

Package ‘scanr’

August 21, 2026

Type Package

Title Sequential Change-Point Detection via Nonparametric Inference

Version 0.1.1

Description Detects change points in long univariate time series using the SCAN framework. The implementation uses a native Rust backend exposed to R via 'extendr'.

License GPL-3

Language en-US

Encoding UTF-8

RoxygenNote 8.0.0

SystemRequirements Cargo (Rust's package manager), rustc, Quarto CLI

Imports ggplot2, jsonlite, rlang

Suggests knitr, quarto, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder quarto

URL <https://github.com/Prabashoka/scanr>,
<https://prabashoka.github.io/scanr-vignette/>

BugReports <https://github.com/Prabashoka/scanr/issues>

Config/rextendr/version 0.4.2

NeedsCompilation yes

Config/roxygen2/version 8.0.0

Author Ashoka Prabashwara [aut, cre],
Patricia Menéndez [aut],
Liam Hodgkinson [aut],
Stuart Lee [aut]

Maintainer Ashoka Prabashwara <smashoka123@gmail.com>

Repository CRAN

Date/Publication 2026-08-21 08:10:02 UTC

Contents

covering_metric	2
cpd_metrics	3
default_window_sizes	3
f1_score_cpd	4
ipm_statistic	5
match_change_points	5
one_wasserstein_distance	6
precision_recall_cpd	6
scan_cpd	7
scan_single_window	8
swal_statistic	9
ts_cusum	10
ts_wasserstein	10
vis_change_points	11
vis_swal_curve	12
vis_thresholds	12
vis_time_series	13
vis_vote_scree	14
vis_window_votes	15
Index	16

covering_metric	<i>Segment covering metric</i>
-----------------	--------------------------------

Description

Computes a weighted segment-covering score in $[0, 1]$.

Usage

covering_metric(true_cps, estimated_cps, n)

Arguments

- true_cps Integer vector of true change points.
- estimated_cps Integer vector of estimated change points.
- n Number of observations in the series.

Value

Numeric covering score.

cpd_metrics	<i>Combined change-point accuracy metrics</i>
-------------	---

Description

Combined change-point accuracy metrics

Usage

```
cpd_metrics(true_cps, estimated_cps, n, tolerance = 10L)
```

Arguments

true_cps	Integer vector of true change points.
estimated_cps	Integer vector of estimated change points.
n	Number of observations in the series.
tolerance	Maximum absolute distance allowed for a tolerant match.

Value

A list with matches, precision, recall, F1, and covering score.

Examples

```
cpd_metrics(  
  true_cps = c(25L, 50L),  
  estimated_cps = c(24L, 52L),  
  n = 75L,  
  tolerance = 3L  
)
```

default_window_sizes	<i>Choose default scan window sizes</i>
----------------------	---

Description

Randomly samples window sizes from a discrete uniform distribution for a series of length n . The upper bound defaults to $\text{floor}(n^{2/3})$ and is always capped at $\text{floor}(n / 2)$ so that a complete left and right window can fit around a candidate split.

Usage

```
default_window_sizes(
  n,
  min_window = 15L,
  max_window = NULL,
  n_windows = 5L,
  seed = NULL
)
```

Arguments

<code>n</code>	Number of observations in the series.
<code>min_window</code>	Smallest window size in the grid.
<code>max_window</code>	Optional largest window size. If NULL, uses $\text{floor}(n^{(2/3)})$.
<code>n_windows</code>	Number of distinct window sizes to sample uniformly between the lower and upper bounds. If more sizes are requested than available integers, all available sizes are returned.
<code>seed</code>	Optional non-negative integer seed for reproducible sampling. If NULL, the current R random-number generator state is used.

Value

A sorted integer vector of sampled window sizes.

Examples

```
default_window_sizes(n = 1000L, min_window = 15L)
default_window_sizes(
  n = 1000L,
  min_window = 20L,
  max_window = 100L,
  n_windows = 9L,
  seed = 123L
)
```

f1_score_cpd

Tolerant F1 score for change-point detection

Description

Tolerant F1 score for change-point detection

Usage

```
f1_score_cpd(true_cps, estimated_cps, tolerance = 10L)
```

Arguments

true_cps	Integer vector of true change points.
estimated_cps	Integer vector of estimated change points.
tolerance	Maximum absolute distance allowed for a match.

Value

Numeric F1 score.

ipm_statistic	<i>Integral probability metric statistic</i>
---------------	--

Description

Integral probability metric statistic

Usage

```
ipm_statistic(left, right)
```

Arguments

left	Numeric vector.
right	Numeric vector.

Value

Numeric distance.

match_change_points	<i>Match true and estimated change points</i>
---------------------	---

Description

Greedly matches true and estimated change points by smallest distance, allowing each point to be used at most once.

Usage

```
match_change_points(true_cps, estimated_cps, tolerance = 10L)
```

Arguments

true_cps	Integer vector of true change points.
estimated_cps	Integer vector of estimated change points.
tolerance	Maximum absolute distance allowed for a match.

