Package ‘PHSMM’

February 9, 2021

Type Package
Title Penalised Maximum Likelihood Estimation for Hidden Semi-Markov Models
Version 1.0
Date 2021-01-28
Author Jennifer Pohle
Maintainer Jennifer Pohle <jennifer.pohle@uni-bielefeld.de>
License GPL-3
Imports Rcpp (>= 1.0.5)
Depends R (>= 4.0.0)
LinkingTo Rcpp, RcppArmadillo
LazyData true
NeedsCompilation yes
Repository CRAN
Date/Publication 2021-02-09 10:10:02 UTC

R topics documented:

Bernoulli distribution ......................................................... 2
decodeHSMM ................................................................. 3
muskox ............................................................................ 4
n2wHSMM ........................................................................... 5
nll_Rcpp ............................................................................ 7
npliHSMM .......................................................................... 7
Bernoulli distribution

Description

Probability mass function and cumulative distribution function of the Bernoulli distribution.

Usage

dbern(y, prob)
pbern(y, prob)

Arguments

y a vector of zeros and ones.
prob probability.

Details

The code relies on the functions dbinom and pbinom with size=1 and log=FALSE.

Value

dbern returns the probability mass function, pbern returns the cumulative distribution function.

References

See the documentation for dbinom and pbinom for more details.

Examples

dbern(c(0,1), 0.4)
pbern(c(0,1), 0.4)
**decodeHSMM**

*State decoding*

**Description**

State decoding for the HSMM estimated using `pmleHSM`. Decoding is based on the Viterbi algorithm and the corresponding HMM model representation.

**Usage**

```r
decodeHSMM(y, mod)
```

**Arguments**

- `y` vector containing the observed time series.
- `mod` model object as returned by `pmleHSM`.

**Value**

Returns a vector containing the decoded states.

**References**

For more details about the Viterbi algorithm, see for example:


**Examples**

```r
# running this example might take a few minutes
#
# 1.) 2-state gamma-HSMM for hourly muskox step length
# with an unstructured start of length of 10
#
# initial values
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
mu0<-c(5,150)
sigma0<-c(3,180)
#
# fit 2-state gamma-HSMM with lambda=c(100,100)
# and difference order 3
# estimation might take a few minutes
PHSMM<-pmleHSM(y=muskox$step,N=2,p_list=p_list0,mu=mu0,
sigma=sigma0,lambda=c(100,100),order_diff=3,
y_dist='gamma')
```
# state decoding
s_HSMM<-decodeHSMM(muskox$step,mod=PHSMM)
# plot sequence of the decoded time series
plot(muskox$step[1:1000],type='h',xlab='time (h)',ylab='step (m)',
     main=' ',col=s_HSMM)
legend('topright',c('state 1','state 2'),lwd=2,col=1:2)

# running this example might take a few minutes
#
# 2.) 3-state gamma-HSMM for hourly muskox step length
# with an unstructured start of length of 10
#
# initial values
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[3]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
omega0<-matrix(0.5,3,3)
diag(omega0)<-0
mu0<-c(5,100,350)
sigma0<-c(3,90,300)

# fit 3-state gamma-HSMM with lambda=c(1000,1000,1000)
# and difference order 3
# estimation might take some minutes
PHSMM<-pmleHSMM(y=muskox$step,N=3,p_list=p_list0,mu=mu0,
                 sigma=sigma0,omega=omega0,
                 lambda=c(1000,1000,1000),
                 order_diff=3,y_dist='gamma')

# state decoding
s_HSMM<-decodeHSMM(muskox$step,mod=PHSMM)
# plot sequence of the decoded time series
plot(muskox$step[1:1000],type='h',xlab='time (h)',ylab='step (m)',
     main=' ',col=s_HSMM)
legend('topright',c('state 1','state 2' ,'state 3'),lwd=2,col=1:3)

---

**muskox**

*Muskox movement data*

**Description**

Example data based on the movement of a muskox tracked in northeast Greenland.

