

# Package ‘ecoCopula’

March 2, 2022

**Title** Graphical Modelling and Ordination using Copulas

**Version** 1.0.2

**Description** Creates 'graphs' of species associations (interactions) and ordination biplots from co-occurrence data by fitting discrete gaussian copula graphical models. Methods described in Popovic, GC., Hui, FK.C., Warton, DI., (2018) <doi:10.1016/j.jmva.2017.12.002>.

**Depends** R (>= 3.5.0), mvabund (>= 4.2)

**Imports** glasso, plyr, sna, MASS, tweedie, igraph, betareg, doParallel, mgcv, glm2, ordinal, compiler, foreach, stats

**License** LGPL (>= 2.1)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.2

**Suggests** knitr, rmarkdown, testthat, tidyr, ggplot2, RColorBrewer, ggraph, labdsv, tidygraph

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Gordana Popovic [aut, cre],  
David Warton [ctb],  
Francis K.C. Hui [ctb],  
Michelle Lim [ctb]

**Maintainer** Gordana Popovic <g.popovic@unsw.edu.au>

**Repository** CRAN

**Date/Publication** 2022-03-02 00:20:02 UTC

## R topics documented:

cgr . . . . .	2
cord . . . . .	3
fitted.stackedsdm . . . . .	5
plot.cgr . . . . .	6
plot.cord . . . . .	7

plot.stackedsdm . . . . .	9
predict.stackedsdm . . . . .	9
print.cgr . . . . .	11
print.cord . . . . .	11
residuals.stackedsdm . . . . .	12
simulate.cord . . . . .	13
spider . . . . .	14
stackedsdm . . . . .	16
summary.cgr . . . . .	17
summary.cord . . . . .	18

<b>Index</b>	<b>19</b>
--------------	-----------

---

cgr

*Fitting Gaussian copula graphical lasso to co-occurrence data*


---

## Description

cgr is used to fit a Gaussian copula graphical model to multivariate discrete data, like species co-occurrence data in ecology. This function fits the model and estimates the shrinkage parameter using BIC. Use [plot.cgr](#) to plot the resulting graph.

## Usage

```
cgr(
  obj,
  lambda = NULL,
  n.lambda = 100,
  n.samp = 500,
  method = "BIC",
  seed = NULL
)
```

## Arguments

obj	object of either class <a href="#">manyglm</a> , or <a href="#">manyany</a> with ordinal models <a href="#">clm</a>
lambda	vector, values of shrinkage parameter lambda for model selection (optional, see detail)
n.lambda	integer, number of lambda values for model selection (default = 100), ignored if lambda supplied
n.samp	integer (default = 500), number of sets residuals used for importance sampling (optional, see detail)
method	method for selecting shrinkage parameter lambda, either "BIC" (default) or "AIC"
seed	integer (default = 1), seed for random number generation (optional, see detail)

**Value**

Three objects are returned; `best_graph` is a list with parameters for the 'best' graphical model, chosen by the chosen method; `all_graphs` is a list with likelihood, BIC and AIC for all models along lambda path; `obj` is the input object.

**Details**

`cgr` is used to fit a Gaussian copula graphical model to multivariate discrete data, such as co-occurrence (multi species) data in ecology. The model is estimated using importance sampling with `n.samp` sets of randomised quantile or "Dunn-Smyth" residuals (Dunn & Smyth 1996), and the [glasso](#) package for fitting Gaussian graphical models. Models are fit for a path of values of the shrinkage parameter `lambda` chosen so that both completely dense and sparse models are fit. The `lambda` value for the `best_graph` is chosen by BIC (default) or AIC. The seed is controlled so that models with the same data and different predictors can be compared.

**Author(s)**

Gordana Popovic <g.popovic@unsw.edu.au>.

**References**

- Dunn, P.K., & Smyth, G.K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics* 5, 236-244.
- Popovic, G. C., Hui, F. K., & Warton, D. I. (2018). A general algorithm for covariance modeling of discrete data. *Journal of Multivariate Analysis*, 165, 86-100.

