Package 'ctsfeatures'

February 20, 2023

Type Package

Title Analyzing Categorical Time Series

Version 1.0.0

Description An implementation of several functions for feature extraction in categorical time series datasets. Specifically, some features related to marginal distributions and serial dependence patterns can be computed. These features can be used to feed clustering and classification algorithms for categorical time series, among others. The package also includes some interesting datasets containing biological sequences. Practitioners from a broad variety of fields could benefit from the general framework provided by 'ctsfeatures'.

License GPL-2

Encoding UTF-8

LazyData true

LazyDataCompression xz

Depends R (>= 4.0.0)

RoxygenNote 7.1.2

Imports ggplot2, astsa, latex2exp, Rdpack, Bolstad2

RdMacros Rdpack

NeedsCompilation no

Author Angel Lopez-Oriona [aut, cre],
Jose A. Vilar [aut]

Maintainer Angel Lopez-Oriona <oriona38@hotmail.com>

Repository CRAN

Date/Publication 2023-02-20 11:40:08 UTC

R topics documented:

`binarization` *Constructs the binarized time series associated with a given categorical time series*

Description

`binarization` constructs the binarized time series associated with a given categorical time series.

Usage

```
binarization(series, categories)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function constructs the binarized time series, which is defined as $\overline{\boldsymbol{Y}}_t = \{\overline{\boldsymbol{Y}}_1, \ldots, \overline{\boldsymbol{Y}}_T\}$, with $\overline{\boldsymbol{Y}}_k = (\overline{Y}_{k,1}, \ldots, \overline{Y}_{k,r})^\top$ such that $\overline{Y}_{k,i} = 1$ if $\overline{X}_k = i$ ($k = 1, \ldots, T, , i = 1, \ldots, r$). The binarized series is constructed in the form of a matrix whose rows represent time observations and whose columns represent the categories in the original series

Value

The binarized time series.

Author(s)

Ángel López-Oriona, José A. Vilar

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

Examples

```
binarized_series <- binarization(GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Constructing the binarized
# time series for the first CTS in dataset GeneticSequences
```

---

chebycheff_dispersion *Computes the Chebycheff dispersion of a categorical time series*

Description

`chebycheff_dispersion` returns the value of the Chebycheff dispersion for a categorical time series

Usage

```
chebycheff_dispersion(series, categories)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Chebycheff dispersion, $\widehat{c} = \frac{r}{r-1}(1 - \max_i \widehat{p}_i)$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, $i = 1, \ldots, r$.

Value

The value of the Chebycheff dispersion.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
cd <- chebycheff_dispersion(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Chebycheff dispersion
# for the first series in dataset GeneticSequences
```

---

cohens_kappa *Computes the Cohen's kappa of a categorical time series*

Description

cohens_kappa returns the value of the Cohen's kappa for a categorical time series

Usage

```
cohens_kappa(series, lag = 1, categories, features = FALSE)
```

Arguments

series	A CTS.
lag	The considered lag (default is 1).
categories	A vector of type factor containing the corresponding categories.
features	Logical. If features = FALSE (default), the value of the Cohen's kappa is returned. Otherwise, the function returns a matrix with the individual components of the Cohen's kappa.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Cohen's kappa, $\widehat{\kappa}(l) = \frac{\sum_{j=1}^{r}(\widehat{p}_{jj}(l) - \widehat{p}_j^2)}{1 - \sum_{i=1}^{r}\widehat{p}_i^2}$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, and $\widehat{p}_{ij}(l)$ is the natural estimate of the joint probability for categories i and j at lag l, $i, j = 1, \ldots, r$. If features = TRUE, the function returns a vector whose components are the quantities $\widehat{p}_{ii}(l) - \widehat{p}_i^2$, $i = 1, 2, \ldots, r$.

Value

If `features = FALSE` (default), returns the value of the Cohen's kappa. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Cohen's kappa.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
ck <- cohens_kappa(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Cohen's kappa
# for the first series in dataset GeneticSequences
feature_matrix <- cohens_kappa(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

---

conditional_probabilities

Computes the conditional probabilities of a categorical time series

---

Description

`conditional_probabilities` returns a matrix with the conditional probabilities of a categorical time series

Usage

```
conditional_probabilities(series, lag = 1, categories)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the matrix $\widehat{\boldsymbol{P}}^c(l) = \left(\widehat{p}^c_{ij}(l)\right)_{1\leq i,j\leq r}$, with $\widehat{p}^c_{ij}(l) = \frac{TN_{ij}(l)}{(T-l)N_i}$, where N_i is the number of elements equal to i in the realization $\overline{X}_t$ and $N_{ij}(l)$ is the number of pairs $(\overline{X}_t, \overline{X}_{t-l}) = (i, j)$ in the realization $\overline{X}_t$.

Value

A matrix with the conditional probabilities.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
matrix_cp <- conditional_probabilities(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the matrix of
# joint probabilities for the first series in dataset GeneticSequences
```

`cramers_vi` *Computes the Cramer's vi of a categorical time series*

Description

`cramers_vi` returns the value of the Cramer's vi for a categorical time series

Usage

```
cramers_vi(series, lag = 1, categories, features = FALSE)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.
`features`	Logical. If `features = FALSE` (default), the value of the Cramer's vi is returned. Otherwise, the function returns a matrix with the individual components of the Cramer's vi.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Cramer's vi, $\widehat{v}(l) = \sqrt{\frac{1}{r-1}\sum_{i,j=1}^{r}\frac{(\widehat{p}_{ij}(l)-\widehat{p}_i\widehat{p}_j)^2}{\widehat{p}_i\widehat{p}_j}}$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, and $\widehat{p}_{ij}(l)$ is the natural estimate of the joint probability for categories i and j at lag l, $i, j = 1, \ldots, r$. If `features = TRUE`, the function returns the same output as the function `pearson_measure`.

