Package 'mHMMbayes'

October 30, 2019

Type Package

Title Multilevel Hidden Markov Models Using Bayesian Estimation

Version 0.1.1

Depends R (>= 3.5.0)

Imports MCMCpack, mvtnorm, stats, Rdpack

Maintainer Emmeke Aarts <e.aarts@uu.nl>

Description An implementation of the multilevel (also known as mixed or random effects) hidden Markov model using Bayesian estimation in R. The multilevel hidden Markov model (HMM) is a generalization of the well-known hidden Markov model, for the latter see Rabiner (1989) <doi:10.1109/5.18626>. The multilevel HMM is tailored to accommodate (intense) longitudinal data of multiple individuals simultaneously, see e.g., de Haan-Rietdijk et al. <doi:10.1080/00273171.2017.1370364>. Using a multilevel framework, we allow for heterogeneity in the model parameters (transition probability matrix and conditional distribution), while estimating one overall HMM. The model can be fitted on multivariate data with a categorical distribution, and include individual level covariates (allowing for e.g., group comparisons on model parameters). Parameters are estimated using Bayesian estimation utilizing the forward-backward recursion within a hybrid Metropolis within Gibbs sampler. The package also includes various visualization options, a function to simulate data, and a function to obtain the most likely hidden state sequence for each individual using the Viterbi algorithm.

URL https://github.com/emmekeaarts/mHMMbayes

BugReports https://github.com/emmekeaarts/mHMMbayes/issues

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, alluvial, grDevices, RColorBrewer, testthat (>= 2.1.0)

VignetteBuilder knitr

RdMacros Rdpack

NeedsCompilation no

Author Emmeke Aarts [aut, cre]

Repository CRAN

Date/Publication 2019-10-30 11:30:08 UTC

R topics documented:

`mHMM` *Multilevel hidden Markov model using Bayesian estimation*

Description

`mHMM` fits a multilevel (also known as mixed or random effects) hidden Markov model (HMM) to intense longitudinal data with categorical observations of multiple subjects using Bayesian estimation, and creates an object of class mHMM. By using a multilevel framework, we allow for heterogeneity in the model parameters between subjects, while estimating one overall HMM. The function includes the possibility to add covariates at level 2 (i.e., at the subject level) and have varying observation lengths over subjects. For a short description of the package see mHMMbayes. See `vignette("tutorial-mhmm")` for an introduction to multilevel hidden Markov models and the package, and see `vignette("estimation-mhmm")` for an overview of the used estimation algorithms.

Usage

```
mHMM(s_data, gen, xx = NULL, start_val, mcmc, return_path = FALSE,
  print_iter = FALSE, gamma_hyp_prior = NULL, emiss_hyp_prior = NULL,
  gamma_sampler = NULL, emiss_sampler = NULL)
```

Arguments

`s_data`
: A matrix containing the observations to be modelled, where the rows represent the observations over time. In `s_data`, the first column indicates subject id number. Hence, the id number is repeated over rows equal to the number of observations for that subject. The subsequent columns contain the dependent variable(s). Note that the dependent variables have to be numeric, i.e., they cannot be a (set of) factor variable(s). The total number of rows are equal to the sum over the number of observations of each subject, and the number of columns are equal to the number of dependent variables (`n_dep`) + 1. The number of observations can vary over subjects.

`gen`
: List containing the following elements denoting the general model properties:

- `m`: numeric vector with length 1 denoting the number of hidden states
- `n_dep`: numeric vector with length 1 denoting the number of dependent variables
- `q_emiss`: numeric vector with length `n_dep` denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.

`xx`
: An optional list of (level 2) covariates to predict the transition matrix and/or the emission probabilities. Level 2 covariate(s) means that there is one observation per subject of each covariate. The first element in the list `xx` is used to predict the transition matrix. Subsequent elements in the list are used to predict the emission distribution of (each of) the dependent variable(s). Each element in the list is a matrix, with the number of rows equal to the number of subjects. The first column of each matrix represents the intercept, that is, a column only consisting of ones. Subsequent columns correspond to covariates used to predict the transition matrix / emission distribution. See *Details* for more information on the use of covariates.

If `xx` is omitted completely, `xx` defaults to `NULL`, resembling no covariates. Specific elements in the list can also be left empty (i.e., set to `NULL`) to signify that either the transition probability matrix or a specific emission distribution is not predicted by covariates.

`start_val`
: List containing the start values for the transition probability matrix gamma and the emission distribution(s). The first element of the list contains a `m` by `m` matrix with the start values for gamma. The subsequent elements contain `m` by `q_emiss[k]` matrices for the start values for each of the `k` emission distribution(s). Note that `start_val` should not contain nested lists (i.e., lists within lists).

`mcmc`
: List of Markov chain Monte Carlo (MCMC) arguments, containing the following elements:

- `J`: numeric vector with length 1 denoting the number of iterations of the MCMC algorithm
- `burn_in`: numeric vector with length 1 denoting the burn-in period for the MCMC algorithm.

`return_path`
: A logical scalar. Should the sampled state sequence obtained at each iteration and for each subject be returned by the function (`sample_path = TRUE`) or not

(sample_path = FALSE). Note that the sampled state sequence is quite a large object, hence the default setting is sample_path = FALSE. Can be used for local decoding purposes.

print_iter
: A logical scaler. Should the function print the progress of the algorithm by returning the current iteration number at every 10th iteration (print_iter = TRUE) or not (print_iter = FALSE). Defaults to print_iter = FALSE.

gamma_hyp_prior
: An optional list containing user specified parameters of the hyper-prior distribution on the multivariate normal distribution of the intercepts (and regression coefficients given that covariates are used) of the multinomial regression model of the transition probability matrix gamma. The hyper-prior of the mean intercepts is a multivariate Normal distribution, the hyper-prior of the covariance matrix between the set of (state specific) intercepts is an Inverse Wishart distribution.

