Package 'rLifting'

June 19, 2026

Title High-Performance Wavelet Lifting Transforms

Version 1.0.0

Date 2026-06-11

Description Performs Wavelet Lifting Transforms focusing on signal denoising and functional data analysis (FDA). Implements a hybrid architecture with a zero-allocation 'C++' core for high-performance processing. Features include unified offline (batch) denoising, causal (real-time) filtering using a ring buffer engine, and adaptive recursive thresholding.

License MIT + file LICENSE

URL https://github.com/mkyou/rLifting

BugReports https://github.com/mkyou/rLifting/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

VignetteBuilder knitr

LinkingTo Rcpp

Imports Rcpp

Suggests testthat (>= 3.0.0), knitr, rmarkdown, dplyr, tidyr, ggplot2, microbenchmark, wavethresh

Config/testthat/edition 3

Depends R (>= 4.1.0)

NeedsCompilation yes

Author Moises da Silva [aut, cre]

Maintainer Moises da Silva <moisesdff8@gmail.com>

Repository CRAN

Date/Publication 2026-06-19 17:40:02 UTC

Contents

benchmark_adlift *adlift Benchmark Results*

Description

MSE and timing statistics for the **adlift** package (adaptive lifting on irregular grids) on the four Donoho-Johnstone signals. 7-quartile summaries over 1000 simulations per configuration. Used as a reference baseline in the irregular-grid benchmark vignette.

Usage

```
data(benchmark_adlift)
```

Format

A data frame with 384 rows and 19 columns:

Signal Donoho-Johnstone test signal.

Pkg Always `"adlift"`.

Wavelet Predictor family: `"AdaptPred"`, `"CubicPred"`, `"LinearPred"`, or `"QuadPred"`.

Boundary Encoded combination of `adlift::fwtnp` options (neighbours, interpolation, closest-point, mean/median predictor).

N Number of simulations per configuration (1000).

MSE_min, MSE_q1, MSE_median, MSE_mean, MSE_q3, MSE_max, MSE_se Reconstruction MSE statistics.

Time_min, Time_q1, Time_median, Time_mean, Time_q3, Time_max, Time_se Wall-time statistics (seconds).

Source

`data-raw/generate_adlift_benchmark.R`

benchmark_adlift_irregular *adlift Irregular-Grid Benchmark Results*

Description

MSE and timing statistics for the **adlift** package on seven irregular-grid test signals: three physically motivated (`linear_phys`, `trend_events`, `blocks_gapped`) and four Donoho-Johnstone classics resampled on irregular grids (`blocks_dj_irr`, `bumps_dj_irr`, `doppler_dj_irr`, `heavisine_dj_irr`). 7-quartile summaries over 1000 simulations per configuration. Companion to `benchmark_nlt_irregular` and `benchmark_rlifting_irregular`; used by the irregular-grid benchmark vignette.

Usage

```
data(benchmark_adlift_irregular)
```

Format

A data frame with 672 rows and 20 columns:

Signal One of the seven irregular-grid test signals (see Description).

Pkg Always `"adlift"`.

Wavelet Predictor family: `"AdaptPred"`, `"CubicPred"`, `"LinearPred"`, or `"QuadPred"`.

Boundary Encoded combination of `adlift::fwtnp` options.

NoiseSd Per-signal Gaussian noise standard deviation (e.g. 0.15 for `linear_phys/trend_events`, 0.30 for the DJ-irregular signals, 0.50 for `blocks_gapped`).

N Number of simulations per configuration (1000).

MSE_min, MSE_q1, MSE_median, MSE_mean, MSE_q3, MSE_max, MSE_se Reconstruction MSE statistics.

Time_min, Time_q1, Time_median, Time_mean, Time_q3, Time_max, Time_se Wall-time statistics (seconds).

Source

`data-raw/generate_adlift_irregular_benchmark.R`

---

benchmark_nlt *nlt Benchmark Results*

Description

MSE and timing statistics for the **nlt** package (nondecimated lifting transform) on the four Donoho-Johnstone signals. 7-quartile summaries over 1000 simulations per configuration. Used as a reference baseline in the irregular-grid benchmark vignette.

Usage

```
data(benchmark_nlt)
```

Format

A data frame with 384 rows and 19 columns:

Signal Donoho-Johnstone test signal.

