Package 'fdth'

May 27, 2010

Type Package

Title Frequency Distribution Tables, Histograms and Poligons

Version 1.1-2

Date 2010-05-27

Author Jose Claudio Faria & Enio Jelihovschi

Maintainer Jose Claudio Faria <joseclaudio.faria@gmail.com>

Depends R (>= 2.6.0), base, grDevices, graphics

Description Perform frequency distribution tables, associated histograms and poligons from vector, data.frame and matrix objects.

License GPL (>= 2)

LazyLoad yes

R topics documented:

`fdth-package` *Frequency Distribution Tables, Histograms and Poligons*

Description

The `fdth` package contains a set of functions which easily allows the user to make frequency distribution tables (fdt), its associated histograms and frequency poligons (absolut, relative and cumulative). The fdt can be formatted in many ways which may be suited to publication in many different ways (papers, books, etc). The `plot` method (S3) is the histogram which can be dealt with the easiness 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 table (fdt) can be used for both ordinal and continuous variables.

The `R` environment provides a set of functions (generally low level) enabling the user to perfom 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 novices or ocasional users of `R`, it can be laborious to find out all necessary funtions and graphical parameters to do a normatized and pretty fdt and the associated histogram ready for publications.

That is the aim of this package, i.e, to allow the user to do (using a few, simple and flexible high level set of S3 functions) with ease and flexibility both: the fdt and histogram. The input data for univariated is generally a `vector`. For multivariated data can be used both: a `data.frame`, in this case also alowing grouping all numerical variables according to one categorical, or `matrices`.

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

It can be provided also: a) `x` and `k` (number of class intervals); b) `x`, `start` (left endpoint of the first class interval) and `end` (right endpoint of the last class interval); or c) `x`, `start`, `end` and `h` (class interval width). These options make the `fdt` very easy and flexible.

The `fdt` object stores information to be used by methods `summary`, `print` and `plot`. The result of `plot` is a histogram or poligon (absolut, relative or cummulative). 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)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

Maintainer: Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)

See Also

`hist` provided by `graphics`; `table`, `cut` both provided by `base` and `hist.data.frame` provided by `Hmisc` package.

Examples

```
library (fdth)

#======================
# Vectors: univariated
#======================
set.seed(1)
x <- rnorm(n=1e3, mean=5, sd=1)

d <- fdt(x); d

# Histograms
plot(d)
plot(d, main='My title')
plot(d, x.round=3, col='darkgreen')
plot(d, x.las=2)
plot(d, x.round=2, x.las=2, xlab=NULL)
plot(d, x.round=2, x.las=2, xlab=NULL, col=rainbow(11))

plot(d, type='fh')
plot(d, type='rfh')
plot(d, type='rfph')
plot(d, type='cdh')
plot(d, type='cfh')
plot(d, type='cfph')

# Poligons
plot(d, type='fp')
plot(d, type='rfp')
plot(d, type='rfpp')
plot(d, type='cdp')
plot(d, type='cfp')
plot(d, type='cfpp')

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

# Summary
d
summary(d) # the same
print(d)   # the same
show(d)    # the same
summary(d, format=TRUE)                     # It can not be what you want to publications!
summary(d, format=TRUE, pattern='%.2f')    # Huumm ..., good, but ... Can it be better?
summary(d,
        col=c(1:2, 4, 6),
        format=TRUE, pattern='%.2f')       # Yes, it can!

range(x)                                    # To know 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
d[['table']]                                # Stores the feq. dist. table (fdt)
d[['breaks']]                               # Stores the breaks of fdt
```

```
d[['breaks']]['start']                    # Stores the left value of the first class
d[['breaks']]['end']                      # Stores the right value of the last class
d[['breaks']]['h']                        # Stores the class interval
as.logical(d[['breaks']]['right'])        # Stores the right option

# 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=2)

#=============================================
# Data.frames: multivariated with categorical
#=============================================
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))

d <- fdt(mdf); d

# Histograms
plot(d, main=TRUE)
plot(d, col='darkgreen', ylim=c(0, 40), main=TRUE)
plot(d, col=rainbow(8), main=TRUE)

plot(d, type='fh')
plot(d, type='rfh')
plot(d, type='rfph')
plot(d, type='cdh')
plot(d, type='cfh')
plot(d, type='cfph')

# Poligons
plot(d, type='fp')
plot(d, type='rfp')
plot(d, type='rfpp')
plot(d, type='cdp')
plot(d, type='cfp')
plot(d, type='cfpp')

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

# Summary
d
summary(d) # the same
print(d)   # the same
show(d)    # the same
summary(d, format=TRUE)
summary(d, format=TRUE, pattern='%05.2f') # regular expression
summary(d, col=c(1:2, 4, 6), format=TRUE, pattern='%05.2f')

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

