Package ‘ActiveLearning4SPM’

October 7, 2025

Type Package

Title Active Learning for Process Monitoring

Version 0.1.0

Description Implements the methodology introduced in Capezza, Lepore, and Paynabar (2025) <doi:10.1080/00401706.2025.2561744> for process monitoring with limited labeling resources. The package provides functions to (i) simulate data streams with true latent states and multivariate Gaussian observations as done in the paper, (ii) fit partially hidden Markov models (pHMMs)
using a constrained Baum-Welch algorithm with partial labels, and (iii) perform stream-based active learning that balances exploration and exploitation to decide whether to request labels in real time. The methodology is particularly suited for statistical process monitoring in industrial applications where labeling is costly.

Depends R (>= 4.2)

Imports Rcpp, Rfast, mvnfast, rrcov, caTools, abind, pROC, stats

LinkingTo Rcpp, RcppArmadillo

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Suggests covr, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Christian Capezza [aut, cre],
Antonio Lepore [aut],
Kamran Paynabar [aut]

Maintainer Christian Capezza <christian.capezza@unina.it>

Repository CRAN

Date/Publication 2025-10-07 18:30:21 UTC

Contents

`active_learning_pHMM` *Stream-Based Active Learning with a Partially Hidden Markov Model (pHMM)*

Description

Implements the stream-based active learning strategy of Capezza, Lepore, and Paynabar (2025) for process monitoring with partially observed states. At each time step, the method fits a pHMM to the available data, and balances between *exploitation* (reducing predictive uncertainty) and *exploration* (detecting potential out-of-control shifts) to decide whether to request the true label of the current observation. Labeling requests are constrained by a user-defined budget.

Usage

```
active_learning_pHMM(
  y,
  true_x,
  T0,
  B = 0.1,
  weight_exploration = 0.5,
  lambda_MEWMA = 0.3,
  verbose = FALSE
)
```

Arguments

`y`	A numeric matrix of dimension $T \times d$, where each row corresponds to a d-dimensional observation at time t.
`true_x`	Integer vector of true states of length `nrow(y)`, used to assess model predictions. The first `T0` values, assumed to be from an in-control process, must be 1.
`T0`	Integer. Number of initial observations assumed to be labeled as in-control (state 1).
`B`	Numeric between 0 and 1. Labeling budget, expressed as the maximum fraction of observations (after the first `T0`) for which labels may be acquired. Default is `0.1`.
`weight_exploration`	Numeric between 0 and 1. Weight assigned to the exploration criterion. The exploitation weight is computed as `1 - weight_exploration`. Default is `0.5`.

`lambda_MEWMA`	Numeric in (0,1). Smoothing parameter for the MEWMA statistic used in the exploration criterion. Default is `0.3`.
`verbose`	Logical. If `TRUE`, prints the current time index as the algorithm progresses. Default is `FALSE`.

Details

The exploitation criterion is based on the entropy of the state sequence, while the exploration criterion uses a multivariate exponentially weighted moving average (MEWMA) statistic. The two criteria are combined with a user-defined weighting, and labeling stops when the budget is exhausted or at the end of the data stream.

Value

A list with components:

- `decision`: character vector indicating the action taken at each time (`"label_exploitation"`, `"label_exploration"`, or predicted state).
- `xlabeled`: updated state sequence including acquired labels.
- `xhat`: final predicted state sequence.
- `scores`: classification performance metrics (accuracy, precision, recall, F1, AUC) computed against the true states.

Note

This function is intended for simulation studies where the entire observation sequence `y` and the corresponding true states `x` are available in advance. The function uses these to evaluate the active learning strategy under a given budgets. In real-time applications, data and labels would arrive sequentially, and labels would only be obtained if requested by the strategy.

References

Capezza, C., Lepore, A., & Paynabar, K. (2025). Stream-Based Active Learning for Process Monitoring. *Technometrics*. <doi:10.1080/00401706.2025.2561744>.

Examples

```
library(ActiveLearning4SPM)
set.seed(123)
dat <- simulate_stream(T0 = 50, TT = 100, T_min_IC = 20, T_max_IC = 30)
out <- active_learning_pHMM(y = dat$y,
                            true_x = dat$x,
                            T0 = 50,
                            B = 0.1)
table(out$decision)
out$scores$f1
```

fit_pHMM — *Fit a Partially Hidden Markov Model (pHMM)*

Description

Fits a partially hidden Markov model (pHMM) to multivariate time series observations y with partially observed process states x, using a constrained Baum-Welch algorithm. The function allows the user to provide custom initial parameters, and supports constraints on known means and/or covariances, as well as equal or diagonal covariance structures.

