# Package ‘table.glue’

September 30, 2020

**Title** Make and Apply Customized Rounding Specifications for Tables

**Version** 0.0.1

**Description** Translate double and integer valued data into character values formatted for tabulation in manuscripts or other types of academic reports.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Suggests** testthat, covr, knitr, rmarkdown

**Imports** glue, stringi

**URL** https://github.com/bcjaeger/table.glue

**BugReports** https://github.com/bcjaeger/table.glue/issues

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Byron Jaeger [aut, cre] (<https://orcid.org/0000-0001-7399-2299>)

**Maintainer** Byron Jaeger <bcjaeger@uab.edu>

**Repository** CRAN

**Date/Publication** 2020-09-30 08:40:03 UTC

## R topics documented:

---

`format_big` *Format values left of decimal*

---

## Description

Values to the left of the decimal are generally called 'big' since they are larger than values to the right of the decimal. `format_big()` lets you update the settings of a `rounding_specification` object (see round_spec) so that values left of the decimal will be printed with a specific format (see examples).

## Usage

```
format_big(rspec, mark = ",", interval = 3L)
```

## Arguments

| | |
|---|---|
| `rspec` | a `rounding_specification` object (see round_spec). |
| `mark` | a character value used to separate number groups to the left of the decimal point. See prettyNum for more details on this. Set this input to '' to negate it's effect. |
| `interval` | a numeric value indicating the size of number groups for numbers left of the decimal. |

## Value

an object of class `rounding_specification`.

## Examples

```
big_x <- 1234567

rspec <- format_big(round_spec(), mark = '|', interval = 3)

table_value(big_x, rspec) # returns "1|234|567"
```

## `format_decimal` *Format decimal symbol*

### Description

`format_decimal()` lets you update the settings of a `rounding_specification` object (see round_spec) so that the decimal is represented by a user-specified mark.

### Usage

```
format_decimal(rspec, mark = ".")
```

### Arguments

| | |
|---|---|
| `rspec` | a `rounding_specification` object (see round_spec). |
| `mark` | a character value used to represent the decimal point. |

### Value

an object of class `rounding_specification`.

### See Also

Other formatting helpers: `format_small()`

### Examples

```
small_x <- 0.1234567

rspec <- round_spec()
rspec <- round_using_decimal(rspec, digits = 7)
rspec <- format_decimal(rspec, mark = '*')

table_value(small_x, rspec)
```

## `format_missing` *Format missing values*

### Description

`format_missing()` updates a `rounding_specification` object so that missing values are printed as the user specifies.

## Usage

```
format_missing(rspec, replace_na_with)
```

## Arguments

| | |
|---|---|
| rspec | a rounding_specification object (see round_spec). |
| replace_na_with | a character value that replaces missing values. |

## Value

an object of class rounding_specification.

## Examples

```
rspec <- round_spec()
rspec <- format_missing(rspec, 'oh no!')
table_value(x = c(pi, NA), rspec)
```

---

# format_small *Format values right of decimal*

---

## Description

Values to the right of the decimal are generally called 'small' since they are smaller than values to the left of the decimal. format_small() lets you update the settings of a rounding_specification object (see round_spec) so that values right of the decimal will be printed with a specific format (see examples).

## Usage

```
format_small(rspec, mark = "", interval = 5L)
```

## Arguments

| | |
|---|---|
| rspec | a rounding_specification object (see round_spec). |
| mark | a character value used to separate number groups to the right of the decimal point. See prettyNum for more details on this. Set this input to '' to negate it's effect. |
| interval | a numeric value indicating the size of number groups for numbers left of the decimal. |

## Value

an object of class rounding_specification.

