Package 'factas'

July 2, 2014

Type Package

Title Data Mining Methods for Data Streams

Version 2.3

Date 2014-23-01

Author Romain BAR <romain.bar@univ-lorraine.fr>

Maintainer Romain BAR <romain.bar@univ-lorraine.fr>

Description The package contains methods using stochastic approximation to analyse (big) data (streams).

Depends R (>= 2.14.0), plotrix, FactoMineR, Matrix

License GPL

Encoding UTF-8

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-03-31 15:37:37

R topics documented:

`CA` *function performing a Correspondence Analysis on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Correspondence Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meaningful 2D visualizations: the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
CA(data, groups, stream=TRUE, nb_fact, principal_factors=TRUE, principal_axes=FALSE,
eigenvalues=FALSE, corr=FALSE, graphics=FALSE, data_init, exec_time, print_step)
```

Arguments

`data`	matrix : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (`stream=TRUE`), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
`groups`	vector (of dimension 2) : each component corresponds to the number (>1) of categories for each nominative variable.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
`corr`	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
`graphics`	boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted. N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE); if stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).
`data_init`	integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis. It must be at least equal to the number of columns of the dataset.
`exec_time`	real : execution time. If stream=TRUE, the analysis will stop when the first of these two events happens: all the data were taken into account or execution time achieved. Otherwise (stream=FALSE), it's the execution time of `CA_iter`.
`print_step`	integer: this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal canonical factors (or principal canonical axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Details

Reminding that CA is an analysis achieved on two groups of nominative variables only, we precise that the CA function computes the canonical elements (principal factors, principal axes, ...) associated with the first group of nominative variables only. If the user needs the canonical elements associated with the second group only, he may permute the two groups of variables before launching the analysis on that new dataset. If the user needs both canonical elements associated with the two groups, an option is to launch the MCA function (generalization of the CA function to more than two groups) on the dataset: indeed, MCA returns the general elements (principal factors, principal axes, ...) that are strongly linked to the canonical elements: for instance, if the two groups are respectively composed of p and q variables, the principal canonical factors associated with the first (resp. second) group returned by the CA function are of length p (resp. q) whereas the principal general factors returned by the MCA function are of length p+q. The principal canonical factors associated with the first (resp. second) group are colinears to the vectors composed of the p first (resp. q last) components of the principal general factors.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Université de Lorraine, IECL, Université de Lorraine, INRIA : BIGS group.

References

ACP projetée de données séquentielles, Monnez J-M., 42èmes Journées de Statistique (2010)

See Also

CA_iter, MCA, MCA_iter

Examples

```
data(tea)
the<-cbind(tea[,18],tea[,21])
CA(rbind(the,the,the),c(6,7),stream=TRUE,3,principal_factors=TRUE,
principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,graphics=TRUE,200,1,10)
CA(rbind(the,the,the),c(6,7),TRUE,3,TRUE,TRUE,TRUE,TRUE,TRUE,200,1,10)
```

CA_iter *performs a (iterative) Correpondence Analysis on a given data table.*

Description

This function permits user to perform fast Correspondence Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed. If `stream=TRUE`, `CA_iter` performs an initialisation of the function `CA` : considering `data_init` observations, the function `CA_iter` calculates a first estimation of the desired elements and returns them to the `CA` function. If `stream=FALSE`, `CA_iter` performs a Correspondence Analysis on the (entire) given matrix and returns the results to the `CA` function. Most of the arguments of `CA` are inherited from the `CA_iter` function.

Reminding that CA is an analysis achieved on two groups of nominative variables only, we precise that the `CA_iter` function computes the canonical elements (principal factors, principal axes, ...) associated with the first group of nominative variables only. If the user needs the canonical elements associated with the second group only, he may permute the two groups of variables before launching the analysis on that new dataset. If the user needs both canonical elements associated with the two groups, an option is to launch the `MCA_iter` function (generalization of the CA_iter function to more than two groups) on the dataset: indeed, `MCA_iter` returns the general elements (principal factors, principal axes, ...) that are strongly linked to the canonical elements: for instance, if the two groups are respectively composed of p and q variables, the principal canonical factors associated with the first (resp. second) group returned by the CA_iter function are of length p (resp. q) whereas the principal general factors returned by the MCA_iter function are of length p+q. The principal canonical factors associated with the first (resp. second) group are colinears to the vectors composed of the p first (resp. q last) components of the principal general factors.

Usage

```
CA_iter(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

`data`	matrix : if `stream=TRUE`, `CA_iter` computes a first estimation of the elements of interest on the matrix composed of the aggregation of the `data_init` first observations. if `stream=FALSE`, `data` is the entire given data table.
`groups`	vector (of dimension 2) : each component corresponds to the number (>1) of categories for each nominative variable.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.

corr	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
exec_time	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see CA)

Value

the value returned to the CA function is a list:

N	inverse of the covariance matrix of the first group
rbar	mean vector of the first group
sbar	mean vector of the second group
A	list composed of the nb_fact first principal canonical axes
F, F2, G, B, E_SR	useful matrices to define an estimation process
E_SS	(diagonal) covariance matrix of the second group stored as a vector
L1	list composed of the nb_fact first eigenvalues
X	list composed of the nb_fact first principal canonical factors
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

Bar Romain, Université de Lorraine, IECL, INRIA : BIGS group

References

ACP projetée de données séquentielles, Monnez J-M., 42èmes Journées de Statistique (2010)

