

# Package ‘harbinger’

April 24, 2025

**Title** A Unified Time Series Event Detection Framework

**Version** 1.1.717

**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](https://doi.org/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** 7.3.2

**Imports** stats, daltoolbox, tsm, dtwclust, rugarch, forecast, ggplot2,  
changepoint, strucchange, stringr, wavelets, hht, zoo, dplyr

**NeedsCompilation** no

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

Eduardo Bezerra [aut],  
 Rafaelli Coutinho [aut],  
 Esther Pacitti [aut],  
 Fabio Porto [aut],  
 CEFET/RJ [cph] (Federal Center for Technological Education of Rio de  
 Janeiro)

**Maintainer** Eduardo Ogasawara <eogasawara@ieee.org>

**Repository** CRAN

**Date/Publication** 2025-04-24 02:40:02 UTC

## Contents

|                                 |    |
|---------------------------------|----|
| detect . . . . .                | 3  |
| examples_anomalies . . . . .    | 4  |
| examples_changepoints . . . . . | 5  |
| examples_harbinger . . . . .    | 6  |
| examples_motifs . . . . .       | 7  |
| hanct_dtw . . . . .             | 7  |
| hanct_kmeans . . . . .          | 8  |
| hanc_ml . . . . .               | 9  |
| hanr_arima . . . . .            | 10 |
| hanr_emd . . . . .              | 11 |
| hanr_fbiad . . . . .            | 12 |
| hanr_fft . . . . .              | 13 |
| hanr_garch . . . . .            | 14 |
| hanr_histogram . . . . .        | 15 |
| hanr_ml . . . . .               | 16 |
| hanr_red . . . . .              | 17 |
| hanr_remd . . . . .             | 18 |
| hanr_wavelet . . . . .          | 19 |
| han_autoencoder . . . . .       | 20 |
| harbinger . . . . .             | 20 |
| harutils . . . . .              | 21 |
| har_ensemble . . . . .          | 21 |
| har_eval . . . . .              | 22 |
| har_eval_soft . . . . .         | 23 |
| har_plot . . . . .              | 24 |
| hcp_amoc . . . . .              | 25 |
| hcp_binseg . . . . .            | 26 |
| hcp_cf_arima . . . . .          | 27 |
| hcp_cf_ets . . . . .            | 28 |
| hcp_cf_lr . . . . .             | 29 |
| hcp_chow . . . . .              | 30 |
| hcp_garch . . . . .             | 31 |
| hcp_gft . . . . .               | 32 |
| hcp_pelt . . . . .              | 32 |
| hcp_red . . . . .               | 33 |

*detect*

3

hcp\_scp . . . . .

34

hdis\_mp . . . . .

35

hdis\_sax . . . . .

36

hmo\_mp . . . . .

37

hmo\_sax . . . . .

38

hmo\_xsax . . . . .

39

hmu\_pca . . . . .

40

mas . . . . .

41

trans\_sax . . . . .

42

trans\_xsax . . . . .

43

**Index**

**44**

---

|        |                                     |
|--------|-------------------------------------|
| detect | <i>Detect events in time series</i> |
|--------|-------------------------------------|

---

**Description**

Event detection using a fitted Harbinger model

**Usage**

detect(obj, ...)

**Arguments**

- objharbinger object
- ...optional arguments.

**Value**

a data frame with the index of observations and if they are identified or not as an event, and their type

**Examples**

#See examples of detectors for anomalies, change points, and motifs  
#at <https://cefet-rj-dal.github.io/harbinger>

---

|                    |                                          |
|--------------------|------------------------------------------|
| examples_anomalies | <i>Time series for anomaly detection</i> |
|--------------------|------------------------------------------|

---

## Description

A list of time series for anomaly detection

- simple: a simple synthetic time series
- contextual: a contextual synthetic time series
- simple: a trend synthetic time series
- trend: a simple synthetic time series
- multiple: a multiple anomalies synthetic time series
- sequence: a sequence synthetic time series
- tt: a train-test synthetic time series
- tt\_warped: a warped train-test synthetic time series

#'

## Usage

```
data(examples_anomalies)
```

## Format

A list of time series for anomaly detection.