Hence, the list gamma_hyp_prior contains the following elements:

- gamma_mu0: a list containing m matrices; one matrix for each row of the transition probability matrix gamma. Each matrix contains the hypothesized mean values of the intercepts. Hence, each matrix consists of one row (when not including covariates in the model) and m - 1 columns
- gamma_K0: numeric vector with length 1 denoting the number of hypothetical prior subjects on which the vector of means gamma_mu0 is based
- gamma_nu: numeric vector with length 1 denoting the degrees of freedom of the Inverse Wishart distribution
- gamma_V: matrix of m - 1 by m - 1 containing the hypothesized variance-covariance matrix between the set of intercepts.

Note that gamma_K0, gamma_nu and gamma_V are assumed equal over the states. The mean values of the intercepts (and regression coefficients of the covariates) denoted by gamma_mu0 are allowed to vary over the states.

The default values for the hyper-prior on gamma are: all elements of the matrices contained in gamma_mu0 set to 0, gamma_K0 set to 1, gamma_nu set to 3 + m - 1, and the diagonal of gamma_V (i.e., the variance) set to 3 + m - 1 and the off-diagonal elements (i.e., the covariance) set to 0.

See *Details* below if covariates are used for changes in the settings of the arguments of gamma_hyp_prior.

emiss_hyp_prior
: An optional list containing user specified parameters of the hyper-prior distribution on the multivariate normal distribution of the intercepts (and regression coefficients given that covariates are used) of the multinomial regression model of the emission distribution. The hyper-prior for the mean intercepts is a multivariate Normal distribution, the hyper-prior for the covariance matrix between the set of (state specific) intercepts is an Inverse Wishart distribution.

Hence, the list emiss_hyp_prior contains the following elements:

- emiss_mu0: a list of lists: emiss_mu0 contains ndep lists, i.e., one list for each dependent variable k. Each of these lists contains m matrices; one matrix for each set of emission probabilities within a state. The matrices contain the hypothesized mean values of the intercepts. Hence, each matrix consists of one row (when not including covariates in the model) and q_emiss[k] - 1 columns

- `emiss_K0`: a list containing `ndep` elements corresponding to each of the dependent variables, where each element is a numeric vector with length 1 denoting the number of hypothetical prior subjects on which the vector of means `emiss_mu0` is based
- `emiss_nu`: a list containing `ndep` elements corresponding to each of the dependent variables, where each element is a numeric vector with length 1 denoting the degrees of freedom of the Inverse Wishart distribution
- `emiss_V`: a list containing `ndep` elements corresponding to each of the dependent variables `k`, where each element is a matrix of `q_emiss[k] - 1` by `q_emiss[k]` - 1 containing the hypothesized variance-covariance matrix between the set of intercepts.

Note that `emiss_K0`, `emiss_nu` and `emiss_V` are assumed equal over the states. The mean values of the intercepts (and regression coefficients of the covariates) denoted by `emiss_mu0` are allowed to vary over the states.

The default values for the hyper-prior on the emission distribution(s) are: all elements of the matrices contained in `emiss_K0` set to 0, `emiss_K0` set to 1, `emiss_nu` set to `3 + q_emiss[k] - 1`, and the diagonal of `gamma_V` (i.e., the variance) set to `3 + q_emiss[k] - 1` and the off-diagonal elements (i.e., the covariance) set to 0.

See *Details* below if covariates are used for changes in the settings of the arguments of `emiss_hyp_prior`.

`gamma_sampler` An optional list containing user specified settings for the proposal distribution of the random walk (RW) Metropolis sampler for the subject level parameter estimates of the intercepts modeling the transition probability matrix. The list `gamma_sampler` contains the following elements:

- `gamma_int_mle0`: a numeric vector with length `m` - 1 denoting the start values for the maximum likelihood estimates of the intercepts for the transition probability matrix gamma, based on the pooled data (data over all subjects)
- `gamma_scalar`: a numeric vector with length 1 denoting the scale factor `s`. That is, The scale of the proposal distribution is composed of a covariance matrix Sigma, which is then tuned by multiplying it by a scaling factor `s^2`
- `gamma_w`: a numeric vector with length 1 denoting the weight for the overall log likelihood (i.e., log likelihood based on the pooled data over all subjects) in the fractional likelihood.

Default settings are: all elements in `gamma_int_mle0` set to 0, `gamma_scalar` set to 2.93 / sqrt(`m` - 1), and `gamma_w` set to 0.1. See the section *Scaling the proposal distribution of the RW Metropolis sampler* in `vignette("estimation-mhmm")` for details.

`emiss_sampler` An optional list containing user specified settings for the proposal distribution of the random walk (RW) Metropolis sampler for the subject level parameter estimates of the intercepts modeling the emission distributions of the dependent variables `k`. The list `emiss_sampler` contains the following elements:

- `emiss_int_mle0`: a list containing `ndep` elements corresponding to each of the dependent variables `k`, where each element is a a numeric vector with length `q_emiss[k]` - 1 denoting the start values for the maximum likelihood estimates of the intercepts for the emission distribution, based on the pooled data (data over all subjects)

- `emiss_scalar`: a list containing `ndep` elements corresponding to each of the dependent variables, where each element is a numeric vector with length 1 denoting the scale factor `s`. That is, The scale of the proposal distribution is composed of a covariance matrix Sigma, which is then tuned by multiplying it by a scaling factor `s^2`
- `emiss_w`: a list containing `ndep` elements corresponding to each of the dependent variables, where each element is a numeric vector with length 1 denoting the weight for the overall log likelihood (i.e., log likelihood based on the pooled data over all subjects) in the fractional likelihood.

Default settings are: all elements in `emiss_int_mle0` set to 0, `emiss_scalar` set to 2.93 / sqrt(`q_emiss[k]` - 1), and `emiss_w` set to 0.1. See the section *Scaling the proposal distribution of the RW Metropolis sampler* in `vignette("estimation-mhmm")` for details.

Details

Covariates specified in `xx` can either be dichotomous or continuous variables. Dichotomous variables have to be coded as 0/1 variables. Categorical or factor variables can as yet not be used as predictor covariates. The user can however break up the categorical variable in multiple dummy variables (i.e., dichotomous variables), which can be used simultaneously in the analysis. Continuous predictors are automatically centered. That is, the mean value of the covariate is subtracted from all values of the covariate such that the new mean equals zero. This is done such that the presented probabilities in the output (i.e., for the population transition probability matrix and population emission probabilities) correspond to the predicted probabilities at the average value of the covariate(s).

