Package 'lazyWeave'

April 3, 2025

Type Package

Title LaTeX Wrappers for R Users

Version 3.0.3

Maintainer Benjamin Nutter <benjamin.nutter@gmail.com>

Description Provides the functionality to write LaTeX code from within R without having to learn LaTeX. Functionality also exists to create HTML and Markdown code. While the functionality still exists to write complete documents with lazyWeave, it is generally easier to do so with with markdown and knitr. lazyWeave's main strength now is the ability to design custom and complex tables for reporting results.

License GPL

LazyLoad yes

Depends R (>= 2.10.0)

Imports Hmisc, labelVector

Suggests testthat

RoxygenNote 7.3.2

NeedsCompilation no

Author Benjamin Nutter [aut, cre]

Repository CRAN

Date/Publication 2025-04-03 12:20:02 UTC

Contents

`ComparisonTable` *Comparison Tables*

Description

Produce a table of comparisons for reports and manuscripts

Usage

```
catconttable(
  data,
  vars,
  byVar,
  vars.cat = NULL,
  fisher = NULL,
  fisher.arg = NULL,
  cmh = NULL,
  row.score = NULL,
  col.score = NULL,
  normal = NULL,
```

```
  var.equal = NULL,
  median = NULL,
  odds = NULL,
  odds.scale = NULL,
  odds.unit = NULL,
  none = NULL,
  row.p = TRUE,
  alpha = 0.05,
  B = 1000,
  seed = NULL
)

cattable(
  data,
  vars,
  byVar,
  fisher = NULL,
  fisher.arg = NULL,
  cmh = NULL,
  row.score = NULL,
  col.score = NULL,
  mcnemar = NULL,
  correct = NULL,
  none = NULL,
  odds = NULL,
  row.p = TRUE,
  alpha = 0.05,
  minl = 5
)

conttable(
  data,
  vars,
  byVar,
  normal = NULL,
  var.equal = NULL,
  median = NULL,
  none = NULL,
  odds = NULL,
  odds.scale = NULL,
  odds.unit = NULL,
  alpha = 0.05,
  B = 1000,
  seed = NULL
)
```

Arguments

data	A ccf.df or data.frame with the variables to be compared and the grouping variable.
vars	A character vector naming the variables to be compared
byVar	A character(1) giving the grouping variable. This allows more than one level. Numeric variables are coerced to factors.
vars.cat	A character vector that can be used to specify which, if any, numeric variables in vars should be treated as categorical.
fisher	A character vector giving the names of variables that should be compared with Fisher's Exact test. Currently, there is no implementation to determine this automatically.
fisher.arg	A list of additional arguments to pass to fisher.test
cmh	A character vector giving the names of variables that should be compared with Manthel-Haenszel's Test for Linear Trend. This is not yet written and will be ignored.
row.score	Currently ignored
col.score	Currently ignored
normal	A character vector that assigns variables in vars as normally distributed.
var.equal	A character vector that assigns variables in vars as having equal variance. This is used to determine the proper form of a t-test.
median	A character vector that assigns variables that shoudl be summarized with a median, quartiles, or min and max.
odds	A character vector giving the names of variables for which odds ratios should be calculated. For categorical measures, this is the primary test of comparison. For numeric measures, this is calculated in addition to another test.
odds.scale	For numeric variables only. A list with named elements that gives the scale on which the odds ratio should be presented. For example, if the odds for variable x should be presented in 5 year increments, we would use odds.scale=list(x = 5).
odds.unit	For numeric variables only. A list with named elements that gives the units on which the odds ratio should be presented. For example, if the odds of variable x should be presented in 5 year increments, we would use odds.unit=list(x="years").
none	A character vector naming variables in vars for which no comparison should be made.
row.p	Toggles if row or column proportions are calculated.
alpha	Significance levels for tests.
B	The number of Bootstrap samples for bootstrapped confidence intervals.
seed	The seed to use in starting the Bootstrapping.
mcnemar	a character vector giving the names of variables that should be compared using McNemar's test.
correct	Character vector giving the variables for which a continuity correction should be applied in McNemar's test.

`minl` Minimum length for levels abbreviations. The function `abbreviate` is used to create unique rownames for each level of a variable in the output data frame. If the abbreviations are short, they may not be readable. This allows the user to make the length longer.

Details

`catconttable` is a wrapper that determines the type of variable and calls either cattable or conttable as appropriate. For this to work properly, all factor variables must be defined before the function call.

In contrast, if cattable is called directly, variables are coerced to factors, which could lead to peculiar results if a numeric value is given.

Author(s)

Benjamin Nutter

See Also

`write.ctable`

Examples

```
#Read in the delivery dataset from the lazyWeave package
data(Delivery)

#Use conttable to summarize maternal age, ga weeks, weight (grams)
#and grava by delivery type.  The dataset name is specified under the "data="
#option, the variables of interest are listed under "vars=", and the K-level by variable
#is specified under "byVar=".

#Default is to report mean and bootstrapped 95% CI for mean.  Tests of location are by
#default either Wilcoxon rank sum (K=2) or Kruskal-Wallis (K>2) rank sum.  The "seed="
#option allows for reproducibility by setting the seed for getting bootstrapped samples.

d_type.contable <- conttable(data=Delivery,
                             vars=c("maternal.age", "ga.weeks", "wt.gram", "grava"),
                                    byVar="delivery.type")

#Specifying weights by delivery type as a normally distributed variables, reports means,
#standard deviations and a t-test of equality of the means for delivery type.  Variables listed
#under "var.equal=" are assumed to have equal variances in all levels of byVar.  Otherwise,
#variances are allowed to be unequal.

d_type.conttable <- conttable(data=Delivery,
                         vars=c("maternal.age", "ga.weeks", "wt.gram", "grava", "apgar1"),
                               byVar="delivery.type",
                               normal=c("wt.gram", "maternal.age"),
                               var.equal="ga.weeks")

#List variables under "median=" to report median, 25th and 75th percentiles.
```

```
d_type.conttable <- conttable(data=Delivery,
                        vars=c("maternal.age", "ga.weeks", "wt.gram", "grava", "apgar1"),
                              byVar="delivery.type",
                              normal=c("wt.gram", "maternal.age"),
                              var.equal="ga.weeks",
                              median=c("grava","apgar1"))

#Use cattable to summarize child sex, laceration, and laceration degree by delivery type.
#Row percent, overall counts, and counts by delivery type are reported.  Column percents can
#be specified by the row.p=FALSE option.
#By default chi-square tests of independence are performed.

d_type.cattable <- cattable(data=Delivery,
                            vars=c("child.sex", "laceration"),
                            byVar="delivery.type")

#For variables listed under "fisher=" Fisher's exact test of independence is performed.
#The reported test statistic is the odds ratio.

d_type.cattable <- cattable(data=Delivery,
                            vars=c("child.sex", "laceration"),
                            fisher=c("child.sex"),
                            byVar="delivery.type")


#All variables listed in a single table

d_type.catconttable <- catconttable(data=Delivery,
                              vars=c("maternal.age", "ga.weeks", "child.sex", "wt.gram",
                                          "grava", "apgar1", "laceration"),
                                    median=c("grava", "apgar1"),
                                    normal="maternal.age",
                                    fisher="child.sex",
                                    byVar="delivery.type")

## Not run:
  #Code for writing ctable objects to a file.  See write.ctable() for more information

  #Write to PDF
  options(lazyReportFormat='latex')
  lazy.write(
    lazy.file.start(),
    write.ctable(d_type.catconttable),
    lazy.file.end(),
    OutFile="SampleOutput.tex")

#Generate a pdf in the working directory
  lazy.build("SampleOutput.tex")

  unlink("SampleOutput.tex")
  unlink("SampleOutput.pdf")

## End(Not run)
```

`Delivery` *Obstetric Delivery Log*

Description

Description of 100 hypothetical deliveries from a midwestern hospital.

