Package: fdth (via r-universe)

November 6, 2026

Version 1.5-3

Date 2026-05-28

Title Frequency Distribution Tables, Histograms and Polygons

Depends R (>= 3.5.0)

Imports xtable

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Description Perform frequency distribution tables, associated histograms and polygons from vector, data.frame and matrix objects for numerical and categorical variables.

License GPL (>= 2)

URL https://github.com/jcfaria/fdth

Encoding UTF-8

NeedsCompilation no

Repository https://jcfaria.r-universe.dev

Date/Publication 2026-10-06 07:59:07 UTC

RemoteUrl https://github.com/jcfaria/fdth

RemoteRef HEAD

RemoteSha cbb2a03aa44445d22b289c0e464ca353a22c76ab

Contents

fdth-package 2
amplitude 10
fdt 11
fdt_cat 15
make.fdt 18
mean.fdt 20
median.fdt 22
mfv 23

plot.fdt 26
print.fdt 35
quantile.fdt 39
sd 43
summary.fdt 44
ta 49
var 50
xtable.fdt 51

Index **57**

`fdth-package` *Frequency distribution tables, histograms and polygons*

Description

The **fdth** package contains a set of functions that allow users to create frequency distribution tables (‘`fdt`’) and their associated histograms and frequency polygons (absolute, relative and cumulative). The ‘`fdt`’ can be formatted in many ways suitable for publication (papers, books, etc). The S3 `plot` method produces histograms with the convenience and flexibility of a high-level function.

Details

The frequency of a particular observation is the number of times the observation occurs in the data. The distribution of a variable is the pattern of frequencies of the observation.

Frequency distribution tables (‘`fdt`’) can be used for ordinal, continuous, and categorical variables.

The `R` environment provides a set of functions (generally low level) enabling the user to perform a ‘`fdt`’ and the associated graphical representation, the histogram. A ‘`fdt`’ plays an important role to summarize data information and is the basis for the estimation of probability density function used in parametrical inference.

However, for novice or occasional users of `R`, it can be laborious to find out all necessary functions and graphical parameters to do a normalized and clear ‘`fdt`’ tables and associated histograms ready for publication.

That is the aim of this package, i.e., to allow users to create both ‘`fdt`’ tables and histograms easily and flexibly. The most common input for univariate data is a `vector`. For multivariate data, both a `data.frame` in this case also allowing grouping all numerical variables according to one categorical, or `matrices`.

The simplest way to run ‘`fdt`’ and ‘`fdt_cat`’ is by supplying only the ‘`x`’ object, for example: `d <- fdt(x)`. In this case all necessary default values (‘`breaks`’ and ‘`right`’) ("Sturges" and `FALSE` respectively) will be used, if the ‘`x`’ object is categorical then just use `d <- fdt_cat(x)`.

If the variable is continuous, you can also supply:

- ‘`x`’ and ‘`k`’ (number of class intervals);
- ‘`x`’, ‘`start`’ (left endpoint of the first class interval) and ‘`end`’ (right endpoint of the last class interval); or
- ‘`x`’, ‘`start`’, ‘`end`’ and ‘`h`’ (class interval width).

These options make the 'fdt' very easy and flexible.

The 'fdt' and 'fdt_cat' object store information to be used by methods summary, print and plot. The result of plot is a histogram or polygon (absolute, relative or cumulative). The methods summary, print and plot provide a reasonable set of parameters to format and plot the 'fdt' object in a pretty (and publishable) way.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

hist provided by **graphics** and table, cut both provided by **base**.

Examples

```
library (fdth)

# Numerical
#=====================
# Vectors: univariate
#=====================
x <- rnorm(n = 1e3,
           mean = 5,
           sd = 1)

(ft <- fdt(x))

# Histograms
plot(ft)  # Absolute frequency histogram

plot(ft,
     main = 'My title')

plot(ft,
     x.round = 3,
     col = 'darkgreen')

plot(ft,
     xlas = 2)

plot(ft,
     x.round = 3,
     xlas = 2,
     xlab = NULL)

plot(ft,
     v = TRUE,
     cex = .8,
     x.round = 3,
```

```
     xlas = 2,
     xlab = NULL,
     col = rainbow(11))

plot(ft,
     type = 'fh')    # Absolute frequency histogram

plot(ft,
     type = 'rfh')   # Relative frequency histogram

plot(ft,
     type = 'rfph')  # Relative frequency (%) histogram

plot(ft,
     type = 'cdh')   # Cumulative density histogram

plot(ft,
     type = 'cfh')   # Cumulative frequency histogram

plot(ft,
     type = 'cfph')  # Cumulative frequency (%) histogram

# Polygons
plot(ft,
     type = 'fp')    # Absolute frequency polygon

plot(ft,
     type = 'rfp')   # Relative frequency polygon

plot(ft,
     type = 'rfpp')  # Relative frequency (%) polygon

plot(ft,
     type = 'cdp')   # Cumulative density polygon

plot(ft,
     type = 'cfp')   # Cumulative frequency polygon

plot(ft,
     type = 'cfpp')  # Cumulative frequency (%) polygon

# Density
plot(ft,
     type = 'd')     # Density

# Summary
ft

summary(ft)  # same result

print(ft)    # same result

show(ft)     # same result
```

```
summary(ft,
        format = TRUE)      # This may not be what you want for publication.

summary(ft,
        format = TRUE,
        pattern = '%.2f')   # Better, but can it be improved?

summary(ft,
        col = c(1:2, 4, 6),
        format = TRUE,
        pattern = '%.2f')   # Yes, it can!

range(x)                 # To inspect the range of x

summary(fdt(x,
            start = 1,
            end = 9,
            h = 1),
        col = c(1:2, 4, 6),
        format = TRUE,
        pattern = '%d')     # Is it nice now?

# The fdt.object
ft[['table']]                       # Stores the frequency distribution table (fdt)
ft[['breaks']]                      # Stores the breaks of fdt
ft[['breaks']]['start']             # Stores the left value of the first class
ft[['breaks']]['end']               # Stores the right value of the last class
ft[['breaks']]['h']                 # Stores the class interval
as.logical(ft[['breaks']]['right'])  # Stores the right option

# Theoretical curve and fdt
y <- rnorm(1e5,
           mean = 5,
           sd = 1)

ft <- fdt(y,
          k = 100)

plot(ft,
     type = 'd',                      # density
     col = heat.colors(100))

curve(dnorm(x,
            mean = 5,
            sd = 1),
      n = 1e3,
      add = TRUE,
      lwd = 3,
      col = 'dark blue')

#=======================================================
# Data.frames: multivariate with categorical variables
```

```
#=====================================================
mdf <- data.frame(X1 = rep(LETTERS[1:4], 25),
                  X2 = as.factor(rep(1:10, 10)),
                  Y1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  Y2 = rnorm(100,
                             60,
                             4),
                  Y3 = rnorm(100,
                             50,
                             4),
                  Y4 = rnorm(100,
                             40,
                             4),
                  stringsAsFactors = TRUE)

#(ft <- fdt(mdf)) # Error message due to presence of NA values

(ft <- fdt(mdf,
           na.rm = TRUE))

# Histograms
plot(ft,
     v = TRUE)

plot(ft,
     col = rainbow(8))

plot(ft,
     type = 'fh')

plot(ft,
     type = 'rfh')

plot(ft,
     type = 'rfph')

plot(ft,
     type = 'cdh')

plot(ft,
     type = 'cfh')

plot(ft,
     type = 'cfph')

# Polygons
plot(ft,
```

```
     v = TRUE,
     type = 'fp')

plot(ft,
     type = 'rfp')

plot(ft,
     type = 'rfpp')

plot(ft,
     type = 'cdp')

plot(ft,
     type = 'cfp')

plot(ft,
     type = 'cfpp')

# Density
plot(ft,
     type = 'd')

# Summary
ft

summary(ft)  # same result

print(ft)    # same result

show(ft)     # same result

summary(ft,
        format = TRUE)

summary(ft,
        format = TRUE,
        pattern = '%05.2f')  # regular expression

summary(ft,
        col = c(1:2, 4, 6),
        format = TRUE,
        pattern = '%05.2f')

print(ft,
      col = c(1:2, 4, 6))

print(ft,
      col = c(1:2, 4, 6),
      format = TRUE,
      pattern = '%05.2f')

# Using by
levels(mdf$X1)
```

```
plot(fdt(mdf,
         k = 5,
         by = 'X1',
         na.rm = TRUE),
     col = rainbow(5))

levels(mdf$X2)

summary(fdt(iris,
            k = 5),
        format = TRUE,
        patter = '%04.2f')

plot(fdt(iris,
         k = 5),
     col = rainbow(5))

levels(iris$Species)

summary(fdt(iris,
            k = 5,
            by = 'Species'),
        format = TRUE,
        patter = '%04.2f')

plot(fdt(iris,
         k = 5,
         by = 'Species'),
     v = TRUE)

#=======================
# Matrices: multivariate
#=======================
summary(fdt(state.x77),
        col = c(1:2,
                4,
                6),
        format = TRUE)

plot(fdt(state.x77))

# Very big
summary(fdt(volcano,
            right = TRUE),
        col = c(1:2,
                  4,
                  6),
        round = 3,
        format = TRUE,
        pattern = '%05.1f')

plot(fdt(volcano,
```

```
         right = TRUE))

## Categorical
x <- sample(x = letters[1:5],
            size = 5e2,
            rep = TRUE)

(fdt.c <- fdt_cat(x))

(fdt.c <- fdt_cat(x,
                  sort = FALSE))

#=========================================================
# Data.frame: multivariate with two categorical variables
#=========================================================
mdf <- data.frame(c1 = sample(LETTERS[1:3],
                              1e2,
                              rep = TRUE),
                  c2 = as.factor(sample(1:10,
                                        1e2,
                                        rep = TRUE)),
                  n1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  n2 = rnorm(100,
                             60,
                             4),
                  n3 = rnorm(100,
                             50,
                             4),
                  stringsAsFactors = TRUE)

str(mdf)

(fdt.c <- fdt_cat(mdf))

(fdt.c <- fdt_cat(mdf,
                  dec = FALSE))

(fdt.c <- fdt_cat(mdf,
                  sort = FALSE))

(fdt.c <- fdt_cat(mdf,
                  by = 'c1'))

#=========================
# Matrix: two categorical
#=========================
x <- matrix(sample(x = letters[1:10],
```

```
                size = 100,
                rep = TRUE),
            nc = 2,
            dimnames = list(NULL,
                            c('c1', 'c2')))

head(x)

(fdt.c <- fdt_cat(x))
```

