Package ‘ztable’

February 19, 2015

**Title**  Zebra-Striped Tables in LaTeX and HTML Formats

**Version**  0.1.5

**Description**  Makes zebra-striped tables (tables with alternating row colors) in LaTeX and HTML formats easily from a data.frame, matrix, lm, aov, anova, glm, coxph, nls, fitdistr, mytable and cbind.mytable objects.

**Depends**  R (>= 3.1.2)

**License**  GPL-2

**LazyData**  true

**Suggests**  MASS, survival, testthat, knitr, moonBook

**VignetteBuilder**  knitr

**Author**  Keon-Woong Moon [aut, cre]

**Maintainer**  Keon-Woong Moon <cardiomoon@gmail.com>

**NeedsCompilation**  no

**Repository**  CRAN

**Date/Publication**  2015-02-15 07:37:56

### R topics documented:

- `addCellColor` .......................... 3
- `addcgroup` ............................ 3
- `addColColor` ........................... 4
- `addrgroup` ............................ 4
- `addRowColor` ........................... 5
- `addSubColNames` ......................... 5
- `align2html` ............................ 6
- `align2lines` ............................ 6
- `align2nd` .............................. 7
- `alignCheck` ............................ 7
- `alignCount` ............................ 7
- `caption2minipage` ....................... 8
- `cGroupSpan` ............................ 8
- `colGroupCount` ........................ 9
R topics documented:

- data2table ......................................................... 9
- define_colors .................................................. 9
- extractAlign ....................................................... 10
- getNewAlign ...................................................... 10
- getNewSpanCol ................................................... 10
- getNewSpanRow ................................................... 11
- getspanRowData .................................................. 11
- getspanRowLength ............................................... 12
- hlines ............................................................. 12
- isGroupCol ......................................................... 13
- isspanCol .......................................................... 13
- isspanRow .......................................................... 14
- make.cell.color ................................................... 14
- myhtmlStyle ......................................................... 15
- name2rgb ........................................................... 15
- parallelTables ..................................................... 16
- parallelTablesHTML ............................................ 17
- parallelTablesLatex ............................................. 17
- print.ztable ....................................................... 18
- printHTMLHead ................................................... 18
- printLatexHead ................................................... 18
- printRowGroup .................................................... 19
- print_ztable ....................................................... 19
- repColor ............................................................ 19
- spanCol .............................................................. 20
- spanColWidth ....................................................... 20
- spanRow .............................................................. 21
- tableLength ......................................................... 21
- totalCol ............................................................. 22
- tr ................................................................. 22
- tr2 ................................................................. 22
- update_ztable ..................................................... 23
- validColor ........................................................ 26
- validColor2 ....................................................... 26
- vline2align ........................................................ 27
- vlines ............................................................... 27
- zcolors .............................................................. 28
- ztable ............................................................... 28
- ztable.cbind.mytable ........................................... 30
- ztable.mytable .................................................... 30
- ztable2html ........................................................ 31
- ztable2latex ....................................................... 31
- ztable2viewer ...................................................... 32
- ztable_sub ........................................................ 32

Index 37
addCellColor

Add column colors of an object of ztable

Description
Add column colors of an object of ztable

Usage
addCellColor(z, rows, cols, color)

Arguments
- z: An object of ztable
- rows: An integer vector indicating specific rows
- cols: An integer vector indicating specific columns
- color: A character vector indicating color

Examples
z = ztable(head(iris))
z = addRowColor(z, c(1, 3), color = "platinum")
z = addColColor(z, 2, color = "syan")
z = addCellColor(z, cols = c(5, 4), rows = 5, color = "red")
z

addcgroup

Add column groups of an object of ztable

Description
Add column groups of an object of ztable

Usage
addcgroup(z, cgroup, n.cgroup, cgroupcolor = NULL)

Arguments
- z: An object of ztable
- cgroup: A character vector or matrix indicating names of column group. Default value is NULL
- n.cgroup: A integer vector or matrix indicating the numbers of columns included in each cgroup. Default value is NULL
- cgroupcolor: A character vector or matrix indicating the background colors of each cells.
addColColor  
Add column colors of an object of ztable

Description
Add column colors of an object of ztable

Usage
addColColor(z, cols, color)

Arguments

z  An object of ztable
cols  An integer vector indicating specific columns
color  A character vector indicating color

Examples
z = ztable(head(iris))
z = addColColor(z, c(1,3), color = "platinum")
z

addrgroup  
Add row groups of an object of ztable

Description
Add row groups of an object of ztable

Usage
addrgroup(z, rgroup, n.rgroup, cspan.rgroup = NULL)

Arguments

z  An object of ztable
rgroup  A character vector indicating names of row group. Default value is NULL
n.rgroup  A integer vector indicating the numbers of rows included in each rgroup Dafault value is NULL
cspan.rgroup  An integer indicating the column span of rgroup
**addRowColor**

