Package ‘ziphsmm’

May 22, 2018

Type Package

Title Zero-Inflated Poisson Hidden (Semi-)Markov Models

Version 2.0.6

Date 2018-05-28

Author Zekun (Jack) Xu, Ye Liu

Maintainer Zekun (Jack) Xu <zekunxu@gmail.com>

Description Fit zero-inflated Poisson hidden (semi-)Markov models with or without covariates by directly minimizing the negative log likelihood function using the gradient descent algorithm. Multiple starting values should be used to avoid local minima.

Depends R(>= 3.0.0)

License GPL

Imports Rcpp, pracma

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-05-22 10:15:03 UTC

R topics documented:

`CAT` *Pseudo activity counts (per minute) data for cats*

Description

Pseudo activity counts (per minute) data for cats

Usage

```
CAT
```

Format

A data frame with 4320 rows and 5 variables:

id cat ID: 1,2,3

hour hour of the day: 1,2,...,24

minute minute of the hour: 1,2,...,60

night night time indicator

activity activity count data

`convolution` *Convolution of two real vectors of the same length.*

Description

Convolution of two real vectors of the same length.

Usage

```
convolution(vec1, vec2)
```

Arguments

`vec1`	the first vector
`vec2`	the second vector

Value

a vector of full convolution

`dist_learn` *Distributed learning for a longitudinal continuous-time zero-inflated Poisson hidden Markov model, where zero-inflation only happens in State 1. Assume that priors, transition rates and state-dependent parameters can be subject-specific, clustered by group, or common. But at least one set of the parameters have to be common across all subjects.*

Description

Distributed learning for a longitudinal continuous-time zero-inflated Poisson hidden Markov model, where zero-inflation only happens in State 1. Assume that priors, transition rates and state-dependent parameters can be subject-specific, clustered by group, or common. But at least one set of the parameters have to be common across all subjects.

Usage

```
dist_learn(ylist, timelist, prior_init, tpm_init, emit_init, zero_init,
  yceil = NULL, rho = 1, priorclust = NULL, tpmclust = NULL,
  emitclust = NULL, zeroclust = NULL, group, maxit = 100, tol = 1e-04,
  ncores = 1, method = "Nelder-Mead", print = TRUE, libpath = NULL, ...)
```

Arguments

`ylist`	list of observed time series values for each subject
`timelist`	list of time indices
`prior_init`	a vector of initial values for prior probability for each state
`tpm_init`	a matrix of initial values for transition rate matrix
`emit_init`	a vector of initial values for the means for each poisson distribution
`zero_init`	a scalar initial value for the structural zero proportion
`yceil`	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
`rho`	tuning parameters in the distributed learning algorithm. Default to 1.
`priorclust`	a vector to specify the grouping for state prior. Default to NULL, which means no grouping.
`tpmclust`	a vector to specify the grouping for state transition rates. Default to NULL, which means no grouping.
`emitclust`	a vector to specify the grouping for Poisson means. Default to NULL, which means no grouping.
`zeroclust`	a vector to specify the grouping for structural zero proportions. Default to NULL, which means no grouping.
`group`	a list containing group information.
`maxit`	maximum number iteration. Default to 100.
`tol`	tolerance in the terms of the relative change in the norm of the common coefficients. Default to 1e-4.
`ncores`	number of cores to be used for parallel programming. Default to 1.
`method`	method for the distributed optimization in the ADMM framework.
`print`	whether to print each iteration. Default to TRUE.
`libpath`	path for the ziphsmm library if not the default set up. Default to NULL.
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 3(1), pp.1-122.

Examples

```
## Not run:
set.seed(930518)
nsubj <- 10
ns <- 5040
ylist <- vector(mode="list",length=nsubj)
timelist <- vector(mode="list",length=nsubj)
prior1 <- c(0.5,0.2 ,0.3 )
omega1 <- matrix(c(-0.3,0.2,0.1,
                  0.1,-0.2,0.1,
                  0.15,0.2,-0.35),3,3,byrow=TRUE)
prior2 <- c(0.3,0.3 ,0.4 )
omega2 <- matrix(c(-0.5,0.25,0.25,
                   0.2,-0.4,0.2,
                   0.15,0.3,-0.45),3,3,byrow=TRUE)
emit <- c(50,200,600)
zero <- c(0.2,0,0)
for(n in 1:nsubj){
 timeindex <- rep(1,ns)
 for(i in 2:ns) timeindex[i] <- timeindex[i-1] + sample(1:4,1)
 timelist[[n]] <- timeindex
 if(n<=5){
   result <- hmmsim.cont(ns, 3, prior1, omega1, emit, zero, timeindex)
   ylist[[n]] <- result$series
 }else{
   result <- hmmsim.cont(ns, 3, prior2, omega2, emit, zero, timeindex)
   ylist[[n]] <- result$series
 }
}
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(50, 225, 650)
zero_init <- 0.2
tpm_init <- matrix(c(-0.3,0.2,0.1,0.1,-0.2,0.1,0.15,0.2,-0.35),3,3,byrow=TRUE)
M <- 3
priorclust <- NULL
tpmclust <- c(1,1,1,1,1,2,2,2,2,2)
zeroclust <- rep(1,10)
emitclust <- rep(1,10)
group <- vector(mode="list",length=2)
group[[1]] <- 1:5; group[[2]] <- 6:10
result <- dist_learn(ylist, timelist, prior_init, tpm_init,
                    emit_init, zero_init,NULL, rho=1,priorclust,tpmclust,
                    emitclust,zeroclust,group,ncores=1,
                    maxit=50, tol=1e-4, method="CG", print=TRUE)

## End(Not run)
```

`dist_learn2` *Distributed learning for a longitudinal continuous-time zero-inflated Poisson hidden Markov model, where zero-inflation only happens in State 1 and covariates are for state-dependent zero proportion and means. Assume that priors, transition rates, state-dependent intercepts and slopes can be subject-specific, clustered by group, or common. But at least one set of the parameters have to be common across all subjects.*

Description

Distributed learning for a longitudinal continuous-time zero-inflated Poisson hidden Markov model, where zero-inflation only happens in State 1 and covariates are for state-dependent zero proportion and means. Assume that priors, transition rates, state-dependent intercepts and slopes can be subject-specific, clustered by group, or common. But at least one set of the parameters have to be common across all subjects.

Usage

```
dist_learn2(ylist, xlist, timelist, prior_init, tpm_init, emit_init, zero_init,
  yceil = NULL, rho = 1, priorclust = NULL, tpmclust = NULL,
  emitclust = NULL, zeroclust = NULL, slopeclust = NULL, group,
  maxit = 100, tol = 1e-04, ncores = 1, method = "Nelder-Mead",
  print = TRUE, libpath = NULL, ...)
```

Arguments

`ylist`	list of observed time series values for each subject
`xlist`	list of design matrices for each subject.
`timelist`	list of time indices
`prior_init`	a vector of initial values for prior probability for each state
`tpm_init`	a matrix of initial values for transition rate matrix
`emit_init`	a vector of initial values for the means for each poisson distribution
`zero_init`	a scalar initial value for the structural zero proportion
`yceil`	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
`rho`	tuning parameter in the distributed learning algorithm. Default to 1.
`priorclust`	a vector to specify the grouping for state prior. Default to NULL, which means no grouping.
`tpmclust`	a vector to specify the grouping for state transition rates. Default to NULL, which means no grouping.
`emitclust`	a vector to specify the grouping for the intercepts in Poisson regressions. Default to NULL, which means no grouping.
`zeroclust`	a vector to specify the grouping for the intercepts in ZIP regression. Default to NULL, which means no grouping.

`slopeclust`	a vector to specify the grouping for the slopes in Poisson and ZIP regressions. Default to NULL, which means no grouping.
`group`	a list containing group information.
`maxit`	maximum number iteration. Default to 100.
`tol`	tolerance in the terms of the relative change in the norm of the common coefficients. Default to 1e-4.
`ncores`	number of cores to be used for parallel programming. Default to 1.
`method`	method for the distributed optimization in the ADMM framework.
`print`	whether to print each iteration. Default to TRUE.
`libpath`	path for the ziphsmm library if not the default set up. Default to NULL.
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 3(1), pp.1-122.

Examples

```
## Not run:
set.seed(12933)
nsubj <- 20
ns <- 4000
ylist <- vector(mode="list",length=nsubj)
xlist <- vector(mode="list",length=nsubj)
timelist <- vector(mode="list",length=nsubj)

priorparm1 <- 0
priorparm2 <- 1
tpmparm1 <- c(-2,-2)
tpmparm2 <- c(0,0)
zeroparm <- c(-2,0)
emitparm <- c(4,0, 6,0)
zeroindex <- c(1,0)
for(n in 1:nsubj){

 xlist[[n]] <- matrix(rep(c(0,1,0,1),rep(1000,4)),nrow=4000,ncol=1)

 timeindex <- rep(1,4000)
 for(i in 2:4000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)
 timelist[[n]] <- timeindex

 if(n<=10){
   workparm <- c(priorparm1,tpmparm1,zeroparm,emitparm)
```

```
 }else{
   workparm <- c(priorparm2,tpmparm2,zeroparm,emitparm)
 }

 result <- hmmsim2.cont(workparm,2,4000,zeroindex,emit_x=xlist[[n]],
                        zeroinfl_x=xlist[[n]],timeindex=timeindex)
 ylist[[n]] <- result$series
}

prior_init=c(0.5,0.5)
tpm_init=matrix(c(-0.1,0.1,0.1,-0.1),2,2,byrow=TRUE)
zero_init=0.2
emit_init=c(50,400)

####
M <- 2
priorclust <- c(rep(1,10),rep(2,10))
tpmclust <- c(rep(1,10),rep(2,10))
zeroclust <- NULL
emitclust <- NULL
slopeclust <- rep(1,20)

group <- vector(mode="list",length=2)
group[[1]] <- 1:10; group[[2]] <- 11:20
###
time <- proc.time()
result <- dist_learn2(ylist, xlist, timelist, prior_init, tpm_init,
                     emit_init, zero_init, NULL, rho=1, priorclust,tpmclust,
                     emitclust,zeroclust,slopeclust,group,ncores=1,
                     maxit=10, tol=1e-4, method="CG",print=TRUE)
proc.time() - time

## End(Not run)
```

`dist_learn3` *Distributed learning for a longitudinal continuous-time zero-inflated Poisson hidden Markov model, where zero-inflation only happens in State 1 with covariates in the state-dependent parameters and transition rates.*

Description

Distributed learning for a longitudinal continuous-time zero-inflated Poisson hidden Markov model, where zero-inflation only happens in State 1 with covariates in the state-dependent parameters and transition rates.