`amplitude` *Total range from frequency distribution table classes*

Description

S3 methods for the total range (amplitude total) of an object. For `fdt`, it is computed as `end - start` from class limits.

Usage

```
## S3 generic
amplitude(x, ...)

## S3 methods: numerical
## Default S3 method:
amplitude(x, ...)

## S3 method for class 'fdt'
amplitude(x, ...)

## S3 method for class 'fdt.multiple'
amplitude(x, ...)
```

Arguments

`x`	a numeric vector or a `fdt` (simple or multiple) object.
`...`	additional arguments passed to methods.

Details

`amplitude.default` computes `max(x) - min(x)`. `amplitude.fdt` computes `breaks["end"] - breaks["start"]`. `amplitude.fdt.multiple` applies `amplitude.fdt` to each variable.

Value

`amplitude.default` and `amplitude.fdt` return a numeric value. `amplitude.fdt.multiple` returns a list of numeric values, one per variable.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

`var.fdt`, `sd.fdt`, `fdt`.

Examples

```
library(fdth)

x <- rnorm(1e3,
           mean = 20,
           sd = 2)

# From a numeric vector
amplitude(x)

# From an fdt object
amplitude(fdt(x))

# From a data.frame (fdt.multiple)
amplitude(fdt(iris[, 1:4]))
```

`fdt` *Frequency distribution table for numerical data*

Description

An S3 set of methods to easily create frequency distribution tables ('`fdt`') from `vector`, `data.frame` and `matrix` objects.

Usage

```
## S3 generic
fdt(x, ...)

## S3 methods
## Default S3 method:
fdt(x,
    k,
    start,
    end,
    h,
    breaks = c('Sturges', 'Scott', 'FD'),
    right = FALSE,
```

```
    na.rm = FALSE, ...)

## S3 method for class 'data.frame'
fdt(x,
    k,
    by,
    breaks = c('Sturges', 'Scott', 'FD'),
    right = FALSE,
    na.rm = FALSE, ...)

## S3 method for class 'matrix'
fdt(x,
    k,
    breaks = c('Sturges', 'Scott', 'FD'),
    right = FALSE,
    na.rm = FALSE, ...)
```

Arguments

x	a vector, data.frame or matrix object. If 'x' is data.frame or matrix it must contain at least one numeric column.
k	number of class intervals.
start	left endpoint of the first class interval.
end	right endpoint of the last class interval.
h	class interval width.
by	categorical variable used for grouping each numeric variable, useful only on data.frame.
breaks	method used to determine the number of interval classes, c("Sturges", "Scott", "FD").
right	right endpoints open (default = FALSE).
na.rm	logical. Should missing values be removed? (default = FALSE).
...	potential further arguments (required by generic).

Details

The simplest way to run 'fdt' is by supplying only the 'x' object, for example: `nm <- fdt(x)`. In this case all necessary default values ('breaks' and 'right') ("Sturges" and FALSE respectively) will be used.

It can also be provided as:

- 'x' and 'k' (number of class intervals);
- 'x', 'start' (left endpoint of the first class interval) and 'end' (right endpoint of the last class interval); or
- 'x', 'start', 'end' and 'h' (class interval width).

These options make 'fdt' very easy and flexible.

The 'fdt' object stores information used by methods summary, print, plot, mean, median and mfv. The result of plot is a histogram. The methods summary, print and plot provide a reasonable set of parameters to format and plot the 'fdt' object in a clear (and publishable) way.

Value

For fdt the method fdt.default returns a list of class fdt.default with the slots:

\samp{table}	A data.frame storing the 'fdt';
\samp{breaks}	A vector of length 4 storing 'start', 'end', 'h' and 'right' of the 'fdt' generated by this method.

The methods fdt.data.frame and fdt.matrix return a list of class fdt.multiple. This list has one slot for each numeric (fdt) variable of the 'x' provided. Each slot, corresponding to each numeric variable, stores the same slots of the fdt.default described above.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

hist provided by **graphics** and table, cut both provided by **base**.

Examples

```
library(fdth)

#========
# Vector
#========
x <- rnorm(n = 1e3,
           mean = 5,
           sd = 1)

# x
(ft <- fdt(x))

# x, alternative breaks
(ft <- fdt(x,
           breaks = 'Scott'))

# x, k
(ft <- fdt(x,
           k = 10))

# x, star, end
range(x)
```

```
(ft <- fdt(x,
           start = floor(min(x)),
           end = floor(max(x) + 1)))

# x, start, end, h
(ft <- fdt(x,
           start = floor(min(x)),
           end = floor(max(x) + 1),
           h = 1))

# Effect of right
sort(x <- rep(1:3, 3))

(ft <- fdt(x,
           start = 1,
           end = 4,
           h = 1))

(ft <- fdt(x,
           start = 0,
           end = 3,
           h = 1,
           right = TRUE))

#=========================================================
# Data.frame: multivariate with two categorical variables
#=========================================================
mdf <- data.frame(c1 = sample(LETTERS[1:3],
                              1e2,
                              TRUE),
                  c2 = as.factor(sample(1:10,
                                        1e2,
                                        TRUE)),
                  n1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  n2 = rnorm(100,
                             60,
                             4),
                  n3 = rnorm(100,
                             50,
                             4),
                  stringsAsFactors = TRUE)

str(mdf)

#(ft <- fdt(mdf))  # Error message due to presence of NA values
```

```
(ft <- fdt(mdf,
           na.rm = TRUE))

# By factor
(ft <- fdt(mdf,
           k = 5,
           by = 'c1',
           na.rm = TRUE))

# choose FD criteria
(ft <- fdt(mdf,
           breaks = 'FD',
           by = 'c1',
           na.rm = TRUE))

# k
(ft <- fdt(mdf,
           k = 5,
           by = 'c2',
           na.rm = TRUE))

(ft <- fdt(iris[c(1:2, 5)],
           k = 10))

(ft <- fdt(iris[c(1:2, 5)],
           k = 5,
           by = 'Species'))

#========================
# Matrices: multivariate
#========================
(ft <-fdt(state.x77))

summary(ft,
        format = TRUE)

summary(ft,
        format = TRUE,
        pattern = '%.2f')
```

`fdt_cat` *Frequency distribution table for categorical data*

Description

An S3 set of methods to easily create categorical frequency distribution tables ('`fdt_cat`') from `vector`, `data.frame` and `matrix` objects.

