Package ‘mhsmm’

February 20, 2015

Type Package
Title Inference for Hidden Markov and Semi-Markov Models
Version 0.4.14
Date 2014-01-08
Author Jared O'Connell <jaredoconnell@gmail.com>, Søren Højsgaard
         <sorenh@math.aau.dk>
Maintainer Jared O'Connell <jaredoconnell@gmail.com>
Description Parameter estimation and prediction for hidden Markov and semi-Markov models for data with multiple observation sequences. Suitable for equidistant time series data, with multivariate and/or missing data. Allows user defined emission distributions.
License GPL (>= 2)
LazyLoad yes
Depends mvtnorm
Repository CRAN
Encoding latin1
NeedsCompilation yes
Date/Publication 2014-12-22 10:37:39

R topics documented:

  addStates .................................................. 2
  dmvnorm.hsmm ........................................... 3
  dnorm.hsmm ............................................. 4
  dpois.hsmm ............................................ 5
  gammafit ................................................. 6
  hmmfit ................................................... 7
  hmmmpec .................................................. 8
  hsmmmfit ............................................... 9
  hsmmspec .............................................. 11
  mstep.mvnorm ........................................... 12
  mstep.norm ............................................. 13
addStates

Add a bar representing state sequence.

Description

Add a colour coded horizontal bar representing the state sequence to a plot of (presumably time-series) data.

Usage

```r
addStates(states, x=NULL, ybot = axTicks(2)[1],
          ytop = ybot + (axTicks(2)[2] - axTicks(2)[1]) / 5,
          dy = ytop - ybot,
          greyscale = FALSE, leg = NA, J = length(unique(states)),
          time.scale = 1, shifthx = 0)
```

Arguments

- **states**: A vector of integers representing the states traversed.
- **x**: The time values where the states are observed \((1:length(states) - shifthx)/time.scale\) if NULL.
- **ybot**: Vertical bottom limit of the bar.
- **ytop**: Vertical top limit of the bar.
dmvnorm.hsmm


dy Height of the bar.
greyscale If TRUE produces a bar in greyscale.
leg Array of state names, if present, produces a legend.
J Number of states
time.scale Resolution of the timescale
shiftx Shift the bar forward or backwards horizontal by shiftx distance.

Author(s)

Soren Hojsgaard sorenh@math.aau.dk

See Also

addStates

Examples

plot(rnorm(100),type='l')
addStates(rep(c(1,2),each=50))

plot(seq(0.01,1,.01),rnorm(100),type='l')
addStates(rep(c(1,2),each=50),seq(0.01,1,.01))

______________________________
| dmvnorm.hsmm | Emission ensity function for a multivariate normal emission distribution |
______________________________

Description

Calculates the density of observations x for state j given the parameters in model. This is used for a multivariate Gaussian emission distribution of a HMM or HSMM and is a suitable prototype for user's to make their own custom distributions.

Usage

dmvnorm.hsmm(x, j, model)

Arguments

x Observed value
j State
model A hsmmspec or hmmspec object

Details

This is used by hmm and hsmm to calculate densities for use in the E-step of the EM algorithm. It can also be used as a template for users wishing to building their own emission distributions.
dnorm.hsmm

Value

A vector of probability densities.

Author(s)

Jared O’Connell jaredoconnell@gmail.com

See Also

mstep.mvnorm, rmvnorm.hsmm

Examples

```r
J <- 2
initial <- rep(1/J, J)
P <- matrix(c(3, 5, 7, 5), nrow = J)
b <- list(mu = list(c(-3, 0), c(1, 2)), sigma = list(diag(2), matrix(c(4, 2, 2, 3), ncol = 2)))
model <- hsmmspec(init = initial, trans = P, parms.emission = b, dens.emission = dmvnorm.hsmm)
model
train <- simulate(model, nsim = 300, seed = 1234, rand.emis = rmvnorm.hsmm)
plot(train, xlim = c(0, 100))
h1 <- hmmfit(train, model, mstep = mstep.mvnorm)
```

Description

Calculates the density of observations x for state j given the parameters in model. This is used for the Gaussian emission distribution of a HMM or HSMM and is a suitable prototype for user’s to make their own custom distributions.

Usage

```r
dnorm.hsmm(x, j, model)
```

Arguments

- **x**: Observed value
- **j**: State
- **model**: A hsmmspec or hmmspec object

Details

This is used by hmm and hsmm to calculate densities for use in the E-step of the EM algorithm. It can also be used as a template for users wishing to building their own emission distributions.
**dpois.hsmm**

**Value**

A vector of probability densities.

