The cocorresp Package

December 1, 2005

Type Package

Title Co-correspondence analysis ordination methods for community ecology

Version 0.1-3

Date 2005-11-30

Depends vegan


Maintainer Gavin L. Simpson <gavin.simpson@ucl.ac.uk>

Description Fits predictive and symmetric co-correspondence analysis (CoCA) models to relate one data matrix to another data matrix. More specifically, CoCA maximises the weighted covariance between the weighted averaged species scores of one community and the weighted averaged species scores of another community. CoCA attempts to find patterns that are common to both communities.

License GPL Version 2

Encoding UTF-8

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>beetles</td>
<td>2</td>
</tr>
<tr>
<td>bryophyte</td>
<td>2</td>
</tr>
<tr>
<td>coca</td>
<td>3</td>
</tr>
<tr>
<td>cocorresp-package</td>
<td>6</td>
</tr>
<tr>
<td>coinertia</td>
<td>7</td>
</tr>
<tr>
<td>coinertial</td>
<td>8</td>
</tr>
<tr>
<td>corAxis</td>
<td>10</td>
</tr>
<tr>
<td>crossval</td>
<td>11</td>
</tr>
<tr>
<td>fitted.symcoca</td>
<td>12</td>
</tr>
<tr>
<td>mcChi</td>
<td>14</td>
</tr>
<tr>
<td>pasteCall</td>
<td>15</td>
</tr>
<tr>
<td>permutest.coca</td>
<td>16</td>
</tr>
<tr>
<td>plot.predcoca</td>
<td>18</td>
</tr>
<tr>
<td>plot.symcoca</td>
<td>19</td>
</tr>
<tr>
<td>rescale</td>
<td>21</td>
</tr>
</tbody>
</table>
bryophyte

Bryophytes and vascular plants in Carpathian spring meadows

Description

The data consist of observations on 30 bryophyte and 123 vascular plant species in 70 spring meadows (sites, samples). The species data are a subset of only those species occurring in five or more meadows.

Index

beetles

Carabid beetles and vascular plants in Dutch roadside verges

Description

Counts of 126 beetle taxa and abundances of 173 vascular plant taxa (expressed on the 1-9 van der Maarel scale) in 30 road-side verges in the Netherlands.

Usage

data(beetles)

data(plants)

Details

This is the complete dataset of Raemakers et al (2001). ter Braak and Schaffers (2004) only analyse a subset of 91 beetle taxa.

Source


Examples

data(beetles)

data(plants)
Usage

data(bryophyte)
data(vascular)

Source


Examples

data(bryophyte)
data(vascular)

coca

*Fit Co- Correspondence Analysis Ordination Models*

Description

coca is used to fit Co- Correspondence Analysis (CoCA) models. It can fit predictive or symmetric models to two community data matrices containing species abundance data. predcoca.simpls, predcoca.eigen and symcoca perform the actual model fitting.

Usage

coca(y, ...)

## Default S3 method:
coca(y, x, method = c("predictive", "symmetric"),
    reg.method = c("simpls", "eigen"), weights = NULL,
    n.axes = NULL, symmetric = FALSE, ...)

## S3 method for class 'formula':
coca(formula, data, method = c("predictive", "symmetric"),
    reg.method = c("simpls", "eigen"), weights = NULL,
    n.axes = NULL, symmetric = FALSE, ...)

predcoca.eigen(y, x, R0 = NULL, n.axes = NULL, nam.dat = NULL)
predcoca.simpls(y, x, R0 = NULL, n.axes = NULL, nam.dat = NULL)
symcoca(y, x, n.axes = NULL, R0 = NULL,
    symmetric = FALSE, nam.dat = NULL)
coca

Arguments

y      a data frame containing the response community data matrix.
x      a data frame containing the predictor community data matrix.
formula  a symbolic description of the model to be fit. The details of model specification are given below.
data    an optional data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which coca is called.
method    a character string indicating which co-correspondence analysis method to use. One of "predictive"(default), or "symmetric", can be abbreviated.
reg.method One of "simpls" (default) or "eigen". If method is "predictive" then reg.method controls whether the co-correspondence analysis should be fitted using the SIMPLS algorithm or via an eigen analysis.
weights, R0 a vector of length nrow(y) of user supplied weights for R0. If weights = NULL (default) then the weights are determined from y (default) or x and y (symmetric = TRUE only).
n.axes  the number of CoCA axes to extract. If missing (default) the n.axes is min(ncol(y), ncol(x), nrow(y) - 1).
symmetric if method is "symmetric" then symmetric determines whether weights for R0 are symmetric and taken as the average of the row sums of x and y (symmetric = TRUE). If symmetric = FALSE (default) then the weights R0 are taken as the row sums of y unless user defined weights are provided via argument weights. Ignored if method is "predictive".
nam.dat  an optional list with elements namY and namX containing the names of y and x respectively. Used to label printed output. If missing the names of are deduced from y and x.
... additional arguments to be passed to lower level methods.