Usage

```
dist_learn3(ylist, xlist, timelist, M, initparm, yceil = NULL, rho = 1,
  priorclust = NULL, tpmclust = NULL, tpmslopeclust = NULL,
```

```
emitclust = NULL, zeroclust = NULL, slopeclust = NULL, group,
maxit = 100, tol = 1e-04, ncores = 1, seed = 0,
method = "Nelder-Mead", print = TRUE, libpath = NULL, ...)
```

Arguments

`ylist`	list of observed time series values for each subject
`xlist`	list of design matrices for each subject.
`timelist`	list of time indices
`M`	number of latent states
`initparm`	matrix of initial working parameters for prior, transition, zero proportion, and emission parameters.
`yceil`	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
`rho`	tuning parameter in the distributed learning algorithm. Default to 1.
`priorclust`	a vector to specify the grouping for state prior. Default to NULL, which means no grouping.
`tpmclust`	a vector to specify the grouping for the intercepts in state transition rates. Default to NULL, which means no grouping.
`tpmslopeclust`	a vector to specify the grouping for the slopes in state transition rates. Default to NULL, which means no grouping.
`emitclust`	a vector to specify the grouping for the intercepts in Poisson regressions. Default to NULL, which means no grouping.
`zeroclust`	a vector to specify the grouping for the intercepts in ZIP regression. Default to NULL, which means no grouping.
`slopeclust`	a vector to specify the grouping for the slopes in Poisson and ZIP regressions. Default to NULL, which means no grouping.
`group`	a list containing group information.
`maxit`	maximum number iteration. Default to 100.
`tol`	tolerance in the terms of the relative change in the norm of the common coefficients. Default to 1e-4.
`ncores`	number of cores to be used for parallel programming. Default to 1.
`seed`	a seed for the random initialization of the algorithm
`method`	method for the distributed optimization in the ADMM framework.
`print`	whether to print each iteration. Default to TRUE.
`libpath`	path for the ziphsmm library if not the default set up. Default to NULL.
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 3(1), pp.1-122.

Examples

```
## Not run:
set.seed(12933)
nsubj <- 10
ns <- 2000
ylist <- vector(mode="list",length=nsubj)
xlist <- vector(mode="list",length=nsubj)
timelist <- vector(mode="list",length=nsubj)

priorparm <- 0
tpmparm <- c(-2,0.1,-2,-0.2)
zeroindex <- c(1,0)
zeroparm <- c(0,0.5)
emitparm <- c(2,0.2,3,0.3)
workparm <- NULL

for(n in 1:nsubj){

 xlist[[n]] <- matrix(rep(c(0,1),rep(1000,2)),nrow=2000,ncol=1)

 timeindex <- rep(1,2000)
 for(i in 2:2000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)
 timelist[[n]] <- timeindex

   workparm <- rbind(workparm,c(priorparm,tpmparm,zeroparm,emitparm))

 result <- hmmsim3.cont(workparm,2,2000,zeroindex,x=xlist[[n]],timeindex=timeindex)
 ylist[[n]] <- result$series
}


####
M <- 2
priorclust <- c(rep(1,5),rep(2,5))
tpmclust <- c(rep(1,5),rep(2,5))
tpmslopeclust <- c(rep(1,5),rep(2,5))
zeroclust <- NULL
emitclust <- NULL
slopeclust <- rep(1,10)

group <- vector(mode="list",length=2)
group[[1]] <- 1:5; group[[2]] <- 6:10
###
time <- proc.time()
result <- dist_learn3(ylist, xlist, timelist, 2,workparm,
                     NULL, rho=1, priorclust,tpmclust,tpmslopeclust,
```

```
                       emitclust,zeroclust,slopeclust,group,ncores=1,
                       maxit=20, tol=1e-4, method="CG",print=TRUE)
proc.time() - time

## End(Not run)
```

`dzip` *pmf for zero-inflated poisson*

Description

pmf for zero-inflated poisson

Usage

```
dzip(p, theta, y, loga)
```

Arguments

`p`	proportion of structural zero's
`theta`	the poisson mean
`y`	the observed value
`loga`	Logical. Whether to return the log probability or not.

Value

the probability mass of the zero-inflated poisson distribution

---

`fasthmmfit` *Fast gradient descent / stochastic gradient descent algorithm to learn the parameters in a specialized zero-inflated hidden Markov model, where zero-inflation only happens in State 1. And if there were covariates, they could only be the same ones for the state-dependent log Poisson means and the logit structural zero proportion.*

Description

Fast gradient descent / stochastic gradient descent algorithm to learn the parameters in a specialized zero-inflated hidden Markov model, where zero-inflation only happens in State 1. And if there were covariates, they could only be the same ones for the state-dependent log Poisson means and the logit structural zero proportion.

Usage

```
fasthmmfit(y, x = NULL, ntimes = NULL, M, prior_init, tpm_init, emit_init,
  zero_init, yceil = NULL, stochastic = FALSE, nmin = 1000,
  nupdate = 100, power = 0.7, rate = c(1, 0.05), method = "Nelder-Mead",
  hessian = FALSE, ...)
```

Arguments

`y`	observed time series values
`x`	matrix of covariates for the log poisson means and logit zero proportion. Default to NULL.
`ntimes`	a vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y).
`M`	number of latent states
`prior_init`	a vector of initial values for prior probability for each state
`tpm_init`	a matrix of initial values for transition probability matrix
`emit_init`	a vector of initial values for the means for each poisson distribution
`zero_init`	a scalar initial value for the structural zero proportion
`yceil`	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
`stochastic`	Logical. Should the stochastic gradient descent methods be used.
`nmin`	a scalar for the minimum number of observations before the first iteration of stochastic gradient descent. Default to 1000.
`nupdate`	a scalar specifying the total number of updates for stochastic gradient descent. Default to 100.
`power`	a scalar representing the power of the learning rate, which should lie between (0.5,1]. Default to 0.7
`rate`	a vector of learning rate in stochastic gradient descent for the logit parameters and log parameters. Default to c(1,0.05).
`method`	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.
`hessian`	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the generalized logit of prior probabilities (except for state 1), the generalized logit of the transition probability matrix(except 1st column), the logit of non-zero zero proportions, and the log of each state-dependent poisson means
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
#1. no covariates
set.seed(135)
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
result <- hmmsim(n=10000,M=3,prior=prior_init, tpm_parm=omega,
                emit_parm=emit_init,zeroprop=zero_init)
y <- result$series

time <- proc.time()
fit1 <-  fasthmmfit(y,x=NULL,ntimes=NULL,M=3,prior_init,omega,
              emit_init,0.5, hessian=FALSE,
              method="BFGS", control=list(trace=1))
proc.time() - time


#2. with covariates
priorparm <- 0
tpmparm <- c(-2,2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)

designx <- matrix(rnorm(20000),nrow=10000,ncol=2)
x <- cbind(1,designx) #has to make the additional 1st column of 1 for intercept
result <- hmmsim2(workparm,2,10000,zeroindex,emit_x=designx,zeroinfl_x=designx)
y <- result$series

time <- proc.time()
fit2 <-  fasthmmfit(y=y,x=x,ntimes=NULL,M=2,prior_init=c(0.5,0.5),
              tpm_init=matrix(c(0.9,0.1,0.1,0.9),2,2),
              zero_init=0.4,emit_init=c(7,21), hessian=FALSE,
              method="BFGS", control=list(trace=1))
proc.time() - time
fit2

#3. stochastic gradient descent without covariates
#no covariates
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
result <- hmmsim(n=50000,M=3,prior=prior_init, tpm_parm=omega,
                emit_parm=emit_init,zeroprop=zero_init)
```

```
y <- result$series

initparm2 <- c(-1,-0.5,  -0.3,-0.3,-0.4,-0.4,0.5,0.5,  0,2,3,4)
time <- proc.time()
fitst <- fasthmmfit(y=y,x=NULL,ntimes=NULL,M=3,prior_init=c(0.4,0.3,0.3),
              tpm_init=matrix(c(0.6,0.3,0.1,0.3,0.4,0.3,0.1,0.3,0.6),3,3,byrow=TRUE),
              zero_init=0.3,emit_init=c(8,35,67),stochastic=TRUE,
              nmin=1000,nupdate=1000,power=0.6,rate=c(1,0.05))
proc.time() - time
str(fitst)

#with covariates
priorparm <- 0
tpmparm <- c(-2,2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)

designx <- matrix(rnorm(100000),nrow=50000,ncol=2)
x <- cbind(1,designx) #has to make the additional 1st column of 1 for intercept
result <- hmmsim2(workparm,2,50000,zeroindex,emit_x=designx,zeroinfl_x=designx)
y <- result$series

initparm <- c(0, -1.8,1.8, 0,-0.8,0.8, 1.8,0.6,-0.6,3.1,0.4,-0.3)

time <- proc.time()
fitst <- fasthmmfit(y=y,x=x,ntimes=NULL,M=2,prior_init=c(0.4,0.6),
              tpm_init=matrix(c(0.8,0.2,0.2,0.8),2,2,byrow=TRUE),
              zero_init=0.3,emit_init=c(10,25),stochastic=TRUE,
              nmin=1000,nupdate=1000,power=0.6,rate=c(1,0.05))
proc.time() - time
str(fitst)
```

`fasthmmfit.cont` *Fast gradient descent algorithm to learn the parameters in a specialized continuous-time zero-inflated hidden Markov model, where zero-inflation only happens in State 1. And if there were covariates, they could only be the same ones for the state-dependent log Poisson means and the logit structural zero proportion.*

Description

Fast gradient descent algorithm to learn the parameters in a specialized continuous-time zero-inflated hidden Markov model, where zero-inflation only happens in State 1. And if there were covariates, they could only be the same ones for the state-dependent log Poisson means and the logit structural zero proportion.

Usage

```
fasthmmfit.cont(y, x = NULL, M, prior_init, tpm_init, emit_init, zero_init,
  yceil = NULL, timeindex, method = "Nelder-Mead", hessian = FALSE, ...)
```

Arguments

y	observed time series values
x	matrix of covariates for the log poisson means and logit zero proportion. Default to NULL.
M	number of latent states
prior_init	a vector of initial values for prior probability for each state
tpm_init	a matrix of initial values for transition rate matrix
emit_init	a vector of initial values for the means for each poisson distribution
zero_init	a scalar initial value for the structural zero proportion
yceil	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
timeindex	a vector containing the time points
method	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the generalized logit of prior probabilities (except for state 1), the generalized logit of the transition probability matrix(except 1st column), the logit of non-zero zero proportions, and the log of each state-dependent poisson means
...	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Liu, Yu-Ying, et al. "Efficient learning of continuous-time hidden markov models for disease progression." Advances in neural information processing systems. 2015.

Examples

