

# Package ‘helpeRs’

December 24, 2025

**Title** Helper Functions for High-Quality Presentation of Quantitative Research

**Version** 0.0

**Description** Helper functions for high-quality presentation of quantitative research. Provides tools for generating publication-ready regression tables in LaTeX format, with support for clustered standard errors, bootstrap inference, and instrumental variables. Also includes visualization utilities for creating heat maps and other graphics.

**Depends** R (>= 3.3.3)

**License** GPL-3

**Encoding** UTF-8

**Imports** fields,  
ggExtra,  
ggplot2,  
interp,  
lmtest,  
missForest,  
plyr,  
sandwich,  
stargazer,  
stringr,  
viridis

**RoxygenNote** 7.3.3

**Suggests** MASS,  
testthat (>= 3.0.0)

**Config/testthat/edition** 3

## Contents

cols2numeric . . . . .	2
colSummmmary . . . . .	3
f2n . . . . .	4
fixZeroEndings . . . . .	4
FullTransformer . . . . .	5
GetTableEntry . . . . .	6
heatMap . . . . .	8
heatmap2 . . . . .	10
image2 . . . . .	12

MakeHeatMap . . . . .	13
Stargazer2FullTable . . . . .	15
Tables2Tex . . . . .	16
vcovCluster . . . . .	18
WidenMargins . . . . .	19
<b>Index</b>	<b>21</b>

---

cols2numeric	<i>Convert data frame columns to numeric when possible</i>
--------------	--

---

## Description

Attempts to coerce each column of a data frame to numeric using [f2n](#). Columns that cannot be coerced (i.e., result in all NA values) are left unchanged. This is useful for cleaning data where numeric values may be stored as factors or character strings.

## Usage

```
cols2numeric(x)
```

## Arguments

x                      A data frame to process.

## Value

A data frame with the same dimensions as x. Columns that could be converted to numeric are returned as numeric; others retain their original type.

## See Also

[f2n](#) for converting individual vectors, [colSummary](#) for summarizing columns

## Examples

```
# Data frame with mixed types
df <- data.frame(
  a = factor(c("1.5", "2.5", "3.5")),
  b = c("x", "y", "z"),
  c = c(1, 2, 3),
  stringsAsFactors = FALSE
)

# Convert what can be converted
df_numeric <- cols2numeric(df)
sapply(df_numeric, class)
# a becomes numeric, b stays character, c stays numeric
```

---

colSummary*Summarize each column of a data frame*

---

## Description

Computes a single summary value for each column of a data frame. Numeric columns are summarized by their mean, while non-numeric columns are summarized by their mode (most frequent value). This is primarily used when constructing prediction grids for visualization, where you need representative values for variables not being varied.

## Usage

```
colSummary(x)
```

## Arguments

**x**                      A data frame to summarize.

## Details

The function uses [f2n](#) internally to attempt numeric conversion, so factor columns with numeric-looking levels will be treated as numeric.

## Value

A named vector with one element per column of **x**. Names correspond to column names. Numeric summaries are means; non-numeric summaries are the most frequent value.

## See Also

[cols2numeric](#) for converting columns to numeric, [MakeHeatMap](#) which uses this function internally

## Examples

```
# Summarize mtcars
colSummary(mtcars[, 1:4])

# Mixed numeric and character columns
df <- data.frame(
  x = c(1, 2, 3, 4),
  y = c("a", "b", "a", "a")
)
colSummary(df) # Returns mean of x and mode of y
```

---

f2n	<i>Convert factors to numeric values</i>
-----	--

---

### Description

Converts a vector to numeric by first coercing to character. This is the correct way to extract numeric values from factors, as direct coercion with `as.numeric()` returns factor level indices rather than the underlying values.

### Usage

```
f2n(x)
```

### Arguments

x	A vector to convert, typically a factor whose levels are numeric strings (e.g., <code>factor(c("1.5", "2.3", "3.1"))</code> ).
---	--

### Value

A numeric vector. Values that cannot be coerced to numeric become NA with a warning.