Value

If `features = FALSE` (default), returns the value of the Cramer's vi. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Cramer's vi.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
cv <- cramers_vi(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Cramer's vi
# for the first series in dataset GeneticSequences
feature_matrix <- cramers_vi(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

---

cts_plot *Constructs a categorical time series plot*

Description

`cts_plot` constructs a categorical time series plot

Usage

```
cts_plot(series, categories, title = "Time series plot")
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`title`	The title of the graph.

Details

Constructs a categorial time series plot for a given CTS.

Value

The categorical time series plot.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2018). *An introduction to discrete-valued time series*. John Wiley and Sons.

Examples

```
time_series_plot <- cts_plot(series = GeneticSequences$data[[1]][1 : 50],
categories = factor(c('a', 'c', 'g', 't'))) # Constructs a categorical
# time series plot for the first 50 observations of the first  time series in
# dataset GeneticSequences
```

`cycle_control_chart` *Constructs a control chart for the cycle lengths of a categorical series*

Description

`cycle_control_chart` constructs a control chart for the cycle lengths of a categorical series

Usage

```
cycle_control_chart(
  series,
  categories,
  mu_t,
  lcl_t,
  ucl_t,
  plot = TRUE,
  title = "Control chart (cycles)",
  ...
)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`mu_t`	The mean of the process measuring the cycle lengths.
`lcl_t`	The lower control limit.
`ucl_t`	The upper control limit.
`plot`	Logical. If `plot = TRUE` (default), returns the control chart. Otherwise, returns the standardized statistic.
`title`	The title of the graph.
`...`	Additional parameters for the function.

Details

Constructs a control chart of a CTS based on cycle lengths. The chart is based on the standardized statistic $T_t = T_t^{(L)} + T_t^{(U)}$, with $T_t^{(L)} = \min\left(0, \frac{C_t - \mu_t}{|LCL_t - \mu_t|}\right)$ and $T_t^{(U)} = \max\left(0, \frac{C_t - \mu_t}{|UCL_t - \mu_t|}\right)$, where Z_t expresses the length of a cycle ending with a specific category, μ_t denotes the mean of Z_t and LCL_t and UCL_t are lower and upper individual control limits, respectively. Note that an out-of-control alarm is signalled if $T_t < -1$ or $T_t > 1$.

Value

If `plot = TRUE` (default), represents the control chart for the cycle lengths. Otherwise, the function returns a matrix with the values of the standardized statistic for each time t

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2008). "Visual analysis of categorical time series." *Statistical Methodology*, **5**(1), 56–71.

Examples

```
cycle_cc <- cycle_control_chart(series = SyntheticData1$data[[1]],
categories = factor(c('1', '2', '3')), mu_t = c(1, 1.5, 1),
lcl_t = rep(10, 600), ucl_t = rep(10, 600)) # Representing
# a control chart for the cycle lengths
cycle_cc <- cycle_control_chart(series = SyntheticData1$data[[1]],
categories = factor(c('1', '2', '3')), mu_t = c(1, 1.5, 1),
lcl_t = rep(10, 600), ucl_t = rep(10, 600), plot = FALSE) # Computing the
# corresponding standardized statistic
```

`entropy` *Computes the entropy of a categorical time series*

Description

`entropy` returns the value of the entropy for a categorical time series

Usage

```
entropy(series, categories, features = FALSE)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`features`	Logical. If `features = FALSE` (default), the value of the entropy is returned. Otherwise, the function returns a vector with the individual components of the entropy.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated entropy, $\widehat{e} = \frac{-1}{\ln(r)} \sum_{i=1}^{r} \widehat{p}_i \ln \widehat{p}_i$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, $i = 1, \ldots, r$. If `features = TRUE`, the function returns a vector whose components are the quantities $\widehat{p}_i \ln(\widehat{p}_i)$, $i = 1, 2, \ldots, r$.

Value

The value of the entropy.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
et <- entropy(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the entropy
# for the first series in dataset GeneticSequences
```

`GeneticSequences` *GeneticSequences*

Description

Categorical time series (CTS) of DNA sequences from different viruses

Usage

```
data(GeneticSequences)
```

Format

A `list` with two elements, which are:

`data` A list with 32 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a categorical time series containing four categories (DNA bases). The numeric vector `classes` is formed by integers from 1 to 4, indicating that there are 4 different classes in the database. Each class is associated with a different family of viruses. For more information, see López-Oriona et al. (2023).