```
priorparm <- 0
tpmparm <- c(-1,-2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)
```

```
designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim2.cont(workparm,2,1000,zeroindex,emit_x=designx,
                      zeroinfl_x=designx,timeindex=timeindex)
y <- result$series
state <- result$state

fit2 <-  fasthmmfit.cont(y=y,x=designx,M=2,prior_init=c(0.5,0.5),
  tpm_init=matrix(c(-0.2,0.2,0.1,-0.1),2,2,byrow=TRUE),
  zero_init=0.4,emit_init=c(7,21), timeindex=timeindex,
  hessian=FALSE, method="BFGS", control=list(trace=1))
```

`fasthmmfit.cont3` *Fast gradient descent algorithm to learn the parameters in a specialized continuous-time zero-inflated hidden Markov model, where zero-inflation only happens in State 1 with covariates in the state-dependent parameters and transition rates.*

Description

Fast gradient descent algorithm to learn the parameters in a specialized continuous-time zero-inflated hidden Markov model, where zero-inflation only happens in State 1 with covariates in the state-dependent parameters and transition rates.

Usage

```
fasthmmfit.cont3(y, x, M, initparm, yceil = NULL, timeindex,
  method = "Nelder-Mead", hessian = FALSE, ...)
```

Arguments

`y`	observed time series values
`x`	matrix of covariates in the state-dependent parameters and transition rates.
`M`	number of latent states
`initparm`	vector of initial working parameters for prior, transition, zero proportion, and emission parameters.
`yceil`	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
`timeindex`	a vector containing the time points
`method`	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.
`hessian`	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the generalized logit of prior probabilities (except for state 1), the generalized logit of the transition probability matrix(except 1st column), the logit of non-zero zero proportions, and the log of each state-dependent poisson means
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Liu, Yu-Ying, et al. "Efficient learning of continuous-time hidden markov models for disease progression." Advances in neural information processing systems. 2015.

Examples

```
## Not run:
priorparm <- 0
tpmparm <- c(-2,0.1,-0.1,-2,-0.2,0.2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)

designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim3.cont(workparm,2,1000,zeroindex,x=designx,timeindex=timeindex)
y <- result$series
state <- result$state

fit2 <-  fasthmmfit.cont3(y=y,x=designx,M=2,initparm=workparm,
  timeindex=timeindex,
  hessian=FALSE, method="BFGS", control=list(trace=1))

## End(Not run)
```

---

fasthsmmfit

Fast gradient descent / stochastic gradient descent algorithm to learn the parameters in a specialized zero-inflated hidden semi-Markov model, where zero-inflation only happens in State 1. And if there were covariates, they could only be the same ones for the state-dependent log Poisson means and the logit structural zero proportion. In addition, the dwell time distributions are nonparametric for all hidden states.

Description

Fast gradient descent / stochastic gradient descent algorithm to learn the parameters in a specialized zero-inflated hidden semi-Markov model, where zero-inflation only happens in State 1. And if there were covariates, they could only be the same ones for the state-dependent log Poisson means and the logit structural zero proportion. In addition, the dwell time distributions are nonparametric for all hidden states.

Usage

```
fasthsmmfit(y, x = NULL, ntimes = NULL, M, trunc, dt_init, prior_init,
  tpm_init, emit_init, zero_init, yceil = NULL, stochastic = FALSE,
  nmin = 1000, nupdate = 100, power = 0.7, rate = c(1, 0.05),
  method = "Nelder-Mead", hessian = FALSE, ...)
```

Arguments

`y`	observed time series values
`x`	matrix of covariates for the log poisson means and logit zero proportion. Default to NULL.
`ntimes`	a vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y).
`M`	number of latent states
`trunc`	a vector specifying truncation at the maximum number of dwelling time in each state.
`dt_init`	a matrix whose i,j th element is the probability of staying in state i for duration j, which is the nonparametric state duration distributions.
`prior_init`	a vector of initial values for prior probability for each state
`tpm_init`	a matrix of initial values for transition probability matrix
`emit_init`	a vector of initial values for the means for each poisson distribution
`zero_init`	a scalar initial value for the structural zero proportion
`yceil`	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
`stochastic`	Logical. Should the stochastic gradient descent methods be used.
`nmin`	a scalar for the minimum number of observations before the first iteration of stochastic gradient descent. Default to 1000.
`nupdate`	a scalar specifying the total number of updates for stochastic gradient descent. Default to 100.
`power`	a scalar representing the power of the learning rate, which should lie between (0.5,1]. Default to 0.7
`rate`	a vector of learning rate in stochastic gradient descent for the logit parameters and log parameters. Default to c(1,0.05).
`method`	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.
`hessian`	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the generalized logit of prior probabilities (except for state 1), the generalized logit of the transition probability matrix(except 1st column), the logit of non-zero zero proportions, and the log of each state-dependent poisson means
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
#1. no covariates
set.seed(123)
prior_init <- c(0.5,0.2,0.3)
dt_init <- matrix(c(0.4,0.3,0.2,0.1,0.5,0.2,0.2,0.1,
                   0.25,0.25,0.25,0.25),3,4,byrow=TRUE)
emit_init <- c(10,50,100)
zeroprop <- c(0.6,0,0)
omega <- matrix(c(0,0.3,0.7,0.4,0,0.6,0.5,0.5,0),3,3,byrow=TRUE)
sim1 <- hsmmsim(n=1000,M=3,prior=prior_init,dt_dist="nonparametric",
               dt_parm=dt_init, tpm_parm=omega,
               emit_parm=emit_init,zeroprop=zeroprop)
str(sim1)
y <- sim1$series


fitnost <- fasthsmmfit(y,NULL,NULL,3,trunc=c(4,4,4),
           dt_init=matrix(c(0.3,0.3,0.2,0.2,
           0.4,0.2,0.2,0.2,0.25,0.25,0.25,0.25),3,4,byrow=TRUE),
           prior_init=c(0.3,0.3,0.4),
           tpm_init=matrix(c(0,0.7,0.3,0.5,0,0.5,0.3,0.7,0),3,3,byrow=TRUE),
           emit_init=c(8,40,80),zero_init=0.4,method="BFGS",control=list(trace=1))

decode1 <- hsmmviterbi(y, ntimes=NULL, 3, trunc=c(4,4,4), fitnost$prior,
                      dt_dist="nonparametric",
                      fitnost$dt, fitnost$tpm,
                      fitnost$emit, c(fitnost$zeroprop,0,0))

#2. with covariates
dtparm <-  matrix(c(0.4,0.3,0.2,0.1,0.7,0.2,0.1,0,0.25,0.25,0.25,0.25),3,4,byrow=TRUE)
priorparm <- c(0,0)
zeroindex <- c(1,0,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2,4,0.4,-0.4)
tpmparm <- c(-1,0,1)
workparm <- c(priorparm,zeroparm,emitparm,tpmparm)
trunc <- c(4,3,4)

designx <- matrix(rnorm(4000),nrow=2000,ncol=2)
result <- hsmmsim2(workparm,3,2000,zeroindex,"nonparametric",
                  emit_x=designx,zeroinfl_x=designx,dt_x=dtparm)
y <- result$series
```

```
fitnost <- fasthsmmfit(y,designx,NULL,3,trunc=c(4,3,4),
           dt_init=matrix(c(0.3,0.3,0.2,0.2,0.4,0.2,0.2,0.2,
           0.25,0.25,0.25,0.25),3,4,byrow=TRUE),
           prior_init=c(0.3,0.3,0.4),
           tpm_init=matrix(c(0,0.8,0.2,0.4,0,0.6,0.2,0.8,0),3,3,byrow=TRUE),
           emit_init=c(8,40,80),zero_init=0.4,method="BFGS",control=list(trace=1))

decode2 <- hsmmviterbi2(y,NULL,3,trunc=c(4,3,4),
                        fitnost$working_parm[-(1:8)],dt_x=fitnost$dt,
                        dt_dist="nonparametric", zero_init=c(1,0,0),
                        emit_x=designx,zeroinfl_x=designx)

#3. stochastic gradient descent without covariates
prior_init <- c(0.5,0.2,0.3)
dt_init <- matrix(c(0.4,0.3,0.2,0.1,0.5,0.2,0.2,0.1,
                   0.25,0.25,0.25,0.25),3,4,byrow=TRUE)
emit_init <- c(10,50,100)
zeroprop <- c(0.6,0,0)
omega <- matrix(c(0,0.3,0.7,0.4,0,0.6,0.5,0.5,0),3,3,byrow=TRUE)
sim1 <- hsmmsim(n=50000,M=3,prior=prior_init,dt_dist="nonparametric",
               dt_parm=dt_init, tpm_parm=omega,
               emit_parm=emit_init,zeroprop=zeroprop)
y <- sim1$series

fitst <- fasthsmmfit(y,NULL,NULL,3,trunc=c(4,4,4),
         dt_init=matrix(c(0.4,0.3,0.2,0.1,0.4,0.2,0.2,0.2,
         0.25,0.25,0.25,0.25),3,4,byrow=TRUE),
         prior_init=c(0.3,0.3,0.4),
         tpm_init=matrix(c(0,0.8,0.2,0.5,0,0.5,0.2,0.8,0),3,3,byrow=TRUE),
         emit_init=c(15,40,90),zero_init=0.4,stochastic=TRUE,
         nmin=500,nupdate=500,power=0.6,rate=c(0.5,0.08))
str(fitst)

#4. stochastic descent without covariates
dtparm <-  matrix(c(0.4,0.3,0.2,0.1,0.7,0.2,0.1,0,
           0.25,0.25,0.25,0.25),3,4,byrow=TRUE)
priorparm <- c(0,0)
zeroindex <- c(1,0,0)
zeroparm <- c(0,-0.5,0.5)
emitparm <- c(2,0.1,-0.1,3,0.3,-0.2,4,0.4,-0.4)
tpmparm <- c(-1,0,1)
workparm <- c(priorparm,zeroparm,emitparm,tpmparm)
trunc <- c(4,3,4)

designx <- matrix(rnorm(100000),nrow=50000,ncol=2)
result <- hsmmsim2(workparm,3,50000,zeroindex,"nonparametric",
                  emit_x=designx,zeroinfl_x=designx,dt_x=dtparm)

y <- result$series

fitst <- fasthsmmfit(y,designx,NULL,3,trunc=c(4,3,4),
         dt_init=matrix(c(0.4,0.3,0.2,0.1,0.6,0.3,0.1,0,
```

```
          0.25,0.25,0.25,0.25),3,4,byrow=TRUE),
          prior_init=c(0.3,0.3,0.4),
          tpm_init=matrix(c(0,0.8,0.2,0.5,0,0.5,0.2,0.8,0),3,3,byrow=TRUE),
          emit_init=c(15,40,90),zero_init=0.6,stochastic=TRUE,
          nmin=500,nupdate=500,power=0.6,rate=c(0.3,0.05))
str(fitst)
```