Usage

```
## S3 generic
fdt_cat(x, ...)

## S3 methods
## Default S3 method:
fdt_cat(x,
        sort = TRUE,
        decreasing = TRUE, ...)

## S3 method for class 'data.frame'
fdt_cat(x,
        by,
        sort = TRUE,
        decreasing = TRUE, ...)

## S3 method for class 'matrix'
fdt_cat(x,
        sort = TRUE,
        decreasing = TRUE, ...)
```

Arguments

x	a vector, data.frame or matrix object. If 'x' is data.frame or matrix it must contain at least one character/factor column.
by	categorical variable used for grouping each categorical response, useful only on data.frame.
sort	logical. Should the fdt_cat be sorted by the absolute frequency into ascending or descending order? (default = TRUE).
decreasing	logical. Should the sort order be increasing or decreasing? (default = TRUE).
...	optional further arguments (required by generic).

Details

The simplest way to run 'fdt_cat' is supplying only the 'x' object, for example: ct <- fdt_cat(x). In this case all necessary default values ('sort = TRUE' and 'decreasing = TRUE') will be used.

These options make the 'fdt_cat' very easy and flexible.

The 'fdt_cat' object stores information to be used by methods summary, print, plot and mfv. The result of plot is a bar plot. The methods summary.fdt_cat, print.fdt_cat and plot.fdt_cat provide a reasonable set of parameters to format and plot the 'fdt_cat' object in a clear (and publishable) way.

Value

For fdt_cat the method fdt_cat.default returns a data.frame storing the 'fdt'.

The methods `fdt_cat.data.frame` and `fdt_cat.matrix` return a list of class `fdt_cat..multiple`. This `list` has one slot for each categorical variable of the supplied 'x'. Each slot, corresponding to each categorical variable, stores the same slots of the `fdt_cat.default` described above.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

`hist` provided by **graphics** and `table`, `cut` both provided by **base**.

Examples

```
library(fdth)

# Categorical
x <- sample(x = letters[1:5],
            size = 5e2,
            rep = TRUE)

table(x)
sum(table(x))

(ft.c <- fdt_cat(x))

(ft.c <- fdt_cat(x,
                 sort = FALSE))

#=========================================================
# Data.frame: multivariate with two categorical variables
#=========================================================
mdf <- data.frame(c1 = sample(LETTERS[1:3],
                              1e2,
                              rep = TRUE),
                  c2 = as.factor(sample(1:10,
                                        1e2,
                                        rep = TRUE)),
                  n1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  n2 = rnorm(100,
                             60,
                             4),
                  n3 = rnorm(100,
                             50,
```

```
                                4),
                  stringsAsFactors = TRUE)

str(mdf)

(ft.c <- fdt_cat(mdf))

(ft.c <- fdt_cat(mdf,
                 dec = FALSE))

(ft.c <- fdt_cat(mdf,
                 sort = FALSE))

(ft.c <- fdt_cat(mdf,
                 by = 'c1'))

#==================================
# Matrix: two categorical variables
#==================================
x <- matrix(sample(x = letters[1:10],
                   size = 100,
                   rep = TRUE),
            nc = 2,
            dimnames = list(NULL,
                            c('c1', 'c2')))

head(x)

(ft.c <- fdt_cat(x))
```

---

make.fdt *Frequency distribution table for continuous and categorical variables*

Description

Creates a complete fdt from a minimal set of information.
Useful to reproduce a previous fdt when the original data vector is not known.

Usage

```
make.fdt(f,
         start,
         end,
         right = FALSE)

make.fdt_cat(f,
             categories = NULL,
             sort = TRUE,
             decreasing = TRUE)
```

Arguments

f	a numeric vector object of frequencies.
start	the left value of the interval of the first class.
end	the last value of the interval of the last class.
right	intervals open on the right (default = FALSE).
categories	the levels of the categorical variable.
sort	the levels of the categorical variable will be ordered by frequency. The default is TRUE.
decreasing	if sort argument is TRUE, the order will be decreasing by default.

Details

Given the starting and ending values of the continuous-variable table or the levels of the categorical variable plus the number of intervals and the absolute frequency values the functions make.fdt and make.fdt_cat reconstruct complete fdt or fdt_cat tables.

Value

The function make.fdt returns a list with the slots:

table	a data.frame storing the 'fdt'.
breaks	a vector of length 4 storing 'start', 'end', 'h' and 'right' of the 'fdt' generated by this method.

The function make.fdt_cat returns a list with the slots:

Category	the levels of the categorical variable.
f	absolute frequency, numeric
rf	relative frequency, numeric
rf(%)	relative frequency in percentages, numeric
cf	cumulative frequency; numeric
cf(%)	cumulative frequency in percentages, numeric

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

table and cut provided by **base** package.

Examples

```
library(fdth)

# Numeric
# Making one reference fdt
set.seed(33)
x <- rnorm(1e3,
           20,
           2)

(ft.r <- fdt(x))

# Making a brand new
(ft.n  <- make.fdt(f = ft.r$table$f,
                   start = 13.711,
                   end = 27.229))   # Good, but can it be improved?

summary(ft.n,
        format = TRUE,
        pattern = '%.3f')           # Is it nice now?


# Categorical
x <- sample(letters[1:5],
            1e3,
            rep = TRUE)

# Making one reference fdt
(ft.r <- fdt_cat(x))

# Making a brand new
(ft.n <- make.fdt_cat(f = ft.r$f,
                      categ = ft.r$Category))
```

`mean.fdt` *Mean of frequency distribution table (numerical variable)*

Description

S3 method for the arithmetic mean of a `fdt`.
Useful to estimate the arithmetic mean (when the real data vector is not known) from a previous `fdt`.

Usage

```
## S3 method: numerical
## S3 method for class 'fdt'
mean(x, ...)
```

Arguments

x	a fdt (simple or multiple) object.
...	required by generic.

Details

mean.fdt calculates the mean value based on a known formula using the midpoint of each interval class. mean.fdt.multiple calls mean.fdt for each variable, that is, each column of the data.frame.

Value

mean.fdt returns a numeric vector containing the mean value of the fdt. mean.fdt.multiple returns a list, where each element is a numeric vector containing the mean value of the fdt for each variable.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

median.fdt, mfv.

Examples

```
library(fdth)

mdf <- data.frame(x = rnorm(1e3,
                            20,
                            2),
                  y = rnorm(1e3,
                            30,
                            3),
                  z = rnorm(1e3,
                            40,
                            4))

head(mdf)

# From a data.frame
apply(mdf,
      2,
      mean)

# From a fdt object
mean(fdt(mdf))
```

`median.fdt` *Median of frequency distribution table (numerical variable)*

Description

S3 method for the median of a `fdt`.
Useful to estimate the median (when the real data vector is not known) from a previous `fdt`.

Usage

```
## S3 method: numerical
## S3 method for class 'fdt'
median(x, ...)
```

Arguments

`x`	a `fdt` (simple or multiple) object.
`...`	required by generic.

Details

`median.fdt` calculates the median value based on a known formula. `median.fdt.multiple` calls `median.fdt` for each variable, that is, each column of the data.frame.

Value

`median.fdt` returns a numeric vector containing the value of the median of the `fdt`. `median.fdt.multiple` returns a list, where each element is a numeric vector containing the value of the median of the `fdt` for each variable.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

`mean.fdt`, `mfv`.

Examples

```
library(fdth)

mdf <- data.frame(x = rnorm(1e3,
                            20,
                            2),
                  y = rnorm(1e3,
```

```
                            30,
                            3),
                  z = rnorm(1e3,
                            40,
                            4))

head(mdf)

# From a data.frame
apply(mdf,
      2,
      median)

# From a fdt object
median(fdt(mdf))
```

`mfv` *Most frequent value (statistical mode) of frequency distribution table (numerical and categorical variable)*

Description

S3 methods for the most frequent value (statistical mode) of a `fdt`.
Useful to estimate the most frequent value (statistical mode). It may also be used with a previous `fdt` when the original data vector is not known.