### See Also

[cols2numeric](#) for converting multiple columns of a data frame

### Examples

```
# Factor with numeric levels
x <- factor(c("1.5", "2.3", "3.1"))

# Wrong way - returns level indices (1, 2, 3)
as.numeric(x)

# Correct way - returns actual values
f2n(x) # Returns c(1.5, 2.3, 3.1)
```

---

fixZeroEndings	<i>Ensure numbers have a fixed number of decimal places</i>
----------------	---

---

### Description

Pads numeric strings with trailing zeros so that all values have exactly `roundAt` digits after the decimal point. This ensures consistent formatting in regression tables where column alignment matters.

### Usage

```
fixZeroEndings(zr, roundAt = 2)
```

**Arguments**

zr	A character or numeric vector to process. Numeric values are first converted to character.
roundAt	Integer specifying the desired number of decimal places. Default is 2.

**Value**

A character vector with values padded to have exactly roundAt decimal places. Values without a decimal point receive one followed by the appropriate number of zeros.

**See Also**

[GetTableEntry](#) which uses this function to format coefficient estimates

**Examples**

```
# Pad to 2 decimal places (default)
fixZeroEndings(c("1.5", "2", "3.14"))
# Returns: c("1.50", "2.00", "3.14")

# Pad to 3 decimal places
fixZeroEndings(c(1.5, 2.33, 3), roundAt = 3)
# Returns: c("1.500", "2.330", "3.000")
```

---

FullTransformer

*Clean and reorder regression tables for publication*


---

**Description**

Takes a matrix of regression output and performs string replacements, row name cleaning, and reordering so that the resulting table is suitable for inclusion in LaTeX documents. This function is typically called internally by [Tables2Tex](#) but can be used directly for custom table processing.

**Usage**

```
FullTransformer(t_FULL, COLNAMES_VEC, name_conversion_matrix = NULL)
```

**Arguments**

t_FULL	Matrix or data frame containing the raw regression table output, typically from <a href="#">GetTableEntry</a> .
COLNAMES_VEC	Character vector of column names to apply to the final table. Length must match the number of columns in t_FULL.
name_conversion_matrix	Optional two-column matrix for renaming row labels. Column 1 contains regex patterns to match against row names, column 2 contains the replacement names. If NULL (default), no renaming is performed.

## Details

The function performs several transformations:

- Renames rows using the optional `name_conversion_matrix`
- Moves summary statistics (R-squared, observations, etc.) to the bottom
- Expands camelCase variable names into readable format
- Handles `as.factor()` variable names by extracting factor levels
- Preserves common acronyms (GDP, AIC, MMD, SMD)

## Value

A transformed data frame with cleaned row names, reordered rows (summary statistics at bottom), and the specified column names.

## See Also

[Tables2Tex](#) which calls this function to process full tables, [GetTableEntry](#) for extracting table entries from models

## Examples

```
## Not run:
# Create a simple table
t_raw <- data.frame(Model1 = c("0.5 (2.1)*", "0.95", "100"))
row.names(t_raw) <- c("gdpPerCapita", "Adjusted R-squared", "Observations")

# Clean and rename
name_mat <- matrix(c("gdpPerCapita", "GDP per Capita"), ncol = 2)
t_clean <- FullTransformer(t_raw, COLNAMES_VEC = "Model 1",
                           name_conversion_matrix = name_mat)

## End(Not run)
```

---

GetTableEntry

*Extract regression results as a formatted table row*

---

## Description

Builds a one-row data frame containing coefficient estimates, standard errors or t-statistics, significance stars, and summary statistics from a fitted model. Supports both analytical standard errors (including clustered and heteroskedasticity-robust) and bootstrap-based inference.

