Package ‘transx’

November 27, 2020

Title Transform Univariate Time Series

Version 0.0.1

Description Univariate time series operations that follow an opinionated design.

The main principle of ‘transx’ is to keep the number of observations the same.

Operations that reduce this number have to fill the observations gap.

License GPL-3

Imports rlang

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

URL https://github.com/kvasilopoulos/transx

BugReports https://github.com/kvasilopoulos/transx/issues

Suggests dplyr, ggplot2, cli, testthat, knitr, DescTools, outliers,

rmarkdown, mFilter, covr

VignetteBuilder knitr

Language en-US

Depends R (>= 2.10)

NeedsCompilation no

Author Kostas Vasilopoulos [aut, cre]

(<https://orcid.org/0000-0002-9769-6395>)

Maintainer Kostas Vasilopoulos <k.vasilopoulo@gmail.com>

Repository CRAN

Date/Publication 2020-11-27 11:40:02 UTC

R topics documented:

demean-demedian ......................................................... 2
diffx-rdiffx-ldiffx .................................................... 3
dtrend ................................................................. 4
demean-demedian

Removes measure of centrality from the series

Description

Maturing

Removes the mean, the median or the mode from the series.

Usage

demean(x, na.rm = getOption("transx.na.rm"))

demedian(x, na.rm = getOption("transx.na.rm"))

demod(x, na.rm = getOption("transx.na.rm"))
Arguments

x [univariate vector]
   Univariate vector, numeric or ts object with only one dimension.
na.rm [logical(1): getOption("transx.na.rm")]
   A value indicating whether NA values should be stripped before the computation proceeds.

Value

Returns a vector with the same class and attributes as the input vector.

Examples

```r
x <- c(2, 5, 10, 20, 30)
summary(x)
demean(x)
demedian(x)
demode(x)
```

Description

Maturing

Returns suitably lagged and iterated difference

- `diffx` computes simple differences.
- `rdiffx` computes percentage differences.
- `ldiffx` computes logged differences.

Usage

```r
diffx(x, n = 1L, order = 1L, rho = 1, fill = NA)
rdiffx(x, n = 1L, order = 1L, rho = NULL, fill = NA)
ldiffx(x, n = 1L, order = 1L, rho = 1, fill = NA)
```
Arguments

x        [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

n        [positive integer(1): 1L]
Value indicating which lag to use.

order    [positive integer(1): 1L]
Value indicating the order of the difference.

rho      [numeric(1): NULL]
Value indicating the autocorrelation parameter. The purpose of this parameter is
to provide quasi-differencing assuming the value falls within 0 and 1.

fill     [numeric or function: NA]
Numeric value(s) or function used to fill observations.

Examples

x <- c(2, 4, 8, 20)
diffx(x)
rdiffx(x)
ldiffx(x)

dtrend  Deterministic Trend

description

Stable

Remove global deterministic trend information from the series.

• dt_lin removes the linear trend.
• dt_quad removes the quadratic trend.
• dt_poly removes the nth-degree polynomial trend.

Usage

dtrend_lin(x, bp = NULL, na.rm = getOption("transx.na.rm"))
dtrend_quad(x, bp = NULL, na.rm = getOption("transx.na.rm"))
dtrend_poly(x, degree, bp = NULL, na.rm = getOption("transx.na.rm"))
fill_linear

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

bp [positive integer(1)]
Break points to define piecewise segments of the data.

na.rm [logical(1): getOption("transx.na.rm")]
A value indicating whether NA values should be stripped before the computation proceeds.

degree [positive integer(1)]
Value indicating the degree of polynomial

Value

Returns a vector with the same class and attributes as the input vector.

Examples

set.seed(123)
t <- 1:20

# Linear trend
x <- 3*sin(t) + t
plot(x, cbind(x, dtrend_lin(x)))

# Quadratic trend
x2 <- 3*sin(t) + t + t^2
plot(x, cbind(x2, quad = dtrend_quad(x2)))

# Introduce a breaking point at point = 10
xbp <- 3*sin(t) + t
plot(x, cbind(xbp, lin = dtrend_lin(xbp), lin_bp = dtrend_lin(xbp, bp = 10)))

fill_linear Fill with "linear approximation"

Description

Fill with "linear approximation"

Usage

fill_linear(body, idx, ...)