**Author(s)**

Jared O'Connell jaredoconnell@gmail.com

---

**dpois.hsmm**

*Emission density function for Poisson emission distribution*

**Description**

Calculates the density of observations $x$ for state $j$ given the parameters in model. This is used for a Poisson emission distribution of a HMM or HSMM and is a suitable prototype for user’s to make their own custom distributions.

**Usage**

`dpois.hsmm(x, j, model)`

**Arguments**

- **x** Observed value
- **j** State
- **model** A `hsmmspec` or `hmmspec` object

**Details**

This is used by `hmm` and `hsmm` to calculate densities for use in the E-step of the EM algorithm. It can also be used as a template for users wishing to building their own emission distributions.

**Value**

A vector of probability densities.

**Author(s)**

Jared O'Connell jaredoconnell@gmail.com

**See Also**

`mstep.pois`, `rpois.hsmm`
gammafit

Parameter estimation for the Gamma distribution

Summary

Estimates parameters for the Gamma distribution using the Method of Maximum Likelihood, works with weighted data.

Usage

\[ \text{gammafit}(x, \text{wt} = \text{NULL}) \]

Arguments

- \( x \): A vector of observations
- \( \text{wt} \): Optional set of weights

Value

- \( \text{shape} \): The shape parameter
- \( \text{scale} \): The scale parameter (equal to 1/rate)

Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


Examples

\[ \text{gammafit}(\text{rgamma}(1000, \text{shape}=10, \text{scale}=13)) \]
hmmfit

**hmmfit**  
*fit a hidden Markov model*

**Description**

Estimates parameters of a HMM using the EM algorithm.

**Usage**

```r
hmmfit(x, start.val, mstep=mstep.norm, lock.transition=FALSE, tol=1e-08, maxit=1000)
```

**Arguments**

- `x`  
  A hsmm.data object (see Details)
- `start.val`  
  Starting parameters for the model (see Details)
- `mstep`  
  Re-estimates the parameters of density function on each iteration
- `lock.transition`  
  If TRUE will not re-estimate the transition matrix
- `maxit`  
  Maximum number of iterations
- `tol`  
  Convergence tolerance

**Value**

- `start`  
  A vector of the starting probabilities for each state
- `a`  
  The transition matrix of the embedded Markov chain
- `emission`  
  A list of the parameters of the emission distribution

**Author(s)**

Jared O’Connell jaredoconnell@gmail.com

**References**


**See Also**

- `predict.hmm`
Examples

```r
J <- 3
initial <- rep(1/J, J)
P <- matrix(c(.8,.5,.1,.05,.2,.5,.15,.3,.4), nrow=J)
b <- list(mu=c(-3,0,2), sigma=c(2,1,.5))
model <- hmmspec(init=initial, trans=P, parms.emission=b, dens.emission=dnorm.hsmm)

train <- simulate(model, nsim=300, seed=1234, rand.emis=rnorm.hsmm)
plot(train, xlim=c(0,100))

init0 <- rep(1/J, J)
P0 <- matrix(1/J, nrow=J, ncol=J)
b0 <- list(mu=c(-3,1,3), sigma=c(1,1,1))
startval <- hmmspec(init=init0, trans=P0, parms.emission=b0, dens.emission=dnorm.hsmm)
h1 <- hmmfit(train, startval, mstep=mstep.norm)

plot(h1$loglik, type='b', ylab='Log-likelihood', xlab='Iteration')
summary(h1)

# proportion of incorrect states
mean(train$s1 != predict(h1, train)$s)

# simulate a new test set
test <- simulate(model, nsim=c(100,200,300), seed=1234, rand.emis=rnorm.hsmm)
mean(test$s1 != predict(h1, test)$s)
```

## hmmspec

### Specification of HMMs

**Description**

Creates a model specficiation for a hidden Markov model.

**Usage**

`hmmspec(init, trans, parms.emission, dens.emission, rand.emission=NULL, mstep=NULL)`

**Arguments**

- `init`: Distribution of states at t=1 ie. P(S=s) at t=1
- `trans`: The transition matrix of the Markov chain
- `parms.emission`: A list containing the parameters of the emission distribution
- `dens.emission`: Density function of the emission distribution.
- `rand.emission`: The function used to generate observations from the emission distribution
- `mstep`: Re-estimates the parameters of density function on each iteration
**hsmmfit**

**Value**

A hmspec object

**Author(s)**

Jared O'Connell jaredoconnell@gmail.com

**References**


**See Also**

simulate.hmspec, simulate.hmspec, hmmfit, predict.hmm

---

**hsmmfit**

*fit a hidden semi-Markov model*

**Description**

Estimates parameters of a HSMM using the EM algorithm.

