

Package ‘doudpackage’

September 25, 2026

Title Create Elegant Table 1 in HTML/'LaTeX' for Bio-Statistics

Version 2.2.0

Description Creates the ``table one'' of bio-medical papers. Fill it with your data and the name of the variable which you'll make the group(s) out of and it will make univariate and bivariate analysis, and parse the result into an HTML or 'LaTeX' table ready to paste into a paper.

License GPL (>= 3)

Encoding UTF-8

Language en-US

Imports dplyr, kableExtra, methods, parallel, purrr, stats, tibble, tidyr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

URL <https://github.com/tiago972/doudpackage>

BugReports <https://github.com/tiago972/doudpackage/issues>

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Edouard Baudouin [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-8324-6908>>)

Maintainer Edouard Baudouin <edouardpierre.baudouin@aphp.fr>

Repository CRAN

Date/Publication 2026-09-25 14:10:12 UTC

Contents

anaBiv	2
anaBiv,data.frame,character-method	3
anaBiv,listVar,character-method	4
descTab	4
ft_ana_biv	6

ft_desc_tab	7
ft_parse	7
initialize,parseClass-method	8
initialize,Var-method	9
initialize,VarGroup-method	9
listVar-class	10
parseClass	10
parseClass-class	11
parseClassFun	12
parseClassFun,parseClass-method	13
Var	15
Var-class	15
VarGroup-class	16
[,parseClass-method	16
[,Var-method	17
[,VarGroup-method	17
[<-,parseClass-method	18
[<-,Var-method	18
[<-,VarGroup-method	19
Index	20

anaBiv	<i>anaBiv generic function</i>
--------	--------------------------------

Description

Generic function of anaBiv which gives bivariate analysis according to group

Usage

```
anaBiv(var, group, parallel, ...)
```

Arguments

var	listVar object or data.frame
group	Name of the factor variable to make subgroups with
parallel	Logical. Make analysis using parallel from <code>parallel::mclapply()</code> .
...	Further arguments: normality and digits.p, as in <code>descTab()</code> .

Value

A list of VarGroup object or data.frame

anaBiv,data.frame,character-method
anaBiv data.frame function

Description

Generic function of anaBiv which gives bivariate analysis according to group

Usage

```
## S4 method for signature 'data.frame,character'
anaBiv(var, group, parallel, normality = "normal", digits.p = 3L, ...)
```

Arguments

var	listVar object or data.frame
group	Name of the factor variable to make subgroups with
parallel	Logical. Make analysis using parallel from parallel::mclapply() .
normality	One of "normal", "non normal" or "assess", as in descTab() .
digits.p	Integer. Significant digits for p value.
...	Further arguments: normality and digits.p, as in descTab() .

Value

A list of VarGroup object or data.frame

Examples

```
# A small simulated clinical trial
set.seed(42)
n <- 200
patients <- data.frame(
  arm      = factor(sample(c("Placebo", "Treatment"), n, replace = TRUE)),
  age      = round(rnorm(n, mean = 65, sd = 10)),
  crp      = round(rlnorm(n, meanlog = 2, sdlog = 1), 1),
  sex      = factor(sample(c("Female", "Male"), n, replace = TRUE)),
  diabetes = factor(sample(c("No", "Yes"), n, replace = TRUE, prob = c(0.7, 0.3))),
  nyha     = factor(sample(c("I", "II", "III", "IV"), n, replace = TRUE),
                    ordered = TRUE)
)
patients$crp[sample(n, 15)] <- NA

# p values only, without the descriptive table
res <- anaBiv(patients, group = "arm", parallel = FALSE, normality = "assess")
data.frame(variable = sapply(res, function(x) x["name"]),
           pvalue = sapply(res, function(x) x["pvalue"]))
```

anaBiv,listVar,character-method
anaBiv data.frame function

Description

Generic function of anaBiv which gives bivariate analysis according to group

Usage

```
## S4 method for signature 'listVar,character'
anaBiv(var, group, parallel, ...)
```

Arguments

var	listVar object or data.frame
group	Name of the factor variable to make subgroups with
parallel	Logical. Make analysis using parallel from parallel::mclapply() .
...	Further arguments: normality and digits.p, as in descTab() .

