Package: nscancor (via r-universe)

August 23, 2024

Version 0.7.0-6

Title Non-Negative and Sparse CCA

Description Two implementations of canonical correlation analysis (CCA) that are based on iterated regression. By choosing the appropriate regression algorithm for each data domain, it is possible to enforce sparsity, non-negativity or other kinds of constraints on the projection vectors. Multiple canonical variables are computed sequentially using a generalized deflation scheme, where the additional correlation not explained by previous variables is maximized. nscancor() is used to analyze paired data from two domains, and has the same interface as cancor() from the 'stats' package (plus some extra parameters). mcancor() is appropriate for analyzing data from three or more domains. See <https://sigg-iten.ch/learningbits/2014/01/20/canonical-correlation-analysis-under-constraints/> and Sigg et al. (2007) <doi:10.1109/MLSP.2007.4414315> for more details.

URL https://sigg-iten.ch/research/

BugReports https://github.com/chrsigg/nscancor/issues

License GPL (>=2)

Depends R (>= 3.6.0)

Imports methods, stats

Suggests CCA, glmnet, MASS, roxygen2, testthat (>= 0.8), V8

RoxygenNote 7.2.3

Encoding UTF-8

Repository https://chrsigg.r-universe.dev

RemoteUrl https://github.com/chrsigg/nscancor

RemoteRef HEAD

RemoteSha 8f3eb56a577abee7643b0d82e53255456b256ab4

Contents

`acor` *Additional Explained Correlation*

Description

`acor` computes the additional standard correlation explained by each canonical variable, taking into account the possible non-conjugacy of the canonical vectors. The result of the analysis is returned as a list of class `nscancor`.

Usage

```
acor(
  x,
  xcoef,
  y,
  ycoef,
  xcenter = TRUE,
  ycenter = TRUE,
  xscale = FALSE,
  yscale = FALSE
)
```

Arguments

`x`	a numeric matrix which provides the data from the first domain
`xcoef`	a numeric data matrix with the canonical vectors related to `x` as its columns.
`y`	a numeric matrix which provides the data from the second domain
`ycoef`	a numeric data matrix with the canonical vectors related to `y` as its columns.
`xcenter`	a logical value indicating whether the empirical mean of (each column of) `x` should be subtracted. Alternatively, a vector of length equal to the number of columns of `x` can be supplied. The value is passed to `scale`.
`ycenter`	analogous to `xcenter`
`xscale`	a logical value indicating whether the columns of `x` should be scaled to have unit variance before the analysis takes place. The default is `FALSE` for consistency with `cancor`. Alternatively, a vector of length equal to the number of columns of `x` can be supplied. The value is passed to `scale`.
`yscale`	analogous to `xscale`

Details

The additional correlation is measured after projecting the corresponding canonical vectors to the ortho-complement space spanned by the previous canonical variables. This procedure ensures that the correlation explained by non-conjugate canonical vectors is not counted multiple times. See Mackey (2009) for a presentation of generalized deflation in the context of principal component analysis (PCA), which was adapted here to CCA.

`acor` is also useful to build a partial CCA model, to be completed with additional canonical variables computed using `nscancor`.

Value

A list of class `nscancor` containing the following elements:

`cor`	the additional correlation explained by each pair of canonical variables
`xcoef`	copied from the input arguments
`ycoef`, `ycenter`, `yscale`	copied from the input arguments
`xp`	the deflated data matrix corresponding to `x`
`yp`	analogous to `xp`

References

Mackey, L. (2009) Deflation Methods for Sparse PCA. In *Advances in Neural Information Processing Systems* (pp. 1017–1024).

Examples

```
data(nutrimouse, package = "CCA")

x <- nutrimouse$gene[ , 1:5]
y <- nutrimouse$lipid
cc <- cancor(x, y)

# Re-compute explained correlation
ac <- acor(x, cc$xcoef, y, cc$ycoef)

# Results should agree
print(cc$cor)
print(ac$cor)
```

colCardinalities *Cardinality of Column Vectors*

Description

Computes the cardinality (the number of non-zero elements) of each column of the matrix $\mathbf{w}$.

Usage

```
colCardinalities(w)
```

Arguments

w	a numeric matrix, e.g. xcoef as returned by nscancor

Value

A vector containing the number of non-zero elements of each column of w

Examples

```
# returns c(2, 1)
colCardinalities(matrix(c(1, 0, 2, -1, 0, 0), ncol = 2))
```

macor *Multi-Domain Additional Explained Correlation*

Description

macor generalizes acor to the case of more than two data domains.

