Package 'mHMMbayes'

July 11, 2025

Type Package

Title Multilevel Hidden Markov Models Using Bayesian Estimation

Version 1.1.1

Depends R (>= 3.6.0)

Imports MCMCpack, mvtnorm, stats, Rdpack, Rcpp

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 either a categorical, normal, or Poisson 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. Missing data (NA) in the dependent variables is accommodated assuming MAR. 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://CRAN.R-project.org/package=mHMMbayes

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

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

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

VignetteBuilder knitr

RdMacros Rdpack

SystemRequirements GNU make

LinkingTo Rcpp

NeedsCompilation yes

Author Emmeke Aarts [aut, cre],
Sebastian Mildiner Moraga [aut]

Repository CRAN

Date/Publication 2025-07-11 14:10:02 UTC

Contents

`int_to_prob` *Transforming a set of Multinomial logit regression intercepts to probabilities*

Description

`int_to_prob` transforms a set of Multinomial logit regression intercepts to the corresponding state transition or categorical emission observation probabilities. Note that the first state or category is assumed to be the reference category, hence no intercept is to specified for the first state or category.

Usage

```
int_to_prob(int_matrix)
```

Arguments

int_matrix	A matrix with (number of states OR categories - 1) columns and number of rows to be determined by the user. For obtaining the set of probabilities of the complete transition probability matrix gamma or categorical emission distribution matrix, the number of rows equals the number of states m. The first state / category is assumed to be the reference category, no intercept is to be specified for this first category.

Details

Designed to ease the specification of informative hyper-prior values for the mean intercepts of the transition probability matrix gamma and categorical emission distribution(s) of the multilevel hidden Markov model through the functions prior_gamma and prior_emiss_cat. No check is performed on correct specifications of the dimensions.

Value

int_to_prob returns a matrix containing probabilities with each row summing to one, with the number of columns equal to the number of states / categories and the number of rows equal to the number rows specified in the input matrix.

See Also

prob_to_int for transforming a set of probabilities to a set of Multinomial logit regression intercepts, prior_gamma and prior_emiss_cat for specifying informative hyper-priors for the the multilevel hidden Markov model and mHMM to fit a multilevel hidden Markov model.

Examples

```
# example for transition probability matrix gamma with 3 states
m <- 3
gamma_int <- matrix(c(-1, -1,
                      3,  0,
                      0,  2), ncol = m-1, nrow = m, byrow = TRUE)
gamma_prob <- int_to_prob(gamma_int)
gamma_prob
```

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, continuous (i.e., normally distributed), or count (i.e., Poisson distributed) 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,
  data_distr = "categorical",
  gen,
  xx = NULL,
  start_val,
  mcmc,
  return_path = FALSE,
  show_progress = TRUE,
  gamma_hyp_prior = NULL,
  emiss_hyp_prior = NULL,
  gamma_sampler = NULL,
  emiss_sampler = NULL
)
```

Arguments

s_data	A matrix containing the observations to be modeled, 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 in case of categorical dependent variable(s), input are integers (i.e., whole numbers) that range from 1 to q_emiss (see below) and is numeric (i.e., not 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.
data_distr	String vector with length 1 describing the observation type of the data. Currently supported are 'categorical' , 'continuous', and 'count'. Note that

for multivariate data, all dependent variables are assumed to be of the same observation type. The default equals to `data_distr = 'categorical'`.

`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`: only to be specified if the data represents categorical data. 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 `n_dep` elements each contain a matrix with the start values for the emission distribution(s). In case of categorical observations: a `m` by `q_emiss[k]` matrix denoting the probability of observing each categorical outcome (columns) within each state (rows). In case of continuous observations: a `m` by 2 matrix denoting the mean (first column) and standard deviation (second column) of the Normal emission distribution within each state (rows). In case of count observations: a `m` by 1 matrix denoting the mean (column) of the Poisson emission distribution within each state on the natural (i.e., real positive domain, not logarithmic) scale (rows). 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.

show_progress A logical scaler. Should the function show a text progress bar in the R console to represent the progress of the algorithm (show_progress = TRUE) or not (show_progress = FALSE). Defaults to show_progress = TRUE.

gamma_hyp_prior An optional object of class mHMM_prior_gamma containing user specified parameter values for the hyper-prior distribution on the transition probability matrix gamma, generated by the function prior_gamma.

emiss_hyp_prior An object of the class mHMM_prior_emiss containing user specified parameter values for the hyper-prior distribution on the categorical or Normal emission distribution(s), generated by the function prior_emiss_cat, prior_emiss_cont or prior_emiss_count. Mandatory in case of continuous and count observations, optional in case of categorical observations.

gamma_sampler An optional object of the class mHMM_pdRW_gamma containing user specified settings for the proposal distribution of the random walk (RW) Metropolis sampler on the subject level transition probability matrix parameters, generated by the function pd_RW_gamma.

emiss_sampler An optional object of the class mHMM_pdRW_emiss containing user specified settings for the proposal distribution of the random walk (RW) Metropolis sampler on the subject level emission distribution(s) parameters, generated by either the function pd_RW_emiss_cat or pd_RW_emiss_count. Only applicable in case of categorical and count observations.

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).

Value

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

PD_subj A list containing one list per subject with the elements trans_prob, cat_emiss, cont_emiss, or count_emiss in case of categorical, continuous, and count observations, respectively, and log_likl, providing the subject parameter estimates over the iterations of the MCMC sampler. trans_prob relates to the transition probabilities gamma, cat_emiss to the categorical emission distribution (emission probabilities), cont_emiss to the continuous emission distributions (subsequently the the emission means and the (fixed over subjects) emission standard deviation), count_emiss to the count emission distribution means in the natural (real-positive) scale, and log_likl to the log likelihood over the MCMC iterations. Iterations are contained in the rows, the parameters in the columns.

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_V_int_bar A matrix containing the variance components for the subject-level intercepts (between subject variances) 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 variance components for the subject level intercepts. Note that only the intercept variances (and not the co-variances) are returned.

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.

input Overview of used input specifications: the distribution type of the observations `data_distr`, the number of states `m`, the number of used dependent variables `n_dep`, (in case of categorical observations) 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`.

Additionally, in case of categorical observations, the `mHMM` return object contains:

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_V_int_bar` A list containing one matrix per dependent variable, denoting the variance components for the subject-level intercepts (between subject variances) of the multinomial logistic regression modeling the categorical emission 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 variance components for the subject level intercepts. Note that only the intercept variances (and not the co-variances) are returned.

`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.

In case of continuous observations, the `mHMM` return object contains:

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

`emiss_varmu_bar` A list containing one matrix per dependent variable, denoting the variance between the subject level means of the Normal emission distributions. over the iterations of the Gibbs sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group level variance in the mean.

`emiss_sd_bar` A list containing one matrix per dependent variable, denoting the (fixed over subjects) standard deviation of the Normal emission distributions over the iterations of the Gibbs sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group level emission variances.

`emiss_cov_bar` A list containing one matrix per dependent variable, denoting the group level regression coefficients predicting the emission means within each of the dependent variables over the iterations of the 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_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 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`.

In case of count observations, the `mHMM` return object contains:

`emiss_mu_bar` A list containing one matrix per dependent variable, denoting the group-level means of the Poisson emission distribution of each dependent variable over the MCMC iterations. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group-level Poisson emission means in the positive scale. If covariates were included in the analysis, the group-level means represent the predicted mean counts given that the covariate is at the average value for continuous covariates, or given that the covariate equals zero for dichotomous covariates.

`emiss_varmu_bar` A list containing one matrix per dependent variable, denoting the variance between the subject-level means of the Poisson emission distribution(s) over the iterations of the MCMC sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group-level variance(s) between subject-specific means (in the positive scale).

`emiss_logmu_bar` A list containing one matrix per dependent variable, denoting the group-level logmeans of the Poisson emission distribution of each dependent variable over the MCMC iterations. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group-level Poisson emission logmeans in the logarithmic scale. If covariates were included in the analysis, the group-level means represent the predicted logmean counts given that the covariate is at the average value for continuous covariates, or given that the covariate equals zero for dichotomous covariates.

`emiss_logvarmu_bar` A list containing one matrix per dependent variable, denoting the logvariance between the subject-level logmeans of the Poisson emission distribution(s) over the iterations of the MCMC sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group-level logvariance(s) between subject-specific means (in the logarithmic scale).

`emiss_cov_bar` A list containing one matrix per dependent variable, denoting the group level regression coefficients predicting the emission means in the logarithmic scale within each of the dependent variables over the iterations of the MCMC sampler. The iterations of the sampler are contained in the rows of the matrix, and the columns contain the group-level regression coefficients in the logarithmic scale.

`input` Overview of used input specifications: the number of states `m`, the number of used dependent variables `n_dep`, 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`.

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 (categorical) example data, see ?nonverbal

# 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 (see ?nonverbal_cov) to predict the
# emission distribution for each of the 4 dependent variables:

n_subj <- 10
xx_emiss <- rep(list(matrix(c(rep(1, n_subj),nonverbal_cov$std_CDI_change),
                            ncol = 2, nrow = n_subj)), n_dep)
xx <- c(list(matrix(1, ncol = 1, nrow = n_subj)), xx_emiss)
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 categorical simulated data
# Simulate data for 10 subjects with each 100 observations:
n_t <- 100
n <- 10
m <- 2
n_dep <- 1
q_emiss <- 3
gamma <- matrix(c(0.8, 0.2,
                  0.3, 0.7), ncol = m, byrow = TRUE)
emiss_distr <- list(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, gen = list(m = m, n_dep = n_dep, 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 = c(list(gamma), emiss_distr),
                  mcmc = list(J = 11, burn_in = 5))

###### Example on continuous simulated data
 # simulating multivariate continuous data
n_t     <- 100
n       <- 10
m       <- 3
n_dep   <- 2

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 <- list(matrix(c( 50, 10,
                              100, 10,
                              150, 10), nrow = m, byrow = TRUE),
                    matrix(c(5, 2,
                             10, 5,
                             20, 3), nrow = m, byrow = TRUE))

data_cont <- sim_mHMM(n_t = n_t, n = n, data_distr = 'continuous',
                      gen = list(m = m, n_dep = n_dep),
                      gamma = gamma, emiss_distr = emiss_distr,
                      var_gamma = .1, var_emiss = c(5^2, 0.2^2))

# Specify hyper-prior for the continuous emission distribution
manual_prior_emiss <- prior_emiss_cont(
                        gen = list(m = m, n_dep = n_dep),
                        emiss_mu0 = list(matrix(c(30, 70, 170), nrow = 1),
                                         matrix(c(7, 8, 18), nrow = 1)),
                        emiss_K0 = list(1, 1),
                        emiss_V =  list(rep(5^2, m), rep(0.5^2, m)),
                        emiss_nu = list(1, 1),
                        emiss_a0 = list(rep(1.5, m), rep(1, m)),
                        emiss_b0 = list(rep(20, m), rep(4, m)))

# Run 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_3st_cont_sim <- mHMM(s_data = data_cont$obs,
                         data_distr = 'continuous',
                         gen = list(m = m, n_dep = n_dep),
                         start_val = c(list(gamma), emiss_distr),
                         emiss_hyp_prior = manual_prior_emiss,
                         mcmc = list(J = 11, burn_in = 5))




###### Example on multivariate count data
# Simulate data with one covariate for each count dependent variable

n_t     <- 200     # Number of observations on the dependent variable
m       <- 3        # Number of hidden states
n_dep   <- 2        # Number of dependent variables
n_subj  <- 30        # Number of subjects

gamma   <- matrix(c(0.9, 0.05, 0.05,
                    0.2, 0.7, 0.1,
                    0.2,0.3, 0.5), ncol = m, byrow = TRUE)

emiss_distr <- list(matrix(c(20,
                             10,
                             5), nrow = m, byrow = TRUE),
```

```
                    matrix(c(50,
                             3,
                             20), nrow = m, byrow = TRUE))

# Define between subject variance to use on the simulating function:
# here, the variance is varied over states within the dependent variable.
var_emiss <- list(matrix(c(5.0, 3.0, 1.5), nrow = m),
                  matrix(c(5.0, 5.0, 5.0), nrow = m))

# Simulate count data:
data_count <- sim_mHMM(n_t = n_t,
                       n = n_subj,
                       data_distr = "count",
                       gen = list(m = m, n_dep = n_dep),
                       gamma = gamma,
                       emiss_distr = emiss_distr,
                       var_gamma = 0.1,
                       var_emiss = var_emiss,
                       return_ind_par = TRUE)


# Transition probabilities
start_gamma <- diag(0.8, m)
start_gamma[lower.tri(start_gamma) | upper.tri(start_gamma)] <-
                                            (1 - diag(start_gamma)) / (m - 1)

# Emission distribution
start_emiss <- list(matrix(c(20,10, 5), nrow = m, byrow = TRUE),
                    matrix(c(50, 3,20), nrow = m, byrow = TRUE))

# Specify hyper-prior for the count emission distribution
manual_prior_emiss <- prior_emiss_count(
  gen = list(m = m, n_dep = n_dep),
  emiss_mu0 = list(matrix(c(20, 10, 5), byrow = TRUE, ncol = m),
                   matrix(c(50, 3, 20), byrow = TRUE, ncol = m)),
  emiss_K0  = rep(list(0.1),n_dep),
  emiss_nu  = rep(list(0.1),n_dep),
  emiss_V   = rep(list(rep(10, m)),n_dep)
)

# Run 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_3st_count_sim <- mHMM(s_data = data_count$obs,
                          data_distr = 'count',
                          gen = list(m = m, n_dep = n_dep),
                          start_val = c(list(start_gamma), start_emiss),
                          emiss_hyp_prior = manual_prior_emiss,
                          mcmc = list(J = 11, burn_in = 5),
                          show_progress = TRUE)

summary(out_3st_count_sim)
```