These data are simulated in a way to be similar to data collected from a midwestern hospital. However they are not actual clinical data.

Usage

```
data(Delivery)
```

Format

A data frame with 100 rows and 15 variables.

maternal.id	Mother's ID number (repeats for twins)
child.id	Child ID; given by appending a letter to the maternal id. The letter represents the birth order when multipl
maternal.age	Mother's age at time of delivery.
grava	Total number of pregnancies the mother has experienced (including current pregnancy)
para	Total number of prior live births
ga.weeks	Gestational age; the whole number of weeks the mother has been pregnant.
ga.days	Gestational age; a number between 0 and 6. This may be added to `ga.weeks` to get the total gestational ag
delivery.type	Denotes if the delivery was vaginal or cesarean.
child.sex	Biological sex of the child
vaginal.type	Denotes if a vaginal delivery was spontaneous (natural) or instrument assisted
wt.gram	Birth weight in grams
apgar1	1 minute APGAR score
apgar5	5 minute APGAR score
laceration	Binary indicator for laceration. 0 = No, 1 = Yes. Only applies to vaginal deliveries
laceration.degree	Description of the severity of laceration.

Source

Simulation of Midwestern Hospital Delivery Log.

`is_significant` *Test the significance of a p-value*

Description

Test if a p-value is significant. This is specifically designed to handle numeric output, or output from `pvalString`

Usage

```
is_significant(pvalue, alpha = 0.05)
```

Arguments

`pvalue`	The p-value to be tested
`alpha`	The significance level for comparison

Details

In instances where `pvalue` has a leading '<' or '>', the inequality is stripped and the remaining characters are coerced to a numeric. A logical vector comparing pvalue to alpha is returned where the value is `TRUE` if `pvalue <= alpha`

This function was built with the intent of using it to identify rows in descriptive tables (such as `cattable` and `conttable`) with significant results. These rows could then be highlighted using bold print automatically. This might prove useful for large tables.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
is_significant(c(.10, .06, .051, .05, .049, .02, .01))
is_significant(c("> .10", "< .05", "< 0.001"), alpha=.01)

## End(Not run)
```

`lazy.build` *Compile a PDF or HTML Report*

Description

Executes a command to build a pdf file based on a .tex or .html file. For HTML files, compiles the figure into a subfolder and places all of the contents into a zip file

Usage

```
lazy.build(
  filename,
  pdf.zip = NULL,
  quiet = TRUE,
  clean = TRUE,
  replace = TRUE,
  ...
)
```

Arguments

`filename`	Character string giving the location of the file to be built. Usually a .tex or .html file.
`pdf.zip`	filename where the completed document should be saved. Usually a .pdf or .zip file.
`quiet`	Sets the system call flag for a quiet (non-verbose) run.
`clean`	Sets the system call for cleaning up the folder after completion. If `TRUE`, extra files generated during processing will be deleted.
`replace`	when `pdf.zip` is not `NULL`, this determines if the new file will overwrite an existing file.
`...`	Additoinal arguments to be passed to `tools::texi2dvi`

Details

For TEX files, a call is made using `tools::texi2dvi` to compile a PDF file.

For HTML files, the referenced figures are gathered, copied into a subdirectory and the HTML document and the figures are placed into a zip folder for convenient transfer. All of the image links in the HTML code are modified to reflect the new location. Lastly, a text file is added with instructions for unzipping the files for convenient viewing (but don't worry, no one ever reads this).

Author(s)

Benjamin Nutter

`lazy.citation` *Add R or R Package Citation*

Description

Generates code for the citation of R or an R package.

Usage

```
lazy.citation(
  pkg = NULL,
  author = TRUE,
  title = TRUE,
  org = TRUE,
  address = TRUE,
  volume = TRUE,
  year = TRUE,
  note = TRUE
)
```

Arguments

`pkg`	a character(1) vector giving the name of a package. If `NULL`, a citation for R is produced.
`author`	Include author name
`title`	Include title of package
`org`	Include organization name
`address`	Include address
`volume`	Include volume
`year`	include year of publication
`note`	include the note on the citation.

Details

Not every option is populated in every package. Future improvements might include automatic detection of NULL fields, but for now, observing the output with all the options set to `TRUE` will tell you which ones are empty.

Author(s)

Benjamin Nutter

Examples

```
lazy.citation()
lazy.citation(pkg="lazyWeave", org=FALSE, address=FALSE, volume=FALSE)
```

`lazy.counter` *Create and Manage Counters for LaTeX Documents*

Description

Provides the code to create, manipulate, or extract values of counters in a LaTeX or HTML document

Usage

```
lazy.counter(
  counter,
  value,
  oldcounter,
  fn = c("new", "addto", "set", "use", "value")
)
```

Arguments

`counter`	A character(1) giving the name of the counter to be created and/or manipulated
`value`	An integer. For `fn="addto"`, it is the value by which the counter is to be increased. For `fn="set"`, it is the value to which the counter should be set
`oldcounter`	character(1). An optional argument for `fn="new"`. It must be a previously named counter. If present, the new counter will be reset whenever `oldcounter` is incremented
`fn`	Selects the LaTeX function to be used

Details

Counters are used to provide table, figure, and section numbers. After each use of each command, the counter is incremented so that it can be referred to in later uses. New counters may be defined by users, but several LaTeX environments have default counters that do not need to be defined. These are `part`, `chapter`, `section`, `subsection`, `subsubsection`, `paragraph`, `subparagraph`, `page`, `equation`, `figure`, `table`, `footnote`, `mpfootnote`. Any of these may be manipulated by `lazyWeave`.

Referring to and manipulating counters is done using `lazy.counter`. Different actions are achieved by changing the `fn` argument.

`fn="new"` creates a new counter with name `counter`.

`fn="addto"` adds `value` to the current value of `counter`.

`fn="set"` changes the current value of `counter` to `value`.

`fn="use"` designates `counter` for use in the current environment.

`fn="value"` returns the value of `counter`. This value isn't printed, but can be useful for doing arithmetic with counter values.

The HTML counters in `counter` are the defaults and will reference global counters

```
options("html.counter.table")
 options("html.counter.table"
 options("html.counter.table"
 options("html.counter.table"
 options("html.counter.table"
 options("html.counter.table"
 options("html.counter.table"
```

Additional and custom counters may be defined if desired, in which case a new option will be defined as `options("html.custom.[countername]")`.

Extracting a current value does not increment the counter–this must be done manually (and is done manually when used in `lazy.table`, `lazy.figure`,\ `lazy.footnote`, and `lazy.section`.

Author(s)

Benjamin Nutter

Examples

```
lazy.counter("myCounter")
lazy.counter("myCounter", value=3, fn="set")
lazy.counter("myCounter", value=2, fn="addto")
lazy.counter("myCounter", fn="use")
lazy.counter("myCounter", fn="value")

lazy.counter("table", fn="use")
```

---

lazy.figure *Include Figures in Latex Documents*

Description

Generates the code to place a figure in a Latex document

Usage

```
lazy.figure(
  filename,
  caption = NULL,
  align = "center",
  height = 3,
  width = 3,
  units = "in",
  counter,
  counterSet = NULL,
  label = NULL,
  placement = "h",
```