Value

A list of VarGroup object or data.frame

descTab	<i>Create a table of descriptive analysis of a dataset</i>
---------	--

Description

Displays together the univariate analysis (mean/median; SD/IQR; proportions) and the bivariate analysis (t test/Wilcoxon, ANOVA/Kruskal-Wallis, Chi2 or Fisher) of every variable of a dataset. The univariate analysis can be sub-grouped by a variable of interest of n levels; the test applied to each variable follows from its type and from normality.

Usage

```
descTab(
  data,
  group = "",
  quanti = TRUE,
  quali = TRUE,
  na.print = FALSE,
  pvalue = TRUE,
  digits.p = 3L,
  digits.qt = 1L,
```

```

    digits.q1 = 1L,
    normality = "normal",
    parallel = FALSE,
    mc.cores = 0
  )

```

Arguments

<code>data</code>	A dataset. Needs to be a <code>data.frame</code> or a <code>tibble</code> .
<code>group</code>	Optional. The name of the factor variable to make sub-groups comparisons with; it must have at least two non empty levels. Omitted, "" or <code>NULL</code> , the table has a single Total column. Rows with a missing value for it are dropped, and its unused levels are dropped, both with a warning.
<code>quanti</code> , <code>quali</code> , <code>na.print</code> , <code>pvalue</code>	Logical. If false, won't display quantitative/qualitative/Missing values/pvalues variable results. <code>quanti</code> and <code>quali</code> cannot both be <code>FALSE</code> .
<code>digits.p</code>	Integer. Significant digits for p value
<code>digits.qt</code>	Integer. Significant digits for mean/median, SD/IQR
<code>digits.q1</code>	Integer. Significant digits for proportions
<code>normality</code>	One of "normal", "non normal" or "assess". "normal" applies the parametric tests (mean (SD), t test/ANOVA) to every quantitative variable, "non normal" the non parametric ones (median (IQR), Wilcoxon/Kruskal-Wallis), and "assess" decides variable by variable with a Shapiro-Wilk test at the 5% level.
<code>parallel</code>	Logical. Make analysis using parallel from <code>parallel::mclapply()</code> .
<code>mc.cores</code>	If parallel is <code>TRUE</code> , how many cores to use. The default, 0, uses all the cores but one.

Details

Only numeric, integer and factor (including ordered factor) variables are described. A variable of any other type is ignored with a warning, as is a variable whose test cannot be computed: its p value is then NA and the rest of the table is still produced.

Value

An S4 object of class `parseClass`. Its ["table"] element is a `data.frame` (whether or not there is a group, and also when data is a `tibble`) with a var column, one column per level of group, a Total column and, if there is a group and `pvalue = TRUE`, a pvalue column.

See Also

[parseClassFun\(\)](#) to turn the result into an HTML/LaTeX table.

Examples

```
# A small simulated clinical trial
set.seed(42)
n <- 200
patients <- data.frame(
  arm      = factor(sample(c("Placebo", "Treatment"), n, replace = TRUE)),
  age      = round(rnorm(n, mean = 65, sd = 10)),
  crp      = round(rlnorm(n, meanlog = 2, sdlog = 1), 1),
  sex      = factor(sample(c("Female", "Male"), n, replace = TRUE)),
  diabetes = factor(sample(c("No", "Yes"), n, replace = TRUE, prob = c(0.7, 0.3))),
  nyha     = factor(sample(c("I", "II", "III", "IV"), n, replace = TRUE),
                    ordered = TRUE)
)
patients$crp[sample(n, 15)] <- NA

# Compare the two arms. With normality = "assess", age (normal) is described
# by mean (SD) with a t test, crp (skewed) by median (IQR) with a Wilcoxon test
tab <- descTab(patients, group = "arm", normality = "assess", na.print = TRUE)
tab["table"]

# No group: a single Total column, no test
descTab(patients)["table"]

# Quantitative variables only, two decimals
descTab(patients, group = "arm", quali = FALSE, digits.qt = 2)["table"]

# Render it (see parseClassFun() for the layout options)
parseClassFun(tab)
```

ft_ana_biv

This function is deprecated, please use anaBiv(). [anaBiv\(\)](#)

Description

This function is deprecated, please use [anaBiv\(\)](#). [anaBiv\(\)](#)

