

Package: yacca (via r-universe)

October 11, 2024

Type Package

Title Yet Another Canonical Correlation Analysis Package

Version 1.4-2

Date 2022-03-08

Depends R (>= 1.8.0), utils

Suggests RSpectra

Author Carter T. Butts <buttsc@uci.edu>

Maintainer Carter T. Butts <buttsc@uci.edu>

Description An alternative canonical correlation/redundancy analysis function, with associated print, plot, and summary methods. A method for generating helio plots is also included.

License GPL (>= 3)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2022-03-07 07:40:02 UTC

Contents

yacca-package	2
cca	2
F.test.cca	6
helio.plot	8

Index	10
--------------	-----------

yacca-package

Yet Another Canonical Correlation Analysis Package

Description

This package provides an alternative canonical correlation/redundancy analysis function, with associated print, plot, and summary methods. A method for generating helio plots is also included.

Details

For details on using the package, see [cca](#) and [helio.plot](#).

Author(s)

Carter T. Butts <buttsc@uci.edu>

Maintainer: Carter T. Butts <buttsc@uci.edu>

References

Mardia, K. V.; Kent, J. T.; and Bibby, J. M. 1979. *Multivariate Analysis*. London: Academic Press.

cca

Canonical Correlation Analysis

Description

Performs a canonical correlation (and canonical redundancy) analysis on two sets of variables.

Usage

```
cca(x, y, xlab = colnames(x), ylab = colnames(y), xcenter = TRUE,  
    ycenter = TRUE, xscale = FALSE, yscale = FALSE,  
    standardize.scores = TRUE, use = "complete.obs", na.rm = TRUE,  
    use.eigs = FALSE, max.dim = Inf, reg.param = NULL)
```

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

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

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

Arguments

<code>x</code>	for <code>cca</code> , a single vector or a matrix whose columns contain the <code>x</code> variables. Otherwise, a <code>cca</code> object.
<code>y</code>	a single vector or a matrix whose columns contain the <code>y</code> variables.
<code>xlab</code>	an optional vector of <code>x</code> labels.
<code>ylab</code>	an optional vector of <code>y</code> labels.
<code>xcenter</code>	boolean; demean the <code>x</code> variables?
<code>ycenter</code>	boolean; demean the <code>y</code> variables?
<code>xscale</code>	boolean; scale the <code>x</code> variables to unit variance?
<code>yscale</code>	boolean; scale the <code>y</code> variables to unit variance?
<code>standardize.scores</code>	boolean; rescale scores (and coefficients) to produce scores of unit variance?
<code>use</code>	use argument to be passed to <code>var</code> when creating covariance matrices.
<code>na.rm</code>	boolean; remove missing values during redundancy analysis?
<code>use.eigs</code>	boolean; use <code>eigs</code> instead of <code>eigen</code> for diagonalization?
<code>max.dim</code>	maximum number of canonical variates to extract (only relevant if less than the minimum of the number of columns of <code>x</code> and <code>y</code>)
<code>reg.param</code>	an optional L2 regularization parameter (or vector thereof).
<code>object</code>	a <code>cca</code> object.
<code>...</code>	additional arguments.

Details

Canonical correlation analysis (CCA) is a form of linear subspace analysis, and involves the projection of two sets of vectors (here, the variable sets `x` and `y`) onto a joint subspace. The goal of (CCA) is to find a sequence of linear transformations of each variable set, such that the correlations between the transformed variables are maximized (under the proviso that each transformed variable must be orthogonal to those preceding it). These transformed variables – known as “canonical variates” (CVs) – can be thought of as expressing the common variation across the data sets, in a manner analogous to the role of principal components in within-set analysis (see, e.g., `princomp`). Since the rank of the joint subspace is equal to the minimum of the ranks of the two spaces spanned by the initial data vectors, it follows that the number of CVs will usually be equal to the minimum of the number of `x` and `y` variables (perhaps fewer, if the sets are not of full rank or if `max.dim` is used to constrain the number of variables extracted).