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

`gini_index` *Computes the gini index of a categorical time series*

Description

`gini_index` returns the value of the gini index for a categorical time series

Usage

```
gini_index(series, categories)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated gini index, $\widehat{g} = \frac{r}{r-1}(1 - \sum_{i=1}^{r} \widehat{p}_i^2)$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, $i = 1, \ldots, r$.

Value

The value of the gini index.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
gi <- gini_index(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the gini index
# for the first series in dataset GeneticSequences
```

---

gk_lambda *Computes the Goodman and Kruskal's lambda of a categorical time series*

Description

gk_lambda returns the value of the Goodman and Kruskal's lambda for a categorical time series

Usage

```
gk_lambda(series, lag = 1, categories, features = FALSE)
```

Arguments

series	A CTS.
lag	The considered lag (default is 1).
categories	A vector of type factor containing the corresponding categories.
features	Logical. If features = FALSE (default), the value of Goodman and Kruskal's lambda is returned. Otherwise, the function returns a matrix with the individual components of Goodman and Kruskal's lambda

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Goodman and Kruskal's lambda, $\widehat{\lambda}(l) = \frac{\sum_{j=1}^{r} \max_i \widehat{p}_{ij}(l) - \max_i \widehat{p}_i}{1 - \max_i \widehat{p}_i}$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, and $\widehat{p}_{ij}(l)$ is the natural estimate of the joint probability for categories i and j at lag l, $i, j = 1, \ldots, r$. If features = TRUE, the function returns a vector whose components are the quantities $\max_i \widehat{p}_{ij}(l)$, $i = 1, 2, \ldots, r$.

Value

If features = FALSE (default), returns the value of the Goodman and Kruskal's lambda. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Goodman and Kruskal's lambda.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
gkl <- gk_lambda(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Goodman and Kruskal's lambda
# for the first series in dataset GeneticSequences
feature_matrix <- gk_lambda(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

---

gk_tau *Computes the Goodman and Kruskal's tau of a categorical time series*

Description

gk_tau returns the value of the Goodman and Kruskal's tau for a categorical time series

Usage

```
gk_tau(series, lag = 1, categories, features = FALSE)
```

Arguments

series	A CTS.
lag	The considered lag (default is 1).
categories	A vector of type factor containing the corresponding categories.
features	Logical. If features = FALSE (default), the value of Goodman and Kruskal's tau is returned. Otherwise, the function returns a matrix with the individual components of Goodman and Kruskal's tau.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Goodman and Kruskal's tau, $\widehat{\tau}(l) = \frac{\sum_{i,j=1}^{r} \frac{\widehat{p}_{ij}(l)^2}{\widehat{p}_j} - \sum_{i=1}^{r} \widehat{p}_i^2}{1 - \sum_{i=1}^{r} \widehat{p}_i^2}$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, and $\widehat{p}_{ij}(l)$ is the natural estimate of the joint probability for categories i and j at lag l, $i, j = 1, \ldots, r$. If features = TRUE, the function returns a matrix whose components are the quantities $\frac{\widehat{p}_{ij}(l)^2}{\widehat{p}_j}$, $i, j = 1, 2, \ldots, r$.

Value

If features = FALSE (default), returns the value of the Goodman and Kruskal's tau. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Goodman and Kruskal's tau.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
gkt <- gk_tau(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Goodman and Kruskal's tau
# for the first series in dataset GeneticSequences
feature_matrix <- gk_tau(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

---

ifs_circle_transformation

Constructs the IFS circle transformation of a categorical time series

---

Description

`ifs_circle_transformation` constructs the IFS circle transformation of a categorical time series.

Usage

```
ifs_circle_transformation(
  series,
  categories,
  alpha,
  beta,
  title = "IFS circle transformation",
  ...
)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`alpha`	Parameter alpha in the circle transformation.
`beta`	Parameter beta in the circle transformation.
`title`	The title of the graph.
`...`	Additional parameters for the function.

Details

Constructs the IFS circle transformation for a given CTS, which is useful to identify cycles of arbitrary length.

Value

The IFS circle transformation.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2008). "Visual analysis of categorical time series." *Statistical Methodology*, **5**(1), 56–71.

Examples

```
ct <- ifs_circle_transformation(GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')),
alpha = 0.1, beta = 0.1) # Constructing the IFS circle transformation
# for the first CTS in dataset GeneticSequences
```

---

joint_probabilities *Computes the joint probabilities of a categorical time series*

Description

`joint_probabilities` returns a matrix with the joint probabilities of a categorical time series

Usage

```
joint_probabilities(series, lag = 1, categories)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the matrix $\widehat{\boldsymbol{P}}(l) = \left(\widehat{p}_{ij}(l)\right)_{1 \leq i,j \leq r}$, with $\widehat{p}_{ij}(l) = \frac{N_{ij}(l)}{T-l}$, where $N_{ij}(l)$ is the number of pairs $(\overline{X}_t, \overline{X}_{t-l}) = (i, j)$ in the realization $\overline{X}_t$.

Value

A matrix with the joint probabilities.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
matrix_jp <- joint_probabilities(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the matrix of
# joint probabilities for the first series in dataset GeneticSequences
```

---

marginal_control_chart

Constructs a control chart for the marginal distribution of a categorical series

---

Description

`marginal_control_chart` constructs a control chart for the marginal distribution of a categorical series

Usage

```
marginal_control_chart(
  series,
  categories,
  c,
  sigma,
  lambda = 0.99,
  k = 3.3,
  min_max = FALSE,
  plot = TRUE,
  title = "Control chart (marginal)",
  ...
)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`c`	The hypothetical marginal distribution.
`sigma`	A matrix containing the variances for each category (columns) and each time t (rows).
`lambda`	The constant lambda to construct the EWMA estimator.
`k`	The constant k to construct the k sigma limits.