Usage

```
## S3 generic
mfv(x, ...)

## S3 methods: numerical and categorical
## Default S3 method:
mfv(x, ...)

## S3 method for class 'fdt'
mfv(x, ...)

## S3 method for class 'fdt.multiple'
mfv(x, ...)

## S3 method for class 'fdt_cat'
mfv(x, ...)

## S3 method for class 'fdt_cat.multiple'
mfv(x, ...)
```

Arguments

x	for mfv.default, a numeric or categorical vector; for the other methods, a fdt or fdt_cat (simple or multiple) object.
...	further arguments (required by the generic).

Details

mfv.fdt and mfv.fdt_cat calculate the most frequent value (mfv) based on a known formula. mfv.fdt.multiple and mfv.fdt_cat.multiple call mfv.fdt or mfv.fdt_cat, respectively, for each variable, that is, each column of the data.frame.

Value

mfv.default returns a vector containing the mfv value(s) of x. In multimodal cases, this vector has length greater than one. mfv.fdt returns a numeric vector containing the mfv value(s) of the fdt. In multimodal cases, this vector has length greater than one. mfv.fdt.multiple returns a list, where each element is a numeric vector containing the mfv value(s) of the fdt for each variable. mfv.fdt_cat returns a character vector containing the mfv value(s) of the fdt_cat. mfv.fdt_cat.multiple returns a list, where each element is a character vector containing the mfv value(s) of the fdt_cat for each variable.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

mean.fdt, median.fdt, quantile.fdt.

Examples

```
library(fdth)

## Numerical (multimodal examples)
vx <- c(rep(3,
            5),
        rep(5,
            5),
        sample(6:10,
               5))
vx
mfv(vx)      # Two modes: 3 and 5

tb <- fdt(vx)
tb
mfv(tb)      # Mode estimated from grouped data (two modal classes)

vy <- c(rep(3,
```

```
            5),
        sample(6:9,
               10,
               rep = TRUE),
        rep(10,
            5))
vy

tb2 <- fdt(vy,
           start = 3,
           end = 11,
           h = 1)
tb2
mfv(tb2)     # Two modes in non-adjacent classes

vz <- c(rep(1.2,
            6),
        rep(6.4,
            6),
        rep(3.3,
            2),
        rep(4.7,
            2))
vz

tb3 <- fdt(vz,
           start = 0,
           end = 8,
           h = 1)
tb3
mfv(tb3)     # Two modes in non-adjacent classes (deterministic example)


## Categorical
mdf <- data.frame(c1 = sample(letters[1:5],
                              1e3,
                              rep = TRUE),
                  c2 = sample(letters[6:10],
                              1e3,
                              rep = TRUE),
                  c3 = sample(letters[11:21],
                              1e3,
                              rep = TRUE),
                  stringsAsFactors = TRUE)
head(mdf)

mfv(mdf$c1)   # From vector c1
mfv(mdf$c2)   # From vector c2
mfv(mdf$c3)   # From vector c3

(ft <- fdt_cat(mdf))

mfv(ft)       # From grouped data in a fdt object
```

plot.fdt *Plot fdt.default and fdt.multiple objects*

Description

S3 methods for `fdt.default` and `fdt.multiple` objects. It is possible to plot histograms and polygons (absolute, relative and cumulative).

Usage

```
## S3 methods
## S3 method for class 'fdt.default'
plot(x,
     type = c('fh', 'fp',
              'rfh', 'rfp',
              'rfph', 'rfpp',
              'd',
              'cdh', 'cdp',
              'cfh', 'cfp',
              'cfph', 'cfpp'),

     v = FALSE,
     v.round = 2,
     v.pos = 3,
     xlab = "Class limits",
     xlas = 0,
     ylab = NULL,
     col = "gray",
     xlim = NULL,
     ylim = NULL,
     main = NULL,
     x.round = 2, ...)

## S3 method for class 'fdt.multiple'
plot(x,
     type = c('fh', 'fp',
              'rfh', 'rfp',
              'rfph', 'rfpp',
              'd',
              'cdh', 'cdp',
              'cfh', 'cfp',
              'cfph', 'cfpp'),
     v = FALSE,
     v.round = 2,
     v.pos = 3,
     xlab = "Class limits",
     xlas = 0,
```

```
     ylab = NULL,
     col = "gray",
     xlim = NULL,
     ylim = NULL,
     main = NULL,
     main.vars = TRUE,
     x.round = 2,
     grouped = FALSE,
     args.legend = NULL, ...)

## S3 method for class 'fdt_cat.default'
plot(x,
     type = c('fb', 'fp', 'fd',
              'rfb', 'rfp', 'rfd',
              'rfpb', 'rfpp', 'rfpd',
              'cfb', 'cfp', 'cfd',
              'cfpb', 'cfpp', 'cfpd',
              'pa'),
     v = FALSE,
     v.round = 2,
     v.pos = 3,
     xlab = NULL,
     xlas = 0,
     ylab = NULL,
     y2lab = NULL,
     y2cfp = seq(0,
                 100,
                 25),
     col = gray(.4),
     xlim = NULL,
     ylim = NULL,
     main = NULL,
     box = FALSE, ...)

## S3 method for class 'fdt_cat.multiple'
plot(x,
     type = c('fb', 'fp', 'fd',
              'rfb', 'rfp', 'rfd',
              'rfpb', 'rfpp', 'rfpd',
              'cfb', 'cfp', 'cfd',
              'cfpb', 'cfpp', 'cfpd',
              'pa'),
     v = FALSE,
     v.round = 2,
     v.pos = 3,
     xlab = NULL,
     xlas = 0,
     ylab = NULL,
```

```
y2lab = NULL,
y2cfp = seq(0,
            100,
            25),
col = gray(.4),
xlim = NULL,
ylim = NULL,
main = NULL,
main.vars = TRUE,
box = FALSE, ...)
```

Arguments

x	A 'fdt' object.
type	the type of the plot: 'fb:' absolute frequency barplot, 'fh:' absolute frequency histogram, 'fp:' absolute frequency polygon, 'fd:' absolute frequency dotchart, 'rfb:' relative frequency barplot, 'rfh:' relative frequency histogram, 'rfp:' relative frequency polygon, 'rfd:' relative frequency dotchart, 'rfpb:' relative frequency (%) barplot, 'rfph:' relative frequency (%) histogram, 'rfpp:' relative frequency (%) polygon, 'rfpd:' relative frequency (%) dotchart, 'd:' density, 'cdh:' cumulative density histogram, 'cdp:' cumulative density polygon, 'cfb:' cumulative frequency barplot, 'cfh:' cumulative frequency histogram, 'cfp:' cumulative frequency polygon, 'cfd:' cumulative frequency dotchart, 'cdpb:' cumulative frequency (%) barplot, 'cdph:' cumulative frequency (%) histogram, 'cfpp:' cumulative frequency (%) polygon, 'cfpd:' cumulative frequency (%) dotchart. 'pa:' pareto chart.
v	logical flag: should the values be added to the plot?
v.round	if v = TRUE, it rounds the values to the specified number of decimal places (default 0).

v.pos	if v = TRUE, a position specifier for the text. Values of 1, 2, 3 and 4, respectively indicate positions below, to the left of, above and to the right of the coordinates (default 3).
xlab	a label for the 'x' axis.
xlas	an integer which controls the orientation of the 'x' axis labels: '0:' parallel to the axes, '2:' perpendicular to the axes.
ylab	a label for the 'y' axis.
y2lab	a label for the 'y2' axis.
y2cfp	a cumulative percent frequency for the 'y2' axis. The default is seq(0, 100, 25).
col	a vector of colors.
xlim	the 'x' limits of the plot.
ylim	the 'y' limits of the plot.
main	title of the plot(s). This option has priority over 'main.vars', i.e., if any value is provided, the variable names will not be used as title of the plot(s). For fdt.multiple, the value should be a vector of characters, in this case, the R's recycling rule will be used.
main.vars	logical flag: should the variables names be added as title of each plot (default TRUE)?
x.round	a numeric value to round the 'x' ticks: '0:' parallel to the axes, '1:' horizontal, '2:' perpendicular to the axes, '3:' vertical.
box	if TRUE (the default), a box will be placed on the graphic. Only for categorical variables.
grouped	if TRUE, the lines of polygon types will be plotted on the same graphic. It works only if the by argument is used.
args.legend	list of additional arguments to be passed to legend; names of the list are used as argument names. Only used if grouped = TRUE. The default is NULL.
...	optional plotting parameters.

