Package ‘mbclusterwise’

November 22, 2016

Type Package

Title Clusterwise Multiblock Analyses

Version 1.0

Date 2016-11-22

Author Stephanie Bougeard

Maintainer Stephanie Bougeard <stephanie.bougeard@anses.fr>

Description Perform clusterwise multiblock analyses (clusterwise multiblock Partial Least Squares, clusterwise multiblock Redundancy Analysis or a regularized method between the two latter ones) associated with a F-fold cross-validation procedure to select the optimal number of clusters and dimensions.

License GPL (>= 2.0)

Depends ade4, doParallel, foreach, kknn, parallel

NeedsCompilation no

Repository CRAN

Date/Publication 2016-11-22 17:08:03

R topics documented:

mbclusterwise-package *Clusterwise Multiblock Analyses*

Description

Perform clusterwise multiblock analyses (clusterwise multiblock Partial Least Squares, clusterwise multiblock Redundancy Analysis or a regularized method between the two latter ones) associated with a F-fold cross-validation procedure to select the optimal number of clusters and dimensions.

Details

The DESCRIPTION file:

Package:	mbclusterwise
Type:	Package
Title:	Clusterwise Multiblock Analyses
Version:	1.0
Date:	2016-11-22
Author:	Stephanie Bougeard
Maintainer:	Stephanie Bougeard <stephanie.bougeard@anses.fr>
Description:	Perform clusterwise multiblock analyses (clusterwise multiblock Partial Least Squares, clusterwise multiblock
License:	GPL(>=2.0)
Depends:	ade4, doParallel, foreach, kknn, parallel

Index of help topics:

```
cw.multiblock           Clusterwise multiblock analyses
cw.predict              Prediction procedure for clusterwise multiblock
                        analyses
cw.tenfold              F-Fold cross-validation for clusterwise
                        multiblock analyses
mbclusterwise-package   Clusterwise Multiblock Analyses
mbpcaiv.fast            Multiblock principal component analysis with
                        instrumental variables (also called multiblock
                        Redundancy Analysis)
mbpls.fast              Multiblock partial least squares
mbregular               Regularized multiblock regression
simdata.red             Simulated toy data with two groups to test the
                        mbclusterwise package
```

Author(s)

Stephanie Bougeard
Maintainer: Stephanie Bougeard <stephanie.bougeard@anses.fr>

References

Bougeard, S., Abdi, H., Saporta, G., Niang, N., Submitted, Clusterwise analysis for multiblock component methods.

See Also

ade4

Examples

```
data(simdata.red)
Data.X <- simdata.red[c(1:10, 21:30), 1:10]
Data.Y <- simdata.red[c(1:10, 21:30), 11:13]
## Note that the options (INIT=2) and (parallel.level = "low") are chosen to quickly
## illustrate the function.
## For real data, instead choose (INIT=20) to avoid local optima and (parallel.level = "high")
## to improve the computing speed.
res.cw <- cw.multiblock(Y = Data.Y, X = Data.X, blo = c(5, 5), option = "none", G = 2, H = 1,
          INIT = 2, method = "mbpls", Gamma = NULL, parallel.level = "low")
```

---

cw.multiblock *Clusterwise multiblock analyses*

Description

Function to perform a clusterwise multiblock analyses (clusterwise multiblock Partial Least Squares, clusterwise multiblock Redundancy Analysis or clusterwise regularized multiblock regression) of several explanatory blocks $(X_1, \ldots, X_K)$ to explain a dependent dataset Y.

Usage

```
cw.multiblock(Y, X, blo, option = c("none", "uniform"), G, H, INIT = 20,
    method = c("mbpls", "mbpcaiv", "mbregular"), Gamma = NULL,
    parallel.level = c("high", "low"))
```

Arguments

Y	a matrix or data frame containing the dependent variable(s)
X	a matrix or data frame containing the explanatory variables
blo	a vector of the numbers of variables in each explanatory dataset
option	an option for the block weighting (by default, the first option is chosen): 'none' the block weight is equal to the block inertia 'uniform' the block weight is equal to $1/K$ for $(X_1, \ldots, X_K)$ and to 1 for X and Y
G	an integer giving the expected number of clusters
H	an integer giving the expected number of dimensions of the component-based model

INIT	an integer giving the number of initializations required for the clusterwise analysis (20 by default)
method	an option for the multiblock method to be applied (by default, the first option is chosen): 'mbpls' multiblock Partial Least Squares is applied 'mbpcaiv' multiblock Redundancy Analysis is applied 'mbregular' multiblock regularized regression is applied
Gamma	a numeric value of the regularization parameter for the multiblock regularized regression comprised between 0 and 1 (NULL by default). The value (Gamma=0) leads to multiblock Redundancy Analysis and (Gamma=1) to multiblock PLS
parallel.level	Level of parallel computing, i.e. initializations are carried out simultaneously (high by default) 'high' includes all the processing units of your computer 'low' includes only two processing units of your computer