Details
coca is the main user-callable function. predcoca.simpls, predcoca.eigen and symcoca perform the actual model fitting but are not meant to be called by the user as coca pre-processes the input data before calling these functions.

A typical model has the form response ~ terms where response is the (numeric) response data frame and terms is a series of terms which specifies a linear predictor for response. A typical form for terms is ., which is shorthand for "all variables" in data. If . is used, data must also be provided. If specific species (variables) are required then terms should take the form spp1 + spp2 + spp3.

The default is to fit a predictive CoCA model using SIMPLS via a modified version of simpls.fit from package pls. Alternatively, reg.method = "eigen" fits the model using an older, slower eigen analysis version of the SIMPLS algorithm. reg.method = "eigen" is about 100% slower than reg.method = "simpls".

Value
coca returns a list with method and reg.method determining the actual components returned.
nam.dat    list with components namY and namX containing the names of the response and the predictor(s) respectively.
call the matched call.

method the CoCA method used, one of "predictive" or "symmetric".

scores the species and site scores of the fitted model.

loadings the site loadings of the fitted model for the response and the predictor. (Predictive CoCA via SIMPLS only.)

tagged the fitted values for the response. A list with 2 components Yhat (the fitted values) and Yhat1 (the transformed fitted values. (Predictive CoCA via SIMPLS only.)

varianceExp list with components Yblock and Xblock containing the variances in the response and the predictor respectively, explained by each fitted PLS axis. (Predictive CoCA via SIMPLS only.)

totalVar list with components Yblock and Xblock containing the total variance in the response and the predictor respectively. (Predictive CoCA via SIMPLS only.)

lambda the Eigenvalues of the analysis.

n.axes the number of fitted axes

Ychi a list containing the mean-centered chi-square matrices for the response (Ychi1) and the predictor (Ychi2). (Predictive CoCA only.)

R0 the (possibly user-supplied) row weights used in the analysis.

X X-Matrix (symmetric CoCA only).

residuals Residuals of a symmetric model (symmetric CoCA only).

inertia list with components total and residual containing the total and residual inertia for the response and the predictor (symmetric CoCA only).

rowsum a list with the row sums for the response (rsum1) and the predictor (rsum2) (symmetric CoCA only).

colsum a list with the column sums for the response (csum1) and the predictor (csum2) (symmetric CoCA only).

Author(s)


References


See Also

crossval for cross-validation and permutest.coca for permutation test to determine the number of PLS axes to retain in for predictive CoCA.

summary.precoca and summary.symcoca for summary methods.
Examples

```r
## symmetric CoCA
data(beetles)
beetles <- log(beetles + 1)
data(plants)
## fit the model
bp.sym <- coca(beetles ~ ., data = plants, method = "symmetric")
bp.sym
summary(bp.sym)
plot(bp.sym)

## predictive CoCA using SIMPLS and formula interface
bp.pred <- coca(beetles ~ ., data = plants)
## should retain only the useful PLS components for a parsimonious model
## Not run:
## Leave-one-out crossvalidation - this takes a while
crossval(beetles, plants)
## so 2 axes are sufficient
## permutation test to assess significant PLS components - takes a while
bp.perm <- permutest.coca(bp.pred, permutations = 99)
bp.perm
summary(bp.perm)
## End(Not run)
## agrees with the Leave-one-out cross-validation
## refit the model with only 2 PLS components
bp.pred <- coca(beetles ~ ., data = plants, n.axes = 2)
bp.pred
summary(bp.pred)
plot(bp.pred)

## predictive CoCA using Eigen-analysis
data(bryophyte)
data(vascular)
carp.pred <- coca(y = bryophyte, x = vascular, reg.method = "eigen")
carp.pred
## determine important PLS components - takes a while
## Not run:
crossval(bryophyte, vascular)
(carp.perm <- permutest.coca(carp.pred, permutations = 99))
## End(Not run)
## 2 components again, refit
carp.pred <- coca(y = bryophyte, x = vascular, reg.method = "eigen", n.axes = 2)
carp.pred
## plot
plot(carp.pred)
```
Description

Fits predictive and symmetric co-correspondence analysis (CoCA) models to relate one data matrix to another data matrix. More specifically, CoCA maximises the weighted covariance between the weighted averaged species scores of one community and the weighted averaged species scores of another community. CoCA attempts to find patterns that are common to both communities.

Details

Package: cocorresp
Type: Package
Version: 0.1-3
Date: 2005-11-30
Depends: vegan
License: GPL Version 2
Encoding: UTF-8

Author(s)

Function simpls based on simpls.fit (package pls) by Ron Wehrens and Bjorn-Helge Mevik.
Maintainer: Gavin L. Simpson <gavin.simpson@ucl.ac.uk>

coinertia

Co-inertia analysis

Description

Performs a co-inertia of the triplets \((Q_1, K_1, R_0)\) and \((Q_2, K_2, R_0)\).

