Package 'ggchangepoint'

June 26, 2026

Type Package

Title Combines Changepoint Analysis with 'ggplot2'

Version 0.3.0

Description R provides fantastic tools for changepoint analysis, but plots generated by the tools do not have the 'ggplot2' style. This tool, however, combines 'changepoint', 'changepoint.np' and 'ecp' together, and uses 'ggplot2' to visualize changepoints. It provides a unified 'ggcpt' S3 result class, 'broom'-style tidy/glance/augment methods, 'autoplot()', composable geoms ('geom_changepoint()', 'geom_cpt_segment()', 'geom_cpt_ci()', 'stat_changepoint()'), a unified 'cpt_detect()' dispatcher with method introspection via 'cpt_methods()', wrappers for several optional engines (WBS, WBS2, NOT, MOSUM, FPOP, Isolate-Detect, TGUH), a method comparison module, accuracy metrics, data simulation, and canonical test signals.

License GPL (>= 3)

Encoding UTF-8

Imports changepoint, changepoint.np, dplyr, ecp, generics, ggplot2 (>= 3.4.0), lifecycle, Rdpack, tibble, utils

RdMacros Rdpack

RoxygenNote 7.3.2

Suggests rmarkdown, knitr, testthat (>= 3.0.0), wbs, breakfast, not, mosum, fpop, IDetect, future, future.apply

VignetteBuilder knitr

URL https://pursuitofdatascience.github.io/ggchangepoint/

BugReports https://github.com/PursuitOfDataScience/ggchangepoint/issues

NeedsCompilation no

Author Youzhi Yu [aut, cre]

Maintainer Youzhi Yu <yuyouzhi666@icloud.com>

Repository CRAN

Date/Publication 2026-06-26 15:30:08 UTC

Contents

annotate_segments *Annotate segments with alternating shading*

Description

Adds alternating shaded rectangles to highlight segments between changepoints.

Usage

```
annotate_segments(cp, n, fill = c("grey90", "white"), alpha = 0.5, ...)
```

Arguments

cp	Changepoint indices (including 0 and n).
n	Length of the series.
fill	Colors for alternating segments. Defaults to c("grey90", "white").
alpha	Alpha for fill. Defaults to 0.5.
...	Additional arguments passed to annotate.

Value

A list of ggplot annotations.

augment.ggcpt *Augment a ggcpt object*

Description

Returns the original data with added columns: seg_id, .fitted, .resid, and is_changepoint.

Usage

```
## S3 method for class 'ggcpt'
augment(x, ...)
```

Arguments

x	A ggcpt object.
...	Additional arguments (ignored).

Value

A tibble with the original data plus augment columns.

`autoplot.ggcpt` *Autoplot a ggcpt object*

Description

Renders a changepoint detection result as a ggplot. The raw series is drawn as a line (with optional points), changepoints are shown as vertical lines, and (optionally) fitted segment levels are overlaid.

Usage

```
## S3 method for class 'ggcpt'
autoplot(
  object,
  show_segments = FALSE,
  show_ci = FALSE,
  cptline_alpha = 1,
  cptline_color = "blue",
  cptline_type = "solid",
  cptline_linewidth = 0.5,
  show_points = NULL,
  show_line = TRUE,
  ...
)
```

Arguments

`object`	A ggcpt object.
`show_segments`	Logical. Whether to draw the fitted segment means. Defaults to `FALSE`.
`show_ci`	Logical. Whether to draw confidence intervals for changepoint locations (if available). Defaults to `FALSE`.
`cptline_alpha`	Alpha for changepoint lines. Defaults to `1`.
`cptline_color`	Color for changepoint lines. Defaults to `"blue"`.
`cptline_type`	Linetype for changepoint lines. Defaults to `"solid"`.
`cptline_linewidth`	Linewidth for changepoint lines. Defaults to `0.5`.
`show_points`	Logical. Whether to draw data points. Auto-off above 500 obs.
`show_line`	Logical. Whether to draw the line. Defaults to `TRUE`.
`...`	Additional arguments passed to `ggcptplot_internal`.

Value

A ggplot object.

cpt_detect *Unified changepoint detection dispatcher*

Description

Runs one or more changepoint detection methods on a sequence and returns a tidy ggcpt result object. This is the recommended entry point for most users.

