

Package: dyadicMarkov (via r-universe)

September 21, 2026

Type Package

Title Pattern Estimation and Identification for Dyadic Sequences Using
Transition Matrices in R

Version 0.1.3.9000

Description Provides methods for analyzing categorical dyadic
sequences using transition matrices within the Longitudinal
Actor-Partner Interdependence Model and Markov-chain framework.
The package supports empirical transition counts, maximum
likelihood estimation of transition probabilities, and
identification of univariate and bivariate patterns of
interaction in dyadic sequences.

License MIT + file LICENSE

URL <https://github.com/BoellenruecherM/dyadicMarkov-public>,
<https://boellenruecherM.github.io/dyadicMarkov-public/>

BugReports <https://github.com/BoellenruecherM/dyadicMarkov-public/issues>

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports stats

Suggests testthat (>= 3.0.0), knitr, rmarkdown, srr

VignetteBuilder knitr

Roxygen list(markdown = TRUE, roclets = c("`namespace", "`rd",
"`srr::srr_stats_roclet"))

Config/testthat/edition 3

LazyData true

Config/roxygen2/version 8.0.0

Repository <https://ropensci-staging.r-universe.dev>

Date/Publication 2026-09-21 06:56:38 UTC

RemoteUrl <https://github.com/BoellenruecherM/dyadicMarkov-public>

RemoteRef ropensci-review-refinements

RemoteSha 96bd1d421b7b3e1ec4ac079c1b8feb04c18b1d16

Contents

bivariateCase	2
completePattern	3
countEmp	4
countEmpBivariate	5
dyadic_bivariate_example	6
dyadic_univariate_example	7
mleEstimation	7
partialPattern	8
plot.dyadic_pattern	9
sensitivity_simulation_data	10
univariatePattern	11
Index	12

bivariateCase	<i>Bivariate case identification for dyadic Markov chains</i>
---------------	---

Description

Identifies the bivariate case as "trivial", "univariate", "partial", or "complete" using two chi-squared tests against constrained binary bivariate structures.

Usage

```
bivariateCase(empirical, alpha = 0.05)
```

Arguments

empirical	A numeric matrix with 16 rows and 2 columns containing finite non-negative counts, as returned by countEmpBivariate . Integer and double storage modes are both accepted.
alpha	A single number in (0, 1) giving the significance level. Default is 0.05.

Details

The returned case corresponds to the global approach of the binary bivariate method. The scientific method treats the global comparisons as nested-model comparisons within a likelihood-ratio test (LRT) framework. `bivariateCase()` implements the two chi-squared tests for the A1 and B1 comparisons. `dyadicMarkov` evaluates these tests using Pearson's chi-squared statistic,

$$X^2 = \sum_{ij} (O_{ij} - E_{ij})^2 / E_{ij},$$

where O and E are the empirical and theoretical transition counts. A global comparison is treated as rejected when $p \leq \alpha$. The result determines whether the sequence analyzed is treated as trivial, univariate, partial bivariate, or complete bivariate before the separate local AIC-based identification of the pattern of interaction.

Value

A list with class `c("dyadic_case", "list")` containing components `testUnivariate`, `testPartial`, `case`, and `metadata` fields `alpha` and `call`. It remains usable as an ordinary list. When `empirical` was created by `countEmpBivariate()`, its stored sequence metadata is propagated so that `plot()` can display the four observed state strips.

Examples

```
chainFM_V1 <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM_V1 <- c(2L, 1L, 2L, 1L, 1L, 2L)
chainFM_V2 <- c(1L, 1L, 2L, 2L, 1L, 2L)
chainSM_V2 <- c(2L, 2L, 1L, 1L, 2L, 1L)
emp <- countEmpBivariate(
  chainFM_V1, chainSM_V1, chainFM_V2, chainSM_V2,
  states = 2L
)
bivariateCase(emp, alpha = 0.05)
```

completePattern

*Complete bivariate pattern identification by AIC***Description**

Compares the complete bivariate patterns C, D1–D4, and E1–E4 using AIC and returns the selected pattern.

Usage

```
completePattern(empirical)
```

Arguments

`empirical` A numeric matrix with 16 rows and 2 columns containing finite non-negative counts, as returned by `countEmpBivariate`. Integer and double storage modes are both accepted.

Details

Conditional on the complete bivariate case, the G-squared deviance is computed for each C, D1–D4, and E1–E4 candidate before calculating

$$AIC = G^2 + 2k,$$

where

$$G^2 = 2 \sum_{ij} O_{ij} \log(O_{ij}/E_{ij}).$$

The candidate with the smallest AIC is selected. If candidates have exactly equal AIC values, the first candidate in the documented comparison order is selected.

