Package 'TidyDensity'

April 26, 2024

Title Functions for Tidy Analysis and Generation of Random Data

Version 1.4.0

Description To make it easy to generate random numbers based upon the underlying stats distribution functions. All data is returned in a tidy and structured format making working with the data simple and straight forward. Given that the data is returned in a tidy 'tibble' it lends itself to working with the rest of the 'tidyverse'.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

URL https://github.com/spsanderson/TidyDensity

BugReports https://github.com/spsanderson/TidyDensity/issues

Depends R (>= 4.1.0)

Imports magrittr, rlang (>= 0.4.11), dplyr, ggplot2, plotly, tidyr, purrr, actuar, methods, stats, patchwork, survival, nloptr, broom, tidyselect, data.table, stringr

Suggests rmarkdown, knitr, EnvStats

VignetteBuilder knitr

NeedsCompilation no

Author Steven Sanderson [aut, cre, cph] (<https://orcid.org/0009-0006-7661-8247>)

Maintainer Steven Sanderson <spsanderson@gmail.com>

Repository CRAN

Date/Publication 2024-04-26 13:20:02 UTC

R topics documented:

`bootstrap_density_augment` *Bootstrap Density Tibble*

Description

Add density information to the output of `tidy_bootstrap()`, and `bootstrap_unnest_tbl()`.

Usage

```
bootstrap_density_augment(.data)
```

Arguments

.data	The data that is passed from the tidy_bootstrap() or bootstrap_unnest_tbl() functions.

Details

This function takes as input the output of the tidy_bootstrap() or bootstrap_unnest_tbl() and returns an augmented tibble that has the following columns added to it: x, y, dx, and dy.

It looks for an attribute that comes from using tidy_bootstrap() or bootstrap_unnest_tbl() so it will not work unless the data comes from one of those functions.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bootstrap: bootstrap_p_augment(), bootstrap_p_vec(), bootstrap_q_augment(), bootstrap_q_vec(), bootstrap_stat_plot(), bootstrap_unnest_tbl(), tidy_bootstrap()

Other Augment Function: bootstrap_p_augment(), bootstrap_q_augment()

Examples

```
x <- mtcars$mpg

tidy_bootstrap(x) |>
  bootstrap_density_augment()

tidy_bootstrap(x) |>
  bootstrap_unnest_tbl() |>
  bootstrap_density_augment()
```

`bootstrap_p_augment` *Augment Bootstrap P*

Description

Takes a numeric vector and will return the ecdf probability.

Usage

```
bootstrap_p_augment(.data, .value, .names = "auto")
```

Arguments

`.data`	The data being passed that will be augmented by the function.
`.value`	This is passed `rlang::enquo()` to capture the vectors you want to augment.
`.names`	The default is "auto"

Details

Takes a numeric vector and will return the ecdf probability of that vector. This function is intended to be used on its own in order to add columns to a tibble.

Value

A augmented tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Augment Function: `bootstrap_density_augment()`, `bootstrap_q_augment()`

Other Bootstrap: `bootstrap_density_augment()`, `bootstrap_p_vec()`, `bootstrap_q_augment()`, `bootstrap_q_vec()`, `bootstrap_stat_plot()`, `bootstrap_unnest_tbl()`, `tidy_bootstrap()`

Examples

```
x <- mtcars$mpg
tidy_bootstrap(x) |>
  bootstrap_unnest_tbl() |>
  bootstrap_p_augment(y)
```

`bootstrap_p_vec` *Compute Bootstrap P of a Vector*

Description

This function takes in a vector as it's input and will return the ecdf probability of a vector.

Usage

```
bootstrap_p_vec(.x)
```

Arguments

`.x`	A numeric

Details

A function to return the ecdf probability of a vector.

Value

A vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bootstrap: `bootstrap_density_augment()`, `bootstrap_p_augment()`, `bootstrap_q_augment()`, `bootstrap_q_vec()`, `bootstrap_stat_plot()`, `bootstrap_unnest_tbl()`, `tidy_bootstrap()`

Other Vector Function: `bootstrap_q_vec()`, `cgmean()`, `chmean()`, `ckurtosis()`, `cmean()`, `cmedian()`, `csd()`, `cskewness()`, `cvar()`, `tidy_kurtosis_vec()`, `tidy_scale_zero_one_vec()`, `tidy_skewness_vec()`

Examples

```
x <- mtcars$mpg
bootstrap_p_vec(x)
```

`bootstrap_q_augment` *Augment Bootstrap Q*

Description

Takes a numeric vector and will return the quantile.

Usage

```
bootstrap_q_augment(.data, .value, .names = "auto")
```

Arguments

`.data`	The data being passed that will be augmented by the function.
`.value`	This is passed `rlang::enquo()` to capture the vectors you want to augment.
`.names`	The default is "auto"

Details

Takes a numeric vector and will return the quantile of that vector. This function is intended to be used on its own in order to add columns to a tibble.

Value

A augmented tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Augment Function: `bootstrap_density_augment()`, `bootstrap_p_augment()`

Other Bootstrap: `bootstrap_density_augment()`, `bootstrap_p_augment()`, `bootstrap_p_vec()`, `bootstrap_q_vec()`, `bootstrap_stat_plot()`, `bootstrap_unnest_tbl()`, `tidy_bootstrap()`

Examples

```
x <- mtcars$mpg

tidy_bootstrap(x) |>
  bootstrap_unnest_tbl() |>
  bootstrap_q_augment(y)
```

bootstrap_q_vec *Compute Bootstrap Q of a Vector*

Description

This function takes in a vector as it's input and will return the quantile of a vector.

Usage

```
bootstrap_q_vec(.x)
```

Arguments

.x	A numeric

Details

A function to return the quantile of a vector.

Value

A vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bootstrap: bootstrap_density_augment(), bootstrap_p_augment(), bootstrap_p_vec(), bootstrap_q_augment(), bootstrap_stat_plot(), bootstrap_unnest_tbl(), tidy_bootstrap()

Other Vector Function: bootstrap_p_vec(), cgmean(), chmean(), ckurtosis(), cmean(), cmedian(), csd(), cskewness(), cvar(), tidy_kurtosis_vec(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
x <- mtcars$mpg

bootstrap_q_vec(x)
```

`bootstrap_stat_plot` *Bootstrap Stat Plot*

Description

This function produces a plot of a cumulative statistic function applied to the bootstrap variable from `tidy_bootstrap()` or after `bootstrap_unnest_tbl()` has been applied to it.

Usage

```
bootstrap_stat_plot(
  .data,
  .value,
  .stat = "cmean",
  .show_groups = FALSE,
  .show_ci_labels = TRUE,
  .interactive = FALSE
)
```

Arguments

`.data`	The data that comes from either `tidy_bootstrap()` or after `bootstrap_unnest_tbl()` is applied to it.
`.value`	The value column that the calculations are being applied to.
`.stat`	The cumulative statistic function being applied to the `.value` column. It must be quoted. The default is "cmean".
`.show_groups`	The default is FALSE, set to TRUE to get output of all simulations of the bootstrap data.
`.show_ci_labels`	The default is TRUE, this will show the last value of the upper and lower quantile.
`.interactive`	The default is FALSE, set to TRUE to get a plotly plot object back.

Details

This function will take in data from either `tidy_bootstrap()` directly or after apply `bootstrap_unnest_tbl()` to its output. There are several different cumulative functions that can be applied to the data.The accepted values are:

- "cmean" - Cumulative Mean
- "chmean" - Cumulative Harmonic Mean
- "cgmean" - Cumulative Geometric Mean
- "csum" = Cumulative Sum
- "cmedian" = Cumulative Median

- "cmax" = Cumulative Max
- "cmin" = Cumulative Min
- "cprod" = Cumulative Product
- "csd" = Cumulative Standard Deviation
- "cvar" = Cumulative Variance
- "cskewness" = Cumulative Skewness
- "ckurtosis" = Cumulative Kurtotsis

Value

A plot either ggplot2 or plotly.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bootstrap: `bootstrap_density_augment()`, `bootstrap_p_augment()`, `bootstrap_p_vec()`, `bootstrap_q_augment()`, `bootstrap_q_vec()`, `bootstrap_unnest_tbl()`, `tidy_bootstrap()`

Other Autoplot: `tidy_autoplot()`, `tidy_combined_autoplot()`, `tidy_four_autoplot()`, `tidy_multi_dist_autoplot` `tidy_random_walk_autoplot()`

Examples

```
x <- mtcars$mpg

tidy_bootstrap(x) |>
  bootstrap_stat_plot(y, "cmean")

tidy_bootstrap(x, .num_sims = 10) |>
  bootstrap_stat_plot(y,
    .stat = "chmean", .show_groups = TRUE,
    .show_ci_label = FALSE
  )
```

---

bootstrap_unnest_tbl *Unnest Tidy Bootstrap Tibble*

Description

Unnest the data output from `tidy_bootstrap()`.

Usage

```
bootstrap_unnest_tbl(.data)
```

Arguments

.data	The data that is passed from the tidy_bootstrap() function.

Details

This function takes as input the output of the tidy_bootstrap() function and returns a two column tibble. The columns are sim_number and y

It looks for an attribute that comes from using tidy_bootstrap() so it will not work unless the data comes from that function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bootstrap: bootstrap_density_augment(), bootstrap_p_augment(), bootstrap_p_vec(), bootstrap_q_augment(), bootstrap_q_vec(), bootstrap_stat_plot(), tidy_bootstrap()

Examples

```
tb <- tidy_bootstrap(.x = mtcars$mpg)
bootstrap_unnest_tbl(tb)

bootstrap_unnest_tbl(tb) |>
  tidy_distribution_summary_tbl(sim_number)
```

---

cgmean *Cumulative Geometric Mean*

Description

A function to return the cumulative geometric mean of a vector.

Usage

```
cgmean(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative geometric mean of a vector. `exp(cummean(log(.x)))`

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: `bootstrap_p_vec()`, `bootstrap_q_vec()`, `chmean()`, `ckurtosis()`, `cmean()`, `cmedian()`, `csd()`, `cskewness()`, `cvar()`, `tidy_kurtosis_vec()`, `tidy_scale_zero_one_vec()`, `tidy_skewness_vec()`

Examples

```
x <- mtcars$mpg

cgmean(x)
```

---

check_duplicate_rows *Check for Duplicate Rows in a Data Frame*

Description

This function checks for duplicate rows in a data frame.

Usage

```
check_duplicate_rows(.data)
```

Arguments

`.data`	A data frame.

Details

This function checks for duplicate rows by comparing each row in the data frame to every other row. If a row is identical to another row, it is considered a duplicate.

Value

A logical vector indicating whether each row is a duplicate or not.