*Add row colors of an object of ztable*

---

**Description**

Add row colors of an object of ztable

**Usage**

```r
addRowColor(z, rows, color)
```

**Arguments**

- `z` An object of ztable
- `rows` An integer vector indicating specific rows
- `color` A character vector indicating color

**Examples**

```r
z = ztable(head(iris))
z = addRowColor(z, c(1, 3), color = "platinum")
z
```

---

**addSubColNames**

*Add a adjunctive name below columnn name in a ztable*

---

**Description**

Add a adjunctive name below columnn name in a ztable

**Usage**

```r
addSubColNames(z, subcolnames)
```

**Arguments**

- `z` An object of ztable
- `subcolnames` A character vector
align2html

*Convert the align in Latex format to html format*

**Description**

Convert the align in Latex format to html format

**Usage**

`align2html(align)`

**Arguments**

- `align`: A character of align in Latex format

align2lines

*count the vertical column lines from align of Latex format*

**Description**

count the vertical column lines from align of Latex format

**Usage**

`align2lines(align)`

**Arguments**

- `align`: A string of align Latex format

**Value**

a numeric vector consists of vertical lines of each column
align2nd  
*Delete first components of align*

**Description**
Delete first components of align

**Usage**
align2nd(align)

**Arguments**
- **align**  
  A character for define the align of column in Latex format

---

alignCheck  
*Check the validity of align*

**Description**
Check the validity of align

**Usage**
alignCheck(align, ncount, addrow)

**Arguments**
- **align**  
  A character for define the align of column in Latex format
- **ncount**  
  An integer equals of ncol function
- **addrow**  
  An integer

---

alignCount  
*Count the number of align*

**Description**
Count the number of align

**Usage**
alignCount(align)

**Arguments**
- **align**  
  A character for define the align of column in Latex format
caption2minipage  Convert long caption to minipage

Description

Convert long caption to minipage

Usage

caption2minipage(z, caption)

Arguments

z  An object of ztable
caption  A character vector to convert


cGroupSpan  Count the colspan of each colgroup

Description

Count the colspan of each colgroup

Usage

cGroupSpan(z)

Arguments

z  An object of ztable

Value

A matrix indicating the column span occupied by each colgroup
colGroupCount

Count the colgroup of an object of ztable

Description
Count the colgroup of an object of ztable

Usage
colGroupCount(z)

Arguments
z An object of class ztable

Value
A vector indicating the position of colgroup

data2table
Convert data to formatted data for table

Description
Convert data to formatted data for table

Usage
data2table(z)

Arguments
z An object of class "ztable"

define_colors
Define colors

Description
Define colors of mycolors

Usage
define_colors(mycolors)

Arguments
mycolors chracters vectors of color names
**extractAlign**  
*Extract column position information only (without vertical line specifier)*

**Description**  
Extract column position information only (without vertical line specifier).

**Usage**

```r
extractAlign(align)
```

**Arguments**

- `align`  
  A character string indicating align for latex table

---

**getNewAlign**  
*Make a character string indicating the alignment of components of table.*

**Description**  
Make a character string indicating the alignment of components of table.

**Usage**

```r
getNewAlign(z)
```

**Arguments**

- `z`  
  An object of ztable

---

**getNewSpanCol**  
*Calculating new spanCol with spanCol plus space made by column group*

**Description**  
Calculating new spanCol with spanCol plus space made by column group

**Usage**

```r
getNewSpanCol(z)
```

**Arguments**

- `z`  
  An object of ztable
getNewSpanRow

Calculating new spanRow with spanRow plus space made by row group

Description
Calculating new spanRow with spanRow plus space made by row group

Usage
getNewSpanRow(z)

Arguments
z An object of ztable

getspanRowData
Gets the spanRaw start column

Description
Gets the spanRaw start column

Usage
getspanRowData(z, i, j)

Arguments
z An object of ztable
i An integer indicating the row of specific cell
j An integer indicating the column of specific cell

Value
An integer indicating column where spanRaw start. This function is for latex multirow
getspanRowLength  

Description

Gets spanRow length

Usage

getspanRowLength(z, i, j)

Arguments

z  
An object of ztable

i  
An integer indicating the row of specific cell

j  
An integer indicating the column of specific cell

Value

row count when spanRow starts, 0 when column spans.

hlines  

Add or delete horizontal lines in a ztable

Description

Add or delete horizontal lines in a ztable

Usage

hlines(z, type = NULL, add = NULL, del = NULL)

Arguments

z  
An object of ztable

type  
An integer or one of c("none", "all")

add  
An integer vector indicating rows where the horizontal lines added

del  
An integer vector indicating rows where the horizontal lines deleted
isGroupCol

Returns whether or not column with position start plus length is group column

Description

Returns whether or not column with position start plus length is group column

Usage

isGroupCol(start, length, colCount)

Arguments

start An integer indicating start column position
length An integer indicating spanCol length
colCount An integer vector calculating from colGroupCount()

isspanCol

Idetify the spanCol status of a cell

Description

Idetify the spanCol status of a cell

Usage

isspanCol(z, i, j)

Arguments

z An object of ztable
i An integer indicating the row of specific cell
j An integer indicating the column of specific cell

Value

column plus space count when spanCol starts, 0 when column spans, minus value when spanCol ends, NULL when no span.
### isspanRow

**Identify the spanRow status of a cell**

**Description**

Identify the spanRow status of a cell

**Usage**

```r
isspanRow(z, i, j)
```

**Arguments**

- `z`: An object of ztable
- `i`: An integer indicating the row of specific cell
- `j`: An integer indicating the column of specific cell

**Value**

columns count plus spaces by rgroup when spanRow starts, 0 when row spans, minus value when spanRow ends, NULL when no span.