Usage

```
cpt_detect(x, method = "pelt", change_in = "mean", penalty = "MBIC", ...)
```

Arguments

x	A numeric vector (the data series).
method	Detection method. See cpt_methods() for the complete table. Methods available in this release: "pelt", "binseg", "segneigh", "amoc", "np", "ecp", "fpop", "wbs", "wbs2", "not", "mosum", "idetect", "tguh". Methods that ship with optional (Suggests) engine packages will prompt you to install them if missing. Planned methods — "smuce", "hsmuce", "kcp", "cpm", "robust", "decafs", "sn", "inspect", "sbs", "bcp", "bocpd", "strucchange", "segmented" — are listed in cpt_methods() with their target release.
change_in	What to detect change in. One of "mean", "var", "meanvar", "slope", "distribution". Defaults to "mean".
penalty	Penalty type or value. Either a character string ("MBIC", "BIC", "AIC", "Hannan-Quinn") or a numeric penalty value. Defaults to "MBIC". Numeric penalties are honoured by the functional-pruning method (fpop); the changepoint-based methods expect one of the character options.
...	Additional arguments passed to the specific wrapper.

Value

A ggcpt object.

Examples

```
set.seed(2022)
x <- c(rnorm(100, 0, 1), rnorm(100, 10, 1))
result <- cpt_detect(x, method = "pelt", change_in = "mean")
result
ggplot2::autoplot(result)
```

cpt_methods *Introspect available changepoint detection methods*

Description

Returns a tibble describing every method the package knows about — those that are wired and those that are planned — along with their capabilities and installation status. Useful for discovering what can be run and what needs to be installed.

Usage

```
cpt_methods()
```

Value

A tibble with columns:

method Method name as passed to `cpt_detect()`.

change_in What types of change the method can detect.

engine The upstream R package that implements the method.

status `"available"` (wired in this release) or `"planned"` (future).

installed `TRUE` if the engine package is installed, `FALSE` if it is a `Suggests` engine that is missing, `NA` for planned methods.

target_release The release that plans to wire this method, or `NA` for currently available methods.

Examples

```
cpt_methods()
```

cpt_metrics *Changepoint accuracy metrics*

Description

Computes standard accuracy metrics comparing predicted changepoints to ground truth, including precision/recall/F1 with margin, covering metric, Hausdorff distance, (adjusted) Rand index, annotation error, and MAE/RMSE of matched locations.

Usage

```
cpt_metrics(pred, truth, n, margin = 5)
```

Arguments

pred	Predicted changepoint indices (integer vector).
truth	Ground truth changepoint indices (integer vector).
n	Length of the series.
margin	Tolerance margin for matching (default 5).

Value

A tibble with columns: n, n_pred, n_truth, precision, recall, f1, covering, hausdorff, rand_index, annotation_error, mae_matched, rmse_matched.

Examples

```
cpt_metrics(c(100, 200), c(100, 200), n = 300)
cpt_metrics(c(101, 205), c(100, 200), n = 300, margin = 5)
```

---

cpt_metrics_annotated *Multi-annotator evaluation*

Description

Computes averaged covering and F1 scores against multiple annotation sets, as used in the Turing Change Point Dataset benchmark.

Usage

```
cpt_metrics_annotated(pred, annotations, n, margin = 5)
```

Arguments

pred	Predicted changepoint indices.
annotations	A list of ground-truth annotation vectors.
n	Length of the series.
margin	Tolerance margin (default 5).

Value

A tibble with averaged metrics.

`cpt_penalty` *Construct changepoint penalties*

Description

Helper to construct standard penalty values for use with changepoint detection methods. Returns a numeric penalty value.

Usage

```
cpt_penalty(type, n = NULL, k = 1, value = NULL)
```

Arguments

`type`	Penalty type: `"None"`, `"BIC"` (or `"SIC"`), `"MBIC"`, `"AIC"`, `"Hannan-Quinn"`, `"sSIC"`, or `"Manual"`.
`n`	Series length. Required for BIC, MBIC, AIC, Hannan-Quinn, sSIC.
`k`	Number of parameters per changepoint (typically 2 for mean+variance, 1 for mean-only). Defaults to 1.
`value`	Numeric value for `Manual` type.