`min_max`	Logical. If `min_max = FALSE` (default), the standard control chart for the marginal distribution is plotted. Otherwise, the reduced control chart is plotted, i.e., only the minimum and maximum values of the standardized statistics (with respect to the set of categories) are considered.
`plot`	Logical. If `plot = TRUE` (default), returns the control chart. Otherwise, returns the standardized statistics or their maximum and minimum value for each time t.
`title`	The title of the graph.
`...`	Additional parameters for the function.

Details

Constructs a control chart of a CTS with range $\mathcal{V} = \{1, \ldots, r\}$ based on the marginal distribution. The chart relies on the standardized statistic $T_{t,i} = \frac{\hat{\pi}_{t,i}^{(\lambda)} - p_i}{k \cdot \sigma_{t,i}}$, where the $\hat{\pi}_{t,i}^{(\lambda)}$, $i = 1, \ldots, r$, are the components of the EWMA estimator of the marginal distribution, p_i is the marginal probability of category i, $\sigma_{t,i}$ is the variance of $\hat{\pi}_{t,i}^{(\lambda)}$ and k is a constant set by the user. If `min_max = FALSE`, then only the statistics $T_t^{\min} = \min_{i \in \mathcal{V}} T_{t,i}$ and $T_t^{\max} = \max_{i \in \mathcal{V}} T_{t,i}$ are plotted. An out-of-control alarm is signalled if the statistics are below -1 or above 1.

Value

If `plot = TRUE` (default), represents the control chart for the marginal distribution. Otherwise, the function returns a matrix with the values of the standardized statistics for each time t

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2008). "Visual analysis of categorical time series." *Statistical Methodology*, **5**(1), 56–71.

Examples

```
cycle_md <- marginal_control_chart(series = SyntheticData1$data[[1]],
categories = factor(c('1', '2', '3')), c = c(0.3, 0.3, 0.4),
sigma = matrix(rep(c(1, 1, 1), 600), nrow = 600)) # Representing
# a control chart for the marginal distribution
cycle_md <- marginal_control_chart(series = SyntheticData1$data[[1]],
categories = factor(c('1', '2', '3')), c = c(0.3, 0.3, 0.4),
sigma = matrix(rep(c(1, 1, 1), 600), nrow = 600)) # Computing the
# corresponding standardized statistic
```

`marginal_probabilities` *Computes the marginal probabilities of a categorical time series*

Description

`marginal_probabilities` returns a vector with the marginal probabilities of a categorical time series

Usage

```
marginal_probabilities(series, categories)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the vector $\widehat{\boldsymbol{p}} = (\widehat{p}_1, \ldots, \widehat{p}_r)$, with $\widehat{p}_i = \frac{N_i}{T}$, where N_i is the number of elements equal to i in the realization $\overline{X}_t$.

Value

A vector with the marginal probabilities.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
vector_mp <- marginal_probabilities(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the vector of
# marginal probabilities for the first series in dataset GeneticSequences
```

`pattern_histogram` *Constructs the pattern histogram associated with a given category of a categorical time series*

Description

`pattern_histogram` constructs the pattern histogram associated with a given category of a categorical time series.

Usage

```
pattern_histogram(
  series,
  category,
  plot = TRUE,
  title = paste0("Pattern histogram (", category, ")"),
  ...
)
```

Arguments

`series`	A CTS.
`category`	The selected category.
`plot`	Logical. If `plot = TRUE` (default), returns the pattern histogram. Otherwise, returns the frequencies of cycle lengths associated with the corresponding category.
`title`	The title of the graph.
`...`	Additional parameters for the function.

Details

Constructs the pattern histogram for a specific category of a CTS. This graph represents the frequencies of the cycles for the corresponding category according to their length.

Value

The pattern histogram.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2008). "Visual analysis of categorical time series." *Statistical Methodology*, **5**(1), 56–71.

Examples

```
ph <- pattern_histogram(GeneticSequences$data[[1]],
category = 'a') # Constructing the pattern histogram
# for the first CTS in dataset GeneticSequences concerning the category 'a'
cycle_lengths <- pattern_histogram(GeneticSequences$data[[1]],
category = 'a', plot = FALSE) # Obtaining the frequencies of cycle lengths
```

`pearson_measure` *Computes the Pearson measure of a categorical time series*

Description

`pearson_measure` returns the value of the Pearson measure for a categorical time series

Usage

```
pearson_measure(series, lag = 1, categories, features = FALSE)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.
`features`	Logical. If `features = FALSE` (default), the value of the Pearson measure is returned. Otherwise, the function returns a matrix with the individual components of the Pearson measure.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Pearson measure, $\widehat{X}^2_T(l) = T \sum_{i,j=1}^{r} \frac{(\widehat{p}_{ij}(l) - \widehat{p}_i \widehat{p}_j)^2}{\widehat{p}_i \widehat{p}_j}$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, and $\widehat{p}_{ij}(l)$ is the natural estimate of the joint probability for categories i and j at lag l, $i, j = 1, \ldots, r$. If `features = TRUE`, the function returns a matrix whose components are the quantities $\frac{(\widehat{p}_{ij}(l) - \widehat{p}_i \widehat{p}_j)^2}{\widehat{p}_i \widehat{p}_j}$, $i, j = 1, 2, \ldots, r$.

Value

If `features = FALSE` (default), returns the value of the Pearson measure. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Pearson measure.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
pm <- pearson_measure(series = SyntheticData1$data[[1]],
categories = factor(c(1, 2, 3))) # Computing the Pearson measure
# for the first series in dataset GeneticSequences
feature_matrix <- pearson_measure(series = SyntheticData1$data[[1]],
categories = factor(c(1, 2, 3)), features = TRUE) # Computing the corresponding
# matrix of features
```