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 for an object containing a fitted multilevel hidden Markov model, either at the group level, i.e., representing the average emission distribution over all subjects, or at the subject level, returning the emission distribution 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 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 will be used.

Value

`obtain_emiss` creates an object of the class `mHMM_emiss`. Depending on the specification at the input variable `level`, the output is either a list of matrices with the emission distribution 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 for a specific dependent variable.

See Also

`mHMM` for fitting the multilevel hidden Markov model.

Examples

```
###### Example on package data, see ?nonverbal

# 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 (TPM) for an object containing a fitted multilevel hidden Markov model. The TPM can be obtained either at 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 will be used.

Value

obtain_gamma creates an object 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, the output 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, 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")
```

`pd_RW_emiss_cat` *Proposal distribution settings RW Metropolis sampler for mHMM categorical emission distribution(s)*

Description

`pd_RW_emiss_cat` provides a framework to manually specify the settings of the proposal distribution of the random walk (RW) Metropolis sampler of categorical emission distribution(s) of the multilevel hidden Markov model, and creates on object of the class `mHMM_pdRW_emiss`. The RW metropolis sampler is used for sampling the subject level parameter estimates relating to the emission distributions of the dependent variables `k`, that is, the Multinomial logistic regression intercepts.

Usage

```
pd_RW_emiss_cat(gen, emiss_int_mle0, emiss_scalar, emiss_w)
```

Arguments

`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`: only to be specified if the data represents categorical data. Numeric vector with length `n_dep` denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.
`emiss_int_mle0`	A list containing `n_dep` elements corresponding to each of the dependent variables `k`, where each element is a matrix with `m` rows and `q_emiss[k]` - 1 columns denoting the starting values for the maximum likelihood (ML) estimates of the Multinomial logit regression intercepts of the emission distribution(s). ML parameters to be estimated are based on the pooled data (data over all subjects).

`emiss_scalar` A list containing `n_dep` 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 `n_dep` 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.

Details

When no manual values for the settings of the proposal distribution of the random walk (RW) Metropolis sampler are specified at all (that is, the function `pd_RW_emiss_cat` is not used), 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.

Within the function `mHMM`, the acceptance rate of the RW metropolis sampler relating to the emission distribution(s) can be tracked using the output parameter `emiss_naccept`. An acceptance rate of about 23% is considered optimal when many parameters are being updated at once (Gelman, Carlin, Stern & Rubin, 2014).

Value

`pd_RW_emiss_cat` returns an object of class `mHMM_pdRW_emiss`, containing settings of the proposal distribution of the random walk (RW) Metropolis sampler on the categorical emission distribution(s) of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function `mHMM`, and checked for correct input dimensions. The object contains the following components:

`gen` A list containing the elements `m`, `n_dep`, and `q_emiss`, used for checking equivalent general model properties specified under `pd_RW_emiss_cat` and `mHMM`.

`emiss_int_mle0` A list containing `n_dep` elements, where each element is a matrix containing the starting values for the maximum likelihood (ML) estimates of the Multinomial logit regression intercepts of the emission distribution(s).

`emiss_scalar` A list containing `n_dep` elements denoting the scale factor `s` of the proposal distribution.

`emiss_w` A list containing `n_dep` elements denoting denoting the weight for the overall log likelihood in the fractional likelihood.

References

Gelman A, Carlin JB, Stern HS, Rubin DB (2014). *Bayesian Data Analysis vol. 2*. Taylor & Francis.

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

Examples

```
###### Example using package example data, see ?nonverbal
# specifying general model properties:
m <- 3
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying manual values for RW metropolis sampler on emission distributions
emiss_int_mle0 <- list(matrix(c( 2,  0,
                                -2, -2,
                                 0, -1), byrow = TRUE, nrow = m, ncol = q_emiss[1] - 1),
                       matrix(c( 2,
                                 2,
                                 2), byrow = TRUE, nrow = m, ncol = q_emiss[2] - 1),
                       matrix(c(-2, -2,
                                 2,  0,
                                 0, -1), byrow = TRUE, nrow = m, ncol = q_emiss[3] - 1),
                       matrix(c( 2,
                                 2,
                                 2), byrow = TRUE, nrow = m, ncol = q_emiss[4] - 1))
emiss_scalar <- list(c(2), c(3), c(2), c(3))
emiss_w <- rep(list(c(0.2)), n_dep)
manual_emiss_sampler <- pd_RW_emiss_cat(gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                                        emiss_int_mle0 = emiss_int_mle0,
                                        emiss_scalar = emiss_scalar,
                                        emiss_w = emiss_w)

# specifying starting values
start_TM <- diag(.7, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .1
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05,
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          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,
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9,
                          0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# 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_3st_RWemiss <- mHMM(s_data = nonverbal,
```

```
                        gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                        start_val = c(list(start_TM), start_EM),
                        emiss_sampler = manual_emiss_sampler,
                        mcmc = list(J = 11, burn_in = 5))

out_3st_RWemiss
summary(out_3st_RWemiss)

# checking acceptance rate (for illustrative purposes, in the example,
# J is too low for getting a fair indication)
div_J <- function(x, J) x / J
J_it <- 11 - 1 # accept/reject starts at iteration 2 of MCMC algorithm
RW_emiss_accept <- sapply(out_3st_RWemiss$emiss_naccept, div_J, J_it, simplify = FALSE)

# average acceptance rate over all subjects per parameter
# rows represent each of the n_dep dependent variables, columns represent the m states
t(sapply(RW_emiss_accept, apply, MARGIN = 2, mean))
```

---

pd_RW_emiss_count *Proposal distribution settings RW Metropolis sampler for mHMM Poisson-lognormal emission distribution(s)*

Description

pd_RW_emiss_count provides a framework to manually specify the settings of the proposal distribution of the random walk (RW) Metropolis sampler of Poisson emission distribution(s) of the multilevel hidden Markov model, and creates on object of the class mHMM_pdRW_emiss. The RW metropolis sampler is used for sampling the subject level parameter estimates relating to the emission distributions of the dependent variables k, that is, the Poisson parameters lambda.

Usage

```
pd_RW_emiss_count(gen, emiss_scalar, emiss_w)
```

Arguments

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: only to be specified if the data represents categorical data. Numeric vector with length n_dep denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.

`emiss_scalar` A list containing `n_dep` 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 `n_dep` 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.

Details

When no manual values for the settings of the proposal distribution of the random walk (RW) Metropolis sampler are specified at all (that is, the function `pd_RW_emiss_count` is not used), `emiss_scalar` set to 2.38, 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.

Within the function `mHMM`, the acceptance rate of the RW metropolis sampler relating to the emission distribution(s) can be tracked using the output parameter `emiss_naccept`. An acceptance rate of about 45% is considered optimal when a single parameter is being updated (Gelman, Carlin, Stern & Rubin, 2014).

