same as mfrow in par. If NULL (default) it is determined automatically.
mar	vector of length 4, which defines the margin sizes: c(bottom, left, top, right). Defaults to c(4,5,2,1).
xlim.list	list of vectors with length of two to define the intervals for an x axis in each covariate plot. Defaults to NULL when the interval is defined by the range of point estimates and confidence intervals
ind.spp	vector of species indices to construct the caterpillar plot for
...	additional graphical arguments.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>, Francis K.C. Hui, Sara Taskinen, Bert van der Veen

Examples

```
# Extract subset of the microbial data to be used as an example
data(microbialdata)
X <- microbialdata$Xenv
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                    decreasing = TRUE)[21:40]]
fit <- gllvm(y, X, formula = ~ pH + Phosp, family = poisson())
coefplot(fit)
## Not run:
# Fit  gllvm model with environmental covariances and reduced rank
fitRR <- gllvm(y = y, X = X, num.RR = 2, family = "negative.binomial")
coefplot(fitRR, mfrow=c(2,3))

## End(Not run)
```

confint.gllvm *Confidence intervals for model parameters*

Description

Computes confidence intervals for parameters in a fitted gllvm model.

Usage

```
## S3 method for class 'gllvm'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

object	an object of class 'gllvm'.
parm	a specification of which parameters are to be given confidence intervals, a vector of names. Examples of options are "beta0", "Xcoef",theta", "phi". If missing, all parameters are considered.
level	the confidence level. Scalar between 0 and 1.
...	not used.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>

Examples

```
## Not run:
## Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
X <- as.matrix(antTraits$env[,1:2])
# Fit gllvm model
fit <- gllvm(y = y, X = X, family = poisson())
# 95 % confidence intervals for coefficients of X variables
confint(fit, level = 0.95, parm = "Xcoef")

## End(Not run)
```

ecoCoefs *Functions to extract ecological quantities of the latent variables from a GLLVM, if species are a quadratic function of the latent variables.*

Description

Extracts species optima and tolerances, potentially with standard errors (derived with the Delta method).

Usage

```
## S3 method for class 'gllvm'
optima(object, sd.errors = TRUE, ...)

## S3 method for class 'gllvm'
tolerances(object, sd.errors = TRUE, ...)
```

Arguments

object	an object of class 'gllvm'.
sd.errors	logical. If TRUE, also returns standard errors.
...	Not used.

Details

Currently no separate method for calculating species maxima or gradient length are implemented. Gradient length can be inferred from the standard deviation of the latent variables, which is reported by summary.gllvm.

Author(s)

Bert van der Veen

eSpider *Hunting spider data*

Description

Extended dataset of counts of hunting spiders in a dune area in the Netherlands, measured at 100 pitfall traps.

Usage

```
data(eSpider)
```

Format

abund A data frame with abundances of 12 hunting spider species measured at 100 sites.

X A matrix of 26 predictor variables at 28 of the 100 sites.

nonNA An vector of indices indicating at which sites the predictor variables were measured.

Details

Counts of hunting spiders in a dune area in the Netherlands, measured with 100 different pitfall traps. This dataset was published with permission from the CANOCO FORTRAN package (version 4 or higher) example datasets.

Species names have been abbreviated, corresponding to: Alopacce = *Alopecosa accentuata*, Alopcune = *Alopecosa cuneata*, Alopfabr = *Alopecosa fabrilis*, Arctlute = *Arctosa lutetiana*, Arctperi = *Arctosa perita*, Auloalbi = *Aulonia albimana*, Pardlugu = *Pardosa lugubris*, Pardmont = *Pardosa monticola*, Pardnigr = *Pardosa nigriceps*, Pardpull = *Pardosa pullata*, Trocterr = *Trochosa terricola*, Zoraspin = *Zora spinimana*.

Environmental measurements were taken at 28 of the 100 pitfall traps measuring soil properties(Water content: "conWate", Humus content: "conHumu", Acidity (pH-KCl)), vegetation ("BareSand": percentage bare sand, "FallTwig": cover on the ground by leaves and twigs, "CovMoss": cover by mosses and lichens, "CovHerb": cover by the herb and grass layer (including maximum height, minimum height, "CovCala": cover by *Calamagrostis epigejos*, cover by *Carex arenaria*, "CovFest": cover by *Festuca ovina*, "CovCory": cover by *Corynephorus canescens*, "CovUrti": cover by *Urtica dioica*, "CovMoeh": cover by *Moehringia trinervia*), "CovShru": cover by the shrub layer (minimum and maximum height, and "CovLigu": cover by *Ligustrum vulgare*), "CovTree": cover by the tree layer (including maximum height, cover by *Populus tremula* or *Crataegus monogyna*)), and light properties ("LuxGrey": lux at equal grey sky, "LuxSun": lux at cloudless sky, "LuxRef": lux by reflection of the soil surface)

The original publication of Canonical Correspondence Analysis used standardized versions of the `log` and `log1p` transformed predictors "ConWate", "BareSand", "FallTwig", "CovMoss", "CovHerb", "LuxRef".

References

ter Braak, C.J.F. and Smilauer, P. (1998). CANOCO reference manual and user's guide to CANOCO for Windows: software for canonical community ordination (version 4). Microcomputer Power, New York, New York, USA. ter Braak, C.J.F. (1986). Canonical correspondence analysis: a new eigenvector technique for multivariate direct gradient analysis. Ecology, 67(5), 1167-1179. Van der Aart, P. J. M. and Smeenk-Enserink, N. (1975). Correlations between distributions of hunting spiders (*Lycosidae*, *Ctenidae*) and environmental characteristics in a dune area. Netherlands Journal of Zoology, 25(1), 1-45.

Examples

```
data(eSpider)
Y <- eSpider$abund[eSpider$nonNA, ]
X <- eSpider$X[eSpider$nonNA, ]
model <- gllvm(y = Y, X = X,
    lv.formula = ~log(ConWate) + log1p(BareSand) + log1p(FallTwig) +
```

```
        log1p(CovMoss) + log1p(CovHerb) + log(LuxRef),
        num.RR = 2,
        family = "poisson")
```

`fungi` *Wood-decaying fungi data*

Description

Dataset of 1666 binary observations for 215 fungal species, in different 53 European Beech forests spread across 8 regions.

Usage

```
data(fungi)
```

Format

Y A data frame with the presence-absences of 215 fungal species measured at 1666 logs.

X A data frame of 8 predictor variables.

- **DBH.CM** Diameter at breast height (cm)
- **AVERDP** Decay stage of logs on a 1-5 scale
- **CONNECT10** Connectivity of the surrounding forest at 10km scale
- **TEMPR** Annual temperature range in degrees Celsius
- **PRECIP** Annual precipitation in milimeters
- **log.Area** ln(area in hectares) of reserves
- **REGION** Site groups identified based on spatial clusters
- **RESERVE** Site name

TR A data frame of the traits used in Abrego et al. (2022).

tree The phylogenetic tree.

C The phylogenetic covariance matrix.

dist The phylogenetic distance matrix.

Details

Observations of fungi species inhabiting European beech logs, in different European countries. The countries have been grouped in eight different regions. Logs were surveyed in 53 different reserves (or sites). Included environment and trait covariates are limited to those analyzed in the original article, though more are available in the published dataset on datadryad.org.

References

Abrego, N., Bässler, C., Christensen, M., and Heilmann-Clausen, J. (2022). Traits and phylogenies modulate the environmental responses of wood-inhabiting fungal communities across spatial scales. Journal of Ecology, 110(4), 784-798.

Abrego, N., Bässler, C., Christensen, M., and Heilmann-Clausen, J. (2022). Data and code from: Traits and phylogenies modulate the environmental responses of wood-inhabiting fungal communities across spatial scales [Dataset]. Dryad. https://doi.org/10.5061/dryad.t76hdr82r

Examples

```
## Not run:
data(fungi)
Y <- fungi$Y
X <- fungi$X
TR <- fungi$TR
C <- fungi$C
dist <- fungi$dist

# cbind is added to treat species-specific random intercept terms
# for when species-specific fixed intercepts are collected to a global intercept
# with beta0com = TRUE as is the case
#model <- gllvm(y = Y, X = cbind(int = 1, X), TR = TR,
#        formula = ~DBH.CM + AVERDP + I(AVERDP^2) + CONNECT10 + TEMPR + PRECIP +
#        log.AREA + (DBH.CM + AVERDP + I(AVERDP^2) + CONNECT10 + TEMPR + PRECIP +
#        log.AREA):(FB.type + Sp.log.vol.µ3 + Lifestyle),
#        family = "binomial", num.lv = 0, studyDesign = X[,c("REGION", "RESERVE")],
#                     colMat = list(C, dist = dist), colMat.rho.struct = "term",
#                     row.eff = ~(1 | REGION/RESERVE), sd.errors = FALSE,
#                     randomX = ~int + DBH.CM + AVERDP + I(AVERDP^2) +
#                                 CONNECT10 + TEMPR + PRECIP + log.AREA,
#                     beta0com = TRUE, nn.colMat = 10, maxit = 20000)

## End(Not run)
```

`getEnvironCov.gllvm` *Extract species covariances due to environmental random effects from gllvm object*

Description

Calculates the species environment covariance matrix for a gllvm object.

Usage

```
## S3 method for class 'gllvm'
getEnvironCov(object, x = NULL, relative = TRUE, ...)
```

Arguments

`object`	an object of class 'gllvm'.
`x`	(optional) vector of covariate values to calculate the covariance for. Defaults to a vector of 1s. If both 'randomX' and random species effects are present in the model this should be a list of length two.
`relative`	defaults to TRUE. For `getEnvironCor`, if there are residual latent effects in the model (such as due to num.lv or num.lv.c), this scales the covariance matrix by the sum of the environmental and residual variances instead, so that the diagonal of the correlation matrix is smaller than one. This matrix then represents the contribution of the environment to the overall species associations in the model.
`...`	arguments passed on to `getResidualCov.gllvm`

Details

Species covariance matrix due to the environment is calculated.

Covariances due to the covariates can only be calculated when random effects are included in the model, and are thus limited to reduced rank models (those including constrained and concurrent ordinations) fitted with random slopes, models fitted with random effects via the formula interface, or the fourth corner model fitted with random slopes. For full rank models with random slopes, i.e., with the formula interface or the fourth corner model, the covariances of species are formulated as:

$$\Sigma_e = kronecker(C\rho + (1-\rho)I_p, R),$$

where C is a correlation matrix for the columns in the response (e.g., a Phylogenetic matrix), ρ the signal parameter, and R the covariance matrix for the random effects. Here,

$$I = kronecker(I_p, x)$$

, with x a vector of covariate values for each of the random effects, which defaults to a vector of 1s. when there are covariate-specific phylogenetic signal parameters in the model, this is instead:

$$\Sigma_e = kronecker(x_i', I_m) * bdiag(L_k) * kronecker(\Sigma_r, I_m) * bdiag(L_k') * kronecker(x_i, I_m),$$

where $bdiag(L_k)$ is a block-diagonal lower triangular matrix, and each L_k the lower triangular matrix of the covariance matrix for each covariate.

For reduced rank models, the covariance is separately defined for the different variance structures of the canonical coefficients in the package. With LV-specific variances and without correlations, we have:

$$\Sigma_e = \Theta * S * \Theta',$$

where Θ is the matrix of loadings, and S the (diagonal) covariance matrix for the canonical coefficients. With predictor-specific variances, we instead have:

$$\Sigma_e = \sum_{k=1}^{K} \Theta(I_d * \sigma_k^2)\Theta',$$

with I_d an identity matrix for the number of constrained and informed latent variables, and σ_k^2 the variance per predictor for the canonical coefficients. When correlations are included, we have:

$$\Sigma_e = kronecker(x_i', I_m) kronecker(\Sigma, \Theta\Theta') kronecker(x_i, I_m),$$

and similarly for LV-specific variances with correlations. Expressions for the quadratic models in the package are determined similarly but not documented here for brevity.

Value

Function returns the following components:

`cov`	species covariances due to covariates
`trace.randomB`	trace of the covariance matrix due to random canonical coefficients
`trace.randomB.quad`	trace of the covariance matrix components due to squared model terms
`trace.col.eff`	trace of the covariance matrix due to random column (species) effects

Author(s)

Bert van der Veen

See Also

`getEnvironCor` ,`getResidualCov.gllvm`, `getResidualCor.gllvm`,.

Examples

```
## Not run:
# Example with the spider dataset
data(eSpider)
y = eSpider$abund[eSpider$nonNA,]
X = eSpider$X[eSpider$nonNA,]
fit <- gllvm(y, X = scale(X), num.RR = 2,
            randomB = "P", family = "negative.binomial")
envcov <- getEnvironCov(fit)
envcov$trace.randomB
# As proportion of variance in the model
envcov$trace.randomB/sum(envcov$trace.randomB)

## End(Not run)
```

getLoadings.gllvm *Extract loadings*

Description

Extract loadings (species scores) from a gllvm object.

Usage

```
## S3 method for class 'gllvm'
getLoadings(object, ...)
```

Arguments

object	an object of class 'gllvm'.
...	not used

Details

Function retrieves the loadings a.k.a. species scores for a GLLVM. For the optima of a quadratic response model, see optima.gllvm

getLV.gllvm *Extract latent variables*

Description

Extract latent variables from gllvm object.

Usage

```
## S3 method for class 'gllvm'
getLV(object, type = NULL, ...)
```

Arguments

object	an object of class 'gllvm'.
type	type of latent variable scores to retrieve from a gllvm object. For models with unconstrained latent variables, defaults to "residual". For models with constrained latent variables, defaults to conditional. Alternatively, "marginal" returns linear combination scores without residual error.
...	not used

Details

Function retrieves the site scores for a GLLVM. Each type corresponds to a separate term of the model. For a GLLVM with unconstrained latent variables the default is "residual". "Residual" scores represent the error term in concurrent ordination, and are not available for constrained ordination.

For GLLVMs with informed latent variables, "conditional" returns the complete site scores, due to both fixed- and latent effects, where the latent effect is always scaled by the diagonal of the species loadings so that it can be small relative to the fixed-effects. "Conditional" here means conditional on the random-effect i.e. the residual.

Type "marginal" returns linear combination scores, i.e. the site scores only due to fixed-effects. These are available for constrained and concurrent ordination.

If both unconstrained and constrained latent variables are included in the model, type "marginal" returns linear combination scores for constrained latent variables but "residual" scores for unconstrained latent variables.

---

getPredictErr.gllvm *Extract prediction errors for latent variables from gllvm object*

Description

Calculates the prediction errors for latent variables and random effects for gllvm model.

Usage