If covariates are specified and the user wants to set the values for the parameters of the hyper-prior distributions manually, the specification of the elements in the arguments of the hyper-prior parameter values of the normal distribution on the means change as follows: the number of rows in the matrices `gamma_mu0` and `emiss_mu0` are equal to 1 + the number of covariates used to predict the transition probability matrix for `gamma_mu0` and the emission distribution for `emiss_mu0` (i.e., the first row correspond to the hyper-prior mean values of the intercepts, the subsequent rows correspond to the hyper-prior mean values of the regression coefficients connected to each of the covariates), and `gamma_K0` and `emiss_K0` are now a matrix with the number of hypothetical prior subjects on which the vectors of means (i.e., the rows in `gamma_mu0` or `emiss_mu0`) are based on the diagonal, and off-diagonal elements equal to 0. Note that the hyper-prior parameter values of the inverse Wishart distribution on the covariance matrix remains unchanged, as the estimates of the regression coefficients for the covariates are fixed over subjects.

Value

`mHMM` returns an object of class `mHMM`, which has `print` and `summary` methods to see the results. The object contains the following components:

`PD_subj` A list containing one matrix per subject with the subject level parameter estimates and the log likelihood over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows, and the columns contain the subject level (parameter) estimates of subsequently the emission probabilities, the transition probabilities and the log likelihood.

`gamma_prob_bar` A matrix containing the group level parameter estimates of the transition probabilities over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows, and the columns contain the group level parameter estimates. If covariates were included in the analysis, the group level probabilities represent the predicted probability given that the covariate is at the average value for continuous covariates, or given that the covariate equals zero for dichotomous covariates.

`gamma_int_bar` A matrix containing the group level intercepts of the multinomial logistic regression modeling the transition probabilities over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows, and the columns contain the group level intercepts.

`gamma_cov_bar` A matrix containing the group level regression coefficients of the multinomial logistic regression predicting the transition probabilities over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows, and the columns contain the group level regression coefficients.

`gamma_int_subj` A list containing one matrix per subject denoting the subject level intercepts of the multinomial logistic regression modeling the transition probabilities over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows, and the columns contain the subject level intercepts.

`gamma_naccept` A matrix containing the number of accepted draws at the subject level RW Metropolis step for each set of parameters of the transition probabilities. The subjects are contained in the rows, and the columns contain the sets of parameters.

`emiss_prob_bar` A list containing one matrix per dependent variable, denoting the group level emission probabilities of each dependent variable over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group level emission probabilities. If covariates were included in the analysis, the group level probabilities represent the predicted probability given that the covariate is at the average value for continuous covariates, or given that the covariate equals zero for dichotomous covariates.

`emiss_int_bar` A list containing one matrix per dependent variable, denoting the group level intercepts of each dependent variable of the multinomial logistic regression modeling the probabilities of the emission distribution over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group level intercepts.

`emiss_cov_bar` A list containing one matrix per dependent variable, denoting the group level regression coefficients of the multinomial logistic regression predicting the emission probabilities within each of the dependent variables over the iterations of the hybrid Metropolis within Gibbs sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group level regression coefficients.

`emiss_int_subj` A list containing one list per subject denoting the subject level intercepts of each dependent variable of the multinomial logistic regression modeling the probabilities of the emission distribution over the iterations of the hybrid Metropolis within Gibbs sampler. Each lower level list contains one matrix per dependent variable, in which iterations of the sampler are contained in the rows, and the columns contain the subject level intercepts.

`emiss_naccept` A list containing one matrix per dependent variable with the number of accepted draws at the subject level RW Metropolis step for each set of parameters of the emission distribution. The subjects are contained in the rows, and the columns of the matrix contain the sets of parameters.

`input` Overview of used input specifications: the number of states `m`, the number of used dependent variables `n_dep`, the number of output categories for each of the dependent variables `q_emiss`, the number of iterations `J` and the specified burn in period `burn_in` of the hybrid Metropolis within Gibbs sampler, the number of subjects `n_subj`, the observation length for each subject `n_vary`, and the column names of the dependent variables `dep_labels`.

`sample_path` A list containing one matrix per subject with the sampled hidden state sequence over the hybrid Metropolis within Gibbs sampler. The time points of the dataset are contained in the rows, and the sampled paths over the iterations are contained in the columns. Only returned if `return_path = TRUE`.

References

Rabiner LR (1989). "A tutorial on hidden Markov models and selected applications in speech recognition." *Proceedings of the IEEE*, **77**(2), 257–286.

Scott SL (2002). "Bayesian methods for hidden Markov models: Recursive computing in the 21st century." *Journal of the American Statistical Association*, **97**(457), 337–351.

Altman RM (2007). "Mixed hidden Markov models: an extension of the hidden Markov model to the longitudinal data setting." *Journal of the American Statistical Association*, **102**(477), 201–210.

Rossi PE, Allenby GM, McCulloch R (2012). *Bayesian statistics and marketing*. John Wiley \& Sons.

Zucchini W, MacDonald IL, Langrock R (2017). *Hidden Markov models for time series: an introduction using R*. Chapman and Hall/CRC.

See Also

`sim_mHMM` for simulating multilevel hidden Markov data, `vit_mHMM` for obtaining the most likely hidden state sequence for each subject using the Viterbi algorithm, `obtain_gamma` and `obtain_emiss` for obtaining the transition or emission distribution probabilities of a fitted model at the group or subject level, and `plot.mHMM` for plotting the posterior densities of a fitted model.

