

# Package ‘cocorresp’

May 7, 2009

**Type** Package

**Title** Co-correspondence analysis methods

**Version** 0.1-7

**Date** 2008-06-25

**Depends** stats, graphics, vegan

**Author** Original Matlab routines by C.J.F. ter Braak and A.P. Schaffers. R port by Gavin L. Simpson.  
Function `simpls` based on `simpls.fit` (package `pls`) by Ron Wehrens and Bjorn-Helge Mevik.

**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-2

**Repository** CRAN

**Repository/R-Forge/Project** cocorresp

**Repository/R-Forge/Revision** 11

**Date/Publication** 2009-05-06 21:45:42

## R topics documented:

<code>cocorresp-package</code> . . . . .	2
<code>beetles</code> . . . . .	3
<code>bryophyte</code> . . . . .	3
<code>coca</code> . . . . .	4
<code>coinertia</code> . . . . .	8
<code>coinertial</code> . . . . .	9

corAxis . . . . .	10
crossval . . . . .	11
fitted.symcoca . . . . .	13
mcChi . . . . .	14
permutest.coca . . . . .	16
plot.predcoca . . . . .	18
plot.symcoca . . . . .	20
rescale . . . . .	22
resid.symcoca . . . . .	23
scaleChi . . . . .	24
scores.predcoca . . . . .	25
simpls . . . . .	26
summary.predcoca . . . . .	28

<b>Index</b>	<b>30</b>
--------------	-----------

---

cocorresp-package *Co-correspondence analysis ordination methods for community ecology*

---

## 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  
 Packaged: Wed Nov 30 12:54:49 2005; gavin

## Author(s)

Original Matlab routines by C.J.F. ter Braak and A.P. Schaffers. R port by Gavin L. Simpson. Function `simpls` based on `simpls.fit` (package `pls`) by Ron Wehrens and Bjorn-Helge Mevik.

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

---

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

Raemakers, I.P., Schaffers, A.P., Sykora, V. and Heijerman, T. (2001) The importance of plant communities in road verges as habitat for insects. *Proceedings of the Section Experimental and Applied Entomology of the Netherlands Entomological Society* **12**, 101–106.

ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

### Examples

data (beetles)

data (plants)

---

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.

### Usage

data (bryophyte)

data (vascular)

## Source

Hajek, M., Hekera, P. and Hajkova, P. (2002) Spring fen vegetations and water chemistry in the Western Carpathian flysch zone. *Folia geobotanica* **37**, 205–224

ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85(3)**, 834–846

## 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)
```

## Arguments

<code>y</code>	a data frame containing the response community data matrix.
<code>x</code>	a data frame containing the predictor community data matrix.
<code>formula</code>	a symbolic description of the model to be fit. The details of model specification are given below.
<code>data</code>	an optional data frame containing the variables in the model. If not found in <code>data</code> , the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>coca</code> is called.
<code>method</code>	a character string indicating which co-correspondence analysis method to use. One of "predictive" (default), or "symmetric", can be abbreviated.
<code>reg.method</code>	One of "simpls" (default) or "eigen". If <code>method</code> is "predictive" then <code>reg.method</code> controls whether the co-correspondence analysis should be fitted using the SIMPLS algorithm or via an eigen analysis.
<code>weights, R0</code>	a vector of length <code>nrow(y)</code> of user supplied weights for $R_0$ . If <code>weights = NULL</code> (default) then the weights are determined from <code>y</code> (default) or <code>x</code> and <code>y</code> ( <code>symmetric = TRUE</code> only).
<code>n.axes</code>	the number of CoCA axes to extract. If missing (default) the <code>n.axes</code> is $\min(\text{ncol}(y), \text{ncol}(x), \text{nrow}(y))$ , 1.
<code>symmetric</code>	if <code>method</code> is "symmetric" then <code>symmetric</code> determines whether weights for $R_0$ are symmetric and taken as the average of the row sums of <code>x</code> and <code>y</code> ( <code>symmetric = TRUE</code> ). If <code>symmetric = FALSE</code> (default) then the weights $R_0$ are taken as the row sums of <code>y</code> unless user defined weights are provided via argument <code>weights</code> . Ignored if <code>method</code> is "predictive".
<code>nam.dat</code>	an optional list with elements <code>namY</code> and <code>namX</code> containing the names of <code>y</code> and <code>x</code> respectively. Used to label printed output. If missing the names of are deduced from <code>y</code> and <code>x</code> .
<code>...</code>	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.

<code>nam.dat</code>	list with components <code>namY</code> and <code>namX</code> containing the names of the response and the predictor(s) respectively.
<code>call</code>	the matched call.
<code>method</code>	the CoCA method used, one of "predictive" or "symmetric".
<code>scores</code>	the species and site scores of the fitted model.
<code>loadings</code>	the site loadings of the fitted model for the response and the predictor. (Predictive CoCA via SIMPLS only.)
<code>fitted</code>	the fitted values for the response. A list with 2 components <code>Yhat</code> (the fitted values) and <code>Yhat1</code> (the transformed fitted values. (Predictive CoCA via SIMPLS only.)
<code>varianceExp</code>	list with components <code>Yblock</code> and <code>Xblock</code> containing the variances in the response and the predictor respectively, explained by each fitted PLS axis. (Predictive CoCA via SIMPLS only.)
<code>totalVar</code>	list with components <code>Yblock</code> and <code>Xblock</code> containing the total variance in the response and the predictor respectively. (Predictive CoCA via SIMPLS only.)
<code>lambda</code>	the Eigenvalues of the analysis.
<code>n.axes</code>	the number of fitted axes
<code>Ychi</code>	a list containing the mean-centered chi-square matrices for the response ( <code>Ychi1</code> ) and the predictor ( <code>Ychi2</code> ). (Predictive CoCA only.)
<code>R0</code>	the (possibly user-supplied) row weights used in the analysis.
<code>X</code>	X-Matrix (symmetric CoCA only).
<code>residuals</code>	Residuals of a symmetric model (symmetric CoCA only).
<code>inertia</code>	list with components <code>total</code> and <code>residual</code> containing the total and residual inertia for the response and the predictor (symmetric CoCA only).
<code>rowsum</code>	a list with the row sums for the response ( <code>rsum1</code> ) and the predictor ( <code>rsum2</code> ) (symmetric CoCA only).
<code>colsum</code>	a list with the column sums for the response ( <code>csum1</code> ) and the predictor ( <code>csum2</code> ) (symmetric CoCA only).

**Author(s)**

Original Matlab code by C.J.F. ter Braak and A.P. Schaffers. R port by Gavin L. Simpson. Formula method for `coca` uses a modified version of `ordiParseFormula` by Jari Oksanen to handle formulae.

**References**

ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

**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.predcoca](#) and [summary.symcoca](#) for summary methods.

**Examples**

```
## symmetric CoCA
data(beetles)
## log transform the beetle data
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)
```

---

```
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

```
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 $R_0$ .
n.axes	number of axes to calculate the co-inertia analysis for.
object	an object of class <code>coinertia</code> .
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)**

Original Matlab code by C.J.F. ter Braak and A.P. Schaffers. R port by Gavin L. Simpson.

**References**

Ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

Doledec, S and Chessel, D. (1994) Co-inertia analysis: a method for studying species-environment relationships. *Freshwater Biology* **31**, 277–294

**See Also**

[symcoca](#) for the function that calls `coinertia` and `coinertiaI` for co-inertia analysis using identity matrices for  $K_1$ ,  $K_2$ , and  $R_0$

---

`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**

```
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:

<code>weights</code>	A list with components <code>X</code> and <code>Y</code> containing the left and right singular vectors respectively of the SVD on the triplets.
<code>scores</code>	A list with components <code>X</code> and <code>Y</code> , containing the row scores of the <code>X</code> and <code>Y</code> species matrices. These are the result of a matrix multiplication of <code>X</code> by the left singular vectors and <code>Y</code> by the right singular vectors.
<code>lambda</code>	the Eigenvalues of the analysis (the square of the singular values from the SVD).
<code>call</code>	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)**

Original Matlab code by C.J.F. ter Braak and A.P. Schaffers. R port by Gavin L. Simpson.