`grad_zipnegloglik_cov_cont`

gradient for negative log likelihood function in zero-inflated Poisson hidden Markov model with covariates, where zero-inflation only happens in state 1

Description

gradient for negative log likelihood function in zero-inflated Poisson hidden Markov model with covariates, where zero-inflation only happens in state 1

Usage

```
grad_zipnegloglik_cov_cont(parm, y, covariates, M, ntimes, timeindex, udiff)
```

Arguments

`parm`	working parameters
`y`	observed series
`covariates`	design matrix of covariates including the intercept
`M`	number of hidden states
`ntimes`	length of the observed series
`timeindex`	vector of observed time points
`udiff`	unique time intervals

Value

gradient for negative log likelihood

`grad_zipnegloglik_nocov_cont`
gradient for negative log likelihood function from zero-inflated Poisson hidden Markov model without covariates, where zero-inflation only happens in state 1

Description

gradient for negative log likelihood function from zero-inflated Poisson hidden Markov model without covariates, where zero-inflation only happens in state 1

Usage

```
grad_zipnegloglik_nocov_cont(parm, M, y, ntimes, timeindex, udiff)
```

Arguments

parm	working parameters
M	number of hidden states
y	observed series
ntimes	length of the observed series
timeindex	vector of observed time points
udiff	unique time intervals

Value

gradient for negative log likelihood

`hmmfit`
Estimate the parameters of a general zero-inflated Poisson hidden Markov model by directly minimizing of the negative log-likelihood function using the gradient descent algorithm.

Description

Estimate the parameters of a general zero-inflated Poisson hidden Markov model by directly minimizing of the negative log-likelihood function using the gradient descent algorithm.

Usage

```
hmmfit(y, ntimes = NULL, M, prior_init, tpm_init, emit_init, zero_init,
  prior_x = NULL, tpm_x = NULL, emit_x = NULL, zeroinfl_x = NULL,
  method = "Nelder-Mead", hessian = FALSE, ...)
```

Arguments

`y`	observed time series values
`ntimes`	a vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y)
`M`	number of latent states
`prior_init`	a vector of initial values for prior probability for each state
`tpm_init`	a matrix of initial values for transition probability matrix
`emit_init`	a vector of initial values for the means for each poisson distribution
`zero_init`	a vector of initial values for the structural zero proportions in each state
`prior_x`	matrix of covariates for generalized logit of prior probabilites (excluding the 1st probability). Default to NULL.
`tpm_x`	matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
`emit_x`	matrix of covariates for the log poisson means. Default to NULL.
`zeroinfl_x`	matrix of covariates for the nonzero structural zero proportions. Default to NULL.
`method`	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.
`hessian`	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the generalized logit of prior probabilities (except for state 1), the generalized logit of the transition probability matrix(except 1st column), the logit of non-zero zero proportions, and the log of each state-dependent poisson means
`...`	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
result <- hmmsim(n=1000,M=3,prior=prior_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zero_init)
y <- result$series
fit <- hmmfit(y=y,M=3,prior_init=prior_init,tpm_init=omega,
```

```
     emit_init=emit_init,zero_init=zero_init,
     method="Nelder-Mead",hessian=TRUE,control=list(maxit=500,trace=1))
str(fit)

#variances for the 12 working parameters, which are the logit of prior probabilities
#for state 2 and state 3, the generalized logit of the transition probability matrix
#(tpm[1,2],tpm[1,3],tpm[2,2],tpm[2,3],tpm[3,2],tpm[3,3]), the logit of structural zero
#proportions for state 1, and log of poisson means for state 1, 2, and 3
#logit of non-zero zero proportions, and the log of poisson means
variance <- diag(solve(fit$obsinfo))

#with covariates
data(CAT)
y <- CAT$activity
x <- data.matrix(CAT$night)
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,50,100)
zero_init <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
fit2 <-  hmmfit(y,rep(1440,3),3,prior_init,omega,
     emit_init,zero_init, emit_x=x,zeroinfl_x=x,hessian=FALSE,
     method="Nelder-Mead", control=list(maxit=500,trace=1))
fit2

## Not run:
#two zero-inflated poissons
prior_init <- c(0.5,0.5)
emit_init <- c(10,50)
zero_init <- c(0.6,0.3)
omega <- matrix(c(0.9,0.1,0.2,0.8),2,2,byrow=TRUE)
result <- hmmsim(n=1000,M=2,prior=prior_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zero_init)
y <- result$series
fit <- hmmfit(y=y,M=2,prior_init=prior_init,tpm_init=omega,
     emit_init=emit_init,zero_init=zero_init,
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=500,trace=1))
str(fit)

#four regular poissons
prior_init <- c(0.4,0.2,0.2,0.2)
emit_init <- c(10,40,70,100)
zero_init <- c(0,0,0,0)
omega <- matrix(c(0.3,0.3,0.2,0.2,0.4,0.2,0.3,0.1,
                  0.2,0.2,0.3,0.3,0.1,0.2,0.3,0.4),4,4,byrow=TRUE)
result <- hmmsim(n=1000,M=4,prior=prior_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zero_init)
y <- result$series
fit <- hmmfit(y=y,M=4,prior_init=prior_init,tpm_init=omega,
     emit_init=emit_init,zero_init=zero_init,
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=500,trace=1))
str(fit)

## End(Not run)
```

hmmsim — *Simulate a hidden Markov series and its underlying states with zero-inflated emission distributions*

Description

Simulate a hidden Markov series and its underlying states with zero-inflated emission distributions

Usage

```
hmmsim(n, M, prior, tpm_parm, emit_parm, zeroprop)
```

Arguments

n	length of the simulated series
M	number of hidden states
prior	a vector of prior probability for each state
tpm_parm	transition probability matrix
emit_parm	a vector containing means for each poisson distribution
zeroprop	a vector containing structural zero proportions in each state

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,50,100)
zeroprop <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
result <- hmmsim(n=1000,M=3,prior=prior_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zeroprop)
str(result)
```

hmmsim.cont *Simulate a hidden Markov series and its underlying states with zero-inflated emission distributions*

Description

Simulate a hidden Markov series and its underlying states with zero-inflated emission distributions

Usage

```
hmmsim.cont(n, M, prior, tpm_parm, emit_parm, zeroprop, timeindex)
```

Arguments

n	length of the simulated series
M	number of hidden states
prior	a vector of prior probability for each state
tpm_parm	transition rate matrix
emit_parm	a vector containing means for each poisson distribution
zeroprop	a vector containing structural zero proportions in each state
timeindex	a vector containing the time points

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(-0.3,0.2,0.1,0.1,-0.2,0.1,0.2,0.2,-0.4),3,3,byrow=TRUE)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:3,1)
result <- hmmsim.cont(n=1000,M=3,prior=prior_init, tpm_parm=omega,
          emit_parm=emit_init,zeroprop=zero_init,timeindex=timeindex)
```

hmmsim2	*Simulate a hidden Markov series and its underlying states with covariates*

Description

Simulate a hidden Markov series and its underlying states with covariates

Usage

```
hmmsim2(workparm, M, n, zeroindex, prior_x = NULL, tpm_x = NULL,
  emit_x = NULL, zeroinfl_x = NULL)
```

Arguments

workparm	working parameters
M	number of latent states
n	length of the simulated series
zeroindex	a vector specifying whether a certain state is zero-inflated
prior_x	matrix of covariates for generalized logit of prior probabilites (excluding the 1st probability). Default to NULL.
tpm_x	matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
emit_x	matrix of covariates for the log poisson means. Default to NULL.
zeroinfl_x	matrix of covariates for the nonzero structural zero proportions. Default to NULL.

Value

a matrix with 1st column of simulated series and 2nd column of corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
## Not run:
priorparm <- 0
tpmparm <- c(0,-0.5,0.5,0,-0.2,0.8)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
```

```
designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim2(workparm,2,1000,zeroindex,tpm_x=designx,
      emit_x=designx,zeroinfl_x=designx)

y <- result$series

prior_init <- c(0.5,0.5)
emit_init <- c(10,30)
zero_init <- c(0.6,0)
omega <- matrix(c(0.9,0.1,0.2,0.8),2,2,byrow=TRUE)


fit <-  hmmfit(y,NULL,2,prior_init,omega,
     emit_init,zero_init, emit_x=designx,zeroinfl_x=designx,
     tpm_x=designx,hessian=FALSE,
     method="Nelder-Mead", control=list(maxit=2000,trace=1))

decode <- hmmviterbi2(y,NULL,2,fit$working_parameters,zero_init=c(1,0),
            emit_x=designx,zeroinfl_x=designx, tpm_x=designx,
            plot=TRUE, xlab="time", ylab="count",
            xlim=c(0,360),ylim=c(0,200))

sum(decode!=result$state)

## End(Not run)
```

`hmmsim2.cont` *Simulate a continuous-time hidden Markov series and its underlying states with covariates*

Description

Simulate a continuous-time hidden Markov series and its underlying states with covariates

Usage

```
hmmsim2.cont(workparm, M, n, zeroindex, emit_x = NULL, zeroinfl_x = NULL,
  timeindex)
```

Arguments

workparm	working parameters
M	number of latent states
n	length of the simulated series
zeroindex	a vector specifying whether a certain state is zero-inflated
emit_x	matrix of covariates for the log poisson means. Default to NULL.
zeroinfl_x	matrix of covariates for the nonzero structural zero proportions. Default to NULL.
timeindex	a vector containing the time points

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
priorparm <- 0
tpmparm <- c(-1,-2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)

designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim2.cont(workparm,2,1000,zeroindex,emit_x=designx,
                       zeroinfl_x=designx,timeindex=timeindex)
y <- result$series
state <- result$state
```

---

hmmsim3.cont — *Simulate a continuous-time hidden Markov series and its underlying states with covariates in state-dependent parameters and transition rates.*

Description

Simulate a continuous-time hidden Markov series and its underlying states with covariates in state-dependent parameters and transition rates.

Usage

```
hmmsim3.cont(workparm, M, n, zeroindex, x, timeindex)
```

Arguments

workparm	working parameters
M	number of latent states
n	length of the simulated series
zeroindex	a vector specifying whether a certain state is zero-inflated
x	matrix of covariates for the log poisson means, structural zero proportions and transition rates.
timeindex	a vector containing the time points

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
priorparm <- 0
tpmparm <- c(-2,0.1,-0.1,-2,0.2,-0.2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)

designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim3.cont(workparm,2,1000,zeroindex,x=designx,timeindex=timeindex)
y <- result$series
state <- result$state
```