Pkg Always `"nlt"`.

Wavelet Predictor family inherited from **adlift**.

Boundary Encoded combination of nlt/adlift options.

N Number of simulations per configuration (1000).

MSE_min, MSE_q1, MSE_median, MSE_mean, MSE_q3, MSE_max, MSE_se Reconstruction MSE statistics.

Time_min, Time_q1, Time_median, Time_mean, Time_q3, Time_max, Time_se Wall-time statistics (seconds).

Source

`data-raw/generate_nlt_benchmark.R`

`benchmark_nlt_irregular`

nlt Irregular-Grid Benchmark Results

Description

MSE and timing statistics for the **nlt** package on seven irregular-grid test signals (see `benchmark_adlift_irregular` for the signal set). 7-quartile summaries over 1000 simulations per configuration. Used by the irregular-grid benchmark vignette as a reference baseline.

Usage

```
data(benchmark_nlt_irregular)
```

Format

A data frame with 672 rows and 20 columns:

Signal One of the seven irregular-grid test signals.

Pkg Always `"nlt"`.

Wavelet Predictor family inherited from **adlift**: `"AdaptPred"`, `"CubicPred"`, `"LinearPred"`, or `"QuadPred"`.

Boundary Encoded combination of nlt/adlift options.

NoiseSd Per-signal Gaussian noise standard deviation.

N Number of simulations per configuration (1000).

MSE_min, MSE_q1, MSE_median, MSE_mean, MSE_q3, MSE_max, MSE_se Reconstruction MSE statistics.

Time_min, Time_q1, Time_median, Time_mean, Time_q3, Time_max, Time_se Wall-time statistics (seconds).

Source

`data-raw/generate_nlt_irregular_benchmark.R`

benchmark_rlifting *rLifting Offline/Causal Benchmark Results*

Description

MSE and timing statistics for rLifting denoising across the four Donoho-Johnstone signals, three modes (offline / causal-batch / stream), multiple wavelets, boundary modes, threshold rules, and shrinkage methods. Reported as 7-quartile summaries (min / q1 / median / mean / q3 / max / se) over 1000 simulations per configuration. Used by `vignette("v02-thresholding-and-tuning")` and `vignette("v03-causal-stream")`.

Usage

```
data(benchmark_rlifting)
```

Format

A data frame with 3600 rows and 40 columns:

Signal Donoho-Johnstone test signal: `"blocks"`, `"bumps"`, `"doppler"`, or `"heavisine"`.

Pkg Always `"rLifting"`.

Mode `"offline"`, `"causal"`, or `"stream"`.

Wavelet Lifting scheme: `"haar"`, `"cdf53"`, etc.

Boundary Boundary extension mode.

Method Composite label combining threshold rule and shrinkage.

ThresholdMethod Threshold rule: `"universal"` or `"sure"`.

Shrinkage Shrinkage rule: `"hard"`, `"soft"`, `"semisoft"`, or `"scad"`.

AlphaUsed, BetaUsed Threshold-recursion parameters used.

Version Generator version tag (`"v2"` for current).

N Number of simulations per configuration (1000).

MSE_min, MSE_q1, MSE_median, MSE_mean, MSE_q3, MSE_max, MSE_se Mean-squared-error statistics against the noise-free signal.

MSE_settled_min, MSE_settled_q1, MSE_settled_median, MSE_settled_mean, MSE_settled_q3, MSE_settled_max, MSE statistics computed after dropping the warm-up window (causal/stream modes only).

Time_total_min, Time_total_q1, Time_total_median, Time_total_mean, Time_total_q3, Time_total_max, Time_total Total wall time per call (seconds).

Per_sample_us_min, Per_sample_us_q1, Per_sample_us_median, Per_sample_us_mean, Per_sample_us_q3, Per_sa Per-sample time (microseconds), i.e. Time_total / signal length.

Source

`data-raw/generate_rlifting_benchmark_v2.R`

benchmark_rlifting_irregular *rLifting Irregular-Grid Benchmark Results*

Description

MSE and timing statistics for rLifting denoising on seven irregular-grid test signals (see benchmark_adlift_irregular for the signal set), covering all three modes (offline, causal, stream), six built-in wavelets, five boundary modes, and twelve threshold/shrinkage method combinations (universal/sure x hard/soft/semisoft/scad, with and without tune_alpha_beta where applicable). 7-quartile summaries over 1000 simulations per configuration. Companion to benchmark_adlift_irregular and benchmark_nlt_irregular; used by the irregular-grid benchmark vignette.