## Source

[Harbinger package](#)

## References

[Harbinger package](#)

## Examples

```
data(examples_anomalies)
serie <- examples_anomalies$simple
```

---

examples\_changepoints *Time series for change point detection*

---

## Description

A list of time series for change point

- simple: a simple synthetic time series
- sinusoidal: a sinusoidal synthetic time series
- incremental: a incremental synthetic time series
- abrupt: a abrupt synthetic time series
- volatility: a volatility synthetic time series

#'

## Usage

```
data(examples_changepoints)
```

## Format

A list of time series for change point detection.

## Source

Harbinger package

## References

Harbinger package

## Examples

```
data(examples_changepoints)
serie <- examples_changepoints$simple
```

---

|                    |                                        |
|--------------------|----------------------------------------|
| examples_harbinger | <i>Time series for event detection</i> |
|--------------------|----------------------------------------|

---

## Description

A list of time series for event detection

- nonstationarity: a synthetic nonstationarity time series
- global\_temperature\_yearly: yearly global temperature of the world
- global\_temperature\_monthly: monthly global temperature of the world
- multidimensional: multidimensional time 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](#)

## Examples

```
data(examples_harbinger)
serie <- examples_harbinger$seattle_daily
```

---

|                 |                                               |
|-----------------|-----------------------------------------------|
| examples_motifs | <i>Time series for change point detection</i> |
|-----------------|-----------------------------------------------|

---

**Description**

A list of time series for change point

- simple: a simple synthetic time series
- mitdb100: sample of mitdb 100 time series
- mitdb102: sample of mitdb 102 time series

#'

**Usage**

```
data(examples_motifs)
```

**Format**

A list of time series for motif discovery.

**Source**

Harbinger package

**References**

Harbinger package

**Examples**

```
data(examples_motifs)
serie <- examples_motifs$simple
```

---

|           |                                   |
|-----------|-----------------------------------|
| hanct_dtw | <i>Anomaly detector using DTW</i> |
|-----------|-----------------------------------|

---

**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

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series regression model
model <- hanct_dtw()

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

detection <- detect(model, dataset$serie)

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

---

|              |                                      |
|--------------|--------------------------------------|
| hanct_kmeans | <i>Anomaly detector using kmeans</i> |
|--------------|--------------------------------------|

---

**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

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series regression model
model <- hanct_kmeans()

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

detection <- detect(model, dataset$serie)

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

---

hanc\_ml

*Anomaly detector based on machine learning classification*

---

**Description**

Anomaly detection using daltoolbox classification. A training and test set should be used. The training set must contain labeled events. A set of preconfigured of classification methods are described in <https://cefet-rj-dal.github.io/daltoolbox/>. They include: cla\_majority, cla\_dtree, cla\_knn, cla\_mlp, cla\_nb, cla\_rf, cla\_svm

**Usage**

```
hanc_ml(model)
```

**Arguments**

model                      DALToolbox classification model

**Value**

hanc\_ml object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

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

# normalizing the data
norm <- minmax()
norm <- fit(norm, train)
train_n <- transform(norm, train)

# establishing decision tree method
model <- hanc_ml(cla_dtree("event", slevels))

# fitting the model
model <- fit(model, train_n)

# evaluating the detections during testing
test_n <- transform(norm, test)

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

---

hanr\_arima

*Anomaly detector using ARIMA.*

---

**Description**

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

**Usage**

```
hanr_arima()
```

**Value**

hanr\_arima object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series regression model
model <- hanr_arima()

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

detection <- detect(model, dataset$serie)

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

---

hanr\_emd

*Anomaly detector using EMD*

---

**Description**

Anomaly detection using EMD 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 hht library.