Value

A list containing the following components is returned:

call	the matching call
error	a vector containing the value of the criterion to be minimized (overall prediction error) ; this error is performed on the centered and scaled data
beta.cr	a list of array that contain the intercept and the regression coefficients associated with the centered and scaled data for each of the G clusters
beta.raw	a list of array that contain the intercept and the regression coefficients associated with the raw data for each of the G clusters
hopt	the real number of dimensions of the component-based model (hopt is sometimes lower than the expected H)
Ypred.cr	a list of matrices that contain the predicted dependent values associated with the centered and scaled data for each of the G clusters
Ypred.raw	a list of matrices that contain the predicted dependent values associated with the raw data for each of the G clusters
cluster	a vector containing the observation assignation to the G expected clusters (when G>1 only)

Author(s)

Stephanie Bougeard (<stephanie.bougeard@anses.fr>)

References

Bougeard, S., Abdi, H., Saporta, G., Niang, N., Submitted, Clusterwise analysis for multiblock component methods.

See Also

cw.tenfold, cw.predict

Examples

```
data(simdata.red)
Data.X <- simdata.red[c(1:10, 21:30), 1:10]
Data.Y <- simdata.red[c(1:10, 21:30), 11:13]
## Note that the options (INIT=2) and (parallel.level = "low") are chosen to quickly
## illustrate the function.
## For real data, instead choose (INIT=20) to avoid local optima and (parallel.level = "high")
## to improve the computing speed.
res.cw <- cw.multiblock(Y = Data.Y, X = Data.X, blo = c(5, 5), option = "none", G = 2,
          H = 1, INIT = 2, method = "mbpcaiv", Gamma = NULL, parallel.level = "low")
```

`cw.predict` *Prediction procedure for clusterwise multiblock analyses*

Description

Function to perform the prediction of new observations by means of clusterwise multiblock analysis

Usage

```
cw.predict(Xnew, res.cw)
```

Arguments

`Xnew`	a data frame containing new observation values for the explanatory variables
`res.cw`	a list of results created by the function `cw.multiblock`

Value

A list containing the following components is returned:

`clusternew`	a vector containing the new observation assignation to the G expected clusters (when G>1 only)
`Ypred.cr`	a matrix that contain the predicted dependent values associated with the centered and scaled data for each of the G clusters
`Ypred.raw`	a matrix that contain the predicted dependent values associated with the raw data for each of the G clusters

Author(s)

Stephanie Bougeard (`<stephanie.bougeard@anses.fr>`)

References

Bougeard, S., Abdi, H., Saporta, G., Niang, N., Submitted, Clusterwise analysis for multiblock component methods.

See Also

cw.multiblock, cw.tenfold

Examples

```
 data(simdata.red)
 Data.X      <- simdata.red[c(1:10, 21:30), 1:10]
 Data.Y      <- simdata.red[c(1:10, 21:30), 11:13]
 Data.X.test <- simdata.red[c(16:20, 36:40), 1:10]
 ## Note that the options (INIT=2) and (parallel.level = "low") are chosen to quickly
 ## illustrate the function.
## For real data, instead choose (INIT=20) to avoid local optima and (parallel.level = "high")
 ## to improve the computing speed.
res.cw     <- cw.multiblock(Y = Data.Y, X = Data.X, blo = c(5, 5), option = "none", G = 2,
                H = 1, INIT = 2, method = "mbpls", Gamma = NULL, parallel.level = "low")
 rescw.pred  <- cw.predict(Data.X.test, res.cw)
```

---

cw.tenfold *F-Fold cross-validation for clusterwise multiblock analyses*

Description

Function to perform a F-fold cross-validation applied to clusterwise multiblock analyses. This function is usually applied to various numbers of clusters and of dimensions to select their optimal values.

