# Package ‘TSrepr’

January 20, 2025

**Type** Package

**Title** Time Series Representations

**Version** 1.1.0

**Date** 2020-07-12

**Description** Methods for representations (i.e. dimensionality reduction, preprocessing, feature extraction) of time series to help more accurate and effective time series data mining.
Non-data adaptive, data adaptive, model-based and data dictated (clipped) representation methods are implemented. Also various normalisation methods (min-max, z-score, Box-Cox, Yeo-Johnson),
and forecasting accuracy measures are implemented.

**License** GPL-3 | file LICENSE

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 2.10)

**Imports** Rcpp (>= 0.12.12), MASS, quantreg, wavelets, mgcv, dtt

**LinkingTo** Rcpp

**RoxygenNote** 7.1.0

**URL** https://petolau.github.io/package/,
https://github.com/PetoLau/TSrepr/

**BugReports** https://github.com/PetoLau/TSrepr/issues

**Suggests** knitr, rmarkdown, ggplot2, data.table, moments, testthat

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** Peter Laurinec [aut, cre] (<https://orcid.org/0000-0002-3501-8783>)

**Maintainer** Peter Laurinec <tsreprpackage@gmail.com>

**Repository** CRAN

**Date/Publication** 2020-07-13 06:50:15 UTC

# Contents

---

`clipping` *Creates bit-level (clipped representation) from a vector*

---

### Description

The `clipping` computes bit-level (clipped representation) from a vector.

### Usage

```
clipping(x)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |

### Details

Clipping transforms time series to bit-level representation.

It is defined as follows:

$$repr_t = 1 if x_t > \mu, 0 otherwise,$$

where $x_t$ is a value of a time series and $\mu$ is average of a time series.

### Value

the integer vector of zeros and ones

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Bagnall A, Ratanamahatana C, Keogh E, Lonardi S, Janacek G (2006) A bit level representation for time series data mining with shape based similarity. Data Mining and Knowledge Discovery 13(1):11-40

Laurinec P, and Lucka M (2018) Interpretable multiple data streams clustering with clipped streams representation for the improvement of electricity consumption forecasting. Data Mining and Knowledge Discovery. Springer. DOI: 10.1007/s10618-018-0598-2

### See Also

`trending`

## Examples

```
clipping(rnorm(50))
```

---

`coefComp` *Functions for linear regression model coefficients extraction*

---

## Description

The functions computes regression coefficients from a linear model.

## Usage

```
lmCoef(X, Y)

rlmCoef(X, Y)

l1Coef(X, Y)
```

## Arguments

| | |
|---|---|
| X | the model (design) matrix of independent variables |
| Y | the vector of dependent variable (time series) |

## Value

The numeric vector of regression coefficients

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

lm, rlm, rq

## Examples

```
design_matrix <- matrix(rnorm(10), ncol = 2)
lmCoef(design_matrix, rnorm(5))

rlmCoef(design_matrix, rnorm(5))

l1Coef(design_matrix, rnorm(5))
```

## denorm_atan *Arctangent denormalisation*

### Description

The denorm_atan denormalises time series from Arctangent function.

### Usage

```
denorm_atan(x)
```

### Arguments

| | |
|---|---|
| x | the numeric vector (time series) |

### Value

the numeric vector of denormalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

denorm_z, denorm_min_max

### Examples

```
denorm_atan(runif(50))
```

## denorm_boxcox *Two-parameter Box-Cox denormalisation*

### Description

The denorm_boxcox denormalises time series by two-parameter Box-Cox method.

### Usage

```
denorm_boxcox(x, lambda = 0.1, gamma = 0)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) to be denormalised |
| `lambda` | the numeric value - power transformation parameter (default is 0.1) |
| `gamma` | the non-negative numeric value - parameter for holding the time series positive (offset) (default is 0) |

### Value

the numeric vector of denormalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`denorm_z`, `denorm_min_max`, `denorm_atan`

### Examples

```
denorm_boxcox(runif(50))
```

---

## denorm_min_max *Min-Max denormalisation*

---

### Description

The `denorm_min_max` denormalises time series by min-max method.