**Usage**

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

**Arguments**

|        |        |
|--------|--------|
| noise  | noise  |
| trials | trials |

**Value**

hanr\_emd object

**Examples**

```

library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series emd detector
model <- hanr_emd()

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

detection <- detect(model, dataset$serie)

# filtering detected events
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](https://doi.org/10.1109/IJCNN55064.2022.9892088).

**Examples**

```

library(daltoolbox)

#loading the example database
data(examples_anomalies)

```

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

# setting up time series regression model
model <- hanr_fbiad()

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

detection <- detect(model, dataset$serie)

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

---

hanr\_fft

*Anomaly detector using FFT*

---

## Description

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

## Usage

```
hanr_fft()
```

## Value

hanr\_fft object

## Examples

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

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

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

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

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

---

hanr\_garch

*Anomaly detector using GARCH*

---

## Description

Anomaly detection using GARCH The GARCH model adjusts to the time series. Observations distant from the model are labeled as anomalies. It wraps the ugarch model presented in the rugarch library.

## Usage

```
hanr_garch()
```

## Value

hanr\_garch object

## Examples

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series regression model
model <- hanr_garch()

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

detection <- detect(model, dataset$serie)

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

---

|                |                                         |
|----------------|-----------------------------------------|
| hanr_histogram | <i>Anomaly detector using histogram</i> |
|----------------|-----------------------------------------|

---

**Description**

Anomaly detector using histogram

**Usage**

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

**Arguments**

`density_threshold`

It is the minimum frequency for a bin to not be considered an anomaly. Default value is 5%.

**Value**

hanr\_histogram object histogram based method to detect anomalies in time series. Bins with smaller amount of observations are considered anomalies. Values below first bin or above last bin are also considered anomalies.>.

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series regression model
model <- hanr_histogram()

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

detection <- detect(model, dataset$serie)

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

---

hanr\_ml

*Anomaly detector based on machine learning regression.*


---

### Description

Anomaly detection using daltoolbox regression The regression model adjusts to the time series. Observations distant from the model are labeled as anomalies. A set of preconfigured regression methods are described in <https://cefet-rj-dal.github.io/daltoolbox/>. They include: ts\_elm, ts\_conv1d, ts\_lstm, ts\_mlp, ts\_rf, ts\_svm

### Usage

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

### Arguments

|         |                             |
|---------|-----------------------------|
| model   | DALToolbox regression model |
| sw_size | sliding window size         |

### Value

hanr\_ml object

### Examples

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series regression model
model <- hanr_ml(ts_elm(ts_norm_gminmax(), input_size=4, nhid=3, actfun="purelin"))

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

detection <- detect(model, dataset$serie)

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

---

hanr\_red

---

*Anomaly and change point detector using RED*

---

**Description**

Anomaly and change point detection using RED The RED 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_red(sw_size = 30, noise = 0.001, trials = 5)
```

**Arguments**

|         |                                  |
|---------|----------------------------------|
| sw_size | sliding window size (default 30) |
| noise   | noise                            |
| trials  | trials                           |

**Value**

hanr\_red object

**Examples**

```
library(daltoolbox)
library(zoo)

#loading the example database
data(examples_anomalies)

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

# setting up time series emd detector
model <- hanr_red()

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

detection <- detect(model, dataset$serie)

# filtering detected events
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  | noise  |
| trials | trials |

## Value

hanr\_remd object

## Examples

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series emd detector
model <- hanr_remd()

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

detection <- detect(model, dataset$serie)

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

---

|              |                                       |
|--------------|---------------------------------------|
| hanr_wavelet | <i>Anomaly detector using Wavelet</i> |
|--------------|---------------------------------------|

---

**Description**

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

**Usage**

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

**Arguments**

filter                    Availables wavelet filters: haar, d4, la8, bl14, c6

**Value**

hanr\_wavelet object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

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

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

detection <- detect(model, dataset$serie)

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

---

|                 |                                           |
|-----------------|-------------------------------------------|
| han_autoencoder | <i>Anomaly detector using autoencoder</i> |
|-----------------|-------------------------------------------|

---

**Description**

Anomaly detector using autoencoder

**Usage**

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

**Arguments**

|              |                                                                                                   |
|--------------|---------------------------------------------------------------------------------------------------|
| input_size   | Establish the input size for the autoencoder anomaly detector. It is the size of the output also. |
| encode_size  | The encode size for the autoencoder.                                                              |
| encoderclass | The class of daltoolbox encoder-decoder.                                                          |
| ...          | optional arguments for encoder-decoder class.                                                     |

**Value**

han\_autoencoder object histogram based method to detect anomalies in time series. Bins with smaller amount of observations are considered anomalies. Values below first bin or above last bin are also considered anomalies.>.

**Examples**