**See also**

[plot.cgr](#)

**Examples**

```
abund <- spider$abund[,1:5]
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_graph=cgr(spider_mod)
plot(spid_graph,pad=1)
```

---

cord

*Model based ordination with Gaussian copulas*

---

**Description**

Model based ordination with Gaussian copulas

**Usage**

```
cord(obj, nlv = 2, n.samp = 500, seed = NULL)
```

## Arguments

obj	object of either class <code>manyglm</code> , or <code>manyany</code> with ordinal models <code>clm</code>
nlv	number of latent variables (default = 2, for plotting on a scatterplot)
n.samp	integer (default = 500), number of sets residuals used for importance sampling (optional, see detail)
seed	integer (default = NULL), seed for random number generation (optional)

## Value

loadings latent factor loadings scores latent factor scores sigma covariance matrix estimated with nlv latent variables theta precision matrix estimated with nlv latent variables BIC BIC of estimated model logL log-likelihood of estimated model

## Details

`cord` is used to fit a Gaussian copula factor analytic model to multivariate discrete data, such as co-occurrence (multi species) data in ecology. The model is estimated using importance sampling with `n.samp` sets of randomised quantile or "Dunn-Smyth" residuals (Dunn & Smyth 1996), and the `factanal` function. The seed is controlled so that models with the same data and different predictors can be compared.

## Author(s)

Gordana Popovic <g.popovic@unsw.edu.au>.

## References

- Dunn, P.K., & Smyth, G.K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics* 5, 236-244.
- Popovic, G. C., Hui, F. K., & Warton, D. I. (2018). A general algorithm for covariance modeling of discrete data. *Journal of Multivariate Analysis*, 165, 86-100.

## See also

[plot.cord](#)

## Examples

```
abund <- spider$abund
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_lv=cord(spider_mod)
plot(spid_lv,biplot = TRUE)
```

---

fitted.stackedsdm      *Fitted values from a stackedsdm object*

---

### Description

Fitted values from a stackedsdm object

### Usage

```
## S3 method for class 'stackedsdm'
fitted(object, ...)
```

### Arguments

object	An object of class stackedsdm
...	Not used

### Value

A matrix of fitted values.

### Details

Extracts the fitted values from stackedsdm object.

### Author(s)

Francis K.C. Hui <francis.hui@anu.edu.au>.

### Examples

```
library(mvabund)
data(spider)
X <- spider$x
abund <- spider$abund

# Example 1: Simple example
myfamily <- "negative.binomial"
# Example 1: Funkier example where Species are assumed to have different distributions
# Fit models including all covariates are linear terms, but exclude for bare sand
fit0 <- stackedsdm(abund, formula_X = ~. -bare.sand, data = X, family = myfamily, ncores=2)
fitted(fit0)

# Example 2: Funkier example where Species are assumed to have different distributions
abund[,1:3] <- (abund[,1:3]>0)*1 # First three columns for presence absence
myfamily <- c(rep(c("binomial"), 3),
              rep(c("negative.binomial"), (ncol(abund)-3)))
fit0 <- stackedsdm(abund, formula_X = ~ bare.sand, data = X, family = myfamily, ncores=2)
fitted(fit0)
```

---

plot.cgr

*Plot graph of direct species associations.*


---

## Description

Plot graph of direct species associations.

## Usage

```
## S3 method for class 'cgr'
plot(
  x,
  P = NULL,
  vary.edge.lwd = FALSE,
  edge.col = c("light blue", "pink"),
  label = colnames(x$obj$fitted),
  vertex.col = "blue",
  label.cex = 0.8,
  edge.lwd = ifelse(vary.edge.lwd, 10, 4),
  edge.lty = c(1, 1),
  ...
)
```

## Arguments

x	is a cgr object, e.g. from output of <a href="#">cgr</a> .
P	locations of graph nodes, if NULL (default) these are generated with a Fruchterman Reingold algorithm.
vary.edge.lwd	is logical, TRUE will vary line width according to the strength of partial correlation, default (FALSE) uses fixed line width.
edge.col	takes two colours as arguments - the first is the colour used for positive partial correlations, the second is the colour of negative partial correlations.
label	is a vector of labels to apply to each variable, defaulting to the column names supplied in the data.
vertex.col	the colour of graph nodes.
label.cex	is the size of labels.
edge.lwd	is line width, defaulting to 10*partial correlation when varying edge width, and 4 otherwise.
edge.lty	is a vector of two integers specifying the line types for positive and negative partial correlations, respectively. Both default to solid lines.
...	other parameters to be passed through to plotting <a href="#">gplot</a> , in particular pad, the amount to pad the plotting range is useful if labels are being clipped. For details see the <a href="#">gplot</a> help file.