## Usage

```
GetTableEntry(
  my_lm,
  clust_id,
  iv_round = 2,
  NAME = "",
  iv = F,
```

```

    inParens = "tstat",
    seType = "analytical",
    bootDataLocation = "./",
    bootDataNameTag = "Data",
    bootFactorVars = NULL,
    bootExcludeCovars = NULL,
    keepCoef1 = FALSE,
    superunit_covariateName = "country",
    superunit_label = "Countries"
)

```

### Arguments

<code>my_lm</code>	A fitted model object (typically from <code>lm()</code> , <code>glm()</code> , or <code>ivreg()</code> ). The data used to fit the model must be a data.frame.
<code>clust_id</code>	Character string giving the name of the clustering variable for clustered standard errors. If NULL, heteroskedasticity-robust standard errors (HC1) are used instead.
<code>iv_round</code>	Integer specifying the number of decimal places for rounding estimates and statistics. Default is 2.
<code>NAME</code>	Character string used as the column name for the resulting row. Default is an empty string.
<code>iv</code>	Logical; if TRUE, includes instrumental variable diagnostic statistics (Weak instruments and Wu-Hausman tests) in the output. Only applicable for models fitted with <code>ivreg()</code> . Default is FALSE.
<code>inParens</code>	Character string specifying what to display in parentheses: "tstat" (default) for t-statistics or "se" for standard errors.
<code>seType</code>	Character string specifying the type of standard errors: "analytical" (default) uses sandwich estimators, "boot" uses bootstrap standard errors from pre-computed replications on disk.
<code>bootDataLocation</code>	Character string giving the folder path containing bootstrap replication datasets. Only used when <code>seType = "boot"</code> .
<code>bootDataNameTag</code>	Character string giving the file name prefix for bootstrap data files. Files should be named {bootDataNameTag}_0.csv for the original data and {bootDataNameTag}_1.csv, etc. for replications.
<code>bootFactorVars</code>	Character vector of variable names to treat as factors during bootstrap estimation.
<code>bootExcludeCovars</code>	Character vector of covariate names to exclude from the imputation step during bootstrap processing.
<code>keepCoef1</code>	Logical; if TRUE, includes the first coefficient (typically the intercept) in the output. Default is FALSE.
<code>superunit_covariateName</code>	Character string giving the name of the variable used to count higher-level units (e.g., countries in panel data). Default is "country".
<code>superunit_label</code>	Character string used as the row label for the higher-level unit count in the output table. Default is "Countries".

## Details

The output format is designed for downstream processing by [Tables2Tex](#), with coefficients formatted as "estimate (stat)\*" where stat is either the t-statistic or standard error (controlled by `inParens`), and \* indicates  $p < 0.05$ .

## Value

A one-row data frame where each column corresponds to a coefficient or summary statistic. Columns include:

- Formatted coefficient estimates with significance stars
- Fit statistic (Adjusted R-squared for linear models, AIC for GLMs)
- Number of observations
- Count of higher-level units (e.g., countries)
- IV diagnostics (if `iv = TRUE`)

## See Also

[Tables2Tex](#) for generating complete LaTeX tables from multiple models, [vcovCluster](#) for the clustered standard error implementation

## Examples

```
## Not run:
# Fit a simple linear model
data(mtcars)
fit <- lm(mpg ~ wt + hp, data = mtcars)

# Extract with robust standard errors
entry <- GetTableEntry(fit, clust_id = NULL, NAME = "Model 1")

# Extract with clustered standard errors
entry_clust <- GetTableEntry(fit, clust_id = "cyl", NAME = "Model 2")

## End(Not run)
```

---

heatMap

---

*Create an interpolated heat map from scattered data*


---

## Description

Interpolates irregularly-spaced (x, y, z) observations onto a regular grid using the **interp** package and visualizes the result using [image.plot](#) from the **fields** package. This is useful for visualizing relationships in data where observations are not on a regular grid.