Author(s)

Steven P. Sanderson II, MPH

See Also

duplicated, anyDuplicated

Other Utility: convert_to_ts(), quantile_normalize(), tidy_mcmc_sampling(), util_beta_aic(), util_binomial_aic(), util_cauchy_aic(), util_chisq_aic(), util_exponential_aic(), util_gamma_aic(), util_geometric_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_pareto_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
data <- data.frame(
  x = c(1, 2, 3, 1),
  y = c(2, 3, 4, 2),
  z = c(3, 2, 5, 3)
)

check_duplicate_rows(data)
```

---

chmean *Cumulative Harmonic Mean*

Description

A function to return the cumulative harmonic mean of a vector.

Usage

```
chmean(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative harmonic mean of a vector. `1 / (cumsum(1 / .x))`

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), ckurtosis(), cmean(), cmedian(), csd(), cskewness(), cvar(), tidy_kurtosis_vec(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
x <- mtcars$mpg

chmean(x)
```

---

ci_hi *Confidence Interval Generic*

Description

Gets the upper 97.5% quantile of a numeric vector.

Usage

```
ci_hi(.x, .na_rm = FALSE)
```

Arguments

.x	A vector of numeric values
.na_rm	A Boolean, defaults to FALSE. Passed to the quantile function.

Details

Gets the upper 97.5% quantile of a numeric vector.

Value

A numeric value.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Statistic: ci_lo(), tidy_kurtosis_vec(), tidy_range_statistic(), tidy_skewness_vec(), tidy_stat_tbl()

Examples

```
x <- mtcars$mpg
ci_hi(x)
```

`ci_lo` *Confidence Interval Generic*

Description

Gets the lower 2.5% quantile of a numeric vector.

Usage

```
ci_lo(.x, .na_rm = FALSE)
```

Arguments

`.x`	A vector of numeric values
`.na_rm`	A Boolean, defaults to FALSE. Passed to the quantile function.

Details

Gets the lower 2.5% quantile of a numeric vector.

Value

A numeric value.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Statistic: `ci_hi()`, `tidy_kurtosis_vec()`, `tidy_range_statistic()`, `tidy_skewness_vec()`, `tidy_stat_tbl()`

Examples

```
x <- mtcars$mpg
ci_lo(x)
```

`ckurtosis` *Cumulative Kurtosis*

Description

A function to return the cumulative kurtosis of a vector.

Usage

```
ckurtosis(.x)
```

Arguments

`.x`	A numeric vector

Details

A function to return the cumulative kurtosis of a vector.

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: `bootstrap_p_vec()`, `bootstrap_q_vec()`, `cgmean()`, `chmean()`, `cmean()`, `cmedian()`, `csd()`, `cskewness()`, `cvar()`, `tidy_kurtosis_vec()`, `tidy_scale_zero_one_vec()`, `tidy_skewness_vec()`

Examples

```
x <- mtcars$mpg

ckurtosis(x)
```

cmean *Cumulative Mean*

Description

A function to return the cumulative mean of a vector.

Usage

```
cmean(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative mean of a vector. It uses `dplyr::cummean()` as the basis of the function.

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: `bootstrap_p_vec()`, `bootstrap_q_vec()`, `cgmean()`, `chmean()`, `ckurtosis()`, `cmedian()`, `csd()`, `cskewness()`, `cvar()`, `tidy_kurtosis_vec()`, `tidy_scale_zero_one_vec()`, `tidy_skewness_vec()`

Examples

```
x <- mtcars$mpg

cmean(x)
```

cmedian *Cumulative Median*

Description

A function to return the cumulative median of a vector.

Usage

```
cmedian(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative median of a vector.

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), chmean(), ckurtosis(), cmean(), csd(), cskewness(), cvar(), tidy_kurtosis_vec(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
x <- mtcars$mpg

cmedian(x)
```

color_blind *Provide Colorblind Compliant Colors*

Description

8 Hex RGB color definitions suitable for charts for colorblind people.

Usage

```
color_blind()
```

convert_to_ts *Convert Data to Time Series Format*

Description

This function converts data in a data frame or tibble into a time series format. It is designed to work with data generated from `tidy_` distribution functions. The function can return time series data, pivot it into long format, or both.

Usage

```
convert_to_ts(.data, .return_ts = TRUE, .pivot_longer = FALSE)
```

Arguments

`.data`	A data frame or tibble to be converted into a time series format.
`.return_ts`	A logical value indicating whether to return the time series data. Default is TRUE.
`.pivot_longer`	A logical value indicating whether to pivot the data into long format. Default is FALSE.

Details

The function takes a data frame or tibble as input and processes it based on the specified options. It performs the following actions:

1. Checks if the input is a data frame or tibble; otherwise, it raises an error.
2. Checks if the data comes from a `tidy_` distribution function; otherwise, it raises an error.
3. Converts the data into a time series format, grouping it by "sim_number" and transforming the "y" column into a time series.
4. Returns the result based on the chosen options:
   - If `ret_ts` is set to TRUE, it returns the time series data.
   - If `pivot_longer` is set to TRUE, it pivots the data into long format.
   - If both options are set to FALSE, it returns the data as a tibble.

Value

The function returns the processed data based on the chosen options:

- If `ret_ts` is set to TRUE, it returns time series data.
- If `pivot_longer` is set to TRUE, it returns the data in long format.
- If both options are set to FALSE, it returns the data as a tibble.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Convert data to time series format without returning time series data
x <- tidy_normal()
result <- convert_to_ts(x, FALSE)
head(result)

# Example 2: Convert data to time series format and pivot it into long format
x <- tidy_normal()
result <- convert_to_ts(x, FALSE, TRUE)
head(result)

# Example 3: Convert data to time series format and return the time series data
x <- tidy_normal()
result <- convert_to_ts(x)
head(result)
```

---

csd *Cumulative Standard Deviation*

Description

A function to return the cumulative standard deviation of a vector.

Usage

```
csd(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative standard deviation of a vector.

Value

A numeric vector. Note: The first entry will always be NaN.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), chmean(), ckurtosis(), cmean(), cmedian(), cskewness(), cvar(), tidy_kurtosis_vec(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
x <- mtcars$mpg

csd(x)
```

---

cskewness *Cumulative Skewness*

Description

A function to return the cumulative skewness of a vector.

Usage

```
cskewness(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative skewness of a vector.

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), chmean(), ckurtosis(), cmean(), cmedian(), csd(), cvar(), tidy_kurtosis_vec(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
x <- mtcars$mpg

cskewness(x)
```

---

cvar *Cumulative Variance*

Description

A function to return the cumulative variance of a vector.

Usage

```
cvar(.x)
```

Arguments

.x	A numeric vector

Details

A function to return the cumulative variance of a vector. exp(cummean(log(.x)))

Value

A numeric vector. Note: The first entry will always be NaN.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), chmean(), ckurtosis(), cmean(), cmedian(), csd(), cskewness(), tidy_kurtosis_vec(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
x <- mtcars$mpg

cvar(x)
```

---

dist_type_extractor *Extract Distribution Type from Tidy Distribution Object*

Description

Get the distribution name in title case from the tidy_ distribution function.

Usage

```
dist_type_extractor(.x)
```

Arguments

.x	The attribute list passed from a tidy_ distribution function.

Details

This will extract the distribution type from a tidy_ distribution function output using the attributes of that object. You must pass the attribute directly to the function. It is meant really to be used internally.

You should be passing if using manually the $tibble_type attribute.

Value

A character string

Author(s)

Steven P. Sanderson II,

Examples

```
tn <- tidy_normal()
atb <- attributes(tn)
dist_type_extractor(atb$tibble_type)
```

`quantile_normalize` *Perform quantile normalization on a numeric matrix/data.frame*

Description

This function will perform quantile normalization on two or more distributions of equal length. Quantile normalization is a technique used to make the distribution of values across different samples more similar. It ensures that the distributions of values for each sample have the same quantiles. This function takes a numeric matrix as input and returns a quantile-normalized matrix.

Usage

```
quantile_normalize(.data, .return_tibble = FALSE)
```

Arguments

`.data`	A numeric matrix where each column represents a sample.
`.return_tibble`	A logical value that determines if the output should be a tibble. Default is 'FALSE'.

Details

This function performs quantile normalization on a numeric matrix by following these steps:

1. Sort each column of the input matrix.
2. Calculate the mean of each row across the sorted columns.
3. Replace each column's sorted values with the row means.
4. Unsort the columns to their original order.

Value

A list object that has the following:

1. A numeric matrix that has been quantile normalized.
2. The row means of the quantile normalized matrix.
3. The sorted data
4. The ranked indices

Author(s)

Steven P. Sanderson II, MPH

See Also

rowMeans: Calculate row means.

apply: Apply a function over the margins of an array.

order: Order the elements of a vector.

Other Utility: check_duplicate_rows(), convert_to_ts(), tidy_mcmc_sampling(), util_beta_aic(), util_binomial_aic(), util_cauchy_aic(), util_chisq_aic(), util_exponential_aic(), util_gamma_aic(), util_geometric_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_pareto_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
# Create a sample numeric matrix
data <- matrix(rnorm(20), ncol = 4)

# Perform quantile normalization
normalized_data <- quantile_normalize(data)
normalized_data

as.data.frame(normalized_data$normalized_data) |>
  sapply(function(x) quantile(x, probs = seq(0, 1, 1 / 4)))

quantile_normalize(
data.frame(sample1 = rnorm(30),
           sample2 = rnorm(30)),
           .return_tibble = TRUE)
```

---

td_scale_color_colorblind

Provide Colorblind Compliant Colors

Description

Provide Colorblind Compliant Colors

Usage

```
td_scale_color_colorblind(..., theme = "td")
```

Arguments

...	Data passed to the function
theme	This defaults to td and that is the only allowed value

td_scale_fill_colorblind *Provide Colorblind Compliant Colors*

Description

Provide Colorblind Compliant Colors

Usage

```
td_scale_fill_colorblind(..., theme = "td")
```

Arguments

...	Data passed to the function
theme	This defaults to td and that is the only allowed value

tidy_autoplot *Automatic Plot of Density Data*

Description

This is an auto plotting function that will take in a tidy_ distribution function and a few arguments, one being the plot type, which is a quoted string of one of the following:

- density
- quantile
- probablity
- qq
- mcmc

If the number of simulations exceeds 9 then the legend will not print. The plot subtitle is put together by the attributes of the table passed to the function.

Usage

```
tidy_autoplot(
  .data,
  .plot_type = "density",
  .line_size = 0.5,
  .geom_point = FALSE,
  .point_size = 1,
  .geom_rug = FALSE,
  .geom_smooth = FALSE,
  .geom_jitter = FALSE,
  .interactive = FALSE
)
```

Arguments

.data	The data passed in from a tidy_distribution function like tidy_normal()
.plot_type	This is a quoted string like 'density'
.line_size	The size param ggplot
.geom_point	A Boolean value of TREU/FALSE, FALSE is the default. TRUE will return a plot with ggplot2::ggeom_point()
.point_size	The point size param for ggplot
.geom_rug	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of ggplot2::geom_rug()
.geom_smooth	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of ggplot2::geom_smooth() The aes parameter of group is set to FALSE. This ensures a single smoothing band returned with SE also set to FALSE. Color is set to 'black' and linetype is 'dashed'.
.geom_jitter	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of ggplot2::geom_jitter()
.interactive	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return an interactive plotly plot.

Details

This function will spit out one of the following plots:

- density
- quantile
- probability
- qq
- mcmc

Value

A ggplot or a plotly plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Autoplot: bootstrap_stat_plot(), tidy_combined_autoplot(), tidy_four_autoplot(), tidy_multi_dist_autoplot(), tidy_random_walk_autoplot()

Examples

```
tidy_normal(.num_sims = 5) |>
  tidy_autoplot()

tidy_normal(.num_sims = 20) |>
  tidy_autoplot(.plot_type = "qq")
```

`tidy_bernoulli` *Tidy Randomly Generated Bernoulli Distribution Tibble*

Description

This function will generate `n` random points from a Bernoulli distribution with a user provided, `.prob`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_bernoulli(.n = 50, .prob = 0.1, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

`.n`	The number of randomly generated points you want.
`.prob`	The probability of success/failure.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the `rbinom()`, and its underlying `p`, `d`, and `q` functions. The *Bernoulli* distribution is a special case of the *Binomial* distribution with `size = 1` hence this is why the `binom` functions are used and set to size = 1.

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Bernoulli_distribution

Other Discrete Distribution: tidy_binomial(), tidy_hypergeometric(), tidy_negative_binomial(), tidy_poisson(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_binomial(), tidy_zero_truncated_poisson()

Other Bernoulli: util_bernoulli_param_estimate(), util_bernoulli_stats_tbl()

Examples

