Package 'samurais'

July 23, 2025

Type Package

Title Statistical Models for the Unsupervised Segmentation of Time-Series ('SaMUraiS')

Version 0.1.0

Description Provides a variety of original and flexible user-friendly statistical latent variable models and unsupervised learning algorithms to segment and represent time-series data (univariate or multivariate), and more generally, longitudinal data, which include regime changes. 'samurais' is built upon the following packages, each of them is an autonomous time-series segmentation approach: Regression with Hidden Logistic Process ('RHLP'), Hidden Markov Model Regression ('HMMR'), Multivariate 'RHLP' ('MRHLP'), Multivariate 'HMMR' ('MHMMR'), Piece-Wise regression ('PWR'). For the advantages/differences of each of them, the user is referred to our mentioned paper references.

URL https://github.com/fchamroukhi/SaMUraiS

License GPL (>= 3)

Depends R (>= 2.10)

Imports methods, stats, MASS, Rcpp

Suggests knitr, rmarkdown

LinkingTo Rcpp, RcppArmadillo

Collate samurais-package.R RcppExports.R utils.R dynamicProg.R fitPWRFisher.R mkStochastic.R hmmProcess.R MData.R ParamHMMR.R ParamMHMMR.R ParamMRHLP.R ParamPWR.R ParamRHLP.R StatHMMR.R StatMHMMR.R StatMRHLP.R StatPWR.R StatRHLP.R ModelHMMR.R ModelMHMMR.R ModelMRHLP.R ModelPWR.R ModelRHLP.R emHMMR.R emMHMMR.R emMRHLP.R emRHLP.R selectHMMR.R selectMHMMR.R selectMRHLP.R selectRHLP.R data-multivrealdataset.R data-multivtoydataset.R data-univrealdataset.R data-univtoydataset.R

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation yes

Author Faicel Chamroukhi [aut] (ORCID:
<https://orcid.org/0000-0002-5894-3103>),
Marius Bartcus [aut],
Florian Lecocq [aut, cre]

Maintainer Florian Lecocq <florian.lecocq@outlook.com>

Repository CRAN

Date/Publication 2019-07-28 09:50:02 UTC

Contents

samurais-package . . . 3
emHMMR . . . 4
emMHMMR . . . 5
emMRHLP . . . 7
emRHLP . . . 8
fitPWRFisher . . . 9
hmmProcess . . . 10
MData-class . . . 11
mkStochastic . . . 11
ModelHMMR-class . . . 12
ModelMHMMR-class . . . 13
ModelMRHLP-class . . . 14
ModelPWR-class . . . 15
ModelRHLP-class . . . 16
multivrealdataset . . . 17
multivtoydataset . . . 18
ParamHMMR-class . . . 19
ParamMHMMR-class . . . 20
ParamMRHLP-class . . . 21
ParamPWR-class . . . 22
ParamRHLP-class . . . 23
selectHMMR . . . 24
selectMHMMR . . . 25
selectMRHLP . . . 26
selectRHLP . . . 27
StatHMMR-class . . . 28
StatMHMMR-class . . . 30
StatMRHLP-class . . . 31
StatPWR-class . . . 32
StatRHLP-class . . . 33
univrealdataset . . . 34
univtoydataset . . . 35

Index **36**

samurais-package *SaMUraiS: StAtistical Models for the UnsupeRvised segmentAtIon of time-Series*

Description

`samurais` is a toolbox including many original and flexible user-friendly statistical latent variable models and efficient unsupervised algorithms to segment and represent time-series data (univariate or multivariate), and more generally, longitudinal data, which include regime changes.

`samurais` contains the following time series segmentation models:

- RHLP;
- HMM/HMMR;
- PWR;
- MRHLP;
- MHMMR;

For the advantages/differences of each of them, the user is referred to our mentioned paper references.

To learn more about `samurais`, start with the vignettes: `browseVignettes(package = "samurais")`

Author(s)

Maintainer: Florian Lecocq <florian.lecocq@outlook.com>

Authors:

- Faicel Chamroukhi <faicel.chamroukhi@unicaen.fr> (0000-0002-5894-3103)
- Marius Bartcus <marius.bartcus@gmail.com>

References

Chamroukhi, F., and Hien D. Nguyen. 2019. *Model-Based Clustering and Classification of Functional Data.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. https://doi.org/10.1002/widm.1298.

Chamroukhi, F. 2015. *Statistical Learning of Latent Data Models for Complex Data Analysis.* Habilitation Thesis (HDR), Universite de Toulon. https://chamroukhi.com/Dossier/FChamroukhi-Habilitation.pdf.

Trabelsi, D., S. Mohammed, F. Chamroukhi, L. Oukhellou, and Y. Amirat. 2013. *An Unsupervised Approach for Automatic Activity Recognition Based on Hidden Markov Model Regression.* IEEE Transactions on Automation Science and Engineering 3 (10): 829–335. https://chamroukhi.com/papers/Chamroukhi-MHMMR-IeeeTase.pdf.

Chamroukhi, F., D. Trabelsi, S. Mohammed, L. Oukhellou, and Y. Amirat. 2013. *Joint Segmentation of Multivariate Time Series with Hidden Process Regression for Human Activity Recognition.* Neurocomputing 120: 633–44. https://chamroukhi.com/papers/chamroukhi_et_al_neucomp2013b.pdf.

Chamroukhi, F., A. Same, G. Govaert, and P. Aknin. 2010. *A Hidden Process Regression Model for Functional Data Description. Application to Curve Discrimination.* Neurocomputing 73 (7-9): 1210–21. https://chamroukhi.com/papers/chamroukhi_neucomp_2010.pdf.

Chamroukhi, F. 2010. *Hidden Process Regression for Curve Modeling, Classification and Tracking.* Ph.D. Thesis, Universite de Technologie de Compiegne. https://chamroukhi.com/papers/FChamroukhi-Thesis.pdf.

Chamroukhi, F., A. Same, G. Govaert, and P. Aknin. 2009. *Time Series Modeling by a Regression Approach Based on a Latent Process.* Neural Networks 22 (5-6): 593–602. https://chamroukhi.com/papers/Chamroukhi_Neural_Networks_2009.pdf.

See Also

Useful links:

- https://github.com/fchamroukhi/SaMUraiS

`emHMMR` — *emHMMR implemens the EM (Baum-Welch) algorithm to fit a HMMR model.*

Description

emHMMR implements the maximum-likelihood parameter estimation of the HMMR model by the Expectation-Maximization (EM) algorithm, known as Baum-Welch algorithm in the context of HMMs.

Usage

```
emHMMR(X, Y, K, p = 3, variance_type = c("heteroskedastic",
  "homoskedastic"), n_tries = 1, max_iter = 1500, threshold = 1e-06,
  verbose = FALSE)
```

Arguments

`X`	Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.
`Y`	Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.
`K`	The number of regimes/segments (HMMR components).
`p`	Optional. The order of the polynomial regression. By default, `p` is set at 3.
`variance_type`	Optional character indicating if the model is "homoskedastic" or "heteroskedastic" (i.e same variance or different variances for each of the K regmies). By default the model is "heteroskedastic".
`n_tries`	Optional. Number of runs of the EM algorithm. The solution providing the highest log-likelihood will be returned. If `n_tries` > 1, then for the first run, parameters are initialized by uniformly segmenting the data into K segments, and for the next runs, parameters are initialized by randomly segmenting the data into K contiguous segments.

max_iter	Optional. The maximum number of iterations for the EM algorithm.
threshold	Optional. A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the EM as stopping criteria.
verbose	Optional. A logical value indicating whether or not values of the log-likelihood should be printed during EM iterations.

Details

emHMMR function implements the EM algorithm for the HMMR model. This function starts with an initialization of the parameters done by the method `initParam` of the class ParamHMMR, then it alternates between the E-Step (method of the class StatHMMR) and the M-Step (method of the class ParamHMMR) until convergence (until the relative variation of log-likelihood between two steps of the EM algorithm is less than the `threshold` parameter).

Value

EM returns an object of class ModelHMMR.