Usage

```
data(benchmark_rlifting_irregular)
```

Format

A data frame with 7560 rows and 47 columns:

Signal One of the seven irregular-grid test signals.

Pkg Always "rLifting".

Mode "offline", "causal", or "stream".

Wavelet Lifting scheme: "haar", "db2", "cdf53", "cdf97", "dd4", or "lazy".

Boundary Boundary extension mode: "symmetric", "periodic", "zero", "local_linear", or "one_sided".

Method Composite label combining threshold rule, shrinkage, and tuned/untuned status (e.g. "universal_tuned_soft").

ThresholdMethod Threshold rule: "universal" or "sure".

Shrinkage Shrinkage rule: "hard", "soft", "semisoft", or "scad".

AlphaUsed, BetaUsed Threshold-recursion parameters used (post-tuning when applicable).

NoiseSd Per-signal Gaussian noise standard deviation.

N Number of simulations per configuration (1000).

MSEpos_min, MSEpos_q1, MSEpos_median, MSEpos_mean, MSEpos_q3, MSEpos_max, MSEpos_se MSE statistics with position-aware processing.

MSEfix_min, MSEfix_q1, MSEfix_median, MSEfix_mean, MSEfix_q3, MSEfix_max, MSEfix_se MSE statistics with position-ignoring processing (offline only; NA for causal/stream).

Ratio_min, Ratio_q1, Ratio_median, Ratio_mean, Ratio_q3, Ratio_max, Ratio_se Per-simulation ratio MSEpos / MSEfix summarised over the 1000 simulations (offline only).

Timepos_min, Timepos_q1, Timepos_median, Timepos_mean, Timepos_q3, Timepos_max, Timepos_se Wall-time statistics with position-aware processing (seconds).

Timefix_min, Timefix_q1, Timefix_median, Timefix_mean, Timefix_q3, Timefix_max, Timefix_se Wall-time statistics with position-ignoring processing (offline only).

Details

Each configuration is run twice in offline mode: with position-aware processing (`t = t_phys` passed in, irregular path active) and position-ignoring (`t = NULL`, uniform-grid treatment). The `Ratio_*` columns report `MSEpos / MSEfix` per simulation as a direct measure of the value of irregular handling for that configuration. Causal and stream modes report only position-aware results (`MSEpos_*`/`Timepos_*`); the position-ignoring columns are `NA` for those rows.

Source

`data-raw/generate_rlifting_irregular_benchmark.R`

---

benchmark_wavethresh *wavethresh Benchmark Results*

Description

MSE and timing statistics for the **wavethresh** package on the four Donoho-Johnstone signals, across its native wavelets and boundary/threshold combinations. 7-quartile summaries over 1000 simulations per configuration. Used as a reference baseline in the offline benchmark vignette.

Usage

```
data(benchmark_wavethresh)
```

Format

A data frame with rows per (Signal, Wavelet, Boundary) and 19 columns:

Signal Donoho-Johnstone test signal.

Pkg Always `"wavethresh"`.

Wavelet Daubechies filter family identifier (e.g. `"co1"` for Daubechies-extremal-phase order 1).

Boundary Combination of wavethresh threshold policy and shrinkage rule (e.g. `"BayesThresh_soft"`, `"cv_hard"`).

N Number of simulations per configuration (1000).

MSE_min, MSE_q1, MSE_median, MSE_mean, MSE_q3, MSE_max, MSE_se Reconstruction MSE statistics.

Time_min, Time_q1, Time_median, Time_mean, Time_q3, Time_max, Time_se Wall-time statistics (seconds).

Source

`data-raw/generate_wavethresh_benchmark.R`

`compute_adaptive_threshold` *Calculate Adaptive Threshold (Universal / Recursive)*

Description

Estimates the per-level noise threshold from the finest-level detail coefficients and applies the recursive Liu et al. (2014) decay across levels. This is the step-by-step entry point for the universal threshold rule.