Examples

```
###### Example on package example data

# specifying general model properties:
m <- 2
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
start_TM <- diag(.8, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .2
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[2]), # looking patient
                 matrix(c(0.90, 0.05, 0.05,
                             0.05, 0.90, 0.05), byrow = TRUE,
                           nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                    matrix(c(0.1, 0.9,
                             0.1, 0.9), byrow = TRUE, nrow = m,
                           ncol = q_emiss[4])) # looking therapist

# Run a model without covariate(s):
# Note that for reasons of running time, J is set at a ridiculous low value.
# One would typically use a number of iterations J of at least 1000,
# and a burn_in of 200.
out_2st <- mHMM(s_data = nonverbal,
                gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                start_val = c(list(start_TM), start_EM),
                mcmc = list(J = 11, burn_in = 5))

out_2st
summary(out_2st)

# plot the posterior densities for the transition and emission probabilities
plot(out_2st, component = "gamma", col =c("darkslategray3", "goldenrod"))

# Run a model including a covariate. Here, the covariate (standardized CDI
# change) predicts the emission distribution for each of the 4 dependent
# variables:

n_subj <- 10
xx <- rep(list(matrix(1, ncol = 1, nrow = n_subj)), (n_dep + 1))
for(i in 2:(n_dep + 1)){
  xx[[i]] <- cbind(xx[[i]], nonverbal_cov$std_CDI_change)
}
out_2st_c <- mHMM(s_data = nonverbal, xx = xx,
                  gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                  start_val = c(list(start_TM), start_EM),
                  mcmc = list(J = 11, burn_in = 5))


###### Example on simulated data
# Simulate data for 10 subjects with each 100 observations:
n_t <- 100
n <- 10
m <- 2
q_emiss <- 3
gamma <- matrix(c(0.8, 0.2,
                  0.3, 0.7), ncol = m, byrow = TRUE)
emiss_distr <- matrix(c(0.5, 0.5, 0.0,
                        0.1, 0.1, 0.8), nrow = m, ncol = q_emiss, byrow = TRUE)
data1 <- sim_mHMM(n_t = n_t, n = n, m = m, q_emiss = q_emiss, gamma = gamma,
                  emiss_distr = emiss_distr, var_gamma = .5, var_emiss = .5)

# Specify remaining required analysis input (for the example, we use simulation
# input as starting values):
n_dep <- 1
q_emiss <- 3
```

```
# Run the model on the simulated data:
out_2st_sim <- mHMM(s_data = data1$obs,
                    gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                    start_val = list(gamma, emiss_distr),
                    mcmc = list(J = 11, burn_in = 5))
```

mHMMbayes — *mHMMbayes: multilevel hidden Markov models using Bayesian estimation.*

Description

With the `R` package `mHMMbayes` you can fit multilevel hidden Markov models. The multilevel hidden Markov model (HMM) is a generalization of the well-known hidden Markov model, tailored to accommodate (intense) longitudinal data of multiple individuals simultaneously. Using a multilevel framework, we allow for heterogeneity in the model parameters (transition probability matrix and conditional distribution), while estimating one overall HMM. The model has a great potential of application in many fields, such as the social sciences and medicine. The model can be fitted on multivariate data with a categorical distribution, and include individual level covariates (allowing for e.g., group comparisons on model parameters). Parameters are estimated using Bayesian estimation utilizing the forward-backward recursion within a hybrid Metropolis within Gibbs sampler.

Details

The `mHMMbayes` package provides three main functions: `mHMM` , `sim_mHMM` and `vit_mHMM`, described below. For a more elaborate guide to the package `mHMMbayes`, see the tutorial-mhmm vignette: `vignette("tutorial-mhmm",package = "mHMMbayes")` . For extensive information on the estimation of the parameters in the package, see the estimation-mhmm vignette: `vignette("estimation-mhmm",package = "mHMMbayes")`.

`mHMM`

The function `mHMM` fits a multilevel hidden Markov model to (intense longitudinal) data from multiple subjects using Bayesian estimation. By using a multilevel framework, one general 'population' HMM is estimated, while heterogeneity between subjects is accommodated. The function can handle covariates at the subject level varying observation lengths over subjects. Estimation is performed using a hybrid Metropolis within Gibbs sampler, and completes the forward backward algorithm for all subjects in a sequential manner.

`sim_mHMM`

The function `sim_mHMM` simulates data for multiple subjects, for which the data have categorical observations that follow a hidden Markov model (HMM) with an multilevel structure. The multilevel structure implies that each subject is allowed to have it's own set of parameters, and that the parameters at the subject level (level 1) are tied together by a population distribution at level 2 for each of the corresponding parameters. The shape of the population distribution for each of the parameters is a normal (i.e., Gaussian) distribution. In addition to (natural and/or unexplained) heterogeneity between subjects, the subjects parameters can also depend on a (set of) covariate(s).

`vit_mHMM`

The function `vit_mHMM` obtains the most likely hidden state sequence for each subject, given the data and the subject specific parameter estimates. The function does this by utilizing the Viterbi algorithm.

`nonverbal` *Nonverbal communication of patients and therapist*

Description

A dataset containing the nonverbal communication of 10 patient-therapist couples, recorded for 15 minutes at a frequency of 1 observation per second (= 900 observations per couple).

Usage

```
nonverbal
```

Format

A matrix with 10 * 900 rows and 5 variables:

id id variable of patient - therapist couple to distinguish which observation belongs to which couple

p_verbalizing verbalizing behavior of the patient, consisting of 1 = not verbalizing, 2 = verbalizing, 3 = back channeling

p_looking looking behavior of the patient, consisting of 1 = not looking at therapist, 2 = looking at therapist

t_verbalizing verbalizing behavior of the therapist, consisting of 1 = not verbalizing, 2 = verbalizing, 3 = back channeling

t_looking looking behavior of the therapist, consisting of 1 = not looking at patient, 2 = looking at patient

`nonverbal_cov` *Predictors of nonverbal communication*

Description

A dataset containing predictors of nonverbal communication of 10 patient-therapist couples.

Usage

```
nonverbal_cov
```

Format

A matrix with 10 rows and 3 variables:

diagnosis Diagnosis of the patient, consisting of 0 = depression, 1 = anxiety

std_CDI_change Change in measure for depression (CDI) before and after therapy, standardized scale

std_SCA_change Change in measure for anxiety (SCARED) before and after therapy, standardized scale

`obtain_emiss` *Obtain the emission distribution probabilities for a fitted multilevel HMM*

Description

`obtain_emiss` obtains the emission distribution probabilities (also known as conditional probabilities) for a fitted multilevel hidden Markov model, for either the group level, i.e., representing the average emission distribution probabilities over all subjects, or at the subject level, returning the emission distribution probabilities for each subject.