Value

A list with class `c("dyadic_pattern", "list")` containing components `aic` (a data frame with columns `pattern`, `matrix`, and `aic`), `pattern` (the selected pattern label), and `call`. It remains usable as an ordinary list. When empirical was created by `countEmpBivariate()`, its stored sequence metadata is propagated so that `plot()` can display the four observed state strips.

Examples

```
chainFM_V1 <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM_V1 <- c(2L, 1L, 2L, 1L, 1L, 2L)
chainFM_V2 <- c(1L, 1L, 2L, 2L, 1L, 2L)
chainSM_V2 <- c(2L, 2L, 1L, 1L, 2L, 1L)
emp <- countEmpBivariate(
  chainFM_V1, chainSM_V1, chainFM_V2, chainSM_V2,
  states = 2L
)
completePattern(emp)
```

countEmp

*Empirical transition counts for univariate dyadic sequences***Description**

Computes empirical transition counts for the sequences of the first and second member. Rows correspond to dyadic states of the two members, and columns correspond to the state of the first member at the next time point.

Usage

```
countEmp(chainFM, chainSM, states)
```

Arguments

chainFM	Numeric vector of integer-valued observed states for the first member (FM). Must have the same length as chainSM.
chainSM	Numeric vector of integer-valued observed states for the second member (SM). Must have the same length as chainFM.
states	A single positive integer greater than or equal to 2 giving the number of states.

Details

Rows correspond to current dyadic states (FM_t, SM_t) . For general states, the row index is computed as $1 + \text{states} * (FM_t - 1) + (SM_t - 1)$. Columns correspond to the state of the first member at the next time point, FM_{t+1} .

Value

An integer matrix with class `c("dyadic_counts", "matrix", "array")`, with $states^2$ rows and $states$ columns. It remains usable as an ordinary matrix. `print()`, `summary()`, and `utils::toLatex()` methods are available for this result.

Examples

```
chainFM <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM <- c(2L, 1L, 2L, 1L, 1L, 2L)
countEmp(chainFM, chainSM, states = 2L)
```

countEmpBivariate	<i>Empirical transition counts for bivariate dyadic sequences</i>
-------------------	---

Description

Computes empirical transition counts for bivariate categorical dyadic sequences with two variables. This function currently supports `states = 2` only.

Usage

```
countEmpBivariate(chainFM_V1, chainSM_V1, chainFM_V2, chainSM_V2, states = 2L)
```

Arguments

chainFM_V1	Numeric vector of observed states for variable 1 for the first member. Values must be either 1 or 2, and the vector must have the same length as the other three chain vectors.
chainSM_V1	Numeric vector of observed states for variable 1 for the second member. Values must be either 1 or 2, and the vector must have the same length as the other three chain vectors.
chainFM_V2	Numeric vector of observed states for variable 2 for the first member. Values must be either 1 or 2, and the vector must have the same length as the other three chain vectors.
chainSM_V2	Numeric vector of observed states for variable 2 for the second member. Values must be either 1 or 2, and the vector must have the same length as the other three chain vectors.
states	A single integer equal to 2. Other state counts are not currently supported. Default is 2.

Details

The bivariate counter currently supports `states = 2` only. Rows represent the previous dyadic states of variable 1 and variable 2. Writing K for the number of states, the row index at time t is

$$r_t = 1 + K^2[K(FM_{V1,t} - 1) + (SM_{V1,t} - 1)] + K(FM_{V2,t} - 1) + (SM_{V2,t} - 1)$$

Columns correspond to the state of the first member on variable 1 at the next time point, $FM_{V1,t+1}$.

Value

An integer matrix with class `c("dyadic_counts", "matrix", "array")` with 16 rows and 2 columns when `states = 2`. It remains usable as an ordinary matrix. The four validated input sequences and the binary state count are retained in the `"dyadic_sequences"` attribute so that downstream bivariate results can display the observed state strips with `plot()`.

Examples

```
chainFM_V1 <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM_V1 <- c(2L, 1L, 2L, 1L, 1L, 2L)
chainFM_V2 <- c(1L, 1L, 2L, 2L, 1L, 2L)
chainSM_V2 <- c(2L, 2L, 1L, 1L, 2L, 1L)
emp <- countEmpBivariate(
  chainFM_V1, chainSM_V1, chainFM_V2, chainSM_V2,
  states = 2L
)
dim(emp)
```

dyadic_bivariate_example

Synthetic bivariate dyadic sequence example

Description

A synthetic bivariate dyadic sequence with 90 observations, designed for package workflow examples.

Usage

```
dyadic_bivariate_example
```

Format

A data frame with 90 rows and 5 columns:

time Index of the measurement occasion.

FM_V1 Integer variable 1 state for the first member, taking values 1 or 2.

SM_V1 Integer variable 1 state for the second member, taking values 1 or 2.