Usage

```r
coinertia(X, Dp, Y, Dq, Dn, n.axes)
```

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

Arguments

- **X**  
  *Q_1*, matrix of expected abundances under row-column independence in the original \(Y\) species matrix when treated as a contingency table.

- **Dp**  
  *\(K_1\)*, species (column) weights for \(X\).

- **Y**  
  *\(Q_2\)*, matrix of expected abundances under row-column independence in the original \(X\) species matrix when treated as a contingency table.

- **Dq**  
  *\(K_2\)*, species (column) weights for \(Y\).

- **Dn**  
  site weights \(R_0\).
coinertia

n.axes number of axes to calculate the co-inertia analysis for.

object an object of class coinertia.

axes the number of axes to display when printing.

... arguments passed to other functions. Currently ignored.

Value

A list with the following components:

U1 column weights of X.

U2 column weights of Y.

X1 rowscores of X.

X2 rowscores of Y.

lambda the Eigenvalues (squares of the singular values).

n.axes number of axes requested.

call the matched function call.

Note

This function is not yet meant to be called directly by the user. If you wish to use it directly, see the function definition for symcoca which demonstrates how to prepare the relevant input matrices.

Note that in this function, X corresponds to the input matrix y and Y corresponds to the input matrix x in symcoca. Confusing! This will be changed in a future release but for now the arguments follow those of the original Matlab code - perhaps a little too closely!

Author(s)


References


See Also

symcoca for the function that calls coinertia and coinertiaI for co-inertia analysis using identity matrices for K1, K2, and R0.
**coinertiaI**  
*Coinertia analysis with identity matrices*

**Description**
Performs a co-inertia of the triplets \((Q_1, K_1, R_0)\) and \((Q_2, K_2, R_0)\) with identity matrices \(K_1, K_2, R_0\).

**Usage**

```r
coinertiaI(X, Y, fast = TRUE)
```

**Arguments**

- **X**: Species matrix X.
- **Y**: Species Matrix Y.
- **fast**: If "TRUE" only return the row scores of Y.

**Details**
Argument *fast* is used to return only the row scores of Y in function `permutest.coca`, which speeds the permutation test considerably.

**Value**
If *fast = TRUE*, a matrix of row scores for matrix Y (see scores below). If *fast = FALSE* a list with the following components:

- **weights**: A list with components X and Y containing the left and right singular vectors respectively of the SVD on the triplets.
- **scores**: A list with components
dQuote{X} and Y, containing the row scores of the X and Y species matrices. These are the result of a matrix multiplication of X by the left singular vectors and Y by the right singular vectors.
- **lambda**: the Eigenvalues of the analysis (the square of the singular values from the SVD).
- **call**: the matched function call.

**Note**
This function is not meant to be called directly by the user. If you wish to use it study the code in `permutest.coca` to see how it should be called.

**Author(s)**

**References**
See Also

coinertia

corAxis

Correlation between ordination axes

Description

Calculates the Pearson product-moment correlation coefficient for the site scores of ordination axes.

Usage

```
corAxis(x, ...) 

## Default S3 method: 
corAxis(x, ...) 

## S3 method for class 'symcoca': 
corAxis(x, axes = c(1:min(6, x$n.axes)), ...) 
```

Arguments

- `x`: an ordination object. Only methods for objects of class `symcoca` are currently available.
- `axes`: the number of axes to calculate the correlation coefficients for.
- `...`: arguments to be passed on to other methods.

Value

A named vector containing the correlation coefficients for the requested axes.

Note

The arguments for `cor` are hard coded at their defaults, see `cor` for details. A more flexible version is planned that will allow arguments to be passed to `cor`.

Author(s)

Gavin L. Simpson

See Also

- `cor`, for the main analysis function.
**Examples**

```r
## load some data
data(beetles)
data(plants)

## log transform the beetle data
beetles <- log(beetles + 1)

## symmetric Co-CA model
beetles.sym <- coca(beetles ~ ., data = plants, method = "symmetric")

## correlations between axes
corAxis(beetles.sym)
```

---

**crossval**

*Cross-validation for predictive Co-Correspondence Analysis models*

**Description**

Performs a leave-one-out cross-validation of a predictive Co-Correspondence Analysis model.

**Usage**

```r
crossval(y, x, n.axes = min(dim(x), dim(y)) - 1, 
centre = TRUE, verbose = TRUE)
```

```r
## S3 method for class 'crossval':
summary(object, axes = c(1:min(6, object$n.axes)), ...)
```

**Arguments**

- `y` the response species matrix.
- `x` the predictor species matrix.
- `n.axes` the number of axes to calculate the leave-one-out cross-validation for. Default is to perform the CV for all extractable axes.
- `centre` centre `y` and `x` during analysis? Currently ignored as it may not be necessary.
- `verbose` if TRUE, the default, print information on the progress of the cross-validation procedure.
- `object` an object of class `crossval` as returned by `crossval`.
- `axes` the number of axes to summarise results for.
- `...` further arguments to `print` - currently ignored.