Details

The result is a single histogram or polygon (absolute, relative or cumulative) for fdt.default or a set of histograms or polygons (absolute, relative or cumulative) for fdt.multiple objects. Both 'default and multiple' try to compute the maximum number of histograms that will fit on one page, then it draws a matrix of histograms. More than one graphical device may be opened to show all histograms.

The result is a single bar plot, polygon, dot chart (absolute, relative or cumulative) and Pareto chart for fdt_cat.default or a set of the same graphs for fdt_cat.multiple objects. Both 'default and multiple' try to compute the maximum number of histograms that will fit on one page, then it draws a matrix of graphs listed above. More than one graphical device may be opened to show all graphs.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

Examples

```
library(fdth)

#==============================
# Vectors: univariate numerical
#==============================
x <- rnorm(n = 1e3,
           mean = 5,
           sd = 1)

(ft <- fdt(x))

# Histograms
plot(ft)  # Absolute frequency histogram

plot(ft,
     main = 'My title')

plot(ft,
     x.round = 3,
     col = 'darkgreen')

plot(ft,
     xlas = 2)

plot(ft,
     x.round = 3,
     xlas = 2,
     xlab = NULL)

plot(ft,
     v = TRUE,
     cex = .8,
     x.round = 3,
     xlas = 2,
     xlab = NULL,
     col = rainbow(11))

plot(ft,
     type = 'fh')    # Absolute frequency histogram

plot(ft,
     type = 'rfh')   # Relative frequency histogram

plot(ft,
     type = 'rfph')  # Relative frequency (%) histogram
```

```
plot(ft,
     type = 'cdh')   # Cumulative density histogram

plot(ft,
     type = 'cfh')   # Cumulative frequency histogram

plot(ft,
     type = 'cfph')  # Cumulative frequency (%) histogram

# Polygons
plot(ft,
     type = 'fp')    # Absolute frequency polygon

plot(ft,
     type = 'rfp')   # Relative frequency polygon

plot(ft,
     type = 'rfpp')  # Relative frequency (%) polygon

plot(ft,
     type = 'cdp')   # Cumulative density polygon

plot(ft,
     type = 'cfp')   # Cumulative frequency polygon

plot(ft,
     type = 'cfpp')  # Cumulative frequency (%) polygon

# Density
plot(ft,
     type = 'd')     # Density

# Theoretical curve and fdt
x <- rnorm(1e5,
           mean = 5,
           sd = 1)

plot(fdt(x,
         k = 100),
     type = 'd',
     col = heat.colors(100))

curve(dnorm(x,
            mean = 5,
            sd = 1),
      col = 'darkgreen',
      add = TRUE,
      lwd = 3)


#=================================
# Vectors: univariate categorical
```

```
#================================
x <- sample(letters[1:5],
            1e3,
            rep = TRUE)

(ft.c <- fdt_cat(x))

# Barplot: the default
plot(ft.c)

# Barplot
plot(ft.c,
     type = 'fb')

# Polygon
plot(ft.c,
     type = 'fp')

# Dotchart
plot(ft.c,
     type = 'fd')

# Pareto chart
plot(ft.c,
     type = 'pa')

#=======================================================
# Data.frames: multivariate with categorical variables
#=======================================================
mdf <- data.frame(X1 = rep(LETTERS[1:4], 25),
                  X2 = as.factor(rep(1:10, 10)),
                  Y1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  Y2 = rnorm(100,
                             60,
                             4),
                  Y3 = rnorm(100,
                             50,
                             4),
                  Y4 = rnorm(100,
                             40,
                             4),
                  stringsAsFactors = TRUE)

str(mdf)

# Histograms
(ft <- fdt(mdf,
```

```
            na.rm = TRUE))

plot(ft,
     v = TRUE,
     cex = .8)

plot(ft,
     col = 'darkgreen',
     ylim = c(0, 40))

plot(ft,
     col = rainbow(8),
     ylim = c(0, 40),
     main = LETTERS[1:4])

plot(ft,
     type = 'fh')

plot(ft,
     type = 'rfh')

plot(ft,
     type = 'rfph')

plot(ft,
     type = 'cdh')

plot(ft,
     type = 'cfh')

plot(ft,
     type = 'cfph')

# Polygons
plot(ft,
     v = TRUE,
     type = 'fp')

plot(ft,
     type = 'rfp')

plot(ft,
     type = 'rfpp')

plot(ft,
     type = 'cdp')

plot(ft,
     type = 'cfp')

plot(ft,
     type = 'cfpp')
```

```
# Density
plot(ft,
     type = 'd')

levels(mdf$X1)

plot(fdt(mdf,
         k = 5,
         by = 'X1',
         na.rm = TRUE),
     ylim = c(0, 12))

levels(mdf$X2)

plot(fdt(mdf,
         breaks = 'FD',
         by = 'X2',
         na.rm = TRUE))

plot(fdt(mdf,
         k = 5,
         by = 'X2',
         na.rm = TRUE))  # It is difficult to compare

plot(fdt(mdf,
         k = 5,
         by = 'X2',
         na.rm = TRUE),
     ylim = c(0, 8))  # Easy

plot(fdt(iris,
         k = 5))

plot(fdt(iris,
         k = 5),
     col = rainbow(5))

plot(fdt(iris,
         k = 5,
         by = 'Species'),
     v = TRUE)

ft <- fdt(iris,
          k = 10)

plot(ft)

plot(ft,
     type = 'd')

# Categorical data
(ft.c <- fdt_cat(mdf))
plot(ft.c)
```

```
plot(ft.c,
     type = 'fd',
     pch = 19)

#========================
# Matrices: multivariate
#========================
plot(fdt(state.x77))

plot(fdt(volcano))
```

---

print.fdt *Print methods for fdt objects*

Description

S3 methods to return a data.frame (the frequency distribution table - fdt) for fdt.default and fdt.multiple objects; data.frame (the frequency distribution table - fdt_cat) for fdt_cat.default and fdt_cat.multiple objects.

Usage

```
## S3 methods
## S3 method for class 'fdt.default'
print(x,
      columns = 1:6,
      round = 2,
      format.classes = FALSE,
      pattern = '%09.3e',
      row.names = FALSE,
      right = TRUE, ...)

## S3 method for class 'fdt.multiple'
print(x,
      columns = 1:6,
      round = 2,
      format.classes = FALSE,
      pattern = '%09.3e',
      row.names = FALSE,
      right = TRUE, ...)

## S3 method for class 'fdt_cat.default'
print(x,
      columns = 1:6,
      round = 2,
      row.names = FALSE,
      right = TRUE, ...)
```

```
## S3 method for class 'fdt_cat.multiple'
print(x,
      columns = 1:6,
      round = 2,
      row.names = FALSE,
      right = TRUE, ...)
```

Arguments

x	a 'fdt' object.
columns	a vector of integers to select columns of the data.frame table.
round	rounds 'fdt' columns to the specified number of decimal places (default 2).
format.classes	logical, if TRUE the first column of the data.frame table will be formatted using a regular expression. The default is "%09.3e".
pattern	same as fmt in sprintf.
row.names	logical (or character vector), indicating whether (or what) row names should be printed. The default is FALSE.
right	logical, indicating whether or not strings should be right-aligned. The default is right-alignment.
...	potential further arguments (required by generic).

Details

For print.fdt, it is possible to select which columns of the table (a data.frame) will be shown, as well as the pattern of the first column, for print.fdt_cat it is only possible to select which columns of the table (a data.frame) will be shown. The columns are:

1. 'Class limits'
2. 'f' - absolute frequency
3. 'rf' - relative frequency
4. 'rf(%)' - relative frequency, %
5. 'cf' - cumulative frequency
6. 'cf(%)' - cumulative frequency, %

The available parameters offer an easy and powerful way to format the 'fdt' for publications and other purposes.