FM_V2 Integer variable 2 state for the first member, taking values 1 or 2.

SM_V2 Integer variable 2 state for the second member, taking values 1 or 2.

Details

The bivariate workflow classifies this example as complete using `bivariateCase` with `alpha = 0.05`. The complete bivariate pattern selected by `completePattern` is D2.

dyadic_univariate_example

Synthetic univariate dyadic sequence example

Description

A synthetic dyadic sequence with 90 observations, designed for package workflow examples.

Usage

```
dyadic_univariate_example
```

Format

A data frame with 90 rows and 3 columns:

time Index of the measurement occasion.

FM Integer state for the first member, taking values 1 or 2.

SM Integer state for the second member, taking values 1 or 2.

Details

The package workflow classifies this example as PM (A3) using [univariatePattern](#) with `states = 2`.

mleEstimation

Maximum likelihood estimation of transition probabilities

Description

Estimates transition probabilities by maximum likelihood from an empirical count matrix returned by [countEmp](#) or [countEmpBivariate](#).

Usage

```
mleEstimation(empirical)
```

Arguments

`empirical` An empirical count matrix.

Details

Each row of `empirical` is normalized independently. Rows with zero total count are assigned a uniform probability vector, so each row of the returned matrix sums to one.

Value

A numeric matrix with class `c("dyadic_mle", "matrix", "array")` containing estimated transition probabilities with the same dimensions as `empirical`. It remains usable as an ordinary matrix. `print()`, `summary()`, and `utils::toLatex()` methods are available for this result.

Examples

```
chainFM <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM <- c(2L, 1L, 2L, 1L, 1L, 2L)
emp <- countEmp(chainFM, chainSM, states = 2L)
mleEstimation(emp)
```

partialPattern

Partial bivariate pattern identification by AIC

Description

Compares the partial bivariate patterns B1, B2, and B3 using AIC and returns the selected pattern.

Usage

```
partialPattern(empirical)
```

Arguments

`empirical` A numeric matrix with 16 rows and 2 columns containing finite non-negative counts, as returned by `countEmpBivariate`. Integer and double storage modes are both accepted.

Details

Conditional on the partial bivariate case, the G-squared deviance is computed for each B1, B2, and B3 candidate before calculating

$$AIC = G^2 + 2k,$$

where

$$G^2 = 2 \sum_{ij} O_{ij} \log(O_{ij}/E_{ij}).$$

The candidate with the smallest AIC is selected. If candidates have exactly equal AIC values, the first candidate in the documented comparison order is selected.

Value

A list with class `c("dyadic_pattern", "list")` containing components `aic` (a data frame with candidate patterns and AIC values), `pattern` (the selected pattern label), and `call`. It remains usable as an ordinary list. When `empirical` was created by `countEmpBivariate()`, its stored sequence metadata is propagated so that `plot()` can display the four observed state strips.

Examples

```

chainFM_V1 <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM_V1 <- c(2L, 1L, 2L, 1L, 1L, 2L)
chainFM_V2 <- c(1L, 1L, 2L, 2L, 1L, 2L)
chainSM_V2 <- c(2L, 2L, 1L, 1L, 2L, 1L)
emp <- countEmpBivariate(
  chainFM_V1, chainSM_V1, chainFM_V2, chainSM_V2,
  states = 2L
)
partialPattern(emp)

```

plot.dyadic_pattern *Plot dyadic categorical state strips*

Description

Displays each categorical sequence retained by a dyadicMarkov result as a horizontal state strip. A strip consists of adjacent coloured intervals: each interval represents one measurement occasion, and its colour identifies the observed state. Consecutive occasions in the same state therefore form a continuous run. Vertical position identifies the member and, for bivariate results, the variable.

Usage

```

## S3 method for class 'dyadic_pattern'
plot(x, col = NULL, main = NULL, cex = NULL, ...)

## S3 method for class 'dyadic_case'
plot(x, col = NULL, main = NULL, cex = NULL, ...)

```

Arguments

x	A dyadic_pattern or dyadic_case object.
col	Character vector containing exactly one distinct, non-fully-transparent colour per state, or NULL for an accessible default palette. Colours identify states consistently across every strip.
main	Optional main title. If NULL, a title appropriate to the univariate or bivariate sequence is used.
cex	Optional positive text scaling factor. If NULL, the default is 1.
...	Additional arguments are not currently supported.

Details

Univariate results contain two strips, one for each member, and support any integer states ≥ 2 . Bivariate results contain four strips for the two members on the main and second variables; the currently developed bivariate method supports states = 2 only. The plot displays observed sequences, not fitted probabilities or inferred dependency structures. Statistical identification results remain available through print() and summary().

Value

Invisibly returns the validated plotting specification used to draw the state strips.