See Also

ModelHMMR, ParamHMMR, StatHMMR

Examples

```
data(univtoydataset)
hmmr <- emHMMR(univtoydataset$x, univtoydataset$y, K = 5, p = 1, verbose = TRUE)

hmmr$summary()

hmmr$plot()
```

---

emMHMMR *emMHMMR implemens the EM (Baum-Welch) algorithm to fit a MHMMR model.*

Description

emMHMMR implements the maximum-likelihood parameter estimation of the MHMMR model by the Expectation-Maximization (EM) algorithm, known as Baum-Welch algorithm in the context of HMMs.

Usage

```
emMHMMR(X, Y, K, p = 3, variance_type = c("heteroskedastic",
  "homoskedastic"), n_tries = 1, max_iter = 1500, threshold = 1e-06,
  verbose = FALSE)
```

Arguments

X	Numeric vector of length m representing the covariates/inputs $x_1, \ldots, x_m$.
Y	Matrix of size (m, d) representing a d dimension time series observed at points $1, \ldots, m$.
K	The number of regimes (MHMMR components).
p	Optional. The order of the polynomial regression. By default, p is set at 3.
variance_type	Optional character indicating if the model is "homoskedastic" or "heteroskedastic". By default the model is "heteroskedastic".
n_tries	Optional. Number of runs of the EM algorithm. The solution providing the highest log-likelihood will be returned. If n_tries > 1, then for the first run, parameters are initialized by uniformly segmenting the data into K segments, and for the next runs, parameters are initialized by randomly segmenting the data into K contiguous segments.
max_iter	Optional. The maximum number of iterations for the EM algorithm.
threshold	Optional. A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the EM as stopping criteria.
verbose	Optional. A logical value indicating whether or not values of the log-likelihood should be printed during EM iterations.

Details

emMHMMR function implements the EM algorithm. This function starts with an initialization of the parameters done by the method `initParam` of the class ParamMHMMR, then it alternates between the E-Step (method of the class StatMHMMR) and the M-Step (method of the class ParamMHMMR) until convergence (until the relative variation of log-likelihood between two steps of the EM algorithm is less than the `threshold` parameter).

Value

EM returns an object of class ModelMHMMR.

See Also

ModelMHMMR, ParamMHMMR, StatMHMMR

Examples

```
data(multivtoydataset)

mhmmr <- emMHMMR(multivtoydataset$x, multivtoydataset[,c("y1", "y2", "y3")],
                 K = 5, p = 1, verbose = TRUE)

mhmmr$summary()

mhmmr$plot()
```

emMRHLP *emMRHLP implemens the EM algorithm to fit a MRHLP model.*

Description

emMRHLP implements the maximum-likelihood parameter estimation of the MRHLP model by the Expectation-Maximization (EM) algorithm.

Usage

```
emMRHLP(X, Y, K, p = 3, q = 1, variance_type = c("heteroskedastic",
  "homoskedastic"), n_tries = 1, max_iter = 1500, threshold = 1e-06,
  verbose = FALSE, verbose_IRLS = FALSE)
```

Arguments

X	Numeric vector of length m representing the covariates/inputs $x_1, \ldots, x_m$.
Y	Matrix of size (m, d) representing a d dimension function of X observed at points $1, \ldots, m$. Y is the observed response/output.
K	The number of regimes (MRHLP components).
p	Optional. The order of the polynomial regression. By default, p is set at 3.
q	Optional. The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1 (which is the default value).
variance_type	Optional character indicating if the model is "homoskedastic" or "heteroskedastic". By default the model is "heteroskedastic".
n_tries	Optional. Number of runs of the EM algorithm. The solution providing the highest log-likelihood will be returned. If n_tries > 1, then for the first run, parameters are initialized by uniformly segmenting the data into K segments, and for the next runs, parameters are initialized by randomly segmenting the data into K contiguous segments.
max_iter	Optional. The maximum number of iterations for the EM algorithm.
threshold	Optional. A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the EM as stopping criteria.
verbose	Optional. A logical value indicating whether or not values of the log-likelihood should be printed during EM iterations.
verbose_IRLS	Optional. A logical value indicating whether or not values of the criterion optimized by IRLS should be printed at each step of the EM algorithm.

Details

emMRHLP function implements the EM algorithm of the MRHLP model. This function starts with an initialization of the parameters done by the method initParam of the class ParamMRHLP, then it alternates between the E-Step (method of the class StatMRHLP) and the M-Step (method of the class ParamMRHLP) until convergence (until the relative variation of log-likelihood between two steps of the EM algorithm is less than the threshold parameter).

Value

EM returns an object of class ModelMRHLP.

See Also

ModelMRHLP, ParamMRHLP, StatMRHLP

Examples

```
data(multivtoydataset)

mrhlp <- emMRHLP(multivtoydataset$x, multivtoydataset[,c("y1", "y2", "y3")],
                 K = 5, p = 1, verbose = TRUE)

mrhlp$summary()

mrhlp$plot()
```

---

emRHLP *emRHLP implements the EM algorithm to fit a RHLP model.*

Description

emRHLP implements the maximum-likelihood parameter estimation of the RHLP model by the Expectation-Maximization (EM) algorithm.

Usage

```
emRHLP(X, Y, K, p = 3, q = 1, variance_type = c("heteroskedastic",
  "homoskedastic"), n_tries = 1, max_iter = 1500, threshold = 1e-06,
  verbose = FALSE, verbose_IRLS = FALSE)
```

Arguments

X	Numeric vector of length m representing the covariates/inputs $x_1, \ldots, x_m$.
Y	Numeric vector of length m representing the observed response/output $y_1, \ldots, y_m$.
K	The number of regimes (RHLP components).
p	Optional. The order of the polynomial regression. By default, p is set at 3.
q	Optional. The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1 (which is the default value).
variance_type	Optional character indicating if the model is "homoskedastic" or "heteroskedastic". By default the model is "heteroskedastic".
n_tries	Optional. Number of runs of the EM algorithm. The solution providing the highest log-likelihood will be returned. If n_tries > 1, then for the first run, parameters are initialized by uniformly segmenting the data into K segments, and for the next runs, parameters are initialized by randomly segmenting the data into K contiguous segments.

max_iter	Optional. The maximum number of iterations for the EM algorithm.
threshold	Optional. A numeric value specifying the threshold for the relative difference of log-likelihood between two steps of the EM as stopping criteria.
verbose	Optional. A logical value indicating whether or not values of the log-likelihood should be printed during EM iterations.
verbose_IRLS	Optional. A logical value indicating whether or not values of the criterion optimized by IRLS should be printed at each step of the EM algorithm.

Details

emRHLP function implements the EM algorithm for the RHLP model. This function starts with an initialization of the parameters done by the method `initParam` of the class ParamRHLP, then it alternates between the E-Step (method of the class StatRHLP) and the M-Step (method of the class ParamRHLP) until convergence (until the relative variation of log-likelihood between two steps of the EM algorithm is less than the `threshold` parameter).

Value

EM returns an object of class ModelRHLP.

See Also

ModelRHLP, ParamRHLP, StatRHLP

Examples

```
data(univtoydataset)

rhlp <- emRHLP(univtoydataset$x, univtoydataset$y, K = 3, p = 1, verbose = TRUE)

rhlp$summary()

rhlp$plot()
```

---

fitPWRFisher *fitPWRFisher implements an optimized dynamic programming algorithm to fit a PWR model.*

Description

fitPWRFisher is used to fit a Piecewise Regression (PWR) model by maximum-likelihood via an optimized dynamic programming algorithm. The estimation performed by the dynamic programming algorithm provides an optimal segmentation of the time series.

Usage

```
fitPWRFisher(X, Y, K, p = 3)
```

Arguments

X	Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.
Y	Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.
K	The number of regimes/segments (PWR components).
p	Optional. The order of the polynomial regression. By default, p is set at 3.

Details

fitPWRFisher function implements an optimized dynamic programming algorithm of the PWR model. This function starts with the calculation of the "cost matrix" then it estimates the transition points given K the number of regimes thanks to the method computeDynamicProgram (method of the class ParamPWR).

Value

fitPWRFisher returns an object of class ModelPWR.