**Usage**

```r
hsmmfit(x, model, mstep=NULL, M=NA, maxit=100, lock.transition=FALSE, lock.d=FALSE, graphical=FALSE)
```

**Arguments**

- `x` : A hsmm.data object (see Details)
- `model` : Starting parameters for the model (see hmspec)
- `mstep` : Re-estimates the parameters of density function on each iteration
- `maxit` : Maximum number of iterations
- `M` : Maximum number of time spent in a state (truncates the waiting distribution)
- `lock.transition` : If TRUE will not re-estimate the transition matrix
- `lock.d` : If TRUE will not re-estimate the sojourn time density
- `graphical` : If TRUE will plot the sojourn densities on each iteration
Value

- **start**: A vector of the starting probabilities for each state
- **a**: The transition matrix of the embedded Markov chain
- **emission**: A list of the parameters of the emission distribution
- **waiting**: A list of the parameters of the waiting distribution

Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


See Also

- `hsmmspec`, `simulate.hsmmspec`, `predict.hsmm`

Examples

```r
J <- 3
d <- 3
init <- c(0, 0, 1)
P <- matrix(c(0.1, 0.5, 0.6, 0.5, 0.9, 0), nrow = J)
B <- list(mu = c(10, 15, 20), sigma = c(2, 1, 1.5))
d <- list(lambda = c(10, 30, 60), shift = c(10, 100, 30), type = 'poisson')
model <- hsmmspec(init = P, parms = emission = B, sojourn = d, dens = emission = dnorm, hmm)
train <- simulate(model, r = rnorm, hmm, nsim = 100, seed = 123456)
plot(train, xlim = c(0, 400))
start.poi <- hsmmspec(init = rep(1/J, J),
  transition = matrix(c(0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5), nrow = J),
  parms = emission = list(mu = c(4, 12, 23),
    sigma = c(1, 1, 1)),
  sojourn = list(lambda = c(9, 25, 40), shift = c(5, 95, 45), type = 'poisson'),
  dens = emission = dnorm, hmm)

M <- 500
# not run (takes some time)
# h.poisson <- hsmmfit(train, start.poisson, mstep = mstep.norm, M = M)
# plot(h.poisson$loglik, type = 'b', ylab = 'Log-likelihood', xlab = 'Iteration') ## has it converged?
# summary(h.poisson)
# predicted <- predict(h.poisson, train)
# table(train$s, predicted$s) ## classification matrix
# mean(predicted$s != train$s) ## misclassification rate
d <- cbind(dunif(1:M, 0, 50), dunif(1:M, 100, 175), dunif(1:M, 50, 130))
start.np <- hsmmspec(init = rep(1/J, J),
```
hsmmspec

Hidden semi-Markov model specification

Description

Creates a model specification of a hidden semi-Markov model.

Usage

hsmmspec(init, transition, parms.emission, sojourn, dens.emission,
         rand.emission=NULL, mstep=NULL)

Arguments

init Distribution of states at t=1 ie. P(S=s) at t=1
transition The transition matrix of the embedded Markov chain (diagonal must be 0)
parms.emission A list containing the parameters of the emission distribution
sojourn A list containing the parameters and type of sojourn distribution (see Details)
dens.emission Density function of the emission distribution
rand.emission The function used to generate observations from the emission distribution
mstep Re-estimates the parameters of density function on each iteration
Details

The sojourn argument provides a list containing the parameters for the available sojourn distributions. Available sojourn distributions are shifted Poisson, Gamma and non-parametric.

In the case of the Gamma distribution, sojourn is a list with vectors shape and scale (the Gamma parameters in dgamma), both of length J. Where J is the number of states. See reprocows for an example using Gamma sojourn distributions.

In the case of shifted Poisson, sojourn is list with vectors shift and lambda, both of length J. See hsmmf for an example using shifted Poisson sojourn distributions.

In the case of non-parametric, sojourn is a list containing a M x J matrix. Where entry (i,j) is the probability of a sojourn of length i in state j. See hsmmf for an example using shifted non-parametric sojourn distributions.

Value

An object of class hsmmspec

Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


See Also

hsmmf, simulate.hsmmspec, predict.hsmm

Description

Re-estimates the parameters of a multivariate normal emission distribution as part of the EM algorithm for HMMs and HSMMs. This is called by the hmm and hsmm functions. It is a suitable prototype function for users wishing to design their own emission distributions.

Usage

mstep.mvnorm(x, wt)
mstep.norm

Arguments

x   A vector of observed values
wt  A T x J matrix of weights. Column entries are the weights for respective states.

Details

Users may write functions that take the same arguments and return the same values for their own custom emission distributions.