**Value**

a plot of species associations after accounting for the effect of all other species, positive/negative are blue/pink. The matrix of node positions (P) is returned silently.

**See Also**

[gplot](#), [cgr](#)

**Examples**

```
abund <- spider$abund[,1:5]
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_graph=cgr(spider_mod)
plot(spid_graph, edge.col=c("forestgreen","darkorchid4"),
      vertex.col = "black",vary.edge.lwd=TRUE)

library(tidyr)
library(tidygraph)
library(ggraph)

igraph_out<-spid_graph$best_graph$igraph_out

igraph_out %>% ggraph('fr') + # see ?layout_tbl_graph_igraph
  geom_edge_fan0(aes( colour = partcor, width=partcor)) +
  scale_edge_width(range = c(0.5, 3))+
  scale_edge_color_gradient2(low="#b2182b",mid="white",high="#2166ac")+
  geom_node_text(aes(label=name), repel = TRUE)+
  geom_node_point(aes(size=1.3))+
  theme_void() +
  theme(legend.position = 'none')
```

---

plot.cord

*Plots an ordination of latent variables and their corresponding coefficients (biplot).*

---

**Description**

Plots an ordination of latent variables and their corresponding coefficients (biplot).

**Usage**

```
## S3 method for class 'cord'
plot(
  x,
  biplot = FALSE,
  site.col = "black",
  sp.col = "blue",
```

```

alpha = 0.7,
arrow = TRUE,
site.text = FALSE,
labels = dimnames(x$obj$fitted),
...
)

```

### Arguments

<code>x</code>	is a cord object, e.g. from output of cord
<code>biplot</code>	TRUE if both latent variables and their coefficients are plotted, FALSE if only latent variables
<code>site.col</code>	site number colour (default is black), vector of length equal to the number of sites
<code>sp.col</code>	species name colour (default is blue), vector of length equal to the number of sites (if arrow=TRUE)
<code>alpha</code>	scaling factor for ratio of scores to loadings (default is 0.7)
<code>arrow</code>	should arrows be plotted for species loadings (default is TRUE)
<code>site.text</code>	should sites be labeled by row names of data (default is FALSE, points are drawn)
<code>labels</code>	the labels for sites and species (for biplots only) (default is data labels)
<code>...</code>	other parameters to be passed through to plotting functions.

### Value

an ordination plot.

### Examples

```

X <- spider$x
abund <- spider$abund
spider_mod <- stackedsdm(abund,~1, data = X, ncores=2)
spid_lv=cord(spider_mod)
#colour sites according to second column of x (bare sand)
cols=ifelse(spider$x[,2]>0,"black","red")
plot(spid_lv,biplot = FALSE,site.col=cols, site.text = TRUE)

```

```

library(ggplot2)
library(RColorBrewer)
alpha= 2.5
site_res <- data.frame(spid_lv$scores,X)
sp_res <- data.frame(spid_lv$loadings,species=colnames(abund))
ggplot()+
  geom_point(aes(x=Factor1,y=Factor2,color = reflection ),site_res)+
  geom_text(aes(x = Factor1*alpha, y = Factor2*alpha,label = species),data=sp_res)+
  scale_color_gradientn(colours = brewer.pal(n = 10, name = "PuOr"))+
  theme_classic()

```



---

plot.stackedsdm      *Plot residuals of stackedsdm.*

---

### Description

Plot residuals of stackedsdm.

