

# Package: alkahest (via r-universe)

July 7, 2024

**Title** Pre-Processing XY Data from Experimental Methods

**Version** 1.1.1

**Maintainer** Nicolas Frerebeau

<nicolas.frerebeau@u-bordeaux-montaigne.fr>

**Description** A lightweight, dependency-free toolbox for pre-processing XY data from experimental methods (i.e. any signal that can be measured along a continuous variable). This package provides methods for baseline estimation and correction, smoothing, normalization, integration and peaks detection. Baseline correction methods includes polynomial fitting as described in Lieber and Mahadevan-Jansen (2003) <doi:10.1366/000370203322554518>, Rolling Ball algorithm after Kneen and Annegarn (1996) <doi:10.1016/0168-583X(95)00908-6>, SNIP algorithm after Ryan et al. (1988) <doi:10.1016/0168-583X(88)90063-8>, 4S Peak Filling after Liland (2015) <doi:10.1016/j.mex.2015.02.009> and more.

**License** GPL (>= 3)

**URL** <https://packages.tesselle.org/alkahest/>,  
<https://github.com/tesselle/alkahest>

**BugReports** <https://github.com/tesselle/alkahest/issues>

**Depends** R (>= 3.5.0)

**Imports** grDevices, methods, stats, utils

**Suggests** knitr, Matrix, rmarkdown, tinytest

**VignetteBuilder** knitr

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**Collate** 'AllGenerics.R' 'alkahest-internal.R' 'alkahest-package.R'  
 'baseline.R' 'data.R' 'integrate.R' 'peaks.R' 'replace.R'  
 'resample.R' 'rescale.R' 'signal.R' 'smooth.R' 'windows.R'  
 'xrd.R'

**Repository** <https://tesselle.r-universe.dev>

**RemoteUrl** <https://github.com/tesselle/alkahest>

**RemoteRef** v1.1.1

**RemoteSha** df22cfc7cd25edb00bfbe50d95a41ab526f0b007

## Contents

baseline_linear . . . . .	3
baseline_peakfilling . . . . .	4
baseline_polynomial . . . . .	5
baseline_rollingball . . . . .	7
baseline_rubberband . . . . .	8
baseline_snip . . . . .	9
BEGe . . . . .	11
integrate_rectangle . . . . .	12
integrate_trapezoid . . . . .	13
ka2_strip_penalized . . . . .	14
LaBr . . . . .	15
peaks_find . . . . .	16
peaks_fwhm . . . . .	18
replace_negative . . . . .	19
replace_threshold . . . . .	20
resample_bin . . . . .	21
resample_down . . . . .	22
resample_interpolate . . . . .	24
rescale_area . . . . .	25
rescale_range . . . . .	26
rescale_total . . . . .	28
rescale_transform . . . . .	29
signal_bind . . . . .	30
signal_correct . . . . .	32
signal_drift . . . . .	33
signal_mean . . . . .	34
signal_shift . . . . .	35
smooth_likelihood . . . . .	36
smooth_loess . . . . .	38
smooth_rectangular . . . . .	39
smooth_savitzky . . . . .	41
smooth_triangular . . . . .	42
smooth_whittaker . . . . .	44
subset . . . . .	46
window_sliding . . . . .	47

<i>baseline_linear</i>	3
window_tumbling . . . . .	48
XRD . . . . .	50
<b>Index</b>	<b>51</b>

---

<i>baseline_linear</i>	<i>Linear Baseline Estimation</i>
------------------------	-----------------------------------

---

## Description

Linear Baseline Estimation

## Usage

```
baseline_linear(x, y, ...)

## S4 method for signature 'numeric,numeric'
baseline_linear(x, y, points = range(x))

## S4 method for signature 'ANY,missing'
baseline_linear(x, points = range(x))
```

## Arguments

<code>x, y</code>	A <b>numeric</b> vector. If <code>y</code> is missing, an attempt is made to interpret <code>x</code> in a suitable way (see <code>grDevices::xy.coords()</code> ).
<code>...</code>	Currently not used.
<code>points</code>	A <b>numeric</b> vector specifying the data points to be used in the fitting process (in <code>x</code> unit).

## Value

Returns a **list** with two components `x` and `y`.

## Author(s)

N. Frerebeau

## See Also

`signal_correct()`

Other baseline estimation methods: `baseline_peakfilling()`, `baseline_polynomial()`, `baseline_rollingball()`, `baseline_rubberband()`, `baseline_snip()`

**Examples**

```
## X-ray diffraction
data("XRD")

## Plot spectrum
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Linear baseline
baseline <- baseline_linear(XRD, points = c(25, 34))

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(baseline, type = "l", col = "red")

## Correct baseline
XRD$count <- XRD$count - baseline$y

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
```

---

baseline\_peakfilling *4S Peak Filling*

---

**Description**

Baseline estimation by iterative mean suppression.

**Usage**

```
baseline_peakfilling(x, y, ...)

## S4 method for signature 'numeric,numeric'
baseline_peakfilling(x, y, n, m, by = 10, lambda = 1600, d = 2, sparse = FALSE)

## S4 method for signature 'ANY,missing'
baseline_peakfilling(x, n, m, by = 10, lambda = 1600, d = 2, sparse = FALSE)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
n	An <b>integer</b> value giving the number of iterations.
m	An odd <b>integer</b> giving the half window size.
by	A length-one <b>numeric</b> vector giving the number of buckets to divide x into.
lambda	An <b>integer</b> giving the smoothing parameter. The larger lambda is, the smoother the curve (see <code>smooth_whittaker()</code> ).
d	An <b>integer</b> specifying the order of the penalty (see <code>smooth_whittaker()</code> ).
sparse	A <b>logical</b> scalar: should sparse matrices be used for computation (see <code>smooth_whittaker()</code> )? If TRUE, <b>Matrix</b> is required.

**Author(s)**

N. Frerebeau

**References**

Liland, K. H. (2015). 4S Peak Filling - baseline estimation by iterative mean suppression. *MethodsX*, 2, 135-140. doi:10.1016/j.mex.2015.02.009.

**See Also**

[signal\\_correct\(\)](#), [smooth\\_whittaker\(\)](#)

Other baseline estimation methods: [baseline\\_linear\(\)](#), [baseline\\_polynomial\(\)](#), [baseline\\_rollingball\(\)](#), [baseline\\_rubberband\(\)](#), [baseline\\_snip\(\)](#)

**Examples**

```
## X-ray diffraction
data("XRD")

## 4S Peak Filling baseline
baseline <- baseline_peakfilling(XRD, n = 10, m = 5, by = 10, sparse = TRUE)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(baseline, type = "l", col = "red")
```

---

baseline\_polynomial    *Polynomial Baseline Estimation*

---

**Description**

Polynomial Baseline Estimation

**Usage**

```
baseline_polynomial(x, y, ...)

## S4 method for signature 'numeric,numeric'
baseline_polynomial(x, y, d = 3, tolerance = 0.001, stop = 100)

## S4 method for signature 'ANY,missing'
baseline_polynomial(x, d = 3, tolerance = 0.001, stop = 100)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
d	An <b>integer</b> giving the degree of the polynomial. Must be less than the number of unique points.
tolerance	A <b>numeric</b> scalar giving the tolerance of difference between iterations.
stop	An <b>integer</b> giving the stopping rule (i.e. maximum number of iterations).

**Value**

Returns a **list** with two components x and y.

**Author(s)**

N. Frerebeau

**References**

Lieber, C. A. and Mahadevan-Jansen, A. (2003). Automated Method for Subtraction of Fluorescence from Biological Raman Spectra. *Applied Spectroscopy*, 57(11): 1363-67. doi:[10.1366/000370203322554518](https://doi.org/10.1366/000370203322554518).

**See Also**

[signal\\_correct\(\)](#)

Other baseline estimation methods: [baseline\\_linear\(\)](#), [baseline\\_peakfilling\(\)](#), [baseline\\_rollingball\(\)](#), [baseline\\_rubberband\(\)](#), [baseline\\_snip\(\)](#)

**Examples**

```
## X-ray diffraction
data("XRD")

## Subset from 20 to 70 degrees
XRD <- signal_select(XRD, from = 20, to = 70)

## Plot spectrum
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Polynomial baseline
baseline <- baseline_polynomial(XRD, d = 4, tolerance = 0.02, stop = 1000)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(baseline, type = "l", col = "red")
```

---

baseline\_rollingball *Rolling Ball Baseline Estimation*

---

## Description

Rolling Ball Baseline Estimation

## Usage

```
baseline_rollingball(x, y, ...)  
  
## S4 method for signature 'numeric,numeric'  
baseline_rollingball(x, y, m, s)  
  
## S4 method for signature 'ANY,missing'  
baseline_rollingball(x, m, s)
```

## Arguments

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Currently not used.
m	An odd <a href="#">integer</a> giving the window size (i.e. the number of adjacent points to be used; see <a href="#">window_sliding()</a> ) for minimization/maximization.
s	An odd <a href="#">integer</a> giving the window size (i.e. the number of adjacent points to be used; see <a href="#">window_sliding()</a> ) for smoothing.

## Value

Returns a [list](#) with two components x and y.

## Note

There will be  $(m - 1)/2$  points both at the beginning and at the end of the data series for which a complete  $m$ -width window cannot be obtained. To prevent data loss, progressively wider/narrower windows are used at both ends of the data series.

## Author(s)

N. Frerebeau

## References

Kneen, M. A. and Annegarn, H. J. (1996). Algorithm for Fitting XRF, SEM and PIXE X-Ray Spectra Backgrounds. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 109/110: 209-213. doi:10.1016/0168583X(95)009086.

