# Package 'inTextSummaryTable'

September 25, 2025

**Type** Package

**Title** Creation of in-Text Summary Table

**Version** 3.3.5

**Date** 2025-09-24

**Description** Creation of tables of summary statistics or counts for clinical data (for 'TLFs'). These tables can be exported as in-text table (with the 'flextable' package) for a Clinical Study Report (Word format) or a 'topline' presentation (PowerPoint format), or as interactive table (with the 'DT' package) to an html document for clinical data review.

**Imports** clinUtils (>= 0.1.0), cowplot, flextable (>= 0.5.5), ggplot2, ggrepel, magrittr, methods, officer, plyr, reshape2 (>= 1.4), scales, stats, utils

**Suggests** htmltools, knitr, rmarkdown, testthat, xml2, tools, tibble, extrafont

**URL** https://github.com/openanalytics/inTextSummaryTable

**BugReports** https://github.com/openanalytics/inTextSummaryTable/issues

**License** MIT + file LICENSE

**RoxygenNote** 7.3.3

**VignetteBuilder** knitr

**SystemRequirements** pandoc (to export an interactive summary table to a file)

**NeedsCompilation** no

**Author** Laure Cougnaud [aut, cre], Michela Pasetto [aut], Margaux Faes [rev] (tests), Open Analytics [cph]

**Maintainer** Laure Cougnaud <laure.cougnaud@openanalytics.eu>

**Repository** CRAN

**Date/Publication** 2025-09-25 09:30:08 UTC

# Contents

checkVar . . . 3
checkVarLabInclude . . . 4
combine . . . 4
combine.summaryTable . . . 5
combineColTotal . . . 6
combineVariables . . . 6
computeSummaryStatistics . . . 8
computeSummaryStatisticsTable . . . 10
convertSummaryStatisticsTableToFlextable . . . 16
convertVarFlag . . . 18
convertVarRowVarColVarToFactor . . . 18
convertVectToBinary . . . 19
createFlextableWithHeader . . . 20
cv . . . 21
export . . . 22
export.summaryTable . . . 22
exportFlextableToDocx . . . 27
exportSummaryStatisticsTableToDT . . . 28
exportSummaryStatisticsTableToFlextable . . . 30
formatPercentage . . . 33
formatSummaryStatisticsTable . . . 34
formatSummaryStatisticsTableFlextable . . . 35
geomCV . . . 38
geomMean . . . 39
geomSD . . . 40
geomSE . . . 41
getColorPaletteTable . . . 42
getDimPage . . . 42
getListing . . . 44
getMaxNDecimals . . . 45
getMaxNDecimalsData . . . 46
getNDecimals . . . 47
getNDecimalsData . . . 48
getNDecimalsRule . . . 49
getPatternPosition . . . 50
getStats . . . 50
getStatsData . . . 52
getSummaryStatisticsTable . . . 54
inTextSummaryTable . . . 63
inTextSummaryTable-common-args . . . 64
inTextSummaryTable-DT-args . . . 66
inTextSummaryTable-flextable-args . . . 67
inTextSummaryTable-stats . . . 68
inTextSummaryTable-stats-utility . . . 71
pageDimPresentation . . . 71
postProcessVarFlag . . . 72

se . . . . . 72
subjectProfileSummaryPlot . . . . . 73
subjectProfileSummaryTable . . . . . 77
summaryTable . . . . . 79
tableColorsPresentation . . . . . 81
tableColorsReport . . . . . 81
uniqueVarWithOrder . . . . . 82
writeTable . . . . . 82

**Index** **84**

---

`checkVar` *Check if variable(s) are present in reference: either in columns in a dataset or in reference set.*

---

## Description

Filter variables not present in the data or in reference set with a warning, and only returned filtered vector, or NULL if empty.

## Usage

```
checkVar(
  var,
  varLabel,
  varUncheck = NULL,
  varRef,
  refLabel = ifelse(!missing(varRef), "reference variable", "data"),
  data,
  msgType = c("warning", "error")
)
```

## Arguments

| | |
|---|---|
| `var` | String with variable to check. |
| `varLabel` | String with label for `var`, e.g. name of associated parameter. |
| `varUncheck` | (Named) character vector with extra variables in `var` which shouldn't be checked. |
| `varRef` | (Named) character vector with set of reference variables. |
| `refLabel` | String with label for the reference |
| `data` | Data.frame with data. |
| `msgType` | String with type of message returned, either a 'warning' (default) or an error. |

## Value

Depending on `msgType`:

- `warning`: warning is printed in the console, and a `var` filtered with element not in `data` or in `refSet` is returned. If filtered `var` is empty, NULL is returned.
- `error`: an error is triggered.

### Author(s)

Laure Cougnaud

---

## checkVarLabInclude *Check the* varLabInclude *variable.*

---

### Description

This function ensures that:

- variable name is included if more than one variable are specified
- variable name is not included if no variable is specified

### Usage

```
checkVarLabInclude(var, varLabInclude = length(var) > 1)
```

### Arguments

| | |
|---|---|
| var | String with variable to check. |
| varLabInclude | Logical, if TRUE the name of the summary statistic variable(s) (var) are included in the table. This is automatically set to TRUE if more than one variable(s) and is specified, and FALSE if only one variable is specified. |

### Value

(Updated) varLabInclude

### Author(s)

Laure Cougnaud

---

## combine *Combine objects*

---

### Description

Combine objects

### Usage

```
combine(...)
```

### Arguments

| | |
|---|---|
| ... | Extra parameters for the corresponding method. |

## See Also

combine.summaryTable to combine summaryTable objects.

---

combine.summaryTable *Combine summary statistics table*

---

## Description

- A new table is created, combining the tables by rows.
- The attributes of the combined summary table are obtained by combining the attributes of all summary tables (and removing duplicates).

## Usage

```
## S3 method for class 'summaryTable'
combine(..., summaryTables, combineVar = NULL, combineDir = c("row", "col"))
```

## Arguments

| | |
|---|---|
| ... | summaryTable objects. |
| summaryTables | List of summaryTable objects. |
| combineVar | (optional) String with name of a new variable tracking from which table each record originally come from.<br>The label for each table is extracted from the names of the summaryTables list, or 1:length(tables) if the list is not named.<br>If not specified (by default), the tables will be combined but the information on which input table: each record from the combined table belongs to will not be retained. |
| combineDir | String indicating on which direction: 'row' or 'col' (a.k.a column) the information on the table appartenance (combineVar) will be displayed when the table is exported. |

## Details

- In case only a set of tables contain categorical variable, in nested rows, so the variable and variableGroup variables, these variables are included as last in the 'row variables' attribute of the combined table.
- Only one row is retained for the columns totals per column variable (the first one in order of appearance).
  The column totals are not included if the column variable(s) are not the same across tables, or if the totals differ between tables.

## Value

A combined summaryTable.

### Author(s)

Laure Cougnaud

---

`combineColTotal` *Combine the column total for a combined summary table*

---

### Description

Combine the column total for a combined summary table

### Usage

```
combineColTotal(summaryTable, attrs)
```

### Arguments

| | |
|---|---|
| `summaryTable` | Combined summary table |
| `attrs` | Nested list with attributes of each summary table. |

### Value

summary table, with combined total columns

### Author(s)

Laure Cougnaud

---

`combineVariables` *Create a data.frame combining a set of records from same or multiple variables.*

---

### Description

This typically converts the data from a wide to a long format. For each variable, a subset of interest based on a condition can be specified.

### Usage

```
combineVariables(
  data,
  paramsList,
  newVar,
  labelVars = NULL,
  fctTest = "==",
  includeAll = FALSE,
  labelAll = "Any"
)
```

## Arguments

`data` Data.frame with dataset to consider for the summary table.

`paramsList` nested list of parameters, specifying how the records of interest should be selected.
There are two ways to select a subset of interest:

- by specifying one unique variable of interest with:
  - `var`: string with column of `data` of interest
  - `value`: value of `var` of interest (only used if `var` is specified).
    If not specified only the values different than NA and '' are considered.
  - `fctTest`: string with name or directly comparison function to apply on `var` to select subset of interest versus `value`.
    The function should take `var` as first parameter and `value` to compare to as second parameter and returns a logical vector with TRUE or FALSE (of length `var`) if the condition is fullfilled.
    If not specified, the records with `var` equal to `value` are retained (`fctTest` is set to '==').
  - label specification:
    - `label`: string with label for the condition, includde in the new 'variable' column.
      If not specified and:
      - `var` is specified: label is extracted from `labelVars` if available or set to `var` otherwise.
      - `var` is not specified: label should be specified.
    - `labelExtra`: string with extra label, will be concatenated with `label`
- by specifying a combination of variable of interest with:
  - `exprs`: string with expression of columns of `data` to select subset of interest
  - `label`: string with complete label for the group

`newVar` String with name of new variable to construct.

`labelVars` (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

`fctTest` Default function to use to compare `var` and `value` specified in each sublist of `paramsList`.
This is only used if `fctTest` is not specified in each sublist.

`includeAll` Logical, if TRUE (FALSE by default) include also the entire data as an additional subgroup.

`labelAll` String of group label for the entire data in case `includeAll` is TRUE.

## Value

Data.frame with records from `data` extracted based on the different conditions specified in `paramsList`. This data.frame contains an additional variable (labelled based on `newVar`) mentioning the specific

condition for which the record was extracted (based `label`, `labelExtra`, `labelVars`).
This variable is a factor whose levels are ordered based on the order of the condition specified in `paramsList`.

## Author(s)

Laure Cougnaud

---

`computeSummaryStatistics` *Compute summary statistics of interest of an unique variable of interest.*

---

## Description

Additionally, this function run extra checks on the data:

- an error message is triggered if any subject (identified by `subjectVar`) have different values in a continuous `var`
- an indicative message is triggered if multiple but identical records are available for `subjectVar` and a continuous `var`

## Usage

```
computeSummaryStatistics(
  data,
  var = NULL,
  varTotalInclude = FALSE,
  statsExtra = NULL,
  subjectVar = "USUBJID",
  filterEmptyVar = TRUE,
  type = "auto",
  checkVarDiffBySubj = c("error", "warning", "none"),
  msgLabel = NULL,
  msgVars = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | Data.frame with dataset to consider for the summary table. |
| `var` | Character vector with variable(s) of `data`, to compute statistics on.<br>If NULL (by default), counts by row/column variable(s) are computed.<br>To also return counts of the `rowVar` in case other `var` are specified, you can include: 'all' in the `var`.<br>Missing values, if present, are filtered (also for the report of number of subjects/records). |

varTotalInclude
: Logical (FALSE by default) Should the total across all categories of `var` be included for the count table? Only used if `var` is a categorical variable.

statsExtra
: (optional) Named list with functions for additional custom statistics to be computed.
Each function:

  - has as parameter, either: 'x': the variable (`var`) to compute the summary statistic on or 'data': the entire dataset
  - returns the corresponding summary statistic as a numeric vector

  For example, to additionally compute the coefficient of variation, this can be set to: `list(statCVPerc = function(x) sd(x)/mean(x)*100)` (or `cv`).

subjectVar
: String, variable of `data` with subject ID, 'USUBJID' by default.

filterEmptyVar
: Logical, if TRUE doesn't return any results if the variable is empty, otherwise return 0 for the counts and NA for summary statistics. Criterias to consider a variable empty are:

  - for a continuous variable: all missing (NA)
  - for a categorical variable: all missing or **category is included in the factor levels but not available in `data`**

  By default, an empty variable are filtered.

type
: String with type of table:

  - 'summaryTable': summary table with statistics for numeric variable
  - 'countTable': count table
  - 'auto' (by default): 'summaryTable' if the variable is numeric, 'countTable' otherwise

checkVarDiffBySubj
: String, 'error' (default), 'warning', or 'none'. Should an error, a warning, or nothing be produced if a continuous variable (`var`) contains different values for the same subject?

msgLabel
: (optional) String with label for the data (NULL by default), included in the message/warning for checks.

msgVars
: (optional) Character vector with columns of `data` containing extra variables (besides `var` and `subjectVar`) that should be included in the message/warning for checks.

## Value

Data.frame with summary statistics in columns, depending if `type` is:

- 'summary':
  - 'statN': number of subjects
  - 'statm': number of records
  - 'statMean': mean of `var`
  - 'statSD': standard deviation of `var`
  - 'statSE': standard error the mean of `var`

  - 'statMedian': median of `var`
  - 'statMin': minimum of `var`
  - 'statMax': maximum of `var`
- 'count':
  - 'variableGroup': factor with groups of `var` for which counts are reported
  - 'statN': number of subjects
  - 'statm': number of records

## Author(s)

Laure Cougnaud

---

`computeSummaryStatisticsTable` *Compute summary statistics for a specific dataset and variables of interest*

---

## Description

Compute summary statistics for a specific dataset and variables of interest

## Usage

```
computeSummaryStatisticsTable(
  data,
  var = NULL,
  varFlag = NULL,
  varInclude0 = FALSE,
  varLab = NULL,
  varLabInclude = length(var) > 1,
  varGeneralLab = "Variable",
  varSubgroupLab = "Variable group",
  varIgnore = NULL,
  varIncludeTotal = FALSE,
  varTotalInclude = FALSE,
  varTotalInSepRow = FALSE,
  colVar = NULL,
  colVarDataLevels = NULL,
  colVarTotal = colVar,
  colVarTotalPerc = colVarTotal,
  colTotalInclude = FALSE,
  colTotalLab = "Total",
  colInclude0 = FALSE,
  rowVar = NULL,
  rowVarDataLevels = NULL,
  rowVarLab = NULL,
```