Value

Returns the emission slot of a hmspec or hsmmspec object

mu  A list of length J contain the mean vectors
sigma A list of length J containing the covariance matrices

Author(s)

Jared O'Connell jaredoconnell@gmail.com

See Also

dmvnorm.hsm, rmvnorm.hsm

Examples

J<2
initial <- rep(1/J,J)
P <- matrix(c(.3,.5,.7,.5),nrow=J)
b <- list(mu=list(c(-3,0),c(1,2)),sigma=list(diag(2),matrix(c(4,2,2,3), ncol=2)))
model <- hmspec(init=initial, trans=P, parms.emission=b, dens.emission=dmvnorm.hsm)
model
train <- simulate(model, nsim=300, seed=1234, rand.emis=rmvnorm.hsm)
plot(train,xlim=c(0,100))
h1 = hmmfit(train,model,mstep=mstep.mvnorm)

mstep.norm

Performs re-estimation (the M-step) for a normal emission distribution

Description

Re-estimates the parameters of a normal emission distribution as part of the EM algorithm for HMMs and HSMMs. This is called by the hmm and hsmm functions. It is a suitable prototype function for users wishing to design their own emission distributions.

Usage

mstep.norm(x, wt)
mstep.pois

Arguments

x A vector of observed values
wt A T x J matrix of weights. Column entries are the weights for respective states.

Details

Users may write functions that take the same arguments and return the same values for their own custom emission distributions.

Value

Returns the emission slot of a hmmspec or hsmmspec object

mu Vector of length J containing the means
sigma Vector of length J containing the variances

Author(s)

Jared O’Connell jaredoconnell@gmail.com

mstep.pois Performs re-estimation (the M-step) for a Poisson emission distribution

Description

Re-estimates the parameters of a Poisson emission distribution as part of the EM algorithm for HMMs and HSMMs. This is called by the hmm and hsmm functions. It is a suitable prototype function for users wishing to design their own emission distributions.

Usage

mstep.pois(x, wt)

Arguments

x A vector of observed values
wt A T x J matrix of weights. Column entries are the weights for respective states.

Details

Users may write functions that take the same arguments and return the same values for their own custom emission distributions.

Value

Returns the emission slot of a hmmspec or hsmmspec object

lambda Vector of length J containing the Poisson parameters for each state j
Author(s)
Dared O’Connell jaredoconnell@gmail.com

See Also
rpois.hsmm, dpois.hsmm

Examples

J <- 3
initial <- rep(1/J,J)
P <- matrix(c(.8,.5,.1,.05,.2,.5,.15,.3,.4),nrow=J)
b <- list(lambda=c(1,3,6))
model <- hmspec(init=initial, trans=P, parms.emission=b,dens.emission=dpois.hsmm)
model
train <- simulate(model, nsim=300, seed=1234, rand.emis=rpois.hsmm)
plot(train, xlim=c(0,100))
h1 <- hmmsfit(train, model, mstep=mstep.pois)

plot.hsmm

Plot function for hsmms

Description
Displays the densities for the sojourn distributions of each state.

Usage

## S3 method for class 'hsmm'
plot(x, ...)

Arguments

x
A hsmm object

... Arguments passed to plot

Author(s)
Jared O’Connell jaredoconnell@gmail.com
plot.hsmm.data

Plot function for hsmm data

Description

Produces a plot of the observed sequences, and displays a coloured bar signifying the hidden states (if available)

Usage

```r
## S3 method for class 'hsmm.data'
plot(x, ...)
```

Arguments

- `x`: A hsmm.data object
- `...`: Arguments passed to `plot.ts`

Author(s)

Jared O’Connell jaredoconnell@gmail.com

See Also

- `addStates`

Examples

```r
J<-3
initial <- rep(1/J,J)
P <- matrix(c(.8,.5,.1,.05,.2,.5,.15,.3,.4),nrow=J)
b <- list(mu=c(-3,0,2),sigma=c(2,1,5))
model <- hmmSspec(init=initial, trans=P, parms.emission=b, dens.emission=dnorm.hsmm)

train <- simulate(model, nsim=300, seed=1234, rand.emis=rnorm.hsmm)
plot(train, xlim=c(0,100))
```

predict.hmm

Prediction function for hmm

Description

Predicts the underlying state sequence for an observed sequence `newdata` given a `hmm` model
predict.hmm

Usage

## S3 method for class 'hmm'
predict(object, newdata, method = "viterbi", ...)

Arguments

- object: An object of class hmm
- newdata: A vector or data.frame of observations
- method: Prediction method (see details)
- ...: further arguments passed to or from other methods.

