Package ‘harbinger’

February 11, 2026

Title A Unified Time Series Event Detection Framework

Version 1.2.767

Description By analyzing time series, it is possible to observe significant changes in the behavior of observations that frequently characterize events. Events present themselves as anomalies, change points, or motifs. In the literature, there are several methods for detecting events. However, searching for a suitable time series method is a complex task, especially considering that the nature of events is often unknown. This work presents Harbinger, a framework for integrating and analyzing event detection methods. Harbinger contains several state-of-the-art methods described in Salles et al. (2020) <doi:10.5753/sbbd.2020.13626>.

License MIT + file LICENSE

URL `https://cefet-rj-dal.github.io/harbinger/`,
`https://github.com/cefet-rj-dal/harbinger`

BugReports `https://github.com/cefet-rj-dal/harbinger/issues`

Encoding UTF-8

Depends R (>= 4.1.0)

RoxygenNote 7.3.3

Imports tspredit, changepoint, daltoolbox, forecast, ggplot2, hht, RcppHungarian, dplyr, dtwclust, rugarch, stats, stringr, strucchange, tsmp, wavelets, zoo

NeedsCompilation no

Author Eduardo Ogasawara [aut, ths, cre] (ORCID: <`https://orcid.org/0000-0002-0466-0626`>),
Antonio Castro [aut],
Antonio Mello [aut],
Diego Carvalho [ctb],
Eduardo Bezerra [ctb],
Ellen Paixão [aut],
Fernando Fraga [aut],
Heraldo Borges [aut],
Janio Lima [aut],
Jessica Souza [aut],
Lais Baroni [aut],

Lucas Tavares [aut],
Michel Reis [aut],
Rebecca Salles [aut],
CEFET/RJ [cph]

Maintainer Eduardo Ogasawara <eogasawara@ieee.org>

Repository CRAN

Date/Publication 2026-02-11 01:20:02 UTC

Contents

A1Benchmark *Yahoo Webscope S5 – A1 Benchmark (Real)*

Description

Part of the Yahoo Webscope S5 labeled anomaly detection dataset. A1 contains real-world time series with binary anomaly labels. Useful for evaluating anomaly detection methods on real traffic-like data. Labels available: Yes.

Usage

```
data(A1Benchmark)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1371/journal.pone.0262463

References

Yoshihara K, Takahashi K (2022) A simple method for unsupervised anomaly detection: An application to Web time series data. PLoS ONE 17(1).

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(A1Benchmark)
# Access the first series and visualize
s <- A1Benchmark[[1]]
plot(ts(s$value), main = names(A1Benchmark)[1], ylab = "value")
mean(s$event)  # proportion of labeled anomalies
```

`A2Benchmark` *Yahoo Webscope S5 – A2 Benchmark (Synthetic)*

Description

Part of the Yahoo Webscope S5 dataset. A2 contains synthetic time series with labeled anomalies designed to stress-test algorithms. Labels available: Yes.

Usage

```
data(A2Benchmark)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1371/journal.pone.0262463

References

Yoshihara K, Takahashi K (2022) A simple method for unsupervised anomaly detection: An application to Web time series data. PLoS ONE 17(1).

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(A2Benchmark)
s <- A2Benchmark[[1]]
summary(s$value)
```

`A3Benchmark` *Yahoo Webscope S5 – A3 Benchmark (Synthetic with Outliers)*

Description

Part of the Yahoo Webscope S5 dataset. A3 contains synthetic time series with labeled outliers/anomalies. Labels available: Yes.

Usage

```
data(A3Benchmark)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1371/journal.pone.0262463

References

Yoshihara K, Takahashi K (2022) A simple method for unsupervised anomaly detection: An application to Web time series data. PLoS ONE 17(1).

Examples

```
library(harbinger)
data(A3Benchmark)
s <- A3Benchmark[[1]]
# Quick visualization with harbinger
har_plot(harbinger(), s$value)
```

`A4Benchmark` *Yahoo Webscope S5 – A4 Benchmark (Synthetic with Anomalies and CPs)*

Description

Part of the Yahoo Webscope S5 dataset. A4 contains synthetic time series with labeled anomalies and change points. Labels available: Yes.

Usage

```
data(A4Benchmark)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1371/journal.pone.0262463

References

Yoshihara K, Takahashi K (2022) A simple method for unsupervised anomaly detection: An application to Web time series data. PLoS ONE 17(1).

Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(A4Benchmark)
s <- A4Benchmark[[1]]
mean(s$event)  # proportion of anomalous or change-point timestamps
```

`detect` *Detect events in time series*

Description

Generic S3 generic for event detection using a fitted Harbinger model. Concrete methods are implemented by each detector class.

Usage

```
detect(obj, ...)
```

Arguments

`obj`	A `harbinger` detector object.
`...`	Additional arguments passed to methods.

Value

A data frame with columns: `idx` (index), `event` (logical), and `type` (character: "anomaly", "changepoint", or ""). Some detectors may also attach attributes (e.g., `threshold`) or columns (e.g., `seq`, `seqlen`).

Examples

```
# See detector-specific examples in the package site for usage patterns
# and plotting helpers.
```

`examples_anomalies` *Time series for anomaly detection*

Description

A list of time series designed for anomaly detection tasks.

- simple: simple synthetic series with isolated anomalies.
- contextual: contextual anomalies relative to local behavior.
- trend: synthetic series with trend and anomalies.
- multiple: multiple anomalies.
- sequence: repeated anomalous sequences.
- tt: train-test split synthetic series.
- tt_warped: warped train-test synthetic series.

#'

Usage

```
data(examples_anomalies)
```

Format

A list of time series for anomaly detection.

Source

Harbinger package

References

Harbinger package

Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
data(examples_anomalies)
# Select a simple anomaly series
serie <- examples_anomalies$simple
head(serie)
```

`examples_changepoints` *Time series for change point detection*

Description

A list of time series for change point experiments.

- simple: simple synthetic series with one change point.
- sinusoidal: sinusoidal pattern with a regime change.
- incremental: gradual change in mean/variance.
- abrupt: abrupt level shift.
- volatility: variance change.

#'

Usage

```
data(examples_changepoints)
```

Format

A list of time series for change point detection.

Source

Harbinger package

References

Harbinger package

Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
data(examples_changepoints)
# Select a simple change point series
serie <- examples_changepoints$simple
head(serie)
```

`examples_harbinger` *Time series for event detection*

Description

A list of time series for demonstrating event detection tasks.

- nonstationarity: synthetic nonstationary time series.
- global_temperature_yearly: yearly global temperature.
- global_temperature_monthly: monthly global temperature.
- multidimensional: multivariate series with a change point.
- seattle_week: Seattle weekly temperature in 2019.
- seattle_daily: Seattle daily temperature in 2019.

#'

Usage

```
data(examples_harbinger)
```

Format

A list of time series.

Source

Harbinger package

References

Harbinger package

Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
data(examples_harbinger)
# Inspect a series (Seattle daily temperatures)
serie <- examples_harbinger$seattle_daily
head(serie)
```

---

examples_motifs *Time series for motif/discord discovery*

Description

A list of time series for motif (repeated patterns) and discord (rare patterns) discovery.

- simple: simple synthetic series with motifs.
- mitdb100: sample from MIT-BIH arrhythmia database (record 100).
- mitdb102: sample from MIT-BIH arrhythmia database (record 102).

#'

Usage

```
data(examples_motifs)
```

Format

A list of time series for motif discovery.

