Package ‘runner’

May 17, 2020

Title Running Operations for Vectors

Type Package

Version 0.3.7

Depends R (>= 3.0)

Language en-US

Encoding UTF-8

Maintainer Dawid Kałędkowski <dawid.kaledkowski@gmail.com>

Description Fully supports rolling windows operations by controlling window length, window lag, time indexing. With runner one can apply any R function on rolling windows. Package eases work with equally and unequally spaced time series.

License GPL (>= 2)

BugReports https://github.com/gogonzo/runner/issues

LinkingTo Rcpp

Imports methods, Rcpp

Suggests dplyr, knitr, magrittr, rmarkdown, spelling, testthat

VignetteBuilder knitr

RoxygenNote 7.1.0

NeedsCompilation yes

Author Dawid Kałędkowski [aut, cre] (<https://orcid.org/0000-0001-9533-457X>)

Repository CRAN

Date/Publication 2020-05-17 01:10:06 UTC

R topics documented:

`fill_run` *Fill NA with previous non-NA element*

Description

Fill `NA` with last non-NA element.

Usage

```
fill_run(x, run_for_first = FALSE, only_within = FALSE)
```

Arguments

`x`	(`vector`, `data.frame`, `matrix`) Input in runner custom function `f`.
`run_for_first`	If first elements are filled with `NA`, `run_for_first` = `TRUE` allows to fill all initial `NA` with nearest non-NA value. By default `run_for_first` = `TRUE`
`only_within`	`NA` are replaced only if previous and next non-NA values are the same. By default `only_within` = `TRUE`

Value

vector - `x` containing all `x` elements with `NA` replaced with previous non-NA element.

Examples

```
fill_run(c(NA, NA,1:10, NA, NA), run_for_first = TRUE)
fill_run(c(NA, NA,1:10, NA, NA), run_for_first = TRUE)
fill_run(c(NA, NA,1:10, NA, NA), run_for_first = FALSE)
fill_run(c(NA, NA, 1, 2, NA, NA, 2, 2, NA, NA, 1, NA, NA), run_for_first = TRUE, only_within = TRUE)
```

k_by *Converts k and lag from time-unit-interval to int*

Description

Converts k and lag from time-unit-interval to int

Usage

```
k_by(k, idx, param)
```

Arguments

k	(`integer` vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k)` == `length(x)` different window size for each element. One can also specify `k` in the same way as by in `seq.POSIXt`. More in details.
idx	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
param	name of the parameter to be printed in error message

Examples

```
k <-  "1 month"
idx <- seq(as.POSIXct("2019-01-01 03:02:01"), as.POSIXct("2020-01-01 03:02:01"), by = "month")
k_difftime <- runner:::k_by(k, idx, param = "k")
idx - k_difftime
```

lag_run *Lag dependent on variable*

Description

Vector of input lagged along integer vector

Usage

```
lag_run(x, lag = 1L, idx = integer(0), nearest = FALSE)
```

Arguments

x	(vector, data.frame, matrix) Input in runner custom function f.
lag	(integer vector or single value) Denoting window lag. If lag is a single value then window lag is constant for all elements, otherwise if length(lag) == length(x) different window size for each element. Negative value shifts window forward. One can also specify lag in the same way as by in seq.POSIXt. More in details.
idx	(integer, Date, POSIXt) Optional integer vector containing sorted (ascending) index of observation. By default idx is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then k and lag are depending on idx. Length of idx have to be equal of length x.
nearest	logical single value. Applied when idx is used, then nearest = FALSE returns observation lagged exactly by the specified number of "periods". When nearest = TRUE function returns latest observation within lag window.

Examples

```
lag_run(1:10, lag = 3)
lag_run(letters[1:10], lag = -2, idx = c(1, 1, 1, 2, 3, 4, 6, 7, 8, 10))
lag_run(letters[1:10], lag = 2, idx = c(1, 1, 1, 2, 3, 4, 6, 7, 8, 10), nearest = TRUE)
```

---

length_run *Length of running windows*

Description

Number of elements in k-long window calculated on idx vector. If idx is an ‘as.integer(date)‘ vector, then k=number of days in window - then the result is number of observations within k days window.