References

Tueller, S. J., Van Dorn, R. A., and Bobashev, G. V. (2016). Visualization of categorical longitudinal and time series data. *Methods Report RTI Press*, 2016. doi:10.3768/rtipress.2016.mr.0033.1602.

Examples

```
chainFM <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM <- c(2L, 1L, 2L, 1L, 1L, 2L)
univariate <- univariatePattern(
  chainFM,
  chainSM,
  states = 2L
)
plot(univariate)

chainFM_V2 <- c(1L, 1L, 2L, 2L, 1L, 2L)
chainSM_V2 <- c(2L, 2L, 1L, 1L, 2L, 1L)
empirical <- countEmpBivariate(
  chainFM,
  chainSM,
  chainFM_V2,
  chainSM_V2,
  states = 2L
)
bivariate <- bivariateCase(empirical)
plot(bivariate)
```

sensitivity_simulation_data

Sensitivity-analysis simulation datasets

Description

Fixed archived simulation datasets used in the sensitivity-analysis vignette for sequence lengths 30, 60, 90, 180, and 720. The datasets contain binary bivariate dyadic sequences and are re-analysed by the current package; they are not re-simulated during package use or testing.

Format

Each object is a data frame with 4,000 rows, representing 1,000 dyads observed for two members and two variables. The columns TM1, ..., TMn contain the simulated binary states for the corresponding sequence length. The remaining columns are members, variable, caseSimulated, dyad, and local. The five datasets were simulated separately for sequence lengths 30, 60, 90, 180, and 720 and are therefore not nested prefixes of one another. The caseSimulated column records the interaction pattern used to simulate each member-variable sequence, whereas local records the archived

classification produced by the historical analysis. Historical terminology is retained in these fixed simulation datasets; current package output uses the current scientific nomenclature.

univariatePattern	<i>Univariate pattern identification for dyadic Markov chains</i>
-------------------	---

Description

Computes empirical transition counts, estimates transition probabilities by maximum likelihood, and performs likelihood-ratio tests against the actor-only and partner-only constrained models to identify the univariate pattern of interaction.

Usage

```
univariatePattern(chainFM, chainSM, states, alpha = 0.05)
```

Arguments

chainFM	Numeric vector of observed states for the first member (FM). Values must be integers. Must have the same length as chainSM.
chainSM	Numeric vector of observed states for the second member (SM). Values must be integers. Must have the same length as chainFM.
states	A single positive integer greater than or equal to 2 giving the number of states.
alpha	A single number in (0, 1) giving the significance level. Default is 0.05.

Details

The nested full/restricted comparisons form the univariate likelihood-ratio test (LRT) procedure. dyadicMarkov evaluates each LRT comparison using Pearson's chi-squared statistic,

$$X^2 = \sum_{ij} (O_{ij} - E_{ij})^2 / E_{ij},$$

where O and E are the empirical and theoretical transition counts. A comparison is treated as rejected when $p \leq \alpha$. Pattern labels summarize which structure is retained by the tests: IM (A0) denotes an independence pattern, APM (A1) an actor-partner pattern, AM (A2) an actor-only pattern, and PM (A3) a partner-only pattern.

Value

A list with class `c("dyadic_pattern", "list")` containing two `htest` objects (`TEST.AM`, `TEST.PM`), the retained input sequences `chainFM` and `chainSM`, a string pattern, and metadata fields `alpha`, `states`, and `call`. It remains usable as an ordinary list.

Examples

```
chainFM <- c(1L, 2L, 1L, 2L, 2L, 1L)
chainSM <- c(2L, 1L, 2L, 1L, 1L, 2L)
univariatePattern(chainFM, chainSM, states = 2L, alpha = 0.05)
```

Index

* datasets

- dyadic_bivariate_example, [6](#)
- dyadic_univariate_example, [7](#)

bivariateCase, [2](#), [6](#)

completePattern, [3](#), [6](#)

countEmp, [4](#), [7](#)

countEmpBivariate, [2](#), [3](#), [5](#), [7](#), [8](#)

data_complete_180
 (sensitivity_simulation_data),
 [10](#)

data_complete_30
 (sensitivity_simulation_data),
 [10](#)

data_complete_60
 (sensitivity_simulation_data),
 [10](#)

data_complete_720
 (sensitivity_simulation_data),
 [10](#)

data_complete_90
 (sensitivity_simulation_data),
 [10](#)

dyadic_bivariate_example, [6](#)

dyadic_univariate_example, [7](#)

mleEstimation, [7](#)

partialPattern, [8](#)

plot.dyadic_case (plot.dyadic_pattern),
 [9](#)

plot.dyadic_pattern, [9](#)

sensitivity_simulation_data, [10](#)

univariatePattern, [7](#), [11](#)