```
  alt = "Image Not Found",
  cat = getOption("lazyWeave_cat")
)
```

Arguments

`filename`	Character string giving the location of the file to be included
`caption`	Text giving the caption for the figure
`align`	Character string stating the alignment. Valid options are `"left"`, `"right"`, or `"center"`
`height`	The height of the figure
`width`	The width of the figure
`units`	The units for height and width. Defaults to `"in"`.
`counter`	Name of a counter to use to number the table
`counterSet`	The number to which `counter` should be set. In other words, the figure number for this figure
`label`	Name of a label
`placement`	Controls the placement of the figure. Options are `"ht"`, `"t"`, `"b"`, `"p"`, `"H"` and can be supplemented with `"!"`. See "Details" for more explanation
`alt`	For HTML documents only–when `filename` cannot be found, this text is printed in the figure's place
`cat`	Logical. Determines if the output is returned as a character string or returned via the `cat` function (printed to console). The default value is set by `options()$lazyWeave_cat`. This argument allows for selective override of the default.

Details

For LaTeX files, `placement` options are used as follows:

ht	Place the float here, i.e., approximately at the same point it occurs
t	Position at the top of the page
b	Position at the bottom of the page
p	Put on a special page for floats only
H	Places the float at precisely the location in the LaTeX code. Requires the float package

The `"!"` may be used after any of these in order to override LaTeX float rules and force your selection. More can be learned by reading about floats in a LaTeX manual.

For HTML files, the file can be any type supported by the browser. JPEGs and PNGs seem to work well.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
pdf("MPG.pdf", height=4, width=4)
hist(mtcars$mpg)
dev.off()

lazy.figure("MPG.pdf")

lazy.write(
lazy.file.start(),
lazy.figure("MPG.pdf",
caption="Distribution of Miles per Gallon in mtcars dataset",
height=5, width=5, label="MPGgraph"),
lazy.file.end(),
OutFile="Example-1.tex")

unlink("MPG.pdf")
unlink("Example-1.tex")
unlink("Example-1.pdf")

## End(Not run)
```

---

lazy.file.end *End LaTeX Documents*

Description

Provides the code to end a LaTeX document

Usage

```
lazy.file.end()
```

Author(s)

Benjamin Nutter

Examples

```
lazy.file.end()
```

`lazy.file.start` *Initiate New LaTeX, HTML, or Markdown Files*

Description

Write code to open a new LaTeX document with packages, classes, a title, and page settings

Usage

```
lazy.file.start(
  docClass = "article",
  packages = NULL,
  counters = NULL,
  layout = "",
  page = "arabic",
  ligatures = TRUE,
  title = NULL,
  author = NULL,
  date = "",
  initialize = TRUE
)
```

Arguments

`docClass`	a character string giving a valid LaTeX document class. For example, `article`, `slide`, `report`, `book`
`packages`	A character vector of additional LaTeX packages to use
`counters`	A character vector of additional counters to initialize
`layout`	LaTeX code for page layout. Remember to escape backslashes!
`page`	A character string denoting the page numbering style. Options are `"arabic"`, `"roman"`, `"Roman"`, `"alph"`, `"Alph"`.
`ligatures`	Determines if ligatures are enabled. See the references for a link about ligatures
`title`	A title for the document
`author`	Author of the document
`date`	Date to be printed on the title page
`initialize`	For HTML and markdow files and when `TRUE`, the function `lazy.options` is called and all of the counters are reset to 1. Font, family, and size defaults are also reset

Details

Titles are only made when either `title` or `author` are not `NULL`.

Packages automatically included are "xcolor", "graphicx", "colortbl", "float", "soul", "hyperref", "placeins", and "Sweave". Any user defined templates made in conjuction with `lazyWeave` must include these packages in order to use figures and underlined text.

With page, the options produce the following:

arabic	Arabic numbers
roman	Lower case roman numerals (i, ii, iii, ...)
Roman	Upper case roman numerals (I, II, III, ...)
alph	Lower case alphabetic ordering (a, b, c, ...)
Alph	Upper case alphabetic ordering (A, B, C, ...)

Author(s)

Benjamin Nutter

References

Ligatures: https://en.wikibooks.org/wiki/LaTeX/Text_Formatting#Ligatures

Examples

```
#* lazy.file.start does not currently work with markdown documents
#* First, we set the lazyReportFormat option to "latex"
orig_option <- getOption("lazyReportFormat")
options(lazyReportFormat="latex")
lazy.file.start(docClass="report",
  packages=c("pslatex", "palatino", "avant"),
  title="Report Name", author="Your Name")

#* Return the original option setting
options(lazyReportFormat=orig_option)
```

---

lazy.footnote *Add a Footnote*

Description

Adds a footnote to the text in a .tex file. In HTML files, an endnote is produced. Links are established between the text and the endnote for user convenience

Usage

```
lazy.footnote(
  text,
  number = NULL,
  translate = FALSE,
  name,
  ref,
  counter = "footnote",
  size = 8
)
```

Arguments

text	Text for the footnote
number	Footnote number
translate	Determines if latexTranslate is applied to text.
name	For HTML, a reference name to the endnote
ref	For HTML, a reference name to go back to the text (from the endnote).
counter	For HTML, counter to use for numbering the endnotes
size	For HTML, the text size to use for the endnote

Details

Be warned that this is not a perfect function on the LaTeX side. If you use special characters that require that latexTranslate be turned off, you'll also need to turn of latexTranslate in lazy.write. I'm including this as is for ease of use, but it does have some kinks to work out.

name and ref are used to create links between the footnote marking in the text and the actual footnote. Clicking on the links created will allow the reader to go back and forth between them. The names may be similar, but exact matches may confuse the browser.

Author(s)

Benjamin Nutter

Examples

```
lazy.footnote("Enter a reference to an article in this argument", number=3)
lazy.footnote(lazy.citation(), number=4)
```

---

lazy.insert.code *Insert Code to a TeX or HTML document*

Description

Places TeX code from a file into a TeX document. Text is placed within a verbatim environment. For HTML documents, a verbatim-like environment is created

Usage

```
lazy.insert.code(file, prompt.symbol = options()$prompt, lines)
```

Arguments

file	Filename containing the code to be inserted
prompt.symbol	Character to be placed at the left most side of the page. Defaults to the system's current prompt symbol
lines	A vector giving the lines in file to be inserted into the document

Details

Text is inserted in a verbatim environment to preserve whitespace. This function is performed better by `Sweave` and `knitr`. Those packages will also display any result printed by the code. This function will not display results.

With HTML, the font family is set to monospace, and the font is set to courier. All of the spaces are replaced with a non-breaking space to preserve the white space in the code.

Author(s)

Benjamin Nutter

`lazy.label` *Reference Tables, Figures, Sections, and Pages*

Description

Provides the code to label and reference objects that have a label statement with them

Usage

```
lazy.label(label)

lazy.ref(label, text, page = FALSE, link = TRUE)
```

Arguments

`label`	A character(1) giving the name of the to be created or referenced
`text`	For HTML, the text to be hyperlinked for the reference. If missing, this is set to `"(link)"`
`page`	Indicates if the page number on which the label lies should be returned or the object number. This only applies to LaTeX files
`link`	for LaTeX files, should the reference link to the object

Author(s)

Benjamin Nutter

Examples

```
lazy.label("Label1")
lazy.ref("Label1")
lazy.ref("Label1", page=TRUE)
```

lazy.landscape *Landscape Page Orientation*

Description

Allows the user to turn on or off landscape page orientation

Usage

```
lazy.landscape(begin = TRUE)
```

Arguments

begin	If TRUE, begins the landscape environment. Otherwise, the environment is closed

Details

This function has no effect on HTML or markdown files. The page orientation is determined by the browser

Value

Returns a string that either begins or ends a landscape environment

Author(s)

Benjamin Nutter

Examples

```
lazy.landscape()
lazy.landscape(begin=FALSE)
```

lazy.link *Links to Webpages or External Documents*

Description

While lazy.ref provides the option of linking to areas within a document, lazy.link provides the option of linking to areas outside of the document. Web pages are perhaps the most obvious example, but links could also go to files on a directory

Usage

```
lazy.link(url, text, web = TRUE, secure = FALSE)
```

Arguments

url	A character(1) giving the URL for the link or a file path
text	The text to be highlighted as the link. If this is missing, url is used
web	When TRUE, "http://" is added to url, (if not already present), to ensure a link to the web. For files on a local dis, set this to FALSE
secure	Should the link be to a secure site "https://".