### make.cell.color

**Make a data.frame named "cellcolor" from ztable call**

**Description**

Make a data.frame named "cellcolor" from ztable call

**Usage**

```r
make.cell.color(x, zebra, zebra.color, zebra.type, zebra.list, zebra.colnames, zebra.rownames)
```

**Arguments**

- `x`: a data.frame
- `zebra`: Null or an integer of 0 or 1 or 2. The arguments zebra and zebra.color are used to make a Zebra striping table(table with alternating background colors) easily. A value of 1 sets background color of all odd rows/columns with specified with zebra.color. A value of 2 sets all even rows/columns. A value of 0 sets background colors of all rows/columns with colors specified with zebra.color. When zebra is 1 or 2, the parameters of prefix.rows and commands ignored. Default is NULL.
zebra.color A color name or a numeric value indicating pre-defined color. When parameter zebra is 0 or 1 or 2 and zebra.color is NULL, then zebra.color is set to "platinum". Numeric values between 1 to 13 is converted to predefined color names. The predefined color names are c("peach", "peach-orange", "peachpuff", "peach-yellow", "pear", "pearl", "peridot", "periwinkle", "pastelred", "pastelgray"). Default is NULL.

zebra.type An integer of 0 or 1 or 2 or 3. A value of 1 sets background colors by row. A value of 2 sets background colors by column. A value of 0 sets background colors of all cells. A value of 3 sets background colors of cells specified with zebra.list. Default value is 1.

zebra.list A list consists of y,x,color. zebra.list is used only when zebra.type=3. zebra.list sets the cells specified with rows of vector "y" and columns of vector "x" with "color". The y and x are integer vector indicating rows and columns. NA value of y or x indicating all columns or rows. The color is an character vector consists of names of color.

zebra.colnames whether or not use background colors in column names row, Default value is FALSE

zebra.rownames whether or not use background colors in row names column, Default value is TRUE

---

myhtmlStyle print html style

**Description**

print html style

**Usage**

myhtmlStyle()

---

name2rgb Find rgb value from color name

**Description**

Find rgb value from color name

**Usage**

name2rgb(name)

**Arguments**

name a valid color name
parallelTables

Place two or more ztables or figures side by side in Latex or HTML format

Description

Place two or more ztables or figures side by side in Latex or HTML format. Requires Latex "boxed-minipage" package in preamble. The ztable for this purpose should be made by function ztable with tabular="TRUE".

Usage

parallelTables(width, listTables, type = "latex")

Arguments

width: a numeric vector specifies the width to which the tables or figures should be scaled
listTables: a list consists of object of "ztable" or valid figure name
type: Type of table to produce. Possible values for type are "latex" or "html". Default value is "latex".

Examples

require(ztable)
z1=ztable(head(mtcars[1:3]),tabular=TRUE)
parallelTables(c(0.5,0.3),list(z1,z2))
parallelTables(c(0.5,0.5),list(z1,z2, type="html"))
z1=ztable(head(iris[1:3]),turn=TRUE,angle=10,zebra=1)
z2=ztable(head(iris[1:3]),turn=TRUE,angle=-10,zebra=2)
parallelTables(c(0.5,0.5),list(z1,z2))
parallelTablesHTML

**Description**

Place two or more ztables or figures side by side in HTML format. The ztable for this purpose should be made by function ztable with tabular="TRUE".

**Usage**

```r
parallelTablesHTML(width, listTables)
```

**Arguments**

- `width`: a numeric vector specifies the width to which the tables or figures should be scaled
- `listTables`: a list consists of object of "ztable" or valid figure name

---

parallelTablesLatex

**Description**

Place two or more ztables or figures side by side in Latex format. The ztable for this purpose should be made by function ztable with tabular="TRUE".