**Usage**

muskox
Format

A data frame with 6825 GPS-based observations and 5 columns:

- date: date
- tday: time of day
- x: UTM easting coordinate
- y: UTM northing coordinate
- step: hourly step length calculated based on the coordinates of time t and t+1

Source


References

The data set is based on a subset of the data described and analysed in:


---

n2wHSMM Parameter transformation from natural to working parameters

Description

Parameter transformation from the natural (constraint) HSMM parameters into unconstraint working parameters which are used in the numerical maximum likelihood estimation. Not intended to be run by the user (internal function, called by the function pmleHSMM).

Usage

n2wHSMM(N, p_list, mu, sigma=NULL, omega=NULL, delta=NULL, y_dist=c("norm","gamma","pois","bern"), stationary=TRUE, p_ref=2)

Arguments

N number of states of the HSMM, integer greater than 1.

p_list list containing the parameters of the states’ dwell-time distributions. The list consists of N probability vectors, i.e. one vector for each state. Each probability vector contains the state dwell-time probabilities for the unstructured start and, as last element, the probability mass captured in the geometric tail. Thus, each vector must sum to one and automatically determines the length of the unstructured start of the according state dwell-time distribution.
mu vector of length \( N \) containing the state-dependent mean values if gamma, normal or Poisson distributions are chosen and the state-dependent probabilities if the Bernoulli distribution is chosen. The values must be sorted in an ascending order.

sigma vector of length \( N \) containing the state-dependent standard deviations if gamma or normal distributions are chosen. NULL (default) otherwise.

omega conditional transition probability matrix of the underlying semi-Markov chain. Only needed if the number of states is greater than 2, NULL (default) otherwise. In the former case, omega is a matrix with \( N \) rows and \( N \) columns, its diagonal elements must be zero and its rows must sum to one.

delta vector of length \( N \) containing the initial distribution. Only needed if stationary=FALSE, otherwise delta is ignored and can be set to NULL (default).

y_dist character determining the class of state-dependent distributions. Supported values are "norm" (normal distribution), "gamma" (gamma distribution), "pois" (Poisson distribution) and "bern" (Bernoulli distribution).

stationary Logical, if TRUE (default), stationarity is assumed, if FALSE, the underlying state-sequence is assumed to enter a new state at time \( t=1 \) and it is necessary to define the initial distribution delta.

p_ref positive integer determining the reference dwell-time probability used for the multinomial logit parameter transformation. Default value is 2. Only needs to be changed if the dwell-time probability for dwell time \( r=2 \) is estimated very close to zero in order to avoid numerical problems.

Details

The transformation from natural to working parameters is needed to carry out an unconstraint optimisation. The function includes log-transformations for positive parameters and (multinomial) logit-transformations for probabilities, probability vectors and matrices.

Value

A vector of unconstraint working parameters characterising the HSMM.

Examples

# natural parameters for 2-state HSMM with state-dependent normal distributions
p_list0<-list() # list of dwell-time distribution vectors,
# vector elements must sum to one
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.1),1-pgeom(9,0.1))
mu0<-c(-10,10) # mean values
sigma0<-c(3,5) # standard deviations
# parameter transformation:

n2wHSMM(N=2,p_list=p_list0,mu=mu0,sigma=sigma0,y_dist='norm',stationary=TRUE)
**nll_Rcpp**

Negative log-likelihood evaluation

Description

Evaluation of the negative HMM log-likelihood function based on the forward algorithm, written in C++ (internal function, called by the function `npl1HSMM`).

Usage

```
nll_Rcpp(allprobs, gamma, delta, T_y)
```

Arguments

- `allprobs`: matrix containing the state-dependent distribution values for each observation (row) and state (column), respectively.
- `gamma`: transition probability matrix.
- `delta`: initial distribution.
- `T_y`: number of observations.