## See Also

Other formatting helpers: `format_decimal()`

## Examples

```
small_x <- 0.1234567

rspec <- round_spec()
rspec <- round_using_decimal(rspec, digits = 7)
rspec <- format_small(rspec, mark = '*', interval = 1)

table_value(small_x, rspec)
```

---

`round_half_up` *Set rules for rounding ties*

---

## Description

Rounding a number x to the nearest integer requires some tie-breaking rule for those cases when x is exactly half-way between two integers, that is, when the fraction part of x is exactly 0.5. The `round_half_up()` function implements a tie-breaking rule that consistently rounds half units upward. Although this creates a slight bias toward larger rounded outputs, it is widely used in many disciplines. The `round_half_even()` function breaks ties by rounding to the nearest even unit.

## Usage

```
round_half_up(rspec)

round_half_even(rspec)
```

## Arguments

| | |
|---|---|
| `rspec` | a `rounding_specification` object (see round_spec). |

## Value

an object of class `rounding_specification`.

## See Also

Other rounding helpers: `round_using_magnitude()`

## Examples

```
# note base R behavior rounds to even:
round(0.5) # --> 0
round(1.5) # --> 2
round(2.5) # --> 2

# make rspec that rounds up
rspec <- round_half_up(round_spec())
rspec <- round_using_decimal(rspec, digits = 0)

# check
table_value(0.5, rspec) # --> 1
table_value(1.5, rspec) # --> 2
table_value(2.5, rspec) # --> 3

# make rspec that rounds even
rspec <- round_half_even(round_spec())
rspec <- round_using_decimal(rspec, digits = 0)

# check
table_value(0.5, rspec) # --> 0
table_value(1.5, rspec) # --> 2
table_value(2.5, rspec) # --> 2
```

---

`round_spec` *Make a rounding specification*

---

## Description

`round_spec()` creates a rounding specification object with default settings. The settings of a rounding specification object can be updated using functions in the `round_` (see round_half_up, round_half_even, round_using_signif, round_using_decimal, and round_using_magnitude) and `format_` (see format_missing, format_big, format_small, and format_decimal) families.

## Usage

```
round_spec(force_default = FALSE)
```

## Arguments

| | |
|---|---|
| `force_default` | a logical value. If `TRUE`, then `round_spec()` ignores global options and uses its factory default values. If `FALSE`, `round_spec()` will access global options to determine its settings. |

### Details

Rounding specifications are meant to be passed into the table_glue and table_value functions. The specification can also be passed into `table_` functions implicitly by saving a rounding specification into the global options.

The `round_spec()` function intentionally uses no input arguments. This is to encourage users to develop rounding specifications using the `round_` and `format_` families in conjunction with the pipe (%>%) operator.

### Value

an object of class `rounding_specification`.

### Examples

```
rspec <- round_spec()

table_value(x = pi, rspec)
```

---

## `round_using_magnitude` *Set rules for rounding numbers*

---

### Description

These functions update a `rounding_specification` object (see round_spec) so that a particular approach to rounding is applied:

- round to a dynamic decimal place based on magnitude of the rounded number (`round_using_magnitude()`)
- round to a specific number of significant digits (`round_using_signif()`)
- round to a specific decimal place (`round_using_decimal()`)

### Usage

```
round_using_magnitude(rspec, digits = c(2, 1, 0), breaks = c(1, 10, Inf))

round_using_signif(rspec, digits = 2)

round_using_decimal(rspec, digits = 1)
```

### Arguments

| | |
|---|---|
| `rspec` | a `rounding_specification` object (see round_spec). |
| `digits` | for `round_using_decimal()` and `round_using_signif`, a numeric value specifying the number of decimal places and significant digits to round to, respectively. For `round_using_magnitude()`, `digits` should be a numeric vector of equal length to `breaks` that indicates how many decimals to round to in the numeric range designated by `breaks`. (see notes for example). |

breaks (only relevant if rounding based on magnitude) a positive, monotonically increasing numeric vector designating rounding boundaries.