See Also

ModelPWR, ParamPWR, StatPWR

Examples

```
data(univtoydataset)

pwr <- fitPWRFisher(univtoydataset$x, univtoydataset$y, K = 5, p = 1)

pwr$summary()

pwr$plot()
```

---

hmmProcess *hmmProcess calculates the probability distribution of a random process following a Markov chain*

Description

hmmProcess calculates the probability distribution of a random process following a Markov chain

Usage

```
hmmProcess(prior, trans_mat, n)
```

Arguments

`prior`	Numeric vector or a one row matrix of length K representing the prior probabilities of the Markov chain.
`trans_mat`	Matrix of size (K, K) representing the transition matrix of the Markov chain.
`n`	Numeric. Number of variables of the Markov chain.

Details

hmmProcess calculates the distribution $P(Z_1, \ldots, Z_n; \pi, A)$ of a Markov chain $(Z_1, \ldots, Z_n)$ with prior probability π and transition matrix A.

The calculation is based on the following formula:

$P(Z_i = k) = \sum_l P(Z_i = k, Z_{i-1} = l) = \sum_l P(Z_i = k | Z_{i-1} = l) \times P(Z_{i-1} = l) = \sum_l A_{lk} \times P(Z_{i-1})$

Value

Matrix of size (n, K) giving the distribution of process given the K-state Markov chain parameters.

`MData-class` *A Reference Class which represents multivariate data.*

Description

MData is a reference class which represents multivariate objects. The data can be ordered by time (multivariate time series). In the last case, the field `X` represents the time.

Fields

`X` Numeric vector of length *m*.

`Y` Matrix of size (m, d) representing a d dimension function of `X` observed at points $1, \ldots, m$.

`mkStochastic` *mkStochastic ensures that it is a stochastic vector, matrix or array.*

Description

mkStochastic ensures that it is a stochastic vector, matrix or array.

Usage

```
mkStochastic(M)
```

Arguments

`M`	A vector, matrix or array to transform.

Details

mkStochastic ensures that the giving argument is a stochastic vector, matrix or array, i.e., that the sum over the last dimension is 1.

Value

A vector, matrix or array for which the sum over the last dimension is 1.

`ModelHMMR-class` *A Reference Class which represents a fitted HMMR model.*

Description

ModelHMMR represents an estimated HMMR model.

Fields

`param` An object of class ParamHMMR. It contains the estimated values of the parameters.

`stat` An object of class StatHMMR. It contains all the statistics associated to the HMMR model.

Methods

`plot(what = c("predicted", "filtered", "smoothed", "regressors", "loglikelihood"), ...)`
Plot method.

`what` The type of graph requested:

- `"predicted"` = Predicted time series and predicted regime probabilities (fields `predicted` and `predict_prob` of class StatHMMR).
- `"filtered"` = Filtered time series and filtering regime probabilities (fields `filtered` and `filter_prob` of class StatHMMR).
- `"smoothed"` = Smoothed time series, and segmentation (fields `smoothed` and `klas` of the class StatHMMR).
- `"regressors"` = Polynomial regression components (fields `regressors` and `tau_tk` of class StatHMMR).
- `"loglikelihood"` = Value of the log-likelihood for each iteration (field `stored_loglik` of class StatHMMR).

`...` Other graphics parameters.

By default, all the graphs mentioned above are produced.

`summary(digits = getOption("digits"))` Summary method.

`digits` The number of significant digits to use when printing.

See Also

ParamHMMR, StatHMMR

Examples

```
data(univtoydataset)

hmmr <- emHMMR(univtoydataset$x, univtoydataset$y, K = 5, p = 1, verbose = TRUE)

# hmmr is a ModelHMMR object. It contains some methods such as 'summary' and 'plot'
hmmr$summary()
hmmr$plot()

# hmmr has also two fields, stat and param which are reference classes as well

# Log-likelihood:
hmmr$stat$loglik

# Parameters of the polynomial regressions:
hmmr$param$beta
```

`ModelMHMMR-class` *A Reference Class which represents a fitted MHMMR model.*

Description

ModelMHMMR represents an estimated MHMMR model.

Fields

`param` A ParamMHMMR object. It contains the estimated values of the parameters.

`stat` A StatMHMMR object. It contains all the statistics associated to the MHMMR model.

Methods

`plot(what = c("predicted", "filtered", "smoothed", "regressors", "loglikelihood"), ...)`
Plot method.

`what` The type of graph requested:

- `"predicted"` = Predicted time series and predicted regime probabilities (fields `predicted` and `predict_prob` of class StatMHMMR).
- `"filtered"` = Filtered time series and filtering regime probabilities (fields `filtered` and `filter_prob` of class StatMHMMR).
- `"smoothed"` = Smoothed time series, and segmentation (fields `smoothed` and `klas` of class StatMHMMR).
- `"regressors"` = Polynomial regression components (fields `regressors` and `tau_tk` of class StatMHMMR).
- `"loglikelihood"` = Value of the log-likelihood for each iteration (field `stored_loglik` of class StatMHMMR).

`...` Other graphics parameters.

By default, all the above graphs are produced.

`summary(digits = getOption("digits"))` Summary method.

`digits` The number of significant digits to use when printing.

See Also

ParamMHMMR, StatMHMMR

Examples

```
data(multivtoydataset)

mhmmr <- emMHMMR(multivtoydataset$x, multivtoydataset[,c("y1", "y2", "y3")],
                 K = 5, p = 1, verbose = TRUE)

# mhmmr is a ModelMHMMR object. It contains some methods such as 'summary' and 'plot'
mhmmr$summary()
mhmmr$plot()

# mhmmr has also two fields, stat and param which are reference classes as well

# Log-likelihood:
mhmmr$stat$loglik

# Parameters of the polynomial regressions:
mhmmr$param$beta
```

ModelMRHLP-class *A Reference Class which represents a fitted MRHLP model.*

Description

ModelMRHLP represents an estimated MRHLP model.

Fields

`param` A ParamMRHLP object. It contains the estimated values of the parameters.

`stat` A StatMRHLP object. It contains all the statistics associated to the MRHLP model.

Methods

`plot(what = c("regressors", "estimatedsignal", "loglikelihood"), ...)` Plot method.

`what` The type of graph requested:

- `"regressors"` = Polynomial regression components (fields `polynomials` and `pi_ik` of class StatMRHLP).
- `"estimatedsignal"` = Estimated signal (fields `Ex` and `klas` of class StatMRHLP).
- `"loglikelihood"` = Value of the log-likelihood for each iteration (field `stored_loglik` of class StatMRHLP).

`...` Other graphics parameters.

By default, all the graphs mentioned above are produced.

`summary(digits = getOption("digits"))` Summary method.

`digits` The number of significant digits to use when printing.

See Also

ParamMRHLP, StatMRHLP

Examples

```
data(multivtoydataset)

mrhlp <- emMRHLP(multivtoydataset$x, multivtoydataset[,c("y1", "y2", "y3")],
                 K = 5, p = 1, verbose = TRUE)

# mrhlp is a ModelMRHLP object. It contains some methods such as 'summary' and 'plot'
mrhlp$summary()
mrhlp$plot()

# mrhlp has also two fields, stat and param which are reference classes as well

# Log-likelihood:
mrhlp$stat$loglik

# Parameters of the polynomial regressions:
mrhlp$param$beta
```

ModelPWR-class *A Reference Class which represents a fitted PWR model.*

Description

ModelPWR represents an estimated PWR model.

Fields

`param` A ParamPWR object. It contains the estimated values of the parameters.

`stat` A StatPWR object. It contains all the statistics associated to the PWR model.

Methods

`plot(what = c("regressors", "segmentation"), ...)` Plot method.

`what` The type of graph requested:

- `"regressors"` = Polynomial regression components (field `regressors` of class StatPWR).
- `"segmentation"` = Estimated signal (field `mean_function` of class StatPWR).

`...` Other graphics parameters.

By default, all the graphs mentioned above are produced.

`summary(digits = getOption("digits"))` Summary method.

`digits` The number of significant digits to use when printing.