Usage

```
obtain_emiss(object, level = "group", burn_in = NULL)
```

Arguments

`object`	An object of class `mHMM`, generated by the function `mHMM`.
`level`	String specifying if the returned emission distribution probabilities should be at the group level (`level = "group"`), i.e., representing the average emission distribution probabilities over all subjects, or at the subject level (`level = "subject"`).
`burn_in`	An integer which specifies the number of iterations to discard when obtaining the model parameter summary statistics. When left unspecified (`burn_in = NULL`), the burn in period specified when creating the `mHMM` object with the function `mHMM` will be used.

Value

`obtain_emiss` returns the object `est_emiss`. Depending on the specification at the input variable `level`, `est_emiss` is either a list of matrices with the emission distribution probabilities at the group level (if `level = "group"`) for each dependent variable, or a list of lists, where for each dependent variable a list is returned with the number of elements equal to the number of subjects analyzed, if `level = 'subject'`). In the latter scenario, each matrix in the lower level list represents the subject specific emission distribution probabilities for a specific dependent variable.

See Also

`mHMM` for fitting the multilevel hidden Markov model, creating the object `mHMM`.

Examples

```
###### Example on package data

# specifying general model properties:
m <- 2
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
start_TM <- diag(.8, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .2
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[2]), # looking patient
                 matrix(c(0.90, 0.05, 0.05,
                          0.05, 0.90, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# Run a model without covariate(s):
out_2st <- mHMM(s_data = nonverbal,
                gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                start_val = c(list(start_TM), start_EM),
                mcmc = list(J = 11, burn_in = 5))

out_2st
summary(out_2st)

# obtaining the emission probabilities at the group and subject level
obtain_emiss(out_2st, level = "group")
obtain_emiss(out_2st, level = "subject")
```

`obtain_gamma` *Obtain the transition probabilities gamma for a fitted multilevel HMM*

Description

`obtain_gamma` obtains the transition probability matrix for a fitted multilevel hidden Markov model, for either the group level, i.e., representing the average transition probability matrix over all subjects, or at the subject level, returning the transition probability matrices for each subject.

Usage

```
obtain_gamma(object, level = "group", burn_in = NULL)
```

Arguments

`object`	An object of class `mHMM`, generated by the function `mHMM`.
`level`	String specifying if the returned transition probability matrix gamma should be at the group level (`level = "group"`), i.e., representing the average transition probability matrix over all subjects, or at the subject level (`level = "subject"`).
`burn_in`	An integer which specifies the number of iterations to discard when obtaining the model parameter summary statistics. When left unspecified (`burn_in = NULL`), the burn in period specified when creating the `mHMM` object with the function `mHMM` will be used.

Value

`obtain_gamma` returns the object `est_gamma` of the class `mHMM_gamma`. This object can be directly plotted using the function `plot.mHMM_gamma()`, or simply `plot()`. Depending on the specification at the input variable `level`, `est_gamma` is either a matrix with the transition probabilities at the group level (if `level = "group"`), or a list of matrices (with the number of elements equal to the number of subjects analyzed, if `level = 'subject'`), where each matrix in the list represents a subject specific transition probability matrix.

See Also

`mHMM` for fitting the multilevel hidden Markov model, creating the object `mHMM`, and `plot.mHMM_gamma` for plotting the obtained transition probabilities.

Examples

```
###### Example on package data

# specifying general model properties:
m <- 2
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
```

```
start_TM <- diag(.8, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .2
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[2]), # looking patient
                 matrix(c(0.90, 0.05, 0.05,
                          0.05, 0.90, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# Run a model without covariate(s):
out_2st <- mHMM(s_data = nonverbal,
                gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                start_val = c(list(start_TM), start_EM),
                mcmc = list(J = 11, burn_in = 5))

out_2st
summary(out_2st)

# obtaining the transition probabilities at the group and subject level
obtain_gamma(out_2st, level = "group")
obtain_gamma(out_2st, level = "subject")
```

plot.mHMM — *Plotting the posterior densities for a fitted multilevel HMM*

Description

plot.mHMM plots the posterior densities for a fitted multilevel hidden Markov model for the group and subject level parameters simultaneously. The plotted posterior densities are either for the transition probability matrix gamma, or for the emission distribution probabilities.

Usage

```
## S3 method for class 'mHMM'
plot(x, component = "gamma", dep = 1, col, cat_lab,
  dep_lab, lwd1 = 2, lwd2 = 1, lty1 = 1, lty2 = 3, legend_cex,
  burn_in, ...)
```

Arguments

`x`	Object of class `mHMM`, generated by the function `mHMM`.
`component`	String specifying if the displayed posterior densities should be for the transition probability matrix gamma (`component = "gamma"`), or for the emission distribution probabilities (`component = "emiss"`). In case of the latter and the model is based on multiple dependent variables, the user has to indicate for which dependent variable the posterior densities have to be plotted, see `dep`.
`dep`	Integer specifying for which dependent variable the posterior densities should be plotted. Only required if one wishes to plot the emission distribution probabilities and the model is based on multiple dependent variables.
`col`	Vector of colors for the posterior density lines. If one is plotting the posterior densities for gamma, the vector has length `m` (i.e., number of hidden states). If one is plotting the posterior densities for the emission probabilities, the vector has length `q_emiss[k]` (i.e., the number of outcome categories for the dependent variable `k`).
`cat_lab`	Optional vector of strings when plotting the posterior densities of the emission probabilities, denoting the labels of the categorical outcome values. Automatically generated when not provided.
`dep_lab`	Optional string when plotting the posterior densities of the emission probabilities with length 1, denoting the label for the dependent variable plotted. Automatically obtained from the input object `x` when not specified.
`lwd1`	Positive number indicating the line width of the posterior density at the group level.
`lwd2`	Positive number indicating the line width of the posterior density at the subject level.
`lty1`	Positive number indicating the line type of the posterior density at the group level.
`lty2`	Positive number indicating the line type of the posterior density at the subject level.
`legend_cex`	A numerical value giving the amount by which plotting text and symbols in the legend should be magnified relative to the default.
`burn_in`	An integer which specifies the number of iterations to discard when obtaining the model parameter summary statistics. When left unspecified, the burn in period specified at creating the `mHMM` object with the function `mHMM` will be used.
`...`	Arguments to be passed to methods (see `par`)

Value

`plot.mHMM` returns a plot of the posterior densities. Depending on whether (`component = "gamma"`) or (`component = "emiss"`), the plotted posterior densities are either for the transition probability matrix gamma or for the emission distribution probabilities, respectively.

