Package 'markovmix'

December 2, 2023

Title Mixture of Markov Chains with Support of Higher Orders and Multiple Sequences

Version 0.1.3

Maintainer Xiurui Zhu <zxr6@163.com>

Description Fit mixture of Markov chains of higher orders from multiple sequences. It is also compatible with ordinary 1-component, 1-order or single-sequence Markov chains. Various utility functions are provided to derive transition patterns, transition probabilities per component and component priors. In addition, print(), predict() and component extracting/replacing methods are also defined as a convention of mixture models.

License MIT + file LICENSE

Depends R (>= 3.5.0)

Imports dplyr (>= 1.0.8), forcats (>= 1.0.0), pillar (>= 1.9.0), purrr (>= 0.3.4), Rcpp, rlang (>= 1.1.0), tibble (>= 3.1.6), tidyr (>= 1.2.0)

Suggests bench (>= 1.1.2), covr, ggplot2 (>= 3.4.2), testthat (>= 3.1.7)

LinkingTo Rcpp

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Collate 'RcppExports.R' 'data.R' 'utils.R' 'markov_mix_utils.R' 'markov_mix.R' 'markovmix-package.R'

URL https://github.com/zhuxr11/markovmix

BugReports https://github.com/zhuxr11/markovmix/issues

NeedsCompilation yes

Author Xiurui Zhu [aut, cre]

Repository CRAN

Date/Publication 2023-12-02 11:00:05 UTC

R topics documented:

`markovmix-package` *markovmix package*

Description

`markovmix` package Fit mixture of Markov chains of higher orders from multiple sequences. It is also compatible with ordinary 1-component, 1-order or single-sequence Markov chains. Various utility functions are provided to derive transition patterns, transition probabilities per component and component priors. In addition, `print()`, `predict()` and `component extracting/replacing` methods are also defined as a convention of mixture models.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the package.
- 0.1.2 Xiurui Zhu - Update package documentation.

Author(s)

Xiurui Zhu

`Extract.MarkovMix` *Extract or replace components of MarkovMix object*

Description

Operators to extract or replace components of a `MarkovMix` object.

Usage

```
## S3 method for class 'MarkovMix'
x[i]

## S3 replacement method for class 'MarkovMix'
x[i] <- value
```

Arguments

`x`	`MarkovMix` object.
`i`	Indices specifying components to extract or replace.
`value`	Numeric matrix as soft counts for transition patterns (like `get_counts(object = x)`), whose rows correspond to the rows in `get_states_mat(x)` and columns correspond to the number of components to replace.

Note

Change log:

- 0.1.1 Xiurui Zhu - Initiate the functions.
- 0.1.2 Xiurui Zhu - Update function documentation.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: `get_counts()`, `get_order()`, `get_prior()`, `get_prob()`, `get_states_mat()`, `get_states()`, `restate()`

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
```

```
get_order(object = markov_mix_ex)

# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)

# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

`fit_markov_mix` *Fit mixture of Markov chains*

Description

`fit_markov_mix` fits mixture of Markov chains. It supports high-order Markov chains, multiple sequences and mixture components with cluster probabilities.

Usage

```
fit_markov_mix(
  seq_list,
  order. = 1L,
  states = NULL,
  clusters = NULL,
  verbose = TRUE
)
```

Arguments

seq_list	Sequence list containing vectors of the same class.
order.	Integer (1L) indicating the order of the Markov chain.
states	NULL or vector indicating the states in the Markov chain. If NULL, states are inferred from unique non-NA elements in all the sequences. If vector, it should match the class of the sequences. NA elements in the vector are removed.
clusters	NULL or matrix containing clustering probabilities. If NULL, Markov chain is fit without mixture components. If matrix, rows are probabilities of sequences and columns are components. As probabilities of sequences, rows are normalized to sum up to 1.
verbose	Logical (1L) indicating whether additional messages should be printed.

Value

An object of class MarkovMix.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

Examples

```
# Generate a list of integer sequences of different lengths with 4 states
test_states <- seq_len(4L)
test_maxlen <- 10L
set.seed(1111L)
test_seq <- purrr::map(
  seq_len(100L),
  ~ sample(test_states, sample.int(test_maxlen, 1L), replace = TRUE)
)

# Fit a 1-order Markov chain
markov_fit <- fit_markov_mix(
  seq_list = test_seq,
  order. = 1L,
  states = test_states
)
print(markov_fit)

# Fit a mixture of 2-order Markov chain with 3 components
test_n_comp <- 3L
test_clusters <- matrix(
  runif(length(test_seq) * test_n_comp),
  nrow = length(test_seq),
```

```
  ncol = test_n_comp
)
markov_mix_fit <- fit_markov_mix(
  seq_list = test_seq,
  order. = 2L,
  states = test_states,
  clusters = test_clusters
)
print(markov_mix_fit)
```

`get_counts` *Get transition pattern counts from MarkovFit object*

Description

`get_counts` gets transition pattern counts from `MarkovMix` object.