Author(s)

Benjamin Nutter

Examples

```
lazy.link("https://github.com/nutterb/lazyWeave", secure=TRUE)
```

---

lazy.list *Lists in LaTeX and HTML*

Description

Produce code for lists in LaTeX documents

Usage

```
lazy.list(
  item,
  ordered = TRUE,
  counter = NULL,
  counterSet = 1,
  title = NULL,
  style = c("arabic", "Roman", "roman", "Alph", "alph"),
  symbol = c("bullet", "circ", "blacksquare"),
  font,
  family,
  size
)
```

Arguments

item	A vector with the items to be placed in the list
ordered	Denotes if the list is ordered or bulleted
counter	For future inclusion. Specifies what counter should be used for numbering. Currently not in use
counterSet	The value to which counter should be set. In other words, the number for the first item in the list

title	A title for the list
style	A character string denoting how the ordered list should be numbered. Options are "arabic", "roman","Roman", "alph", "Alph".
symbol	A symbol for bulleted lists to be used as the bullet
font	Font to be used in HTML documents. Defaults to the font set in the options
family	Font family to be used in HTML documents. Defaults to the font family in the options
size	Font size to be used in HTML documents. Defaults to the font family in the options

Details

With style, the options produce the following and apply to both LaTeX and HTML:

arabic	Arabic numbers
roman	Lower case roman numerals (i, ii, iii, ...)
Roman	Upper case roman numerals (I, II, III, ...)
alph	Lower case alphabetic ordering (a, b, c, ...)
Alph	Upper case alphabetic ordering (A, B, C, ...)

The given options for symbol follow the HTML conventions, but when options("lazyReportFormat") is "latex", these options are translated into the appropriate equivalent.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
lazy.write(
  lazy.file.start(),
  lazy.text("A vector can be presented as a list as follows"),
  lazy.list(c("First item", "Second item", "Third item",
              "Fourth item", "Fifth item"), style="Alph"),
  lazy.file.end(),
  OutFile="Example 1.tex")

unlink("Example 1.tex")

## End(Not run)
```

lazy.matrix *Convert Matrix to LaTeX Table*

Description

An example of using lazyWeave to produce tables

Usage

```
lazy.matrix(
  x,
  align = "center",
  justify = "center",
  rcol = NULL,
  usecol = "lightgray",
  caption = NULL,
  footnote = NULL,
  placement = "h",
  translate = TRUE,
  cat = getOption("lazyWeave_cat"),
  ...
)
```

Arguments

x	A matrix. Other objects are coerced to matrices
align	Character vector or string giving the alignment for each column of the table. Options are "left", "center", "right".
justify	Character string giving the alignment for the table on the page. Options are "left", "center", "right".
rcol	A vector giving the rows of x to be colored
usecol	A character string or vector denoting the color of the rows in rcol
caption	Caption for the table. This is printed above the table
footnote	Additional footnotes for the table. These are printed below the table.
placement	Controls the placement of the figure. Options are "ht", "t", "b", "p", "H" and can be supplemented with "!". See "Details" for more explanation. These apply only to LaTeX
translate	Toggles if inputs in x should be passed through latexTranslate. This should be set to FALSE if writing custom code
cat	Logical. Determines if the output is returned as a character string or returned via the cat function (printed to console). The default value is set by options()$lazyWeave_cat. This argument allows for selective override of the default.
...	Additional arguments to be passed to lazy.table

Details

The output for `lazy.matrix` is highly inflexible compared to `lazy.table`. It is an example of how to build a reproducible table with a certain formatting style that may be used again and again for consistency.

Row names are always left justified. This cannot be changed.

`placement` options are used as follows:

ht	Place the float here, i.e., approximately at the same point it occurs
t	Position at the top of the page
b	Position at the bottom of the page
p	Put on a special page for floats only
H	Places the float at precisely the location in the LaTeX code. Requires the float package

The `"!"` may be used after any of these in order to override LaTeX float rules and force your selection. More can be learned by reading about floats in a LaTeX manual

Author(s)

Benjamin Nutter

Examples

```
## Not run:
lazy.write(
  lazy.file.start(),
  lazy.text("The mtcars dataset describes a number of vehicles.
      Let's take a look at the data"),
  lazy.matrix(mtcars, rcol=(1:nrow(mtcars))[c(FALSE, TRUE)]),
  lazy.file.end(),
  OutFile="Example 1.tex")

unlink("Example 1.tex")

## End(Not run)
```

---

lazy.page.break *Start New Page in LaTeX*

Description

Insert code to start a new page in a document

Usage

```
lazy.page.break()
```

Details

For HTML documents, page breaks will not show up in the browser, but will show up when printed.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
lazy.write(
lazy.file.start(),
lazy.text("First we type something on the first page"),
lazy.page.break(),
lazy.text("Then we type something on the second page"),
lazy.file.end(),
OutFile="Example 1.tex")

unlink("Example 1.tex")

## End(Not run)
```

---

lazy.page.number *Change Page Numbering*

Description

Allows page numbering style to be changed. For instance, from roman numerals in an introduction to arabic numbers in the body of a report

Usage

```
lazy.page.number(num_style = c("arabic", "roman", "Roman", "alph", "Alph"))
```

Arguments

num_style	A character(1) giving the numbering style for the page

Details

Each time the page numbering is changed, the page counter is reset to 0 (meaning the next page will be numbered 1). If the number needs to be preserved, this can be done using `lazy.counter` with `counter="page"`.

Author(s)

Benjamin Nutter

Examples

```
lazy.page.number("Roman")
```

`lazy.section` *Open Sections in LaTeX Documents*

Description

Provides the code to start a new section, chapter, or subsection.

Usage

```
lazy.section(
  heading,
  type = c("section", "sub", "subsub", "chapter", "sub2"),
  ordered = FALSE,
  counter,
  counterSet = NULL,
  label = NULL,
  font = getOption("html.font.font"),
  family = getOption("html.font.family"),
  size = getOption("html.font.size"),
  leadspace = TRUE,
  floatBarrier = TRUE
)
```

Arguments

`heading`	The name of the section or subsection
`type`	Type of section. `"section"` for section; `"sub"` for subsection; and `"subsub"` for subsubsection. The option `"sub2"` is a relic of me trying to avoid typing two extra characters. I decided it would be better to keep true to the LaTeX descriptors. Thus, `"sub2"` is available for back-compatibility, but I recommend against its use
`ordered`	Logical. Determines if the section is numbered
`counter`	Name of a the counter to be used for this section. When `NULL`, the value defaults to the counter corresponding to the type of section. See `lazy.counter` for more details about counters
`counterSet`	Value to which `counter` should be set. In other words, the number for this section (or similar division).
`label`	The label to be used with `lazy.ref`.
`font`	Font to be used in HTML files
`family`	Font family to be used in HTML files

size	Font size. I'm pretty sure this doesn't actually get used, but haven't gotten around to verifying this. Heading sizes are set using the HTML <H ... > tags
leadspace	For HTML reports, should several lines of white space be placed before the section header. This helps to create a visual break between sections
floatBarrier	Figures and tables in LaTeX are floating objects that LaTeX may choose to place somewhere other than where specified in order to optimize appearance. This is not always desirable. floatBarrier prevents floats from overlapping a section break. This may be turned off if placement of figures and tables is of little consequence

Details

For HTML, Sections titles are printed using the <Hx ...> tags, where x is the heading number. H1 is the largest heading size and H6 is the smallest. Chapter headings use <H2>; sections use <H3>; subsections use <H4>; and subsubsections use <H5>.