Value

A single data.frame for fdt.default and fdt_cat.default, or multiple data.frames for fdt.multiple and fdt_cat.multiple.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

Examples

```
library (fdth)

#====================
# Vectors: univariate
#====================
set.seed(1)

x <- rnorm(n = 1e3,
           mean = 5,
           sd = 1)

ft <- fdt(x)

str(ft)

ft

print(ft) # same result

print(ft,
      format = TRUE)      # It may not be what you want for publication!

print(ft,
      format = TRUE,
      pattern = '%.2f')   # Better, but can it be improved?

print(ft,
      col = c(1:2, 4, 6),
      format = TRUE,
      pattern = '%.2f')   # Yes, it can!

range(x)                 # To inspect the range of x

print(fdt(x,
          start = 1,
          end = 9,
          h = 1),
      col = c(1:2, 4, 6),
      format = TRUE,
      pattern = '%d')     # Is it nice now?

ft[['table']]                       # Stores the frequency distribution table (fdt)
ft[['breaks']]                      # Stores the breaks of fdt
ft[['breaks']]['start']             # Stores the left value of the first class
ft[['breaks']]['end']               # Stores the right value of the last class
ft[['breaks']]['h']                 # Stores the class interval
as.logical(ft[['breaks']]['right'])  # Stores the right option

#=====================================================
# Data.frames: multivariate with categorical variables
#=====================================================
```

```
mdf <- data.frame(X1 = rep(LETTERS[1:4], 25),
                  X2 = as.factor(rep(1:10, 10)),
                  Y1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  Y2 = rnorm(100,
                             60,
                             4),
                  Y3 = rnorm(100,
                             50,
                             4),
                  Y4 = rnorm(100,
                             40,
                             4),
                  stringsAsFactors = TRUE)

(ft <- fdt_cat(mdf))

print(ft)

(ft <- fdt(mdf,
           na.rm = TRUE))

print(ft)

str(ft)

print(ft, # the s
      format = TRUE)

print(ft,
      format = TRUE,
      pattern = '%05.2f')  # regular expression

print(ft,
      col = c(1:2, 4, 6),
      format = TRUE,
      pattern = '%05.2f')

print(ft,
      col = c(1:2, 4, 6))

print(ft,
      col = c(1:2, 4, 6),
      format = TRUE,
      pattern = '%05.2f')

levels(mdf$X1)
```

```
print(fdt(mdf,
          k = 5,
          by = 'X1',
          na.rm = TRUE))

levels(mdf$X2)

print(fdt(mdf,
          breaks = 'FD',
          by = 'X2',
          na.rm = TRUE),
      round = 3)

print(fdt(mdf,
          k = 5,
          by = 'X2',
          na.rm = TRUE),
      format = TRUE,
      round = 3)

print(fdt(iris,
          k = 5),
      format = TRUE,
      patter = '%04.2f')

levels(iris$Species)

print(fdt(iris,
          k = 5,
          by = 'Species'),
      format = TRUE,
      patter = '%04.2f')

#========================
# Matrices: multivariate
#========================
print(fdt(state.x77),
      col = c(1:2, 4, 6),
      format = TRUE)

print(fdt(volcano,
          right = TRUE),
      col = c(1:2, 4, 6),
      round = 3,
      format = TRUE,
      pattern = '%05.1f')
```

---

quantile.fdt *Quantile of frequency distribution table (numerical variable)*

Description

S3 methods for the quantile of a fdt.
Useful to estimate the quantile (when the real data vector is not known) from a previous fdt.

Usage

```
## S3 methods: numerical
## S3 method for class 'fdt'
quantile(x,
         ...,
         i = 1,
         probs = seq(0, 1, 0.25))

## S3 method for class 'fdt.multiple'
quantile(x, ...)
```

Arguments

x	a fdt (simple or multiple) object.
i	an integer vector indicating which quantiles should be computed. Values must be in 1:(length(probs)-1).
probs	a finite numeric vector in [0, 1] defining the quantile scale. It must have at least two values and be sorted in non-decreasing order.
...	potential further arguments (required by generic).

Details

quantile.fdt calculates the quantiles based on a known formula for class intervals. quantile.fdt.multiple calls quantile.fdt for each variable, that is, each column of the data.frame.

Value

quantile.fdt returns a named numeric vector of class fdt.quantile containing the value(s) of the quantile(s) from fdt. Names are derived from the selected probability levels in probs (for example, 25%, 50%, 75%). quantile.fdt.multiple returns a list of class fdt.quantile.multiple, where each element is a fdt.quantile vector containing the quantile(s) of the fdt for each variable. Both classes have xtable methods for LaTeX export.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

median.fdt, var.fdt, mfv, xtable.fdt.

Examples

```
library(fdth)

x <- rnorm(n = 1e3,
           mean = 5,
           sd = 1)

(ft <- fdt(x))

# Quartile from vector
quantile(x)[2:4]

# Quartile from grouped data in a fdt object
quantile(ft,
         i = 1:3)

# Quartile from vector
quantile(x,
         probs = seq(from = 0,
                     to = 1,
                     by = .1))[2:10]

# Decile from grouped data in a fdt object
quantile(ft,
         i = 1:9,
         probs = seq(from = 0,
                     to = 1,
                     by = .1))

# Percentile from vector
quantile(x,
         probs = seq(from = 0,
                     to = 1,
                     by = .01))[2:100]

# Percentile from grouped data in a fdt object
quantile(ft,
         i = 1:99,
         probs = seq(from = 0,
                     to = 1,
                     by = .01))

# From a data.frame
mdf <- data.frame(x = rnorm(1e2,
                            20,
                            2),
                  y = rnorm(1e2,
                            30,
                            3),
                  z = rnorm(1e2,
                            40,
                            4))
```

```
head(mdf)

# From a data.frame (rows 2:4 are the 25%, 50%, and 75% quantiles)
apply(mdf,
      2,
      quantile)[2:4, ]

# From a fdt object
quantile(fdt(mdf),
         i = 1:3)

## A small (but didactic) joke
quantile(fdt(mdf),
         i = 2,
         probs = seq(0,
                     1,
                     0.25))        # The quartile 2
quantile(fdt(mdf),
         i = 5,
         probs = seq(0,
                     1,
                     0.10))        # The decile 5

quantile(fdt(mdf),
         i = 50,
         probs = seq(0,
                     1,
                     0.01))        # The percentile 50

quantile(fdt(mdf),
         i = 500,
         probs = seq(0,
                     1,
                     0.001))       # The permile 500

median(fdt(mdf))                 # The median (all results are the same) ;)

# More than one quantile
quantile(fdt(mdf$x),
         i = 1:3,
         probs = seq(0,
                     1,
                     0.25))  # The three quartiles

quantile(fdt(mdf$x),
         i = 1:9,
         probs = seq(0,
                     1,
                     0.10))  # The nine deciles

# Legacy approach (no longer necessary)
# ql <- numeric()
```

```
#
# for(i in 1:3)
#   ql[i] <- quantile(fdt(mdf$x),
#                     i=i,
#                     probs=seq(0,
#                               1,
#                               0.25))  # The three quartiles
#
# names(ql) <- paste0(c(25,
#                       50,
#                       75),
#                     '%')
# round(ql,
#       2)
```

---

sd *Standard deviation of frequency distribution table (numerical variable)*

Description

S3 methods for the standard deviation of a `fdt`.
Useful to estimate the standard deviation (when the real data vector is not known) from a previous `fdt`.

Usage

```
## S3 generic
sd(x, ...)

## S3 methods: numerical
## Default S3 method:
sd(x, ...)

## S3 method for class 'fdt'
sd(x, ...)

## S3 method for class 'fdt.multiple'
sd(x, ...)
```

Arguments

`x`	a `fdt` (simple or multiple) object.
`...`	required to be generic.

Details

`sd.fdt` calculates the value of the standard deviation based on a known formula. `sd.fdt.multiple` calls `sd.fdt` for each variable, that is, each column of the data.frame.