Value

Returns the negative log-likelihood value.

---

**npl1HSMM**

negative penalised HSMM log-likelihood function

Description

Evaluates the negative penalised log-likelihood function of the HSMM (internal function, called by the function `pmleHSMM`).

Usage

```
npl1HSMM(parvect, N, y, R_vec, lambda, order_diff,
y_dist=c("norm", "gamma", "pois", "bern"),
stationary=TRUE, T_y, p_ref=2)
```
Arguments

parvect vector of unconstraint working parameter as returned by the function `n2wHSMM`.

N number of states of the HSMM, integer greater than 1.

y vector containing the observed time series.

R_vec vector of length N containing the lengths of the unstructured starts of the dwell-time distributions.

lambda vector of length N containing the smoothing parameter values to weight the penalty term.

order_diff order of the differences used for the penalty term, positive integer which does not exceed the length of the unstructured starts.

y_dist character determining the class of state-dependent distributions. Supported values are "norm" (normal distribution), "gamma" (gamma distribution), "pois" (Poisson distribution) and "bern" (Bernoulli distribution).

stationary Logical, if TRUE (default), stationarity is assumed, if FALSE, the underlying state-sequence is assumed to enter a new state at time t=1.

T_y length of the observed time series.

p_ref positive integer determining the reference dwell-time probability used for the multinomial logit parameter transformation. Default value is 2. Only needs to be changed if the dwell-time probability for dwell time \( r=2 \) is estimated very close to zero in order to avoid numerical problems.

Details

The penalised log-likelihood function relies on the exact HMM representation of the HSMM and is evaluated using the forward algorithm which is implemented in C++ to speed up the calculation.

Value

Returns the value of the negative penalised HSMM log-likelihood function for the given parameters and time series.

References


Examples

```r
# 3-state gamma HSMM and hourly muskox step length
# natural parameters
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[3]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
```
omega0 <- matrix(0.5, 3, 3)
diag(omega0) <- 0
mu0 <- c(5, 100, 350)
sigma0 <- c(3, 90, 300)
R_vec <- sapply(p_list0, length) - 1  # lengths of the unstructured starts

# working parameter vector
parvect <- n2wHSMM(N = 3, p_list = p_list0, mu = mu0, sigma = sigma0,
                   omega = omega0, y_dist = 'gamma')

# evaluate the negative penalised log-likelihood function
npllHSMM(parcvect, N = 3, muskox$step, R_vec = R_vec, lambda = c(1000, 1000, 1000),
         order_diff = 2, y_dist = 'gamma', T_y = nrow(muskox))

---

**plotDw**  
*Plot of the estimated HSMM dwell-time distributions.*

**Description**
Plots the HSMM dwell-time distributions estimated using `pmleHSMM`.

**Usage**
```r
plotDw(mod, R_max, state = 'all', mfrow = NULL)
```

**Arguments**
- **mod**: model object as returned by `pmleHSMM`.
- **R_max**: integer, maximum dwell time for which the dwell-time probabilities are plotted.
- **state**: value determining the states for which the distributions are plotted. Either "all" (default) for plotting the dwell-time distributions of all states, or positive integer in 1,...,N.
- **mfrow**: If NULL (default) and state="all", the probability mass functions are plotted one below the other. Otherwise, a vector of length 2 which determines the number of rows (first element) and the number of columns (second argument) of the matrix of plots.