### Usage

```
denorm_min_max(x, min, max)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `min` | the minimum value |
| `max` | the maximal value |

### Value

the numeric vector of denormalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, Lucká M (2018) Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Comput Sci, 8(1):38–50, DOI: 10.1515/comp-2018-0006

### See Also

`norm_min_max`, `norm_min_max_list`

### Examples

```
# Normalise values and save normalisation parameters:
norm_res <- norm_min_max_list(rnorm(50, 5, 2))
# Denormalise new data with previous computed parameters:
denorm_min_max(rnorm(50, 4, 2), min = norm_res$min, max = norm_res$max)
```

---

## `denorm_yj` *Yeo-Johnson denormalisation*

---

### Description

The `denorm_yj` denormalises time series by Yeo-Johnson method

### Usage

```
denorm_yj(x, lambda = 0.1)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) to be denormalised |
| `lambda` | the numeric value - power transformation parameter (default is 0.1) |

### Value

the numeric vector of denormalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`denorm_z`, `denorm_min_max`, `denorm_boxcox`

### Examples

```
denorm_yj(runif(50))
```

denorm_z *Z-score denormalisation*

## Description

The denorm_z denormalises time series by z-score method.

## Usage

```
denorm_z(x, mean, sd)
```

## Arguments

| | |
|---|---|
| x | the numeric vector (time series) |
| mean | the mean value |
| sd | the standard deviation value |

## Value

the numeric vector of denormalised values

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## References

Laurinec P, Lucká M (2018) Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Comput Sci, 8(1):38–50, DOI: 10.1515/comp-2018-0006

## See Also

norm_z, norm_z_list

## Examples

```
# Normalise values and save normalisation parameters:
norm_res <- norm_z_list(rnorm(50, 5, 2))
# Denormalise new data with previous computed parameters:
denorm_z(rnorm(50, 4, 2), mean = norm_res$mean, sd = norm_res$sd)
```

## `elec_load` — *2 weeks of electricity load data from 50 consumers.*

### Description

A dataset containing the electricity consumption time series from 50 consumers of the length of 2 weeks. Every day is 48 measurements (half-hourly data). Each row represents one consumers time series.

### Usage

```
elec_load
```

### Format

A data frame with 50 rows and 672 variables.

### Source

Anonymized.

## `fast_stat` — *Fast statistic functions (helpers)*

### Description

Fast statistic functions (helpers) for representations computation.

### Usage

```
maxC(x)

minC(x)

meanC(x)

sumC(x)

medianC(x)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector |

### Value

the numeric value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
maxC(rnorm(50))

minC(rnorm(50))

meanC(rnorm(50))

sumC(rnorm(50))

medianC(rnorm(50))
```

---

## maape *MAAPE*

---

### Description

the maape computes MAAPE (Mean Arctangent Absolute Percentage Error) of a forecast.

### Usage

```
maape(x, y)
```

### Arguments

| | |
|---|---|
| x | the numeric vector of real values |
| y | the numeric vector of forecasted values |

### Value

the numeric value in %

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Sungil Kim, Heeyoung Kim (2016) A new metric of absolute percentage error for intermittent demand forecasts, International Journal of Forecasting 32(3):669-679

### Examples

```
maape(runif(50), runif(50))
```

`mae` *MAE*

### Description

The `mae` computes MAE (Mean Absolute Error) of a forecast.

### Usage

```
mae(x, y)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector of real values |
| `y` | the numeric vector of forecasted values |

### Value

the numeric value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
mae(runif(50), runif(50))
```

`mape` *MAPE*

### Description

the `mape` computes MAPE (Mean Absolute Percentage Error) of a forecast.

### Usage

```
mape(x, y)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector of real values |
| `y` | the numeric vector of forecasted values |

### Value

the numeric value in %

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
mape(runif(50), runif(50))
```

---

## mase *MASE*

---

### Description

The `mase` computes MASE (Mean Absolute Scaled Error) of a forecast.

### Usage

```
mase(real, forecast, naive)
```

### Arguments

| | |
|---|---|
| `real` | the numeric vector of real values |
| `forecast` | the numeric vector of forecasted values |
| `naive` | the numeric vector of naive forecast |

### Value

the numeric value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
mase(rnorm(50), rnorm(50), rnorm(50))
```

## mdae *MdAE*

### Description

The `mdae` computes MdAE (Median Absolute Error) of a forecast.

### Usage

```
mdae(x, y)
```

### Arguments

| | |
|---|---|
| x | the numeric vector of real values |
| y | the numeric vector of forecasted values |

### Value

the numeric value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
mdae(runif(50), runif(50))
```

## mse *MSE*

### Description

The `mse` computes MSE (Mean Squared Error) of a forecast.