```
## S3 method for class 'gllvm'
getPredictErr(object, CMSEP = TRUE, cov = FALSE, ...)
```

Arguments

object	an object of class 'gllvm'.
CMSEP	logical, if TRUE conditional mean squared errors for predictions are calculated. If FALSE, prediction errors are based on covariances of the variational distributions for method ="VA" and method ="EVA".
cov	if TRUE, return as covariances/variances of predictions. Otherwise FALSE (default) return as standard errors of predictions.
...	not used

Details

Calculates conditional mean squared errors for predictions. If variational approximation is used, prediction errors can be based on covariances of the variational distributions, and therefore they do not take into account the uncertainty in the estimation of (fixed) parameters.

Value

Function returns following components:

lvs	prediction errors for latent variables
row.effects	prediction errors for random row effects if included

Author(s)

Francis K.C. Hui, Jenni Niku, David I. Warton

Examples

```
## Not run:
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
# Fit gllvm model
fit <- gllvm(y = y, family = poisson())
# prediction errors for latent variables:
getPredictErr(fit)

## End(Not run)
```

---

getResidualCor.gllvm *Extract residual correlations from gllvm object*

Description

Calculates the residual correlation matrix for gllvm model.

Usage

```
## S3 method for class 'gllvm'
getResidualCor(object, adjust = 1, x = NULL, ...)
```

Arguments

object	an object of class 'gllvm'.
adjust	The type of adjustment used for negative binomial and binomial distribution when computing residual correlation matrix. Options are 0 (no adjustment), 1 (the default adjustment) and 2 (alternative adjustment for NB distribution). See details.
x	(optional) vector of covariate values to calculate the covariance for, when applicable.
...	not used

Details

Residual correlation matrix is calculated based on the residual covariance matrix, see details from getResidualCov.gllvm.

Author(s)

Francis K.C. Hui, Jenni Niku, David I. Warton

Examples

```
#'# Extract subset of the microbial data to be used as an example
data(microbialdata)
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                     decreasing = TRUE)[21:40]]
fit <- gllvm(y, family = poisson())
fit$logL
cr <- getResidualCor(fit)
cr[1:5,1:5]
## Not run:
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
# Fit gllvm model
fit <- gllvm(y = y, family = poisson())
# residual correlations:
cr <- getResidualCor(fit)
# Plot residual correlations:
install.packages("corrplot", "gclus")
library(corrplot)
library(gclus)
corrplot(cr[order.single(cr), order.single(cr)], diag = F,
  type = "lower", method = "square", tl.cex = 0.8, tl.srt = 45, tl.col = "red")

## End(Not run)
```

---

getResidualCov.gllvm *Extract residual covariance matrix from gllvm object*

Description

Calculates the residual covariance matrix for gllvm model.

Usage

```
## S3 method for class 'gllvm'
getResidualCov(object, adjust = 1, x = NULL, ...)
```

Arguments

object	an object of class 'gllvm'.
adjust	The type of adjustment used for negative binomial, binomial and normal distribution when computing residual correlation matrix. Options are 0 (no adjustment), 1 (the default adjustment) and 2 (alternative adjustment for NB distribution), see details.
x	(optional) vector of covariate values to calculate the covariance for, when applicable.
...	not used.

Details

Residual covariance matrix, storing information on species co-occurrence that is not explained by the environmental variables (if included), is calculated using the matrix of latent variables loadings, that is, $\Theta\Theta'$, and the dispersion parameter related to the distribution of choice, is applicable (e.g. in the case of negative-binomial distributed responses).

When the responses are modelled using the negative binomial distribution, the residual variances for each species must be adjusted for overdispersion. The two possible adjustment terms are $log(\phi_j+1)$ (`adjust = 1`) and $\psi^{(1)}(1/\phi_j)$ (`adjust = 2`), where $\psi^{(1)}$ is the trigamma function.

The negative binomial model can be written using different parameterizations. The residual covariance with `adjust = 1` can be obtained using the lognormal-Poisson parametrization, that is,

$$Y_{ij} \sim Poisson(\mu_{ij}\lambda_j),$$

where $\lambda_j \sim lognormal(-\sigma^2/2, \sigma^2)$ and $\sigma^2 = log(\phi_j + 1)$ and $log(\mu_{ij}) = \eta_{ij}$. Now $E[Y_{ij}] = \mu_{ij}$ and variance $V(\mu_{ij}) = \mu_{ij} + \mu_{ij}^2(exp(\sigma^2) - 1) = \mu_{ij} + \mu_{ij}^2\phi_j$, which are the same as for the NB distribution. Therefore, on linear predictor scale, we have the variance

$$V(log(\mu_{ij}\lambda_j)) = V(log\mu_{ij}) + V(log\lambda_j) = V(u_i'\theta_j) + \sigma^2 = \theta_j'\theta_j + log(\phi_j + 1).$$

which leads to the residual covariance matrix $\Theta\Theta'+\Psi$, where Ψ is the diagonal matrix with $log(\phi_j+1)$ as diagonal elements (`adjust = 1`).

Or, for a GLLVM where species are a quadratic function of the latent variables, we instead have

$$V(log(\mu_{ij}\lambda_j)) = V(log\mu_{ij}) + V(log\lambda_j) = V(u_i'\theta_j - u_i'D_ju_i) + \sigma^2$$
$$= \theta_j'\theta_j + 2diag(D_j)'diag(D_j)log(\phi_j + 1).$$

which leads to the residual covariance matrix $\Theta\Theta'+2\Gamma_j\Gamma_j'+diag(\Phi)$, where Γ_j holds the quadratic coefficients. Since the quadratic coefficients are constrained to be positive, the residual covariance in the latter case is, given the same coefficients on the linear term, equal or more positive than in the linear case.

The residual covariance matrix with `adjust = 2` can be obtained by using Poisson-Gamma parametrization

$$Y_{ij} \sim Poisson(\mu_{ij}\lambda_j),$$

where $\lambda_j \sim Gamma(1/\phi_j, 1/\phi_j)$ and μ_{ij} is as above. The mean and the variance are of similar form as above and we have that

$$V(log(\mu_{ij}\lambda_j)) = V(log\mu_{ij}) + V(log\lambda_j) = \theta_j'\theta_j + \psi^{(1)}(1/\phi_j),$$

where $\psi^{(1)}$ is the trigamma function.

For the negative binomial (1) parameterization, we instead have $\lambda_{ij} \sim Gamma(\mu_{ij}\phi_j, \phi_j)$, so that $V(log\lambda_{ij} = \psi^{(1)}(\mu_{ij}\phi_j)$. As this makes the variance of the link-scale model-dependent, no correction is currently applied.

In the case of binomial distribution, the adjustment terms (`adjust = 1`) are 1 for probit link and $\pi^2/3$ for logit link. These are obtained by treating binomial model as latent variable model. Assume

$$Y_{ij}^* = \eta_{ij} + e_{ij},$$

where $e_{ij} \sim N(0,1)$ for probit model, and $e_{ij} \sim logistic(0,1)$ for logit model. Then binary response is defined as $Y_{ij} = 1$, if $Y_{ij}^* > 0$ and 0 otherwise. Now we have that $\mu_{ij} = P(Y_{ij} = 1) = P(Y_{ij}^* > 0) = P(\eta_{ij} > -e_{ij}) = P(e_{ij} <= \eta_{ij})$ which leads to probit and logit models. On linear predictor scale we then have that

$$V(\eta_{ij} + e_{ij}) = V(\eta_{ij}) + V(e_{ij}).$$

For the probit model, the residual covariance matrix is then $\Theta\Theta' + I_m$, and for the logit model $\Theta\Theta' + \pi^2/3I_m$. Similarly as above, for a GLLVM where species are a quadratic function of the latent variables, the term $2\Gamma_j\Gamma_j'$ is added to the residual covariance matrix.

For normal distribution, we can write

$$Y_{ij} = \eta_{ij} + e_{ij},$$

where $e_{ij} \sim N(0, \phi_j^2)$ and thus we have that

$$V(\eta_{ij} + e_{ij}) = V(\eta_{ij}) + V(e_{ij}).$$

For the gaussian model, the residual covariance matrix is then $\Theta\Theta' + diag(\Phi^2)$.

Value

Function returns following components:

`cov`	residual covariance matrix
`trace`	trace of the residual covariance matrix, the total variance explained
`var.q`	trace of the residual covariance matrix per latent variable, variance explained per latent variable
`var.q2`	trace of the squared term of the residual covariance matrix per latent variable, for quadratic responses. Variance explained per latent variable by the quadratic term

Author(s)

Francis K.C. Hui, Jenni Niku, David I. Warton, Bert van der Veen

Examples

```
## Not run:
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
# Fit gllvm model
fit <- gllvm(y = y, family = poisson())
# residual covariance:
rescov <- getResidualCov(fit)
rescov$cov
# Trace of the covariance matrix
rescov$trace
# Variance explained per latent variable
rescov$var.q

## End(Not run)
```

---

gllvm *Generalized Linear Latent Variable Models*

Description

Fits generalized linear latent variable model for multivariate data. The model can be fitted using Laplace approximation method or variational approximation method.

Usage

```
gllvm(
  y = NULL,
  X = NULL,
  TR = NULL,
  data = NULL,
  formula = NULL,
  family,
  num.lv = NULL,
  num.lv.c = 0,
  num.RR = 0,
  lv.formula = NULL,
  lvCor = NULL,
  studyDesign = NULL,
  dist = list(matrix(0)),
  distLV = matrix(0),
  colMat = NULL,
  colMat.rho.struct = "single",
  corWithin = FALSE,
  corWithinLV = FALSE,
  quadratic = FALSE,
  row.eff = FALSE,
```