```
# Using by
levels(mdf$X1)
summary(fdt(mdf, k=5, by='X1'))
plot(fdt(mdf, k=5, by='X1'), col=rainbow(5), main=TRUE)

levels(mdf$X2)
summary(fdt(mdf, breaks='FD', by='X2'), round=3)
plot(fdt(mdf, breaks='FD', by='X2'), main=TRUE)

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

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

# Big fdt
require(MASS)
levels(Cars93$Origin)
summary(fdt(Cars93, k=5, by='Origin'), format=TRUE)
plot(fdt(Cars93, k=5, by='Origin'), col=heat.colors(5), main=TRUE)

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

# 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), main=TRUE)
```

fdt *Frequency Distribution Table*

Description

A S3 set of methods to easily perform frequency distribution table (fdt) from `vector`, `data.frame` and `matrix` objects.

Usage

```
  fdt(x, ...)

  ## Default S3 method:
fdt(x, k, start, end, h, breaks=c("Sturges", "Scott", "FD"),
    right=FALSE, ...)
  ## S3 method for class 'data.frame':
fdt(x, k, by, breaks=c("Sturges", "Scott", "FD"),
    right=FALSE, ...)
  ## S3 method for class 'matrix':
fdt(x, k, breaks=c("Sturges", "Scott", "FD"),
    right=FALSE, ...)
```

Arguments

`x`	A numeric `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.frames`.
`breaks`	Method used to determine the number of interval classes, c("Sturges", "Scott", "FD").
`right`	Right endpoints open (default = FALSE).
`...`	Potencial further arguments (required by generic).

Details

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

It can also be provided: a) `x` and `k`; b) `x`, `start` and `end`; or c) `x`, `start`, `end` and `h`. These options make the `fdt` very easy and flexible to use.

The `fdt` object stores information to be used by methods `summary`, `print` and `plot`. 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 pretty (and publishable) way.

Value

The method `fdt.default` returns a list of class `fdt.default` 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.
`data`	A vector of the data `x` provided.

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

Author(s)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

See Also

`hist` provided by `graphics`; `table`, `cut` both provided by `base` and `hist.data.frame` provided by `Hmisc` package.

Examples

```
library(fdth)

#======================
# Vectors: univariated
#======================
set.seed(1); x <- rnorm(n=1e3, mean=5, sd=1)

# x
d <- fdt(x); d

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

# x, k
d <- fdt(x, k=20); d

# x, star, end
range(x)
d <- fdt(x, start=1.5, end=9); d

# x, start, end, h
d <- fdt(x, start=1, end=9, h=1); d

# Effect of right
x <- rep(1:3, 3); sort(x)
d <- fdt(x, start=1, end=4, h=1); d

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

#=============================================
# Data.frames: multivariated with categorical
#=============================================
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))