### Usage

```
mse(x, y)
```

### Arguments

| | |
|---|---|
| x | the numeric vector of real values |
| y | the numeric vector of forecasted values |

### Value

the numeric value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
mse(runif(50), runif(50))
```

---

## norm_atan *Arctangent normalisation*

---

### Description

The `norm_atan` normalises time series by Arctangent to max (-1,1) range.

### Usage

```
norm_atan(x)
```

### Arguments

| | |
|---|---|
| x | the numeric vector (time series) |

### Value

the numeric vector of normalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`norm_z`, `norm_min_max`

### Examples

```
norm_atan(rnorm(50))
```

## norm_boxcox *Two-parameter Box-Cox normalisation*

### Description

The `norm_boxcox` normalises time series by two-parameter Box-Cox normalisation.

### Usage

```
norm_boxcox(x, lambda = 0.1, gamma = 0)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `lambda` | the numeric value - power transformation parameter (default is 0.1) |
| `gamma` | the non-negative numeric value - parameter for holding the time series positive (offset) (default is 0) |

### Value

the numeric vector of normalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`norm_z`, `norm_min_max`, `norm_atan`

### Examples

```
norm_boxcox(runif(50))
```

## norm_min_max *Min-Max normalisation*

### Description

The `norm_min_max` normalises time series by min-max method.

### Usage

```
norm_min_max(x)
```

### Arguments

`x` the numeric vector (time series)

### Value

the numeric vector of normalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`norm_z`

### Examples

```
norm_min_max(rnorm(50))
```

---

## `norm_min_max_list` *Min-Max normalization list*

---

### Description

The `norm_min_max_list` normalises time series by min-max method and returns normalization parameters (min and max).

### Usage

```
norm_min_max_list(x)
```

### Arguments

`x` the numeric vector (time series)

### Value

the list composed of:

**norm_values** the numeric vector of normalised values of time series

**min** the min value

**max** the max value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

norm_z_list

## Examples

```
norm_min_max_list(rnorm(50))
```

---

norm_min_max_params *Min-Max normalisation with parameters*

---

## Description

The norm_min_max_params normalises time series by min-max method with defined parameters.

## Usage

```
norm_min_max_params(x, min, max)
```

## Arguments

| | |
|---|---|
| x | the numeric vector (time series) |
| min | the numeric value |
| max | the numeric value |

## Value

the numeric vector of normalised values

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

norm_z_params

## Examples

```
norm_min_max_params(rnorm(50), 0, 1)
```

## norm_yj *Yeo-Johnson normalisation*

### Description

The `norm_yj` normalises time series by Yeo-Johnson normalisation.

### Usage

```
norm_yj(x, lambda = 0.1)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `lambda` | the numeric value - power transformation parameter (default is 0.1) |

### Value

the numeric vector of normalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`norm_z`, `norm_min_max`, `norm_boxcox`

### Examples

```
norm_yj(runif(50))
```

## norm_z *Z-score normalisation*

### Description

The `norm_z` normalises time series by z-score.

### Usage

```
norm_z(x)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |

## Value

the numeric vector of normalised values

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

norm_min_max

## Examples

```
norm_z(runif(50))
```

---

norm_z_list *Z-score normalization list*

---

## Description

The norm_z_list normalizes time series by z-score and returns normalization parameters (mean and standard deviation).

## Usage

```
norm_z_list(x)
```

## Arguments

x the numeric vector (time series)

## Value

the list composed of:

**norm_values** the numeric vector of normalised values of time series

**mean** the mean value

**sd** the standard deviation

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

norm_min_max_list

### Examples

```
norm_z_list(runif(50))
```

---

## norm_z_params — *Z-score normalisation with parameters*

---

### Description

The `norm_z_params` normalises time series by z-score with defined mean and standard deviation.

### Usage

```
norm_z_params(x, mean, sd)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `mean` | the numeric value |
| `sd` | the numeric value - standard deviation |

### Value

the numeric vector of normalised values

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

`norm_min_max_params`

### Examples

```
norm_z_params(runif(50), 0.5, 1)
```

## repr_dct *DCT representation*

### Description

The repr_dct computes DCT (Discrete Cosine Transform) representation from a time series.

### Usage

```
repr_dct(x, coef = 10)
```

### Arguments

| | |
|---|---|
| x | the numeric vector (time series) |
| coef | the number of coefficients to extract from DCT |

### Details

The length of the final time series representation is equal to set coef parameter.

### Value

the numeric vector of DCT coefficients

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

repr_dft, repr_dwt, dtt

### Examples

```
repr_dct(rnorm(50), coef = 4)
```

`repr_dft` *DFT representation by FFT*

## Description

The `repr_dft` computes DFT (Discrete Fourier Transform) representation from a time series by FFT (Fast Fourier Transform).

## Usage

```
repr_dft(x, coef = 10)
```

## Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `coef` | the number of coefficients to extract from FFT |

## Details

The length of the final time series representation is equal to set `coef` parameter.

## Value

the numeric vector of DFT coefficients

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

`repr_dwt`, `repr_dct`, `fft`

## Examples