```
tidy_bernoulli()
```

---

tidy_beta *Tidy Randomly Generated Beta Distribution Tibble*

Description

This function will generate n random points from a beta distribution with a user provided, .shape1, .shape2, .ncp or non-centrality parameter, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_beta(
  .n = 50,
  .shape1 = 1,
  .shape2 = 1,
  .ncp = 0,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape1	A non-negative parameter of the Beta distribution.
.shape2	A non-negative parameter of the Beta distribution.
.ncp	The non-centrality parameter of the Beta distribution.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying stats::rbeta(), and its underlying p, d, and q functions. For more information please see stats::rbeta()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://statisticsglobe.com/beta-distribution-in-r-dbeta-pbeta-qbeta-rbeta

https://en.wikipedia.org/wiki/Beta_distribution

Other Continuous Distribution: tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Beta: tidy_generalized_beta(), util_beta_param_estimate(), util_beta_stats_tbl()

Examples

```
tidy_beta()
```

`tidy_binomial` *Tidy Randomly Generated Binomial Distribution Tibble*

Description

This function will generate `n` random points from a binomial distribution with a user provided, `.size`, `.prob`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_binomial(
  .n = 50,
  .size = 0,
  .prob = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.size`	Number of trials, zero or more.
`.prob`	Probability of success on each trial.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rbinom()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rbinom()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda366i.htm

Other Discrete Distribution: tidy_bernoulli(), tidy_hypergeometric(), tidy_negative_binomial(), tidy_poisson(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_binomial(), tidy_zero_truncated_poisson()

Other Binomial: tidy_negative_binomial(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_b
util_binomial_param_estimate(), util_binomial_stats_tbl(), util_negative_binomial_param_estimate()

Examples

```
tidy_binomial()
```

---

tidy_bootstrap *Bootstrap Empirical Data*

Description

Takes an input vector of numeric data and produces a bootstrapped nested tibble by simulation number.

Usage

```
tidy_bootstrap(
  .x,
  .num_sims = 2000,
  .proportion = 0.8,
  .distribution_type = "continuous"
)
```

Arguments

.x	The vector of data being passed to the function. Must be a numeric vector.
.num_sims	The default is 2000, can be set to anything desired. A warning will pass to the console if the value is less than 2000.
.proportion	How much of the original data do you want to pass through to the sampling function. The default is 0.80 (80%)
.distribution_type	This can either be 'continuous' or 'discrete'

Details

This function will take in a numeric input vector and produce a tibble of bootstrapped values in a list. The table that is output will have two columns: `sim_number` and `bootstrap_samples`

The `sim_number` corresponds to how many times you want the data to be resampled, and the `bootstrap_samples` column contains a list of the boostrapped resampled data.

Value

A nested tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bootstrap: `bootstrap_density_augment()`, `bootstrap_p_augment()`, `bootstrap_p_vec()`, `bootstrap_q_augment()`, `bootstrap_q_vec()`, `bootstrap_stat_plot()`, `bootstrap_unnest_tbl()`

Examples

```
x <- mtcars$mpg
tidy_bootstrap(x)
```

---

tidy_burr *Tidy Randomly Generated Burr Distribution Tibble*

Description

This function will generate `n` random points from a Burr distribution with a user provided, `.shape1`, `.shape2`, `.scale`, `.rate`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_burr(
  .n = 50,
  .shape1 = 1,
  .shape2 = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape1	Must be strictly positive.
.shape2	Must be strictly positive.
.rate	An alternative way to specify the .scale.
.scale	Must be strictly positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying actuar::rburr(), and its underlying p, d, and q functions. For more information please see actuar::rburr()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: tidy_beta(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Burr: tidy_inverse_burr(), util_burr_param_estimate(), util_burr_stats_tbl()

Examples

```
tidy_burr()
```

`tidy_cauchy` *Tidy Randomly Generated Cauchy Distribution Tibble*

Description

This function will generate `n` random points from a cauchy distribution with a user provided, `.location`, `.scale`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_cauchy(
  .n = 50,
  .location = 0,
  .scale = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.location`	The location parameter.
`.scale`	The scale parameter, must be greater than or equal to 0.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rcauchy()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rcauchy()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3663.htm

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Cauchy: `util_cauchy_param_estimate()`, `util_cauchy_stats_tbl()`

Examples

```
tidy_cauchy()
```

`tidy_chisquare` *Tidy Randomly Generated Chisquare (Non-Central) Distribution Tibble*

Description

This function will generate `n` random points from a chisquare distribution with a user provided, `.df`, `.ncp`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_chisquare(
  .n = 50,
  .df = 1,
  .ncp = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.df	Degrees of freedom (non-negative but can be non-integer)
.ncp	Non-centrality parameter, must be non-negative.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rchisq()`, and its underlying p, d, and q functions. For more information please see `stats::rchisq()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3666.htm

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Chisquare: `util_chisquare_param_estimate()`, `util_chisquare_stats_tbl()`

Examples

```
tidy_chisquare()
```

tidy_combined_autoplot *Automatic Plot of Combined Multi Dist Data*

Description

This is an auto plotting function that will take in a `tidy_` distribution function and a few arguments, one being the plot type, which is a quoted string of one of the following:

- `density`
- `quantile`
- `probablity`
- `qq`
- `mcmc`

If the number of simulations exceeds 9 then the legend will not print. The plot subtitle is put together by the attributes of the table passed to the function.

Usage

```
tidy_combined_autoplot(
  .data,
  .plot_type = "density",
  .line_size = 0.5,
  .geom_point = FALSE,
  .point_size = 1,
  .geom_rug = FALSE,
  .geom_smooth = FALSE,
  .geom_jitter = FALSE,
  .interactive = FALSE
)
```

Arguments

`.data`	The data passed in from a the function `tidy_multi_dist()`
`.plot_type`	This is a quoted string like 'density'
`.line_size`	The size param ggplot
`.geom_point`	A Boolean value of TREU/FALSE, FALSE is the default. TRUE will return a plot with `ggplot2::ggeom_point()`
`.point_size`	The point size param for ggplot
`.geom_rug`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_rug()`
`.geom_smooth`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_smooth()` The `aes` parameter of group is set to FALSE. This ensures a single smoothing band returned with SE also set to FALSE. Color is set to 'black' and `linetype` is 'dashed'.

`.geom_jitter`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_jitter()`
`.interactive`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return an interactive `plotly` plot.

Details

This function will spit out one of the following plots:

- `density`
- `quantile`
- `probability`
- `qq`
- `mcmc`

Value

A ggplot or a plotly plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Autoplot: `bootstrap_stat_plot()`, `tidy_autoplot()`, `tidy_four_autoplot()`, `tidy_multi_dist_autoplot()`, `tidy_random_walk_autoplot()`

Examples

```
combined_tbl <- tidy_combine_distributions(
  tidy_normal(),
  tidy_gamma(),
  tidy_beta()
)

combined_tbl

combined_tbl |>
  tidy_combined_autoplot()

combined_tbl |>
  tidy_combined_autoplot(.plot_type = "qq")
```

tidy_combine_distributions *Combine Multiple Tidy Distributions of Different Types*

Description

This allows a user to specify any n number of tidy_ distributions that can be combined into a single tibble. This is the preferred method for combining multiple distributions of different types, for example a Gaussian distribution and a Beta distribution.

This generates a single tibble with an added column of dist_type that will give the distribution family name and its associated parameters.

Usage

```
tidy_combine_distributions(...)
```

Arguments

...	The ... is where you can place your different distributions

Details

Allows a user to generate a tibble of different tidy_ distributions

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Multiple Distribution: tidy_multi_single_dist()

Examples

```
tn <- tidy_normal()
tb <- tidy_beta()
tc <- tidy_cauchy()

tidy_combine_distributions(tn, tb, tc)

## OR

tidy_combine_distributions(
```

```
  tidy_normal(),
  tidy_beta(),
  tidy_cauchy(),
  tidy_logistic()
)
```

`tidy_distribution_comparison` *Compare Empirical Data to Distributions*

Description

Compare some empirical data set against different distributions to help find the distribution that could be the best fit.

Usage

```
tidy_distribution_comparison(
  .x,
  .distribution_type = "continuous",
  .round_to_place = 3
)
```

Arguments

`.x`	The data set being passed to the function
`.distribution_type`	What kind of data is it, can be one of `continuous` or `discrete`
`.round_to_place`	How many decimal places should the parameter estimates be rounded off to for distibution construction. The default is 3

Details

The purpose of this function is to take some data set provided and to try to find a distribution that may fit the best. A parameter of `.distribution_type` must be set to either `continuous` or `discrete` in order for this the function to try the appropriate types of distributions.

The following distributions are used:

Continuous:

- tidy_beta
- tidy_cauchy
- tidy_chisquare
- tidy_exponential
- tidy_gamma

- tidy_logistic
- tidy_lognormal
- tidy_normal
- tidy_pareto
- tidy_uniform
- tidy_weibull

Discrete:

- tidy_binomial
- tidy_geometric
- tidy_hypergeometric
- tidy_poisson

The function itself returns a list output of tibbles. Here are the tibbles that are returned:

- comparison_tbl
- deviance_tbl
- total_deviance_tbl
- aic_tbl
- kolmogorov_smirnov_tbl
- multi_metric_tbl

The `comparison_tbl` is a long `tibble` that lists the values of the `density` function against the given data.

The `deviance_tbl` and the `total_deviance_tbl` just give the simple difference from the actual density to the estimated density for the given estimated distribution.

The `aic_tbl` will provide the `AIC` for liklehood of the distribution.

The `kolmogorov_smirnov_tbl` for now provides a `two.sided` estimate of the `ks.test` of the estimated density against the empirical.

The `multi_metric_tbl` will summarise all of these metrics into a single tibble.

Value

An invisible list object. A tibble is printed.

Author(s)

Steven P. Sanderson II, MPH

Examples

```
xc <- mtcars$mpg
output_c <- tidy_distribution_comparison(xc, "continuous")

xd <- trunc(xc)
output_d <- tidy_distribution_comparison(xd, "discrete")

output_c
output_d
```

`tidy_distribution_summary_tbl`

Tidy Distribution Summary Statistics Tibble

Description

This function returns a summary statistics tibble. It will use the y column from the `tidy_` distribution function.

Usage

```
tidy_distribution_summary_tbl(.data, ...)
```

Arguments

`.data`	The data that is going to be passed from a a `tidy_` distribution function.
`...`	This is the grouping variable that gets passed to `dplyr::group_by()` and `dplyr::select()`.

Details

This function takes in a `tidy_` distribution table and will return a tibble of the following information:

- `sim_number`
- `mean_val`
- `median_val`
- `std_val`
- `min_val`
- `max_val`
- `skewness`
- `kurtosis`
- `range`
- `iqr`
- `variance`
- `ci_hi`
- `ci_lo`

The kurtosis and skewness come from the package `healthyR.ai`

Value

A summary stats tibble

Author(s)

Steven P. Sanderson II, MPH

Examples