Arguments

body [numeric vector]
The body of the vector.
idx [integer vector]
the index to replace with.

... Further arguments passed to \link[stats]{approx}

Value

Returns a vector with the same class and attributes as the input vector.

Examples

```r
x <- c(5,3,2,2,5)
xlen <- length(x)
n <- 2
n <- pmin(n, xlen)
idx <- 1:n
body <- x[seq_len(xlen - n)]
fill_linear(body, idx)
```

fill_locf

Fill with "Last Observation Carried Forward"

Description

Fill with "Last Observation Carried Forward"

Usage

```r
fill_locf(body, idx, fail = NA)
```

Arguments

body [numeric vector]
The body of the vector.
idx [integer vector]
the index to replace with.
fail [numeric(1) or numeric vector: fill]
In case it fails to fill some values.

Value

Returns a vector with the same class and attributes as the input vector.
fill_noCB

Examples

```r
tax <- c(5,3,2,2,5)
lagx(x, n = 2, fill = fill_locf)
leadx(x, n = 2, fill = fill_locf)

lagx(x, n = 2, fill = fill_noCB)
leadx(x, n = 2, fill = fill_noCB)
```

**Description**

Fill with "Next observation carried backwards"

**Usage**

```r
fill_noCB(body, idx, fail = NA)
```

**Arguments**

- `body` [numeric vector]
  The body of the vector.
- `idx` [integer vector]
  the index to replace with.
- `fail` [numeric(1) or numeric vector: fill]
  In case it fails to fill some values.

**Value**

Returns a vector with the same class and attributes as the input vector.

**Examples**

```r
x <- c(5,3,2,2,5)
leadx(x, n = 2, fill = fill_locf)

xlen <- length(x)
n <- 2
n <- pmin(n, xlen)
idx <- (xlen - n + 1):xlen
body <- x[-seq_len(n)]
fill_locf(body, idx, NA)
```
### fill_spline

*Fill with "cubic spline interpolation"*

**Description**
Fill with "cubic spline interpolation"

**Usage**

```r
fill_spline(body, idx, ...)
```

**Arguments**
- `body` [numeric vector]
  - The body of the vector.
- `idx` [integer vector]
  - the index to replace with.
- `...` Further arguments passed to `\link[stats]{spline}`

**Value**
Returns a vector with the same class and attributes as the input vector.

**Examples**

```r
x <- c(5,3,NA,2,5)
fill_spline(x, 3)
```

---

### filter_bk

*Baxter-King Filter*

**Description**

This function computes the cyclical component of the Baxter-King filter.

**Usage**

```r
filter_bk(x, fill = NA, ...)
```

**Arguments**
- `x` [univariate vector]
  - Univariate vector, numeric or ts object with only one dimension.
- `fill` [numeric or function: NA]
  - Numeric value(s) or function used to fill observations.
- `...` Further arguments passed to `bkfilter`.
filter_bw

Examples

unemp <- ggplot2::economics$unemploy
unemp_cycle <- filter_bw(unemp)
plotx(cbind(unemp, unemp_cycle))

filter_bw

Butterworth Filter

Description

Maturing

This function computes the cyclical component of the Butterworth filter.

Usage

filter_bw(x, ...)

Arguments

x
[univariate vector]

Univariate vector, numeric or ts object with only one dimension.

...
Further arguments passed to bwfilter.

Examples

unemp <- ggplot2::economics$unemploy
unemp_cycle <- filter_bw(unemp, freq = 10)
plotx(cbind(unemp, unemp_cycle))

filter_cf

Christiano-Fitzgerald Filter

Description

Maturing

This function computes the cyclical component of the Christiano-Fitzgerald filter.

Usage

filter_cf(x, ...)
Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

... Further arguments passed to cffilter.

Examples

unemp <- ggplot2::economics$unemploy
unemp_cycle <- filter_cf(unemp)
plotx(cbind(unemp, unemp_cycle))

filter_hamilton

Hamilton Filter

Description

Maturing
This function computes the cyclical component of the Hamilton filter.