**Value**
Plot of the estimated HSMM dwell-time distributions.

**Examples**
```r
# running this example might take a few minutes
#
# 1.) 2-state gamma-HSMM for hourly muskox step length
# with an unstructured start of length of 10
#
# initial values
```
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
mu0<-c(5,150)
sigma0<-c(3,180)

# fit 2-state gamma-HSMM with lambda=c(100,100)
# and difference order 3
# estimation might take a few minutes
PHSMM<-pmleHSMM(y=muskox$step,N=2,p_list=p_list0,mu=mu0,
sigma=sigma0,lambda=c(100,100),order_diff=3,
y_dist="gamma")

# plot the estimated dwell-time distributions
# for dwell-times up to 12
plotDw(mod=PHSMM,R_max=12)
plotDw(mod=PHSMM,R_max=12,state=1)
plotDw(mod=PHSMM,R_max=12,mfrow=c(1,2))

# running this example might take a few minutes
#
# 2.) 3-state gamma-HSMM for hourly muskox step length
# with an unstructured start of length of 10
#
# initial values
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[3]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
omega0<-matrix(0.5,3,3)
diag(omega0)<-0
mu0<-c(5,100,350)
sigma0<-c(3,90,300)

# fit 3-state gamma-HSMM with lambda=c(1000,1000,1000)
# and difference order 3
# estimation might take some minutes
PHSMM<-pmleHSMM(y=muskox$step,N=3,p_list=p_list0,mu=mu0,
sigma=sigma0,omega=omega0,
lambda=c(1000,1000,1000),
order_diff=3,y_dist="gamma")

# plot the estimated dwell-time distributions
# for dwell-times up to 15
plotDw(mod=PHSMM,R_max=15)
plotDw(mod=PHSMM,R_max=15,state=1)
plotDw(mod=PHSMM,R_max=15,mfrow=c(1,3))
pmleHSMM

HSMM penalised maximum likelihood estimation

Description

Estimates the parameters of a hidden semi-Markov model (HSMM) for univariate time series using numerical penalised maximum likelihood estimation. The dwell times are modelled using distributions with an unstructured start and a geometric tail. During the estimation, (higher-order) differences between adjacent dwell-time probabilities are penalised to derive smooth and flexible estimates. The function allows for normal-, gamma-, Poisson- or Bernoulli-distributions in the state-dependent process.

Usage

pmleHSMM(y, N, p_list, mu, sigma=NULL, omega=NULL, delta=NULL, lambda, order_diff, y_dist=c("norm","gamma","pois","bern"), stationary=TRUE, p_ref=2, print.level=0, iterlim=10000, stepmax=NULL, hessian=FALSE, gradtol=10^(-6))

Arguments

y        vector containing the observed time series.
N        number of states of the HSMM, integer greater than 1.
p_list   list of vectors containing the starting values for the state dwell-time distributions. The list comprises N vectors, i.e. one vector for each state. Each vector contains the state’s dwell-time probabilities for the unstructured start and, as last element, the probability mass captured in the geometric tail. Thus, each vector must sum to one and automatically determines the length of the unstructured start of the according state dwell-time distribution.
mu       starting values for the state-dependent mean values if normal, gamma or Poisson distributions are used to model the state-dependent observations. State-dependent probabilities if Bernoulli distributions are chosen. The vector is of length N and the values must be sorted in an ascending order (to avoid label switching).
sigma    starting values for the state-dependent standard deviations if gamma or normal state-dependent distributions are used. In that case, sigma is a vector of length N, otherwise it is NULL (default).
omega    starting values for the conditional transition probability matrix of the underlying semi-Markov chain. Only needed if the number of states exceeds 2, otherwise it is NULL (default). In the former case, omega is a matrix with N rows and N columns, its diagonal elements must be zero and its rows must sum to one.
delta    starting values for the initial distribution of the underlying semi-Markov chain if stationary=FALSE. In that case, delta is a vector of dimension N and its elements must sum to one. For stationary=TRUE, delta is ignored and can be set to NULL (default).
lambda vector of length N containing the smoothing parameter values to weight the penalty term.

order_diff order of the differences used for the penalty term, positive integer which does not exceed the length of the unstructured starts (as determined by p_list).

y_dist character determining the class of state-dependent distributions used to model the observations. Supported values are "norm" (normal distribution), "gamma" (gamma distribution), "pois" (Poisson distribution) and "bern" (Bernoulli distribution).