```
library(dplyr)

tn <- tidy_normal(.num_sims = 5)
tb <- tidy_beta(.num_sims = 5)

tidy_distribution_summary_tbl(tn)
tidy_distribution_summary_tbl(tn, sim_number)

data_tbl <- tidy_combine_distributions(tn, tb)

tidy_distribution_summary_tbl(data_tbl)
tidy_distribution_summary_tbl(data_tbl, dist_type)
```

---

tidy_empirical *Tidy Empirical*

Description

This function takes in a single argument of .x a vector and will return a tibble of information similar to the `tidy_` distribution functions. The `y` column is set equal to `dy` from the density function.

Usage

```
tidy_empirical(.x, .num_sims = 1, .distribution_type = "continuous")
```

Arguments

`.x`	A vector of numbers
`.num_sims`	How many simulations should be run, defaults to 1.
`.distribution_type`	A string of either "continuous" or "discrete". The function will default to "continuous"

Details

This function takes in a single argument of .x a vector

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

Examples

```
x <- mtcars$mpg
tidy_empirical(.x = x, .distribution_type = "continuous")
tidy_empirical(.x = x, .num_sims = 10, .distribution_type = "continuous")
```

---

tidy_exponential *Tidy Randomly Generated Exponential Distribution Tibble*

Description

This function will generate n random points from a exponential distribution with a user provided, .rate, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_exponential(.n = 50, .rate = 1, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

.n	The number of randomly generated points you want.
.rate	A vector of rates
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rexp()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rexp()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3667.htm

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Exponential: `tidy_inverse_exponential()`, `util_exponential_param_estimate()`, `util_exponential_stats_`

Examples

```
tidy_exponential()
```

---

tidy_f *Tidy Randomly Generated F Distribution Tibble*

Description

This function will generate `n` random points from a rf distribution with a user provided, `df1`,`df2`, and `ncp`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_f(
  .n = 50,
  .df1 = 1,
  .df2 = 1,
  .ncp = 0,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.df1	Degrees of freedom, Inf is allowed.
.df2	Degrees of freedom, Inf is allowed.
.ncp	Non-centrality parameter.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rf()`, and its underlying p, d, and q functions. For more information please see `stats::rf()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3665.htm

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other F Distribution: `util_f_stats_tbl()`

Examples

```
tidy_f()
```

`tidy_four_autoplot` *Automatic Plot of Density Data*

Description

This is an auto plotting function that will take in a `tidy_` distribution function and a few arguments, one being the plot type, which is a quoted string of one of the following:

- `density`
- `quantile`
- `probablity`
- `qq`
- `mcmc`

If the number of simulations exceeds 9 then the legend will not print. The plot subtitle is put together by the attributes of the table passed to the function.

Usage

```
tidy_four_autoplot(
  .data,
  .line_size = 0.5,
  .geom_point = FALSE,
  .point_size = 1,
  .geom_rug = FALSE,
  .geom_smooth = FALSE,
  .geom_jitter = FALSE,
  .interactive = FALSE
)
```

Arguments

`.data`	The data passed in from a tidy_`distribution` function like `tidy_normal()`
`.line_size`	The size param ggplot
`.geom_point`	A Boolean value of TREU/FALSE, FALSE is the default. TRUE will return a plot with `ggplot2::ggeom_point()`
`.point_size`	The point size param for ggplot
`.geom_rug`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_rug()`
`.geom_smooth`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_smooth()` The `aes` parameter of group is set to FALSE. This ensures a single smoothing band returned with SE also set to FALSE. Color is set to 'black' and `linetype` is 'dashed'.
`.geom_jitter`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_jitter()`
`.interactive`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return an interactive `plotly` plot.

Details

This function will spit out one of the following plots:

- density
- quantile
- probability
- qq
- mcmc

Value

A ggplot or a plotly plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Autoplot: bootstrap_stat_plot(), tidy_autoplot(), tidy_combined_autoplot(), tidy_multi_dist_autoplo
tidy_random_walk_autoplot()

Examples

```
tidy_normal(.num_sims = 5) |>
  tidy_four_autoplot()
```

---

tidy_gamma *Tidy Randomly Generated Gamma Distribution Tibble*

Description

This function will generate n random points from a gamma distribution with a user provided, .shape, .scale, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_gamma(
  .n = 50,
  .shape = 1,
  .scale = 0.3,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	This is strictly 0 to infinity.
.scale	The standard deviation of the randomly generated data. This is strictly from 0 to infinity.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rgamma()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rgamma()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.statology.org/fit-gamma-distribution-to-dataset-in-r/

https://en.wikipedia.org/wiki/Gamma_distribution

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Gamma: `tidy_inverse_gamma()`, `util_gamma_param_estimate()`, `util_gamma_stats_tbl()`

Examples

```
tidy_gamma()
```

tidy_generalized_beta *Tidy Randomly Generated Generalized Beta Distribution Tibble*

Description

This function will generate n random points from a generalized beta distribution with a user provided, .shape1, .shape2, .shape3, .rate, and/or .sclae, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_generalized_beta(
  .n = 50,
  .shape1 = 1,
  .shape2 = 1,
  .shape3 = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape1	A non-negative parameter of the Beta distribution.
.shape2	A non-negative parameter of the Beta distribution.
.shape3	A non-negative parameter of the Beta distribution.
.rate	An alternative way to specify the .scale parameter.
.scale	Must be strictly positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rbeta()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rbeta()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://statisticsglobe.com/beta-distribution-in-r-dbeta-pbeta-qbeta-rbeta

https://en.wikipedia.org/wiki/Beta_distribution

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Beta: `tidy_beta()`, `util_beta_param_estimate()`, `util_beta_stats_tbl()`

Examples

```
tidy_generalized_beta()
```

`tidy_generalized_pareto`

Tidy Randomly Generated Generalized Pareto Distribution Tibble

---

Description

This function will generate `n` random points from a generalized Pareto distribution with a user provided, `.shape1`, `.shape2`, `.rate` or `.scale` and number of #' random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.

- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_generalized_pareto(
  .n = 50,
  .shape1 = 1,
  .shape2 = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape1	Must be positive.
.shape2	Must be positive.
.rate	An alternative way to specify the .scale argument
.scale	Must be positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying actuar::rgenpareto(), and its underlying p, d, and q functions. For more information please see actuar::rgenpareto()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Pareto: tidy_inverse_pareto(), tidy_pareto(), tidy_pareto1(), util_pareto_param_estimate(), util_pareto_stats_tbl()

Examples

```
tidy_generalized_pareto()
```

---

tidy_geometric *Tidy Randomly Generated Geometric Distribution Tibble*

Description

This function will generate n random points from a geometric distribution with a user provided, .prob, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_geometric(.n = 50, .prob = 1, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

.n	The number of randomly generated points you want.
.prob	A probability of success in each trial 0 < prob <= 1.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rgeom()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rgeom()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Geometric_distribution

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Geometric: `tidy_zero_truncated_geometric()`, `util_geometric_param_estimate()`, `util_geometric_stats_tbl()`

Examples

```
tidy_geometric()
```

`tidy_hypergeometric` *Tidy Randomly Generated Hypergeometric Distribution Tibble*

Description

This function will generate n random points from a hypergeometric distribution with a user provided, m,nn, and k, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_hypergeometric(
  .n = 50,
  .m = 0,
  .nn = 0,
  .k = 0,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.m	The number of white balls in the urn
.nn	The number of black balls in the urn
.k	The number of balls drawn fro the urn.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying stats::rhyper(), and its underlying p, d, and q functions. For more information please see stats::rhyper()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Hypergeometric_distribution

Other Discrete Distribution: tidy_bernoulli(), tidy_binomial(), tidy_negative_binomial(), tidy_poisson(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_binomial(), tidy_zero_truncated_poisson()

Other Hypergeometric: util_hypergeometric_param_estimate(), util_hypergeometric_stats_tbl()

Examples

```
tidy_hypergeometric()
```

---

tidy_inverse_burr *Tidy Randomly Generated Inverse Burr Distribution Tibble*

Description

This function will generate n random points from an Inverse Burr distribution with a user provided, .shape1, .shape2, .scale, .rate, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_inverse_burr(
  .n = 50,
  .shape1 = 1,
  .shape2 = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape1	Must be strictly positive.
.shape2	Must be strictly positive.
.rate	An alternative way to specify the .scale.
.scale	Must be strictly positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying actuar::rinvburr(), and its underlying p, d, and q functions. For more information please see actuar::rinvburr()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Burr: tidy_burr(), util_burr_param_estimate(), util_burr_stats_tbl()

Other Inverse Distribution: tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull()

Examples

```
tidy_inverse_burr()
```

`tidy_inverse_exponential` *Tidy Randomly Generated Inverse Exponential Distribution Tibble*

Description

This function will generate `n` random points from an inverse exponential distribution with a user provided, `.rate` or `.scale` and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_inverse_exponential(
  .n = 50,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.rate`	An alternative way to specify the `.scale`
`.scale`	Must be strictly positive.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rinvexp()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rinvexp()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Exponential: tidy_exponential(), util_exponential_param_estimate(), util_exponential_stats_tbl()

Other Inverse Distribution: tidy_inverse_burr(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull()

Examples

```
tidy_inverse_exponential()
```

---

tidy_inverse_gamma *Tidy Randomly Generated Inverse Gamma Distribution Tibble*

Description

This function will generate n random points from an inverse gamma distribution with a user provided, .shape, .rate, .scale, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_inverse_gamma(
  .n = 50,
  .shape = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	Must be strictly positive.
.rate	An alternative way to specify the .scale
.scale	Must be strictly positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying actuar::rinvgamma(), and its underlying p, d, and q functions. For more information please see actuar::rinvgamma()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Gamma: tidy_gamma(), util_gamma_param_estimate(), util_gamma_stats_tbl()

Other Inverse Distribution: tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull()

Examples

```
tidy_inverse_gamma()
```

`tidy_inverse_normal` *Tidy Randomly Generated Inverse Gaussian Distribution Tibble*

Description

This function will generate `n` random points from an Inverse Gaussian distribution with a user provided, `.mean`, `.shape`, `.dispersion`The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_inverse_normal(
  .n = 50,
  .mean = 1,
  .shape = 1,
  .dispersion = 1/.shape,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.mean`	Must be strictly positive.
`.shape`	Must be strictly positive.
`.dispersion`	An alternative way to specify the `.shape`.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rinvgauss()`. For more information please see `rinvgauss()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Gaussian: `tidy_normal()`, `util_normal_param_estimate()`, `util_normal_stats_tbl()`

Other Inverse Distribution: `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`

Examples

```
tidy_inverse_normal()
```

`tidy_inverse_pareto` *Tidy Randomly Generated Inverse Pareto Distribution Tibble*

Description

This function will generate `n` random points from an inverse pareto distribution with a user provided, `.shape`, `.scale`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_inverse_pareto(
  .n = 50,
  .shape = 1,
  .scale = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	Must be positive.
.scale	Must be positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rinvpareto()`, and its underlying p, d, and q functions. For more information please see `actuar::rinvpareto()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Pareto: `tidy_generalized_pareto()`, `tidy_pareto()`, `tidy_pareto1()`, `util_pareto_param_estimate()`, `util_pareto_stats_tbl()`

Other Inverse Distribution: `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_weibull()`

Examples

```
tidy_inverse_pareto()
```

`tidy_inverse_weibull` *Tidy Randomly Generated Inverse Weibull Distribution Tibble*

Description

This function will generate `n` random points from a weibull distribution with a user provided, `.shape`, `.scale`, `.rate`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_inverse_weibull(
  .n = 50,
  .shape = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.shape`	Must be strictly positive.
`.rate`	An alternative way to specify the `.scale`.
`.scale`	Must be strictly positive.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rinvweibull()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rinvweibull()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Weibull: `tidy_weibull()`, `util_weibull_param_estimate()`, `util_weibull_stats_tbl()`

Other Inverse Distribution: `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`

Examples