```
  rowOrder = "auto",
  rowOrderTotalFilterFct = NULL,
  rowOrderCatLast = NULL,
  rowVarTotalInclude = NULL,
  rowVarTotalInSepRow = NULL,
  rowVarTotalByVar = NULL,
  rowVarTotalPerc = NULL,
  rowInclude0 = FALSE,
  type = "auto",
  subjectVar = "USUBJID",
  dataTotal = NULL,
  dataTotalPerc = dataTotal,
  dataTotalRow = NULL,
  dataTotalCol = NULL,
  stats = NULL,
  statsVarBy = NULL,
  statsExtra = NULL,
  statsGeneralLab = "Statistic",
  statsPerc = c("statN", "statm"),
  filterFct = NULL,
  labelVars = NULL,
  byVar = NULL,
  byVarLab = NULL,
  checkVarDiffBySubj = "error"
)
```

## Arguments

| | |
|---|---|
| `data` | Data.frame with dataset to consider for the summary table. |
| `var` | Character vector with variable(s) of `data`, to compute statistics on.<br>If NULL (by default), counts by row/column variable(s) are computed.<br>To also return counts of the `rowVar` in case other `var` are specified, you can include: 'all' in the `var`.<br>Missing values, if present, are filtered (also for the report of number of subjects/records). |
| `varFlag` | Character vector, subset of `var` with variable(s) of type 'flag' (with 'Y', 'N' or '' for empty/non specified value). Only the counts for records flagged (with 'Y') are retained. |
| `varInclude0` | Logical, should rows with no counts for the count `var` or `varFlag` variable(s) be included in the table? Either:<br>• logical of length 1, if TRUE (FALSE by default) rows with no count are included for all `var`<br>• a character vector containing categorical `var` for which zero counts rows should be included |
| `varLab` | Named character vector with label for each variable specified in `var`. By default, extracted from the `labelVars`. if not available, `var` is used. |

varLabInclude Logical, if TRUE the name of the summary statistic variable(s) (var) are included in the table. This is automatically set to TRUE if more than one variable(s) and is specified, and FALSE if only one variable is specified.

varGeneralLab String with general label for variable specified in var. In case of multiple variable in var, this will be included in the table header (see 'rowVarLab' attribute of the output).

varSubgroupLab String with general label for sub-group of categorical variable(s) for count table, 'Variable group' by default. This will be included in the final table header (see 'rowVarLab' attribute of the output).

varIgnore Vector with elements to ignore in the var variable(s). The data records with such elements in var are **filtered** from the data at the start of the workflow.

varIncludeTotal

This argument is deprecated, please use: 'varTotalInclude' instead.

varTotalInclude

Should the total across all categories of var be included for the count table? Only used for categorical variables (and var not 'all'). Either:

- logical of length 1, if TRUE (FALSE by default) include the total for all categorical var
- a character vector containing categorical var for which the total should be included

varTotalInSepRow

Logical, should the total per variable be included in a separated row (by default) or in the row containing the header of the variable?

colVar Character vector with variable(s) to be included in columns. If multiple variables are specified, the variables should be sorted in hierarchical order, and are included in multi-columns layout.
Use: 'variable' to include the variables to summarize: var (if multiple) in different columns.

colVarDataLevels

Data.frame with unique combinations of colVar to be included in columns. Each column should correspond to colVar and as factor if the elements should be ordered in the final table.

colVarTotal String with column(s) considered to compute the total by, reported in the header of the table, by default same as colVar. Use: 'variable' to compute total by var (if multiple).

colVarTotalPerc

String with column(s) considered to compute the total by, used as denominator for the percentage computation, by default same as colVarTotal. Use: 'variable' to compute total by var (if multiple).

colTotalInclude

Logical, if TRUE (FALSE by default) include the summary statistics across columns in a separated column.

colTotalLab String, label for the total column 'Total' by default.

`colInclude0` Logical, if TRUE (FALSE by default), include columns with no records, based on all combinations of the `columnVar` (assuming nested variable(s)). If variable(s) are not nested, possible combinations can be specified via `colVarDataLevels`.

`rowVar` Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table.

`rowVarDataLevels`
Data.frame with unique combinations of `rowVar` to be included in columns. Each column should correspond to `colVar` and as factor if the elements should be ordered in the final table.

`rowVarLab` Named character vector with label for the `rowVar` variable(s).

`rowOrder` Specify how the rows should be ordered in the final table, either a:

- String among:
  - 'auto' (by default): if the variable is a factor, keep its order, otherwise order alphabetically
  - 'alphabetical': order alphabetically
  - 'total': order rows in decreasing order of the total number of subjects across all columns for this specific category.
- Function with input the summary table and output the ordered elements of the `rowVar`

To specify different ordering methods for different `rowVar`, specify a list of such elements, named with the `rowVar` variable. For the table output of `computeSummaryStatisticsTable` (long format), this order is also reflected in the `levels` of the row factor variable.

`rowOrderTotalFilterFct`
Function used to filter the data used to order the rows based on total counts (in case `rowOrder` is 'total'), To order rows based on one specific column category, e.g. to order based on the counts in the treatment column: function(x) subset(x, TRTP == "treatmentX")

`rowOrderCatLast`
String with category to be printed in the last row of each `rowVar` (if any, set to NULL if none).

`rowVarTotalInclude`
Character vector with `rowVar` for which the total should be reported.
If the higher row variable is specified, the total across all rows is reported.
For the export, these variable(s) are formatted as factor with **'Total' as the first level**.

`rowVarTotalInSepRow`
Character vector with `rowVarTotalInclude` (not in `rowVarInSepCol`) for which the total should be included in a separated row labelled 'Total'. Otherwise (by default) the total is included in the header row of each category.

`rowVarTotalByVar`
Character vector with a row variable used to categorize the row total.
Note that this is only used if row total(s) is/are requested via `rowVarTotalInclude`, and this variable should also be included in `rowVar`. This can be specified also for a specific row variable if the vector is named.

For example: `c(ADECOD = "AESEV")` to compute total by severity for row adverse event term in a typical adverse event count table (by System Organ Class and Adverse Event Term).

`rowVarTotalPerc`

Character vector with row variables by which the total should be computed for the denominator for the percentage computation. By default the total is only computed only by column (NULL by default). If the total should be based on the total number of records per variable, `rowVarTotalPerc` should be set to 'variable'.

`rowInclude0` Logical, if TRUE (FALSE by default), include rows with no records, based on all combinations of the `rowVar` (assuming nested variable(s)).

`type` String with type of table:

- 'summaryTable': summary table with statistics for numeric variable
- 'countTable': count table
- 'auto' (by default): 'summaryTable' if the variable is numeric, 'countTable' otherwise

`subjectVar` String, variable of `data` with subject ID, 'USUBJID' by default.

`dataTotal` Data.frame used to extract the Total number of subject per column in column header ('N = [X]'). It should contain the variables specified by `colVarTotal`. If not specified, the total number of subjects is extracted from the `data`.

`dataTotalPerc` Data.frame used to extract the total counts per column for the computation of the percentage.
By default, `dataTotal` is used.
It should contain the variables specified by `colVarTotalPerc`.

`dataTotalRow` Data.frame used to extract the total count across all elements of the row variable, list of such data.frame for each `rowVar` variable.
If the dataset is specified by row variable, the list should be named with: variable X if the total across elements of variable X should be included. By default, `data` is used.

`dataTotalCol` Data.frame from which the total across columns is extracted (in case `colTotalInclude` is TRUE) or list of such data.frame for each `rowVar` variable.
If the dataset is specified by row variable, the list should be named with: with:

- last row variable: for the dataset used in the total column for the most nested row variable
- higher row variable (X+1): for the dataset used for the total column and row total of X
- 'total': for the dataset used for the total column and general row total

If only a subset of the variables is specified in this list, `data` is used for the remaining variable(s) (or 'total') if needed.
This dataset (the one for 'total' if a list) is also used for:

- the header of the total column in case `dataTotal` is not specified
- the denominator of the percentages in the total column in case `dataTotalPerc` is not specified

By default, `data` is used.

`stats` (optional) Statistic(s) of interest to compute, either:

- string with the name of a default set of statistics available in the package, see section 'Formatted statistics' in `in-text table statistics`.
  See the corresponding `type` parameter of the `getStatsData` for more information on how the statistic is internally extracted.
- (expert mode) named list of language object (see `is.language`) of base summary statistics of interest, see section: 'Base statistics' in `in-text table statistics`.
  The names are reported in the header.
  If `stats` if of length 1, the name of the summary statistic is not included in the table.
  The statistics can be specified separately:
  - for each `var` (if multiple), by naming each element of the list: `list(varName1 = list(...), varName2 = list())`
  - and/or for each element in: `statsVarBy`, by naming each sublist.

`statsVarBy` String with variable in `rowVar`/`colVar` which the statistics should be computed by.
In this case, `stats` (nested list or not) should be additionally nested to specify the statistics for each element in `statsVarBy`.

`statsExtra` (optional) Named list with functions for additional custom statistics to be computed.
Each function:

- has as parameter, either: 'x': the variable (`var`) to compute the summary statistic on or 'data': the entire dataset
- returns the corresponding summary statistic as a numeric vector

For example, to additionally compute the coefficient of variation, this can be set to: `list(statCVPerc = function(x) sd(x)/mean(x)*100)` (or `cv`).

`statsGeneralLab`
String with general label for statistics, 'Statistic' by default. Only included if no `statsVar` if longer than 1.

`statsPerc` String with 'base statistical variable' used to compute the percentage, either:

- 'statN' (by default): the number of subjects
- 'statm': the number of records

`filterFct` (optional) Function taking as input the summary table with computed statistics and returning a subset of the summary table.
Note: The filtering function should also handle records with :

- total for the column header: `isTotal` set to TRUE, and `colVar`/`rowVar` is NA.
  For example: `filterFct = function(data) subset(data, isTotal & myColVar == "group 1")`
- `rowVar`/`colVar` set to 'Total'/`colTotalLab` if `rowVarTotalInclude`/`colTotalInclude` is specified

`labelVars` (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

byVar Variable(s) of data for which separated table(s) should be created.

byVarLab String with label for byVar, used to set the names of the output list of table(s).

checkVarDiffBySubj
String, 'error' (default), 'warning', or 'none'. Should an error, a warning, or nothing be produced if a continuous variable (var) contains different values for the same subject (by row/column)?

## Value

An object summaryTable or list of such objects if byVar is specified.

## Author(s)

Laure Cougnaud

---

# convertSummaryStatisticsTableToFlextable

*Convert summary statistics table to flextable*

---

## Description

Convert summary statistics table to flextable

## Usage

```
convertSummaryStatisticsTableToFlextable(
  summaryTable,
  colHeaderMerge = TRUE,
  landscape = (style == "presentation"),
  margin = 1,
  rowPadBase = 14.4,
  title = NULL,
  footer = NULL,
  style = "report",
  colorTable = getColorPaletteTable(style = style),
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  fontsize = switch(style, report = 8, presentation = 10),
  file = NULL,
  pageDim = NULL,
  columnsWidth = NULL
)
```

## Arguments

| | |
|---|---|
| summaryTable | A summaryTable object. |
| colHeaderMerge | (flextable output) Logical, if TRUE (FALSE by default) the column header is merged. |
| landscape | (flextable output) Logical, if TRUE the file is in landscape format.<br>By default: FALSE if style is 'report' and TRUE if style is 'presentation'. |
| margin | (flextable output) Margin in the document in inches (1 by default). This is used to specify the width of the table, from: [pageDim[1] - 2 * margin]. |
| rowPadBase | (flextable output) Base padding for row (in points), 14.4 by default (corresponds to 0.2 inches) |
| title | Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. If multiple are specified, specified for each element of byVar (in order of the levels). |
| footer | (flextable output) Character vector with footer(s) for the table. Set to NULL (by default) if no footer should be included. |
| style | (flextable output) String with table style, either 'report' or 'presentation'. This parameter affects the fontsize, font family, color of the text and background, and table dimensions of the table. |
| colorTable | (flextable output) Named character vector with color for the table background/body/text/line, e.g. created with the getColorPaletteTable function. |
| fontname | (flextable output) String with font name, by default: 'Times' if style is 'report' and 'Tahoma' if style is 'presentation'. |
| fontsize | (flextable output) Integer with font size, by default: 8 if style is 'report' and 10 if style is 'presentation'. |
| file | String with path of the file where the table should be exported. The file should have the extension: '.html'. If NULL, the summary table is not exported but only returned as output. If byVar is specified, each table is exported to a separated file with the suffix: 'file_[i].html' with i the index of the file. |
| pageDim | Numeric vector of length 2 with page width and height.<br>Depending on outputType:<br>• 'flextable': in inches<br>• 'DT': in number of rows in the table.<br>Currently only the height is used (e.g. c(NA, 4)) |
| columnsWidth | (expert mode) Column widths of the table. This is only used for flextable and DT tables.<br>For flextable, note that the widths should be set to fit into the document page (see getDimPage). |

## Value

flextable object with summary table If summaryTable is a list of summary tables, returns a list of flextable.

### Author(s)

Laure Cougnaud

---

`convertVarFlag` *Convert flag variable to a format such as only the flagged records are counted in the summary table.*

---

### Description

Convert flag variable to a format such as only the flagged records are counted in the summary table.

### Usage

```
convertVarFlag(x)
```

### Arguments

| | |
|---|---|
| x | Character or factor variable with flag variable, should contain elements: 'Y' and 'N', or '' (for missing value). |

### Value

Formatted factor variable with levels: 'Y' & 'N'. Empty strings have been converted to NA.

### Author(s)

Laure Cougnaud

---

`convertVarRowVarColVarToFactor`
*Convert* `rowVar`, `colVar` *and character* `var` *in* `data` *to factor*

---

### Description

Convert `rowVar`, `colVar` and character `var` in `data` to factor

### Usage

```
convertVarRowVarColVarToFactor(data, rowVar = NULL, colVar = NULL, var = NULL)
```

### Arguments

| | |
|---|---|
| data | Data.frame with dataset to consider for the summary table. |
| rowVar | Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table. |
| colVar | Character vector with variable(s) to be included in columns. If multiple variables are specified, the variables should be sorted in hierarchical order, and are included in multi-columns layout.<br>Use: 'variable' to include the variables to summarize: var (if multiple) in different columns. |
| var | Character vector with variable(s) of data, to compute statistics on.<br>If NULL (by default), counts by row/column variable(s) are computed.<br>To also return counts of the rowVar in case other var are specified, you can include: 'all' in the var.<br>Missing values, if present, are filtered (also for the report of number of subjects/records). |

### Value

Updated data

### Author(s)

Laure Cougnaud

---

## convertVectToBinary *Convert vector to a bincode of 0/1 based on consecutive values in the vector.*

---

### Description

Convert vector to a bincode of 0/1 based on consecutive values in the vector.

### Usage

```
convertVectToBinary(x)
```

### Arguments

| | |
|---|---|
| x | Vector. |

### Value

Integer vector of same length than x.

### Author(s)

Laure Cougnaud

### Examples

```
x <- c("group1", "group1", "group1", "group2", "group2", "group3", "group4", "group4")
convertVectToBinary(x = x)
```

---

`createFlextableWithHeader` *Create a flextable, setting the column names to syntactic names if it is not the case.*

---

### Description

Create a flextable, setting the column names to syntactic names if it is not the case.

### Usage

```
createFlextableWithHeader(
  data,
  headerDf = NULL,
  title = NULL,
  includeRownames = TRUE,
  colHeaderMerge = TRUE
)
```

### Arguments

| | |
|---|---|
| `data` | Data.frame with data. |
| `headerDf` | (optional) Data.frame with header. This should contain the same number of columns than `data` (+ if `includeRownames` is TRUE) and optionally multiple rows. Neighbouring cells with same content will be represented merged in the output. |
| `title` | Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. |
| `includeRownames` | Logical, if TRUE (by default) rownames are included in the `flextable` object. |
| `colHeaderMerge` | (flextable output) Logical, if TRUE (FALSE by default) the column header is merged. |

### Value

list with:

- 'ft': `flextable`
- 'colsData': Named vector with original column names, with names set to new syntactic names.

### Author(s)

Laure Cougnaud

---

`cv` *Compute the percentage coefficient of variation, (in a scale from 0 to 100).*

---

### Description

The coefficient of variation is computed as: $\frac{\sigma(x)}{\bar{x}} * 100$, with:

- $\sigma(x)$: standard deviation of `x`
- $\bar{x}$: arithmetic mean of `x`

### Usage

```
cv(x, na.rm = FALSE)
```

### Arguments

| | |
|---|---|
| `x` | Numeric vector. |
| `na.rm` | Logical, should NA value(s) be removed (FALSE by default)? |

### Value

Numeric vector of length 1 with coefficient of variation.

### Author(s)

Laure Cougnaud

### See Also

Other stats utility functions: `geomCV()`, `geomMean()`, `geomSD()`, `geomSE()`, `se()`

### Examples

```
# coefficient of variation of normal distribution tends to 100%
cv(rnorm(n = 1000, mean = 1, sd = 1))
```

## export *Export an object*

### Description

Export an object

### Usage

```
export(...)
```

### Arguments

| | |
|---|---|
| ... | Extra parameters for the corresponding method. |

### See Also

export.summaryTable to export summaryTable objects.

## export.summaryTable *Export a summary table to* docx, pptx *or* html *format (interactive table)*

### Description

The use of export is recommended.
exportSummaryStatisticsTable is retained for back-compatibility.

### Usage

```
## S3 method for class 'summaryTable'
export(...)

exportSummaryStatisticsTable(
  summaryTable,
  rowVar = getAttribute(summaryTable, "rowVar"),
 rowVarLab = getAttribute(summaryTable, "rowVarLab", default = getLabelVar(rowVar,
    labelVars = labelVars)),
  rowVarInSepCol = NULL,
  rowVarFormat = NULL,
  rowVarTotalInclude = getAttribute(summaryTable, "rowVarTotalInclude"),
  rowTotalLab = NULL,
  rowVarTotalInSepRow = getAttribute(summaryTable, "rowVarTotalInSepRow"),
  rowAutoMerge = TRUE,
  colVar = getAttribute(summaryTable, "colVar"),
  colTotalLab = getAttribute(summaryTable, "colTotalLab", default = "Total"),
```

```
  colHeaderTotalInclude = TRUE,
  colHeaderMerge = TRUE,
  statsVar = getAttribute(summaryTable, "statsVar"),
 statsLayout = getAttribute(summaryTable, "statsLayout", default = ifelse("DT" %in%
    outputType, "col", "row")),
  statsValueLab = "StatisticValue",
  statsLabInclude = NULL,
  emptyValue = "-",
  labelVars = NULL,
  file = NULL,
  title = NULL,
  outputType = "flextable",
  pageDim = NULL,
  columnsWidth = NULL,
  landscape = (style == "presentation"),
  margin = 1,
  rowPadBase = 14.4,
  footer = NULL,
  style = "report",
  colorTable = getColorPaletteTable(style = style),
  fontsize = switch(style, report = 8, presentation = 10),
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  vline = "none",
  hline = "auto",
  expandVar = NULL,
  noEscapeVar = NULL,
  barVar = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `...` | (DT output) Extra parameters passed to the `getClinDT` |
| `summaryTable` | A `summaryTable` object. |
| `rowVar` | Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table. |
| `rowVarLab` | Named character vector with label for the `rowVar` variable(s). |
| `rowVarInSepCol` | Character vector with `rowVar` that should be included in separated columns. By default (NULL), all row variables are nested in the first column of the table. To include the groups within a `var` variable in a separated column, set: `rowVarInSepCol == 'variableGroup'`. |
| `rowVarFormat` | (flextable output) Named list with special formatting for the `rowVar`. Currently, only possibility is to set the variable elements in bold, with: list(var1 = "bold"). (Use 'variable' for `var` or 'variableGroup' for group within categorical variables.) |

`rowVarTotalInclude`
: Character vector with `rowVar` for which the total should be reported.
If the higher row variable is specified, the total across all rows is reported.
For the export, these variable(s) are formatted as factor with **'Total' as the first level**.