Value

`pd_RW_emiss_count` returns an object of class `mHMM_pdRW_emiss`, containing settings of the proposal distribution of the random walk (RW) Metropolis sampler on the categorical emission distribution(s) of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function `mHMM`, and checked for correct input dimensions. The object contains the following components:

`gen` A list containing the elements `m` and `n_dep`, used for checking equivalent general model properties specified under `pd_RW_emiss_count` and `mHMM`.

`emiss_scalar` A list containing `n_dep` elements denoting the scale factor `s` of the proposal distribution.

`emiss_w` A list containing `n_dep` elements denoting denoting the weight for the overall log likelihood in the fractional likelihood.

References

Gelman A, Carlin JB, Stern HS, Rubin DB (2014). *Bayesian Data Analysis vol. 2*. Taylor & Francis.

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

Examples

```
###### Example using simulated data
# specifying general model properties:
n_t     <- 200     # Number of observations on the dependent variable
m       <- 3        # Number of hidden states
```

```
n_dep   <- 2        # Number of dependent variables
n_subj  <- 30        # Number of subjects

gamma   <- matrix(c(0.9, 0.05, 0.05,
                    0.2, 0.7, 0.1,
                    0.2,0.3, 0.5), ncol = m, byrow = TRUE)

emiss_distr <- list(matrix(c(20,
                             10,
                             5), nrow = m, byrow = TRUE),
                    matrix(c(50,
                             3,
                             20), nrow = m, byrow = TRUE))

# Define between subject variance to use on the simulating function:
# here, the variance is varied over states within the dependent variable.
var_emiss <- list(matrix(c(5.0, 3.0, 1.5), nrow = m),
                  matrix(c(5.0, 5.0, 5.0), nrow = m))

# Simulate count data:
data_count <- sim_mHMM(n_t = n_t,
                       n = n_subj,
                       data_distr = "count",
                       gen = list(m = m, n_dep = n_dep),
                       gamma = gamma,
                       emiss_distr = emiss_distr,
                       var_gamma = 0.1,
                       var_emiss = var_emiss,
                       return_ind_par = TRUE)


# Transition probabilities
start_gamma <- diag(0.8, m)
start_gamma[lower.tri(start_gamma) | upper.tri(start_gamma)] <- (1 - diag(start_gamma)) / (m - 1)

# Emission distribution
start_emiss <- list(matrix(c(20,10, 5), nrow = m, byrow = TRUE),
                    matrix(c(50, 3,20), nrow = m, byrow = TRUE))

# Specify hyper-prior for the count emission distribution
manual_prior_emiss <- prior_emiss_count(
  gen = list(m = m, n_dep = n_dep),
  emiss_mu0 = list(matrix(c(20, 10, 5), byrow = TRUE, ncol = m),
                   matrix(c(50, 3, 20), byrow = TRUE, ncol = m)),
  emiss_K0  = rep(list(0.1),n_dep),
  emiss_nu  = rep(list(0.1),n_dep),
  emiss_V   = rep(list(rep(10, m)),n_dep)
)

# Specify the desired values for the sampler
manual_emiss_sampler <- pd_RW_emiss_count(gen = list(m = m, n_dep = n_dep),
                                          emiss_scalar = rep(list(2.38),n_dep),
                                          emiss_w = rep(list(0.1), n_dep))
```

```
# Run 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_3st_count_RWemiss <- mHMM(s_data = data_count$obs,
                          data_distr = 'count',
                          gen = list(m = m, n_dep = n_dep),
                          start_val = c(list(start_gamma), start_emiss),
                          emiss_hyp_prior = manual_prior_emiss,
                          emiss_sampler = manual_emiss_sampler,
                          mcmc = list(J = 11, burn_in = 5),
                          show_progress = TRUE)

# Examine acceptance rates over dependent variable, individual, and states:
lapply(out_3st_count_RWemiss$emiss_naccept, function(e) e/out_3st_count_RWemiss$input$J)

# Finally, take the average acceptance rate by dependent variable and state:
lapply(out_3st_count_RWemiss$emiss_naccept, function(e) colMeans(e/out_3st_count_RWemiss$input$J))
```

---

pd_RW_gamma *Proposal distribution settings RW Metropolis sampler for mHMM transition probability matrix gamma*

Description

pd_RW_gamma provides a framework to manually specify the settings of the proposal distribution of the random walk (RW) Metropolis sampler of the transition probability matrix gamma of the multilevel hidden Markov model, and creates on object of the class mHMM_pdRW_gamma. The RW metropolis sampler is used for sampling the subject level parameter estimates relating to the transition probability matrix gamma, that is, the Multinomial logistic regression intercepts.

Usage

```
pd_RW_gamma(m, gamma_int_mle0, gamma_scalar, gamma_w)
```

Arguments

m	Numeric vector with length 1 denoting the number of hidden states.
gamma_int_mle0	A matrix with m rows and m - 1 columns denoting the starting values for the maximum likelihood (ML) estimates of the Multinomial logit regression intercepts of the transition probability matrix gamma. ML parameters to be estimated are 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.

Details

When no manual values for the settings of the proposal distribution of the random walk (RW) Metropolis sampler are specified at all (that is, the function pd_RW_gamma is not used), 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.

Within the function mHMM, the acceptance rate of the RW metropolis sampler relating to the transition probability matrix gamma can be tracked using the output parameter gamma_naccept. An acceptance rate of about 23% is considered optimal when many parameters are being updated at once (Gelman, Carlin, Stern & Rubin, 2014).

Value

pd_RW_gamma returns an object of class mHMM_pdRW_gamma, containing settings of the proposal distribution of the random walk (RW) Metropolis sampler on the transition probability matrix gamma of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function mHMM, and checked for correct input dimensions. The object contains the following components:

m Numeric vector denoting the number of hidden states, used for checking equivalent general model properties specified under pd_RW_gamma and mHMM.

gamma_int_mle0 A matrix containing the starting values for the maximum likelihood (ML) estimates of the Multinomial logit regression intercepts of the transition probability matrix gamma.

gamma_scalar A numeric vector with length 1 denoting the scale factor s of the proposal distribution.

gamma_w A numeric vector with length 1 denoting denoting the weight for the overall log likelihood in the fractional likelihood.

References

Gelman A, Carlin JB, Stern HS, Rubin DB (2014). *Bayesian Data Analysis vol. 2*. Taylor & Francis.

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

Examples

```
###### Example using package example data, see ?nonverbal
# specifying general model properties:
m <- 3

# specifying manual values for RW metropolis sampler on gamma
gamma_int_mle0 <- matrix(c( -2, -2,
```

```
                            2,  0,
                            0,  3), byrow = TRUE, nrow = m, ncol = m - 1)
gamma_scalar <- c(2)
gamma_w <- c(0.2)
manual_gamma_sampler <- pd_RW_gamma(m = m, gamma_int_mle0 = gamma_int_mle0,
                                    gamma_scalar = gamma_scalar,
                                    gamma_w = gamma_w)

# specifying starting values
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

start_TM <- diag(.7, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .1
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05,
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          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,
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9,
                          0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# 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_3st_RWgamma <- mHMM(s_data = nonverbal,
                        gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                        start_val = c(list(start_TM), start_EM),
                        gamma_sampler = manual_gamma_sampler,
                        mcmc = list(J = 11, burn_in = 5))
out_3st_RWgamma
summary(out_3st_RWgamma)

# checking acceptance rate (for illustrative purposes, in the example,
# J is too low for getting a fair indication)
J_it <- 11 - 1 # accept/reject starts at iteration 2 of MCMC algorithm
out_3st_RWgamma$gamma_naccept / J_it
# average acceptance rate over all subjects per parameter
apply(out_3st_RWgamma$gamma_naccept / J_it, 2, mean)
```

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 (categorical data) or means and standard deviation (continuous data).

Usage

```
## S3 method for class 'mHMM'
plot(
  x,
  component = "gamma",
  dep = 1,
  col,
  dep_lab,
  cat_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. Defaults to dep = 1.
col	Vector of colors for the posterior density lines. If one is plotting the posterior densities for gamma, or the posterior densities of Normally distributed emission probabilities, the vector has length m (i.e., number of hidden states). If one is plotting the posterior densities for categorical emission probabilities, the vector has length q_emiss[k] (i.e., the number of outcome categories for the dependent variable k).

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.
cat_lab	Optional vector of strings when plotting the posterior densities of categorical emission probabilities, denoting the labels of the categorical outcome values. Automatically generated when not provided.
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)

Details

Note that the standard deviation of the (variable and) state specific Normal emission distribution in case of continuous data is fixed over subjects. Hence, for the standard deviation, only the posterior distribution at the group level is plotted.

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

```
###### Example on package example data, see ?nonverbal
# First run the function mHMM on 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):
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, see ?nonverbal
# 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))
```

`prior_emiss_cat` *Specifying informative hyper-prior on the categorical emission distribution(s) of the multilevel hidden Markov model*

Description

`prior_emiss_cat` provides a framework to manually specify an informative hyper-prior on the categorical emission distribution(s). `prior_emiss_cat` creates an object of class `mHMM_prior_emiss` used by the function `mHMM`, and additionally attaches the class `cat` to signal use for categorical observations. Note that the hyper-prior distribution on the categorical emission probabilities are on the intercepts (and, if subject level covariates are used, regression coefficients) of the Multinomial logit model used to accommodate the multilevel framework of the data, instead of on the probabilities directly. The set of hyper-prior distributions consists of a multivariate Normal hyper-prior distribution on the vector of means (i.e., intercepts and regression coefficients), and an Inverse Wishart hyper-prior distribution on the covariance matrix.

Usage