Usage

```
length_run(k = integer(1), lag = integer(1), idx = integer(0))
```

Arguments

k	(integer vector or single value) Denoting size of the running window. If k is a single value then window size is constant for all elements, otherwise if length(k) == length(x) different window size for each element. One can also specify k in the same way as by in seq.POSIXt. More in details.

lag	(integer vector or single value) Denoting window lag. If lag is a single value then window lag is constant for all elements, otherwise if length(lag) == length(x) different window size for each element. Negative value shifts window forward. One can also specify lag in the same way as by in seq.POSIXt. More in details.
idx	(integer, Date, POSIXt) Optional integer vector containing sorted (ascending) index of observation. By default idx is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then k and lag are depending on idx. Length of idx have to be equal of length x.

Examples

```
length_run(k = 3, idx = c(1, 2, 2, 4, 5, 5, 5, 5, 5, 5))
```

---

max_run   *Running maximum*

Description

min_run calculates running max on given x numeric vector, specified k window size.

Usage

```
max_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_rm = TRUE,
  na_pad = FALSE
)
```

Arguments

x	(vector, data.frame, matrix) Input in runner custom function f.
k	(integer vector or single value) Denoting size of the running window. If k is a single value then window size is constant for all elements, otherwise if length(k) == length(x) different window size for each element. One can also specify k in the same way as by in seq.POSIXt. More in details.
lag	(integer vector or single value) Denoting window lag. If lag is a single value then window lag is constant for all elements, otherwise if length(lag) == length(x) different window size for each element. Negative value shifts window forward. One can also specify lag in the same way as by in seq.POSIXt. More in details.

idx	(integer, Date, POSIXt) Optional integer vector containing sorted (ascending) index of observation. By default idx is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then k and lag are depending on idx. Length of idx have to be equal of length x.
at	(integer, Date, POSIXt, character vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also POSIXt sequence increment seq.POSIXt. More in details.
na_rm	logical single value (default na_rm = TRUE) - if TRUE sum is calculating excluding NA.
na_pad	(logical single value) Whether incomplete window should return NA (if na_pad = TRUE) Incomplete window is when some parts of the window are out of range.

Value

max numeric vector of length equals length of x.

Examples

```
set.seed(11)
x1 <- sample( c(1,2,3), 15, replace=TRUE)
x2 <- sample( c(NA,1,2,3), 15, replace=TRUE)
k  <- sample( 1:4, 15, replace=TRUE)
max_run(x1) # simple cumulative maximum
max_run(x2, na_rm = TRUE) # cumulative maximum with removing NA.
max_run(x2, na_rm = TRUE, k=4) # maximum in 4-element window
max_run(x2, na_rm = FALSE, k=k) # maximum in varying k window size
```

---

mean_run *Running mean*

Description

Running mean in specified window of numeric vector.

Usage

```
mean_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_rm = TRUE,
  na_pad = FALSE
)
```

Arguments

x	`numeric` vector which running function is calculated on
k	(`integer`‘ vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k) == length(x)` different window size for each element.
lag	(`integer` vector or single value) Denoting window lag. If `lag` is a single value then window lag is constant for all elements, otherwise if `length(lag) == length(x)` different window size for each element. Negative value shifts window forward.
idx	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
at	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at.
na_rm	`logical` single value (default `na_rm = TRUE`) - if `TRUE` sum is calculating excluding `NA`.
na_pad	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.

Value

mean numeric vector of length equals length of `x`.

Examples

```
set.seed(11)
x1 <- rnorm(15)
x2 <- sample(c(rep(NA,5), rnorm(15)), 15, replace = TRUE)
k <- sample(1:15, 15, replace = TRUE)
mean_run(x1)
mean_run(x2, na_rm = TRUE)
mean_run(x2, na_rm = FALSE )
mean_run(x2, na_rm = TRUE, k=4)
```

`minmax_run` *Running min/max*

Description

`min_run` calculates running minimum-maximum on given `x` numeric vector, specified `k` window size.

Usage

```
minmax_run(x, metric = "min", na_rm = TRUE)
```

Arguments

`x`	(`vector`, `data.frame`, `matrix`) Input in runner custom function `f`.
`metric`	`character` what to return, minimum or maximum
`na_rm`	`logical` single value (default `na_rm` = `TRUE`) - if `TRUE` sum is calculating excluding `NA`.

Value

list.

`min_run` *Running minimum*

Description

`min_run` calculates running min on given `x` numeric vector, specified `k` window size.

Usage

```
min_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_rm = TRUE,
  na_pad = FALSE
)
```

Arguments

`x`	(`vector`, `data.frame`, `matrix`) Input in runner custom function `f`.
`k`	(`integer` vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k)` == `length(x)` different window size for each element. One can also specify `k` in the same way as by in `seq.POSIXt`. More in details.