**See Also**

[signal\\_correct\(\)](#)

Other baseline estimation methods: [baseline\\_linear\(\)](#), [baseline\\_peakfilling\(\)](#), [baseline\\_polynomial\(\)](#), [baseline\\_rubberband\(\)](#), [baseline\\_snip\(\)](#)

**Examples**

```
## X-ray diffraction
data("XRD")

## Subset from 20 to 70 degrees
XRD <- signal_select(XRD, from = 20, to = 70)

## Plot spectrum
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Rolling Ball baseline
baseline <- baseline_rollingball(XRD, m = 201, s = 151)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(baseline, type = "l", col = "red")
```

---

baseline\_rubberband     *Rubberband Baseline Estimation*

---

**Description**

Rubberband Baseline Estimation

**Usage**

```
baseline_rubberband(x, y, ...)

## S4 method for signature 'numeric,numeric'
baseline_rubberband(x, y, noise = 0, spline = TRUE, ...)

## S4 method for signature 'ANY,missing'
baseline_rubberband(x, noise = 0, spline = TRUE, ...)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Extra arguments to be passed to <a href="#">stats::smooth.spline()</a> .
noise	A length-one <b>numeric</b> vector giving the noise level. Only used if method is "rubberband".
spline	A <b>logical</b> scalar: should spline interpolation through the support points be used instead of linear interpolation? Only used if method is "rubberband".



## Details

A convex envelope of the spectrum is determined and the baseline is estimated as the part of the convex envelope lying below the spectrum. Note that the rubber band does not enter the concave regions (if any) of the spectrum.

## Value

Returns a [list](#) with two components x and y.

## Note

`baseline_rubberband()` is slightly modified from C. Beleites' [hyperSpec::spc.rubberband\(\)](#).

## Author(s)

N. Frerebeau

## See Also

[signal\\_correct\(\)](#)

Other baseline estimation methods: [baseline\\_linear\(\)](#), [baseline\\_peakfilling\(\)](#), [baseline\\_polynomial\(\)](#), [baseline\\_rollingball\(\)](#), [baseline\\_snip\(\)](#)

## Examples

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Plot spectrum
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Rubberband baseline
baseline <- baseline_rubberband(BEGe)

plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")
lines(baseline, type = "l", col = "red")
```

## Description

Sensitive Nonlinear Iterative Peak clipping algorithm.

**Usage**

```
baseline_snip(x, y, ...)

## S4 method for signature 'numeric,numeric'
baseline_snip(x, y, LLS = FALSE, decreasing = FALSE, n = 100)

## S4 method for signature 'ANY,missing'
baseline_snip(x, LLS = FALSE, decreasing = FALSE, n = 100)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
LLS	A <b>logical</b> scalar: should the LLS operator be applied on x before employing SNIP algorithm? Only used if method is "SNIP".
decreasing	A <b>logical</b> scalar: should a decreasing clipping window be used?
n	An <b>integer</b> value giving the number of iterations.

**Value**

Returns a **list** with two components x and y.

**Author(s)**

N. Frerebeau

**References**

Morháč, M., Kliman, J., Matoušek, V., Veselský, M. & Turzo, I. (1997). Background elimination methods for multidimensional gamma-ray spectra. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 401(1), p. 113-132. doi:10.1016/S01689002(97)010231

Morháč, M. & Matoušek, V. (2008). Peak Clipping Algorithms for Background Estimation in Spectroscopic Data. *Applied Spectroscopy*, 62(1), p. 91-106. doi:10.1366/000370208783412762

Ryan, C. G., Clayton, E., Griffin, W. L., Sie, S. H. & Cousens, D. R. (1988). SNIP, a statistics-sensitive background treatment for the quantitative analysis of PIXE spectra in geoscience applications. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 34(3), p. 396-402. doi:10.1016/0168583X(88)900638

**See Also**

[signal\\_correct\(\)](#)

Other baseline estimation methods: [baseline\\_linear\(\)](#), [baseline\\_peakfilling\(\)](#), [baseline\\_polynomial\(\)](#), [baseline\\_rollingball\(\)](#), [baseline\\_rubberband\(\)](#)

## Examples

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Plot spectrum
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## SNIP baseline
baseline <- baseline_snip(BEGe, LLS = FALSE, decreasing = FALSE, n = 100)

plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")
lines(baseline, type = "l", col = "red")
```

---

BEGe

*Gamma-Ray Spectrometry*

---

## Description

Gamma-Ray Spectrometry

## Usage

BEGe

## Format

A `data.frame` with 8192 rows (channels) and 2 variables.

**energy** (keV)

**count**

## See Also

Other datasets: [LaBr](#), [XRD](#)

## Examples

```
data("BEGe")
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")
```

---

integrate\_rectangle *Rectangle Rule*

---

### Description

Approximates the definite integral by using the rectangle rule.

### Usage

```
integrate_rectangle(x, y, ...)  
  
## S4 method for signature 'numeric,numeric'  
integrate_rectangle(x, y, right = FALSE)  
  
## S4 method for signature 'ANY,missing'  
integrate_rectangle(x, right = FALSE)
```

### Arguments

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Currently not used.
right	A <a href="#">logical</a> scalar: should the right rule be used instead of the left rule?

### Value

Returns a [list](#) with two components x and y.

### Author(s)

N. Frerebeau

### See Also

Other integration methods: [integrate\\_trapezoid\(\)](#)

### Examples

```
## Calculate the area under the sine curve from 0 to pi  
# integrate(f = function(x) x^3, lower = 0, upper = 2)  
x <- seq(0, 2, len = 101)  
y <- x^3  
  
plot(x, y, type = "l")  
  
integrate_rectangle(x, y, right = FALSE) # 3.9204  
integrate_rectangle(x, y, right = TRUE) # 4.0804  
integrate_trapezoid(x, y) # 4.0004
```

---

integrate\_trapezoid *Trapezoidal Rule*

---

### Description

Approximates the definite integral by using the trapezoidal rule.

### Usage

```
integrate_trapezoid(x, y, ...)  
  
## S4 method for signature 'numeric,numeric'  
integrate_trapezoid(x, y)  
  
## S4 method for signature 'ANY,missing'  
integrate_trapezoid(x)
```

### Arguments

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Currently not used.

### Value

Returns a [list](#) with two components x and y.

### Author(s)

N. Frerebeau

### See Also

Other integration methods: [integrate\\_rectangle\(\)](#)

### Examples

```
## Calculate the area under the sine curve from 0 to pi  
# integrate(f = function(x) x^3, lower = 0, upper = 2)  
x <- seq(0, 2, len = 101)  
y <- x^3  
  
plot(x, y, type = "l")  
  
integrate_rectangle(x, y, right = FALSE) # 3.9204  
integrate_rectangle(x, y, right = TRUE) # 4.0804  
integrate_trapezoid(x, y) # 4.0004
```

---

ka2\_strip\_penalized *Strip XRD ka2*


---

## Description

Strip XRD ka2

## Usage

```
ka2_strip_penalized(x, y, ...)

## S4 method for signature 'numeric,numeric'
ka2_strip_penalized(
  x,
  y,
  lambda,
  wave = c(1.5406, 1.54443),
  tau = 0.5,
  nseg = 1,
  progress = interactive()
)

## S4 method for signature 'ANY,missing'
ka2_strip_penalized(
  x,
  lambda,
  wave = c(1.5406, 1.54443),
  tau = 0.5,
  nseg = 1,
  progress = interactive()
)
```

## Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
lambda	An <b>integer</b> giving the smoothing parameter. The larger lambda is, the smoother the curve.
wave	A length-two <b>numeric</b> vector giving the characteristic wavelengths of the anode material (defaults to copper).
tau	A length-one <b>numeric</b> vector giving the ratio between $\alpha_1$ and $\alpha_2$ line intensities (defaults to 1/2).
nseg	A length-one <b>numeric</b> vector specifying the number of equally sized segments for B-spline basis matrix computation.
progress	A <b>logical</b> scalar: should a progress bar be displayed?

**Value**

Returns a [list](#) with two components x and y.

**Note**

**Matrix** is required.

**Author(s)**

J. J. de Rooi *et al.* (original R code).

**References**

de Rooi, J. J., van der Pers, N. M., Hendrikx, R. W. A., Delhez, R., Böttger A. J. and Eilers, P. H. C. (2014). Smoothing of X-ray diffraction data and Ka2 elimination using penalized likelihood and the composite link model. *Journal of Applied Crystallography*, 47: 852-860. doi:10.1107/S1600576714005809

**Examples**

```
## Not run:
## X-ray diffraction
data("XRD")

## Subset from 20 to 40 degrees
XRD <- signal_select(XRD, from = 20, to = 40)

## Plot diffractogram
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Penalized likelihood smoothing
lambda <- seq(from = 1, to = 8, length.out = 40)
lambda <- 10^lambda

likelihood <- smooth_likelihood(XRD, lambda = lambda, d = 3)
lines(likelihood, col = "red")

## Strip ka2
ka2 <- ka2_strip_penalized(XRD, lambda = lambda, tau = 0.5, nseg = 1)
lines(ka2, col = "blue")

## End(Not run)
```

**Description**

Gamma-Ray Spectrometry

**Usage**

```
LaBr
```

**Format**

A `data.frame` with 1024 rows (channels) and 2 variables.

**energy** (keV)

**count**

**See Also**

Other datasets: [BEGe](#), [XRD](#)

**Examples**

```
data("LaBr")
plot(LaBr, type = "l", xlab = "Energy (keV)", ylab = "Count")
```

---

peaks\_find

*Find Peaks*

---

**Description**

Finds local maxima in sequential data.