Source

Harbinger package

References

Harbinger package

Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
data(examples_motifs)
# Select a simple motif series
serie <- examples_motifs$simple
head(serie)
```

`gecco` *GECCO Challenge 2018 – Water Quality Time Series*

Description

Benchmark time series for water quality monitoring composed of multiple sensors and an associated binary event label. This dataset supports research in anomaly and event detection for environmental data streams. See cefet-rj-dal/united for usage guidance and links to the preprocessing steps used to build the package-ready object. Labels available: Yes.

Usage

```
data(gecco)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

GECCO Challenge 2018 (legacy challenge page unavailable)

References

Genetic and Evolutionary Computation Conference (GECCO), Association for Computing Machinery (ACM). See also: Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(gecco)
# Select the first univariate series and inspect
series <- gecco[[1]]
summary(series$value)
# Plot values with event markers
plot(ts(series$value), main = names(gecco)[1], ylab = "value")
```

hanct_dtw *Anomaly detector using DTW*

Description

Anomaly detection using DTW The DTW is applied to the time series. When seq equals one, observations distant from the closest centroids are labeled as anomalies. When seq is grater than one, sequences distant from the closest centroids are labeled as discords. It wraps the tsclust presented in the dtwclust library.

Usage

```
hanct_dtw(seq = 1, centers = NA)
```

Arguments

`seq`	sequence size
`centers`	number of centroids

Value

`hanct_dtw` object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure DTW-based detector
model <- hanct_dtw()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected events
```

```
print(detection[(detection$event),])
```

`hanct_kmeans` *Anomaly detector using kmeans*

Description

Anomaly detection using kmeans The kmeans is applied to the time series. When seq equals one, observations distant from the closest centroids are labeled as anomalies. When seq is grater than one, sequences distant from the closest centroids are labeled as discords. It wraps the kmeans presented in the stats library.

Usage

```
hanct_kmeans(seq = 1, centers = NA)
```

Arguments

`seq`	sequence size
`centers`	number of centroids

Value

`hanct_kmeans` object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure k-means detector
model <- hanct_kmeans()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
```

```
detection <- detect(model, dataset$serie)

# Show detected events
print(detection[(detection$event),])
```

---

hanc_ml *Anomaly detector based on ML classification*

Description

Supervised anomaly detection using a DALToolbox classifier trained with labeled events. Predictions above a probability threshold are flagged.

A set of preconfigured classification methods are listed at https://cefet-rj-dal.github.io/daltoolbox/ (e.g., cla_majority, cla_dtree, cla_knn, cla_mlp, cla_nb, cla_rf, cla_svm).

Usage

```
hanc_ml(model, threshold = 0.5)
```

Arguments

model	A DALToolbox classification model.
threshold	Numeric. Probability threshold for positive class.

Value

hanc_ml object.

References

- Bishop CM (2006). Pattern Recognition and Machine Learning. Springer.
- Hyndman RJ, Athanasopoulos G (2021). Forecasting: Principles and Practice. OTexts.
- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load labeled anomaly dataset
data(examples_anomalies)

# Use train-test example
dataset <- examples_anomalies$tt
dataset$event <- factor(dataset$event, labels=c("FALSE", "TRUE"))
slevels <- levels(dataset$event)
```

```
# Split into training and test
train <- dataset[1:80,]
test <- dataset[-(1:80),]

# Normalize features
norm <- minmax()
norm <- fit(norm, train)
train_n <- daltoolbox::transform(norm, train)

# Configure a decision tree classifier
model <- hanc_ml(cla_dtree("event", slevels))

# Fit the classifier
model <- fit(model, train_n)

# Evaluate detections on the test set
test_n <- daltoolbox::transform(norm, test)

detection <- detect(model, test_n)
print(detection[(detection$event),])
```

`hanr_arima` *Anomaly detector using ARIMA*

Description

Fits an ARIMA model to the series and flags observations with large model residuals as anomalies. Wraps ARIMA from the `forecast` package.

Usage

```
hanr_arima()
```

Details

The detector estimates ARIMA(p,d,q) and computes standardized residuals. Residual magnitudes are summarized via a distance function and thresholded with outlier heuristics from `harutils()`.

Value

`hanr_arima` object.

References

- Box GEP, Jenkins GM, Reinsel GC, Ljung GM (2015). Time Series Analysis: Forecasting and Control. Wiley.

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure ARIMA anomaly detector
model <- hanr_arima()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`hanr_emd` *Anomaly detector using EMD*

Description

Empirical Mode Decomposition (CEEMD) to extract intrinsic mode functions and flag anomalies from high-frequency components. Wraps `hht::CEEMD`.

Usage

```
hanr_emd(noise = 0.1, trials = 5)
```

Arguments

`noise`	Numeric. Noise amplitude for CEEMD.
`trials`	Integer. Number of CEEMD trials.

Value

`hanr_emd` object

References

- Huang NE, et al. (1998). The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proc. Royal Society A.

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure EMD-based anomaly detector
model <- hanr_emd()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`hanr_fbiad` *Anomaly detector using FBIAD*

Description

Anomaly detector using FBIAD

Usage

```
hanr_fbiad(sw_size = 30)
```

Arguments

`sw_size`	Window size for FBIAD

Value

hanr_fbiad object Forward and Backward Inertial Anomaly Detector (FBIAD) detects anomalies in time series. Anomalies are observations that differ from both forward and backward time series inertia doi:10.1109/IJCNN55064.2022.9892088.

References

- Lima, J., Salles, R., Porto, F., Coutinho, R., Alpis, P., Escobar, L., Pacitti, E., Ogasawara, E. Forward and Backward Inertial Anomaly Detector: A Novel Time Series Event Detection Method. Proceedings of the International Joint Conference on Neural Networks, 2022. doi:10.1109/IJCNN55064.2022.9892088

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure FBIAD detector
model <- hanr_fbiad()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`hanr_fft` *Anomaly detector using FFT*

Description

High-pass filtering via FFT to isolate high-frequency components; anomalies are flagged where the filtered magnitude departs strongly from baseline.

Usage

```
hanr_fft()
```

Details

The spectrum is computed by FFT, a cutoff is selected from the power spectrum, low frequencies are nulled, and the inverse FFT reconstructs a high-pass signal. Magnitudes are summarized and thresholded using `harutils()`.

Value

`hanr_fft` object

References

- Sobrinho, E. P., Souza, J., Lima, J., Giusti, L., Bezerra, E., Coutinho, R., Baroni, L., Pacitti, E., Porto, F., Belloze, K., Ogasawara, E. Fine-Tuning Detection Criteria for Enhancing Anomaly Detection in Time Series. In: Simpósio Brasileiro de Banco de Dados (SBBD). SBC, 29 Sep. 2025. doi:10.5753/sbbd.2025.247063

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure FFT-based anomaly detector
model <- hanr_fft()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`hanr_fft_amoc` *Anomaly Detector using FFT with AMOC Cutoff*

Description

This function implements an anomaly detection method that uses the Fast Fourier daltoolbox::transform (FFT) combined with an automatic frequency cutoff strategy based on the AMOC (At Most One Change) algorithm. The model analyzes the power spectrum of the time series and detects the optimal cutoff frequency — the point where the frequency content significantly changes — using a changepoint detection method from the `changepoint` package.

All frequencies below the cutoff are removed from the spectrum, and the inverse FFT reconstructs a filtered version of the original signal that preserves only high-frequency components. The resulting residual signal is then analyzed to identify anomalous patterns based on its distance from the expected behavior.

This function extends the HARBINGER framework and returns an object of class `hanr_fft_amoc`.