**Details**

Performs a leave-one-out cross-validation of a predictive Co-Correspondence Analysis model. It can be slow depending on the number of columns in the matrices, and of course the number of sites.
crossval

Value

Returns a large list with the following components:

- `dimx, dimy` the dimensions of the input matrices `x` and `y` respectively.
- `press0` the `press_0` statistic.
- `n.axes` the number of axes tested.
- `CVfit` the cross-validatory fit.
- `varianceExp` list with components `Yblock` and `Xblock` containing the variances in the response and the predictor respectively, explained by each fitted PLS axis.
- `totalVar` list with components `Yblock` and `Xblock` containing the total variance in the response and the predictor respectively.
- `nam.dat` list with components `namY` and `namX` containing the names of the response and the predictor(s) respectively.
- `call` the R call used.

Note

This function is not a bit out-of-date compared to some of the other functions. It should have a formula interface like `coca` or work on the results from `coca`, although that will have to be altered to store a copy of the data?

Author(s)


See Also

The model fitting function `coca`

Examples

```r
## load the data sets
data(beetles)
data(plants)
## log transform the bettle data
beetles <- log(beetles + 1)

## predictive CoCA using SIMPLS and formula interface
bp.pred <- coca(beetles ~ ., data = plants)
## should retain only the useful PLS components for a parsimonious model

## Not run:
## Leave-one-out crossvalidation - this takes a while
crossval(beetles, plants)
## so 2 axes are sufficient
## End(Not run)

bp.pred <- coca(beetles ~ ., data = plants, n.axes = 2)
bp.pred
summary(bp.pred)
```
fitted.symcoca  

**Fitted values of a Symmetric Co-Correspondence analysis model.**

**Description**

Calculates and extracts the fitted values of a Symmetric Co-Correspondence analysis model.

**Usage**

```r
## S3 method for class 'symcoca':
fitted(object, ...)
```

**Arguments**

- `object`  
an object of class "symcoca"
- `...`  
arguments to be passed to other methods.

**Value**

A list with the following components:

- `Yhat1`  
the fitted values for the “response” matrix.
- `Yhat2`  
the fitted values for the “predictor” matrix.
- `nam.dat`  
a vector containing the names of the “response” and “predictor” matrices respectively. Used for printing the results.

**Note**

This function needs an update and to allow option to restrict fitted values to specified axes, and the names of the returned objects need making more obvious!

**Author(s)**


**References**


**See Also**

The model fitting function `coca`
## symmetric CoCA

data(beetles)
beetles <- log(beetles + 1)
data(plants)

## log transform the bettle data
bp.sym <- coca(beetles ~ ., data = plants, method = "symmetric")
bp.sym

## fitted values
bp.fit <- fitted(bp.sym)
bp.fit

---

**mcChi**

*Standardised chi-square residuals*

### Description

Scales a matrix, \( Y \), to its standardised chi-square residuals \( \frac{(o-e)}{\sqrt{e}} \) (if \( R_0 = R \), where \( R \) contains the row sums of matrix \( Y \)) so that further analysis can be unweighted.

### Usage

```r
mcChi(Y, R0, eps = 1e-06)
```

### Arguments

- **Y**: a matrix for which standardised chi-square residuals are to be calculated.
- **R0**: row weights.
- **eps**: tolerance - leave as default.

### Details

This function implements equation 8 of ter Braak and Schaffers (2004) by firstly applying equation 7 to form matrix \( Q \) using row and column sums of \( Y \) as weights, and, secondly, by applying equation 8 to form a matrix of standardised chi-square residuals from \( Q \) by pre-multiplication of \( Q \) by \( \sqrt{R_0} \) and post-multiplication of \( Q \) by \( \sqrt{K} \), where \( K \) is the column sums of \( Y \).

### Value

A list with the following components:

- **Ychi**: the matrix of standardised chi-squared residuals of \( Y \)
- **Kn**: the column sums (\( K \)) of \( Y \) divided by \( \text{sum}(K) \)

### Note

This function is not intended for casual use by users.

### Author(s)

References


Examples
data(beetles)
## log transform the bettle data
beetles <- as.matrix(log(beetles + 1))
## row sums for R0
rsum <- rowSums(beetles)
## calculate chi-square residuals
res <- mcChi(beetles, rsum)
res$Ychi

---

**pasteCall**

Pretty printing of R language calls

Description

A helper function to facilitate printing of R calls stored in R objects.

Usage

```
pasteCall(call, prefix = "Call:"
```

Arguments

- `call`: an object of class `call`.
- `prefix`: a prefix to prepend to the concatenated, textual representation of the call.

Details

The function deparses the matched call. If the length of this deparsed object is greater than 1, `paste` is used to paste together each section of the call, separating each section with a space. The concatenated or original call then has the `prefix` prepended to it.

Value

A character vector of length one containing the original call.

Note

This function is used in a number of the print methods in the cocorresp package, with `writeLines` and `strwrap` to achieve prettier printing of calls within the printed output.