Usage

filter_hamilton(x, p = 4, horizon = 8, fill = NA)

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

p [integer(1): 4]
A value indicating the number of lags

horizon [integer(1): 8]
A value indicating the number of periods to look ahead.

fill [numeric or function: NA]
Numeric value(s) or function used to fill observations.

Value

Returns a vector with the same class and attributes as the input vector.

Examples

unemp <- ggplot2::economics$unemploy
unemp_cycle <- filter_hamilton(unemp)
plotx(cbind(unemp, unemp_cycle))
filter_hp  

Hodrick-Prescot Filter

Description

Maturing
This function computes the cyclical component of the Hodrick-Prescot filter.

Usage

filter_hp(x, ...)

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

... Further arguments passed to hpfilter.

See Also

select_lambda

Examples

unemp <- ggplot2::economics$unemploy
unemp_cycle <- filter_hp(unemp, freq = select_lambda("monthly"))
plotx(cbind(unemp, unemp_cycle))

filter_tr  

Trigonometric regression Filter

Description

Maturing
This function computes the cyclical component of the trigonometric regression filter.

Usage

filter_tr(x, ...)

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

... Further arguments passed to trfilter.
leadx-lagx

Examples

```r
unemp <- ggplot2::economics$unemploy
unemp_cycle <- filter_tr(unemp, pl=8, pu=40)
plotx(cbind(unemp, unemp_cycle))
```

gmean

- **Geometric Mean value**

**Description**
Compute the sample geometric mean.

**Usage**

```r
gmean(x, na.rm = getOption("transx.na.rm"))
```

**Arguments**

- `x` [univariate vector]
  Univariate vector, numeric or ts object with only one dimension.
- `na.rm` [logical(1): getOption("transx.na.rm")]
  A value indicating whether NA values should be stripped before the computation proceeds.

**Value**
Returns a vector with the same class and attributes as the input vector.

leadx-lagx

- **Compute lagged or leading values**

**Description**

**Stable**
Find the "previous" (lagx()) or "next" (leadx()) values in a vector. Useful for comparing values behind of or ahead of the current values.

**Usage**

```r
lagx(x, n = 1L, fill = NA)
leadx(x, n = 1L, fill = NA)
```
modex

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

n [positive integer(1): 1L]
Value indicating the number of positions to lead or lag by.

fill [numeric or function: NA]
Numeric value(s) or function used to fill observations.

Details

This functions has been taken and modified from the dplyr package, however, to reduce dependencies they are not imported.

Value

Returns a vector with the same class and attributes as the input vector.

Examples

x <- c(5,3,2,2,5)
lagx(x)
lagx(x, fill = mean)
lagx(x, fill = fill_nocb)

leadx(x)
leadx(x, fill = fill_locf)

modex

Mode value

Description

Compute the sample median.

Usage

modex(x, na.rm = getOption("transx.na.rm"))
modex_int(x, na.rm = getOption("transx.na.rm"))

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

na.rm [logical(1): getOption("transx.na.rm")]
A value indicating whether NA values should be stripped before the computation proceeds.
out_iqr  

Detect outliers with Tukey’s method

Description
Maturing

Usage
out_iqr(x, cutoff = 1.5, fill = NA, ...)

Arguments

- **x**  
  [univariate vector]  
  Univariate vector, numeric or ts object with only one dimension.
- **cutoff**  
  [numeric(1): 1.5]  
- **fill**  
  [numeric or function: NA]  
  Numeric value(s) or function used to fill observations.
- **...**  
  further arguments passed to quantile.

Examples

```r
out_iqr(c(0,1,3,4,20))
```

out_pt  

Detect outliers with Percentiles

Description
Maturing

Usage
out_pt(x, pt_low = 0.1, pt_high = 0.9, fill = NA)

Arguments

- **x**  
  [univariate vector]  
  Univariate vector, numeric or ts object with only one dimension.
- **pt_low**  
  the lowest quantile
- **pt_high**  
  the highest quantile
- **fill**  
  [numeric or function: NA]  
  Numeric value(s) or function used to fill observations.
out_score_z

Examples

```r
x <- c(1, 3, -1, 5, 10, 100)
out_pt(x)
```

---

out_score_z

Detect outliers with zscore

Description

Maturing

Usage

```r
out_score_z(x, cutoff = 3, fill = NA, ...)
```

Arguments

- `x` [univariate vector]
  Univariate vector, numeric or ts object with only one dimension.
- `cutoff` [numeric(1): 3]
- `fill` [numeric or function: NA]
  Numeric value(s) or function used to fill observations.
- `...` Further arguments passed to `score`.