**References**

Doledec, S and Chessel, D. (1994) Co-inertia analysis: a method for studying species-environment relationships. *Freshwater Biology* **31**, 277–294.

**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 <code>symcoca</code> 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**

```
## 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)
```

**Description**

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

**Usage**

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

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

**Arguments**

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

**Value**

Returns a large list with the following components:

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

Gavin L. Simpson, based on Matlab code by C.J.F. ter Braak and A.P. Schaffers.

**See Also**

The model fitting function [coca](#)

**Examples**

```
## load the data sets
data(beetles)
data(plants)
## log transform the beetle 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**

```
## 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)**

Gavin L. Simpson, based on Matlab code by C.J.F. ter Braak and A.P. Schaffers.

**References**

Ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

**See Also**

The model fitting function `coca`

**Examples**

```
## symmetric CoCA
data(beetles)
## log transform the beetle data
beetles <- log(beetles + 1)
data(plants)
## fit the model
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  $(o - e) / \sqrt{e}$  (if  $R_0 = R$ , where R contains the row sums of matrix Y) so that further analysis can be unweighted

**Usage**

```
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 sum(K)

**Note**

This function is not intended for casual use by users.

**Author(s)**

Gavin L. Simpson, based on Matlab code by C.J.F. ter Braak and A.P. Schaffers.

**References**

Ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

**Examples**

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

---

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

```
## S3 method for class 'coca':
permutest(x, R0 = NULL, permutations = 99,
          n.axes = x$n.axes, verbose = TRUE, ...)

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

## Arguments

<code>x</code>	an object of class "predcoca".
<code>R0</code>	row weights to use in the analysis. If missing, the default, these are determined from <code>x</code> .
<code>permutations</code>	the number of permutations to perform.
<code>n.axes</code>	The number of axes to test. Defaults to the number of axes stated in <code>x\$n.axes</code> .
<code>verbose</code>	if TRUE, the default, print information on the progress of the permutation test procedure.
<code>object</code>	an object of class "permutest.coca".
<code>...</code>	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:

<code>pval</code>	a vector of $P$ -values for each ordination axis.
<code>permstat</code>	a vector of values for the test statistic for each axis.
<code>total.inertia</code>	the total inertia in the response matrix.
<code>inertia</code>	a vector containing the inertia explained by each ordination axis.
<code>fitax</code>	a vector containing the fit of each axis to response.
<code>pcent.fit</code>	a vector containing the fit of each axis to the response as a percentage of the total inertia (variance).
<code>n.axes</code>	the number of axes in the ordination.
<code>call</code>	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 ( $R_0$ ) as `R0`. This argument will likely be removed in future versions.

**Author(s)**

Gavin L. Simpson, based on Matlab code by C.J.F. ter Braak and A.P. Schaffers.

**References**

- ter Braak, C.J.F. and de Jong, S. (1998) The objective function of partial least squares regression. *Journal of Chemometrics* **12**, 41–54.
- ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846.
- ter Braak, C.J.F. and Smilauer, P. (2002) *Canoco reference manual and CanoDraw for Windows user's guide: software for canonical community ordination. Version 4.5*. New York: Microcomputer Power.

**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**