Usage

```
hanr_fft_amoc()
```

Value

`hanr_fft_amoc` object

References

- Sobrinho, E. P., Souza, J., Lima, J., Giusti, L., Bezerra, E., Coutinho, R., Baroni, L., Pacitti, E., Porto, F., Belloze, K., Ogasawara, E. Fine-Tuning Detection Criteria for Enhancing Anomaly Detection in Time Series. In: Simpósio Brasileiro de Banco de Dados (SBBD). SBC, 29 Sep. 2025. doi:10.5753/sbbd.2025.247063

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure FFT+AMOC detector
model <- hanr_fft_amoc()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Inspect detected anomalies
print(detection[detection$event, ])
```

`hanr_fft_amoc_cusum` *Anomaly Detector using FFT with AMOC and CUSUM Cutoff*

Description

This function implements an anomaly detection method based on the Fast Fourier daltoolbox::transform (FFT) and a changepoint-based cutoff strategy using the AMOC (At Most One Change) algorithm applied to the cumulative sum (CUSUM) of the power spectrum.

The method first computes the FFT of the input time series and extracts the power spectrum. It then applies a CUSUM transformation to the spectral data to emphasize gradual changes or shifts in spectral energy. Using the AMOC algorithm, it detects a single changepoint in the CUSUM-transformed spectrum, which serves as a cutoff index to remove the lower-frequency components.

The remaining high-frequency components are then reconstructed into a time-domain signal via inverse FFT, effectively isolating rapid or local deviations. Anomalies are detected by evaluating the distance between this filtered signal and the original series, highlighting points that deviate significantly from the expected pattern.

This method is suitable for series where spectral shifts are subtle and a single significant change in behavior is expected.

This function extends the HARBINGER framework and returns an object of class `hanr_fft_amoc_cusum`.

Usage

```
hanr_fft_amoc_cusum()
```

Value

`hanr_fft_amoc_cusum` object

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure FFT+CUSUM+AMOC detector
model <- hanr_fft_amoc_cusum()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Inspect detected anomalies
print(detection[detection$event, ])
```

`hanr_fft_binseg` *Anomaly Detector using FFT with Binary Segmentation Cutoff*

Description

This function implements an anomaly detection method that combines the Fast Fourier daltoolbox::transform (FFT) with a spectral cutoff strategy based on the Binary Segmentation (BinSeg) algorithm for changepoint detection.

The method analyzes the power spectrum of the input time series and applies the BinSeg algorithm to identify a changepoint in the spectral density, corresponding to a shift in the frequency content.

Frequencies below this changepoint are considered part of the underlying trend or noise and are removed from the signal.

The modified spectrum is then transformed back into the time domain via inverse FFT, resulting in a high-pass filtered version of the series. Anomalies are identified by measuring the distance between the original and the filtered signal, highlighting unusual deviations from the dominant signal behavior.

This function is part of the HARBINGER framework and returns an object of class `hanr_fft_binseg`.

Usage

```
hanr_fft_binseg()
```

Value

`hanr_fft_binseg` object

References

- Sobrinho, E. P., Souza, J., Lima, J., Giusti, L., Bezerra, E., Coutinho, R., Baroni, L., Pacitti, E., Porto, F., Belloze, K., Ogasawara, E. Fine-Tuning Detection Criteria for Enhancing Anomaly Detection in Time Series. In: Simpósio Brasileiro de Banco de Dados (SBBD). SBC, 29 Sep. 2025. doi:10.5753/sbbd.2025.247063

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure FFT+BinSeg detector
model <- hanr_fft_binseg()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Inspect detected anomalies
print(detection[detection$event, ])
```

`hanr_fft_binseg_cusum` *Anomaly Detector using FFT with BinSeg and CUSUM Cutoff*

Description

This function implements an anomaly detection method that combines the Fast Fourier daltoolbox::transform (FFT) with a changepoint-based cutoff strategy using the Binary Segmentation (BinSeg) method applied to the cumulative sum (CUSUM) of the frequency spectrum.

The method first computes the FFT of the input time series and obtains its power spectrum. Then, it applies a CUSUM transformation to the spectral density to enhance detection of gradual transitions or accumulated changes in energy across frequencies. The Binary Segmentation method is applied to the CUSUM-transformed spectrum to identify a changepoint that defines a cutoff frequency.

Frequencies below this cutoff are removed from the spectrum, and the signal is reconstructed using the inverse FFT. This produces a filtered signal that retains only the high-frequency components, emphasizing potential anomalies.

Anomalies are then detected by measuring the deviation of the filtered signal from the original one, and applying an outlier detection mechanism based on this residual.

This function extends the HARBINGER framework and returns an object of class `hanr_fft_binseg_cusum`.

Usage

```
hanr_fft_binseg_cusum()
```

Value

`hanr_fft_binseg_cusum` object

References

- Sobrinho, E. P., Souza, J., Lima, J., Giusti, L., Bezerra, E., Coutinho, R., Baroni, L., Pacitti, E., Porto, F., Belloze, K., Ogasawara, E. Fine-Tuning Detection Criteria for Enhancing Anomaly Detection in Time Series. In: Simpósio Brasileiro de Banco de Dados (SBBD). SBC, 29 Sep. 2025. doi:10.5753/sbbd.2025.247063

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

#Using simple example
dataset <- examples_anomalies$simple
head(dataset)

# setting up time series fft detector
model <- hanr_fft_binseg_cusum()
```

```
# fitting the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# filtering detected events
print(detection[(detection$event),])
```

`hanr_fft_sma` *Anomaly Detector using Adaptive FFT and Moving Average*

Description

This function implements an anomaly detection model based on the Fast Fourier daltoolbox::transform (FFT), combined with an adaptive moving average filter. The method estimates the dominant frequency in the input time series using spectral analysis and then applies a moving average filter with a window size derived from that frequency. This highlights high-frequency deviations, which are likely to be anomalies.

The residuals (original signal minus smoothed version) are then processed to compute the distance from the expected behavior, and points significantly distant are flagged as anomalies. The detection also includes a grouping strategy to reduce false positives by selecting the most representative point in a cluster of consecutive anomalies.

This function extends the HARBINGER framework and returns an object of class `hanr_fft_sma`.

Usage

```
hanr_fft_sma()
```

Value

`hanr_fft_sma` object

References

- Sobrinho, E. P., Souza, J., Lima, J., Giusti, L., Bezerra, E., Coutinho, R., Baroni, L., Pacitti, E., Porto, F., Belloze, K., Ogasawara, E. Fine-Tuning Detection Criteria for Enhancing Anomaly Detection in Time Series. In: Simpósio Brasileiro de Banco de Dados (SBBD). SBC, 29 Sep. 2025. doi:10.5753/sbbd.2025.247063

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)
```

```
#Using simple example
dataset <- examples_anomalies$simple
head(dataset)

# setting up time series fft detector
model <- hanr_fft_sma()

# fitting the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# filtering detected events
print(detection[(detection$event),])
```

---

hanr_garch *Anomaly detector using GARCH*

Description

Fits a GARCH model to capture conditional heteroskedasticity and flags observations with large standardized residuals as anomalies. Wraps `rugarch`.

Usage

```
hanr_garch()
```

Details

A sGARCH(1,1) with ARMA(1,1) mean is estimated. Standardized residuals are summarized and thresholded via `harutils()`.

Value

`hanr_garch` object.

References

- Engle RF (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. Econometrica, 50(4):987–1007.
- Bollerslev T (1986). Generalized Autoregressive Conditional Heteroskedasticity. Journal of Econometrics, 31(3):307–327.

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure GARCH anomaly detector
model <- hanr_garch()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`hanr_histogram` *Anomaly detector using histograms*

Description

Flags observations that fall into low-density histogram bins or outside the observed bin range.

Usage