**Usage**

```r
parallelTablesLatex(width, listTables)
```

**Arguments**

- `width`: a numeric vector specifies the width to which the tables or figures should be scaled
- `listTables`: a list consists of object of "ztable" or valid figure name
print.ztable  
*Print an object of class "ztable"*

**Description**

Print an object of class "ztable"

**Usage**

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

**Arguments**

- `x`  
  An object of class "ztable"
- `...`  
  further argument passed to other function

printHTMLHead  
*Print HTML head if ztable object a has a colgroup*

**Description**

Print HTML head if ztable object a has a colgroup

**Usage**

```r
printHTMLHead(z)
```

**Arguments**

- `z`  
  An object of ztable

printLatexHead  
*Print the head of latex table if the object of ztable has a colgroup*

**Description**

Print the head of latex table if the object of ztable has a colgroup

**Usage**

```r
printLatexHead(z)
```

**Arguments**

- `z`  
  An object of ztable
printRowGroup  

Print Row Groups in a latex table

Description

Print Row Groups in a latex table

Usage

printRowGroup(z, i)

Arguments

z An object of class ztable
i An integer indicating row

print_ztable  

Print an object of class "ztable"

Description

Print an object of class "ztable"

Usage

print_ztable(z)

Arguments

z An object of class "ztable"

repColor  

Make vector x from vector color

Description

Internal function of make.cell.color

Usage

repColor(x, color)

Arguments

x A destination vector
color A character vector consists of color names
**spanCol**

*Merging data cells of ztable object in columns*

**Description**

Merging data cells of ztable object in columns

**Usage**

`spanCol(z, row, from, to, color = NULL)`

**Arguments**

- `z` An object of ztable
- `row` An integer indicating the row of merging data cell
- `from` An integer indicating start column of merging data cell
- `to` An integer indicating end column of merging data cell
- `color` An optional character indicating the background color of merging cell

---

**spanColWidth**

*Calculate the spanColWidth when spanCol start*

**Description**

Calculate the spanColWidth when spanCol start

**Usage**

`spanColWidth(z, i, j)`

**Arguments**

- `z` An object of ztable
- `i` An integer indicating the row of specific cell
- `j` An integer indicating the column of specific cell

**Value**

column count when spanCol start
**spanRow**

*Merging data cells of ztable object in rows*

**Description**

Merging data cells of ztable object in rows

**Usage**

```r
spanRow(z, col, from, to, color = NULL)
```

**Arguments**

- `z` : An object of ztable
- `col` : An integer indicating the column of merging data cell
- `from` : An integer indicating start row of merging data cell
- `to` : An integer indicating end row of merging data cell
- `color` : An optional character indicating the background color of merging cell

---

**tableLength**

*Convert data to formatted data for table*

**Description**

Convert data to formatted data for table

**Usage**

```r
tableLength(z)
```

**Arguments**

- `z` : An object of class "ztable"
### totalCol

**Calculating total columns of ztable**

**Description**
Calculating total columns of ztable

**Usage**
```
totalCol(z)
```

**Arguments**
- `z`: An object of ztable

### tr

**Subfunction used in ztable2latex**

**Description**
Subfunction used in ztable2latex

**Usage**
```
tr(string)
```

**Arguments**
- `string`: a character vector

### tr2

**Subfunction used in ztable2html**

**Description**
Subfunction used in ztable2html

**Usage**
```
tr2(string)
```

**Arguments**
- `string`: a character vector
update_ztable

Description

Update options of ztable before print

Usage

update_ztable(z, size = NULL, color = NULL, tablewidth = NULL,
type = NULL, include.rownames = NULL, placement = NULL,
position = NULL, show.heading = NULL, show.footer = NULL,
caption = NULL, caption.placement = NULL, caption.position = NULL,
caption.bold = NULL, align = NULL, digits = NULL, display = NULL,
sidewaystable = NULL, longtable = NULL, rotate = NULL, turn = NULL,
angle = NULL, wrappable = NULL, wrappetablewidth = NULL, tabular = NULL,
label = NULL, hline.after = NULL, booktabs = NULL, prefix.rows = NULL,
commands = NULL, top.command = NULL, zebra = NULL, zebra.color = NULL,
zebra.type = NULL, zebra.list = NULL, zebra.colnames = NULL,
zebra.rownames = NULL, colnames.bold = NULL, include.colnames = NULL,
cgroup = NULL, n.cgroup = NULL, rgroup = NULL, n.rgroup = NULL,
cspan.rgroup = NULL)

Arguments

z An object of class "ztable"
size An integer from 1 to 10 indicating font size = c("tiny","scriptsize","footnotesize","small","normalsize","large","Large","LARGE","huge","Huge") respectively.
color A character indicating color of ztable
tablewidth A numeric indicating desired table width as a ratio to linewidth. Default value is 0.3.
type character indicating formats of ztable, either "html" or "latex".
include.rownames A logical value whether or not include rownames in the table
placement The table will have placement given by placement where placement must be NULL or contain only elements of "h","t","b","p","!","H".
position The table will be have placed at the center of the paper if position is "center" or "c", and at the left side of the paper if it equals "left" or "l", and at the right side of the paper if it equals "right" or "r". The position is translated to specified latex environments such as "flushright" or "flushleft" or "center" (provided as a character vector) will enclose the tabular environment.
show.heading A logical value whether or not include headings in the table.
show.footer A logical value whether or not include headings in the table.
caption A character
caption.placement
The caption will be have placed at the top of the table if caption.placement is "top" and at the bottom of the table if it equals "bottom".