Usage

```
macor(x, coef, center = TRUE, scale_ = FALSE)
```

Arguments

x	a list of numeric matrices which contain the data from the different domains
coef	a list of matrices containing the canonical vectors related to each data domain. Each matrix contains the respective canonical vectors as its columns.
center	a list of logical values indicating whether the empirical mean of (each column of) the corresponding data matrix should be subtracted. Alternatively, a list of vectors can be supplied, where each vector specifies the mean to be subtracted from the corresponding data matrix. Each list element is passed to scale.

scale_ a list of logical values indicating whether the columns of the corresponding data matrix should be scaled to have unit variance before the analysis takes place. The default is FALSE for consistency with acor. Alternatively, a list of vectors can be supplied, where each vector specifies the standard deviations used to rescale the columns of the corresponding data matrix. Each list element is passed to scale.

Value

A list of class mcancor with the following elements:

cor	a multi-dimensional array containing the additional correlations explained by each pair of canonical variables. The first two dimensions correspond to the domains, and the third dimension corresponds to the different canonical variables per domain.
coef	copied from the input arguments
center	the list of empirical means used to center the data matrices
scale	the list of empirical standard deviations used to scale the data matrices
xp	the list of deflated data matrices corresponding to x

Examples

```
x <- matrix(runif(10*5), 10)
y <- matrix(runif(10*5), 10)
z <- matrix(runif(10*5), 10)

xcoef <- matrix(rnorm(2*5), 5)
ycoef <- matrix(rnorm(2*5), 5)
zcoef <- matrix(rnorm(2*5), 5)

# Explained multi-domain correlation
macor(list(x, y, z), list(xcoef, ycoef, zcoef))$cor
```

---

mcancor *Non-Negative and Sparse Multi-Domain CCA*

Description

Performs a canonical correlation analysis (CCA) on multiple data domains, where constraints such as non-negativity or sparsity are enforced on the canonical vectors. The result of the analysis is returned as a list of class mcancor.

Usage

```
mcancor(
  x,
  center = TRUE,
  scale_ = FALSE,
  nvar = min(sapply(x, dim)),
  predict,
  cor_tol = NULL,
  nrestart = 10,
  iter_tol = 0,
  iter_max = 50,
  partial_model = NULL,
  verbosity = 0
)
```

Arguments

x	a list of numeric matrices which contain the data from the different domains
center	a list of logical values indicating whether the empirical mean of (each column of) the corresponding data matrix should be subtracted. Alternatively, a list of vectors can be supplied, where each vector specifies the mean to be subtracted from the corresponding data matrix. Each list element is passed to scale.
scale_	a list of logical values indicating whether the columns of the corresponding data matrix should be scaled to have unit variance before the analysis takes place. The default is FALSE for consistency with nscancor. Alternatively, a list of vectors can be supplied, where each vector specifies the standard deviations used to rescale the columns of the corresponding data matrix. Each list element is passed to scale.
nvar	the number of canonical variables to be computed for each domain. With the default setting, canonical variables are computed until at least one data matrix is fully deflated.
predict	a list of regression functions to predict the sum of the canonical variables of all other domains. The formal arguments for each regression function are the design matrix x corresponding to the data from the current domain, the regression target sc as the sum of the canonical variables for all other domains, and cc as a counter of which canonical variable is currently computed (e.g. for enforcing different constraints for subsequent canonical vectors of a given domain). See the examples for an illustration.
cor_tol	a threshold indicating the magnitude below which canonical variables should be omitted. Variables are omitted if the sum of all their correlations are less than or equal to cor_tol times the sum of all correlations of the first canonical variables of all domains. With the default NULL setting, no variables are omitted.
nrestart	the number of random restarts for computing the canonical variables via iterated regression steps. The solution achieving maximum explained correlation over all random restarts is kept. A value greater than one can help to avoid poor local maxima.

iter_tol	If the relative change of the objective is less than iter_tol between iterations, the procedure is assumed to have converged to a local optimum.
iter_max	the maximum number of iterations to be performed. The procedure is terminated if either the iter_tol or the iter_max criterion is satisfied.
partial_model	NULL or an object of class mcancor. The computation can be continued from a partial model by providing an mcancor object (either from a previous run of this function or from macor) and setting nvar to a value greater than the number of canonical variables contained in the partial model. See the examples for an illustration.
verbosity	an integer specifying the verbosity level. Greater values result in more output, the default is to be quiet.