```
hanr_histogram(density_threshold = 0.05)
```

Arguments

`density_threshold`
: Numeric between 0 and 1. Minimum bin density to avoid being considered an anomaly (default 0.05).

Value

`hanr_histogram` object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure histogram-based detector
model <- hanr_histogram()

# Fit the model (no-op)
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

---

hanr_ml *Anomaly detector based on ML regression*

Description

Trains a regression model to forecast the next value from a sliding window and flags large prediction errors as anomalies. Uses DALToolbox regressors.

A set of preconfigured regression methods are described at https://cefet-rj-dal.github.io/daltoolbox/ (e.g., `ts_elm`, `ts_conv1d`, `ts_lstm`, `ts_mlp`, `ts_rf`, `ts_svm`).

Usage

```
hanr_ml(model, sw_size = 15)
```

Arguments

`model`	A DALToolbox regression model.
`sw_size`	Integer. Sliding window size.

Value

`hanr_ml` object.

References

- Hyndman RJ, Athanasopoulos G (2021). Forecasting: Principles and Practice. OTexts.
- Goodfellow I, Bengio Y, Courville A (2016). Deep Learning. MIT Press.

Examples

```
library(daltoolbox)
library(tspredit)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure a time series regression model
model <- hanr_ml(tspredit::ts_elm(tspredit::ts_norm_gminmax(),
                   input_size=4, nhid=3, actfun="purelin"))

# Fit the model
model <- daltoolbox::fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

---

hanr_remd — *Anomaly detector using REMD*

Description

Anomaly detection using REMD The EMD model adjusts to the time series. Observations distant from the model are labeled as anomalies. It wraps the EMD model presented in the forecast library.

Usage

```
hanr_remd(noise = 0.1, trials = 5)
```

Arguments

noise	nosie
trials	trials

Value

`hanr_remd` object

References

- Souza, J., Paixão, E., Fraga, F., Baroni, L., Alves, R. F. S., Belloze, K., Dos Santos, J., Bezerra, E., Porto, F., Ogasawara, E. REMD: A Novel Hybrid Anomaly Detection Method Based on EMD and ARIMA. Proceedings of the International Joint Conference on Neural Networks, 2024. doi:10.1109/IJCNN60899.2024.10651192

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure REMD detector
model <- hanr_remd()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`hanr_rtad` *Anomaly and change point detector using RTAD*

Description

Anomaly and change point detection using RTAD The RTAD model adjusts to the time series. Observations distant from the model are labeled as anomalies. It wraps the EMD model presented in the hht library.

Usage

```
hanr_rtad(sw_size = 30, noise = 0.001, trials = 5, sigma = sd)
```

Arguments

`sw_size`	sliding window size (default 30)
`noise`	noise
`trials`	trials
`sigma`	function to compute the dispersion

Value

`hanr_rtad` object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)
library(zoo)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure RTAD detector
model <- hanr_rtad()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected events
print(detection[(detection$event),])
```

`hanr_wavelet` *Anomaly detector using Wavelets*

Description

Multiresolution decomposition via wavelets; anomalies are flagged where aggregated wavelet detail coefficients indicate unusual energy.

Usage

```
hanr_wavelet(filter = "haar")
```

Arguments

`filter`	Character. Available wavelet filters: `haar`, `d4`, `la8`, `bl14`, `c6`.

Details

The series is decomposed with MODWT and detail bands are aggregated to compute a magnitude signal that is thresholded using `harutils()`.

Value

`hanr_wavelet` object

References

- Mallat S (1989). A theory for multiresolution signal decomposition: The wavelet representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(7):674–693.

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure wavelet-based anomaly detector
model <- hanr_wavelet()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected anomalies
print(detection[(detection$event),])
```

`han_autoencoder` *Anomaly detector using autoencoders*

Description

Trains an encoder-decoder (autoencoder) to reconstruct sliding windows of the series; large reconstruction errors indicate anomalies.

Usage

```
han_autoencoder(input_size, encode_size, encoderclass = autoenc_base_ed, ...)
```

Arguments

`input_size`	Integer. Input (and output) window size for the autoencoder.
`encode_size`	Integer. Size of the encoded (bottleneck) representation.
`encoderclass`	DALToolbox encoder-decoder constructor to instantiate.
`...`	Additional arguments forwarded to `encoderclass`.

Value

`han_autoencoder` object

References

- Sakurada M, Yairi T (2014). Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. MLSDA 2014.

Examples

```
library(daltoolbox)
library(tspredit)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure an autoencoder-based anomaly detector
model <- han_autoencoder(input_size = 5, encode_size = 3)

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)
```

```
# Inspect detected anomalies
print(detection[detection$event, ])
```

`harbinger` *Harbinger*

Description

Base class for time series event detection in the Harbinger framework. It provides common state handling and helper methods used by anomaly, change point, and motif detectors. Concrete detectors extend this class and implement their own `fit()` and/or `detect()` S3 methods.

Usage

```
harbinger()
```

Details

Internally, this class stores references to the original series, indices of non-missing observations, and helper structures to restore detection results in the original series index space. It also exposes utility hooks for distance computation and outlier post-processing provided by `harutils()`.

Value

A `harbinger` object that can be extended by detectors.

References

- Harbinger documentation: https://cefet-rj-dal.github.io/harbinger
- Salles, R., Escobar, L., Baroni, L., Zorrilla, R., Ziviani, A., Kreischer, V., Delicato, F., Pires, P. F., Maia, L., Coutinho, R., Assis, L., Ogasawara, E. Harbinger: Um framework para integração e análise de métodos de detecção de eventos em séries temporais. Anais do Simpósio Brasileiro de Banco de Dados (SBBD). In: Anais do XXXV Simpósio Brasileiro de Bancos de Dados. SBC, 28 Sep. 2020. doi:10.5753/sbbd.2020.13626

Examples

```
# See the specific detector examples for anomalies, change points, and motifs
# at https://cefet-rj-dal.github.io/harbinger
```

`harutils` *Harbinger Utilities*

Description

Utility object that groups common distance measures, threshold heuristics, and outlier grouping rules used by Harbinger detectors.

Usage

```
harutils()
```

Details

Provided helpers include:

- L1 and L2 distance aggregations over vectors or rows of matrices/data frames.
- Thresholding heuristics: boxplot-based (IQR), Gaussian 3-sigma, and a ratio-based rule.
- Grouping strategies for contiguous outliers: keep first index or keep highest-magnitude index.
- Optional fuzzification over detections to propagate influence within a tolerance window.

These utilities centralize common tasks and ensure consistent behavior across detectors.

Value

A `harutils` object exposing the helper functions.

References

- Tukey JW (1977). Exploratory Data Analysis. Addison-Wesley. (boxplot/IQR heuristic)
- Shewhart WA (1931). Economic Control of Quality of Manufactured Product. D. Van Nostrand. (three-sigma rule)
- Silva, E. P., Balbi, H., Pacitti, E., Porto, F., Santos, J., Ogasawara, E. Cutoff Frequency Adjustment for FFT-Based Anomaly Detectors. In: Simpósio Brasileiro de Banco de Dados (SBBD). SBC, 14 Oct. 2024. doi:10.5753/sbbd.2024.243319

Examples

```
# Basic usage of utilities
utils <- harutils()

# Compute L2 distance on residuals
res <- c(0.1, -0.5, 1.2, -0.3)
d2 <- utils$har_distance_l2(res)
print(d2)

# Apply 3-sigma outlier rule and keep only first index of contiguous runs
idx <- utils$har_outliers_gaussian(d2)
```

```
flags <- utils$har_outliers_checks_firstgroup(idx, d2)
print(which(flags))
```

`har_ensemble` *Harbinger Ensemble*

Description

Majority-vote ensemble across multiple Harbinger detectors with optional temporal fuzzification to combine nearby detections.