stationary Logical, if TRUE (default), stationarity is assumed, if FALSE, an initial distribution is estimated and the underlying state-sequence is assumed to enter a new state at time t=1.

p_ref positive integer determining the reference dwell-time probability used for the multinomial logit parameter transformation. Default value is 2. Only needs to be changed if the dwell-time probability for dwell time r=2 is estimated very close to zero in order to avoid numerical problems.

print.level print level for the optimisation procedure nlm. Default value is 0 corresponding to no printing. See nlm for more details.

iterlim maximum number of iterations for the optimisation procedure nlm. Default value is 10000.

stepmax stepmax value for nlm. The default value NULL corresponds to the nlm default. See nlm for more details.

hessian Logical, if TRUE, the hessian matrix is calculated and returned by nlm, if FALSE (default), the hessian is not calculated.

gradtol tolerance value for a convergence criterion used by nlm. Default value is 10^(-6). See nlm for more details.

Details

The numerical penalised maximum likelihood estimation requires starting values for each HSMM parameter. If these starting values are poorly chosen, the algorithm might fail in finding the global optimum of the penalised log-likelihood function. Therefore, it is highly recommended to repeat the estimation several times using different sets of starting values.

The maximisation of the penalised log-likelihood function is carried out using the optimisation routine nlm. The likelihood evaluation is written in C++ to speed up the estimation.

Value

An HSMM model object, i.e. a list containing:

p_list list containing the penalised maximum likelihood estimates for the state dwell-time distributions.

mu vector containing the penalised maximum likelihood estimates for the state-dependent mean values or state-dependent probabilities, depending on the chosen class of state-dependent distributions.
sigma vector containing the penalised maximum likelihood estimates for the state-dependent standard deviations if state-dependent gamma or normal distributions are used.
delta vector containing the equilibrium distribution if stationary=TRUE, otherwise vector of the estimated initial distribution.
omega penalised maximum likelihood estimate of the conditional HSMM transition probability matrix.
Gamma transition probability matrix corresponding to the HMM representation of the estimated HSMM.
npl1 minimum negative penalised log likelihood value as found by nlm.
gr gradient of the negative penalised log-likelihood function as returned by nlm.
iterations number of iterations until convergence.
code_conv convergence code as returned by nlm.
w_parvect vector of the penalised maximum likelihood working parameters.
stationary Logical, as specified for the estimation.
y_dist state-dependent distribution, as specified for the estimation.

References

See nlm for further details on the numerical optimisation routine.

For details on the model formulation and likelihood function, see:


Examples

# running this example might take a few minutes
#
# 1.) fit 2-state gamma-HSMM to hourly muskox step length
# using a length of 10 for the unstructured start
#
# initial values
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
mu0<-c(5,150)
sigma0<-c(3,180)
#
# fit 2-state gamma-HSMM with lambda=c(100,100)
# and difference order 3
# estimation might take a few minutes
pseudoResHSMM <- pmleHSMM(y = muskox$step, N = 2, p_list = p_list0, mu = mu0,
  sigma = sigma0, lambda = c(100, 100), order_diff = 3,
y_dist = 'gamma')

# running this example might take a few minutes
#
# 2.) fit 3-state gamma-HSMM to hourly muskox step length
# using a length of 10 for the unstructured start
#
# initial values
p_list0 <- list()
p_list0[[1]] <- c(dgeom(0:9, 0.2), 1 - pgeom(9, 0.2))
p_list0[[2]] <- c(dgeom(0:9, 0.2), 1 - pgeom(9, 0.2))
p_list0[[3]] <- c(dgeom(0:9, 0.2), 1 - pgeom(9, 0.2))
omega0 <- matrix(0.5, 3, 3)
diag(omega0) <- 0
mu0 <- c(5, 1, 100, 350)
sigma0 <- c(3, 9, 300)

# fit 3-state gamma-HSMM with lambda = c(1000, 1000, 1000)
# and difference order 3
# estimation might take some minutes
PHSMM <- pmleHSMM(y = muskox$step, N = 3, p_list = p_list0, mu = mu0,
  sigma = sigma0, omega = omega0,
  lambda = c(1000, 1000, 1000),
  order_diff = 3, y_dist = 'gamma')

pseudoResHSMM  

HSMM pseudo-residuals

Description
Pseudo-residuals based on the one-step ahead forecast distributions under the HSMM which was
estimated using pmleHSMM. This function can only be used for HSMMs with state-dependent normal
or gamma distributions.

