Package: edecob (via r-universe)

August 30, 2024

Title Event Detection Using Confidence Bounds

Version 1.2.2

Description Detects sustained change in digital bio-marker data using simultaneous confidence bands. Accounts for noise using an auto-regressive model. Based on Buehlmann (1998) ``Sieve bootstrap for smoothing in nonstationary time series'' <doi:10.1214/aos/1030563978>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Depends R (>= 3.5.0)

Imports stats (>= 3.5.0), ggplot2 (>= 3.1.0), rlang (>= 0.4.0), utils (>= 3.5.0), graphics (>= 3.5.0)

Suggests survival (>= 2.43)

Repository https://manzhch.r-universe.dev

RemoteUrl https://github.com/manzhch/edecob

RemoteRef HEAD

RemoteSha 21b0d2ead6e111d54795dd4c36abe187da927cca

Contents

`bt_smoother` *Bootstrap the Smoother*

Description

First, fit an autoregressive model on the residuals of the smoother. Then bootstrap the errors of the autoregressive model. Afterwards, reconstruct the measurements by adding the bootstrapped error, the autoregressive model, and the smoother. We can again calculate the smoother using these reconstructed measurements to obtain the bootstrapped smoother (which can later be used to construct the simultaneous confidence bounds). For details see below.

Usage

```
bt_smoother(
  data,
  smoother,
  resample_method,
  smoother_pts,
  resid,
  bt_tot_rep,
  ...
)
```

Arguments

`data`	A data frame in long format containing the data for which events is to be detected. This means that each measurement corresponds to a row and the columns are (in order): source (the device or person from which the data was collected), point in time, and measurement value. If custom detection bounds are chosen, the folloing two columns must be added: lower detection bound, and upper detection bound. The source is expected to be a string; the time point are integers; measurements, and detection bounds are expected to be numerical. The detection bounds are in absolute value in the same unit as the values and each is expected to be identical for the same source. In case detection is wanted for a one sided change (e.g. give an event if the confidence bounds drop below a threshold) then the upper or lower detection bound can be chosen to be Inf or -Inf respectively.
`smoother`	A string specifying which smoother is to be used. Use `mov_med` for the moving median. When using the moving median, the parameter `med_win` must be given to specify the size of the window over which the moving median is to be taken. Defaults to the moving median.
`resample_method`	A string that determines how to resample the errors of the autoregression for the bootstrap. Default is `all`, meaning that the epsilon of a certain time point are

	resampled from all time points. `past` only considers epsilon corresponding to a time point prior to the one being resampled. `window` resamples the epsilon from the window given by `resample_win`.
`smoother_pts`	A data frame containing the smoother with columns time_point and value. Preferably the output of one of the smoother functions within this package.
`resid`	A vector of the same length as the number of rows of data containing the difference between the smoother and the measurements. Preferably the output of `smoother_resid`.
`bt_tot_rep`	The number of iterations for the bootstrap computation. Because of run time, it is recommended to keep this number below 500. Defaults to 100.
`...`	Additional parameters to be given to the function. Possible parameters for the model are `order` and `min_pts_in_win`. For the moving median, a `med_win` is required. When resampling from window, a `resample_win` may be given. The parameter `min_pts_in_win` defines the minimal number of measurements required to be in the time window for the median to be calculated. Defaults to 1. If the parameter `order` is given, that number will be the (maximal) order of the autoregressive model. If no `order` is given, it will be determined using the Akaike information criterion. When the moving median is used as the smoother, `med_win` is expected. If no `med_win` is given, it will default to `c(-42, 42)`. When resampling from window, one can choose the window size for the resampling window with `resample_win` by giving a window like e.g. `c(-14,14)`..

Details

An autoregressive (AR) model is used for the residuals of the smoother:

$$Y(t) = S(t) + \eta(t)$$

$$\eta(t) = \sum_{j=1}^{p} \phi_j \eta(t-j) + \epsilon$$

where t is the point in time, $Y(t)$ the data point, $S(t)$ a smoother, $\eta(t)$ the residual of the smoother, p the order of the AR model, ϕ_j the coefficients of the AR model, and ϵ the error of the AR model.