Value

A numeric penalty value.

Penalty semantics across engines

The same penalty name may be interpreted differently by different engines:

- **changepoint-based methods** (PELT, BinSeg, SegNeigh, AMOC): accept character penalties (`"MBIC"`, `"BIC"`, `"AIC"`, `"Hannan-Quinn"`, `"None"`) and pass them to the upstream **changepoint** package. These methods do *not* accept raw numeric penalty values.
- **Functional-pruning methods** (`fpop`): accept numeric penalties only. When a character penalty is supplied via `cpt_detect()`, it is resolved to a numeric value using `cpt_penalty()` before dispatch.
- **Search-based methods** (WBS, WBS2, NOT, MOSUM, IDetect, TGUH): use internal model-selection criteria (e.g., sSIC, threshold) and generally *ignore* the `penalty` argument. Specify thresholds via the wrapper's own arguments.
- `MBIC` in `cpt_penalty()` uses the Zhang-Siegmund (2007) formula $0.5(k+1)\log n + \log\binom{n}{k}$, which differs from the **changepoint** package's MBIC. Use the character `"MBIC"` with **changepoint**-based methods to get the engine's native MBIC.

Examples

```
cpt_penalty("BIC", n = 100)
cpt_penalty("AIC", n = 100)
cpt_penalty("Manual", value = 5)
```

`cpt_simulate` *Generate simulated changepoint data*

Description

Creates a synthetic time series with known changepoints for testing and benchmarking.

Usage

```
cpt_simulate(
  n,
  changepoints = integer(),
  change_in = c("mean", "var", "meanvar", "slope"),
  params = NULL,
  noise = c("gauss", "t", "ar1", "rw"),
  sd = 1,
  df = 3,
  rho = 0,
  seed = NULL
)

rcpt(...)
```

Arguments

`n`	Length of the series.
`changepoints`	Integer vector of changepoint locations (last index of each segment before the change).
`change_in`	What changes: `"mean"`, `"var"`, `"meanvar"`, or `"slope"`.
`params`	A list of parameters per segment. For `mean` changes, a vector of segment means. For `var` changes, a vector of segment sds. For `meanvar`, a list of lists with `mean` and `sd` per segment. For `slope`, a list with `intercept` and `slope` per segment.
`noise`	Noise type: `"gauss"` (Gaussian), `"t"` (Student-t), `"ar1"` (AR(1)), or `"rw"` (random walk).
`sd`	Noise standard deviation (for Gaussian and t). Defaults to 1.
`df`	Degrees of freedom for t-noise. Defaults to 3.
`rho`	AR(1) autocorrelation parameter. Defaults to 0.
`seed`	Optional seed for reproducibility.
`...`	Passed to `cpt_simulate`.

Value

A tibble with columns `index`, `value`, and `seg_id`. The true changepoints are stored in the `true_changepoints` attribute.

Examples

```
dat <- cpt_simulate(200, changepoints = c(100), change_in = "mean",
                    params = c(0, 10), seed = 2022)
attr(dat, "true_changepoints")
```

`cpt_wrapper` *Changepoint wrapper*

Description

This function wraps a number of `cpt` functions from the changepoint package and the `cpt.np()` function from the changepoint.np package. It is handy that users can use this function to get the same changepoint results as these functions output individually. Moreover, it returns a tibble that inherits the tidyverse style. Functions from the changepoint package do require data normality assumption by default, yet changepoint.np is a non-parametric way to detect changepoints and let data speak by itself. If user sets `change_in` as `np` (or `cpt_np`), a seed should be set before using the function for the sake of reproducibility. For more details on the changepoint and changepoint.np packages, please refer to their documentation.

Usage

```
cpt_wrapper(data, change_in = "mean_var", cp_method = "PELT", ...)
```

Arguments

`data`	A numeric vector.
`change_in`	Choice of `mean_var`, `mean`, `var`, and `np` (or `cpt_np` for backward compatibility). Each choice corresponds to `cpt.meanvar()`, `cpt.mean()`, `cpt.var()` and `cpt.np()` respectively. The default is `mean_var`.
`cp_method`	A wide range of choices (i.e., `AMOC`, `PELT`, `SegNeigh` or `BinSeg`). Please note when `change_in` is `np` or `cpt_np`, `PELT` is the only option.
`...`	Extra arguments for each `cpt` function mentioned in the `change_in` section.