```
  sd.errors = TRUE,
  offset = NULL,
  method = "VA",
  randomB = FALSE,
  randomX = NULL,
  beta0com = FALSE,
  zeta.struc = "species",
  plot = FALSE,
  link = "probit",
  Ntrials = matrix(1),
  Power = 1.1,
  seed = NULL,
  scale.X = TRUE,
  return.terms = TRUE,
  gradient.check = FALSE,
  disp.formula = NULL,
 control = list(reltol = 1e-10, reltol.c = 1e-08, TMB = TRUE, optimizer = ifelse((num.RR
  + num.lv.c) <= 1 | randomB != FALSE, "optim", "alabama"), max.iter = 6000, maxit =
   6000, trace = FALSE, optimizer.trace = 0, optim.method = NULL, nn.colMat = 10,
    colMat.approx = "NNGP"),
 control.va = list(Lambda.struc = "unstructured", Ab.struct = ifelse(is.null(colMat),
  "blockdiagonal", "MNunstructured"), Ab.struct.rank = 1, Ar.struc = "diagonal",
   diag.iter = 1, Ab.diag.iter = 0, Lambda.start = c(0.3, 0.3, 0.3), NN = 10),
 control.start = list(starting.val = "res", n.init = 1, n.init.max = 10, jitter.var = 0,
   jitter.var.br = 0, start.fit = NULL, start.lvs = NULL, randomX.start = "res",
  quad.start = 0.01, start.struc = "LV", scalmax = 10, MaternKappa = 1.5, rangeP =
  NULL, zetacutoff = NULL, start.optimizer = "nlminb", start.optim.method = "BFGS"),
  setMap = NULL,
  ...
)
```

Arguments

y	(n x m) matrix of responses.
X	matrix or data.frame of environmental covariates.
TR	matrix or data.frame of trait covariates.
data	data in long format, that is, matrix of responses, environmental and trait covariates and row index named as "id". When used, model needs to be defined using formula. This is alternative data input for y, X and TR.
formula	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted (for column-specific effects).
family	distribution function for responses, or a vector of families for mixed response type model. Options are `"negative.binomial"` and `"negative.binomial1"` (with log link), `poisson(link = "log")`, `binomial` (with probit, logit, or cloglog link), zero-inflated binomial (`ZIB`), zero-and-N-inflated binomial (`ZNIB`) zero-inflated poisson (`"ZIP"`), zero-inflated negative-binomial (`"ZINB"`), `gaussian(link = "identity")`, Tweedie (`"tweedie"`) (with log link), `"gamma"` (with log link),

"exponential" (with log link), beta ("beta") (with logit and probit link, for "LA" and "EVA"-method), "ordinal" (with "VA" and "EVA"-method, with probit or logit link), beta hurdle "betaH" (for "VA" and "EVA"-method) and "orderedBeta" (for "VA" and "EVA"-method). Note: "betaH" and "orderedBeta" with "VA"-method are actually fitted using a hybrid approach such that EVA is applied to the beta distribution part of the likelihood.

num.lv — number of latent variables, d, in gllvm model. Non-negative integer, less than number of response variables (m). Defaults to 2, if num.lv.c=0 and num.RR=0, otherwise 0.

num.lv.c — number of latent variables, d, in gllvm model to inform, i.e., with residual term. Non-negative integer, less than number of response (m) and equal to, or less than, the number of predictor variables (k). Defaults to 0. Requires specification of "lv.formula" in combination with "X" or "datayx". Can be used in combination with num.lv and fixed-effects, but not with traits.

num.RR — number of latent variables, d, in gllvm model to constrain, without residual term (reduced rank regression). Cannot yet be combined with traits.

lv.formula — an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted (for latent variables).

lvCor — correlation structure for latent variables, defaults to NULL Correlation structure for latent variables can be defined via formula, eg. ~struc(1|groups), where option to 'struc' are corAR1 (AR(1) covariance), corExp (exponentially decaying, see argument 'dist'), corCS (compound symmetry), and propto (proportional covariance, used as propto(a+b|group, matrix)). The grouping variable needs to be included either in studyDesign. Works at the moment only with unconstrained ordination without quadratic term.

studyDesign — variables related to eg. sampling/study design, used for defining correlation structure of the latent variables and row effects.

dist — list of length equal to the number of row effects with correlation structure corExp that holds the matrix of coordinates or time points.

distLV — matrix of coordinates or time points used for LV correlation structure corExp.

colMat — either a list of length 2 with matrix of similarity for the column effects and named matrix "dist" of pairwise distances (of columns, to use in selecting nearest neighbours) for a sparse approximation of the matrix inverse in the likelihood, or only a (p.d.) matrix of similarity for the column effects for a normal inverse calculation.

colMat.rho.struct — either single (default) or term indicating whether the signal parameter should be shared for covariates, or not.

corWithin — logical. Vector of length equal to the number of row effects. For structured row effects with correlation, If TRUE, correlation is set between row effects of the observation units within group. Correlation and groups can be defined using row.eff. Defaults to FALSE, when correlation is set for row parameters between groups.

corWithinLV — logical. For LVs with correlation, If TRUE, correlation is set between rows of the observation units within group. Defaults to FALSE, when correlation is set for rows between groups.

quadratic — either FALSE(default), TRUE, or LV. If FALSE models species responses as a linear function of the latent variables. If TRUE models species responses as a quadratic function of the latent variables. If LV assumes species all have the same quadratic coefficient per latent variable.

row.eff — FALSE, fixed, "random" or formula to define the structure for the community level row effects, indicating whether row effects are included in the model as a fixed or as a random effects. Defaults to FALSE when row effects are not included. Structured random row effects can be defined via formula, eg. ~(1|groups), when unique row effects are set for each group, not for all rows, the grouping variable needs to be included in studyDesign. Correlation structure between random group effects/intercepts can also be set using ~struc(1|groups), where option to 'struc' are corAR1 (AR(1) covariance), corExp (exponentially decaying, see argument 'dist') and corCS (compound symmetry). Correlation structure can be set between or within groups, see argument 'corWithin'.

sd.errors — logical. If TRUE (default) standard errors for parameter estimates are calculated.

offset — vector or matrix of offset terms.

method — model can be fitted using Laplace approximation method (method = "LA") or variational approximation method (method = "VA"), or with extended variational approximation method (method = "EVA") when VA is not applicable. If particular model has not been implemented using the selected method, model is fitted using the alternative method as a default. Defaults to "VA".

randomB — either FALSE(default), "LV", "P", "single", or "iid". Fits concurrent or constrained ordination (i.e. models with num.lv.c or num.RR) with random slopes for the predictors. "LV" assumes LV-specific variance parameters, "P" predictor specific, and "single" the same across LVs and predictors.

randomX — formula for species specific random effects of environmental variables in fourth corner model. Defaults to NULL, so that no random slopes are included by default.

beta0com — logical. If FALSE column-specific intercepts are assumed. If TRUE, column-specific intercepts are collected to a common value.

zeta.struc — structure for cut-offs in the ordinal model. Either "common", for the same cut-offs for all species, or "species" for species-specific cut-offs. For the latter, classes are arbitrary per species, each category per species needs to have at least one observations. Defaults to "species".

plot — logical. If TRUE ordination plots will be printed in each iteration step when TMB = FALSE. Defaults to FALSE.

link — link function for binomial family if method = "LA" and beta family. Options are "logit" and "probit" and "cloglog".

Ntrials — number of trials for binomial, ZIB and ZNIB families.

Power — fixed power parameter in Tweedie model. Scalar from interval (1,2). Defaults to 1.1. If set to NULL it is estimated (note: experimental).

seed — a single seed value if n.init=1, and a seed value vector of length n.init if n.init>1. Defaults to NULL, when new seed is not set for single initial fit and seeds are is randomly generated if multiple initial fits are set.

`scale.X` logical. If `TRUE`, covariates are scaled when fourth corner model is fitted.

`return.terms` logical. If `TRUE` 'terms' object is returned.

`gradient.check` logical. If `TRUE` gradients are checked for large values (>0.01) even if the optimization algorithm did converge.

`disp.formula` a vector of indices, or alternatively formula, for the grouping of dispersion parameters (e.g. in a negative-binomial distribution, ZINB, tweedie), shape parameters (gamma, Beta, ordered Beta, hurdle Beta models) or variance parameters (gaussian distribution). Defaults to NULL so that all species have their own dispersion parameter. Is only allowed to include categorical variables. If a formula, data should be included as named rows in y.

`control` A list with the following arguments controlling the optimization:

***reltol*:** convergence criteria for log-likelihood, defaults to 1e-10.

***reltol.c*:** convergence criteria for equality constraints in ordination with predictors, defaults to 1e-8.

***TMB*:** logical, if `TRUE` model will be fitted using Template Model Builder (TMB). TMB is always used if `method = "LA"`. Defaults to `TRUE`.

***optimizer*:** if `TMB=TRUE`, log-likelihood can be optimized using `"optim"` (default) or `"nlminb"`. For ordination with predictors (num.RR>0 or num.lv.c>0) this can additionally be one of `alabama`(default), `nloptr(agl)` or `nloptr(sqp)`.

***max.iter*:** maximum number of iterations when `TMB = FALSE` or for `optimizer = "nlminb"` when `TMB = TRUE`, defaults to 6000.

***maxit*:** maximum number of iterations for optimizer, defaults to 6000.

***trace*:** logical, if `TRUE` in each iteration step information on current step will be printed. Defaults to `FALSE`.

***optimizer.trace*:** numeric, see the trace argument that is specific to each optimizer, what is printed is specific to the optimizer that is used.

***optim.method*:** optimization method to be used if optimizer is `"optim"`,`"alabama"`, or `"nloptr"`, but the latter two are only available in combination with at least two latent variables (i.e., num.RR+num.lv.c>1). Defaults to `"BFGS"`, but to `"L-BFGS-B"` for Tweedie family due the limited-memory use. For optimizer='alabama' this can be any `"optim"` method, or `"nlminb"`. If optimizer = 'nloptr(agl)' this can be one of: "NLOPT_LD_CCSAQ", "NLOPT_LD_SLSQP", "NLOPT_LD_TNEWTON_PRECOND" (default), "NLOPT_LD_TNEWTON", "NLOPT_LD_MMA".

***nn.colMat*:** number of nearest neighbours for calculating inverse of "colMat" when `colMat.approx = "NNGP"`, defaults to 10. Otherwise, if `colMat.approx = "band"`, nn.colMat is the bandwidth of the approximation. If set to the number of columns in the response data, a standard inverse is used instead.

`control.va` A list with the following arguments controlling the variational approximation method:

***Lambda.struc*:** covariance structure of VA distributions for latent variables when `method = "VA"`, "unstructured" or "diagonal".

***Ab.struct*:** covariance structure of VA distributions for random slopes when `method = "VA"`, ordered in terms of complexity: "diagonal", "MNdiagonal" (only with colMat), "blockdiagonal" (default without colMat), "MNunstructured" (default, only with colMat), "diagonalCL1" ,"CL1" (only with

colMat), "CL2" (only with colMat),"diagonalCL2" (only with colMat), or "unstructured" (only with colMat).

***Ab.struct.rank*:** number of columns for the cholesky of the variational covariance matrix to use, defaults to 1. Only applicable with "MNunstructured", "diagonalCL1", "CL1","diagonalCL2", and "unstructured".

***Ar.struc*:** covariance structure of VA distributions for random row effects when `method = "VA"`, "unstructured" or "diagonal". Defaults to "diagonal". "Unstructured" is block diagonal for ordinary random effects.

***diag.iter*:** non-negative integer which can sometimes be used to speed up the updating of variational (covariance) parameters in VA method. Can sometimes improve the accuracy. If `TMB = TRUE` either 0 or 1. Defaults to 1.

***Ab.diag.iter*:** As above, but for variational covariance of random slopes.

***Lambda.start*:** starting values for variances in VA distributions for latent variables, random row effects and random slopes in variational approximation method. Defaults to 0.3.

***NN*:** Number of nearest neighbors for NN variational covariance. Defaults to 10.

`control.start` A list with the following arguments controlling the starting values:

***starting.val*:** starting values can be generated by fitting model without latent variables, and applying factorial analysis to residuals to get starting values for latent variables and their coefficients (`starting.val = "res"`). Another options are to use zeros as a starting values (`starting.val = "zero"`) or initialize starting values for latent variables with (n x num.lv) matrix. Defaults to `"res"`, which is recommended.

***n.init*:** number of initial runs. Uses multiple runs and picks up the one giving highest log-likelihood value. Defaults to 1.

***n.init.max*:** maximum number of refits try try for n.init without improvement, defaults to 10.

***start.fit*:** object of class 'gllvm' which can be given as starting parameters for count data (poisson, NB, or ZIP).

***start.lvs*:** initialize starting values for latent variables with (n x num.lv) matrix. Defaults to `NULL`.

***jitter.var*:** jitter variance for starting values of latent variables. Defaults to 0, meaning no jittering.

***jitter.var.br*:** jitter variance for starting values of random slopes. Defaults to 0, meaning no jittering.

***randomX.start*:** starting value method for the random slopes. Options are `"zero"` and `"res"`. Defaults to `"res"`.

***start.struc*:** starting value method for the quadratic term. Options are `"LV"` (default) and `"all"`.

***quad.start*:** starting values for quadratic coefficients. Defaults to 0.01.

***MaternKappa*:** Starting value for smoothness parameter of Matern covariance function. Defaults to 3/2.

***scalmax*:** Sets starting value for the scale parameter for the coordinates. Defaults to 10, when the starting value for scale parameter scales the distances of coordinates between 0-10.

***rangeP*:** Sets starting value for the range parameter for the correlation structure.

***zetacutoff*:** Either vector of length 2 or a matrix of dimension (a number of species x 2). Sets starting value for the cutoff parameters of the ordered beta model.

***start.optimizer*:** optimizer for starting value generation, see "optimizer" for more information.

***start.optim.method*:** optimizer method for starting value generation, see "optim.method" for more information.

`setMap` under development, not properly tested, except for ordinal beta cutoffs (zeta) and for rho_lvc. a list of a set of parameters to be fixed. Parameters to be fixed need to be defined with factors. Other arguments may overwrite these definitions.

... Not used.

Details

Fits generalized linear latent variable models as in Hui et al. (2015 and 2017) and Niku et al. (2017). Method can be used with two types of latent variable models depending on covariates. If only site related environmental covariates are used, the expectation of response Y_{ij} is determined by

$$g(\mu_{ij}) = \eta_{ij} = \alpha_i + \beta_{0j} + x_i'\beta_j + u_i'\theta_j,$$

where $g(.)$ is a known link function, u_i are d-variate latent variables ($d«m$), α_i is an optional community level row effect at site i, and it can be fixed or random effect (also other structures are possible, see below), β_{0j} is an intercept term for species j, β_j and θ_j are column specific coefficients related to covariates and the latent variables, respectively.

Quadratic model: Alternatively, a more complex version of the model can be fitted with `quadratic = TRUE`, where species are modeled as a quadratic function of the latent variables:

$$g(\mu_{ij}) = \eta_{ij} = \alpha_i + \beta_{0j} + x_i'\beta_j + u_i'\theta_j - u_i'D_ju_i$$

. Here, D_j is a diagonal matrix of positive only quadratic coefficients, so that the model generates concave shapes only. This implementation follows the ecological theoretical model where species are generally recognized to exhibit non-linear response curves. For a model with quadratic responses, quadratic coefficients are assumed to be the same for all species:

$$D_j = D$$

. This model requires less information per species and can be expected to be more applicable to most datasets. The quadratic coefficients D can be used to calculate the length of ecological gradients. For quadratic responses, it can be useful to provide the latent variables estimated with a GLLVM with linear responses, or estimated with (Detrended) Correspondence Analysis. The latent variables can then be passed to the `start.lvs` argument inside the `control.start` list, which in many cases gives good results.

Ordination with predictors: For GLLVMs with both linear and quadratic response model, a series of predictors x_{lv} can be included to explain the latent variables:

$$g(\mu_{ij}) = \alpha_i + \beta_{0j} + x_i'\beta_j + (B'x_{lv,i} + \epsilon_i)'\gamma_j - (B'x_{lv,i} + \epsilon_i)'D_j(B'x_{lv,i} + \epsilon_i),$$

where $z_i = B'x_{lv,i} + \epsilon_i$ are latent variables informed by the predictors, but not constrained compared to unconstrained ordination as in methods such as CCA or RDA. Omitting the predictors results in an unconstrained ordination, and omitting ϵ_i in the usual constrained ordination, which can also be fitted.

Fourth corner model: An alternative model is the fourth corner model (Brown et al., 2014, Warton et al., 2015) which will be fitted if also trait covariates are included. The expectation of response Y_{ij} is

$$g(\mu_{ij}) = \alpha_i + \beta_{0j} + x_i'(\beta_x + b_j) + TR_j'\beta_t + vec(B) * kronecker(TR_j, X_i) + u_i'\theta_j - u_i'D_ju_i$$

where g(.), u_i, β_{0j} and θ_j are defined as above. Vectors β_x and β_t are the main effects or coefficients related to environmental and trait covariates, respectively, matrix B includes interaction terms. Vectors b_j are optional species-specific random slopes for environmental covariates. The interaction/fourth corner terms are optional as well as are the main effects of trait covariates.

Structured row effects: In addition to the sample-specific community level random effects, α_i, it is also possible to set arbitrary structure/design for the row effects. That is, assume that observations / rows $i = 1, ..., n$ in the data matrix are from groups $t = 1, ..., T$, so that each row i belongs to one of the groups, denote $G(i) \in \{1, ..., T\}$. Each group t has a number of observations n_t, so that $\sum_{t=1}^{T} n_t = n$. Now we can set random intercept for each group t, (see argument '`row.eff`'):

$$g(\mu_{ij}) = \eta_{ij} = \alpha_{G(i)} + \beta_{0j} + x_i'\beta_j + u_i'\theta_j,$$

There is also a possibility to set correlation structure for the random intercepts between groups, so that $(\alpha_1, ..., \alpha_T)^\top \sim N(0, \Sigma_r)$. That might be the case, for example, when the groups are spatially or temporally dependent. Another option is to set row specific random intercepts α_i, but to set the correlation structure for the observations within groups, (see argument '`corWithin`'). That is, we can set $corr(\alpha_i, \alpha_{i'}) = C(i, i') \neq 0$ according to some correlation function C, when $G(i) = G(i')$. This model is restricted to the case, where each group has equal number of observations (rows), that is $n_t = n_{t'}$ for all $t, t' \in \{1, ..., T\}$.

The correlation structures available in the package are

`corAR1` autoregressive process of order 1.

`corExp` exponentially decaying, see argument '`dist`'.

`corCS` compound symmetry.

Starting values: The method is sensitive for the choices of initial values of the latent variables. Therefore it is recommendable to use multiple runs and pick up the one giving the highest log-likelihood value (see argument '`n.init`'). However, sometimes this is computationally too demanding, and default option `starting.val = "res"` is recommended. For more details on different starting value methods, see Niku et al., (2018).

Models are implemented using TMB (Kristensen et al., 2015) applied to variational approximation (Hui et al., 2017), extended variational approximation (Korhonen et al., 2021) and Laplace approximation (Niku et al., 2017).

With ordinal family response classes must start from 0 or 1.

Distributions:

Mean and variance for distributions are defined as follows.

For count data `family = poisson()`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij}$, or

`family = "negative.binomial"`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij} + \mu_{ij}^2\phi_j$, or

`family = "negative.binomial1"`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij} + \mu_{ij}\phi_j$, or

`family = "ZIP"`: Expectation $E[Y_{ij}] = (1-p_j)\mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij}(1-p_j)(1+\mu_{ij}p_j)$.

`family = "ZINB"`: Expectation $E[Y_{ij}] = (1-p_j)\mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij}(1-p_j)(1+\mu_{ij}(\phi_j+p_j))$.

For binary data `family = binomial()`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = N_{trials}\mu_{ij}(1-\mu_{ij})$.

`family = "ZIB"`: Expectation $E[Y_{ij}] = (1-p_j)N_j\mu_{ij}$, variance $V(\mu_{ij}) = N_j\mu_{ij}(1-p_j)(1+N_j\mu_{ij}p_j)$.

`family = "ZNIB"`: Expectation $E[Y_{ij}] = p_j^N N_j + (1-p_j^0-p_j^N)N_j\mu_{ij}$, variance $V(\mu_{ij}) = p_j^N N_j^2 + (1-p_j^0-p_j^N)N_j\mu_{ij}(1-\mu_{ij}+N_j\mu_{ij}) - E[Y_{ij}]^2$.

`family = "beta.binomial"`: Expectation $E[Y_{ij}] = N_j\mu_{ij}$, variance $V(\mu_{ij}) = N_j\mu_{ij}(1-\mu_{ij})(1+(N_j-1)\phi_j/(\phi_j+1))$ where $\phi_j > 0$ is a species-specific overdispersion parameter. Only available for `method = "LA"`.

For percent cover data $0 < Y_{ij} < 1$ `family = "beta"`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij}(1-\mu_{ij})/(1+\phi_j)$.

For positive continuous data `family = "gamma"`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij}^2/\phi_j$, where ϕ_j is species specific shape parameter.

For non-negative continuous data `family = "exponential"`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \mu_{ij}^2$.

For non-negative continuous or biomass data `family = "tweedie"` Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(\mu_{ij}) = \phi_j * \mu_{ij}^{\nu}$, where ν is a power parameter of Tweedie distribution. See details Dunn and Smyth (2005).

For ordinal data `family = "ordinal"`: Cumulative probit model, see Hui et.al. (2016).

For normal distributed data `family = gaussian()`: Expectation $E[Y_{ij}] = \mu_{ij}$, variance $V(y_{ij}) = \phi_j^2$.

Value

An object of class "gllvm" includes the following components:

`call`	function call.
`y`	(n x m) matrix of responses.
`X`	matrix or data.frame of environmental covariates.
`X.design`	design matrix of environmental covariates.
`lv.X`	design matrix or data.frame of environmental covariates for latent variables.
`lv.X.design`	design matrix or data.frame of environmental covariates for latent variables.
`TR`	Trait matrix.
`formula`	Formula for predictors.

`lv.formula` Formula of latent variables in constrained and concurrent ordination.

`randomX` Formula for species specific random effects in fourth corner model.

`Xd` design matrix for species specific random effects in fourth corner model.

`randomB` Boolean flag for random slopes in constrained and concurrent ordination.

`num.lv` Number of unconstrained latent variables.

`num.lv.c` Number of latent variables in concurrent ordination.

`num.RR` Number of latent variables in constrained ordination.

`Ntrials` Number of trials in a binomial model.

`method` Method used for integration.

`family` Response distribution.

`row.eff` Type of row effect used.

`n.init` Number of model runs for best fit.

`disp.group` Groups for dispersion parameters.

`sd` List of standard errors.

`lvs` Latent variables.

`params` List of parameters

theta latent variables' loadings relative to the diagonal entries of loading matrix

sigma.lv diagonal entries of latent variables' loading matrix

LvXcoef Predictor coefficients (or predictions for random slopes) related to latent variables, i.e. canonical coefficients

beta0 column specific intercepts

Xcoef coefficients related to environmental covariates X

B coefficients in fourth corner model, and RE means

Br column random effects

sigmaB scale parameters for column-specific random effects

rho.sp (positive) correlation parameter for influence strength of "colMat"

row.params.random row-specific random effects

row.params.fixed row-specific fixed effects

sigma scale parameters for row-specific random effects

phi dispersion parameters ϕ for negative binomial or Tweedie family, probability of zero inflation for ZIP family, standard deviation for gaussian family or shape parameter for gamma/beta family

inv.phi dispersion parameters $1/\phi$ for negative binomial

`Power` power parameter ν for Tweedie family

`sd` list of standard errors of parameters

`prediction.errors`
list of prediction covariances for latent variables and variances for random row effects when method `"LA"` is used

`A`, `Ar`, `Ab_lv`, `spArs`
: covariance matrices for variational densities of latent variables, random row effects, random slopes, and colum effects respectively

`seed`
: Seed used for calculating starting values

`TMBfn`
: TMB objective and derivative functions

`logL`
: log likelihood

`convergence`
: convergence code of optimizer

`quadratic`
: flag for quadratic model

`Hess`
: List holding matrices of second derivatives

`beta0com`
: Flag for common intercept

`cstruc`
: Correlation structure for row effects

`cstruclv`
: Correlation structure for LVs

`dist`
: Matrix of coordinates or time points used for row effects

`distLV`
: Matrix of coordinates or time points used for LVs

`col.eff`
: list of components for column random effects

Ab.struct variational covariance structure of fitted model

Ab.struct.rank fitted rank of variational covariance matrix

col.eff flag indicating if column random effects are included

spdr design matrix

colMat.rho.struct character vector for signal parameter

`terms`
: Terms object for main predictors

`start`
: starting values for model

`optim.method`
: Optimization method when using 'optim', 'alabama', or 'nloptr'

Note

If function gives warning: 'In f(x, order = 0) : value out of range in 'lgamma", optimizer have visited an area where gradients become too big. It is automatically fixed by trying another step in the optimization process, and can be ignored if errors do not occur.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>, Wesley Brooks, Riki Herliansyah, Francis K.C. Hui, Pekka Korhonen, Sara Taskinen, Bert van der Veen, David I. Warton

References

Brown, A. M., Warton, D. I., Andrew, N. R., Binns, M., Cassis, G., and Gibb, H. (2014). The fourth-corner solution - using predictive models to understand how species traits interact with the environment. Methods in Ecology and Evolution, 5:344-352.

Dunn, P. K. and Smyth, G. K. (2005). Series evaluation of tweedie exponential dispersion model densities. Statistics and Computing, 15:267-280.

Hui, F. K. C., Taskinen, S., Pledger, S., Foster, S. D., and Warton, D. I. (2015). Model-based approaches to unconstrained ordination. Methods in Ecology and Evolution, 6:399-411.

Hui, F. K. C., Warton, D., Ormerod, J., Haapaniemi, V., and Taskinen, S. (2017). Variational approximations for generalized linear latent variable models. Journal of Computational and Graphical Statistics. Journal of Computational and Graphical Statistics, 26:35-43.

Kasper Kristensen, Anders Nielsen, Casper W. Berg, Hans Skaug, Bradley M. Bell (2016). TMB: Automatic Differentiation and Laplace Approximation. Journal of Statistical Software, 70(5), 1-21.

Korhonen, P., Hui, F. K. C., Niku, J., and Taskinen, S. (2021). Fast, universal estimation of latent variable models using extended variational approximations. Stat Comput 33, 26 (2023).

Niku, J., Warton, D. I., Hui, F. K. C., and Taskinen, S. (2017). Generalized linear latent variable models for multivariate count and biomass data in ecology. Journal of Agricultural, Biological, and Environmental Statistics, 22:498-522.

Niku, J., Brooks, W., Herliansyah, R., Hui, F. K. C., Taskinen, S., and Warton, D. I. (2018). Efficient estimation of generalized linear latent variable models. PLoS One, 14(5):1-20.

Sweeney, J., Haslett, J., & Parnell, A. C. (2014). The zero & $ N $-inflated binomial distribution with applications. arXiv preprint arXiv:1407.0064.

Warton, D. I., Guillaume Blanchet, F., O'Hara, R. B., Ovaskainen, O., Taskinen, S., Walker, S. C. and Hui, F. K. C. (2015). So many variables: Joint modeling in community ecology. Trends in Ecology & Evolution, 30:766-779.

See Also

coefplot.gllvm, confint.gllvm, ordiplot.gllvm, plot.gllvm, summary.gllvm.

Examples