Usage

```
get_counts(object, check = TRUE)
```

Arguments

`object`	`MarkovMix` object.
`check`	Logical (1L) indicating whether to check `object` at the beginning.

Value

A numeric matrix indicates transition pattern (soft) counts, where each row corresponds to a transition pattern and each column corresponds to a component.

Note

Change log:

- 0.1.2 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: `Extract.MarkovMix`, `get_order()`, `get_prior()`, `get_prob()`, `get_states_mat()`, `get_states()`, `restate()`

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
get_order(object = markov_mix_ex)

# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)

# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

---

get_order *Get the order of Markov chains*

Description

get_order gets the order of Markov chains from MarkovMix object.

Usage

```
get_order(object, check = TRUE)
```

Arguments

object	MarkovMix object.
check	Logical (1L) indicating whether to check object at the beginning.

Value

An integer as the order of Markov chains.

Note

Change log:

- 0.1.2 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: Extract.MarkovMix, get_counts(), get_prior(), get_prob(), get_states_mat(), get_states(), restate()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
get_order(object = markov_mix_ex)

# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)

# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
```

```
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

`get_prior` *Get component priors from MarkovFit object*

Description

`get_prior` gets component priors from `MarkovMix` object, normalized to sum up to 1.

Usage

```
get_prior(object, check = TRUE)
```

Arguments

`object`	`MarkovMix` object.
`check`	Logical (1L) indicating whether to check `object` at the beginning.

Value

A numeric vector indicates component priors.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: Extract.MarkovMix, get_counts(), get_order(), get_prob(), get_states_mat(), get_states(), restate()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
get_order(object = markov_mix_ex)

# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)

# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

---

get_prob *Get probability matrix from MarkovFit object*

Description

get_prob gets probability matrix from MarkovMix object. It normalizes each column in the count matrix to sum up to 1.

Usage

```
get_prob(object, check = TRUE)
```

Arguments

object	MarkovMix object.
check	Logical (1L) indicating whether to check object at the beginning.

Value

A numeric matrix indicating probabilities of each state transition pattern in each component.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: Extract.MarkovMix, get_counts(), get_order(), get_prior(), get_states_mat(), get_states(), restate()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
get_order(object = markov_mix_ex)

# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)
```

```
# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

`get_states` *Get the states of Markov chains*

Description

`get_states` gets the states of Markov chains from `MarkovMix` object.

Usage

```
get_states(object, check = TRUE)
```

Arguments

`object`	`MarkovMix` object.
`check`	Logical (1L) indicating whether to check `object` at the beginning.

Value

A vector as the states used in Markov chains.

Note

Change log:

- 0.1.2 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: Extract.MarkovMix, get_counts(), get_order(), get_prior(), get_prob(), get_states_mat(), restate()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
get_order(object = markov_mix_ex)

# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)

# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

get_states_mat *Get state transition patterns from MarkovFit object*

Description

get_states_mat gets state transition patterns from MarkovMix object. The number of columns is the order of the (mixture of) Markov chain(s) plus 1 (the destination state). Each column is arranged in the ascending order of the states. The last column serves as the destination state and iterates the fastest.

Usage

```
get_states_mat(object, check = TRUE)
```

Arguments

object	MarkovMix object.
check	Logical (1L) indicating whether to check object at the beginning.

Value

A matrix indicating the state transition patterns.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: Extract.MarkovMix, get_counts(), get_order(), get_prior(), get_prob(), get_states(), restate()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Derive transition pattern soft counts
get_counts(object = markov_mix_ex)

# Derive the order of Markov chains
get_order(object = markov_mix_ex)
```

```
# Derive the states of Markov chains
get_states(object = markov_mix_ex)

# Derive state transition patterns
get_states_mat(markov_mix_ex)

# Derive probability matrices
get_prob(markov_mix_ex)

# Derive component priors
get_prior(markov_mix_ex)

# Combine state transition patterns and their probabilities
cbind(
  as.data.frame(get_states_mat(markov_mix_ex)),
  as.data.frame(get_prob(markov_mix_ex))
)

# Extract 1 or more components
markov_mix_ex[2L]
markov_mix_ex[c(1L, 3L)]

# Replace 1 or more components
nrow_value <- length(get_states(object = markov_mix_ex, check = FALSE))^
  (get_order(object = markov_mix_ex, check = FALSE) + 1L)
markov_mix_ex2 <- markov_mix_ex
markov_mix_ex2[2L] <- runif(nrow_value)
print(markov_mix_ex2)
markov_mix_ex3 <- markov_mix_ex
markov_mix_ex3[c(1L, 3L)] <- matrix(runif(nrow_value * 2L), ncol = 2L)
print(markov_mix_ex3)
```

`MarkovMix-class` *MarkovMix class*

Description

An object of class `MarkovMix` is a list containing the following components:

counts Numeric matrix containing soft counts of sub-sequence patterns in each component. For (non-mixture) Markov chains, the matrix contains only 1 column and counts are actually integers, but they are still stored as numeric values.

order Integer (1L) as the order of (mixture) Markov chain(s).

states Vector as the states in the (mixture) Markov chain(s).

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the class.

Author(s)

Xiurui Zhu

`markov_mix_ex` *Mixture of Markov chain example*

Description

A mixture of 2-order Markov chain fit from 100 random sequences with 4 states (A, B, C, D) and 3 components.

Usage