caption.position
The caption will be have placed at the center of the table if caption.position is "center" or "c", and at the left side of the table if it equals "left" or "l", and at the right side of the table if it equals "right" or "r".

caption.bold
whether or not use bold font for caption

align
Character vector : nchar equal to the number of columns of the resulting table indicating the alignment of the corresponding columns.

digits
Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table

display
Character vector of length equal to the number of columns of the resulting table indicating the format for the corresponding columns. Since the row names are printed in the first column, the length of display is one greater than ncol(x) if x is a data.frame. These values are passed to the formatC function. Use "d" (for integers), "f", "e", "E", "g", "G", "fg" (for reals), or "s" (for strings). "f" gives numbers in the usual xxx.xxx format; "e" and "E" give n.ddde+nn or n.dddE+nn (scientific format); "g" and "G" put x[i] into scientific format only if it saves space to do so. "fg" uses fixed format as "f", but digits as number of significant digits. Note that this can lead to quite long result strings.

sidewaystable
Logical value whether or not set the tabular environment= "sidewaystable". Requires Latex "rotating" package in preamble.

longtable
Logical value whether or not set the tabular environment= "longtable". Requires Latex "longtable" package in preamble.

rotate
Logical value whether or not set the tabular environment= "rotate". No special arrangement is made to find space for the result. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle(counterclockwise).

turn
Logical value whether or not set the tabular environment= "turn". In this environment, Latex leaves space for the rotated table. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle.

angle
An integer indicate the angle to rotate(degree); range -180 to 180.

wraptable
Logical value whether or not set the tabular environment= "wraptable". Requires Latex "wrapfig" package in preamble.

wraptablewidth
A integer indicate wraptable width in centimeter.
	
	
tabular
Logical value whether or not set the tabular environment. If TRUE, no tabular environment is set.

label
Character vector of length 1 containing the LaTeX label or HTML anchor. Set to NULL to suppress the label.

hline.after
A vector of numbers between -1 and "nrow(x)", inclusive, indicating the rows after which a horizontal line should appear. If NULL is used no lines are produced. Default value is c(-1,0,nrow(x)) which means draw a line before and after the columns names and at the end of the table. Repeated values are allowed.
booktabs Logical value. If TRUE, the toprule, midrule and bottomrule tags from the LaTeX "booktabs" package are used rather than hline for the horizontal line tags. Requires LaTeX 'booktabs' package in preamble.

prefix.rows A numeric vector contains the position of rows on which extra Latex commands should be added as a prefix.

commands A character vector of the length 1 or same length of the nrow of data.frame which contains the command that should be added as a prefix at the specified rows.

top.command A character vector of the length 1 which contains the command that should be added as a prefix at the colnames row.

zebra Null or a integer of 1 or 2. The arguments zebra and zebra.color are used to make a Zebra striping table (table with alternating background colors) easily. A value of 1 sets background color of all odd rows with specified with zebra.color. A value of 2 sets all even rows. when zebra is 1 or 2, the parameters of prefix.rows and commands ignored.

zebra.color A color name or a numeric value indicating pre-defined color. When parameter zebra is 0 or 1 or 2 and zebra.color is NULL, then zebra.color is set to "platinum". Numeric values between 1 to 13 is converted to predefined color names. The predefined color names are c("peach", "peach-orange", "peachpuff", "peach-yellow", "pear", "pearl", "peridot", "periwinkle", "pastelred", "pastelgray").

zebra.type An integer of 0 or 1 or 2 or 3. A value of 1 sets background colors by row. A value of 2 sets background colors by column. A value of 0 sets background colors of all cells. A value of 3 sets background colors of cells specified with zebra.list. Default value is 1.

zebra.list A list consists of y,x,color. zebra.list is used only when zebra.type=3. zebra.list sets the cells specified with cells[y,x] with "color". The y and x are integer indicating rows and columns. NA value of y or x indicating all columns or rows.

zebra.colnames whether or not use background colors in column names row, Default value is FALSE

zebra.rownames whether or not use background colors in row names column, Default value is TRUE

colnames.bold whether or not use bold font for column names.

include.colnames Logical. If TRUE the column names is printed.

cgroup A character vector or matrix indicating names of column group. Default value is NULL

n.cgroup A integer vector or matrix indicating the numbers of columns included in each cgroup Daault value is NULL

rgroup A character vector indicating names of row group. Default value is NULL

n.rgroup A integer vector indicating the numbers of rows included in each rgroup Daault value is NULL

cspan.rgroup The number of columns that an rgroup should span. It spans by default all columns but you may want to limit this if you have column colors that you want to retain.
validColor  
\textit{Find valid color name}

\begin{description}
\item[Description] 
Find valid color name
\item[Usage] 
validColor(a, mycolor)
\item[Arguments] 
\begin{itemize}
\item a \hspace{1cm} An integer or a character
\item mycolor \hspace{1cm} predefined color names
\end{itemize}
\item[Value] 
a valid Latex color name
\end{description}

validColor2  
\textit{Find valid color name}

\begin{description}
\item[Description] 
Find valid color name
\item[Usage] 
validColor2(a)
\item[Arguments] 
\begin{itemize}
\item a \hspace{1cm} An integer or a character
\end{itemize}
\item[Value] 
a valid Latex color name
\end{description}
vline2align** Make a latex "align" from a string and vertical line specifier**