## Details

digits and breaks must be used in coordination with each other when rounding based on magnitude. For example, using breaks = c(1,10,Inf) and decimals = c(2,1,0),

- numbers whose absolute value is < 1 are rounded to 2 decimal places,
- numbers whose absolute value is >= 1 and < 10 are rounding to 1 decimal place, and
- numbers whose absolute value is >= 10 are rounding to 0 decimal places. The use of magnitude to guide rounding rules is extremely flexible and can be used for many different applications (e.g., see table_pvalue). Rounding by magnitude is similar in some ways to rounding to a set number of significant digits but not entirely the same (see examples).

## Value

an object of class rounding_specification.

## See Also

Other rounding helpers: round_half_up()

## Examples

```
x <- c(pi, exp(1))
x <- c(x, x*10, x*100, x*1000)

# make one specification using each rounding approach
specs <- list(
  magnitude = round_using_magnitude(round_spec()),
  decimal = round_using_decimal(round_spec()),
  signif = round_using_signif(round_spec())
)

# apply all three rounding specifications to x
# notice how the rounding specifications are in agreement
# for smaller values of x but their answers are different
# for larger values of x.

sapply(specs, function(rspec) table_value(x, rspec))

# output:
#  magnitude   decimal    signif
# [1,] "3.1"     "3.1"     "3.1"
# [2,] "2.7"     "2.7"     "2.7"
# [3,] "31"      "31.4"    "31.0"
# [4,] "27"      "27.2"    "27.0"
# [5,] "314"     "314.2"   "310.0"
# [6,] "272"     "271.8"   "270.0"
```

```
# [7,] "3,142"   "3,141.6" "3,100.0"
# [8,] "2,718"   "2,718.3" "2,700.0"
```

---

table_ester *Round estimates and their corresponding errors*

---

## Description

Though they are not easy to find in print, there are some general conventions for rounding numbers. When rounding a summary statistic such as the mean or median, the number of rounded digits shown should be governed by the precision of the statistic. For instance, authors are usually asked to present means plus or minus standard deviations in published research, or regression coefficients plus or minus the standard error. The convention applied here is to

1. find the place of the first significant digit of the error
2. round the estimate to that place
3. round the error to 1 additional place
4. present the combination in a form such as estimate (error) or estimate +/- error

## Usage

```
table_ester(
  estimate,
  error,
  form = "{estimate} ± {error}",
  majority_rule = FALSE
)

table_estin(
  estimate,
  lower,
  upper,
  form = "{estimate} ({lower}, {upper})",
  majority_rule = FALSE
)
```

## Arguments

| | |
|---|---|
| `estimate` | a numeric vector of estimate values. |
| `error` | a numeric vector of error values. All errors should be >0. |
| `form` | a character value that indicates how the error and estimate should be formatted together. Users can specify anything they like as long as they use the terms `estimate` and `error` to refer to the estimate and error values, respectively, and encapsulate those terms inside of curly brackets, i.e., . For instance, if estimate = 1.23 and error = 0.45, then `form` = "estimate (error)" will return "1.2 (0.45)", a common format used in presentation of the point and error combination. The default `form` gives output in the form of 1.2 +/- 0.45. |

majority_rule a logical value. If TRUE, then the most common digit used for rounding will be used to round every number given. Within a single table, consistency in saving digits may be desirable, so all numbers may be rounded to the place indicated by the majority of the numbers. Notably, if a user wants to exercise more control over the number of decimals shown, they should use `table_glue()` with a customized rounding specification (see round_spec).

lower the lower-bound of an interval for the estimate.

upper the upper-bound of an interval for the estimate.