```
## load some data
data(beetles)
## log transform the beetle 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**

```
## S3 method for class 'predcoca':
plot(x, choices = c(1:2),
     display = c("species", "site"), oneFig = TRUE,
     type = c("points", "text", "none"),
     ask = prod(par("mfcol")) < 2 && dev.interactive(),
     cex = c(0.7, 0.7), pch = c(par("pch"), 3),
     col = c("black", "red"), ylab, xlab, main, sub = "",
     axes = TRUE, ann = par("ann"), ...)
```

**Arguments**

x an object of class "predcoca", the result of a call to [symcoca](#).

choices	a vector of length 2 indicating which predictive CoCA axes to plot.
display	which sets of scores are drawn. See <code>scores.symcoca</code> .
oneFig	if TRUE, biplots for the response and predictor are plotted on the same figure using <code>par(mfrow = c(1, 2))</code> . If FALSE each biplot is plotted in turn on the active device. The first plot will be overwritten unless <code>ask = TRUE</code> , see below.
type	one of "points", "text", or "none". Determines how the site and species scores are displayed. If <code>type = "points"</code> , scores are plotted as points using plotting characters given in argument <code>pch</code> . If <code>type = "text"</code> , then the row names of the scores matrices are plotted. If <code>type = "none"</code> , then the scores are not plotted.
ask	logical, if TRUE, the user is <i>asked</i> before each plot, see <code>par</code>
cex	a vector of length 2, containing the character expansion factors to use for the samples (sites) and the species.
col	a vector of length 2, containing the colours used to draw the samples (sites) and the species markers.
pch	a vector of length 2, containing the plotting character to use for the samples (sites) and the species.
xlab, ylab	labels for the x and y axes, defaults to "Axis X (lamda_X = Y)", where 'X' is the CoCa axis plotted and 'Y' is the Eigenvalue for axis 'X'.
main	a main title for the plot, defaults to the names given in <code>x\$nam.dat</code> . User supplied values currently ignored.
sub	a sub title for the plots. Currently the same sub title is applied to each biplot.
axes	a logical value indicating whether both axes should be drawn on the plot.
ann	logical, if TRUE plots are annotated and not if FALSE, currently ignored.
...	other graphical parameters as in 'par' may also be passed as arguments.

### Note

This is an improved plot method for `preccoca` objects, that is far more configurable. More intuitive handling of user supplied values for `main`, `xlab`, `ylab` and `sub` will be provided in the next version.

### Author(s)

Gavin L. Simpson.

### References

Ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

### See Also

`coca`, `plot.default`

**Examples**

```
## predictive CoCA
data(beetles)

## log transform the beetle 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)
```

---

plot.symcoca

*Biplots for symmetric co-correspondence analysis*


---

**Description**

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

**Usage**

```
## S3 method for class 'symcoca':
plot(x, choices = c(1:2),
      display = c("species", "site"), scaling = FALSE,
      oneFig = TRUE, type = c("points", "text", "none"),
      ask = prod(par("mfcol")) < 2 && dev.interactive(),
      cex = c(0.7, 0.7), pch = c(par("pch"), 3),
      col = c("black", "red"), ylab, xlab, main, sub = "",
      axes = TRUE, ann = par("ann"), ...)
```

**Arguments**

x	an object of class "symcoca", the result of a call to <a href="#">symcoca</a> .
choices	a vector of length 2 indicating which predictive CoCA axes to plot.
display	which sets of scores are drawn. See <a href="#">scores.symcoca</a> .
scaling	logical, whether scaling should be applied. See <a href="#">scores.symcoca</a> .
oneFig	if TRUE, biplots for the response and predictor are plotted on the same figure using <code>par(mfrow = c(1, 2))</code> . If FALSE each biplot is plotted in turn on the active device. The first plot will be overwritten unless <code>ask = TRUE</code> , see below.