Formally, we may describe the CCA solution as follows. Given data matrices X and Y , let Σ_{XX} , Σ_{XY} , Σ_{YX} and Σ_{YY} be the respective sample covariance matrices for X versus itself, X versus Y , Y versus X , and Y versus itself. Now, for some i less than or equal to the minimum rank of X and Y , let u_i be the i th eigenvector of $\Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{YX}$, with corresponding eigenvalue λ_i . Then the vector u_i contains the coefficients projecting X onto the i th canonical variate; the corresponding scores are given by Xu_i . Similarly, let v_i be the i th eigenvector of $\Sigma_{YY}^{-1} \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY}$. Then v_i contains the coefficients projecting Y onto the i th canonical variate (with scores Yv_i). The eigenvalue in the second case will be the same as the first, and corresponds to the square of the i th canonical correlation for the CCA solution – that is, the correlation between the X and Y scores on

the i th canonical variate. Since the canonical correlation structure is unaffected by rescaling of the canonical variate scores, it is common to adjust the coefficients u_i and v_i to ensure that the resulting scores have unit variance; this option is controlled here via the `standardize.scores` argument.

CCA output can be fairly complex. Quantities of particular interest include the correlations between the original variables in each set and their respective canonical variates (*structural correlations* or *loadings*), the coefficients which take the original variables into the CVs, and of course the correlations between the CV scores in one set and their corresponding scores in the opposite set (the *canonical correlations*). The canonical correlations provide a basic measure of concordance between the transformed variables, but are surprisingly uninformative by themselves; canonical redundancies (see below) are of more typical interest. Interpretation of CVs is usually performed by inspection of loadings, which reveal the extent to which each CV is associated with particular variables in each set. The squared loadings, in particular, convey the fraction of variance in each original variable which is accounted for by a given CV (though not necessarily by the variables in the opposite set!).

A common interest in the context of CCA is the extent to which the variance of one set of variables can be accounted for by the other (in the usual least squares sense). While it is tempting to interpret the squared canonical correlations in this manner, this is incorrect: the squared canonical correlations convey the fraction of variance in the CV scores from one variable set which can be accounted for by scores from the other, but say nothing about the extent to which the CVs themselves account for variation in the original variables. The variance in one set explainable by the other is instead expressed via the so-called *redundancy index*, which combines the squared canonical correlations with the *canonical adequacy* (within-set variance accounted for) for each CV. The use of the redundancy index in this way is sometimes called “(canonical) redundancy analysis”, although it is simply an alternate means of presenting CCA results.

As the name of the technique implies, CCA is a symmetric procedure: the designation of one variable set as x and another as y is arbitrary, and may be reversed without incident. (Note, however, that the coefficients and redundancies are set-specific, and will also be reversed in this case.) CCA with one x or y variable is equivalent to OLS regression (with the squared canonical correlation corresponding to the R^2), and CCA on one variable pair yields the familiar Pearson product-moment correlation. Centering and scaling data prior to analysis is equivalent to working with correlation matrices in the underlying analysis (with interpretation/effects analogous to the principal components case).

Finding the CCA solution can pose numerical challenges, ironically more so when the degree of potential dimension reduction is highest. In recalcitrant cases, it can be useful to apply regularization to the solution for purposes of stabilization. The optional `reg.param` can be used for this purpose: if given as a single numeric value, it adds an L2 (aka “ridge”) penalty to each variable set with the corresponding multiplier value. `reg.param` can also be given as a vector of length 2, in which case the first value is applied to the x variables and the second is applied to the y variables. Relatedly, in high-dimension/low-rank problems it can be useful to extract a much smaller number of canonical variates than the nominal maximum. This can be controlled by `max.dim`, though the default diagonalization method computes the entire eigendecomposition prior to canonical variate extraction. In such cases, it can be helpful to employ the alternative diagonalization method controlled by the use of `eigs` argument to compute only those dimensions that are actually required. Experience suggests that this method (`eigs`) is less stable than the base `eigen`, but it can be much faster in high-dimensional settings.