**Description**

Make a latex "align" from a string and vertical line specifier

**Usage**

vline2align(align, vlines)

**Arguments**

- **align**: A character string indicating align of latex table
- **vlines**: An integer vector indicating vertical line position

vlines** Add or delete vertical lines in a ztable**

**Description**

Add or delete vertical lines in a ztable

**Usage**

vlines(z, type = NULL, add = NULL, del = NULL)

**Arguments**

- **z**: An object of ztable
- **type**: An integer or one of c("none","all")
- **add**: An integer vector indicating columns where the width of vertical lines added
- **del**: An integer vector indicating columns where the width of vertical lines subtracted
zcolors

**Definition of Latex Color**

**Description**
A dataset containing the name of color and Hex-triplets and latex definition

**Usage**
zcolors

**Format**
A data frame with 749 rows and 3 variables:

- **name** Color name
- **rgb** Hex triplet of color
- **definition** Latex command of color definition

**Details**
To use this color definition, a latex package "color" should be included in your preamble.

---

ztable

**Exporting a R object to an object of class "ztable"**

**Description**
Exporting a R object to an object of class "ztable"

**Usage**
ztable(x, digits = NULL, ...)

## Default S3 method:
ztable(x, digits = NULL, ...)

## S3 method for class 'data.frame'
ztable(x, digits = NULL, ...)

## S3 method for class 'matrix'
ztable(x, digits = NULL, ...)

## S3 method for class 'lm'
ztable(x, digits = NULL, ...)
## Arguments

- `x`  
  An R object, mainly data.frame

- `digits`  
  Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table

- `...`  
  Additional arguments to be passed to `ztable_sub`

## Methods (by class)

- `default`
- `data.frame`
- `matrix`
- `lm`
- `fitdistr`
- `nls`
- `aov`
- `anova`
- `glm`
- `coxph`
- `prcomp`
- `summary.prcomp`
ztable.cbind.mytable  Make ztable from object cbind.mytable

Description
Make ztable from object cbind.mytable

Usage
## S3 method for class 'cbind.mytable'
ztable(x, digits = NULL, ...)

Arguments
x An object of cbind.mytable
digits Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table
...
arguments to be passed to ztable_sub

Examples
require(moonBook)
res=mytable(sex+DM~., data=acs)
z=ztable(res)
z

ztable.mytable  Make ztable from object mytable

Description
Make ztable from object mytable

Usage
## S3 method for class 'mytable'
ztable(x, digits = NULL, ...)

Arguments
x An object of mytable
digits Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table
...
arguments to be passed to ztable_sub
Examples

```r
require(moonBook)
res=mytable(sex~.,data=acs)
z=ztable(res)
z
```

---

**ztable2html**

*Print an object of class "ztable" to html table*

---

**Description**

Print an object of class "ztable" to html table

**Usage**

`ztable2html(z, xdata)`

**Arguments**

- `z` An object of class "ztable"
- `xdata` A formatted data.frame

---

**ztable2latex**

*Print an object of class "ztable" to Latex table*

---

**Description**

Print an object of class "ztable" to Latex table

**Usage**

`ztable2latex(z, xdata)`

**Arguments**

- `z` An object of class "ztable"
- `xdata` A formatted data.frame
ztable2viewer

Print an object of ztable via rstudio::viewer

Description

Print an object of ztable via rstudio::viewer

Usage

ztable2viewer(z)

Arguments

z  An object of ztable

ztable_sub

Exporting "data.frame" to an object of class "ztable"

Description

Exporting "data.frame" to an object of class "ztable"

Usage

ztable_sub(x, size = 5, color = getOption("ztable.color", "black"),
  tablewidth = 0.3, type = getOption("ztable.type", "latex"),
  include.rownames = getOption("ztable.include.rownames", TRUE),
  placement = "!htb", position = "c",
  show.header = getOption("ztable.show.header", TRUE),
  show.footer = getOption("ztable.show.footer", TRUE), caption = NULL,
  caption.placement = getOption("ztable.caption.placement", "top"),
  caption.position = getOption("ztable.caption.position", "c"),
  caption.bold = getOption("ztable.caption.bold", FALSE), align = NULL,
  digits = NULL, display = NULL, sidewaystable = FALSE,
  longtable = FALSE, rotate = FALSE, turn = FALSE, angle = 0,
  wraptable = FALSE, wraptablewidth = 12, tabular = FALSE, label = NULL,
  hline.after = NULL, booktabs = getOption("ztable.booktabs", TRUE),
  prefix.rows = NULL, commands = NULL, top.command = NULL,
  zebra = getOption("ztable.zebra", NULL),
  zebra.color = getOption("ztable.zebra.color", NULL),
  zebra.type = getOption("ztable.zebra.type", 1),
  zebra.colnames = getOption("ztable.zebra.colnames", FALSE),
  zebra.rownames = getOption("ztable.zebra.rownames", TRUE),
  zebra.list = NULL, colnames.bold = getOption("ztable.colnames.bold", FALSE),
  include.colnames = getOption("ztable.include.colnames", TRUE),
  cgroup = NULL, n.cgroup = NULL, rgroup = NULL, n.rgroup = NULL,
  cspan.rgroup = NULL)
### Arguments