Examples

```
## Not run:
  lazy.write(
  lazy.file.start(),
  lazy.section("Section A", ordered=TRUE),
  lazy.text("Notice that Section A is numbered"),
  lazy.section("Subsection", type="sub", ordered=FALSE),
  lazy.text("But the subsection is not numbered"),
  lazy.file.end(),
  OutFile="Example-1.tex")

  unlink("Example-1.tex")

## End(Not run)
```

---

lazy.table *Tables in LaTeX*

Description

Generate code for custom LaTeX tables

Usage

```
lazy.table(
  x,
  align = "center",
  cspan = 1,
  cwidth = NULL,
  cwidth.units = "in",
```

```
  cborder = NULL,
  cborder.thick = 1,
  cborder.style = "solid black",
  rborder = NULL,
  rbspan = NULL,
  rborder.thick = 1,
  rborder.style = "solid black",
  rcol = NULL,
  usecol = "lightgray",
  font,
  family,
  size,
  justify = "center",
  placement = "h",
  open = TRUE,
  close = TRUE,
  caption = NULL,
  footnote = NULL,
  label = NULL,
  counter = NULL,
  counterSet = NULL,
  translate = TRUE,
  cat = getOption("lazyWeave_cat")
)
```

Arguments

x	Matrix to be put into the table. Other objects are coerced to matrices. Vectors are coerced to a row vector
align	Character vector or string giving the alignment for each column. Options are "left", "center", "right".
cspan	A vector specifying how many columns of the table each column of x should span. This is used when using successive calls to latex.table to build tables with complex headers
cwidth	specify the width of each column
cwidth.units	Units of measure for the column width. For example, "in", "cm", or "mm"
cborder	A vector denoting vertical border positions. Borders are placed to the right of the given columns. See "Details".
cborder.thick	For HTML, column border thickness denoted in pixels
cborder.style	A valid HTML descriptor for the color of the column border
rborder	A vector denoting horizontal border positions. Borders are placed at the bottom of the given rows. See "Details".
rbspan	A vector or list giving the start and stop positions of the horizontal borders. Use a list when borders should go from columns 1 - 3 and 5 - 7, but not at column 4.
rborder.thick	For HTML, row border thickness denoted in pixels
rborder.style	A valid HTML descriptor for the color of the row border

rcol	A vector denoting which rows should be colored
usecol	A character vector or string giving the color to be used for the rows in rcol. The color must be a recognized LaTeX color
font	HTML font for the paragraph. Defaults to the HTML option (see setHtmlOptions).
family	HTML font family for the paragraph. Defaults to the HTML option (see setHtmlOptions).
size	Text size of the paragraph. Defaults to the HTML option (see setHtmlOptions). May be an integer or a LaTeX size descriptor. See "Details" for options
justify	Character string giving the alignment for the table on the page. Options are "left", "center", "right".
placement	Controls the placement of the figure. Options are "ht", "t", "b", "p", "H" and can be supplemented with "!". See "Details" for more explanation
open	Logical. Indicates if a new table environment should be opened
close	Logical. Indicates if the current table environment should be closed.
caption	Caption for the table. Currently, captions are placed above tables.
footnote	Additional footnotes to be placed below tables
label	The label to be used by lazy.ref
counter	The name of the counter to be used for this table
counterSet	The value to which counter should be set. In other words, the number of this table
translate	Toggles if inputs in x should be passed through latexTranslate. This should be set to FALSE if writing custom code
cat	Logical. Determines if the output is returned as a character string or returned via the cat function (printed to console). The default value is set by options()$lazyWeave_cat. This argument allows for selective override of the default.

Details

cborder (or column border) will create vertical borders in the table. Borders are placed on the right side of the specified columns. If a border is needed on the far left side of the table, use 0.

rborder (or row border) acts similarly, but for rows. Borders are placed under the specified rows. Use 0 if a line is needed at the top of a table.

Multiple calls to latex.table may be used to make complex tables. For instance, a header may be desired with group names that span over three summary values (See example 2). In these instances, *it is the responsibility of the user to make sure the number of columns in each call is the same as in the other calls.* There is no way in lazyWeave to check the column consistency of tables.

placement options are used as follows:

ht	Place the float here, i.e., approximately at the same point it occurs
t	Position at the top of the page
b	Position at the bottom of the page
p	Put on a special page for floats only
H	Places the float at precisely the location in the LaTeX code. Requires the float package

The "!" may be used after any of these in order to override LaTeX float rules and force your selection. More can be learned by reading about floats in a LaTeX manual.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
#*** Example 1: Simple Table
tab.text <- lazy.table(mtcars[, c(1, 2, 4, 6)], align="right",
                       cborder=c(0, 4), rborder=c(0, nrow(mtcars)))

lazy.write(
  lazy.file.start(),
  tab.text,
  lazy.file.end(),
  OutFile="Example 1.tex")

unlink("Example 1.tex")

#*** Example 2: Complex Table
person <- c("Rachel", "John", "Elizabeth", "George", "Ryan")
veg <- c("", "x", "x", "", "x")
meat <- c("x", "", "", "x", "")
soup <- c("x", "", "x", "x", "")
salad <- c("", "x", "", "", "x")
ice <- c("", "x", "x", "x", "")
cake <- c("x", "", "", "", "x")

dinner <- cbind(person, veg, meat, soup, salad, ice, cake)
colnames(dinner) <- c("Name", "Vegetarian", "Meat",
                      "Soup", "Salad", "Ice Cream", "Cake")

tab1 <- lazy.table(c("", "Entree", "Side", "Dessert"),
                   cspan=c(1, 2, 2, 2),
                   rborder=c(0, 0, 1), rbspan=2:7,
                   caption="Dinner Orders", close=FALSE)
tab2 <- lazy.table(colnames(dinner),
                   align=c("left", rep("center", 6)),
                   cborder=c(3, 5),
                   open=FALSE, close=FALSE)
tab3 <- lazy.table(dinner,
                   align=c("left", rep("center", 6)),
                   cborder=c(1, 3, 5),
                   rborder=c(0, nrow(dinner)), open=FALSE)

lazy.write(
  lazy.file.start(),
  tab1, tab2, tab3,
  lazy.file.end(),
  OutFile="Example 2.tex")

unlink("Example 2.tex")
```

```
## End(Not run)
```

`lazy.text` *Paragraphs in LaTeX*

Description

Write paragraphs in LaTeX code

Usage

