

# Package ‘statar’

July 6, 2017

**Title** Tools Inspired by 'Stata' to Manipulate Tabular Data

**Version** 0.6.5

**Description** A set of tools inspired by 'Stata' to explore data.frames ('summarize', 'tabulate', 'xtile', 'pctile', 'binscatter', elapsed quarters/month, lead/lag).

**License** GPL-2

**URL** <https://github.com/matthieugomez/statar>

**BugReports** <https://github.com/matthieugomez/statar/issues>

**Depends** R (>= 3.2.0)

**Imports** data.table, dplyr (>= 0.6.0), ggplot2 (>= 2.0.0), lazyeval, matrixStats, methods, rlang, stringr, tidyr

**Suggests** knitr, lubridate, rmarkdown, testthat

**VignetteBuilder** knitr

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Matthieu Gomez [aut, cre]

**Maintainer** Matthieu Gomez <mattg@princeton.edu>

**Repository** CRAN

**Date/Publication** 2017-07-06 16:35:15 UTC

## R topics documented:

elapsed	2
fill_gap	3
is.panel	4
join	5
n_narm	6
pctile	7
statar	7
stat_binmean	7
sum_up	9

tab . . . . .	9
tempname . . . . .	10
tlead-flag . . . . .	11
winsorize . . . . .	12
xtile . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

elapsed	<i>Elapsed dates (monthly, quarterly)</i>
---------	---

---

### Description

Elapsed dates (monthly, quarterly)

### Usage

as.quarterly(x)

is.quarterly(x)

as.monthly(x)

is.monthly(x)

### Arguments

x                    a vector

### Details

Monthly and quarterly dates are stored as integers, representing the number of elapsed calendar periods since 01/01/1970. As `yearmonth` and `yearqtr` the package `zoo`, these dates are printed in a way that fits their frequency (YYYqq, YYYMMM). The only difference is that, `monthly`, and `quarterly` are integers, which removes issues due to floating points (particularly important when merging). This also allows to use arithmetic on periods, ie `date + 1` adds one period rather than one day.

Methods to convert from and to Dates or POSIXlt are provided. In particular, you may use `lubridate` [week month](#) and [year](#) to extract information from elapsed dates.

### Examples

```
library(lubridate)
library(dplyr)
date <- mdy(c("04/03/1992", "01/04/1992", "03/15/1992"))
datem <- as.monthly(date)
is.monthly(datem)
as.quarterly(date)
as.character(datem)
datem + 1
```

```
df <- data_frame(datem)
# filter(df, month(datem) == 1)
lag(c(1, 2, 3), n = 1, along_with = datem)
seq(datem[1], datem[2])
as.Date(datem)
as.POSIXlt(datem)
as.POSIXct(datem)
week(datem)
```

---

fill\_gap

---

*Add rows corresponding to gaps in some variable*


---

### Description

Add rows corresponding to gaps in some variable

### Usage

```
fill_gap(x, ..., full = FALSE, roll = FALSE, rollends = if (roll ==
  "nearest") c(TRUE, TRUE) else if (roll >= 0) c(FALSE, TRUE) else c(TRUE,
  FALSE))
```

### Arguments

x	A data frame
...	a time variable
full	A boolean. When full = FALSE (default) rows are filled with respect to min and max of ... within each group. When full = TRUE, rows are filled with respect to min and max of ... in the whole datasets.
roll	When roll is a positive number, this limits how far values are carried forward. roll=TRUE is equivalent to roll=+Inf. When roll is a negative number, values are rolled backwards; i.e., next observation carried backwards (NOCB). Use -Inf for unlimited roll back. When roll is "nearest", the nearest value is used.
rollends	A logical vector length 2 (a single logical is recycled). When rolling forward (e.g. roll=TRUE) if a value is past the last observation within each group defined by the join columns, rollends[2]=TRUE will roll the last value forwards. rollends[1]=TRUE will roll the first value backwards if the value is before it. If rollends=FALSE the value of i must fall in a gap in x but not after the end or before the beginning of the data, for that group defined by all but the last join column. When roll is a finite number, that limit is also applied when rolling the end

**Examples**

```
library(dplyr)
library(lubridate)
df <- data_frame(
  id    = c(1, 1, 1, 2),
  datem = as.monthly(mdy(c("04/03/1992", "01/04/1992", "03/15/1992", "05/11/1992"))),
  value = c(4.1, 4.5, 3.3, 3.2)
)
df %>% group_by(id) %>% fill_gap(datem)
df %>% group_by(id) %>% fill_gap(datem, full = TRUE)
df %>% group_by(id) %>% fill_gap(datem, roll = "nearest")
df %>% group_by(id) %>% fill_gap(datem, roll = "nearest", full = TRUE)
```