### Value

a character vector

### References

Blackstone, Eugene H. "Rounding numbers" (2016): *The Journal of Thoracic and Cardiovascular Surgery*. DOI: https://doi.org/10.1016/j.jtcvs.2016.09.003

### See Also

Other table helpers: `table_glue()`, `table_pvalue()`, `table_value()`

### Examples

```
# ---- examples are taken from Blackstone, 2016 ----

# Example 1: ----
# Mean age is 72.17986, and the standard deviation (SD) is 9.364132.
## Steps:
## - Nine is the first significant figure of the SD.
## - Nine is in the ones place. Thus...
##   + round the mean to the ones place (i.e., round(x, digits = 0))
##   + round the SD to the tenths place (i.e., round(x, digits = 1))
table_ester(estimate = 72.17986, error = 9.364132)
# > [1] 72 +/- 9.4

# an estimated lower and upper bound for 95% confidence limits
lower <- 72.17986 - 1.96 * 9.364132
upper <- 72.17986 + 1.96 * 9.364132
table_estin(estimate = 72.17986, lower = lower, upper = upper,
            form = "{estimate} (95% CI: {lower}, {upper})")
# > [1] "72 (95% CI: 54, 91)"

# Example 2: ----
# Mean cost is $72,347.23, and the standard deviation (SD) is $23,994.06.
## Steps:
## - Two is the first significant figure of the SD.
## - Nine is in the ten thousands place. Thus...
##   + round mean to the 10-thousands place (i.e., round(x, digits = -4))
##   + round SD to the thousands place (i.e., round(x, digits = -3))
```

```
table_ester(estimate = 72347.23, error = 23994.06)
# > [1] "70,000 +/- 24,000"

# an estimated lower and upper bound for 95% confidence limits
lower <- 72347.23 - 1.96 * 23994.06
upper <- 72347.23 + 1.96 * 23994.06
table_estin(estimate = 72347.23, lower = lower, upper = upper,
            form = "{estimate} (95% CI: {lower} - {upper})")
# > [1] "70,000 (95% CI: 30,000 - 120,000)"
```

---

table_glue — *Expressive rounding for table values*

---

## Description

Expressive rounding for table values

## Usage

```
table_glue(..., rspec = NULL, .sep = "", .envir = parent.frame())
```

## Arguments

| | |
|---|---|
| `...` | strings to round and format. Multiple inputs are concatenated together. Named arguments are **not** supported. |
| `rspec` | a `rounding_specification` object. If no `rspec` is given, a default setting will round values to decimal places based on the magnitude of the values. |
| `.sep` | Separator used to separate elements |
| `.envir` | environment to evaluate each expression in. |

## Value

a character vector of length equal to the vectors supplied in `...`

## See Also

Other table helpers: `table_ester()`, `table_pvalue()`, `table_value()`

## Examples

```
x <- runif(10)
y <- runif(10)

table_glue("{x} / {y} = {x/y}")
```

```
table_glue("{x}", "({100 * y}%)", .sep = ' ')

df = data.frame(x = 1:10, y=1:10)

table_glue("{x} / {y} = {as.integer(x/y)}", .envir = df)
table_glue("{x} / {y} = {as.integer(x/y)}")

with(df, table_glue("{x} / {y} = {as.integer(x/y)}"))

mtcars$car <- rownames(mtcars)
# use the default rounding specification
table_glue(
  "the {car} gets ~{mpg} miles/gallon and weighs ~{wt} thousand lbs",
  .envir = mtcars[1:3, ]
)

# use your own rounding specification
rspec <- round_spec()
rspec <- round_using_decimal(rspec, digits = 1)

table_glue(
  "the {car} gets ~{mpg} miles/gallon and weighs ~{wt} thousand lbs",
  rspec = rspec,
  .envir = mtcars[1:3, ]
)
```

---

`table_pvalue` *Round p-values*

---

## Description

When presenting p-values, journals tend to request a lot of finessing. `table_pvalue()` is meant to do almost all of the finessing for you. The part it does not do is *interpret* the p-value. For that, please see the guideline on interpretation of p-values by the American Statistical Association (Wasserstein, 2016). The six main statements on p-value usage are included in the "Interpreting p-values" section below.