```
tidy_inverse_weibull()
```

---

tidy_kurtosis_vec *Compute Kurtosis of a Vector*

Description

This function takes in a vector as it's input and will return the kurtosis of that vector. The length of this vector must be at least four numbers. The kurtosis explains the sharpness of the peak of a distribution of data.
`((1/n) * sum(x - mu})^4) / (((()1/n) * sum(x - mu)^2)^2`

Usage

```
tidy_kurtosis_vec(.x)
```

Arguments

`.x`	A numeric vector of length four or more.

Details

A function to return the kurtosis of a vector.

Value

The kurtosis of a vector

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Kurtosis

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), chmean(), ckurtosis(), cmean(), cmedian(), csd(), cskewness(), cvar(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Other Statistic: ci_hi(), ci_lo(), tidy_range_statistic(), tidy_skewness_vec(), tidy_stat_tbl()

Other Vector Function: bootstrap_p_vec(), bootstrap_q_vec(), cgmean(), chmean(), ckurtosis(), cmean(), cmedian(), csd(), cskewness(), cvar(), tidy_scale_zero_one_vec(), tidy_skewness_vec()

Examples

```
tidy_kurtosis_vec(rnorm(100, 3, 2))
```

---

tidy_logistic *Tidy Randomly Generated Logistic Distribution Tibble*

Description

This function will generate n random points from a logistic distribution with a user provided, .location, .scale, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresonds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_logistic(
  .n = 50,
  .location = 0,
  .scale = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.location	The location parameter
.scale	The scale parameter
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rlogis()`, and its underlying p, d, and q functions. For more information please see `stats::rlogis()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Logistic_distribution

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Logistic: `tidy_paralogistic()`, `util_logistic_param_estimate()`, `util_logistic_stats_tbl()`

Examples

```
tidy_logistic()
```

tidy_lognormal *Tidy Randomly Generated Lognormal Distribution Tibble*

Description

This function will generate n random points from a lognormal distribution with a user provided, .meanlog, .sdlog, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_lognormal(
  .n = 50,
  .meanlog = 0,
  .sdlog = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.meanlog	Mean of the distribution on the log scale with default 0
.sdlog	Standard deviation of the distribution on the log scale with default 1
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying stats::rlnorm(), and its underlying p, d, and q functions. For more information please see stats::rlnorm()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3669.htm

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Lognormal: util_lognormal_param_estimate(), util_lognormal_stats_tbl()

Examples

```
tidy_lognormal()
```

---

tidy_mcmc_sampling *Tidy MCMC Sampling*

Description

This function performs Markov Chain Monte Carlo (MCMC) sampling on the input data and returns tidy data and a plot representing the results.

Usage

```
tidy_mcmc_sampling(.x, .fns = "mean", .cum_fns = "cmean", .num_sims = 2000)
```

Arguments

.x	The data vector for MCMC sampling.
.fns	The function(s) to apply to each MCMC sample. Default is "mean".
.cum_fns	The function(s) to apply to the cumulative MCMC samples. Default is "cmean".
.num_sims	The number of simulations. Default is 2000.

Details

Perform MCMC sampling and return tidy data and a plot.

The function takes a data vector as input and performs MCMC sampling with the specified number of simulations. It applies user-defined functions to each MCMC sample and to the cumulative MCMC samples. The resulting data is formatted in a tidy format, suitable for further analysis. Additionally, a plot is generated to visualize the MCMC samples and cumulative statistics.

Value

A list containing tidy data and a plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Generate MCMC samples
set.seed(123)
data <- rnorm(100)
result <- tidy_mcmc_sampling(data, "median", "cmedian", 500)
result
```

`tidy_mixture_density` *Tidy Mixture Data*

Description

Create mixture model data and resulting density and line plots.

Usage

```
tidy_mixture_density(...)
```

Arguments

`...`	The random data you want to pass. Example rnorm(50,0,1) or something like tidy_normal(.mean = 5, .sd = 1)

Details

This function allows you to make mixture model data. It allows you to produce density data and plots for data that is not strictly of one family or of one single type of distribution with a given set of parameters.

For example this function will allow you to mix say tidy_normal(.mean = 0, .sd = 1) and tidy_normal(.mean = 5, .sd = 1) or you can mix and match distributions.

The output is a list object with three components.

1. Data

- input_data (The random data passed)
- dist_tbl (A tibble of the passed random data)
- density_tbl (A tibble of the x and y data from `stats::density()`)

1. Plots

- line_plot - Plots the dist_tbl
- dens_plot - Plots the density_tbl

1. Input Functions

- input_fns - A list of the functions and their parameters passed to the function itself

Value

A list object

Author(s)

Steven P. Sanderson II, MPH

Examples

```
output <- tidy_mixture_density(rnorm(100, 0, 1), tidy_normal(.mean = 5, .sd = 1))

output$data

output$plots

output$input_fns
```

`tidy_multi_dist_autoplot` *Automatic Plot of Multi Dist Data*

Description

This is an auto plotting function that will take in a `tidy_` distribution function and a few arguments, one being the plot type, which is a quoted string of one of the following:

- `density`
- `quantile`
- `probablity`
- `qq`
- `mcmc`

If the number of simulations exceeds 9 then the legend will not print. The plot subtitle is put together by the attributes of the table passed to the function.

Usage

```
tidy_multi_dist_autoplot(
  .data,
  .plot_type = "density",
  .line_size = 0.5,
  .geom_point = FALSE,
  .point_size = 1,
  .geom_rug = FALSE,
  .geom_smooth = FALSE,
  .geom_jitter = FALSE,
  .interactive = FALSE
)
```

Arguments

`.data`	The data passed in from a the function `tidy_multi_dist()`
`.plot_type`	This is a quoted string like 'density'
`.line_size`	The size param ggplot
`.geom_point`	A Boolean value of TREU/FALSE, FALSE is the default. TRUE will return a plot with `ggplot2::ggeom_point()`
`.point_size`	The point size param for ggplot
`.geom_rug`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_rug()`
`.geom_smooth`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_smooth()` The `aes` parameter of group is set to FALSE. This ensures a single smoothing band returned with SE also set to FALSE. Color is set to 'black' and `linetype` is 'dashed'.

`.geom_jitter`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of `ggplot2::geom_jitter()`
`.interactive`	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return an interactive `plotly` plot.

Details

This function will spit out one of the following plots:

- `density`
- `quantile`
- `probability`
- `qq`
- `mcmc`

Value

A ggplot or a plotly plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Autoplot: `bootstrap_stat_plot()`, `tidy_autoplot()`, `tidy_combined_autoplot()`, `tidy_four_autoplot()`, `tidy_random_walk_autoplot()`

Examples

```
tn <- tidy_multi_single_dist(
  .tidy_dist = "tidy_normal",
  .param_list = list(
    .n = 100,
    .mean = c(-2, 0, 2),
    .sd = 1,
    .num_sims = 5,
    .return_tibble = TRUE
  )
)

tn |>
  tidy_multi_dist_autoplot()

tn |>
  tidy_multi_dist_autoplot(.plot_type = "qq")
```

`tidy_multi_single_dist` *Generate Multiple Tidy Distributions of a single type*

Description

Generate multiple distributions of data from the same `tidy_` distribution function.

Usage

```
tidy_multi_single_dist(.tidy_dist = NULL, .param_list = list())
```

Arguments

`.tidy_dist`	The type of `tidy_` distribution that you want to run. You can only choose one.
`.param_list`	This must be a `list()` object of the parameters that you want to pass through to the TidyDensity `tidy_` distribution function.

Details

Generate multiple distributions of data from the same `tidy_` distribution function. This allows you to simulate multiple distributions of the same family in order to view how shapes change with parameter changes. You can then visualize the differences however you choose.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Multiple Distribution: `tidy_combine_distributions()`

Examples

```
tidy_multi_single_dist(
  .tidy_dist = "tidy_normal",
  .param_list = list(
    .n = 50,
    .mean = c(-1, 0, 1),
    .sd = 1,
    .num_sims = 3,
    .return_tibble = TRUE
  )
```

```
)

tidy_multi_single_dist(
  .tidy_dist = "tidy_normal",
  .param_list = list(
    .n = 50,
    .mean = c(-1, 0, 1),
    .sd = 1,
    .num_sims = 3,
    .return_tibble = FALSE
  )
)
```

`tidy_negative_binomial` *Tidy Randomly Generated Negative Binomial Distribution Tibble*

Description

This function will generate `n` random points from a negative binomial distribution with a user provided, `.size`, `.prob`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_negative_binomial(
  .n = 50,
  .size = 1,
  .prob = 0.1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.size	target for number of successful trials, or dispersion parameter (the shape parameter of the gamma mixing distribution). Must be strictly positive, need not be integer.
.prob	Probability of success on each trial where 0 < .prob <= 1.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rnbinom()`, and its underlying p, d, and q functions. For more information please see `stats::rnbinom()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Discrete Distribution: `tidy_bernoulli()`, `tidy_binomial()`, `tidy_hypergeometric()`, `tidy_poisson()`, `tidy_zero_truncated_binomial()`, `tidy_zero_truncated_negative_binomial()`, `tidy_zero_truncated_poisson()`

Other Binomial: `tidy_binomial()`, `tidy_zero_truncated_binomial()`, `tidy_zero_truncated_negative_binomial()` `util_binomial_param_estimate()`, `util_binomial_stats_tbl()`, `util_negative_binomial_param_estimate()`

Examples

```
tidy_negative_binomial()
```

`tidy_normal` *Tidy Randomly Generated Gaussian Distribution Tibble*

Description

This function will generate `n` random points from a Gaussian distribution with a user provided, `.mean`, `.sd` - standard deviation and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `dnorm`, `pnorm` and `qnorm` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting p_ function of the distribution family.
- `q` The values from the resulting q_ function of the distribution family.