Usage
pseudoResHSMM(y, mod)

Arguments

y  
vector containing the observations.

mod 
model object as returned by pmleHSMM.
Details

A good model fit is indicated by standard normally distributed pseudo-residuals.

Value

Returns a vector containing the forecast pseudo-residuals.

References

For more details about pseudo-residuals in the context of HMMs, see:


Examples

```r
# running this example might take a few minutes
#
# 1.) 2-state gamma-HSMM for hourly muskox step length
# with an unstructured start of length of 10
#
# initial values
p_list0<-list()
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
mu0<-c(5,150)
sigma0<-c(3,180)
#
# fit 2-state gamma-HSMM with lambda=c(100,100)
# and difference order 3
# estimation might take a few minutes
PHSMM<-pmleHSMM(y=muskox$step,N=2,p_list=p_list0,mu=mu0,
                  sigma=sigma0,lambda=c(100,100),order_diff=3,
                  y_dist='gamma')
#
# pseudo-residuals
pseudoRes<-pseudoResHSMM(y=muskox$step,PHSMM)
hist(pseudoRes,probability=TRUE)
z<-seq(-3,3,0.01)
lines(z,dnorm(z),col='blue')

# running this example might take a few minutes
#
# 2.) 3-state gamma-HSMM for hourly muskox step length
# with an unstructured start of length of 10
#
# initial values
p_list0<-list()
```
tpmHMM

Transition probability matrix of the HMM representation

Description

Construction of the transition probability matrix corresponding to the HMM which exactly represents the HSMM. Not intended to be run by the user (internal function, called by the functions `w2nHSMM` and `pmleHSMM`).

Usage

```r
tpmHMM(N, omega, d_r, R_vec, eps=1e-10)
```

Arguments

- **N**: number of states of the HSMM, integer greater than 1.
- **omega**: conditional transition probability matrix of the HSMM with N rows and N columns. The diagonal elements must be zero and the rows must sum to one.
- **d_r**: list of vectors containing the dwell-time probabilities of the unstructured starts.
- **R_vec**: vector of length N containing the lengths of the unstructured starts of the state dwell-time distributions.
- **eps**: to avoid negative probabilities due to numerical underflow. Default is `1e-10`.

Value

Returns the transition probability matrix of the HMM which exactly represents the HSMM.
References

For details on the code and construction of the matrix, see:
Langrock, R. and Zucchini W. (2011): Hidden Markov models with arbitrary state dwell-time dis-

Examples

```r
# list of dwell-time probability vectors
# (vector elements should not sum to one)
d_r<-list()
d_r[[1]]<-c(dgeom(0:9,0.2))
d_r[[2]]<-c(dgeom(0:9,0.1))

# transition probability matrix:
Gamma<-tpmHMM(N=2,omega=matrix(c(0,1,1,0),2,2),
               d_r=d_r,R_vec=sapply(d_r,length))
```

**Description**

Transforms unconstraint HSMM working parameters back into (constraint) natural parameters. Not intended to be run by the user (internal function, called by the functions `pmleHMM` and `npHMM`).