---

is.panel

*Check whether a data.frame is a panel*


---

**Description**

Check whether a data.frame is a panel

**Usage**

```
is.panel(x, ..., .dots)
```

**Arguments**

x	a data frame
...	a time variable
.dots	Used to work around non standard evaluation

**Value**

The function is.panel check that there are no duplicate combinations of the variables in ... and that no observation is missing for the last variable in ... (the time variable).

**Examples**

```
library(dplyr)
df <- data_frame(
  id1    = c(1, 1, 1, 2, 2),
  id2    = 1:5,
  year   = c(1991, 1993, NA, 1992, 1992),
  value  = c(4.1, 4.5, 3.3, 3.2, 5.2)
)
df %>% group_by(id1) %>% is.panel(year)
df1 <- df %>% filter(!is.na(year))
df1 %>% is.panel(year)
df1 %>% group_by(id1) %>% is.panel(year)
df1 %>% group_by(id1, id2) %>% is.panel(year)
```

---

join	<i>Join two data frames together</i>
------	--------------------------------------

---

### Description

Join two data frames together

### Usage

```
join(x, y, kind, on = intersect(names(x), names(y)), suffixes = c(".x",  
  ".y"), check = m ~ m, gen = FALSE, inplace = FALSE, update = FALSE,  
  type)
```

### Arguments

x	The master data.frame
y	The using data.frame
kind	The kind of (SQL) join among "full" (default), "left", "right", "inner", "semi", "anti" and "cross".
on	Character vectors specifying variables to match on. Default to common names between x and y.
suffixes	A character vector of length 2 specifying suffix of overlapping columns. Default to ".x" and ".y".
check	A formula checking for the presence of duplicates. Specifying 1~m (resp m~1, 1~1) checks that joined variables uniquely identify observations in x (resp y, both).
gen	Name of new variable to mark result, or the boolean FALSE (default) if no such variable should be created. The variable equals 1 for rows in master only, 2 for rows in using only, 3 for matched rows.
inplace	A boolean. In case "kind"= "left" and RHS of check is 1, the merge can be one in-place.
update	A boolean. For common variables in x and y not specified in "on", replace missing observations by the non missing observations in y.
type	Deprecated

### Value

A data.frame that joins rows in master and using datasets. Importantly, if x or y are not keyed, the join may change their row orders.

## Examples

```
library(dplyr)
x <- data.frame(a = rep(1:2, each = 3), b=1:6)
y <- data.frame(a = 0:1, bb = 10:11)
join(x, y, kind = "full")
join(x, y, kind = "left", gen = "_merge")
join(x, y, kind = "right", gen = "_merge")
join(x, y, kind = "inner", check = m~1)
join(x, y, kind = "semi")
join(x, y, kind = "anti")
y <- rename(y, b = bb)
join(x, y, kind = "full", on = "a")
join(x, y, kind = "full", on = "a", suffixes = c("", ".i"))
y <- data.frame(a = 0:1, bb = 10:11)
join(x, y, kind = "left", check = m~1)
x <- data.frame(a = c(1,2), b=c(NA, 2))
y <- data.frame(a = c(1,2), b = 10:11)
join(x, y, kind = "left", on = "a", update = TRUE)
join(x, y, kind = "left", on = "a", check = m~1, update = TRUE)
```

---

n\_narm

*Count number of non missing observations*

---

## Description

Count number of non missing observations

## Usage

```
n_narm(...)
```

## Arguments

... a sequence of vectors, matrices and data frames.

## Examples

```
n_narm(1:100, c(NA, 1:99))
```

---

pctile *Weighted quantile of type 2 (similar to Stata \_pctile)*

---

**Description**

Weighted quantile of type 2 (similar to Stata \_pctile)

**Usage**

```
pctile(x, probs = c(0.25, 0.5, 0.75), wt = NULL, na.rm = FALSE)
```

**Arguments**

x	A vector
probs	A vector of probabilities
wt	A weight vector
na.rm	Should missing values be returned?

---

statar *A package for applied research*

---

**Description**

A package for applied research

---

stat\_binmean *Plot the mean of y over the mean of x within bins of x.*

---

**Description**

Plot the mean of y over the mean of x within bins of x.