Usage

```
ft_ana_biv(...)
```

Arguments

... None

Value

No return value, deprecated

ft_desc_tab	<i>This function is depreciated, please use <code>anaBiv()</code>. descTab()</i>
-------------	--

Description

This function is depreciated, please use `anaBiv()`. [descTab\(\)](#)

Usage

```
ft_desc_tab(...)
```

Arguments

...	None
-----	------

Value

No return value, depreciated

ft_parse	<i>This function is depreciated, please use parseClassFun()</i>
----------	---

Description

This function is depreciated, please use [parseClassFun\(\)](#)

Usage

```
ft_parse(...)
```

Arguments

...	None
-----	------

Value

No return value, depreciated

```
initialize,parseClass-method
```

S4 class initialization function

Description

Initialization function for parseClass object `initialize,parseClass-method()`

Usage

```
## S4 method for signature 'parseClass'
initialize(
  .Object,
  table,
  group,
  pvalue,
  na.print,
  quanti,
  quali,
  var_list,
  data,
  digits.qt,
  digits.ql
)
```

Arguments

<code>.Object</code>	The object to create
<code>table</code>	The result of <code>descTab</code>
<code>group</code>	The variable from which to make subgroups
<code>pvalue, na.print, quanti, quali</code>	Values from <code>descTab</code> <code>descTab()</code>
<code>var_list</code>	An object of listVar <code>listVar-class()</code>
<code>data</code>	The dataset provided in <code>descTab</code>
<code>digits.qt, digits.ql</code>	As provided in <code>descTab</code>

Value

parseClass object

initialize, Var-method *S4 class initialization function*

Description

Initialization function for Var [initialize, Var-method\(\)](#)

Usage

```
## S4 method for signature 'Var'
initialize(.Object, name, type, normal)
```

Arguments

.Object	Object to be initialized
name	A character taking name of the variable
type	A character taking name of the variable type
normal	Logical, if variable, is numeric; is it normal

Value

Var Object

initialize, VarGroup-method
S4 class initialization function

Description

Initialization function for VarGroup [initialize, VarGroup-method\(\)](#)

Usage

```
## S4 method for signature 'VarGroup'
initialize(
  .Object,
  x,
  group_var,
  pvalue,
  parsed_name,
  value,
  missing.value,
  missing.value.name
)
```

Arguments

.Object	Object to be initialized
x	A Var object
group_var	The subgroup for which proportions, mean/sd were calculated and missing values
pvalue	The calculated pvalue
parsed_name	The name of the variable parsed with the n (%), mean (SD)
value	The values calculated parsed
missing.value	Missing values numbers and proportions n (%)
missing.value.name	Missing values concatenate with the level of the variable if it factor

Value

VarGroup object

listVar-class	S4 class
---------------	----------

Description

A class of list of Var object

Slots

List a list of Var

parseClass	S4 class initialization function
------------	----------------------------------

Description

Initialization function for parseClass object [initialize,parseClass-method\(\)](#)

Usage

```
parseClass(
  table,
  group,
  pvalue,
  na.print,
  quanti,
  quali,
  var_list,
  data,
  digits.qt,
  digits.ql
)
```

Arguments

table	The result of descTab
group	The variable from which to make subgroups
pvalue, na.print, quanti, quali	Values from descTab descTab()
var_list	An object of listVar listVar-class()
data	The dataset provided in descTab
digits.qt, digits.ql	As provided in descTab

Value

parseClass object

parseClass-class	<i>S4 class</i>
------------------	-----------------

Description

A S4 class containing all the information needed for parsClassFun the missing values and the group for which it was calculated

Slots

table	The result of descTab
group	The variable from which to make subgroups
pvalue, na.print, quanti, quali	Values from descTab descTab()
var_list	An object of listVar listVar-class()
data	The dataset provided in descTab
digits.qt, digits.ql	As provided in descTab

parseClassFun	<i>Make the LaTeX/HTML table. Generic function</i>
---------------	--

Description

Make the LaTeX/HTML table. Generic function

Usage

```
parseClassFun(
  table,
  col.order = NULL,
  levels_to_keep = NULL,
  group_rows_labels = NULL,
  font = "arial"
)
```

Arguments

table	The output of descTab() or anaBiv() , an S4 object.
col.order	Optional. A vector containing the column order. If set, must contain at least all levels of group, without duplicates. Three columns created are "var", "Total", and "pvalue" which can be present in the vector
levels_to_keep	Optional, named list. If the variable is binary, which level to keep. Default is the last level of levels(variable). Must be as: list("variable name" = "level to keep").
group_rows_labels	Optional, named list. Create row labels in order to regroup them. Must be as list("label" = c("var1", "var2"), "label2" = c("var3", "var4")). Variable names are matched exactly, and a variable with no row in the table is an error. Rows are reordered to follow the order of the labels and of the variables within each label.
font	The HTML font of the table. Default "arial".