See Also

ParamPWR, StatPWR

Examples

```
data(univtoydataset)

pwr <- fitPWRFisher(univtoydataset$x, univtoydataset$y, K = 5, p = 1)

# pwr is a ModelPWR object. It contains some methods such as 'summary' and 'plot'
pwr$summary()
pwr$plot()

# pwr has also two fields, stat and param which are reference classes as well

# Value of the objective function:
pwr$stat$objective

# Parameters of the polynomial regressions:
pwr$param$beta
```

ModelRHLP-class *A Reference Class which represents a fitted RHLP model.*

Description

ModelRHLP represents an estimated RHLP model.

Fields

`param` A ParamRHLP object. It contains the estimated values of the parameters.

`stat` A StatRHLP object. It contains all the statistics associated to the RHLP model.

Methods

`plot(what = c("regressors", "estimatedsignal", "loglikelihood"), ...)` Plot method.

`what` The type of graph requested:

- `"regressors"` = Polynomial regression components (fields `polynomials` and `pi_ik` of class StatRHLP).
- `"estimatedsignal"` = Estimated signal (fields `Ex` and `klas` of class StatRHLP).
- `"loglikelihood"` = Value of the log-likelihood for each iteration (field `stored_loglik` of class StatRHLP).

`...` Other graphics parameters.

By default, all the graphs mentioned above are produced.

`summary(digits = getOption("digits"))` Summary method.

`digits` The number of significant digits to use when printing.

See Also

ParamRHLP, StatRHLP

Examples

```
data(univtoydataset)

rhlp <- emRHLP(univtoydataset$x, univtoydataset$y, K = 3, p = 1, verbose = TRUE)

# rhlp is a ModelMHMMR object. It contains some methods such as 'summary' and 'plot'
rhlp$summary()
rhlp$plot()

# rhlp has also two fields, stat and param which are reference classes as well

# Log-likelihood:
rhlp$stat$loglik

# Parameters of the polynomial regressions:
rhlp$param$beta
```

`multivrealdataset` *Time series representing the three acceleration components recorded over time with body mounted accelerometers during the activity of a given person.*

Description

This dataset is provided for illustration only and represents the three acceleration components recorded over time with body mounted accelerometers during the activity of a given person. These data consist therefore of multidimensional time series with several regime changes over time, each regime is associated with an activity.

Usage

```
multivrealdataset
```

Format

A data frame with 2253 rows and 4 columns:

x The covariate variable (the sampling time).

y1 X axis component of the acceleration.

y2 Y axis component of the acceleration.

y3 Z axis component of the acceleration.

`multivtoydataset` *A simulated non-stationary multidimensional time series with regime changes.*

Description

A simulated non-stationary multidimensional time series with five regimes (segments). This time series is used for illustration.

Usage

```
multivtoydataset
```

Format

A data frame with 670 rows and 4 columns:

x The covariate variable (the sampling time for time series).

y1 The first dimension of the time series. The latter has been generated as follows:

- First regime: 100 values of standard Normally distributed random numbers.
- Second regime: 120 values of Normally distributed random numbers with mean 7 and unit variance.
- Third regime: 200 values of Normally distributed random numbers with mean 4 and unit variance.
- Fourth regime: 100 values of Normally distributed random numbers with mean -1 and unit variance.
- Fifth regime: 150 values of Normally distributed random numbers with mean 3.5 and unit variance.

y2 The second dimension of the time series. The latter has been generated as follows:

- First regime: 100 values of Normally distributed random numbers with mean 1 and unit variance.
- Second regime: 120 values of Normally distributed random numbers with mean 5 and unit variance.
- Third regime: 200 values of Normally distributed random numbers with mean 6 and unit variance.
- Fourth regime: 100 values of Normally distributed random numbers with mean -2 and unit variance.
- Fifth regime: 150 values of Normally distributed random numbers with mean 2 and unit variance.

y3 The third dimension of the time series. The latter has been generated as follows:

- First regime: 100 values of Normally distributed random numbers with mean -2 and unit variance.
- Second regime: 120 values of Normally distributed random numbers with mean 10 and unit variance.

- Third regime: 200 values of Normally distributed random numbers with mean 8 and unit variance.
- Fourth regime: 100 values of Normally distributed random numbers and unit variance.
- Fifth regime: 150 values of Normally distributed random numbers with mean 5 and unit variance.

`ParamHMMR-class` *A Reference Class which contains parameters of a HMMR model.*

Description

ParamHMMR contains all the parameters of a HMMR model. The parameters are calculated by the initialization Method and then updated by the Method implementing the M-Step of the EM algorithm.

Fields

`X` Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.

`Y` Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.

`m` Numeric. Length of the response/output vector `Y`.

`K` The number of regimes (HMMR components).

`p` The order of the polynomial regression.

`variance_type` Character indicating if the model is homoskedastic (`variance_type = "homoskedastic"`) or heteroskedastic (`variance_type = "heteroskedastic"`). By default the model is heteroskedastic.

`prior` The prior probabilities of the Markov chain. `prior` is a row matrix of dimension $(1, K)$.

`trans_mat` The transition matrix of the Markov chain. `trans_mat` is a matrix of dimension (K, K).

`mask` Mask applied to the transition matrices `trans_mat`. By default, a mask of order one is applied.

`beta` Parameters of the polynomial regressions. $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ is a matrix of dimension $(p+1, K)$, with `p` the order of the polynomial regression. `p` is fixed to 3 by default.

`sigma2` The variances for the `K` regimes. If HMMR model is heteroskedastic (`variance_type = "heteroskedastic"`) then `sigma2` is a matrix of size $(K, 1)$ (otherwise HMMR model is homoskedastic (`variance_type = "homoskedastic"`) and `sigma2` is a matrix of size $(1, 1)$).

`nu` The degree of freedom of the HMMR model representing the complexity of the model.

`phi` A list giving the regression design matrices for the polynomial and the logistic regressions.

Methods

`initParam(try_algo = 1)` Method to initialize parameters `mask`, `prior`, `trans_mat`, `beta` and `sigma2`.

If `try_algo = 1` then `beta` and `sigma2` are initialized by segmenting the time series `Y` uniformly into `K` contiguous segments. Otherwise, `beta` and `sigma2` are initialized by segmenting randomly the time series `Y` into `K` segments.

`MStep(statHMMR)` Method which implements the M-step of the EM algorithm to learn the parameters of the HMMR model based on statistics provided by the object `statHMMR` of class StatHMMR (which contains the E-step).

ParamMHMMR-class *A Reference Class which contains parameters of a MHMMR model.*

Description

ParamMHMMR contains all the parameters of a MHMMR model. The parameters are calculated by the initialization Method and then updated by the Method implementing the M-Step of the EM algorithm.

Fields

`mData` MData object representing the sample (covariates/inputs `X` and observed multivariate responses/outputs `Y`).

`K` The number of regimes (MHMMR components).

`p` The order of the polynomial regression.

`variance_type` Character indicating if the model is homoskedastic (`variance_type = "homoskedastic"`) or heteroskedastic (`variance_type = "heteroskedastic"`). By default the model is heteroskedastic.

`prior` The prior probabilities of the Markov chain. `prior` is a row matrix of dimension $(1, K)$.

`trans_mat` The transition matrix of the Markov chain. `trans_mat` is a matrix of dimension (K, K).

`mask` Mask applied to the transition matrices `trans_mat`. By default, a mask of order one is applied.

`beta` Parameters of the polynomial regressions. $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ is an array of dimension $(p + 1, d, K)$, with `p` the order of the polynomial regression. `p` is fixed to 3 by default.

`sigma2` The variances for the `K` regimes. If MRHLP model is heteroskedastic (`variance_type = "heteroskedastic"`) then `sigma2` is an array of size (d, d, K) (otherwise MRHLP model is homoskedastic (`variance_type = "homoskedastic"`) and `sigma2` is a matrix of size (d, d)).

`nu` The degree of freedom of the MHMMR model representing the complexity of the model.

`phi` A list giving the regression design matrices for the polynomial and the logistic regressions.

Methods

`initParam(try_algo = 1)` Method to initialize parameters `prior`, `trans_mat`, `beta` and `sigma2`.