See Also

`mHMM` for fitting the multilevel hidden Markov model, creating the object `mHMM`.

Examples

```
###### First run the function mHMM on the nonverbal data

# specifying general model properties:
m <- 2
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
start_TM <- diag(.8, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .2
start_EM <- list(matrix(c(0.05, 0.90, 0.05, 0.90, 0.05, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9, 0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[2]), # looking patient
                 matrix(c(0.90, 0.05, 0.05, 0.05, 0.90, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9, 0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# Run a model without covariate(s):
out_2st <- mHMM(s_data = nonverbal, gen = list(m = m, n_dep = n_dep,
                q_emiss = q_emiss), start_val = c(list(start_TM), start_EM),
                mcmc = list(J = 11, burn_in = 5))

## plot the posterior densities for gamma
plot(out_2st, component = "gamma")
```

`plot.mHMM_gamma` *Plotting the transition probabilities gamma for a fitted multilevel HMM*

Description

`plot.mHMM_gamma` plots the transition probability matrix for a fitted multilevel hidden Markov model, by means of an alluvial plot (also known as Sankey diagram or riverplot) using the R package `alluvial`. The plotted transition probability matrix either represents the probabilities at the group level, i.e., representing the average transition probability matrix over all subjects, or at the subject level. In case of the latter, the user has to specify for which subject the transition probability matrix should be plotted.

Usage

```
## S3 method for class 'mHMM_gamma'
plot(x, subj_nr = NULL, cex = 0.8, col, hide, ...)
```

Arguments

`x`	An object of class `mHMM_gamma`, generated by the function `obtain_gamma`.
`subj_nr`	An integer specifying for which specific subject the transition probability matrix should be plotted. Only required if the input object represents the subject specific transition probability matrices.
`cex`	An integer specifying scaling of fonts of category labels. When not specified, defaults to `cex = 0.8`.
`col`	An optional vector with length `m` * `m` (i.e., where `m` denotes the number of hidden states) specifying the used colors in the alluvial plot.
`hide`	An optional logical vector with length `m` * `m` (i.e., where `m` denotes the number of hidden states) specifying whether particular stripes should be plotted. When not specified, omits the lines representing a value of exactly zero.
`...`	Arguments to be passed to alluvial (see `alluvial`)

Value

`plot.mHMM_gamma` returns a plot of the transition probability matrix. Depending on whether the input object represents the transition probabilities at the group level or the subject specific transition probability matrices, the returned plot represents either the group transition probability matrix, or the transition probability matrix for a given subject, specified by `subject_nr`.

See Also

`mHMM` for fitting the multilevel hidden Markov model, creating the object `mHMM`, and `obtain_gamma` to obtain the transition probabilities gamma for a fitted multilevel HMM, creating the object `mHMM_gamma`.

Examples

```
#' ###### Example on package data
# specifying general model properties:
m <- 2
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
start_TM <- diag(.8, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .2
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[2]), # looking patient
                 matrix(c(0.90, 0.05, 0.05,
                          0.05, 0.90, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                          ncol = q_emiss[4])) # looking therapist

# Run a model without covariate(s):
out_2st <- mHMM(s_data = nonverbal,
                gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                start_val = c(list(start_TM), start_EM),
                mcmc = list(J = 11, burn_in = 5))

out_2st
summary(out_2st)

# obtaining the transition probabilities at the group and subject level
est_gamma_group <- obtain_gamma(out_2st, level = "group")

# plot the obtained transition probabilities
plot(est_gamma_group, col = rep(c("green", "blue"), each = m))
```

`sim_mHMM` *Simulate data using a multilevel hidden Markov model*

Description

sim_mHMM simulates data for multiple subjects, for which the data have categorical observations that follow a hidden Markov model (HMM) with an multilevel structure. The multilevel structure implies that each subject is allowed to have its own set of parameters, and that the parameters at the subject level (level 1) are tied together by a population distribution at level 2 for each of the corresponding parameters. The shape of the population distribution for each of the parameters is a normal (i.e., Gaussian) distribution. In addition to (natural and/or unexplained) heterogeneity between subjects, the subjects parameters can also depend on a (set of) covariate(s).

Usage

```
sim_mHMM(n_t, n, m, q_emiss, gamma, emiss_distr, beta = NULL,
  xx_vec = NULL, var_gamma = 1, var_emiss = 1,
  return_ind_par = FALSE)
```

Arguments

n_t	The length of the observed sequence to be simulated for each subject. To only simulate subject specific transition probability matrices gamma and emission distributions (and no data), set t to 0.
n	The number of subjects for which data is simulated.
m	The number of hidden states in the HMM used for simulating data.
q_emiss	The number of categories of the simulated observations.