`rowTotalLab`
: (flextable output) string with label for the row with total.

`rowVarTotalInSepRow`
: Character vector with `rowVarTotalInclude` (not in `rowVarInSepCol`) for which the total should be included in a separated row labelled 'Total'. Otherwise (by default) the total is included in the header row of each category.

`rowAutoMerge`
: (flextable output) Logical, if TRUE (by default) automatically merge rows, e.g. in case there is only one sub-category (e.g. categorical variable with only one group) or only one statistic per category.

`colVar`
: Character vector with variable(s) to be included in columns. If multiple variables are specified, the variables should be sorted in hierarchical order, and are included in multi-columns layout.
Use: 'variable' to include the variables to summarize: `var` (if multiple) in different columns.

`colTotalLab`
: String, label for the total column 'Total' by default.

`colHeaderTotalInclude`
: Logical, if TRUE include the total of number of patients (`'statN'`) in the column header.

`colHeaderMerge`
: (flextable output) Logical, if TRUE (FALSE by default) the column header is merged.

`statsVar`
: Character vector with columns of `summaryTable` with statistic variables. For the export: if not specified, all columns of `data` besides row, column variables, 'variable', 'variableGroup' and 'isTotal' are considered.

`statsLayout`
: String with layout for the statistics names (in case more than one statistic is included), among:
  - row (by default for 'flextable' output):
    All statistics are included in different rows in the first column of the table (after the row variable(s))
  - 'col' (by default for 'DT' output):
    Statistics are included in separated columns (last row of the header).
    This option is not compatible with categorical variable(s).
  - 'rowInSepCol':
    Statistics are included in different rows, but in a separated column than the `rowVar` variable(s)

`statsValueLab`
: String with label for the statistic value, 'StatisticValue' by default.
This is only included in the table if the statistics provided in `stats` are not named and if no `colVar` is specified.

`statsLabInclude`
: Logical, if TRUE include the statistic label in the table.
By default only included if more than one statistic variables are available in the table.

`emptyValue` String with placeholder used to fill the table for missing values, '-' by default. This value is typically used e.g. if not all statistics are computed for all specified row/col/var variables.

`labelVars` (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

`file` (Optional) Name of the file the table should be exported to, either:

- string (of length 1). In this case, depending on the file extension, the following is exported:
  - 'txt': summary table in long format ('data.frame-base' `outputType`)
  - 'docx': summary table in final format is exported ('flextable' `outputType`)
  - 'html': interactive summary table is exported ('DT' `outputType`)
- named character vector in case of multiple exports. The names should correspond to the options in `outputType`:
  - for 'data.frame-base' and 'data.frame': filename with 'txt' extension
  - for 'flextable': filename with 'docx' extension
  - for 'DT': filename with 'html' extension

If NULL (by default), the summary table is not exported but only returned as output. If `byVar` is specified, each table is exported to a separated file with the suffix: 'file_[i].[ext]' with i the index of the file (and [ext] the file extension).

`title` Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. If multiple are specified, specified for each element of `byVar` (in order of the levels).

`outputType` String with output type:

- 'flextable' (by default): `flextable` object, with format for CSR, compatible with Word/PowerPoint export
- 'DT': `datatable` interactive table, compatible with html export
- 'data.frame': data.frame in wide format (with elements in `colVar` in different columns)
- 'data.frame-base': data.frame in long format (with elements in `colVar` in different rows), useful for QC

`pageDim` Numeric vector of length 2 with page width and height.
Depending on `outputType`:

- 'flextable': in inches
- 'DT': in number of rows in the table.
  Currently only the height is used (e.g. `c(NA, 4)`)

`columnsWidth` (expert mode) Column widths of the table. This is only used for flextable and DT tables.
For flextable, note that the widths should be set to fit into the document page (see `getDimPage`).

`landscape` (flextable output) Logical, if TRUE the file is in landscape format.
By default: FALSE if `style` is 'report' and TRUE if `style` is 'presentation'.

| | |
|---|---|
| margin | (flextable output) Margin in the document in inches (1 by default). This is used to specify the width of the table, from: [pageDim[1] - 2 * margin]. |
| rowPadBase | (flextable output) Base padding for row (in points), 14.4 by default (corresponds to 0.2 inches) |
| footer | (flextable output) Character vector with footer(s) for the table. Set to NULL (by default) if no footer should be included. |
| style | (flextable output) String with table style, either 'report' or 'presentation'. This parameter affects the fontsize, font family, color of the text and background, and table dimensions of the table. |
| colorTable | (flextable output) Named character vector with color for the table background/body/text/line, e.g. created with the getColorPaletteTable function. |
| fontsize | (flextable output) Integer with font size, by default: 8 if style is 'report' and 10 if style is 'presentation'. |
| fontname | (flextable output) String with font name, by default: 'Times' if style is 'report' and 'Tahoma' if style is 'presentation'. |
| vline | (flextable output) String mentioning how vertical lines should be included in the body of the table, either:<br>• 'none' (default): no vertical lines included<br>• 'auto': vertical lines included between sub-groups |
| hline | (flextable output) String mentioning how horizontal lines should be included in the body of the table, either:<br>• 'none': no horizontal lines included<br>• 'auto' (default): horizontal lines included between sub-groups |
| expandVar | (DT output) Character vector with variables of the summary table which should be expanded in the data. |
| noEscapeVar | (DT output) Character vector with variables of summaryTable which shouldn't be escaped in the table (e.g. containing URLs). |
| barVar | (DT output) Character vector with variables of summaryTable that should be represented as a bar. |

## Value

Depending on the outputType:

- 'data.frame-base': input summary table in a long format with all computed statistics
- 'data.frame': summary table in a wide format ( different columns for each colVar), with specified labels
- 'flextable' (by default): flextable object with summary table
- 'DT': datatable object with summary table

If multiple outputType are specified, a list of those objects, named by outputType.
If byVar is specified, each object consists of a list of tables, one for each element in byVar.

### Functions

- `export(summaryTable)`: export `summaryTable` object

### Author(s)

Laure Cougnaud

---

`exportFlextableToDocx` *Export flextable to docx file*

---

### Description

Export flextable to docx file

### Usage

```
exportFlextableToDocx(
  object,
  file,
  landscape = FALSE,
  breaksAfter = if (!inherits(object, "flextable")) seq_along(object) else 1
)
```

### Arguments

| | |
|---|---|
| `object` | `flextable` object, or list of such objects |
| `file` | String with path of the file where the table should be exported. The file should have the extension: '.html'. If NULL, the summary table is not exported but only returned as output. If `byVar` is specified, each table is exported to a separated file with the suffix: 'file_[i].html' with i the index of the file. |
| `landscape` | (flextable output) Logical, if TRUE the file is in landscape format.<br>By default: FALSE if `style` is 'report' and TRUE if `style` is 'presentation'. |
| `breaksAfter` | In case `object` is list: integer vector with indices of list item after which a page break should be included in the final document. |

### Value

no returned value, the `object` is exported to a docx file.

### Author(s)

Laure Cougnaud

exportSummaryStatisticsTableToDT *Export summary table to an interactive DT table, e.g. to be exported into an html document.*

## Description

Export summary table to an interactive DT table, e.g. to be exported into an html document.

## Usage

```
exportSummaryStatisticsTableToDT(
  summaryTable,
  rowVar = getAttribute(summaryTable, "rowVar"),
 rowVarLab = getAttribute(summaryTable, "rowVarLab", default = getLabelVar(rowVar,
    labelVars = labelVars)),
  rowVarInSepCol = NULL,
  statsVar = getAttribute(summaryTable, "statsVar"),
  statsLayout = getAttribute(summaryTable, "statsLayout", default = "col"),
  statsValueLab = "StatisticValue",
  title = NULL,
  expandVar = NULL,
  noEscapeVar = NULL,
  barVar = NULL,
  pageDim = NULL,
  labelVars = NULL,
  file = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| summaryTable | Summary table as provided by the formatSummaryStatisticsTable. |
| rowVar | Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table. |
| rowVarLab | Named character vector with label for the rowVar variable(s). |
| rowVarInSepCol | Character vector with rowVar that should be included in separated columns. By default (NULL), all row variables are nested in the first column of the table. To include the groups within a var variable in a separated column, set: rowVarInSepCol == 'variableGroup'. |
| statsVar | Character vector with columns of summaryTable with statistic variables. For the export: if not specified, all columns of data besides row, column variables, 'variable', 'variableGroup' and 'isTotal' are considered. |
| statsLayout | String with layout for the statistics names (in case more than one statistic is included), among: |

- row (by default for 'flextable' output):
  All statistics are included in different rows in the first column of the table (after the row variable(s))
- 'col' (by default for 'DT' output):
  Statistics are included in separated columns (last row of the header).
  This option is not compatible with categorical variable(s).
- 'rowInSepCol':
  Statistics are included in different rows, but in a separated column than the `rowVar` variable(s)

`statsValueLab` String with label for the statistic value, 'StatisticValue' by default.
This is only included in the table if the statistics provided in `stats` are not named and if no `colVar` is specified.

`title` Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. If multiple are specified, specified for each element of `byVar` (in order of the levels).

`expandVar` (DT output) Character vector with variables of the summary table which should be expanded in the data.

`noEscapeVar` (DT output) Character vector with variables of `summaryTable` which shouldn't be escaped in the table (e.g. containing URLs).

`barVar` (DT output) Character vector with variables of `summaryTable` that should be represented as a bar.

`pageDim` Numeric vector of length 2 with page width and height.
Depending on `outputType`:

- 'flextable': in inches
- 'DT': in number of rows in the table.
  Currently only the height is used (e.g. `c(NA, 4)`)

`labelVars` (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

`file` String with path of the file where the table should be exported. The file should have the extension: '.docx'. If NULL, the summary table is not exported but only returned as output. If `byVar` is specified, each table is exported to a separated file with the suffix: 'file_[i].docx' with i the index of the file.

`...` (DT output) Extra parameters passed to the `getClinDT`

## Value

A `datatable` object.

## Author(s)

Laure Cougnaud

`exportSummaryStatisticsTableToFlextable` *Export summary table to a flextable object, e.g. to be exported in Word or PowerPoint.*

## Description

Export summary table to a flextable object, e.g. to be exported in Word or PowerPoint.

## Usage

```
exportSummaryStatisticsTableToFlextable(
  summaryTable,
  rowVar = getAttribute(summaryTable, "rowVar"),
  rowVarInSepCol = NULL,
  rowVarTotalInclude = getAttribute(summaryTable, "rowVarTotalInclude"),
  statsLayout = getAttribute(summaryTable, "statsLayout", default = "row"),
  statsVar = getAttribute(summaryTable, "statsVar"),
  statsLabInclude = getAttribute(summaryTable, "statsLabInclude", default =
    length(statsVar) > 1),
 rowVarLab = getAttribute(summaryTable, "rowVarLab", default = getLabelVar(rowVar,
    labelVars = labelVars)),
  rowVarTotalInSepRow = NULL,
  vline = c("none", "auto"),
  hline = c("auto", "none"),
  rowAutoMerge = TRUE,
  rowVarFormat = NULL,
  rowTotalLab = NULL,
  colHeaderMerge = TRUE,
  landscape = (style == "presentation"),
  margin = 1,
  rowPadBase = 14.4,
  title = NULL,
  footer = NULL,
  style = "report",
  colorTable = getColorPaletteTable(style = style),
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  fontsize = switch(style, report = 8, presentation = 10),
  file = NULL,
  pageDim = NULL,
  columnsWidth = NULL,
  labelVars = NULL
)
```

## Arguments

| | |
|---|---|
| `summaryTable` | Summary table as provided by the `formatSummaryStatisticsTable` |

`rowVar` Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table.

`rowVarInSepCol` Character vector with `rowVar` that should be included in separated columns. By default (NULL), all row variables are nested in the first column of the table.
To include the groups within a `var` variable in a separated column, set: `rowVarInSepCol == 'variableGroup'`.

`rowVarTotalInclude`
Character vector with `rowVar` for which the total should be reported.
If the higher row variable is specified, the total across all rows is reported.
For the export, these variable(s) are formatted as factor with **'Total' as the first level**.

`statsLayout` String with layout for the statistics names (in case more than one statistic is included), among:

- row (by default for 'flextable' output):
  All statistics are included in different rows in the first column of the table (after the row variable(s))
- 'col' (by default for 'DT' output):
  Statistics are included in separated columns (last row of the header).
  This option is not compatible with categorical variable(s).
- 'rowInSepCol':
  Statistics are included in different rows, but in a separated column than the `rowVar` variable(s)

`statsVar` Character vector with columns of `summaryTable` with statistic variables. For the export: if not specified, all columns of `data` besides row, column variables, 'variable', 'variableGroup' and 'isTotal' are considered.

`statsLabInclude`
Logical, if TRUE include the statistic label in the table.
By default only included if more than one statistic variables are available in the table.

`rowVarLab` Named character vector with label for the `rowVar` variable(s).

`rowVarTotalInSepRow`
Character vector with `rowVarTotalInclude` (not in `rowVarInSepCol`) for which the total should be included in a separated row labelled 'Total'. Otherwise (by default) the total is included in the header row of each category.

`vline` (flextable output) String mentioning how vertical lines should be included in the body of the table, either:

- 'none' (default): no vertical lines included
- 'auto': vertical lines included between sub-groups

`hline` (flextable output) String mentioning how horizontal lines should be included in the body of the table, either:

- 'none': no horizontal lines included
- 'auto' (default): horizontal lines included between sub-groups

`rowAutoMerge` (flextable output) Logical, if TRUE (by default) automatically merge rows, e.g. in case there is only one sub-category (e.g. categorical variable with only one group) or only one statistic per category.

`rowVarFormat` (flextable output) Named list with special formatting for the `rowVar`. Currently, only possibility is to set the variable elements in bold, with: list(var1 = "bold"). (Use 'variable' for `var` or 'variableGroup' for group within categorical variables.)

`rowTotalLab` (flextable output) string with label for the row with total.

`colHeaderMerge` (flextable output) Logical, if TRUE (FALSE by default) the column header is merged.

`landscape` (flextable output) Logical, if TRUE the file is in landscape format.
By default: FALSE if `style` is 'report' and TRUE if `style` is 'presentation'.

`margin` (flextable output) Margin in the document in inches (1 by default). This is used to specify the width of the table, from: [`pageDim[1]` - 2 * margin].

`rowPadBase` (flextable output) Base padding for row (in points), 14.4 by default (corresponds to 0.2 inches)

`title` Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. If multiple are specified, specified for each element of `byVar` (in order of the levels).

`footer` (flextable output) Character vector with footer(s) for the table. Set to NULL (by default) if no footer should be included.

`style` (flextable output) String with table style, either 'report' or 'presentation'. This parameter affects the fontsize, font family, color of the text and background, and table dimensions of the table.

`colorTable` (flextable output) Named character vector with color for the table background/body/text/line, e.g. created with the `getColorPaletteTable` function.

`fontname` (flextable output) String with font name, by default: 'Times' if `style` is 'report' and 'Tahoma' if `style` is 'presentation'.

`fontsize` (flextable output) Integer with font size, by default: 8 if `style` is 'report' and 10 if `style` is 'presentation'.

`file` String with path of the file where the table should be exported. The file should have the extension: '.html'. If NULL, the summary table is not exported but only returned as output. If `byVar` is specified, each table is exported to a separated file with the suffix: 'file_[i].html' with i the index of the file.

`pageDim` Numeric vector of length 2 with page width and height.
Depending on `outputType`:

- 'flextable': in inches
- 'DT': in number of rows in the table.
  Currently only the height is used (e.g. `c(NA, 4)`)

`columnsWidth` (expert mode) Column widths of the table. This is only used for flextable and DT tables.
For flextable, note that the widths should be set to fit into the document page (see `getDimPage`).

| | |
|---|---|
| labelVars | (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.<br>Labels specified via dedicated parameter: e.g. rowVarLab, colVarLab, varLab have priority on this parameter. |

## Value

flextable object with summary table If summaryTable is a list of summary tables, returns a list of flextable.

## Author(s)

Laure Cougnaud

---

# formatPercentage *Format a percentage.*

---

## Description

The following rules are used:

- percentage = 0%: '0'
- 0% < percentage < 0.1%: '<0.1'
- 99.9% < percentage < 100%: '>99.9'
- percentage = 100%: '100'
- missing value (NA) (class without valid data): '-'
- other: 'x.x' (1 decimal)

## Usage