---

hmmsmooth.cont — *Compute the posterior state probabilities for continuous-time hidden Markov models without covariates where zero-inflation only happens in state 1*

Description

Compute the posterior state probabilities for continuous-time hidden Markov models without covariates where zero-inflation only happens in state 1

Usage

```
hmmsmooth.cont(y, M, prior_init, tpm_init, emit_init, zero_init, timeindex)
```

Arguments

`y`	the observed series to be decoded
`M`	number of latent states
`prior_init`	a vector of prior probability values
`tpm_init`	transition rate matrix
`emit_init`	a vector containing means for each poisson distribution
`zero_init`	a scalar containing structural zero proportion in state 1
`timeindex`	a vector containing the time points

Value

posterior state probabilities

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(-0.3,0.2,0.1,0.1,-0.2,0.1,0.2,0.2,-0.4),3,3,byrow=TRUE)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:3,1)
result <- hmmsim.cont(n=1000,M=3,prior=prior_init, tpm_parm=omega,
          emit_parm=emit_init,zeroprop=zero_init,timeindex=timeindex)
y <- result$series
fit2 <-  fasthmmfit.cont(y,x=NULL,M=3,prior_init,omega,
                        emit_init,0.5,timeindex=timeindex,hessian=FALSE,
                        method="BFGS", control=list(maxit=500,trace=1))
post <- hmmsmooth.cont(y,3,fit2$prior,fit2$tpm,fit2$emit,
           fit2$zeroprop,timeindex=timeindex)
```

`hmmsmooth.cont2` *Compute the posterior state probabilities for continuous-time hidden Markov models where zero-inflation only happens in state 1 and covariates can only be included in the state-dependent parameters*

Description

Compute the posterior state probabilities for continuous-time hidden Markov models where zero-inflation only happens in state 1 and covariates can only be included in the state-dependent parameters

Usage

```
hmmsmooth.cont2(y, x, M, prior, tpm, zeroparm, emitparm, timeindex)
```

Arguments

`y`	the observed series to be decoded
`x`	matrix of covariates for the log poisson means and logit zero proportion. Default to NULL.
`M`	number of latent states

prior	prior parameters from the fitted continuous-time hidden Markov model
tpm	transition rate parameters from the fitted continuous-time hidden Markov model
zeroparm	parameters for the structural zero proportions in the fitted continuous-time hidden Markov model
emitparm	parameters for the Poisson means in the fitted continuous-time hidden Markov model
timeindex	a vector containing the time points

Value

posterior state probabilities

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
set.seed(2910)
priorparm <- 0
tpmparm <- c(-1,-2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(4,0.2,-0.2,5,0.1,-0.1)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)

designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim2.cont(workparm,2,1000,zeroindex,emit_x=designx,
                      zeroinfl_x=designx,timeindex=timeindex)
y <- result$series
state <- result$state

fit2 <-  fasthmmfit.cont(y=y,x=designx,M=2,prior_init=c(0.5,0.5),
  tpm_init=matrix(c(-0.2,0.2,0.1,-0.1),2,2,byrow=TRUE),
  zero_init=0.4,emit_init=c(50,150), timeindex=timeindex,
  hessian=FALSE, method="BFGS", control=list(trace=1))
post <- hmmsmooth.cont2(y,designx,2,fit2$prior,fit2$tpm,fit2$zeroparm,
       fit2$emitparm,timeindex)
```

`hmmsmooth.cont3` *Compute the posterior state probabilities for continuous-time hidden Markov models with covariates in the state-dependent parameters and transition rates*

Description

Compute the posterior state probabilities for continuous-time hidden Markov models with covariates in the state-dependent parameters and transition rates

Usage

```
hmmsmooth.cont3(y, x, M, prior, tpmparm, zeroparm, emitparm, timeindex)
```

Arguments

`y`	the observed series to be decoded
`x`	matrix of covariates in the state-dependent parameters and transition rates.
`M`	number of latent states
`prior`	prior parameters from the fitted continuous-time hidden Markov model
`tpmparm`	parameters from the fitted continuous-time hidden Markov model
`zeroparm`	parameters for the structural zero proportions in the fitted continuous-time hidden Markov model
`emitparm`	parameters for the Poisson means in the fitted continuous-time hidden Markov model
`timeindex`	a vector containing the time points

Value

posterior state probabilities

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
## Not run:
set.seed(2910)
priorparm <- 0
tpmparm <- c(-2,0.1,-0.1,-2,-0.2,0.2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
```

```
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)

designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
result <- hmmsim3.cont(workparm,2,1000,zeroindex,x=designx,timeindex=timeindex)
y <- result$series
state <- result$state

fit2 <-  fasthmmfit.cont3(y=y,x=designx,M=2,
  initparm=workparm, timeindex=timeindex,
  hessian=FALSE, method="CG", control=list(trace=1))
post <- hmmsmooth.cont3(y,designx,2,fit2$prior,fit2$tpm,fit2$zeroparm,
       fit2$emitparm,timeindex)

## End(Not run)
```

---

hmmviterbi *Viterbi algorithm to decode the latent states for hidden Markov models*

Description

Viterbi algorithm to decode the latent states for hidden Markov models

Usage

```
hmmviterbi(y, ntimes = NULL, M, prior_init, tpm_init, emit_init, zero_init,
  plot = FALSE, xlim = NULL, ylim = NULL, ...)
```

Arguments

y	the observed series to be decoded
ntimes	vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y)
M	number of latent states
prior_init	a vector of prior probability values
tpm_init	transition probability matrix
emit_init	a vector containing means for each poisson distribution
zero_init	a vector containing structural zero proportions in each state
plot	whether a plot should be returned
xlim	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
ylim	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
...	further arguments to be passed to the plot() function

Value

the decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
result <- hmmsim(n=1000,M=3,prior=prior_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zero_init)
y <- result$series
state <- result$state
fit <- hmmfit(y=y,M=3,prior_init=prior_init,tpm_init=omega,
     emit_init=emit_init,zero_init=zero_init,
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=500,trace=1))

decode <- hmmviterbi(y,NULL,3,fit$prior,fit$tpm,fit$emit_parm,fit$zeroprop,
                      plot=TRUE,xlab="Time",ylab="Count")
#check the missclassification rate
sum(decode!=state)/length(state)

## Not run:
decode <- hmmviterbi(y,NULL,3,fit$prior,fit$tpm,fit$emit_parm,fit$zeroprop,
                      plot=TRUE,xlim=c(0,100),ylim=c(0,100),
                      xlab="Time",ylab="Count")

## End(Not run)
```

---

hmmviterbi.cont — *Viterbi algorithm to decode the latent states for continuous-time hidden Markov models without covariates*

Description

Viterbi algorithm to decode the latent states for continuous-time hidden Markov models without covariates

Usage

```
hmmviterbi.cont(y, M, prior_init, tpm_init, emit_init, zero_init, timeindex,
  plot = FALSE, xlim = NULL, ylim = NULL, ...)
```

Arguments

y	the observed series to be decoded
M	number of latent states
prior_init	a vector of prior probability values
tpm_init	transition rate matrix
emit_init	a vector containing means for each poisson distribution
zero_init	a vector containing structural zero proportions in each state
timeindex	a vector containing the time points
plot	whether a plot should be returned
xlim	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
ylim	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
...	further arguments to be passed to the plot() function

Value

the decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,40,70)
zero_init <- c(0.5,0,0)
omega <- matrix(c(-0.3,0.2,0.1,0.1,-0.2,0.1,0.2,0.2,-0.4),3,3,byrow=TRUE)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:3,1)
result <- hmmsim.cont(n=1000,M=3,prior=prior_init, tpm_parm=omega,
          emit_parm=emit_init,zeroprop=zero_init,timeindex=timeindex)
y <- result$series
fit2 <-  fasthmmfit.cont(y,x=NULL,M=3,prior_init,omega,
                        emit_init,0.5,timeindex=timeindex,hessian=FALSE,
                        method="BFGS", control=list(maxit=500,trace=1))
decode2 <- hmmviterbi.cont(y,3,fit2$prior,fit2$tpm,fit2$emit,
           c(fit2$zeroprop,0,0),timeindex=timeindex)
```

hmmviterbi2 — *Viterbi algorithm to decode the latent states in hidden Markov models with covariate values*

Description

Viterbi algorithm to decode the latent states in hidden Markov models with covariate values

Usage

```
hmmviterbi2(y, ntimes = NULL, M, workparm, zero_init, prior_x = NULL,
  tpm_x = NULL, emit_x = NULL, zeroinfl_x = NULL, plot = FALSE,
  xlim = NULL, ylim = NULL, ...)
```

Arguments

y	the observed series to be decoded
ntimes	vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y)
M	number of latent states
workparm	a vector of values for working parameters, which is the last element returned from hmmfit() function. This consists the generalized logit of prior probabilities (except for the 1st state), generalized logit of transition probability matrix (except for the 1st column), the logit of nonzero structural zero proportions, and the log poisson means
zero_init	a vector containing structural zero proportions in each state, e.g. set zero_init[i] to be 0 if the i-th state is a regular poisson, and otherwise 1.
prior_x	matrix of covariates for generalized logit of prior probabilites (excluding the 1st probability). Default to NULL.
tpm_x	matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
emit_x	matrix of covariates for the log poisson means. Default to NULL.
zeroinfl_x	matrix of covariates for the nonzero structural zero proportions. Default to NULL.
plot	whether a plot should be returned
xlim	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
ylim	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
...	further arguments to be passed to the plot() function

Value

decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
data(CAT)
y <- CAT$activity
x <- data.matrix(CAT$night)
prior_init <- c(0.5,0.2,0.3)
emit_init <- c(10,50,100)
zero_init <- c(0.5,0,0)
omega <- matrix(c(0.5,0.3,0.2,0.4,0.3,0.3,0.2,0.4,0.4),3,3,byrow=TRUE)
fit2 <-  hmmfit(y,rep(1440,3),3,prior_init,omega,
     emit_init,zero_init, emit_x=x,zeroinfl_x=x,hessian=FALSE,
     method="Nelder-Mead", control=list(maxit=500,trace=1))
decode <- hmmviterbi2(y,rep(1440,3),3,fit2$working_parameters,zero_init=c(1,0,0),
            emit_x=x,zeroinfl_x=x, plot=TRUE, xlab="time", ylab="count",
            xlim=c(0,360),ylim=c(0,200))
```

---

hmmviterbi2.cont — *Viterbi algorithm to decode the latent states in continuous-time hidden Markov models with covariates*

Description

Viterbi algorithm to decode the latent states in continuous-time hidden Markov models with covariates

Usage

```
hmmviterbi2.cont(y, M, workparm, zero_init, emit_x = NULL,
  zeroinfl_x = NULL, timeindex, plot = FALSE, xlim = NULL, ylim = NULL,
  ...)
```

Arguments

`y`	the observed series to be decoded
`M`	number of latent states
`workparm`	a vector of values for working parameters, which is the last element returned from hmmfit() function. This consists the generalized logit of prior probabilities (except for the 1st state), generalized logit of transition probability matrix (except for the 1st column), the logit of nonzero structural zero proportions, and the log poisson means
`zero_init`	a vector containing structural zero proportions in each state, e.g. set zero_init[i] to be 0 if the i-th state is a regular poisson, and otherwise 1.