Value

A tibble including which point(s) is/are the changepoint along with raw changepoint value corresponding to that changepoint. Changepoint locations follow the convention of the `changepoint` package: the last index of the left segment.

References

Killick R, Eckley I (2014). "changepoint: An R package for changepoint analysis." *Journal of statistical software*, **58**(3), 1–19.

Examples

```
set.seed(2022)
cpt_wrapper(c(rnorm(100,0,1),rnorm(100,0,10)))
cpt_wrapper(c(rnorm(100,0,1),rnorm(100,10,1)))
```

`ecp_wrapper` *ecp wrapper*

Description

The ecp package provides a non-parametric way to detect changepoints. Unlike the changepoint package, it does not assume raw data to have any formal distribution. This wrapper function wraps two functions from the ecp package, i.e., `e.divisive()` and `e.agglo()`. Users can use either function by switching the `algorithm` argument. Before using the wrapper function, seed should be set for the sake of reproducibility.

Usage

```
ecp_wrapper(data, algorithm = "divisive", min_size = 2, seed = NULL, ...)
```

Arguments

`data`	A numeric vector (for univariate) or matrix/data.frame (for multivariate).
`algorithm`	Either `divisive` or `agglo`. `divisive` is the default.
`min_size`	Minimum number of observations between change points. By default is 2. This argument is only applied when `algorithm = "divisive"`.
`seed`	Optional. A seed for reproducibility of the stochastic permutation test.
`...`	Extra arguments to pass on either from `e.divisive()` or `e.agglo()`.

Value

A tibble includes which point(s) is/are the changepoint along with raw changepoint value corresponding to that changepoint. Changepoint locations follow the `ecp` package convention: the first index of the right segment. When no changepoint is found, an empty tibble is returned (0 rows).

References

James NA, Matteson DS (2013). "ecp: An R package for nonparametric multiple change point analysis of multivariate data." *arXiv preprint arXiv:1309.3295*.

Examples

```
set.seed(2022)
ecp_wrapper(c(rnorm(100,0,1),rnorm(100,0,10)))
ecp_wrapper(c(rnorm(100,0,1),rnorm(100,10,1)))
```

fpop_wrapper *FPOP wrapper — Functional Pruning Optimal Partitioning*

Description

Wraps the fpop package for optimal changepoint detection via functional pruning.

Usage

```
fpop_wrapper(x, penalty = NULL, ...)
```

Arguments

x	A numeric vector.
penalty	Penalty value. Defaults to 2 * log(length(x)) (BIC).
...	Additional arguments passed to fpop::Fpop().

Value

A ggcpt object.

geom_changepoint *Changepoint vertical rules geom*

Description

Draws vertical lines at changepoint locations. Mimics geom_vline but designed to work with the tidy changepoint data frames returned by the package. Can be used as a standalone layer: geom_changepoint(data = cp_df, aes(xintercept = cp)).

Usage

```
geom_changepoint(
  mapping = NULL,
  data = NULL,
  ...,
  na.rm = FALSE,
  show.legend = NA
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::aes(). Requires xintercept.
data	A data frame with changepoint information.
...	Other arguments passed to geom_vline.
na.rm	If FALSE, missing values are removed.
show.legend	Whether to show legend.

Value

A ggplot layer.

`geom_cpt_ci` *Changepoint confidence interval geom*

Description

Draws horizontal whiskers for changepoint-location confidence intervals (e.g. from MOSUM, stepR, strucchange, segmented).

Usage

```
geom_cpt_ci(mapping = NULL, data = NULL, ..., na.rm = FALSE, show.legend = NA)
```

Arguments

`mapping`	Aesthetic mappings. Requires `x`, `xmin`, `xmax`, and `y`.
`data`	A data frame with CI information.
`...`	Other arguments passed to `geom_errorbarh`.
`na.rm`	If `FALSE`, missing values are removed.
`show.legend`	Whether to show legend.

Value

A ggplot layer.

`geom_cpt_segment` *Changepoint segment level geom*

Description

Draws horizontal segments representing the estimated level of each segment between changepoints. Typically used with data from `augment()`.

Usage