### Usage

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

### Arguments

x                    is a stackedsdm object.  
...                  not used

### Examples

```
abund <- spider$abund  
spider_mod <- stackedsdm(abund,~1, data = spider$x)  
plot(spider_mod)
```

---

predict.stackedsdm      *Predictions from a stackedsdm object*

---

### Description

Predictions from a stackedsdm object

### Usage

```
## S3 method for class 'stackedsdm'  
predict(  
  object,  
  newdata = NULL,  
  type = "link",  
  se.fit = FALSE,  
  na.action = na.pass,  
  ...  
)
```

**Arguments**

<code>object</code>	An object of class <code>stackedsdm</code>
<code>newdata</code>	Optionally, a data frame in which to look for variables with which to predict. If omitted, the covariates from the existing dataset are used.
<code>type</code>	The type of prediction required. This can be supplied as either a single character string, when is applied to all species, or a vector of character strings of the same length as <code>ncol(object\$y)</code> specifying the type of predictions desired for each species. The exact type of prediction allowed depends precisely on the distribution, but for many there is at least "link" which is on the scale of the linear predictors, and "response" which is on the scale of the response variable. The values of this argument can be abbreviated.
<code>se.fit</code>	Logical switch indicating if standard errors are required.
<code>na.action</code>	Function determining what should be done with missing values in <code>newdata</code> . The default is to predict NA..
<code>...</code>	not used

**Value**

A list where the k-th element is the result of applying the `predict` method to the k-th fitted model in `object$fits`.

**Details**

This function simply applies a for loop, cycling through each fitted model from the `stackedsdm` object and then attempting to construct the relevant predictions by applying the relevant `predict` method. Please keep in mind no formatting is done to the predictions.

**Author(s)**

Francis K.C. Hui <francis.hui@anu.edu.au>.

**Examples**

```
X <- spider$x
abund <- spider$abund

# Example 1: Simple example
myfamily <- "negative.binomial"
# Fit models including all covariates are linear terms, but exclude for bare sand
fit0 <- stackedsdm(abund, formula_X = ~. -bare.sand, data = X, family = myfamily, ncores=2)
predict(fit0, type = "response")

# Example 2: Funkier example where Species are assumed to have different distributions
abund[,1:3] <- (abund[,1:3]>0)*1 # First three columns for presence absence
myfamily <- c(rep(c("binomial"), 3),
             rep(c("negative.binomial"), 5),
             rep(c("tweedie"), 4)
            )
```

```
fit0 <- stackedsdm(abund, formula_X = ~ bare.sand, data = X, family = myfamily, ncores=2)
predict(fit0, type = "response")
```

---

print.cgr

*Print function for cgr object*

---

### Description

Print function for cgr object

### Usage

```
## S3 method for class 'cgr'
print(x, ...)
```

### Arguments

x	is a cgr object, e.g. from output of <a href="#">cgr</a> .
...	not used

### See Also

[cgr](#)

### Examples

```
abund <- spider$abund[,1:5]
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_graph=cgr(spider_mod)
print(spid_graph)
```

---

print.cord

*Print function for cord object*

---

### Description

Print function for cord object

### Usage

```
## S3 method for class 'cord'
print(x, ...)
```

**Arguments**

x is a cord object, e.g. from output of [cord](#).  
 ... not used

**See Also**

[cord](#)

**Examples**

```
abund <- spider$abund
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_lv=cord(spider_mod)
print(spid_lv)
```

---

residuals.stackedsdm *Calculate residuals from a stackedsdm object*

---

**Description**

Calculate residuals from a stackedsdm object

**Usage**

```
## S3 method for class 'stackedsdm'
residuals(object, type = "dunnsmyth", seed = NULL, ...)
```

**Arguments**

object An object of class stackedsdm;  
 type Determined what type of residuals to calculate. The current options include Dunn-Smyth residuals (default; "dunnsmyth"), raw response residuals ("response") or probability integral transform residuals ("PIT");  
 seed For Dunn-Smyth and PIT residuals applied to discrete responses, random jittering is added, and the seed can be used to seed to jittering.  
 ... not used

**Value**

A matrix of residuals

**Details**

Calculated the residuals from stackedsdm object.

**Author(s)**

Francis K.C. Hui <francis.hui@anu.edu.au>.