`emit_x`	matrix of covariates for the log poisson means. Default to NULL.
`zeroinfl_x`	matrix of covariates for the nonzero structural zero proportions. Default to NULL.
`timeindex`	a vector containing the time points
`plot`	whether a plot should be returned
`xlim`	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
`ylim`	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
`...`	further arguments to be passed to the plot() function

Value

decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
priorparm <- 0
tpmparm <- c(-1,-2)
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,3,0.3,-0.2)
workparm <- c(priorparm,tpmparm,zeroparm,emitparm)
timeindex <- rep(1,1000)
for(i in 2:1000) timeindex[i] <- timeindex[i-1] + sample(1:4,1)

designx <- matrix(rnorm(2000),nrow=1000,ncol=2)
x <- cbind(1,designx) #has to make the additional 1st column of 1 for intercept
result <- hmmsim2.cont(workparm,2,1000,zeroindex,emit_x=designx,
                      zeroinfl_x=designx,timeindex=timeindex)
y <- result$series
state <- result$state

fit2 <-  fasthmmfit.cont(y=y,x=designx,M=2,prior_init=c(0.5,0.5),
  tpm_init=matrix(c(-0.2,0.2,0.1,-0.1),2,2,byrow=TRUE),
  zero_init=0.4,emit_init=c(7,21), timeindex=timeindex,
  hessian=FALSE, method="BFGS", control=list(trace=1))
decode2 <- hmmviterbi2.cont(y,2,fit2$working_parm,c(1,0),
  emit_x=designx, zeroinfl_x=designx,
  timeindex=timeindex,plot=FALSE)
```

`hsmmfit` *Estimate the parameters of a general zero-inflated Poisson hidden semi-Markov model by directly minimizing of the negative log-likelihood function using the gradient descent algorithm.*

Description

Estimate the parameters of a general zero-inflated Poisson hidden semi-Markov model by directly minimizing of the negative log-likelihood function using the gradient descent algorithm.

Usage

```
hsmmfit(y, ntimes = NULL, M, trunc, prior_init, dt_dist, dt_init, tpm_init,
  emit_init, zero_init, prior_x = NULL, dt_x = NULL, tpm_x = NULL,
  emit_x = NULL, zeroinfl_x = NULL, method = "Nelder-Mead",
  hessian = FALSE, ...)
```

Arguments

`y`	observed time series values
`ntimes`	A vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(data)
`M`	number of hidden states
`trunc`	a vector specifying truncation at the maximum number of dwelling time in each state. The higher the truncation, the more accurate the approximation but also the more computationally expensive.
`prior_init`	a vector of initial value for prior probability for each state
`dt_dist`	dwell time distribution, can only be "log", "geometric", or "shiftedpoisson"
`dt_init`	a vector of initial value for the parameter in each dwell time distribution, which should be a vector of p's for dt_dist == "log" and a vector of theta's for dt_dist=="shiftpoisson"
`tpm_init`	a matrix of initial values for the transition probability matrix, whose diagonal elements should be zero's
`emit_init`	a vector initial value for the vector containing means for each poisson distribution
`zero_init`	a vector initial value for the vector containing structural zero proportions in each state
`prior_x`	matrix of covariates for generalized logit of prior probabilites (excluding the 1st probability). Default to NULL.
`dt_x`	matrix of covariates for the dwell time distribution parameters
`tpm_x`	matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
`emit_x`	matrix of covariates for the log poisson means. Default to NULL.

zeroinfl_x	matrix of covariates for the nonzero structural zero proportions. Default to NULL.
method	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the logit of parameter p for each log-series dwell time distribution or the log of parameter theta for each shifted-poisson dwell time distribution, the generalized logit of prior probabilities (except for the 1st state),the logit of each nonzero structural zero proportions, the log of each state-dependent poisson means, and the generalized logit of the transition probability matrix(except 1st column and the diagonal elements)
...	Further arguments passed on to the optimization methods

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
#2 zero-inflated poissons
prior_init <- c(0.5,0.5)
emit_init <- c(10,30)
dt_init <- c(10,6)
trunc <- c(20,10)
zeroprop <- c(0.5,0.3)
omega <- matrix(c(0,1,1,0),2,2,byrow=TRUE)
sim2 <- hsmmsim(n=1000,M=2,prior=prior_init,dt_dist="shiftpoisson",
         dt_parm=dt_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zeroprop)
str(sim2)
y <- sim2$series
fit2 <- hsmmfit(y=y,M=2,trunc=trunc,prior_init=prior_init,dt_dist="shiftpoisson",
     dt_init=dt_init,
     tpm_init=omega,emit_init=emit_init,zero_init=zeroprop,
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=500,trace=1))
str(fit2)


## Not run:
#1 zero-inflated poisson and 3 regular poissons
prior_init <- c(0.5,0.2,0.2,0.1)
dt_init <- c(0.8,0.7,0.6,0.5)
emit_init <- c(10,30,70,130)
```

```
trunc <- c(10,10,10,10)
zeroprop <- c(0.6,0,0,0)  #only the 1st-state is zero-inflated
omega <- matrix(c(0,0.5,0.3,0.2,0.4,0,0.4,0.2,
                  0.2,0.6,0,0.2,0.1,0.1,0.8,0),4,4,byrow=TRUE)
sim1 <- hsmmsim(n=2000,M=4,prior=prior_init,dt_dist="log",
         dt_parm=dt_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zeroprop)
str(sim1)
y <- sim1$series
fit <- hsmmfit(y=y,M=4,trunc=trunc,prior_init=prior_init,dt_dist="log",dt_init=dt_init,
     tpm_init=omega,emit_init=emit_init,zero_init=zeroprop,
     method="Nelder-Mead",hessian=TRUE,control=list(maxit=500,trace=1))
str(fit)

#variances for the 20 working parameters, which are the logit of parameter p for
#the 4 log-series dwell time distributions, the generalized logit of prior probabilities
#for state 2,3,4, the logit of each nonzero structural zero proportions in state 1,
#the log of 4 state-dependent poisson means, and the generalized logit of the
#transition probability matrix(which are tpm[1,3],tpm[1,4], tpm[2,3],tpm[2,4],
#tpm[3,2],tpm[3,4],tpm[4,2],tpm[4,3])
variance <- diag(solve(fit$obsinfo))


#1 zero-inflated poisson and 2 poissons with covariates
data(CAT)
y <- CAT$activity
x <- data.matrix(CAT$night)
prior_init <- c(0.5,0.3,0.2)
dt_init <- c(0.9,0.6,0.3)
emit_init <- c(10,20,30)
zero_init <- c(0.5,0,0) #assuming only the 1st state has structural zero's
tpm_init <- matrix(c(0,0.3,0.7,0.4,0,0.6,0.5,0.5,0),3,3,byrow=TRUE)
trunc <- c(10,7,4)
fit2 <-  hsmmfit(y,rep(1440,3),3,trunc,prior_init,"log",dt_init,tpm_init,
     emit_init,zero_init,emit_x=x,zeroinfl_x=x,hessian=FALSE,
     method="Nelder-Mead", control=list(maxit=500,trace=1))
fit2

#another example with covariates for 2 zero-inflated poissons
data(CAT)
y <- CAT$activity
x <- data.matrix(CAT$night)
prior_init <- c(0.5,0.5)
dt_init <- c(10,5)
emit_init <- c(10, 30)
zero_init <- c(0.5,0.2)
tpm_init <- matrix(c(0,1,1,0),2,2,byrow=TRUE)
trunc <- c(10,5)
fit <-  hsmmfit(y,NULL,2,trunc,prior_init,"shiftpoisson",dt_init,tpm_init,
     emit_init,zero_init,dt_x=x,emit_x=x,zeroinfl_x=x,tpm_x=x,hessian=FALSE,
     method="Nelder-Mead", control=list(maxit=500,trace=1))
fit
```

```
## End(Not run)
```

---

hsmmfit_exp *Simulate a hidden semi-Markov series and its underlying states with covariates where the latent state distributions have accelerated failure time structure whose base densities are exponential*

Description

Simulate a hidden semi-Markov series and its underlying states with covariates where the latent state distributions have accelerated failure time structure whose base densities are exponential

Usage

```
hsmmfit_exp(y, M, trunc, dtrate, dtparm, prior, zeroparm, emitparm, tpmparm,
  dt_x, zeroinfl_x, emit_x, tpm_x, yceil = NULL, method = "Nelder-Mead",
  hessian = FALSE, ...)
```

Arguments

y	observed time series values
M	number of latent states
trunc	a vector specifying truncation at the maximum number of dwelling time in each state.
dtrate	a vector for the scale parameters in the base exponential density for the latent state durations.
dtparm	a matrix of coefficients for the accelerated failure time model in each latent state
prior	a vector of prior probabilities
zeroparm	a vector of regression coefficients for the structural zero proportion in state 1
emitparm	a matrix of regression coefficients for the Poisson regression in each state
tpmparm	a vector of coefficients for the multinomial logistic regression in the transition probabilities
dt_x	a matrix of covariates for the latent state durations
zeroinfl_x	a matrix of covariates for the zero proportion
emit_x	a matrix of covariates for the Poisson means
tpm_x	a matrix of covariates for the transition
yceil	a scalar defining the ceiling of y, above which the values will be truncated. Default to NULL.
method	method to be used for direct numeric optimization. See details in the help page for optim() function. Default to Nelder-Mead.

hessian	Logical. Should a numerically differentiated Hessian matrix be returned? Note that the hessian is for the working parameters, which are the generalized logit of prior probabilities (except for state 1), the generalized logit of the transition probability matrix(except 1st column), the logit of non-zero zero proportions, and the log of each state-dependent poisson means
...	Further arguments passed on to the optimization methods

Value

the maximum likelihood estimates of the zero-inflated hidden Markov model

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

---

hsmmsim — *Simulate a hidden semi-Markov series and its corresponding states according to the specified parameters*

Description

Simulate a hidden semi-Markov series and its corresponding states according to the specified parameters

Usage

```
hsmmsim(n, M, prior, dt_dist = "nonparametric", dt_parm, tpm_parm, emit_parm,
  zeroprop)
```