Value

An object of class `cca`, whose elements are as follows:

<code>corr</code>	Canonical correlations.
<code>corrsq</code>	Squared canonical correlations (shared variance across canonical variates).
<code>xcoef</code>	Coefficients for the x variables on each canonical variate.
<code>ycoef</code>	Coefficients for the y variables on each canonical variate.
<code>canvarx</code>	Canonical variate scores for the x variables.
<code>canvary</code>	Canonical variate scores for the y variables.
<code>xstructcorr</code>	Structural correlations (loadings) for x variables on each canonical variate.
<code>ystructcorr</code>	Structural correlations (loadings) for y variables on each canonical variate.
<code>xstructcorrsq</code>	Squared structural correlations for x variables on each canonical variate (i.e., fraction of x variance associated with each variate).
<code>ystructcorrsq</code>	Squared structural correlations for y variables on each canonical variate (i.e., fraction of y variance associated with each variate).
<code>xcrosscorr</code>	Canonical cross-loadings for x variables on the y scores for each canonical variate.
<code>ycrosscorr</code>	Canonical cross-loadings for y variables on the x scores for each canonical variate.
<code>xcrosscorrsq</code>	Squared canonical cross-loadings for x variables on the y scores for each canonical variate (i.e., the fraction of variance in each x variable attributable to y through the respective CVs).
<code>ycrosscorrsq</code>	Squared canonical cross-loadings for y variables on the x scores for each canonical variate (i.e., the fraction of variance in each y variable attributable to x through the respective CVs).
<code>xcancom</code>	Canonical communalities for x variables (for each x variable, fraction associated with all canonical variates).
<code>ycancom</code>	Canonical communalities for y variables (for each y variable, fraction associated with all canonical variates).
<code>xcanvad</code>	Canonical variate adequacies for x variables (for each canonical variate, fraction of total x variance for which it is associated).
<code>ycanvad</code>	Canonical variate adequacies for y variables (for each canonical variate, fraction of total y variance for which it is associated).
<code>xvrd</code>	Canonical redundancies for x variables (i.e., total fraction of x variance accounted for by y variables, through each canonical variate).
<code>yvrd</code>	Canonical redundancies for y variables (i.e., total fraction of y variance accounted for by x variables, through each canonical variate).
<code>xrd</code>	Total canonical redundancy for x variables (i.e., total fraction of x variance accounted for by y variables, through all canonical variates).
<code>yrd</code>	Total canonical redundancy for y variables (i.e., total fraction of y variance accounted for by x variables, through all canonical variates).

chisq	Sequential χ^2 values for tests of each respective canonical variate using Bartlett's omnibus statistic.
df	Degrees of freedom for Bartlett's test.
xlab	Variable names for x.
ylab	Variable names for y.
reg.param	Regularization parameter (if any).

Author(s)

Carter T. Butts <buttsc@uci.edu>

References

Mardia, K. V.; Kent, J. T.; and Bibby, J. M. 1979. *Multivariate Analysis*. London: Academic Press.

See Also

[F.test.cca](#), [cancor](#), [princomp](#)

Examples

```
#Example parallels the R builtin cancort example
data(LifeCycleSavings)
pop <- LifeCycleSavings[, 2:3]
oec <- LifeCycleSavings[, -(2:3)]
cca.fit <- cca(pop, oec)
cca.regfit <- cca(pop, oec, reg.param=1) # Some minimal regularization

#View the results
cca.fit
summary(cca.fit)
plot(cca.fit)
cca.regfit      #Not a vast difference, usually....
```

F.test.cca

F Test for Canonical Correlations Using Rao's Approximation

Description

Tests a series of canonical correlations (sequentially) against the null hypothesis that the tested coefficient and all succeeding coefficients are zero.

Usage

```
F.test.cca(x, ...)

## S3 method for class 'F.test.cca'
print(x, ...)
```

Arguments

x a cca object.
 ... additional arguments.

Details

Several related tests have been proposed for the evaluation of canonical correlations (including Bartlett's Chi-squared test, which is computed by default within `cca`). This function employs Rao's statistic (related to Wilks' Lambda) as the basis for an F test of each coefficient (and all others in ascending sequence) against the hypothesis that the associated population correlations are zero.

Value

An object of class `F.test.cca`, whose elements are as follows:

corr Canonical correlations.
 statistic Squared canonical correlations (shared variance across canonical variates).
 parameter Coefficients for the x variables on each canonical variate.
 p.value Coefficients for the y variables on each canonical variate.
 method Canonical variate scores for the x variables.
 data.name Canonical variate scores for the y variables.