```
geom_cpt_segment(
  mapping = NULL,
  data = NULL,
  ...,
  na.rm = FALSE,
  show.legend = NA
)
```

Arguments

mapping	Aesthetic mappings. Requires x, xend, y, yend.
data	A data frame with segment information.
...	Other arguments passed to geom_segment.
na.rm	If FALSE, missing values are removed.
show.legend	Whether to show legend.

Value

A ggplot layer.

---

ggcptplot *Plot for the changepoint package*

Description

The plot for changepoints detected by the changepoint package is a line plot for the raw data and the vertical lines representing each changepoint. The x-axis is the row number of the raw data in the original data vector. The plot inherits ggplot2, meaning users can add ggplot2 functions on top the changepoint plot for customization.

Usage

```
ggcptplot(
  data,
  change_in = "mean_var",
  cp_method = "PELT",
  ...,
  cptline_alpha = 1,
  cptline_color = "blue",
  cptline_type = "solid",
  cptline_linewidth = 0.5,
  cptline_size = lifecycle::deprecated(),
  index = NULL,
  show_points = NULL,
  show_line = TRUE
)
```

Arguments

data	A numeric vector.
change_in	Choice of mean_var, mean, var, and np (or cpt_np for backward compatibility). Each choice corresponds to cpt.meanvar(), cpt.mean(), cpt.var() and cpt.np() respectively. The default is mean_var.

cp_method	A wide range of choices (i.e., AMOC, PELT, SegNeigh or BinSeg). Please note when change_in is np or cpt_np, PELT is the only option.
...	Extra arguments for each cpt function mentioned in the change_in section.
cptline_alpha	The value of alpha for the vertical changepoint line(s), default is 1, meaning no transparency.
cptline_color	The color for the vertical changepoint line(s), default is blue.
cptline_type	The linetype for the vertical changepoint line(s), default is solid.
cptline_linewidth	The linewidth for the vertical changepoint line(s), default is 0.5.
cptline_size	Deprecated. Use cptline_linewidth instead.
index	Optional. A vector of x-axis labels (e.g. dates) of the same length as data.
show_points	Logical. Whether to draw data points. Defaults to TRUE when length(data) <= 500, FALSE otherwise.
show_line	Logical. Whether to draw the line. Defaults to TRUE.

Value

A line plot with data points along with the vertical lines representing changepoints.

Examples

```
ggcptplot(c(rnorm(100,0,1),rnorm(100,0,10)))
ggcptplot(c(rnorm(100,0,1),rnorm(100,10,1)))
```

---

ggcpt_compare *Compare multiple changepoint detection methods*

Description

Runs several detectors on the same data and returns a faceted or overlaid ggplot comparison. Respects future::plan() for parallel execution if the future.apply package is available.

Usage

```
ggcpt_compare(
  x,
  methods = c("pelt", "binseg", "amoc"),
  layout = c("facet", "overlay"),
  change_in = "mean",
  seed = NULL,
  ...
)
```

Arguments

x	A numeric vector (the data series).
methods	Character vector of method names (passed to cpt_detect).
layout	Layout type. "facet" (default) shows one panel per method; "overlay" draws all changepoints in one panel, colour-coded.
change_in	What to detect change in. Passed to each detector.
seed	Optional seed for reproducible parallelism. Passed to future.apply::future_lapply() as future.seed.
...	Additional arguments passed to each detector.

Value

A ggplot object.

Examples

```
set.seed(2022)
x <- c(rnorm(100, 0, 1), rnorm(100, 10, 1))
ggcpt_compare(x, methods = c("pelt", "binseg"))
```

---

ggcpt_compare_table *Comparison table*

Description

Returns a tidy tibble combining the results of multiple detectors on the same series.

Usage