Examples

```r
out_score_z(c(0,0.1,2,1,3,2.5,2,.5,6,4,100))
```

---

out_score_zrob

Detect outliers Iglewicz and Hoaglin (1993) robust z-score method

Description

Maturing

Usage

```r
out_score_zrob(x, cutoff = 3.5, fill = NA, ...)
```
out_threshold

Detect outliers with upper and lower threshold

Description
Maturing

Usage
out_threshold(x, tlow = NULL, thigh = NULL, fill = NA)

Arguments
x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.
tlow [numeric(1): NULL]
The lower threshold.
thigh [numeric(1): NULL]
The upper threshold.
fill [numeric or function: NA]
Numeric value(s) or function used to fill observations.

Value
Returns a vector with the same class and attributes as the input vector.

Examples
x <- c(1, 3, -1, 5, 10, 100)
out_threshold(x, tlow = 0, fill = 0)
out_threshold(x, thigh = 9, fill = function(x) quantile(x, 0.9))
Description

Maturing

Replace extremely values that are defined by \texttt{min} and \texttt{max}.

Usage

\begin{verbatim}
out_winsorise(x, min = quantile(x, 0.05), max = quantile(x, 0.95))
out_winsorize(x, min = quantile(x, 0.05), max = quantile(x, 0.95))
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{x} \texttt{[univariate vector]}
    \texttt{x} \texttt{Univariate vector, numeric or ts object with only one dimension.}
  \item \texttt{min} \texttt{[numeric(1): quantile(x, 0.05)]}
    \texttt{The lower bound, all values lower than this will be replaced by this value.}
  \item \texttt{max} \texttt{[numeric(1): quantile(x, 0.95)]}
    \texttt{The upper bound, all values above than this will be replaced by this value.}
\end{itemize}

Value

Returns a vector with the same class and attributes as the input vector.

See Also

\texttt{Winsorize}

Examples

\begin{verbatim}
x <- c(1, 3, -1, 5, 10, 100)
out_winsorise(x)
\end{verbatim}
pow

nth Power Transformation

Description

Stable

Usage

\[ \text{pow(x, pow = NULL, modulus = FALSE)} \]

Arguments

\begin{itemize}
\item \textbf{x} \hspace{1cm} \text{[univariate vector]}
  \hspace{1cm} \text{Univariate vector, numeric or ts object with only one dimension.}
\item \textbf{pow} \hspace{1cm} \text{[numeric(1): NA]}
  \hspace{1cm} \text{The nth power.}
\item \textbf{modulus} \hspace{1cm} \text{positive}
\end{itemize}

Value

Returns a vector with the same class and attributes as the input vector.

Examples

\begin{itemize}
\item \text{pow(2, 2)}
\item \text{pow(-2, 2)}
\item \text{pow(-2, 2, TRUE)}
\end{itemize}

pow_boxcox

Box-Cox Transformations

Description

Maturing

Usage

\[ \text{pow_boxcox(x, lambda = NULL, lambda2 = NULL, ...)} \]
Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

lambda [numeric(1): NULL]
Transformation exponent, \(\lambda\).

lambda2 [numeric(1): NULL]
Transformation exponent, \(\lambda_2\).

... Further arguments passed to pow.

Value

Returns a vector with the same class and attributes as the input vector.

References


Examples

```r
set.seed(123)
x <- runif(10)
pow_boxcox(x, 3)
```

Description

Maturing

The transformation was reported to be successful in transform unimodal skewed distribution into normal distribution, but is not quite useful for bimodal or U-shaped distribution.

Usage

```r
pow_manly(x, lambda = NULL)
```

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.

lambda [numeric(1): NULL]
Transformation exponent, \(\lambda\).