Author(s)

Nicholas L. Crookston <ncrookston@fs.fed.us>
 Carter T. Butts <buttsc@uci.edu>

References

Mardia, K. V.; Kent, J. T.; and Bibby, J. M. 1979. *Multivariate Analysis*. London: Academic Press.

See Also

[cca](#)

Examples

```
#Example: perceived personal attributes versus professional performance
#for US Judges
data(USJudgeRatings)
personal <- USJudgeRatings[,c("INTG", "DMNR", "DILG", "FAMI", "PHYS")]
performance <- USJudgeRatings[,c("CFMG", "DECI", "PREP", "ORAL", "WRIT")]
cca.fit <- cca(personal, performance)

#Test the canonical correlations (see also summary(cca.fit))
F.test.cca(cca.fit)
```

helio.plot

Helio Plots

Description

Displays data using a circular layout; function is designed to be used with `cca` objects, but could perhaps be rigged for use in other circumstances.

Usage

```
helio.plot(c, cv = 1, xvlab = c$xlabs, yvlab = c$ylabs,
  x.name = "X Variables", y.name = "Y Variables", lab.cex = 1,
  wid.fact = 0.75, main = "Helio Plot",
  sub = paste("Canonical Variate", cv, sep = "" ), zero.rad = 30,
  range.rad = 20, name.padding = 5, name.cex = 1.5,
  axis.circ = c(-1, 1), x.group = rep(0, dim(c$xstructcorr)[1]),
  y.group = rep(0, dim(c$ystructcorr)[1]), type = "correlation")
```

Arguments

<code>c</code>	object to be plotted (generally output from <code>cca</code>).
<code>cv</code>	the canonical variate to display.
<code>xvlab</code>	X variable labels.
<code>yvlab</code>	Y variable labels.
<code>x.name</code>	name for the X variable set.
<code>y.name</code>	name for the Y variable set.
<code>lab.cex</code>	character expansion for plot labels.
<code>wid.fact</code>	width multiplier for data bars.
<code>main</code>	plot main title.
<code>sub</code>	plot subtitle.
<code>zero.rad</code>	radius for the zero-value reference circle.
<code>range.rad</code>	difference between inner and outer plotting radius.
<code>name.padding</code>	offset for variable names.
<code>name.cex</code>	character expansion for variable names.
<code>axis.circ</code>	location to draw axis circles.
<code>x.group</code>	optional grouping vector for X variables.
<code>y.group</code>	optional grouping vector for Y variables.
<code>type</code>	one of "correlation" or "variance", depending on the type of data to be displayed.

Details

Helio plots display data in radial bars, with larger values pointing outward from a base reference circle and smaller (more negative) values pointing inward). Such plots are well-suited to the display of multivariate information with several groups of variables, as with canonical correlation analysis.

Value

None.

Author(s)

Carter T. Butts <buttsc@uci.edu>

See Also

[cca](#)

Examples

```
data(LifeCycleSavings)
pop <- LifeCycleSavings[, 2:3]
oec <- LifeCycleSavings[, -(2:3)]
cca.fit <- cca(pop, oec)

#Show loadings on first canonical variate
helio.plot(cca.fit, x.name="Population Variables",
           y.name="Economic Variables")

#Show variances on second canonical variate
helio.plot(cca.fit, cv=2, x.name="Population Variables",
           y.name="Economic Variables", type="variance")
```

Index

- * **hplot**
 - helio.plot, 8
- * **htest**
 - F.test.cca, 6
- * **models**
 - cca, 2
- * **multivariate**
 - cca, 2
 - F.test.cca, 6
- * **package**
 - yacca-package, 2

cancor, 6
cca, 2, 2, 7–9

eigen, 3, 4
eigs, 3, 4

F.test.cca, 6, 6

helio.plot, 2, 8

plot.cca (cca), 2
princomp, 3, 6
print.cca (cca), 2
print.F.test.cca (F.test.cca), 6
print.summary.cca (cca), 2

summary.cca (cca), 2

var, 3

yacca (yacca-package), 2
yacca-package, 2