Value

sd.fdt returns a numeric vector containing the value of the standard deviation of the fdt. sd.fdt.multiple returns a list, where each element is a numeric vector containing the value of the standard deviation of the fdt for each variable.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

var.fdt, mean.fdt.

Examples

```
library(fdth)

mdf <- data.frame(x = rnorm(1e3,
                            20,
                            2),
                  y = rnorm(1e3,
                            30,
                            3),
                  z = rnorm(1e3,
                            40,
                            4))

head(mdf)

# From a data.frame
apply(mdf,
      2,
      sd)

# From a fdt object
sd(fdt(mdf))

unlist(sd(fdt(mdf)))
```

---

summary.fdt *Summary methods for fdt objects*

Description

S3 methods to return a data.frame (the frequency distribution table - 'fdt') for fdt.default, fdt.multiple, fdt_cat.default and fdt_cat.multiple objects.

Usage

```
## S3 methods
## S3 method for class 'fdt.default'
summary(object,
        columns = 1:6,
        round = 2,
        format.classes = FALSE,
        pattern = "%09.3e",
        row.names = FALSE,
        right = TRUE, ...)

## S3 method for class 'fdt.multiple'
summary(object,
        columns = 1:6,
        round = 2,
        format.classes = FALSE,
        pattern = "%09.3e",
        row.names = FALSE,
        right = TRUE, ...)

## S3 method for class 'fdt_cat.default'
summary(object,
        columns = 1:6,
        round = 2,
        row.names = FALSE,
        right = TRUE, ...)

## S3 method for class 'fdt_cat.multiple'
summary(object,
        columns = 1:6,
        round = 2,
        row.names = FALSE,
        right = TRUE, ...)
```

Arguments

object	a fdt or fdt_cat object.
columns	a vector of integers to select columns of the data.frame table.
round	rounds 'fdt' columns to the specified number of decimal places (default 2).
format.classes	logical, if TRUE the first column of the data.frame table will be formatted using a regular expression. The default is "%09.3e".
pattern	same as fmt in sprintf.
row.names	logical (or character vector), indicating whether (or what) row names should be printed. The default is FALSE.
right	logical, indicating whether or not strings should be right-aligned. The default is right-alignment.
...	optional further arguments (required by generic).

Details

It is possible to select what columns of the table (a `data.frame`) will be shown, as well as the pattern of the first column. The columns are:

1. '`Class limits`'
2. '`f`' - absolute frequency
3. '`rf`' - relative frequency
4. '`rf(%)`' - relative frequency, %
5. '`cf`' - cumulative frequency
6. '`cf(%)`' - cumulative frequency, %

The available parameters offer an easy and powerful way to format the '`fdt`' for publications and other purposes.

Value

A single `data.frame` for `fdt.default` or multiple `data.frames` for `fdt.multiple`.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

Examples

```
library (fdth)

#=====================
# Vectors: univariate
#=====================
set.seed(1)

x <- rnorm(n = 1e3,
           mean = 5,
           sd = 1)

ft <- fdt(x)

str(ft)

ft

summary(ft)  # same result

summary(ft,
        format = TRUE)      # It may not be what you want for publication!

summary(ft,
        format = TRUE,
```

```
        pattern = '%.2f')   # Better, but can it be improved?

summary(ft,
        col = c(1:2, 4, 6),
        format = TRUE,
        pattern = '%.2f')   # Yes, it can!

range(x)                 # To inspect the range of x

summary(fdt(x,
            start = 1,
            end = 9,
            h = 1),
        col = c(1:2, 4, 6),
        format = TRUE,
        pattern = '%d')     # Is it nice now?

ft[['table']]                       # Stores the frequency distribution table (fdt)
ft[['breaks']]                      # Stores the breaks of fdt
ft[['breaks']]['start']             # Stores the left value of the first class
ft[['breaks']]['end']               # Stores the right value of the last class
ft[['breaks']]['h']                 # Stores the class interval
as.logical(ft[['breaks']]['right'])  # Stores the right option

#=====================================================
# Data.frames: multivariate with categorical variables
#=====================================================
mdf <- data.frame(X1 = rep(LETTERS[1:4], 25),
                  X2 = as.factor(rep(1:10, 10)),
                  Y1 = c(NA,
                         NA,
                         rnorm(96,
                               10,
                               1),
                         NA,
                         NA),
                  Y2 = rnorm(100,
                             60,
                             4),
                  Y3 = rnorm(100,
                             50,
                             4),
                  Y4 = rnorm(100,
                             40,
                             4),
                  stringsAsFactors = TRUE)

ft_c <- fdt_cat(mdf)

summary(ft_c)

ft <- fdt(mdf,
          na.rm = TRUE)
```

```
str(ft)

summary(ft)  # same result

summary(ft,
        format = TRUE)

summary(ft,
        format = TRUE,
        pattern = '%05.2f')  # regular expression

summary(ft,
        col = c(1:2, 4, 6),
        format = TRUE,
        pattern = '%05.2f')

print(ft,
      col = c(1:2, 4, 6))

print(ft,
      col = c(1:2, 4, 6),
      format = TRUE,
      pattern = '%05.2f')

levels(mdf$X1)

summary(fdt(mdf,
            k = 5,
            by = 'X1',
            na.rm = TRUE))

levels(mdf$X2)

summary(fdt(mdf,
            breaks = 'FD',
            by = 'X2',
            na.rm = TRUE),
        round = 3)

summary(fdt(mdf,
            k = 5,
            by = 'X2',
            na.rm = TRUE),
        format = TRUE,
        round = 3)

summary(fdt(iris,
            k = 5),
        format = TRUE,
        patter = '%04.2f')

levels(iris$Species)
```

```
summary(fdt(iris,
            k = 5,
            by = 'Species'),
        format = TRUE,
        patter = '%04.2f')

#========================
# Matrices: multivariate
#========================
summary(fdt(state.x77),
        col = c(1:2, 4, 6),
        format = TRUE)

summary(fdt(volcano,
            right = TRUE),
        col = c(1:2, 4, 6),
        round = 3,
        format = TRUE,
        pattern = '%05.1f')
```

`ta` *Shortcut for total amplitude*

Description

`ta` is a shortcut to `amplitude`.

Usage

```
ta(x, ...)
```

Arguments

`x`	an object accepted by `amplitude`.
`...`	additional arguments passed to `amplitude`.

Details

This function is equivalent to calling `amplitude(x, ...)`.

Value

Returns the same output as `amplitude`.

See Also

`amplitude`.

Examples

```
library(fdth)

x <- rnorm(1e3,
           20,
           2)

ta(x)
ta(fdt(x))
ta(fdt(iris[, 1:4]))
```

---

var *Variance of frequency distribution table (numerical variable)*

Description

S3 methods for the variance of a `fdt`.
Useful to estimate the variance (when the real data vector is not known) from a previous `fdt`.

Usage

```
## S3 generic
var(x, ...)

## S3 methods: numerical
## Default S3 method:
var(x, ...)

## S3 method for class 'fdt'
var(x, ...)

## S3 method for class 'fdt.multiple'
var(x, ...)
```

Arguments

`x`	a `fdt` (simple or multiple) object.
`...`	required to be generic.

Details

`var.fdt` calculates the value of the variance based on a known formula. `var.fdt.multiple` calls `var.fdt` for each variable, that is, each column of the data.frame.

Value

`var.fdt` returns a numeric vector containing the value of the variance of the `fdt`. `var.fdt.multiple` returns a list, where each element is a numeric vector containing the value of the variance of the `fdt` for each variable.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

`sd.fdt`, `mean.fdt`.

Examples

```
library(fdth)

mdf <- data.frame(x = rnorm(1e2,
                            20,
                            2),
                  y = rnorm(1e2,
                            30,
                            3),
                  z = rnorm(1e2,
                            40,
                            4))

head(mdf)

# From a data.frame
apply(mdf,
      2,
      var)

# From a fdt object
var(fdt(mdf))

unlist(var(fdt(mdf)))
```

`xtable.fdt` *LaTeX table of the frequency distribution table*

Description

This function returns a LaTeX table of the `fdt`, `fdt.multiple`, `fdt.quantile`, `fdt.quantile.multiple`, `fdt_cat`, and `fdt_cat.multiple` objects of the `xtable` class.