**Usage**

```
peaks_find(x, y, ...)
```

```
## S4 method for signature 'numeric,numeric'
peaks_find(x, y, method = "MAD", SNR = 2, m = NULL, ...)
```

```
## S4 method for signature 'ANY,missing'
peaks_find(x, method = "MAD", SNR = 2, m = NULL, ...)
```

**Arguments**

<code>x, y</code>	A <code>numeric</code> vector. If <code>y</code> is missing, an attempt is made to interpret <code>x</code> in a suitable way (see <code>grDevices::xy.coords()</code> ).
<code>...</code>	Extra parameters to be passed to internal methods.
<code>method</code>	A <code>character</code> string specifying the method to be used for background noise estimation (see below).
<code>SNR</code>	An <code>integer</code> giving the signal-to-noise-ratio for peak detection (see below).
<code>m</code>	An odd <code>integer</code> giving the window size (i.e. the number of adjacent points to be used). If <code>NULL</code> , 5% of the data points is used as the half window size.



## Details

A local maximum has to be the highest one in the given window and has to be higher than  $SNR \times noise$  to be recognized as peak.

The following methods are available for noise estimation:

MAD Median Absolute Deviation.

Note that to improve peak detection, it may be helpful to smooth the data and remove the baseline beforehand.

## Value

Returns a [list](#) with two components x and y.

## Note

There will be  $(m - 1)/2$  points both at the beginning and at the end of the data series for which a complete  $m$ -width window cannot be obtained. To prevent data loss, progressively wider/narrower windows are used at both ends of the data series.

Adapted from Stasia Grinberg's [findPeaks](#) function.

## Author(s)

N. Frerebeau

## See Also

Other peaks detection methods: [peaks\\_fwhm\(\)](#)

## Examples

```
## X-ray diffraction
data("XRD")

## 4S Peak Filling baseline
baseline <- baseline_peakfilling(XRD, n = 10, m = 5, by = 10, sparse = TRUE)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(baseline, type = "l", col = "red")

## Correct baseline
XRD <- signal_drift(XRD, lag = baseline, subtract = TRUE)

## Find peaks
peaks <- peaks_find(XRD, SNR = 3, m = 11)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(peaks, type = "p", pch = 16, col = "red")
abline(h = attr(peaks, "noise"), lty = 2) # noise threshold

## Half-Width at Half-Maximum
```

```
x <- seq(-4, 4, length = 1000)
y <- dnorm(x)

peaks_fwhm(x, y, center = 0) # Expected: 2 * sqrt(2 * log(2))
```

---

peaks_fwhm	<i>Half-Width at Half-Maximum</i>
------------	-----------------------------------

---

### Description

Estimates the Half-Width at Half-Maximum (FWHM) for a given peak.

### Usage

```
peaks_fwhm(x, y, ...)
```

```
## S4 method for signature 'numeric,numeric'
peaks_fwhm(x, y, center)
```

```
## S4 method for signature 'ANY,missing'
peaks_fwhm(x, center)
```

### Arguments

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Currently not used.
center	A <a href="#">numeric</a> value giving the peak position in x units.

### Details

It tries to get the smallest possible estimate.

### Value

A [numeric](#) value.

### Author(s)

N. Frerebeau

### See Also

Other peaks detection methods: [peaks\\_find\(\)](#)

**Examples**

```
## X-ray diffraction
data("XRD")

## 4S Peak Filling baseline
baseline <- baseline_peakfilling(XRD, n = 10, m = 5, by = 10, sparse = TRUE)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(baseline, type = "l", col = "red")

## Correct baseline
XRD <- signal_drift(XRD, lag = baseline, subtract = TRUE)

## Find peaks
peaks <- peaks_find(XRD, SNR = 3, m = 11)

plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
lines(peaks, type = "p", pch = 16, col = "red")
abline(h = attr(peaks, "noise"), lty = 2) # noise threshold

## Half-Width at Half-Maximum
x <- seq(-4, 4, length = 1000)
y <- dnorm(x)

peaks_fwhm(x, y, center = 0) # Expected: 2 * sqrt(2 * log(2))
```

---

replace_negative	<i>Replace Negative Values</i>
------------------	--------------------------------

---

**Description**

Replace Negative Values

**Usage**

```
replace_negative(x, y, ...)
```

## S4 method for signature 'numeric,numeric'

```
replace_negative(x, y, value = 0)
```

## S4 method for signature 'ANY,missing'

```
replace_negative(x, value = 0)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Extra parameters to be passed to threshold.
value	A <b>numeric</b> value to replace negative values.

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**See Also**

Other replacement methods: [replace\\_threshold\(\)](#)

---

replace\_threshold      *Replace Values Below a Given Threshold*

---

**Description**

Replace Values Below a Given Threshold

**Usage**

```
replace_threshold(x, y, threshold, ...)

## S4 method for signature 'numeric,numeric,`function`'
replace_threshold(x, y, threshold, value = 0, ...)

## S4 method for signature 'ANY,missing,`function`'
replace_threshold(x, threshold, value = 0, ...)

## S4 method for signature 'numeric,numeric,numeric'
replace_threshold(x, y, threshold, value = 0, ...)

## S4 method for signature 'ANY,missing,numeric'
replace_threshold(x, threshold, value = 0, ...)
```

**Arguments**

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
threshold	A <a href="#">numeric</a> value or a <a href="#">function</a> that takes a numeric vector as argument and returns a single numeric value.
...	Extra parameters to be passed to threshold.
value	A <a href="#">numeric</a> value to replace values below threshold.

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**See Also**Other replacement methods: [replace\\_negative\(\)](#)

---

`resample_bin`*Bin*

---

**Description**

Averages x values and applies a function to the corresponding y values.

**Usage**

```
resample_bin(x, y, ...)

## S4 method for signature 'numeric,numeric'
resample_bin(x, y, by, f = mean, ...)

## S4 method for signature 'ANY,missing'
resample_bin(x, y, by, f = sum)
```

**Arguments**

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Extra parameters to be passed to f.
by	An <a href="#">integer</a> specifying the binning ratio (i.e. the number of points to be grouped together; see <a href="#">window_tumbling()</a> ).
f	A <a href="#">function</a> that takes a numeric vector of intensities as argument and returns a single numeric vector. Used to estimate the local representative value in each bin (defaults to <a href="#">sum()</a> ; see examples).

**Value**Returns a [list](#) with two components x and y.**Author(s)**

N. Frerebeau

**See Also**Other resampling methods: [resample\\_down\(\)](#), [resample\\_interpolate\(\)](#)

## Examples

```
## X-ray diffraction
data("XRD")

## Plot spectrum
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Bin by 3
XRD_bin_mean <- resample_bin(XRD, by = 3, f = mean)
XRD_bin_min <- resample_bin(XRD, by = 3, f = min)

plot(XRD, type = "l", xlim = c(25, 35),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD_bin_mean, type = "l", col = "red")
lines(XRD_bin_min, type = "l", col = "green")

## Downsample by 10
XRD_down <- resample_down(XRD, by = 10)

plot(XRD, type = "l", xlim = c(20, 40),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD_down, type = "l", col = "red")

## Linearly interpolate
XRD_approx <- resample_interpolate(XRD, from = 20, to = 40, by = 0.02)

plot(XRD, type = "l", xlim = c(20, 40),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD_approx, type = "l", col = "red")
```

---

resample\_down

*Downsample*

---

## Description

Downsample

## Usage

```
resample_down(x, y, ...)
```

## S4 method for signature 'numeric,numeric'

```
resample_down(x, y, by)
```

## S4 method for signature 'ANY,missing'

```
resample_down(x, y, by)
```

**Arguments**

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Currently not used.
by	An <a href="#">integer</a> specifying the downsampling factor.

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**See Also**

Other resampling methods: [resample\\_bin\(\)](#), [resample\\_interpolate\(\)](#)

**Examples**

```
## X-ray diffraction
data("XRD")

## Plot spectrum
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Bin by 3
XRD_bin_mean <- resample_bin(XRD, by = 3, f = mean)
XRD_bin_min <- resample_bin(XRD, by = 3, f = min)

plot(XRD, type = "l", xlim = c(25, 35),
     xlab = expression(2*theta), ylab = "Count")
lines(XRD_bin_mean, type = "l", col = "red")
lines(XRD_bin_min, type = "l", col = "green")

## Downsample by 10
XRD_down <- resample_down(XRD, by = 10)

plot(XRD, type = "l", xlim = c(20, 40),
     xlab = expression(2*theta), ylab = "Count")
lines(XRD_down, type = "l", col = "red")

## Linearly interpolate
XRD_approx <- resample_interpolate(XRD, from = 20, to = 40, by = 0.02)

plot(XRD, type = "l", xlim = c(20, 40),
     xlab = expression(2*theta), ylab = "Count")
lines(XRD_approx, type = "l", col = "red")
```

---

resample\_interpolate *Linearly Interpolate*

---

## Description

Linearly Interpolate

## Usage

```
resample_interpolate(x, y, ...)  
  