See Also

CA, MCA, MCA_iter

CCA *function performing a Canonical Correlation Analysis on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Canonical Correlation Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meanningful 2D visualizations: the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
CCA(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,principal_axes=FALSE,
eigenvalues=FALSE,corr=FALSE,graphics=FALSE,data_init,exec_time,print_step)
```

Arguments

`data`	matrix : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (`stream=TRUE`), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
`groups`	vector : each component of `groups` gives the dimension/length of the associated group of variables.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
`corr`	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
`graphics`	boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted.

N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). if stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).

`data_init` integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis. It must be at least equal to the number of columns of the dataset.

`exec_time` real : execution time. If stream=TRUE, the analysis will stop when the first of these two events happen: all the were taken into account or execution time achieved. Otherwise, it's the execution time of `CCA_iter`

`print_step` integer : this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal canonical factors (or principal canonical axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Details

Reminding that CCA is an analysis achieved on two groups of quantitative variables only, we precise that the `CCA` function computes the canonical elements (principal factors, principal axes, ...) associated with the first group of variables only. If the user needs the canonical elements associated with the second group only, he may permute the two groups of variables before launching the analysis on that new dataset. If the user needs both canonical elements associated with the two groups, an option is to launch the `GCCA` function (generalization of the CCA function to more than two groups) on the dataset : indeed, `GCCA` returns the general elements (principal factors, principal axes, ...) that are strongly linked to the canonical elements: for instance, if the two groups are respectively composed of p and q variables, the principal canonical factors associated with the first (resp. second) group returned by the CCA function are of length p (resp. q) whereas the principal general factors returned by the GCCA function are of length p+q. The principal canonical factors associated with the first (resp. second) group are colinears to the vectors composed of the p first (resp. q last) components of the principal general factors.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

Stochastic approximation of the factors of a generalized canonical correlation analysis, Monnez J-M., Statistics and Probability Letters 78, 14 (2008) 2210-2216

See Also

`CA_iter`, `GCCA`, `GCCA_iter`

Examples

```
data(psy)
CCA(psy,c(3,4),stream=TRUE,3,principal_factors=TRUE,principal_axes=TRUE,
eigenvalues=TRUE,corr=TRUE,graphics=TRUE,40,1,10)
```

`CCA_iter` *performs a (iterative) CCA on a given data table.*

Description

This function permits user to perform fast Canonical Correlation Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed. If `stream=TRUE`, `CCA_iter` performs an initialisation of the function `CCA` : considering `data_init` observations, the function `CCA_iter` calculates a first estimation of the desired elements and returns them to function `CCA`. If `stream=FALSE`, `CCA_iter` performs a Canonical Correlation Analysis on the (entire) given matrix and returns the results to function `CCA`. Most of the arguments of the `CCA` function are inherited from the `CCA_iter` function.

Reminding that CCA is an analysis achieved on two groups of variables only, we precise that the `CCA_iter` function computes the canonical elements (principal factors, principal axes, ...) associated with the first group of variables only. If the user needs the canonical elements associated with the second group only, he may permute the two groups of variables before launching the analysis on that new dataset. If the user needs both canonical elements associated with the two groups, an option is to launch the `GCCA_iter` function (generalization of the CCA_iter function to more than two groups) on the dataset: indeed, `GCCA_iter` returns the general elements (principal factors, principal axes, ...) that are strongly linked to the canonical elements: for instance, if the two groups are respectively composed of p and q variables, the principal canonical factors associated with the first (resp. second) group returned by the CCA_iter function are of length p (resp. q) whereas the principal general factors returned by the GCCA_iter function are of length p+q. The principal canonical factors associated with the first (resp. second) group are colinears to the vectors composed of the p first (resp. q last) components of the principal general factors.

Usage

```
CCA_iter(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

`data`	matrix : if `stream=TRUE`, `CCA_iter` computes a first estimation of the elements of interest on the matrix composed of the aggregation of the `data_init` first observations; if `stream=FALSE`, `data` is the entire given data table.

groups	vector : each component of groups gives the dimension/length of the associated group of variables.
stream	boolean: stream=TRUE if you consider data streams. stream=FALSE otherwise.
nb_fact	integer: number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
principal_factors	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.
principal_axes	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
eigenvalues	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
corr	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
exec_time	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see CCA)

Value

the value returned to the CCA function is a list:

A	list composed of the nb_fact first principal canonical axes
D	vector useful to define an estimation processus
rbar	mean vector of the first group
sbar	mean vector of the second group
F	matrix useful to define an estimation processus
M	covariance matrix of the first group
L1	list composed of the nb_fact first eigenvalues
X	list composed of the nb_fact first principal canonical factors
N	inverse of the covariance matrix of the first group
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

Bar Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

Stochastic approximation of the factors of a generalized canonical correlation analysis, Monnez J-M., Statistics and Probability Letters 78, 14 (2008) 2210-2216

See Also

CA, GCCA, GCCA_iter

CDA *function performing a Canonical Discriminant Analysis on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Canonical Discriminant Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meaningful 2D visualizations: the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
CDA(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,principal_axes=FALSE,
eigenvalues=FALSE,corr=FALSE,graphics=FALSE,data_init,exec_time,print_step)
```

Arguments

`data`	matrix : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (`stream=TRUE`), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
`groups`	vector (of dimension 2) : the first component (>1) corresponds to the dimension/length of the group of quantitative variables, the second component (>1) gives the number of categories for the nominative variable.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.

`corr` boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.

`graphics` boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted.

N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). If stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).

`data_init` integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis. It must be at least equal to the number of columns of the dataset.

`exec_time` real : execution time. If stream=TRUE, the analysis will stop when one of these two events happen: all the data have been taken into account or execution time achieved. Otherwise, it's the execution time of `CDA_iter`

`print_step` integer: this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal canonical factors (or principal canonical axes, ...) is computed but not necessarily printed : it will be printed only every `print_step` observations.

Details

Reminding that CDA is an analysis achieved on two groups of variables only (one quantitative and the other nominative or qualitative), we precise that the `CDA` function computes the canonical elements (principal factors, principal axes, ...) associated with the first group of (quantitative) variables only.

To compute the function, the group of quantitative variables must be the first group in the dataset and the values taken by the nominative variables must be exclusive categories. Indeed, a preprocessing is done on the nominative variables to obtain a codage composed of 1 and 0, essential to achieve the analysis.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Univeristé de Lorraine, IECL, INRIA : BIGS group.

References

ACP projetée de données séquentielles, Monnez J.-M. 42èmes Journées de Statistique (2010)

See Also

CDA_iter

Examples