If `try_algo = 1` then `beta` and `sigma2` are initialized by segmenting the time series `Y` uniformly into `K` contiguous segments. Otherwise, `beta` and `sigma2` are initialized by segmenting randomly the time series `Y` into `K` segments.

`MStep(statMHMMR)` Method which implements the M-step of the EM algorithm to learn the parameters of the MHMMR model based on statistics provided by the object `statMHMMR` of class StatMHMMR (which contains the E-step).

ParamMRHLP-class *A Reference Class which contains the parameters of a MRHLP model.*

Description

ParamMRHLP contains all the parameters of a MRHLP model. The parameters are calculated by the initialization Method and then updated by the Method implementing the M-Step of the EM algorithm.

Fields

`mData` MData object representing the sample (covariates/inputs `X` and observed responses/outputs `Y`).

`K` The number of regimes (MRHLP components).

`p` The order of the polynomial regression.

`q` The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1.

`variance_type` Character indicating if the model is homoskedastic (`variance_type = "homoskedastic"`) or heteroskedastic (`variance_type = "heteroskedastic"`). By default the model is heteroskedastic.

`W` Parameters of the logistic process. $\boldsymbol{W} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{K-1})$ is a matrix of dimension $(q+1, K-1)$, with `q` the order of the logistic regression. `q` is fixed to 1 by default.

`beta` Parameters of the polynomial regressions. $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ is an array of dimension $(p+1, d, K)$, with `p` the order of the polynomial regression. `p` is fixed to 3 by default.

`sigma2` The variances for the `K` regimes. If MRHLP model is heteroskedastic (`variance_type = "heteroskedastic"`) then `sigma2` is an array of size (d, d, K) (otherwise MRHLP model is homoskedastic (`variance_type = "homoskedastic"`) and `sigma2` is a matrix of size (d, d)).

`nu` The degree of freedom of the MRHLP model representing the complexity of the model.

`phi` A list giving the regression design matrices for the polynomial and the logistic regressions.

Methods

`initParam(try_algo = 1)` Method to initialize parameters `W`, `beta` and `sigma2`.

If `try_algo = 1` then `beta` and `sigma2` are initialized by segmenting the time series `Y` uniformly into `K` contiguous segments. Otherwise, `W`, `beta` and `sigma2` are initialized by segmenting randomly the time series `Y` into `K` segments.

`MStep(statMRHLP, verbose_IRLS)` Method which implements the M-step of the EM algorithm to learn the parameters of the MRHLP model based on statistics provided by the object `statMRHLP` of class StatMRHLP (which contains the E-step).

ParamPWR-class *A Reference Class which contains the parameters of a PWR model.*

Description

ParamPWR contains all the parameters of a PWR model. The parameters are calculated by the initialization Method and then updated by the Method dynamic programming (here dynamic programming)

Fields

`X` Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.

`Y` Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.

`m` Numeric. Length of the response/output vector `Y`.

`K` The number of regimes (PWR components).

`p` The order of the polynomial regression. `p` is fixed to 3 by default.

`gamma` Set of transition points. `gamma` is a column matrix of size $(K+1, 1)$.

`beta` Parameters of the polynomial regressions. `beta` is a matrix of dimension $(p+1, K)$, with `p` the order of the polynomial regression. `p` is fixed to 3 by default.

`sigma2` The variances for the `K` regimes. `sigma2` is a matrix of size $(K, 1)$.

`phi` A list giving the regression design matrices for the polynomial and the logistic regressions.

Methods

`computeDynamicProgram(C1, K)` Method which implements the dynamic programming based on the cost matrix `C1` and the number of regimes/segments `K`.

`computeParam()` Method which estimates the parameters `beta` and `sigma2` knowing the transition points `gamma`.

`ParamRHLP-class` *A Reference Class which contains parameters of a RHLP model.*

Description

ParamRHLP contains all the parameters of a RHLP model. The parameters are calculated by the initialization Method and then updated by the Method implementing the M-Step of the EM algorithm.

Fields

`X` Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.

`Y` Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.

`m` Numeric. Length of the response/output vector `Y`.

`K` The number of regimes (RHLP components).

`p` The order of the polynomial regression.

`q` The dimension of the logistic regression. For the purpose of segmentation, it must be set to 1.

`variance_type` Character indicating if the model is homoskedastic (`variance_type = "homoskedastic"`) or heteroskedastic (`variance_type = "heteroskedastic"`). By default the model is heteroskedastic.

`W` Parameters of the logistic process. $\boldsymbol{W} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{K-1})$ is a matrix of dimension $(q+1, K-1)$, with `q` the order of the logistic regression. `q` is fixed to 1 by default.

`beta` Parameters of the polynomial regressions. $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K)$ is a matrix of dimension $(p+1, K)$, with `p` the order of the polynomial regression. `p` is fixed to 3 by default.

`sigma2` The variances for the `K` regimes. If RHLP model is heteroskedastic (`variance_type = "heteroskedastic"`) then `sigma2` is a matrix of size $(K, 1)$ (otherwise RHLP model is homoskedastic (`variance_type = "homoskedastic"`) and `sigma2` is a matrix of size $(1, 1)$).

`nu` The degree of freedom of the RHLP model representing the complexity of the model.

`phi` A list giving the regression design matrices for the polynomial and the logistic regressions.

Methods

`initParam(try_algo = 1)` Method to initialize parameters `W`, `beta` and `sigma2`.

If `try_algo = 1` then `beta` and `sigma2` are initialized by segmenting the time series `Y` uniformly into `K` contiguous segments. Otherwise, `W`, `beta` and `sigma2` are initialized by segmenting randomly the time series `Y` into `K` segments.

`MStep(statRHLP, verbose_IRLS)` Method which implements the M-step of the EM algorithm to learn the parameters of the RHLP model based on statistics provided by the object `statRHLP` of class StatRHLP (which contains the E-step).

`selectHMMR` *selectHMMR implements a model selection procedure to select an optimal HMMR model with unknown structure.*

Description

selectHMMR implements a model selection procedure to select an optimal HMMR model with unknown structure.

Usage

```
selectHMMR(X, Y, Kmin = 1, Kmax = 10, pmin = 0, pmax = 4,
  criterion = c("BIC", "AIC"), verbose = TRUE)
```

Arguments

`X`	Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.
`Y`	Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.
`Kmin`	The minimum number of regimes (HMMR components).
`Kmax`	The maximum number of regimes (HMMR components).
`pmin`	The minimum order of the polynomial regression.
`pmax`	The maximum order of the polynomial regression.
`criterion`	The criterion used to select the HMMR model ("BIC", "AIC").
`verbose`	Optional. A logical value indicating whether or not a summary of the selected model should be displayed.

Details

selectHMMR selects the optimal HMMR model among a set of model candidates by optimizing a model selection criteria, including the Bayesian Information Criterion (BIC). This function first fits the different HMMR model candidates by varying the number of regimes `K` from `Kmin` to `Kmax` and the order of the polynomial regression `p` from `pmin` to `pmax`. The model having the highest value of the chosen selection criterion is then selected.

Value

selectHMMR returns an object of class ModelHMMR representing the selected HMMR model according to the chosen `criterion`.

See Also

ModelHMMR

Examples

```
data(univtoydataset)

selectedhmmr <- selectHMMR(X = univtoydataset$x, Y = univtoydataset$y,
                           Kmin = 2, Kmax = 6, pmin = 0, pmax = 2)

selectedhmmr$plot()
```

`selectMHMMR` *selectMHMMR implements a model selection procedure to select an optimal MHMMR model with unknown structure.*

Description

selectMHMMR implements a model selection procedure to select an optimal MHMMR model with unknown structure.

Usage

```
selectMHMMR(X, Y, Kmin = 1, Kmax = 10, pmin = 0, pmax = 4,
  criterion = c("BIC", "AIC"), verbose = TRUE)
```

Arguments

`X`	Numeric vector of length m representing the covariates/inputs $x_1, \ldots, x_m$.
`Y`	Matrix of size (m, d) representing a d dimension time series observed at points $1, \ldots, m$.
`Kmin`	The minimum number of regimes (c components).
`Kmax`	The maximum number of regimes (MHMMR components).
`pmin`	The minimum order of the polynomial regression.
`pmax`	The maximum order of the polynomial regression.
`criterion`	The criterion used to select the MHMMR model ("BIC", "AIC").
`verbose`	Optional. A logical value indicating whether or not a summary of the selected model should be displayed.