## S4 method for signature 'numeric,numeric'  
resample_interpolate(x, y, from, to, by, ...)  
  
## S4 method for signature 'ANY,missing'  
resample_interpolate(x, y, from, to, by, ...)
```

## Arguments

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Extra arguments to be passed to <a href="#">stats::approx()</a> .
from	A length-one <a href="#">numeric</a> vector giving the starting value of the sequence where interpolation is to take place.
to	A length-one <a href="#">numeric</a> vector giving the end value of the sequence where interpolation is to take place.
by	A length-one <a href="#">numeric</a> vector specifying the increment of the sequence.

## Value

Returns a [list](#) with two components x and y.

## Author(s)

N. Frerebeau

## See Also

Other resampling methods: [resample\\_bin\(\)](#), [resample\\_down\(\)](#)

## Examples

```
## X-ray diffraction  
data("XRD")  
  
## Plot spectrum  
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
```



```

## Bin by 3
XRD_bin_mean <- resample_bin(XRD, by = 3, f = mean)
XRD_bin_min <- resample_bin(XRD, by = 3, f = min)

plot(XRD, type = "l", xlim = c(25, 35),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD_bin_mean, type = "l", col = "red")
lines(XRD_bin_min, type = "l", col = "green")

## Downsample by 10
XRD_down <- resample_down(XRD, by = 10)

plot(XRD, type = "l", xlim = c(20, 40),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD_down, type = "l", col = "red")

## Linearly interpolate
XRD_approx <- resample_interpolate(XRD, from = 20, to = 40, by = 0.02)

plot(XRD, type = "l", xlim = c(20, 40),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD_approx, type = "l", col = "red")

```

---

rescale\_area

*Normalize intensities by AUC*


---

## Description

Rescales intensities so that the area under the curve (AUC) is equal to 1.

## Usage

```
rescale_area(x, y, ...)
```

```
## S4 method for signature 'numeric,numeric'
rescale_area(x, y, method = c("rectangle", "trapezoid"), ...)
```

```
## S4 method for signature 'ANY,missing'
rescale_area(x, method = c("rectangle", "trapezoid"), ...)
```

## Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
method	A <b>character</b> string specifying the method for integration. It must be one of "rectangle" or "trapezoid". Any unambiguous substring can be given.

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**See Also**

Other normalization methods: [rescale\\_range\(\)](#), [rescale\\_total\(\)](#), [rescale\\_transform\(\)](#)

**Examples**

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Plot spectrum
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Normalize by area under the curve
BEGe_area <- rescale_area(BEGe)
plot(BEGe_area, type = "l", xlab = "Energy (keV)", ylab = "Count")
integrate_rectangle(BEGe)
integrate_rectangle(BEGe_area)

## Rescale so that intensities sum to 1
BEGe_total <- rescale_total(BEGe, total = 1)
plot(BEGe_total, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Rescale intensities to 0-1
BEGe_range <- rescale_range(BEGe, min = 0, max = 1)
plot(BEGe_range, type = "l", xlab = "Energy (keV)", ylab = "Count")
```

---

rescale\_range

*Rescales intensities to have specified minimum and maximum*

---

**Description**

Rescales intensities to have specified minimum and maximum.

**Usage**

```
rescale_range(x, y, ...)
```

```
rescale_min(x, y, ...)
```

```
rescale_max(x, y, ...)  
  
## S4 method for signature 'numeric,numeric'  
rescale_range(x, y, min = 0, max = 1)  
  
## S4 method for signature 'ANY,missing'  
rescale_range(x, min = 0, max = 1)  
  
## S4 method for signature 'numeric,numeric'  
rescale_min(x, y, min = 0)  
  
## S4 method for signature 'ANY,missing'  
rescale_min(x, min = 0)  
  
## S4 method for signature 'numeric,numeric'  
rescale_max(x, y, max = 1)  
  
## S4 method for signature 'ANY,missing'  
rescale_max(x, max = 1)
```

### Arguments

x, y	A <a href="#">numeric</a> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
...	Currently not used.
min	A length-one <a href="#">numeric</a> vector specifying the output minimum.
max	A length-one <a href="#">numeric</a> vector specifying the output maximum.

### Value

Returns a [list](#) with two components x and y.

### Author(s)

N. Frerebeau

### See Also

Other normalization methods: [rescale\\_area\(\)](#), [rescale\\_total\(\)](#), [rescale\\_transform\(\)](#)

### Examples

```
## gamma-ray spectrometry  
data("BEGe")  
  
## Subset from 2.75 to 200 keV  
BEGe <- signal_select(BEGe, from = 3, to = 200)  
  
## Plot spectrum
```

```

plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Normalize by area under the curve
BEGe_area <- rescale_area(BEGe)
plot(BEGe_area, type = "l", xlab = "Energy (keV)", ylab = "Count")
integrate_rectangle(BEGe)
integrate_rectangle(BEGe_area)

## Rescale so that intensities sum to 1
BEGe_total <- rescale_total(BEGe, total = 1)
plot(BEGe_total, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Rescale intensities to 0-1
BEGe_range <- rescale_range(BEGe, min = 0, max = 1)
plot(BEGe_range, type = "l", xlab = "Energy (keV)", ylab = "Count")

```

---

rescale_total	<i>Rescale intensities to sum to a specified value</i>
---------------	--

---

## Description

Rescales intensities to sum to a specified value.

## Usage

```

rescale_total(x, y, ...)

## S4 method for signature 'numeric,numeric'
rescale_total(x, y, total = 1)

## S4 method for signature 'ANY,missing'
rescale_total(x, total = 1)

```

## Arguments

<code>x, y</code>	A <b>numeric</b> vector. If <code>y</code> is missing, an attempt is made to interpret <code>x</code> in a suitable way (see <code>grDevices::xy.coords()</code> ).
<code>...</code>	Currently not used.
<code>total</code>	A length-one <b>numeric</b> vector specifying the output total. Defaults to 1, i.e. normalizes by total intensity.

## Value

Returns a **list** with two components `x` and `y`.

## Author(s)

N. Frerebeau

**See Also**

Other normalization methods: [rescale\\_area\(\)](#), [rescale\\_range\(\)](#), [rescale\\_transform\(\)](#)

**Examples**

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Plot spectrum
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Normalize by area under the curve
BEGe_area <- rescale_area(BEGe)
plot(BEGe_area, type = "l", xlab = "Energy (keV)", ylab = "Count")
integrate_rectangle(BEGe)
integrate_rectangle(BEGe_area)

## Rescale so that intensities sum to 1
BEGe_total <- rescale_total(BEGe, total = 1)
plot(BEGe_total, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Rescale intensities to 0-1
BEGe_range <- rescale_range(BEGe, min = 0, max = 1)
plot(BEGe_range, type = "l", xlab = "Energy (keV)", ylab = "Count")
```

---

rescale_transform	<i>Transform Intensities</i>
-------------------	------------------------------

---

**Description**

Transform Intensities

**Usage**

```
rescale_transform(x, y, ...)
```

## S4 method for signature 'numeric,numeric'

```
rescale_transform(x, y, f, ...)
```

## S4 method for signature 'ANY,missing'

```
rescale_transform(x, f, ...)
```

## Arguments

- `x, y` A **numeric** vector. If `y` is missing, an attempt is made to interpret `x` in a suitable way (see `grDevices::xy.coords()`).
- `...` Extra arguments to be passed to `f`.
- `f` A **function** that takes a numeric vector of intensities as argument and returns a numeric vector.

## Details

Transformation of intensities can be used to improve the identification of peaks with a low signal-to-noise ratio.

## Value

Returns a **list** with two components `x` and `y`.

## Author(s)

N. Frerebeau

## See Also

Other normalization methods: `rescale_area()`, `rescale_range()`, `rescale_total()`

## Examples

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Plot spectrum
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Transform intensities
BEGe_trans <- rescale_transform(BEGe, f = sqrt)
plot(BEGe_trans, type = "l", xlab = "Energy (keV)", ylab = "sqrt(Count)")
```

---

signal\_bind

*Bind*

---

## Description

Combines XY objects.

**Usage**

```
signal_bind(...)  
  
## S4 method for signature 'ANY'  
signal_bind(...)
```

**Arguments**

... Any object that can be interpreted in a suitable way (see [grDevices::xy.coords\(\)](#)).

**Value**

Returns a [matrix](#) of intensities.

**Author(s)**

N. Frerebeau

**See Also**

Other signal processing methods: [signal\\_correct\(\)](#), [signal\\_drift\(\)](#), [signal\\_mean\(\)](#), [signal\\_shift\(\)](#), [subset\(\)](#)

**Examples**

```
## X-ray diffraction  
data("XRD")  
  
XRD1 <- signal_drift(XRD, lag = 1500)  
  
## Bind  
XRD_bind <- signal_bind(XRD, XRD1)  
XRD_bind[, 1:10]  
  
## Mean  
XRD_mean <- signal_mean(XRD, XRD1)  
  
plot(NULL, type = "l", xlim = c(10, 70) , ylim = c(3000, 36000),  
      xlab = expression(2*theta), ylab = "Count")  
lines(XRD, type = "l")  
lines(XRD1, type = "l")  
lines(XRD_mean, type = "l", col = "red")
```

---

signal\_correct      *Baseline Correction*

---

### Description

Baseline Correction

### Usage

```
signal_correct(x, y, ...)

## S4 method for signature 'numeric,numeric'
signal_correct(x, y, method = c("linear", "rubberband", "SNIP", "4S"), ...)

## S4 method for signature 'ANY,missing'
signal_correct(x, method = c("linear", "rubberband", "SNIP", "4S"), ...)
```

### Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Extra arguments to be passed to <code>baseline_*()</code> (see details).
method	A <b>character</b> string specifying the method for baseline estimation. It must be one of "linear", "rubberband", "SNIP" or "4S" (see details). Any unambiguous substring can be given.

### Details

Available methods for baseline estimation:

linear Linear baseline estimation (see `baseline_linear()`).

rubberband Rubberband baseline estimation (see `baseline_rubberband()`).