```
data(wolves)
CDA(rbind(wolves,wolves,wolves),c(9,4),stream=TRUE,3,principal_factors=TRUE,
principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,graphics=TRUE,25,2,100)
```

CDA_iter *performs a (iterative) CDA on a given data table.*

Description

This function permits user to perform fast Canonical Discriminant Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed. If stream=TRUE, CDA_iter performs an initialisation of the function CDA : considering data_init observations, the function CDA_iter calculates a first estimation of the desired elements and returns them to function CDA. If stream=FALSE, CDA_iter performs a Canonical Discriminant Analysis on the (entire) given matrix and returns the results to function CDA. Most of the arguments of CDA are inherited from the CDA_iter function.

Reminding that CDA is an analysis achieved on two groups of variables only (one quantitative and the other nominative or qualitative), we precise that the CDA_iter function computes the canonical elements (principal factors, principal axes, ...) associated with the first group of (quantitative) variables only.

To compute the function, the group of quantitative variables must be the first group in the dataset and the values taken by the nominative variables must be exclusive categories. Indeed, a pre-process is done on the nominative variables to obtain a codage composed of 1 and 0, essential to achieve the analysis.

Usage

```
CDA_iter(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

data	matrix : if stream=TRUE, CDA_iter computes a first estimation of the elements of interest on the matrix composed of the aggregation of the data_init first observations; if stream=FALSE, data is the entire given data table.
groups	vector (of dimension 2) : the first component (>1) corresponds to the dimension/length of the group of quantitative variables, the second component (>1) gives the number of categories for the nominative variable.

stream	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
nb_fact	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
principal_factors	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.
principal_axes	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
eigenvalues	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
corr	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
exec_time	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see CDA)

Value

the value returned to the CDA function is a list:

rbar	mean vector of the first group
sbar	mean vector of the second group
A	list composed of the nb_fact first principal canonical axes
N_calc	covariance matrix of the first group
M_calc	inverse of the covariance matrix of the first group
J_calc, K_calc, A2_calc, B_calc, Cov_SR_calc, Cov_S_calc	useful matrices to define an estimation processus
D_calc	vector usefull to define an estimation processus
L1	list composed of the nb_fact first eigenvalues
X	list composed of the nb_fact first principal canonical factors
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

Bar Romain, Université de Lorraine, IECL, INRIA : BIGS group

References

ACP projetée de données séquentielles, Monnez J-M., 42èmes Journées de Statistique (2010)

See Also

CDA

decath *data frame with decathlon results*

Description

give results of athletes in decathlon

Usage

```
data(decath)
```

Format

A data frame with 41 rows and ten columns: the ten columns correspond to the performance of the athletes for the ten events of the decathlon.

`100m` a numeric giving the time to run 100m

`long jump` a numeric giving the length in long jump

`shot put` a numeric giving the length in shot put

`high jump` a numeric giving the high in high jump

`400m` a numeric giving the time to run 400m

`110m hurdles` a numeric giving the time to run 110m hurdles

`discus` a numeric giving the time to run 100m

`pole vault` a numeric giving the high in pole vault

`javeline` a numeric giving the length in javeline

`1500m` a numeric giving the time to run 1500m

factas *function usefull to perform fast data-mining, especially for high-dimensional and on-line data (Big Data).*

Description

Principal function of the package. It permits user to perform fast (factorial) analyses on (high dimensional) on-line data. Several kind of analyses are possible:

- (Normalized) Principal Component Analysis (PCA)
- Multiple Factor Analysis (MFA)
- Canonical Correlation Analysis (CCA)
- Generalized Canonical Correlation Analysis (GCCA)
- Canonical Discriminant Analysis (also termed Discriminant Factor Analysis or Multiple Discriminant Analysis)(CDA)

- Correpondence Analysis (CA)
- Multiple Correpondence Analysis (MCA)

Using stochastic processes, several estimation may be calculated depending on the user goals: principal factors, principal axes, eigenvalues and correlation coefficients between the original variables and the principal factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meaningful 2D visualizations: the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
factas(type, data, groups = NULL, stream = TRUE, nb_fact,
principal_factors = TRUE, principal_axes = FALSE, eigenvalues = FALSE,
corr = FALSE, graphics=FALSE, data_init, exec_time, print_step)
```

Arguments

`type`	character string : type of analysis to perform (PCA,MFA,CCA,GCCA,CDA, CA or MCA) (between quotes !)
`data`	data.frame : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (`stream=TRUE`), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
`groups`	vector : For all the analyses except PCA (for which the variables are always considered individually) and CA (for which we always consider 2 (nominative) variables), the function requires: the dimension/length of the groups of quantitative variables OR the number of categories for each nominative variable. For instance, if you consider 3 groups respectively composed of 5,7 and 2 quantitative variables, or 3 nominative variables owning respectively 5,7 and 2 categories, just type c(5,7,2). N.B. : In CCA, CDA and CA, only two groups of variables are considered. More precisely, in CCA, the two groups are composed of quantitative variables. If you have to consider more than 2 groups, just use GCCA (generalization of CCA to more than 2 groups). The vector "groups" will be of the form c(.,.). In CDA, the first group is composed of quantitative variables while the second one is only composed of 1 nominative variable (qualitative variable such that there is no meaningful ordering, or ranking, of the categories). The vector "groups" will be of the form c(.,1). At last, in CA, the two groups are composed of 1 nominative variable. There is no need to define "groups".
`stream`	boolean : stream=TRUE if you consider data stream; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal factors, nb_fact=3

`principal_factors`
: boolean : principal_factors=TRUE if you need the value of some first principal factors; principal_factors=FALSE otherwise.

`principal_axes`
: boolean : principal_axes=TRUE if you need the value of some first principal axes; principal_axes=FALSE otherwise.

`eigenvalues`
: boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.

`corr`
: boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal factors; corr=FALSE otherwise.

`data_init`
: integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis.

`exec_time`
: real : execution time. (if stream=TRUE, the analysis will stop when the first of these two events happen: all the data have been taken into account or execution time has been achieved.)

`graphics`
: boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted.

  N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). if stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).

`print_step`
: integer : this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal factors (or principal axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, IECN, Université de Lorraine, INRIA : BIGS group.

References

"Convergence d'un processus d'estimation stochastique en analyse factorielle" by Jean-Marie Monnez in "Publ.Inst.Statist.Univ.Paris", 1994, vol.38, n°1 p.37-55

"Approximation stochastique en analyse des données: Methodes et Applications" by Abdelhalim Bouamaine and Jean Marie Monnez, 2012, Broché, Editions universitaires europeennes.

More specific references are given in each function description file.

Examples