```
prior_emiss_cat(
  gen,
  emiss_mu0,
  emiss_K0 = NULL,
  emiss_V = NULL,
  emiss_nu = NULL,
  n_xx_emiss = NULL
)
```

Arguments

`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`: only to be specified if the data represents categorical data. Numeric vector with length `n_dep` denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.

`emiss_mu0` A list of lists: `emiss_mu0` contains `n_dep` lists, i.e., one list for each dependent variable `k`. Each list `k` contains `m` matrices; one matrix for each set of emission probabilities within a state. The matrices contain the hypothesized hyper-prior mean values of the intercepts of the Multinomial logit model on the categorical emission probabilities. Hence, each matrix consists of one row (when not including covariates in the model) and `q_emiss[k]` - 1 columns. If covariates are used, the number of rows in each matrix in the list is equal to 1 + n_xx (i.e., the first row corresponds 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).

`emiss_K0` Optional list containing `n_dep` elements corresponding to each dependent variable `k`. Each element `k` is a numeric vector with length 1 (when no covariates are used) denoting the number of hypothetical prior subjects on which the set of hyper-prior mean intercepts specified in `emiss_mu0` are based. When covariates are used: each element is a numeric vector with length 1 + n_xx denoting the number of hypothetical prior subjects on which the set of intercepts (first value) and set of regression coefficients (subsequent values) are based.

`emiss_V` Optional list containing `n_dep` elements corresponding to each dependent variable `k`, where each element `k` is a matrix of `q_emiss[k]` - 1 by `q_emiss[k]` - 1 containing the variance-covariance of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

`emiss_nu` Optional list containing `n_dep` elements corresponding to each dependent variable `k`. Each element `k` is a numeric vector with length 1 denoting the degrees of freedom of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

`n_xx_emiss` Optional numeric vector with length `n_dep` denoting the number of (level 2) covariates used to predict the emission distribution of each of the dependent variables `k`. When omitted, the model assumes no covariates are used to predict the emission distribution(s).

Details

Estimation of the mHMM proceeds within a Bayesian context, hence a hyper-prior distribution has to be defined for the group level parameters. Default, non-informative priors are used unless specified otherwise by the user. For each dependent variable, each row of the categorical emission probability matrix (i.e., the probability to observe each category (columns) within each of the states (rows)) has its own set of Multinomial logit intercepts, which are assumed to follow a multivariate normal distribution. Hence, the hyper-prior distributions for the intercepts consists of a multivariate Normal hyper-prior distribution on the vector of means, and an Inverse Wishart hyper-prior distribution on the covariance matrix. Note that only the general model properties (number of states `m`, number of dependent variables `n_dep`, and number of observed categories for each dependent variable `q_emiss`) and values of the hypothesized hyper-prior mean values of the Multinomial logit intercepts have to be specified by the user, default values are available for all other hyper-prior distribution parameters.

Given that the hyper-priors are specified on the intercepts of the Multinomial logit model intercepts instead of on the categorical emission probabilities directly, specifying a hyper-prior can seem rather daunting. However, see the function `prob_to_int` and `int_to_prob` for translating probabilities to a set of Multinomial logit intercepts and vice versa.

Note that emiss_K0, emiss_nu and emiss_V are assumed equal over the states. When the hyper-prior values for emiss_K0, emiss_nu and emiss_V are not manually specified, the default values are as follows. 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. In addition, when no manual values for the hyper-prior on the categorical emission distribution are specified at all (that is, the function prior_emiss_cat is not used), all elements of the matrices contained in emiss_mu0 are set to 0 in the function mHMM.

Note that in case covariates are specified, the hyper-prior parameter values of the inverse Wishart distribution on the covariance matrix remain unchanged, as the estimates of the regression coefficients for the covariates are fixed over subjects.

Value

prior_emiss_cat returns an object of class mHMM_prior_emiss, containing informative hyper-prior values for the categorical emission distribution(s) of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function mHMM, and thoroughly checked for correct input dimensions. The object contains the following components:

gen A list containing the elements m, n_dep, and q_emiss, used for checking equivalent general model properties specified under prior_emiss_cat and mHMM.

emiss_mu0 A list of lists containing the hypothesized hyper-prior mean values of the intercepts of the Multinomial logit model on the categorical emission probabilities.

emiss_K0 A list containing n_dep elements denoting the number of hypothetical prior subjects on which the set of hyper-prior mean intercepts specified in emiss_mu0 are based.

emiss_nu A list containing n_dep elements denoting the degrees of freedom of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

emiss_V A list containing n_dep elements containing the variance-covariance of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

n_xx_emiss A numeric vector denoting the number of (level 2) covariates used to predict the emission distribution of each of the dependent variables. When no covariates are used, n_xx_emiss equals NULL.

See Also

prior_gamma for manually specifying an informative hyper-prior on the transition probability matrix gamma, prob_to_int for transforming a set of probabilities to a set of Multinomial logit regression intercepts, and mHMM for fitting a multilevel hidden Markov model.

Examples

```
###### Example using package example data, see ?nonverbal
# specifying general model properties:
m <- 3
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# hypothesized mean emission probabilities
prior_prob_emiss_cat <- list(matrix(c(0.10, 0.80, 0.10,
```

```
                                    0.80, 0.10, 0.10,
                                    0.40, 0.40, 0.20), byrow = TRUE,
                                  nrow = m, ncol = q_emiss[1]), # vocalizing patient,
                                  # prior belief: state 1 - much talking, state 2 -
                                  # no talking, state 3 - mixed
                           matrix(c(0.30, 0.70,
                                    0.30, 0.70,
                                    0.30, 0.70), byrow = TRUE, nrow = m,
                                  ncol = q_emiss[2]), # looking patient
                                  # prior belief: all 3 states show frequent looking
                                  # behavior
                           matrix(c(0.80, 0.10, 0.10,
                                    0.10, 0.80, 0.10,
                                    0.40, 0.40, 0.20), byrow = TRUE,
                                  nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                                  # prior belief: state 1 - no talking, state 2 -
                                  # frequent talking, state 3 - mixed
                           matrix(c(0.30, 0.70,
                                    0.30, 0.70,
                                    0.30, 0.70), byrow = TRUE, nrow = m,
                                  ncol = q_emiss[4])) # looking therapist
                                  # prior belief: all 3 states show frequent looking
                                  # behavior

# using the function prob_to_int to obtain intercept values for the above specified
# categorical emission distributions
prior_int_emiss <- sapply(prior_prob_emiss_cat, prob_to_int)
emiss_mu0 <- rep(list(vector(mode = "list", length = m)), n_dep)
for(k in 1:n_dep){
  for(i in 1:m){
  emiss_mu0[[k]][[i]] <- matrix(prior_int_emiss[[k]][i,], nrow = 1)
  }
}

emiss_K0 <- rep(list(c(1)), n_dep)
emiss_nu <- list(c(5), c(4), c(5), c(4))
emiss_V <- list(diag(5, q_emiss[1] - 1),
                diag(4, q_emiss[2] - 1),
                diag(5, q_emiss[3] - 1),
                diag(4, q_emiss[4] - 1))

manual_prior_emiss <- prior_emiss_cat(gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                                  emiss_mu0 = emiss_mu0, emiss_K0 = emiss_K0,
                                  emiss_nu = emiss_nu, emiss_V = emiss_V)


# using the informative hyper-prior in a model

# specifying starting values
start_TM <- diag(.7, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .1
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05,
```

```
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
                 matrix(c(0.1, 0.9,
                          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,
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                 matrix(c(0.1, 0.9,
                          0.1, 0.9,
                          0.1, 0.9), byrow = TRUE, nrow = m,
                        ncol = q_emiss[4])) # looking therapist

# 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_3st_infemiss <- mHMM(s_data = nonverbal,
                    gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                    start_val = c(list(start_TM), start_EM),
                    emiss_hyp_prior = manual_prior_emiss,
                    mcmc = list(J = 11, burn_in = 5))

out_3st_infemiss
summary(out_3st_infemiss)
```

`prior_emiss_cont` *Specifying informative hyper-prior on the continuous emission distribution(s) of the multilevel hidden Markov model*

Description

`prior_emiss_cont` provides a framework to manually specify an informative hyper-prior on the Normal (i.e., Gaussian) emission distributions. `prior_emiss_cont` creates an object of class `mHMM_prior_emiss` used by the function `mHMM`, and additionally attaches the class `cont` to signal use for continuous observations. The set of hyper-prior distributions consists of a Normal-Inverse-Gamma distribution (i.e., assuming both unknown population mean and variance between subject level means) on the vector of means (i.e., intercepts and regression coefficients), and an Inverse gamma distribution (i.e., assuming a known mean) on each of the (fixed over subjects) emission variances.

Usage