**Usage**

```
stat_binmean(mapping = NULL, data = NULL, geom = "point",
  position = "identity", show.legend = NA, inherit.aes = TRUE,
  na.rm = FALSE, n = 20, ...)
```

**Arguments**

mapping	Set of aesthetic mappings created by <a href="#">aes</a> or <a href="#">aes_</a> . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code> , the default, the data is inherited from the plot data as specified in the call to <a href="#">ggplot</a> . A <code>data.frame</code> , or other object, will override the plot data. All objects will be fortified to produce a data frame. See <a href="#">fortify</a> for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code> , and will be used as the layer data.
geom	The geometric object to use display the data
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes.
inherit.aes	If <code>FALSE</code> , overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. <a href="#">borders</a> .
na.rm	If <code>FALSE</code> (the default), removes missing values with a warning. If <code>TRUE</code> silently removes missing values.
n	number of x-bins. Default to 20. Set to zero if you want to use distinct value of x for grouping.
...	other arguments passed on to <a href="#">layer</a> . These are often aesthetics, used to set an aesthetic to a fixed value, like <code>color = "red"</code> or <code>size = 3</code> . They may also be parameters to the paired geom/stat.

**Value**

a `data.frame` with additional columns:

xtile	bins for x
x	mean of x
y	mean of y

**Examples**

```
library(ggplot2)
g <- ggplot(iris, aes(x = Sepal.Width , y = Sepal.Length))
g + stat_binmean(n = 10)
g + stat_binmean(n = 10) + stat_smooth(method = "lm", se = FALSE)
g + stat_binmean(n = 0)
g <- ggplot(iris, aes(x = Sepal.Width , y = Sepal.Length, color = Species))
g + stat_binmean(n = 10)
g + stat_binmean(n = 10) + stat_smooth(method = "lm", se = FALSE)
```



---

sum_up	<i>Gives summary statistics (corresponds to Stata command summarize)</i>
--------	--

---

**Description**

Gives summary statistics (corresponds to Stata command summarize)

**Usage**

```
sum_up(df, ..., d = FALSE, wt = NULL)
```

**Arguments**

df	a data.frame
...	Variables to include. Defaults to all non-grouping variables. See the <a href="#">select</a> documentation.
d	Should detailed summary statistics be printed?
wt	Weights. Default to NULL.

**Value**

a data.frame

**Examples**

```
library(dplyr)
N <- 100
df <- data_frame(
  id = 1:N,
  v1 = sample(5, N, TRUE),
  v2 = sample(1e6, N, TRUE)
)
sum_up(df)
sum_up(df, v2, d = TRUE)
sum_up(df, v2, wt = v1)
df %>% group_by(v1) %>% sum_up(starts_with("v"))
```

---

tab	<i>Returns cross tabulation</i>
-----	---------------------------------

---

**Description**

Returns cross tabulation

**Usage**

```
tab(x, ..., wt = NULL, na.rm = FALSE, sort = TRUE)
```

**Arguments**

x	a vector or a data.frame
...	Variable(s) to include. If length is two, a special cross tabulation table is printed although a long data.frame is always (invisibly) returned.
wt	Frequency weights. Default to NULL.
na.rm	Remove missing values. Default to FALSE
sort	Boolean. Default to TRUE

**Value**

a data.frame sorted by variables in ..., and with columns "Freq.", "Percent", and "Cum." for counts.

**Examples**

```
# setup
library(dplyr)
N <- 1e2 ; K = 10
df <- data_frame(
  id = sample(c(NA,1:5), N/K, TRUE),
  v1 = sample(1:5, N/K, TRUE)
)
# one-way tabulation
df %>% tab(id)
df %>% tab(id, wt = v1)
# two-way tabulation
df %>% tab(id, v1)
df %>% filter(id >= 3) %>% tab(id)
```

---

tempname

---

*Create unique names within a list, a data.frame, or an environment*


---

**Description**

Create unique names within a list, a data.frame, or an environment

**Usage**

```
tempname(where = globalenv(), n = 1, prefix = ".temp", inherits = TRUE)
```

**Arguments**

where	A character vector, list or an environment
n	An integer that specifies length of the output
prefix	A character vector that specifies prefix for new name
inherits	Should the name unique also in the enclosing frames of the environment?