```
repr_dft(rnorm(50), coef = 4)
```

`repr_dwt` *DWT representation*

## Description

The `repr_dwt` computes DWT (Discrete Wavelet Transform) representation (coefficients) from a time series.

## Usage

```
repr_dwt(x, level = 4, filter = "d4")
```

## Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `level` | the level of DWT transformation (default is 4) |
| `filter` | the filter name (default is "d6"). Can be: "haar", "d4", "d6", ..., "d20", "la8", "la10", ..., "la20", "bl14", "bl18", "bl20", "c6", "c12", ..., "c30". See more info at `wt.filter`. |

## Details

This function extracts DWT coefficients. You can use various wavelet filters, see all of them here `wt.filter`. The number of extracted coefficients depends on the `level` selected. The final representation has length equal to floor(n / 2^level), where n is a length of original time series.

## Value

the numeric vector of DWT coefficients

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## References

Laurinec P, Lucka M (2016) Comparison of representations of time series for clustering smart meter data. In: Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2016, pp 458-463

## See Also

`repr_dft`, `repr_dct`, `dwt`

### Examples

```
# Interpretation: DWT with Daubechies filter of length 4 and
# 3rd level of DWT coefficients extracted.
repr_dwt(rnorm(50), filter = "d4", level = 3)
```

---

## repr_exp — *Exponential smoothing seasonal coefficients as representation*

---

### Description

The `repr_exp` computes exponential smoothing seasonal coefficients.

### Usage

```
repr_exp(x, freq, alpha = TRUE, gamma = TRUE)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `freq` | the frequency of the time series |
| `alpha` | the smoothing factor (default is TRUE - automatic determination of smoothing factor), or number between 0 to 1 |
| `gamma` | the seasonal smoothing factor (default is TRUE - automatic determination of seasonal smoothing factor), or number between 0 to 1 |

### Details

This function extracts exponential smoothing seasonal coefficients and uses them as time series representation. You can set smoothing factors (`alpha, gamma`) manually, but recommended is automatic method (set to `TRUE`). The trend component is not included in computations.

### Value

the numeric vector of seasonal coefficients

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, Lucka M (2016) Comparison of representations of time series for clustering smart meter data. In: Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2016, pp 458-463

Laurinec P, Loderer M, Vrablecova P, Lucka M, Rozinajova V, Ezzeddine AB (2016) Adaptive time series forecasting of energy consumption using optimized cluster analysis. In: Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on, IEEE, pp 398-405

### See Also

repr_lm, repr_gam, repr_seas_profile,HoltWinters

### Examples

```
repr_exp(rnorm(96), freq = 24)
```

---

## repr_feaclip — *FeaClip representation of time series*

---

### Description

The repr_feaclip computes representation of time series based on feature extraction from bit-level (clipped) representation.

### Usage

```
repr_feaclip(x)
```

### Arguments

| | |
|---|---|
| x | the numeric vector (time series) |

### Details

FeaClip is method of time series representation based on feature extraction from run lengths (RLE) of bit-level (clipped) representation. It extracts 8 key features from clipped representation.

There are as follows:

$$repr = \{max_1 - max.fromrunlengthsofones,$$

$$sum_1 - sumofrunlengthsofones,$$

$$max_0 - max.fromrunlengthsofzeros,$$

$$crossings - lengthofRLEencoding - 1,$$

$$f_0 - numberoffirstzeros,$$

$$l_0 - numberoflastzeros,$$

$$f_1 - numberoffirstones,$$

$$l_1 - numberoflastones\}.$$

### Value

the numeric vector of length 8

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, and Lucka M (2018) Interpretable multiple data streams clustering with clipped streams representation for the improvement of electricity consumption forecasting. Data Mining and Knowledge Discovery. Springer. DOI: 10.1007/s10618-018-0598-2

### See Also

repr_featrend, repr_feacliptrend

### Examples

```
repr_feaclip(rnorm(50))
```

---

## repr_feacliptrend *FeaClipTrend representation of time series*

---

### Description

The `repr_feacliptrend` computes representation of time series based on feature extraction from bit-level representations (clipping and trending).