```
prior_emiss_cont(
  gen,
```

```
  emiss_mu0,
  emiss_K0,
  emiss_V,
  emiss_nu,
  emiss_a0,
  emiss_b0,
  n_xx_emiss = NULL
)
```

Arguments

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: only to be specified if the data represents categorical data. Numeric vector with length n_dep denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.
emiss_mu0	A list containing n_dep matrices, i.e., one list for each dependent variable k. Each matrix contains the hypothesized hyper-prior means of the Normal emission distributions in each of the states. Hence, each matrix consists of one row (when not including covariates in the model) and m columns. If covariates are used, the number of rows in each matrix in the list is equal to 1 + n_xx (i.e., the first row corresponds to the hyper-prior means, the subsequent rows correspond to the hyper-prior values of the regression coefficients connected to each of the covariates).
emiss_K0	A list containing n_dep elements corresponding to each dependent variable k. Each element k is a numeric vector with length 1 (when no covariates are used) denoting the number of hypothetical prior subjects on which the set of hyper-prior means specified in emiss_mu0 are based. When covariates are used: each element is a numeric vector with length 1 + n_xx denoting the number of hypothetical prior subjects on which the set of means (first value) and set of regression coefficients (subsequent values) are based.
emiss_V	A list containing n_dep elements corresponding to each of the dependent variables k, where each element k is a vector with length m containing the hypothesized variance between the subject (emission distribution) means, which are assumed to follow a Inverse Gamma hyper-prior distribution (note: here, the Inverse Gamma hyper-prior distribution is parametrized as a scaled inverse chi-squared distribution).
emiss_nu	A list containing n_dep elements corresponding to each dependent variable k. Each element k is a numeric vector with length 1 denoting the degrees of freedom of the Inverse Gamma hyper-prior distribution on the between subject variance of the emission distribution means (note: here, the Inverse Gamma hyper-prior distribution is parametrized as a scaled inverse chi-squared distribution).
emiss_a0	A list containing n_dep elements corresponding to each of the dependent variables k, where each element k is a vector with length m containing the shape values of the Inverse Gamma hyper-prior on each of the (fixed over subjects)

emission standard deviation^2 of the Normal emission distributions (note: here the standard Inverse Gamma parametrization is used).

`emiss_b0` A list containing `n_dep` elements corresponding to each of the dependent variables `k`, where each element `k` is a vector with length `m` containing the scale values of the Inverse Gamma hyper-prior on each of the (fixed over subjects) emission standard deviation^2 of the Normal emission distributions (note: here the standard Inverse Gamma parametrization is used).

`n_xx_emiss` Optional numeric vector with length `n_dep` denoting the number of (level 2) covariates used to predict the emission distribution of each of the dependent variables `k`. When omitted, the model assumes no covariates are used to predict the emission distribution(s).

Details

Estimation of the mHMM proceeds within a Bayesian context, hence a hyper-prior distribution has to be defined for the group level parameters. To avoid problems with 'label switching' when dealing with continuous emission distribution(s) (i.e., switching of the labels of the hidden states while sampling from the MCMC), the user is forced to specify hyper-prior parameter values when using continuous emission distributions (i.e., default, non-informative priors are not available for continuous emission distributions).

Note that `emiss_K0` and `emiss_nu` are assumed equal over the states. Also note that in case covariates are specified, the hyper-prior parameter values of the inverse Wishart distribution on the covariance matrix remain unchanged, as the estimates of the regression coefficients for the covariates are fixed over subjects.

Value

`prior_emiss_cont` returns an object of class `mHMM_prior_emiss`, containing informative hyper-prior values for the continuous emission distribution(s) of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function `mHMM`, and thoroughly checked for correct input dimensions. The object contains the following components:

`gen` A list containing the elements `m`, and `n_dep`, used for checking equivalent general model properties specified under `prior_emiss_cont` and `mHMM`.

`emiss_mu0` A lists containing the hypothesized hyper-prior means of the Normal distribution on the continuous emission probabilities.

`emiss_K0` A list containing `n_dep` elements denoting the number of hypothetical prior subjects on which the set of hyper-prior means specified in `emiss_mu0` are based.

`emiss_V` A list containing `n_dep` elements containing the variance of the Inverse Gamma hyper-prior distribution on the between subject variance of the emission distribution means.

`emiss_nu` A list containing `n_dep` elements denoting the degrees of freedom of the Inverse Gamma hyper-prior distribution on the between subject variance of the emission distribution means.

`emiss_a0` A list containing `n_dep` elements denoting the shape values of the Inverse Gamma hyper-prior on each of the (fixed over subjects) emission standard deviation^2 of the Normal emission distributions.

`emiss_b0` A list containing `n_dep` elements denoting the scale values of the Inverse Gamma hyper-prior on each of the (fixed over subjects) emission standard deviation^2 of the Normal emission distributions.

`n_xx_emiss` A numeric vector denoting the number of (level 2) covariates used to predict the emission distribution of each of the dependent variables. When no covariates are used, `n_xx_emiss` equals `NULL`.

See Also

`prior_gamma` for manually specifying an informative hyper-prior on the transition probability matrix gamma, and `mHMM` for fitting a multilevel hidden Markov model.

Examples

```
###### Example using simulated data
# specifying general model properties:
m <- 3
n_dep <- 2

# hypothesized hyper-prior values for the continuous emission distribution
manual_prior_emiss <- prior_emiss_cont(
                        gen = list(m = m, n_dep = n_dep),
                        emiss_mu0 = list(matrix(c(30, 70, 170), nrow = 1),
                                         matrix(c(7, 8, 18), nrow = 1)),
                        emiss_K0 = list(1, 1),
                         emiss_V =  list(rep(5^2, m), rep(0.5^2, m)),
                        emiss_nu = list(1, 1),
                        emiss_a0 = list(rep(1.5, m), rep(1, m)),
                        emiss_b0 = list(rep(20, m), rep(4, m)))

# to use the informative priors in a model, simulate multivariate continuous data
n_t     <- 100
n       <- 10

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 <- list(matrix(c( 50, 10,
                              100, 10,
                              150, 10), nrow = m, byrow = TRUE),
                    matrix(c(5, 2,
                             10, 5,
                             20, 3), nrow = m, byrow = TRUE))

data_cont <- sim_mHMM(n_t = n_t, n = n, data_distr = 'continuous',
                      gen = list(m = m, n_dep = n_dep),
                      gamma = gamma, emiss_distr = emiss_distr,
                      var_gamma = .1, var_emiss = c(5^2, 0.2^2))

# using the informative hyper-prior in a 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_3st_cont_sim_infemiss <- mHMM(s_data = data_cont$obs,
                    data_distr = "continuous",
                    gen = list(m = m, n_dep = n_dep),
                    start_val = c(list(gamma), emiss_distr),
                    emiss_hyp_prior = manual_prior_emiss,
                    mcmc = list(J = 11, burn_in = 5))

out_3st_cont_sim_infemiss
summary(out_3st_cont_sim_infemiss)
```

`prior_emiss_count` *Specifying informative hyper-priors on the count emission distribution(s) of the multilevel hidden Markov model*

Description

`prior_emiss_count` provides a framework to manually specify an informative hyper-prior on the count emission distributions. `prior_emiss_count` creates an object of class `mHMM_prior_emiss` used by the function `mHMM`, and additionally attaches the class `count` to signal use for count observations. The set of hyper-prior distributions consists of a lognormal-Inverse-Gamma distribution (i.e., assuming both unknown population mean and variance between subject level means) on the vector of Poisson means (i.e., intercepts and regression coefficients).

Usage

```
prior_emiss_count(
  gen,
  emiss_mu0,
  emiss_K0,
  emiss_V,
  emiss_nu,
  n_xx_emiss = NULL,
  log_scale = FALSE
)
```

Arguments

`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`: only to be specified if the data represents categorical data. Numeric vector with length `n_dep` denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.

`emiss_mu0` A list containing `n_dep` matrices, i.e., one list for each dependent variable `k`. Each matrix contains the hypothesized hyper-prior means of the Poisson emission distribution in each of the states in the natural (positive real numbers) scale. Hence, each matrix consists of one row (when not including covariates in the model) and `m` columns. If covariates are used, the number of rows in each matrix in the list is equal to 1 + n_xx_emiss (i.e., the first row corresponds to the hyper-prior means, the subsequent rows correspond to the hyper-prior values of the regression coefficients connected to each of the covariates). If covariates are used to predict the emission distribution, then `emiss_mu0` should be specified in the logarithmic scale, and `log_scale` set to `TRUE`.

`emiss_K0` A list containing `n_dep` elements corresponding to each dependent variable `k`. Each element `k` is a numeric vector with length 1 (when no covariates are used) denoting the number of hypothetical prior subjects on which the set of hyper-prior means specified in `emiss_mu0` are based. When covariates are used: each element is a numeric vector with length 1 + n_xx denoting the number of hypothetical prior subjects on which the set of means (first value) and set of regression coefficients (subsequent values) are based.

`emiss_V` A list containing `n_dep` elements corresponding to each of the dependent variables `k`, where each element `k` is a vector with length `m` containing the hypothesized variance between the subject (emission distribution) means in the natural (positive real numbers) scale, which are assumed to follow a Inverse Gamma hyper-prior distribution (note: here, the Inverse Gamma hyper-prior distribution is parametrized as a scaled inverse chi-squared distribution). If covariates are used to predict the emission distribution, then `emiss_V` should be specified in the logarithmic scale, and `log_scale` set to `TRUE`.

`emiss_nu` A list containing `n_dep` elements corresponding to each dependent variable `k`. Each element `k` is a numeric vector with length 1 denoting the degrees of freedom of the Inverse Gamma hyper-prior distribution on the between subject variance of the emission distribution means (note: here, the Inverse Gamma hyper-prior distribution is parametrized as a scaled inverse chi-squared distribution).

`n_xx_emiss` Optional numeric vector with length `n_dep` denoting the number of (level 2) covariates used to predict the emission distribution of each of the dependent variables `k`. When omitted, the model assumes no covariates are used to predict the emission distribution(s).

`log_scale` A logical scalar. Should `emiss_mu0` and `emiss_V` be specified in the logarithmic scale (`log_scale = TRUE`) or the natural scale (`log_scale = FALSE`). The default equals `log_scale = FALSE`.

Details

Estimation of the mHMM proceeds within a Bayesian context, hence a hyper-prior distribution has to be defined for the group level parameters. To avoid problems with 'label switching' when dealing with continuous emission distribution(s) (i.e., switching of the labels of the hidden states while sampling from the MCMC), the user is forced to specify hyper-prior parameter values when using count emission distributions (i.e., default, non-informative priors are not available for count emission distributions).

Note that `emiss_K0` and `emiss_nu` are assumed equal over the states. Also note that in case covariates are specified, the hyper-prior parameter values of the inverse Wishart distribution on the

covariance matrix remain unchanged, as the estimates of the regression coefficients for the covariates are fixed over subjects.

Also note that for simplicity the hyper-prior means and variances of the lognormal distribution, `emiss_mu0` and `emiss_V`, by default have to be specified in the natural (positive real numbers) scale and not in the logarithmic scale. `prior_emiss_count` returns the corresponding values of the parameters on the logarithmic scale. If the user wants to manually specify these values on the logarithmic scale, please set the argument `log_scale` to `TRUE` in `prior_emiss_count`. If covariates are used to predict the emission distribution, then the logarithmic scale should be used for the inputs `emiss_mu0` and `emiss_V`, and set `log_scale = TRUE`. To aid the user in transforming the variance to the logarithmic scale, the function 'var_to_logvar()' can be used.

Value

`prior_emiss_count` returns an object of class `mHMM_prior_emiss`, containing informative hyper-prior values for the continuous emission distribution(s) of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function `mHMM`, and thoroughly checked for correct input dimensions. The object contains the following components:

`gen` A list containing the elements `m`, and `n_dep`, used for checking equivalent general model properties specified under `prior_emiss_count` and `mHMM`.

`emiss_mu0` A lists containing the hypothesized hyper-prior means of the the Poisson distribution used to model the count emissions.

`emiss_K0` A list containing `n_dep` elements denoting the number of hypothetical prior subjects on which the set of hyper-prior means specified in `emiss_mu0` are based.

`emiss_V` A list containing `n_dep` elements containing the variance of the Inverse Gamma hyper-prior distribution on the between subject variance of the emission distribution means.

`emiss_nu` A list containing `n_dep` elements denoting the degrees of freedom of the Inverse Gamma hyper-prior distribution on the between subject variance of the emission distribution means.

`n_xx_emiss` A numeric vector denoting the number of (level 2) covariates used to predict the emission distribution of each of the dependent variables. When no covariates are used, `n_xx_emiss` equals `NULL`.

See Also

`prior_gamma` for manually specifying an informative hyper-prior on the transition probability matrix gamma, and `mHMM` for fitting a multilevel hidden Markov model.

Examples