```
#See an example of using `autoenc_ed` at this
#https://github.com/cefet-rj-dal/harbinger/blob/master/anomalies/han_autoenc_ed
```

---

|           |                  |
|-----------|------------------|
| harbinger | <i>Harbinger</i> |
|-----------|------------------|

---

**Description**

Ancestor class for time series event detection

**Usage**

```
harbinger()
```

**Value**

Harbinger object

**Examples**

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

---

harutils

*Harbinger Utils*

---

**Description**

Utility class that contains major distance measures, threshold limits, and outliers grouping functions

**Usage**

```
harutils()
```

**Value**

Harbinger Utils

**Examples**

```
# See ?hanc_ml for an example of anomaly detection using machine learning classification
```

---

har\_ensemble

*Harbinger Ensemble*

---

**Description**

Ensemble detector

**Usage**

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

**Arguments**

```
...          list of detectors
```

**Value**

Harbinger object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time series emd detector
model <- har_ensemble(hanr_fbiad(), hanr_arima(), hanr_emd())

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

detection <- detect(model, dataset$serie)

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

---

har\_eval

---

*Evaluation of event detection*


---

**Description**

Evaluation of event detection (traditional hard evaluation)

**Usage**

```
har_eval()
```

**Value**

har\_eval object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

dataset <- examples_anomalies$simple
head(dataset)

# setting up time change point using GARCH
model <- hcp_garch()
```

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

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

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

# evaluating the detections
evaluation <- evaluate(har_eval(), detection$event, dataset$event)
print(evaluation$confMatrix)

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

---

har\_eval\_soft*Evaluation of event detection*

---

**Description**

Evaluation of event detection using SoftED [doi:10.48550/arXiv.2304.00439](https://doi.org/10.48550/arXiv.2304.00439)

**Usage**

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

**Arguments**

sw\_size                      tolerance window size

**Value**

har\_eval\_soft object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time change point using GARCH
model <- hcp_garch()
```

```

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

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

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

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

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

```

---

har\_plot

*Plot event detection on a time series*


---

### Description

It accepts as harbinger, a time series, a data.frame of events, a parameter to mark the detected change points, a threshold for the y-axis and an index for the time series

### Usage

```

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

```

### Arguments

|           |                    |
|-----------|--------------------|
| obj       | harbinger detector |
| serie     | time series        |
| detection | detection          |
| event     | events             |
| mark.cp   | show change points |
| ylim      | limits for y-axis  |

|           |                                                                                                                                |
|-----------|--------------------------------------------------------------------------------------------------------------------------------|
| idx       | labels for x observations                                                                                                      |
| pointsize | default point size                                                                                                             |
| colors    | default colors for event detection: green is TP, blue is FN, red is FP, purple means observations that are part of a sequence. |

### Value

A time series plot with marked events

### Examples

```
library(daltoolbox)

#loading the example database
data(examples_anomalies)

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

# setting up time change point using GARCH
model <- hanr_arima()

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

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

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

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

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

---

hcp\_amoc

*At most one change (AMOC) method*

---

### Description

Change-point detection method that focus on identify one change point in mean/variance [doi: 10.1093/biomet/57.1.1](https://doi.org/10.1093/biomet/57.1.1). It wraps the amoc implementation available in the changepoint library.

**Usage**

```
hcp_amoc()
```

**Value**

hcp\_amoc object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_amoc()

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

# execute the detection method
detection <- detect(model, dataset$serie)

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

---

hcp\_binseg

*Binary segmentation (BinSeg) method*

---

**Description**

Change-point detection method that focus on identify change points in mean/variance [doi:10.2307/2529204](https://doi.org/10.2307/2529204). It wraps the BinSeg implementation available in the changepoint library.