### Usage

```
repr_feacliptrend(x, func, pieces = 2L, order = 4L)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `func` | the aggregation function for FeaTrend procedure (sumC or maxC) |
| `pieces` | the number of parts of time series to split |
| `order` | the order of simple moving average |

### Details

FeaClipTrend combines FeaClip and FeaTrend representation methods. See documentation of these two methods (check See Also section).

### Value

the numeric vector of frequencies of features

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, and Lucka M (2018) Interpretable multiple data streams clustering with clipped streams representation for the improvement of electricity consumption forecasting. Data Mining and Knowledge Discovery. Springer. DOI: 10.1007/s10618-018-0598-2

### See Also

repr_featrend, repr_feaclip

### Examples

```
repr_feacliptrend(rnorm(50), maxC, 2, 4)
```

---

## repr_featrend *FeaTrend representation of time series*

---

### Description

The `repr_featrend` computes representation of time series based on feature extraction from bit-level (trending) representation.

### Usage

```
repr_featrend(x, func, pieces = 2L, order = 4L)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `func` | the function of aggregation, can be sumC or maxC or similar aggregation function |
| `pieces` | the number of parts of time series to split (default to 2) |
| `order` | the order of simple moving average (default to 4) |

### Details

FeaTrend is method of time series representation based on feature extraction from run lengths (RLE) of bit-level (trending) representation. It extracts number of features from trending representation based on number of pieces defined. From every piece, 2 features are extracted. You can define what feature will be extracted, recommended functions are max and sum. For example if max is selected, then maximum value of run lengths of ones and zeros are extracted.

### Value

the numeric vector of the length pieces

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

repr_feaclip, repr_feacliptrend

### Examples

```
# default settings
repr_featrend(rnorm(50), maxC)

# compute FeaTrend for 4 pieces and make more smoothed ts by order = 8
repr_featrend(rnorm(50), sumC, 4, 8)
```

---

## repr_gam *GAM regression coefficients as representation*

---

### Description

The `repr_gam` computes seasonal GAM regression coefficients. Additional exogenous variables can be also added.

### Usage

```
repr_gam(x, freq = NULL, xreg = NULL)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `freq` | the frequency of the time series. Can be vector of two frequencies (seasonalities) or just an integer of one frequency. |
| `xreg` | the numeric vector or the data.frame with additional exogenous regressors |

### Details

This model-based representation method extracts regression coefficients from a GAM (Generalized Additive Model). The extraction of seasonal regression coefficients is automatic. The maximum number of seasonalities is 2 so it is possible to compute representation for double-seasonal time series. The first set seasonality (frequency) is main, so for example if we have hourly time series (`freq = c(24, 24*7)`), the number of extracted daily seasonal coefficients is 24 and the number of weekly seasonal coefficients is 7, because the length of second seasonality representation is always freq_1 / freq_2. The smooth function for seasonal variables is set to cubic regression spline. There is also possibility to add another independent variables (`xreg`).

### Value

the numeric vector of GAM regression coefficients

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, Lucka M (2016) Comparison of representations of time series for clustering smart meter data. In: Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2016, pp 458-463

Laurinec P, Loderer M, Vrablecova P, Lucka M, Rozinajova V, Ezzeddine AB (2016) Adaptive time series forecasting of energy consumption using optimized cluster analysis. In: Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on, IEEE, pp 398-405

Laurinec P, Lucká M (2018) Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Comput Sci, 8(1):38–50, DOI: 10.1515/comp-2018-0006

### See Also

`repr_lm`, `repr_exp`, `gam`

### Examples

```
repr_gam(rnorm(96), freq = 24)
```

---

`repr_list` — *Computation of list of representations list of time series with different lengths*

---

### Description

The `repr_list` computes list of representations from list of time series

### Usage

```
repr_list(
  x,
  func = NULL,
  args = NULL,
  normalise = FALSE,
  func_norm = norm_z,
  windowing = FALSE,
  win_size = NULL
)
```

### Arguments

| | |
|---|---|
| x | the list of time series, where time series can have different lengths |
| func | the function that computes representation |
| args | the list of additional (or required) parameters of func (function that computes representation) |
| normalise | normalise (scale) time series before representations computation? (default is FALSE) |
| func_norm | the normalisation function (default is norm_z) |
| windowing | perform windowing? (default is FALSE) |
| win_size | the size of the window |

### Details

This function computes representation to an every member of a list of time series (that can have different lengths) and returns list of time series representations. It can be combined with windowing (see repr_windowing) and normalisation of time series.

### Value

the numeric list of representations of time series

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

repr_windowing, repr_matrix

### Examples

```
# Create random list of time series with different lengths
list_ts <- list(rnorm(sample(8:12, 1)), rnorm(sample(8:12, 1)), rnorm(sample(8:12, 1)))
repr_list(list_ts, func = repr_sma,
 args = list(order = 3))