**Usage**

```

heatMap(
  x,
  y,
  z,
  main = "",
  N,
  yaxt = NULL,
  xlab = "",
  ylab = "",
  horizontal = F,
  useLog = "",
  legend.width = 1,
  ylim = NULL,
  xlim = NULL,
  zlim = NULL,
  add.legend = T,
  legend.only = F,
  vline = NULL,
  col_vline = "black",
  hline = NULL,
  col_hline = "black",
  cex.lab = 2,
  cex.main = 2,
  myCol = NULL,
  includeMarginals = F,
  marginalJitterSD_x = 0.01,
  marginalJitterSD_y = 0.01,
  openBrowser = F
)

```

**Arguments**

<code>x, y, z</code>	Numeric vectors of equal length defining the coordinates and values to interpolate. <code>x</code> and <code>y</code> are the spatial coordinates, <code>z</code> is the value at each location.
<code>main</code>	Character string for the plot title. Default is empty.
<code>N</code>	Integer specifying the number of grid cells in each direction for interpolation. Higher values produce smoother plots but take longer.
<code>yaxt</code>	Optional character vector of labels for the y-axis. If <code>NULL</code> (default), numeric axis labels are used.
<code>xlab, ylab</code>	Character strings for axis labels. Default is empty.
<code>horizontal</code>	Logical; if <code>TRUE</code> , draw the color legend horizontally below the plot. Default is <code>FALSE</code> (vertical legend on right).
<code>useLog</code>	Character string specifying which axes to log-transform. Can include " <code>x</code> ", " <code>y</code> ", and/or " <code>z</code> " (e.g., " <code>xyz</code> " for all axes). Default is empty (no log transformation).
<code>legend.width</code>	Numeric value controlling the width of the color legend. Default is 1.
<code>xlim, ylim, zlim</code>	Numeric vectors of length 2 giving the plot limits for each axis. If <code>NULL</code> (default), limits are computed from the data.
<code>add.legend</code>	Logical; if <code>FALSE</code> , the color legend is suppressed. Default is <code>TRUE</code> .

<code>legend.only</code>	Logical; if TRUE, draw only the legend without the main plot. Useful for creating custom layouts. Default is FALSE.
<code>vline, hline</code>	Numeric values specifying positions of vertical or horizontal reference lines. Default is NULL (no lines).
<code>col_vline, col_hline</code>	Colors for the reference lines. Default is "black".
<code>cex.lab, cex.main</code>	Numeric values for character expansion of axis labels and title. Default is 2 for both.
<code>myCol</code>	Optional color palette vector. If NULL, uses the default <code>image.plot</code> palette.
<code>includeMarginals</code>	Logical; if TRUE, adds 1D marginal rug plots showing the distribution of x and y values. Default is FALSE.
<code>marginalJitterSD_x, marginalJitterSD_y</code>	Numeric values controlling the amount of jitter applied to marginal points, as a fraction of the standard deviation. Default is 0.01.
<code>openBrowser</code>	Logical; if TRUE, enters debug mode via <code>browser()</code> for interactive inspection. Default is FALSE.

### Value

Invisibly returns NULL. Called for its side effect of producing a plot.

### See Also

[heatmap2](#) for plotting matrices directly, [MakeHeatMap](#) for model-based prediction heat maps, [image.plot](#) for the underlying plotting function

### Examples

```
## Not run:
# Create some scattered data
set.seed(42)
x <- runif(100, 0, 10)
y <- runif(100, 0, 10)
z <- sin(x) * cos(y) + rnorm(100, sd = 0.1)

# Plot as interpolated heat map
heatMap(x, y, z, N = 50, main = "Example Heat Map",
        xlab = "X Variable", ylab = "Y Variable")

## End(Not run)
```

---

heatmap2

---

*Quick heat map visualization of a matrix*


---

### Description

A convenience wrapper around [heatMap](#) for fast visualization of matrix-like objects. Can use either base R graphics or **ggplot2** for a more polished appearance.

**Usage**

```
heatmap2(
  mat,
  row_labels = NULL,
  col_labels = NULL,
  use_gg = FALSE,
  scale = FALSE,
  log = FALSE,
  includeMarginals = FALSE,
  ...
)
```