Usage

```
compute_adaptive_threshold(lwt_obj, alpha = 0.3, beta = 1.2)
```

Arguments

`lwt_obj`	Object returned by `lwt()`.
`alpha`	Recursive adjustment parameter (Eq. 9 of Liu et al., 2014).
`beta`	Initial threshold scale factor (Eq. 9 of Liu et al., 2014).

Details

To use SureShrink instead, call `denoise_signal_offline()` or the causal/stream functions with `threshold_method = "sure"` — the SURE branch lives in the C++ engine and is not exposed as a standalone R routine. For automatic selection of `alpha` and `beta`, see `tune_alpha_beta`.

Value

Object of class `adaptive_thresholds` (a list of thresholds).

References

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

Liu, Z., Mi, Y., & Mao, Y. (2014). Improved real-time denoising method based on lifting wavelet transform. *Measurement Science Review*, 14(3), 152–159. doi:10.2478/msr20140020

custom_wavelet *Create a custom wavelet*

Description

Wrapper to create a `lifting_scheme` object from manual steps.

Usage

```
custom_wavelet(name, steps, norm = c(1, 1))
```

Arguments

`name`	Identifier name for the wavelet.
`steps`	List of steps created via `lift_step`.
`norm`	Normalization vector c(K, 1/K).

Value

An object of class `lifting_scheme`.

Examples

```
p1 = lift_step("predict", c(1), position = "center")
u1 = lift_step("update", c(0.5), position = "center")
w = custom_wavelet("HaarManual", list(p1, u1), c(1.41, 0.707))
```

denoise_signal_causal *Causal Batch Denoising (Turbo Simulation)*

Description

Processes a complete signal simulating the sequential arrival of data. Uses the specialized 'C++' class `WaveletEngine` to perform causal filtering efficiently on a historical dataset.

Usage

```
denoise_signal_causal(
  signal,
  scheme,
  levels = 1,
  window_size = 256,
  alpha = 0.3,
  beta = 1.2,
  threshold_method = "universal",
```

```
  shrinkage = NULL,
  a = 3.7,
  method = NULL,
  extension = "symmetric",
  update_freq = 1,
  t = NULL,
  ll_k = 4L
)
```

Arguments

signal	Complete vector of the noisy signal.
scheme	lifting_scheme object.
levels	Decomposition levels.
window_size	Window size.
alpha	Threshold decay parameter (universal rule only). Ignored when threshold_method = "sure".
beta	Threshold gain factor (universal rule only). Ignored when threshold_method = "sure".
threshold_method	Threshold-selection rule. One of "universal" or "sure" (per-level SURE-minimising threshold, capped at the universal value; alpha and beta are unused).
shrinkage	Shrinkage rule: "hard", "soft", "semisoft" (default), or "scad".
a	SCAD shape parameter (must be > 2; default 3.7 per Fan & Li 2001). Used only when shrinkage = "scad".
method	Deprecated. Use shrinkage instead.
extension	Boundary treatment: "symmetric", "periodic", "zero", "local_linear", or "one_sided".
update_freq	Frequency of threshold updates. Set to 0 to freeze thresholds at the warm-up estimate (a warning is emitted). Negative values are rejected.
t	Optional numeric vector of sample time positions (irregular grid). Must be sorted and the same length as signal. Ignored by extension = "one_sided" (with a warning).
ll_k	Local-linear neighbourhood size, used only when extension = "local_linear". Default 4L; minimum 2; clamped to window_size if larger.

Value

Filtered vector (same length as input).

`denoise_signal_offline` *Offline Denoising (Global Batch)*

Description

Performs denoising on the entire signal at once using a non-causal approach. Uses global statistics for recursive threshold calculation (Eq. 9). This function is fully optimized in 'C++' (Zero-Allocation).