`gamma`	A matrix with `m` rows and `m` columns containing the average population transition probability matrix used for simulating the data. That is, the probability to switch from hidden state *i* (row *i*) to hidden state *j* (column *j*).
`emiss_distr`	A matrix with `m` rows and `q_emiss` columns containing the average population emission distribution of the (categorical) observations given the hidden states. That is, the probability of observing category *k* (column *k*) in state *i* (row *i*).
`beta`	List of two matrices containing the regression parameters to predict `gamma` and/or `emiss_distr` in combination with `xx_vec` using multinomial logistic regression. The first matrix is used to predict the transition probability matrix `gamma`. The second matrix is used to predict the emission distribution `emiss_distr` of the dependent variable. In both matrices, one regression parameter is specified for each element in `gamma` and `emiss_distr`, with the following exception. The first element in each row of `gamma` and/or `emiss_distr` is used as reference category in the multinomial logistic regression. As such, no regression parameters can be specified for these parameters. Hence, the first matrix in the list `beta` to predict `gamma` consist of a matrix with the number of rows equal to `m` and the number of columns equal to `m - 1`. The second matrix in the list `beta` to predict `emiss_distr` consist of a matrix with the number of rows equal to `m` and the number of columns equal to `q_emiss - 1`. See *details* for more information. Note that if `beta` is specified, `xx_vec` has to be specified as well. If `beta` is omitted completely, `beta` defaults to NULL, resembling no prediction of `gamma` or `emiss_distr` using covariates. One of the two elements in the list can also be left empty (i.e., set to `NULL`) to signify that either the transition probability matrix or a specific emission distribution is not predicted by covariates.
`xx_vec`	List of two vectors containing the covariate(s) to predict `gamma` and/or `emiss_distr` using the regression parameters specified in `beta`. The covariate used to predict `gamma` and `emiss_distr` can either be the same covariate, two different covariates, or a covariate for one element and none for the other. At this point, it is only possible to use one covariate for both `gamma` and `emiss_distr`. The first vector of the list `xx_vec` is used to predict the transition matrix. The second vector of the list `xx_vec` is used to predict the emission distribution of the dependent variable. For both vectors, the number of observations should be equal to `n` the number of subjects to be simulated. If `xx_vec` is omitted completely, `xx_vec` defaults to NULL, resembling no covariates at all. One of the two elements in the list can also be left empty (i.e., set to `NULL`) to signify that either the transition probability matrix or the emission distribution is not predicted by covariates.
`var_gamma`	An integer denoting the variance between subjects in the transition probability matrix. Note that this value corresponds to the variance of the parameters of the multinomial distribution (i.e., the intercepts of the regression equation of the multinomial distribution used to sample the transition probability matrix), see details below. In addition, only one variance value can be specified for the complete transition probability matrix, hence the variance is assumed fixed across all components. The default equals 1, which corresponds to quite some variation between subjects. A less extreme value would be 0.5. If one wants to simulate data from exactly the same HMM for all subjects, var_gamma should be set to 0.

`var_emiss`	An integer denoting the variance between subjects in the emission distribution. Note that this value corresponds to the variance of the parameters of the multinomial distribution (i.e., the intercepts of the regression equation of the multinomial distribution used to sample the components of the emission distribution), see details below. In addition, only one variance value can be specified for the complete emission distribution, hence the variance is assumed fixed across all components. The default equals 1, which corresponds to quite some variation between subjects. A less extreme value would be 0.5. If one wants to simulate data from exactly the same HMM for all subjects, var_emiss should be set to 0.
`return_ind_par`	A logical scalar. Should the subject specific transition probability matrix `gamma` and emission probability matrix `emiss_distr` be returned by the function (`return_ind_par = TRUE`) or not (`return_ind_par = FALSE`). The default equals `return_ind_par = FALSE`.

Details

In simulating the data, having a multilevel structure means that the parameters for each subject are sampled from the population level distribution of the corresponding parameter. The user specifies the population distribution for each parameter: the average population transition probability matrix and its variance, and the average population emission distribution and its variance. For now, the variance is assumed fixed for all components of the transition probability matrix and for all components of the emission distribution, and the simulated data can only consist of one dependent variable. In addition, at this point only one dependent variable can be simulated. That is, the hidden Markov model is a univariate hidden Markov model.

Note: the subject specific) initial state distributions (i.e., the probability of each of the states at the first time point) needed to simulate the data are obtained from the stationary distributions of the subject specific transition probability matrices gamma.

`beta`: As the first element in each row of `gamma` is used as reference category in the multinomial logistic regression, the first matrix in the list `beta` used to predict transition probability matrix `gamma` has a number of rows equal to `m` and the number of columns equal to `m - 1`. The first element in the first row corresponds to the probability of switching from state one to state two. The second element in the first row corresponds to the probability of switching from state one to state three, and so on. The last element in the first row corresponds to the probability of switching from state one to the last state. The same principle holds for the second matrix in the list `beta` used to predict the emission distribution `emiss_distr`: the first element in the first row corresponds to the probability of observing category two in state one. The second element in the first row corresponds to the probability of observing category three is state one, and so on. The last element in the first row corresponds to the probability of observing the last category in state one.

Value

The following components are returned by the function `sim_mHMM`:

`states` A matrix containing the simulated hidden state sequences, with one row per hidden state per subject. The first column indicates subject id number. The second column contains the simulated hidden state sequence, consecutively for all subjects. Hence, the id number is repeated over the rows (with the number of repeats equal to the length of the simulated hidden state sequence `T` for each subject).

obs A matrix containing the simulated observed outputs, with one row per simulated observation per subject. The first column indicates subject id number. The second column contains the simulated observation sequence, consecutively for all subjects. Hence, the id number is repeated over rows (with the number of repeats equal to the length of the simulated observation sequence T for each subject).

gamma A list containing n elements with the simulated subject specific transition probability matrices gamma. Only returned if return_ind_par is set to TRUE.

emiss_distr A list containing n elements with the simulated subject specific emission probability matrices emiss_distr. Only returned if return_ind_par is set to TRUE.

See Also

mHMM for analyzing multilevel hidden Markov data.