The bootstrap procedure is as follows:

1. Compute the smoother $S(t)$.
2. Compute the residuals $\eta(t_i) = Y(t_i) - S(t_i)$.
3. Fit an AR(p) model to $\eta(t_i)$ to obtain the coefficients $\phi_1, \ldots, \phi_p$ and residuals $\epsilon(t_i) = \eta(t_i) - \sum_{j=1}^{p} \phi_j \eta(t_i - t_{i-j})$.
4. Resample $\epsilon(t_i)*$ from $\epsilon(t_{p+1}), \ldots, \epsilon(t_n)$ to obtain

   $$Y(t_i)* = S(t_i) + \eta(t_i)*,$$

   where

   $$\eta(t_i)* = \sum_{j=1}^{p} \phi_j \eta(t_{i-j}) * + \epsilon(t_{i-j}) * .$$

5. Compute $S(.)* = g(Y(t_1), \ldots, Y(t_n))$ where g is the function with which the smoother is calculated.
6. Repeat steps 4 and 5 `bt_tot_rep` times.

Value

A data frame containing the bootstrap repetitions of the smoother. The column are subject identifier, time point, value, and the bootstrap repetition the value corrsponds to.

References

Bühlmann, P. (1998). Sieve Bootstrap for Smoothing in Nonstationary Time Series. *The Annals of Statistics*, 26(1), 48-83.

`conf_band` *Confidence Bounds of the Smoother*

Description

Calculate the confidence bounds of the smoother function using the bootstrap.

Usage

```
conf_band(bt_smoother, smoother_pts, bt_tot_rep, conf_band_lvl)
```

Arguments

`bt_smoother`	A data frame containing the bootstrapped smoother. Use the output of `bt_smoother`.
`smoother_pts`	A data frame containing the smoother. Preferably the output of one of the smoother functions included in this package.
`bt_tot_rep`	The number of iterations for the bootstrap computation. Because of run time, it is recommended to keep this number below 500. Defaults to 100.
`conf_band_lvl`	The confidence level for the simultaneous confidence bands. Defaults to 0.95. When detection of events using only the smoother is desired, `conf_band_lvl` can be chosen to be 0.

Details

The procedure is as follows:

1. We compute the quantiles
$$q_x(t_i), q_{1-x}(t_i) i = 1, \ldots, N$$
where
$$q_x(t_i) = inf\,\{u; P^*[S(t_i)^*_b - S(t_i) \leq u] \geq x\}$$
is a pointwise bootstrap quantile, $S(t_i)^*_b$ the bootstrapped smoother, and N the number of measurements or rows in `data`, in our case the number of rows.

2. We vary the pointwise error x until

$$P^*[q_x(t_i) \leq S(t_i)_b^* - S(t_i) \leq q_{1-x}(t_i) \forall i = 1, \ldots, N] \approx 1 - \alpha.$$

   In other words, until the ratio of bootstrap curves that have all their points within $[q_x(t_i), q_{1-x}(t_i)]$ is approximately $1 - \alpha$.

3. We define

$$I_n(t_i) = [S(t_i) + q_x(t_i), S(t_i) + q_{1-x}(t_i)] \forall i = 1, \ldots, N$$

   the confidence bounds. Then $I_n(t_i); i = 1, \ldots, N$ is a consistent simultaneous confidence band of level $1 - \alpha$.

Value

A data frame containing the upper confidence bound, the lower confidence bound, and the time point corresponding to the bounds.

References

Bühlmann, P. (1998). Sieve Bootstrap for Smoothing in Nonstationary Time Series. *The Annals of Statistics*, 26(1), 48-83.

---

detect_event *Detect Events*

Description

Detect events using the confidence bounds. An event is detected if all the points of the upper or lower bound of the confidence band are below or above the threshold for min_change_dur consecutive days.

Usage

```
detect_event(conf_band, detec_lower, detec_upper, min_change_dur)
```

Arguments

conf_band	A data frame containing the confidence bounds. Ideally the output of conf_band.
detec_lower	The lower detection bound in the same units as the values in data.
detec_upper	The upper detection bound in the same units as the values in data.
min_change_dur	The minimal number of time units that the confidence bounds need to stay inside the detection bounds in order for an event to be detected. Defaults to 84, i.e. 12 weeks.

Value

A list of four values:

`event_detected`	gives whether an event was detected
`event_onset`	gives the time_point at which the event was detected
`event_duration`	gives the duration the event is sustained
`event_stop`	gives whether the detected event is censored

`edecob` *Event DEtection using COnfidence Bounds*

Description

Calculate a smoother of longitudinal data of the same measure and bootstrap the errors of the autoregressive model fitted on the smoother to form simultaneous confidence bounds of a certain level (mathematical details below). Define an event if the simultaneous confidence bound is within a chosen interval for a predefined amount of time. When data from multiple sources is provided, the calculation will be done separately for each source.