Author(s)

Gavin L. Simpson.
Examples

```r
# From the example in ?call
c1 <- call("round", 10.5)
pcl <- pasteCall(c1)
writeLines(strwrap(pcl))

# A silly example of a long call, wrapped on to 2 lines
dat <- as.data.frame(matrix(rnorm(100), nc = 10))
(names(dat) <- paste("var", 1:10, sep = ""))
mod <- lm(var5 ~ var1 + var2 + var3 + var4 + var6 + var7 + var8 +
          var9 + var10, data = dat)
writeLines(strwrap(pasteCall(mod$call)))
```

---

`permutest.coca`  
*Permutation test for predictive co-correspondence analysis models*

Description

A permutation test for predictive co-correspondence analysis models to assess the significance of each CoCA ordination axes.

Usage

```r
permutest.coca(x, R0 = NULL, permutations = 99,
n.axes = x$n.axes, verbose = TRUE, ...)
```

```r
## S3 method for class 'permutest.coca':
summary(object, ...)
```

Arguments

- `x`  
an object of class "predcoca".
- `R0`  
row weights to use in the analysis. If missing, the default, these are determined from `x`.
- `permutations`  
the number of permutations to perform.
- `n.axes`  
The number of axes to test. Defaults to the number of axes stated in `x$n.axes`.
- `verbose`  
if TRUE, the default, print information on the progress of the permutation test procedure.
- `object`  
an object of class "permutest.coca".
- `...`  
arguments to be passed to other methods.

Details

An alternative approach to cross-validation (see `crossval`) to select the number of axes to retain in a predictive co-correspondence analysis is to test the statistical significance of each ordination axis using permutation tests.

The test statistic used is the $F$-ratio based on the fit of the first axis to the response data (ter Braak and Smilauer 2002). The second and subsequent axes are tested by treating previous axes as co-variables.
To be precise, this approach does not test the significance of SIMPLS axes, but those of NIPALS-PLS axes (ter Braak and de Jong 1998).

Value

A list with the following components:

- `pval`: a vector of \( P \)-values for each ordination axis.
- `permstat`: a vector of values for the test statistic for each axis.
- `total.inertia`: the total inertia in the response matrix.
- `inertia`: a vector containing the inertia explained by each ordination axis.
- `fitax`: a vector containing the fit of each axis to response.
- `pcent.fit`: a vector containing the fit of each axis to the response as a percentage of the total inertia (variance).
- `n.axes`: the number of axes in the ordination.
- `call`: the matched call.

Warning

This function is slow. Beware setting argument `permutations` higher than the default. Determine how long it takes for the default 99 permutations to complete before going crazy and asking for thousands of permutations - you’ve been warned, have a good book to hand.

Note

Argument \( R0 \) is provided for compatibility with the original MATLAB code. The R usage paradigm makes this argument redundant in the current code and it may be invalid to supply different row weights (\( R0 \)) as \( R0 \). This argument will likely be removed in future versions.

Author(s)


References


See Also

coca, for the model fitting function, crossval, for a leave-one-out cross-validation procedure, which is the preferred way to select axes in a predictive co-correspondence analysis.
Examples

```r
## load some data
data(beetles)
## log transform the bettle data
beetles <- log(beetles + 1)
data(plants)
## predictive CoCA using SIMPLS and formula interface
bp.pred <- coca(beetles ~ ., data = plants)

## should retain only the useful PLS components for a parsimonious model
## Not run:
## Leave-one-out crossvalidation - this takes a while
crossval(beetles, plants)
## so 2 axes are sufficient

## permutation test to assess significant PLS components - takes a while
bp.perm <- permutest.coca(bp.pred, permutations = 99)
bp.perm
summary(bp.perm)

## permutation test, this time only testing the first 2 axes
bp.perm <- permutest.coca(bp.pred, permutations = 99, n.axes = 2)
bp.perm
summary(bp.perm)
## End(Not run)
```

---

**plot.predcoca**

**Biplots for predictive co-correspondence analysis**

### Description

Produces biplots of the response and predictor from the results of a predictive co-correspondence analysis.

### Usage

```r
## S3 method for class 'predcoca':
plot(x, pages = NULL, axes = c(1:2),
cex = c(0.8, 0.8), pch = c(par("pch"), 3),
ylab = NULL, xlab = NULL, ann = par("ann"),
lab.plot = TRUE, ...)
```

### Arguments

- **x**: an object of class "predcoca", the result of a call to `predcoca.eigen` or `predcoca.simpls`.
- **pages**: the number of pages to print the biplots on. Currently ignored.
- **axes**: a vector of length 2 indicating which predictive CoCA axes to plot.
- **cex**: a vector of length 2, containing the character expansion factors to use for the samples (sites) and the species.
- **pch**: a vector of length 2, containing the plotting character to use for the samples (sites) and the species.
ylab  the label for the y-axis of the biplots, currently ignored.
xlab  the label for the x-axis of the biplots, currently ignored.
ann   logical, if TRUE plots are annotated and not if FALSE, currently ignored.
lab.plot logical, if TRUE, the biplots are given a title, derived from the names of the
        original data matrices. See coca.
...  graphical parameters can be given as arguments to plot.default, these are
currently ignored.