Usage

```
fit_pHMM(
  y,
  xlabeled,
  nstates,
  ppi_start = NULL,
  A_start = NULL,
  mean_start,
  covariance_start = NULL,
  known_mean = NULL,
  known_covariance = NULL,
  equal_covariance = FALSE,
  covariance_structure = "full",
  max_iter = 200,
  tol = 0.001,
  verbose = FALSE
)
```

Arguments

`y`	A numeric matrix of dimension $T \times d$, where each row corresponds to a d-dimensional observation at time t.
`xlabeled`	An integer vector of length T with partially observed states. Known states must be integers in $1, \dots, N$; unknown states should be coded as `NA`.
`nstates`	Integer. The total number of hidden states to fit.
`ppi_start`	Numeric vector of length `nstates` giving the initial state distribution. If `NULL`, defaults to `c(1,0,...,0)`.
`A_start`	Numeric `nstates` $\times$ `nstates` transition probability matrix. If `NULL`, defaults to a transition matrix with diagonal entries equal to `1-0.01*(nstates-1)` and all off-diagonal entries equal to `0.01`.
`mean_start`	List of length `nstates` containing numeric mean vectors for the emission distributions.

covariance_start	List of covariance matrices for the emission distributions. Must be of length nstates, unless equal_covariance = TRUE, in which case it must be of length 1. If NULL, defaults to identity matrices.
known_mean	Optional list of known mean vectors. Use NA for unknown elements.
known_covariance	Optional list of known covariance matrices. Use NA for unknown elements.
equal_covariance	Logical. If TRUE, all states are constrained to share a common covariance matrix.
covariance_structure	Character string specifying the covariance structure. Either "full" (default) or "diagonal".
max_iter	Maximum number of EM iterations. Default is 200.
tol	Convergence tolerance for log-likelihood and parameter change. Default is 1e-3.
verbose	Logical. If TRUE, prints log-likelihood progress at each iteration.

Value

A list with components:

- y, xlabeled: the input data.
- log_lik, log_lik_vec: final and trace of log-likelihood.
- iter: number of EM iterations performed.
- logB, log_alpha, log_beta, log_gamma, log_xi: posterior quantities from the Baum-Welch algorithm.
- logAhat, mean_hat, covariance_hat, log_pi_hat: estimated model parameters.
- AIC, BIC: information criteria for model selection.

References

Capezza, C., Lepore, A., & Paynabar, K. (2025). Stream-Based Active Learning for Process Monitoring. *Technometrics*. <doi:10.1080/00401706.2025.2561744>.

Examples

```
library(ActiveLearning4SPM)
set.seed(123)
dat <- simulate_stream(T0 = 100, TT = 500)
y <- dat$y
xlabeled <- dat$x
d <- ncol(dat$y)
xlabeled[sample(1:600, 300)] <- NA
out <- fit_pHMM(y = y,
                xlabeled = xlabeled,
                nstates = 3,
                mean_start = list(rep(0, d), rep(1, d), rep(-1, d)),
                  equal_covariance = TRUE)
out$AIC
```

fit_pHMM_auto *Automatic Initialization and Fitting of a Partially Hidden Markov Model (pHMM)*

Description

Fits a partially hidden Markov model (pHMM) to multivariate time series observations y with partially observed states x, using the constrained Baum-Welch algorithm. Unlike `fit_pHMM`, this function does not require user-specified initial parameters. Instead, it implements a customized initialization strategy designed for process monitoring with highly imbalanced classes, as described in the supplementary material of Capezza, Lepore, and Paynabar (2025).

Usage

```
fit_pHMM_auto(
  y = y,
  xlabeled = xlabeled,
  tol = 0.001,
  max_nstates = 5,
  ntry = 10
)
```

Arguments

`y`	A numeric matrix of dimension $T \times d$, where each row corresponds to a d-dimensional observation at time t.
`xlabeled`	An integer vector of length T with partially observed states. Known states must be integers in $1, \ldots, N$; unknown states should be coded as `NA`.
`tol`	Convergence tolerance for log-likelihood and parameter change. Default is `1e-3`.
`max_nstates`	Maximum number of hidden states to consider during the initialization procedure. Default is `5`.
`ntry`	Number of candidate initializations for each new state. Default is `10`.