Details

If method="viterbi", this technique applies the Viterbi algorithm for HMMs, producing the most likely sequence of states given the observed data. If method="smoothed", then the individually most likely (or smoothed) state sequence is produced, along with a matrix with the respective probabilities for each state.

Value

Returns a hsmm.data object, suitable for plotting.

- newdata: A vector or data.frame of observations
- s: A vector containing the reconstructed state sequence
- N: The lengths of each sequence
- p: A matrix where the rows represent time steps and the columns are the probability for the respective state (only produced when method="smoothed")

Author(s)

Jared O'Connell jaredoconnell@gmail.com

References


See Also

hmmfit, hmspec

Examples

##See examples in 'hmmfit'
predict.hmmspec  

*Prediction function for hmmspec*

**Description**
Predicts the underlying state sequence for an observed sequence `newdata` given a `hmmspec` model.

**Usage**
```r
## S3 method for class 'hmmspec'
predict(object, newdata, method = "viterbi", ...)
```

**Arguments**
- `object`: An object of class `hmm`.
- `newdata`: A vector or data.frame of observations.
- `method`: Prediction method (see details).
- `...`: Further arguments passed to or from other methods.

**Details**
If `method="viterbi"`, this technique applies the Viterbi algorithm for HMMs, producing the most likely sequence of states given the observed data. If `method="smoothed"`, then the individually most likely (or smoothed) state sequence is produced, along with a matrix with the respective probabilities for each state. This function differs from `predict.hmm` in that it takes the output from `hmmspec` ie. this is useful when users already know their parameters and wish to make predictions.

**Value**
Returns a `hmm.data` object, suitable for plotting.
- `newdata`: A vector or data.frame of observations.
- `s`: A vector containing the reconstructed state sequence.
- `N`: The lengths of each sequence.
- `p`: A matrix where the rows represent time steps and the columns are the probability for the respective state (only produced when `method="smoothed"`).

**Author(s)**
Jared O'Connell jaredoconnell@gmail.com

**References**
predict.hsmm

See Also

hmmspec

Examples

J <- 3
initial <- rep(1/J, J)
P <- matrix(c(.8,.5,.1,0.05,.2,.5,.15,.3,.4), nrow = J)
b <- list(mu = c(-3, 0, 2), sigma = c(2, 1, .5))
model <- hmmspec(init = initial, trans = P, parms.emission = b, dens.emission = dnorm.hsmm)
train <- simulate(model, nsi = 300, seed = 1234, rand.emis = rnorm.hsmm)
mean(predict(model, train)$s! = train$s) # error rate when true model is known

predict.hsmm  Prediction for hsmms

Description

Predicts the underlying state sequence for an observed sequence newdata given a hsmm model

Usage

## S3 method for class 'hsmm'
predict(object, newdata, method = "viterbi", ...)

Arguments

object An object of type hsmm
newdata A vector or dataframe of observations
method Prediction method (see details)
... further arguments passed to or from other methods.

Details

If method = "viterbi", this technique applies the Viterbi algorithm for HSMMs, producing the most likely sequence of states given the observed data. If method = "smoothed", then the individually most likely (or smoothed) state sequence is produced, along with a matrix with the respective probabilities for each state.

Value

Returns a hsmm.data object, suitable for plotting.

newdata A vector or dataframe of observations
s A vector containing the reconstructed state sequence
N The lengths of each sequence
p A matrix where the rows represent time steps and the columns are the probability for the respective state (only produced when method = "smoothed")
Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


See Also

hsmmfit,predict.hsmmspec

Examples

## See 'hsmmfit' for examples

predict.hsmmspec  Prediction for hsmmspec

Description

Predicts the underlying state sequence for an observed sequence newdata given a hsmm model

Usage

## S3 method for class 'hsmmspec'
predict(object, newdata, method = "viterbi", M=NA, ...)

Arguments

object  An object of type hsmmspec
newdata A vector or dataframe of observations
method  Prediction method (see details)
M  Maximum number of time spent in a state (truncates the waiting distribution)
...  further arguments passed to or from other methods.

Details

If method="viterbi", this technique applies the Viterbi algorithm for HSMMs, producing the most likely sequence of states given the observed data. If method="smoothed", then the individually most likely (or smoothed) state sequence is produced, along with a matrix with the respective probabilities for each state. This method is different to predict.hsmm in that it takes the output from hsmmspec as input ie. it is useful for people who already know their model parameters.
Value

Returns a hsmm.data object, suitable for plotting.