Usage

```
har_ensemble(...)
```

Arguments

`...`	One or more detector objects.

Value

A `har_ensemble` object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use a simple example
dataset <- examples_anomalies$simple
head(dataset)

# Configure an ensemble of detectors
model <- har_ensemble(hanr_arima(), hanr_arima(), hanr_arima())
# model <- har_ensemble(hanr_fbiad(), hanr_arima(), hanr_emd())

# Fit all ensemble members
model <- fit(model, dataset$serie)

# Run ensemble detection
detection <- detect(model, dataset$serie)
```

```
# Show detected events
print(detection[(detection$event),])
```

har_eval *Evaluation of event detection*

Description

Hard evaluation of event detection producing confusion matrix and common metrics (accuracy, precision, recall, F1, etc.).

Usage

```
har_eval()
```

Value

`har_eval` object

References

- Salles, R., Lima, J., Reis, M., Coutinho, R., Pacitti, E., Masseglia, F., Akbarinia, R., Chen, C., Garibaldi, J., Porto, F., Ogasawara, E. SoftED: Metrics for soft evaluation of time series event detection. Computers and Industrial Engineering, 2024. doi:10.1016/j.cie.2024.110728

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

dataset <- examples_anomalies$simple
head(dataset)

# Configure a change-point detector (GARCH)
model <- hcp_garch()

# Fit the detector
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected events
print(detection[(detection$event),])

# Evaluate detections
evaluation <- evaluate(har_eval(), detection$event, dataset$event)
```

```
print(evaluation$confMatrix)

# Plot the results
grf <- har_plot(model, dataset$serie, detection, dataset$event)
plot(grf)
```

`har_eval_soft` *Evaluation of event detection (SoftED)*

Description

Soft evaluation of event detection using SoftED doi:10.48550/arXiv.2304.00439.

Usage

```
har_eval_soft(sw_size = 15)
```

Arguments

`sw_size`	Integer. Tolerance window size for soft matching.

Value

`har_eval_soft` object

References

- Salles, R., Lima, J., Reis, M., Coutinho, R., Pacitti, E., Masseglia, F., Akbarinia, R., Chen, C., Garibaldi, J., Porto, F., Ogasawara, E. SoftED: Metrics for soft evaluation of time series event detection. Computers and Industrial Engineering, 2024. doi:10.1016/j.cie.2024.110728

Examples

```
library(daltoolbox)

# Load anomaly example data
data(examples_anomalies)

# Use the simple series
dataset <- examples_anomalies$simple
head(dataset)

# Configure a change-point detector (GARCH)
model <- hcp_garch()

# Fit the detector
model <- fit(model, dataset$serie)

# Run detection
```

```
detection <- detect(model, dataset$serie)

# Show detected events
print(detection[(detection$event),])

# Evaluate detections (SoftED)
evaluation <- evaluate(har_eval_soft(), detection$event, dataset$event)
print(evaluation$confMatrix)

# Plot the results
grf <- har_plot(model, dataset$serie, detection, dataset$event)
plot(grf)
```

`har_plot` *Plot event detection on a time series*

Description

Convenience plotting helper for Harbinger detections. It accepts a detector, the input series, an optional detection data.frame, and optional ground-truth events to color-code true positives (TP), false positives (FP), and false negatives (FN). It can also mark detected change points and draw reference horizontal lines.

Usage

```
har_plot(
  obj,
  serie,
  detection = NULL,
  event = NULL,
  mark.cp = TRUE,
  ylim = NULL,
  idx = NULL,
  pointsize = 0.5,
  colors = c("green", "blue", "red", "purple"),
  yline = NULL
)
```

Arguments

`obj`	A `harbinger` detector used to produce `detection`.
`serie`	Numeric vector with the time series to plot.
`detection`	Optional detection data.frame as returned by `detect()`.
`event`	Optional logical vector with ground-truth events (same length as `serie`).
`mark.cp`	Logical; if TRUE, marks detected change points with dashed vertical lines.
`ylim`	Optional numeric vector of length 2 for y-axis limits.

`idx`	Optional x-axis labels or indices (defaults to `seq_along(serie)`).
`pointsize`	Base point size for observations.
`colors`	Character vector of length 4 with colors for TP, FN, FP, and motif segments.
`yline`	Optional numeric vector with y values to draw dotted horizontal lines.

Value

A `ggplot` object showing the time series with detected events highlighted.

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load an example anomaly dataset
data(examples_anomalies)

# Use the simple time series
dataset <- examples_anomalies$simple
head(dataset)

# Set up an ARIMA-based anomaly detector
model <- hanr_arima()

# Fit the detector
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Inspect detected events
print(detection[(detection$event),])

# Evaluate detections (soft evaluation)
evaluation <- evaluate(har_eval_soft(), detection$event, dataset$event)
print(evaluation$confMatrix)

# Plot the results
grf <- har_plot(model, dataset$serie, detection, dataset$event)
plot(grf)
```

hcp_amoc *At Most One Change (AMOC)*

Description

Change-point detection method focusing on identifying at most one change in mean and/or variance. This is a wrapper around the AMOC implementation from the `changepoint` package.

Usage

```
hcp_amoc()
```

Details

AMOC detects a single most significant change point under a cost function optimized for a univariate series. It is useful when at most one structural break is expected.

Value

`hcp_amoc` object.

References

- Hinkley DV (1970). Inference about the change-point in a sequence of random variables. Biometrika, 57(1):1–17. doi:10.1093/biomet/57.1.1
- Killick R, Fearnhead P, Eckley IA (2012). Optimal detection of changepoints with a linear computational cost. JASA, 107(500):1590–1598.

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure the AMOC detector
model <- hcp_amoc()

# Fit the detector (no-op for AMOC)
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change point(s)
```

```
print(detection[(detection$event),])
```

---

hcp_binseg *Binary Segmentation (BinSeg)*

Description

Multi-change-point detection via Binary Segmentation on mean/variance using the `changepoint` package.

Usage

```
hcp_binseg(Q = 2)
```

Arguments

`Q`	Integer. Maximum number of change points to search for.

Details

Binary Segmentation recursively partitions the series around the largest detected change until a maximum number of change points or stopping criterion is met. This is a fast heuristic widely used in practice.

Value

`hcp_binseg` object.

References

- Vostrikova L (1981). Detecting "disorder" in multidimensional random processes. Soviet Mathematics Doklady, 24, 55–59.
- Killick R, Fearnhead P, Eckley IA (2012). Optimal detection of changepoints with a linear computational cost. JASA, 107(500):1590–1598. dplyr::context

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure the BinSeg detector
model <- hcp_binseg()
```

```
# Fit the detector (no-op for BinSeg)
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

---

hcp_cf_arima *Change Finder using ARIMA*

Description

Change-point detection by modeling residual deviations with ARIMA and applying a second-stage smoothing and thresholding, inspired by ChangeFinder doi:10.1109/TKDE.2006.1599387. Wraps ARIMA from the `forecast` package.

Usage

```
hcp_cf_arima(sw_size = NULL)
```

Arguments

`sw_size`	Integer. Sliding window size for smoothing/statistics.

Value

`hcp_cf_arima` object.

References

- Takeuchi J, Yamanishi K (2006). A unifying framework for detecting outliers and change points from time series. IEEE Transactions on Knowledge and Data Engineering.

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure ChangeFinder-ARIMA detector
```

```
model <- hcp_cf_arima()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

`hcp_cf_ets` *Change Finder using ETS*

Description

Change-point detection by modeling residual deviations with ETS and applying a second-stage smoothing and thresholding, inspired by ChangeFinder doi:10.1109/TKDE.2006.1599387. Wraps ETS from the `forecast` package.