Arguments

n	length of the simulated series
M	number of hidden states
prior	a vector of prior probability for each state
dt_dist	dwell time distribution, which should be "log" or "shiftpoisson" or "nonparametric". Default to "nonparametric".
dt_parm	a vector of dwell time distribution parameters for each state. If dt_dist is "log", then dt_parm is vector of p's; if dt_dist is "shiftpoisson", then dt_parm is vector of theta's; if dt_dist is "nonparametric", then dt_parm is a matrix whose i,j th element is the probability of staying in state i for duration j.
tpm_parm	transition probability matrix, whose diagonal should be zero's.
emit_parm	a vector containing means for each poisson distribution
zeroprop	a vector containing structural zero proportions in each state

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
prior_init <- c(0.5,0.2,0.3)
dt_init <- c(0.8,0.5,0.2)
emit_init <- c(10,50,100)
zeroprop <- c(0.6,0.3,0.1)
omega <- matrix(c(0,0.3,0.7,0.4,0,0.6,0.5,0.5,0),3,3,byrow=TRUE)
sim1 <- hsmmsim(n=1000,M=3,prior=prior_init,dt_dist="log",
         dt_parm=dt_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zeroprop)
str(sim1)

prior_init <- c(0.5,0.5)
dt_init <- c(10,5)
emit_init <- c(10,30)
zeroprop <- c(0.5,0)
omega <- matrix(c(0,1,1,0),2,2,byrow=TRUE)
sim2 <- hsmmsim(n=1000,M=2,prior=prior_init,dt_dist="shiftpoisson",
         dt_parm=dt_init, tpm_parm=omega,
         emit_parm=emit_init,zeroprop=zeroprop)
str(sim2)
hist(sim2$series,main="Histogram of observed values",xlab="observed values")
```

`hsmmsim2` *Simulate a hidden semi-Markov series and its underlying states with covariates*

Description

Simulate a hidden semi-Markov series and its underlying states with covariates

Usage

```
hsmmsim2(workparm, M, n, zeroindex, dt_dist = "nonparametric", dt_x = NULL,
  prior_x = NULL, tpm_x = NULL, emit_x = NULL, zeroinfl_x = NULL)
```

Arguments

- `workparm` — working parameters. The first part is the logit of parameter p for each log-series dwell time distribution or the log of parameter theta for each shifted-poisson dwell time distribution. If dt_dist is "nonparametric", then the first part is empty. The next part is the generalized logit of prior probabilities (except for the 1st state),the logit of each nonzero structural zero proportions, the log of each state-dependent poisson means, and the generalized logit of the transition probability matrix(except 1st column and the diagonal elements).
- `M` — number of latent states
- `n` — length of the simulated series
- `zeroindex` — a vector specifying whether a certain state is zero-inflated
- `dt_dist` — dwell time distribution, can only be "log" or "shiftedpoisson" or "nonparametric". Default to "nonparametric".
- `dt_x` — if dt_dist is "nonparametric", then dt_x is the matrix of nonparametric state durataion probabilities. Otherwise, dt_x is matrix of covariates for the dwell time distribution parameters in log-series or shifted-poisson distributions.Default to NULL.
- `prior_x` — matrix of covariates for generalized logit of prior probabilites (excluding the 1st probability). Default to NULL. Set to NULL if dt_dist is "nonparametric".
- `tpm_x` — matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
- `emit_x` — matrix of covariates for the log poisson means. Default to NULL.
- `zeroinfl_x` — matrix of covariates for the nonzero structural zero proportions. Default to NULL.

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
## Not run:
#example 1

dtparm <- c(2,1)  #in the log scale
priorparm <- 0
zeroindex <- c(1,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,4,0.3,-0.2)
workparm <- c(dtparm,priorparm,zeroparm,emitparm)

designx <- matrix(rnorm(4000),nrow=2000,ncol=2)
```

```
result <- hsmmsim2(workparm,2,2000,zeroindex,"shiftpoisson",
      emit_x=designx,zeroinfl_x=designx)

y <- result$series

dt_init <- c(8,3)
prior_init <- c(0.5,0.5)
emit_init <- c(10,50)
trunc <- c(13,8)
zeroprop <- c(0.8,0)
omega <- matrix(c(0,1,1,0),2,2,byrow=TRUE)


fit1 <- hsmmfit(y=y,M=2,trunc=trunc,prior_init=prior_init,dt_dist="shiftpoisson",
     dt_init=dt_init,emit_x=designx,zeroinfl_x=designx,
     tpm_init=omega,emit_init=emit_init,zero_init=zeroprop,
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=2000,trace=1))

decode <- hsmmviterbi2(y,NULL,2,trunc,fit1$working_parameters,
            dt_dist="shiftpoisson", zero_init=zeroprop,
            emit_x=designx,zeroinfl_x=designx, plot=TRUE, xlab="time", ylab="count",
            xlim=c(0,2000),ylim=c(0,200))
sum(decode!=result$state)



#example 2

dtparm <- c(2,0,-1)  #logit scale
priorparm <- c(0,0)
zeroindex <- c(1,0,0)
zeroparm <- c(0,-1,1)
emitparm <- c(2,0.5,-0.5,4,0.3,-0.2,6,0.2,-0.2)
tpmparm <- c(0,0,0)
workparm <- c(dtparm,priorparm,zeroparm,emitparm,tpmparm)

designx <- matrix(rnorm(4000),nrow=2000,ncol=2)
result <- hsmmsim2(workparm,3,2000,zeroindex,"log",
      emit_x=designx,zeroinfl_x=designx)

y <- result$series

dt_init <- c(0.9,0.5,0.3)
prior_init <- c(0.3,0.4,0.3)
emit_init <- c(10,100,400)
trunc <- c(13,5,3)
zeroprop <- c(0.8,0,0)
omega <- matrix(c(0,0.5,0.5,0.5,0,0.5,0.5,0.5,0),3,3,byrow=TRUE)


fit2 <- hsmmfit(y=y,M=3,trunc=trunc,prior_init=prior_init,dt_dist="log",
     dt_init=dt_init,emit_x=designx,zeroinfl_x=designx,
     tpm_init=omega,emit_init=emit_init,zero_init=zeroprop,
```

```
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=2000,trace=1))

decode <- hsmmviterbi2(y,NULL,3,trunc,fit2$working_parameters,
            dt_dist="shiftpoisson", zero_init=zeroprop,
            emit_x=designx,zeroinfl_x=designx, plot=TRUE, xlab="time", ylab="count",
            xlim=c(0,2000),ylim=c(0,1000))
sum(decode!=result$state)

## End(Not run)
```

---

hsmmsim2_exp *Simulate a hidden semi-Markov series and its underlying states with covariates*

Description

Simulate a hidden semi-Markov series and its underlying states with covariates

Usage

```
hsmmsim2_exp(prior, dtrate, dtparm, zeroparm, emitparm, tpmparm, trunc, M, n,
  dt_x = NULL, tpm_x = NULL, emit_x = NULL, zeroinfl_x = NULL)
```

Arguments

prior	a vector of prior probabilities
dtrate	a vector for the scale parameters in the base exponential density for the latent state durations.
dtparm	a matrix of coefficients for the accelerated failure time model in each latent state
zeroparm	a vector of regression coefficients for the structural zero proportion in state 1
emitparm	a matrix of regression coefficients for the Poisson regression in each state
tpmparm	a vector of coefficients for the multinomial logistic regression in the transition probabilities
trunc	a vector
M	number of latent states
n	length of the simulated series
dt_x	if dt_dist is "nonparametric", then dt_x is the matrix of nonparametric state durataion probabilities. Otherwise, dt_x is matrix of covariates for the dwell time distribution parameters in log-series or shifted-poisson distributions.Default to NULL.
tpm_x	matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
emit_x	matrix of covariates for the log poisson means. Default to NULL.
zeroinfl_x	matrix of covariates for the nonzero structural zero proportions. Default to NULL.

Value

simulated series and corresponding states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

---

hsmmviterbi — *Viterbi algorithm to decode the latent states for hidden semi-Markov models*

Description

Viterbi algorithm to decode the latent states for hidden semi-Markov models

Usage

```
hsmmviterbi(y, ntimes = NULL, M, trunc, prior, dt_dist = "nonparametric",
  dt_parm, tpm_parm, emit_parm, zero_init, plot = TRUE, xlim = NULL,
  ylim = NULL, ...)
```

Arguments

`y`	the observed series to be decoded
`ntimes`	vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y)
`M`	number of latent states
`trunc`	a vector specifying truncation at the maximum number of dwelling time in each state.
`prior`	a vector of prior probability values
`dt_dist`	dwell time distribution, can only be "log", "shiftedpoisson", or "nonparametric". Default to "nonparametric".
`dt_parm`	a vector of parameter values in each dwell time distribution, which should be a vector of p's for dt_dist == "log", or a vector of theta's for dt_dist=="shiftpoisson", or a matrix whose a matrix whose i,j th element is the probability of staying in state i for duration j for dt_dist == "nonparametric".
`tpm_parm`	transition probability matrix
`emit_parm`	a vector containing means for each poisson distribution
`zero_init`	a vector containing structural zero proportions in each state
`plot`	whether a plot should be returned
`xlim`	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
`ylim`	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
`...`	further arguments to be passed to the plot() function

Value

the decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
## Not run:
#3 zero-inflated poissons
prior_init <- c(0.3,0.3,0.4)
dt_init <- c(10,8,6)
emit_init <- c(10,50,100)
zeroprop <- c(0.5,0.3,0.2)
trunc <- c(10,10,10)
omega <- matrix(c(0,0.3,0.7,0.4,0,0.6,0.5,0.5,0),3,3,byrow=TRUE)
result <- hsmmsim(n=1000,M=3,prior=prior_init,dt_dist="shiftpoisson",
         dt_parm=dt_init, tpm_parm=omega,emit_parm=emit_init,zeroprop=zeroprop)
y <- result$series
state <- result$state
fit <- hsmmfit(y=y,ntimes=NULL,M=3,trunc=trunc,prior_init=prior_init,dt_dist="shiftpoisson",
     dt_init=dt_init,tpm_init=omega,emit_init=emit_init,zero_init=zeroprop,
     method="Nelder-Mead",hessian=FALSE,control=list(maxit=500,trace=1))
decode <- hsmmviterbi(y=y,ntimes=NULL,M=3,trunc=trunc,prior=fit$prior,dt_dist="shiftpoisson",
     dt_parm=fit$dt_parm,tpm_parm=fit$tpm,emit_parm=fit$emit_parm,
     zero_init=fit$zeroprop,plot=TRUE,xlim=c(0,1000),ylim=c(0,200))
#check the missclassification rate
sum(decode!=state)/length(state)