## Usage

```
table_pvalue(
  x,
  round_half_to = "even",
  decimals_outer = 3L,
  decimals_inner = 2L,
  alpha = 0.05,
  bound_inner_low = 0.01,
  bound_inner_high = 0.99,
  bound_outer_low = 0.001,
  bound_outer_high = 0.999,
```

```
  miss_replace = "--",
  drop_leading_zero = TRUE
)
```

## Arguments

- `x` — a vector of numeric values. All values should be > 0 and < 1.
- `round_half_to` — a character value indicating how to break ties when the rounded unit is exactly halfway between two rounding points. See round_half_even and round_half_up for details. Valid inputs are 'even' and 'up'.
- `decimals_outer` — number of decimals to print when p > bound_outer_high or p < bound_outer_low.
- `decimals_inner` — number of decimals to print when `bound_outer_low` < p < `bound_outer_high`.
- `alpha` — a numeric value indicating the significance level, i.e. the probability that you will make the mistake of rejecting the null hypothesis when it is true.
- `bound_inner_low` — the lower bound of the inner range.
- `bound_inner_high` — the upper bound of the inner range.
- `bound_outer_low` — the lowest value printed. Values lower than the threshold will be printed as <threshold.
- `bound_outer_high` — the highest value printed. Values higher than the threshold will be printed as >threshold.
- `miss_replace` — a character value that replaces missing values.
- `drop_leading_zero` — a logical value. If `TRUE`, the leading 0 is dropped for all p-values. So, '0.04' will become '.04'. If `FALSE`, no leading zeroes are dropped.

## Value

a character vector

## Interpreting p-values

The American Statistical Association (ASA) defines the p-value as follows:

A p-value is the probability under a specified statistical model that a statistical summary of the data (e.g., the sample mean difference between two compared groups) would be equal to or more extreme than its observed value.

It then provides six principles to guide p-value usage:

1. P-values can indicate how incompatible the data are with a specified statistical model.A p-value provides one approach to summarizing the incompatibility between a particular set of data and a proposed model for the data. The most common context is a model, constructed under a set of assumptions, together with a so-called "null hypothesis". Often the null hypothesis postulates the absence of an effect, such as no difference between two groups, or the absence

of a relationship between a factor and an outcome. The smaller the p-value, the greater the statistical incompatibility of the data with the null hypothesis, if the underlying assumptions used to calculate the p-value hold. This incompatibility can be interpreted as casting doubt on or providing evidence against the null hypothesis or the underlying assumptions.

2. P-values do not measure the probability that the studied hypothesis is true, or the probability that the data were produced by random chance alone. Researchers often wish to turn a p-value into a statement about the truth of a null hypothesis, or about the probability that random chance produced the observed data. The p-value is neither. It is a statement about data in relation to a specified hypothetical explanation, and is not a statement about the explanation itself.
3. Scientific conclusions and business or policy decisions should not be based only on whether a p-value passes a specific threshold. Practices that reduce data analysis or scientific inference to mechanical "bright-line" rules (such as "p < 0.05") for justifying scientific claims or conclusions can lead to erroneous beliefs and poor decision making. A conclusion does not immediately become "true" on one side of the divide and "false" on the other. Researchers should bring many contextual factors into play to derive scientific inferences, including the design of a study, the quality of the measurements, the external evidence for the phenomenon under study, and the validity of assumptions that underlie the data analysis. Pragmatic considerations often require binary, "yes-no" decisions, but this does not mean that p-values alone can ensure that a decision is correct or incorrect. The widespread use of "statistical significance" (generally interpreted as "p<0.05") as a license for making a claim of a scientific finding (or implied truth) leads to considerable distortion of the scientific process.
4. Proper inference requires full reporting and transparency P-values and related analyses should not be reported selectively. Conducting multiple analyses of the data and reporting only those with certain p-values (typically those passing a significance threshold) renders the reported p-values essentially uninterpretable. Cherry picking promising findings, also known by such terms as data dredging, significance chasing, significance questing, selective inference, and "p-hacking," leads to a spurious excess of statistically significant results in the published literature and should be vigorously avoided. One need not formally carry out multiple statistical tests for this problem to arise: Whenever a researcher chooses what to present based on statistical results, valid interpretation of those results is severely compromised if the reader is not informed of the choice and its basis. Researchers should disclose the number of hypotheses explored during the study, all data collection decisions, all statistical analyses conducted, and all p-values computed. Valid scientific conclusions based on p-values and related statistics cannot be drawn without at least knowing how many and which analyses were conducted, and how those analyses (including p-values) were selected for reporting.
5. A p-value, or statistical significance, does not measure the size of an effect or the importance of a result. Statistical significance is not equivalent to scientific, human, or economic significance. Smaller p-values do not necessarily imply the presence of larger or more important effects, and larger p-values do not imply a lack of importance or even lack of effect. Any effect, no matter how tiny, can produce a small p-value if the sample size or measurement precision is high enough, and large effects may produce unimpressive p-values if the sample size is small or measurements are imprecise. Similarly, identical estimated effects will have different p-values if the precision of the estimates differs.
6. By itself, a p-value does not provide a good measure of evidence regarding a model or hypothesis. Researchers should recognize that a p-value without context or other evidence provides limited information. For example, a p-value near 0.05 taken by itself offers only weak evidence against the null hypothesis. Likewise, a relatively large p-value does not imply evidence