`phi2_measure` *Computes the Phi2 measure of a categorical time series*

Description

`phi2_measure` returns the value of the Phi2 measure for a categorical time series

Usage

```
phi2_measure(series, lag = 1, categories, features = FALSE)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.
`features`	Logical. If `features = FALSE` (default), the value of the Phi2 measure is returned. Otherwise, the function returns a matrix with the individual components of the Phi2 measure.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Phi2 measure, $\widehat{\Phi}^2(l) = \frac{\widehat{X}^2_T(l)}{T}$, where $\widehat{X}^2_T$ is the estimated Pearson measure. If `features = TRUE`, the function returns the same output as the function `pearson_measure`.

Value

If `features = FALSE` (default), returns the value of the Phi2 measure. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Phi2 measure.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
phi2m <- phi2_measure(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Phi2 measure
# for the first series in dataset GeneticSequences
feature_matrix <- phi2_measure(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

---

plot_cohens_kappa *Constructs a serial dependence plot based on Cohen's kappa*

Description

plot_cohens constructs a serial dependence plot of a categorical time series based on Cohen's kappa

Usage

```
plot_cohens_kappa(
  series,
  categories,
  max_lag = 10,
  alpha = 0.05,
  plot = TRUE,
  title = "Serial dependence plot",
  bar_width = 0.12,
  ...
)
```

Arguments

series	A CTS.
categories	A vector of type factor containing the corresponding categories.
max_lag	The maximum lag represented in the plot (default is 10).
alpha	The significance level for the corresponding hypothesis test (default is 0.05).
plot	Logical. If plot = TRUE (default), returns the serial dependence plot. Otherwise, returns a list with the values of Cohens's kappa, the critical value and the corresponding p-values.

`title`	The title of the graph.
`bar_width`	The width of the corresponding bars.
`...`	Additional parameters for the function.

Details

Constructs a serial dependence plot based on Cohens's kappa, $\widehat{\kappa}(l)$, for several lags. A dashed lined is incorporated indicating the critical value of the test based on the following asymptotic approximation (under the i.i.d. assumption)

$$\sqrt{\frac{T}{V(\widehat{\boldsymbol{p}})}}\left(\widehat{\kappa}(l)+\frac{1}{T}\right)\sim N\big(0,1\big),$$

where T is the series length, $\widehat{\boldsymbol{p}}=(\widehat{p}_1,\ldots,\widehat{p}_r)$ is the vector of estimated marginal probabilities for the r categories of the series and $V(\widehat{\boldsymbol{p}})=1-\frac{1+2\sum_{i=1}^{r}\widehat{p}_i^3-3\sum_{i=1}^{r}\widehat{p}_i^2}{(1-\sum_{i=1}^{r}\widehat{p}_i^2)^2}$.

Value

If `plot = TRUE` (default), returns the serial dependence plot based on Cramer's vi. Otherwise, the function returns a list with the values of Cohens's kappa, the critical value and the corresponding p-values.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2011). "Empirical measures of signed serial dependence in categorical time series." *Journal of Statistical Computation and Simulation*, **81**(4), 411–429.

Examples

```
plot_ck <- plot_cohens_kappa(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), max_lag = 3) # Representing
# the serial dependence plot
list_ck <- plot_cohens_kappa(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), max_lag = 3, plot = FALSE) # Obtaining
# the values of Cohens's kappa, the critical value and the p-values
```

`plot_cramers_vi` *Constructs a serial dependence plot based on Cramer's vi*

Description

`plot_cramers` constructs a serial dependence plot of a categorical time series based on Cramer's vi

Usage

```
plot_cramers_vi(
  series,
  categories,
  max_lag = 10,
  alpha = 0.05,
  plot = TRUE,
  title = "Serial dependence plot",
  bar_width = 0.12,
  ...
)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`max_lag`	The maximum lag represented in the plot (default is 10).
`alpha`	The significance level for the corresponding hypothesis test (default is 0.05).
`plot`	Logical. If `plot = TRUE` (default), returns the serial dependence plot. Otherwise, returns a list with the values of Cramer's vi, the critical value and the corresponding p-values.
`title`	The title of the graph.
`bar_width`	The width of the corresponding bars.
`...`	Additional parameters for the function.

Details

Constructs a serial dependence plot based on Cramer's vi, $\widehat{v}(l)$, for several lags. A dashed lined is incorporated indicating the critical value of the test based on the following asymptotic approximation (under the i.i.d. assumption)

$$T(r-1)\widehat{v}(l)^2 \sim \chi^2_{(r-1)^2},$$

where T is the series length and r is the number of categories in the time series.

Value

If `plot = TRUE` (default), returns the serial dependence plot based on Cramer's vi. Otherwise, the function returns a list with the values of Cramer's vi, the critical value and the corresponding p-values.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH (2013). "Serial dependence of NDARMA processes." *Computational Statistics and Data Analysis*, **68**, 213–238.