Usage

```
denoise_signal_offline(
  signal,
  scheme,
  alpha = 0.3,
  beta = 1.2,
  levels = 3,
  threshold_method = "universal",
  shrinkage = NULL,
  a = 3.7,
  method = NULL,
  extension = "symmetric",
  t = NULL,
  ll_k = 4L
)
```

Arguments

`signal`	Numeric vector containing the complete signal.
`scheme`	A `lifting_scheme` object.
`alpha`	Recursive threshold parameter (universal rule only). Ignored when `threshold_method = "sure"`.
`beta`	Threshold scale factor (universal rule only). Ignored when `threshold_method = "sure"`.
`levels`	Number of decomposition levels.
`threshold_method`	Threshold-selection rule. One of `"universal"` (Donoho-Johnstone universal threshold with the recursive per-level decay parameterised by `alpha` and `beta`) or `"sure"` (SureShrink: per-level SURE-minimising threshold, capped at the universal value; `alpha` and `beta` are unused).
`shrinkage`	Shrinkage rule applied above the threshold: `"hard"`, `"soft"`, `"semisoft"` (default), or `"scad"`.
`a`	SCAD shape parameter (must be > 2; default 3.7 per Fan & Li 2001). Used only when `shrinkage = "scad"`.

method	Deprecated. Use shrinkage instead. If provided, takes precedence over shrinkage with a deprecation warning.
extension	Extension mode: "symmetric", "periodic", "zero", "local_linear", or "one_sided".
t	Optional numeric vector of sample positions for irregular grids. Must be sorted and have the same length as signal. When supplied, irregular-grid Lagrange interpolation is applied in the predict steps (the scheme is validated by .check_irregular_scheme). Ignored by extension = "one_sided" (with a warning).
ll_k	Local-linear neighbourhood size, used only when extension = "local_linear". Default 4L; minimum 2; clamped to the signal length if larger.

Value

Filtered numeric vector (same length as input).

---

diagnose_wavelet *Complete Wavelet Diagnosis*

Description

Runs a battery of physical and mathematical tests on a wavelet.

Usage

```
diagnose_wavelet(wavelet_name, config, verbose = TRUE, plot = TRUE)
```

Arguments

wavelet_name	Name string or a lifting_scheme object.
config	Configuration list (is_ortho, vm_degrees, max_taps).
verbose	Print results to console handling? (Defaults to TRUE).
plot	Boolean. Visualize basis functions during diagnosis? (Defaults to TRUE).

Value

An object of class wavelet_diagnosis (S3), which is a list containing the results of each test. The object has a dedicated print method.

doppler_example *Noisy Doppler Signal Example*

Description

Synthetic Doppler signal contaminated with Gaussian noise. Used in `vignette("v01-introduction")` and the boundary-mode comparison.

Usage

```
data(doppler_example)
```

Format

A data frame with 2048 rows and 3 columns:

index Time index (1..2048).

original The pure Doppler signal.

noisy The signal with added Gaussian noise (sd = 0.5).

ilwt *Inverse Lifting Wavelet Transform ('C++' Accelerated)*

Description

Reconstructs the original signal from wavelet coefficients. Optimized with 'C++' backend.

Usage

```
ilwt(lwt_obj, scheme = NULL)
```

Arguments

`lwt_obj`	Object of class `lwt` returned by `lwt()`. The fields `extension`, `ll_k`, and `t` carried by the object are reused to mirror the forward transform; the inverse cannot be invoked with a different boundary mode or grid.
`scheme`	(Optional) `lifting_scheme` object. If NULL, uses the one from `lwt_obj`.

Value

Numeric vector containing the reconstructed signal.

Examples

```
s = c(1, 2, 3, 4)
sch = lifting_scheme("haar")
fwd = lwt(s, sch)
rec = ilwt(fwd)
print(rec)
```

---

lifting_scheme *Lifting Scheme Constructor*

Description

Creates an S3 object containing the prediction (P) and update (U) steps required for the Lifting Transform.

Usage

```
lifting_scheme(wavelet = "haar", custom_steps = NULL, custom_norm = NULL)
```

Arguments

wavelet	Wavelet name (string). Options: "haar", "db2", "cdf53", "cdf97", "dd4", "lazy".
custom_steps	List of custom steps (optional). If provided, ignores internal lookup.
custom_norm	Normalization vector (optional).

Value

An object of class `lifting_scheme`.

---

lift_step *Create an individual Lifting Step*

Description

Helper function to create prediction (P) or update (U) steps, abstracting the complexity of index management.