`lag`	(`integer` vector or single value) Denoting window lag. If `lag` is a single value then window lag is constant for all elements, otherwise if `length(lag) == length(x)` different window size for each element. Negative value shifts window forward. One can also specify `lag` in the same way as by in `seq.POSIXt`. More in details.
`idx`	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
`at`	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also `POSIXt` sequence increment `seq.POSIXt`. More in details.
`na_rm`	`logical` single value (default `na_rm = TRUE`) - if `TRUE` sum is calculating excluding `NA`.
`na_pad`	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.

Value

min numeric vector of length equals length of `x`.

Examples

```
set.seed(11)
x1 <- sample(c(1, 2, 3), 15, replace = TRUE)
x2 <- sample(c(NA, 1, 2, 3), 15, replace = TRUE)
k  <- sample(1:4, 15, replace = TRUE)
min_run(x1)
min_run(x2, na_rm = TRUE)
min_run(x2, na_rm = TRUE, k = 4)
min_run(x2, na_rm = FALSE, k = k)
```

`reformat_k` *Formats time-unit-interval to valid for runner*

Description

Formats time-unit-interval to valid for runner

Usage

```
reformat_k(k, only_positive = TRUE)
```

Arguments

k	(k or lag) object from runner to be formatted
only_positive	for k is TRUE, for lag is FALSE

Examples

```
runner:::reformat_k("1 days")
runner:::reformat_k("day")
runner:::reformat_k("10 days")
runner:::reformat_k("-10 days", only_positive = FALSE)
runner:::reformat_k(c("-10 days", "2 months"), only_positive = FALSE)
```

runner *Apply running function*

Description

Applies custom function on running windows.

Usage

```
runner(
  x,
  f = function(x) x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_pad = FALSE,
  type = "auto",
  ...
)

## S3 method for class 'data.frame'
runner(
  x,
  f = function(x) x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_pad = FALSE,
  type = "auto",
  ...
)

## Default S3 method:
```

```
runner(
  x,
  f = function(x) x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_pad = FALSE,
  type = "auto",
  ...
)
```

Arguments

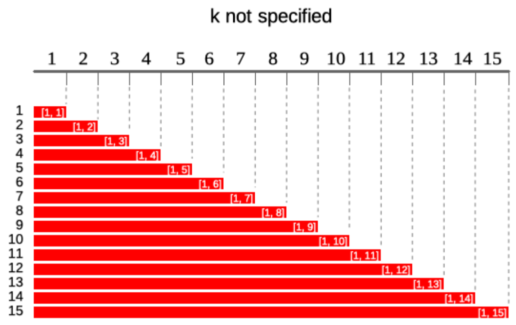
x	(vector, data.frame, matrix) Input in runner custom function f.
f	(function) Applied on windows created from x. This function is meant to summarize windows and create single element for each window, but one can also specify function which return multiple elements (runner output will be a list). By default runner returns windows as is (f = function(x)).
k	(integer vector or single value) Denoting size of the running window. If k is a single value then window size is constant for all elements, otherwise if length(k) == length(x) different window size for each element. One can also specify k in the same way as by in seq.POSIXt. More in details.
lag	(integer vector or single value) Denoting window lag. If lag is a single value then window lag is constant for all elements, otherwise if length(lag) == length(x) different window size for each element. Negative value shifts window forward. One can also specify lag in the same way as by in seq.POSIXt. More in details.
idx	(integer, Date, POSIXt) Optional integer vector containing sorted (ascending) index of observation. By default idx is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then k and lag are depending on idx. Length of idx have to be equal of length x.
at	(integer, Date, POSIXt, character vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also POSIXt sequence increment seq.POSIXt. More in details.
na_pad	(logical single value) Whether incomplete window should return NA (if na_pad = TRUE) Incomplete window is when some parts of the window are out of range.
type	(character single value) output type ("auto", "logical", "numeric", "integer", "character"). runner by default guess type automatically. In case of failure of "auto" please specify desired type.

`...`	(optional) other arguments passed to the function `f`.