```
data(decath)
factas(type="PCA",data=decath,stream=TRUE,nb_fact=3,principal_factors=TRUE,
principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,graphics=FALSE,data_init=20,
exec_time=1,print_step=10)


data(vins_de_loire)
factas(type="MFA",rbind(vins_de_loire,vins_de_loire,vins_de_loire),c(5,3,10,9,2),
stream=TRUE,3,principal_factors=TRUE,principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,
30,graphics=FALSE,1,print_step=10)


data(psy)
factas(type="CCA",psy,c(3,4),stream=TRUE,nb_fact=3,principal_factors=TRUE,
principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,graphics=FALSE,data_init=20,
exec_time=1,print_step=10)
```

GCCA *function performing a Generalized Canonical Correlation Analysis on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Generalized Canonical Correlation Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meanningful 2D visualizations: the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Reminding that GCCA is an analysis achieved on q groups of variables (q > or = 2), we precise that the GCCA function computes the general elements (principal factors, principal axes, ...) . If the user needs the canonical elements associated with the k^th group (k=1,...,q), it's possible to deduce them from the general elements: the principal canonical factors associated with the k^th group are colinears to the vectors composed of the relevant components of the principal general factors.

Usage

```
GCCA(data,groups,stream=TRUE,nb_fact,principal_factors=FALSE,principal_axes=TRUE,
eigenvalues=FALSE,corr=FALSE,graphics=FALSE,data_init,exec_time,print_step)
```

Arguments

`data`	matrix : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (`stream=TRUE`), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
`groups`	vector : each component of `groups` gives the length of the associated group of variables.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal general factors, nb_fact=3
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal general factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal general axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
`corr`	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
`graphics`	boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted. N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). if stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).
`data_init`	integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis. It must be at least equal to the number of columns of the dataset.
`exec_time`	real : execution time. If stream=TRUE, the analysis will stop when one of these two events happen: all the data have been taken into account or execution time achieved. Otherwise, it's the execution time of `GCCA_iter`.
`print_step`	integer : this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal general factors (or principal canonical axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Details

GCCA is a generalization of CCA in the sense that it performs an analysis on more than 2 groups of quantitative variables.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

Stochastic approximation of the factors of a generalized canonical correlation analysis, Monnez J-M., Statistics and Probability Letters 78, 14 (2008) 2210-2216

See Also

GCCA_iter, CCA, CCA_iter

Examples

```
data(vins_de_loire)
GCCA(rbind(vins_de_loire,vins_de_loire,vins_de_loire),c(5,3,10,9,2),stream=TRUE,3,
principal_factors=TRUE,principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,30,graphics=FALSE,
1,print_step=10)
```

GCCA_iter *performs a (iterative) GCCA on a given data table.*

Description

This function permits user to perform fast Generalized Canonical Correlation Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal general factors, principal general axes, eigenvalues and correlation coefficients between the original variables and the principal general factors can be computed. If `stream=TRUE`, `GCCA_iter` performs an initialisation of the function GCCA : considering `data_init` observations, the function `GCCA_iter` calculates a first estimation of the desired elements and returns them to function `GCCA`. If `stream=FALSE`, `GCCA_iter` performs a Generalized Canonical Correlation Analysis on the (entire) given matrix and returns the results to function `GCCA`. Most of the arguments of `GCCA` are inherited from the `GCCA_iter` function.

Reminding that GCCA is an analysis achieved on q groups of variables (q > or = 2), we precise that the `GCCA_iter` function computes the general elements (principal factors, principal axes, ...) . If the user needs the canonical elements associated with the k^th group (k=1,. . . ,q), it's possible to

deduce them from the general elements: the principal canonical factors associated with the k^th group are colinears to the vectors composed of the relevant components of the principal general factors.

Usage

```
GCCA_iter(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

data	matrix : if stream=TRUE, GCCA_iter computes a first estimation of the elements of interest on the matrix composed of the aggregation of the data_init first observations; if stream=FALSE, data is the entire given data table.
groups	vector : each component of groups gives the length of the associated group of variables.
stream	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
nb_fact	integer : number of elements to be calculated. For instance, if you need the three first principal general factors, nb_fact=3
principal_factors	boolean : principal_factors=TRUE if you need the value of some first principal general factors; principal_factors=FALSE otherwise.
principal_axes	boolean : principal_axes=TRUE if you need the value of some first principal general axes; principal_axes=FALSE otherwise.
eigenvalues	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
corr	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal general factors; corr=FALSE otherwise.
exec_time	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see GCCA).