SNIP Sensitive Nonlinear Iterative Peak clipping algorithm (see `baseline_snip()`).

4S 4S Peak Filling (see `baseline_peakfilling()`).

### Value

Returns a **list** with two components x and y.

### Author(s)

N. Frerebeau

### See Also

Other signal processing methods: `signal_bind()`, `signal_drift()`, `signal_mean()`, `signal_shift()`, `subset()`



**Examples**

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Drift
baseline <- baseline_snip(BEGe)
BEGe_drif <- signal_drift(BEGe, lag = baseline, subtract = TRUE)

plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")
lines(BEGe_drif, type = "l", col = "red")

## Correct
BEGe_corr <- signal_correct(BEGe, method = "SNIP")

plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")
lines(BEGe_corr, type = "l", col = "red")
```

---

signal\_drift

*Drift Intensities*


---

**Description**

Drift Intensities

**Usage**

```
signal_drift(x, y, lag, ...)

## S4 method for signature 'numeric,numeric,numeric'
signal_drift(x, y, lag)

## S4 method for signature 'ANY,missing,ANY'
signal_drift(x, lag, subtract = FALSE)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <a href="#">grDevices::xy.coords()</a> ).
lag	A <b>numeric</b> vector specifying the offset or any object that can be interpreted in a suitable way (see <a href="#">grDevices::xy.coords()</a> )
...	Currently not used.
subtract	A <b>logical</b> scalar: should lag be subtracted to y?

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**See Also**

Other signal processing methods: [signal\\_bind\(\)](#), [signal\\_correct\(\)](#), [signal\\_mean\(\)](#), [signal\\_shift\(\)](#), [subset\(\)](#)

**Examples**

```
## gamma-ray spectrometry
data("BEGe")

## Subset from 2.75 to 200 keV
BEGe <- signal_select(BEGe, from = 3, to = 200)

## Drift
BEGe_plus <- signal_drift(BEGe, lag = 250)
BEGe_minus <- signal_drift(BEGe, lag = 250, subtract = TRUE)

plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")
lines(BEGe_plus, type = "l", col = "red")
lines(BEGe_minus, type = "l", col = "green")
```

---

signal\_mean

*Mean Intensities*

---

**Description**

Mean Intensities

**Usage**

```
signal_mean(...)

## S4 method for signature 'ANY'
signal_mean(...)
```

**Arguments**

... Any object that can be interpreted in a suitable way (see [grDevices::xy.coords\(\)](#)).

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**See Also**

Other signal processing methods: [signal\\_bind\(\)](#), [signal\\_correct\(\)](#), [signal\\_drift\(\)](#), [signal\\_shift\(\)](#), [subset\(\)](#)

**Examples**

```
## X-ray diffraction
data("XRD")

XRD1 <- signal_drift(XRD, lag = 1500)

## Bind
XRD_bind <- signal_bind(XRD, XRD1)
XRD_bind[, 1:10]

## Mean
XRD_mean <- signal_mean(XRD, XRD1)

plot(NULL, type = "l", xlim = c(10, 70) , ylim = c(3000, 36000),
      xlab = expression(2*theta), ylab = "Count")
lines(XRD, type = "l")
lines(XRD1, type = "l")
lines(XRD_mean, type = "l", col = "red")
```

---

signal\_shift

*Shift the X Scale*

---

**Description**

Shifts the x scale by a given value.

**Usage**

```
signal_shift(x, y, lag, ...)

## S4 method for signature 'numeric,numeric'
signal_shift(x, y, lag)

## S4 method for signature 'ANY,missing'
signal_shift(x, lag)
```

**Arguments**

`x, y` A **numeric** vector. If `y` is missing, an attempt is made to interpret `x` in a suitable way (see `grDevices::xy.coords()`).

`lag` A **numeric** vector specifying the offset.

`...` Currently not used.

**Value**

Returns a **list** with two components `x` and `y`.

**Author(s)**

N. Frerebeau

**See Also**

Other signal processing methods: `signal_bind()`, `signal_correct()`, `signal_drift()`, `signal_mean()`, `subset()`

**Examples**

```
## X-ray diffraction
data("XRD")

## Plot spectrum
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Shift by one degree
XRD_offset <- signal_shift(XRD, lag = 1)
lines(XRD_offset, type = "l", col = "red")
```

---

smooth\_likelihood      *Penalized Likelihood Smoothing*

---

**Description**

Penalized Likelihood Smoothing

**Usage**

```
smooth_likelihood(x, y, ...)
```

```
## S4 method for signature 'numeric,numeric'
smooth_likelihood(x, y, lambda, d = 2, SE = FALSE, progress = interactive())
```

```
## S4 method for signature 'ANY,missing'
smooth_likelihood(x, lambda, d = 2, SE = FALSE, progress = interactive())
```

## Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
lambda	An <b>integer</b> giving the smoothing parameter. The larger lambda is, the smoother the curve.
d	An <b>integer</b> specifying the order of the penalty.
SE	A <b>logical</b> scalar: should standard errors be returned?
progress	A <b>logical</b> scalar: should a progress bar be displayed?

## Value

Returns a **list** with two components x and y.

## Note

**Matrix** is required.

## Author(s)

J. J. de Rooi *et al.* (original R code).

## References

de Rooi, J. J., van der Pers, N. M., Hendrikx, R. W. A., Delhez, R., Böttger A. J. and Eilers, P. H. C. (2014). Smoothing of X-ray diffraction data and Ka2 elimination using penalized likelihood and the composite link model. *Journal of Applied Crystallography*, 47: 852-860. doi:10.1107/S1600576714005809

## See Also

Other smoothing methods: `smooth_loess()`, `smooth_rectangular()`, `smooth_savitzky()`, `smooth_triangular()`, `smooth_whittaker()`

## Examples

```
## Not run:
## X-ray diffraction
data("XRD")

## Subset from 20 to 40 degrees
XRD <- signal_select(XRD, from = 20, to = 40)

## Plot diffractogram
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")

## Penalized likelihood smoothing
lambda <- seq(from = 1, to = 8, length.out = 40)
lambda <- 10^lambda
```

```
likelihood <- smooth_likelihood(XRD, lambda = lambda, d = 3)
lines(likelihood, col = "red")

## Strip ka2
ka2 <- ka2_strip_penalized(XRD, lambda = lambda, tau = 0.5, nseg = 1)
lines(ka2, col = "blue")

## End(Not run)
```

---

smooth\_loess

*Loess Smoothing*

---

### Description

Smooths intensities by loess fitting.

### Usage

```
smooth_loess(x, y, ...)

## S4 method for signature 'numeric,numeric'
smooth_loess(x, y, span = 0.75, ...)

## S4 method for signature 'ANY,missing'
smooth_loess(x, span = 0.75, ...)
```

### Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Extra arguments to be passed to <code>stats::loess()</code> .
span	An <b>integer</b> specifying the degree of smoothing (see <code>stats::loess()</code> ).

### Value

Returns a **list** with two components x and y.

### Author(s)

N. Frerebeau

### See Also

Other smoothing methods: `smooth_likelihood()`, `smooth_rectangular()`, `smooth_savitzky()`, `smooth_triangular()`, `smooth_whittaker()`

## Examples

```
## Simulate data with some noise
x <- seq(-4, 4, length = 100)
y <- dnorm(x) + rnorm(100, mean = 0, sd = 0.01)

## Plot spectrum
plot(x, y, type = "l", xlab = "", ylab = "")

## Rectangular smoothing
unweighted <- smooth_rectangular(x, y, m = 3)
plot(unweighted, type = "l", xlab = "", ylab = "")

## Triangular smoothing
weighted <- smooth_triangular(x, y, m = 5)
plot(weighted, type = "l", xlab = "", ylab = "")

## Loess smoothing
loess <- smooth_loess(x, y, span = 0.75)
plot(loess, type = "l", xlab = "", ylab = "")

## Savitzky-Golay filter
savitzky <- smooth_savitzky(x, y, m = 21, p = 2)
plot(savitzky, type = "l", xlab = "", ylab = "")

## Whittaker smoothing
whittaker <- smooth_whittaker(x, y, lambda = 1600, d = 2)
plot(whittaker, type = "l", xlab = "", ylab = "")
```

---

smooth_rectangular	<i>Rectangular Smoothing</i>
--------------------	------------------------------

---

## Description

Unweighted sliding-average or rectangular Smoothing.

## Usage

```
smooth_rectangular(x, y, ...)
```

## S4 method for signature 'numeric,numeric'

```
smooth_rectangular(x, y, m = 3)
```

## S4 method for signature 'ANY,missing'

```
smooth_rectangular(x, m)
```

## Arguments

**x, y** A **numeric** vector. If y is missing, an attempt is made to interpret x in a suitable way (see `grDevices::xy.coords()`).

... Currently not used.  
 m An odd [integer](#) giving the window size (i.e. the number of adjacent points to be used; see [window\\_sliding\(\)](#)).

### Details

It replaces each point in the signal with the average of  $m$  adjacent points.

### Value

Returns a [list](#) with two components  $x$  and  $y$ .

### Note

There will be  $(m - 1)/2$  points both at the beginning and at the end of the data series for which a complete  $m$ -width window cannot be obtained. To prevent data loss, progressively wider/narrower windows are used at both ends of the data series.