Details

selectMHMMR selects the optimal MHMMR model among a set of model candidates by optimizing a model selection criteria, including the Bayesian Information Criterion (BIC). This function first fits the different MHMMR model candidates by varying the number of regimes `K` from `Kmin` to `Kmax` and the order of the polynomial regression `p` from `pmin` to `pmax`. The model having the highest value of the chosen selection criterion is then selected.

Value

selectMHMMR returns an object of class ModelMHMMR representing the selected MHMMR model according to the chosen `criterion`.

See Also

ModelMHMMR

Examples

```
data(multivtoydataset)
x <- multivtoydataset$x
y <- multivtoydataset[, c("y1", "y2", "y3")]

selectedmhmmr <- selectMHMMR(X = x, Y = y, Kmin = 2, Kmax = 6,
                             pmin = 0, pmax = 2)

selectedmhmmr$summary()
```

`selectMRHLP` *selecMRHLP implements a model selection procedure to select an optimal MRHLP model with unknown structure.*

Description

selecMRHLP implements a model selection procedure to select an optimal MRHLP model with unknown structure.

Usage

```
selectMRHLP(X, Y, Kmin = 1, Kmax = 10, pmin = 0, pmax = 4,
  criterion = c("BIC", "AIC"), verbose = TRUE)
```

Arguments

`X`	Numeric vector of length m representing the covariates/inputs $x_1, \ldots, x_m$.
`Y`	Matrix of size (m, d) representing a d dimension function of `X` observed at points $1, \ldots, m$. `Y` is the observed response/output.
`Kmin`	The minimum number of regimes (MRHLP components).
`Kmax`	The maximum number of regimes (MRHLP components).
`pmin`	The minimum order of the polynomial regression.
`pmax`	The maximum order of the polynomial regression.
`criterion`	The criterion used to select the MRHLP model ("BIC", "AIC").
`verbose`	Optional. A logical value indicating whether or not a summary of the selected model should be displayed.

Details

selectMRHLP selects the optimal MRHLP model among a set of model candidates by optimizing a model selection criteria, including the Bayesian Information Criterion (BIC). This function first fits the different MRHLP model candidates by varying the number of regimes `K` from `Kmin` to `Kmax` and the order of the polynomial regression `p` from `pmin` to `pmax`. The model having the highest value of the chosen selection criterion is then selected.

Value

selectMRHLP returns an object of class ModelMRHLP representing the selected MRHLP model according to the chosen `criterion`.

See Also

ModelMRHLP

Examples

```
data(multivtoydataset)

# Let's select a MRHLP model on a multivariate time series with 3 regimes:
data <- multivtoydataset[1:320, ]
x <- data$x
y <- data[, c("y1", "y2", "y3")]

selectedmrhlp <- selectMRHLP(X = x, Y = y, Kmin = 2, Kmax = 4,
                             pmin = 0, pmax = 1)

selectedmrhlp$summary()
```

`selectRHLP` *selecRHLP implements a model selection procedure to select an optimal RHLP model with unknown structure.*

Description

selecRHLP implements a model selection procedure to select an optimal RHLP model with unknown structure.

Usage

```
selectRHLP(X, Y, Kmin = 1, Kmax = 10, pmin = 0, pmax = 4,
  criterion = c("BIC", "AIC"), verbose = TRUE)
```

Arguments

`X`	Numeric vector of length *m* representing the covariates/inputs $x_1, \ldots, x_m$.
`Y`	Numeric vector of length *m* representing the observed response/output $y_1, \ldots, y_m$.
`Kmin`	The minimum number of regimes (RHLP components).
`Kmax`	The maximum number of regimes (RHLP components).
`pmin`	The minimum order of the polynomial regression.
`pmax`	The maximum order of the polynomial regression.
`criterion`	The criterion used to select the RHLP model ("BIC", "AIC").
`verbose`	Optional. A logical value indicating whether or not a summary of the selected model should be displayed.

Details

selectRHLP selects the optimal MRHLP model among a set of model candidates by optimizing a model selection criteria, including the Bayesian Information Criterion (BIC). This function first fits the different RHLP model candidates by varying the number of regimes `K` from `Kmin` to `Kmax` and the order of the polynomial regression `p` from `pmin` to `pmax`. The model having the highest value of the chosen selection criterion is then selected.

Value

selectRHLP returns an object of class ModelRHLP representing the selected RHLP model according to the chosen `criterion`.

See Also

ModelRHLP

Examples

```
data(univtoydataset)

# Let's select a RHLP model on a time series with 3 regimes:
data <- univtoydataset[1:320,]

selectedrhlp <- selectRHLP(X = data$x, Y = data$y,
                           Kmin = 2, Kmax = 4, pmin = 0, pmax = 1)

selectedrhlp$summary()
```

`StatHMMR-class` *A Reference Class which contains statistics of a HMMR model.*

Description

StatHMMR contains all the statistics associated to a HMMR model. It mainly includes the E-Step of the EM algorithm calculating the posterior distribution of the hidden variables (ie the smoothing probabilities), as well as the calculation of the prediction and filtering probabilities, the log-likelihood at each step of the algorithm and the obtained values of model selection criteria..

Fields

`tau_tk` Matrix of size (m, K) giving the posterior probability that the observation Y_i originates from the k-th regression model.

`alpha_tk` Matrix of size (m, K) giving the forwards probabilities: $P(Y_1, \ldots, Y_t, z_t = k)$.

`beta_tk` Matrix of size (m, K), giving the backwards probabilities: $P(Y_{t+1}, \ldots, Y_m | z_t = k)$.

`xi_tkl` Array of size $(m-1, K, K)$ giving the joint post probabilities: $xi_t k[t, k, l] = P(z_t = k, z_{t-1} = l|\boldsymbol{Y})$ for $t = 2, \ldots, m$.

`f_tk` Matrix of size (m, K) giving the cumulative distribution function $f(y_t|z_t = k)$.

`log_f_tk` Matrix of size (m, K) giving the logarithm of the cumulative distribution `f_tk`.

`loglik` Numeric. Log-likelihood of the HMMR model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each iteration of the EM algorithm.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \ldots, K$.

`z_ik` Hard segmentation logical matrix of dimension (m, K) obtained by the Maximum a posteriori (MAP) rule: $z\_ik = 1$ if $z\_ik = \arg\max_s P(z_i = s|\boldsymbol{Y}) = tau\_tk$; 0 otherwise, $k = 1, \ldots, K$.

`state_probs` Matrix of size (m, K) giving the distribution of the Markov chain. $P(z_1, \ldots, z_m; \pi, \boldsymbol{A})$ with π the prior probabilities (field `prior` of the class ParamHMMR) and $\boldsymbol{A}$ the transition matrix (field `trans_mat` of the class ParamHMMR) of the Markov chain.

`BIC` Numeric. Value of BIC (Bayesian Information Criterion).

`AIC` Numeric. Value of AIC (Akaike Information Criterion).

`regressors` Matrix of size (m, K) giving the values of the estimated polynomial regression components.

`predict_prob` Matrix of size (m, K) giving the prediction probabilities: $P(z_t = k|y_1, \ldots, y_{t-1})$.

`predicted` Row matrix of size $(m, 1)$ giving the sum of the polynomial components weighted by the prediction probabilities `predict_prob`.

`filter_prob` Matrix of size (m, K) giving the filtering probabilities $Pr(z_t = k|y_1, \ldots, y_t)$.

`filtered` Row matrix of size $(m, 1)$ giving the sum of the polynomial components weighted by the filtering probabilities.

`smoothed_regressors` Matrix of size (m, K) giving the polynomial components weighted by the posterior probability `tau_tk`.

`smoothed` Row matrix of size $(m, 1)$ giving the sum of the polynomial components weighted by the posterior probability `tau_tk`.