Usage

```
tidy_normal(.n = 50, .mean = 0, .sd = 1, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

`.n`	The number of randomly generated points you want.
`.mean`	The mean of the randomly generated data.
`.sd`	The standard deviation of the randomly generated data.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rnorm()`, `stats::pnorm()`, and `stats::qnorm()` functions to generate data from the given parameters. For more information please see `stats::rnorm()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Gaussian: tidy_inverse_normal(), util_normal_param_estimate(), util_normal_stats_tbl()

Examples

```
tidy_normal()
```

---

tidy_paralogistic *Tidy Randomly Generated Paralogistic Distribution Tibble*

Description

This function will generate n random points from a paralogistic distribution with a user provided, .shape, .rate, .scale and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_paralogistic(
  .n = 50,
  .shape = 1,
  .rate = 1,
  .scale = 1/.rate,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	Must be strictly positive.
.rate	An alternative way to specify the .scale
.scale	Must be strictly positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying actuar::rparalogis(), and its underlying p, d, and q functions. For more information please see actuar::rparalogis()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Logistic_distribution

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Logistic: tidy_logistic(), util_logistic_param_estimate(), util_logistic_stats_tbl()

Examples

```
tidy_paralogistic()
```

tidy_pareto *Tidy Randomly Generated Pareto Distribution Tibble*

Description

This function will generate n random points from a pareto distribution with a user provided, .shape, .scale, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_pareto(
  .n = 50,
  .shape = 10,
  .scale = 0.1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	Must be positive.
.scale	Must be positive.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying actuar::rpareto(), and its underlying p, d, and q functions. For more information please see actuar::rpareto()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto1(), tidy_t(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Pareto: tidy_generalized_pareto(), tidy_inverse_pareto(), tidy_pareto1(), util_pareto_param_estimat
util_pareto_stats_tbl()

Examples

```
tidy_pareto()
```

---

tidy_pareto1 *Tidy Randomly Generated Pareto Single Parameter Distribution Tibble*

Description

This function will generate n random points from a single parameter pareto distribution with a user provided, .shape, .min, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_pareto1(
  .n = 50,
  .shape = 1,
  .min = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	Must be positive.
.min	The lower bound of the support of the distribution.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rpareto1()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rpareto1()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Pareto: `tidy_generalized_pareto()`, `tidy_inverse_pareto()`, `tidy_pareto()`, `util_pareto_param_estimate` `util_pareto_stats_tbl()`

Examples

```
tidy_pareto1()
```

tidy_poisson *Tidy Randomly Generated Poisson Distribution Tibble*

Description

This function will generate n random points from a Poisson distribution with a user provided, .lambda, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_poisson(.n = 50, .lambda = 1, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

.n	The number of randomly generated points you want.
.lambda	A vector of non-negative means.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying stats::rpois(), and its underlying p, d, and q functions. For more information please see stats::rpois()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://r-coder.com/poisson-distribution-r/

https://en.wikipedia.org/wiki/Poisson_distribution

Other Poisson: tidy_zero_truncated_poisson(), util_poisson_param_estimate(), util_poisson_stats_tbl()

Other Discrete Distribution: tidy_bernoulli(), tidy_binomial(), tidy_hypergeometric(), tidy_negative_binomial(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_binomial(), tidy_zero_truncated_poisson()

Examples

```
tidy_poisson()
```

---

tidy_random_walk *Tidy Random Walk*

Description

Takes in the data from a tidy_ distribution function and applies a random walk calculation of either cum_prod or cum_sum to y.

Usage

```
tidy_random_walk(
  .data,
  .initial_value = 0,
  .sample = FALSE,
  .replace = FALSE,
  .value_type = "cum_prod"
)
```

Arguments

.data	The data that is being passed from a tidy_ distribution function.
.initial_value	The default is 0, this can be set to whatever you want.
.sample	This is a boolean value TRUE/FALSE. The default is FALSE. If set to TRUE then the y value from the tidy_ distribution function is sampled.
.replace	This is a boolean value TRUE/FALSE. The default is FALSE. If set to TRUE AND .sample is set to TRUE then the replace parameter of the sample function will be set to TRUE.
.value_type	This can take one of three different values for now. These are the following: • "cum_prod" - This will take the cumprod of y • "cum_sum" - This will take the cumsum of y

Details

Monte Carlo simulations were first formally designed in the 1940's while developing nuclear weapons, and since have been heavily used in various fields to use randomness solve problems that are potentially deterministic in nature. In finance, Monte Carlo simulations can be a useful tool to give a sense of how assets with certain characteristics might behave in the future. While there are more complex and sophisticated financial forecasting methods such as ARIMA (Auto-Regressive Integrated Moving Average) and GARCH (Generalised Auto-Regressive Conditional Heteroskedasticity) which attempt to model not only the randomness but underlying macro factors such as seasonality and volatility clustering, Monte Carlo random walks work surprisingly well in illustrating market volatility as long as the results are not taken too seriously.

Value

An ungrouped tibble.

Author(s)

Steven P. Sanderson II, MPH

Examples

```
tidy_normal(.sd = .1, .num_sims = 25) %>%
  tidy_random_walk()
```

---

tidy_random_walk_autoplot

Automatic Plot of Random Walk Data

Description

This is an auto-plotting function that will take in a `tidy_` distribution function and a few arguments with regard to the output of the visualization.

If the number of simulations exceeds 9 then the legend will not print. The plot subtitle is put together by the attributes of the table passed to the function.

Usage

```
tidy_random_walk_autoplot(
  .data,
  .line_size = 0.5,
  .geom_rug = FALSE,
  .geom_smooth = FALSE,
  .interactive = FALSE
)
```

Arguments

.data	The data passed in from a tidy_distribution function like tidy_normal()
.line_size	The size param ggplot
.geom_rug	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of ggplot2::geom_rug()
.geom_smooth	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return the use of ggplot2::geom_smooth() The aes parameter of group is set to FALSE. This ensures a single smoothing band returned with SE also set to FALSE. Color is set to 'black' and linetype is 'dashed'.
.interactive	A Boolean value of TRUE/FALSE, FALSE is the default. TRUE will return an interactive plotly plot.

Details

This function will produce a simple random walk plot from a tidy_ distribution function.

Value

A ggplot or a plotly plot.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Autoplot: bootstrap_stat_plot(), tidy_autoplot(), tidy_combined_autoplot(), tidy_four_autoplot(), tidy_multi_dist_autoplot()

Examples

```
tidy_normal(.sd = .1, .num_sims = 5) |>
  tidy_random_walk(.value_type = "cum_sum") |>
  tidy_random_walk_autoplot()

tidy_normal(.sd = .1, .num_sims = 20) |>
  tidy_random_walk(.value_type = "cum_sum", .sample = TRUE, .replace = TRUE) |>
  tidy_random_walk_autoplot()
```

`tidy_range_statistic` *Get the range statistic*

Description

Takes in a numeric vector and returns back the range of that vector

Usage

```
tidy_range_statistic(.x)
```

Arguments

`.x`	A numeric vector

Details

Takes in a numeric vector and returns the range of that vector using the `diff` and `range` functions.

Value

A single number, the range statistic

Author(s)

Steven P. Sandeson II, MPH

See Also

Other Statistic: `ci_hi()`, `ci_lo()`, `tidy_kurtosis_vec()`, `tidy_skewness_vec()`, `tidy_stat_tbl()`

Examples

```
tidy_range_statistic(seq(1:10))
```

`tidy_scale_zero_one_vec` *Vector Function Scale to Zero and One*

Description

Takes a numeric vector and will return a vector that has been scaled from `[0,1]`

Usage

```
tidy_scale_zero_one_vec(.x)
```

Arguments

`.x`	A numeric vector to be scaled from `[0,1]` inclusive.

Details

Takes a numeric vector and will return a vector that has been scaled from `[0,1]` The input vector must be numeric. The computation is fairly straightforward. This may be helpful when trying to compare the distributions of data where a distribution like beta which requires data to be between 0 and 1

$$y[h] = (x - min(x))/(max(x) - min(x))$$

Value

A numeric vector

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Vector Function: `bootstrap_p_vec()`, `bootstrap_q_vec()`, `cgmean()`, `chmean()`, `ckurtosis()`, `cmean()`, `cmedian()`, `csd()`, `cskewness()`, `cvar()`, `tidy_kurtosis_vec()`, `tidy_skewness_vec()`

Examples

```
vec_1 <- rnorm(100, 2, 1)
vec_2 <- tidy_scale_zero_one_vec(vec_1)

dens_1 <- density(vec_1)
dens_2 <- density(vec_2)
max_x <- max(dens_1$x, dens_2$x)
max_y <- max(dens_1$y, dens_2$y)
plot(dens_1,
```

```
  asp = max_y / max_x, main = "Density vec_1 (Red) and vec_2 (Blue)",
  col = "red", xlab = "", ylab = "Density of Vec 1 and Vec 2"
)
lines(dens_2, col = "blue")
```

`tidy_skewness_vec` *Compute Skewness of a Vector*

Description

This function takes in a vector as it's input and will return the skewness of that vector. The length of this vector must be at least four numbers. The skewness explains the 'tailedness' of the distribution of data.
`((1/n) * sum(x - mu})^3) / (((()1/n) * sum(x - mu)^2)^(3/2)`

Usage

```
tidy_skewness_vec(.x)
```

Arguments

`.x`	A numeric vector of length four or more.

Details

A function to return the skewness of a vector.

Value

The skewness of a vector

Author(s)

Steven P. Sanderson II, MPH

See Also

https://en.wikipedia.org/wiki/Skewness

Other Statistic: `ci_hi()`, `ci_lo()`, `tidy_kurtosis_vec()`, `tidy_range_statistic()`, `tidy_stat_tbl()`

Other Vector Function: `bootstrap_p_vec()`, `bootstrap_q_vec()`, `cgmean()`, `chmean()`, `ckurtosis()`, `cmean()`, `cmedian()`, `csd()`, `cskewness()`, `cvar()`, `tidy_kurtosis_vec()`, `tidy_scale_zero_one_vec()`

Examples

```
tidy_skewness_vec(rnorm(100, 3, 2))
```

tidy_stat_tbl *Tidy Stats of Tidy Distribution*

Description

A function to return the `stat` function values of a given `tidy_` distribution output.

Usage

```
tidy_stat_tbl(
  .data,
  .x = y,
  .fns,
  .return_type = "vector",
  .use_data_table = FALSE,
  ...
)
```

Arguments

`.data`	The input data coming from a `tidy_` distribution function.
`.x`	The default is `y` but can be one of the other columns from the input data.
`.fns`	The default is `IQR`, but this can be any `stat` function like `quantile` or `median` etc.
`.return_type`	The default is "vector" which returns an `sapply` object.
`.use_data_table`	The default is FALSE, TRUE will use data.table under the hood and still return a tibble. If this argument is set to TRUE then the `.return_type` parameter will be ignored.
`...`	Addition function arguments to be supplied to the parameters of `.fns`

Details

A function to return the value(s) of a given `tidy_` distribution function output and chosen column from it. This function will only work with `tidy_` distribution functions.

There are currently three different output types for this function. These are:

- "vector" - which gives an `sapply()` output
- "list" - which gives an `lapply()` output, and
- "tibble" - which returns a `tibble` in long format.

Currently you can pass any stat function that performs an operation on a vector input. This means you can pass things like `IQR`, `quantile` and their associated arguments in the `...` portion of the function.

This function also by default will rename the value column of the tibble to the name of the function. This function will also give the column name of sim_number for the tibble output with the corresponding simulation numbers as the values.

For the sapply and lapply outputs the column names will also give the simulation number information by making column names like sim_number_1 etc.

There is an option of .use_data_table which can greatly enhance the speed of the calculations performed if used while still returning a tibble. The calculations are performed after turning the input data into a data.table object, performing the necessary calculation and then converting back to a tibble object.

Value

A return of object of either sapply lapply or tibble based upon user input.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Statistic: ci_hi(), ci_lo(), tidy_kurtosis_vec(), tidy_range_statistic(), tidy_skewness_vec()

Examples

```
tn <- tidy_normal(.num_sims = 3)

p <- c(0.025, 0.25, 0.5, 0.75, 0.95)

tidy_stat_tbl(tn, y, quantile, "vector", probs = p, na.rm = TRUE)
tidy_stat_tbl(tn, y, quantile, "list", probs = p)
tidy_stat_tbl(tn, y, quantile, "tibble", probs = p)
tidy_stat_tbl(tn, y, quantile, .use_data_table = TRUE, probs = p, na.rm = TRUE)
```

---

tidy_t *Tidy Randomly Generated T Distribution Tibble*

Description

This function will generate n random points from a rt distribution with a user provided, df, ncp, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.

- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_t(.n = 50, .df = 1, .ncp = 0, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

.n	The number of randomly generated points you want.
.df	Degrees of freedom, Inf is allowed.
.ncp	Non-centrality parameter.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying stats::rt(), and its underlying p, d, and q functions. For more information please see stats::rt()

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3664.htm

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_triangular(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other T Distribution: util_t_stats_tbl()

Examples

```
tidy_t()
```

`tidy_triangular` *Generate Tidy Data from Triangular Distribution*

Description

This function generates tidy data from the triangular distribution.

Usage

```
tidy_triangular(
  .n = 50,
  .min = 0,
  .max = 1,
  .mode = 1/2,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of x values for each simulation.
`.min`	The minimum value of the triangular distribution.
`.max`	The maximum value of the triangular distribution.
`.mode`	The mode (peak) value of the triangular distribution.
`.num_sims`	The number of simulations to perform.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

The function takes parameters for the triangular distribution (minimum, maximum, mode), the number of x values (`n`), the number of simulations (`num_sims`), and an option to return the result as a tibble (`return_tibble`). It performs various checks on the input parameters to ensure validity. The result is a data frame or tibble with tidy data for further analysis.