Usage

```
## S3 method for class 'fdt'
xtable(x,
       caption = NULL,
       label = NULL,
```

```
       align = NULL,
       digits = NULL,
       display = NULL,
       auto = FALSE,
       ...)

## S3 method for class 'fdt.multiple'
xtable(x,
       caption = NULL,
       label = NULL,
       align = NULL,
       digits = NULL,
       display = NULL,
       ...)

## S3 method for class 'fdt.quantile'
xtable(x,
       caption = NULL,
       label = NULL,
       align = NULL,
       digits = NULL,
       display = NULL,
       auto = FALSE,
       ...)

## S3 method for class 'fdt.quantile.multiple'
xtable(x,
       caption = NULL,
       label = NULL,
       align = NULL,
       digits = NULL,
       display = NULL,
       ...)

## S3 method for class 'fdt_cat'
xtable(x,
       caption = NULL,
       label = NULL,
       align = NULL,
       digits = NULL,
       display = NULL,
       auto = FALSE,
       ...)

## S3 method for class 'fdt_cat.multiple'
xtable(x,
       caption = NULL,
       label = NULL,
```

```
        align = NULL,
        digits = NULL,
        display = NULL,
        ...)
```

Arguments

x	A fdt, fdt.multiple, fdt.quantile, fdt.quantile.multiple, fdt_cat, or fdt_cat.multiple class object.
caption	Character vector of length 1 or 2 containing the table caption or title. See the xtable function for more details.
label	Character vector of length 1 containing the LaTeX label or HTML anchor. See the xtable function for more details.
align	Character vector of length equal to the number of columns of the resulting table, indicating the alignment of the corresponding columns. See the xtable function for more details.
digits	Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table, or matrix of the same size as the resulting table, indicating the number of digits to display in the corresponding columns. See the xtable function for more details.
display	Character vector of length equal to the number of columns of the resulting table, indicating the format for the corresponding columns. See the xtable function for more details.
auto	Logical, indicating whether to apply automatic format when no value is passed to align, digits, or display
...	Additional arguments.

Details

The functions latex.fdt was deprecated. We understand over the years that creating a method for the generic xtable function would be inevitable, given the advancement of the xtable package and its support by the academic community.

Then, the fdt, fdt.multiple, fdt.quantile, fdt.quantile.multiple, fdt_cat, and fdt_cat.multiple methods were created for the generic xtable function.

Objects of the fdt.multiple, fdt.quantile.multiple, and fdt_cat.multiple class, when using the xtable function, will have the xtableList class. Although it may seem confusing, the xtableList function in the xtable package has no generic function and therefore it was not possible to create a method of type xtableList.fdt.multiple. Therefore, a method of the xtable.fdt.multiple class was created, but the function xtableList is being used internally.

More examples than those provided in the manual can be seen in the vignette.

It is possible to select what columns of the table (a data.frame) will be shown, as well as the pattern of the first column. The columns are:

1. 'Class limits'
2. 'f' - Absolute frequency

3. 'rf' - Relative frequency
4. 'rf(%)' - Relative frequency, %
5. 'cf' - Cumulative frequency
6. 'cf(%)' - Cumulative frequency, %

Value

An object of the class xtable, xtableList, or related classes from the **xtable** package.

Author(s)

Faria, J. C.
Allaman, I. B
Jelihovschi, E. G.

See Also

xtable, xtableList, quantile.fdt

Examples

```
library(fdth)
library(xtable)

# +++++ Quantitative data
## Example 1: The simplest possible
t1 <- fdt(rnorm(n = 1e3,
                mean = 10,
                sd = 2))

t1x <- xtable(t1)
t1x

## Example 2
print(t1x,
      include.rownames = FALSE)

## Example 3
newclass <- gsub("[\\[\\)]",
                 "",
                 t1x[,1],
                 perl = TRUE)
t3x <- t1x
t3x[,1] <- newclass

print(t3x,
      include.rownames = FALSE,
      sanitize.text.function = function(x)gsub(",",
                                               "\\dashv",
                                               x,
                                               perl = TRUE))
```

```
## Not run:
## Example 4
clim <- t1$table[1]
clim1 <- sapply(clim,
                as.character)
right <- t1$breaks[4]
pattern <- "%05.2f"
clim2 <- make.fdt.format.classes(clim1,
                                 right,
                                 pattern)
clim3 <- sapply(clim2,
                function(x) paste0("$",
                                   x,
                                   "$"))
t4x <- t1x
t4x[,1] <- clim3

print(t4x,
      include.rownames = FALSE)


## End(Not run)
## Example 5
t5 <- fdt(iris,
          by = "Species")
attr(t5, "subheadings") <- paste0("Variable = ", names(t5))
xtable(t5)

## Quantiles estimated from grouped data
q1 <- quantile(t1,
               i = 1:3)
xtable(q1)

q5 <- quantile(fdt(iris[, 1:4]),
               i = 1:3)
attr(q5, "subheadings") <- names(q5)
xtable(q5)

# +++++ Qualitative data
## Example 6
t6 <- fdt_cat(sample(LETTERS[1:3],
                     replace = TRUE,
                     size = 30))

t6x <- xtable(t6)
t6x

t61 <- fdt_cat(data.frame(c1 = sample(LETTERS[1:3],
                                      replace = TRUE,
                                      size = 10),
                          c2 = sample(letters[4:5],
                                      replace = TRUE,
                                      size = 10),
```

```
                            stringsAsFactors = TRUE))

attr(t61, "subheadings") <- paste0("Variable = ", names(t61))
t61x <- xtable(t61)
t61x
```

Index

∗ **distribution**
fdt, 11
fdt_cat, 15
fdth-package, 2
plot.fdt, 26
print.fdt, 35
summary.fdt, 44
∗ **fdt_cat**
fdt, 11
fdt_cat, 15
∗ **fdt**
amplitude, 10
fdt, 11
fdt_cat, 15
fdth-package, 2
plot.fdt, 26
print.fdt, 35
summary.fdt, 44
ta, 49
xtable.fdt, 51
∗ **frequency**
fdt, 11
fdt_cat, 15
fdth-package, 2
plot.fdt, 26
print.fdt, 35
summary.fdt, 44
∗ **histogram**
fdth-package, 2
plot.fdt, 26
∗ **plot**
plot.fdt, 26
∗ **print**
print.fdt, 35
∗ **range**
amplitude, 10
ta, 49
∗ **summary**
summary.fdt, 44
∗ **table**
fdt, 11
fdt_cat, 15
fdth-package, 2
plot.fdt, 26
print.fdt, 35
summary.fdt, 44
∗ **xable**
xtable.fdt, 51

amplitude, 10

cut, *3*, *13*, *17*, *19*

fdt, 11
fdt_cat, 15
fdth *(*fdth-package*)*, 2
fdth-package, 2

hist, *3*, *13*, *17*

make.fdt, 18
make.fdt_cat *(*make.fdt*)*, 18
mean.fdt, 20
median.fdt, 22
mfv, 23, *40*

plot.fdt, 26
plot.fdt.default *(*plot.fdt*)*, 26
plot.fdt.multiple *(*plot.fdt*)*, 26
plot.fdt_cat.default *(*plot.fdt*)*, 26
plot.fdt_cat.multiple *(*plot.fdt*)*, 26
print.fdt, 35
print.fdt.default *(*print.fdt*)*, 35
print.fdt.multiple *(*print.fdt*)*, 35
print.fdt_cat.default *(*print.fdt*)*, 35
print.fdt_cat.multiple *(*print.fdt*)*, 35

quantile.fdt, *24*, 39, *54*

sd, 43

sprintf, *36*, *45*
summary.fdt, 44
summary.fdt.default *(*summary.fdt*)*, 44
summary.fdt.multiple *(*summary.fdt*)*, 44
summary.fdt_cat.default *(*summary.fdt*)*, 44
summary.fdt_cat.multiple *(*summary.fdt*)*, 44

ta, 49
table, *3*, *13*, *17*, *19*

var, 50

xtable, *54*
xtable.fdt, *40*, 51
xtable.fdt.multiple *(*xtable.fdt*)*, 51
xtable.fdt.quantile *(*xtable.fdt*)*, 51
xtable.fdt_cat *(*xtable.fdt*)*, 51
xtableList, *54*