Usage

```
edecob(
  data,
  smoother = "mov_med",
  resample_method = "all",
  min_change_dur = 84,
  conf_band_lvl = 0.95,
  bt_tot_rep = 100,
  time_unit = "day",
  detect = "below",
  detect_factor = 1,
  bline_period = 14,
  ...
)
```

Arguments

`data`	A data frame in long format containing the data for which events is to be detected. This means that each measurement corresponds to a row and the columns are (in order): source (the device or person from which the data was collected), point in time, and measurement value. If custom detection bounds are chosen, the folloing two columns must be added: lower detection bound, and upper detection bound. The source is expected to be a string; the time point are integers; measurements, and detection bounds are expected to be numerical. The detection bounds are in

	absolute value in the same unit as the values and each is expected to be identical for the same source. In case detection is wanted for a one sided change (e.g. give an event if the confidence bounds drop below a threshold) then the upper or lower detection bound can be chosen to be Inf or -Inf respectively.
`smoother`	A string specifying which smoother is to be used. Use `mov_med` for the moving median. When using the moving median, the parameter `med_win` must be given to specify the size of the window over which the moving median is to be taken. Defaults to the moving median.
`resample_method`	A string that determines how to resample the errors of the autoregression for the bootstrap. Default is `all`, meaning that the epsilon of a certain time point are resampled from all time points. `past` only considers epsilon corresponding to a time point prior to the one being resampled. `window` resamples the epsilon from the window given by `resample_win`.
`min_change_dur`	The minimal number of time units that the confidence bounds need to stay inside the detection bounds in order for an event to be detected. Defaults to 84, i.e. 12 weeks.
`conf_band_lvl`	The confidence level for the simultaneous confidence bands. Defaults to 0.95. When detection of events using only the smoother is desired, `conf_band_lvl` can be chosen to be 0.
`bt_tot_rep`	The number of iterations for the bootstrap computation. Because of run time, it is recommended to keep this number below 500. Defaults to 100.
`time_unit`	A string containing the unit of time used, in singular form. Defaults to day.
`detect`	A string specifying how the detection bounds are to be chosen. `below`, `above`, and `custom` can be chosen. `below` detects decreases in value, `above` detects increases in value, and `custom` can be used to manually add detection bounds for each subject. When `above` or `below` are used, the detection bound will be x percent above or below the median of the first y days, where x is `detect_factor` and y is `detect period`.
`detect_factor`	A number specifying the factor by which the median of the fist `bline_period` days is to be multiplied to obtain the detection bounds. E.g. 0.9 sets the detection bound 10 percent below the said median.
`bline_period`	A number specifying the number of time units from which data should be taken to calculate the median to obtain the detection bounds.
`...`	Additional parameters to be given to the function. Possible parameters for the model are `order` and `min_pts_in_win`. For the moving median, a `med_win` is required. When resampling from window, a `resample_win` may be given. The parameter `min_pts_in_win` defines the minimal number of measurements required to be in the time window for the median to be calculated. Defaults to 1. If the parameter `order` is given, that number will be the (maximal) order of the autoregressive model. If no `order` is given, it will be determined using the Akaike information criterion. When the moving median is used as the smoother, `med_win` is expected. If no `med_win` is given, it will default to `c(-42, 42)`.

When resampling from window, one can choose the window size for the resampling window with `resample_win` by giving a window like e.g. `c(-14,14)`..

Details

For the moving median, the med_win is the total size of the window, meaning that for the value corresponding to day `x`, the data points from day `x` + `med_win[1]` to `x` + `med_win[2]` will be used for the calculation of the median.

If there is no data for two times `med_win[2]-med_win[1]` consecutive time units, there will be time points at which no confidence bound can be calculated. In this case, it will be assumed that the confidence bound is outside of the detection interval when detecting sustained change.

In case there are multiple instances where the algorithm would detect a sustained change (i.e. if after the first sustained change the confidence bounds leave the detection interval and then return into it for longer than `min_change_dur` time units) then only the first sustained change would be detected.

Please note that the event onset could be on a date where there are no actual measurements. This can happen when there is a gap in the data. In this case, the confidence bounds will extend into the gap. If the confidence bounds in this period are outside the detection interval and remain outside for the next `min_change_duration` time units, the event onset will be in this gap.