**Usage**

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

**Arguments**

Q                      The maximum number of change-points to search for using the BinSeg method

**Value**

hcp\_binseg object

## Examples

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_binseg()

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

# execute the detection method
detection <- detect(model, dataset$serie)

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

---

hcp\_cf\_arima

*Change Finder using ARIMA*

---

## Description

Change-point detection is related to event/trend change detection. Change Finder ARIMA detects change points based on deviations relative to ARIMA model [doi:10.1109/TKDE.2006.1599387](https://doi.org/10.1109/TKDE.2006.1599387). It wraps the ARIMA model presented in the forecast library.

## Usage

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

## Arguments

sw\_size                      Sliding window size

## Value

hcp\_cf\_arima object

## Examples

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_cf_arima()

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

detection <- detect(model, dataset$serie)

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

---

hcp\_cf\_ets

*Change Finder using ETS*

---

## Description

Change-point detection is related to event/trend change detection. Change Finder ETS detects change points based on deviations relative to trend component (T), a seasonal component (S), and an error term (E) model [doi:10.1109/TKDE.2006.1599387](https://doi.org/10.1109/TKDE.2006.1599387). It wraps the ETS model presented in the forecast library.

## Usage

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

## Arguments

sw\_size                      Sliding window size

## Value

hcp\_cf\_ets object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_cf_ets()

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

detection <- detect(model, dataset$serie)

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

---

hcp\_cf\_lr*Change Finder using LR*

---

**Description**

Change-point detection is related to event/trend change detection. Change Finder LR detects change points based on deviations relative to linear regression model [doi:10.1109/TKDE.2006.1599387](https://doi.org/10.1109/TKDE.2006.1599387).

**Usage**

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

**Arguments**

sw\_size                      Sliding window size

**Value**

hcp\_cf\_lr object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)
```

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

# setting up change point method
model <- hcp_cf_lr()

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

detection <- detect(model, dataset$serie)

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

---

hcp\_chow

*Chow test method*


---

## Description

Change-point detection method that focus on identifying structural changes [doi:10.18637/jss.v007.i02](https://doi.org/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)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_chow()

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

# execute the detection method
```

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

# filtering detected events
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](https://doi.org/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

### Examples

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

#Using volatility example
dataset <- examples_changepoints$volatility
head(dataset)

# setting up change point method
model <- hcp_garch()

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

detection <- detect(model, dataset$serie)

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

---

|         |                                           |
|---------|-------------------------------------------|
| hcp_gft | <i>Generalized Fluctuation Test (GFT)</i> |
|---------|-------------------------------------------|

---

**Description**

GFT detection method focuses on identifying structural changes [doi:10.18637/jss.v007.i02](https://doi.org/10.18637/jss.v007.i02). It wraps the breakpoints implementation available in the strucchange library.

**Usage**

```
hcp_gft()
```

**Value**

hcp\_chow object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_gft()

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

# execute the detection method
detection <- detect(model, dataset$serie)

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

---

|          |                                               |
|----------|-----------------------------------------------|
| hcp_pelt | <i>Pruned exact linear time (PELT) method</i> |
|----------|-----------------------------------------------|

---

**Description**

Change-point detection method that focus on identifying multiple exact change points in mean/variance [doi:10.1080/01621459.2012.737745](https://doi.org/10.1080/01621459.2012.737745). It wraps the BinSeg implementation available in the change-point library.

