Package: rfastlowess (via r-universe)

May 13, 2026

Title High-Performance LOWESS Smoothing for R

Version 1.3.0

Description Provides high-performance LOWESS smoothing (Locally Weighted Scatterplot Smoothing) using a 'Rust' backend. Supports various weight functions, robustness iterations, streaming/online processing, cross-validation, and uncertainty quantification. Applicable to genomic data, time series, and general-purpose smoothing tasks.

License MIT + file LICENSE | Apache License (== 2.0)

SystemRequirements 'Rust' programming language (via 'rustup' or system tools)

Encoding UTF-8

Roxygen list(markdown = TRUE, roclets = c(``namespace'', ``rd'', ``srr::srr_stats_roclet''))

biocViews Preprocessing, StatisticalMethod, Regression, FunctionalPrediction, QualityControl, KEGG

Imports BiocGenerics

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

BugReports https://github.com/thisisamirv/lowess-project/issues

URL https://github.com/thisisamirv/lowess-project, https://lowess.readthedocs.io/

Config/rextendr/version 0.5.0

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libclang-dev

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

Date/Publication 2026-05-13 07:23:25 UTC

RemoteUrl https://github.com/thisisamirv/lowess-project

RemoteRef HEAD

RemoteSha 96219cc9a9049fdfb537baf6408fea994dec8129

RemoteSubdir bindings/r

Contents

`rfastlowess-package` *rfastlowess: High-performance LOWESS Smoothing for R*

Description

A high-performance LOWESS (Locally Weighted Scatterplot Smoothing) implementation built on the Rust `fastLowess` crate.

Main Classes

- `Lowess`: Primary interface for batch processing
- `StreamingLowess`: Chunked processing for large datasets
- `OnlineLowess`: Sliding window for real-time data

Documentation

For comprehensive documentation, tutorials, and API reference, see: https://lowess.readthedocs.io/

Author(s)

Maintainer: Amir Valizadeh <thisisamirv@gmail.com> (ORCID) [funder]

Authors:

- Amir Valizadeh <thisisamirv@gmail.com> (ORCID) [funder]

See Also

Useful links:

- https://github.com/thisisamirv/lowess-project
- https://lowess.readthedocs.io/
- Report bugs at https://github.com/thisisamirv/lowess-project/issues

Examples

```
# Basic smoothing
x <- seq(1, 10, length.out = 100)
y <- sin(x) + rnorm(100, sd = 0.2)
model <- Lowess(fraction = 0.3)
result <- model$fit(x, y)
plot(x, y)
lines(result$x, result$y, col = "red", lwd = 2)
```

Lowess *LOWESS Batch Smoothing*

Description

Create a stateful LOWESS model for batch smoothing.

Usage

```
Lowess(
    fraction = 0.67,
    iterations = 3L,
    delta = NULL,
    weight_function = "tricube",
    robustness_method = "bisquare",
    scaling_method = "mad",
    boundary_policy = "extend",
    confidence_intervals = NULL,
    prediction_intervals = NULL,
    return_diagnostics = FALSE,
    return_residuals = FALSE,
    return_robustness_weights = FALSE,
    zero_weight_fallback = "use_local_mean",
    auto_converge = NULL,
    cv_fractions = NULL,
    cv_method = "kfold",
    cv_k = 5L,
    parallel = TRUE
)
```

Arguments

`fraction`	Smoothing fraction (0 to 1). Default: 0.67.
`iterations`	Robustness iterations. Default: 3.
`delta`	Interpolation threshold. NULL = auto.
`weight_function`	Kernel name. Default: "tricube".
`robustness_method`	Method: "bisquare", "huber", "talwar".
`scaling_method`	Scale estimation: "mad", "mar".
`boundary_policy`	Edge handling: "extend", "reflect", "zero", "noboundary".
`confidence_intervals`	Confidence level (e.g., 0.95). NULL disables.
`prediction_intervals`	Prediction level (e.g., 0.95). NULL disables.
`return_diagnostics`	Return fit metrics. Default: FALSE.
`return_residuals`	Return residuals. Default: FALSE.
`return_robustness_weights`	Return weights. Default: FALSE.
`zero_weight_fallback`	Fallback: "use_local_mean", "return_original", "return_none".
`auto_converge`	Convergence tolerance. NULL disables.
`cv_fractions`	Fractions for cross-validation. NULL disables.
`cv_method`	CV method: "kfold", "loocv".
`cv_k`	Folds for k-fold CV. Default: 5.
`parallel`	Enable parallel processing. Default: TRUE.

Value

A Lowess object.

Examples

```
x <- seq(0, 10, length.out = 100)
y <- sin(x) + rnorm(100, 0, 0.1)
model <- Lowess(fraction = 0.2)
result <- model$fit(x, y)
plot(x, y)
lines(x, result$y, col = "red")
```

Nullable *Nullable Value Wrapper*

Description

Wraps a value to be passed to Rust as an Option.

Usage

```
Nullable(x)
```

Arguments

x	Value to wrap or NULL.

Value

The value itself. This is a helper for rextendr conversion.

Examples

```
Nullable(5)
Nullable(NULL)
```

OnlineLowess *LOWESS Online Smoothing*

Description

Create a stateful LOWESS model for real-time online data.