Methods

`computeLikelihood(paramHMMR)` Method to compute the log-likelihood based on some parameters given by the object `paramHMMR` of class ParamHMMR.

`computeStats(paramHMMR)` Method used in the EM algorithm to compute statistics based on parameters provided by the object `paramHMMR` of class ParamHMMR.

`EStep(paramHMMR)` Method used in the EM algorithm to update statistics based on parameters provided by the object `paramHMMR` of class ParamHMMR (prior and posterior probabilities).

`MAP()` MAP calculates values of the fields `z_ik` and `klas` by applying the Maximum A Posteriori Bayes allocation rule.

$z\_ik = 1$ if $z\_ik = \arg\max_s P(z_i = s|\boldsymbol{Y}) = tau\_tk$; 0 otherwise

See Also

ParamHMMR

`StatMHMMR-class` *A Reference Class which contains statistics of a MHMMR model.*

Description

StatMHMMR contains all the statistics associated to a MHMMR model. It mainly includes the E-Step of the EM algorithm calculating the posterior distribution of the hidden variables (ie the smoothing probabilities), as well as the calculation of the prediction and filtering probabilities, the log-likelihood at each step of the algorithm and the obtained values of model selection criteria..

Fields

`tau_tk` Matrix of size (m, K) giving the posterior probability that the observation Y_i originates from the k-th regression model.

`alpha_tk` Matrix of size (m, K) giving the forwards probabilities: $P(Y_1, \ldots, Y_t, z_t = k)$.

`beta_tk` Matrix of size (m, K), giving the backwards probabilities: $P(Y_{t+1}, \ldots, Y_m | z_t = k)$.

`xi_tkl` Array of size $(m-1, K, K)$ giving the joint post probabilities: $xi_t k[t, k, l] = P(z_t = k, z_{t-1} = l|\boldsymbol{Y})$ for $t = 2, \ldots, m$.

`f_tk` Matrix of size (m, K) giving the cumulative distribution function $f(y_t|z_t = k)$.

`log_f_tk` Matrix of size (m, K) giving the logarithm of the cumulative distribution `f_tk`.

`loglik` Numeric. Log-likelihood of the MHMMR model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each iteration of the EM algorithm.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \ldots, K$.

`z_ik` Hard segmentation logical matrix of dimension (m, K) obtained by the Maximum a posteriori (MAP) rule: $z\_ik = 1$ if $z\_ik = \arg\max_s P(z_i = s|\boldsymbol{Y}) = tau\_tk$; 0 otherwise, $k = 1, \ldots, K$.

`state_probs` Matrix of size (m, K) giving the distribution of the Markov chain. $P(z_1, \ldots, z_m; \pi, \boldsymbol{A})$ with π the prior probabilities (field `prior` of the class ParamMHMMR) and $\boldsymbol{A}$ the transition matrix (field `trans_mat` of the class ParamMHMMR) of the Markov chain.

`BIC` Numeric. Value of BIC (Bayesian Information Criterion).

`AIC` Numeric. Value of AIC (Akaike Information Criterion).

`regressors` Matrix of size (m, K) giving the values of the estimated polynomial regression components.

`predict_prob` Matrix of size (m, K) giving the prediction probabilities: $P(z_t = k|y_1, \ldots, y_{t-1})$.

`predicted` Row matrix of size $(m, 1)$ giving the sum of the polynomial components weighted by the prediction probabilities `predict_prob`.

`filter_prob` Matrix of size (m, K) giving the filtering probabilities $Pr(z_t = k|y_1, \ldots, y_t)$.

`filtered` Row matrix of size $(m, 1)$ giving the sum of the polynomial components weighted by the filtering probabilities.

`smoothed_regressors` Matrix of size (m, K) giving the polynomial components weighted by the posterior probability `tau_tk`.

`smoothed` Row matrix of size $(m, 1)$ giving the sum of the polynomial components weighted by the posterior probability `tau_tk`.

Methods

`computeLikelihood(paramMHMMR)` Method to compute the log-likelihood based on some parameters given by the object `paramMHMMR` of class ParamMHMMR.

`computeStats(paramMHMMR)` Method used in the EM algorithm to compute statistics based on parameters provided by the object `paramMHMMR` of class ParamMHMMR.

`EStep(paramMHMMR)` Method used in the EM algorithm to update statistics based on parameters provided by the object `paramMHMMR` of class ParamMHMMR (prior and posterior probabilities).

`MAP()` MAP calculates values of the fields `z_ik` and `klas` by applying the Maximum A Posteriori Bayes allocation rule.
$z\_ik = 1$ if $z\_ik = \arg\max_s P(z_i = s|\boldsymbol{Y}) = tau\_tk$; 0 otherwise

See Also

ParamMHMMR

`StatMRHLP-class` *A Reference Class which contains statistics of a MRHLP model.*

Description

StatMRHLP contains all the statistics associated to a MRHLP model. It mainly includes the E-Step of the EM algorithm calculating the posterior distribution of the hidden variables, as well as the calculation of the log-likelihood at each step of the algorithm and the obtained values of model selection criteria..

Fields

`pi_ik` Matrix of size (m, K) representing the prior/logistic probabilities $\pi_k(x_i; \boldsymbol{\Psi}) = P(z_i = k|\boldsymbol{x}; \Psi)$ of the latent variable $z_i, i = 1, \ldots, m$.

`z_ik` Hard segmentation logical matrix of dimension (m, K) obtained by the Maximum a posteriori (MAP) rule: $z\_ik = 1$ if $z\_ik = \arg\max_s \pi_s(x_i; \boldsymbol{\Psi})$; 0 otherwise, $k = 1, \ldots, K$.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \ldots, K$.

`tau_ik` Matrix of size (m, K) giving the posterior probability that the observation Y_i originates from the k-th regression model.

`polynomials` Array of size (m, d, K) giving the values of the estimated polynomial regression components.

`weighted_polynomials` Array of size (m, d, K) giving the values of the estimated polynomial regression components weighted by the prior probabilities `pi_ik`.

`Ex` Matrix of size *(m, d)*. `Ex` is the curve expectation (estimated signal): sum of the polynomial components weighted by the logistic probabilities `pi_ik`.

`loglik` Numeric. Observed-data log-likelihood of the MRHLP model.

`com_loglik` Numeric. Complete-data log-likelihood of the MRHLP model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each EM iteration.

`stored_com_loglik` Numeric vector. Stored values of the Complete log-likelihood at each EM iteration.

`BIC` Numeric. Value of BIC (Bayesian Information Criterion).

`ICL` Numeric. Value of ICL (Integrated Completed Likelihood).

`AIC` Numeric. Value of AIC (Akaike Information Criterion).

`log_piik_fik` Matrix of size (m, K) giving the values of the logarithm of the joint probability $P(y_i,\ z_i = k|\boldsymbol{x}, \boldsymbol{\Psi}), i = 1, \ldots, m$.

`log_sum_piik_fik` Column matrix of size *m* giving the values of $\log \sum_{k=1}^{K} P(y_i,\ z_i = k|\boldsymbol{x}, \boldsymbol{\Psi})$, $i = 1, \ldots, m$.

Methods

`computeLikelihood(reg_irls)` Method to compute the log-likelihood. `reg_irls` is the value of the regularization part in the IRLS algorithm.

`computeStats(paramMRHLP)` Method used in the EM algorithm to compute statistics based on parameters provided by the object `paramMRHLP` of class ParamMRHLP.

`EStep(paramMRHLP)` Method used in the EM algorithm to update statistics based on parameters provided by the object `paramMRHLP` of class ParamMRHLP (prior and posterior probabilities).

`MAP()` MAP calculates values of the fields `z_ik` and `klas` by applying the Maximum A Posteriori Bayes allocation rule.

$z_{ik} = 1$ if $k = \arg\max_s \pi_s(x_i; \boldsymbol{\Psi});\ 0$ otherwise

See Also

ParamMRHLP

`StatPWR-class` *A Reference Class which contains statistics of a PWR model.*

Description

StatPWR contains all the statistics associated to a PWR model.