Details

mcancor generalizes nscancor to the case where more than two data domains are available for an analysis. Its objective is to maximize the sum of all pairwise correlations of the canonical variables.

Value

mcancor returns a list of class mcancor with the following elements:

cor	a multi-dimensional array containing the additional correlations explained by each pair of canonical variables. The first two dimensions correspond to the domains, and the third dimension corresponds to the different canonical variables per domain (see also macor).
coef	a list of matrices containing the canonical vectors related to each data domain. The canonical vectors are stored as the columns of each matrix.
center	the list of empirical means used to center the data matrices
scale	the list of empirical standard deviations used to scale the data matrices
xp	the list of deflated data matrices corresponding to x

See Also

macor, nscancor, scale

Examples

```
# As of version 1.2.1 of the PMA package, breastdata.rda is no longer
# contained in the package and needs to be downloaded separately
breastdata_url <- "https://statweb.stanford.edu/~tibs/PMA/breastdata.rda"
breastdata_file <- tempfile("breastdata_", fileext = ".rda")
status <- download.file(breastdata_url, breastdata_file, mode = "wb")
if (status > 0)
  stop("Unable to download from", breastdata_url)
load(breastdata_file)

# Three data domains: a subset of genes, and CGH spots for the first and
# second chromosome
```

```
x <- with(
  breastdata,
  list(t(rna)[ , 1:100], t(dna)[ , chrom == 1], t(dna)[ , chrom == 2])
)

# Sparse regression functions with different cardinalities for different domains
generate_predict <- function(dfmax) {
  force(dfmax)
  return(
    function(x, sc, cc) {
      en <- glmnet::glmnet(x, sc, alpha = 0.05, intercept = FALSE, dfmax = dfmax)
      W <- coef(en)
      return(W[2:nrow(W), ncol(W)])
    }
  )
}
predict <- lapply(c(20, 10, 10), generate_predict)

# Compute two canonical variables per domain
mcc <- mcancor(x, predict = predict, nvar = 2)

# Compute another canonical variable for each domain
mcc <- mcancor(x, predict = predict, nvar = 3, partial_model = mcc)
mcc$cor
```

---

nscancor *Non-Negative and Sparse CCA*

Description

Performs a canonical correlation analysis (CCA) where constraints such as non-negativity or sparsity are enforced on the canonical vectors. The result of the analysis is returned as a list of class nscancor, which contains a superset of the elements returned by cancor.

Usage

```
nscancor(
  x,
  y,
  xcenter = TRUE,
  ycenter = TRUE,
  xscale = FALSE,
  yscale = FALSE,
  nvar = min(dim(x), dim(y)),
  xpredict,
  ypredict,
  cor_tol = NULL,
```

```
  nrestart = 10,
  iter_tol = 0,
  iter_max = 50,
  partial_model = NULL,
  verbosity = 0
)
```

Arguments

x	a numeric matrix which provides the data from the first domain
y	a numeric matrix which provides the data from the second domain
xcenter	a logical value indicating whether the empirical mean of (each column of) x should be subtracted. Alternatively, a vector of length equal to the number of columns of x can be supplied. The value is passed to scale.
ycenter	analogous to xcenter
xscale	a logical value indicating whether the columns of x should be scaled to have unit variance before the analysis takes place. The default is FALSE for consistency with cancor. Alternatively, a vector of length equal to the number of columns of x can be supplied. The value is passed to scale.
yscale	analogous to xscale
nvar	the number of canonical variables to be computed for each domain. With the default setting, canonical variables are computed until either x or y is fully deflated.
xpredict	the regression function to predict the canonical variable for x, given y. The formal arguments are the design matrix y, the regression target xc as the current canonical variable for x, and cc as a counter of the current pair of canonical variables (e.g. for enforcing different constraints for different canonical vectors). See the examples for an illustration.
ypredict	analogous to xpredict
cor_tol	a threshold indicating the magnitude below which canonical variables should be omitted. Variables are omitted if their explained correlations are less than or equal to cor_tol times the correlation of the first pair of canonical variables. With the default NULL setting, no variables are omitted.
nrestart	the number of random restarts for computing the canonical variables via iterated regression steps. The solution achieving maximum explained correlation over all random restarts is kept. A value greater than one can help to avoid poor local maxima.
iter_tol	If the relative change of the objective is less than iter_tol between iterations, the procedure is assumed to have converged to a local optimum.
iter_max	the maximum number of iterations to be performed. The procedure is terminated if either the iter_tol or the iter_max criterion is satisfied.
partial_model	NULL or an object of class nscancor. The computation can be continued from a partial model by providing an nscancor object (either from a previous run of this function or from acor) and setting nvar to a value greater than the number of canonical variables contained in the partial model. See the examples for an illustration.

verbosity	an integer specifying the verbosity level. Greater values result in more output, the default is to be quiet.