```
# Extract subset of the microbial data to be used as an example
data(microbialdata)
X <- microbialdata$Xenv
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                     decreasing = TRUE)[21:40]]
fit <- gllvm(y, X, formula = ~ pH + Phosp, family = poisson())
fit$logL
ordiplot(fit)
coefplot(fit)


# Inclusion of structured random row effect
sDesign<-data.frame(Site = microbialdata$Xenv$Site)
fit <- gllvm(y, X, formula = ~ pH + Phosp, family = poisson(),
            studyDesign=sDesign, row.eff=~(1|Site))

## Load a dataset from the mvabund package
library(mvabund)
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
X <- as.matrix(antTraits$env)
TR <- antTraits$traits
# Fit model with environmental covariates Bare.ground and Shrub.cover
```

```
fit <- gllvm(y, X, formula = ~ Bare.ground + Shrub.cover,
            family = poisson())
ordiplot(fit)
coefplot.gllvm(fit)

## Example 1: Fit model with two unconstrained latent variables
# Using variational approximation:
fitv0 <- gllvm(y, family = "negative.binomial", method = "VA")
ordiplot(fitv0)
plot(fitv0, mfrow = c(2,2))
summary(fitv0)
confint(fitv0)

## Example 1a: Fit concurrent ordination model with two latent variables and with
# quadratic response model
# We scale and centre the  predictors to improve convergence
fity1 <- gllvm(y, X = scale(X), family = "negative.binomial",
              num.lv.c=2, method="VA")
ordiplot(fity1, biplot = TRUE)

#'## Example 1b: Fit constrained ordination model with two latent variables and with
# random canonical coefficients
fity2 <- gllvm(y, X = scale(X), family = "negative.binomial",
              num.RR=2, randomB="LV", method="VA")

# Using Laplace approximation: (this line may take about 30 sec to run)
fitl0 <- gllvm(y, family = "negative.binomial", method = "LA")
ordiplot(fitl0)

# Poisson family:
fit.p <- gllvm(y, family = poisson(), method = "LA")
ordiplot(fit.p)
# Use poisson model as a starting parameters for ZIP-model, this line
# may take few minutes to run
fit.z <- gllvm(y, family = "ZIP", method = "LA",
              control.start = list(start.fit = fit.p))
ordiplot(fit.z)


## Example 2: gllvm with environmental variables
# Fit model with two latent variables and all environmental covariates,
fitvX <- gllvm(formula = y ~ X, family = "negative.binomial")
ordiplot(fitvX, biplot = TRUE)
coefplot.gllvm(fitvX)
# Fit model with environmental covariates Bare.ground and Shrub.cover
fitvX2 <- gllvm(y, X, formula = ~ Bare.ground + Shrub.cover,
 family = "negative.binomial")
ordiplot(fitvX2)
coefplot.gllvm(fitvX2)
# Use 5 initial runs and pick the best one
fitvX_5 <- gllvm(y, X, formula = ~ Bare.ground + Shrub.cover,
 family = "negative.binomial", control.start=list(n.init = 5, jitter.var = 0.1))
ordiplot(fitvX_5)
```

```
coefplot.gllvm(fitvX_5)

## Example 3: Data in long format
# Reshape data to long format:
datalong <- reshape(data.frame(cbind(y,X)), direction = "long",
                    varying = colnames(y), v.names = "y")
head(datalong)
fitvLong <- gllvm(data = datalong, formula = y ~ Bare.ground + Shrub.cover,
               family = "negative.binomial")

## Example 4: Fourth corner model
# Fit fourth corner model with two latent variables
fitF1 <- gllvm(y = y, X = X, TR = TR, family = "negative.binomial")
coefplot.gllvm(fitF1)
# Fourth corner can be plotted also with next lines
#fourth = fitF1$fourth.corner
#library(lattice)
#a = max( abs(fourth) )
#colort = colorRampPalette(c("blue","white","red"))
#plot.4th = levelplot(t(as.matrix(fourth)), xlab = "Environmental Variables",
#              ylab = "Species traits", col.regions = colort(100),
#              at = seq( -a, a, length = 100), scales = list( x = list(rot = 45)))
#print(plot.4th)

# Specify model using formula
fitF2 <- gllvm(y = y, X = X, TR = TR,
 formula = ~ Bare.ground + Canopy.cover * (Pilosity + Webers.length),
 family = "negative.binomial")
ordiplot(fitF2)
coefplot.gllvm(fitF2)

## Include species specific random slopes to the fourth corner model
fitF3 <- gllvm(y = y, X = X, TR = TR,
 formula = ~ Bare.ground + Canopy.cover * (Pilosity + Webers.length),
 family = "negative.binomial", randomX = ~ Bare.ground + Canopy.cover,
 control.start = list(n.init = 3))
ordiplot(fitF3)
coefplot.gllvm(fitF3)


## Example 5: Fit Tweedie model
# Load coral data
data(tikus)
ycoral <- tikus$abund
# Let's consider only years 1981 and 1983
ycoral <- ycoral[((tikus$x$time == 81) + (tikus$x$time == 83)) > 0, ]
# Exclude species which have observed at less than 4 sites
ycoral <- ycoral[-17, (colSums(ycoral > 0) > 4)]
# Fit Tweedie model for coral data (this line may take few minutes to run)
fit.twe <- gllvm(y = ycoral, family = "tweedie", method = "EVA", seed=111)
fit.twe