Note
This is a simple attempt at a plot method for predictive co-correspondence analysis biplots. A
number of the arguments do nothing and some of the functionality is hard-coded within the function
definition. This will be updated in a future release of the package.

Author(s)
Gavin L. Simpson.

References
to relate two community compositions. Ecology 85(3), 834–846

See Also
coca, plot.default

Examples
## predictive CoCA
data(beetles)
## log transform the bettle data
beetles <- log(beetles + 1)
data(plants)
## predictive CoCA using SIMPLS and formula interface
bp.pred <- coca(beetles ~ ., data = plants)
## draw the biplots
plot(bp.pred)
Usage

```r
## S3 method for class 'symcoca':
plot(x, pages = NULL, axes = c(1:2),
     scaling = 1, cex = c(0.8, 0.8), pch = c(par("pch"), 3),
     ylab = NULL, xlab = NULL, ann = par("ann"),
     lab.plot = TRUE, ...)```

Arguments

- `x`: an object of class "symcoca", the result of a call to `symcoca`.
- `pages`: the number of pages to print the biplots on. Currently ignored.
- `axes`: a vector of length 2 indicating which predictive CoCA axes to plot.
- `scaling`: the scaling to be applied. See `scores.symcoca`.
- `cex`: a vector of length 2, containing the character expansion factors to use for the samples (sites) and the species.
- `pch`: a vector of length 2, containing the plotting character to use for the samples (sites) and the species.
- `ylab`: the label for the y-axis of the biplots, currently ignored.
- `xlab`: the label for the x-axis of the biplots, currently ignored.
- `ann`: logical, if TRUE plots are annotated and not if FALSE, currently ignored.
- `lab.plot`: logical, if TRUE, the biplots are given a title, derived from the names of the original data matrices. See `coca`.
- `...`: graphical parameters can be given as arguments to `plot.default`, these are currently ignored.

Note

This is a simple attempt at a plot method for symmetric co-correspondence analysis biplots. A number of the arguments do nothing and some of the functionality is hard-coded within the function definition. This will be updated in a future release of the package.

Author(s)

Gavin L. Simpson.

References


See Also

coca, plot.default
Examples

```r
## symmetric CoCA
data(beetles)
## log transform the bettle data
beetles <- log(beetles + 1)
data(plants)
## fit the model
bp.sym <- coca(beetles ~ ., data = plants, method = "symmetric")
## draw the biplots
plot(bp.sym)
```

**rescale**

Rescales CoCA species scores to the quarter root of the eigenvalues.

**Description**

Rescales CoCA species scores to the quarter root of the eigenvalues.

**Usage**

```r
rescale(object, ...)
```

```r
## Default S3 method:
rescale(object, ...)
```

```r
## S3 method for class 'symcoca':
rescale(object, axes = c(1:object$n.axes), ...)
```

**Arguments**

- `object`: an R object. Currently only objects of class "symcoca" are supported.
- `axes`: the number of axes to rescale
- `...`: other arguments to be passed to rescale methods. Currently not used here.

**Details**

Currently only implemented for objects of class "symcoca".

**Value**

Returns a list with the following components:

- `U1`: rescaled species scores for the response
- `U2`: rescaled species scores for the predictor

**Author(s)**


**See Also**

`symcoca`
Examples

data(bryophyte)
data(vascular)
bryo.sym <- coca(bryophyte ~ ., data = vascular,
  method = "symmetric")
rescale(bryo.sym, axes = 1:2)

coca (Continued)

data(bryophyte)
data(vascular)
bryo.sym <- coca(bryophyte ~ ., data = vascular,
  method = "symmetric")
rescale(bryo.sym, axes = 1:2)

resid.symcoca | Extract Model Residuals

Description

Extracts the residuals of the fitted model of a symmetric CoCA to the response and the predictor.

Usage

## S3 method for class 'symcoca':
resid(object, ...)

Arguments

object | an object of class "symcoca".
...
| arguments to be passed to other methods.

Value

A list containing the residuals for the response and the predictor with the following components:

Y | residuals of the fit to the response.
X | residuals of the fit to the predictor.

Author(s)

Gavin L. Simpson

See Also

symcoca

Examples

data(bryophyte)
data(vascular)
bryo.sym <- coca(bryophyte ~ ., data = vascular,
  method = "symmetric")
resid(bryo.sym)
**scaleChi**

*Standardised chi-square residuals*

**Description**

Scales a matrix, \( Y \), to is standardised chi-square residuals \((o - e) / \sqrt{e}\) (given \( K_n \) and \( R_0 \) metrics derived from an external matrix \( Y_0 \)) so that further analysis can be unweighted.

**Usage**