Value

the value returned to the GCCA function is a list:

A	list composed of the nb_fact first principal canonical axes
C	covariance matrix of the entire dataset
M	list composed of the inverses of the covariance matrices of the k^th group (k=1,. . . ,d)
M1	matrix : metric associated to the GCCA : blockwise diagonal matrix composed of the elements of M
L1	list composed of the nb_fact first eigenvalues
X	list composed of the nb_fact first principal canonical factors
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

Bar Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

Stochastic approximation of the factors of a generalized canonical correlation analysis, Monnez J-M., Statistics and Probability Letters 78, 14 (2008) 2210-2216

See Also

GCCA_iter, CCA, CCA_iter

MCA *function performing a Multiple Correspondence Analysis on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Multiple Correspondence Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meaningful 2D visualizations : the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
MCA(data,groups,stream=TRUE,nb_fact,principal_factors=FALSE,principal_axes=TRUE,
eigenvalues=FALSE,corr=FALSE,graphics=FALSE,data_init,exec_time,print_step)
```

Arguments

data	matrix : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (stream=TRUE), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
groups	vector :each component of groups gives the number (>1) of categories for each nominative variable.
stream	boolean : stream=TRUE if you consider data stream; stream=FALSE otherwise.
nb_fact	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.

`principal_factors`
: boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.

`principal_axes`
: boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.

`eigenvalues`
: boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.

`corr`
: boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.

`graphics`
: boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted.

  N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). If stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).

`data_init`
: integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis. It must be at least equal to the number of columns of the dataset.

`exec_time`
: real : execution time. If stream=TRUE, the analysis will stop when one of these two events happen: all the data have been taken into account or execution time achieved. Otherwise, it's the execution time of `MCA_iter`

`print_step`
: integer : this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal canonical factors (or principal canonical axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Details

MCA is a generalization of CA in the sense that it can perform the analysis on more than 2 groups of nominative variables.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

http://en.wikipedia.org/wiki/Multiple_correspondence_analysis

See Also

MCA_iter, CA, CA_iter

Examples

```
data(tea)
the<-cbind(tea[,18],tea[,21],tea[,22],tea[,23],tea[,24])
MCA(rbind(the,the,the),c(6,7,2,5,4),stream=TRUE,3,principal_factors=TRUE,
principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,graphics=TRUE,300,2,100)
```

MCA_iter *performs a (iterative) MCA on a given data table.*

Description

This function permits user to perform fast Multiple Correspondence Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal canonical factors, principal canonical axes, eigenvalues and correlation coefficients between the original variables and the principal canonical factors can be computed. If stream=TRUE, MCA_iter performs an initialisation of the function MCA : considering data_init observations, the function MCA_iter calculates a first estimation of the desired elements and returns them to function MCA. If stream=FALSE, MCA_iter performs a Multiple Correspondence Analysis on the (entire) given matrix and returns the results to function MCA. Most of the arguments of MCA are inherited from the MCA_iter function.

Usage

```
MCA_iter(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

data	matrix : if stream=TRUE, MCA_iter computes a first estimation of the elements of interest on the matrix composed of the aggregation of the data_init first observations; if stream=FALSE, data is the entire given data table.
groups	vector : each component of groups gives the number (>1) of categories for each nominative variable.
stream	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
nb_fact	integer : number of elements to be calculated. For instance, if you need the three first principal canonical factors, nb_fact=3.
principal_factors	boolean : principal_factors=TRUE if you need the value of some first principal canonical factors; principal_factors=FALSE otherwise.

principal_axes	boolean : principal_axes=TRUE if you need the value of some first principal canonical axes; principal_axes=FALSE otherwise.
eigenvalues	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
corr	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal canonical factors; corr=FALSE otherwise.
exec_time	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see CA).

Value

the value returned to the MCA function is a list:

rbar	mean vector of the data set
A	list composed of the nb_fact first principal canonical axes
C	covariance matrix of the entire dataset
M	list composed of the inverses of the covariance matrices of the k^th group (k=1,. . . ,d)
M1	metric associated to the MCA: blockwise diagonal matrix composed of the elements of M
L1	list composed of the nb_fact first eigenvalues
X	list composed of the nb_fact first principal canonical factors
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

Bar Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

http://en.wikipedia.org/wiki/Multiple_correspondence_analysis

See Also

MCA, CA, CA_iter

MFA *function performing a Multiple Factor Analysis in the sense of Escofier-Pages on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Multiple Factor Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal factors, principal axes, eigenvalues and correlation coefficients between the original variables and the principal factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meanningful 2D visualizations : the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
MFA(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,principal_axes=FALSE,
eigenvalues=FALSE,corr=FALSE,graphics=FALSE,data_init,exec_time,print_step)
```

Arguments

data	matrix : data to be analysed. The columns will represent your variables and the rows will represent your observations (sometimes called records, subjects or cases). If you consider on-line data (stream=TRUE), of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
groups	vector : each component of groups gives the dimension/length of the associated group of variables.
stream	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
nb_fact	integer : number of elements to be calculated. For instance, if you need the three first principal factors, nb_fact=3
principal_factors	boolean : principal_factors=TRUE if you need the value of some first principal factors; principal_factors=FALSE otherwise.
principal_axes	boolean : principal_axes=TRUE if you need the value of some first principal axes; principal_axes=FALSE otherwise.
eigenvalues	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
corr	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal factors; corr=FALSE otherwise.

`data_init` integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis. It must be at least equal to the number of columns of the dataset.

`exec_time` real : execution time. If stream=TRUE, the analysis will stop when one of these two events happen: all the data have been taken into account or execution time achieved. Otherwise, it's the execution time of `MFA_iter`.

`graphics` boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted.

N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). if stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).

`print_step` integer : this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal factors (or principal axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Details

MFA is assumed here to be performable only on groups composed of quantitative variables.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Université de Lorraine, IECL, INRIA: BIGS group.

References

Multiple Factor Analysis: Main Features and Application to Sensory Data, Jérôme Pagès, 2004. Approximation stochastique en analyse factorielle multiple, Jean-Marie Monnez, Publications de l'Institut de Statistique de l'Université de Paris L, 3 (2006) 27-45

See Also

`MFA_iter`

Examples

```
data(vins_de_loire)
MFA(rbind(vins_de_loire,vins_de_loire,vins_de_loire),c(5,3,10,9,2),stream=TRUE,3,
principal_factors=TRUE,principal_axes=TRUE,eigenvalues=TRUE,corr=TRUE,30,graphics=TRUE,
```

