Package 'countHMM'

October 12, 2022

Type Package

Title Penalized Estimation of Flexible Hidden Markov Models for Time Series of Counts

Version 0.1.0

Author Timo Adam

Maintainer Timo Adam <timo.adam@uni-bielefeld.de>

Description Provides tools for penalized estimation of flexible hidden Markov models for time series of counts w/o the need to specify a (parametric) family of distributions. These include functions for model fitting, model checking, and state decoding. For details, see Adam, T., Langrock, R., and Weiß, C.H. (2019): Penalized Estimation of Flexible Hidden Markov Models for Time Series of Counts. <arXiv:1901.03275>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-04-24 13:00:02 UTC

R topics documented:

fitMod *fitMod*

Description

Estimates the parameters of a hidden Markov model using maximum penalized likelihood estimation. For details, see Adam *et al.* (2019).

Usage

```
fitMod(x,N=2,probs0=NULL,gamma0=NULL,delta0=NULL,stationary=TRUE,lambda=NULL,sup=NULL,m=3,
inflation=NULL)
```

Arguments

x	Vector containing the observed time series of counts.
N	Integer, number of states. Default is N=2.
probs0	Matrix with N columns which contains initial parameter values for the state-dependent probabilities for each count. If probs0=NULL (default), then the initial parameter values are generated automatically (not recommended).
gamma0	Initial parameter values for the transition probabilities of the Markov chain underlying the observed counts. Matrix with N rows and N columns. If gamma0=NULL (default), then the initial parameter values are generated automatically (not recommended).
delta0	Initial parameter values for the initial probabilities of the Markov chain underlying the observed counts. Vector of length N. If delta0=NULL (default), then the stationary distribution is used.
stationary	Logical, determines whether the initial distribution of the Markov chain underlying the observed counts is the stationary distribution. Default is stationary=TRUE.
lambda	Vector of length N which contains the smoothing parameters associated with the different state-dependent distributions. Default is lambda=rep(0,N).
sup	Integer, determines the upper bound of the support of the state-dependent distributions. If NULL (default), then the maximum of x is used.
m	Integer, order of the difference penalties. Default is m=3.
inflation	Count probabilities to be excluded from penalization (e.g. in the presence of zero-inflation). Default is inflation=NULL.

Value

An object of type countHMM.

References

Adam, T., Langrock, R., and Weiß, C.H. (2019): Penalized Estimation of Flexible Hidden Markov Models for Time Series of Counts. arXiv:https://arxiv.org/pdf/1901.03275.pdf.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# model fitting
lambda = rep(10^4,2)
fitMod(x=x,lambda=lambda)
```

---

nLogLike *Penalized negative log-likelihood*

Description

Computes the penalized negative log-likelihood using the forward algorithm as described in Adam *et al.* (2019). Not intended to be run by the user (internal function, called by the function `fitMod`).

Usage

```
nLogLike(parvect,x,N,stationary,lambda,sup,m,inflation)
```

Arguments

`parvect`	Vector of working parameters (as returned by `pn2pw`).
`x`	Vector of observed counts.
`N`	Integer, number of states.
`stationary`	Logical, determines whether the initial distribution of the Markov chain underlying the observed counts is the stationary distribution.
`lambda`	Vector of length `N` which contains the smoothing parameters associated with the different state-dependent distributions.
`sup`	Integer, determines the upper bound of the support of the state-dependent distributions. If `NULL`, then the maximum of `x` is used.
`m`	Integer, order of the difference penalties.
`inflation`	Count probabilities to be excluded from penalization (e.g. in the presence of zero-inflation).

Value

Numeric, the penalized negative log-likelihood.

References

Adam, T., Langrock, R., and Weiß, C.H. (2019): Penalized Estimation of Flexible Hidden Markov Models for Time Series of Counts. arXiv:https://arxiv.org/pdf/1901.03275.pdf.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# computing the penalized negative log-likelihood
parvect = pn2pw(N=2,probs=cbind(dpois(x=0:41,lambda=14),dpois(x=0:41,lambda=26)),
gamma=matrix(c(0.95,0.05,0.05,0.95),ncol=2),delta=NULL,stationary=TRUE)
lambda = rep(10^4,2)
nLogLike(parvect=parvect,x=x,N=2,stationary=TRUE,lambda=lambda,sup=41,m=3,inflation=FALSE)
```

---

plotMod *plotMod*

Description

Plots the estimated state-dependent distributions.

Usage

```
plotMod(mod)
```

Arguments

mod	An object of type countHMM (as returned by the function fitMod).