Usage

```
lift_step(
  type = c("predict", "update"),
  coeffs,
  start_idx = NULL,
  position = "center",
  degree = NULL
)
```

Arguments

type	Step type: "predict" (P) or "update" (U).
coeffs	Numeric vector containing the filter coefficients.
start_idx	(Optional) Manual start index. If provided, ignores the position parameter. Use this for fine-grained control. The filter reads neighbours at offsets start_idx + 0..(length(coeffs) - 1) relative to the current index.
position	Automatic index adjustment, used only when start_idx is NULL: • "center": centres the filter (default). start_idx = -floor((length(coeffs) - 1) / 2). • "left": causal filter (looks into the past). start_idx = -length(coeffs) + 1. • "right": anti-causal filter (looks into the future). start_idx = 0.
degree	(Optional) Polynomial degree the predict step reproduces exactly. Drives the irregular-grid Lagrange interpolation (see vignette("v06-extensions")). If NULL, inferred as length(coeffs) - 1 when type == "predict" and sum(coeffs) == 1 (interpolating filter); otherwise -1 (filter not interpretable as polynomial interpolation, e.g. CDF 9/7 or DB2 predicts). Update steps always carry degree = -1.

Value

A list list(type, coeffs, start_idx, degree) formatted for the internal lifting engine.

---

lwt *Lifting Wavelet Transform (Forward)*

Description

Performs the Forward Wavelet Transform using the Lifting Scheme. Optimized with 'C++' backend.

Usage

```
lwt(signal, scheme, levels = 1, extension = "symmetric", t = NULL, ll_k = 4L)
```

Arguments

signal	Numeric vector containing the input signal.
scheme	A lifting_scheme object.
levels	Integer. Number of decomposition levels.
extension	Boundary extension mode: "symmetric" (default), "periodic", "zero", "local_linear" (linear extrapolation from boundary samples), or "one_sided" (asymmetric filter renormalisation at the boundary).

t	Optional numeric vector of sample positions for irregular grids. Must be sorted and have the same length as signal. When supplied, irregular-grid Lagrange interpolation is applied in the predict steps and lwt_obj$t is stored for use by ilwt(). Ignored by extension = "one_sided" (with a warning).
ll_k	Local-linear neighbourhood size, used only when extension = "local_linear". Default 4L; minimum 2; clamped to the signal length if larger.

Value

An object of class lwt. It is a list containing coeffs (list of details d1..dn and approximation an), scheme (the scheme object used), levels, original_len, extension, ll_k, and t.

Examples

```
data = c(1, 2, 3, 4, 5, 6, 7, 8)
sch = lifting_scheme("haar")
res = lwt(data, sch, levels = 2)
print(res)
```

---

new_wavelet_stream *Create an Adaptive Wavelet Stream Processor ('C++' Core)*

Description

Generates a stateful function backed by a high-performance 'C++' Ring Buffer engine. It implements Sliding Window + Lifting Decomposition + Adaptive Thresholding in highly efficient time per sample.

Usage

```
new_wavelet_stream(
  scheme,
  window_size = 256,
  levels = 1,
  alpha = 0.3,
  beta = 1.2,
  threshold_method = "universal",
  shrinkage = NULL,
  a = 3.7,
  method = NULL,
  extension = "symmetric",
  update_freq = 1,
  irregular = FALSE,
  ll_k = 4L
)
```

Arguments

scheme	A lifting_scheme object.
window_size	Sliding window size (W). Must be > 8.
levels	Decomposition levels (default 1).
alpha	Threshold decay parameter (universal rule only). Ignored when threshold_method = "sure".
beta	Threshold gain factor (universal rule only). Ignored when threshold_method = "sure".
threshold_method	Threshold-selection rule. One of "universal" or "sure" (per-level SURE-minimising threshold, capped at the universal value; alpha and beta are unused).
shrinkage	Shrinkage rule: "hard", "soft", "semisoft" (default), or "scad".
a	SCAD shape parameter (must be > 2; default 3.7 per Fan & Li 2001). Used only when shrinkage = "scad".
method	Deprecated. Use shrinkage instead.
extension	Boundary handling: "symmetric", "periodic", "zero", "local_linear", or "one_sided".
update_freq	How often to recompute threshold statistics (default 1). Set to 0 to freeze thresholds at the warm-up estimate (a warning is emitted). Negative values are rejected.
irregular	Logical. If TRUE, the returned closure accepts a second argument t_val (the sample's time position) and applies position-aware interpolation in the predict steps.
ll_k	Local-linear neighbourhood size, used only when extension = "local_linear". Default 4L; minimum 2; clamped to window_size if larger.