```
ggcpt_compare_table(
  x,
  methods = c("pelt", "binseg", "amoc"),
  change_in = "mean",
  ...
)
```

Arguments

x	A numeric vector (the data series).
methods	Character vector of method names.
change_in	What to detect change in.
...	Additional arguments passed to each detector.

Value

A tibble with columns method, cp, cp_value.

ggcpt_eval *Evaluation visualization*

Description

Overlays predictions and ground truth on the series with tolerance windows, colouring true positives, false positives, and misses.

Usage

```
ggcpt_eval(pred, truth, data_vec, margin = 5)
```

Arguments

pred	Predicted changepoint indices.
truth	Ground truth changepoint indices.
data_vec	The original data vector (for context).
margin	Tolerance margin (default 5).

Value

A ggplot object.

ggcpt_methods *Coerce, format, and plot ggcpt objects*

Description

Convenience S3 methods for working with ggcpt objects: coerce the changepoints to a tibble or data frame, render a one-line summary string, or produce the default plot (a base-graphics fallback that delegates to autoplot.ggcpt).

Usage

```
## S3 method for class 'ggcpt'
as_tibble(x, ..., .name_repair = NULL)

## S3 method for class 'ggcpt'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S3 method for class 'ggcpt'
format(x, ...)

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

Arguments

x	A ggcpt object.
...	Additional arguments passed to methods.
.name_repair	Passed to as_tibble.
row.names, optional	Passed to as.data.frame.

Value

as_tibble() and as.data.frame() return the changepoints table; format() returns a length-one character string; plot() returns a ggplot object.

Examples

```
set.seed(2022)
res <- cpt_detect(c(rnorm(50), rnorm(50, 5)), method = "pelt")
as_tibble(res)
as.data.frame(res)
format(res)
```

---

ggecpplot *Plot for the ecp package*

Description

The plot for changepoints detected by the ecp package is a line plot for the raw data and the vertical lines representing each changepoint. The x-axis is the row number of the raw data in the original data vector. The plot inherits ggplot2, meaning users can add ggplot2 functions on top the changepoint plot for customization.

Usage

```
ggecpplot(
  data,
  algorithm = "divisive",
  min_size = 2,
  ...,
  cptline_alpha = 1,
  cptline_color = "blue",
  cptline_type = "solid",
  cptline_linewidth = 0.5,
  cptline_size = lifecycle::deprecated(),
  index = NULL,
  show_points = NULL,
  show_line = TRUE
)
```

Arguments

data	A numeric vector (for univariate) or matrix/data.frame (for multivariate).
algorithm	Either divisive or agglo. divisive is the default.
min_size	Minimum number of observations between change points. By default is 2. This argument is only applied when algorithm = "divisive".
...	Extra arguments to pass on either from e.divisive() or e.agglo().
cptline_alpha	The value of alpha for the vertical changepoint line(s), default is 1, meaning no transparency.
cptline_color	The color for the vertical changepoint line(s), default is blue.
cptline_type	The linetype for the vertical changepoint line(s), default is solid.
cptline_linewidth	The linewidth for the vertical changepoint line(s), default is 0.5.
cptline_size	Deprecated. Use cptline_linewidth instead.
index	Optional. A vector of x-axis labels (e.g. dates) of the same length as data.
show_points	Logical. Whether to draw data points. Defaults to TRUE when length(data) <= 500, FALSE otherwise.
show_line	Logical. Whether to draw the line. Defaults to TRUE.

Value

A line plot with data points along with the vertical lines representing changepoints.

Examples

```
ggecpplot(c(rnorm(100,0,1),rnorm(100,0,10)))
ggecpplot(c(rnorm(100,0,1),rnorm(100,10,1)))
```

---

glance.ggcpt *Glance at a ggcpt object*

Description

Returns a one-row summary of a changepoint detection result.

Usage

```
## S3 method for class 'ggcpt'
glance(x, ...)
```

Arguments

x	A ggcpt object.
...	Additional arguments (ignored).

Value

A one-row tibble with columns: `n`, `n_changepoints`, `method`, `change_in`, `penalty_type`, `penalty_value`, `cp_convention`, `total_cost` (if available), `runtime` (NA).

---

idetect_wrapper *Isolate-Detect wrapper*

Description

Wraps the `IDetect` package. Requires the `IDetect` package.