# return normalised representations, and normalise time series by min-max normalisation
repr_list(list_ts, func = repr_sma,
 args = list(order = 3), normalise = TRUE, func_norm = norm_min_max)
```

## repr_lm — *Regression coefficients from linear model as representation*

### Description

The repr_lm computes seasonal regression coefficients from a linear model. Additional exogenous variables can be also added.

### Usage

```
repr_lm(x, freq = NULL, method = "lm", xreg = NULL)
```

### Arguments

| | |
|---|---|
| x | the numeric vector (time series) |
| freq | the frequency of the time series. Can be vector of two frequencies (seasonalities) or just an integer of one frequency. |
| method | the linear regression method to use. It can be "lm", "rlm" or "l1". |
| xreg | the data.frame with additional exogenous regressors or the single numeric vector |

### Details

This model-based representation method extracts regression coefficients from a linear model. The extraction of seasonal regression coefficients is automatic. The maximum number of seasonalities is 2 so it is possible to compute representation for double-seasonal time series. The first set seasonality (frequency) is main, so for example if we have hourly time series (freq = c(24, 24*7)), the number of extracted daily seasonal coefficients is 24 and the number of weekly seasonal coefficients is 7, because the length of second seasonality representation is always freq_1 / freq_2. There is also possibility to add another independent variables (xreg).

You have three possibilities for selection of a linear model method.

- "lm" is classical OLS regression.
- "rlm" is robust linear model using psi huber function and is implemented in MASS package.
- "l1" is L1 quantile regression model (also robust linear regression method) implemented in package quantreg.

### Value

the numeric vector of regression coefficients

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## References

Laurinec P, Lucka M (2016) Comparison of representations of time series for clustering smart meter data. In: Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2016, pp 458-463

Laurinec P, Loderer M, Vrablecova P, Lucka M, Rozinajova V, Ezzeddine AB (2016) Adaptive time series forecasting of energy consumption using optimized cluster analysis. In: Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on, IEEE, pp 398-405

Laurinec P, Lucká M (2018) Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Comput Sci, 8(1):38–50, DOI: 10.1515/comp-2018-0006

## See Also

repr_gam, repr_exp

## Examples

```
# Extracts 24 seasonal regression coefficients from the time series by linear model
repr_lm(rnorm(96), freq = 24, method = "lm")

# Try also robust linear models ("rlm" and "l1")
repr_lm(rnorm(96), freq = 24, method = "rlm")
repr_lm(rnorm(96), freq = 24, method = "l1")
```

---

# repr_matrix — *Computation of matrix of representations from matrix of time series*

---

## Description

The `repr_matrix` computes matrix of representations from matrix of time series

## Usage

```
repr_matrix(
  x,
  func = NULL,
  args = NULL,
  normalise = FALSE,
  func_norm = norm_z,
  windowing = FALSE,
  win_size = NULL
)
```

### Arguments

| | |
|---|---|
| x | the matrix, data.frame or data.table of time series, where time series are in rows of the table |
| func | the function that computes representation |
| args | the list of additional (or required) parameters of func (function that computes representation) |
| normalise | normalise (scale) time series before representations computation? (default is FALSE) |
| func_norm | the normalisation function (default is norm_z) |
| windowing | perform windowing? (default is FALSE) |
| win_size | the size of the window |

### Details

This function computes representation to an every row of a matrix of time series and returns matrix of time series representations. It can be combined with windowing (see repr_windowing) and normalisation of time series.

### Value

the numeric matrix of representations of time series

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### See Also

repr_windowing, repr_list

### Examples

```
# Create random matrix of time series
mat_ts <- matrix(rnorm(100), ncol = 10)
repr_matrix(mat_ts, func = repr_paa,
 args = list(q = 5, func = meanC))

# return normalised representations, and normalise time series by min-max normalisation
repr_matrix(mat_ts, func = repr_paa,
 args = list(q = 2, func = meanC), normalise = TRUE, func_norm = norm_min_max)

# with windowing
repr_matrix(mat_ts, func = repr_feaclip, windowing = TRUE, win_size = 5)
```

`repr_paa` *PAA - Piecewise Aggregate Approximation*

### Description

The `repr_paa` computes PAA representation from a vector.

### Usage

```
repr_paa(x, q, func)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `q` | the integer of the length of the "piece" |
| `func` | the aggregation function. Can be meanC, medianC, sumC, minC or maxC or similar aggregation function |

### Details

PAA with possibility to use arbitrary aggregation function. The original method uses average as aggregation function.

### Value

the numeric vector

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Keogh E, Chakrabarti K, Pazzani M, Mehrotra Sh (2001) Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. Knowledge and Information Systems 3(3):263-286

### See Also

`repr_dwt`, `repr_dft`, `repr_dct`, `repr_sma`

### Examples

```
repr_paa(rnorm(11), 2, meanC)
```

## repr_pip *PIP representation*

### Description

The `repr_pip` computes PIP (Perceptually Important Points) representation from a time series.