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Continuous Distribution: tidy_beta(), tidy_burr(), tidy_cauchy(), tidy_chisquare(), tidy_exponential(), tidy_f(), tidy_gamma(), tidy_generalized_beta(), tidy_generalized_pareto(), tidy_geometric(), tidy_inverse_burr(), tidy_inverse_exponential(), tidy_inverse_gamma(), tidy_inverse_normal(), tidy_inverse_pareto(), tidy_inverse_weibull(), tidy_logistic(), tidy_lognormal(), tidy_normal(), tidy_paralogistic(), tidy_pareto(), tidy_pareto1(), tidy_t(), tidy_uniform(), tidy_weibull(), tidy_zero_truncated_geometric()

Other Triangular: util_triangular_param_estimate(), util_triangular_stats_tbl()

Examples

```
tidy_triangular(.return_tibble = TRUE)
```

---

tidy_uniform *Tidy Randomly Generated Uniform Distribution Tibble*

Description

This function will generate n random points from a uniform distribution with a user provided, .min and .max values, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_uniform(.n = 50, .min = 0, .max = 1, .num_sims = 1, .return_tibble = TRUE)
```

Arguments

.n	The number of randomly generated points you want.
.min	A lower limit of the distribution.
.max	An upper limit of the distribution
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::runif()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::runif()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3662.htm

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_weibull()`, `tidy_zero_truncated_geometric()`

Other Uniform: `util_uniform_param_estimate()`, `util_uniform_stats_tbl()`

Examples

```
tidy_uniform()
```

---

tidy_weibull *Tidy Randomly Generated Weibull Distribution Tibble*

Description

This function will generate `n` random points from a weibull distribution with a user provided, `.shape`, `.scale`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_weibull(
  .n = 50,
  .shape = 1,
  .scale = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.shape	Shape parameter defaults to 0.
.scale	Scale parameter defaults to 1.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `stats::rweibull()`, and its underlying `p`, `d`, and `q` functions. For more information please see `stats::rweibull()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://www.itl.nist.gov/div898/handbook/eda/section3/eda3669.htm

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_zero_truncated_geometric()`

Other Weibull: `tidy_inverse_weibull()`, `util_weibull_param_estimate()`, `util_weibull_stats_tbl()`

Examples

```
tidy_weibull()
```

`tidy_zero_truncated_binomial` *Tidy Randomly Generated Binomial Distribution Tibble*

Description

This function will generate `n` random points from a zero truncated binomial distribution with a user provided, `.size`, `.prob`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_zero_truncated_binomial(
  .n = 50,
  .size = 1,
  .prob = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.size`	Number of trials, zero or more.
`.prob`	Probability of success on each trial 0 <= prob <= 1.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rztbinom()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rztbinom()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Discrete Distribution: `tidy_bernoulli()`, `tidy_binomial()`, `tidy_hypergeometric()`, `tidy_negative_binomial()`, `tidy_poisson()`, `tidy_zero_truncated_negative_binomial()`, `tidy_zero_truncated_poisson()`

Other Binomial: `tidy_binomial()`, `tidy_negative_binomial()`, `tidy_zero_truncated_negative_binomial()`, `util_binomial_param_estimate()`, `util_binomial_stats_tbl()`, `util_negative_binomial_param_estimate()`

Other Zero Truncated Distribution: `tidy_zero_truncated_geometric()`, `tidy_zero_truncated_poisson()`

Examples

```
tidy_zero_truncated_binomial()
```

---

tidy_zero_truncated_geometric

Tidy Randomly Generated Zero Truncated Geometric Distribution Tibble

---

Description

This function will generate `n` random points from a zero truncated Geometric distribution with a user provided, `.prob`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_zero_truncated_geometric(
  .n = 50,
  .prob = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.prob	A probability of success in each trial 0 < prob <= 1.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rztgeom()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rztgeom()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Geometric: `tidy_geometric()`, `util_geometric_param_estimate()`, `util_geometric_stats_tbl()`

Other Continuous Distribution: `tidy_beta()`, `tidy_burr()`, `tidy_cauchy()`, `tidy_chisquare()`, `tidy_exponential()`, `tidy_f()`, `tidy_gamma()`, `tidy_generalized_beta()`, `tidy_generalized_pareto()`, `tidy_geometric()`, `tidy_inverse_burr()`, `tidy_inverse_exponential()`, `tidy_inverse_gamma()`, `tidy_inverse_normal()`, `tidy_inverse_pareto()`, `tidy_inverse_weibull()`, `tidy_logistic()`, `tidy_lognormal()`, `tidy_normal()`, `tidy_paralogistic()`, `tidy_pareto()`, `tidy_pareto1()`, `tidy_t()`, `tidy_triangular()`, `tidy_uniform()`, `tidy_weibull()`

Other Zero Truncated Distribution: `tidy_zero_truncated_binomial()`, `tidy_zero_truncated_poisson()`

Examples

```
tidy_zero_truncated_geometric()
```

`tidy_zero_truncated_negative_binomial` *Tidy Randomly Generated Binomial Distribution Tibble*

Description

This function will generate `n` random points from a zero truncated binomial distribution with a user provided, `.size`, `.prob`, and number of random simulations to be produced. The function returns a tibble with the simulation number column the `x` column which corresponds to the `n` randomly generated points, the `d_`, `p_` and `q_` data points as well.

The data is returned un-grouped.

The columns that are output are:

- `sim_number` The current simulation number.
- `x` The current value of `n` for the current simulation.
- `y` The randomly generated data point.
- `dx` The `x` value from the `stats::density()` function.
- `dy` The `y` value from the `stats::density()` function.
- `p` The values from the resulting `p_` function of the distribution family.
- `q` The values from the resulting `q_` function of the distribution family.

Usage

```
tidy_zero_truncated_negative_binomial(
  .n = 50,
  .size = 0,
  .prob = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

`.n`	The number of randomly generated points you want.
`.size`	Number of trials, zero or more.
`.prob`	Probability of success on each trial 0 <= prob <= 1.
`.num_sims`	The number of randomly generated simulations you want.
`.return_tibble`	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rztnbinom()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rztnbinom()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Discrete Distribution: tidy_bernoulli(), tidy_binomial(), tidy_hypergeometric(), tidy_negative_binomial(), tidy_poisson(), tidy_zero_truncated_binomial(), tidy_zero_truncated_poisson()

Other Binomial: tidy_binomial(), tidy_negative_binomial(), tidy_zero_truncated_binomial(), util_binomial_param_estimate(), util_binomial_stats_tbl(), util_negative_binomial_param_estimate()

Examples

```
tidy_zero_truncated_negative_binomial()
```

---

tidy_zero_truncated_poisson

Tidy Randomly Generated Zero Truncated Poisson Distribution Tibble

---

Description

This function will generate n random points from a Zero Truncated Poisson distribution with a user provided, .lambda, and number of random simulations to be produced. The function returns a tibble with the simulation number column the x column which corresponds to the n randomly generated points, the d_, p_ and q_ data points as well.

The data is returned un-grouped.

The columns that are output are:

- sim_number The current simulation number.
- x The current value of n for the current simulation.
- y The randomly generated data point.
- dx The x value from the stats::density() function.
- dy The y value from the stats::density() function.
- p The values from the resulting p_ function of the distribution family.
- q The values from the resulting q_ function of the distribution family.

Usage

```
tidy_zero_truncated_poisson(
  .n = 50,
  .lambda = 1,
  .num_sims = 1,
  .return_tibble = TRUE
)
```

Arguments

.n	The number of randomly generated points you want.
.lambda	A vector of non-negative means.
.num_sims	The number of randomly generated simulations you want.
.return_tibble	A logical value indicating whether to return the result as a tibble. Default is TRUE.

Details

This function uses the underlying `actuar::rztpois()`, and its underlying `p`, `d`, and `q` functions. For more information please see `actuar::rztpois()`

Value

A tibble of randomly generated data.

Author(s)

Steven P. Sanderson II, MPH

See Also

https://openacttexts.github.io/Loss-Data-Analytics/ChapSummaryDistributions.html

Other Poisson: `tidy_poisson()`, `util_poisson_param_estimate()`, `util_poisson_stats_tbl()`

Other Zero Truncated Distribution: `tidy_zero_truncated_binomial()`, `tidy_zero_truncated_geometric()`

Other Discrete Distribution: `tidy_bernoulli()`, `tidy_binomial()`, `tidy_hypergeometric()`, `tidy_negative_binomial()`, `tidy_poisson()`, `tidy_zero_truncated_binomial()`, `tidy_zero_truncated_negative_`

Examples

```
tidy_zero_truncated_poisson()
```

`triangle_plot` *Triangle Distribution PDF Plot*

Description

This function generates a probability density function (PDF) plot for the triangular distribution.

Usage

```
triangle_plot(.data, .interactive = FALSE)
```

Arguments

`.data`	Tidy data from the `tidy_triangular` function.
`.interactive`	A logical value indicating whether to return an interactive plot using plotly. Default is FALSE.

Details

The function checks if the input data is a data frame or tibble, and if it comes from the `tidy_triangular` function. It then extracts necessary attributes for the plot and creates a PDF plot using ggplot2. The plot includes data points and segments to represent the triangular distribution.

Value

The function returns a ggplot2 object representing the probability density function plot for the triangular distribution.

Author(s)

Steven P. Sanderson II, MPH

Examples

```
# Example: Generating a PDF plot for the triangular distribution
data <- tidy_triangular(.n = 50, .min = 0, .max = 1, .mode = 1/2, .num_sims = 1,
.return_tibble = TRUE)
triangle_plot(data)
```

`util_bernoulli_param_estimate` *Estimate Bernoulli Parameters*

Description

This function will attempt to estimate the Bernoulli prob parameter given some vector of values `.x`. The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated Bernoulli data.

Usage

```
util_bernoulli_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be non-negative integers.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will see if the given vector `.x` is a numeric vector. It will attempt to estimate the prob parameter of a Bernoulli distribution.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Bernoulli: `tidy_bernoulli()`, `util_bernoulli_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

tb <- tidy_bernoulli(.prob = .1) |> pull(y)
output <- util_bernoulli_param_estimate(tb)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_bernoulli_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_bernoulli_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Bernoulli: `tidy_bernoulli()`, `util_bernoulli_param_estimate()`

Other Distribution Statistics: `util_beta_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_chisquare_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_logistic_stats_tbl()`, `util_lognormal_stats_tbl()`, `util_negative_binomial_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_t_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_bernoulli() |>
  util_bernoulli_stats_tbl() |>
  glimpse()
```

---

util_beta_aic *Calculate Akaike Information Criterion (AIC) for Beta Distribution*

Description

This function estimates the parameters of a beta distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_beta_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a beta distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a beta distribution fitted to the provided data.