```
###### Example using simulated data
# specifying general model properties:
m <- 3
n_dep <- 2

# hypothesized hyper-prior values for the count emission distribution
manual_prior_emiss <- prior_emiss_count(
                        gen = list(m = m, n_dep = n_dep),
                        emiss_mu0 = list(matrix(c(30, 70, 170), nrow = 1),
```

```
                                        matrix(c(7, 8, 18), nrow = 1)),
                          emiss_K0 = list(1, 1),
                          emiss_V =  list(rep(16, m), rep(4, m)),
                          emiss_nu = list(0.1, 0.1))

# to use the informative priors in a model, simulate multivariate count data
n_t     <- 100
n       <- 10

# Specify group-level transition and emission means
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 <- list(matrix(log(c( 50,
                              100,
                              150)), nrow = m, byrow = TRUE),
                    matrix(log(c(5,
                             10,
                             20)), nrow = m, byrow = TRUE))
# Simulate data
data_count <- sim_mHMM(n_t = n_t, n = n, data_distr = 'count',
                       gen = list(m = m, n_dep = n_dep),
                       gamma = gamma, emiss_distr = emiss_distr,
                       var_gamma = .1, var_emiss = c(.05, 0.01), log_scale = TRUE)

# Specify starting values
start_gamma <- gamma
start_emiss <- list(matrix(c(50,
                             100,
                             150), nrow = m, byrow = TRUE),
                    matrix(c(5,
                             10,
                             20), nrow = m, byrow = TRUE))


# using the informative hyper-prior in a 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_3st_count_sim_infemiss <- mHMM(s_data = data_count$obs,
                    data_distr = "count",
                    gen = list(m = m, n_dep = n_dep),
                    start_val = c(list(start_gamma), start_emiss),
                    emiss_hyp_prior = manual_prior_emiss,
                    mcmc = list(J = 11, burn_in = 5))

out_3st_count_sim_infemiss
summary(out_3st_count_sim_infemiss)
```

prior_gamma *Specifying informative hyper-prior on the transition probability matrix gamma of the multilevel hidden Markov model*

Description

prior_gamma provides a framework to manually specify an informative hyper-prior on the transition probability matrix gamma, and creates an object of class mHMM_prior_gamma used by the function mHMM. Note that the hyper-prior distribution on the transition probabilities are on the intercepts (and, if subject level covariates are used, regression coefficients) of the Multinomial logit model used to accommodate the multilevel framework of the data, instead of on the probabilities directly. The set of hyper-prior distributions consists of a multivariate Normal hyper-prior distribution on the vector of means (i.e., intercepts and regression coefficients), and an Inverse Wishart hyper-prior distribution on the covariance matrix.

Usage

```
prior_gamma(
  m,
  gamma_mu0,
  gamma_K0 = NULL,
  gamma_nu = NULL,
  gamma_V = NULL,
  n_xx_gamma = NULL
)
```

Arguments

m	Numeric vector with length 1 denoting the number of hidden states.
gamma_mu0	A list containing m matrices; one matrix for each row of the transition probability matrix gamma. Each matrix contains the hypothesized hyper-prior mean values of the intercepts of the Multinomial logit model on the transition probabilities gamma. Hence, each matrix consists of one row (when not including covariates in the model) and m - 1 columns. If covariates are used, the number of rows in each matrix in the list is equal to 1 + n_xx_gamma (i.e., the first row corresponds 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).
gamma_K0	Optional numeric vector with length 1 (when no covariates are used) denoting the number of hypothetical prior subjects on which the set of hyper-prior mean intercepts specified in gamma_mu0 are based. When covariates are used: Numeric vector with length 1 + n_xx_gamma denoting the number of hypothetical prior subjects on which the set of intercepts (first value) and set of regression coefficients (subsequent values) are based.
gamma_nu	Optional numeric vector with length 1 denoting the degrees of freedom of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

gamma_V — Optional matrix of m - 1 by m - 1 containing the variance-covariance matrix of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

n_xx_gamma — Optional numeric vector with length 1 denoting the number of (level 2) covariates (excluding the intercept) used to predict the transition probability matrix gamma. When omitted, the model assumes no covariates are used to predict gamma.

Details

Estimation of the mHMM proceeds within a Bayesian context, hence a hyper-prior distribution has to be defined for the group level parameters. Default, non-informative priors are used unless specified otherwise by the user. Each row of the transition probability matrix has its own set of Multinomial logit intercepts, which are assumed to follow a multivariate normal distribution. Hence, the hyper-prior distributions for the intercepts consists of a multivariate Normal hyper-prior distribution on the vector of means, and an Inverse Wishart hyper-prior distribution on the covariance matrix. Note that only the number of states m and values of the hypothesized hyper-prior mean values of the Multinomial logit intercepts have to be specified by the user, default values are available for all other hyper-prior distribution parameters.

Given that the hyper-priors are specified on the intercepts of the Multinomial logit model intercepts instead of on the probabilities of the transition probability matrix gamma directly, specifying a hyper-prior can seem rather daunting. However, see the function prob_to_int and int_to_prob for translating probabilities to a set of Multinomial logit intercepts and vice versa.

Note that gamma_K0, gamma_nu and gamma_V are assumed equal over the states. When the hyper-prior values for gamma_K0, gamma_nu and gamma_V are not manually specified, the default values are as follows. 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. In addition, when no manual values for the hyper-prior on gamma are specified at all (that is, the function prior_gamma is not used), all elements of the matrices contained in gamma_mu0 are set to 0 in the function mHMM.

Note that in case covariates are specified, the hyper-prior parameter values of the inverse Wishart distribution on the covariance matrix remain unchanged, as the estimates of the regression coefficients for the covariates are fixed over subjects.

Value

prior_gamma returns an object of class mHMM_prior_gamma, containing informative hyper-prior values for the transition probability matrix gamma of the multilevel hidden Markov model. The object is specifically created and formatted for use by the function mHMM, and thoroughly checked for correct input dimensions. The object contains the following components:

m Numeric vector denoting the number of hidden states, used for checking equivalent general model properties specified under prior_gamma and mHMM.

gamma_mu0 A list containing the hypothesized hyper-prior mean values of the intercepts of the Multinomial logit model on the transition probability matrix gamma.

gamma_K0 A numeric vector denoting the number of hypothetical prior subjects on which the set of hyper-prior mean intercepts specified in gamma_mu0 are based.

gamma_nu A numeric vector denoting the degrees of freedom of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

gamma_V A matrix containing the variance-covariance of the hyper-prior Inverse Wishart distribution on the covariance of the Multinomial logit intercepts.

n_xx_gamma A numeric vector denoting the number of (level 2) covariates used to predict the transition probability matrix gamma. When no covariates are used, n_xx_gamma equals NULL.

See Also

prior_emiss_cat for manually specifying an informative hyper-prior on the categorical emission distribution(s), prob_to_int for transforming a set of probabilities to a set of Multinomial logit regression intercepts, and mHMM for fitting a multilevel hidden Markov model.

Examples

```
###### Example using package example data, see ?nonverbal
# specifying general model properties:
m <- 3
# representing a prior belief that switching to state 3 does not occur often and
# state 3 has a relative short duration
prior_prob_gamma <- matrix(c(0.70, 0.25, 0.05,
                             0.25, 0.70, 0.05,
                             0.30, 0.30, 0.40), nrow = m, ncol = m, byrow = TRUE)

# using the function prob_to_int to obtain intercept values for the above specified
# transition probability matrix gamma
prior_int_gamma <- prob_to_int(prior_prob_gamma)
gamma_mu0 <- list(matrix(prior_int_gamma[1,], nrow = 1, ncol = m-1),
                  matrix(prior_int_gamma[2,], nrow = 1, ncol = m-1),
                  matrix(prior_int_gamma[3,], nrow = 1, ncol = m-1))

gamma_K0 <- 1
gamma_nu <- 5
gamma_V <- diag(5, m - 1)

manual_prior_gamma <- prior_gamma(m = m, gamma_mu0 = gamma_mu0,
                                  gamma_K0 = gamma_K0, gamma_nu = gamma_nu,
                                  gamma_V = gamma_V)


# using the informative hyper-prior in a model
n_dep <- 4
q_emiss <- c(3, 2, 3, 2)

# specifying starting values
start_TM <- diag(.7, m)
start_TM[lower.tri(start_TM) | upper.tri(start_TM)] <- .1
start_EM <- list(matrix(c(0.05, 0.90, 0.05,
                          0.90, 0.05, 0.05,
                          0.55, 0.45, 0.05), byrow = TRUE,
                        nrow = m, ncol = q_emiss[1]), # vocalizing patient
```

```
                    matrix(c(0.1, 0.9,
                             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,
                             0.55, 0.45, 0.05), byrow = TRUE,
                           nrow = m, ncol = q_emiss[3]), # vocalizing therapist
                    matrix(c(0.1, 0.9,
                             0.1, 0.9,
                             0.1, 0.9), byrow = TRUE, nrow = m,
                           ncol = q_emiss[4])) # looking therapist

# 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_3st_infgamma <- mHMM(s_data = nonverbal,
                    gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                    start_val = c(list(start_TM), start_EM),
                    gamma_hyp_prior = manual_prior_gamma,
                    mcmc = list(J = 11, burn_in = 5))

out_3st_infgamma
summary(out_3st_infgamma)
```

`prob_to_int` *Transforming a set of probabilities to Multinomial logit regression intercepts*

Description

`prob_to_int` transforms a set of state transition or categorical emission observation probabilities to the corresponding Multinomial logit regression intercepts. Note that the first category is assumed to be the reference category, hence no intercept is returned for the first state or category.

Usage