**Arguments**

<code>mat</code>	Numeric matrix or data frame to visualize. Will be coerced to a matrix if necessary.
<code>row_labels</code>	Optional character vector of labels for rows. If NULL (default), uses <code>rownames(mat)</code> .
<code>col_labels</code>	Optional character vector of labels for columns. If NULL (default), uses <code>colnames(mat)</code> .
<code>use_gg</code>	Logical; if TRUE, creates a <b>ggplot2</b> plot instead of using base graphics. Requires the <b>ggplot2</b> package. Default is FALSE.
<code>scale</code>	Logical; if TRUE, standardizes values (subtracts mean, divides by SD) before plotting. Default is FALSE.
<code>log</code>	Logical; if TRUE, applies natural log transformation to values before plotting. Default is FALSE.
<code>includeMarginals</code>	Logical; if TRUE and <code>use_gg = TRUE</code> , adds marginal histograms using <b>ggExtra</b> . For base graphics, adds marginal rug plots. Default is FALSE.
<code>...</code>	Additional arguments passed to <a href="#">heatMap</a> when <code>use_gg = FALSE</code> .

**Details**

When `use_gg = TRUE`, the function creates a tile plot using [geom\\_tile](#) with a white-to-steelblue gradient. Marginal histograms can be added using **ggExtra** if available.

**Value**

When `use_gg = TRUE`, invisibly returns the ggplot object. Otherwise, invisibly returns NULL. Called for its side effect of producing a plot.

**See Also**

[heatMap](#) for the underlying scatter-to-grid interpolation, [image2](#) for simple matrix plotting

**Examples**

```
# Create a correlation matrix
cor_mat <- cor(mtcars[, 1:5])

# Quick base R plot
heatmap2(cor_mat, row_labels = colnames(cor_mat),
         col_labels = colnames(cor_mat))
```

```
## Not run:
# ggplot2 version
heatmap2(cor_mat, use_gg = TRUE)

## End(Not run)
```

---

image2

---

*Plot a matrix as an image with intuitive orientation*


---

## Description

A wrapper around [image](#) that displays matrices with the conventional orientation (first row at top, first column at left). The base `image()` function displays matrices transposed and flipped, which can be confusing. This function corrects that behavior.

## Usage

```
image2(
  mat,
  xaxt = NULL,
  yaxt = NULL,
  col = NULL,
  main = NULL,
  scale_vec = c(1, 1.04),
  cex.axis = 1
)
```

## Arguments

<code>mat</code>	Numeric matrix to display as a heat map image.
<code>xaxt</code>	Optional character vector of labels for the x-axis (columns). If <code>NULL</code> (default), no x-axis labels are drawn.
<code>yaxt</code>	Optional character vector of labels for the y-axis (rows). If <code>NULL</code> (default), no y-axis labels are drawn.
<code>col</code>	Optional vector of colors for the heat map. If <code>NULL</code> , uses the default <code>image()</code> color palette.
<code>main</code>	Optional character string for the plot title.
<code>scale_vec</code>	Numeric vector of length 2 controlling the placement scaling of tick labels on the x and y axes. Default is <code>c(1, 1.04)</code> .
<code>cex.axis</code>	Numeric value for character expansion of axis tick labels. Default is 1.

## Value

Invisibly returns `NULL`. Called for its side effect of producing a plot.

## See Also

[heatmap2](#) for a higher-level heat map function, [heatMap](#) for interpolated heat maps from scattered data

**Examples**

```
# Create a simple matrix
mat <- matrix(1:12, nrow = 3, ncol = 4)
rownames(mat) <- c("A", "B", "C")
colnames(mat) <- c("W", "X", "Y", "Z")

# Plot with row/column labels
image2(mat, xaxt = colnames(mat), yaxt = rownames(mat), main = "Example")
```

MakeHeatMap

*Visualize two-way predictor effects as a heat map***Description**

Creates a prediction surface showing how the expected outcome varies across a grid of two predictor variables, holding all other predictors at their mean (numeric) or mode (categorical) values. The result is saved as a PDF file.