## Example 6: Random row effects
```

```
fitRand <- gllvm(y, family = "negative.binomial", row.eff = "random")
ordiplot(fitRand, biplot = TRUE)
```

---

glmmVA *Fitting Generalized Linear Mixed-Effects Models with Variational Approximation*

Description

Wraps the "gllvm" function to fit a (univariate) generalized linear mixed-effects model (GLMM) in a more familiar syntax. Both fixed and random effects are specified in lme4-style via the formula argument.

Usage

```
glmmVA(
  formula,
  data,
  family,
  response.group = NULL,
 control = list(reltol = 1e-10, optimizer = "optim", max.iter = 6000, maxit = 6000,
    optim.method = "BFGS"),
 control.va = list(Ar.struc = "unstructured", diag.iter = 0, Lambda.start = 0.3),
 control.start = list(starting.val = "zero", n.init = 1, n.init.max = 10, start.fit =
    NULL, scalmax = 10, MaternKappa = 1.5, rangeP = NULL, zetacutoff = NULL,
    start.optimizer = "nlminb", start.optim.method = "BFGS"),
  ...
)
```

Arguments

formula	a formula object describing both the fixed- and random-effects part of the model. A response should be present on the left-hand side of the operator, and otherwise passed as a named 'y' argument to the function. Random effects are written in lme4-style. Some structured random effects are supported, see "gllvm" for more information.
data	an optional data frame containing the variables named in formula.
family	a family as supported by the "gllvm" function. For mixed response type models, a character vector of length equal to the number of observations.
response.group	optional factor of length equal to the number of observations. Observations sharing a level are mapped to the same distribution.
control	A list with the following arguments controlling the optimization:
	***reltol*:** convergence criteria for log-likelihood, defaults to 1e-10.
	***optimizer*:** the log-likelihood can be optimized using "optim" (default) or "nlminb".

***max.iter*:** maximum number of iterations for `optimizer = "nlminb"`, defaults to 6000.

***maxit*:** maximum number of iterations for optimizer, defaults to 6000.

***optim.method*:** optimization method to be used if optimizer is `"optim"`. Defaults to `"BFGS"`, but to `"L-BFGS-B"` for Tweedie family due the limited-memory use.

`control.va` A list with the following arguments controlling the variational approximation method:

***Ar.struc*:** covariance structure of VA distributions, "unstructured" or "diagonal". Defaults to "unstructured". "Unstructured" meaning block diagonal for ordinary random effects, a kronecker product for propto structures with correlaton parameters, and fully unstructured for structured random effects (such as "corExp").

***diag.iter*:** non-negative integer which can sometimes be used to speed up the updating of variational (covariance) parameters, whichh can sometimes improve the fit. Either 0 or 1. Defaults to 0.

***Lambda.start*:** starting value for variances in VA distributions. Defaults to 0.3.

`control.start` A list with the following arguments controlling the starting values:

***starting.val*:** defaults to `"zero"`. See `"gllvm"` for details.

***n.init*:** number of initial runs. Uses multiple runs and picks up the one giving highest log-likelihood value. Defaults to 1.

***n.init.max*:** maximum number of refits try try for n.init without improvement, defaults to 10.

***start.fit*:** object that inherits of class 'gllvm' which can be given as starting parameters.

***MaternKappa*:** Starting value for smoothness parameter of Matern covariance function. Defaults to 3/2.

***scalmax*:** Sets starting value for the scale parameter for the coordinates. Defaults to 10, when the starting value for scale parameter scales the distances of coordinates between 0-10.

***rangeP*:** Sets starting value for the range parameter for the correlation structure.

***zetacutoff*:** A vector of length 2. Sets starting value for the cutoff parameters of the ordered beta model.

***start.optimizer*:** optimizer for starting value generation, see "optimizer" for more information.

***start.optim.method*:** optimizer method for starting value generation, see "optim.method" for more information.

`...` other arguments passed onto the `"gllvm"` function.

Value

An object of class "glmmVA" that inherits from the "gllvm" class.

Author(s)

Bert van der Veen

See Also

gllvm

Examples

```
data(eSpider)
data <- cbind(data.frame(y = c(eSpider$abund[eSpider$nonNA,]),
           species = factor(rep(1:ncol(eSpider$abund), each = length(eSpider$nonNA))),
           site = factor(rep(1:length(eSpider$nonNA), ncol(eSpider$abund)))),
         do.call(rbind, replicate(ncol(eSpider$abund), scale(eSpider$X[eSpider$nonNA,]),
           simplify = FALSE)))
# Example 1: crossed random slope effects
model <- glmmVA(y~species + ConWate + ConHumu + (0+ConHumu|species) + (0+ConWate|species),
                family = "poisson", data = data)

# Example 2: correlated random slopes
model1 <- glmmVA(y~species + ConWate + ConHumu + (0+ConWate+ConHumu|species),
                 family = "poisson", data = data)

# Example 3: joint model for mixed Poisson and and negative-binomial responses.
fam <- c(rep("poisson", 28), rep("negative.binomial", nrow(data)-28))
model2 <- update(model1, family=fam, sd.errors=FALSE)
```

---

goodnessOfFit *Goodness of fit measures for a gllvm*

Description

Several goodness-of-fit measure are currently available and can be calculated for a gllvm model fit and predicted values.

Usage

```
goodnessOfFit(
  object = NULL,
  y = NULL,
  pred = NULL,
  measure = c("cor", "RMSE", "MAE", "MARNE"),
  species = FALSE
)
```

Arguments

object	an object of class 'gllvm', to calculate goodness of a model fit.
y	a response matrix of new observations

`pred` predicted values for response matrix y if you want to calculate prediction accuracy for new values. Note that for ordinal model, you need to give the predicted classes.

`measure` a goodness-of-fit measure to be calculated. Options are `"cor"` (correlation between observed and predicted values), `"scor"` (Spearman correlation between observed and predicted values), `"RMSE"` (root mean squared error of prediction), `"MAE"` (Mean Absolute Error), `"MARNE"` (Mean Absolute Range Normalized Error), `"TjurR2"` (Tjur's R2 measure, only for binary data), `"R2"` (R-squared as the square of the correlation), "AUC", `"sR2"` (R-squared as the square of the spearman correlation). Likelihood based pseudo R2 meaures `"NagelkerkeR2"`, `"McFaddenR2"`, `"CoxSnellR2"` can be calculated currently only for training data to measure the model's goodness of fit for full data, not response specific.

`species` logical, if `TRUE`, goodness-of-fit measures are calculated for each species separately. If FALSE, goodness-of-fit measures are calculated for all species together.

Details

goodnessOfFit is used for evaluating the goodness-of-fit of a model or predictions. Available goodness-of-fit measures are correlation, RMSE, MARNE, and R2 measures. Definitions are below. Denote an observed response j (species) at sample i, $i = 1, ..., n$, as y_{ij}, and predicted value as $\hat{y}_{ij}$.

$$RMSE(\boldsymbol{y_j}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}) = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(y_{ij} - \hat{y}_{ij})^2}$$

$$MAE(\boldsymbol{y_j}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}) = \frac{1}{n}\Sigma_{i=1}^{n}|y_{ij} - \hat{y}_{ij}|$$

$$MARNE(\boldsymbol{y_j}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}) = \frac{1}{n}\Sigma_{i=1}^{n}\frac{|y_{ij} - \hat{y}_{ij}|}{max(\boldsymbol{y_j}) - min(\boldsymbol{y_j})}$$

$$Tjur'sR2(\boldsymbol{y_j}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}) = \frac{1}{n_1}\Sigma\hat{y}_{ij}\mathbf{1}_{y=1}(y_{ij}) - \frac{1}{n_0}\Sigma\hat{y}_{ij}\mathbf{1}_{y=0}(y_{ij})$$

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>

See Also

`gllvm`, `predict.gllvm`

Examples

```
## Not run:
# Fit gllvm model with Poisson family
data(microbialdata)
X <- microbialdata$Xenv
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                     decreasing = TRUE)[21:40]]
fit <- gllvm(y, X, formula = ~ pH + Phosp, family = poisson())
# Calculate metrics
goodnessOfFit(object = fit, measure = c("cor", "RMSE"))


## End(Not run)
```

`kelpforest` *Kelp Forest community Dynamics: Cover of sessile organisms, Uniform Point Contact*

Description

These data describe the cover of sessile invertebrates, understory macroalgae, and bottom substrate types as determined by a uniform point contact method. The presence of over 150 taxa of sessile invertebrates and macroalgae are recorded at 80 uniformly spaced points along permanent 40m x 2m transects. Multiple species can be recorded at any given point. Percent cover of a given species on a transect can be estimated from UPC observations as the fraction of total points at which that species was present x 100. The total percent cover of all species combined using this method can exceed 100%; however, the percent cover of any single species cannot exceed 100%. This specific version of the data includes 61 species of macroalgae, 69 species of sessile invertebrates, and two species of plants.

Usage

```
data(kelpforest)
```

Format

Y A data frame with the percent cover records of 132 sessile invertebrates and understory macroalgae measured at 836 permanent transects.

X A data frame of study design variables and predictors.

- **SITE** Kelp forest site
- **YEAR** Sampling year
- **TRANSECT** Permanent transect identifying number (unique within site), nested within sites
- **KELP_FRONDS** A number of stipes of giant kelp
- **PERCENT_ROCKY** percent rock coverage

SPinfo A data frame of the species information including species names, group and taxonomy.

- **SP_CODE** Species codes corresponding species names in abundance matrix

GROUP Species group; algae, invertebrate or plant

COMMON_NAME, SCIENTIFIC_NAME Species' common and scientific names

TAXON_KINGDOM,TAXON_PHYLUM,TAXON_CLASS,TAXON_ORDER,TAXON_FAMILY, TAXON_GE Species taxonomic information

Details

These data are part of SBC LTERs kelp forest monitoring program, which began in 2000 and was designed to track long-term patterns in species abundance and diversity of reef-associated organisms in the Santa Barbara Channel, California, USA. The sampling locations in this dataset include nine reef sites along the mainland coast of the Santa Barbara Channel and at two sites on the north side of Santa Cruz Island. These sites reflect several oceanographic regimes in the channel and vary in distance from sources of terrestrial runoff. Data collection began in 2000 and this dataset is updated annually.

The time period of data collection varied among the 11 kelp forest sites. Sampling at BULL, CARP, and NAPL began in 2000, sampling at the other 6 mainland sites (AHND, AQUE, IVEE, GOLB, ABUR, MOHK) began in 2001 (transects 3, 5, 6, 7, 8 at IVEE were added in 2011). Data collection at the two Santa Cruz Island sites (SCTW and SCDI) began in 2004.

Detailed description of the data available in the reference and the website https://sbclter.msi.ucsb.edu/data/catalog/package/?p
lter-sbc.15

References

Reed, D, R. Miller. 2023. SBC LTER: Reef: Kelp Forest Community Dynamics: Cover of sessile organisms, Uniform Point Contact ver 33. Environmental Data Initiative. https://doi.org/10.6073/pasta/0af1a5b0d9dde5b4e5
(Accessed: 2023-12-01).

Examples

```
## Not run:
data(kelpforest)
Y <- kelpforest$Y
X <- kelpforest$X
SPinfo <- kelpforest$SPinfo

## End(Not run)
```

---

logLik.gllvm *Log-likelihood of gllvm*

Description

Extracts Log-likelihood from 'gllvm' objects.

Usage

```
## S3 method for class 'gllvm'
logLik(object, ...)
```

Arguments

object	an object of class 'gllvm'.
...	not used.

Author(s)

David I. Warton, Jenni Niku

Examples

```
## Not run:
## Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
# Fit gllvm model
fit <- gllvm(y = y, family = poisson())
# log-Likelihood:
logLik(fit)

## End(Not run)
```

---

microbialdata *Microbial community data*

Description

Microbial community data consist of abundances of 985 bacteria species measured at 56 soil sample sites from three regions, Kilpisjarvi (Finland), Ny-Alesund (Norway), and Mayrhofen (Austria). In addition to bacteria counts, three continuous environmental variables (pH, available phosphorous and soil organic matter) were measured from each soil sample.

Usage

```
data(microbialdata)
```

Format

Y A data frame with abundances of 985 bacteria species measured at 56 soil sample sites

X Environmental variables SOM: soil organic matter, pH: soil pH value and Phosp: available phosphorus and information from the samples, including Region: sampling region (Kilpisjarvi (Finland), Ny-Alesund (Norway), and Mayrhofen (Austria).), Site: sampling site and Soiltype: soil sample type (top soil (T) or bottom soil (B))

References

Kumar, M., Brader, G., Sessitsch, A., Mäki, A., van Elsas, J.D., and Nissinen, R. (2017). Plants Assemble Species Specific Bacterial Communities from Common Core Taxa in Three Arcto-Alpine Climate Zones. Frontiers in Microbiology, 8:12.

Niku, J., Warton, D. I., Hui, F. K. C., and Taskinen, S. (2017). Generalized linear latent variable models for multivariate count and biomass data in ecology. Journal of Agricultural, Biological, and Environmental Statistics, 22:498-522.

`ordiplot.gllvm` *Plot latent variables from gllvm model*

Description

Plots latent variables and their corresponding coefficients (biplot).

Usage

```
## S3 method for class 'gllvm'
ordiplot(
  object,
  biplot = FALSE,
  ind.spp = NULL,
  alpha = 0.5,
  main = NULL,
  which.lvs = c(1, 2),
  predict.region = FALSE,
  level = 0.95,
  jitter = FALSE,
  jitter.amount = 0.2,
  s.colors = 1,
  s.cex = 1.2,
  symbols = FALSE,
  cex.spp = 0.7,
  spp.colors = "blue",
  arrow.scale = 0.8,
  arrow.spp.scale = 0.8,
  arrow.ci = TRUE,
  arrow.lty = "solid",
  fac.center = FALSE,
  spp.arrows = NULL,
  spp.arrows.lty = "dashed",
  cex.env = 0.7,
  lab.dist = 0.1,
  lwd.ellips = 0.5,
  col.ellips = 4,
  lty.ellips = 1,
```