Details

The initialization procedure addresses the multimodality of the likelihood and the sensitivity of the Baum-Welch algorithm to starting values:

1. A one-state model (in-control process) is first fitted using robust estimators of location and scatter.

2. To introduce an additional state, candidate mean vectors are selected from observations that are least well represented by the current model. This is achieved by computing moving averages of the data over window lengths $k = 1, \ldots, 9$, and then calculating the Mahalanobis distances of these smoothed points to existing state means.
3. The `ntry` observations with the largest minimum distances are retained as candidate initializations for the new state's mean.
4. For each candidate, a pHMM is initialized with:
   - Existing means fixed to their previous estimates.
   - The new state's mean set to the candidate vector.
   - A shared covariance matrix fixed to the robust estimate from the in-control state.
   - Initial state distribution π concentrated on the IC state.
   - Transition matrix with diagonal entries $1 - 0.01(N - 1)$ and off-diagonal entries 0.01.
5. Each initialized model is fitted with the Baum-Welch algorithm, and the one achieving the highest log-likelihood is retained.
6. This process is repeated until up to `max_nstates` states are considered.

This strategy leverages prior process knowledge (dominant in-control regime) and focuses the search on under-represented regions of the data space, which improves convergence and reduces sensitivity to random initialization.

Value

A list with the same structure as returned by `fit_pHMM`:

- `y`, `xlabeled`: the input data.
- `log_lik`, `log_lik_vec`: final and trace of log-likelihood.
- `iter`: number of EM iterations performed.
- `logB`, `log_alpha`, `log_beta`, `log_gamma`, `log_xi`: posterior quantities from the Baum-Welch algorithm.
- `logAhat`, `mean_hat`, `covariance_hat`, `log_pi_hat`: estimated model parameters.
- `AIC`, `BIC`: information criteria for model selection.

References

Capezza, C., Lepore, A., & Paynabar, K. (2025). Stream-Based Active Learning for Process Monitoring. *Technometrics*. <doi:10.1080/00401706.2025.2561744>.

Supplementary Material, Section B: Initialization of the Partially Hidden Markov Model. Available at <https://doi.org/10.1080/00401706.2025.2561744>.

Examples

```
library(ActiveLearning4SPM)
set.seed(123)
dat <- simulate_stream(T0 = 100, TT = 500)
y <- dat$y
xlabeled <- dat$x
```

```
d <- ncol(dat$y)
xlabeled[sample(1:600, 300)] <- NA
obj <- fit_pHMM_auto(y = y,
                     xlabeled = xlabeled,
                     tol = 1e-3,
                     max_nstates = 5,
                     ntry = 10)
obj$AIC
```

simulate_stream *Simulate Process Monitoring Data for Stream-Based Active Learning*

Description

Generate a sequence of latent states and corresponding multivariate Gaussian observations for process monitoring. The process has three possible states:

1. state 1: in-control (IC),
2. state 2: out-of-control (OC),
3. state 3: out-of-control (OC).

Usage

```
simulate_stream(
  d = 10,
  TT = 500,
  T0 = 100,
  T_min_IC = 60,
  T_max_IC = 85,
  T_OC = 5,
  mean = NULL,
  covariance = NULL
)
```

Arguments

`d`	Integer. Number of variables (dimension of the multivariate observations). Default is 10.
`TT`	Integer. Length of the sequence after the initial IC portion. Default is 500.
`T0`	Integer. Length of the initial IC sequence known to belong to state 1. Default is 100.
`T_min_IC`, `T_max_IC`	Integers. Minimum and maximum length of consecutive IC observations before switching to an OC state.
`T_OC`	Integer. Fixed length of each OC state sequence.

`mean`	List of three numeric vectors of length `d`, representing the mean vectors of states 1 (IC), 2 (OC), and 3 (OC). If `NULL` (default), simple default values are used.
`covariance`	List of three `d x d` covariance matrices, one for each state. If `NULL` (default), pre-defined (equal) covariance matrices are used.

Details

The first `T0` observations are fixed in state 1 (IC). Then, in the following `TT` observations, only state 2 appears in the first half, and only state 3 appears in the second half. Within each half, runs of state 1 (IC) of random length between `T_min_IC` and `T_max_IC` alternate with fixed-length runs of the corresponding OC state of length `T_OC`.

Value

A list with elements:

`x`	Integer vector of latent states of length `T0 + TT`.
`y`	Matrix of simulated multivariate observations with `T0 + TT` rows and `d` columns.

Examples

```
library(ActiveLearning4SPM)
sim <- simulate_stream()
table(sim$x)
```

Index