**Examples**

```
X <- spider$x
abund <- spider$abund

# Example 1: Simple example
myfamily <- "negative.binomial"
# Example 1: Funkier example where Species are assumed to have different distributions
# Fit models including all covariates are linear terms, but exclude for bare sand
fit0 <- stackedsdm(abund, formula_X = ~. -bare.sand, data = X, family = myfamily, ncores=2)
residuals(fit0)

# Example 2: Funkier example where Species are assumed to have different distributions
abund[,1:3] <- (abund[,1:3]>0)*1 # First three columns for presence absence
myfamily <- c(rep(c("binomial"), 3),
              rep(c("negative.binomial"), (ncol(abund)-3)))
fit0 <- stackedsdm(abund, formula_X = ~ bare.sand, data = X, family = myfamily, ncores=2)
residuals(fit0)
```

---

simulate.cord

*Simulates new data from a given cord object*

---

**Description**

Simulates new data from a given cord object

**Usage**

```
## S3 method for class 'cord'
simulate(object, nsim = 1, seed = NULL, newdata = object$obj$data, ...)
```

**Arguments**

object	is a cord object, e.g. from output of cord
nsim	Number of simulations, defaults to 1. If nsim > 1, the simulated data will be appended.
seed	Random number seed, defaults to a random seed number.
newdata	A data frame in which to look for X covariates with which to simulate.
...	not used Defaults to the X covariates in the fitted model.

**Examples**

```

abund = spider$abund

spider_mod_ssdm = stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_lv_ssdm = cord(spider_mod_ssdm)
simulate(spid_lv_ssdm, nsim=2)

# using mvabund
library(mvabund) #for manyglm
abund=mvabund(abund)
spider_mod = manyglm(abund~1)
spid_lv = cord(spider_mod)
simulate(spid_lv)

spider_mod_X = manyglm(abund ~ soil.dry + bare.sand, data=spider$x)
spid_lv_X = cord(spider_mod_X)
Xnew = spider$x[1:10,]
simulate(spid_lv_X, newdata = Xnew)
simulate(spid_lv_X, nsim=2, newdata = Xnew)

spider_mod_X_ssdm = stackedsdm(abund, formula_X = ~. -bare.sand, data = spider$x, ncores=2)
spid_lv_X_ssdm = cord(spider_mod_X_ssdm)
simulate(spid_lv_X_ssdm, newdata = Xnew)

```

---

spider

*Spider data*


---

**Description**

Abundance of hunting spiders and associated environmental variables

**Usage**

```
spider
```

**Format**

A list containing the elements

**abund** A matrix with 28 observations of abundance of 12 hunting spider species.

**x** A data frame of six (transformed) environmental variables at each of the 28 sites.

**trait** A data frame of three species trait variables for each of the 12 species.

**Details**

The matrix `abund` has the following species abundances (column name abbreviation in brackets)

- *Alopecosa accentuata* (Alopacce)
- *Alopecosa cuneata* (Alopcune)
- *Alopecosa fabrilis* (Alopfabr)
- *Arctosa lutetiana* (Arctlute)
- *Arctosa perita*(Arctperi)
- *Aulonia albimana* (Auloalbi)
- *Pardosa lugubris* (Pardlugu)
- *Pardosa monticola* (Pardmont)
- *Pardosa nigriceps* (Pardnigr)
- *Pardosa pullata* (Pardpull)
- *Trochosa terricola* (Trocterr)
- *Zora spinimana* (Zoraspin)

The data frame `x` has the following  $\log(x+1)$ -transformed environmental variables

- `soil.dry` - Soil dry mass
- `bare.sand` - Cover bare sand
- `fallen.leaves` - Cover fallen leaves / twigs
- `moss` - Cover moss
- `herb.layer` - Cover herb layer
- `reflection` - Reflection of the soil surface with a cloudless sky

The data frame `trait` has the following variables

- `length` (numeric) - Length (log-transformed), averaged across typical lengths (in centimetres) for male and females
- (factor) - Predominant colour, "yellow" or "dark"
- (factor) - Whether the spider typically has markings on it: "none", "spots" or "stripes"

**Source**

Data attributed to van der Aart & Smeenk-Enserink (1975), obtained from the `spider2` directory, CANOCO FORTRAN package, with trait data added by David Warton, exported from `mvabund` R package.