Usage

```
idetect_wrapper(x, seed = NULL, ...)
```

Arguments

`x`	A numeric vector.
`seed`	Optional seed for reproducibility.
`...`	Additional arguments passed to `IDetect::ID()`.

Value

A `ggcpt` object.

---

is_ggcpt *Test if an object is a ggcpt object*

Description

Test if an object is a ggcpt object

Usage

```
is_ggcpt(x)
```

Arguments

`x`	An object to test.

Value

`TRUE` if `x` inherits from `ggcpt`.

mosum_wrapper *MOSUM wrapper — Moving Sum*

Description

Wraps the mosum package for moving-sum-based changepoint detection.

Usage

```
mosum_wrapper(x, G = NULL, multiscale = FALSE, seed = NULL, ...)
```

Arguments

x	A numeric vector.
G	Bandwidth. If NULL, automatically selected.
multiscale	Logical. Use multiscale MOSUM? Defaults to FALSE.
seed	Optional seed for reproducibility.
...	Additional arguments passed to mosum::mosum().

Value

A ggcpt object.

new_ggcpt *Create a ggcpt object*

Description

Create a ggcpt object

Usage

```
new_ggcpt(
  changepoints = tibble::tibble(cp = integer(), cp_value = numeric()),
  segments = tibble::tibble(seg_id = integer(), start = integer(), end = integer(), n =
    integer(), param_estimate = numeric()),
  data = tibble::tibble(index = integer(), value = numeric()),
  method = character(),
  change_in = character(),
  penalty = list(type = NA_character_, value = NA_real_),
  fit = NULL,
  call = NULL,
  cp_convention = "left",
  runtime = NA_real_
)
```

Arguments

changepoints	A tibble with columns cp and cp_value.
segments	A tibble with segment information: seg_id, start, end, n, param_estimate.
data	A tibble with index and value.
method	Character. The detection method used.
change_in	Character. What was detected (e.g. "mean", "var", "meanvar").
penalty	A list with type and value.
fit	The raw upstream object.
call	The matched call.
cp_convention	Character. The convention for reporting changepoint locations: "left" (last index of left segment, used by changepoint) or "right" (first index of right segment, used by ecp). Defaults to "left".
runtime	Numeric. Elapsed detection time in seconds, if measured. Defaults to NA.

Value

An object of class ggcpt.

---

not_wrapper *NOT wrapper — Narrowest-Over-Threshold*

Description

Wraps the not package for changepoint detection via the Narrowest-Over-Threshold method.

Usage

```
not_wrapper(x, contrast = "pcwsConstMean", seed = NULL, ...)
```

Arguments

x	A numeric vector.
contrast	Contrast type. One of "pcwsConstMean", "pcwsLinContMean", "pcwsLinMean", "pcwsConstMeanVar". Defaults to "pcwsConstMean".
seed	Optional seed for reproducibility.
...	Additional arguments passed to not::not().

Value

A ggcpt object.

print.ggcpt *Print a ggcpt object*

Description

Print a ggcpt object

Usage

```
## S3 method for class 'ggcpt'
print(x, ...)
```

Arguments

x	A ggcpt object.
...	Additional arguments (ignored).

signal_blocks *Blocks test signal*

Description

The classic Donoho-Johnstone blocks test signal with known changepoints.

Usage

```
signal_blocks(n = 2048, seed = NULL)
```

Arguments

n	Length of the signal. Defaults to 2048.
seed	Optional seed.

Value

A tibble with columns index and value. The true_changepoints attribute contains the known changepoint locations.

References

Donoho, D. L. and Johnstone, I. M. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3), 425-455.

signal_fms *FMS (Four-Metric-Segments) test signal*

Description

A piecewise-constant test signal from the WBS/NOT literature.

Usage

```
signal_fms(n = 2000, seed = NULL)
```

Arguments

n	Length of the signal. Defaults to 2000.
seed	Optional seed.

Value

A tibble with columns `index` and `value`.

signal_mix *Mix test signal*

Description

A piecewise-constant/linear signal from the literature.

Usage

```
signal_mix(n = 2000, seed = NULL)
```

Arguments

n	Length of the signal. Defaults to 2000.
seed	Optional seed.

Value

A tibble with columns `index` and `value`.

signal_stairs *Stairs test signal*

Description

A monotonically stepping signal (staircase).