Usage

```
cw.tenfold(Y, X, blo, option = c("none", "uniform"), G, H, FOLD = 10, INIT = 20,
   method = c("mbpls", "mbpcaiv", "mbregular"), Gamma = NULL,
   parallel.level = c("high", "low"))
```

Arguments

Y	a matrix or data frame containing the dependent variable(s)
X	a matrix or data frame containing the explanatory variables
blo	vector of the numbers of variables in each explanatory dataset
option	an option for the block weighting (by default, the first option is chosen): 'none' the block weight is equal to the block inertia 'uniform' the block weight is equal to $1/K$ for $(X_1, \ldots, X_K)$ and to 1 for X and Y
G	an integer giving the number of clusters
H	an integer giving the number of dimensions of the component-based model
FOLD	an integer giving the number of folds of the F-Fold cross-validation procedure comprised between 2 and 10 (10 by default)

INIT	an integer giving the number of initializations required for the clusterwise analysis (20 by default)
method	an option for the multiblock method to be applied (by default, the first option is chosen): 'mbpls' multiblock Partial Least Squares is applied 'mbpcaiv' multiblock Redundancy Analysis is applied 'mbregular' multiblock regularized regression is applied
Gamma	a numeric value of the regularization parameter for the multiblock regularized regression comprised between 0 and 1 (NULL by default). The value (Gamma=0) leads to multiblock Redundancy Analysis and (Gamma=1) to multiblock PLS
parallel.level	Level of parallel computing, i.e. initializations are carried out simultaneously (high by default) 'high' includes all the processing units of your computer 'low' includes only two processing units of your computer

Value

A list containing the following components is returned:

call	the matching call
sqrmse.cal	the squared Root Mean Squared Error from the F calibration datasets
sqrmse.val	the squared Root Mean Squared Error from the F prediction datasets

Author(s)

Stephanie Bougeard (<stephanie.bougeard@anses.fr>)

References

Bougeard, S., Abdi, H., Saporta, G., Niang, N., Submitted, Clusterwise analysis for multiblock component methods.

See Also

cw.multiblock, cw.predict

Examples

```
data(simdata.red)
Data.X <- simdata.red[c(1:8, 21:28), 1:10]
Data.Y <- simdata.red[c(1:8, 21:28), 11:13]
res1   <- list()
res2   <- list()

## Note that the options (INIT=2) and (parallel.level = "low") are chosen to quickly
## illustrate the function.
## For real data, instead choose (INIT=20) to avoid local optima and (parallel.level = "high")
## to improve the computing speed.
```

```
for (H in c(1:2)){
  print(paste("H=", H, sep=""))
 res1[[H]] <- cw.tenfold(Y = Data.Y, X = Data.X, blo = c(5, 5), option = "none", G = 1, H,
    FOLD = 2, INIT = 2, method = "mbpls", Gamma = NULL, parallel.level = "low")
 res2[[H]] <- cw.tenfold(Y = Data.Y, X = Data.X, blo = c(5, 5), option = "none", G = 2, H,
     FOLD = 2, INIT = 2, method = "mbpls", Gamma = NULL, parallel.level = "low")
}
res1.cal <- unlist(lapply(1:2, function(x) mean(sqrt(res1[[x]]$sqrmse.cal), na.rm=TRUE)))
res1.val <- unlist(lapply(1:2, function(x) mean(sqrt(res1[[x]]$sqrmse.val), na.rm=TRUE)))
res2.cal <- unlist(lapply(1:2, function(x) mean(sqrt(res2[[x]]$sqrmse.cal), na.rm=TRUE)))
res2.val <- unlist(lapply(1:2, function(x) mean(sqrt(res2[[x]]$sqrmse.val), na.rm=TRUE)))

rmse.cal <- rbind(res1.cal, res2.cal)
rmse.val <- rbind(res1.val, res2.val)
rownames(rmse.cal) <- rownames(rmse.val) <- paste("G", 1:2, sep = "=")
colnames(rmse.cal) <- colnames(rmse.val) <- paste("H", 1:2, sep = "=")

par(mfrow=c(1,2))
matplot(t(rmse.cal), type = "o", ylab = "RMSE of calibration", xlab = "Model dimension (H)",
     main = "Calibration", col = c("steelblue", "darkorange"), pch = c(0, 5), lwd = c(3, 3))
     legend("center", inset = .05, legend = rownames(rmse.cal), pch = c(0, 5), lwd = c(3, 3),
       col = c("steelblue", "darkorange"), horiz = TRUE, title = "Cluster number (G)")
matplot(t(rmse.val), type = "o", ylab = "RMSE of prediction", xlab = "Model dimension (H)",
     main = "Prediction", col = c("steelblue", "darkorange"), pch = c(0, 5), lwd = c(3, 3))
     legend("center", inset = .05, legend = rownames(rmse.val), pch = c(0, 5), lwd = c(3, 3),
       col = c("steelblue", "darkorange"), horiz = TRUE, title = "Cluster number (G)")
```

`mbpcaiv.fast` *Multiblock principal component analysis with instrumental variables (also called multiblock Redundancy Analysis)*

Description

Function to perform a multiblock Redundancy Analysis of several explanatory blocks $(X_1, \ldots, X_K)$, defined as an object of class `ktab` (from `ade4`), to explain a dependent dataset Y, defined as an object of class `dudi` (from `ade4`). This function is based on the same code and gives the same results as the `mbpcaiv` function from the `ade4` package with additional ones developed for the clusterwise procedure.