Fields

`z_ik` Logical matrix of dimension (m, K) giving the class vector.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \ldots, K$.

`mean_function` Approximation of the time series given the estimated parameters. `mean_function` is a matrix of size $(m, 1)$.

`regressors` Matrix of size (m, K) giving the values of the estimated polynomial regression components.

`objective` Numeric. Value of the objective function.

Methods

`computeStats(paramPWR)` Method used at the end of the dynamic programming algorithm to compute statistics based on parameters provided by `paramPWR`.

See Also

ParamPWR

`StatRHLP-class` *A Reference Class which contains statistics of a RHLP model.*

Description

StatRHLP contains all the statistics associated to a RHLP model. It mainly includes the E-Step of the EM algorithm calculating the posterior distribution of the hidden variables, as well as the calculation of the log-likelihood at each step of the algorithm and the obtained values of model selection criteria..

Fields

`pi_ik` Matrix of size (m, K) representing the prior/logistic probabilities $\pi_k(x_i; \mathbf{\Psi}) = P(z_i = k|\boldsymbol{x}; \Psi)$ of the latent variable $z_i, i = 1, \ldots, m$.

`z_ik` Hard segmentation logical matrix of dimension (m, K) obtained by the Maximum a posteriori (MAP) rule: $z\_ik = 1$ if $z\_ik =$ arg max$_s$ $\pi_s(x_i; \mathbf{\Psi})$; 0 otherwise, $k = 1, \ldots, K$.

`klas` Column matrix of the labels issued from `z_ik`. Its elements are $klas(i) = k$, $k = 1, \ldots, K$.

`tau_ik` Matrix of size (m, K) giving the posterior probability that the observation Y_i originates from the k-th regression model.

`polynomials` Matrix of size (m, K) giving the values of the estimated polynomial regression components.

`Ex` Column matrix of dimension *m*. `Ex` is the curve expectation (estimated signal): sum of the polynomial components weighted by the logistic probabilities `pi_ik`.

`loglik` Numeric. Observed-data log-likelihood of the RHLP model.

`com_loglik` Numeric. Complete-data log-likelihood of the RHLP model.

`stored_loglik` Numeric vector. Stored values of the log-likelihood at each EM iteration.

`stored_com_loglik` Numeric vector. Stored values of the Complete log-likelihood at each EM iteration.

`BIC` Numeric. Value of BIC (Bayesian Information Criterion).

`ICL` Numeric. Value of ICL (Integrated Completed Likelihood).

`AIC` Numeric. Value of AIC (Akaike Information Criterion).

`log_piik_fik` Matrix of size (m, K) giving the values of the logarithm of the joint probability $P(y_i,\ z_i = k|\boldsymbol{x}, \mathbf{\Psi})$, $i = 1, \ldots, m$.

`log_sum_piik_fik` Column matrix of size *m* giving the values of log $\sum_{k=1}^{K} P(y_i,\ z_i = k|\boldsymbol{x}, \mathbf{\Psi})$, $i = 1, \ldots, m$.

Methods

`computeLikelihood(reg_irls)` Method to compute the log-likelihood. `reg_irls` is the value of the regularization part in the IRLS algorithm.

`computeStats(paramRHLP)` Method used in the EM algorithm to compute statistics based on parameters provided by the object `paramRHLP` of class ParamRHLP.

`EStep(paramRHLP)` Method used in the EM algorithm to update statistics based on parameters provided by the object `paramRHLP` of class ParamRHLP (prior and posterior probabilities).

`MAP()` MAP calculates values of the fields `z_ik` and `klas` by applying the Maximum A Posteriori Bayes allocation rule.

$z_{ik} = 1$ if $k = \arg\max_s \pi_s(x_i; \mathbf{\Psi})$; 0 otherwise

See Also

ParamRHLP

`univrealdataset` *Time series representing the electrical power consumption during a railway switch operation*

Description

This dataset is provided for illustration only; It is issued from the switch railway monitoring domain. The switch mechanism enables trains to be guided from one track to another at a railway junction. During each switch operation, a set of measurements are recorded. Each measurement represents the consumed electrical power. The resulting time series present regime changes.

Usage

```
univrealdataset
```

Format

A data frame with 562 rows and 3 columns:

x The covariate variables which are the sampling time in this time-series case.

y1 Measurements of the electrical power consumed during time for a first example of switch operations.

y2 Measurements of the electrical power consumed during during time for another example of switch operations.

`univtoydataset` *A simulated non-stationary time series with regime changes.*

Description

A simulated non-stationary time series with regime changes. This time series is used for illustration.

Usage

```
univtoydataset
```

Format

A data frame with 670 rows and 2 columns:

x The covariate variable which is the time in this time-series case.

y The time series. The latter has been generated as follows:

- First regime: 100 values of standard Normally distributed random numbers.
- Second regime: 120 values of Normally distributed random numbers with mean 7 and unit variance.
- Third regime: 200 values of Normally distributed random numbers with mean 4 and unit variance.
- Fourth regime: 100 values of Normally distributed random numbers with mean -2 and unit variance.
- Fifth regime: 150 values of Normally distributed random numbers with mean 3.5 and unit variance.

Index

∗ **datasets**
    multivrealdataset, 17
    multivtoydataset, 18
    univrealdataset, 34
    univtoydataset, 35

emHMMR, 4
emMHMMR, 5
emMRHLP, 7
emRHLP, 8

fitPWRFisher, 9

hmmProcess, 10
HMMR, *28*

MData, *20, 21*
MData *(*MData-class*)*, 11
MData-class, 11
MHMMR, *30*
mkStochastic, 11
ModelHMMR, *5*, *24*
ModelHMMR *(*ModelHMMR-class*)*, 12
ModelHMMR-class, 12
ModelMHMMR, *6*, *25, 26*
ModelMHMMR *(*ModelMHMMR-class*)*, 13
ModelMHMMR-class, 13
ModelMRHLP, *8*, *27*
ModelMRHLP *(*ModelMRHLP-class*)*, 14
ModelMRHLP-class, 14
ModelPWR, *10*
ModelPWR *(*ModelPWR-class*)*, 15
ModelPWR-class, 15
ModelRHLP, *9*, *28*
ModelRHLP *(*ModelRHLP-class*)*, 16
ModelRHLP-class, 16
MRHLP, *31*
multivrealdataset, 17
multivtoydataset, 18

ParamHMMR, *5*, *12*, *29*
ParamHMMR *(*ParamHMMR-class*)*, 19
ParamHMMR-class, 19
ParamMHMMR, *6*, *13, 14*, *30, 31*
ParamMHMMR *(*ParamMHMMR-class*)*, 20
ParamMHMMR-class, 20
ParamMRHLP, *7, 8*, *14, 15*, *32*
ParamMRHLP *(*ParamMRHLP-class*)*, 21
ParamMRHLP-class, 21
ParamPWR, *10*, *15, 16*, *33*
ParamPWR *(*ParamPWR-class*)*, 22
ParamPWR-class, 22
ParamRHLP, *9*, *16, 17*, *34*
ParamRHLP *(*ParamRHLP-class*)*, 23
ParamRHLP-class, 23
PWR, *32*

RHLP, *33*

samurais *(*samurais-package*)*, 3
samurais-package, 3
selectHMMR, 24
selectMHMMR, 25
selectMRHLP, 26
selectRHLP, 27
StatHMMR, *5*, *12*, *20*
StatHMMR *(*StatHMMR-class*)*, 28
StatHMMR-class, 28
StatMHMMR, *6*, *13, 14*, *21*
StatMHMMR *(*StatMHMMR-class*)*, 30
StatMHMMR-class, 30
StatMRHLP, *7, 8*, *14, 15*, *22*
StatMRHLP *(*StatMRHLP-class*)*, 31
StatMRHLP-class, 31
StatPWR, *10*, *15, 16*
StatPWR *(*StatPWR-class*)*, 32
StatPWR-class, 32
StatRHLP, *9*, *16, 17*, *23*
StatRHLP *(*StatRHLP-class*)*, 33
StatRHLP-class, 33

univrealdataset, 34
univtoydataset, 35