Usage

```
hcp_cf_ets(sw_size = 7)
```

Arguments

`sw_size`	Integer. Sliding window size for smoothing/statistics.

Value

`hcp_cf_ets` object.

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure ChangeFinder-ETS detector
model <- hcp_cf_ets()

# Fit the model
model <- fit(model, dataset$serie)
```

```
# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

`hcp_cf_lr` *Change Finder using Linear Regression*

Description

Change-point detection by modeling residual deviations with linear regression and applying a second-stage smoothing and thresholding, inspired by ChangeFinder doi:10.1109/TKDE.2006.1599387.

Usage

```
hcp_cf_lr(sw_size = 30)
```

Arguments

`sw_size`	Integer. Sliding window size for smoothing/statistics.

Value

`hcp_cf_lr` object.

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure ChangeFinder-LR detector
model <- hcp_cf_lr()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

hcp_chow *Chow Test (structural break)*

Description

Change-point detection for linear models using F-based structural break tests from the strucchange package doi:10.18637/jss.v007.i02. It wraps the Fstats and breakpoints implementation available in the strucchange library.

Usage

```
hcp_chow()
```

Value

hcp_chow object

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure the Chow detector
model <- hcp_chow()

# Fit the detector (no-op for Chow)
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

hcp_garch *Change Finder using GARCH*

Description

Change-point detection is related to event/trend change detection. Change Finder GARCH detects change points based on deviations relative to linear regression model doi:10.1109/TKDE.2006.1599387. It wraps the GARCH model presented in the rugarch library.

Usage

```
hcp_garch(sw_size = 5)
```

Arguments

sw_size	Sliding window size

Value

hcp_garch object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a volatility example
dataset <- examples_changepoints$volatility
head(dataset)

# Configure ChangeFinder-GARCH detector
model <- hcp_garch()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

---

hcp_gft *Generalized Fluctuation Test (GFT)*

Description

Structural change detection using generalized fluctuation tests via `strucchange::breakpoints()` doi:10.18637/jss.v007.i02.

Usage

```
hcp_gft()
```

Value

`hcp_gft` object

References

- Zeileis A, Leisch F, Kleiber C, Hornik K (2002). strucchange: An R package for testing for structural change in linear regression models. Journal of Statistical Software, 7(2). doi:10.18637/jss.v007.i02
- Zeileis A, Kleiber C, Krämer W, Hornik K (2003). Testing and dating of structural changes in practice. Computational Statistics & Data Analysis, 44(1):109–123.

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure the GFT detector
model <- hcp_gft()

# Fit the detector (no-op for GFT)
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

---

hcp_pelt *Pruned Exact Linear Time (PELT)*

Description

Multiple change-point detection using the PELT algorithm for mean/variance with a linear-time cost under suitable penalty choices. This function wraps the PELT implementation in the `changepoint` package.

Usage

```
hcp_pelt()
```

Details

PELT performs optimal partitioning while pruning candidate change-point locations to achieve near-linear computational cost.

Value

`hcp_pelt` object.

References

- Killick R, Fearnhead P, Eckley IA (2012). Optimal detection of changepoints with a linear computational cost. JASA, 107(500):1590–1598.

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure the PELT detector
model <- hcp_pelt()

# Fit the detector (no-op for PELT)
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

`hcp_scp` *Seminal change point*

Description

Change-point detection is related to event/trend change detection. Seminal change point detects change points based on deviations of linear regression models adjusted with and without a central observation in each sliding window <10.1145/312129.312190>.

Usage

```
hcp_scp(sw_size = 30)
```

Arguments

sw_size	Sliding window size

Value

hcp_scp object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Configure seminal change-point detector
model <- hcp_scp()

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected change points
print(detection[(detection$event),])
```

---

hdis_mp *Discord discovery using Matrix Profile*

Description

Discovers rare, dissimilar subsequences (discords) using Matrix Profile as implemented in the `tsmp` package doi:10.32614/RJ-2020-021.

Usage

```
hdis_mp(mode = "stamp", w, qtd)
```

Arguments

mode	Character. Algorithm: one of "stomp", "stamp", "simple", "mstomp", "scrimp", "valmod", "pmp".
w	Integer. Subsequence window size.
qtd	Integer. Number of discords to return (>= 3 recommended).

Value

`hdis_mp` object.

References

- Yeh CCM, et al. (2016). Matrix Profile I/II: All-pairs similarity joins and scalable time series motifs/discrod discovery. IEEE ICDM.
- Tavenard R, et al. tsmp: The Matrix Profile in R. The R Journal (2020). doi:10.32614/RJ-2020-021

Examples

```
library(daltoolbox)

# Load motif/discord example data
data(examples_motifs)

# Use a simple sequence example
dataset <- examples_motifs$simple
head(dataset)

# Configure discord discovery via Matrix Profile
model <- hdis_mp("stamp", 4, 3)

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected discords
print(detection[(detection$event),])
```

`hdis_sax` *Discord discovery using SAX*

Description

Discord discovery using SAX doi:10.1007/s10618-007-0064-z

Usage

```
hdis_sax(a, w, qtd = 2)
```

Arguments

a	alphabet size
w	word size
qtd	number of occurrences to be classified as discords

Value

`hdis_sax` object

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load motif/discord example data
data(examples_motifs)

# Use a simple sequence example
dataset <- examples_motifs$simple
head(dataset)

# Configure discord discovery via SAX
model <- hdis_sax(26, 3, 3)

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected discords
print(detection[(detection$event),])
```

hmo_mp *Motif discovery using Matrix Profile*

Description

Discovers repeated subsequences (motifs) using Matrix Profile methods as implemented in the `tsmp` package doi:10.32614/RJ-2020-021.

Usage

```
hmo_mp(mode = "stamp", w, qtd)
```

Arguments

`mode`	Character. Algorithm: one of "stomp", "stamp", "simple", "mstomp", "scrimp", "valmod", "pmp".
`w`	Integer. Subsequence window size.
`qtd`	Integer. Minimum number of occurrences to classify as a motif.

Value

`hmo_mp` object.

References

- Yeh CCM, et al. (2016). Matrix Profile I/II: All-pairs similarity joins and scalable time series motifs/discrod discovery. IEEE ICDM.
- Tavenard R, et al. tsmp: The Matrix Profile in R. The R Journal (2020). doi:10.32614/RJ-2020-021

Examples

```
library(daltoolbox)

# Load motif example data
data(examples_motifs)

# Use a simple sequence example
dataset <- examples_motifs$simple
head(dataset)

# Configure motif discovery via Matrix Profile
model <- hmo_mp("stamp", 4, 3)

# Fit the model
model <- fit(model, dataset$serie)

# Run detection
```

```
detection <- detect(model, dataset$serie)

# Show detected motifs
print(detection[(detection$event),])
```

---

hmo_sax *Motif discovery using SAX*

Description

Discovers repeated subsequences (motifs) using a Symbolic Aggregate approXimation (SAX) representation doi:10.1007/s10618-007-0064-z. Subsequences are discretized and grouped by symbolic words; frequently occurring words indicate motifs.

Usage

```
hmo_sax(a, w, qtd = 2)
```

Arguments

a	Integer. Alphabet size.
w	Integer. Word/window size.
qtd	Integer. Minimum number of occurrences to classify as a motif.

Value

hmo_sax object.

References

- Lin J, Keogh E, Lonardi S, Chiu B (2007). A symbolic representation of time series, with implications for streaming algorithms. Data Mining and Knowledge Discovery 15, 107–144.