**Examples**

```
tempname(c("temp1", "temp3"), 4)
tempname(globalenv())
tempname(data.frame(temp = 1), n = 3)
```

---

tlead-tlag	<i>lead and lag with respect to a time variable</i>
------------	---

---

**Description**

lead and lag with respect to a time variable

**Usage**

```
tlead(x, n = 1L, time, default = NA)
tlag(x, n = 1L, time, default = NA)
```

**Arguments**

x	a vector of values
n	a positive integer of length 1, giving the number of positions to lead or lag by. When the package lubridate is loaded, it can be a period when using with time (see the lubridate function minutes, hours, days, weeks, months and years)
time	time variable
default	value used for non-existent rows. Defaults to NA.

**Examples**

```
date <- c(1989, 1991, 1992)
value <- c(4.1, 4.5, 3.3)
tlag(value, 1, time = date) # returns value in year - 1
library(lubridate)
date <- as.monthly(mdy(c("01/04/1992", "03/15/1992", "04/03/1992")))
tlag(value, time = date)
library(dplyr)
df <- data_frame(
  id = c(1, 2, 2),
  date = date,
  value = value
)
df %>% group_by(id) %>% mutate(value1 = tlag(value, n = 1, time = date))
```

---

winsorize	<i>Winsorize a numeric vector</i>
-----------	-----------------------------------

---

**Description**

Winsorize a numeric vector

**Usage**

```
winsorize(x, probs = NULL, cutpoints = NULL, replace = c(cutpoints[1],
  cutpoints[2]), verbose = TRUE)
```

```
winsorise(x, probs = NULL, cutpoints = NULL, replace = c(cutpoints[1],
  cutpoints[2]), verbose = TRUE)
```

**Arguments**

x	A vector of values
probs	A vector of probabilities that can be used instead of cutpoints. Quantiles are computed as the inverse of the empirical distribution function (type = 1)
cutpoints	Cutpoints under and above which are defined outliers. Default is (median - five times interquartile range, median + five times interquartile range). Compared to bottom and top percentile, this takes into account the whole distribution of the vector.
replace	Values by which outliers are replaced. Default to cutpoints. A frequent alternative is NA.
verbose	Boolean. Should the percentage of replaced values printed?

**Examples**

```
v <- c(1:4, 99)
winsorize(v)
winsorize(v, replace = NA)
winsorize(v, probs = c(0.01, 0.99))
winsorize(v, cutpoints = c(1, 50))
```

---

xtile	<i>Bin variable in groups (similar to Stata xtile)</i>
-------	--

---

**Description**

Bin variable in groups (similar to Stata xtile)

**Usage**

```
xtile(x, n = NULL, probs = NULL, cutpoints = NULL, wt = NULL)
```

**Arguments**

x	A vector
n	A numeric specifying number of quantiles. Can be used instead of cutpoints
probs	A vector of probabilities that can be used instead of cutpoints. Quantiles are computed as the inverse of the empirical distribution function (type = 1)
cutpoints	Cutpoints to use when nq is not specified. For instance cutpoints = 0.4 creates two groups, one for observations equal or below 0.4, one for observations superior to 0.4.
wt	A variable specifying weight in case the option n_quantiles is specified.

**Value**

An integer vector representing groups corresponding to cutpoints. Includes missing values when present in the original vector.

**Examples**

```
x <- c(NA, 1:10)
xtile(x, n = 3) # 3 groups based on terciles
xtile(x, probs = c(0.3, 0.7)) # 3 groups based on two quantiles
xtile(x, cutpoints = c(2, 3)) # 3 groups based on two cutpoints
```

# Index

aes, 8  
aes\_, 8  
as.monthly (elapsed), 2  
as.quarterly (elapsed), 2  
  
borders, 8  
  
elapsed, 2  
  
fill\_gap, 3  
fortify, 8  
  
ggplot, 8  
  
is.monthly (elapsed), 2  
is.panel, 4  
is.quarterly (elapsed), 2  
  
join, 5  
  
layer, 8  
  
month, 2  
monthly (elapsed), 2  
  
n\_narm, 6  
  
pctile, 7  
  
quarterly, (elapsed), 2  
  
select, 9  
stat\_binmean, 7  
statar, 7  
statar-package (statar), 7  
sum\_up, 9  
  
tab, 9  
tempname, 10  
tlag (tlead-tlag), 11  
tlead (tlead-tlag), 11  
tlead-tlag, 11  
  
week, 2  
winsorise (winsorize), 12  
winsorize, 12  
  
xtile, 12  
  
year, 2