- **newdata**: A vector or data.frame of observations
- **s**: A vector containing the reconstructed state sequence
- **N**: The lengths of each sequence
- **p**: A matrix where the rows represent time steps and the columns are the probability for the respective state (only produced when method="smoothed")

Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


See Also

hsmmspec, predict.hsmm

Examples

```r
J <- 3
init <- c(0,0,1)
P <- matrix(c(0,1,.4,.5,0,.6,.5,.9,0), nrow=J)
B <- list(mu=c(10,15,20), sigma=c(2,1,1.5))
d <- list(lambda=c(10,30,60), shift=c(10,100,30), type='poisson')
model <- hsmmspec(init,P,parms.emission=B,sojourn=d,dens.emission=dnorm.hsmm)
train <- simulate(model,r=rnorm.hsmm,nsim=100,seed=123456)
mean(predict(model,train,M=500)$s=train$s) # error rate when true model is known
```

print.hmm

Print method for hmm objects

Description

Prints the slots of a hmm object

Usage

```
## S3 method for class 'hmm'
print(x, ...)
```
Arguments

x An object of type hmm

... further arguments passed to or from other methods.

Author(s)

Jared O'Connell jaredoconnell@gmail.com

Description

Prints the parameters contained in the object

Usage

## S3 method for class 'hmmspec'
print(x, ...)

Arguments

x An object of type hmmspec

... further arguments passed to or from other methods.

Author(s)

Jared O'Connell jaredoconnell@gmail.com

Description

Prints the parameters contained in the object

Usage

## S3 method for class 'hmmspec'
print(x, ...)

Arguments

x An object of type hmmspec

... further arguments passed to or from other methods.
**reproai**

**Author(s)**

Jared O’Connell jaredoconnell@gmail.com

---

**Description**

This is an auxiliary data set to the `cows` data set containing times of artificial insemination for respective cows. Only the day of insemination was recorded so time of day is always midday.

**Usage**

`reproai`

**Format**

`reproai` is a dataframe with 12 rows and `id` being the cow’s id and `daysNfromNcalving` recording the number of days from calving when insemination occurred.

**Source**

Danish Cattle Research Centre

**References**


---

**reprocows**

**Reproductive data from seven dairy cows**

**Description**

This data set contains hourly observations on progesterone and an activity index at hourly intervals since calving on seven dairy cows.

**Usage**

`reprocows`
reprocows

Format

reprocows is a data frame containing 13040 rows. id is the cow ID, progesterone is a measurement of the hormone in ng/L taken from a milk sample, activity is a relative measure of activity calculated from a pedometer.

There are a large number of missing values as progesterone is measured only at milking time (and at a farm manager’s discretion). Missing values in activity occur due to hardware problems can occur with pedometers.

Source

Danish Cattle Research Centre

References


Examples

data(reprocows)
data(reproai)
data(reproppa)
tm = 1600

J <- 3
init <- c(1,0,0)
trans <- matrix(c(0,0,0,1,0,1,0),nrow=J)
emis <- list(mu=c(0,2.5,0),sigma=c(1,1,1))

N <- as.numeric(table(reprocows$id))
train <- list(x=reprocows$activity,N=N)
class(train) <- "hsmm.data"
tmp <- gammafit(reproppa * 24)
M <- max(N)

d <- cbind(dgamma(1:M,shape=tmp$shape,scale=tmp$scale),
    # ppa sojourn directly estimated from ppa data set
dunif(1:M,4,30),
    # oestrus between 4 and 30 hours
    dunif(1:M,15*24,40*24))
    #not-oestrus between 15 and 40 days

startval <- hsmmspec(init,trans,parms.emission=emis.list(d=d,type='gamma'),
    dens.emission=dnorm,hsmm)
#not run (takes some time)
#h.activity <- hmmfit(train,startval,mstep=mstep.norm,maxit=10,M=M,lock.transition=TRUE)
reproppa

Observed lengths of post-partum anoestrus for 73 dairy cows

Description
This data set contains the observed length of post-partum anoestrus (in days) for 73 dairy cattle.

Usage
reproppa

Format
reproppa a vector containing 73 integers.

Source
Danish Cattle Research Centre

References

rmvnormNhsmm
Random number generation from a multivariate normal distributed emission distribution

Description
This generates values from a multivariate normal distributed emission state \( j \) given parameters in \( \text{model} \).

Usage
rmvnormNhsmm(j, \text{model})

Arguments
\( j \) An integer representing the state
model A hmspec or hsmmspec object

Details
This is essentially a wrapper for \( \text{rnorm} \). Users may build functions with the same arguments and return values so they can use their own custom emission distributions.
Value

A single value from the emission distribution.