### Author(s)

N. Frerebeau

### See Also

Other smoothing methods: [smooth\\_likelihood\(\)](#), [smooth\\_loess\(\)](#), [smooth\\_savitzky\(\)](#), [smooth\\_triangular\(\)](#), [smooth\\_whittaker\(\)](#)

### Examples

```
## Simulate data with some noise
x <- seq(-4, 4, length = 100)
y <- dnorm(x) + rnorm(100, mean = 0, sd = 0.01)

## Plot spectrum
plot(x, y, type = "l", xlab = "", ylab = "")

## Rectangular smoothing
unweighted <- smooth_rectangular(x, y, m = 3)
plot(unweighted, type = "l", xlab = "", ylab = "")

## Triangular smoothing
weighted <- smooth_triangular(x, y, m = 5)
plot(weighted, type = "l", xlab = "", ylab = "")

## Loess smoothing
loess <- smooth_loess(x, y, span = 0.75)
plot(loess, type = "l", xlab = "", ylab = "")

## Savitzky-Golay filter
savitzky <- smooth_savitzky(x, y, m = 21, p = 2)
plot(savitzky, type = "l", xlab = "", ylab = "")
```



```
## Whittaker smoothing
whittaker <- smooth_whittaker(x, y, lambda = 1600, d = 2)
plot(whittaker, type = "l", xlab = "", ylab = "")
```

---

smooth\_savitzky      *Savitzky-Golay Filter*

---

## Description

Savitzky-Golay Filter

## Usage

```
smooth_savitzky(x, y, ...)

## S4 method for signature 'numeric,numeric'
smooth_savitzky(x, y, m = 3, p = 2)

## S4 method for signature 'ANY,missing'
smooth_savitzky(x, m, p)
```

## Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
m	An odd <b>integer</b> giving the window size (i.e. the number of adjacent points to be used).
p	An <b>integer</b> giving the degree of the polynomial to be used.

## Details

This method is based on the least-squares fitting of polynomials to segments of  $m$  adjacent points.

## Value

Returns a **list** with two components x and y.

## Note

There will be  $(m - 1)/2$  points both at the beginning and at the end of the data series for which a complete  $m$ -width window cannot be obtained. To prevent data loss, the original  $(m - 1)/2$  points at both ends of the data series are preserved.

## Author(s)

N. Frerebeau

## References

- Gorry, P. A. (1990). General Least-Squares Smoothing and Differentiation by the Convolution (Savitzky-Golay) Method. *Analytical Chemistry*, 62(6), p. 570-573. doi:10.1021/ac00205a007.
- Savitzky, A. & Golay, M. J. E. (1964). Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36(8), p. 1627-1639. doi:10.1021/ac60214a047.

## See Also

Other smoothing methods: [smooth\\_likelihood\(\)](#), [smooth\\_loess\(\)](#), [smooth\\_rectangular\(\)](#), [smooth\\_triangular\(\)](#), [smooth\\_whittaker\(\)](#)

## Examples

```
## Simulate data with some noise
x <- seq(-4, 4, length = 100)
y <- dnorm(x) + rnorm(100, mean = 0, sd = 0.01)

## Plot spectrum
plot(x, y, type = "l", xlab = "", ylab = "")

## Rectangular smoothing
unweighted <- smooth_rectangular(x, y, m = 3)
plot(unweighted, type = "l", xlab = "", ylab = "")

## Triangular smoothing
weighted <- smooth_triangular(x, y, m = 5)
plot(weighted, type = "l", xlab = "", ylab = "")

## Loess smoothing
loess <- smooth_loess(x, y, span = 0.75)
plot(loess, type = "l", xlab = "", ylab = "")

## Savitzky-Golay filter
savitzky <- smooth_savitzky(x, y, m = 21, p = 2)
plot(savitzky, type = "l", xlab = "", ylab = "")

## Whittaker smoothing
whittaker <- smooth_whittaker(x, y, lambda = 1600, d = 2)
plot(whittaker, type = "l", xlab = "", ylab = "")
```

---

smooth\_triangular      *Triangular Smoothing*

---

## Description

Weighted sliding-average or triangular smoothing.

**Usage**

```
smooth_triangular(x, y, ...)  
  
## S4 method for signature 'numeric,numeric'  
smooth_triangular(x, y, m = 3)  
  
## S4 method for signature 'ANY,missing'  
smooth_triangular(x, m)
```

**Arguments**

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
m	An odd <b>integer</b> giving the window size (i.e. the number of adjacent points to be used; see <code>window_sliding()</code> ).

**Details**

It replaces each point in the signal with the weighted mean of  $m$  adjacent points.

**Value**

Returns a **list** with two components x and y.

**Note**

There will be  $(m - 1)/2$  points both at the beginning and at the end of the data series for which a complete  $m$ -width window cannot be obtained. To prevent data loss, progressively wider/narrower windows are used at both ends of the data series.

**Author(s)**

N. Frerebeau

**See Also**

Other smoothing methods: `smooth_likelihood()`, `smooth_loess()`, `smooth_rectangular()`, `smooth_savitzky()`, `smooth_whittaker()`

**Examples**

```
## Simulate data with some noise  
x <- seq(-4, 4, length = 100)  
y <- dnorm(x) + rnorm(100, mean = 0, sd = 0.01)  
  
## Plot spectrum  
plot(x, y, type = "l", xlab = "", ylab = "")
```

```

## Rectangular smoothing
unweighted <- smooth_rectangular(x, y, m = 3)
plot(unweighted, type = "l", xlab = "", ylab = "")

## Triangular smoothing
weighted <- smooth_triangular(x, y, m = 5)
plot(weighted, type = "l", xlab = "", ylab = "")

## Loess smoothing
loess <- smooth_loess(x, y, span = 0.75)
plot(loess, type = "l", xlab = "", ylab = "")

## Savitzky-Golay filter
savitzky <- smooth_savitzky(x, y, m = 21, p = 2)
plot(savitzky, type = "l", xlab = "", ylab = "")

## Whittaker smoothing
whittaker <- smooth_whittaker(x, y, lambda = 1600, d = 2)
plot(whittaker, type = "l", xlab = "", ylab = "")

```

---

smooth\_whittaker      *Whittaker Smoothing*

---

## Description

Whittaker Smoothing

## Usage

```

smooth_whittaker(x, y, ...)

## S4 method for signature 'numeric,numeric'
smooth_whittaker(x, y, lambda = 1600, d = 2, sparse = FALSE)

## S4 method for signature 'ANY,missing'
smooth_whittaker(x, lambda = 1600, d = 2, sparse = FALSE)

```

## Arguments

x, y	A <b>numeric</b> vector. If y is missing, an attempt is made to interpret x in a suitable way (see <code>grDevices::xy.coords()</code> ).
...	Currently not used.
lambda	An <b>integer</b> giving the smoothing parameter. The larger lambda is, the smoother the curve.
d	An <b>integer</b> specifying the order of the penalty.
sparse	A <b>logical</b> scalar: should sparse matrices be used for computation? If TRUE, <b>Matrix</b> is required.

**Value**

Returns a [list](#) with two components x and y.

**Author(s)**

N. Frerebeau

**References**

Eilers, P. H. C. (2003). A Perfect Smoother. *Analytical Chemistry*, 75(14): 3631-36. [doi:10.1021/ac034173t](https://doi.org/10.1021/ac034173t).

**See Also**

Other smoothing methods: [smooth\\_likelihood\(\)](#), [smooth\\_loess\(\)](#), [smooth\\_rectangular\(\)](#), [smooth\\_savitzky\(\)](#), [smooth\\_triangular\(\)](#)

**Examples**

```
## Simulate data with some noise
x <- seq(-4, 4, length = 100)
y <- dnorm(x) + rnorm(100, mean = 0, sd = 0.01)

## Plot spectrum
plot(x, y, type = "l", xlab = "", ylab = "")

## Rectangular smoothing
unweighted <- smooth_rectangular(x, y, m = 3)
plot(unweighted, type = "l", xlab = "", ylab = "")

## Triangular smoothing
weighted <- smooth_triangular(x, y, m = 5)
plot(weighted, type = "l", xlab = "", ylab = "")

## Loess smoothing
loess <- smooth_loess(x, y, span = 0.75)
plot(loess, type = "l", xlab = "", ylab = "")

## Savitzky-Golay filter
savitzky <- smooth_savitzky(x, y, m = 21, p = 2)
plot(savitzky, type = "l", xlab = "", ylab = "")

## Whittaker smoothing
whittaker <- smooth_whittaker(x, y, lambda = 1600, d = 2)
plot(whittaker, type = "l", xlab = "", ylab = "")
```

---

subset	<i>Subset</i>
--------	---------------

---

### Description

- `signal_select()` allows to subset by values of `x`.
- `signal_slice()` allows to subset by position along `x`.

### Usage

```

signal_select(x, y, ...)

signal_slice(x, y, ...)

## S4 method for signature 'numeric,numeric'
signal_select(x, y, from, to)

## S4 method for signature 'ANY,missing'
signal_select(x, from, to)

## S4 method for signature 'numeric,numeric'
signal_slice(x, y, subset)

## S4 method for signature 'ANY,missing'
signal_slice(x, subset)

```

### Arguments

<code>x, y</code>	A <b>numeric</b> vector. If <code>y</code> is missing, an attempt is made to interpret <code>x</code> in a suitable way (see <code>grDevices::xy.coords()</code> ).
<code>...</code>	Currently not used.
<code>from, to</code>	A <b>numeric</b> value giving the first and last value (in <code>x</code> unit) to be selected.
<code>subset</code>	An <b>integer</b> vector giving either positive values to keep, or negative values to drop. The values provided must be either all positive or all negative (coerced to integer as by <code>as.integer()</code> ).