Usage

```
mbpcaiv.fast(dudiY, ktabX, scale = FALSE, option = c("none", "uniform"), H)
```

Arguments

`dudiY`	an object of class `dudi` (from `ade4`) containing the dependent variable(s)
`ktabX`	an object of class `ktab` (from `ade4`) containing the blocks of explanatory variables

scale	a logical value indicating whether the explanatory variables should be standardized
option	an option for the block weighting (by default, the first option is chosen): 'none' the block weight is equal to the block inertia 'uniform' the block weight is equal to $1/K$ for $(X_1, \ldots, X_K)$ and to 1 for X and Y
H	an integer giving the number of dimensions

Value

A list containing the following components is returned:

crit.reg	the regression error
lX	a matrix of the global components associated with the whole explanatory dataset (scores of the individuals)
XYcoef	a list of matrices of the regression coefficients of the whole explanatory dataset onto the dependent dataset
intercept	a list of matrices of the regression intercepts of the whole explanatory dataset onto the dependent dataset
fitted	a list of matrices which contain the predicted dependent values

Author(s)

Stephanie Bougeard (<stephanie.bougeard@anses.fr>)

References

Bougeard, S., Qannari, E.M., Lupo, C. and Hanafi, M. (2011). From multiblock partial least squares to multiblock redundancy analysis. A continuum approach. Informatica, 22(1), 11-26

See Also

cw.multiblock, cw.tenfold, cw.predict, mbpcaiv

Examples

```
data(simdata.red)
Data.X <- simdata.red[c(1:15, 21:35), 1:10]
Data.Y <- simdata.red[c(1:15, 21:35), 11:13]
library(ade4)
dudiy  <- dudi.pca(df = Data.Y, center = FALSE, scale = FALSE, scannf = FALSE)
ktabx  <- ktab.data.frame(df = data.frame(Data.X), blocks = c(5,5),
  tabnames = paste("Tab", c(1:2), sep = "."))
res    <- mbpcaiv.fast(dudiy, ktabx, scale = FALSE, option = "none", H = 2)
```

mbpls.fast *Multiblock partial least squares*

Description

Function to perform a multiblock Partial Least Squares (PLS) of several explanatory blocks $(X_1, \ldots, X_K)$ defined as an object of class ktab (from ade4), to explain a dependent dataset Y defined as an object of class dudi (from ade4). This function is based on the same code and gives the same results as the mbpls function from the ade4 package with additional ones developed for the clusterwise procedure.

Usage

```
mbpls.fast(dudiY, ktabX, scale = FALSE, option = c("none", "uniform"), H)
```

Arguments

dudiY	an object of class dudi (from ade4) containing the dependent variable(s)
ktabX	an object of class ktab (from ade4) containing the blocks of explanatory variables
scale	a logical value indicating whether the explanatory variables should be standardized
option	an option for the block weighting (by default, the first option is chosen): 'none' the block weight is equal to the block inertia 'uniform' the block weight is equal to $1/K$ for $(X_1, \ldots, X_K)$ and to 1 for X and Y
H	an integer giving the number of dimensions

Value

A list containing the following components is returned:

crit.reg	the regression error
lX	a matrix of the global components associated with the whole explanatory dataset (scores of the individuals)
XYcoef	a list of matrices of the regression coefficients of the whole explanatory dataset onto the dependent dataset
intercept	a list of matrices of the regression intercepts of the whole explanatory dataset onto the dependent dataset
fitted	a list of matrices which contain the predicted dependent values

Author(s)

Stephanie Bougeard (<stephanie.bougeard@anses.fr>)

References

Bougeard, S., Qannari, E.M., Lupo, C. and Hanafi, M. (2011). From multiblock partial least squares to multiblock redundancy analysis. A continuum approach. Informatica, 22(1), 11-26

See Also

`cw.multiblock`, `cw.tenfold`, `cw.predict`, `mbpls`

Examples

```
data(simdata.red)
Data.X <- simdata.red[c(1:15, 21:35), 1:10]
Data.Y <- simdata.red[c(1:15, 21:35), 11:13]
library(ade4)
dudiy  <- dudi.pca(df = Data.Y, center = FALSE, scale = FALSE, scannf = FALSE)
ktabx  <- ktab.data.frame(df = data.frame(Data.X), blocks = c(5,5),
  tabnames = paste("Tab", c(1:2), sep = "."))
res    <- mbpls.fast(dudiy, ktabx, scale = FALSE, option = "none", H = 2)
```