**Usage**

```
MakeHeatMap(
  factor1,
  factor2,
  outcome,
  dat,
  lm_obj,
  pdf_path,
  extrap_factor1 = 1,
  extrap_factor2 = 1,
  useLog = "",
  OUTCOME_SCALER = 1,
  OutcomeTransformFxn = function(x) {
    x
  },
  openBrowser = F
)
```

**Arguments**

factor1, factor2	Character strings giving the names of the two predictor variables to vary. These must be columns in dat.
outcome	Character string giving the name of the outcome variable. Used only for excluding from the summary computation.
dat	Data frame containing the data used to fit the model. Used to determine variable ranges and compute summary values.
lm_obj	Fitted linear model object (from <code>lm()</code> or similar) used for generating predictions.
pdf_path	Character string giving the file path where the PDF plot will be saved.

extrap_factor1, extrap_factor2	Numeric multipliers controlling how far beyond the observed data range the prediction grid extends. Values > 1 extrapolate beyond the data; values < 1 restrict to a subset of the range. Default is 1 (exact data range).
useLog	Character string specifying which axes to log-transform. Can include "x", "y", and/or "z". Default is empty (no transformation).
OUTCOME_SCALER	Numeric value by which predictions are multiplied before plotting. Useful for unit conversions. Default is 1.
OutcomeTransformFxn	Function applied to predictions before scaling. Default is the identity function <code>function(x) x</code> .
openBrowser	Logical; if TRUE, pauses execution via <code>browser()</code> for interactive debugging. Default is FALSE.

## Details

The function uses [colSummary](#) to compute representative values for predictors not being varied, then generates predictions across a 50x50 grid spanning the ranges of the two focal predictors. The predictions are visualized using [heatMap](#) with the **viridis** color palette.

## Value

Invisibly returns NULL. The function is called for its side effect of writing a PDF file to `pdf_path`.

## See Also

[heatMap](#) for the underlying plotting function, [colSummary](#) for computing representative predictor values

## Examples

```
## Not run:
# Fit a model
fit <- lm(mpg ~ wt + hp + cyl, data = mtcars)

# Create heat map of mpg as function of wt and hp
MakeHeatMap(
  factor1 = "wt",
  factor2 = "hp",
  outcome = "mpg",
  dat = mtcars,
  lm_obj = fit,
  pdf_path = "mpg_heatmap.pdf"
)

## End(Not run)
```

---

Stargazer2FullTable	<i>Convert a stargazer table to a self-contained longtable</i>
---------------------	--

---

### Description

Transforms the LaTeX output from [stargazer](#) into a longtable environment suitable for multi-page regression tables. This function handles the conversion automatically, including header repetition on continued pages.

### Usage

```
Stargazer2FullTable(stargazer_text, fontsize = "footnotesize")
```

### Arguments

stargazer_text	Character vector containing the raw lines produced by <code>stargazer()</code> (typically captured with <code>capture.output()</code> ).
fontsize	Character string specifying the LaTeX font size environment to wrap the table in. Common values include "footnotesize", "scriptsize", or "tiny". Default is "footnotesize".

### Details

The function performs several transformations:

- Replaces `table` with `longtable` environment
- Wraps content in the specified font size environment
- Adds `\endfirsthead` and `\endhead` markers for header repetition
- Includes "(Continued from previous page)" note on subsequent pages
- Removes incompatible stargazer formatting directives

### Value

A character vector of LaTeX code ready to be written to a .tex file using `write()` or `cat()`.

### LaTeX Requirements

The output requires the `longtable` package in your LaTeX preamble:

```
\usepackage{longtable}
```

### See Also

[Tables2Tex](#) which calls this function internally, [FullTransformer](#) for table cleaning before conversion

## Examples

```
## Not run:
# Capture stargazer output and convert to longtable
library(stargazer)
fit <- lm(mpg ~ wt + hp, data = mtcars)
sg_output <- capture.output(stargazer(fit))

# Convert to longtable format
lt_output <- Stargazer2FullTable(sg_output, fontsize = "scriptsize")

# Write to file
write(lt_output, file = "full_table.tex")

## End(Not run)
```

---

Tables2Tex

---

*Generate publication-ready regression tables in LaTeX*


---

## Description

Collates a list of regression model objects, extracts their coefficient estimates and statistics via [GetTableEntry](#), and writes publication-ready LaTeX table files using [stargazer](#).