Value

A plot of the estimated state-dependent distributions.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# model fitting
lambda = rep(10^4,2)
mod = fitMod(x=x,lambda=lambda)
# plotting the estimated state-dependent distributions
plotMod(mod)
```

plotObs *plotObs*

Description

Plots the Viterbi-decoded time series using different colors for the different states.

Usage

```
plotObs(mod)
```

Arguments

mod	An object of type countHMM (as returned by the function fitMod).

Value

A plot of the Viterbi-decoded time series.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# model fitting
lambda = rep(10^4,2)
mod = fitMod(x=x,lambda=lambda)
# plotting the Viterbi-decoded time series
plotObs(mod)
```

plotRes *Quantile-quantile and autocorrelation function plots of the pseudo-residuals.*

Description

Plots quantile-quantile and autocorrelation function plots of the midpoints of the pseudo residual segments. For details, see *Zucchini et al. (2016).*

Usage

```
plotRes(mod)
```

Arguments

mod	An object of type countHMM (as returned by the function fitMod).

Value

A plot with two windows, the first of which displays the quantile-quantile function and the second of which displays the autocorrelation function of the pseudo-residuals.

References

Zucchini W., MacDonald, I.L., and Langrock, R. (2016): *Hidden Markov models for time series: An introduction using R, 2nd edition*. Chapman & Hall/CRC, Boca Raton.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# model fitting
lambda = rep(10^4,2)
mod = fitMod(x=x,lambda=lambda)
# plotting the pseudo residuals
plotRes(mod)
```

---

pn2pw *pn2pw*

Description

Transforming natural parameters to working parameters. Not intended to be run by the user (internal function, called by the functions `nLogLike` and `fitMod`).

Usage

```
pn2pw(N,probs,gamma,delta,stationary)
```

Arguments

`N`	Number of states.
`probs`	Matrix with `N` columns which contains the state-dependent probabilities for each count.
`gamma`	Transition probability matrix of the Markov chain underlying the observed counts with `N` rows and `N` columns.
`delta`	Initial distribution vector of length `N` of the Markov chain underlying the observed counts. If `NULL`, then the stationary distribution is returned.
`stationary`	Logical, determines whether the initial distribution of the Markov chain underlying the observed counts is the stationary distribution.

Value

A vector of working parameters.

Examples

```
# transforming natural parameters to working paramters
pn2pw(N=2,probs=cbind(dpois(x=0:41,lambda=14),dpois(x=0:41,lambda=26)),
gamma=matrix(c(0.95,0.05,0.05,0.95),ncol=2),delta=NULL,stationary=TRUE)
```

---

psRes *psRes*

Description

Computes the pseudo-residuals.

Usage

```
psRes(mod)
```

Arguments

mod	An object of type `countHMM` (as returned by the function `fitMod`).

Value

A matrix with 3 rows, the first of which corresponds to the lower limits, the second of which corresponds to the midpoints, and the third of which corresponds to the upper limits of the pseudo-residual segments. For details, see Zucchini *et al*. (2016).

References

Zucchini W., MacDonald, I.L., and Langrock, R. (2016): Hidden Markov Models for Time Series: An Introduction Using R, 2nd Edition. Chapman & Hall/CRC. doi:https://doi.org/10.1201/b20790.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# model fitting
lambda = rep(10^4,2)
mod = fitMod(x=x,lambda=lambda)
# computing the pseudo-residuals
psRes(mod)
```

pw2pn *pw2pn*

Description

Transforming working parameters to natural parameters. Not intended to be run by the user (internal function, called by the functions `nLogLike` and `fitMod`).

Usage

```
pw2pn(N,parvect,stationary,sup)
```

Arguments

`N`	Integer, number of states.
`parvect`	Vector of working parameters.
`stationary`	Logical, determines whether the initial distribution of the Markov chain underlying the observed counts is the stationary distribution.
`sup`	Integer, determines the upper bound of the support of the state-dependent distributions.

Value

A list of natural parameters.

Examples

```
# transforming natural parameters to working paramters
parvect = pn2pw(N=2,probs=cbind(dpois(x=0:41,lambda=14),dpois(x=0:41,lambda=26)),
gamma=matrix(c(0.95,0.05,0.05,0.95),ncol=2),delta=NULL,stationary=TRUE)
# transforming working parameters to natural parameters
pw2pn(N=2,parvect=parvect,stationary=TRUE,sup=41)
```

stateDec *stateDec*

Description

Decodes the states of the Markov chain underlying the observed time series of counts using the Viterbi algorithm. For details, see Zucchini *et al*. (2016).

Usage

```
stateDec(mod)
```

Arguments

mod	An object of type countHMM (as returned by the function fitMod).

Value

A vector of Viterbi-decoded states.

References

Zucchini W., MacDonald, I.L., and Langrock, R. (2016): Hidden Markov Models for Time Series: An Introduction Using R, 2nd Edition. Chapman & Hall/CRC. doi:https://doi.org/10.1201/b20790.

Examples

```
# importing example data
x = read.table("http://www.hmms-for-time-series.de/second/data/earthquakes.txt")$V2
# model fitting
lambda = rep(10^4,2)
mod = fitMod(x=x,lambda=lambda)
# decoding the states
stateDec(mod)
```

Index