### Value

Returns a **list** with two components `x` and `y`.

### Author(s)

N. Frerebeau

### See Also

Other signal processing methods: `signal_bind()`, `signal_correct()`, `signal_drift()`, `signal_mean()`, `signal_shift()`

## Examples

```
## gamma-ray spectrometry
data("BEGe")

## Plot spectrum
plot(BEGe, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Subset from 2.75 keV to 200 keV
BEGe_1 <- signal_select(BEGe, from = 3, to = 200)

## Plot spectrum
plot(BEGe_1, type = "l", xlab = "Energy (keV)", ylab = "Count")

## Subset from the 20th to the 1250th value
BEGe_2 <- signal_slice(BEGe, subset = 20:1250)

## Plot spectrum
plot(BEGe_2, type = "l", xlab = "Energy (keV)", ylab = "Count")
```

---

window\_sliding

*Sliding Windows*

---

## Description

There will be  $(m - 1)/2$  points both at the beginning and at the end of the data series for which a complete  $m$ -width window cannot be obtained. To prevent data loss, progressively wider/narrower windows are evaluated at both ends of the data series.

## Usage

```
window_sliding(n, m, ...)

## S4 method for signature 'integer,integer'
window_sliding(n, m, i = NULL)

## S4 method for signature 'numeric,numeric'
window_sliding(n, m, i = NULL)
```

## Arguments

n	An <a href="#">integer</a> giving the length of the data series (will be coerced with <a href="#">as.integer()</a> and hence truncated toward zero).
m	An odd <a href="#">integer</a> giving the window size, i.e. the number of adjacent points to be used (will be coerced with <a href="#">as.integer()</a> and hence truncated toward zero).
...	Currently not used.
i	A vector <a href="#">integer</a> specifying the indices of the data points for which windows should be returned. If NULL (the default), windows are evaluated for each data point.

**Value**

Returns a length- $n$  [list](#) of [integer](#) vectors (indices of the data points in each window).

**Author(s)**

N. Frerebeau

**See Also**

Other moving windows: [window\\_tumbling\(\)](#)

**Examples**

```
## Length of the data series
n <- 10

## Progressive sliding windows
sliding <- window_sliding(n = n, m = 5)

plot(NULL, xlim = c(1, n), ylim = c(1, 10.5), xlab = "Index", ylab = "Window")
for (i in seq_along(sliding)) {
  w <- sliding[[i]]
  text(x = w, y = rep(i, length(w)), labels = w, pos = 3)
  lines(w, rep(i, length(w)), type = "l", lwd = 2)
}

## Tumbling windows
## (compare with drop = TRUE)
tumbling <- window_tumbling(n = n, m = 3, drop = FALSE)

plot(NULL, xlim = c(1, n), ylim = c(1, 5.5), xlab = "Index", ylab = "Window")
for (i in seq_along(tumbling)) {
  w <- tumbling[[i]]
  text(x = w, y = rep(i, length(w)), labels = w, pos = 3)
  lines(w, rep(i, length(w)), type = "l", lwd = 2)
}
```

---

window\_tumbling

*Tumbling Windows*

---

**Description**

Tumbling Windows



**Usage**

```

window_tumbling(n, m, ...)

## S4 method for signature 'integer,integer'
window_tumbling(n, m, drop = FALSE)

## S4 method for signature 'numeric,numeric'
window_tumbling(n, m, drop = FALSE)

```

**Arguments**

n	An <a href="#">integer</a> giving the length of the data series (will be coerced with <a href="#">as.integer()</a> and hence truncated toward zero).
m	An <a href="#">integer</a> giving the window size, i.e. the number of adjacent points to be used (will be coerced with <a href="#">as.integer()</a> and hence truncated toward zero).
...	Currently not used.
drop	A <a href="#">logical</a> scalar: if m is not a multiple of n, should the last data points be removed so that all windows have the same length?

**Value**

Returns a [list](#) of [integer](#) vectors (indices of the data points in each window).

**Author(s)**

N. Frerebeau

**See Also**

Other moving windows: [window\\_sliding\(\)](#)

**Examples**

```

## Length of the data series
n <- 10

## Progressive sliding windows
sliding <- window_sliding(n = n, m = 5)

plot(NULL, xlim = c(1, n), ylim = c(1, 10.5), xlab = "Index", ylab = "Window")
for (i in seq_along(sliding)) {
  w <- sliding[[i]]
  text(x = w, y = rep(i, length(w)), labels = w, pos = 3)
  lines(w, rep(i, length(w)), type = "l", lwd = 2)
}

## Tumbling windows
## (compare with drop = TRUE)
tumbling <- window_tumbling(n = n, m = 3, drop = FALSE)

```

```
plot(NULL, xlim = c(1, n), ylim = c(1, 5.5), xlab = "Index", ylab = "Window")
for (i in seq_along(tumbling)) {
  w <- tumbling[[i]]
  text(x = w, y = rep(i, length(w)), labels = w, pos = 3)
  lines(w, rep(i, length(w)), type = "l", lwd = 2)
}
```

---

XRD

*Powder X-ray Diffraction*

---

### Description

Powder X-ray Diffraction

### Usage

XRD

### Format

A `data.frame` with 2989 rows and 2 variables.

**theta**

**count**

### See Also

Other datasets: [BEGe](#), [LaBr](#)

### Examples

```
data("XRD")
plot(XRD, type = "l", xlab = expression(2*theta), ylab = "Count")
```

# Index

## \* **baseline estimation methods**

- baseline\_linear, 3
- baseline\_peakfilling, 4
- baseline\_polynomial, 5
- baseline\_rollingball, 7
- baseline\_rubberband, 8
- baseline\_snip, 9

## \* **datasets**

- BEGe, 11
- LaBr, 15
- XRD, 50

## \* **integration methods**

- integrate\_rectangle, 12
- integrate\_trapezoid, 13

## \* **moving windows**

- window\_sliding, 47
- window\_tumbling, 48

## \* **normalization methods**

- rescale\_area, 25
- rescale\_range, 26
- rescale\_total, 28
- rescale\_transform, 29

## \* **peaks detection methods**

- peaks\_find, 16
- peaks\_fwhm, 18

## \* **replacement methods**

- replace\_negative, 19
- replace\_threshold, 20

## \* **resampling methods**

- resample\_bin, 21
- resample\_down, 22
- resample\_interpolate, 24

## \* **signal processing methods**

- signal\_bind, 30
- signal\_correct, 32
- signal\_drift, 33
- signal\_mean, 34
- signal\_shift, 35
- subset, 46

## \* **smoothing methods**

- smooth\_likelihood, 36
- smooth\_loess, 38
- smooth\_rectangular, 39
- smooth\_savitzky, 41
- smooth\_triangular, 42
- smooth\_whittaker, 44

## \* **specialized tools**

- ka2\_strip\_penalized, 14

as.integer(), 46, 47, 49

baseline\_linear, 3, 5, 6, 8–10

baseline\_linear(), 32

baseline\_linear, ANY, missing-method  
(baseline\_linear), 3

baseline\_linear, numeric, numeric-method  
(baseline\_linear), 3

baseline\_linear-method  
(baseline\_linear), 3

baseline\_peakfilling, 3, 4, 6, 8–10

baseline\_peakfilling(), 32

baseline\_peakfilling, ANY, missing-method  
(baseline\_peakfilling), 4

baseline\_peakfilling, numeric, numeric-method  
(baseline\_peakfilling), 4

baseline\_peakfilling-method  
(baseline\_peakfilling), 4

baseline\_polynomial, 3, 5, 5, 8–10

baseline\_polynomial, ANY, missing-method  
(baseline\_polynomial), 5

baseline\_polynomial, numeric, numeric-method  
(baseline\_polynomial), 5

baseline\_polynomial-method  
(baseline\_polynomial), 5

baseline\_rollingball, 3, 5, 6, 7, 9, 10

baseline\_rollingball, ANY, missing-method  
(baseline\_rollingball), 7

baseline\_rollingball, numeric, numeric-method  
(baseline\_rollingball), 7

- baseline\_rollingball-method
  - (baseline\_rollingball), 7
- baseline\_rubberband, 3, 5, 6, 8, 8, 10
- baseline\_rubberband(), 32
- baseline\_rubberband, ANY, missing-method
  - (baseline\_rubberband), 8
- baseline\_rubberband, numeric, numeric-method
  - (baseline\_rubberband), 8
- baseline\_rubberband-method
  - (baseline\_rubberband), 8
- baseline\_snip, 3, 5, 6, 8, 9, 9
- baseline\_snip(), 32
- baseline\_snip, ANY, missing-method
  - (baseline\_snip), 9
- baseline\_snip, numeric, numeric-method
  - (baseline\_snip), 9
- baseline\_snip-method (baseline\_snip), 9
- BEGe, 11, 16, 50
  
- character, 16, 25, 32
  
- data.frame, 11, 16, 50
  
- function, 20, 21, 30
  
- grDevices::xy.coords(), 3, 4, 6–8, 10, 12–14, 16, 18–21, 23–25, 27, 28, 30–34, 36–39, 41, 43, 44, 46
  
- hyperSpec::spc.rubberband(), 9
  
- integer, 4, 6, 7, 10, 14, 16, 21, 23, 37, 38, 40, 41, 43, 44, 46–49
- integrate\_rectangle, 12, 13
- integrate\_rectangle, ANY, missing-method
  - (integrate\_rectangle), 12
- integrate\_rectangle, numeric, numeric-method
  - (integrate\_rectangle), 12
- integrate\_rectangle-method
  - (integrate\_rectangle), 12
- integrate\_trapezoid, 12, 13
- integrate\_trapezoid, ANY, missing-method
  - (integrate\_trapezoid), 13
- integrate\_trapezoid, numeric, numeric-method
  - (integrate\_trapezoid), 13
- integrate\_trapezoid-method
  - (integrate\_trapezoid), 13
  