## Usage

```
Tables2Tex(
  reg_list,
  clust_id,
  seType = "analytical",
  checkmark_list = NULL,
  addrow_list = NULL,
  saveFolder = "./",
  nameTag = "Table",
  saveFull = T,
  tabCaption = "",
  model.names = NULL,
  NameConversionMat = NULL,
  DoFullTableKey = T,
  superunit_covariateName = "country",
  superunit_label = "Countries",
  font.size = "footnotesize",
  inParens = "tstat",
  keepCofef1 = FALSE,
  font.size.full = "footnotesize"
)
```

## Arguments

<code>reg_list</code>	A list of fitted model objects, or a character vector of object names to evaluate. Models should be fitted using <code>lm()</code> , <code>glm()</code> , or <code>ivreg()</code> . The data argument to these models must be a <code>data.frame</code> (not a matrix or tibble).
-----------------------	---



<code>clust_id</code>	Character string giving the name of the clustering variable for clustered standard errors, or NULL for heteroskedasticity-robust standard errors.
<code>seType</code>	Character string specifying the type of standard errors: "analytical" (default) or "boot" for bootstrap.
<code>checkmark_list</code>	Optional named list of binary vectors indicating which models include certain features. Names become row labels, values of 1 produce checkmarks in the table.
<code>addrow_list</code>	Optional named list of vectors to append as additional rows to the table. Each list element becomes a row with the element name as the row label.
<code>saveFolder</code>	Character string giving the folder path where LaTeX files will be written. Default is the current directory.
<code>nameTag</code>	Character string used as the base name for output files. Files are named <code>tab{nameTag}_SE{seType}.tex</code> and <code>FULL_tab{nameTag}_SE{seType}.tex</code> .
<code>saveFull</code>	Logical; if TRUE (default), also produces a full table containing all coefficients using the longtable environment.
<code>tabCaption</code>	Character string for the table caption.
<code>model.names</code>	Optional character vector of column headings for each model. If NULL, defaults to "Model 1", "Model 2", etc.
<code>NameConversionMat</code>	Optional two-column matrix for filtering and renaming row labels. Column 1 contains regex patterns to match, column 2 contains replacement names. Rows not matching any pattern are dropped from the condensed table (but retained in the full table).
<code>DoFullTableKey</code>	Logical; if TRUE (default), adds a reference to the full table in the condensed table's caption.
<code>superunit_covariateName</code>	Character string giving the variable name used to count higher-level units (e.g., "country" for panel data).
<code>superunit_label</code>	Character string for the row label showing the count of higher-level units. Default is "Countries".
<code>font.size</code>	Character string specifying the LaTeX font size for the condensed table (e.g., "footnotesize", "scriptsize").
<code>inParens</code>	Character string specifying what to display in parentheses: "tstat" (default) or "se" for standard errors.
<code>keepCoef1</code>	Logical; if TRUE, includes the first coefficient (typically the intercept) in the output. Default is FALSE.
<code>font.size.full</code>	Character string specifying the LaTeX font size for the full table. Default is "footnotesize".

## Details

This is the main user-facing function for the regression table workflow. It produces two outputs:

1. A condensed table showing only the covariates specified in `NameConversionMat` (or all if NULL)
2. A full table (if `saveFull = TRUE`) containing all coefficients in longtable format for multi-page output

**Value**

Invisibly returns NULL. The function is called for its side effect of writing LaTeX files to saveFolder.

**LaTeX Requirements**

The generated tables require the following LaTeX packages:

- longtable for full tables
- amssymb for checkmark symbols

**See Also**

[GetTableEntry](#) for extracting individual model results, [FullTransformer](#) for table cleaning, [Stargazer2FullTable](#) for longtable conversion

**Examples**

```
## Not run:
# Fit multiple models
fit1 <- lm(mpg ~ wt, data = mtcars)
fit2 <- lm(mpg ~ wt + hp, data = mtcars)
fit3 <- lm(mpg ~ wt + hp + cyl, data = mtcars)

# Generate tables
Tables2Tex(
  reg_list = list(fit1, fit2, fit3),
  clust_id = NULL,
  saveFolder = "./tables/",
  nameTag = "MPG_Models",
  tabCaption = "Determinants of Fuel Efficiency",
  model.names = c("Base", "Controls", "Full")
)

## End(Not run)
```

---

vcovCluster

---

*Cluster-robust covariance matrix estimator*


---

**Description**

Computes a clustered sandwich covariance matrix for a fitted model using the method of Arellano (1987). This provides standard errors that are robust to arbitrary within-cluster correlation.