type	one of "points", "text", or "none". Determines how the site and species scores are displayed. If type = "points", scores are plotted as points using plotting characters given in argument pch. If type = "text", then the row names of the scores matrices are plotted. If type = "none", then the scores are not plotted.
ask	logical, if TRUE, the user is <i>asked</i> before each plot, see <a href="#">par</a> .
cex	a vector of length 2, containing the character expansion factors to use for the samples (sites) and the species.
col	a vector of length 2, containing the colours used to draw the samples (sites) and the species markers.
pch	a vector of length 2, containing the plotting character to use for the samples (sites) and the species.
xlab, ylab	labels for the x and y axes, defaults to "Axis X (lamda_X = Y)", where 'X' is the CoCa axis plotted and 'Y' is the Eigenvalue for axis 'X'.
main	a main title for the plot, defaults to the names given in x\$nam.dat. User supplied values currently ignored.
sub	a sub title for the plots. Currently the same sub title is applied to each biplot.
axes	a logical value indicating whether both axes should be drawn on the plot.
ann	logical, if TRUE plots are annotated and not if FALSE, currently ignored.
...	other graphical parameters as in 'par' may also be passed as arguments.

### Note

This is an improved plot method for `symcoca` objects, that is far more configurable. More intuitive handling of user supplied values for `main`, `xlab`, `ylab` and `sub` will be provided in the next version.

### Author(s)

Gavin L. Simpson.

### References

Ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

### See Also

[coca, plot.default](#)

### Examples

```
## symmetric CoCA
data(beetles)

## log transform the beetle 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*

---

### Description

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

### Usage

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

## Default S3 method:

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

## S3 method for class 'symcoca':

```
rescale(object, axes = c(1:object$n.axes), ...)
```

### Arguments

<code>object</code>	an R object. Currently only objects of class "symcoca" are supported.
<code>axes</code>	the number of axes to rescale
<code>...</code>	other arguments to be passed to <code>rescale</code> 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 predator

### Author(s)

Matlab original by C.J.F. ter Braak and A.P. Schaffers. R port by Gavin L. Simpson.

**See Also**[symcoca](#)**Examples**

```

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

```

---

resid.symcoca	<i>Extract Model Residuals</i>
---------------	--------------------------------

---

**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	<i>Standardised chi-square residuals</i>
----------	--

---

**Description**

Scales a matrix, Y, to its 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**

```
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)**

Gavin L. Simpson, based on Matlab code by C.J.F. ter Braak and A.P. Schaffers.

**References**

Ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

---

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.

### 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 <a href="#">rescale.symcoca</a> .
...	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)

Gavin L. Simpson, based on Matlab code by C.J.F. ter Braak and A.P. Schaffers.

## References

ter Braak, C.J.F and Schaffers, A.P. (2004) Co-Correspondence Analysis: a new ordination method to relate two community compositions. *Ecology* **85**(3), 834–846

## See Also

[scores](#), for further details on the method.

## Examples

```
## load some data
data(beetles)
data(plants)
## log transform the beetle 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

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

## Arguments

X	a matrix of observations. NAs and Infs are not allowed.
Y	a vector or matrix of responses. NAs and Infs 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 modifications 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:

<code>coefficients</code>	an array of regression coefficients for 1, ..., <code>ncomp</code> components. The dimensions of <code>coefficients</code> are <code>c(nvar, npred, ncomp)</code> with <code>nvar</code> the number of <code>X</code> variables and <code>npred</code> the number of variables to be predicted in <code>Y</code> .
<code>scores</code>	a matrix of scores.
<code>loadings</code>	a matrix of loadings.
<code>Yscores</code>	a matrix of <code>Y</code> -scores.
<code>Yloadings</code>	a matrix of <code>Y</code> -loadings.
<code>projection</code>	the projection matrix used to convert <code>X</code> to scores.
<code>Xmeans</code>	a vector of means of the <code>X</code> variables.
<code>Ymeans</code>	a vector of means of the <code>Y</code> variables.
<code>fitted.values</code>	an array of fitted values. The dimensions of <code>fitted.values</code> are <code>c(nobj, npred, ncomp)</code> with <code>nobj</code> the number of samples and <code>npred</code> the number of <code>Y</code> variables.
<code>residuals</code>	an array of regression residuals. It has the same dimensions as <code>fitted.values</code> .
<code>Xvar</code>	a vector with the amount of <code>X</code> -variance explained by each number of components.
<code>Yvar</code>	a vector with the amount of <code>Y</code> -variance explained by each number of components.
<code>Xtotvar</code>	Total variance in <code>X</code> .
<code>Ytotvar</code>	Total variance in <code>Y</code> .

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**

de Jong, S. (1993) SIMPLS: an alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, **18**, 251–263.

**See Also**

[coca](#)

---

summary.predcoca     *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 <a href="#">coca</a> .
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 <code>predcoca.simpls</code> .
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 <code>loadings1</code> and <code>loadings2</code> , which refer to the response and the predictor matrices respectively. (Only for predictive CoCA models.)
varianceExp	a list with components <code>Yblock</code> and <code>Xblock</code> 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 <code>Yblock</code> and <code>Xblock</code> containing the total variance in the response and the predictor data sets respectively
inertia	a list with components <code>total</code> and <code>residual</code> 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**