Value

A data frame with columns true, estimated, and distance.

one_wasserstein_distance

One-dimensional Wasserstein distance

Description

One-dimensional Wasserstein distance

Usage

```
one_wasserstein_distance(left, right)
```

Arguments

left Numeric vector.

right Numeric vector.

Value

Numeric distance.

Examples

```
one_wasserstein_distance(c(0, 1, 2), c(1, 2, 3))
```

precision_recall_cpd *Tolerant precision and recall for change-point detection*

Description

Tolerant precision and recall for change-point detection

Usage

```
precision_recall_cpd(true_cps, estimated_cps, tolerance = 10L)
```

Arguments

true_cps Integer vector of true change points.

estimated_cps Integer vector of estimated change points.

tolerance Maximum absolute distance allowed for a match.

Value

Named numeric vector with precision and recall.

scan_cpd

*Detect change points in a univariate time series***Description**

Detect change points in a univariate time series

Usage

```
scan_cpd(
  x,
  window_sizes = NULL,
  alpha = 0.05,
  n_boot = 400L,
  vote_threshold = 0.5,
  min_window = 15L,
  max_window = NULL,
  n_windows = 5L,
  block_length = NULL,
  taper = c("tukey", "none"),
  tolerance = NULL,
  random_state = NULL,
  n_jobs = NULL,
  return_all = TRUE,
  change_type = c("distribution", "mean", "var"),
  eps = 1e-12,
  batch_size = 32L
)
```

Arguments

x	Numeric vector.
window_sizes	Optional positive integer vector of scan window sizes.
alpha	Significance level, either as a proportion such as 0.05 or a percentage such as 5.
n_boot	Number of tapered block bootstrap replications.
vote_threshold	Minimum normalized ensemble vote score for retained change points.
min_window	Minimum default window size when window_sizes is NULL.
max_window	Maximum default window size when window_sizes is NULL.
n_windows	Number of evenly spaced default window sizes when window_sizes is NULL.
block_length	Optional tapered block bootstrap block length.
taper	Taper shape, either "tukey" or "none".
tolerance	Distance used to merge nearby candidates across windows.
random_state	Optional non-negative integer seed.

n_jobs	Optional number of Rust/Rayon worker threads. Defaults to 1. Use -1 for all detected cores.
return_all	Whether to keep per-window diagnostics and raw output.
change_type	One of "distribution", "mean", or "var".
eps	Small positive value used to avoid division by zero.
batch_size	Bootstrap batch size used by the Rust backend.

Value

An object of class `scanr_result`.

Examples

```
set.seed(123)
x <- c(rnorm(30), rnorm(30, mean = 3))
fit <- scan_cpd(
  x,
  window_sizes = 10L,
  n_boot = 10L,
  random_state = 123L,
  change_type = "mean"
)
fit
```

scan_single_window	<i>Run SCAN for one window size</i>
--------------------	-------------------------------------

Description

Run SCAN for one window size

Usage

```
scan_single_window(
  x,
  window_size,
  alpha = 0.05,
  n_boot = 400L,
  block_length = NULL,
  taper = c("tukey", "none"),
  random_state = NULL,
  change_type = c("distribution", "mean", "var"),
  eps = 1e-12,
  batch_size = 32L
)
```

Arguments

x	Numeric vector.
window_size	Positive integer scan window size.
alpha	Significance level, either as a proportion such as 0.05 or a percentage such as 5.
n_boot	Number of tapered block bootstrap replications.
block_length	Optional tapered block bootstrap block length.
taper	Taper shape, either "tukey" or "none".
random_state	Optional non-negative integer seed.
change_type	One of "distribution", "mean", or "var".
eps	Small positive value used to avoid division by zero.
batch_size	Bootstrap batch size used by the Rust backend.

Value

An object of class `scanr_window_result`.

Examples

```
set.seed(123)
x <- c(rnorm(20), rnorm(20, mean = 3))
scan_single_window(
  x,
  window_size = 8L,
  n_boot = 10L,
  random_state = 123L,
  change_type = "mean"
)
```

swal_statistic	<i>Local SCAN/Wasserstein split statistic</i>
----------------	---

Description

Local SCAN/Wasserstein split statistic

Usage

```
swal_statistic(x, change_type = c("distribution", "mean", "var"))
```

Arguments

x	Numeric vector.
change_type	One of "distribution", "mean", or "var".

Value

Integer split position.

ts_cusum	<i>Localize a mean change with a CUSUM statistic</i>
----------	--

Description

Localize a mean change with a CUSUM statistic

Usage

```
ts_cusum(x)
```

Arguments

x Numeric vector.

Value

Integer split position.

Examples

```
ts_cusum(c(rep(0, 5), rep(4, 5)))
```

ts_wasserstein	<i>Localize a distributional change with a Wasserstein statistic</i>
----------------	--

Description

Localize a distributional change with a Wasserstein statistic

Usage

```
ts_wasserstein(x)
```

Arguments

x Numeric vector.