```
  type = NULL,
  rotate = TRUE,
  ...
)
```

Arguments

object	an object of class 'gllvm'.
biplot	TRUE if both latent variables and their coefficients are plotted, FALSE if only latent variables.
ind.spp	the number of response variables (usually, species) to include on the biplot. The default is none, or all if biplot = TRUE.
alpha	a numeric scalar between 0 and 1 that is used to control the relative scaling of the latent variables and their coefficients, when constructing a biplot.
main	main title.
which.lvs	indices of two latent variables to be plotted if number of the latent variables is more than 2. A vector with length of two. Defaults to c(1,2).
predict.region	if TRUE or "sites" prediction regions for the predicted latent variables are plotted, defaults to FALSE. EXTENSION UNDER DEVELOPMENT: if "species" uncertainty estimate regions for the estimated latent variable loadings are plotted. Works only if biplot = TRUE.
level	level for prediction regions.
jitter	if TRUE, jittering is applied on points.
jitter.amount	numeric, positive value indicating an amount of jittering for each point, defaults to 0.2 (jitter range).
s.colors	colors for sites
s.cex	size of site labels
symbols	logical, if TRUE sites are plotted using symbols, if FALSE (default) site numbers are used
cex.spp	size of species labels in biplot
spp.colors	colors for species, defaults to "blue"
arrow.scale	positive value, to scale arrows
arrow.spp.scale	positive value, to scale arrows of species
arrow.ci	represent statistical uncertainty for arrows in constrained or concurrent ordination using confidence or prediction interval? Defaults to TRUE
arrow.lty	linetype for arrows in constrained ordination
fac.center	logical. If TRUE place labels for binary variables at their estimated location.
spp.arrows	plot species scores as arrows if outside of the range of the plot? Defaults to FALSE for linear response models and TRUE for quadratic response models.
spp.arrows.lty	linetype for species arrows
cex.env	size of labels for arrows in constrained ordination

lab.dist	distance between label and arrow heads. Value between 0 and 1
lwd.ellips	line width for prediction ellipses. See graphical parameter lwd.
col.ellips	colors for prediction ellipses.
lty.ellips	line type for prediction ellipses. See graphical parameter lty.
type	which type of ordination plot to construct. Options are "residual", "conditional", and "marginal". Defaults to "residual" for GLLVMs with unconstrained latent variables and "conditional" otherwise.
rotate	logical, if TRUE (default) latent variables are rotated to their principal direction using singular value decomposition
...	additional graphical arguments.

Details

Function constructs a scatter plot of two latent variables, i.e. an ordination plot. Latent variables are re-rotated to their principal direction using singular value decomposition, so that the first plotted latent variable does not have to be the first latent variable in the model. If only one latent variable is in the fitted model, latent variables are plotted against their corresponding row indices. The latent variables are labeled using the row index of the response matrix y.

Coefficients related to latent variables are plotted in the same figure with the latent variables if biplot = TRUE. They are labeled using the column names of y. The number of latent variable coefficients to be plotted can be controlled by ind.spp. An argument alpha is used to control the relative scaling of the latent variables and their coefficients. If alpha = 0.5, the latent variables and their coefficients are on the same scale. For details for constructing a biplot, see Gabriel (1971).

For a quadratic response model, species optima are plotted. Any species scores that are outside the range of the predicted site scores are not directly plotted, but their main direction is indicated with arrows instead. This ensures that the plot remains on a reasonable scale.

Effects of environmental variables in constrained ordination are indicated with arrows. If any of the arrows exceeds the range of the plot, arrows are scaled to 80 but so that the relative contribution of predictors is maintained. If standard errors are available in the provided model, the slopes of environmental variables for which the 95 are slightly less intensely coloured.

For constrained ordination, a conditional plot includes both fixed- and random-effects to optimally represent species co-occurrence patterns, corresponding to "conditional" site scores in getLV.gllvm. Marginal corresponds to an ordination plot that excludes residual patterns (i.e. excluding the random-effect), so that it is only available with num.lv.c>0 or num.RR>0. A conditional plot requires num.lv.c>0. The "residual" type corresponds to an ordination diagram of only residual patterns. See getLV.gllvm for details.

Note

- If error is occurred when using ordiplot(), try full name of the function ordiplot.gllvm() as functions named 'ordiplot' might be found in other packages as well.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>, Francis K.C. Hui, Bert van der Veen

References

Gabriel, K. R. (1971). The biplot graphic display of matrices with application to principal component analysis. Biometrika, 58, 453-467.

See Also

getLV.gllvm.

Examples

```
#'# Extract subset of the microbial data to be used as an example
data(microbialdata)
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                     decreasing = TRUE)[21:40]]
fit <- gllvm(y, family = poisson())
fit$logL
ordiplot(fit, predict.region = TRUE)
## Not run:
#'## Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
fit <- gllvm(y, family = poisson())
# Ordination plot:
ordiplot(fit)
# Biplot with 10 species
ordiplot(fit, biplot = TRUE, ind.spp = 10)

## End(Not run)
```

---

phyloplot.gllvm *Plot phylogenetic random effects from gllvm*

Description

Plots phylogenetic random effects with the phylogeny, and community effects

Usage

```
## S3 method for class 'gllvm'
phyloplot(
  object,
  tree = NULL,
  comm.eff = TRUE,
  row.eff = FALSE,
  which.Xcoef = NULL,
  xlim = NULL,
  level = 0.95,
  col = c("#E69F00", "white", "#009E73"),
```

```
  col.sym = TRUE,
  tick.length = 2,
  mar.spec = c(3, 2, 0, 0),
  mar.phy = c(0, 2, 2, 0),
  mar.comm = c(3, 0.5, 2, 1.5),
  cex = 0.6,
  lwd = 1,
  col.edge = "black",
  pch = "x",
  heights = c(0.55, 0.35),
  widths = c(0.64, 0.1),
  phy.place = "top",
  ...
)
```

Arguments

object	an object of class 'gllvm'.
tree	an object of class "
comm.eff	logical, defaults to TRUE. If present in the model, should community effects be plotted?
row.eff	logical, defaults to FALSE. If present in the model, should row effects (e.g., community responses to covariates) be included?
which.Xcoef	List of length 2 with names to subset the effects to plot. The first vector is for the species plot, the second for community effects.
xlim	vector of length two. Limits for the x-axis of the caterpillar plot. Defaults to NULL, in which case the limits are chosen based on the confidence intervals.
level	the confidence level. Scalar between 0 and 1.
col	vector of three colors (defaults to c("#E69F00","white","#009E73")) passed to colorRampPalette for species random effects.
col.sym	logical, defaults to TRUE. Then, the color scale of the species random effects plot is symmetrical (so that zero is nearly in the middle), so that both the lower and upper limit are determined by the largest absolute value. If FALSE, the lower and upper limits are determined by the smallest and largest values, respectively.
tick.length	numerical, defaults to 2. Length of ticks for the species-specific random effects plot.
mar.spec	vector of length 4, which defines the margins sizes for the species random effects plot. Defaults to c(3, 2, 0, 0).
mar.phy	vector of length 4, which defines the margins sizes for plotting the phylogeny. Defaults to c(0, 2, 2, 0).
mar.comm	vector of length 4, which defines the margins sizes for the caterpillar plot. Defaults to c(3, 0.5, 2, 1.5).
cex	the magnification to be used for text in the plot. Defaults to 0.6.
lwd	line thickness for the branches in the phylogeny and the confidence intervals in the caterpillar plot. Defaults to 1.

col.edge	character. Color of branches in the phylogeny.
pch	symbol used in the catter pillar plot. Defaults to "x".
heights	vector of length two. Relative row heights, defaults to c(0.55, 0.35).
widths	vector of length two. Relative column widths, defaults to c(0.64, 0.10).
phy.place	not (yet) in use.
...	additional not in use.

Details

Plots phylogenetically structured random effects together with the phylogeny, and with community-level effects (i.e., effects that are the same across species). If standard errors have been calculated for the model, the prediction intervals for species random effects are checked, and crossed out (i.e., displayed as white) if they cross zero.

Author(s)

Bert van der Veen

References

van der Veen, B., O'Hara, R.B. (2024). Fast fitting of Fast fitting of phylogenetic mixed effects models. arXiv.

Examples

```
## Not run:
# Load dataset
data(fungi)
Y <- fungi$Y
# Scale the predictor
X <- fungi$X
X[,"DBH.CM"] <- scale(X[, "DBH.CM"])
tree <- fungi$tree # the tree
colMat <- fungi$C # e.g., from ape::vcv(tree)
dist <- fungi$dist # e.g., from ape::cophenetic.phylo(tree)
order <- gllvm:::findOrder(covMat = colMat, distMat = dist, nn = 15,
                           order = order(dist[1:length(tree$tip.label), nrow(dist)],
                           decreasing = TRUE))$order
order <- tree$tip.label[order]
model <- gllvm(y = Y[,order], X = X,
               formula = ~(DBH.CM|1), beta0com = TRUE,
               family = "binomial", num.lv = 0, nn.colMat = 15,
               colMat = list(colMat[order,order], dist = dist[order,order]),
               colMat.rho.struct = "term")
phyloplot(model, tree)

## End(Not run)
```

plot.gllvm *Plot Diagnostics for an gllvm Object*

Description

Five plots (selectable by which) are currently available: a plot of residuals against linear predictors of fitted values, a Normal Q-Q plot of residuals with a simulated point-wise 95% confidence interval envelope, residuals against row index and column index and scale location plot.

Usage

```
## S3 method for class 'gllvm'
plot(
  x,
  which = 1:5,
  caption = c("Residuals vs linear predictors", "Normal Q-Q", "Residuals vs row",
    "Residuals vs column", "Scale-Location"),
  var.colors = NULL,
  add.smooth = TRUE,
  envelopes = TRUE,
  reps = 150,
  envelope.col = c("blue", "lightblue"),
  n.plot = NULL,
  spp = NULL,
  ...
)
```

Arguments

x	an object of class 'gllvm'.
which	if a subset of the plots is required, specify a subset of the numbers 1:5, see caption below. For objects of class "glmmVA" it should be a subset of 1:3.
caption	captions to appear above the plots.
var.colors	colors for responses, vector with length of number of response variables or 1. Defaults to NULL, when different responses have different colors.
add.smooth	logical with default TRUE. Indicates if a smoother should be added.
envelopes	logical, indicating if simulated point-wise confidence interval envelope will be added to Q-Q plot, defaults to TRUE
reps	number of replications when simulating confidence envelopes for normal Q-Q plot
envelope.col	colors for envelopes, vector with length of two
n.plot	number of species (response variables) to be plotted. Defaults to NULL when all response variables are plotted. Might be useful when data is very high dimensional.

spp	integer vector or NULL, defaults to NULL. If provided, residuals are computed only for the selected species (column indices in response matrix).
...	additional graphical arguments.

Details

plot.gllvm is used for model diagnostics. Dunn-Smyth residuals (randomized quantile residuals) (Dunn and Smyth, 1996) are used in plots. Colors indicate different species.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>

References

Dunn, P. K., and Smyth, G. K. (1996). Randomized quantile residuals. Journal of Computational and Graphical Statistics, 5, 236-244.

Hui, F. K. C., Taskinen, S., Pledger, S., Foster, S. D., and Warton, D. I. (2015). Model-based approaches to unconstrained ordination. Methods in Ecology and Evolution, 6:399-411.

See Also

gllvm, residuals.gllvm

Examples

```
## Not run:
# Fit gllvm model with Poisson family
data(microbialdata)
X <- microbialdata$Xenv
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                     decreasing = TRUE)[21:40]]
fit <- gllvm(y, X, formula = ~ pH + Phosp, family = poisson())
# Plot residuals
plot(fit, mfrow = c(3,2))


## End(Not run)
```

---

predict.gllvm *Predict Method for gllvm Fits*

Description

Obtains predictions from a fitted generalized linear latent variable model object.

Usage

```
## S3 method for class 'gllvm'
predict(
  object,
  newX = NULL,
  newTR = NULL,
  newLV = NULL,
  type = "link",
  level = 1,
  offset = TRUE,
  se.fit = FALSE,
  alpha = 0.95,
  seed = 42,
  ordinal.cat = NULL,
  spp = NULL,
  ...
)
```

Arguments

object	an object of class 'gllvm'.
newX	A new data frame of environmental variables. Should include all relevant variables from the studyDesign and X data frames. If omitted, the original matrices are used.
newTR	A new data frame of traits for each response taxon. If omitted, the original matrix of traits is used.
newLV	A new matrix of latent variables. If omitted, the original matrix of latent variables is used. Note that number of rows/sites must be the same for newX (if X covariates are included in the model).
type	the type of prediction required. The default ("link") is on the scale of the linear predictors; the alternatives are "response" and "class" (predicted classes, works only for binomial model with 1 trial or ordinal model). "response" is on the scale of the response variable. That is, the predictions for the binomial model are predicted probabilities. In case of ordinal data, type = "response" gives predicted probabilities for each level of ordinal variable.
level	specification for how to predict. Level one (level = 1) attempts to use the predicted site scores from variational approximations or laplace approximation or given site scores in newLV. Level 0 sets the latent variable to zero. Defaults to 1.
offset	specification whether of not offset values are included to the predictions in case they are in the model, defaults to TRUE when offset values that are used to fit the model are included to the predictions. Alternatives are matrix/vector (number of rows must match with the newX) of new offset values or FALSE, when offsets are ignored.
se.fit	logical. If TRUE, performs 1000 simulations from the asymptotic covariance matrix for fixed effects, and from the CMSEP covariance matrix for the random effects. If an integer is provided, it is used as the number of simulations instead.

alpha	numerical between 0 and 1, defaults to 0.95. The confidence level for se.fit.
seed	numeric, defaults to 42. Seed used for simulation in se.fit.
ordinal.cat	integer or NULL, defaults to NULL. For ordinal models with type = "response", selects a single category to return instead of the full K×n×p array. When set, the return value is an n×p matrix of probabilities for that category. Primarily used internally by predictSR and goodnessOfFit to reduce memory use.
spp	integer vector or NULL, defaults to NULL. If provided, predictions are computed only for the selected species (column indices into the response matrix), returning an $n \times$ length(spp) matrix. All parameter-level computations (beta0, Xcoef, theta, etc.) are subsetted accordingly, so no unnecessary computation is performed for the remaining species.
...	not used.

Details

If newX, newTR and newLV are omitted the predictions are based on the data used for fitting the model. Notice that newTR need to match with the number of species in the original data. Instead, new sites can be specified in newX. If predictors newX (and newTR) are given, and newLV is not, latent variables are not used in the predictions.

Value

A matrix containing requested predictor types.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>, David Warton

Examples