```
markov_mix_ex
```

Format

A `MarkovMix` object.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the dataset.

Author(s)

Xiurui Zhu

`predict.MarkovMix` *Predict probabilities with MarkovMix object and new sequence list*

Description

`predict.MarkovMix` predicts probabilities with `MarkovMix` object and new sequence list. NA values are returned for sequences with no valid sub-sequences to distinguish them from those that are truly not observed (probabilities = 0) in the transition matrices.

Usage

```
## S3 method for class 'MarkovMix'
predict(object, newdata, aggregate. = TRUE, ...)
```

Arguments

object	MarkovMix object.
newdata	Sequence list containing vectors of the same class.
aggregate.	Logical (1L) indicating whether probabilities from each component should be weighted mean by component priors (TRUE) or not (FALSE).
...	Currently ignored for this method.

Value

For aggregate. = TRUE, a numeric vector of probabilities. For aggregate. = TRUE, a numeric matrix of probabilities from each component.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix methods: print.MarkovMix()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Generate a new list of sequences
set.seed(2222L)
new_maxlen <- 8L
new_seq_list <- purrr::map(
  seq_len(50L),
  ~ sample(get_states(object = markov_mix_ex, check = FALSE),
           sample.int(new_maxlen, 1L),
           replace = TRUE)
)

# Predict the probabilities of sequences
predict(markov_mix_ex, newdata = new_seq_list)

# Predict the probabilities of sequences from each component
predict(markov_mix_ex, newdata = new_seq_list, aggregate. = FALSE)
```

`print.MarkovMix` *Print MarkovMix object*

Description

`print.MarkovMix` prints `MarkovMix` object in a user-friendly form, including component priors and transition matrices.

Usage

```
## S3 method for class 'MarkovMix'
print(x, sep = "->", print_max = 20L, print_min = 10L, ...)
```

Arguments

`x`	`MarkovMix` object.
`sep`	Character (1L) used as separator between states in the row names of transition matrix.
`print_max, print_min`	Integers as the numbers of rows to print each transition matrix. See `pillar_options` for details.
`...`	Currently ignored for this method.

Value

Input `x`, invisibly.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix methods: `predict.MarkovMix()`

Examples

```
# Generate a list of integer sequences of different lengths with 4 states
test_states <- seq_len(4L)
test_maxlen <- 10L
set.seed(1111L)
test_seq <- purrr::map(
  seq_len(100L),
  ~ sample(test_states, sample.int(test_maxlen, 1L), replace = TRUE)
)

# Fit a 1-order Markov chain
markov_fit <- fit_markov_mix(
  seq_list = test_seq,
  order. = 1L,
  states = test_states
)
print(markov_fit)

# Fit a mixture of 2-order Markov chain with 3 components
test_n_comp <- 3L
test_clusters <- matrix(
  runif(length(test_seq) * test_n_comp),
  nrow = length(test_seq),
  ncol = test_n_comp
)
markov_mix_fit <- fit_markov_mix(
  seq_list = test_seq,
  order. = 2L,
  states = test_states,
  clusters = test_clusters
)
print(markov_mix_fit)
```

`restate` *Reorganize states in MarkovMix object*

Description

`restate` reorganizes states in `MarkovMix` object with a function.

Usage

```
restate(.object, .fun, .check = TRUE, ...)
```

Arguments

`.object`	`MarkovMix` object.
`.fun`	Function to process each column in state transition patterns as factors, such as those in `forcats` package.

.check	Logical (1L) indicating whether to check object at the beginning.
...	Additional arguments passed on to .fun.

Value

A MarkovMix object with modified states and count matrix.

Note

Change log:

- 0.1.0 Xiurui Zhu - Initiate the function.

Author(s)

Xiurui Zhu

See Also

Other MarkovMix utilities: Extract.MarkovMix, get_counts(), get_order(), get_prior(), get_prob(), get_states_mat(), get_states()

Examples

```
# Load example MarkovMix object
data("markov_mix_ex")

# Reverse states (using function)
markov_mix_new1 <- restate(
  .object = markov_mix_ex,
  .fun = forcats::fct_rev
)
print(markov_mix_new1)

# Reorder states by hand (using function name with additional arguments)
markov_mix_new2 <- restate(
  .object = markov_mix_ex,
  .fun = "levels<-",
  value = c("B", "D", "C", "A")
)
print(markov_mix_new2)

# Merge state D into C (using purrr-style lambda function)
markov_mix_new3 <- restate(
  .object = markov_mix_ex,
  .fun = ~ forcats::fct_recode(.x, "C" = "D")
)
print(markov_mix_new3)
```

Index