Examples

```
plot_cv <- plot_cramers_vi(series = SyntheticData1$data[[1]],
categories = factor(c(1, 2, 3)), max_lag = 3) # Representing
# the serial dependence plot
list_cv <- plot_cramers_vi(series = SyntheticData1$data[[1]],
categories = factor(c(1, 2, 3)), max_lag = 3, plot = FALSE) # Obtaining
# the values of Cramer's vi, the critical value and the p-values
```

ProteinSequences *ProteinSequences*

Description

Categorical time series (CTS) of protein sequences from different species

Usage

```
data(ProteinSequences)
```

Format

A `list` with two elements, which are:

`data` A list with 40 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a categorical time series containing three categories (amino-acids). The numeric vector `classes` is formed by integers from 1 to 4, indicating that there are 4 different classes in the database. Each class is associated with a different family of viruses. For more information, see López-Oriona et al. (2023).

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

`rate_evolution_graph` *Constructs the rate evolution graph for a categorical time series*

Description

`rate_evolution_graph` constructs the rate evolution graph proposed by Ribler (1997).

Usage

```
rate_evolution_graph(series, categories, title = "Rate evolution graph", ...)
```

Arguments

`series`	A CTS.
`categories`	A vector of type factor containing the corresponding categories.
`title`	The title of the graph.
`...`	Additional parameters for the function.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, and the corresponding binarized time series, $\overline{\boldsymbol{Y}}_t = \{\overline{\boldsymbol{Y}}_1, \ldots, \overline{\boldsymbol{Y}}_T\}$, the function constructs the rate evolution graph. Specifically, consider the series of cumulated sums given by $\overline{\boldsymbol{C}}_t = \{\overline{\boldsymbol{C}}_1, \ldots, \overline{\boldsymbol{C}}_T\}$, with $\overline{\boldsymbol{C}}_k = \sum_{s=1}^{k} \overline{\boldsymbol{Y}}_s$, $k = 1, \ldots, T$. The rate evolution graph displays a standard time series plot for each one of the components of $\overline{\boldsymbol{C}}_t$ simultaneously in one graph.

Value

The rate evolution graph.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Ribler RL (1997). *Visualizing categorical time series data with applications to computer and communications network traces*. Ph.D. thesis, Virginia Polytechnic Institute and State University.

Examples

```
reg <- rate_evolution_graph(GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Constructing the rate
# evolution graph for the first time series in dataset GeneticSequences
```

---

sakoda_measure *Computes the Sakoda measure of a categorical time series*

Description

sakoda_measure returns the value of the Sakoda measure for a categorical time series

Usage

```
sakoda_measure(series, lag = 1, categories, features = FALSE)
```

Arguments

series	A CTS.
lag	The considered lag (default is 1).
categories	A vector of type factor containing the corresponding categories.
features	Logical. If features = FALSE (default), the value of the Sakoda measure is returned. Otherwise, the function returns a matrix with the individual components of the Sakoda measure.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated Sakoda measure, $\widehat{p}^*(l) = \sqrt{\frac{r\widehat{\Phi}^2(l)}{(r-1)(1+\widehat{\Phi}^2(l))}}$, where $\widehat{\Phi}^2(l)$ is the estimated Phi2 measure. If features = TRUE, the function returns the same output as the function pearson_measure.

Value

If features = FALSE (default), returns the value of the Sakoda measure. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the Sakoda measure.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
sm <- sakoda_measure(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the Sakoda measure
# for the first series in dataset GeneticSequences
feature_matrix <- sakoda_measure(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

`spectral_envelope` *Computes the spectral envelope of a categorical time series*

Description

`spectral_envelope` computes the spectral envelope a categorical time series

Usage

```
spectral_envelope(binarized_series, plot = TRUE)
```

Arguments

`binarized_series`	A CTS in binarized form.
`plot`	Logical. If `plot = TRUE` (default), returns a plot of the spectral envelope. Otherwise, returns the values of the spectral envelope at each frequency and the corresponding set of optimal scalings

Details

The function represents the spectral envelope of a categorical time series

Value

If `plot = TRUE` (default), returns returns a plot of the spectral envelope. Otherwise, the function returns the values of the spectral envelope at each frequency and the corresponding set of optimal scalings

Author(s)

Ángel López-Oriona, José A. Vilar

References

Stoffer DS, Tyler DE, McDougall AJ (1993). "Spectral analysis for categorical time series: Scaling and the spectral envelope." *Biometrika*, **80**(3), 611–622.

Examples

```
binarized_series <- binarization(GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')))
se <- spectral_envelope(binarized_series = binarized_series) # Representing the spectral envelope
# for the first series in dataset GeneticSequences
spectral_quantities <- spectral_envelope(binarized_series = binarized_series,
plot = FALSE) # Computing the corresponding
# spectral quantities
```

SyntheticData1 *SyntheticData1*

Description

Synthetic dataset containing 80 MTS generated from four different generating processes.

Usage

```
data(SyntheticData1)
```

Format

A `list` with two elements, which are:

`data` A list with 80 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a CTS of length 600 containing three different categories. Series 1-20, 21-40, 41-60 and 61-80 were generated from Markov Chains with different matrices of transition probabilities. Therefore, there are 4 different classes in the dataset.