Examples

```
# simulating data for 10 subjects with each 100 observations
n_t     <- 100
n       <- 10
m       <- 3
q_emiss <- 4
gamma   <- matrix(c(0.8, 0.1, 0.1,
                  0.2, 0.7, 0.1,
                  0.2, 0.2, 0.6), ncol = m, byrow = TRUE)
emiss_distr <- matrix(c(0.5, 0.5, 0.0, 0.0,
                        0.1, 0.1, 0.8, 0.0,
                        0.0, 0.0, 0.1, 0.9), nrow = m, ncol = q_emiss, byrow = TRUE)
data1 <- sim_mHMM(n_t = n_t, n = n, m = m, q_emiss = q_emiss, gamma = gamma,
                  emiss_distr = emiss_distr, var_gamma = 1, var_emiss = 1)
head(data1$obs)
head(data1$states)

# including a covariate to predict (only) the transition probability matrix gamma
beta      <- rep(list(NULL), 2)
beta[[1]] <- matrix(c(0.5, 1.0,
                     -0.5, 0.5,
                      0.0, 1.0), byrow = TRUE, ncol = 2)
xx_vec      <- rep(list(NULL),2)
xx_vec[[1]] <-  c(rep(0,5), rep(1,5))
data2 <- sim_mHMM(n_t = n_t, n = n, m = m, q_emiss = q_emiss, gamma = gamma,
                  emiss_distr = emiss_distr, beta = beta, xx_vec = xx_vec,
                  var_gamma = 1, var_emiss = 1)


# simulating subject specific transition probability matrices and emission distributions only
n_t <- 0
n <- 5
m <- 3
q_emiss <- 4
gamma <- matrix(c(0.8, 0.1, 0.1,
                  0.2, 0.7, 0.1,
                  0.2, 0.2, 0.6), ncol = m, byrow = TRUE)
emiss_distr <- matrix(c(0.5, 0.5, 0.0, 0.0,
                        0.1, 0.1, 0.8, 0.0,
                        0.0, 0.0, 0.1, 0.9), nrow = m, ncol = q_emiss, byrow = TRUE)
data3 <- sim_mHMM(n_t = n_t, n = n, m = m, q_emiss = q_emiss, gamma = gamma,
                  emiss_distr = emiss_distr, var_gamma = 1, var_emiss = 1)
data3

data4 <- sim_mHMM(n_t = n_t, n = n, m = m, q_emiss = q_emiss, gamma = gamma,
                  emiss_distr = emiss_distr, var_gamma = .5, var_emiss = .5)
data4
```

vit_mHMM — *Obtain hidden state sequence for each subject using the Viterbi algorithm*

Description

`vit_mHMM` obtains the most likely state sequence for each subject from an object of class `mHMM` (generated by the function `mHMM`), using (an extended version of) the Viterbi algorithm. This is also known as global decoding.

Usage

```
vit_mHMM(object, s_data, burn_in = NULL)
```

Arguments

`object`	An object of class `mHMM`, generated by the function `mHMM`.
`s_data`	A matrix containing the observations to be modelled, where the rows represent the observations over time. In `s_data`, the first column indicates subject id number. Hence, the id number is repeated over rows equal to the number of observations for that subject. The subsequent columns contain the dependent variable(s). Note that the dependent variables have to be numeric, i.e., they cannot be a (set of) factor variable(s). The total number of rows are equal to the sum over the number of observations of each subject, and the number of columns are equal to the number of dependent variables (`n_dep`) + 1. The number of observations can vary over subjects.
`burn_in`	The number of iterations to be discarded from the MCMC algorithm when inferring the transition probability matrix gamma and the emission distribution of (each of) the dependent variable(s) for each subject from `s_data`. If omitted, defaults to `NULL` and `burn_in` specified at the function `mHMM` will be used.

Details

Note that local decoding is also possible, by inferring the most frequent state at each point in time for each subject from the sampled state path at each iteration of the MCMC algorithm. This information is contained in the output object `return_path` of the function `mHMM`.

Value

The function `vit_mHMM` returns a matrix containing the most likely state at each point in time. Each column represents a subject, and each row represents a point in time. If sequence lengths differ over subjects, states for none existing time points for subjects are filled with `NA`.

References

Viterbi A (1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm." *IEEE transactions on Information Theory*, **13**(2), 260–269.

Rabiner LR (1989). "A tutorial on hidden Markov models and selected applications in speech recognition." *Proceedings of the IEEE*, **77**(2), 257–286.

See Also

`mHMM` for analyzing multilevel hidden Markov data and obtaining the input needed for `vit_mHMM`, and `sim_mHMM` for simulating multilevel hidden Markov data.

Examples

```
###### Example on package example data
###### First fit the multilevel HMM on the nonverbal data

# specifying general model properties:
m <- 2
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
start_TM <- diag(.8, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .2
start_EM <- list(matrix(c(0.05, 0.90, 0.05, 0.90, 0.05, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9, 0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[2]), # looking patient
                 matrix(c(0.90, 0.05, 0.05, 0.05, 0.90, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9, 0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# Fit the multilevel HMM model:
# Note that for reasons of running time, J is set at a ridiculous low value.
# One would typically use a number of iterations J of at least 1000,
# and a burn_in of 200.
out_2st <- mHMM(s_data = nonverbal, gen = list(m = m, n_dep = n_dep,
                q_emiss = q_emiss), start_val = c(list(start_TM), start_EM),
                mcmc = list(J = 3, burn_in = 1))

###### obtain the most likely state sequence with the Viterbi algorithm
states <- vit_mHMM(s_data = nonverbal, object = out_2st)

###### Example on simulated data
```

```
# Simulate data for 10 subjects with each 100 observations:
n_t <- 100
n <- 10
m <- 2
q_emiss <- 3
gamma <- matrix(c(0.8, 0.2,
                  0.3, 0.7), ncol = m, byrow = TRUE)
emiss_distr <- matrix(c(0.5, 0.5, 0.0,
                        0.1, 0.1, 0.8), nrow = m, ncol = q_emiss, byrow = TRUE)
data1 <- sim_mHMM(n_t = n_t, n = n, m = m, q_emiss = q_emiss, gamma = gamma,
                  emiss_distr = emiss_distr, var_gamma = .5, var_emiss = .5)

# Specify remaining required analysis input (for the example, we use simulation
# input as starting values):
n_dep <- 1
q_emiss <- 3

# Fit the model on the simulated data:
# Note that for reasons of running time, J is set at a ridiculous low value.
# One would typically use a number of iterations J of at least 1000,
# and a burn_in of 200.
out_2st_sim <- mHMM(s_data = data1$obs,
                 gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                 start_val = list(gamma, emiss_distr),
                 mcmc = list(J = 11, burn_in = 5))

###### obtain the most likely state sequence with the Viterbi algorithm
states <- vit_mHMM(s_data = data1$obs, object = out_2st_sim)
```

Index