- **x**
  - A data.frame

- **size**
  - An integer from 1 to 10 indicating font size: c("tiny","scriptsize","footnotesize","small","normalsize","large","Large","LARGE","huge","Huge") respectively. Defaulting is 5 (= "normalsize").

- **color**
  - A character indicating color of `ztable`

- **tablewidth**
  - A numeric value indicating desired table width as a ratio to linewidth. This value is only useful when caption is longer than table length. Default value is 0.3.

- **type**
  - Character indicating formats of `ztable`, either "html" or "latex". Default value is "latex"

- **include.rownames**
  - A logical value whether or not include rownames in the table. Default value is TRUE.

- **placement**
  - The table will have placement given by placement where placement must be NULL or contain only elements of "h","t","b","p","!","H". Default value is "!hbt".

- **position**
  - The table will be have placed at the center of the paper if position is "center" or "c", and at the left side of the paper if it equals "left" or "l", and at the right side of the paper if it equals "right" or "r". The position is translated to specified latex environments such as "flushright" or "flushleft" or "center" (provided as a character vector) will enclose the tabular environment. Default value is "center".

- **show.heading**
  - A logical value whether or not include headings in the table. Default value is TRUE.

- **show.footer**
  - A logical value whether or not include headings in the table. Default value is TRUE.

- **caption**
  - A character

- **caption.placement**
  - The caption will be have placed at the top of the table if caption.placement is "top" and at the bottom of the table if it equals "bottom". Default value is "top".

- **caption.position**
  - The caption will be have placed at the center of the table if caption.position is "center" or "c", and at the left side of the table if it equals "left" or "l", and at the right side of the table if it equals "right" or "r". Default value is "center".

- **caption.bold**
  - Whether or not use bold font for caption

- **align**
  - Character vector: nchar equal to the number of columns of the resulting table indicating the alignment of the corresponding columns.

- **digits**
  - Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table

- **display**
  - Character vector of length equal to the number of columns of the resulting table indicating the format for the corresponding columns. Since the row names are printed in the first column, the length of display is one greater than ncol(x) if x is a data.frame. These values are passed to the `formatC` function. Use "d" (for integers), "f", "e", "E", "g", "G", "fg" (for reals), or "s" (for strings). "f" gives
numbers in the usual xxx.xxx format; "e" and "E" give n.ddde+nn or n.dddE+nn (scientific format); "g" and "G" put x[i] into scientific format only if it saves space to do so. "fg" uses fixed format as "f", but digits as number of significant digits. Note that this can lead to quite long result strings. Default value is NULL, the class of x.

**sidewaystable** Logical value whether or not set the tabular environment= "sidewaystable". Requires Latex "rotating" package in preamble. Default value is FALSE.

**longtable** Logical value whether or not set the tabular environment= "longtable". Requires Latex "longtable" package in preamble. Default value is FALSE.

**rotate** Logical value whether or not set the tabular environment= "rotate". No special arrangement is made to find space for the result. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle (counterclockwise). Default value is FALSE.

**turn** Logical value whether or not set the tabular environment= "turn". In this environment, Latex leaves space for the rotated table. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle. Default value is FALSE.

**angle** An integer indicate the angle to rotate (degree); range -180 to 180. Default value is 0.

**wraptable** Logical value whether or not set the tabular environment= "wraptable". Requires Latex "wrapfig" package in preamble. Default value is FALSE.

**wraptablewidth** A integer indicate wraptable width in centimeter. Default=12.

**tabular** Logical value whether or not set the tabular environment. If TRUE, no tabular environment is set. Default value is FALSE.

**label** Character vector of length 1 containing the LaTeX label or HTML anchor. Set to NULL to suppress the label. Default value is NULL.

**hline.after** A vector of numbers between -1 and "nrow(x)", inclusive, indicating the rows after which a horizontal line should appear. If NULL is used no lines are produced. Default value is c(-1,0,nrow(x)) which means draw a line before and after the columns names and at the end of the table. Repeated values are allowed.

**booktabs** Logical value. If TRUE, the \texttt{toprule}, \texttt{midrule} and \texttt{bottomrule} tags from the LaTeX "booktabs" package are used rather than \texttt{hline} for the horizontal line tags. Requires Latex "booktabs" package in preamble. Default value is TRUE.