in favor of the null hypothesis; many other hypotheses may be equally or more consistent with the observed data. For these reasons, data analysis should not end with the calculation of a p-value when other approaches are appropriate and feasible.

### References

Wasserstein, Ronald L., and Nicole A. Lazar. "The ASA statement on p-values: context, process, and purpose." (2016): *The American Statistician*: 129-133. DOI: https://doi.org/10.1080/00031305.2016.1154108

### See Also

Other table helpers: `table_ester()`, `table_glue()`, `table_value()`

### Examples

```
# Guideline by the American Medical Association Manual of Style:
# Round p-values to 2 or 3 digits after the decimal point depending
# on the number of zeros. For example,
## - Change .157 to .16.
## - Change .037 to .04.
## - Don't change .047 to .05, because it will no longer be significant.
## - Keep .003 as is because 2 zeros after the decimal are fine.
## - Change .0003 or .00003 or .000003 to <.001
#
# In addition, the guideline states that "expressing P to more than 3
# significant digits does not add useful information." You may or may not
# agree with this guideline (I do not agree with parts of it),
# but you will (hopefully) appreciate `table_pvalue()` automating these
# recommendations if you submit papers to journals associated with
# the American Medical Association.

pvals_ama <- c(0.157, 0.037, 0.047, 0.003, 0.0003, 0.00003, 0.000003)

table_pvalue(pvals_ama)
# > [1] ".16"   ".04"   ".047"  ".003"  "<.001" "<.001" "<.001"

# `table_pvalue()` will fight valiantly to keep your p-value < alpha if
# it is < alpha. If it's >= alpha, `table_pvalue()` treats it normally.

pvals_close <- c(0.04998, 0.05, 0.050002)

table_pvalue(pvals_close)
# > [1] ".04998" ".05"    ".05"
```

---

`table_value` *General rounding for tables*

---

## Description

`table_value()` casts numeric vectors into character vectors. The main purpose of `table_value()` is to round and format numeric data for presentation.

## Usage

```
table_value(x, rspec = NULL)
```

## Arguments

| | |
|---|---|
| `x` | a vector of numeric values. |
| `rspec` | a `rounding_specification` object. If no `rspec` is given, a default setting will round values to decimal places based on the magnitude of the values. |

## Value

a vector of character values (rounded numbers).

## See Also

Other table helpers: `table_ester()`, `table_glue()`, `table_pvalue()`

## Examples

```
table_value(0.123)
table_value(1.23)
table_value(12.3)

with(mtcars, table_value(disp))
```

# Index