```
formatPercentage(x, nDec = 1)
```

## Arguments

| | |
|---|---|
| x | Numeric vector with percentage(s) |
| nDec | Integer of length 1, number of decimals used to round the percentage, 1 by default. |

## Value

String with formatted percentage

## Author(s)

Laure Cougnaud

## See Also

Other decimals: `getMaxNDecimals()`, `getMaxNDecimalsData()`, `getNDecimals()`, `getNDecimalsData()`

## Examples

```
xPerc <- c(NA, 0, 100, 99.95, 0.012, 34.768)
formatPercentage(x = xPerc)
```

---

`formatSummaryStatisticsTable`

*Format summary statistics table for export*

---

## Description

Format summary statistics table for export

## Usage

```
formatSummaryStatisticsTable(
  summaryTable,
  rowVar = getAttribute(summaryTable, "rowVar"),
  colVar = getAttribute(summaryTable, "colVar"),
  colTotalLab = getAttribute(summaryTable, "colTotalLab", default = "Total"),
  colHeaderTotalInclude = TRUE,
  statsVar = getAttribute(summaryTable, "statsVar"),
  statsLabInclude = NULL,
  statsLayout = "row",
  statsValueLab = "StatisticValue",
  emptyValue = "-"
)
```

## Arguments

| | |
|---|---|
| `summaryTable` | A `summaryTable` object. |
| `rowVar` | Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table. |
| `colVar` | Character vector with variable(s) to be included in columns. If multiple variables are specified, the variables should be sorted in hierarchical order, and are included in multi-columns layout.<br>Use: 'variable' to include the variables to summarize: `var` (if multiple) in different columns. |
| `colTotalLab` | String, label for the total column 'Total' by default. |
| `colHeaderTotalInclude` | Logical, if TRUE include the total of number of patients (`'statN'`) in the column header. |

`statsVar` Character vector with columns of `summaryTable` with statistic variables. For the export: if not specified, all columns of `data` besides row, column variables, 'variable', 'variableGroup' and 'isTotal' are considered.

`statsLabInclude`
Logical, if TRUE include the statistic label in the table.
By default only included if more than one statistic variables are available in the table.

`statsLayout` String with layout for the statistics names (in case more than one statistic is included), among:

- row (by default for 'flextable' output):
  All statistics are included in different rows in the first column of the table (after the row variable(s))
- 'col' (by default for 'DT' output):
  Statistics are included in separated columns (last row of the header).
  This option is not compatible with categorical variable(s).
- 'rowInSepCol':
  Statistics are included in different rows, but in a separated column than the `rowVar` variable(s)

`statsValueLab` String with label for the statistic value, 'StatisticValue' by default.
This is only included in the table if the statistics provided in `stats` are not named and if no `colVar` is specified.

`emptyValue` String with placeholder used to fill the table for missing values, '-' by default.
This value is typically used e.g. if not all statistics are computed for all specified row/col/var variables.

## Value

summaryTable reformatted to wide format

## Author(s)

Laure Cougnaud

---

# formatSummaryStatisticsTableFlextable

*Merge nested rows of a summary table for a format compatible with `flextable`*

---

## Description

Merge nested rows of a summary table for a format compatible with `flextable`

## Usage

```
formatSummaryStatisticsTableFlextable(
  summaryTable,
  rowVar = getAttribute(summaryTable, "rowVar"),
  rowVarInSepCol = NULL,
  rowVarTotalInclude = getAttribute(summaryTable, "rowVarTotalInclude"),
  statsLayout = "row",
  statsVar = getAttribute(summaryTable, "statsVar"),
  statsLabInclude = getAttribute(summaryTable, "statsLabInclude", default =
    length(statsVar) > 1),
 rowVarLab = getAttribute(summaryTable, "rowVarLab", default = getLabelVar(rowVar,
    labelVars = labelVars)),
  rowVarTotalInSepRow = NULL,
  vline = c("none", "auto"),
  hline = c("none", "auto"),
  rowAutoMerge = TRUE,
  rowVarFormat = NULL,
  rowTotalLab = NULL,
  labelVars = NULL
)
```

## Arguments

| | |
|---|---|
| summaryTable | A summaryTable object. |
| rowVar | Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table. |
| rowVarInSepCol | Character vector with rowVar that should be included in separated columns. By default (NULL), all row variables are nested in the first column of the table.<br>To include the groups within a var variable in a separated column, set: rowVarInSepCol == 'variableGroup'. |
| rowVarTotalInclude | Character vector with rowVar for which the total should be reported.<br>If the higher row variable is specified, the total across all rows is reported.<br>For the export, these variable(s) are formatted as factor with **'Total' as the first level**. |
| statsLayout | String with layout for the statistics names (in case more than one statistic is included), among:<br>• row (by default for 'flextable' output):<br>All statistics are included in different rows in the first column of the table (after the row variable(s))<br>• 'col' (by default for 'DT' output):<br>Statistics are included in separated columns (last row of the header).<br>This option is not compatible with categorical variable(s).<br>• 'rowInSepCol':<br>Statistics are included in different rows, but in a separated column than the rowVar variable(s) |

`statsVar` Character vector with columns of `summaryTable` with statistic variables. For the export: if not specified, all columns of `data` besides row, column variables, 'variable', 'variableGroup' and 'isTotal' are considered.

`statsLabInclude`
Logical, if TRUE include the statistic label in the table.
By default only included if more than one statistic variables are available in the table.

`rowVarLab` Named character vector with label for the `rowVar` variable(s).

`rowVarTotalInSepRow`
Character vector with `rowVarTotalInclude` (not in `rowVarInSepCol`) for which the total should be included in a separated row labelled 'Total'. Otherwise (by default) the total is included in the header row of each category.

`vline` (flextable output) String mentioning how vertical lines should be included in the body of the table, either:

- 'none' (default): no vertical lines included
- 'auto': vertical lines included between sub-groups

`hline` (flextable output) String mentioning how horizontal lines should be included in the body of the table, either:

- 'none': no horizontal lines included
- 'auto' (default): horizontal lines included between sub-groups

`rowAutoMerge` (flextable output) Logical, if TRUE (by default) automatically merge rows, e.g. in case there is only one sub-category (e.g. categorical variable with only one group) or only one statistic per category.

`rowVarFormat` (flextable output) Named list with special formatting for the `rowVar`. Currently, only possibility is to set the variable elements in bold, with: list(var1 = "bold"). (Use 'variable' for `var` or 'variableGroup' for group within categorical variables.)

`rowTotalLab` (flextable output) string with label for the row with total.

`labelVars` (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

## Value

summaryTable reformatted in long format, with extra attributes:

- 'header': data.frame with header for each column
- 'padParams': list of list of parameters to be passed to the `padding` function
- 'rowVar': column of output with row variable
- 'rowVarInSepCol': column(s) of output with row variable in separated column(s)
- 'vlineParams' and 'hlineParams': list of list with correspondingly parameters for vertical and horizontal lines
- 'vline': `vline` parameter

- 'formatParams': list of list with special formatting for the table, currently only used if `rowVarFormat` if specified.

If `summaryTable` is a list of summary tables, returns a list of corresponding summary tables in long format.

## Author(s)

Laure Cougnaud

---

`geomCV` *Compute geometric coefficient of variation (in a scale from 0 to 100).*

---

## Description

The geometric coefficient of variation is computed as: $\sqrt{\exp(\sigma(log(x))^2) - 1} * 100$, with:

- log: natural logarithm
- $\sigma$: standard deviation

## Usage

```
geomCV(x, na.rm = FALSE)
```

## Arguments

| | |
|---|---|
| `x` | Numeric vector. |
| `na.rm` | Logical, should NA value(s) be removed (FALSE by default)? |

## Value

Numeric vector of length 1 with geometric coefficient of variation.

## Author(s)

Laure Cougnaud

## See Also

Other stats utility functions: `cv()`, `geomMean()`, `geomSD()`, `geomSE()`, `se()`

## Examples

```
# Geometric coefficient of variation of a sample from a log normal distribution:
geomCV(rlnorm(n = 1000, meanlog = 0, sdlog = 1))
```

geomMean *Compute geometric mean.*

## Description

The geometric mean is computed as: $\exp(\bar{log(x)})$, with:

- log: natural logarithm
- $\bar{log(x)}$: arithmetic mean of log(x)

## Usage

```
geomMean(x, na.rm = FALSE)
```

## Arguments

| | |
|---|---|
| x | Numeric vector. |
| na.rm | Logical, should NA value(s) be removed (FALSE by default)? |

## Value

Numeric vector of length 1 with geometric mean.

## Author(s)

Laure Cougnaud

## See Also

Other stats utility functions: cv(), geomCV(), geomSD(), geomSE(), se()

## Examples

```
# geometric mean of a big sample from log normal distribution
# tends to the mean of the distribution:
geomMean(rlnorm(n = 1000, meanlog = 0, sdlog = 1))
```

geomSD *Compute geometric standard deviation*

## Description

The geometric standard deviation is computed as: $\exp(\sigma(log(x)))$, with:

- log: natural logarithm
- $\sigma$: standard deviation

## Usage

```
geomSD(x, na.rm = FALSE)
```

## Arguments

| | |
|---|---|
| x | Numeric vector. |
| na.rm | Logical, should NA value(s) be removed (FALSE by default)? |

## Value

Numeric vector of length 1 with geometric mean.

## Author(s)

Laure Cougnaud

## See Also

Other stats utility functions: cv(), geomCV(), geomMean(), geomSE(), se()

## Examples

```
# geometric standard deviation of a sample from a log normal distribution:
geomSD(rlnorm(n = 1000, meanlog = 0, sdlog = 1))
```

geomSE *Compute geometric standard error of the mean.*

## Description

The geometric standard error of the mean is computed as: $\exp(se(log(x))$, with:

- log: natural logarithm
- $se$: standard error of the mean, as computed with `se`

## Usage

```
geomSE(x, na.rm = FALSE)
```

## Arguments

| | |
|---|---|
| x | Numeric vector. |
| na.rm | Logical, should NA value(s) be removed (FALSE by default)? |

## Value

Numeric vector of length 1 with geometric standard error of the mean.

## Author(s)

Laure Cougnaud

## See Also

Other stats utility functions: `cv()`, `geomCV()`, `geomMean()`, `geomSD()`, `se()`

## Examples

```
# Geometric standard error of the mean of a sample from a log normal distribution:
geomSE(rlnorm(n = 1000, meanlog = 0, sdlog = 1))
```

## getColorPaletteTable *Get color palette for the tables*

### Description

This function gets the color palettes for the tables specified as global options.

### Usage

```
getColorPaletteTable(style = c("report", "presentation"))
```

### Arguments

| | |
|---|---|
| style | String with style of report. Either 'report' or 'presentation'. By default, the style is 'report'. |

### Details

By default, the function returns the palette of the package. The user can specify a custom palette by setting the global options.

### Value

A named vector with hex colors.

### Examples

```
# report style (the default)
getColorPaletteTable()
# presentation style
getColorPaletteTable(style = "presentation")
# custom palette
customColorTable <- c('header' = "#FFFFFF",'headerBackground' = "#3F4788FF",
'body' = "#000000", 'bodyBackground1' = "#D9D9D9", 'bodyBackground2' = "#D9D9D9",
'footer' = "#000000", 'footerBackground' = "#FFFFFF",'line' = "#FFFFFF")
options(inTextSummaryTable.colors.table.presentation = customColorTable)
getColorPaletteTable("presentation")
```

## getDimPage *Get dimension of the page available for content for standard Word report or PowerPoint presentation.*

### Description

Report is in A4 and presentation dimensions extracted from PowerPoint. The returned dimensions are the page dimensions without the margins.

## Usage

```
getDimPage(
  type = c("width", "height"),
  landscape = (style == "presentation"),
  margin = 1,
  pageDim = NULL,
  style = "report"
)
```

## Arguments

| | |
|---|---|
| type | Character vector with dimension of interest, among: 'width', 'height', multiple are possible. By default: c("width", "height") |
| landscape | Logical, if TRUE the table is presented in landscape format.<br>By default: TRUE for style: 'report', FALSE for style: 'presentation'. |
| margin | Margin in the document in inches, 1 by default. |
| pageDim | (optional) Numeric vector of length 2 with page width and height in inches in portrait format, in case page dimensions differ from the default implemented report/presentation. These dimensions should include the margins. |
| style | String with table style, either 'report' (by default, a4 format) or 'presentation' |

## Value

numeric vector with dimension of interest, in the same order as specified via the type parameter.

## Author(s)

Laure Cougnaud

## Examples

```
## get part of the page available for content
# report A4 portrait format:
getDimPage(type = "width")
getDimPage(type = "height")
# report A4 landscape format:
getDimPage(type = "width", landscape = TRUE)
getDimPage(type = "height", landscape = TRUE)
# Note that the layout is by default set to 'landscape'
getDimPage(type = "width", style = "presentation")
getDimPage(type = "height", style = "presentation")
# custom dimensions: A3 format
getDimPage(type = "width", pageDim = c(11.7, 16.5))
# increase margin
getDimPage(type = "width", margin = 1.5)
# get both dimensions at once
getDimPage(type = c("width", "height"))
# get dimensions of the full page (including margins)
getDimPage(type = c("width", "height"), style = "report", margin = 0)
getDimPage(type = c("width", "height"), style = "presentation", margin = 0)
```

getListing *Format or create flextable for listings.*

## Description

Flextable version >= 0.4.7 and pandoc >= 2.4 is required to included such table in a Rmarkdown document.

## Usage