Value

An HTML/LaTeX file which can be used directly in Rmarkdown and copy paste

See Also

[descTab\(\)](#) which produces the object this function renders.

Examples

```
# A small simulated clinical trial
set.seed(42)
n <- 200
patients <- data.frame(
```

```

    arm      = factor(sample(c("Placebo", "Treatment"), n, replace = TRUE)),
    age      = round(rnorm(n, mean = 65, sd = 10)),
    crp      = round(rlnorm(n, meanlog = 2, sdlog = 1), 1),
    sex      = factor(sample(c("Female", "Male"), n, replace = TRUE)),
    diabetes = factor(sample(c("No", "Yes"), n, replace = TRUE, prob = c(0.7, 0.3))),
    nyha     = factor(sample(c("I", "II", "III", "IV"), n, replace = TRUE),
                      ordered = TRUE)
  )
  patients$crp[sample(n, 15)] <- NA

  tab <- descTab(patients, group = "arm", normality = "assess", na.print = TRUE)

  # Default layout
  parseClassFun(tab)

  # One line for the binary variables, rows grouped under labels, the Total
  # column after the arms, and a custom font
  parseClassFun(
    tab,
    levels_to_keep = list(sex = "Female", diabetes = "Yes"),
    group_rows_labels = list("Demographics" = c("age", "sex"),
                             "Clinical" = c("crp", "diabetes", "nyha")),
    col.order = c("var", "Placebo", "Treatment", "Total", "pvalue"),
    font = "Times New Roman"
  )

```

parseClassFun, parseClass-method

Make the LaTeX/HTML table

Description

This functions takes the S4 output of descTab to create an HTML parsed table

Usage

```

## S4 method for signature 'parseClass'
parseClassFun(
  table,
  col.order = NULL,
  levels_to_keep = NULL,
  group_rows_labels = NULL,
  font = "arial"
)

```

Arguments

table The output of `descTab()` or `anaBiv()`, an S4 object.

col.order	Optional. A vector containing the column order. If set, must contain at least all levels of group, without duplicates. Three columns created are "var", "Total", and "pvalue" which can be present in the vector
levels_to_keep	Optional, named list. If the variable is binary, which level to keep. Default is the last level of levels(variable). Must be as: list("variable name" = "level to keep").
group_rows_labels	Optional, named list. Create row labels in order to regroup them. Must be as list("label" = c("var1", "var2"), "label2" = c("var3", "var4")). Variable names are matched exactly, and a variable with no row in the table is an error. Rows are reordered to follow the order of the labels and of the variables within each label.
font	The HTML font of the table. Default "arial".

Value

An HTML/LaTeX file which can be used directly in Rmarkdown and copy paste

See Also

[descTab\(\)](#) which produces the object this function renders.

Examples

```
# A small simulated clinical trial
set.seed(42)
n <- 200
patients <- data.frame(
  arm      = factor(sample(c("Placebo", "Treatment"), n, replace = TRUE)),
  age      = round(rnorm(n, mean = 65, sd = 10)),
  crp      = round(rlnorm(n, meanlog = 2, sdlog = 1), 1),
  sex      = factor(sample(c("Female", "Male"), n, replace = TRUE)),
  diabetes = factor(sample(c("No", "Yes"), n, replace = TRUE, prob = c(0.7, 0.3))),
  nyha     = factor(sample(c("I", "II", "III", "IV"), n, replace = TRUE),
                    ordered = TRUE)
)
patients$crp[sample(n, 15)] <- NA

tab <- descTab(patients, group = "arm", normality = "assess", na.print = TRUE)

# Default layout
parseClassFun(tab)

# One line for the binary variables, rows grouped under labels, the Total
# column after the arms, and a custom font
parseClassFun(
  tab,
  levels_to_keep = list(sex = "Female", diabetes = "Yes"),
  group_rows_labels = list("Demographics" = c("age", "sex"),
                           "Clinical" = c("crp", "diabetes", "nyha")),
  col.order = c("var", "Placebo", "Treatment", "Total", "pvalue"),
  font = "Times New Roman"
```