Value

A closure processor(new_sample, t_val = NULL) that accepts one sample (and optionally its time position) and returns the filtered value.

---

plot.adaptive_thresholds

Plot method for Adaptive Thresholds

Description

Plot method for Adaptive Thresholds

Usage

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

Arguments

x	Object of class adaptive_thresholds.
...	Additional arguments.

Value

Invisibly returns NULL.

---

plot.lifting_scheme *Plot method for Lifting Scheme*

Description

Plot method for Lifting Scheme

Usage

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

Arguments

x	An object of class lifting_scheme.
...	Additional arguments passed to visualize_wavelet_basis.

Value

Invisibly returns NULL.

---

plot.lwt *Plot method for LWT Decomposition*

Description

Plot method for LWT Decomposition

Usage

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

Arguments

x	An object of class lwt.
...	Additional arguments.

Value

Invisibly returns NULL.

---

print.adaptive_thresholds *Print method for Adaptive Thresholds*

Description

Print method for Adaptive Thresholds

Usage

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

Arguments

x	Object of class adaptive_thresholds.
...	Additional arguments.

Value

Invisibly returns x.

---

print.lifting_scheme *Print method*

Description

Print method

Usage

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

Arguments

x	object of class lifting_scheme.
...	additional arguments.

Value

Invisibly returns NULL. Called for side effects (printing).

print.lwt *Print method for LWT*

Description

Print method for LWT

Usage

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

Arguments

x	An object of class lwt.
...	Additional arguments.

Value

Invisibly returns NULL. Called for side effects (printing).

print.wavelet_diagnosis *Print method for Wavelet Diagnosis*

Description

Print method for Wavelet Diagnosis

Usage

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

Arguments

x	Object of class wavelet_diagnosis.
...	Additional arguments.

Value

Invisibly returns x.

print.wavelet_stream *Print method for Wavelet Stream Processor*

Description

Print method for Wavelet Stream Processor

Usage

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

Arguments

x	Object of class wavelet_stream.
...	Additional arguments.

Value

Invisibly returns x.

rLifting *rLifting: High-Performance Wavelet Lifting Transforms*

Description

A unified framework for Wavelet Transforms using the Lifting Scheme. It provides robust tools for offline signal analysis and functional data analysis (FDA), while also enabling high-performance causal processing for real-time applications via a specialized 'C++' core.

Author(s)

Maintainer: Moises da Silva <moisesdff8@gmail.com>

See Also

Useful links:

- https://github.com/mkyou/rLifting
- Report bugs at https://github.com/mkyou/rLifting/issues

threshold *General Thresholding Wrapper*

Description

General Thresholding Wrapper

Usage

```
threshold(x, lambda, method = "soft", a = 3.7)
```

Arguments

x	Input vector.
lambda	Threshold value.
method	One of "hard", "soft", "semisoft", or "scad".
a	SCAD shape parameter, ignored unless method = "scad". Default 3.7 (Fan-Li canonical).

Value

Numeric vector of the same length as x with thresholded coefficients.

threshold_hard *Hard Thresholding*

Description

Sets coefficients below the threshold to zero, keeping others unchanged. Known as the "keep or kill" policy.

Usage

```
threshold_hard(x, lambda)
```

Arguments

x	Vector of coefficients (details).
lambda	Positive threshold value.

Value

Processed vector.

threshold_scad *SCAD Shrinkage (Antoniadis & Fan, 2001)*

Description

Smoothly Clipped Absolute Deviation shrinkage. Three-region rule with continuity at lambda, 2*lambda and a*lambda. Identity above a*lambda eliminates the bias that soft thresholding imposes on large coefficients, while keeping sparsity in the zero region.

Usage

```
threshold_scad(x, lambda, a = 3.7)
```

Arguments

x	Vector of coefficients.
lambda	Positive threshold value.
a	Shape parameter, a > 2. Default 3.7 (Fan-Li canonical).

Value

Processed vector.

threshold_semisoft *Semisoft Shrinkage (Hyperbolic)*

Description

Implementation based on Liu et al. (2014). Combines the stability of Soft Thresholding with the amplitude precision of Hard Thresholding. Function: `sign(x) * sqrt(x^2 - lambda^2)` for values above lambda.

Usage

```
threshold_semisoft(x, lambda)
```

Arguments

x	Vector of coefficients.
lambda	Positive threshold value.