**Usage**

```
vcovCluster(fm, clvar)
```

**Arguments**

fm	A fitted model object (typically from <code>lm()</code> or <code>glm()</code> ). For <code>polr</code> models from <b>MASS</b> , predictions are handled specially.
clvar	Character string giving the name of the clustering variable. This variable must exist in the data used to fit fm.

**Details**

The function applies the degrees-of-freedom correction  $(M/(M-1)) \times ((N-1)/(N-K))$  where  $M$  is the number of clusters,  $N$  is the number of observations, and  $K$  is the number of parameters.

**Value**

A  $K \times K$  covariance matrix where  $K$  is the number of model coefficients.

**References**

Arellano, M. (1987). Computing Robust Standard Errors for Within-Groups Estimators. *Oxford Bulletin of Economics and Statistics*, 49(4), 431-434.

**See Also**

[GetTableEntry](#) which uses this function for clustered standard errors, [vcovHC](#) for heteroskedasticity-robust covariance without clustering

**Examples**

```
## Not run:
data(mtcars)
fit <- lm(mpg ~ wt + hp, data = mtcars)

# Cluster by number of cylinders
V_clust <- vcovCluster(fit, "cyl")

# Use with coeftest for clustered standard errors
library(lmtest)
coeftest(fit, vcov. = V_clust)

## End(Not run)
```

---

WidenMargins

---

*Widen margins in a LaTeX table string*


---

**Description**

Adds horizontal margin adjustments to a LaTeX table by wrapping the table content in an `adjustwidth` environment from the **ragged2e** package. This is useful when tables are too wide to fit within standard margins.

**Usage**

```
WidenMargins(x)
```

**Arguments**

**x** Character vector containing the LaTeX table code, typically the output from [stargazer](#) or [Tables2Tex](#).

**Details**

The function extends margins by 0.5 inches on each side, effectively allowing the table to be 1 inch wider than the text width.

**Value**

A character vector with the modified LaTeX code. The `adjustwidth` environment is inserted inside the `table` environment.

**LaTeX Requirements**

The output requires the **ragged2e** package in your LaTeX preamble:

```
\usepackage{ragged2e}
```

**See Also**

[Tables2Tex](#) for generating LaTeX tables

**Examples**

```
## Not run:  
# Read an existing table and widen its margins  
tex_lines <- readLines("my_table.tex")  
tex_wide <- WidenMargins(tex_lines)  
writeLines(tex_wide, "my_table_wide.tex")  
  
## End(Not run)
```

# Index

`cols2numeric`, [2](#), [3](#), [4](#)  
`colSummary`, [2](#), [3](#), [14](#)  
  
`f2n`, [2](#), [3](#), [4](#)  
`fixZeroEndings`, [4](#)  
`FullTransformer`, [5](#), [15](#), [18](#)  
  
`geom_tile`, [11](#)  
`GetTableEntry`, [5](#), [6](#), [6](#), [16](#), [18](#), [19](#)  
  
`heatMap`, [8](#), [10–12](#), [14](#)  
`heatmap2`, [10](#), [10](#), [12](#)  
  
`image`, [12](#)  
`image.plot`, [8](#), [10](#)  
`image2`, [11](#), [12](#)  
  
`MakeHeatMap`, [3](#), [10](#), [13](#)  
  
`stargazer`, [15](#), [16](#), [19](#)  
`Stargazer2FullTable`, [15](#), [18](#)  
  
`Tables2Tex`, [5](#), [6](#), [8](#), [15](#), [16](#), [19](#), [20](#)  
  
`vcovCluster`, [8](#), [18](#)  
`vcovHC`, [19](#)  
  
`WidenMargins`, [19](#)