```
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
X <- scale(antTraits$env[, 1:3])
# Fit gllvm model
fit <- gllvm(y = y, X, family = poisson())
# fitted values
predfit <- predict(fit, type = "response")

# linear predictors
predlin <- predict(fit)
# Predict new sites:
# Generate matrix of environmental variables for 10 new sites
xnew <- cbind(rnorm(10), rnorm(10), rnorm(10))
colnames(xnew) <- colnames(X)
predfit <- predict(fit, newX = xnew, type = "response", level = 0)

TR <- (antTraits$tr[, 1:3])
fitt <- gllvm(y = y, X, TR, family = poisson())
# linear predictors
```

```
predlin <- predict(fitt)
# Predict new sites:
# Generate matrix of environmental variables for 10 new sites
xnew <- cbind(rnorm(10), rnorm(10), rnorm(10))
colnames(xnew) <- colnames(X)
# Generate matrix of traits for species
trnew <- data.frame(Femur.length = rnorm(41), No.spines = rnorm(41),
 Pilosity = factor(sample(0:3, 41, replace = TRUE)))
predfit <- predict(fitt, newX = xnew, newTR = trnew, type = "response", level = 0)
```

`predict.glmmVA` *Predict method for glmmVA objects*

Description

Wrapper around `predict.gllvm` that, for mixed-response models (those fitted with a per-observation `family` vector), collapses the internal wide-format prediction matrix back to a long-format numeric vector aligned with the original observations.

Usage

```
## S3 method for class 'glmmVA'
predict(object, ...)
```

Arguments

`object`	an object of class `"glmmVA"`.
`...`	further arguments passed to `predict.gllvm`.

Value

For univariate models: the same object returned by `predict.gllvm`. For mixed-response models: a numeric vector of length `nobs(object)`.

Author(s)

Bert van der Veen

See Also

`predict.gllvm`, `glmmVA`

predictLVs.gllvm *Predict latent variables for gllvm Fits*

Description

Obtains predictions for latent variables from a fitted generalized linear latent variable model object. Currently works only for the variational approximation method.

Usage

```
## S3 method for class 'gllvm'
predictLVs(object, newX = NULL, newY = object$y, ...)
```

Arguments

object	an object of class 'gllvm'.
newX	A new data frame of environmental variables. If omitted, the original matrix of environmental variables is used.
newY	A new response data. Defaults to the dataset used for original model fit.
...	not used.

Details

Obtains predictions for latent variables from a fitted generalized linear latent variable model object.

Value

A matrix containing requested predictor types.

Author(s)

David Warton, Jenni Niku <jenni.m.e.niku@jyu.fi>

Examples

```
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
X <- scale(antTraits$env[, 1:3])
# Fit gllvm model
fit <- gllvm(y = y, X, family = poisson())
# fitted values
predLVs <- predictLVs.gllvm(fit)
```

predictPairwise.gllvm *Predict co-occurrence from a gllvm object*

Description

Obtains the probability of co-occurrnece for species from a fitted generalized linear latent variable model object.

Usage

```
## S3 method for class 'gllvm'
predictPairwise(
  object,
  spp = NULL,
  se.fit = FALSE,
  alpha = 0.95,
  seed = 42,
  ...
)
```

Arguments

object	an object of class 'gllvm'.
spp	a vector of length 2, defaults to NULL. If NULL, returns the probability of co-occurrence for all species in the data.
se.fit	logical, defaults to FALSE. If set to TRUE, returns a simulated confidence interval of the prediction.
alpha	numeric between 0 and 1, defaults to 0.95. Confidence level of the prediction.
seed	numeric, defaults to 42. Seed for the simulation of the confidence interval.
...	arguments passed to "predict.gllvm".

Details

The probability of co-occurrence is simply given by the joint probability of getting two presences for any pair of species. As such, it is simply calculated as the product of the probability of occurrence for any pair of species, so that this function mostly serves as convenience wrapper around "predict.gllvm".

Value

A matrix of size np(p-1)/2 by 2.

Author(s)

Bert van der Veen

See Also

predict.gllvm, predictSR.gllvm

Examples

```
# Load a dataset from the mvabund package
data(spider, package = "mvabund")
y <- as.matrix(spider$abund)
X <- scale(spider$x)
# Fit gllvm model
fit <- gllvm(y = y, X, formula = ~soil.dry, family = poisson())
# fitted values
newX = data.frame(soil.dry = seq(min(X[,"soil.dry"]),
                 max(X[,"soil.dry"]), length.out=100))
predPair <- predictPairwise(fit, spp = c(1,2), newX = newX, level = 0)
plot(x = newX$soil.dry, y = predPair$prob, type = "l",
    xlab = "Soil dry moisture content",
    ylab = "Probability of co-occurrence", ylim = c(0,1))
```

---

predictSR.gllvm *Predict species richness from a gllvm object*

Description

Obtains predictions from a fitted generalized linear latent variable model object.

Usage

```
## S3 method for class 'gllvm'
predictSR(
  object,
  spp = NULL,
  expected = "mean",
  se.fit = 1000,
  level = 1,
  ci = "expected",
  alpha = 0.95,
  seed = 42,
  return.pred = FALSE,
  batch = NULL,
  ...
)

## S3 method for class 'predictSR.gllvm'
plot(x, object, which = c(1, 2), ...)
```

Arguments

object	object of class gllvm
spp	index vector, defaults to NULL. If omitted, returns a prediction with all species in the data.
expected	character, defaults to "mean", in which case the returned measure of central tendency for species richness is the Poisson-Binomial expectation. Alternatively can be "mode".
se.fit	integer or logical, defaults to 1000. Number of simulations for confidence interval. No confidence interval is returned when set to FALSE.
level	specification for how to predict. Level one (level = 1) attempts to use the predicted site scores from variational approximations or laplace approximation or given site scores in newLV. Level 0 sets the latent variable to zero. Defaults to 1.
ci	character vector, defaults to "expected" but can also be "pmf" or both. If "pmf" provides CI on the whole pmf prediction.
alpha	numeric between 0 and 1, defaults to 0.95. Confidence level of the prediction.
seed	numeric, defaults to 42. Seed for the simulation of the confidence interval.
return.pred	logical, defaults to FALSE. Returns the point-estimate results from "predict.gllvm".
batch	integer or NULL, defaults to 200. If provided, species predictions needed to calculate richness are processed in chunks of this size per simulation draw, reducing peak memory at the cost of more predict calls. NULL processes all (selected) species in one call.
...	not used for plot.predictSR.gllvm
x	object of class predictSR
which	which plot to create, 1 is Dunn-Smyth residuals vs Expected species richness, 2 is QQ-plot

Details

This function returns probabilities of richness for species richness from gllvm objects, by first calling "predict.gllvm", and following by predicting from a Poisson-Binomial distribution for richness. The distribution for species richness follows from a sum of binary responses (i.e., occurrence), and is naturally extended to non-normal data types as the probability to get a non-zero observation. Especially with many species and rows, the calculation may take a while.

Value

list with entries "predicted" and "expected". "predicted" includes the prediction from the Poisson-Binomial distribution, returning a matrix of size sites by length(SR), with statistical uncertianty if se.fit = TRUE. "expected" includes the expected species' richness, i.e., a vector of size n, with statistical uncertianty if se.fit = TRUE.

Note

The point estimate ("fit") can sometimes be outside of the simulated intervals when the model fit is poor, the standard errors of parameter estimates are on very different scales, or the number of simulations is too low. Try either 1) scaling and centering covariates in the model, 2) repeatedly refitting the model (see the n.init argument in "gllvm") to find a better set of starting values and improve the fit, or 3) increasing the number of simulations via 'n.init'.

Author(s)

Bert van der Veen

See Also

predict.gllvm, residuals.predictSR.gllvm

Examples

```
# Load a dataset from the mvabund package
data(spider, package = "mvabund")
y <- as.matrix(spider$abund)
X <- scale(spider$x)
# Fit gllvm model
fit <- gllvm(y = y, X, formula = ~soil.dry, family = poisson())
# Fitted values
newX = data.frame(soil.dry = seq(min(X[,"soil.dry"]),
                 max(X[,"soil.dry"]),length.out=100))
predSR <- predictSR(fit, newX = newX, level = 0)

# Visualize the results
par(mfrow = c(2,2))
for(i in 0:ncol(fit$y)){
plot(x = newX$soil.dry, y = predSR$predicted$fit[,i+1],
     xlab = "Soil dry matter content",
     ylab = bquote(p(SR == .(i))), type = "l", ylim = c(0,1))
}
```

---

randomCoefplot.gllvm *Plot random slope coefficients*

Description

Plots random slopes and their prediction intervals.

Usage

```
## S3 method for class 'gllvm'
randomCoefplot(
  object,
  y.label = TRUE,
  which.Xcoef = NULL,
  cex.ylab = 0.5,
  mfrow = NULL,
  mar = c(4, 6, 2, 1),
  xlim.list = NULL,
  order = FALSE,
  ind.spp = NULL,
  ...
)
```

Arguments

object	an object of class 'gllvm'.
y.label	logical, if TRUE (default) colnames of y with respect to coefficients are added to plot.
which.Xcoef	fector indicating which covariate coefficients will be plotted. Can be vector of covariate names or numbers. Default is NULL when all covariate coefficients are plotted.
cex.ylab	the magnification to be used for axis annotation relative to the current setting of cex.
mfrow	same as mfrow in par. If NULL (default) it is determined automatically.
mar	vector of length 4, which defines the margin sizes: c(bottom, left, top, right). Defaults to c(4,5,2,1).
xlim.list	list of vectors with length of two to define the intervals for x axis in each covariate plot. Defaults to NULL when the interval is defined by the range of point estimates and confidence intervals
order	logical, if TRUE (default), coefficients are sorted according to the point estimates
ind.spp	vector of species indices to construct the caterpillar plot for
...	additional graphical arguments.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>, Francis K.C. Hui, Bert van der Veen, Sara Taskinen,

Examples

```
## Not run:
## Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
X <- as.matrix(antTraits$env)
```

```
TR <- antTraits$traits
# Fit model with random slopes
fitF <- gllvm(y = y, X = X, TR = TR,
 formula = ~ Bare.ground + Bare.ground : Webers.length,
 family = poisson(), randomX = ~ Bare.ground)
randomCoefplot(fitF)

## End(Not run)
```

---

ranef *Generic function for extracting random effects*

Description

S3 generic dispatched on the model class.

Usage

```
ranef(object, ...)
```

Arguments

object	a fitted model object of class glmmVA.
...	additional arguments passed to the method.

---

ranef.glmmVA *Extract random effects from a glmmVA object*

Description

Returns the random effects estimate for a model fitted with glmmVA, optionally with their conditional variances.

Usage

```
## S3 method for class 'glmmVA'
ranef(object, condVar = FALSE, ...)
```

Arguments

object	an object of class "glmmVA".
condVar	logical; if TRUE, attach conditional variances as an attribute "condVar" on the returned object. Defaults to FALSE.
...	not used.

Value

A vector of random-effect estimates. When `condVar = TRUE`, the attribute `"condVar"` is a vector of conditional variances. returned by `getPredictErr`.

Author(s)

Bert van der Veen

See Also

`glmmVA`, `getPredictErr`

`residuals.gllvm` *Dunn-Smyth residuals for gllvm model*

Description

Calculates Dunn-Smyth residuals for gllvm model.

Usage

```
## S3 method for class 'gllvm'
residuals(object, ...)
```

Arguments

`object`	an object of class 'gllvm'.
`...`	not used.

Details

Computes Dunn-Smyth residuals (randomized quantile residuals, Dunn and Smyth, 1996) for gllvm model. For the observation Y_{ij} Dunn-Smyth residuals are defined as

$$r_{ij} = \Phi^{-1}(u_{ij}F_{ij}(y_{ij}) + (1 - u_{ij})F_{ij}^{-}(y_{ij})),$$

where $\Phi(.)$ and $F_{ij}(.)$ are the cumulative probability functions of the standard normal distribution and distribution of the responses, respectively, $F_{ij}^{-}(y))$ is the limit as $F_{ij}(y)$ is approached from the negative side, and u_{ij} has been generated at random from the standard uniform distribution.

Value

`residuals`	matrix of residuals
`linpred`	matrix of linear predictors

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>

References

Dunn, P. K., and Smyth, G. K. (1996). Randomized quantile residuals. Journal of Computational and Graphical Statistics, 5, 236-244.

Hui, F. K. C., Taskinen, S., Pledger, S., Foster, S. D., and Warton, D. I. (2015). Model-based approaches to unconstrained ordination. Methods in Ecology and Evolution, 6:399-411.

Examples

```
## Not run:
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
# Fit gllvm model
fit <- gllvm(y = y, family = poisson())
# residuals
res <- residuals(fit)

## End(Not run)
```

`residuals.predictSR.gllvm` *Dunn-Smyth residuals for species richness from a gllvm object*

Description

Calculates Dunn-Smyth residuals for species richness.

Usage

```
## S3 method for class 'predictSR.gllvm'
residuals(object, model, ...)
```

Arguments

`object`	object returned by "`predictSR.gllvm`"
`model`	an object of class 'gllvm'.
`...`	not used.

Details

See "`residuals.gllvm`" for details.

Value

A list of length 2 with 1) the fitted and 2) the dunn-smyth residuals.

Author(s)

Bert van der Veen

See Also

predict.gllvm, predictSR.gllvm, residuals.gllvm

---

se.gllvm *Standard errors for gllvm model*

Description

Calculates Hessian and standard errors for gllvm model.

Usage

```
## S3 method for class 'gllvm'
se(object, ...)
```

Arguments

object	an object of class 'gllvm'.
...	not used.

Details

Computes Hessian and standard errors for gllvm model.

Value

sd	list of standard errors of parameters
Hess	list including Hessian matrix and approximative covariance matrix of parameters

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>

References

Dunn, P. K., and Smyth, G. K. (1996). Randomized quantile residuals. Journal of Computational and Graphical Statistics, 5, 236-244.

Hui, F. K. C., Taskinen, S., Pledger, S., Foster, S. D., and Warton, D. I. (2015). Model-based approaches to unconstrained ordination. Methods in Ecology and Evolution, 6:399-411.

Examples