```
prob_to_int(prob_matrix)
```

Arguments

`prob_matrix`	A matrix with number of states OR categories columns and number of rows to be determined by the user, with rows summing to one. For obtaining the set of Multinomial logit regression intercepts of the complete transition probability matrix gamma or categorical emission distribution matrix, the number of rows equals the number of states `m`.

Details

Designed to ease the specification of informative hyper-prior values for the mean intercepts of the transition probability matrix gamma and categorical emission distribution(s) of the multilevel hidden Markov model through the functions prior_gamma and prior_emiss_cat. No check is performed on correct specifications of the dimensions.

Value

prob_to_int returns a matrix containing Multinomial logit regression intercepts, with the number of columns equal to (number of states or categories - 1) and the number of rows equal to the number rows specified in the input matrix. The first state / category is assumed to be the reference category, hence no intercept is returned for this first category.

See Also

int_to_prob for transforming a set of Multinomial logit regression intercepts to a probabilities, prior_gamma and prior_emiss_cat for specifying informative hyper-priors for the the multilevel hidden Markov model and mHMM to fit a multilevel hidden Markov model.

Examples

```
# example for transition probability matrix gamma with 3 states
m <- 3
gamma_prob <- matrix(c(0.6, 0.2, 0.2,
                       0.1, 0.8, 0.1,
                       0.1, 0.1, 0.8), ncol = m, nrow = m, byrow = TRUE)
gamma_int <- prob_to_int(gamma_prob)
gamma_int
```

---

sim_mHMM *Simulate data using a multilevel hidden Markov model*

Description

sim_mHMM simulates data for multiple subjects, for which the data have either categorical or continuous (i.e., normally distributed) observations that follow a hidden Markov model (HMM) with a 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 distribution. In addition to (natural and/or unexplained) heterogeneity between subjects, the subjects parameters can also depend on a covariate.

Usage

```
sim_mHMM(
  n_t,
  n,
  data_distr = "categorical",
  gen,
  gamma,
  emiss_distr,
  start_state = NULL,
  xx_vec = NULL,
  beta = NULL,
  var_gamma = 0.1,
  var_emiss = NULL,
  return_ind_par = FALSE,
  m,
  n_dep,
  q_emiss,
  log_scale = FALSE
)
```

Arguments

n_t	Numeric vector with length 1 denoting 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	Numeric vector with length 1 denoting the number of subjects for which data is simulated.
data_distr	A character vector of length 1 denoting the distribution adopted for the data given the hidden states. It can take the values 'categorical', 'continuous', or 'count', standing for the for categorical observations following a Multinomial logit, continuous observations following a normal distribution, and count observations following a Poisson distribution, correspondingly.
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: only to be specified if the data represents categorical data. Numeric vector with length n_dep denoting the number of observed categories for the categorical emission distribution for each of the dependent variables.
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 list with n_dep elements containing the average population emission distribution(s) of the observations given the hidden states for each of the dependent variables. If data_distr = 'categorical', each element is a matrix with m

rows and q_emiss[k] columns for each of the k in n_dep emission distribution(s). That is, the probability of observing category q (column q) in state i (row i). If data_distr = 'continuous', each element is a matrix with m rows and 2 columns; the first column denoting the mean of state i (row i) and the second column denoting the standard deviation of state i (row i) of the Normal distribution. If data_distr = 'count', each element is a matrix with m rows and 1 column; the only column denoting the logmean of state i (row i) of the lognormal distribution used as prior for the Poisson emissions (note: by default the logmeans should be specified in the natural scale; see argument log_scale below, unless covariates are used to predict a count emission distribution).

start_state — Optional numeric vector with length 1 denoting in which state the simulated state sequence should start. If left unspecified, the simulated state for time point 1 is sampled from the initial state distribution (which is derived from the transition probability matrix gamma).

xx_vec — List of 1 + n_dep vectors containing the covariate(s) to predict the transition probability matrix gamma and/or (specific) emission distribution(s) emiss_distr using the regression parameters specified in beta (see below). The first element in the list xx_vec 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). This means that the covariate used to predict gamma and emiss_distr can either be the same covariate, different covariates, or a covariate for certain elements and none for the other. At this point, it is only possible to use one covariate for both gamma and emiss_distr. For all elements in the list, the number of observations in the vectors should be equal to the number of subjects to be simulated n. If xx_vec is omitted completely, xx_vec defaults to NULL, resembling no covariates at all. Specific elements in the list can also be left empty (i.e., set to NULL) to signify that either the transition probability matrix or (one of) the emission distribution(s) is not predicted by covariates.

beta — List of 1 + n_dep 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 subsequent matrices are used to predict the emission distribution(s) emiss_distr of the dependent variable(s). For gamma and categorical emission distributions, 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 element 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. For categorical emission distributions, the subsequent elements 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[k] - 1 for each of the k in n_dep emission distribution(s). See *details* for more information. For continuous and count emission distributions, the subsequent elements in the list beta consist of a matrix with the number of rows equal to m and 1 column.

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 and emiss_distr using covariates. One of the 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. If covariates are used to predict a count emission distribution (data_distr = 'count'), then the logarithmic scale should be used for the inputs emiss_distr and var_emiss in addition to beta, and also set log_scale = TRUE.

var_gamma — Either a numeric vector with length 1 or a matrix of (m by m - 1) elements denoting the amount of variance between subjects in the transition probability matrix. Note that the value(s) correspond 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. Also note that if only one variance value is provided, it will be adopted for the complete transition probability matrix, hence the variance is assumed fixed across all components. The default equals 0.1, which corresponds to little variation between subjects. If one wants to simulate data from exactly the same HMM for all subjects, var_gamma should be set to 0. Note that if data for only 1 subject is simulated (i.e., n = 1), var_gamma is set to 0.

var_emiss — Either a numeric vector with length n_dep or a list of n_dep matrices denoting the amount of variance between subjects in the emission distribution(s). When opting for a list of matrices: if data_distr = 'categorical', each element of the list is a matrix with m rows and q_emiss[k] - 1 columns for each of the k in n_dep emission distribution(s) between subject variances; if data_distr = 'continuous', each element is a matrix with m rows and 1 column; the single column denoting the variance between the subjects' means of state *i* (row *i*) of the normal distribution used as prior for the Normal emissions. If data_distr = 'count', each element is a matrix with m rows and 1 column; the single column denoting the logvariance between the subjects' logmeans of state *i* (row *i*) of the lognormal distribution used as prior for the Poisson emissions (note: by default the logvariances should be specified in the natural scale; see argument log_scale below).

For categorical data, 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. For continuous data, this value corresponds to the variance in the mean of the emission distribution(s) across subjects. For count data, it corresponds to the variance in the logmean of the emission distribution(s) across subjects and by should be specified in the natural scale (see argument log_scale below). Note that if only one variance value provided for each emission distribution, the variance is assumed fixed across states (and, for the categorical distribution, categories within a state) within an emission distribution. The default equals 0.1, which corresponds to little variation between subjects given categorical observations. If one wants to simulate data from exactly the same HMM for all subjects, var_emiss should be set to a vector of 0's. Note that if data for only 1 subject is simulated (i.e., n = 1), var_emiss is set to a vector of 0's.

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.

m	The argument m is deprecated; please specify using the input parameter gen.
n_dep	The argument n_dep is deprecated; please specify using the input parameter gen.
q_emiss	The argument q_emiss is deprecated; please specify using the input parameter gen (only to be specified when simulating categorical data).
log_scale	A logical scalar. If data_distr = 'count', should emiss_distr (i.e., the sample average logmeans) and var_emiss (i.e., the between subject logvariances) of the lognormal prior adopted for the count be specified in the logarithmic scale (log_scale = TRUE) or the natural (i.e., real positive numbers) scale (log_scale = FALSE). The default equals log_scale = FALSE. Note that if covariates beta are used to predict the count emission distribution, then the logarithmic scale should be used for the inputs emiss_distr, beta, and var_emiss, and also set log_scale = TRUE.

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 of the mean population parameters is assumed fixed for all components of the transition probability matrix and for all components of the emission distribution.

One can simulate multivariate data. That is, the hidden states depend on more than 1 observed variable simultaneously. The distributions of multiple dependent variables for multivariate data are assumed to be independent, and all distributions for one dataset have to be of the same type (either categorical or continuous).

Note that 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 categorical emission distribution(s) 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.

Note that when simulating count data (data_distr = 'count'), by default the emission distribution parameters emiss_distr and var_emiss are specified in the natural (positive real numbers) scale (as opposite to the logarithmic scale). If the user wants to manually specify these values on the logarithmic scale, please set the argument log_scale = TRUE. Also note that if covariates are used to predict a count emission distribution, then the logarithmic scale should be used for the inputs emiss_distr, beta, and var_emiss, and also set log_scale = TRUE.

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

```
## Examples on univariate categorical data
# simulating data for 10 subjects with each 100 categorical observations
n_t     <- 100
n       <- 10
m       <- 3
n_dep   <- 1
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 <- list(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, gen = list(m = m, n_dep = n_dep, 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, gen = list(m = m, n_dep = n_dep, 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
n_dep   <- 1
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 <- list(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, gen = list(m = m, n_dep = n_dep, 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, gen = list(m = m, n_dep = n_dep, q_emiss = q_emiss),
                gamma = gamma, emiss_distr = emiss_distr, var_gamma = .5, var_emiss = .5)
data4


## Example on multivariate continuous data
# simulating multivariate continuous data
n_t     <- 100
n       <- 10
m       <- 3
n_dep   <- 2

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 <- list(matrix(c( 50, 10,
                              100, 10,
                              150, 10), nrow = m, byrow = TRUE),
                    matrix(c(5, 2,
                             10, 5,
                             20, 3), nrow = m, byrow = TRUE))

data_cont <- sim_mHMM(n_t = n_t, n = n, data_distr = 'continuous',
                      gen = list(m = m, n_dep = n_dep),
                      gamma = gamma, emiss_distr = emiss_distr,
                      var_gamma = .5, var_emiss = c(5^2, 0.2^2))

head(data_cont$states)
head(data_cont$obs)


## Example on multivariate count data without covariates
```

```
n_t     <- 200     # Number of observations on the dependent variable
m       <- 3        # Number of hidden states
n_dep   <- 2        # Number of dependent variables
n_subj  <- 30        # Number of subjects

gamma   <- matrix(c(0.9, 0.05, 0.05,
                    0.2, 0.7, 0.1,
                    0.2,0.3, 0.5), ncol = m, byrow = TRUE)

emiss_distr <- list(matrix(c(20,
                             10,
                             5), nrow = m, byrow = TRUE),
                    matrix(c(50,
                             3,
                             20), nrow = m, byrow = TRUE))