---

stackedsdm

*Stacked species regression models, possibly fitted in parallel*


---

## Description

Stacked species regression models, possibly fitted in parallel

## Usage

```
stackedsdm(
  y,
  formula_X = ~1,
  data = NULL,
  family = "negative.binomial",
  trial_size = 1,
  do_parallel = FALSE,
  ncores = NULL,
  trace = FALSE
)
```

## Arguments

<code>y</code>	A matrix of species responses
<code>formula_X</code>	An object of class <code>formula</code> representing the relationship to the covariates to be fitted. There should be nothing to the left hand side of the "~" sign.
<code>data</code>	Data frame of the covariates
<code>family</code>	Either a single character vector, in which case all responses are assumed to be from this family, or a vector of character strings of the same length as the number of columns of <code>y</code> . Families as strings and not actual family class objects. This could be changed though if desired in the future e.g., for custom link functions. Currently, the following families are supported (hopefully properly!): "gaussian", "negative.binomial" (with quadratic mean-variance relationship), "poisson", "binomial" (with logit link), "tweedie", "Gamma" (with log link), "exponential", "beta" (with logit link), "ordinal" (cumulative logit model), "ztpoisson", "ztnegative.binomial", "zipoisson", "zinegative.binomial".
<code>trial_size</code>	The trial size if any of the responses are binomial. Is either a single number or a matrix with the same dimension as <code>y</code> . If the latter, then all columns that do not correspond to binomial responses are ignored.
<code>do_parallel</code>	Do the separate species model fits in parallel? Defaults to TRUE
<code>ncores</code>	The number of cores to use if separate the species model fits are done in parallel. If <code>do_parallel = TRUE</code> , then it defaults to <code>detectCores() - 2</code>
<code>trace</code>	Print information. This is not actually used currently



**Value**

A object of class `stackedsdm` with the following components: `call` The function call; `fits` A list where the  $j$ -th element corresponds to the fitted model for species  $j$  i.e., the  $j$ -th column in `y`; `linear_predictor` A matrix of the fitted linear predictors; `fitted` A matrix of the fitted values

**Details**

`stackedsdm` behaves somewhat like the `manyglm` or `manyglm` function in the package `mvabund`, in the sense that it fits a separate regression to each species response i.e., column of `y`. The main difference is that different families can be permitted for each species, which thus allows for mixed responses types.

**Author(s)**

Francis K.C. Hui <francis.hui@anu.edu.au>.

**Examples**

```
data(spider)
X <- spider$x
abund <- spider$abund

# Example 1: Simple example
myfamily <- "negative.binomial"
# Example 1: Funkier example where Species are assumed to have different distributions
# Fit models including all covariates are linear terms, but exclude for bare sand
fit0 <- stackedsdm(abund, formula_X = ~. -bare.sand, data = X, family = myfamily, ncores = 2)

# Example 2: Funkier example where Species are assumed to have different distributions
abund[,1:3] <- (abund[,1:3]>0)*1 # First three columns for presence absence
myfamily <- c(rep(c("binomial"), 3),
              rep(c("negative.binomial"), (ncol(abund)-3)))
fit0 <- stackedsdm(abund, formula_X = ~ bare.sand, data = X, family = myfamily, ncores = 2)
```

---

summary.cgr

*Summary function for cgr object*


---

**Description**

Summary function for cgr object

**Usage**

```
## S3 method for class 'cgr'
summary(object, ...)
```

**Arguments**

object            is a cgr object, e.g. from output of [cgr](#).  
...                not used

**See Also**

[cgr](#)

**Examples**

```
abund <- spider$abund[,1:5]
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_graph=cgr(spider_mod)
summary(spid_graph)
```

---

summary.cord

*Summary function for cgr object*

---

**Description**

Summary function for cgr object

**Usage**

```
## S3 method for class 'cord'
summary(object, ...)
```

**Arguments**

object            is a cord object, e.g. from output of [cgr](#).  
...                not used

**See Also**

[cord](#)

**Examples**

```
abund <- spider$abund[,1:5]
spider_mod <- stackedsdm(abund,~1, data = spider$x, ncores=2)
spid_lv=cord(spider_mod)
summary(spid_lv)
```

# Index

## \* datasets

spider, 14

cgr, 2, 6, 7, 11, 18

clm, 2, 4

cord, 3, 12, 18

factanal, 4

fitted.stackedsdm, 5

glasso, 3

gplot, 6, 7

manyan, 2, 4

manyglm, 2, 4

mvabund, 17

plot.cgr, 2, 3, 6

plot.cord, 4, 7

plot.stackedsdm, 9

predict.stackedsdm, 9

print.cgr, 11

print.cord, 11

residuals.stackedsdm, 12

simulate.cord, 13

spider, 14

stackedsdm, 16

summary.cgr, 17

summary.cord, 18