Value

Returns a vector with the same class and attributes as the input vector.
Examples

```r
set.seed(123)
x <- runif(10)
pow_manly(x, 3)
```

---

**pow_tukey**

*Tukey Transformations Transformations*

Description

Maturing

Usage

```r
pow_tukey(x, lambda = NULL, ...)
```

Arguments

- `x` [univariate vector]
  
  Univariate vector, numeric or ts object with only one dimension.

- `lambda` [numeric(1): NULL]
  
  Transformation exponent, $\lambda$.

- `...` Further arguments passed to `pow`.

Value

Returns a vector with the same class and attributes as the input vector.

Examples

```r
set.seed(123)
x <- runif(10)
pow_tukey(x, 2)
```

---

**pow_yj**

*Yeo and Johnson(2000) Transformations*

Description

Maturing

Usage

```r
pow_yj(x, lambda = NULL, ...)
```
Arguments

- **x**  
  Univariate vector, numeric or ts object with only one dimension.

- **lambda**  
  Transformation exponent, λ.

- ...  
  Further arguments passed to `pow`.

Value

Returns a vector with the same class and attributes as the input vector.

References


Examples

```r
set.seed(123)
x <- runif(10)
pow_yj(x, 3)
```

---

**Description**

Maturing

Change the base year.

**Usage**

```r
rebase(x, n = NULL)
rebase_origin(x)
```

**Arguments**

- **x**  
  Univariate vector, numeric or ts object with only one dimension.

- **n**  
  The index of the new base year.

**Value**

Returns a vector with the same class and attributes as the input vector.
Examples

```r
x <- 3:10

# New base would be 5
rebase(x, 5)

# Or the origin
rebase_origin(x)

# Fro the base to be 100 or 0 then:
rebase(x, 5)*100
rebase(x, 5) - 1
```

---

**root**

*nth Root Transformation*

Description

**Stable**

- *root*: nth root
- *root_sqrt*: square root
- *root_cubic*: cubic root

Usage

```r
root(x, root = NULL, modulus = FALSE)

root_sq(x, ...)

root_cubic(x, ...)
```

Arguments

- **x** [univariate vector]
  Univariate vector, numeric or ts object with only one dimension.
- **root** [numeric(1): NA]
  The nth root.
- **modulus** [logical(1): FALSE]
  Transformation will work for data with both positive and negative root.
- **...**
  Further arguments passed to root.
scale_range

Examples

root(4, 2)
root(-4, 2)
root(-4, 2, TRUE)

description

Maturing

Usage

scale_range(x, to, na.rm = getOption("transx.na.rm"))
scale_minmax(x, na.rm = getOption("transx.na.rm"))
scale_unit_len(x, na.rm = getOption("transx.na.rm"))

Arguments

x [univariate vector]
Univariate vector, numeric or ts object with only one dimension.
to [numeric(2): NULL]
Values that will determine the output range.
na.rm [logical(1): getOption("transx.na.rm")]
A value indicating whether NA values should be stripped before the computation proceeds.

Details

To rescale a range between an arbitrary set of values [a, b], the formula becomes:

Value

Returns a vector with the same class and attributes as the input vector.

Examples

x <- c(10, 5, 1, -2)
scale_range(x, c(-1, 2))
scale_minmax(x)
score transformation

Description

Stable

These functions calculate the scores according to:

- `score_z`: Normal(z) distribution
- `score_mad`: Mean absolute deviation
- `score_t`: t-distribution
- `score_chisq`: chi-distribution

Usage

```r
score_z(x, na.rm = getOption("transx.na.rm"))
score_mad(x, na.rm = getOption("transx.na.rm"))
score_t(x, na.rm = getOption("transx.na.rm"))
score_chisq(x, na.rm = getOption("transx.na.rm"))
```

Arguments

- `x` [univariate vector]
  Univariate vector, numeric or ts object with only one dimension.
- `na.rm` [logical(1): getOption("transx.na.rm")]
  A value indicating whether NA values should be stripped before the computation proceeds.

Details

Because function are known with different names:

- `score_z` is identical to `std_mean`
- `score_mad` is identical to `std_median`

Value

Returns a vector with the same class and attributes as the input vector.

See Also

- `scores`
select_lambda

Examples

x <- seq(-3, 3, 0.5)
score_z(x)
score_mad(x)
score_t(x)

Description

Approaches to selecting lambda.