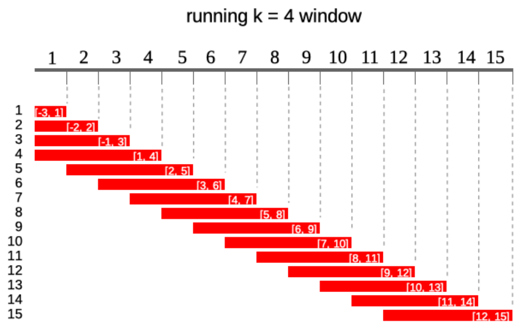
Details

Function can apply any R function on running windows defined by `x`, `k`, `lag`, `idx` and `at`. Running window can be calculated on several ways:

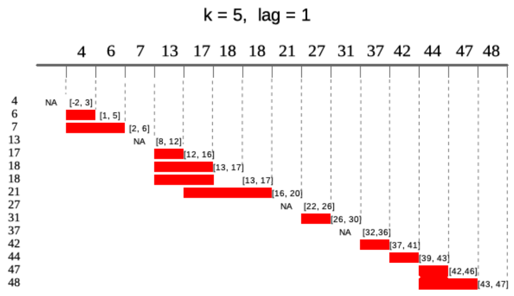
- **Cumulative windows**
  applied when user doesn't specify `k` argument or specify `k = length(x)`, this would mean that `k` is equal to number of available elements



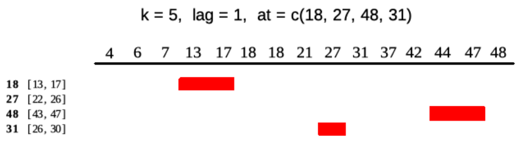
- **Constant sliding windows**
  applied when user specify `k` as constant value keeping `idx` and `at` unspecified. `lag` argument shifts windows left (`lag` > `0`) or right (`lag` < `0`).



- **Windows depending on date**
  If one specifies `idx` this would mean that output windows size might change in size because of unequally spaced indexes. Fox example 5-period window is different than 5-element window, because 5-period window might contain any number of observation (7-day mean is not the same as 7-element mean)



- **Window at specific indices**
  `runner` by default returns vector of the same size as `x` unless one specifies `at` argument. Each element of `at` is an index on which runner calculates function - which means that output of the runner is now of length equal to `at`. Note that one can change index of `x` by specifying `idx`. Illustration below shows output of `runner` for `at = c(18,27,45,31)` which gives windows in ranges enclosed in square brackets. Range for `at = 27` is [22, 26] which is not available in current indices.

  `at` can also be specified as interval of the output defined by `at = "<increment>"` which results in output on following indices `seq.POSIXt(min(idx),max(idx),by = "<increment>")`. Increment of sequence is the same as in `seq.POSIXt` function. It's worth noting that increment interval can't be more frequent than interval of `idx` - for `Date` the most frequent time-unit is a `"day"`, for `POSIXt` a `sec`.

  `k` and `lag` can also be specified as using time sequence increment. Available time units are "sec", "min", "hour", "day", "DSTday", "week", "month", "quarter" or "year". To increment by number of units one can also specify <number> <unit>s for example `lag = "-2 days"`, `k = "5 weeks"`.

Above is not enough since `k` and `lag` can be a vector which allows to stretch and lag/lead each window freely on in time (on indices).

Value

vector with aggregated values for each window. Length of output is the same as `length(x)` or `length(at)` if specified. Type of the output is taken from `type` argument.

Examples

```
# runner returns windows as is by default
runner(1:10)

# mean on k = 3 elements windows
runner(1:10, f = mean, k = 3)
```

```
# mean on k = 3 elements windows with different specification
runner(1:10, k = 3, f = function(x) mean(x, na.rm = TRUE))

# concatenate two columns
runner(
  data.frame(
    a = letters[1:10],
    b = 1:10
  ),
  f = function(x) paste(paste0(x$a, x$b), collapse = "+"),
  type = "character"
)

# concatenate two columns with additional argument
runner(
  data.frame(
    a = letters[1:10],
    b = 1:10
  ),
  f = function(x, xxx) {
    paste(paste0(x$a, xxx, x$b), collapse = " + ")
  },
  xxx = "...",
  type = "character"
)

# number of unique values in each window (varying window size)
runner(letters[1:10],
       k = c(1, 2, 2, 4, 5, 5, 5, 5, 5, 5),
       f = function(x) length(unique(x)))

# concatenate only on selected windows index
runner(letters[1:10],
       f = function(x) paste(x, collapse = "-"),
       at = c(1, 5, 8),
       type = "character")

# 5 days mean
idx <- c(4, 6, 7, 13, 17, 18, 18, 21, 27, 31, 37, 42, 44, 47, 48)
runner::runner(
  x = idx,
  k = "5 days",
  lag = 1,
  idx = Sys.Date() + idx,
  f = function(x) mean(x)
)

# 5 days mean at 4-indices
runner::runner(
  x = 1:15,
  k = 5,
  lag = 1,
  idx = idx,
  at = c(18, 27, 48, 31),
  f = mean
)
```

`run_by` *Set window parameters*

Description

Set window parameters for runner. This function sets the attributes to `x` (only `data.frame`) object and saves user effort to specify window parameters in further multiple runner calls.