```
1,print_step=10)
```

`MFA_iter` *performs a (iterative) MFA on a given data table.*

Description

This function permits user to perform fast Multiple Factorial Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal factors, principal axes, eigenvalues and correlation coefficients between the original variables and the principal factors can be computed. If `stream=TRUE`, `MFA_iter` performs an initialisation of the function `MFA` : considering `data_init` observations, the function `MFA_iter` calculates a first estimation of the desired elements and returns them to function `MFA`. If `stream=FALSE`, `MFA_iter` performs a Multiple Factorial Analysis on the (entire) given matrix and returns the results to function `MFA`. Most of the arguments of `MFA` are inherited from the `MFA_iter` function.

Usage

```
MFA_iter(data,groups,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

`data`	matrix : if `stream=TRUE`, `MFA_iter` computes a first estimation of the elements of interest on the matrix composed of the aggregation of the `data_init` first observations; if `stream=FALSE`, `data` is the entire given data table.
`groups`	vector : each component of `groups` gives the dimension/length of the associated group.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
`corr`	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal factors; corr=FALSE otherwise.
`exec_time`	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see `MFA`).

Value

the value returned to the MFA function is a list:

X	list composed of the nb_fact first principal factors
Xk	list composed of the first principal factors of the PCA on the k^th group, useful to calculate the metric
Ck	list composed of the covariance matrices associated with the k^th group, useful to calculate the metric
Mk	list composed of the classical metric of the PCA on the k^th group (stored as a vector), usefull to calculate the MFA metric
L1	list composed of the nb_fact first eigenvalues
A	list composed of the nb_fact first principal axes
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

BAR Romain, IECL, Université de Lorraine, INRIA : BIGS group.

References

Multiple Factor Analysis : Main Features and Application to Sensory Data, Jérôme Pagès, 2004. Approximation stochastique en analyse factorielle multiple, Jean-Marie Monnez, Publications de l'Institut de Statistique de l'Université de Paris L, 3 (2006) 27-45

See Also

MFA

---

norm *computes the norm of a matrix in the Hilbert-Schmidt sense*

Description

computes the norm of a matrix in the Hilbert-Schmidt sense

Usage

```
norm(A)
```

Arguments

A	matrix

Value

norm of A

Author(s)

Romain Bar, Université de Lorraine, IECL, INRIA: BIGS group

References

http://mathworld.wolfram.com/Hilbert-SchmidtNorm.html

---

orth_Gram_Schmidt *Gram-Schmidt orthogonalization*

Description

this function orthogonalizes a family of vectors in the Gram-Schmidt sense : the metric is a matrix

Usage

```
orth_Gram_Schmidt(M, Y)
```

Arguments

M	matrix : the metric
Y	list : family of vectors to orthogonalize

Value

returns a list : orthogonal family of vectors built with the initial family of vectors (2nd argument : Y) thanks to the orthogonalization process of Gram-Schmidt

Author(s)

Romain Bar, Université de Lorraine, IECL, INRIA: BIGS group

See Also

orth_Gram_Schmidt_metrique_diag

orth_norm_Gram_Schmidt

orth_norm_Gram_Schmidt_metrique_diag

Examples

```
orth_Gram_Schmidt(diag(c(3,2)),list(c(1,2),c(2,3)))

stopifnot( orth_Gram_Schmidt(diag( c(1,2)),list( c(1,2),c(7,5) ) ) [[2]]== c(4,-1))
```

`orth_Gram_Schmidt_metrique_diag` *Gram-Schmidt orthogonalization when the metric is diagonal*

Description

this function orthogonalizes a family of vectors in the Gram-Schmidt sense : the metric is a diagonal matrix stored as a vector

Usage

```
orth_Gram_Schmidt_metrique_diag(M, Y)
```

Arguments

`M`	vector : diagonal of the metric
`Y`	list : family of vectors to orthogonalize

Value

returns a list : orthogonal family of vectors built with the initial family of vectors (2nd argument : Y) thanks to the orthogonalization process of Gram-Schmidt.

Author(s)

BAR Romain, Université de Lorraine, IECL, Université de Lorraine, INRIA : BIGS group.

See Also

`orth_Gram_Schmidt`, `orth_norm_Gram_Schmidt`, `orth_norm_Gram_Schmidt_metrique_diag`

Examples

```
orth_Gram_Schmidt_metrique_diag(c(3,2),list(c(1,2),c(2,3)))

stopifnot( orth_Gram_Schmidt_metrique_diag( c(1,2),list( c(1,2),c(7,5) ) ) [[2]]== c(4,-1))
```

`orth_norm_Gram_Schmidt` *Gram-Schmidt orthonormalization*

Description

this function orthonormalizes a family of vectors in the Gram-Schmidt sense : the metric is a matrix

Usage

```
orth_norm_Gram_Schmidt(M, Y)
```

Arguments

`M`	matrix : the metric
`Y`	list : family of vectors to orthonormalize

Value

returns a list : orthonormal family of vectors built with the initial family of vectors (2nd argument: Y) thanks to the orthonormalization process of Gram-Schmidt.

Author(s)

BAR Romain, Université de Lorraine, IECL, Université de Lorraine, INRIA : BIGS group.

See Also

`orth_Gram_Schmidt_metrique_diag`

`orth_Gram_Schmidt`

`orth_norm_Gram_Schmidt_metrique_diag`

Examples

```
orth_norm_Gram_Schmidt(diag(c(3,2)),list(c(1,2),c(2,3)))
```

`orth_norm_Gram_Schmidt_metrique_diag` *Gram-Schmidt orthonormalization when the metric is diagonal*

Description

this function orthonormalizes a family of vectors in the Gram-Schmidt sense : the metric is a diagonal matrix stored as a vector

Usage

```
orth_norm_Gram_Schmidt_metrique_diag(M, Y)
```

Arguments

`M`	vector : diagonal of the metric
`Y`	list : family of vectors to orthonormalize

Value

returns a list : orthonormal family of vectors built with the initial family of vectors (2nd argument: Y) thanks to the orthonormalization process of Gram-Schmidt.