Value

Processed vector.

References

Liu, Z., Mi, Y., & Mao, Y. (2014). Improved real-time denoising method based on lifting wavelet transform. *Measurement Science Review*, 14(3), 152–159. doi:10.2478/msr20140020

`threshold_soft` *Soft Thresholding*

Description

Sets coefficients below the threshold to zero and shrinks others towards zero. Reduces noise variance but introduces amplitude bias.

Usage

```
threshold_soft(x, lambda)
```

Arguments

`x`	Vector of coefficients.
`lambda`	Positive threshold value.

Value

Processed vector.

`tune_alpha_beta` *Tune Adaptive Threshold Parameters via SURE*

Description

Selects the recursive-threshold parameters `alpha` and `beta` that minimise Stein's Unbiased Risk Estimate (SURE) for soft thresholding applied to the wavelet detail coefficients of the supplied signal.

Usage

```
tune_alpha_beta(
  signal,
  scheme,
  levels = 3,
  extension = "symmetric",
  ll_k = 4L,
  alpha_range = c(0, 10),
  beta_range = c(0.5, 3)
)
```

Arguments

signal	Numeric vector.
scheme	A lifting_scheme object.
levels	Decomposition depth.
extension	Boundary mode (passed to lwt).
ll_k	Local-linear neighborhood size (passed to lwt).
alpha_range	Bounds for alpha; default c(0, 10).
beta_range	Bounds for beta; default c(0.5, 3.0).

Details

SURE is computed under the soft-threshold estimator assumption (Donoho & Johnstone, 1995). The chosen parameters are typically usable with shrinkage = "soft", "semisoft", "hard", or "scad", since all four share the same threshold location.

Value

A list with components alpha, beta, sure (the minimised SURE value), and converged (logical).

---

validate_compact_support

Validate Compact Support (FIR Compliance)

Description

Verifies if the impulse response is finite (FIR Filter).

Usage

```
validate_compact_support(scheme, max_width)
```

Arguments

scheme	Object of class lifting_scheme.
max_width	Maximum expected width (number of taps).

Value

List with status and number of active taps.

validate_orthogonality

Validate Orthogonality (Energy Conservation)

Description

Verifies Parseval's Theorem. Only applicable for orthogonal wavelets.

Usage

```
validate_orthogonality(scheme, expected = TRUE, tol = 1e-09)
```

Arguments

scheme	Object of class lifting_scheme.
expected	Boolean. If TRUE, expects orthogonality.
tol	Tolerance (default 1e-9).

Value

List with status and energy ratio (Out/In).

validate_perfect_reconstruction

Validate Perfect Reconstruction (Stress Test)

Description

Verifies wavelet invertibility against a battery of signals.

Usage

```
validate_perfect_reconstruction(scheme, tol = 1e-09)
```

Arguments

scheme	Object of class lifting_scheme.
tol	Numerical error tolerance (default 1e-9).

Value

List with global status and maximum error found.

validate_shift_sensitivity *Validate Shift Sensitivity (Shift Variance)*

Description

Decimated wavelets are not translation invariant. This test quantifies the variation in detail energy when shifting the input signal by 1 sample.

Usage

```
validate_shift_sensitivity(scheme)
```

Arguments

scheme	Object of class lifting_scheme.

Value

List with status and percentage variation.

validate_vanishing_moments *Validate Vanishing Moments*

Description

Verifies if the wavelet cancels polynomials of a specific degree.

Usage

```
validate_vanishing_moments(scheme, degree = 0, tol = 1e-09)
```

Arguments

scheme	Object of class lifting_scheme.
degree	Polynomial degree (0=Constant, 1=Ramp, 2=Parabola...).
tol	Residual energy tolerance (default 1e-9).

Value

List with status and residual energy.

`visualize_wavelet_basis` *Visualize Basis Functions (Scaling and Wavelet)*

Description

Plots the waveform by iterating the reconstruction over several levels.

Usage

```
visualize_wavelet_basis(scheme, plot = TRUE, levels = 8)
```

Arguments

`scheme`	Object of class `lifting_scheme`.
`plot`	Boolean.
`levels`	Number of cascade levels.

Value

Invisibly returns `NULL`. Called for side effects (plotting).

Index