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

SyntheticData2 *SyntheticData2*

Description

Synthetic dataset containing 80 MTS generated from four different generating processes.

Usage

```
data(SyntheticData2)
```

Format

A `list` with two elements, which are:

`data` A list with 80 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a CTS of length 600 containing three different categories. Series 1-20, 21-40, 41-60 and 61-80 were generated from Hidden Markov Models with different matrices of transition and emission probabilities. Therefore, there are 4 different classes in the dataset.

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

SyntheticData3 *SyntheticData3*

Description

Synthetic dataset containing 80 MTS generated from four different generating processes.

Usage

```
data(SyntheticData3)
```

Format

A `list` with two elements, which are:

`data` A list with 80 MTS.

`classes` A numeric vector indicating the corresponding classes associated with the elements in `data`.

Details

Each element in `data` is a CTS of length 600 containing three different categories. Series 1-20, 21-40, 41-60 and 61-80 were generated from NDARMA processes with different orders and vectors of coefficients. Therefore, there are 4 different classes in the dataset.

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

`total_correlation` *Computes the total correlation of a categorical time series*

Description

`total_correlation` returns the value of the total correlation for a categorical time series

Usage

```
total_correlation(series, lag = 1, categories, features = FALSE)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.
`features`	Logical. If `features = FALSE` (default), the value of the total correlation is returned. Otherwise, the function returns a matrix with the individual components of the total correlation

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, and the binarized time series, which is defined as $\overline{\boldsymbol{Y}}_t = \{\overline{\boldsymbol{Y}}_1, \ldots, \overline{\boldsymbol{Y}}_T\}$, with $\overline{\boldsymbol{Y}}_k = (\overline{Y}_{k,1}, \ldots, \overline{Y}_{k,r})^\top$ such that $\overline{Y}_{k,i} = 1$ if $\overline{X}_k = i$ ($k = 1, \ldots, T, , i = 1, \ldots, r$), the function computes the estimated sum $\widehat{\Psi}(l) = \frac{1}{r^2}\sum_{i,j=1}^{r} \widehat{\psi}_{ij}(l)^2$, where $\widehat{\psi}_{ij}(l)$ is the estimated correlation $\widehat{Corr}(Y_{t,i}, Y_{t-l,j})$, $i, j = 1, \ldots, r$. If `features = TRUE`, the function returns a matrix whose components are the quantities $\widehat{\psi}_{ij}(l), i, j = 1, 2, \ldots, r$.

Value

If `features = FALSE` (default), returns the value of the total correlation. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the total correlation.

Author(s)

Ángel López-Oriona, José A. Vilar

References

López-Oriona Á, Vilar JA, D'Urso P (2023). "Hard and soft clustering of categorical time series based on two novel distances with an application to biological sequences." *Information Sciences*, **624**, 467–492.

Examples

```
tc <- total_correlation(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the total correlation
# for the first series in dataset GeneticSequences
feature_matrix <- total_correlation(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

---

total_mixed_correlation_1

Computes the total mixed l-correlation between a categorical and a real-valued time series

---

Description

`total_mixed_correlation_1` returns the total mixed l-correlation between a categorical and a real-valued time series

Usage

```
total_mixed_correlation_1(
  c_series,
  n_series,
  lag = 1,
  categories,
  features = FALSE
)
```

Arguments

c_series	A CTS.
n_series	A real-valued time series.
lag	The considered lag (default is 1).
categories	A vector of type factor containing the corresponding categories for the CTS.
features	Logical. If features = FALSE (default), the value of the total l-correlation is returned. Otherwise, the function returns a vector with the individual components of the total l-correlation

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, and the binarized time series, which is defined as $\overline{\boldsymbol{Y}}_t = \{\overline{\boldsymbol{Y}}_1, \ldots, \overline{\boldsymbol{Y}}_T\}$, with $\overline{\boldsymbol{Y}}_k = (\overline{Y}_{k,1}, \ldots, \overline{Y}_{k,r})^\top$ such that $\overline{Y}_{k,i} = 1$ if $\overline{X}_k = i$ $(k = 1, \ldots, T, , i = 1, \ldots, r)$, the function computes the estimated total mixed l-correlation given by

$$\widehat{\Psi}_1(l) = \frac{1}{r} \sum_{i=1}^{r} \widehat{\psi}_i(l)^2,$$

where $\widehat{\psi}_i(l) = \widehat{Corr}(Y_{t,i}, Z_{t-l})$, with $\overline{Z}_t = \{\overline{Z}_1, \ldots, \overline{Z}_T\}$ being a T-length real-valued time series. If features = TRUE, the function returns a vector whose components are the quantities $\widehat{\psi}_i(l)$, $i = 1, 2, \ldots, r$.

Value

If features = FALSE (default), returns the value of the total l-correlation. Otherwise, the function returns a vector of features, i.e., the vector contains the features employed to compute the total l-correlation.