**Usage**

```r
w2nHSM(N, parvect, R_vec, y_dist=c("norm","gamma","pois","bern"),
       stationary=TRUE, p_ref=2)
```

**Arguments**

- `N`  number of states of the HSMM, integer greater than 1.
- `parvect`  vector of unconstraint working parameter as obtained by the function `n2wHSM`.
- `R_vec`  vector of length `N` containing the lengths of the unstructured starts of the dwell-time distributions.
- `y_dist`  character determining the class of state-dependent distributions used to model the observations. Supported values are "norm" (normal distribution), "gamma" (gamma distribution), "pois" (Poisson distribution) and "bern" (Bernoulli distribution).
- `stationary`  Logical, if TRUE (default), stationarity is assumed, if FALSE, the underlying state-sequence is assumed to enter a new state at time t=1.
- `p_ref`  positive integer determining the reference dwell-time probability used for the multinomial logit parameter transformation. Default value is 2. Only needs to be changed if the dwell-time probability for dwell time r=2 is estimated very close to zero in order to avoid numerical problems.
Details

The function reverses the transformation of the function `n2wHSMM` and back-transforms the unconstrained parameters into the constraint natural parameters. Note that if \( y_{\text{dist}} = \text{gamma} \), \( \mu \) and \( \sigma \) do not include the mean values and standard deviations, but the shape and rate parameters as required by the density functions \( \text{dgamma} \) and \( \text{pgamma} \). The mean and standard deviations are then assigned to \( \mu_2 \) and \( \sigma_2 \).

Value

A list containing the natural parameters

- **p_list**: list containing the dwell-time distribution vectors for each state. Each of the \( N \) vectors contains the state dwell-time probabilities for the unstructured start and, as last element, the probability mass captured in the geometric tail. Thus, each vector sums to one.
- **mu**: vector of length \( N \). For \( y_{\text{dist}} = \text{norm} \) and \( y_{\text{dist}} = \text{pois} \), it contains the state-dependent mean values, for \( y_{\text{dist}} = \text{gamma} \), it contains the state-dependent shape parameters, and for \( y_{\text{dist}} = \text{bern} \), it contains the state-dependent probabilities.
- **sigma**: vector of length \( N \) containing the state-dependent standard deviations if \( y_{\text{dist}} = \text{norm} \) and the state-dependent rates if \( y_{\text{dist}} = \text{gamma} \). NULL otherwise.
- **omega**: conditional transition probability matrix of the HSMM.
- **delta**: equilibrium distribution if \( \text{stationary} = \text{TRUE} \), initial distribution of length \( N \) if \( \text{stationary} = \text{FALSE} \).
- **d_r**: list containing the dwell-time probabilities of the unstructured starts.
- **Gamma**: transition probability matrix of the HMM which represents the HSMM.

Examples

```r
# natural parameters for 2-state HSMM with state-dependent normal distributions
p_list0<-list() # list of dwell-time distribution vectors, # vector elements must sum to one
p_list0[[1]]<-c(dgeom(0:9,0.2),1-pgeom(9,0.2))
p_list0[[2]]<-c(dgeom(0:9,0.1),1-pgeom(9,0.1))
mu0<-c(-10,10) # mean values
sigma0<-c(3,5) # standard deviations # parameter transformation:
parvec<-n2wHSMM(N=2,p_list=p_list0,mu=mu0,sigma=sigma0,y_dist='norm',stationary=TRUE)
# back-transformation:
par<-w2nHSMM(N=2,parvec=parvec,R_vec=sapply(p_list0,length)-1,y_dist='norm')
```
Index

Bernoulli distribution, 2

dbern (Bernoulli distribution), 2
dbinom, 2
decodeHSMM, 3
dgamma, 18

muskox, 4

n2wHSMM, 5, 8, 17, 18
nll_Rcpp, 7
nlm, 12, 13
npllHSMM, 7, 7, 17

pbern (Bernoulli distribution), 2
pbinom, 2
pgamma, 18
plotDw, 9
pmleHSMM, 3, 5, 7, 9, 11, 14, 16, 17
pseudoResHSMM, 14

tpmHMM, 16

w2nHSMM, 16, 17