The censoring date is based on the last date where the confidence bounds can be calculated. We do not extend the confidence bounds to the last data point so that the confidence bounds don't change in case we obtain new measurements with time points later than the latest time point at which we have a measurement.

The `edecob` function runs the functions `mov_med`, `smoother_resid`, `bt_smoother`, `conf_band`, and `detect_event` in this order for all subjects given. If desired, the functions can also manually be applied for the data to obtain e.g. the confidence bands. Note that in order to run one of these functions, the output of the previous functions are needed.

Value

The output `data` is a list containing as many elements as the number of sources in `data` plus one. Every element in this list will again be a list named after the corresponding sources. Each of these lists contains the following elements:

`event` gives a list with four values: `event_detected`, `event_onset`, `event_duration`, and `event_stop`.

- `event_detected` gives whether an event was detected
- `event_onset` gives the first time point at which the upper or lower bound of the confidence band is inside the detection bounds, and after which it stays inside the detection bounds for at least `min_change_dur` consecutive time units
- `event_duration` gives the number of time units the upper or lower bound of the confidence band stays inside the detection bounds after `event_onset`
- `event_stop` gives whether the confidence bounds stay inside the detection bounds until the last time point at which we can calculate the confidence bound or not.

`conf_band` gives a data frame containing the confidence bands. The columns are source, time point, lower bound, and upper bound of the confidence band.

`smoother_pts` gives a data frame containing the smoother. The columns are source, time point, and the smoother `data` gives the data but with four additional columns: `event_detected`, `event_onset`, `event_duration`, and `event_stop`. They contain the same values as in `event`.

`detec_lower` gives the lower detection bound.

`detec_upper` gives the upper detection bound.

`smoother` gives the smoother used.

`resample_method` gives the resampling method used for the bootstrap.

`min_change_dur` gives the smallest consecutive number of time units the confidence bounds must stay within the detection bounds in order for an event to be detected.

`conf_band_lvl` gives the level of the simultaneous confidence band.

`bt_tot_rep` gives the total amount of bootstrap repetitions performed.

`call` gives the function call.

`col_names` gives the original column names of the data.

`time_unit` gives the unit of time used.

The last element in the output `data` is called `event_info` and is a data frame containing the information from `event` from each patient. `event_info` will thus have the following columns: `source`, `event_detected`, `event_onset`, `event_duration`, and `event_stop`.

Mathematical background

The mathematical background will be explained in the following sections.

Moving Median

Consider a sample $X_1, \dots, X_n$ of size n and the reordering $X_{(1)}, \dots, X_{(n)}$ such that $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$, commonly called the order statistic. Then for n even the median usually defined as

$$median(X_1, \dots, X_n) = X_{(k)}, \text{where } k = n/2.$$

In the case where n is odd the median is defined as

$$median(X_1, \dots, X_n) = 1/2(X_{(k)} + X_{(k+1)}), \text{where } k = n/2.$$

Let the study days at which the measurements $X_1, \dots, X_n$ were taken be $t_1, \dots, t_n$. Let T a fixed positive amount of time. Then the moving median at time point t with window size T is defined as

$$S(t) = median(X_j | t - T/2 \leq t_j \leq t + T/2).$$

The Model

An autoregressive (AR) model is used to model the residuals of the smoother η:

$$Y(t) = S(t) + \eta(t)$$

$$\eta(t) = \sum_{j=1}^{p} \phi_j \eta(t - j) + \epsilon$$

where variable t is the study day, $Y(t)$ the data point at study day t, $S(t)$ a smoother, $\eta(t)$ the difference between the smoother and the measurement at study day t, p the order of the AR model, ϕ_j the coefficients of the AR model, and ϵ the error of the AR model. The order is calculated using the Akaike information criterion (AIC) if it was not given in the function call.