# Define between subject variance to use on the simulating function:
# here, the variance is varied over states within the dependent variable.
var_emiss <- list(matrix(c(5.0, 3.0, 1.5), nrow = m),
                  matrix(c(5.0, 5.0, 5.0), nrow = m))

# Simulate count data:
data_count <- sim_mHMM(n_t = n_t,
                       n = n_subj,
                       data_distr = "count",
                       gen = list(m = m, n_dep = n_dep),
                       gamma = gamma,
                       emiss_distr = emiss_distr,
                       var_gamma = 0.1,
                       var_emiss = var_emiss,
                       return_ind_par = TRUE)


## Example on multivariate count data with covariates
# Simulate data with one covariate for each count dependent variable

n_t     <- 200     # Number of observations on the dependent variable
m       <- 3        # Number of hidden states
n_dep   <- 3        # Number of dependent variables
n_subj  <- 30        # Number of subjects

gamma   <- matrix(c(0.9, 0.05, 0.05,
                    0.2, 0.7, 0.1,
                    0.2,0.3, 0.5), ncol = m, byrow = TRUE)

emiss_distr <- list(matrix(c(20,
                             10,
                             5), nrow = m, byrow = TRUE),
                    matrix(c(15,
                             2,
                             5), nrow = m, byrow = TRUE),
                    matrix(c(50,
                             3,
```

```
                              20), nrow = m, byrow = TRUE))

# Since we have covariates, we specify inputs in the log scale:
emiss_distr_log <- lapply(emiss_distr, function(e) log(e))

# Define list of vectors of covariate values
set.seed(42)
xx_vec <- c(list(NULL),rep(list(rnorm(n_subj,mean = 0, sd = 0.1)),3))

# Define object beta with regression coefficients for the three dependent variables
beta      <- rep(list(NULL), n_dep+1)
beta[[2]] <- matrix(c(1,-1,0), byrow = TRUE, ncol = 1)
beta[[3]] <- matrix(c(2,0,-2), byrow = TRUE, ncol = 1)
beta[[4]] <- matrix(c(-1,3,1), byrow = TRUE, ncol = 1)

# Calculate logvar to use on the simulating function:
logvar <- var_to_logvar(gen = list(m = m, n_dep = n_dep),
                        emiss_mu = emiss_distr,
                        var_emiss =  list(rep(4, m),
                                          rep(2, m),
                                          rep(5, m)),
                        byrow = FALSE)

# Put logvar in the right format:
logvar <- lapply(logvar, function(q) matrix(q, nrow = m))

# Simulate count data:
data_count <- sim_mHMM(n_t = n_t,
                       n = n_subj,
                       data_distr = "count",
                       gen = list(m = m, n_dep = n_dep),
                       gamma = gamma,
                       emiss_distr = emiss_distr_log,
                       xx_vec = xx_vec,
                       beta = beta,
                       var_gamma = 0.1,
                       var_emiss = logvar,
                       return_ind_par = TRUE,
                       log_scale = TRUE)

head(data_count$states)
head(data_count$obs)
```

`var_to_logvar` *Transform the between-subject variance in the positive scale to the logvariance in the logarithmic scale*

Description

`var_to_logvar` returns the desired between-subject logvariance in the logarithmic scale corresponding to the between-subject variance in the (real) positive scale specified as input. The log-

variance is used as the dispersion parameter in the lognormal distribution adopted as prior for the subject-specific Poisson means in the implementation of mHMM and sim_mHMM of mHMMbayes for count distributed data. It takes as inputs the group-level Poisson means emiss_mu and the desired between-subject variance var_emiss, both in the (real) positive scale.

Usage

```
var_to_logvar(gen, emiss_mu, var_emiss, byrow = TRUE)
```

Arguments

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
emiss_mu	A list containing n_dep matrices, i.e., one list for each dependent variable k. Each matrix contains the group-level means of the lognormal emission distributions in each of the states in the natural (positive real numbers) scale. Hence, each matrix consists of either 1 row (when not including covariates in the model) and m columns (see emiss_mu0 in prior_emiss_count), or m rows and 1 column (when not including covariates in the model; see emiss_distr in sim_mHMM), denoting the mean of state *i* (column or row *i*) of the lognormal distribution used as prior for the Poisson emissions. By default it is assumed that the matrices contain 1 row and m columns, as specified for emiss_mu0 in prior_emiss_count (see argument byrow). If emiss_mu were to be specified using m rows and 1 column as for emiss_distr in sim_mHMM, then set the argument byrow = FALSE. Note: in every case the means should be specified in the natural (real positive) scale.
var_emiss	A list containing n_dep elements corresponding to each of the dependent variables k, where each element k is a vector with length m denoting the amount of variance between the subject (emission distribution) means in the natural (positive real numbers) scale. It follows a similar specification as emiss_V in prior_emiss_count.
byrow	A logical scalar indicating whether the emission means are specified in the first row (byrow = TRUE) or the first column (byrow = FALSE) of the n_dep matrices listed in emiss_mu. Use byrow = TRUE if emiss_mu is entered as emiss_mu0 in prior_emiss_count, and byrow = FALSE if emiss_mu is entered as emiss_distr in sim_mHMM. By default, byrow = TRUE.

Value

var_to_logvar Returns a list of n_dep numeric vectors of m elements denoting the state i-specific logvariance (between-subject variance in the logarithmic scale) for the k-dependent variable used as dispersion parameter in the lognormal prior for the Poisson emission distribution.

See Also

mHMM for fitting the multilevel hidden Markov model, and sim_mHMM for simulating data from a multilevel hidden Markov model.

Examples

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


## Example: specifying count priors manually using both positive and log scale:

# Define general parameters:
m       <- 3        # Number of hidden states
n_dep   <- 2        # Number of dependent variables

# Specify priors manually on the positive scale for emuss_mu0 and emiss_V:
manual_prior_emiss_1 <- prior_emiss_count(
  gen = list(m = m, n_dep = n_dep),
  emiss_mu0 = list(matrix(c(30, 70, 170), nrow = 1),
                   matrix(c(7, 8, 18), nrow = 1)),
  emiss_K0 = list(1, 1),
  emiss_V =  list(c(16,25,32),
                  rep(4, m)),
  emiss_nu = list(0.1, 0.1),
  log_scale = FALSE)

# Define logmu and logvar:
logmu <-  list(matrix(log(c(30, 70, 170)), nrow = 1),
               matrix(log(c(7, 8, 18)), nrow = 1))

logvar <- var_to_logvar(gen = list(m = m, n_dep = n_dep),
                        emiss_mu = list(matrix(c(30, 70, 170), nrow = 1),
                                        matrix(c(7, 8, 18), nrow = 1)),
                        var_emiss = list(c(16,25,32),
                                         rep(4, m)),
                        byrow = TRUE)

# Specify priors manually on the log scale for emiss_mu0 and emiss_V:
manual_prior_emiss_2 <- prior_emiss_count(
  gen = list(m = m, n_dep = n_dep),
  emiss_mu0 = logmu,
  emiss_K0 = list(1, 1),
  emiss_V =  logvar,
  emiss_nu = list(0.1, 0.1),
  log_scale = TRUE)

# Check whether they are identical:
identical(manual_prior_emiss_1, manual_prior_emiss_2)
```

`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. Optionally, the state probabilities themselves at each point in time can also be obtained.

Usage

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

Arguments

object	An object of class mHMM, generated by the function mHMM.
s_data	A matrix containing the observations to be modeled, 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 in case of categorical dependent variable(s), input are integers (i.e., whole numbers) that range from 1 to q_emiss (see below) and is numeric (i.e., not 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.
return_state_prob	A logical scaler. Should the function, in addition to the most likely state sequence for each subject, also return the state probabilities at each point in time for each subject (return_state_prob = TRUE) or not (return_state_prob = FALSE). Defaults to return_state_prob = FALSE.

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. 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 second column contains the most likely state at each point in time. If requested, the subsequent columns contain the state probabilities at each point in time for each subject.

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 categorical data, see ?nonverbal
# First fit the multilevel HMM on the 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

# 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
states1 <- vit_mHMM(s_data = nonverbal, object = out_2st)


###### Example on simulated multivariate continuous data
# simulating multivariate continuous data
n_t     <- 100
n       <- 10
m       <- 3
n_dep   <- 2
```

```
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 <- list(matrix(c( 50, 10,
                              100, 10,
                              150, 10), nrow = m, byrow = TRUE),
                    matrix(c(5, 2,
                             10, 5,
                             20, 3), nrow = m, byrow = TRUE))

data_cont <- sim_mHMM(n_t = n_t, n = n, data_distr = 'continuous',
                      gen = list(m = m, n_dep = n_dep), gamma = gamma,
                      emiss_distr = emiss_distr, var_gamma = .1,
                      var_emiss = c(5^2, 0.2^2))

# Specify hyper-prior for the continuous emission distribution
manual_prior_emiss <- prior_emiss_cont(
                        gen = list(m = m, n_dep = n_dep),
                        emiss_mu0 = list(matrix(c(30, 70, 170), nrow = 1),
                                         matrix(c(7, 8, 18), nrow = 1)),
                        emiss_K0 = list(1, 1),
                        emiss_V =  list(rep(5^2, m), rep(0.5^2, m)),
                        emiss_nu = list(1, 1),
                        emiss_a0 = list(rep(1.5, m), rep(1, m)),
                        emiss_b0 = list(rep(20, m), rep(4, m)))

# Run 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_3st_cont_sim <- mHMM(s_data = data_cont$obs,
                    data_distr = "continuous",
                    gen = list(m = m, n_dep = n_dep),
                    start_val = c(list(gamma), emiss_distr),
                    emiss_hyp_prior = manual_prior_emiss,
                    mcmc = list(J = 11, burn_in = 5))

###### obtain the most likely state sequence with the Viterbi algorithm,
# including state probabilities at each time point.
states2 <- vit_mHMM(s_data = data_cont$obs, object = out_3st_cont_sim,
                    return_state_prob = TRUE)
```

Index