### Usage

```
repr_pip(x, times = 10, return = "points")
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `times` | the number of important points to extract (default 10) |
| `return` | what to return? Can be important points ("points"), places of important points in a vector ("places") or "both" (data.frame). |

### Value

the values based on the argument return (see above)

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Fu TC, Chung FL, Luk R, and Ng CM (2008) Representing financial time series based on data point importance. Engineering Applications of Artificial Intelligence, 21(2):277-300

### Examples

```
repr_pip(rnorm(100), times = 12, return = "both")
```

`repr_pla` *PLA representation*

## Description

The `repr_pla` computes PLA (Piecewise Linear Approximation) representation from a time series.

## Usage

```
repr_pla(x, times = 10, return = "points")
```

## Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `times` | the number of important points to extract (default 10) |
| `return` | what to return? Can be "points" (segments), places of points (segments) in a vector ("places") or "both" (data.frame). |

## Value

the values based on the argument return (see above)

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## References

Zhu Y, Wu D, Li Sh (2007) A Piecewise Linear Representation Method of Time Series Based on Feature Points. Knowledge-Based Intelligent Information and Engineering Systems 4693:1066-1072

## Examples

```
repr_pla(rnorm(100), times = 12, return = "both")
```

## repr_sax — *SAX - Symbolic Aggregate Approximation*

### Description

The `repr_sax` creates SAX symbols for a univariate time series.

### Usage

```
repr_sax(x, q = 2, a = 6, eps = 0.01)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `q` | the integer of the length of the "piece" in PAA |
| `a` | the integer of the alphabet size |
| `eps` | is the minimum threshold for variance in x and should be a numeric value. If x has a smaller variance than eps, it will represented as a word using the middle alphabet. |

### Value

the character vector of SAX representation

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Lin J, Keogh E, Lonardi S, Chiu B (2003) A symbolic representation of time series, with implications for streaming algorithms. Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery - DMKD'03

### See Also

`repr_paa`, `repr_pla`

### Examples

```
x <- rnorm(48)
repr_sax(x, q = 4, a = 5)
```

## repr_seas_profile *Mean seasonal profile of time series*

### Description

The `repr_seas_profile` computes mean seasonal profile representation from a time series.

### Usage

```
repr_seas_profile(x, freq, func)
```

### Arguments

| | |
|---|---|
| x | the numeric vector (time series) |
| freq | the integer of the length of the season |
| func | the aggregation function. Can be meanC or medianC or similar aggregation function. |

### Details

This function computes mean seasonal profile representation for a seasonal time series. The length of representation is length of set seasonality (frequency) of a time series. Aggregation function is arbitrary (best choice is for you maybe mean or median).

### Value

the numeric vector

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, Lucka M (2016) Comparison of representations of time series for clustering smart meter data. In: Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2016, pp 458-463

Laurinec P, Loderer M, Vrablecova P, Lucka M, Rozinajova V, Ezzeddine AB (2016) Adaptive time series forecasting of energy consumption using optimized cluster analysis. In: Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on, IEEE, pp 398-405

Laurinec P, Lucká M (2018) Clustering-based forecasting method for individual consumers electricity load using time series representations. Open Comput Sci, 8(1):38–50, DOI: 10.1515/comp-2018-0006

### See Also

`repr_lm`, `repr_gam`, `repr_exp`

### Examples

```
repr_seas_profile(rnorm(48*10), 48, meanC)
```

---

## repr_sma — *Simple Moving Average representation*

---

### Description

The `repr_sma` computes Simple Moving Average (SMA) from a time series.

### Usage

```
repr_sma(x, order)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `order` | the order of simple moving average |

### Value

the numeric vector of smoothed values of the length = length(x) - order + 1

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
repr_sma(rnorm(50), 4)
```

---

## repr_windowing — *Windowing of time series*

---

### Description

The `repr_windowing` computes representations from windows of a vector.