Usage

```
signal_stairs(n = 2000, seed = NULL)
```

Arguments

n	Length of the signal. Defaults to 2000.
seed	Optional seed.

Value

A tibble with columns `index` and `value`.

signal_teeth *Teeth test signal*

Description

A piecewise-constant signal with regularly spaced changepoints.

Usage

```
signal_teeth(n = 2000, seed = NULL)
```

Arguments

n	Length of the signal. Defaults to 2000.
seed	Optional seed.

Value

A tibble with columns `index` and `value`.

`stat_changepoint` *Changepoint detection stat*

Description

Runs changepoint detection inside the ggplot pipeline. Useful for quick exploration: `ggplot(df, aes(t, y)) + geom_line() + stat_changepoint(method = "pelt")`. Draws vertical lines at detected changepoint locations.

Usage

```
stat_changepoint(
  mapping = NULL,
  data = NULL,
  geom = "vline",
  position = "identity",
  ...,
  method = "pelt",
  change_in = "mean",
  na.rm = FALSE,
  show.legend = NA
)
```

Arguments

`mapping`	Aesthetic mappings.
`data`	A data frame.
`geom`	The geometric object to use (default: "vline").
`position`	Position adjustment.
`...`	Other arguments passed to the geom.
`method`	Detection method (passed to `cpt_detect`).
`change_in`	What to detect change in (passed to `cpt_detect`).
`na.rm`	If `FALSE`, missing values are removed.
`show.legend`	Whether to show legend.

Value

A ggplot layer.

summary.ggcpt *Summary of a ggcpt object*

Description

Provides a human-readable digest of a changepoint detection result, including the segment table with levels and lengths, total cost, penalty, and runtime.

Usage

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

## S3 method for class 'summary.ggcpt'
print(x, ...)
```

Arguments

object	A ggcpt object.
...	Additional arguments (ignored).
x	A summary.ggcpt object (for print()).

Value

A list with class summary.ggcpt containing the summary.

tguh_wrapper *TGUH wrapper*

Description

Wraps the breakfast package for Tail-Greedy Unbalanced-Haar detection.

Usage

```
tguh_wrapper(x, ...)
```

Arguments

x	A numeric vector.
...	Additional arguments passed to breakfast::breakfast().

Value

A ggcpt object.

theme_ggcpt *ggchangepoint theme*

Description

A minimal, publication-ready ggplot2 theme for changepoint plots.

Usage

```
theme_ggcpt(base_size = 11, base_family = "")
```

Arguments

base_size	Base font size. Defaults to 11.
base_family	Base font family. Defaults to "".

Value

A ggplot2 theme object.

Examples

```
library(ggplot2)
ggplot(mtcars, aes(wt, mpg)) + geom_point() + theme_ggcpt()
```

tidy.ggcpt *Tidy a ggcpt object*

Description

Returns the changepoints tibble (one row per changepoint).

Usage

```
## S3 method for class 'ggcpt'
tidy(x, ...)
```

Arguments

x	A ggcpt object.
...	Additional arguments (ignored).

Value

A tibble with columns cp, cp_value, and any method-specific columns.

wbs2_wrapper *WBS2 wrapper — Wild Binary Segmentation 2*

Description

Wraps the `breakfast` package. Requires the `breakfast` package.

Usage

```
wbs2_wrapper(x, ...)
```

Arguments

`x`	A numeric vector.
`...`	Additional arguments passed to `breakfast::breakfast()`.

Value

A `ggcpt` object.

wbs_wrapper *WBS wrapper — Wild Binary Segmentation*

Description

Wraps the `wbs` package for randomised changepoint detection via Wild Binary Segmentation.

Usage

```
wbs_wrapper(x, n_intervals = 5000, threshold = NULL, seed = NULL, ...)
```

Arguments

`x`	A numeric vector.
`n_intervals`	Number of random intervals. Defaults to `5000`.
`threshold`	Manual threshold for detection. If `NULL`, uses the strengthened Schwarz Information Criterion (sSIC).
`seed`	Optional seed for reproducibility.
`...`	Additional arguments passed to `wbs::wbs()`.

Value

A `ggcpt` object.

Index