d <- fdt(mdf); d

levels(mdf$X1)
d <- fdt(mdf, k=5, by='X1'); d

d <- fdt(mdf, breaks='FD', by='X1')
str(d)
d

levels(mdf$X2)
d <- fdt(mdf, breaks='FD', by='X2'); d

d <- fdt(mdf, k=5, by='X2'); d

d <- fdt(iris, k=5); d
```

```
d <- fdt(iris, k=10); d

levels(iris$Species)
d <- fdt(iris, k=5, by='Species'); d

require(MASS)
levels(Cars93$Origin)
d <- fdt(Cars93, k=5, by='Origin'); d

d <- fdt(Cars93, breaks='FD', by='Origin'); d

#=========================
# Matrices: multivariated
#=========================
d <-fdt(state.x77); d

d <-fdt(volcano); d
```

`make.fdt.format.classes` *This function formats the presentation's appearance of the interval classes*

Description

This function uses part of the `fdt` results which contains the interval classes description and format it according to a pattern wich is a regular expression, the same as used by `sprintf`. It is mainly for internal use of the fdth package.

Usage

```
make.fdt.format.classes(x, right, pattern)
```

Arguments

`x`	A `fdt` object.
`right`	Intervals right open (default = FALSE).
`pattern`	Same as `fmt` in `sprintf`.

Details

This function uses the object[["table"]] of the `fdt` results (object) which contains the interval classes description and format it according to a pattern wich is a regular expression, the same as used by `sprintf`. It is called by the generic function `summary` and is mainly for internal use of the fdth package.

Value

The function `make.fdt.format.classes` returns a `data.frame` which contains the formatted interval class values.

Note

This function is mainly for internal use in the `fdt` package, and may not remain available (unless we see a good reason).

Author(s)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

See Also

`sprintf` and `gsub` provided by `base` package.

`make.fdt.multiple` *A function which makes the table using the number of interval classes, a method used for breaks, and choose the endpoint*

Description

A function which makes the table by using parameters defining the number of interval classes, the method used for breaks, "Sturges", "Scott", "FD", and the closure of the interval class endpoints being left or right.

Usage

```
make.fdt.multiple(x, k, breaks = c("Sturges", "Scott", "FD"), right)
```

Arguments

`x`	A `data.frame` or `matrix` object containing at last one numeric column.
`k`	Number of class intervals.
`breaks`	Method to determine number of classes, c("Sturges", "Scott", "FD").
`right`	Intervals right open (default = FALSE).

Details

A function which makes the table by using parameters defining the number of interval classes, the method used for breaks, "Sturges", "Scott", "FD", and the closure of the interval class endpoints being left or right.

Value

The function `make.fdt.multiple` 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.
`data`	A vector of the data `x` provided.

Note

This function is mainly for internal use in the `fdt` package, and may not remain available (unless we see a good reason).

Author(s)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

See Also

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

`make.fdt.simple` *A function which makes the table with absolute and relative frequencies, cumulative frequencies in numbers and percentages.*

Description

This function is called by `fdt.default` and `make.fdt.multiple`. It makes a table from a vector of numbers. The table consists of absolute and relative frequencies, cumulative frequencies in numbers and percentages.

Usage

```
make.fdt.simple(x, start, end, h, right)
```

Arguments

`x`	A numeric `vector` object.
`start`	The left value of the interval of the first class.
`end`	The last value of the interval of the last class.
`h`	The class interval.
`right`	Intervals right open (default = FALSE).

Details

This function is called by `fdt.default` and makes a table from a vector of numbers. The table consists of absolute and relative frequencies, cumulative frequencies in numbers and percentages. The result is used by the generic function `summary` to format it in a table suited for publication.

Value

The function `make.fdt.simple` returns a `data.frame` which contains the table with interval classes and frequencies. The following are the columns:

`Class limits`	Interval classes, `character`
`f`	Absolute frequency, `numeric`
`rf`	Relative frequency, `numeric`

`rf(%)`	Relative frequency in percentages, `numeric`
`cf`	Cumulative frequency; `numeric`
`cf(%)`	Cumulative frequency in percentages, `numeric`

Note

This function is mainly for internal use in the `fdt` package, and may not remain available (unless we see a good reason).

Author(s)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

See Also

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

`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 poligons (absolute, relative and cumulative).

Usage