Bootstrap

In the following, the star * denotes a bootstrapped value. The bootstrap procedure is as follows:

1. Compute the smoother $S(t)$.
2. Compute the residuals $\eta(t_i) = Y(t_i) - S(t_i)$.
3. Fit an AR(p) model to $\eta(t_i)$ to obtain the coefficients $\phi_1, \dots, \phi_p$ and $\epsilon(t_i) = \eta(t_i) - \sum_{j=1}^{p} \phi_j \eta(t_i - t_{i-j})$ the error of the AR model.
4. Randomly choose a $\epsilon(t_i)^*$ with replacement from $\epsilon(t_{p+1}), \dots, \epsilon(t_n)$ to obtain
$$Y(t_i)^* = S(t_i) + \eta(t_i)^*,$$
where
$$\eta(t_i)^* = \sum_{j=1}^{p} \phi_j \eta(t_{i-j})^* + \epsilon(t_{i-j})^*$$
the bootstrapped residuals of the smoother.
5. Compute $S(.)^* = g(Y(t_1), \dots, Y(t_n))$ where g is the function with which the smoother is calculated.
6. Repeat steps 4 and 5 `bt_tot_rep` times to obtain $S(t_i)^*_b$ for $\beta = 1, \dots,$ `bt_tot_rep`.

Calculation of the Confidence Bounds

The confidence bounds are calculated as follows:

1. We compute the quantiles
$$q_x(t_i), q_{1-x}(t_i) i = 1, \dots, N$$
where
$$q_x(t_i) = inf \, \{u; P^*[S(t_i)^*_b - S(t_i) \leq u] \geq x\}$$
is a pointwise bootstrap quantile, $S(t_i)^*_b$ the bootstrapped smoother, and N the number of measurements or rows in `data`, in our case the number of rows.
2. We vary the pointwise error x until
$$P^*[q_x(t_i) \leq S(t_i)^*_b - S(t_i) \leq q_{1-x}(t_i) \forall i = 1, \dots, N] \approx 1 - \alpha.$$
In other words, until the ratio of bootstrap curves that have all their points within $[q_x(t_i), q_{1-x}(t_i)]$ is approximately $1 - \alpha$.
3. We define
$$I_n(t_i) = [S(t_i) + q_x(t_i), S(t_i) + q_{1-x}(t_i)] \forall i = 1, \dots, N$$
the confidence bounds. Then $I_n(t_i); i = 1, \dots, N$ is a consistent simultaneous confidence band of level $1 - \alpha$.

References

Bühlmann, P. (1998). Sieve Bootstrap for Smoothing in Nonstationary Time Series. *The Annals of Statistics*, 26(1), 48-83.

Hogg, R., McKean, J. and Craig, A. (2014). *Introduction to mathematical statistics.* Harlow: Pearson Education.

See Also

summary.edecob, plot.edecob

Examples

```
library(edecob)

# Let us examine the example_data dataset
head(example_data, 3)
#>     subject study_day jump_height
#> 1 Subject 1         1    58.13024
#> 2 Subject 1         5    59.48988
#> 3 Subject 1         9    54.14774

# We apply the main fuction of the package onto our example_data
example_event <- edecob(example_data, med_win = c(-21,21), bt_tot_rep = 10,
                        min_change_dur = 70)
names(example_event)
#> [1] "Subject 1"  "Subject 2"  "event_info"

# example_event contains the event data for each source
plot(example_event$`Subject 1`)
plot(example_event$`Subject 2`)

# example_event also contains a data frame containing the event information for all patients
example_event$event_info
#>           event_detected event_onset event_duration event_stop
#> Subject 1           TRUE         119            134       TRUE
#> Subject 2          FALSE         306             60      FALSE

# Using this data frame, we can draw a survival plot
library("survival")
plot(survfit(Surv(time = event_onset, event = event_detected) ~ 1,
             data = example_event$event_info),
     conf.int = FALSE, xlim = c(0,350), ylim = c(0,1), mark.time = TRUE,
     xlab = "Time point", ylab = "Survival probability", main = "Survival plot")
```

`example_data` *Artificially generated data for examples*

Description

A dataset in the long format containing the jump height of 3 subjects over 300 days.

Usage

```
example_data
```

Format

A dataframe with 317 rows and 5 variables:

subject The subject identifier

time_point The point in time from which the data is collected, in days

jump_height The jump height, in cm

detect_lower The lower detection bound for the subject

detect_upper The upper detection bound for the subject

`mov_med` *Moving Median over a Time Window*

Description

Calculates the moving median over a time window around a time point for the all time points between the first and last time point provided.