Examples

```
library(daltoolbox)

# Load motif example data
data(examples_motifs)

# Use a simple sequence example
dataset <- examples_motifs$simple
head(dataset)

# Configure SAX-based motif discovery
model <- hmo_sax(26, 3, 3)

# Fit the model
```

```
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected motifs
print(detection[(detection$event),])
```

---

hmo_xsax *Motif discovery using XSAX*

Description

Discovers repeated subsequences (motifs) using an extended SAX (XSAX) representation that supports a larger alphanumeric alphabet.

Usage

```
hmo_xsax(a, w, qtd)
```

Arguments

a	Integer. Alphabet size.
w	Integer. Word/window size.
qtd	Integer. Minimum number of occurrences to be classified as motifs.

Value

hmo_xsax object.

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

Examples

```
library(daltoolbox)

# Load motif example data
data(examples_motifs)

# Use a simple sequence example
dataset <- examples_motifs$simple
head(dataset)

# Configure XSAX-based motif discovery
model <- hmo_xsax(37, 3, 3)
```

```
# Fit the model
model <- fit(model, dataset$serie)

# Run detection
detection <- detect(model, dataset$serie)

# Show detected motifs
print(detection[(detection$event),])
```

---

hmu_pca *Multivariate anomaly detector using PCA*

Description

Projects multivariate observations onto principal components and flags large reconstruction errors as anomalies. Based on classical PCA.

Usage

```
hmu_pca()
```

Details

The series is standardized, PCA is computed, and data are reconstructed from principal components. The reconstruction error is summarized and thresholded.

Value

`hmu_pca` object.

References

- Jolliffe IT (2002). Principal Component Analysis. Springer.

Examples

```
library(daltoolbox)

# Load multivariate example data
data(examples_harbinger)

# Use a multidimensional time series
dataset <- examples_harbinger$multidimensional
head(dataset)

# Configure PCA-based anomaly detector
model <- hmu_pca()
```

```
# Fit the model (example uses first two columns)
model <- fit(model, dataset[,1:2])

# Run detection
detection <- detect(model, dataset[,1:2])

# Show detected anomalies
print(detection[(detection$event),])

# Evaluate detections
evaluation <- evaluate(model, detection$event, dataset$event)
print(evaluation$confMatrix)
```

`loadfulldata` *Load full dataset from mini data object*

Description

The mini datasets stored in `data/` include an `attr(url)` pointing to the full dataset in `harbinger/`. This helper downloads and loads the full data.

Usage

```
loadfulldata(x, envir = parent.frame())
```

Arguments

`x`	Dataset object or its name (string or symbol).
`envir`	Environment to load the full dataset into.

Value

The full dataset object.

Examples

```
data(A1Benchmark)
A1Benchmark <- loadfulldata(A1Benchmark)
```

mas — *Moving average smoothing*

Description

The `mas()` function returns a simple moving average smoother of the provided time series.

Usage

```
mas(x, order)
```

Arguments

`x`	A numeric vector or univariate time series.
`order`	Order of moving average smoother.

Details

The moving average smoother transformation is given by

$$(1/k) * (x[t] + x[t + 1] + ... + x[t + k - 1])$$

where `k=order`, `t` assume values in the range `1:(n-k+1)`, and `n=length(x)`. See also the `ma` of the `forecast` package.

Value

Numerical time series of length `length(x)-order+1` containing the simple moving average smoothed values.

References

R.H. Shumway and D.S. Stoffer, 2010, Time Series Analysis and Its Applications: With R Examples. 3rd ed. 2011 edition ed. New York, Springer.

Examples

```
# Load change-point example data
data(examples_changepoints)

# Use a simple example
dataset <- examples_changepoints$simple
head(dataset)

# Compute a 5-point moving average
ma <- mas(dataset$serie, 5)
```

`mit_bih_MLII` *MIT-BIH Arrhythmia Database – MLII Lead*

Description

Data collection with real-world time-series. MIT-BIH Arrhythmia Database (MIT-BIH). See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(mit_bih_MLII)
```

Format

A list of time series from the MLII sensor of the MIT-BIH Arrhythmia Database.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1109/51.932724

References

MIT-BIH Arrhythmia Database (MIT-BIH). See also: Moody, G. B., & Mark, R. G. (2001). The impact of the MIT-BIH Arrhythmia Database. IEEE Engineering in Medicine and Biology Magazine, 20(3), 45–50.

Examples

```
data(mit_bih_MLII)
data <- mit_bih_MLII[[1]]
series <- data$value
```

`mit_bih_V1` *MIT-BIH Arrhythmia Database – V1 Lead*

Description

Data collection with real-world time-series. MIT-BIH Arrhythmia Database (MIT-BIH). See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(mit_bih_V1)
```

Format

A list of time series from the V1 sensor of the MIT-BIH Arrhythmia Database.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1109/51.932724

References

MIT-BIH Arrhythmia Database (MIT-BIH). See also: Moody, G. B., & Mark, R. G. (2001). The impact of the MIT-BIH Arrhythmia Database. IEEE Engineering in Medicine and Biology Magazine, 20(3), 45–50.

Examples

```
data(mit_bih_V1)
data <- mit_bih_V1[[1]]
series <- data$value
```

`mit_bih_V2` *MIT-BIH Arrhythmia Database – V2 Lead*

Description

Data collection with real-world time-series. MIT-BIH Arrhythmia Database (MIT-BIH). See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(mit_bih_V2)
```

Format

A list of time series from the V2 sensor of the MIT-BIH Arrhythmia Database.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1109/51.932724

References

MIT-BIH Arrhythmia Database (MIT-BIH). See also: Moody, G. B., & Mark, R. G. (2001). The impact of the MIT-BIH Arrhythmia Database. IEEE Engineering in Medicine and Biology Magazine, 20(3), 45–50.

Examples

```
data(mit_bih_V2)
data <- mit_bih_V2[[1]]
series <- data$value
```

`mit_bih_V5` *MIT-BIH Arrhythmia Database – V5 Lead*

Description

Data collection with real-world time-series. MIT-BIH Arrhythmia Database (MIT-BIH). See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(mit_bih_V5)
```

Format

A list of time series from the V5 sensor of the MIT-BIH Arrhythmia Database.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

doi:10.1109/51.932724

References

MIT-BIH Arrhythmia Database (MIT-BIH). See also: Moody, G. B., & Mark, R. G. (2001). The impact of the MIT-BIH Arrhythmia Database. IEEE Engineering in Medicine and Biology Magazine, 20(3), 45–50.

Examples

```
data(mit_bih_V5)
data <- mit_bih_V5[[1]]
series <- data$value
```

`nab_artificialWithAnomaly`

Numenta Anomaly Benchmark (NAB) – artificialWithAnomaly

Description

Synthetic time series with injected anomalies from the Numenta Anomaly Benchmark (NAB). Designed for evaluating anomaly detection algorithms under controlled conditions. Labels available: Yes. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(nab_artificialWithAnomaly)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

Numenta Anomaly Benchmark (NAB) Dataset

References

Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms – the Numenta Anomaly Benchmark. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

Examples

```
data(nab_artificialWithAnomaly)
s <- nab_artificialWithAnomaly[[1]]
plot(ts(s$value), main = names(nab_artificialWithAnomaly)[1])
```

`nab_realAdExchange` *Numenta Anomaly Benchmark (NAB) – realAdExchange*

Description

Real-world time series with labeled anomalies from ad exchange data (NAB). Useful for evaluating detection methods on operational data. Labels available: Yes. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(nab_realAdExchange)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

Numenta Anomaly Benchmark (NAB) Dataset

References

Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms – the Numenta Anomaly Benchmark. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