**prefix.rows** A numeric vector contains the position of rows on which extra Latex commands should be added as a prefix.

**commands** A character vector of the length 1 or same length of the \texttt{nrow} of data.frame which contains the command that should be added as a prefix at the specified rows. Default value is NULL, i.e. do not add commands.

**top.command** A character vector of the length 1 which contains the command that should be added as a prefix at the colnames row.

**zebra** Null or an integer of 0 or 1 or 2 or 3. The arguments zebra and zebra.color are used to make a Zebra striping table (table with alternating background colors) easily. A value of 1 sets background color of all odd rows/columns with specified with zebra.color. A value of 2 sets all even rows/columns. A value
of 0 sets background colors of all rows/columns with colors specified with zebra.color. When zebra is 1 or 2, the parameters of prefix.rows and commands ignored. When zebra=3, the background colors can be defined by addRowColor, addColColor and addCellColor functions. Default is NULL.

zebra.color A color name or a numeric value indicating pre-defined color. When parameter zebra is 0 or 1 or 2 and zebra.color is NULL, then zebra.color is set to "platinum". Numeric values between 1 to 13 is converted to predefined color names. The predefined color names are c("peach","peach-orange", "peachpuff","peach-yellow","pear","pearl","peridot","periwinkle","pastelred", "pastelgray"). Default is NULL.

zebra.type An integer of 0 or 1 or 2 or 3. A value of 1 sets background colors by row. A value of 2 sets background colors by column. A value of 0 sets background colors of all cells. A value of 3 sets background colors of cells specified with zebra.list. Default value is 1.

zebra.colnames whether or not use background colors in column names row, Default value is FALSE

zebra.rownames whether or not use background colors in row names column, Default value is TRUE

zebra.list A list consists of y,x,color. zebra.list is used only when zebra.type=3. zebra.list sets the cells specified with rows of vector "y" and columns of vector "x" with "color". The y and x are integer vector indicating rows and columns. NA value of y or x indicating all columns or rows. The color is an character vector consists of names of color.

colnames.bold whether or not use bold font for column names, Default value is FALSE

include.colnames Logical. If TRUE the column names is printed. Default value is TRUE.

cgroup A character vector or matrix indicating names of column group. Default value is NULL

n.cgroup A integer vector or matrix indicating the numbers of columns included in each cgroup Dafault value is NULL

rgroup A character vector indicating names of row group. Default value is NULL

n.rgroup A integer vector indicating the numbers of rows included in each rgroup Dafault value is NULL

cspan.rgroup The number of columns that an rgroup should span. It spans by default all columns but you may want to limit this if you have column colors that you want to retain.

Examples

```
require(ztable)
x=head(iris)
ztable(x)
ztable(x,size=3,caption="Table 1. ztable Test")
ztable(x,size=7,caption="Table 1. ztable Test",caption.position="1")
ztable(x,size=7,caption="Table 1. ztable Test",caption.placement="bottom", caption.position="1")
```
```r
fit = lm(mpg ~ ., data = mtcars)
ztable(fit)
data(USArrests)
pr1 <- prcomp(USArrests)
summary(pr1)
ztable(pr1)
require(survival)
data(colon)
attach(colon)
out <- glm(status ~ rx + obstruct + adhere + nodes + extent, data = colon, family = binomial)
ztable(out)
colon$TS = Surv(time, status == 1)
out1 = coxph(TS ~ rx + obstruct + adhere + differ + extent + surg + node4, data = colon)
ztable(out1)
ztable(head(mtcars), zebra = 1)
ztable(head(mtcars), zebra = 1, zebra.type = 2)
```
Index

*Topic datasets
  zcolors, 28

addCellColor, 3
ddggroup, 3
addColColor, 4
addrgroup, 4
addRowColor, 5
addSubColNames, 5
align2html, 6
align2lines, 6
align2nd, 7
alignCheck, 7
alignCount, 7
caption2minipage, 8
cGroupSpan, 8
colGroupCount, 9
data2table, 9
define_colors, 9
extractAlign, 10
getNewAlign, 10
getNewSpanCol, 10
getNewSpanRow, 11
getspanRowData, 11
getspanRowLength, 12
hlines, 12
isGroupCol, 13
isspanCol, 13
isspanRow, 14
make.cell.color, 14
myhtmlStyle, 15
name2rgb, 15
parallelTables, 16
paralelTablesHTML, 17
parallelTablesLatex, 17
print.ztable, 18
print_ztable, 19
printHTMLHead, 18
printLatexHead, 18
printRowGroup, 19
repColor, 19
spanCol, 20
spanColWidth, 20
spanRow, 21
tableLength, 21
totalCol, 22
tr, 22
tr2, 22
update_ztable, 23
validColor, 26
validColor2, 26
vline2align, 27
vlines, 27
zcolors, 28
ztable, 28
ztable.cbind.mtyable, 30
ztable.mtyable, 30
ztable2html, 31
ztable2latex, 31
ztable2viewer, 32
ztable_sub, 29, 30, 32