**Usage**

```
hcp_pelt()
```

**Value**

hcp\_pelt object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_pelt()

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

# execute the detection method
detection <- detect(model, dataset$serie)

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

---

hcp\_red

*Anomaly and change point detector using RED*

---

**Description**

Anomaly and change point detection using RED The RED 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**

```
hcp_red(
  sw_size = 30,
  noise = 0.001,
  trials = 5,
  red_cp = TRUE,
  volatility_cp = TRUE,
  trend_cp = TRUE
)
```

**Arguments**

|               |                                  |
|---------------|----------------------------------|
| sw_size       | sliding window size (default 30) |
| noise         | noise                            |
| trials        | trials                           |
| red_cp        | red change point                 |
| volatility_cp | volatility change point          |
| trend_cp      | trend change point               |

**Value**

hcp\_red object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_red()

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

# execute the detection method
detection <- detect(model, dataset$serie)

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

---

|         |                             |
|---------|-----------------------------|
| hcp_scp | <i>Seminal change point</i> |
|---------|-----------------------------|

---

**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

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_changepoints)

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

# setting up change point method
model <- hcp_scp()

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

detection <- detect(model, dataset$serie)

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

---

hdis\_mp

*Discord discovery using Matrix Profile*


---

**Description**

Discord discovery using Matrix Profile [doi:10.32614/RJ-2020-021](https://doi.org/10.32614/RJ-2020-021)

**Usage**

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

**Arguments**

|      |                                                                                                                                          |
|------|------------------------------------------------------------------------------------------------------------------------------------------|
| mode | mode of computing distance between sequences. Available options include: "stomp", "stamp", "simple", "mstomp", "scrimp", "valmod", "pmp" |
| w    | word size                                                                                                                                |
| qtd  | number of occurrences to be classified as discords                                                                                       |

**Value**

hdis\_mp object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_motifs)

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

# setting up discord discovery method
model <- hdis_mp("stamp", 4, 3)

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

detection <- detect(model, dataset$serie)

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

---

hdis\_sax

*Discord discovery using SAX*


---

**Description**

Discord discovery using SAX [doi:10.1007/s10618-007-0064-z](https://doi.org/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

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_motifs)

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

# setting up discord discovery method
model <- hdis_sax(26, 3, 3)

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

detection <- detect(model, dataset$serie)

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

---

hmo\_mp*Motif discovery using Matrix Profile*

---

**Description**

Motif discovery using Matrix Profile [doi:10.32614/RJ-2020-021](https://doi.org/10.32614/RJ-2020-021)

**Usage**

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

**Arguments**

|      |                                                                                                                                          |
|------|------------------------------------------------------------------------------------------------------------------------------------------|
| mode | mode of computing distance between sequences. Available options include: "stomp", "stamp", "simple", "mstomp", "scrimp", "valmod", "pmp" |
| w    | word size                                                                                                                                |
| qtd  | number of occurrences to be classified as motifs                                                                                         |

**Value**

hmo\_mp object

**Examples**

```

library(daltoolbox)

#loading the example database
data(examples_motifs)

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

# setting up motif discovery method
model <- hmo_mp("stamp", 4, 3)

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

detection <- detect(model, dataset$serie)

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

```

hmo\_sax

*Motif discovery using SAX***Description**

Motif discovery using SAX [doi:10.1007/s10618-007-0064-z](https://doi.org/10.1007/s10618-007-0064-z)

**Usage**

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

**Arguments**

|     |                                                  |
|-----|--------------------------------------------------|
| a   | alphabet size                                    |
| w   | word size                                        |
| qtd | number of occurrences to be classified as motifs |

**Value**

hmo\_sax object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_motifs)

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

# setting up motif discovery method
model <- hmo_sax(26, 3, 3)

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

detection <- detect(model, dataset$serie)

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

---

hmo\_xsax*Motif discovery using xsax*

---

**Description**

Motif discovery using xsax [doi:10.1007/s10618-007-0064-z](https://doi.org/10.1007/s10618-007-0064-z)

**Usage**

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

**Arguments**

|     |                                                  |
|-----|--------------------------------------------------|
| a   | alphabet size                                    |
| w   | word size                                        |
| qtd | number of occurrences to be classified as motifs |

**Value**

hmo\_xsax object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_motifs)

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

# setting up motif discovery method
model <- hmo_xsax(37, 3, 3)

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

detection <- detect(model, dataset$serie)

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

---

hmu\_pca

*Multivariate anomaly detector using PCA*

---

**Description**

Multivariate anomaly detector using PCA [doi:10.1016/0098-3004\(93\)90090-R](https://doi.org/10.1016/0098-3004(93)90090-R)