```
getListing(
  data,
  ft,
  border = TRUE,
  highlight = integer(),
  bgVar = NULL,
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  fontsize = switch(style, report = 8, presentation = 10),
  landscape = (style == "presentation"),
  style = "report",
  margin = 1,
  adjustWidth = TRUE,
  colorTable = getColorPaletteTable(style = style),
  align = TRUE,
  title = NULL,
  pageDim = NULL,
  includeRownames = TRUE
)
```

## Arguments

| | |
|---|---|
| data | data.frame with data used in table. |
| ft | Corresponding flextable. |
| border | Logical, if TRUE add a border. |
| highlight | Integer vector with index(ices) of column(s) to highlight (only applies for style: 'presentation'). 0 for rownames (if present). Colors for:<br>• highlighted columns is specified in colorTable["headerBackgroundHighlight"]<br>• non highlighted columns is specified in colorTable["headerBackground"] |
| bgVar | String with the column of the data used for alternating the body background colors of the table. |
| fontname | String with font name, 'Times' by default. |
| fontsize | Integer with font size, 8 by default. |
| landscape | Logical, if TRUE the table is presented in landscape format.<br>By default: TRUE for style: 'report', FALSE for style: 'presentation'. |

| | |
|---|---|
| style | String with table style, either 'report' (by default, a4 format) or 'presentation' |
| margin | Margin in the document in inches, 1 by default. |
| adjustWidth | Logical, if TRUE adjust column widths, to comply to specification of landscape, margin and pageDim (only set to FALSE if e.g. table dimensions are pre-set with the specified ft). |
| colorTable | Named character vector with color for the table, see output of getColorPaletteTable for required elements. |
| align | Logical, if TRUE (by default), default alignment is set ('center' in all table). |
| title | Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. Only available if ft is not specified. |
| pageDim | (optional) Numeric vector of length 2 with page width and height in inches in portrait format, in case page dimensions differ from the default implemented report/presentation. These dimensions should include the margins. |
| includeRownames | Logical, if TRUE (by default) rownames are included in the flextable object. |

## Value

flextable with style.

## Examples

```
# style: report or presentation
getListing(data = head(mtcars), style = "report")
getListing(data = head(mtcars), style = "presentation")
# remove rownames (included by default)
getListing(data = head(mtcars), style = "presentation", includeRownames = FALSE)
# highlight:
#  all columns
getListing(data = head(mtcars), style = "presentation", highlight = seq_along(mtcars))
# rownames
getListing(data = head(mtcars), style = "presentation", highlight = 0)
# specific columns
getListing(data = head(mtcars), style = "presentation", highlight = c(2, 4))
```

---

# getMaxNDecimals *Get maximum number of decimals in a variable, based on pre-defined rule and/or data.*

---

## Description

The function getNDecimals) extracts the number of decimals in a specific variable.

## Usage

```
getMaxNDecimals(x, ...)
```

### Arguments

| | |
|---|---|
| x | Numeric vector. |
| ... | Any parameters for the getNDecimals function. |

### Value

Integer with maximum number of decimals in a character vector.

### Author(s)

Laure Cougnaud

### See Also

Other decimals: formatPercentage(), getMaxNDecimalsData(), getNDecimals(), getNDecimalsData()

### Examples

```
x <- c(0.99, 5.679, 50.45, 1450)
# extract max number of decimals based on data:
getMaxNDecimals(x, useRule = FALSE, useData = TRUE)
# extract max number of decimals based on pre-defined rule:
getMaxNDecimals(x, useRule = TRUE, useData = FALSE)
# extract max number of decimals based on both rules
# minimum of both is used (by default)
getMaxNDecimals(x, useRule = TRUE, useData = TRUE)
```

---

## getMaxNDecimalsData *Get maximum number of decimals in a variable based on the data (getNDecimalsData)*

---

### Description

Get maximum number of decimals in a variable based on the data (getNDecimalsData)

### Usage

```
getMaxNDecimalsData(x)
```

### Arguments

| | |
|---|---|
| x | Numeric vector. |

### Value

Integer with maximum number of decimals in a character vector.

### Author(s)

Laure Cougnaud x <- c(0.99, 5.679, 50.45, 1450) # extract max number of decimals based on data: getMaxNDecimalsData(x)

### See Also

Other decimals: `formatPercentage()`, `getMaxNDecimals()`, `getNDecimals()`, `getNDecimalsData()`

---

## getNDecimals *Get number of decimals for a specific vector.*

---

### Description

The number of decimals is extracted either:

- from specific implemented rule : see `getNDecimalsRule` for further details
- from the data itself: see `getNDecimalsData` for further details
- both criterias: in this case the minimum of the number of decimals for both criterias is used

### Usage

```
getNDecimals(x, useRule = TRUE, rule = "1", useData = TRUE)
```

### Arguments

| | |
|---|---|
| `x` | Numeric vector. |
| `useRule` | Logical (TRUE by default), should the `rule` be applied? |
| `rule` | Character vector with rule to use to derive the number of parameters. Currently only: '1' is implemented.<br>• '1': standard rule for the number of decimals for individual values for a continuous variable:<br>– value < 1 ('very small values'): 3<br>– value < 10: 2<br>– value in [10, 1000[: 1<br>– value >= 1000: 0 |
| `useData` | Logical (TRUE by default), should the number of decimals be extracted based on the input data `x`? |

### Value

Numeric vector of same length than `x` with the number of decimals.

### Author(s)

Laure Cougnaud

## See Also

Other decimals: formatPercentage(), getMaxNDecimals(), getMaxNDecimalsData(), getNDecimalsData()

## Examples

```
x <- c(0.99, 5.679, 50.45, 1450)
# extract number of decimals based on data:
getNDecimals(x, useRule = FALSE, useData = TRUE)
# extract number of decimals based on pre-defined rule:
getNDecimals(x, useRule = TRUE, useData = FALSE)
# extract number of decimals based on both rules
# minimum of both is used (by default)
getNDecimals(x, useRule = TRUE, useData = TRUE)
```

---

# getNDecimalsData *Get number of decimals based on the data in a numeric vector. Note: NA is returned if the element is missing (NA).*

---

## Description

Get number of decimals based on the data in a numeric vector. Note: NA is returned if the element is missing (NA).

## Usage

```
getNDecimalsData(x)
```

## Arguments

| | |
|---|---|
| x | Numeric vector. |

## Value

Numeric vector of same length than x with the number of decimals.

## Author(s)

Laure Cougnaud

## See Also

Other decimals: formatPercentage(), getMaxNDecimals(), getMaxNDecimalsData(), getNDecimals()

## Examples

```
x <- c(0.99, 5.679, 50.45, 1450)
getNDecimalsData(x)
```

`getNDecimalsRule` *Get number of decimals based pre-defined rule(s).*

## Description

Note: NA is returned if the element is missing (NA).

## Usage

```
getNDecimalsRule(x, rule = c("1"))
```

## Arguments

| | |
|---|---|
| `x` | Numeric vector. |
| `rule` | Character vector with rule to use to derive the number of parameters. Currently only: '1' is implemented.<br>• '1': standard rule for the number of decimals for individual values for a continuous variable:<br>– value < 1 ('very small values'): 3<br>– value < 10: 2<br>– value in [10, 1000[: 1<br>– value >= 1000: 0 |

## Value

Numeric vector of same length than `x` with the number of decimals.

## Author(s)

Laure Cougnaud

## Examples

```
x <- c(0.99, 5.679, 50.45, 1450)
getNDecimalsRule(x = x)
```

## getPatternPosition *Get position(s) (start, end) of a pattern in a string.*

### Description

Get position(s) (start, end) of a pattern in a string.

### Usage

```
getPatternPosition(x, startPattern, endPattern = "\\}", format)
```

### Arguments

| | |
|---|---|
| x | String. |
| startPattern | String with start pattern. |
| endPattern | String with end pattern. |
| format | String with type of formatting |

### Value

Matrix with columns: 'start' and 'end' with start and end position(x) of the pattern, 'format' with the format and 'pattern' with the regex pattern for the full match. NULL if no match.

### Author(s)

Laure Cougnaud

## getStats *Get default set of statistics for one particular variable.*

### Description

This set of statistics can be passed directly to the stats parameter of the of the package functions.

### Usage

```
getStats(
  type = "summary",
  includeName = TRUE,
  x = NULL,
  nDecCont = getMaxNDecimals,
  nDecN = 0,
  nDecm = nDecN,
  formatPercentage = inTextSummaryTable::formatPercentage
)
```

## Arguments

| | |
|---|---|
| type | Character vector with type of statistics (multiple are possible). Available statistics are specified in the section 'Formatted statistics' and formatting in 'Statistics formatting' in in-text table statistics. |
| includeName | Logical, should the statistics name be included (TRUE by default)? This is applied for the statistic names used in each for the set defined in type; and for the label of the list if type is of length 2. If there are multiple type or statistics within a set, the names are retained (to avoid confusion). |
| x | (optional, recommended for continuous variable) Numeric vector for which the statistics should be computed on.<br>This is used to derive the number of decimals to include for a continuous variable.<br>If not specified, the values are rounded with formatC. |
| nDecCont | Integer with base number of decimals for continuous variable, or function returning this number based on x (getNDecimals by default). |
| nDecN, nDecm | Integer with number of decimals for number of subjects/records (0 by default). |
| formatPercentage | Function used to format the percentages (see formatPercentage for default behaviour). |

## Value

Expression (or call object) containing function to extract requested summary statistics. If multiple type are specified, they are combined to a list. Names of the list will be typically used to name the statistic in the summary table.

## Author(s)

Laure Cougnaud

## See Also

getStatsData

## Examples

```
## default set of statistics are available for:

# for count table:
getStats("count")
getStats("n (%)")
getStats("n")
getStats("%")
getStats("m")
getStats("%m")
getStats("m (%)")
# for continuous variable:
getStats("summary")
```

```
getStats("mean (se)")
getStats("mean (sd)")
getStats("median (range)")
getStats("median\n(range)")
getStats(c("Mean", "SE"))

## to not include statistic name in the table
getStats("median\n(range)", includeName = FALSE)
getStats(c("summary", "median\n(range)"), includeName = FALSE)

## to extract the number of decimals based on a continuous variable (see ?getMaxNDecimals)
exampleData <- data.frame(
  USUBJID = 1 : 4,
  WEIGHT = c(67, 78, 83, 61),
  SEX = c("F", "M", "M", "F"),
  stringsAsFactors = FALSE
)
getStats(type = c('median (range)', 'mean (se)'), x = exampleData$WEIGHT)
# compare with when 'x' is not specified:
getStats(type = c('median (range)', 'mean (se)'))

## custom function to format the percentages:
getStats(type = "count", formatPercentage = function(x) round(x, 2))
```

---

`getStatsData` *Get default set of statistics for variables of interest and specific dataset.*

---

## Description

This set of statistics can be passed directly to the `stats` parameter of the package functions.
By default, statistics are extracted based on the variable(s) type and formatted with the default rules implemented in the package.

## Usage

```
getStatsData(
  data,
  var = NULL,
  type = "default",
  extra = NULL,
  args = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | Data.frame with dataset to consider for the summary table. |

| | |
|---|---|
| var | (optional, recommended for continuous variable) Character vector with variable(s) of data, to compute statistics on.<br>If NULL (by default), counts of the entire dataset are computed.<br>It is passed to the x parameter of getStats. |
| type | Character vector with type of statistics to extract, among:<br>• 'default': default sets of statistics, see types: 'summary-default' and 'count-default' in getStats<br>• 'all': all computed statistics, see types: 'summary' and 'count' in getStats<br>• any formatted statistics as implemented in getStats, see section 'Formatted statistics' in in-text table statistics.<br>To specify statistics for a continuous (numeric) or categorical variable separately, this vector can be named with: 'cont' or 'cat' respectively (elements not named are used for both continuous and categorical variables). |
| extra | List with extra statistics to include, or function to apply on each var (e.g. depending on the class of var) to get such statistic. |
| args | (optional) Named list with extra arguments for getStats for continuous (name: 'cont') or categorical variable (name: 'cat') specifically. |
| ... | Extra parameters passed to the getStats function (independent of the variable type). |

## Value

List with statistics to compute, named by var

## Author(s)

Laure Cougnaud

## See Also

getStats

## Examples

```
# default set of statistics (depending if the variable is continuous or categorical)
exampleData <- data.frame(
  USUBJID = 1 : 4,
  WEIGHT = c(67, 78, 83, 61),
  SEX = c("F", "M", "M", "F"),
  stringsAsFactors = FALSE
)
getStatsData(data = exampleData, var = c("WEIGHT", "SEX"))
# all set of statistics (depending if the variable is continuous or categorical)
getStatsData(data = exampleData, var = c("WEIGHT", "SEX"), type = "all")
# custom set of statistics for all variables
getStatsData(data = exampleData, var = c("WEIGHT", "SEX"), type = c("n", "%"))
# custom set of statistics, depending on the type of the variable
getStatsData(data = exampleData, var = c("WEIGHT", "SEX"),
```

```
  type = c(cont = "median (range)", cont = "mean (se)", cat = "n (%)"),
  args = list(cat = list(includeName = FALSE))
)
```