Usage

```
mov_med(data, med_win = c(-42, 42), min_pts_in_win = 1)
```

Arguments

`data`	A data frame in long format containing the data for which events is to be detected. This means that each measurement corresponds to a row and the columns are (in order): source (the device or person from which the data was collected), point in time, and measurement value. If custom detection bounds are chosen, the folloing two columns must be added: lower detection bound, and upper detection bound. The source is expected to be a string; the time point are integers; measurements, and detection bounds are expected to be numerical. The detection bounds are in absolute value in the same unit as the values and each is expected to be identical for the same source. In case detection is wanted for a one sided change (e.g. give an event if the confidence bounds drop below a threshold) then the upper or lower detection bound can be chosen to be Inf or -Inf respectively.
`med_win`	A vector containing two numbers specifying the window over which the moving median is to be taken. More specifically, when given a certain time point, the numbers specify the maximum number of time units difference that a time point can have such that that data point will be considered for the moving median. Note that the first number must be smaller than the second number. The numbers may be negative.
`min_pts_in_win`	The minimal number of measurements required to be in the time window in order for the median to be calculated.

Details

Consider a sample $X_1, \dots, X_n$ of size n and the reordering $X_{(1)}, \dots, X_{(n)}$ such that $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$, commonly called the order statistic. Then for n even the median usually defined as

$$median(X_1, \dots, X_n) = X_{(k)}, \text{where } k = n/2.$$

In the case where n is odd the median is defined as

$$median(X_1, \dots, X_n) = 1/2(X_{(k)} + X_{(k+1)}), \text{where } k = n/2.$$

Let the study days at which the measurements $X_1, \dots, X_n$ were taken be $t_1, \dots, t_n$. Let T a fixed positive amount of time. Then the moving median at time point t with window size T is defined as

$$S(t) = median(X_j | t - T/2 \leq t_j \leq t + T/2).$$

For the initial time points where the time difference between the first data point and the time point for which we are calculating the median is less than half the `width`, we do not have enough data points to form a window which has the same size to both sides of the time point. In this case fewer data points are used to calculate the median and the time window is not symmetric around the time point for which we are calculating the median.

No median is calculated if the time difference between the last data point and the current time point for which we are calculating the median is less than half the `width`. We do not calculate the median using a smaller time window so that the values do not change upon receiving new data with time points newer than that of the old data.

Value

A data frame containing the values of the moving median, the study day to which it corresponds, the time window from which it was calculated, and the subject id corresponding to the data.

`plot.edecob` *Plot Event Data*

Description

Generates a ggplot2 visualizing the data and event if any detected.

Usage

```
## S3 method for class 'edecob'
plot(x, ...)
```

Arguments

`x`	The output of the `edecob` function for one subject. It is an object of class `edecob` containing the data and the event information.
`...`	Other arguments like `title`, `xlab`, or `ylab`. If the plot should be returned, write `output = TRUE`.

Details

The data points are plotted on the x-axis with the time point on the y-axis. The data from the learning period are gray. The baseline period is outlined by two vertical blue lines. The smoother is plotted in orange. The confidence bound is the blue area. If an event is detected, a red triangle will mark the detection day.

If the parameter `output = TRUE` is given to the function, the function will output the ggplot. The plot can then be manipulated using the usual functions from the ggplot2 package.

Value

None. If `output = TRUE` was written in function call, a `ggplot` object that visualizes the data will be returned. The returned plot will not contain the the text at the bottom.

smoother_resid *Residuals of the Smoother*

Description

Calculate the residuals of the smoother to the data points.

Usage

```
smoother_resid(data, smoother_pts)
```

Arguments

`data`	A data frame in long format containing the data for which events is to be detected. This means that each measurement corresponds to a row and the columns are (in order): source (the device or person from which the data was collected), point in time, and measurement value. If custom detection bounds are chosen, the folloing two columns must be added: lower detection bound, and upper detection bound. The source is expected to be a string; the time point are integers; measurements, and detection bounds are expected to be numerical. The detection bounds are in absolute value in the same unit as the values and each is expected to be identical for the same source. In case detection is wanted for a one sided change (e.g. give an event if the confidence bounds drop below a threshold) then the upper or lower detection bound can be chosen to be Inf or -Inf respectively.
`smoother_pts`	A data frame containing the smoother. Preferably the output of one of the smoother functions included in this package.

Value

A vector of the same length as `data` containing the residuals.

`summary.edecob` *Summarizing Event Detection Results*

Description

`summary` method for class "`edecob`", gives a summary of an edecob object

Usage

```
## S3 method for class 'edecob'
summary(object, ...)
```

Arguments

`object`	An object of class "`edecob`" for which the summary will be shown.
`...`	Other arguments

Value

A short summary whether an event was detected and the parameters of the `edecob` function call.

Index