Details

nscancor computes the canonical vectors (called xcoef and ycoef) using iterated regression steps, where the constraints suitable for each domain are enforced by choosing the appropriate regression method. See Sigg et al. (2007) for an early application of the principle (not yet including generalized deflation).

Because constrained canonical vectors no longer correspond to true eigenvectors of the cross-covariance matrix and are usually not pairwise conjugate (i.e. the canonical variables are not uncorrelated), special attention needs to be paid when computing more than a single pair of canonical vectors. nscancor implements a generalized deflation (GD) scheme which builds on GD for PCA as proposed by Mackey (2009). For each domain, a basis of the space spanned by the previous canonical variables is computed. Then, the correlation of the current pair of canonical variables is maximized after projecting each current canonical vector to the ortho-complement space of its respective basis. This procedure maximizes the additional correlation not explained by previous canonical variables, and is identical to standard CCA if the canonical vectors are the eigenvectors of the cross-covariance matrix.

See the references for further details.

Value

A list of class nscancor containing the following elements:

cor	the additional correlation explained by each pair of canonical variables, see acor.
xcoef	the matrix containing the canonical vectors related to x as its columns
ycoef	analogous to xcoef
xcenter	if xcenter is TRUE the centering vector, else the zero vector (in accordance with cancor)
ycenter	analogous to xcenter
xscale	if xscale is TRUE the scaling vector, else FALSE
yscale	analogous to xscale
xp	the deflated data matrix corresponding to x
yp	analogous to xp

References

Sigg, C. and Fischer, B. and Ommer, B. and Roth, V. and Buhmann, J. (2007) Nonnegative CCA for Audiovisual Source Separation. In *Proceedings of the 2007 IEEE Workshop on Machine Learning for Signal Processing* (pp. 253–258).

Mackey, L. (2009) Deflation Methods for Sparse PCA. In *Advances in Neural Information Processing Systems* (pp. 1017–1024).

See Also

acor, cancor, scale

Examples

```
data(nutrimouse, package = "CCA")

###
# Unconstrained CCA, produces results close to calling
# cancor(nutrimouse$gene[ , 1:5], nutrimouse$lipid)

ypredict <- function(x, yc, cc) {
  return(MASS::ginv(x)%*%yc)
}
xpredict <- function(y, xc, cc) {
  return(MASS::ginv(y)%*%xc)
}
cc <- nscancor(nutrimouse$gene[ , 1:5], nutrimouse$lipid, xpredict = xpredict,
               ypredict = ypredict)
print(cc$cor)

###
# Non-negative sparse CCA using glmnet() as the regression function, where
# different regularizers are enforced on the different data domains and pairs
# of canonical variables.

dfmax_w <- c(40, 15, 10)
ypredict <- function(x, yc, cc) {
  en <- glmnet::glmnet(x, yc, alpha = 0.5, intercept = FALSE,
                       dfmax = dfmax_w[cc], lower.limits = 0)
  W <- coef(en)
  return(W[2:nrow(W), ncol(W)])
}
dfmax_v <- c(7, 5, 5)
xpredict <- function(y, xc, cc) {
  en <- glmnet::glmnet(y, xc, alpha = 0.5, intercept = FALSE,
                       dfmax = dfmax_v[cc])
  V <- coef(en)
  return(V[2:nrow(V), ncol(V)])
}
nscc <- nscancor(nutrimouse$gene, nutrimouse$lipid, nvar = 2,
                 xpredict = xpredict, ypredict = ypredict)

# continue the computation of canonical variables from a partial model
nscc <- nscancor(nutrimouse$gene, nutrimouse$lipid, nvar = 3,
                 xpredict = xpredict, ypredict = ypredict,
                 partial_model = nscc)
print(nscc$cor)
print(nscc$xcoef)
print(nscc$ycoef)
```

Index