```
data(eSpider)
mod <- gllvm(eSpider$abund, num.lv = 2, family = "poisson", sd.errors = FALSE)
# Calculate standard errors after fitting
sdErr <- se(mod)
# Store the standard errors in the right place
mod$sd <-sdErr$sd
# Store the Hessian in the right place
mod$Hess <- sdErr$Hess
```

`simulate.gllvm` *Simulate data from gllvm fit*

Description

Generate new data using the fitted values of the parameters

Usage

```
## S3 method for class 'gllvm'
simulate(object, nsim = 1, seed = NULL, conditional = FALSE, newX = NULL, ...)
```

Arguments

`object`	an object of class 'gllvm'.
`nsim`	an optional positive integer specifying the number of simulated datasets. Defaults to 1.
`seed`	an optional integer to set seed number, passed to set.seed. Defaults to a random seed number.
`conditional`	if `conditional = FALSE` simulates marginally over the latent variables.
`newX`	A new data frame of environmental variables. If omitted, the original matrix of environmental variables is used.
`...`	not used.

Details

simulate function for gllvm objects.

Value

A matrix containing generated data.

Author(s)

David Warton, Jenni Niku <jenni.m.e.niku@jyu.fi>

Examples

```
# Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
X <- scale(antTraits$env[, 1:3])
# Fit gllvm model
fit <- gllvm(y = y, X, family = poisson())
# Simulate data
newdata <- simulate(fit)
```

Skabbholmen *Skabbholmen island data*

Description

Dataset of ordinal observations of plants, on the island Skabbholmen in the Stocholm archipelago. Includes 65 unique sites and 70 species, surveyed in two different years.

Usage

```
data(Skabbholmen)
```

Format

Y A data frame with ordinal of 70 plant species measured at 126 plots.

X A matrix of 2 predictor variables at 126 plots.

species A matrix of full species names and abbreviations used in the community data (Y).

Details

Observations of vascular plant cover in 126 one-square-meter plots divided over four transects. The ordinal responses are on a five-degree Hult-Sernander-Du Rietz scale, and were originally recorded by Wolfgang and Cramer (1987) and additionally analyzed by ter Braak (1987). There is a total of 64 unique sites, that were surveyed in two different years (1978 and 1984), but two plots were only surveyed in one year (thus bringing the total number of rows in the data to 126). The plots were located on an elevation gradient, running from the shoreline to the edge of old-growth forest. Elevation to the shoreline was recorded in centimeters during the sampling in 1978.

This dataset was published with permission from the CANOCO FORTRAN package example datasets.

References

ter Braak, C.J.F. and Smilauer, P. (1998). CANOCO reference manual and user's guide to CANOCO for Windows: software for canonical community ordination (version 4). Microcomputer Power, New York, New York, USA.

ter Braak, C.J.F. (1987). The analysis of vegetation-environment relationships by canonical correspondence analysis. Vegetatio, 69(1), 69-77.

Cramer, W. & Hytteborn, H. (1987). The separation of fluctuation and long-term change in vegetation dynamics of a rising seashore. Vegetatio, 69, 157–167.

Examples

```
## Not run:
data(Skabbholmen)
Y <- Skabbholmen$Y
X <- Skabbholmen$X
model <- gllvm(y = Y, X = X, studyDesign = X,
    lv.formula = ~scale(Elevation) + I(Year-1978),
    num.RR = 2,
    family = "ordinal",
    zeta.struc="common",
    row.eff=~(1|transectID))

## End(Not run)
```

`summary.gllvm` *Summarizing gllvm model fits*

Description

A summary of the fitted 'gllvm' object, including function call, distribution family and model parameters.

Usage

```
## S3 method for class 'gllvm'
summary(
  object,
  by = "all",
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  dispersion = FALSE,
  spp.intercepts = FALSE,
  row.intercepts = FALSE,
  Lvcoefs = FALSE,
  rotate = TRUE,
  type = NULL,
  ...
```

```
)

## S3 method for class 'summary.gllvm'
print(x, ...)

## S3 method for class 'summary.gllvm'
plot(x, component = NULL, ...)
```

Arguments

object	an object of class 'gllvm'
by	By = "all" (default) will return a Wald statistics per predictor and LV if the ordination includes predictors, by = "terms" will return a multivariate Wald statistic per predictor (displayed at first LV), and by = "LV" will do the same but per dimension (displayed at first predictors).
digits	the number of significant digits to use when printing
signif.stars	If TRUE, significance stars are printed for each coefficient, defaults to TRUE
dispersion	option to return dispersion parameters, defaults to FALSE
spp.intercepts	option to return species intercepts, defaults to FALSE
row.intercepts	option to return row intercepts, defaults to FALSE
Lvcoefs	option to return species scores in the ordination, defaults to FALSE. Returns species optima for quadratic model.
rotate	defaults to TRUE. If TRUE rotates the output of the latent variables to principal direction, so that it coincides with the ordiplot results. If both unconstrained and constrained latent variables are included, predictor slopes are not rotated.
type	to match "type" in ordiplot.gllvm
...	not used.
x	a summary object
component	component to be plotted

Details

Various options are available to include extra parameter estimates in the summary, which have been excluded by default, for readability.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>, Bert van der Veen

Examples

```
## Not run:
## Load a dataset from the mvabund package
data(antTraits, package = "mvabund")
y <- as.matrix(antTraits$abund)
# Fit gllvm model
```

```
fit <- gllvm(y = y, family = poisson())
summary(fit)

## Plot canonical coefficients
X = antTraits$env
fit2 <- gllvm(y = y, X = X, family = "negative.binomial", num.RR = 2)
plot(summary(fit2))

## End(Not run)
```

`update.gllvm` *Update and Re-fit a gllvm Model Call*

Description

Models of class gllvm have three formula interfaces: formula, lv.formula, and row.eff. Each can include fixed and random effects.

Usage

```
## S3 method for class 'gllvm'
update(
  object,
  formula = NULL,
  lv.formula = NULL,
  row.eff = NULL,
  eval = TRUE,
  ...
)
```

Arguments

`object`	an object of class 'gllvm'.
`formula`	interface for column-specific effects.
`lv.formula`	interface for latent variable (reduced-rank) effects.
`row.eff`	interface for row-specific (i.e., same for all columns) effects.
`eval`	default to TRUE, in which case the reformulated model is fitted directly. if FALSE returns the new model call.
`...`	not used.

Details

Updates the formula of a gllvm object.

Author(s)

Bert van der Veen

vcov.gllvm *Returns variance-covariance matrix of coefficients in a GLLVM.*

Description

Returns the variance-covariance matrix of the parameters from a GLLVM. If the variance-covariance matrix was not calculated after model fitting, this function will have to calculate the variance-covariance matrix, which may be computational intensive for a large number of species and/or sites.

Usage

```
## S3 method for class 'gllvm'
vcov(object, ...)
```

Arguments

object	an object of class 'gllvm'.
...	not used.

Details

Calculates the variance-covariance matrix of a GLLVM object using se.gllvm, which may be computational intensive with many parameters.The parameters might have unintuitive names. Fixed-effects coefficients are labeled "b", and are ordered per species as: 1) intercepts 2) fixed-effects slopes. Coefficients of the latent variables are labled "lambda" (linear coefficients) or "lambda2".

Author(s)

Bert van der Veen

VP.gllvm *Calculate variance partitioning*

Description

Calculates variance partitioning for gllvm object with function VP() (alias varPartitioning()).

Function plotVarPartitioning() (alias plotVP() or just plot()) plots the results of variance partitioning of a fitted gllvm.

Usage

```
## S3 method for class 'gllvm'
VP(
  object,
  group = NULL,
  groupnames = NULL,
  adj.cov = TRUE,
  grouplvs = FALSE,
  calcr2scaled = FALSE,
  ...
)

## S3 method for class 'VP.gllvm'
print(x, ...)

plotVarPartitioning(
  x,
  main = "Variance Partitioning",
  xlab = "Response",
  ylab = "Variance proportion",
  legend.text = NULL,
  args.legend = list(cex = 0.7, x = "topright", bty = "n", inset = c(0, -0.15)),
  mar = c(4, 4, 6, 2),
  r2scaled = FALSE,
  ...
)

plotVP(x, ...)

## S3 method for class 'VP.gllvm'
plot(x, ...)
```

Arguments

object	an object of class 'gllvm'.
group	a vector of integers identifying grouping of X covariates, the default is to use model terms formula and lv.formula.
groupnames	a vector of strings given as names for the groups defined in group
adj.cov	logical, whether or not to adjust co-variation within the group
grouplvs	logical, whether or not to group latent variables to one group
calcr2scaled	logical, whether or not to also calculate r2 scaled variance partitioning. Defaults to FALSE. If true, squared correlation between data and predictions is used for continuous data (normal, gamma, exponent, beta, tweedie), squared spearman correlation for count and ordinal data, and Tjur's R2 for binary data. Note, interpret these cautiously.
...	additional graphical arguments passed to the barplot function

x	a variance partitioning object for a gllvm produced by function varPartitioning.
main	main title
xlab	a label for the x axis.
ylab	a label for the y axis.
legend.text	a vector of names for the groups, as a default 'groupnames' from varPartitioning. If FALSE, legend not printed.
args.legend	a list of additional arguments to pass to `legend()`.
mar	Margins of the plot. Default `c(4,4,6,2)`
r2scaled	logical, whether or not to plot the response specific R2 -scaled variance partitioning. Defaults to FALSE.

Details

Variance for the linear predictor for response j can be calculated as

$$Var(\eta_j) = \sum_k \beta_{jk}^2 * var(z_{.k}) + 2 \sum_{(k1=1,...,K-1)} \sum_{(k2=k1+1,...,K)} \beta_{j(k1)}\beta_{j(k2)} Cov(Z_{.k1}, Z_{.k2}),$$

where $z_{.k}$ is a vector consisting of predictor/latent variable/row effect etc values for all sampling units i. If $z_{.k}$s are not correlated, covariance term is 0 and thus the variance explained of a response j for predictor $z_{.k}$ is given as $\beta_{jk}^2 * var(z_{.k})/Var(\eta_j)$.

In case of correlated predictors, it is advised to group them into a same group. The variance explained is calculated for the correlated group of predictors together and adjusted with the covariance term.

Author(s)

Jenni Niku <jenni.m.e.niku@jyu.fi>

Examples

```
# Extract subset of the microbial data to be used as an example
data(microbialdata)
X <- microbialdata$Xenv
y <- microbialdata$Y[, order(colMeans(microbialdata$Y > 0),
                     decreasing = TRUE)[21:40]]
fit <- gllvm(y, X[,1:3], formula = ~ pH + Phosp, family = poisson(),
             studyDesign = X[,4:5], row.eff = ~(1|Site))
VP <- VP(fit)
plot(VP)

## Not run:
# Plot the result of  variance partitioning
plot(VP, col = palette(hcl.colors(5, "Roma")), las =2)


## End(Not run)
```

Index

AICc, 3
anova.gllvm, 3

beetle, 4

coefplot *(*coefplot.gllvm*)*, 6
coefplot.gllvm, 6, *32*
colorRampPalette, *46*
confint *(*confint.gllvm*)*, 8
confint.gllvm, 8, *32*

ecoCoefs, 9
eSpider, 9

fungi, 11

getEnvironCor, *13, 14*
getEnvironCor *(*getEnvironCov.gllvm*)*, 12
getEnvironCov *(*getEnvironCov.gllvm*)*, 12
getEnvironCov.gllvm, 12
getLoadings *(*getLoadings.gllvm*)*, 15
getLoadings.gllvm, 15
getLV *(*getLV.gllvm*)*, 15
getLV.gllvm, 15, *44, 45*
getPredictErr, *60*
getPredictErr *(*getPredictErr.gllvm*)*, 16
getPredictErr.gllvm, 16
getResidualCor *(*getResidualCor.gllvm*)*, 17
getResidualCor.gllvm, *14*, 17
getResidualCov *(*getResidualCov.gllvm*)*, 18
getResidualCov.gllvm, *13, 14*, *18*, 18
gllvm, 21, *35–38*, *49*, *57*
glmmVA, 35, *52*, *59, 60*
goodnessOfFit, 37

kelpforest, 39

logLik *(*logLik.gllvm*)*, 40
logLik.gllvm, 40

microbialdata, 41

nlminb, *25*, *35*
nloptr, *25*
nobs *(*AICc*)*, 3

optim, *25*, *35, 36*
optima *(*ecoCoefs*)*, 9
optima.gllvm, *15*
ordiplot *(*ordiplot.gllvm*)*, 42
ordiplot.gllvm, *32*, 42, *66*

phyloplot *(*phyloplot.gllvm*)*, 45
phyloplot.gllvm, 45
plot.gllvm, *32*, 48
plot.predictSR.gllvm *(*predictSR.gllvm*)*, 55
plot.summary.gllvm *(*summary.gllvm*)*, 65
plot.VP.gllvm *(*VP.gllvm*)*, 68
plotVarPartitioning *(*VP.gllvm*)*, 68
plotVP *(*VP.gllvm*)*, 68
predict *(*predict.gllvm*)*, 49
predict.gllvm, *38*, 49, *52*, *54–57*, *62*
predict.glmmVA, 52
predictLVs *(*predictLVs.gllvm*)*, 53
predictLVs.gllvm, 53
predictPairwise *(*predictPairwise.gllvm*)*, 54
predictPairwise.gllvm, 54
predictSR *(*predictSR.gllvm*)*, 55
predictSR.gllvm, *55*, 55, *61, 62*
print.summary.gllvm *(*summary.gllvm*)*, 65
print.VP.gllvm *(*VP.gllvm*)*, 68

randomCoefplot *(*randomCoefplot.gllvm*)*, 57
randomCoefplot.gllvm, 57
ranef, 59
ranef.glmmVA, 59
residuals.gllvm, *49*, 60, *61, 62*

residuals.predictSR.gllvm, *57*, 61

se *(*se.gllvm*)*, 62
se.gllvm, 62, *68*
simulate *(*simulate.gllvm*)*, 63
simulate.gllvm, 63
Skabbholmen, 64
summary.gllvm, *9*, *32*, 65

tolerances *(*ecoCoefs*)*, 9

update *(*update.gllvm*)*, 67
update.gllvm, 67

varPartitioning *(*VP.gllvm*)*, 68
vcov *(*vcov.gllvm*)*, 68
vcov.gllvm, 68
VP *(*VP.gllvm*)*, 68
VP.gllvm, 68