```
  ## S3 method for class 'fdt.default':
plot(x, type=c('fh', 'fp', 'rfh', 'rfp', 'rfph', 'rfpp',
    'd', 'cdh', 'cdp', 'cfh', 'cfp', 'cfph', 'cfpp'),
    xlab="Class limits", ylab=NULL, col="gray", xlim=NULL, ylim=NULL,
    main=NULL, x.round=2, x.las=1, ...)
  ## S3 method for class 'fdt.multiple':
plot(x, type=c('fh', 'fp', 'rfh', 'rfp', 'rfph', 'rfpp',
    'd', 'cdh', 'cdp', 'cfh', 'cfp', 'cfph', 'cfpp'),
    xlab="Class limits", ylab=NULL, col="gray", xlim=NULL, ylim=NULL,
    main=NULL, x.round=2, x.las=1, ...)
```

Arguments

`x`	A `fdt` object.
`type`	The type of the plot. `fh:` Absolut frequency histogram, `fp:` Absolut frequency poligon, `rfh:` Relative frequency histogram, `rfp:` Relative frequency poligon, `rfph:` Relative frequency (%) histogram , `rfpp:` Relative frequency (%) poligon, `d:` Density, `cdh:` Cumulative density histogram, `cdp:` Cumulative density poligon, `cfh:` Cumulative frequency histogram, `cfp:` Cumulative frequency poligon, `cdph` cumulative frequency (%) histogram, `cfpp:` cumulative frequency (%) poligon.
`xlab`	A label for the x axis.

`ylab`	A label for the y axis.
`col`	A `vector` of colors.
`xlim`	The x limits of the plot.
`ylim`	The y limits of the plot.
`main`	A title for the plot.
`x.round`	A numeric value to round the x ticks.
`x.las`	An integer which controls the orientation of the x axis labels (0: parallel to the axes, 1: horizontal, 2: perpendicular to the axes, 3: vertical)
`...`	Optional plotting parameters.

Details

The result is a single histogram or poligon (absolute, relative or cummulative) for `fdt.default` or a set of histograms or poligon (absolute, relative or cummulative) 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.

Author(s)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

See Also

`hist.data.frame` provided by `Hmisc` package.

Examples

```
library(fdth)

#======================
# Vectors: univariated
#======================
set.seed(1)
x <- rnorm(n=1e3, mean=5, sd=1)
d <- fdt(x); d

# Histograms
plot(d)
plot(d, main='My title')
plot(d, x.round=3, col='darkgreen')
plot(d, x.las=2)
plot(d, x.round=2, x.las=2, xlab=NULL)
plot(d, x.round=2, x.las=2, xlab=NULL, col=rainbow(11))

plot(d, type='fh')
plot(d, type='rfh')
plot(d, type='rfph')
plot(d, type='cdh')
plot(d, type='cfh')
plot(d, type='cfph')
```

```
# Poligons
plot(d, type='fp')
plot(d, type='rfp')
plot(d, type='rfpp')
plot(d, type='cdp')
plot(d, type='cfp')
plot(d, type='cfpp')

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

# 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=2)

#==============================================
# Data.frames: multivariated with categorical
#==============================================
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))

# Histograms
d <- fdt(mdf); d
plot(d, main=TRUE)
plot(d, col='darkgreen', ylim=c(0, 40), main=TRUE)
plot(d, col=rainbow(8), ylim=c(0, 40), main=TRUE)

plot(d, type='fh')
plot(d, type='rfh')
plot(d, type='rfph')
plot(d, type='cdh')
plot(d, type='cfh')
plot(d, type='cfph')