```r
scaleChi(Y, Kn, R0, eps = 1e-06)
```

**Arguments**

- \( Y \): a matrix for which standardised chi-square residuals are to be calculated.
- \( Kn \): the column sums (K) of \( Y \) divided by sum(K).
- \( R0 \): row weights.
- \( eps \): a tolerance.

**Value**

- \( Yr \): the matrix of standardised chi-squared residuals of \( Y \).

**Note**

This function is not intended for casual use by users.

**Author(s)**


**References**


---

**scores.predcoca**

*Get Species or Site Scores from an Ordination*

**Description**

Function to access either species or site scores for specified axes in co-correspondence analysis ordination methods.
scores.predcoca

Usage

## S3 method for class 'predcoca':
scores(x, choices = c(1, 2),
       display = c("site","species"), ...)

## S3 method for class 'symcoca':
scores(x, choices = c(1, 2),
       display = c("site","species"), scaling = 1, ...)

Arguments

- **x**
  - an ordination result
- **display**
  - partial match to access scores for “sites” or “species”.
- **choices**
  - the ordination axes to return.
- **scaling**
  - whether the species scores should be rescaled to the quarter root of the eigenvalues using `rescale.symcoca`.
- **...**
  - arguments to be passed to other methods.

Details

Implements a `scores` method for symmetric co-correspondence analysis ordination results.

Value

A list with two components containing matrices of the requested scores:

- **species**
  - A list with two components, `U1` and `U2`, containing the species scores for the response matrix `Y` and the predictor matrix `X` respectively.
- **sites**
  - A list with two components, `X1` and `X2`, containing the site scores for the response matrix `Y` and the predictor matrix `X` respectively.
- **...**

Author(s)


References


See Also

`scores`, for further details on the method.
### Examples

```r
## load some data
data(beetles)
data(plants)
## log transform the bettle data
beetles <- log(beetles + 1)
## fit the model, a symmetric CoCA
bp.sym <- coca(beetles ~ ., data = plants, method = "symmetric")
## extract the scores
scores(bp.sym)

## predictive CoCA using SIMPLS and formula interface
bp.pred <- coca(beetles ~ ., data = plants)
scores(bp.pred)
```

---

### simpls

#### Modified version of Sijmen de Jong’s SIMPLS

#### Description

Fits a PLSR model with the SIMPLS algorithm, modified to allow a weighted analysis.

#### Usage

```r
simpls(X, Y, ncomp, stripped = FALSE, ...)
```

#### Arguments

- `X`: a matrix of observations. NAs and Ins are not allowed.
- `Y`: a vector or matrix of responses. NAs and Ins are not allowed.
- `ncomp`: the number of components to be used in the modelling.
- `stripped`: logical. If TRUE the calculations are stripped as much as possible for speed; this is meant for use with cross-validation or simulations when only the coefficients are needed. Defaults to FALSE.
- `...`: other arguments. Currently ignored.

#### Details

This function is a modified version of `simpls.fit` from package pls. Four modification have been made:

1. The input matrices X and Y are not centered,
2. The scores (tt in the code) are not centered,
3. Added code to calculate the total variance in the Y matrix, Ytotvar, and the variance in Y accounted for by each PLS axis, Yvar (See Value below), and
4. Additional components are returned if argument `stripped` is TRUE.

This function should not be called directly, but through the generic function `coca`.

SIMPLS is much faster than the NIPALS algorithm, especially when the number of X variables increases, but gives slightly different results in the case of multivariate Y. SIMPLS truly maximises the covariance criterion. According to de Jong, the standard PLS2 algorithms lie closer to ordinary least-squares regression where a precise fit is sought; SIMPLS lies closer to PCR with stable predictions.
Value

A list containing the following components is returned:

coefficients an array of regression coefficients for 1, ..., ncomp components. The dimensions of coefficients are \(c(nvar, npred, ncomp)\) with \(nvar\) the number of \(X\) variables and \(npred\) the number of variables to be predicted in \(Y\).

scores a matrix of scores.

loadings a matrix of loadings.

Yscores a matrix of \(Y\)-scores.

Yloadings a matrix of \(Y\)-loadings.

projection the projection matrix used to convert \(X\) to scores.

Xmeans a vector of means of the \(X\) variables.

Ymeans a vector of means of the \(Y\) variables.

fitted.values an array of fitted values. The dimensions of fitted.values are \(c(nobj, npred, ncomp)\) with \(nobj\) the number samples and \(npred\) the number of \(Y\) variables.

residuals an array of regression residuals. It has the same dimensions as fitted.values.

Xvar a vector with the amount of \(X\)-variance explained by each number of components.

Yvar a vector with the amount of \(Y\)-variance explained by each number of components.

Xtotvar Total variance in \(X\).

Ytotvar Total variance in \(Y\).

If stripped is TRUE, only the components coefficients, Xmeans and Ymeans, Xvar and Yvar, and Xtotvar and Ytotvar are returned.

Author(s)

Based on simpls.fit by Ron Wehrens and Bjorn-Helge Mevik, with simple modifications by Gavin L. Simpson.

References


See Also
coca
Summarizing Co-CA Model Fits

Description

summary methods for classes "predcoca" and "symcoca". These provide a summary of the main results of a Co-Correspondence Analysis model.

Usage

## S3 method for class 'predcoca':
summary(object, axes = c(1:min(6, object$n.axes)),
        display = c("species", "site"), ...)

## S3 method for class 'symcoca':
summary(object, axes = c(1:min(6, object$n.axes)),
        display = c("species", "site"), scaling = 1, ...)

Arguments

object an object of class "predcoca" or "symcoca". Generally the result of a call to coca.
axes the number of CoCA axes to return in the result set.
display one or both of "species" and/or "site"
scaling for objects of class "symcoca" only, the scaling to be applied to the results. One of "1" or "2". See below for details of scalings used.
...
arguments to be passed to other methods.

Value

A list with the some of the following components:
cocaScores The site and or species scores for the axes requested.
call The call used to fit the model.
lambda The eigenvalues for the axes requested. Not for predcoca.simpls.
namY, namX the names of the response and predictor either supplied by the user or derived from the original call.
loadings a list with two components loadings1 and loadings2, which refer to the response and the predictor matrices respectively. (Only for predictive CoCA models.)
varianceExp a list with components Yblock and Xblock containing the amount of variance explained on each CoCA axis in the response and the predictor respectively. (Only for predictive CoCA models.)
totalVar a list with components Yblock and Xblock containing the total variance in the response and the predictor data sets respectively
inertia a list with components total and residual containing the total and residual inertia (variance) in the response and the predictor matrices of a symmetric CoCA model. (Only for symmetric CoCA models.)
scaling the scaling used/requested. (Only for symmetric CoCA models.)
Author(s)

Gavin L. Simpson

See Also

The model fitting function `coca`

Examples