---

`getSummaryStatisticsTable` *Get summary statistics table*

---

## Description

Get summary statistics table

## Usage

```
getSummaryStatisticsTable(
  data,
  var = NULL,
  varFlag = NULL,
  varLab = NULL,
  varLabInclude = length(var) > 1,
  varInclude0 = FALSE,
  varIgnore = NULL,
  varGeneralLab = "Variable",
  varSubgroupLab = "Variable group",
  varIncludeTotal = FALSE,
  varTotalInclude = FALSE,
  varTotalInSepRow = FALSE,
  rowVar = NULL,
  rowVarLab = NULL,
  rowVarDataLevels = NULL,
  rowOrder = "auto",
  rowOrderTotalFilterFct = NULL,
  rowOrderCatLast = NULL,
  rowVarInSepCol = NULL,
  rowVarFormat = NULL,
  rowVarTotalInclude = NULL,
  rowVarTotalByVar = NULL,
  rowVarTotalInSepRow = NULL,
  rowTotalLab = NULL,
  rowInclude0 = FALSE,
  rowAutoMerge = TRUE,
  emptyValue = "-",
  rowVarTotalPerc = NULL,
  colVar = NULL,
  colVarTotal = colVar,
  colVarTotalPerc = colVarTotal,
  colInclude0 = FALSE,
```

```
  colVarDataLevels = NULL,
  colTotalInclude = FALSE,
  colTotalLab = "Total",
  stats = NULL,
  statsExtra = NULL,
  statsVarBy = NULL,
  statsPerc = c("statN", "statm"),
  statsGeneralLab = "Statistic",
  statsValueLab = "StatisticValue",
  statsLabInclude = NULL,
  subjectVar = "USUBJID",
  filterFct = NULL,
  dataTotal = NULL,
  dataTotalPerc = dataTotal,
  dataTotalRow = NULL,
  dataTotalCol = NULL,
  type = "auto",
  byVar = NULL,
  byVarLab = NULL,
  checkVarDiffBySubj = "error",
  labelVars = NULL,
  outputType = "flextable",
  statsLayout = ifelse("DT" %in% outputType, "col", "row"),
  landscape = (style == "presentation"),
  margin = 1,
  rowPadBase = 14.4,
  title = NULL,
  footer = NULL,
  file = NULL,
  style = "report",
  colorTable = getColorPaletteTable(style = style),
  colHeaderTotalInclude = TRUE,
  colHeaderMerge = TRUE,
  fontsize = switch(style, report = 8, presentation = 10),
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  vline = "none",
  hline = "auto",
  pageDim = NULL,
  columnsWidth = NULL,
  expandVar = NULL,
  noEscapeVar = NULL,
  barVar = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| data | Data.frame with dataset to consider for the summary table. |

`var` Character vector with variable(s) of `data`, to compute statistics on.
If NULL (by default), counts by row/column variable(s) are computed.
To also return counts of the `rowVar` in case other `var` are specified, you can include: 'all' in the `var`.
Missing values, if present, are filtered (also for the report of number of subjects/records).

`varFlag` Character vector, subset of `var` with variable(s) of type 'flag' (with 'Y', 'N' or '' for empty/non specified value). Only the counts for records flagged (with 'Y') are retained.

`varLab` Named character vector with label for each variable specified in `var`. By default, extracted from the `labelVars`. if not available, `var` is used.

`varLabInclude` Logical, if TRUE the name of the summary statistic variable(s) (`var`) are included in the table. This is automatically set to TRUE if more than one variable(s) and is specified, and FALSE if only one variable is specified.

`varInclude0` Logical, should rows with no counts for the count `var` or `varFlag` variable(s) be included in the table? Either:

- logical of length 1, if TRUE (FALSE by default) rows with no count are included for all `var`
- a character vector containing categorical `var` for which zero counts rows should be included

`varIgnore` Vector with elements to ignore in the `var` variable(s). The `data` records with such elements in `var` are **filtered** from the data at the start of the workflow.

`varGeneralLab` String with general label for variable specified in `var`. In case of multiple variable in `var`, this will be included in the table header (see 'rowVarLab' attribute of the output).

`varSubgroupLab` String with general label for sub-group of categorical variable(s) for count table, 'Variable group' by default. This will be included in the final table header (see 'rowVarLab' attribute of the output).

`varIncludeTotal`
This argument is deprecated, please use: 'varTotalInclude' instead.

`varTotalInclude`
Should the total across all categories of `var` be included for the count table? Only used for categorical variables (and `var` not 'all'). Either:

- logical of length 1, if TRUE (FALSE by default) include the total for all categorical `var`
- a character vector containing categorical `var` for which the total should be included

`varTotalInSepRow`
Logical, should the total per variable be included in a separated row (by default) or in the row containing the header of the variable?

`rowVar` Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table.

`rowVarLab` Named character vector with label for the `rowVar` variable(s).

`rowVarDataLevels`
: Data.frame with unique combinations of `rowVar` to be included in columns. Each column should correspond to `colVar` and as factor if the elements should be ordered in the final table.

`rowOrder`
: Specify how the rows should be ordered in the final table, either a:

  - String among:
    - 'auto' (by default): if the variable is a factor, keep its order, otherwise order alphabetically
    - 'alphabetical': order alphabetically
    - 'total': order rows in decreasing order of the total number of subjects across all columns for this specific category.
  - Function with input the summary table and output the ordered elements of the `rowVar`

  To specify different ordering methods for different `rowVar`, specify a list of such elements, named with the `rowVar` variable. For the table output of `computeSummaryStatisticsTable` (long format), this order is also reflected in the `levels` of the row factor variable.

`rowOrderTotalFilterFct`
: Function used to filter the data used to order the rows based on total counts (in case `rowOrder` is 'total'), To order rows based on one specific column category, e.g. to order based on the counts in the treatment column: function(x) subset(x, TRTP == "treatmentX")

`rowOrderCatLast`
: String with category to be printed in the last row of each `rowVar` (if any, set to NULL if none).

`rowVarInSepCol`
: Character vector with `rowVar` that should be included in separated columns. By default (NULL), all row variables are nested in the first column of the table.
  To include the groups within a `var` variable in a separated column, set: `rowVarInSepCol == 'variableGroup'`.

`rowVarFormat`
: (flextable output) Named list with special formatting for the `rowVar`. Currently, only possibility is to set the variable elements in bold, with: list(var1 = "bold"). (Use 'variable' for `var` or 'variableGroup' for group within categorical variables.)

`rowVarTotalInclude`
: Character vector with `rowVar` for which the total should be reported.
  If the higher row variable is specified, the total across all rows is reported.
  For the export, these variable(s) are formatted as factor with **'Total' as the first level**.

`rowVarTotalByVar`
: Character vector with a row variable used to categorize the row total.
  Note that this is only used if row total(s) is/are requested via `rowVarTotalInclude`, and this variable should also be included in `rowVar`. This can be specified also for a specific row variable if the vector is named.
  For example: `c(ADECOD = "AESEV")` to compute total by severity for row adverse event term in a typical adverse event count table (by System Organ Class and Adverse Event Term).

rowVarTotalInSepRow
: Character vector with rowVarTotalInclude (not in rowVarInSepCol) for which the total should be included in a separated row labelled 'Total'. Otherwise (by default) the total is included in the header row of each category.

rowTotalLab
: (flextable output) string with label for the row with total.

rowInclude0
: Logical, if TRUE (FALSE by default), include rows with no records, based on all combinations of the rowVar (assuming nested variable(s)).

rowAutoMerge
: (flextable output) Logical, if TRUE (by default) automatically merge rows, e.g. in case there is only one sub-category (e.g. categorical variable with only one group) or only one statistic per category.

emptyValue
: String with placeholder used to fill the table for missing values, '-' by default. This value is typically used e.g. if not all statistics are computed for all specified row/col/var variables.

rowVarTotalPerc
: Character vector with row variables by which the total should be computed for the denominator for the percentage computation. By default the total is only computed only by column (NULL by default). If the total should be based on the total number of records per variable, rowVarTotalPerc should be set to 'variable'.

colVar
: Character vector with variable(s) to be included in columns. If multiple variables are specified, the variables should be sorted in hierarchical order, and are included in multi-columns layout.
Use: 'variable' to include the variables to summarize: var (if multiple) in different columns.

colVarTotal
: String with column(s) considered to compute the total by, reported in the header of the table, by default same as colVar. Use: 'variable' to compute total by var (if multiple).

colVarTotalPerc
: String with column(s) considered to compute the total by, used as denominator for the percentage computation, by default same as colVarTotal. Use: 'variable' to compute total by var (if multiple).

colInclude0
: Logical, if TRUE (FALSE by default), include columns with no records, based on all combinations of the columnVar (assuming nested variable(s)). If variable(s) are not nested, possible combinations can be specified via colVarDataLevels.

colVarDataLevels
: Data.frame with unique combinations of colVar to be included in columns. Each column should correspond to colVar and as factor if the elements should be ordered in the final table.

colTotalInclude
: Logical, if TRUE (FALSE by default) include the summary statistics across columns in a separated column.

colTotalLab
: String, label for the total column 'Total' by default.

stats
: (optional) Statistic(s) of interest to compute, either:

- string with the name of a default set of statistics available in the package, see section 'Formatted statistics' in in-text table statistics.
  See the corresponding type parameter of the getStatsData for more information on how the statistic is internally extracted.
- (expert mode) named list of language object (see is.language) of base summary statistics of interest, see section: 'Base statistics' in in-text table statistics.
  The names are reported in the header.
  If stats if of length 1, the name of the summary statistic is not included in the table.
  The statistics can be specified separately:
  - for each var (if multiple), by naming each element of the list: list(varName1 = list(...), varName2 = list())
  - and/or for each element in: statsVarBy, by naming each sublist.

statsExtra (optional) Named list with functions for additional custom statistics to be computed.
Each function:

- has as parameter, either: 'x': the variable (var) to compute the summary statistic on or 'data': the entire dataset
- returns the corresponding summary statistic as a numeric vector

For example, to additionally compute the coefficient of variation, this can be set to: list(statCVPerc = function(x) sd(x)/mean(x)*100) (or cv).

statsVarBy String with variable in rowVar/colVar which the statistics should be computed by.
In this case, stats (nested list or not) should be additionally nested to specify the statistics for each element in statsVarBy.

statsPerc String with 'base statistical variable' used to compute the percentage, either:

- 'statN' (by default): the number of subjects
- 'statm': the number of records

statsGeneralLab
String with general label for statistics, 'Statistic' by default. Only included if no statsVar if longer than 1.

statsValueLab String with label for the statistic value, 'StatisticValue' by default.
This is only included in the table if the statistics provided in stats are not named and if no colVar is specified.

statsLabInclude
Logical, if TRUE include the statistic label in the table.
By default only included if more than one statistic variables are available in the table.

subjectVar String, variable of data with subject ID, 'USUBJID' by default.

filterFct (optional) Function taking as input the summary table with computed statistics and returning a subset of the summary table.
Note: The filtering function should also handle records with :

- total for the column header: `isTotal` set to TRUE, and `colVar`/`rowVar` is NA.
  For example: `filterFct = function(data) subset(data, isTotal & myColVar == "group 1")`
- `rowVar`/`colVar` set to 'Total'/`colTotalLab` if `rowVarTotalInclude`/`colTotalInclude` is specified

`dataTotal` Data.frame used to extract the Total number of subject per column in column header ('N = [X]'). It should contain the variables specified by `colVarTotal`. If not specified, the total number of subjects is extracted from the `data`.

`dataTotalPerc` Data.frame used to extract the total counts per column for the computation of the percentage.
By default, `dataTotal` is used.
It should contain the variables specified by `colVarTotalPerc`.

`dataTotalRow` Data.frame used to extract the total count across all elements of the row variable, list of such data.frame for each `rowVar` variable.
If the dataset is specified by row variable, the list should be named with: variable X if the total across elements of variable X should be included. By default, `data` is used.

`dataTotalCol` Data.frame from which the total across columns is extracted (in case `colTotalInclude` is TRUE) or list of such data.frame for each `rowVar` variable.
If the dataset is specified by row variable, the list should be named with: with:

- last row variable: for the dataset used in the total column for the most nested row variable
- higher row variable (X+1): for the dataset used for the total column and row total of X
- 'total': for the dataset used for the total column and general row total

If only a subset of the variables is specified in this list, `data` is used for the remaining variable(s) (or 'total') if needed.
This dataset (the one for 'total' if a list) is also used for:

- the header of the total column in case `dataTotal` is not specified
- the denominator of the percentages in the total column in case `dataTotalPerc` is not specified

By default, `data` is used.

`type` String with type of table:

- 'summaryTable': summary table with statistics for numeric variable
- 'countTable': count table
- 'auto' (by default): 'summaryTable' if the variable is numeric, 'countTable' otherwise

`byVar` Variable(s) of `data` for which separated table(s) should be created.

`byVarLab` String with label for `byVar`, used to set the names of the output list of table(s).

`checkVarDiffBySubj`
String, 'error' (default), 'warning', or 'none'. Should an error, a warning, or nothing be produced if a continuous variable (`var`) contains different values for the same subject (by row/column)?

`labelVars` (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

`outputType` String with output type:

- 'flextable' (by default): `flextable` object, with format for CSR, compatible with Word/PowerPoint export
- 'DT': `datatable` interactive table, compatible with html export
- 'data.frame': data.frame in wide format (with elements in `colVar` in different columns)
- 'data.frame-base': data.frame in long format (with elements in `colVar` in different rows), useful for QC

`statsLayout` String with layout for the statistics names (in case more than one statistic is included), among:

- row (by default for 'flextable' output):
  All statistics are included in different rows in the first column of the table (after the row variable(s))
- 'col' (by default for 'DT' output):
  Statistics are included in separated columns (last row of the header).
  This option is not compatible with categorical variable(s).
- 'rowInSepCol':
  Statistics are included in different rows, but in a separated column than the `rowVar` variable(s)

`landscape` (flextable output) Logical, if TRUE the file is in landscape format.
By default: FALSE if `style` is 'report' and TRUE if `style` is 'presentation'.

`margin` (flextable output) Margin in the document in inches (1 by default). This is used to specify the width of the table, from: `[pageDim[1]` - 2 * margin].

`rowPadBase` (flextable output) Base padding for row (in points), 14.4 by default (corresponds to 0.2 inches)

`title` Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. If multiple are specified, specified for each element of `byVar` (in order of the levels).

`footer` (flextable output) Character vector with footer(s) for the table. Set to NULL (by default) if no footer should be included.

`file` (Optional) Name of the file the table should be exported to, either:

- string (of length 1). In this case, depending on the file extension, the following is exported:
  - 'txt': summary table in long format ('data.frame-base' `outputType`)
  - 'docx': summary table in final format is exported ('flextable' `outputType`)
  - 'html': interactive summary table is exported ('DT' `outputType`)
- named character vector in case of multiple exports. The names should correspond to the options in `outputType`:
  - for 'data.frame-base' and 'data.frame': filename with 'txt' extension
  - for 'flextable': filename with 'docx' extension

- for 'DT': filename with 'html' extension

If NULL (by default), the summary table is not exported but only returned as output. If `byVar` is specified, each table is exported to a separated file with the suffix: 'file_[i].[ext]' with i the index of the file (and [ext] the file extension).

`style` (flextable output) String with table style, either 'report' or 'presentation'. This parameter affects the fontsize, font family, color of the text and background, and table dimensions of the table.

`colorTable` (flextable output) Named character vector with color for the table background/body/text/line, e.g. created with the `getColorPaletteTable` function.

`colHeaderTotalInclude`
Logical, if TRUE include the total of number of patients (`'statN'`) in the column header.

`colHeaderMerge` (flextable output) Logical, if TRUE (FALSE by default) the column header is merged.

`fontsize` (flextable output) Integer with font size, by default: 8 if `style` is 'report' and 10 if `style` is 'presentation'.

`fontname` (flextable output) String with font name, by default: 'Times' if `style` is 'report' and 'Tahoma' if `style` is 'presentation'.

`vline` (flextable output) String mentioning how vertical lines should be included in the body of the table, either:

- 'none' (default): no vertical lines included
- 'auto': vertical lines included between sub-groups

`hline` (flextable output) String mentioning how horizontal lines should be included in the body of the table, either:

- 'none': no horizontal lines included
- 'auto' (default): horizontal lines included between sub-groups

`pageDim` Numeric vector of length 2 with page width and height.
Depending on `outputType`:

- 'flextable': in inches
- 'DT': in number of rows in the table.
  Currently only the height is used (e.g. `c(NA, 4)`)

`columnsWidth` (expert mode) Column widths of the table. This is only used for flextable and DT tables.
For flextable, note that the widths should be set to fit into the document page (see `getDimPage`).

`expandVar` (DT output) Character vector with variables of the summary table which should be expanded in the data.

`noEscapeVar` (DT output) Character vector with variables of `summaryTable` which shouldn't be escaped in the table (e.g. containing URLs).

`barVar` (DT output) Character vector with variables of `summaryTable` that should be represented as a bar.

`...` (DT output) Extra parameters passed to the `getClinDT`

## Value

Depending on the outputType:

- 'data.frame-base': input summary table in a long format with all computed statistics
- 'data.frame': summary table in a wide format ( different columns for each colVar), with specified labels
- 'flextable' (by default): flextable object with summary table
- 'DT': datatable object with summary table

If multiple outputType are specified, a list of those objects, named by outputType.
If byVar is specified, each object consists of a list of tables, one for each element in byVar.

## Author(s)

Laure Cougnaud

---

inTextSummaryTable *inTextSummaryTable: creation of in-text summary table*

---

## Description

Tables of summary statistics or count tables are created. These tables can be exported as in-text table to a Clinical Study Report (Word format), a topline presentation (PowerPoint format), or as interactive table to an html document.

## Details

- To get started with the package, see: `vignette("inTextSummaryTable-introduction", package = "inTextSummaryTable")`
- To get example code for standard in-text tables created with the package, see: `vignette("inTextSummaryTable-standardTables", package = "inTextSummaryTable")`
- The main function: getSummaryStatisticsTable enables to create ready in-text table.

## Author(s)

**Maintainer**: Laure Cougnaud <laure.cougnaud@openanalytics.eu>

Authors:

- Michela Pasetto <michela.pasetto@openanalytics.eu>

Other contributors:

- Margaux Faes (tests) [reviewer]
- Open Analytics [copyright holder]

## See Also

Useful links:

- https://github.com/openanalytics/inTextSummaryTable
- Report bugs at https://github.com/openanalytics/inTextSummaryTable/issues

---

`inTextSummaryTable-common-args` *Arguments used across the functions of the inTextSummaryTable package.*

---

## Description

Arguments used across the functions of the inTextSummaryTable package.

## Arguments

| | |
|---|---|
| `data` | Data.frame with dataset to consider for the summary table. |
| `summaryTable` | A `summaryTable` object. |
| `var` | Character vector with variable(s) of `data`, to compute statistics on.<br>If NULL (by default), counts by row/column variable(s) are computed.<br>To also return counts of the `rowVar` in case other `var` are specified, you can include: 'all' in the `var`.<br>Missing values, if present, are filtered (also for the report of number of subjects/records). |
| `varFlag` | Character vector, subset of `var` with variable(s) of type 'flag' (with 'Y', 'N' or '' for empty/non specified value). Only the counts for records flagged (with 'Y') are retained. |
| `varLabInclude` | Logical, if TRUE the name of the summary statistic variable(s) (`var`) are included in the table. This is automatically set to TRUE if more than one variable(s) and is specified, and FALSE if only one variable is specified. |
| `rowVar` | Character vector with variable(s) to be included in the rows. If multiple variables are specified, the variables should be sorted in hierarchical order (e.g. body system class before adverse event term) and are nested in the table. |
| `rowVarInSepCol` | Character vector with `rowVar` that should be included in separated columns. By default (NULL), all row variables are nested in the first column of the table.<br>To include the groups within a `var` variable in a separated column, set: `rowVarInSepCol == 'variableGroup'`. |
| `rowVarLab` | Named character vector with label for the `rowVar` variable(s). |
| `statsVar` | Character vector with columns of `summaryTable` with statistic variables. For the export: if not specified, all columns of `data` besides row, column variables, 'variable', 'variableGroup' and 'isTotal' are considered. |

`rowVarTotalInclude`
: Character vector with `rowVar` for which the total should be reported.
If the higher row variable is specified, the total across all rows is reported.
For the export, these variable(s) are formatted as factor with **'Total' as the first level**.

`rowVarTotalInSepRow`
: Character vector with `rowVarTotalInclude` (not in `rowVarInSepCol`) for which the total should be included in a separated row labelled 'Total'. Otherwise (by default) the total is included in the header row of each category.

`colVar`
: Character vector with variable(s) to be included in columns. If multiple variables are specified, the variables should be sorted in hierarchical order, and are included in multi-columns layout.
Use: 'variable' to include the variables to summarize: `var` (if multiple) in different columns.

`colTotalInclude`
: Logical, if TRUE (FALSE by default) include the summary statistics across columns in a separated column.

`colTotalLab`
: String, label for the total column 'Total' by default.

`subjectVar`
: String, variable of `data` with subject ID, 'USUBJID' by default.

`statsLayout`
: String with layout for the statistics names (in case more than one statistic is included), among:

  - row (by default for 'flextable' output):
    All statistics are included in different rows in the first column of the table (after the row variable(s))
  - 'col' (by default for 'DT' output):
    Statistics are included in separated columns (last row of the header).
    This option is not compatible with categorical variable(s).
  - 'rowInSepCol':
    Statistics are included in different rows, but in a separated column than the `rowVar` variable(s)

`statsValueLab`
: String with label for the statistic value, 'StatisticValue' by default.
This is only included in the table if the statistics provided in `stats` are not named and if no `colVar` is specified.

`statsExtra`
: (optional) Named list with functions for additional custom statistics to be computed.
Each function:

  - has as parameter, either: 'x': the variable (`var`) to compute the summary statistic on or 'data': the entire dataset
  - returns the corresponding summary statistic as a numeric vector

  For example, to additionally compute the coefficient of variation, this can be set to: `list(statCVPerc = function(x) sd(x)/mean(x)*100)` (or `cv`).

`type`
: String with type of table:

  - 'summaryTable': summary table with statistics for numeric variable
  - 'countTable': count table

- 'auto' (by default): 'summaryTable' if the variable is numeric, 'countTable' otherwise

`statsLabInclude`
: Logical, if TRUE include the statistic label in the table.
By default only included if more than one statistic variables are available in the table.

`title`
: Character vector with title(s) for the table. Set to NULL (by default) if no title should be included. If multiple are specified, specified for each element of `byVar` (in order of the levels).

`pageDim`
: Numeric vector of length 2 with page width and height.
Depending on `outputType`:

- 'flextable': in inches
- 'DT': in number of rows in the table.
  Currently only the height is used (e.g. `c(NA, 4)`)

`columnsWidth`
: (expert mode) Column widths of the table. This is only used for flextable and DT tables.
For flextable, note that the widths should be set to fit into the document page (see `getDimPage`).

`labelVars`
: (optional) Named character vector with label for the row, column variable(s) or variable(s) to summarize.
Labels specified via dedicated parameter: e.g. `rowVarLab`, `colVarLab`, `varLab` have priority on this parameter.

## Value

No return value, used for the documentation of R functions

---

`inTextSummaryTable-DT-args`
: *Common arguments for the functionalities of the inTextSummaryTable package for DT export.*

---

## Description

Common arguments for the functionalities of the inTextSummaryTable package for DT export.

## Arguments

`expandVar`
: (DT output) Character vector with variables of the summary table which should be expanded in the data.

`pageDim`
: (DT output) Numeric vector of length 2 with page width and height, in number of rows (currently only the height is used (e.g. `c(NA, 4)`)

`noEscapeVar`
: (DT output) Character vector with variables of `summaryTable` which shouldn't be escaped in the table (e.g. containing URLs).

barVar (DT output) Character vector with variables of summaryTable that should be represented as a bar.

file String with path of the file where the table should be exported. The file should have the extension: '.docx'. If NULL, the summary table is not exported but only returned as output. If byVar is specified, each table is exported to a separated file with the suffix: 'file_[i].docx' with i the index of the file.

## Value

No return value, used for the documentation of R functions for 'DT' output

---

# inTextSummaryTable-flextable-args

*Common arguments for the functionalities of the inTextSummaryTable package for flextable export.*

---

## Description

Common arguments for the functionalities of the inTextSummaryTable package for flextable export.

## Arguments

style (flextable output) String with table style, either 'report' or 'presentation'. This parameter affects the fontsize, font family, color of the text and background, and table dimensions of the table.

rowTotalLab (flextable output) string with label for the row with total.

rowAutoMerge (flextable output) Logical, if TRUE (by default) automatically merge rows, e.g. in case there is only one sub-category (e.g. categorical variable with only one group) or only one statistic per category.

rowVarFormat (flextable output) Named list with special formatting for the rowVar. Currently, only possibility is to set the variable elements in bold, with: list(var1 = "bold"). (Use 'variable' for var or 'variableGroup' for group within categorical variables.)

rowPadBase (flextable output) Base padding for row (in points), 14.4 by default (corresponds to 0.2 inches)

fontname (flextable output) String with font name, by default: 'Times' if style is 'report' and 'Tahoma' if style is 'presentation'.

fontsize (flextable output) Integer with font size, by default: 8 if style is 'report' and 10 if style is 'presentation'.

margin (flextable output) Margin in the document in inches (1 by default). This is used to specify the width of the table, from: [pageDim[1] - 2 * margin].

colorTable (flextable output) Named character vector with color for the table background/body/text/line, e.g. created with the getColorPaletteTable function.

| | |
|---|---|
| `landscape` | (flextable output) Logical, if TRUE the file is in landscape format.<br>By default: FALSE if `style` is 'report' and TRUE if `style` is 'presentation'. |
| `footer` | (flextable output) Character vector with footer(s) for the table. Set to NULL (by default) if no footer should be included. |
| `vline` | (flextable output) String mentioning how vertical lines should be included in the body of the table, either:<br>• 'none' (default): no vertical lines included<br>• 'auto': vertical lines included between sub-groups |
| `hline` | (flextable output) String mentioning how horizontal lines should be included in the body of the table, either:<br>• 'none': no horizontal lines included<br>• 'auto' (default): horizontal lines included between sub-groups |
| `file` | String with path of the file where the table should be exported. The file should have the extension: '.html'. If NULL, the summary table is not exported but only returned as output. If `byVar` is specified, each table is exported to a separated file with the suffix: 'file_[i].html' with i the index of the file. |
| `colHeaderMerge` | (flextable output) Logical, if TRUE (FALSE by default) the column header is merged. |

## Value

No return value, used for the documentation of R functions for 'flextable' output

---

`inTextSummaryTable-stats`

*Statistics in the in-text table package.*

---

## Description

In the in-text package, different set of statistics are available.

## Details

The statistics are first computed as numeric ('Base statistics' section below), then formatted to be displayed in the table ('Formatted statistics' section below).

## Value

No return value, used for the documentation of the `stats` parameter

**Base statistics**

In the in-text package, the following 'base statistics' are reported in the summary table:

- for a continuous variable:
  - 'statMean': variable mean
  - 'statSD': variable standard deviation
  - 'statSE': variable standard error
  - 'statMedian': variable median
  - 'statMin': variable minimum
  - 'statMax': variable maximum

  During the computation of the statistics, if multiple and different values are available for a specific variable and subject ID (by row/column): an error is triggered.
- for a categorical and continuous variable (or the full table):
  - 'statN': number of subjects
  - 'statm': number of records
  - 'statPercN' (or 'statPercm'): percentage of subjects (or records) for the specific group
  - 'statPercTotalN' (or 'statPercTotalm'): number of subjects (or records) considered for the total (denominator) of the percentage

  The percentage and denominator of the percentage are based on the number of subjects or records depending on the `statsPerc` parameter.

These statistics are reported as numeric and non rounded in the summary table, and are typically used as input for the formatted statistics, or for plots.

**Formatted statistics**

In the in-text package, the following formatted statistics can be reported in the final output table.

- for a continuous variable:
  - base statistics:
    - 'Mean': formatted mean
    - 'Median': formatted median
    - 'SE': formatted standard error
    - 'SD': formatted standard deviation
    - 'Min': formatted minimum
    - 'Max': formatted maximum
  - multiple:
    - 'summary-default': default set of statistics for a continuous variable: 'n', 'Mean', 'SD', 'SE', 'Median', 'Min', 'Max'
    - 'summary': all statistics available for a continuous variable: 'n', 'Mean', 'SD', 'SE', 'Median', 'Min', 'Max', '%', 'm'
  - combined statistics:
    - 'median (range)': median (minimum,maximum)
    - 'median\n(range)': median and (minimum, maximum) below (linebreak)

    - 'mean (sd)': mean and standard deviation
    - 'mean (se)': mean and standard error
    - 'mean (range)': mean and (minimum, maximum)
    - '(min, max)': (minimum, maximum)
- for a categorical or continuous variable (or the full table):
  - base statistics:
    - 'n': formatted number of subjects
    - 'm': formatted number of records
    - '%': formatted percentage of subjects
    - '%m': formatted percentage of records.
      Note: this is only available if the percentage of records is reported (`statsPerc` set to 'statm').
  - multiple:
    - 'count-default': default set of statistics for a categorical variable: 'n', '%'
    - 'count': all statistics available for a categorical variable: 'n', '%', 'm'
  - combined statistics:
    - 'n (%)': number of subjects (and associated percentage)
    - 'n/N (%)': number of subjects/total number of subjects (percentage)
    - 'm (%)': number of records (and associated percentage).
      Note: this is only available if the percentage of records is reported (`statsPerc` set to 'statm').

  #' These statistics are specified via the `stats` parameter (or `type` parameter of `getStats`). These statistics are reported as text variables in the summary table (as data.frame), and typically displayed inside the final table.

**Statistics formatting**

- In general, all rounding is handled with `roundHalfUpTextFormat`.
- statistics for continuous variable:
  - if the number of decimals (`nDecCont`) is specified:
    statistics are rounded with the following number of decimals, based on:
    - 'Min', 'Max': `nDecCont`
    - 'Mean', 'SD', 'Median': `nDecCont` + 1
    - 'SE': `nDecCont` + 2

    Note that the number of decimals is extracted from standard rules/data is the variable of interest is specified (e.g. via `var` in `getStatsData`).
  - if the number of decimals is not specified:
    a default format is set via the `formatC` function.
- statistics for counts:
  - number of subjects, records are rounded with the number of decimals specified via `nDecN` or `nDecm` (0 by default)
  - percentages are formatted by default with `formatPercentage`.
  - 'n (%)' and 'm (%)':

    * if the percentage of subjects/records is missing, '-' is reported
    * if the number of subjects/records is 0, '0' is reported instead of '0 (0%)'
    * otherwise the number and percentage of subjects/records are formatted as specified
  - 'n/N (%)':
    * if the percentage of subjects is missing, '-' is reported
    * if the number of subjects is 0, '0' is reported instead of '0/... (0%)'
    * otherwise the number and percentage of subjects and total are formatted as specified

---

`inTextSummaryTable-stats-utility` *Common arguments for the for the statistics utility functions of the inTextSummaryTable package.*

---

## Description

Common arguments for the for the statistics utility functions of the inTextSummaryTable package.

## Arguments

| | |
|---|---|
| `x` | Numeric vector. |
| `na.rm` | Logical, should NA value(s) be removed (FALSE by default)? |

## Value

No return value, used for the documentation of stat utility R functions

---

`pageDimPresentation` *Page dimension for powerpoints*

---

## Description

Page dimension for powerpoints

## Usage