Usage

```
OnlineLowess(
    fraction = 0.2,
    window_capacity = 100L,
    min_points = 2L,
    iterations = 3L,
    delta = NULL,
    weight_function = "tricube",
    robustness_method = "bisquare",
    scaling_method = "mad",
    boundary_policy = "extend",
    update_mode = "incremental",
    auto_converge = NULL,
    return_robustness_weights = FALSE,
    parallel = FALSE
)
```

Arguments

`fraction`	Smoothing fraction (0 to 1). Default: 0.67.
`window_capacity`	Max points in sliding window.
`min_points`	Minimum points before smoothing.
`iterations`	Robustness iterations. Default: 3.
`delta`	Interpolation threshold. NULL = auto.
`weight_function`	Kernel name. Default: "tricube".
`robustness_method`	Method: "bisquare", "huber", "talwar".
`scaling_method`	Scale estimation: "mad", "mar".
`boundary_policy`	Edge handling: "extend", "reflect", "zero", "noboundary".
`update_mode`	Update strategy: "incremental".
`auto_converge`	Convergence tolerance. NULL disables.
`return_robustness_weights`	Return weights. Default: FALSE.
`parallel`	Enable parallel processing. Default: TRUE.

Value

An OnlineLowess object.

Examples

```
model <- OnlineLowess(fraction = 0.2, window_capacity = 20)
x <- 1:50
y <- sin(x * 0.1) + rnorm(50, 0, 0.1)
result <- model$add_points(x, y)
plot(x, y)
lines(x, result$y, col = "red")
```

`plot.LowessResult` *Plot Lowess Result*

Description

Plot Lowess Result

Usage

```
## S3 method for class 'LowessResult'
plot(x, main = "LOWESS Fit", ...)
```

Arguments

x	A LowessResult object.
main	Plot title.
...	Additional arguments passed to plot() and lines().

Value

NULL, invisibly. Called for side effects (plotting).

Examples

```
x <- seq(0, 10, length.out = 100)
y <- sin(x) + rnorm(100, 0, 0.1)
model <- Lowess(fraction = 0.2)
res <- model$fit(x, y)
plot(res)
```

---

print.Lowess *Print Lowess Model*

Description

Print Lowess Model

Usage

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

Arguments

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

Value

The input object x, invisibly.

Examples

```
model <- Lowess(fraction = 0.3)
print(model)
```

print.LowessResult *Print Lowess Result*

Description

Print Lowess Result

Usage

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

Arguments

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

Value

The input object x, invisibly.

Examples

```
x <- seq(0, 10, length.out = 50)
y <- sin(x) + rnorm(50, 0, 0.1)
model <- Lowess(fraction = 0.3)
result <- model$fit(x, y)
print(result)
```

print.OnlineLowess *Print OnlineLowess Model*

Description

Print OnlineLowess Model

Usage

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

Arguments

x	An OnlineLowess object.
...	Additional arguments.

Value

The input object x, invisibly.

Examples

```
model <- OnlineLowess(fraction = 0.2, window_capacity = 20L)
print(model)
```

---

print.StreamingLowess *Print StreamingLowess Model*

Description

Print StreamingLowess Model

Usage

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

Arguments

x	A StreamingLowess object.
...	Additional arguments.

Value

The input object x, invisibly.

Examples

```
model <- StreamingLowess(fraction = 0.3, chunk_size = 50L)
print(model)
```

`StreamingLowess` *LOWESS Streaming Smoothing*

Description

Create a stateful LOWESS model for streaming data.

Usage

```
StreamingLowess(
    fraction = 0.3,
    chunk_size = 5000L,
    overlap = NULL,
    iterations = 3L,
    delta = NULL,
    weight_function = "tricube",
    robustness_method = "bisquare",
    scaling_method = "mad",
    boundary_policy = "extend",
    auto_converge = NULL,
    return_diagnostics = FALSE,
    return_robustness_weights = FALSE,
    parallel = TRUE
)
```

Arguments

`fraction`	Smoothing fraction (0 to 1). Default: 0.67.
`chunk_size`	Points per chunk.
`overlap`	Overlap between chunks.
`iterations`	Robustness iterations. Default: 3.
`delta`	Interpolation threshold. NULL = auto.
`weight_function`	Kernel name. Default: "tricube".
`robustness_method`	Method: "bisquare", "huber", "talwar".
`scaling_method`	Scale estimation: "mad", "mar".
`boundary_policy`	Edge handling: "extend", "reflect", "zero", "noboundary".
`auto_converge`	Convergence tolerance. NULL disables.
`return_diagnostics`	Return fit metrics. Default: FALSE.
`return_robustness_weights`	Return weights. Default: FALSE.
`parallel`	Enable parallel processing. Default: TRUE.

Value

A StreamingLowess object.

Examples

```
x <- seq(0, 10, length.out = 100)
y <- sin(x) + rnorm(100, 0, 0.1)
model <- StreamingLowess(fraction = 0.2, chunk_size = 50)
res1 <- model$process_chunk(x[1:50], y[1:50])
res2 <- model$process_chunk(x[51:100], y[51:100])
final <- model$finalize()
```

Index