Author(s)

BAR Romain, Université de Lorraine, IECL, Université de Lorraine, INRIA : BIGS group.

See Also

`orth_Gram_Schmidt_metrique_diag`, `orth_Gram_Schmidt`, `orth_norm_Gram_Schmidt`

Examples

```
orth_norm_Gram_Schmidt_metrique_diag(c(3,2),list(c(1,2),c(2,3)))
```

PCA *function performing a normalized principal components analysis on a given data table or on (simulated) data streams.*

Description

This function permits user to perform fast Principal Components Analysis on (high dimensional on-line) data.

Using stochastic processes, several estimation may be calculated depending on the user goals: principal factors, principal axes, eigenvalues and correlation coefficients between the original variables and the principal factors can be computed and updated recursively.

Moreover, graphics may be printed in order to give a better visualization and meaning/interpretation of the data. It's possible to get meaningful 2D visualizations: the observations are projected on the plans generated by direction vectors of the principal axes. Correlation circles that show a projection of the initial variables in the factors space are also available.

Usage

```
PCA(data,stream = TRUE,nb_fact,principal_factors=TRUE,principal_axes=FALSE,
 eigenvalues=FALSE,corr=FALSE,graphics=FALSE,data_init,exec_time,print_step)
```

Arguments

`data`	matrix : data to be analysed. The columns will represent the variables and the rows will represent the observations (sometimes called records, subjects or cases). If you consider on-line data (`stream=TRUE`),of course all the observations are not available at the beginning of the analysis. Thus, each observation (line of the matrix) will be treated individually (and then forgotten) to simulate the data stream.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
`corr`	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal factors; corr=FALSE otherwise.
`data_init`	integer : this argument is only relevant when stream=TRUE. It represents the number of data used to intialize the analysis.

`exec_time` real : execution time. If stream=TRUE, the analysis will stop when the first of these two events happen: all the data have been taken into account or execution time achieved. Otherwise, it's the execution time of `PCA_iter`

`graphics` boolean : if graphics=TRUE, 2D visualizations will be plotted, representing projections of the observations on all the plans generated by the `nb_fact` direction vectors of the principal axes. If corr=TRUE, correlation circles that show a projection of the initial variables in the factors spaces will also be plotted; if graphics=FALSE, no graphics will be plotted.

N.B. : For the fist kind of graphics, a gradation of grey was chosen to represent the "age" of the observations. The darker a point is, the newer the observation is. This coding is meaningful for a data stream (stream=TRUE). If stream=FALSE, the "age" concept is less relevant: in this configuration, the darker a point is, the lower the observation is in the matrix (database).

`print_step` integer : this argument is only relevant when stream=TRUE. Each time that a new observation is taken into account, a new estimation of the principal factors (or principal axes, ...) is computed but not necessarily printed: it will be printed only every `print_step` observations.

Value

results that user needs (principal factors, principal axes, eigenvalues and correlation coefficients)

Author(s)

BAR Romain, Université de Lorraine, IECL, INRIA : BIGS group.

References

Une méthode d'ACP de données en ligne, Jean-Marie Monnez, (2009) 153-156

Approximation stochastique en analyse factorielle multiple, Jean-Marie Monnez, Publications de l'Institut de Statistique de l'Université de Paris L, 3 (2006) 27-45

See Also

`PCA_iter`

Examples

```
data(decath)
PCA(decath,stream=TRUE,3,principal_factors=TRUE,principal_axes=TRUE,eigenvalues=TRUE,
corr=TRUE,graphics=TRUE,20,1,10)
```

PCA_iter *performs a normed (iterative) PCA on a given data table.*

Description

This function permits user to perform fast Principal Components Analysis on (high dimensional) data. Using stochastic processes, several estimation may be calculated depending on the user goals: principal factors, principal axes, eigenvalues and correlation coefficients between the original variables and the principal factors can be computed. If `stream=TRUE`, `PCA_iter` performs an initialisation of the function `PCA` : considering `data_init` observations, the function `PCA_iter` calculates a first estimation of the desired elements and returns them to function `PCA`. If `stream=FALSE`, `PCA_iter` performs a Principal Components Analysis on the (entire) given matrix and returns the results to function `PCA`. Most of the arguments of `PCA` are inherited from the `PCA_iter` function.

Usage

```
PCA_iter(data,stream=TRUE,nb_fact,principal_factors=TRUE,
principal_axes=FALSE,eigenvalues=FALSE,corr=FALSE,exec_time)
```

Arguments

`data`	matrix : if `stream`, `PCA_iter` computes a first estimation of the elements of interest on the matrix composed of the aggregation of the `data_init` first observations; if `stream=FALSE`, `data` is the entire given data table.
`stream`	boolean : stream=TRUE if you consider data streams; stream=FALSE otherwise.
`nb_fact`	integer : number of elements to be calculated. For instance, if you need the three first principal factors, nb_fact=3.
`principal_factors`	boolean : principal_factors=TRUE if you need the value of some first principal factors; principal_factors=FALSE otherwise.
`principal_axes`	boolean : principal_axes=TRUE if you need the value of some first principal axes; principal_axes=FALSE otherwise.
`eigenvalues`	boolean : eigenvalues=TRUE if you need the value of some first eigenvalues; eigenvalues=FALSE otherwise.
`corr`	boolean : corr=TRUE if you need the value of correlation coefficient between the original variables and some first principal factors; corr=FALSE otherwise.
`exec_time`	real : execution time. In this function, this parameter is only relevant if stream=FALSE (if stream=TRUE, see `PCA`).