# Poligons
plot(d, type='fp')
plot(d, type='rfp')
plot(d, type='rfpp')
plot(d, type='cdp')
plot(d, type='cfp')
plot(d, type='cfpp')

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

levels(mdf$X1)
plot(fdt(mdf, k=5, by='X1'), ylim=c(0, 12), main=TRUE)

levels(mdf$X2)
plot(fdt(mdf, breaks='FD', by='X2'), main=TRUE)
plot(fdt(mdf, k=5, by='X2'), main=TRUE)               # It is dificult to compare
plot(fdt(mdf, k=5, by='X2'), ylim=c(0, 8), main=TRUE) # Easy

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

d <- fdt(iris, k=10)
plot(d, main=TRUE)
plot(d, type='d', main=TRUE)

require(MASS)
levels(Cars93$Origin)
plot(fdt(Cars93, k=5, by='Origin'), col=heat.colors(5), main=TRUE)

plot(fdt(Cars93, breaks='FD', by='Origin'), main=TRUE)

#=========================
# Matrices: multivariated
#=========================
plot(fdt(state.x77), main=TRUE)

plot(fdt(volcano), main=TRUE)
```

summary.fdt *Summary and Print Methods for fdt Objects*

Description

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

Usage

```
  ## S3 method for class 'fdt.default':
summary(object, columns=1:6, round=2,
    format.classes=FALSE, pattern="%09.3e", ...)
  ## S3 method for class 'fdt.multiple':
summary(object, columns=1:6, round=2,
    format.classes=FALSE, pattern="%09.3e", ...)
  ## S3 method for class 'fdt.default':
print(x, ...)
  ## S3 method for class 'fdt.multiple':
print(x, ...)
```

Arguments

`object`	A `fdt` object.
`x`	A `fdt` object.
`columns`	A `vector` of `integers` to select colums of the `data.frame` table.
`round`	Rounds the fractionary columns of fdt to the specified number of decimal places (default 2).

`format.classes`	Logical, if `TRUE` the first column of the data.frame table will be formated using regular expression. The default is "%09.3e".
`pattern`	Same as `fmt` in `sprintf`.
`...`	Potential further arguments (require by generic).

Details

The methods `print` and `show` are wrappers for the method `summary`.

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` (absolut 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)

Jose Claudio Faria (<`joseclaudio.faria@gmail.com`>)
Enio Jelihovschi (<`eniojelihovs@gmail.com`>)

Examples

```
library (fdth)

#======================
# Vectors: univariated
#======================
set.seed(1)
x <- rnorm(n=1e3, mean=5, sd=1)

d <- fdt(x)
str(d)

d
summary(d)  # the same
print(d)    # the same
show(d)     # the same
summary(d, format=TRUE)                     # It can not be what you want to publications!
summary(d, format=TRUE, pattern='%.2f')     # Huumm ..., good, but ... Can it be better?
summary(d,
        col=c(1:2, 4, 6),
        format=TRUE, pattern='%.2f')        # Yes, it can!

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

d[['table']]                                # Stores the feq. dist. table (fdt)
```

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

#=============================================
# Data.frames: multivariated with categorical
#=============================================
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))

d <- fdt(mdf)
d

str(d)
summary(d) # the same
print(d)   # the same
show(d)    # the same
summary(d, format=TRUE)
summary(d, format=TRUE, pattern='%05.2f') # regular expression
summary(d, col=c(1:2, 4, 6), format=TRUE, pattern='%05.2f')

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

levels(mdf$X1)
summary(fdt(mdf, k=5, by='X1'))

levels(mdf$X2)
summary(fdt(mdf, breaks='FD', by='X2'), round=3)

summary(fdt(mdf, k=5, by='X2'), 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')

require(MASS)
levels(Cars93$Origin)
summary(fdt(Cars93, k=5, by='Origin'), format=TRUE)

#=========================
# Matrices: multivariated
#=========================
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')
```

Index