Author(s)

Jared O’Connell jaredoconnell@gmail.com

See Also

dmvnorm.hsmm, mstep.mvnorm

Examples

```r
J <- 2
initial <- rep(1/J, J)
P <- matrix(c(3, 5, 7, 9), nrow=J)
b <- list(mu=list(c(-3, 0), c(1, 2)), sigma=list(diag(2), matrix(c(4, 2, 2, 3), ncol=2)))
model <- hsmmspec(init=initial, trans=P, parms.emission=b, dens.emission=dmvnorm.hsmm)
model
train <- simulate(model, nsim=300, seed=1234, rand.emis=rmvnorm.hsmm)
plot(train, xlim=c(0, 100))
h1 <- hmmfit(train, model, mstep=mstep.mvnorm)
```

rnorm.hsmm

Random number generation from a normally distributed emission distribution

Description

This generates values from a normally distributed emission state \( j \) given parameters in \( \text{model} \).

Usage

\[
\text{rnorm.hsmm}(j, \text{model})
\]

Arguments

\( j \)  
An integer representing the state

\( \text{model} \)  
A hsmmspec or hsmmspec object

Details

This is essentially a wrapper for \( \text{rnorm} \). Users may build functions with the same arguments and return values so they can use their own custom emission distributions.

Value

A single value from the emission distribution.
**rpois.hsmm**

*Random number generation from a Poisson distributed emission distribution*

---

**Description**

This generates values from a Poisson distributed emission state \( j \) given parameters in \( \text{model} \).

**Usage**

```
rpois.hsmm(j, model)
```

**Arguments**

- `j` An integer representing the state
- `model` A `hmmspec` or `hsmmspec` object

**Details**

This is essentially a wrapper for `rpois`. Users may build functions with the same arguments and return values so they can use their own custom emission distributions.

**Value**

A single value from the emission distribution.

**Author(s)**

Jared O’Connell jaredoconnell@gmail.com

**See Also**

`mstep.pois`, `dpois.hsmm`

**Examples**

```r
J <- 3
initial <- rep(1/J, J)
P <- matrix(c(.8,.5,.1,.05,.2,.5,.15,.3,.4), nrow = J)
b <- list(lambda = c(1,3,6))
model <- hmmspec(init = initial, trans = P, parms.emission = b, dens.emission = dpois.hsmm)
model
train <- simulate(model, nsim = 300, seed = 1234, rand.emis = rpois.hsmm)
plot(train, xlim = c(0, 100))
h1 <- hmmfit(train, model, mstep = mstep.pois)
```
**sim.mc**  
*Markov chain simulation*

**Description**  
Simulates a Markov chain

**Usage**  
sim.mc(init, transition, N)

**Arguments**
- **init**  
The distribution of states at the first time step
- **transition**  
The transition probability matrix of the Markov chain
- **N**  
The number of observations to simulate

**Value**  
A vector of integers representing the state sequence.

**Author(s)**
Jared O'Connell jaredoconnell@gmail.com

**Examples**
```
p <- matrix(c(1,.3,.6,rep(1/3,3),0,.5,.5),ncol=3,byrow=TRUE)
init <- rep(1/3,3)
sim.mc(init,p,10)
```
Arguments

object  A hmm spec object
nsim   An integer or vector of integers (for multiple sequences) specifying the length of the sequence(s)
seed   seed for the random number generator
rand.emission  The function used to generate observations from the emission distribution
...  further arguments passed to or from other methods.

Details

If nsim is a single integer then a HMM of that length is produced. If nsim is a vector of integers, then length(nsim) sequences are generated with respective lengths.

Value

An object of class hmmdata

x  A vector of length \( \sum(n) \) - the sequence(s) of observed values
s  A vector of length \( \sum(n) \) - the sequence(s) of hidden states
N  A vector of the length of each observation sequence (used to segment x and s)

Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


See Also

hmm spec, link{predict.hmm}

Examples

``` R
j<-3
initial <- rep(1/J,J)
P <- matrix(c(.8,.5,.1,.05,.2,.5,.15,.3,.4),nrow=j)
b <- list(mu=c(-3,0,2),sigma=c(2,1,.5))
model <- hmm spec(init=initial, trans=P, parms.emission=b,dens.emission=dnorm,hsmm)
train <- simulate(model, nsim=100, seed=1234, rand.emis=rnorm,hsmm)
plot(train)
```
Description

Simulates values for a specified hidden semi-Markov model

Usage

```r
# S3 method for class 'hsmmspec'
simulate(object, nsim, seed = NULL, rand.emission = NULL, ...)
```

Arguments

- `object`: A `hsmmspec` object
- `nsim`: An integer or vector of integers (for multiple sequences) specifying the length of the sequence(s)
- `seed`: seed for the random number generator
- `rand.emission`: The function used to generate observations from the emission distribution
- `...`: further arguments passed to or from other methods.