Usage

```
run_by(x, idx, k, lag, na_pad, at)
```

Arguments

`x`	(`vector`, `data.frame`, `matrix`) Input in runner custom function `f`.
`idx`	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
`k`	(`integer` vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k) == length(x)` different window size for each element. One can also specify `k` in the same way as by in `seq.POSIXt`. More in details.
`lag`	(`integer` vector or single value) Denoting window lag. If `lag` is a single value then window lag is constant for all elements, otherwise if `length(lag) == length(x)` different window size for each element. Negative value shifts window forward. One can also specify `lag` in the same way as by in `seq.POSIXt`. More in details.
`na_pad`	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.
`at`	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also `POSIXt` sequence increment `seq.POSIXt`. More in details.

Value

x object which runner can be executed on.

Examples

```
library(dplyr)

data <- data.frame(
 index = c(2, 3, 3, 4, 5, 8, 10, 10, 13, 15),
 a = rep(c("a", "b"), each = 5),
 b = 1:10
)

data %>%
 group_by(a) %>%
 run_by(idx = "index", k = 5) %>%
 mutate(
   c = runner(
     x = .,
     f = function(x) {
       paste(x$b, collapse = ">")
     }
   ),
   d = runner(
     x = .,
     f = function(x) {
       sum(x$b)
     }
   )
 )
```

`seq_at` *Creates sequence for at as time-unit-interval*

Description

Creates sequence for at as time-unit-interval

Usage

```
seq_at(at, idx)
```

Arguments

`at`	object from runner
`idx`	object from runner

`streak_run` *Running streak length*

Description

Calculates running series of consecutive elements

Usage

```
streak_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_rm = TRUE,
  na_pad = FALSE
)
```

Arguments

`x`	any type vector which running function is calculated on
`k`	(`integer` vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k) == length(x)` different window size for each element. One can also specify `k` in the same way as by in `seq.POSIXt`. More in details.
`lag`	(`integer` vector or single value) Denoting window lag. If `lag` is a single value then window lag is constant for all elements, otherwise if `length(lag) == length(x)` different window size for each element. Negative value shifts window forward. One can also specify `lag` in the same way as by in `seq.POSIXt`. More in details.
`idx`	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
`at`	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also `POSIXt` sequence increment `seq.POSIXt`. More in details.
`na_rm`	`logical` single value (default `na_rm = TRUE`) - if `TRUE` sum is calculating excluding `NA`.
`na_pad`	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.

Value

streak [numeric] vector of length equals length of x containing number of consecutive occurrences.

Examples

```
set.seed(11)
x1 <- sample(c("a","b"), 15, replace = TRUE)
x2 <- sample(c(NA_character_, "a", "b"), 15, replace = TRUE)
k <- sample(1:4, 15, replace = TRUE)
streak_run(x1) # simple streak run
streak_run(x1, k = 2) # streak run within 2-element window
streak_run(x2, na_pad = TRUE, k = 3) # streak run within k=3 with padding NA
streak_run(x1, k = k) # streak run within varying window size specified by vector k
```

---

sum_run *Running sum*

Description

Running sum in specified window of numeric vector.

Usage

```
sum_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_rm = TRUE,
  na_pad = FALSE
)
```

Arguments

x	numeric vector which running function is calculated on
k	(integer' vector or single value) Denoting size of the running window. If k is a single value then window size is constant for all elements, otherwise if length(k) == length(x) different window size for each element.
lag	(integer vector or single value) Denoting window lag. If lag is a single value then window lag is constant for all elements, otherwise if length(lag) == length(x) different window size for each element. Negative value shifts window forward.

`idx`	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
`at`	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at.
`na_rm`	`logical` single value (default `na_rm = TRUE`) - if `TRUE` sum is calculating excluding `NA`.
`na_pad`	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.

Value

sum code vector of length equals length of `x`.