Author(s)

Ángel López-Oriona, José A. Vilar

Examples

```
tmc1 <- total_mixed_correlation_1(c_series = SyntheticData1$data[[1]],
n_series = rnorm(600), categories = c('1', '2', '3')) # Computing the total mixed l-correlation
# between the first series in dataset SyntheticData1 and white noise
feature_vector <- total_mixed_correlation_1(c_series = SyntheticData1$data[[1]],
n_series = rnorm(600), categories = c('1', '2', '3'), features = TRUE) # Computing the corresponding
# vector of features
```

`total_mixed_correlation_2` *Computes the total mixed q-correlation between a categorical and a real-valued time series*

Description

`total_mixed_correlation_2` returns the total mixed q-correlation between a categorical and a real-valued time series

Usage

```
total_mixed_correlation_2(
  c_series,
  n_series,
  lag = 1,
  categories,
  features = FALSE
)
```

Arguments

`c_series`	A CTS.
`n_series`	A real-valued time series.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories for the CTS.
`features`	Logical. If `features = FALSE` (default), the value of the total q-correlation is returned. Otherwise, the function returns a vector with the individual components of the total l-correlation

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, and the binarized time series, which is defined as $\overline{\boldsymbol{Y}}_t = \{\overline{\boldsymbol{Y}}_1, \ldots, \overline{\boldsymbol{Y}}_T\}$, with $\overline{\boldsymbol{Y}}_k = (\overline{Y}_{k,1}, \ldots, \overline{Y}_{k,r})^\top$ such that $\overline{Y}_{k,i} = 1$ if $\overline{X}_k = i$ ($k = 1, \ldots, T, , i = 1, \ldots, r$), the function computes the estimated total mixed q-correlation given by

$$\widehat{\Psi}_2(l) = \frac{1}{r} \sum_{i=1}^{r} \int_0^1 \widehat{\psi}_i^{\rho}(l)^2 d\rho,$$

where $\widehat{\psi}_i^{\rho}(l) = \widehat{Corr}\big(Y_{t,i}, I(Z_{t-l} \leq q_{Z_t}(\rho))\big)$, with $\overline{Z}_t = \{\overline{Z}_1, \ldots, \overline{Z}_T\}$ being a T-length real-valued time series, $\rho \in (0, 1)$ a probability level, $I(\cdot)$ the indicator function and q_{Z_t} the quantile function of the corresponding real-valued process. If `features = TRUE`, the function returns a vector whose components are the quantities $\int_0^1 \widehat{\psi}_i^{\rho}(l)^2 d\rho$, $i = 1, 2, \ldots, r$.

Value

If `features = FALSE` (default), returns the value of the total q-correlation. Otherwise, the function returns a vector of features, i.e., the vector contains the features employed to compute the total q-correlation.

Author(s)

Ángel López-Oriona, José A. Vilar

Examples

```
tmc2 <- total_mixed_correlation_2(c_series = SyntheticData1$data[[1]],
n_series = rnorm(600),
categories = c('1', '2', '3')) # Computing the total mixed q-correlation
# between the first series in dataset SyntheticData1 and white noise
feature_vector <- total_mixed_correlation_2(c_series = SyntheticData1$data[[1]],
n_series = rnorm(600),
categories = c('1', '2', '3'), features = TRUE) # Computing the corresponding
# vector of features
```

---

uncertainty_coefficient

Computes the uncertainty coefficient of a categorical time series

---

Description

`uncertainty_coefficient` returns the value of the uncertainty coefficient for a categorical time series

Usage

```
uncertainty_coefficient(series, lag = 1, categories, features = FALSE)
```

Arguments

`series`	A CTS.
`lag`	The considered lag (default is 1).
`categories`	A vector of type factor containing the corresponding categories.
`features`	Logical. If `features = FALSE` (default), the value of the uncertainty coefficient is returned. Otherwise, the function returns a matrix with the individual components of the uncertainty coefficient.

Details

Given a CTS of length T with range $\mathcal{V} = \{1, 2, \ldots, r\}$, $\overline{X}_t = \{\overline{X}_1, \ldots, \overline{X}_T\}$, the function computes the estimated uncertainty coefficient, $\widehat{u}(l) = -\frac{\sum_{i,j=1}^{r} \widehat{p}_{ij}(l) \ln\left(\frac{\widehat{p}_{ij}(l)}{\widehat{p}_i \widehat{p}_j}\right)}{\sum_{i=1}^{r} \widehat{p}_i \ln \widehat{p}_i}$, where $\widehat{p}_i$ is the natural estimate of the marginal probability of the ith category, and $\widehat{p}_{ij}(l)$ is the natural estimate of the joint probability for categories i and j at lag l, $i, j = 1, \ldots, r$. If `features = TRUE`, the function returns a matrix whose components are the quantities $\widehat{p}_{ij}(l) \ln\left(\frac{\widehat{p}_{ij}(l)}{\widehat{p}_i \widehat{p}_j}\right)$, $i, j = 1, 2, \ldots, r$.

Value

If `features = FALSE` (default), returns the value of the uncertainty coefficient. Otherwise, the function returns a matrix of features, i.e., the matrix contains the features employed to compute the uncertainty coefficient.

Author(s)

Ángel López-Oriona, José A. Vilar

References

Weiß CH, Göb R (2008). "Measuring serial dependence in categorical time series." *AStA Advances in Statistical Analysis*, **92**, 71–89.

Examples

```
uc <- uncertainty_coefficient(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't'))) # Computing the uncertainty coefficient
# for the first series in dataset GeneticSequences
feature_matrix <- uncertainty_coefficient(series = GeneticSequences$data[[1]],
categories = factor(c('a', 'c', 'g', 't')), features = TRUE) # Computing the corresponding
# matrix of features
```

Index