```
lazy.text(
  ...,
  title = NULL,
  align = "left",
  italic = FALSE,
  bold = FALSE,
  underline = FALSE,
  sep = "",
  translate = TRUE,
  font,
  family,
  size
)
```

Arguments

`...`	Text and other objects to be included in the paragraph
`title`	A title for the paragraph
`align`	alignment of the paragraph. Options are `"left"`,`"center"`, `"right"`.
`italic`	Logical. Indicates if the text should be italicized
`bold`	Logical. Indicates if the text shoudl be bolded
`underline`	Logical. Indicates if the text should be underlined
`sep`	Character. Denotes the separation string for the items in `...` when they are pasted together
`translate`	Toggles if inputs in `x` should be passed through `latexTranslate`. This should be set to `FALSE` if writing custom code.
`font`	HTML font for the paragraph. Defaults to the HTML option (see `setHtmlOptions`).
`family`	HTML font family for the paragraph. Defaults to the HTML option (see `setHtmlOptions`).
`size`	Text size of the paragraph. Defaults to the HTML option (see `setHtmlOptions`). May be an integer or a LaTeX size descriptor. See "Details" for options

Details

Options for text size are

tiny	smallest
scriptsize	
footnotesize	
small	
normalsize	
large	
Large	
LARGE	
huge	
Huge	BIGGEST

When size is denoted as an integer, as necessary for HTML, it is mapped to a LaTeX size using `map.size`. Likewise, the LaTeX descriptors can be mapped to integers using `map.size`.

Additional formatting may be applied using commands such as `textbf{}` for bold text, `emph{}` for italics, and `ul{}` for underlined text (assuming the `soul` package is available), but doing so is probably easier using `lazy.text.format`.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
lazy.write(
  lazy.file.start(),
  lazy.text("Typically we want our paragraphs to be left
    justified.  This is often what we expect to see when reading."),
  lazy.text("However, we may also have occasions where we would
    like to center our text.  It's one of many ways we can make the
    words stand out on the page", align="center"),
  lazy.text("A more traditional way to make the text stand out might be
    to use bold text or italics, such as these", bold=TRUE, italic=TRUE),
  lazy.file.end(),
  OutFile="Example 1.tex")

unlink("Example 1.tex")

## End(Not run)
```

`lazy.text.format` *Format Text*

Description

Applies italic, bold, or underlining to a piece of text. May be used within `lazy.text` to add emphasis when the entire paragraph does not need to be formatted

Usage

```
lazy.text.format(
  text,
  italic = FALSE,
  bold = FALSE,
  underline = FALSE,
  translate = TRUE
)
```

Arguments

`text`	Text to be formatted
`italic`	Logical. Specifies if text should be italic
`bold`	Logical. Specifies if text should be bold
`underline`	Logical. Specifies if text should be underlined
`translate`	Logical. Specifies if text should be passed through `latexTranslate` before printing

Details

This function differs from `lazy.text` in that `lazy.text` produces a paragraph of formatted text while `lazy.text.format` produces smaller blocks. This allows for smaller bits of text to be formatted for emphasis (see the last example below).

Author(s)

Benjamin Nutter

Examples

```
lazy.text.format("This is the text")
lazy.text.format("This is the text", italic=TRUE)
lazy.text.format("This is the text", bold=TRUE)
lazy.text.format("This is the text", italic=TRUE, bold=TRUE)

lazy.text("The percentage of defective light bulbs in this sample was ",
          lazy.text.format("30%", italic=TRUE),
          ". Clearly, this is unacceptable.")
```

lazy.toc *Table of Contents and Other Lists*

Description

Designates the printing of a table of contents, list of figures, or list of tables. Also provides functionality to manually edit the contents of these lists

Usage

```
lazy.toc(
  type = c("contents", "figures", "tables"),
  add = FALSE,
  desc = "",
  withPage = TRUE,
  sec_unit = c("chapter", "section", "subsection", "subsubsection", "part")
)
```

Arguments

type	Type of list to be printed or edited
add	Determines if the list is printed or if an entry is added to the list
desc	Only used when add=TRUE. Gives the descriptive text of the item being added to the list
withPage	Determines if the page number of the entry is printed in the table of contents. Only used when add=TRUE.
sec_unit	Specifies the format for the new entry. For instance, will the new entry in the table of contents appear as a chapter, section, or subsection. Used only when withPage=TRUE.

Details

The level of detail a table of contents maintains is determined by the counter tocdepth. In most cases, it is set to 3, giving chapter, section, and subsection. To include subsubsections, it would be necessary to include lazy.counter("tocdepth", value=4, fn="set"). Use value=5 to include paragraphs, and so forth.

Value

Returns a string that designating that the table of contents is to be written, or an item to be added to a list. This has no effect for HTML documents

Author(s)

Benjamin Nutter

Examples

```
lazy.toc()
lazy.toc("figures")
lazy.toc("tables", TRUE, "A brief description of the table")
lazy.toc("contents", TRUE, "Subsection 3", sec_unit="subsection")
```

`lazy.verbatim.end` *Verbatim Environments*

Description

Text within a verbatim block appears exactly as type, including whitespace. This is useful when inserting code into a document

Usage

```
lazy.verbatim.end()

lazy.verbatim.start()
```

Details

A verbatim block takes any text entered and typsets it exactly as it was entered. White space is preserved and the font changes. This is typically done to display code, since the whitespace may preserve readability.

For HTML documents, this is done by opening a "<p ...>" tag with font family "monospace" and font "courier". These are applicable until `lazy.verbatim.end` is called and the "</p>" tag is placed, closing the verbatim environment.

It should be noted that HTML code in this forced environment will still not render whitespace as in the LaTeX verbatim environment. This can be enforced by running the text in the environment through a function like `gsub(" ", "  ", [text])` (` ` is the HTML character for a non-breaking space).

Functions

- `lazy.verbatim.end()`:

Author(s)

Benjamin Nutter

`lazy.write` *Output LaTeX Code to a File*

Description

Output text and LaTeX code to a file for building

Usage

```
lazy.write(..., OutFile, append = FALSE, collapse = "\n", footnotes = TRUE)
```

Arguments

`...`	Strings, expressions, and statements to be written to a .tex file
`OutFile`	Filename to which the code should be written
`append`	logical. Indicates if the code should be appended to and existing file
`collapse`	Sets the character string that is placed between elements in `...`
`footnotes`	logical. For HTML and markdown only, when `TRUE`, the footnotes stored in `options("lazy.footnotes")` will be appended to the end of the HTML document

Details

The contents of `...` will be pasted together

Author(s)

Benjamin Nutter

`lazyWeave` *Generate Latex or HTML reports from R*

Description

Arguments passed to functions are embedded in Latex code and can be output to a file. Allows the use of Latex and HTML to write reports, functions for building tables, etc.

Details

Depending on the working directory (for Windows users), the user may encounter the error "texi2dvi.exe: Windows API error 5: Access is denied." when trying to build documents. This happens when the working directory is write protected. It is advisable to change the working directory to something not write protected. This can be done permanently by right clicking on the R shortcut icon, selecting properties, and changing the directory in the "Start in:" box.

`lazyWeave` assumes the availability of the packages `xcolor`, `graphicx`, `colortbl`, `soul`, `lscape`, and `Sweave`. If these packages are not available, the package most likely will not function properly.

It should be noted that `lazyWeave` is a rather inefficient way to go about writing reports with LaTeX or HTML. It's only real advantage is it reduces the amount of knowledge a user needs to have about LaTeX (and it could be debated if that's really an advantage).

Use of `lazyWeave` could also be greatly supplemented by some basic familiarity with LaTeX. For example, knowing the commands for bolding (\textbf{}), italicizing (\emph{}), and underlining text (\ul{}) can go a long way to improving the look of reports. It also would help to know how to subscript and superscript terms. Most introductions to LaTeX will cover these basics.

`lazyWeave` is also only intended to provide the most basic functionality of LaTeX, and I have no plans of extending it much further than what is available already. If what is in the package now is not sufficient enough to satisfy your needs, then I strongly suggest you look into using `Sweave`.

All of the functions can be used for LaTeX and HTML reports, but the functionality and appearance may not be identical between formats.