Usage

select_lambda(
  freq = c("quarterly", "annual", "monthly", "weekly"),
  type = c("rot", "ru2002")
)

Arguments

freq [character: "quarterly"]
The frequency of the dataset.
type [character: "rot"]
The methodology to select lambda.

Details

Rule of thumb is from Hodrick and Prescott (1997):

- Lambda = 100*(number of periods in a year)^2
- Annual data = 100 x 1^2 = 100
- Quarterly data = 100 x 4^2 = 1,600
- Monthly data = 100 x 12^2 = 14,400
- Weekly data = 100 x 52^2 = 270,400
- Daily data = 100 x 365^2 = 13,322,500

Ravn and Uhlig (2002) state that lambda should vary by the fourth power of the frequency observation ratio:

- Lambda = 6.25 x (number of periods in a year)^4

Thus, the rescaled default values for lambda are:
• Annual data = 1600 x 1^4 = 6.25
• Quarterly data = 1600 x 4^4 = 1600
• Monthly data = 1600 x 12^4 = 129,600
• Weekly data = 1600 x 12^4 = 33177,600

References

<table>
<thead>
<tr>
<th>skewness</th>
<th>Skewness/Kurtosis Value</th>
</tr>
</thead>
</table>

Description
Compute the sample skewness/kurtosis

Usage
skewness(x, na.rm = getOption("transx.na.rm"))
kurtosis(x, na.rm = getOption("transx.na.rm"))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>[univariate vector] Univariate vector, numeric or ts object with only one dimension.</td>
</tr>
<tr>
<td>na.rm</td>
<td>[logical(1): getOption(&quot;transx.na.rm&quot;)]) A value indicating whether NA values should be stripped before the computation proceeds.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>std</th>
<th>Standardization</th>
</tr>
</thead>
</table>

Description
Maturing
Convert number of standard deviations by which the value of a raw score is above or below the mean value of what is being observed or measured.
std

Usage

std_mean(x, na.rm = getOption("transx.na.rm"))

std_median(x, na.rm = getOption("transx.na.rm"))

Arguments

x [univariate vector]

Univariate vector, numeric or ts object with only one dimension.

na.rm [logical(1): getOption("transx.na.rm")]

A value indicating whether NA values should be stripped before the computation proceeds.

Value

Returns a vector with the same class and attributes as the input vector.

Examples

x <- c(10,2,5,3)
std_mean(x)
scale(x)

std_median(x)
Index

bkfilter, 8
bwfilter, 9

cffilter, 10

demean (demean-demedian), 2
demean-demedian, 2
demedian (demean-demedian), 2
demode (demean-demedian), 2
diffx (diffx-rdiffx-ldiffx), 3
diffx-rdiffx-ldiffx, 3
dtrend, 4
dtrend_lin (dtrend), 4
dtrend_poly (dtrend), 4
dtrend_quad (dtrend), 4

fill_linear, 5
fill_locf, 6
fill_nocb, 7
fill_spline, 8
filter_bk, 8
filter_bw, 9
filter_cf, 9
filter_hamilton, 10
filter_hp, 11
filter_tr, 11

gmean, 12

hpfilter, 11

kurtosis (skewness), 26

lagx (leadx-lagx), 12
ldiffx (diffx-rdiffx-ldiffx), 3
leadx (leadx-lagx), 12
leadx-lagx, 12

modex, 13
modex_int (modex), 13

out_iqr, 14

out_pt, 14
out_score_z, 15
out_score_zrob, 15
out_threshold, 16
out_winsorise, 17
outWinsorize (out_winsorise), 17

pow, 18
pow_boxcox, 18
pow_manly, 19
pow_tukey, 20
pow_yj, 20

rdiffx (diffx-rdiffx-ldiffx), 3
rebase, 21
rebase_origin (rebase), 21
root, 22
root_cubic (root), 22
root_sq (root), 22

scale_minmax (scale_range), 23
scale_range, 23
scale_unit_len (scale_range), 23
score, 24
score_chisq (score), 24
score_mad (score), 24
score_t (score), 24
score_z (score), 24
scores, 24
select_lambda, 25
skewness, 26

std, 26
std_mean (std), 26
std_median (std), 26

trfilter, 11

Winsorize, 17