```r
## continue the example from coca(.)
## summary for symmetric CoCA
bp.summ <- summary(bp.sym, axes = 1:4)
bp.summ

## Different scaling
bp.summ <- summary(bp.sym, axes = 1:4, scaling = 2)
bp.summ

## summary for predictive CoCA
bp.summ <- summary(bp.pred, axes = 1:2)
bp.summ
```
Index

*Topic datasets
  beetles, 1
  bryophyte, 2

*Topic hplot
  plot.predcoca, 17
  plot.symcoca, 19

*Topic models
  coca, 3
  coinertia, 7
  coinertiaI, 8

*Topic multivariate
  coca, 3
  coinertia, 7
  coinertiaI, 8
  crossval, 10
  fitted.symcoca, 12
  mcChi, 13
  permutest.coca, 15
  rescale, 20
  resid.symcoca, 21
  scaleChi, 22
  scores.predcoca, 22
  simpls, 24
  summary.predcoca, 26

*Topic package
  cocorresp-package, 6

*Topic regression
  coca, 3
  simpls, 24

*Topic univar
  corAxis, 9

*Topic utilities
  pasteCall, 14

beetles, 1
bryophyte, 2
call, 14
coca, 3, 11, 13, 17–20, 24–27
cocorresp (cocorresp-package), 6
cocorresp-package, 6
coinertia, 7, 9
coinertiaI, 8, 8
cor, 10
corAxis, 9
crossval, 5, 10, 16, 17
fitted.symcoca, 12
mcChi, 13
ordiParseFormula, 5
paste, 14
pasteCall, 14
permutest.coca, 5, 8, 9, 15
plants (beetles), 1
plot.default, 18–20
plot.predcoca, 17
plot.symcoca, 19
predcoca.eigen, 18
predcoca.eigen (coca), 3
predcoca.simpls, 18, 26
predcoca.simpls (coca), 3
print.coinertia (coinertia), 7
print.crossval (crossval), 10
print.fitted.symcoca (fitted.symcoca), 12
print.permutest.coca (permutest.coca), 15
print.predcoca (coca), 3
print.summary.coinertia (coinertia), 7
print.summary.crossval (crossval), 10
print.summary.permutest.coca (permutest.coca), 15
print.summary.predcoca (summary.predcoca), 26
print.summary.symcoca (summary.predcoca), 26
print.symcoca (coca), 3
rescale, 20
rescale.symcoca, 23
resid.symcoca, 21
scaleChi, 22
scores, 23
scores.predcoca, 22
scores.symcoca, 19
scores.symcoca(scores.predcoca), 22
simpls, 24
simpls.fit, 4, 24, 25
strwrap, 15
summary.coinertia(coinertia), 7
summary.crossval(crossval), 10
summary.permutest.coca
  (permutest.coca), 15
summary.predcoca, 5, 26
summary.symcoca, 5
summary.symcoca
  (summary.predcoca), 26
symcoca, 8, 10, 19, 21
symcoca(coca), 3

vascular(bryophyte), 2

writeLines, 15