Author(s)

Maintainer: Benjamin Nutter <benjamin.nutter@gmail.com>

`mantel.test` *Mantel-Haenszel test for two-way tables*

Description

Performs a Mantel-Haenszel test for linear trend in two way tables

Usage

```
mantel.test(
  x,
  byVar,
  row.scores = c("equal", "midrank"),
  col.scores = c("equal", "midrank")
)
```

Arguments

x	Either a vector of data or a two way table. If a vector is given, it should be a factor variable
byVar	If x is a vector, then byVar is a factor variable to be tabulated against x.
row.scores	choice of scores for the row variable. May be "equal", "midrank", or user defined.
col.scores	Choice of scores for the column variable. May be "equal", "midrank", or user defined.

Details

Currently, there is no check to ensure that either variables submitted are factors.

Data should be ordinal. Nominal data may not have any practical meaning in this test. Sometimes, when a nominal variable has only two levels, this test may still be appropriate, i.e. Yes vs. No.

In 2xJ tables, when arbitrary scores (i.e., 0, 1) are applied to the rows and midranks applied to the columns, this test is equivalent to the Wilcoxon, or Mann-Whitney test.

In Ix2 tables, when monotone scores are applied to the rows and abitrary scores applied to the columns, this test is equivalent to the Cochran-Armitage test. See the reference for further details.

Value

Returns an object of type htest.

statistic	M^2, a Chi-square statistic with 1 degree of freedom.
parameter	degrees of freedom for M^2
p.value	p-value for the test
method	Type of test performed.
correlation	correlation coefficient of linear trend

References

Alan Agresti, *An Introduction to Categorical Data Analysis*, 1996, pp. 34 - 39.

Examples

```
mantel.test(mtcars$gear,mtcars$cyl)

mantel.test(table(mtcars$gear,mtcars$cyl))

mantel.test(table(mtcars$gear,mtcars$cyl), row.scores="midrank")

mantel.test(table(mtcars$gear,mtcars$cyl), col.scores="midrank")
```

`map.size` *Convert text size specifications between LaTeX and HTML formats*

Description

The default text size specifications in LaTeX are character strings, but in HTML, the font size is determined by an integer. In order to be able to switch formats in existing code, there needs to be some way to map HTML text size to LaTeX text size, and vice versa. This is managed by `map.size` in the `lazyWeave` functions so the user won't have to worry about it.

Usage

```
map.size(x, reportFormat = getOption("lazyReportFormat"))
```

Arguments

`x`	The text size specification to be mapped to the desired format
`reportFormat`	The format to which `x` should be mapped

Details

Text sizes are mapped according to the following:

LaTeX	HTML
tiny	5
scriptsize	7
footnotesize	8
small	9
normalsize	10
large	12
Large	14
LARGE	18
huge	20
Huge	24

Author(s)

Benjamin Nutter

`pvalString` *Format P-values for Reports*

Description

Convert numeric p-values to character strings according to pre-defined formatting parameters. Additional formats may be added for required or desired reporting standards.

Usage

```
pvalString(p, format = c("default", "exact", "scientific"), digits = 3, ...)
```

Arguments

`p`	a numeric vector of p-values.
`format`	A character string indicating the desired format for the p-values. See Details for full descriptions.
`digits`	For `"exact"` and `"scientific"`; indicates the number of digits to precede scientific notation.
`...`	Additional arguments to be passed to `format`

Details

When `format = "default"`, p-values are formatted:

1. $p > 0.99$: "> 0.99"
2. $0.99 > p > 0.10$: Rounded to two digits
3. $0.10 > p > 0.001$: Rounded to three digits
4. $0.001 > p$: "< 0.001"

When `format = "exact"`, the exact p-value is printed with the number of significant digits equal to `digits`. P-values smaller that 1*(10^-`digits`) are printed in scientific notation.

When `format = "scientific"`, all values are printed in scientific notation with `digits` digits printed before the e.

@author Benjamin Nutter @examples p <- c(1, .999, .905, .505, .205, .125, .09531, .05493, .04532, .011234, .0003431, .000000342) pvalString(p, format="default") pvalString(p, format="exact", digits=3) pvalString(p, format="exact", digits=2) pvalString(p, format="scientific", digits=3) pvalString(p, format="scientific", digits=4)

`setHtmlOptions` *lazyWeave HTML Report Options*

Description

Sets or changes options for report counters, font, font size, and footnotes. The counter options apply to HTML and Markdown documents, and the font options only apply to HTML.

Usage

```
setHtmlOptions(
  table = NULL,
  figure = NULL,
  footnote = NULL,
  chapter = NULL,
  section = NULL,
  subsection = NULL,
  subsubsection = NULL,
  font.family = NULL,
  font = NULL,
  font.size = NULL
)
```

Arguments

`table`	Sets the HTML table counter
`figure`	Sets the HTML figure counter
`footnote`	Sets the HTML footnote counter
`chapter`	Sets the HTML chapter counter
`section`	Sets the HTML section counter
`subsection`	Sets the HTML section counter
`subsubsection`	Sets the HTML subsubsection counter
`font.family`	Sets the HTML font family
`font`	Sets the HTML font
`font.size`	Sets the HTML font size

Details

For all arguments, a value of `NULL` results in no action.

The HTML counters are used to maintain a somewhat consistent appearance between HTML and LaTeX reports. Since HTML doesn't have counters, a series of variables is inserted into the Global Environment. These are hidden from view and are incremented automatically by `lazyWeave` function. Manipulating these variables directly is strongly discouraged. They can all be managed by `lazy.counter`.

setHtmlOptions is a convenience function that can change all of the global variables simultaneously (as opposed to repeated calls to lazy.counter). However, this is the recommended way to change font family, font, and font size.

To change the report format, use the code options(lazyReportFormat = "latex"), options(lazyReportFormat = "html") or, options(lazyReportFormat = "markdown")

Author(s)

Benjamin Nutter

---

univ *Univariable Table*

Description

Similar to the QHS SAS macro, provides a simple summary of numerical variables

Usage

```
univ(
  data,
  vars,
  byVar,
  alpha = 0.05,
  test = c("t.test", "aov", "wilcox.test", "kruskal.test"),
  test.args = NULL
)
```

Arguments

data	A ccf.df or data.frame containing the variables in vars and byVar.
vars	Character vector naming the variables in data to be summarized.
byVar	A categorical variables giving groups by which statistics should be computed.
alpha	significance level for determining the confidence limits.
test	The test to use for comparing between groups. Currently limited to t-test and ANOVA (parametric tests).
test.args	a list of additional arguments to pass to the function in test.

Details

Statistics available in univ, and the calls to get them are:

n number of non-missing values.

missing number of missing values

mean arithmetic mean

`median` median value

`sd` standard deviation

`lcl` lower confidence limit

`ucl` upper confidence limit

`min` minimum value

`max` maximum value

`p25` 25th percentile

`p75` 75th percentile

`cv` coefficient of variation

`univ` does not perform any kind of inference on the varaibles supplied in the argument. It returns only summary statistics. If comparisons are desired, try using `conttable`. Confidence limits are determined using the t-distribution.