### Usage

```
repr_windowing(x, win_size, func = NULL, args = NULL)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |
| `win_size` | the length of the window |
| `func` | the function for representation computation. For example `repr_feaclip` or `repr_trend`. |
| `args` | the list of additional arguments to the func (representation computation function). The args list must be named. |

### Details

This function applies specified representation method (function) to every non-overlapping window (subsequence, piece) of a time series.

### Value

the numeric vector

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### References

Laurinec P, and Lucka M (2018) Interpretable multiple data streams clustering with clipped streams representation for the improvement of electricity consumption forecasting. Data Mining and Knowledge Discovery. Springer. DOI: 10.1007/s10618-018-0598-2

### See Also

`repr_paa`, `repr_matrix`

### Examples

```
# func without arguments
repr_windowing(rnorm(48), win_size = 24, func = repr_feaclip)

# func with arguments
repr_windowing(rnorm(48), win_size = 24, func = repr_featrend,
 args = list(func = maxC, order = 2, pieces = 2))
```

## rleC — *RLE (Run Length Encoding) written in C++*

### Description

The `rleC` computes RLE from bit-level (clipping or trending representation) vector.

### Usage

```
rleC(x)
```

### Arguments

| | |
|---|---|
| `x` | the integer vector (from `clipping` or `trending`) |

### Value

the list of values and counts of zeros and ones

### Examples

```
# clipping
clipped <- clipping(rnorm(50))
rleC(clipped)
# trending
trended <- trending(rnorm(50))
rleC(trended)
```

## rmse — *RMSE*

### Description

The `rmse` computes RMSE (Root Mean Squared Error) of a forecast.

### Usage

```
rmse(x, y)
```

### Arguments

| | |
|---|---|
| `x` | the numeric vector of real values |
| `y` | the numeric vector of forecasted values |

### Value

the numeric value

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
rmse(runif(50), runif(50))
```

---

smape *sMAPE*

---

### Description

The smape computes sMAPE (Symmetric Mean Absolute Percentage Error) of a forecast.

### Usage

```
smape(x, y)
```

### Arguments

| | |
|---|---|
| x | the numeric vector of real values |
| y | the numeric vector of forecasted values |

### Value

the numeric value in %

### Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

### Examples

```
smape(runif(50), runif(50))
```

`trending` *Creates bit-level (trending) representation from a vector*

## Description

The `trending` Computes bit-level (trending) representation from a vector.

## Usage

```
trending(x)
```

## Arguments

| | |
|---|---|
| `x` | the numeric vector (time series) |

## Details

Trending transforms time series to bit-level representation.

It is defined as follows:

$$repr_t = 1 if x_t - x_{t+1} < 0, 0 otherwise,$$

where $x_t$ is a value of a time series.

## Value

the integer vector of zeros and ones

## Author(s)

Peter Laurinec, <tsreprpackage@gmail.com>

## See Also

`clipping`

## Examples

```
trending(rnorm(50))
```

TSrepr *TSrepr package*

## Description

Package contains methods for time series representations computation. Representation methods of time series are for dimensionality and noise reduction, emphasizing of main characteristics of time series data and speed up of consequent usage of machine learning methods.

## Details

| | |
|---|---|
| Package: | TSrepr |
| Type: | Package |
| Date: | 2018-01-26 - Inf |
| License: | GPL-3 |

The following functions for time series representations are included in the package:

- repr_paa - Piecewise Aggregate Approximation (PAA)
- repr_dwt - Discrete Wavelet Transform (DWT)
- repr_dft - Discrete Fourier Transform (DFT)
- repr_dct - Discrete Cosine Transform (DCT)
- repr_sma - Simple Moving Average (SMA)
- repr_pip - Perceptually Important Points (PIP)
- repr_sax - Symbolic Aggregate Approximation (SAX)
- repr_pla - Piecewise Linear Approximation (PLA)
- repr_seas_profile - Mean seasonal profile
- repr_lm - Model-based seasonal representations based on linear model (lm, rlm, l1)
- repr_gam - Model-based seasonal representations based on generalized additive model (GAM)
- repr_exp - Exponential smoothing seasonal coefficients
- repr_feaclip - Feature extraction from clipping representation (FeaClip)
- repr_featrend - Feature extraction from trending representation (FeaTrend)
- repr_feacliptrend - Feature extraction from clipping and trending representation (FeaClipTrend)

There are also implemented additional useful functions as:

- repr_windowing - applies above mentioned representations to every window of a time series
- repr_matrix - applies above mentioned representations to every row of a matrix of time series
- repr_list - applies above mentioned representations to every member of a list of time series
- norm_z, norm_min_max, norm_boxcox, norm_yj, norm_atan - normalisation functions

- norm_z_params, norm_min_max_params - normalisation functions with defined parameters
- norm_z_list, norm_min_max_list - normalisation functions with output also of scaling parameters
- denorm_z, denorm_min_max, denorm_boxcox, denorm_yj, denorm_atan - denormalisation functions

## Author(s)

Peter Laurinec

Maintainer: Peter Laurinec <tsreprpackage@gmail.com>

# Index