Value

A list with change_point and statistics.

Examples

```
result <- ts_wasserstein(c(rep(0, 5), rep(4, 5)))  
result$change_point
```

vis_change_points	<i>Visualize detected change points from a scan result</i>
-------------------	--

Description

Visualize detected change points from a scan result

Usage

```
vis_change_points(  
  x,  
  result,  
  true_change_points = NULL,  
  index = NULL,  
  x_label = "Time",  
  y_label = "Series",  
  title = NULL,  
  ...  
)
```

Arguments

x	Numeric vector.
result	A scanr_result object.
true_change_points	Optional true change points.
index	Optional x-axis index with the same length as x.
x_label	X-axis label.
y_label	Y-axis label.
title	Plot title.
...	Reserved for future plot options.

Value

A ggplot object.

vis_swal_curve	<i>Visualize the SWAL/Wasserstein localization curve for a region</i>
----------------	---

Description

Visualize the SWAL/Wasserstein localization curve for a region

Usage

```
vis_swal_curve(
  x,
  start,
  end,
  x_label = "Time series",
  y_label = "Scaled Wasserstein statistic",
  title = NULL,
  ...
)
```

Arguments

x	Numeric vector.
start	First observation in the region, using R's one-based indexing.
end	Last observation in the region, inclusive.
x_label	X-axis label.
y_label	Y-axis label.
title	Plot title.
...	Reserved for future plot options.

Value

A ggplot object.

vis_thresholds	<i>Visualize scan statistics and bootstrap thresholds</i>
----------------	---

Description

Visualize scan statistics and bootstrap thresholds

Usage

```
vis_thresholds(
  result,
  window_size = NULL,
  x_label = "Window start",
  y_label = "Statistic",
  title = NULL,
  ...
)
```

Arguments

result	A scanr_result object.
window_size	Optional window size to plot. Defaults to the first available window.
x_label	X-axis label.
y_label	Y-axis label.
title	Plot title.
...	Reserved for future plot options.

Value

A ggplot object.

vis_time_series	<i>Visualize a time series with optional change-point markers</i>
-----------------	---

Description

Visualize a time series with optional change-point markers

Usage

```
vis_time_series(
  x,
  change_points = NULL,
  true_change_points = NULL,
  index = NULL,
  x_label = "Time",
  y_label = "Value",
  title = NULL,
  ...
)
```

Arguments

<code>x</code>	Numeric vector.
<code>change_points</code>	Optional detected change points.
<code>true_change_points</code>	Optional true change points.
<code>index</code>	Optional x-axis index with the same length as <code>x</code> .
<code>x_label</code>	X-axis label.
<code>y_label</code>	Y-axis label.
<code>title</code>	Plot title.
<code>...</code>	Reserved for future plot options.

Value

A ggplot object.

Examples

```
x <- c(1, 2, 3, 8, 9, 10)
vis_time_series(x, change_points = 3L)
```

<code>vis_vote_scree</code>	<i>Visualize retained change-point count by voting threshold</i>
-----------------------------	--

Description

Visualize retained change-point count by voting threshold

Usage

```
vis_vote_scree(
  result,
  x_label = "Voting threshold",
  y_label = "Number of retained change points",
  title = NULL,
  ...
)
```

Arguments

<code>result</code>	A <code>scanr_result</code> object.
<code>x_label</code>	X-axis label.
<code>y_label</code>	Y-axis label.
<code>title</code>	Plot title.
<code>...</code>	Reserved for future plot options.

Value

A ggplot object.

vis_window_votes	<i>Visualize ensemble vote counts for candidate change points</i>
------------------	---

Description

Visualize ensemble vote counts for candidate change points

Usage

```
vis_window_votes(  
  result,  
  x_label_angle = 45,  
  x_label = "Candidate change point",  
  y_label = "Window votes",  
  title = NULL,  
  ...  
)
```

Arguments

- | | |
|---------------|-----------------------------------|
| result | A scanr_result object. |
| x_label_angle | Rotation angle for x-axis labels. |
| x_label | X-axis label. |
| y_label | Y-axis label. |
| title | Plot title. |
| ... | Reserved for future plot options. |

Value

A ggplot object.

Index

covering_metric, [2](#)
cpd_metrics, [3](#)

default_window_sizes, [3](#)

f1_score_cpd, [4](#)

ipm_statistic, [5](#)

match_change_points, [5](#)

one_wasserstein_distance, [6](#)

precision_recall_cpd, [6](#)

scan_cpd, [7](#)
scan_single_window, [8](#)
swal_statistic, [9](#)

ts_cusum, [10](#)
ts_wasserstein, [10](#)

vis_change_points, [11](#)
vis_swal_curve, [12](#)
vis_thresholds, [12](#)
vis_time_series, [13](#)
vis_vote_scree, [14](#)
vis_window_votes, [15](#)