- ka2\_strip\_penalized, 14
- ka2\_strip\_penalized, ANY, missing-method
  - (ka2\_strip\_penalized), 14
- ka2\_strip\_penalized, numeric, numeric-method
  - (ka2\_strip\_penalized), 14
- ka2\_strip\_penalized-method
  - (ka2\_strip\_penalized), 14
  
- LaBr, 11, 15, 50
- list, 3, 6, 7, 9, 10, 12, 13, 15, 17, 20, 21, 23, 24, 26–28, 30, 32, 34, 36–38, 40, 41, 43, 45, 46, 48, 49
- logical, 4, 8, 10, 12, 14, 33, 37, 44, 49
  
- matrix, 31
  
- numeric, 3, 4, 6–8, 10, 12–14, 16, 18–21, 23–25, 27, 28, 30, 32, 33, 36–39, 41, 43, 44, 46
  
- peaks\_find, 16, 18
- peaks\_find, ANY, missing-method
  - (peaks\_find), 16
- peaks\_find, numeric, numeric-method
  - (peaks\_find), 16
- peaks\_find-method (peaks\_find), 16
- peaks\_fwhm, 17, 18
- peaks\_fwhm, ANY, missing-method
  - (peaks\_fwhm), 18
- peaks\_fwhm, numeric, numeric-method
  - (peaks\_fwhm), 18
- peaks\_fwhm-method (peaks\_fwhm), 18
  
- replace\_negative, 19, 21
- replace\_negative, ANY, missing-method
  - (replace\_negative), 19
- replace\_negative, numeric, numeric-method
  - (replace\_negative), 19
- replace\_negative-method
  - (replace\_negative), 19
- replace\_threshold, 20, 20
- replace\_threshold, ANY, missing, function-method
  - (replace\_threshold), 20
- replace\_threshold, ANY, missing, numeric-method
  - (replace\_threshold), 20
- replace\_threshold, numeric, numeric, function-method
  - (replace\_threshold), 20
- replace\_threshold, numeric, numeric, numeric-method
  - (replace\_threshold), 20
- replace\_threshold-method
  - (replace\_threshold), 20

- resample\_bin, [21](#), [23](#), [24](#)
- resample\_bin, ANY, missing-method  
(resample\_bin), [21](#)
- resample\_bin, numeric, numeric-method  
(resample\_bin), [21](#)
- resample\_bin-method (resample\_bin), [21](#)
- resample\_down, [21](#), [22](#), [24](#)
- resample\_down, ANY, missing-method  
(resample\_down), [22](#)
- resample\_down, numeric, numeric-method  
(resample\_down), [22](#)
- resample\_down-method (resample\_down), [22](#)
- resample\_interpolate, [21](#), [23](#), [24](#)
- resample\_interpolate, ANY, missing-method  
(resample\_interpolate), [24](#)
- resample\_interpolate, numeric, numeric-method  
(resample\_interpolate), [24](#)
- resample\_interpolate-method  
(resample\_interpolate), [24](#)
- rescale\_area, [25](#), [27](#), [29](#), [30](#)
- rescale\_area, ANY, missing-method  
(rescale\_area), [25](#)
- rescale\_area, numeric, numeric-method  
(rescale\_area), [25](#)
- rescale\_area-method (rescale\_area), [25](#)
- rescale\_max (rescale\_range), [26](#)
- rescale\_max, ANY, missing-method  
(rescale\_range), [26](#)
- rescale\_max, numeric, numeric-method  
(rescale\_range), [26](#)
- rescale\_max-method (rescale\_range), [26](#)
- rescale\_min (rescale\_range), [26](#)
- rescale\_min, ANY, missing-method  
(rescale\_range), [26](#)
- rescale\_min, numeric, numeric-method  
(rescale\_range), [26](#)
- rescale\_min-method (rescale\_range), [26](#)
- rescale\_range, [26](#), [26](#), [29](#), [30](#)
- rescale\_range, ANY, missing-method  
(rescale\_range), [26](#)
- rescale\_range, numeric, numeric-method  
(rescale\_range), [26](#)
- rescale\_range-method (rescale\_range), [26](#)
- rescale\_total, [26](#), [27](#), [28](#), [30](#)
- rescale\_total, ANY, missing-method  
(rescale\_total), [28](#)
- rescale\_total, numeric, numeric-method  
(rescale\_total), [28](#)
- rescale\_total-method (rescale\_total), [28](#)
- rescale\_transform, [26](#), [27](#), [29](#), [29](#)
- rescale\_transform, ANY, missing-method  
(rescale\_transform), [29](#)
- rescale\_transform, numeric, numeric-method  
(rescale\_transform), [29](#)
- rescale\_transform-method  
(rescale\_transform), [29](#)
- signal\_bind, [30](#), [32](#), [34–36](#), [46](#)
- signal\_bind, ANY-method (signal\_bind), [30](#)
- signal\_bind-method (signal\_bind), [30](#)
- signal\_correct, [31](#), [32](#), [34–36](#), [46](#)
- signal\_correct(), [3](#), [5](#), [6](#), [8–10](#)
- signal\_correct, ANY, missing-method  
(signal\_correct), [32](#)
- signal\_correct, numeric, numeric-method  
(signal\_correct), [32](#)
- signal\_correct-method (signal\_correct),  
[32](#)
- signal\_drift, [31](#), [32](#), [33](#), [35](#), [36](#), [46](#)
- signal\_drift, ANY, missing, ANY-method  
(signal\_drift), [33](#)
- signal\_drift, numeric, numeric, numeric-method  
(signal\_drift), [33](#)
- signal\_drift-method (signal\_drift), [33](#)
- signal\_mean, [31](#), [32](#), [34](#), [34](#), [36](#), [46](#)
- signal\_mean, ANY-method (signal\_mean), [34](#)
- signal\_mean-method (signal\_mean), [34](#)
- signal\_select (subset), [46](#)
- signal\_select, ANY, missing-method  
(subset), [46](#)
- signal\_select, numeric, numeric-method  
(subset), [46](#)
- signal\_select-method (subset), [46](#)
- signal\_shift, [31](#), [32](#), [34](#), [35](#), [35](#), [46](#)
- signal\_shift, ANY, missing-method  
(signal\_shift), [35](#)
- signal\_shift, numeric, numeric-method  
(signal\_shift), [35](#)
- signal\_shift-method (signal\_shift), [35](#)
- signal\_slice (subset), [46](#)
- signal\_slice, ANY, missing-method  
(subset), [46](#)
- signal\_slice, numeric, numeric-method  
(subset), [46](#)
- signal\_slice-method (subset), [46](#)
- smooth\_likelihood, [36](#), [38](#), [40](#), [42](#), [43](#), [45](#)

- smooth\_likelihood, ANY, missing-method  
(smooth\_likelihood), 36
- smooth\_likelihood, numeric, numeric-method  
(smooth\_likelihood), 36
- smooth\_likelihood-method  
(smooth\_likelihood), 36
- smooth\_loess, 37, 38, 40, 42, 43, 45
- smooth\_loess, ANY, missing-method  
(smooth\_loess), 38
- smooth\_loess, numeric, numeric-method  
(smooth\_loess), 38
- smooth\_loess-method (smooth\_loess), 38
- smooth\_rectangular, 37, 38, 39, 42, 43, 45
- smooth\_rectangular, ANY, missing-method  
(smooth\_rectangular), 39
- smooth\_rectangular, numeric, numeric-method  
(smooth\_rectangular), 39
- smooth\_rectangular-method  
(smooth\_rectangular), 39
- smooth\_savitzky, 37, 38, 40, 41, 43, 45
- smooth\_savitzky, ANY, missing-method  
(smooth\_savitzky), 41
- smooth\_savitzky, numeric, numeric-method  
(smooth\_savitzky), 41
- smooth\_savitzky-method  
(smooth\_savitzky), 41
- smooth\_triangular, 37, 38, 40, 42, 42, 45
- smooth\_triangular, ANY, missing-method  
(smooth\_triangular), 42
- smooth\_triangular, numeric, numeric-method  
(smooth\_triangular), 42
- smooth\_triangular-method  
(smooth\_triangular), 42
- smooth\_whittaker, 37, 38, 40, 42, 43, 44
- smooth\_whittaker(), 4, 5
- smooth\_whittaker, ANY, missing-method  
(smooth\_whittaker), 44
- smooth\_whittaker, numeric, numeric-method  
(smooth\_whittaker), 44
- smooth\_whittaker-method  
(smooth\_whittaker), 44
- stats::approx(), 24
- stats::loess(), 38
- stats::smooth.spline(), 8
- subset, 31, 32, 34–36, 46
- sum(), 21
  
- window\_sliding, 47, 49
- window\_sliding(), 7, 40, 43
  
- window\_sliding, integer, integer-method  
(window\_sliding), 47
- window\_sliding, numeric, numeric-method  
(window\_sliding), 47
- window\_sliding-method (window\_sliding),  
47
- window\_tumbling, 48, 48
- window\_tumbling(), 21
- window\_tumbling, integer, integer-method  
(window\_tumbling), 48
- window\_tumbling, numeric, numeric-method  
(window\_tumbling), 48
- window\_tumbling-method  
(window\_tumbling), 48
  
- XRD, 11, 16, 50