## End(Not run)
```

---

hsmmviterbi2 — *Viterbi algorithm to decode the latent states in hidden semi-Markov models with covariates*

Description

Viterbi algorithm to decode the latent states in hidden semi-Markov models with covariates

Usage

```
hsmmviterbi2(y, ntimes = NULL, M, trunc, workparm,
  dt_dist = "nonparametric", zero_init, prior_x = NULL, tpm_x = NULL,
  emit_x = NULL, zeroinfl_x = NULL, dt_x = NULL, plot = FALSE,
  xlim = NULL, ylim = NULL, ...)
```

Arguments

`y`	the observed series to be decoded
`ntimes`	vector specifying the lengths of individual, i.e. independent, time series. If not specified, the responses are assumed to form a single time series, i.e. ntimes=length(y)
`M`	number of latent states
`trunc`	a vector specifying the truncation at the maximum number of dwelling time in each state.
`workparm`	working parameters. The first part is the logit of parameter p for each log-series dwell time distribution or the log of parameter theta for each shifted-poisson dwell time distribution. If dt_dist is "nonparametric", then the first part is empty. The next part is the generalized logit of prior probabilities (except for the 1st state),the logit of each nonzero structural zero proportions, the log of each state-dependent poisson means, and the generalized logit of the transition probability matrix(except 1st column and the diagonal elements).
`dt_dist`	dwell time distribution, can only be "log", "shiftedpoisson" or "nonparametric". Default to "nonparametric".
`zero_init`	a vector containing structural zero proportions in each state, e.g. set zero_init[i] to be 0 if the i-th state is a regular poisson, and otherwise 1.
`prior_x`	matrix of covariates for generalized logit of prior probabilites (excluding the 1st probability). Default to NULL.
`tpm_x`	matrix of covariates for transition probability matrix (excluding the 1st column). Default to NULL.
`emit_x`	matrix of covariates for the log poisson means. Default to NULL.
`zeroinfl_x`	matrix of covariates for the nonzero structural zero proportions. Default to NULL.
`dt_x`	if dt_dist is "nonparametric", then dt_x is the matrix of nonparametric state durataion probabilities. Otherwise, dt_x is matrix of covariates for the dwell time distribution parameters in log-series or shifted-poisson distributions.Default to NULL.
`plot`	whether a plot should be returned
`xlim`	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
`ylim`	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
`...`	further arguments to be passed to the plot() function

Value

decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
## Not run:
data(CAT)
y <- CAT$activity
x <- data.matrix(CAT$night)
prior_init <- c(0.5,0.3,0.2)
dt_init <- c(0.9,0.6,0.3)
emit_init <- c(10,50,100)
zero_init <- c(0.5,0,0) #assuming only the 1st state has structural zero's
tpm_init <- matrix(c(0,0.3,0.7,0.4,0,0.6,0.5,0.5,0),3,3,byrow=TRUE)
trunc <- c(10,7,4)
fit2 <-  hsmmfit(y,rep(1440,3),3,trunc,prior_init,"log",dt_init,tpm_init,
     emit_init,zero_init,emit_x=x,zeroinfl_x=x,hessian=FALSE,
     method="Nelder-Mead", control=list(maxit=500,trace=1))
decode <- hsmmviterbi2(y,rep(1440,3),3,trunc,fit2$working_parameters,
            dt_dist="log", zero_init=c(1,0,0),
            emit_x=x,zeroinfl_x=x, plot=TRUE, xlab="time", ylab="count",
            xlim=c(0,360),ylim=c(0,200))

## End(Not run)
```

---

hsmmviterbi_exp *Viterbi algorithm to decode the latent states in hidden semi-Markov models with covariates where the latent state durations have accelerated failure time structure*

Description

Viterbi algorithm to decode the latent states in hidden semi-Markov models with covariates where the latent state durations have accelerated failure time structure

Usage

```
hsmmviterbi_exp(y, M, trunc, dtrate, dtparm, prior, zeroparm, emitparm, tpmparm,
  dt_x, zeroinfl_x, emit_x, tpm_x, plot = FALSE, xlim = NULL, ylim = NULL,
  ...)
```

Arguments

y	the observed series to be decoded
M	number of latent states
trunc	a vector specifying the truncation at the maximum number of dwelling time in each state.
dtrate	a vector for the scale parameters in the base exponential density for the latent state durations.
dtparm	a matrix of coefficients for the accelerated failure time model in each latent state

prior	a vector of prior probabilities
zeroparm	a vector of regression coefficients for the structural zero proportion in state 1
emitparm	a matrix of regression coefficients for the Poisson regression in each state
tpmparm	a vector of coefficients for the multinomial logistic regression in the transition probabilities
dt_x	a matrix of covariates for the latent state durations
zeroinfl_x	a matrix of covariates for the zero proportion
emit_x	a matrix of covariates for the Poisson means
tpm_x	a matrix of covariates for the transition
plot	whether a plot should be returned
xlim	vector specifying the minimum and maximum on the x-axis in the plot. Default to NULL.
ylim	vector specifying the minimum and maximum on the y-axis in the plot. Default to NULL.
...	further arguments to be passed to the plot() function

Value

decoded series of latent states

References

Walter Zucchini, Iain L. MacDonald, Roland Langrock. Hidden Markov Models for Time Series: An Introduction Using R, Second Edition. Chapman & Hall/CRC

Examples

```
## Not run:
M <- 3
prior <- c(0.5,0.3,0.2)
dtrate <- c(6,5,4)
dtparm <- matrix(c(0.2,0.1,0.2),nrow=3)
zeroparm <- c(0,-0.2)
emitparm <- matrix(c(4,0.3,5,0.2,6,-0.1),3,2,byrow=TRUE)
tpmparm <- c(1,0.2,0.5,-0.2,0,0.2)

emit_x <- matrix(c(rep(1,1000),rep(0,1000)),nrow=2000,ncol=1)
dt_x <- emit_x
tpm_x <- emit_x
zeroinfl_x <- emit_x
trunc <- c(18,15,10)

re <- hsmmsim2_exp(prior,dtrate,dtparm,zeroparm,emitparm,tpmparm,
                  trunc, M, n, dt_x,tpm_x, emit_x, zeroinfl_x)
y <- re$series

rrr <- hsmmfit_exp(y,M,trunc,dtrate,dtparm,prior,zeroparm,emitparm,tpmparm,
```

```
                  dt_x,zeroinfl_x,emit_x,tpm_x,method="BFGS",control=list(trace=1))

decode <- hsmmviterbi_exp(y,M, trunc,dtrate,dtparm,
                          prior,zeroparm,emitparm,tpmparm,
                          dt_x, zeroinfl_x, emit_x, tpm_x)
sum(decode!=re$state)

## End(Not run)
```

`package-ziphsmm` *zero-inflated poisson hidden (semi-)Markov models*

Description

Package:	ziphsmm
Type:	Package
Version:	2.0.6
Date:	2018-05-08
License:	GPL-2
LazyLoad:	yes
LazyData:	yes

Author(s)

Zekun Xu <zekunxu@gmail.com>

Ye Liu Maintainer: Zekun Xu <zekunxu@gmail.com>

`retrieve_cov_cont` *retrieve the natural parameters from the working parameters in zero-inflated Poisson hidden Markov model with covariates, where zero-inflation only happens in state 1*

Description

retrieve the natural parameters from the working parameters in zero-inflated Poisson hidden Markov model with covariates, where zero-inflation only happens in state 1

Usage

```
retrieve_cov_cont(parm, M, ncolx)
```

Arguments

`parm`	working parameters
`M`	number of hidden states
`ncolx`	number of covariates including the intercept

Value

a list of natural parameters

`retrieve_cov_cont3` *retrieve the natural parameters from the working parameters in zero-inflated Poisson hidden Markov model with covariates in state-dependent parameters and transition rates*

Description

retrieve the natural parameters from the working parameters in zero-inflated Poisson hidden Markov model with covariates in state-dependent parameters and transition rates

Usage

```
retrieve_cov_cont3(parm, M, ncolx)
```

Arguments

`parm`	working parameters
`M`	number of hidden states
`ncolx`	number of covariates including the intercept

Value

a list of natural parameters

retrieve_nocov_cont

retrieve the natural parameters from working parameters for a continuous-time zero-inflated Poisson hidden Markov model where zero-inflation only happens in state 1

Description

retrieve the natural parameters from working parameters for a continuous-time zero-inflated Poisson hidden Markov model where zero-inflation only happens in state 1

Usage

```
retrieve_nocov_cont(parm, M)
```

Arguments

parm	working parameters
M	number of hidden states

Value

a list of natural parameters

rzip

generate zero-inflated poisson random variables

Description

generate zero-inflated poisson random variables

Usage

```
rzip(n, p, theta)
```

Arguments

n	length of the random series
p	proportion of structural zero's
theta	the poisson mean

Value

a series of zero-inflated poisson random variables

zipnegloglik_cov_cont *negative log likelihood function for zero-inflated Poisson hidden Markov model with covariates, where zero-inflation only happens in state 1*

Description

negative log likelihood function for zero-inflated Poisson hidden Markov model with covariates, where zero-inflation only happens in state 1

Usage

```
zipnegloglik_cov_cont(parm, y, covariates, M, ntimes, timeindex, udiff)
```

Arguments

parm	working parameters
y	observed series
covariates	design matrix of covariates including the intercept
M	number of hidden states
ntimes	length of the observed series
timeindex	vector of observed time points
udiff	unique time intervals

Value

negative log likelihood

zipnegloglik_cov_cont3 *negative log likelihood function for zero-inflated Poisson hidden Markov model with covariates in state-dependent parameters and transition rates*

Description

negative log likelihood function for zero-inflated Poisson hidden Markov model with covariates in state-dependent parameters and transition rates

Usage

```
zipnegloglik_cov_cont3(parm, y, covariates, M, ntimes, timeindex)
```

Arguments

`parm`	working parameters
`y`	observed series
`covariates`	design matrix of covariates including the intercept
`M`	number of hidden states
`ntimes`	length of the observed series
`timeindex`	vector of observed time points

Value

negative log likelihood

`zipnegloglik_nocov_cont` *negative log likelihood function for zero-inflated Poisson hidden Markov model without covariates, where zero-inflation only happens in state 1*

Description

negative log likelihood function for zero-inflated Poisson hidden Markov model without covariates, where zero-inflation only happens in state 1

Usage

```
zipnegloglik_nocov_cont(parm, M, y, ntimes, timeindex, udiff)
```

Arguments

`parm`	working parameters
`M`	number of hidden states
`y`	observed series
`ntimes`	length of the observed series
`timeindex`	vector of observed time points
`udiff`	unique time intervals

Value

negative log likelihood

Index