Author(s)

Benjamin Nutter

Examples

```
data(Delivery)
#Read in the delivery dataset from the CCFmisc library
#use labelVector package to label variables in univariate tables
Delivery$maternal.age <-
  labelVector::set_label(Delivery$maternal.age, "Maternal Age")
Delivery$ga.weeks <-
  labelVector::set_label(Delivery$ga.weeks, "Gestation weeks")
Delivery$wt.gram <-
  labelVector::set_label(Delivery$wt.gram, "Weight (g)")


#a univariate table of the variables maternal age,
#ga.weeks and wt.grams.  The object resulting
#from univ() can be used in other functions to create html or
#LaTeX tables.

uni <- univ(Delivery,
            vars=c("maternal.age", "ga.weeks", "wt.gram"))

#a univariate table of the variables maternal age,
#ga.weeks and wt.grams by delivery.type.  The object resulting
#from univ() can be used in other functions to create html or
#LaTeX tables.

deliv.uni <- univ(Delivery,
                  vars=c("maternal.age", "ga.weeks", "wt.gram"),
                  byVar="delivery.type")

#if you want to take advantage of the confidence interval
```

```
#output from univ() different alpha levels can be set
#by the alpha= argument.

deliv_99.uni <- univ(Delivery,
                     vars=c("maternal.age", "ga.weeks", "wt.gram"),
                     byVar="delivery.type",
                     alpha=0.01)
```

`write.univ` *Write Univariate Table to a File*

Description

Write the output of `univ` to a file

Usage

```
write.univ(
  x,
  round = 1,
  Factor = TRUE,
  Group = FALSE,
  N = TRUE,
  Missing = FALSE,
  Mean = TRUE,
  SD = TRUE,
  LCL = FALSE,
  UCL = FALSE,
  Min = TRUE,
  P25 = TRUE,
  Median = TRUE,
  P75 = TRUE,
  Max = TRUE,
  CV = FALSE,
  Pval = FALSE,
  pvalFormat = "default",
  pvalArgs = list(),
  cat = getOption("lazyWeave_cat"),
  ...
)
```

Arguments

`x`	An object of type `univ`.
`round`	Number of significant digits to be printed.
`Factor`	Determines if the Factor (variable) description is printed.

Group	Determines if the Group is printed
N	Determines if the number of non missing values is printed
Missing	Determines if the number of missing values is printed
Mean	Determines if the mean is printed
SD	Determines if the standard deviation is printed
LCL	Determines if the lower confidence limit is printed
UCL	Determines if the upper confidence limit is printed
Min	Determines if the minimum value is printed
P25	Determines if the 25th percentile is printed
Median	Determines if the median value is printed
P75	Determines if the 75th percentile is printed
Max	Determines if the maximum value is printed
CV	Determines if the coefficient of variation is printed
Pval	Determines if the p-value is printed
pvalFormat	Character string passed to pvalString and determines the pvalue style to be printed.
pvalArgs	A list of additional arguments to be passed to pvalString
cat	Logical. Determines if the output is returned as a character string or returned via the cat function (printed to console). The default value is set by options()$lazyWeave_cat. This argument allows for selective override of the default.
...	additional arguments to be passed to lazy.matrix

Examples

```
#output will be written to the working directory
getwd()

#write.univ function must be written to either a LaTeX
#or HTML file.  HTML format is through the lazyHTML package.
options(lazyReportFormat="html")

#Delivery dataset from CCFmisc library
data(Delivery)

#label the variables that will be used
Delivery$maternal.age <-
  labelVector::set_label(Delivery$maternal.age, "Maternal Age")
Delivery$ga.weeks <-
  labelVector::set_label(Delivery$ga.weeks, "Gestation weeks")
Delivery$wt.gram <-
  labelVector::set_label(Delivery$wt.gram, "Weight (g)")

#summaries of the continuous variables
#maternal.age, ga.weeks and wt.gram in the
#Delivery dataset.
```

```
deliv.uni <- univ(Delivery,
                  vars=c("maternal.age", "ga.weeks", "wt.gram")
)

#summaries of continuous variables
#by level of delivery.type
delivBy.uni <- univ(Delivery,
                    vars=c("maternal.age", "ga.weeks", "wt.gram"),
                    byVar="delivery.type"
)

#to write univ based table to an HTML file enclose the
#write.univ() in the html_write function as below.
#see documentation for other options.

#To print byVariable group names in the table,
#set the Group=T flag in the write.univ() function.

## Not run:
lazy.write(
    lazy.file.start(),
    write.univ(deliv.uni),
    write.univ(delivBy.uni, Group=TRUE),
    lazy.file.end(),
    OutFile="ExampleFile.html"
  )

  unlink("ExampleFile.html")

## End(Not run)
```

`WritePrintCtable` *Write and Print Comparison Tables*

Description

Print Comparisons to the console or write the table to a file.

Usage

```
## S3 method for class 'ctable'
print(x, ...)

split_ctable(x, max.rows = 35, keepVarTogether = TRUE, ...)

write.ctable(
  x,
  round = 2,
```

```
  percent = TRUE,
  quartile = TRUE,
  cwidth = NULL,
  caption = NULL,
  footnote = NULL,
  byVarN = FALSE,
  size = "\\normalsize",
  descripCombine = TRUE,
  oddsCombine = TRUE,
  markSignificant = FALSE,
  statHeader = "Statistics",
  name = FALSE,
  var.label = TRUE,
  level = TRUE,
  total = TRUE,
  descriptive = TRUE,
  missing = FALSE,
  missing.perc = FALSE,
  testStat = TRUE,
  odds = FALSE,
  pval = TRUE,
  oneLine = FALSE,
  pvalFormat = "default",
  pvalArgs = list(),
  cat = getOption("lazyWeave_cat"),
  ...
)
```

Arguments

x	A ctable object to be printed or written
...	Other arguments to be passed to `print` (for `print.ctable` or `lazy.table` (for `write.ctable`). Currently none are implemented for `print`. In `split_ctable`, any options in `write.ctable` may also be included.
max.rows	The maximum number of rows to be displayed on a page. Variable groups are forced to be kept together.
keepVarTogether	Determines if variables are kept together when splitting a table across mulitple pages. When a single variable has more levels than can remain on one printed page, this must be set to FALSE.
round	The number of decimal places to be displayed for numeric values.
percent	Toggles if percentages or proportions are printed for categorical values
quartile	Toggles if quartiles or min and max are printed for numeric values associated with the `median` argument in `conttable`
cwidth	A vector giving the width of each column in the body of the table. The number of columns is not always obvious, and this vector is not recycled. If the length is

	inappropriate, a warning message will be printed indicating the correct number of columns.
caption	The name of the table.
footnote	A footnote for the table.
byVarN	Toggles if the N per group in byVar are printed in the column headings.
size	A character string denoting the size of the text for the table. This must be latex code, for example "\normalsize" or "\small." Remember to use to backslashes!
descripCombine	Toggles if descriptive statistics are combined. It is strongly recommended that this be left TRUE as the appearance is much better.
oddsCombine	Toggles if odds ratios are combined with the lower up upper confidence limits.
markSignificant	Toggles if significant results are printed in bold text.
statHeader	Character string giving the column heading for statistical summaries.
name	Toggles if the variable name is printed in the table.
var.label	Toggles if the variable label is printed in the table.
level	Toggles if the variable levels are printed in the table. This column is usually needed for categorical variables, and never needed for numeric variables.
total	Toggles if the totals column is printed in the table
descriptive	Toggles if descriptive statistics are printed.
missing	Toggles if the number of missing values are printed in the table.
missing.perc	Toggles if the percentage of missing values is printed in the table.
testStat	Toggles if the test statistics are printed.
odds	Toggles if odds ratios are printed. Only relevant to numeric variables.
pval	Toggles if the pvalue column is printed.
oneLine	When true, binary variables are printed with only one line per variable. This does not affect the printing of numeric variable or variables with more than two levels.
pvalFormat	Character string passed to pvalString and determines the pvalue style to be printed.
pvalArgs	A list of additional arguments to be passed to pvalString
cat	Logical. Determines if the output is returned as a character string or returned via the cat function (printed to console). The default value is set by options()$lazyWeave_cat. This argument allows for selective override of the default.

Author(s)

Benjamin Nutter

Index