Value

the value returned to the PCA function is a list:

`X`	list composed of the `nb_fact` first principal factors

L1	list composed of the nb_fact first eigenvalues
A	list composed of the nb_fact first principal axes
M1	vector : classical metric of the analysis stored as a vector
Corr	list composed of the nb_fact first correlation coefficient

Author(s)

BAR Romain, Université de Lorraine, IECL, INRIA: BIGS group.

See Also

PCA

---

preprocess_CA *preprocessing for Correpondence Analysis*

Description

performs an encoding of the original data set before launching CA

Usage

```
preprocess_CA(df)
```

Arguments

df	matrix : data set before encoding

Value

matrix : data set after encoding

Author(s)

Romain Bar, Université de Lorraine, IECL, INRIA : BIGS group

See Also

preprocess_CDA, preprocess_MCA

preprocess_CDA *preprocessing for Canonical Discriminant Analysis*

Description

performs an encoding of the original data set before launching CDA

Usage

```
preprocess_CDA(df)
```

Arguments

df	matrix : data set before encoding

Value

matrix : data set after encoding

Author(s)

Romain Bar, Universite de Lorraine, IECL, INRIA : BIGS group

See Also

preprocess_MCA, preprocess_CA

preprocess_MCA *preprocessing for Multiple Correpondence Analysis*

Description

performs an encoding of the original data set before launching MCA

Usage

```
preprocess_MCA(df)
```

Arguments

df	matrix : data set before encoding

Value

matrix : data set after encoding

Author(s)

Romain Bar, Université de Lorraine, IECL, INRIA : BIGS group

See Also

preprocess_CDA, preprocess_CA

---

psy *Psychological data*

Description

Investigation about the associations between psychological measures and academic achievement measures.

Usage

```
data(psy)
```

Format

A data frame with 600 observations (rows) on the following 7 quantitative variables (columns) divided in 2 groups: 3 psychological variables and 4 academic variables.

`locus_of_control` a numeric giving the "measurement" of locus of control

`self_concept` a numeric giving the "measurement" of self concept

`motivation` a numeric giving the "measurement" of motivation

`read` a numeric giving the result on a standardized test in reading

`write` a numeric giving the result on a standardized test in writing

`math` a numeric giving the result on a standardized test in mathematics

`science` a numeric giving the result on a standardized test in sciences

---

vins_de_loire *Wine from Loire*

Description

give sensory descriptions of 21 wines from Loire

Usage

```
data(vins_de_loire)
```

Format

A data frame with 21 rows (the number of wines) and 29 columns: quantitative variables corresponding to sensory descriptors.

`Odor.Intensity.before.shaking` a numeric (note between 0 and 5) giving the odor intensity of the wine before shaking

`Aroma.quality.before.shaking` a numeric (note between 0 and 5) giving the aroma quality of the wine before shaking

`Fruity.before.shaking` a numeric (note between 0 and 5) giving the fruity odor intensity of the wine before shaking

`Flower.before.shaking` a numeric (note between 0 and 5) giving the flower odor intensity of the wine before shaking

`Spice.before.shaking` a numeric (note between 0 and 5) giving the spicy odor intensity of the wine before shaking

`Visual.intensity` a numeric (note between 0 and 5) giving the visual intensity of the wine

`Nuance` a numeric (note between 0 and 5) giving the nuance quality of the wine

`Surface.feeling` a numeric (note between 0 and 5) giving the surface feeling of the wine

`Odor.Intensity` a numeric (note between 0 and 5) giving the odor intensity of the wine

`Quality.of.odour` a numeric (note between 0 and 5) giving the quality of odour of the wine

`Fruity` a numeric (note between 0 and 5) giving the fruity odor intensity of the wine

`Flower` a numeric (note between 0 and 5) giving the flower odor intensity of the wine

`Spice` a numeric (note between 0 and 5) giving the spicy odor intensity of the wine

`Plante` a numeric (note between 0 and 5) giving the plante odor intensity of the wine

`Phenolic` a numeric (note between 0 and 5) giving the phenolic odor intensity of the wine

`Aroma.intensity` a numeric (note between 0 and 5) giving the aroma intensity of the wine

`Aroma.persistency` a numeric (note between 0 and 5) giving the aroma persistency of the wine

`Aroma.quality` a numeric (note between 0 and 5) giving the aroma quality of the wine

`Attack.intensity` a numeric (note between 0 and 5) giving the attack intensity of the wine

`Acidity` a numeric (note between 0 and 5) giving the acidity intensity of the wine

`Astringency` a numeric (note between 0 and 5) giving the astringency intensity of the wine

`Alcohol` a numeric (note between 0 and 5) giving the alcohol intensity of the wine

`Balance` a numeric (note between 0 and 5) giving the balance of the wine

`Smooth` a numeric (note between 0 and 5) giving the smoothness of the wine

`Bitterness` a numeric (note between 0 and 5) giving the bitterness of the wine

`Intensity` a numeric (note between 0 and 5) giving the intensity of the wine

`Harmony` a numeric (note between 0 and 5) giving the harmony of the wine

`Overall.quality` a numeric (note between 0 and 5) giving the overall feeling about the wine

`Typical` a numeric (note between 0 and 5) giving the typicity of the wine

wolves *Study on skull wolves*

Description

Skull morphometric data on Rocky Mountain and Arctic wolves (Canis Lupus L.) taken from Morrison (1990), originally from Jolicoeur (1959).

Usage

```
data(wolves)
```

Format

A data frame with 25 observations on the following 10 variables.

`V1` palatal length, a numeric vector

`V2` postpalatal length, a numeric vector

`V3` zygomatic width, a numeric vector

`V4` palatal width outside first upper molars, a numeric vector

`V5` palatal width inside second upper molars, a numeric vector

`V6` postglenoid foramina width, a numeric vector

`V7` interorbital width, a numeric vector

`V8` braincase width, a numeric vector

`V9` crown length, a numeric vector

`V10` a factor with levels ar:f ar:m rm:f rm:m, comprising the combinations of location and sex

Index