Details

If `nsim` is a single integer then a HSMM of that length is produced. If `nsim` is a vector of integers, then `length(nsim)` sequences are generated with respective lengths.

Value

An object of class `hmmdata`

- `x`: A vector of length `sum(N)` - the sequence(s) of observed values
- `s`: A vector of length `sum(N)` - the sequence(s) of hidden states
- `N`: A vector of the length of each observation sequence (used to segment `x` and `s`)

Author(s)

Jared O’Connell jaredoconnell@gmail.com

References


See Also

`hsmmfit`, `hsmmspec`, `predict.hsmm`
smooth.discrete

Examples

J <- 3
init <- c(0,0,1)
P <- matrix(c(0.1,.5,0,.5,9,0),nrow=J)
B <- list(mu=c(10,15,20),sigma=c(2,1,1.5))
d <- list(lambda=c(10,30,60),shift=c(10,100,30),type='poisson')
model <- hsmmspec(init,P,parms.emission=B,sojourn=d,dens.emission=dnorm,hsmm)
train <- simulate(model,rand.emis=rnorm,hsmm,nsim=100,seed=123456)
plot(train,xlim=c(0,400))

smooth.discrete  Smoothing a discrete time series.

Description

The smooth.discrete() function provides a simple smoothing of a time series of discrete values measured at equidistant times. Under the hood of smooth.discrete() is a hidden Markov model.

Usage

smooth.discrete(y, init = NULL, trans = NULL, parms.emission = 0.5,
method = "viterbi", details = 0, ...)

Arguments

y          A numeric vector
init       Initial distribution (by default derived from data; see the vignette for details)
trans      Transition matrix (by default derived from data; see the vignette for details)
parms.emission Matrix describing the conditional probabilities of the observed states given the latent states. (See the vignette for details).
method     Either "viterbi" or "smoothed". The viterbi method gives the jointly most likely sequence; the smoothed method gives the sequence of individually most likely states.
details    Controlling the amount of information printed.
...        Further arguments passed on to the "hmmfit" function.

Details

The parameters are estimated using the Baum-Welch algorithm (a special case of the EM-algorithm).

Value

A list with the following components:
s         The "smoothed" states
model     The underlying hmm (hidden Markov model) object
data      The data
initial   The initial parameters
Summary method for hmm objects

Description

Prints the estimated parameters of a hmm object

Usage

## S3 method for class 'hmm'
summary(object, ...)

Arguments

object A hmm object

... further arguments passed to or from other methods.

Value

An object of class 'summary.hmm'

Author(s)

Søren Højsgaard <sorenh at agrsci.dk>

See Also

hmmspec, hmmfit

Examples

## Please see the vignette

---

summary.hmm

Summary method for hmm objects

---

Author(s)

Søren Højsgaard <sorenh at agrsci.dk>

See Also

hmmspec, hmmfit

Examples

## Please see the vignette

```r
# Please see the vignette
```
Summary function for hsmm

Description

Returns a summary object for a hsmm object

Usage

## S3 method for class 'hsmm'
summary(object, ...)

Arguments

- **object**: An object of type hsmm
- **...**: further arguments passed to or from other methods.

Author(s)

Jared O’Connell jaredoconnell@gmail.com
Index

*Topic datasets
  reproai, 23
  reprocows, 23
  reproppa, 25

*Topic models
  smooth.discrete, 31

addStates, 2, 16
createTransition (smooth.discrete), 31
dmvnorm.hsmm, 3, 13, 26
dnorm.hsmm, 4
dpois.hsmm, 5, 15, 27
gammafit, 6
hmmfit, 7, 9, 32
hmmspec, 8, 29, 32
hsmmfit, 9, 12, 20, 30
hsmspec, 10, 11, 21, 30
mstep.mvnorm, 4, 12, 26
mstep.norm, 13
mstep.pois, 5, 14, 27
plot.hsmm, 15
plot.hsmm.data, 16
predict.hmm, 7, 9, 16
predict.hmmspec, 18
predict.hsmm, 10, 12, 19, 21, 30
predict.hsmspec, 20, 20
predict.smoothDiscrete
  (smooth.discrete), 31
print.hmm, 21
print.hmmspec, 22
print.hsmspec, 22
print.smoothDiscrete (smooth.discrete), 31

reproai, 23
reprocows, 23
reproppa, 25
rmvnorm.hsmm, 4, 13, 25
rnorm.hsmm, 26
rpois.hsmm, 5, 15, 27
sim.mc, 28
simulate.hmmspec, 9, 28
simulate.hsmspec, 10, 12, 30
smooth.discrete, 31
summary.hmm, 32
summary.hsmspec, 33
summary.smoothDiscrete
  (smooth.discrete), 31