```
## 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, 2  
bryophyte, 3

## \*Topic **hplot**

plot.predcoca, 18  
plot.symcoca, 20

## \*Topic **models**

coca, 4  
coinertia, 7  
coinertiaI, 9

## \*Topic **multivariate**

coca, 4  
coinertia, 7  
coinertiaI, 9  
crossval, 11  
fitted.symcoca, 13  
mcChi, 14  
permutest.coca, 15  
rescale, 22  
resid.symcoca, 23  
scaleChi, 24  
scores.predcoca, 25  
simpls, 26  
summary.predcoca, 28

## \*Topic **package**

cocorresp-package, 2

## \*Topic **regression**

coca, 4  
simpls, 26

## \*Topic **univar**

corAxis, 10

beetles, 2

bryophyte, 3

coca, 4, 12, 14, 17, 19, 21, 27–29

cocorresp (*cocorresp-package*), 2

cocorresp-package, 2

coinertia, 7, 10

coinertiaI, 9, 9

cor, 11

corAxis, 10

crossval, 6, 11, 16, 17

fitted.symcoca, 13

mcChi, 14

ordiParseFormula, 6

par, 18, 20

permutest.coca, 6, 9, 10, 15

plants (*beetles*), 2

plot.default, 19, 21

plot.predcoca, 18

plot.symcoca, 20

predcoca.eigen (*coca*), 4

predcoca.simpls, 29

predcoca.simpls (*coca*), 4

print.coinertia (*coinertia*), 7

print.crossval (*crossval*), 11

print.fitted.symcoca  
(*fitted.symcoca*), 13

print.permutest.coca  
(*permutest.coca*), 15

print.predcoca (*coca*), 4

print.summary.coinertia  
(*coinertia*), 7

print.summary.crossval  
(*crossval*), 11

print.summary.permutest.coca  
(*permutest.coca*), 15

print.summary.predcoca  
(*summary.predcoca*), 28

print.summary.symcoca  
(*summary.predcoca*), 28

print.symcoca (*coca*), 4

rescale, 22

rescale.symcoca, 25

resid.symcoca, 23

scaleChi, 24  
scores, 25, 26  
scores.predcoca, 25  
scores.symcoca, 18, 20  
scores.symcoca(scores.predcoca),  
25  
simpls, 26  
simpls.fit, 5, 27, 28  
summary.coinertia(coinertia), 7  
summary.crossval(crossval), 11  
summary.permutest.coca  
(permutest.coca), 15  
summary.predcoca, 6, 28  
summary.symcoca, 6  
summary.symcoca  
(summary.predcoca), 28  
symcoca, 8–10, 18, 20, 22, 23  
symcoca(coca), 4  
vascular(bryophyte), 3