```
pageDimPresentation
```

## Format

An object of class `numeric` of length 2.

## postProcessVarFlag *Post-process the summary statistics table with variable flag.*

### Description

This function is for internal use within the computeSummaryStatisticsTable function.

### Usage

```
postProcessVarFlag(summaryTable, varFlag)
```

### Arguments

| | |
|---|---|
| summaryTable | Summary table as created internally in computeSummaryStatisticsTable. |
| varFlag | Character vector, subset of var with variable(s) of type 'flag' (with 'Y', 'N' or '' for empty/non specified value). Only the counts for records flagged (with 'Y') are retained. |

### Details

This includes:

- converting the records from a flag variable for the 'variableGroup' variable from 'Y' to NA_character_
- filter records from a flag variable with variableGroup set as 'N'

### Value

Summary table with

### Author(s)

Laure Cougnaud

## se *Compute standard error of the mean.*

### Description

The standard error of the mean is computed as: $\frac{\sigma(x)}{\sqrt{length(x)}}$, with:

$\sigma(x)$: standard deviation of x

### Usage

```
se(x, na.rm = FALSE)
```

### Arguments

| | |
|---|---|
| x | Numeric vector. |
| na.rm | Logical, should NA value(s) be removed (FALSE by default)? |

### Value

Numeric vector with standard error of the mean

### Author(s)

Laure Cougnaud

### See Also

Other stats utility functions: cv(), geomCV(), geomMean(), geomSD(), geomSE()

### Examples

```
se(rnorm(1000))
```

---

## subjectProfileSummaryPlot

*Plot subject summary profile.*

---

### Description

The user can either specify a variable for the standard error (seVar), or directly the variables for the minimum and maximum values for the error bars (minVar, maxVar).

### Usage

```
subjectProfileSummaryPlot(
  data,
  xVar = NULL,
  xLab = getLabelVar(xVar, labelVars = labelVars),
  xAxisExpand = waiver(),
  xGap = NULL,
  xGapDiffNew = NULL,
  meanVar = "statMean",
  seVar = if ("statSE" %in% colnames(data)) "statSE",
  minVar = NULL,
  maxVar = NULL,
 yLab = paste(c(sub("^stat", "", meanVar), if (!is.null(minVar) & !is.null(maxVar)) {

     paste0("(", sub("^stat", "", minVar), ", ", sub("^stat", "", maxVar), ")")
 } else
    if (!is.null(seVar)) paste("+-", sub("^stat", "", seVar))), collapse = " "),
```