Examples

```
set.seed(11)
x1 <- rnorm(15)
x2 <- sample(c(rep(NA, 5),rnorm(15)), 15, replace = TRUE)
k <- sample(1:15, 15, replace = TRUE)
sum_run(x1)
sum_run(x2, na_rm = TRUE)
sum_run(x2, na_rm = FALSE)
sum_run(x2, na_rm = TRUE, k = 4)
```

`this_group` *Access group data in mutate*

Description

Access group data in `dplyr::mutate` after `dplyr::group_by`. Function created because data available in `dplyr::group_by %>% mutate` scheme is not filtered by group - in mutate function `.` is still initial dataset. This function creates `data.frame` using `dplyr::groups` information.

Usage

```
this_group(x)
```

Arguments

`x`	(`data.frame`) object which can be `grouped_df` in special case.

Value

data.frame filtered by current `dplyr::groups()`

`which_run` *Running which*

Description

`min_run` calculates running which - returns index of element where `x` == `TRUE`.

Usage

```
which_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  which = "last",
  na_rm = TRUE,
  na_pad = FALSE
)
```

Arguments

`x`	(`vector`, `data.frame`, `matrix`) Input in runner custom function `f`.
`k`	(`integer` vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k)` == `length(x)` different window size for each element. One can also specify `k` in the same way as by in `seq.POSIXt`. More in details.
`lag`	(`integer` vector or single value) Denoting window lag. If `lag` is a single value then window lag is constant for all elements, otherwise if `length(lag)` == `length(x)` different window size for each element. Negative value shifts window forward. One can also specify `lag` in the same way as by in `seq.POSIXt`. More in details.
`idx`	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
`at`	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also `POSIXt` sequence increment `seq.POSIXt`. More in details.

`which`	`character` value "first" or "last" denoting if the first or last `TRUE` index is returned from the window.
`na_rm`	`logical` single value (default `na_rm = TRUE`) - if `TRUE` sum is calculating excluding `NA`.
`na_pad`	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.

Value

integer vector of indexes of the same length as `x`.

Examples

```
set.seed(11)
x1 <- sample(c(1, 2, 3), 15, replace = TRUE)
x2 <- sample(c(NA, 1, 2, 3), 15, replace = TRUE)
k  <- sample(1:4, 15, replace = TRUE)
which_run(x1)
which_run(x2, na_rm = TRUE)
which_run(x2, na_rm = TRUE, k = 4)
which_run(x2, na_rm = FALSE, k = k)
```

`window_run` *List of running windows*

Description

Creates `list` of windows with given arguments settings. Length of output `list` is equal

Usage

```
window_run(
  x,
  k = integer(0),
  lag = integer(1),
  idx = integer(0),
  at = integer(0),
  na_pad = FALSE
)
```

Arguments

`x`	(`vector`, `data.frame`, `matrix`) Input in runner custom function `f`.

k	(`integer` vector or single value) Denoting size of the running window. If `k` is a single value then window size is constant for all elements, otherwise if `length(k) == length(x)` different window size for each element. One can also specify `k` in the same way as by in `seq.POSIXt`. More in details.
lag	(`integer` vector or single value) Denoting window lag. If `lag` is a single value then window lag is constant for all elements, otherwise if `length(lag) == length(x)` different window size for each element. Negative value shifts window forward. One can also specify `lag` in the same way as by in `seq.POSIXt`. More in details.
idx	(`integer`, `Date`, `POSIXt`) Optional integer vector containing sorted (ascending) index of observation. By default `idx` is index incremented by one. User can provide index with varying increment and with duplicated values. If specified then `k` and `lag` are depending on `idx`. Length of `idx` have to be equal of length `x`.
at	(`integer`, `Date`, `POSIXt`, `character` vector) Vector of any size and any value defining output data points. Values of the vector defines the indexes which data is computed at. Can be also `POSIXt` sequence increment `seq.POSIXt`. More in details.
na_pad	(`logical` single value) Whether incomplete window should return `NA` (if `na_pad = TRUE`) Incomplete window is when some parts of the window are out of range.

Value

list of vectors (windows). Length of list is the same as `length(x)` or `length(at)` if specified, and length of each window is defined by `k` (unless window is out of range).

Examples

```
window_run(1:10, k = 3, lag = -1)
window_run(letters[1:10], k = c(1, 2, 2, 4, 5, 5, 5, 5, 5, 5))
```

Index