Examples

```
data(nab_realAdExchange)
s <- nab_realAdExchange[[1]]
mean(s$event)
```

nab_realAWSCloudwatch *Numenta Anomaly Benchmark (NAB) realAWSCloudwatch*

Description

Data collection with real-world time-series. Real data from AWS Cloud with anomalies As part of the Numenta Anomaly Benchmark (NAB), this dataset contains time series with real and synthetic data. The real data comes from network monitoring and cloud computing. On the other hand, synthetic data simulate series with or without anomalies. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(nab_realAWSCloudwatch)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

Numenta Anomaly Benchmark (NAB) Dataset

References

Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms – the Numenta Anomaly Benchmark. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

Examples

```
data(nab_realAWSCloudwatch)
nab_grp <- nab_realAWSCloudwatch[[1]]
serie <- nab_grp[[1]]
```

`nab_realKnownCause` *Numenta Anomaly Benchmark (NAB) realKnownCause*

Description

Data collection with real-world time-series. Real data with anomalies As part of the Numenta Anomaly Benchmark (NAB), this dataset contains time series with real and synthetic data. The real data comes from network monitoring and cloud computing. On the other hand, synthetic data simulate series with or without anomalies. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(nab_realKnownCause)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

Numenta Anomaly Benchmark (NAB) Dataset

References

Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms – the Numenta Anomaly Benchmark. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

Examples

```
data(nab_realKnownCause)
nab_grp <- nab_realKnownCause[[1]]
serie <- nab_grp[[1]]
```

`nab_realTraffic` *Numenta Anomaly Benchmark (NAB) realTraffic*

Description

Data collection with real-world time-series. Real data from online data traffic with anomalies As part of the Numenta Anomaly Benchmark (NAB), this dataset contains time series with real and synthetic data. The real data comes from network monitoring and cloud computing. On the other hand, synthetic data simulate series with or without anomalies. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(nab_realTraffic)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

Numenta Anomaly Benchmark (NAB) Dataset

References

Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms – the Numenta Anomaly Benchmark. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

Examples

```
data(nab_realTraffic)
nab_grp <- nab_realTraffic[[1]]
serie <- nab_grp[[1]]
```

`nab_realTweets` *Numenta Anomaly Benchmark (NAB) realTweets*

Description

Real-world time series with labeled anomalies from Twitter volumes (NAB). Labels available: Yes. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(nab_realTweets)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

Numenta Anomaly Benchmark (NAB) Dataset

References

Lavin, A., & Ahmad, S. (2015). Evaluating real-time anomaly detection algorithms – the Numenta Anomaly Benchmark. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

Examples

```
data(nab_realTweets)
s <- nab_realTweets[[1]]
plot(ts(s$value), main = names(nab_realTweets)[1])
mean(s$event)
```

`oil_3w_Type_1` *Oil Wells Dataset – Type 1*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Type 0 removed from this version due to file size. Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_1)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_1)
s <- oil_3w_Type_1[[1]]
plot(ts(s$p_tpt), main = names(oil_3w_Type_1)[1], ylab = "value")
```

`oil_3w_Type_2` *Oil Wells Dataset – Type 2*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_2)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_2)
s <- oil_3w_Type_2[[1]]
mean(s$event)  # proportion of change points
```

`oil_3w_Type_4` *Oil Wells Dataset – Type 4*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_4)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_4)
serie <- oil_3w_Type_4[[1]]
```

`oil_3w_Type_5` *Oil Wells Dataset – Type 5*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_5)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_5)
serie <- oil_3w_Type_5[[1]]
```

oil_3w_Type_6 *Oil Wells Dataset – Type 6*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_6)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_6)
serie <- oil_3w_Type_6[[1]]
```

`oil_3w_Type_7` *Oil Wells Dataset – Type 7*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_7)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_7)
serie <- oil_3w_Type_7[[1]]
```

oil_3w_Type_8 *Oil Wells Dataset – Type 8*

Description

First realistic dataset with real events in oil well drilling. The data available in this package consist of time series already analyzed and applied in research experiments by the DAL group (Data Analytics Lab). The series are divided into 7 groups (Type_0, Type_1, Type_2, Type_4, Type_5, Type_6, Type_7 and Type_8). Creation date: 2019. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(oil_3w_Type_8)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCI Machine Learning Repository

References

3W dataset (UCI repository). See also: Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of change point detection methods. Signal Processing, 167, 107299.

Examples

```
data(oil_3w_Type_8)
serie <- oil_3w_Type_8[[1]]
```

trans_sax *SAX transformation*

Description

Symbolic Aggregate approXimation (SAX) discretization of a numeric time series. The series is z-normalized, quantile-binned, and mapped to an alphabet of size `alpha`.

Usage

```
trans_sax(alpha)
```

Arguments

`alpha`	Integer. Alphabet size (2–26).

Value

A `trans_sax` transformer object.

References

- Lin J, Keogh E, Lonardi S, Chiu B (2007). A symbolic representation of time series, with implications for streaming algorithms. Data Mining and Knowledge Discovery 15, 107–144.

Examples

```
library(daltoolbox)
vector <- 1:52
model <- trans_sax(alpha = 26)
model <- fit(model, vector)
xvector <- transform(model, vector)
print(xvector)
```

trans_xsax *XSAX transformation*

Description

Extended SAX (XSAX) discretization using a larger alphanumeric alphabet for finer symbolic resolution.

Usage

```
trans_xsax(alpha)
```

Arguments

`alpha`	Integer. Alphabet size (2–36).

Value

A `trans_xsax` transformer object.

References

- Ogasawara, E., Salles, R., Porto, F., Pacitti, E. Event Detection in Time Series. 1st ed. Cham: Springer Nature Switzerland, 2025. doi:10.1007/978-3-031-75941-3

See Also

trans_sax

Examples

```
library(daltoolbox)
vector <- 1:52
model <- trans_xsax(alpha = 36)
model <- fit(model, vector)
xvector <- transform(model, vector)
print(xvector)
```

`ucr_ecg` *UCR Anomaly Archive – ECG*

Description

Data collection with real-world time-series. Real ECG time series with labeled anomalous intervals. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(ucr_ecg)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCR Anomaly Archive

References

UCR Time Series Anomaly Archive. See also: Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(ucr_ecg)
# Access and plot a series
s <- ucr_ecg[[1]]
plot(ts(s$value), main = names(ucr_ecg)[1])
```

---

ucr_int_bleeding *UCR Anomaly Archive – Internal Bleeding*

Description

Data collection with real-world time-series. Real physiological time series with labeled anomalous intervals. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(ucr_int_bleeding)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCR Anomaly Archive

References

UCR Time Series Anomaly Archive. See also: Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(ucr_int_bleeding)
s <- ucr_int_bleeding[[1]]
plot(ts(s$value))
```

`ucr_nasa` *UCR Anomaly Archive – NASA Spacecraft*

Description

Data collection with real-world time-series. Real NASA spacecraft monitoring time series with labeled anomalous intervals. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(ucr_nasa)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCR Anomaly Archive

References

UCR Time Series Anomaly Archive. See also: Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(ucr_nasa)
s <- ucr_nasa[[1]]
mean(s$event)
```

ucr_power_demand *UCR Anomaly Archive – Italian Power Demand*

Description

Data collection with real-world time-series. Real power demand time series with labeled anomalous intervals. See cefet-rj-dal/united for detailed guidance on using this package and the other datasets available in it. Labels available? Yes

Usage

```
data(ucr_power_demand)
```

Format

A list of time series.

Details

This package ships a mini version of the dataset. Use loadfulldata() to download and load the full dataset from the URL stored in attr(url).

Source

UCR Anomaly Archive

References

UCR Time Series Anomaly Archive. See also: Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58.

Examples

```
data(ucr_power_demand)
s <- ucr_power_demand[[1]]
summary(s$value)
```

Index