```
  facetVar = NULL,
  facetScale = "free_y",
  colorVar = NULL,
  colorLab = getLabelVar(colorVar, labelVars = labelVars),
  colorPalette = NULL,
  labelVars = NULL,
  useLinetype = TRUE,
  linetypePalette = NULL,
  useShape = TRUE,
  shapePalette = NULL,
  jitter = NULL,
  title = NULL,
  caption = NULL,
  yTrans = NULL,
  yLim = NULL,
  xLim = NULL,
  yAxisExpand = c(0.05, 0.05),
  yLimExpand = NULL,
  xAxisLabs = NULL,
  sizePoint = 1.5,
  sizeLine = 0.5,
  sizeLabel = 3.87,
  widthErrorBar = 0.9,
  tableText = NULL,
  tableTextFontface = 1,
  tableHeight = 0.1,
  tableYAxisLabs = !is.null(colorVar),
  tablePlotMargin = unit(0, "pt"),
  label = FALSE,
  labelPadding = unit(1, "lines"),
  byVar = NULL,
  hLine = NULL,
  hLineColor = "black",
  hLineLty = "solid",
  vLine = NULL,
  vLineColor = "black",
  vLineLty = "solid",
  style = "report",
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  fontsize = switch(style, report = 8, presentation = 10),
  themeFct = switch(style, report = theme_classic, presentation = theme_bw),
  themeIncludeVerticalGrid = TRUE,
  ggExtra = NULL,
  legendPosition = ifelse(!is.null(tableText), "none", "bottom"),
  ...
)
```

## Arguments

| | |
|---|---|
| data | Data.frame with summary statistics to represent in the plot, e.g. a summaryTable object. |
| xVar | String, variable of data with variable for the x-axis. |
| xLab | String with label for the xVar. |
| xAxisExpand | Object passed to the 'expand' parameter of: scale_x_continuous, (waiver by default). |
| xGap | (optional) Numeric vector of length 2 for which a gap should be created in the x-axis. Only available if xVar is specified and a numeric variable. Records with xVar within xGap are filtered from the plot, vertical lines are included at the min/max of the gap, and the gap is represented as '//' in the x-axis of the plot. |
| xGapDiffNew | Numeric vector of length 2 with new range of the xGap. If not specified, the minimum difference between consecutive x elements in the data is used. |
| meanVar | String, variable of data with the mean variable. |
| seVar | String, variable of data with the standard error. |
| minVar, maxVar | String, variables of data with minimum and maximum value for error bar. If both are specified, seVar is ignored. |
| yLab | String with label for the y-axis. If different labels should be used for different elements of byVar variable, the vector should be named with each corresponding element (collapsed with '.' if multiple). |
| facetVar | Character vector, variable(s) of data for facetting. |
| facetScale | String with type of scale used for facetting, 'free_y' by default (fixed scale in the x-axis and free in the y-axis). |
| colorVar | String, variable of data for coloring. |
| colorLab | String, label for colorVar, used in the legend. |
| colorPalette | (named) Vector with color palette. |
| labelVars | Named string with variable labels (names are the variable code). |
| useLinetype | Logical, if TRUE (FALSE by default) use also linetype to differenciate the variable specified via colorVar in the mean line. |
| linetypePalette | Vector with linetype(s), in case useLinetype is TRUE. |
| useShape | Logical, if TRUE (by default) colorVar is also used for the shape. |
| shapePalette | Named vector with shape palette for colorVar. |
| jitter | Numeric with jitter for the x-axis, only used if colorVar specified. |
| title | String with title for the plot. If different labels should be used for different elements of byVar variable, the vector should be named with each corresponding element (collapsed with '.' if multiple). |
| caption | String with caption for the plot, NULL by default. |
| yTrans | (optional) String with transformation for the y-axis. Currently only 'log10' (or NULL, default) is available. In case error bars go in the negative, their values are set to a 'small enough' value for plotting: min(data)/10 or yLim[1] if yLim is specified. |

| | |
|---|---|
| yLim | Vector of the length 2 with limits for the y-axis. |
| xLim | Vector of the length 2 with limits for the x-axis. |
| yAxisExpand | Expansion constants for the limits for the y-axis. See the documentation of the expand parameter of the scale_y_continuous function for the available values for this parameter. |
| yLimExpand | This parameter is deprecated, use yAxisExpand instead. |
| xAxisLabs | (optional) Named character vector with labels for the x-axis. |
| sizePoint | Size for the point. |
| sizeLine | Size for the line linking means and error bars. |
| sizeLabel | Size for the label, only used if label is not NULL. |
| widthErrorBar | Numeric vector of length 1 with width of error bar. |
| tableText | (optional) Character vector with colname of data or expression from colnames of data to be represented in the table below the plot. By default, no table is displayed. |
| tableTextFontface | Font face for the text included in the table. |
| tableHeight | Numeric of length 1 with height for the table. |
| tableYAxisLabs | Logical, if TRUE (by default) the labels of the colorVar are included in the y-axis of the table. |
| tablePlotMargin | Margin between the plot and the table, expressed as unit, 0 by default. |
| label | Logical or expression or list of expression. Points are labelled with meanVar if set to TRUE, or with the specified expression if label is an expression. If a list is specified, 'textLabel' (required) should contain expression to extract label, and 'textHjust' and 'textVjust' (optional) may contain expression specifying horizontal and vertical adjustment of the label. |
| labelPadding | Amount of padding (space) between each point and its label, 1.5 lines by default. See parameter point.padding of the geom_text_repel function. |
| byVar | Variable(s) of data for which separated plot(s) should be created. |
| hLine | (optional) numeric with y-intercept of line(s) to be added. If different thresholds should be used for different elements of the byVar or facetVar variables, the vector should be named with each corresponding element (collapsed with '.' if multiple). |
| hLineColor | String with color for hLine, 'black' by default. |
| hLineLty | String with linetype for hLine, 'solid' by default. |
| vLine | (optional) numeric with x-intercept of line(s) to be added. If different thresholds should be used for different elements of the byVar or facetVar variables, the vector should be named with each corresponding element (collapsed with '.' if multiple). |
| vLineColor | String with color for vLine, 'black' by default. |
| vLineLty | String with linetype for vLine, 'solid' by defaul |

| | |
|---|---|
| style | String with subject profile style. This affects the parameters: fontname, fontsize and themeFct. |
| fontname | String with font name, by default 'Times' if style is 'report' and 'Tahoma' if style is 'presentation'. |
| fontsize | Numeric vector of length 1 with font size, by default 8 if style is 'report' and 10 if style is 'presentation' |
| themeFct | Function with ggplot2 theme, by default theme_classic if style is 'report' and theme_bw if style is 'presentation'. |
| themeIncludeVerticalGrid | Logical, if TRUE (by default) include theme vertical grid lines (if present in themeFct). |
| ggExtra | Extra ggplot call of function generating such call(s) to be added in main plot. Multiple calls are set via a function, e.g. ggExtra = function(gg){gg + geom_vline(...) + geom_hline(...)}. If different calls should be used for different elements of the byVar variable, the vector should be named with each corresponding element (collapsed with '.' if multiple). |
| legendPosition | String with legend position. By default, 'bottom' of tableText is not specified, 'none' otherwise. |
| ... | Additional parameters for geom_text_repel or geom_text used for the label. |

## Value

ggplot object or list of such objects of byVar is specified.

## Author(s)

Laure Cougnaud

---

# subjectProfileSummaryTable

*Plot a table with* ggplot *of a text variable of interest.*

---

## Description

The labels extracted based on the text parameter and displayed at the x-position based on xVar and the y-position based on colorVar. Each group specified in the color variables are displayed in different lines in the plot.

## Usage

```
subjectProfileSummaryTable(
  data,
  xVar,
  text,
```

```
  xLim = NULL,
  colorVar = NULL,
  colorPalette = NULL,
  colorLab = getLabelVar(colorVar, labelVars = labelVars),
  fontface = 1,
  xLab = NULL,
  labelVars = NULL,
  caption = NULL,
  showLegend = TRUE,
  legendPosition = ifelse(showLegend, "right", "none"),
  yAxisLabs = FALSE,
  xAxisLabs = NULL,
  style = "report",
  fontname = switch(style, report = "Times", presentation = "Tahoma"),
  fontsize = switch(style, report = 8, presentation = 10),
  pointSize = 1.5,
  themeFct = switch(style, report = theme_classic, presentation = theme_bw),
  textSize = fontsize/ggplot2::.pt,
  xTrans = NULL
)
```

## Arguments

| | |
|---|---|
| data | Data.frame (in long format) with data for the table. |
| xVar | String, variable of data with variable for the x-axis. |
| text | Character vector with colnames of data or expression based on colnames of data to extract the text label. |
| xLim | Vector of the length 2 with limits for the x-axis. |
| colorVar | String, variable of data for coloring. |
| colorPalette | (named) Vector with color palette. |
| colorLab | String, label for colorVar, used in the legend. |
| fontface | Numeric, fontface for the text. |
| xLab | String with label for the x-axis. |
| labelVars | Named string with variable labels (names are the variable code). |
| caption | String with caption for the plot, NULL by default. |
| showLegend | Logical, should the legend be displayed? TRUE by default. |
| legendPosition | String with legendPosition, 'right' by default. |
| yAxisLabs | Logical, if TRUE include the labels in the y-axis. |
| xAxisLabs | Vector with labels for the x-axis if xVar is discrete or vector with limits if continuous. |
| style | String with subject profile style. This affects the parameters: fontname, fontsize and themeFct. |
| fontname | String with font name, by default 'Times' if style is 'report' and 'Tahoma' if style is 'presentation'. |

| | |
|---|---|
| fontsize | Numeric vector of length 1 with font size, by default 8 if style is 'report' and 10 if style is 'presentation' |
| pointSize | Numeric indicating the size of points in the legend, 1.5 by default |
| themeFct | Function with ggplot2 theme, by default theme_classic if style is 'report' and theme_bw if style is 'presentation'. |
| textSize | Size for the text. |
| xTrans | (optional) ggplot2 transformation for the x-axis. |

## Value

ggplot object

## Author(s)

Laure Cougnaud and Michela Pasetto

---

summaryTable summaryTable

---

## Description

1. The summary statistics are computed in computeSummaryStatisticsTable, which creates a summaryTable object.
2. This object is exported to diverse formats via export

Multiple summaryTable objects are combined together with combine.summaryTable.

## Details

The summaryTable is an intermediary object of the package. This contains the summary statistics as data.frame. This object contains:

- the row and column variable(s)
- the computed statistic(s):
  If type is:
  - 'summaryTable':
    - 'statN': number of subjects
    - 'statMean': mean of var
    - 'statSD': standard deviation of var
    - 'statSE': standard error of var
    - 'statMedian': median of var
    - 'statMin': minimum of var
    - 'statMax': maximum of var
    - 'statPerc': percentage of subjects

    - 'statPercTotalN': total number of subjects based on `dataTotalPerc`, denominator of `statPerc`
    - 'statm': number of records
  - 'countTable':
    - 'statN': number of subjects
    - 'statPercN' (or 'statPercm'): percentage of subjects (or records depending on `statsPerc`)
    - 'statPercTotalN' (or 'statPercTotalm'): total number of subjects (or records) based on `dataTotalPerc`, and used as denominator of `statPercN` (or 'statPercm')
    - 'statm': number of records

- computed statistics.
  The statistics are stored in columns corresponding to names of the `statsVar`.
  If the specified statistics are not named and of length 1, the statistics are stored in a column called: 'Statistic'.
- variables:
  - 'variable': variable name in case `var` is of length > 1
  - 'variableGroup': in case `var` is of length > 1 and for variable(s) used for count: elements of the variable
- 'isTotal': variable with logical flag, TRUE if the record contain the total by column

Additionally, the output contains an extra attribute 'summaryTable', which is a list composed of:

- 'statsVar': column name(s) of summary table with computed statistics included in the final table
- 'rowVar': column name(s) of summary table with row variable included in the final table. This parameter should be mainly used for qualitative variables and 'nests' together different rows in the final output table.
- 'rowVarLab': labels corresponding to the 'rowVar' attribute
- 'rowVarTotalInclude': row variables whose total will be included: `rowVarTotalInclude` and 'variableGroup' if the variable total should be included
- 'rowVarTotalInSepRow': row variables whose total will be included in a separated row: `rowVarTotalInSepRow` and 'variableGroup' if `varTotalInSepRow`
- 'colVar': column name(s) of summary table with column variable included in the final table
- 'colTotalLab': label for the total

## Value

Not relevant

## tableColorsPresentation *Colors for tables in a presentation style*

### Description

Default colors are

- header: white text on a blue background
- body: black text on a grey background
- footer: black text on a white background.

### Usage

```
tableColorsPresentation
```

### Format

An object of class `character` of length 11.

## tableColorsReport *Colors for tables in a report style*

### Description

Default colors are black text on a white background.

### Usage

```
tableColorsReport
```

### Format

An object of class `character` of length 7.

## uniqueVarWithOrder *Get unique variables with meaningful order.*

### Description

The following framework is followed:

1. get the unique elements in the vectors
2. for each of this element: get the average order across the different vectors
3. put `variable`, if present as second to last element
4. put `variableGroup`, if present, as last element
5. order the unique elements based on the extracted order

### Usage

```
uniqueVarWithOrder(...)
```

### Arguments

| | |
|---|---|
| ... | Lists |

### Value

Vector with unique and ordered elements.

### Author(s)

Laure Cougnaud

## writeTable *Custom function to write table to a text file*

### Description

This function is mainly a wrapper on `write.table`, with the specific options:

- no rownames
- no quoting
- tab separator

### Usage

```
writeTable(x, file, ...)
```

## Arguments

| | |
|---|---|
| x | Data.frame to export to the table, or list of such tables. |
| file | String with text file to export to. |
| ... | Any parameters passed to the write.table function. |

## Value

No returned value, the object x is exported to the specified file.

## Author(s)

Laure Cougnaud

# Index


∗ **datasets**
pageDimPresentation, 71
tableColorsPresentation, 81
tableColorsReport, 81
∗ **decimals**
formatPercentage, 33
getMaxNDecimals, 45
getMaxNDecimalsData, 46
getNDecimals, 47
getNDecimalsData, 48
∗ **stats utility functions**
cv, 21
geomCV, 38
geomMean, 39
geomSD, 40
geomSE, 41
se, 72

checkVar, 3
checkVarLabInclude, 4
combine, 4
combine.summaryTable, *5*, 5, *79*
combineColTotal, 6
combineVariables, 6
computeSummaryStatistics, 8
computeSummaryStatisticsTable, 10, *13*, *57*, *72*, *79*
convertSummaryStatisticsTableToFlextable, 16
convertVarFlag, 18
convertVarRowVarColVarToFactor, 18
convertVectToBinary, 19
createFlextableWithHeader, 20
cv, *9*, *15*, 21, *38–41*, *59*, *65*, *73*

datatable, *25, 26*, *29*, *61*, *63*

export, 22, *79*
export.summaryTable, *22*, 22
exportFlextableToDocx, 27
exportSummaryStatisticsTable *(*export.summaryTable*)*, 22
exportSummaryStatisticsTableToDT, 28
exportSummaryStatisticsTableToFlextable, 30

flextable, *17*, *20*, *25–27*, *33*, *44, 45*, *61*, *63*
formatC, *51*, *70*
formatPercentage, 33, *46–48*, *51*, *70*
formatSummaryStatisticsTable, *28*, *30*, 34
formatSummaryStatisticsTableFlextable, 35

geom_text, *77*
geom_text_repel, *76, 77*
geomCV, *21*, 38, *39–41*, *73*
geomMean, *21*, *38*, 39, *40, 41*, *73*
geomSD, *21*, *38, 39*, 40, *41*, *73*
geomSE, *21*, *38–40*, 41, *73*
getClinDT, *23*, *29*, *62*
getColorPaletteTable, *17*, *26*, *32*, 42, *45*, *62*, *67*
getDimPage, *17*, *25*, *32*, 42, *62*, *66*
getListing, 44
getMaxNDecimals, *34*, 45, *47, 48*
getMaxNDecimalsData, *34*, *46*, 46, *48*
getNDecimals, *34*, *45–47*, 47, *48*, *51*
getNDecimalsData, *34*, *46–48*, 48
getNDecimalsRule, *47*, 49
getPatternPosition, 50
getStats, 50, *53*, *70*
getStatsData, *15*, *51*, 52, *59*, *70*
getSummaryStatisticsTable, 54, *63*
ggplot, *77*, *79*

inTextSummaryTable, 63
inTextSummaryTable-common-args, 64
inTextSummaryTable-DT-args, 66
inTextSummaryTable-flextable-args, 67

inTextSummaryTable-package
(inTextSummaryTable), 63
inTextSummaryTable-stats, 68
inTextSummaryTable-stats-utility, 71
is.language, *15*, *59*

padding, *37*
pageDimPresentation, 71
postProcessVarFlag, 72

roundHalfUpTextFormat, *70*

scale_x_continuous, *75*
scale_y_continuous, *76*
se, *21*, *38–41*, 72
subjectProfileSummaryPlot, 73
subjectProfileSummaryTable, 77
summaryTable, *5*, *16*, *17*, *22*, *23*, *27*, *34*, *36*, *64*, *75*, 79

tableColorsPresentation, 81
tableColorsReport, 81
theme_bw, *77*, *79*
theme_classic, *77*, *79*

uniqueVarWithOrder, 82
unit, *76*

waiver, *75*
write.table, *82*, *83*
writeTable, 82