`mbregular` *Regularized multiblock regression*

Description

Function to perform the regularized multiblock regression which gives results comprised the ones from multiblock Redundancy Analysis (`gamma=0`) and multiblock PLS (`gamma=1`). This method is applied to several explanatory blocks $(X_1, \ldots, X_K)$ defined as an object of class `ktab` (from `ade4`), to explain a dependent dataset Y defined as an object of class `dudi` (from `ade4`).

Usage

```
mbregular(dudiY, ktabX, scale = FALSE, option = c("none", "uniform"), H, gamma)
```

Arguments

`dudiY`	an object of class `dudi` (from `ade4`) containing the dependent variable(s)
`ktabX`	an object of class `ktab` (from `ade4`) containing the blocks of explanatory variables
`scale`	a logical value indicating whether the explanatory variables should be standardized
`option`	an option for the block weighting (by default, the first option is chosen): '`none`' the block weight is equal to the block inertia '`uniform`' the block weight is equal to $1/K$ for $(X_1, \ldots, X_K)$ and to 1 for X and Y
`H`	an integer giving the number of dimensions
`gamma`	a numeric value of the regularization parameter comprised between 0 and 1. The value (`gamma=0`) leads to multiblock Redundancy Analysis and (`gamma=1`) to multiblock PLS

Value

A list containing the following components is returned:

- `crit.reg` — the regression error
- `lX` — a matrix of the global components associated with the whole explanatory dataset (scores of the individuals)
- `XYcoef` — a list of matrices of the regression coefficients of the whole explanatory dataset onto the dependent dataset
- `intercept` — a list of matrices of the regression intercepts of the whole explanatory dataset onto the dependent dataset
- `fitted` — a list of matrices which contain the predicted dependent values

Author(s)

Stephanie Bougeard (<stephanie.bougeard@anses.fr>)

References

Bougeard, S., Qannari, E.M., Lupo, C. and Hanafi, M. (2011). From multiblock partial least squares to multiblock redundancy analysis. A continuum approach. Informatica, 22(1), 11-26

See Also

cw.multiblock, cw.tenfold, cw.predict, mbpcaiv, mbpls

Examples

```
data(simdata.red)
Data.X <- simdata.red[c(1:15, 21:35), 1:10]
Data.Y <- simdata.red[c(1:15, 21:35), 11:13]
library(ade4)
dudiy  <- dudi.pca(df = Data.Y, center = FALSE, scale = FALSE, scannf = FALSE)
ktabx  <- ktab.data.frame(df = data.frame(Data.X), blocks = c(5,5),
 tabnames = paste("Tab", c(1:2), sep = "."))
res    <- mbregular(dudiy, ktabx, scale = FALSE, option = "none", H = 2, gamma = 0.8)
```

---

simdata.red — *Simulated toy data with two groups to test the mbclusterwise package*

Description

This data frame is a toy example with a limited number of observations extracted from the data `simdata` given in the `plspm` package. These are simulated data organized in two clusters showing two different local regression models.

Usage

```
data(simdata.red)
```

Format

A data frame of simulated data with 40 observations on the following 14 variables.

`mv1` first variable of the block Price Fairness (X_1)

`mv2` second variable of the block Price Fairness (X_1)

`mv3` third variable of the block Price Fairness (X_1)

`mv4` fourth variable of the block Price Fairness (X_1)

`mv5` fifth variable of the block Price Fairness (X_1)

`mv6` first variable of the block Quality (X_2)

`mv7` second variable of the block Quality (X_2)

`mv8` third variable of the block Quality (X_2)

`mv9` fourth variable of the block Quality (X_2)

`mv10` fifth variable of the block Quality (X_2)

`mv11` first variable of the block Customer Satisfaction (Y)

`mv12` second variable of the block Customer Satisfaction (Y)

`mv13` third variable of the block Customer Satisfaction (Y)

References

Esposito Vinzi, V., Ringle, C., Squillacciotti, S. and Trinchera, L. (2007) Capturing and treating unobserved heterogeneity by response based segmentation in PLS path modeling. A comparison of alternative methods by computational experiments. Working paper, ESSEC Business School.

Examples

```
data(simdata.red)
simdata.red
Data.X <- simdata.red[c(1:15, 21:35), 1:10]
Data.Y <- simdata.red[c(1:15, 21:35), 11:13]
```

Index