)

Var	<i>S4 class initialization function</i>
-----	---

Description

Initialization function for Var `initialize,Var-method()`

Usage

`Var(name, type = "", normal = TRUE)`

Arguments

- name A character taking name of the variable
- type A character taking name of the variable type
- normal Logical, if variable, is numeric; is it normal

Value

Var Object

Var-class	<i>S4 class</i>
-----------	-----------------

Description

A S4 class containing name, type and normality assessment of variable

Slots

- name A character taking name of the variable
- type A character taking name of the variable type
- normal Logical, if variable, is numeric; is it normal

VarGroup-class	S4 class
----------------	----------

Description

A S4 class containing Var `initialize,Var-method()` It also contains the pvalue, the parsed value the missing values and the group for which it was calculated

Slots

- group_var The subgroup for which proportions, mean/sd were calculated and missing values
- pvalue The calculated pvalue
- parsed_name The name of the variable parsed with the n (%), mean (SD)
- value The values calculated parsed
- missing.value Missing values numbers and proportions n (%)
- missing.value.name Missing values concatenate with the level of the variable if it factor

[,parseClass-method	Method to access S4 Var elements
---------------------	----------------------------------

Description

Method to access parseClass `initialize,parseClass-method()` elements by name

Usage

```
## S4 method for signature 'parseClass'  
x[i]
```

Arguments

- x : Object
- i : Element name

Value

object

[,Var-method	Method to access S4 Var elements
--------------	----------------------------------

Description

Method to access Var elements by name

Usage

```
## S4 method for signature 'Var'  
x[i]
```

Arguments

- x : object
- i : value

Value

object of Var

[,VarGroup-method	Method to access S4 Var elements
-------------------	----------------------------------

Description

Method to access VarGroup [initialize,VarGroup-method\(\)](#) elements by name

Usage

```
## S4 method for signature 'VarGroup'  
x[i]
```

Arguments

- x : object
- i : value

Value

object element

[<-, parseClass-method *Method to modify S4 Var elements*

Description

Method to modify parseClass `initialize, parseClass-method()` elements by name

Usage

```
## S4 replacement method for signature 'parseClass'
x[i] <- value
```

Arguments

x	: Object
i	: Element name
value	: Value to be added

Value

parseClass Object

[<-, Var-method *Method to access S4 Var elements*

Description

Method to modify Var elements by name

Usage

```
## S4 replacement method for signature 'Var'
x[i] <- value
```

Arguments

x	: object
i	: Element name
value	: Value to be added

Value

object

[<- , VarGroup-method Method to access S4 Var elements

Description

Method to modify VarGroup `initialize, VarGroup-method()` elements by name

Usage

```
## S4 replacement method for signature 'VarGroup'  
x[i] <- value
```

Arguments

x	Object
i	Element name
value	Value to be added

Value

object

Index

[,Var-method, [17](#)
[,VarGroup-method, [17](#)
[,parseClass-method, [16](#)
[<-,Var-method, [18](#)
[<-,VarGroup-method, [19](#)
[<-,parseClass-method, [18](#)

anaBiv, [2](#)
anaBiv(), [6](#), [12](#), [13](#)
anaBiv,data.frame,character-method, [3](#)
anaBiv,listVar,character-method, [4](#)

descTab, [4](#)
descTab(), [2-4](#), [7](#), [8](#), [11-14](#)

ft_ana_biv, [6](#)
ft_desc_tab, [7](#)
ft_parse, [7](#)

initialize,parseClass-method, [8](#)
initialize,Var-method, [9](#)
initialize,VarGroup-method, [9](#)

listVar-class, [10](#)

parallel::mclapply(), [2-5](#)
parseClass, [10](#)
parseClass-class, [11](#)
parseClassFun, [12](#)
parseClassFun(), [5](#), [7](#)
parseClassFun,parseClass-method, [13](#)

Var, [15](#)
Var-class, [15](#)
VarGroup-class, [16](#)