**Usage**

```
hmu_pca()
```

**Value**

hmu\_pca object

**Examples**

```
library(daltoolbox)

#loading the example database
data(examples_harbinger)

#Using the time series 9
dataset <- examples_harbinger$multidimensional
head(dataset)

# establishing hmu_pca method
```

```

model <- hmu_pca()

# fitting the model using the two columns of the dataset
model <- fit(model, dataset[,1:2])

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

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

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

```

mas

*Moving average smoothing*

## Description

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

## Usage

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

## Arguments

|                    |                                             |
|--------------------|---------------------------------------------|
| <code>x</code>     | A numeric vector or univariate time series. |
| <code>order</code> | Order of moving average smoother.           |

## Details

The moving average smoother transformation is given by

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

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

```
#loading the example database
data(examples_changepoints)

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

# setting up change point method
ma <- mas(dataset$serie, 5)
```

---

|           |     |
|-----------|-----|
| trans_sax | SAX |
|-----------|-----|

---

Description

SAX

Usage

```
trans_sax(alpha)
```

Arguments

alpha                      alphabet

Value

obj

Examples

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

---

|            |      |
|------------|------|
| trans_xsax | XSAX |
|------------|------|

---

**Description**

XSAX

**Usage**

```
trans_xsax(alpha)
```

**Arguments**

alpha                  alphabet

**Value**

obj

**Examples**

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

# Index

- \* **average**
  - mas, [41](#)
- \* **datasets**
  - examples\_anomalies, [4](#)
  - examples\_changepoints, [5](#)
  - examples\_harbinger, [6](#)
  - examples\_motifs, [7](#)
- \* **moving**
  - mas, [41](#)
- \* **series**
  - mas, [41](#)
- \* **smoother**
  - mas, [41](#)
- \* **time**
  - mas, [41](#)
- \* **transform**
  - mas, [41](#)

[detect](#), [3](#)

[examples\\_anomalies](#), [4](#)

[examples\\_changepoints](#), [5](#)

[examples\\_harbinger](#), [6](#)

[examples\\_motifs](#), [7](#)

[han\\_autoencoder](#), [20](#)

[hanc\\_ml](#), [9](#)

[hanct\\_dtw](#), [7](#)

[hanct\\_kmeans](#), [8](#)

[hanr\\_arima](#), [10](#)

[hanr\\_emd](#), [11](#)

[hanr\\_fbiad](#), [12](#)

[hanr\\_fft](#), [13](#)

[hanr\\_garch](#), [14](#)

[hanr\\_histogram](#), [15](#)

[hanr\\_ml](#), [16](#)

[hanr\\_red](#), [17](#)

[hanr\\_remd](#), [18](#)

[hanr\\_wavelet](#), [19](#)

[har\\_ensemble](#), [21](#)

[har\\_eval](#), [22](#)

[har\\_eval\\_soft](#), [23](#)

[har\\_plot](#), [24](#)

[harbinger](#), [20](#)

[harutils](#), [21](#)

[hcp\\_amoc](#), [25](#)

[hcp\\_binseg](#), [26](#)

[hcp\\_cf\\_arima](#), [27](#)

[hcp\\_cf\\_ets](#), [28](#)

[hcp\\_cf\\_lr](#), [29](#)

[hcp\\_chow](#), [30](#)

[hcp\\_garch](#), [31](#)

[hcp\\_gft](#), [32](#)

[hcp\\_pelt](#), [32](#)

[hcp\\_red](#), [33](#)

[hcp\\_scp](#), [34](#)

[hdis\\_mp](#), [35](#)

[hdis\\_sax](#), [36](#)

[hmo\\_mp](#), [37](#)

[hmo\\_sax](#), [38](#)

[hmo\\_xsax](#), [39](#)

[hmu\\_pca](#), [40](#)

[ma](#), [41](#)

[mas](#), [41](#)

[trans\\_sax](#), [42](#)

[trans\\_xsax](#), [43](#)