Initial parameter estimates: The choice of initial values can impact the convergence of the optimization.

Optimization method: You might explore different optimization methods within optim for potentially better performance. Data transformation: Depending on your data, you may need to apply transformations (e.g., scaling to `[0,1]` interval) before fitting the beta distribution.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted beta distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), convert_to_ts(), quantile_normalize(), tidy_mcmc_sampling(), util_binomial_aic(), util_cauchy_aic(), util_chisq_aic(), util_exponential_aic(), util_gamma_aic(), util_geometric_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_pareto_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rbeta(30, 1, 1)
util_beta_aic(x)
```

---

util_beta_param_estimate

Estimate Beta Parameters

Description

This function will automatically scale the data from 0 to 1 if it is not already. This means you can pass a vector like `mtcars$mpg` and not worry about it.

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated beta data.

Three different methods of shape parameters are supplied:

- Bayes
- NIST mme
- EnvStats mme, see `EnvStats::ebeta()`

Usage

```
util_beta_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be numeric, and all values must be 0 <= x <= 1
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the beta shape1 and shape2 parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Beta: `tidy_beta()`, `tidy_generalized_beta()`, `util_beta_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

x <- mtcars$mpg
output <- util_beta_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

tb <- rbeta(50, 2.5, 1.4)
util_beta_param_estimate(tb)$parameter_tbl
```

`util_beta_stats_tbl` *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_beta_stats_tbl(.data)
```

Arguments

`.data`	The data being passed from a `tidy_` distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of `tidy_` distribution. It is required that data be passed from a `tidy_` distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Beta: `tidy_beta()`, `tidy_generalized_beta()`, `util_beta_param_estimate()`

Other Distribution Statistics: `util_bernoulli_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_chisquare_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_logistic_stats_tbl()`, `util_lognormal_stats_tbl()`, `util_negative_binomial_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_t_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_beta() |>
  util_beta_stats_tbl() |>
  glimpse()
```

`util_binomial_aic` *Calculate Akaike Information Criterion (AIC) for Binomial Distribution*

Description

This function estimates the size and probability parameters of a binomial distribution from the provided data and then calculates the AIC value based on the fitted distribution.

Usage

```
util_binomial_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a binomial distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a binomial distribution fitted to the provided data.

This function fits a binomial distribution to the provided data. It estimates the size and probability parameters of the binomial distribution from the data. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates as starting points for the size and probability parameters of the binomial distribution.

Optimization method: Since the parameters are directly calculated from the data, no optimization is needed.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted binomial distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rbinom(30, size = 10, prob = 0.2)
util_binomial_aic(x)
```

`util_binomial_param_estimate`
Estimate Binomial Parameters

Description

This function will check to see if some given vector `.x` is either a numeric vector or a factor vector with at least two levels then it will cause an error and the function will abort. The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated binomial data.

Usage

```
util_binomial_param_estimate(.x, .size = NULL, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be numeric, and all values must be 0 <= x <= 1
`.size`	Number of trials, zero or more.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the binomial p_hat and size parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Binomial: tidy_binomial(), tidy_negative_binomial(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_binomial(), util_binomial_stats_tbl(), util_negative_binomial_param_esti

Examples

```
library(dplyr)
library(ggplot2)

tb <- rbinom(50, 1, .1)
output <- util_binomial_param_estimate(tb)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_binomial_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_binomial_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Binomial: tidy_binomial(), tidy_negative_binomial(), tidy_zero_truncated_binomial(), tidy_zero_truncated_negative_binomial(), util_binomial_param_estimate(), util_negative_binomial_param

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(), util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_binomial() |>
  util_binomial_stats_tbl() |>
  glimpse()
```

---

util_burr_param_estimate

Estimate Burr Parameters

Description

This function will attempt to estimate the Burr prob parameter given some vector of values `.x`. The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated Burr data.

Usage

```
util_burr_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be non-negative integers.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will see if the given vector `.x` is a numeric vector. It will attempt to estimate the prob parameter of a Burr distribution.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Burr: `tidy_burr()`, `tidy_inverse_burr()`, `util_burr_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

tb <- tidy_burr(.shape1 = 1, .shape2 = 2, .rate = .3) |> pull(y)
output <- util_burr_param_estimate(tb)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_burr_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_burr_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Burr: tidy_burr(), tidy_inverse_burr(), util_burr_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(), util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl(), util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_burr() |>
  util_burr_stats_tbl() |>
  glimpse()
```

---

util_cauchy_aic *Calculate Akaike Information Criterion (AIC) for Cauchy Distribution*

Description

This function estimates the parameters of a Cauchy distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_cauchy_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a Cauchy distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a Cauchy distribution fitted to the provided data.

This function fits a Cauchy distribution to the provided data using maximum likelihood estimation. It first estimates the initial parameters of the Cauchy distribution using the method of moments. Then, it optimizes the negative log-likelihood function using the provided data and the initial parameter estimates. Finally, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates for the initial location and scale parameters of the Cauchy distribution.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted Cauchy distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), convert_to_ts(), quantile_normalize(), tidy_mcmc_sampling(), util_beta_aic(), util_binomial_aic(), util_chisq_aic(), util_exponential_aic(), util_gamma_aic(), util_geometric_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_pareto_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rcauchy(30)
util_cauchy_aic(x)
```

`util_cauchy_param_estimate` *Estimate Cauchy Parameters*

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated cauchy data.

Usage

```
util_cauchy_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the cauchy location and scale parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Cauchy: `tidy_cauchy()`, `util_cauchy_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

x <- tidy_cauchy(.location = 0, .scale = 1)$y
output <- util_cauchy_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_cauchy_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_cauchy_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Cauchy: tidy_cauchy(), util_cauchy_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(), util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl(), util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_cauchy() |>
  util_cauchy_stats_tbl() |>
  glimpse()
```

`util_chisquare_param_estimate`
Estimate Chisquare Parameters

Description

This function will attempt to estimate the Chisquare prob parameter given some vector of values `.x`. The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated Chisquare data.

Usage

```
util_chisquare_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be non-negative integers.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will see if the given vector `.x` is a numeric vector. It will attempt to estimate the prob parameter of a Chisquare distribution. The function first performs tidyeval on the input data to ensure it's a numeric vector. It then checks if there are at least two data points, as this is a requirement for parameter estimation.

The estimation of the chi-square distribution parameters is performed using maximum likelihood estimation (MLE) implemented with the `bbmle` package. The negative log-likelihood function is minimized to obtain the estimates for the degrees of freedom (`doff`) and the non-centrality parameter (`ncp`). Initial values for the optimization are set based on the sample variance and mean, but these can be adjusted if necessary.

If the estimation fails or encounters an error, the function returns `NA` for both `doff` and `ncp`.

Finally, the function returns a tibble containing the following information:

dist_type The type of distribution, which is "Chisquare" in this case.

samp_size The sample size, i.e., the number of data points in the input vector.

min The minimum value of the data points.

max The maximum value of the data points.

mean The mean of the data points.

degrees_of_freedom The estimated degrees of freedom (`doff`) for the chi-square distribution.

ncp The estimated non-centrality parameter (`ncp`) for the chi-square distribution.

Additionally, if the argument `.auto_gen_empirical` is set to `TRUE` (which is the default behavior), the function also returns a combined tibble containing both empirical and chi-square distribution data, obtained by calling `tidy_empirical` and `tidy_chisquare`, respectively.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Chisquare: `tidy_chisquare()`, `util_chisquare_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

tc <- tidy_chisquare(.n = 500, .df = 6, .ncp = 1) |> pull(y)
output <- util_chisquare_param_estimate(tc)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

`util_chisquare_stats_tbl` *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_chisquare_stats_tbl(.data)
```

Arguments

`.data`	The data being passed from a `tidy_` distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of `tidy_` distribution. It is required that data be passed from a `tidy_` distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Chisquare: `tidy_chisquare()`, `util_chisquare_param_estimate()`

Other Distribution Statistics: `util_bernoulli_stats_tbl()`, `util_beta_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_logistic_stats_tbl()`, `util_lognormal_stats_tbl()`, `util_negative_binomial_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_t_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_chisquare() |>
  util_chisquare_stats_tbl() |>
  glimpse()
```

`util_chisq_aic` *Calculate Akaike Information Criterion (AIC) for Chi-Square Distribution*

Description

This function estimates the parameters of a chi-square distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_chisq_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a chi-square distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a chi-square distribution fitted to the provided data.

Value

The AIC value calculated based on the fitted chi-square distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rchisq(30, df = 3)
util_chisq_aic(x)
```

`util_exponential_aic` *Calculate Akaike Information Criterion (AIC) for Exponential Distribution*

Description

This function estimates the rate parameter of an exponential distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_exponential_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to an exponential distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for an exponential distribution fitted to the provided data.

This function fits an exponential distribution to the provided data using maximum likelihood estimation. It estimates the rate parameter of the exponential distribution using maximum likelihood estimation. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the reciprocal of the mean of the data as the initial estimate for the rate parameter.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted exponential distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rexp(30)
util_exponential_aic(x)
```

`util_exponential_param_estimate`
Estimate Exponential Parameters

Description

This function will attempt to estimate the exponential rate parameter given some vector of values. The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated exponential data.

Usage

```
util_exponential_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be numeric.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will see if the given vector `.x` is a numeric vector.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Exponential: tidy_exponential(), tidy_inverse_exponential(), util_exponential_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

te <- tidy_exponential(.rate = .1) |> pull(y)
output <- util_exponential_param_estimate(te)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_exponential_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_exponential_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Exponential: tidy_exponential(), tidy_inverse_exponential(), util_exponential_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl(), util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_exponential() |>
  util_exponential_stats_tbl() |>
  glimpse()
```

---

util_f_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_f_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other F Distribution: tidy_f()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl(), util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_f() |>
  util_f_stats_tbl() |>
  glimpse()
```

---

util_gamma_aic *Calculate Akaike Information Criterion (AIC) for Gamma Distribution*

Description

This function estimates the shape and scale parameters of a gamma distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_gamma_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a gamma distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a gamma distribution fitted to the provided data.

This function fits a gamma distribution to the provided data using maximum likelihood estimation. It estimates the shape and scale parameters of the gamma distribution using maximum likelihood estimation. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates as starting points for the shape and scale parameters of the gamma distribution.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted gamma distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), convert_to_ts(), quantile_normalize(), tidy_mcmc_sampling(), util_beta_aic(), util_binomial_aic(), util_cauchy_aic(), util_chisq_aic(), util_exponential_aic(), util_geometric_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_pareto_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rgamma(30, shape = 1)
util_gamma_aic(x)
```

---

util_gamma_param_estimate

Estimate Gamma Parameters

Description

This function will attempt to estimate the gamma shape and scale parameters given some vector of values. The function will return a list output by default, and if the parameter .auto_gen_empirical is set to TRUE then the empirical data given to the parameter .x will be run through the tidy_empirical() function and combined with the estimated gamma data.

Usage

```
util_gamma_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

.x	The vector of data to be passed to the function. Must be numeric.
.auto_gen_empirical	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the tidy_empirical() output for the .x parameter and use the tidy_combine_distributions(). The user can then plot out the data using $combined_data_tbl from the function output.

Details

This function will see if the given vector `.x` is a numeric vector.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Gamma: `tidy_gamma()`, `tidy_inverse_gamma()`, `util_gamma_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

tg <- tidy_gamma(.shape = 1, .scale = .3) |> pull(y)
output <- util_gamma_param_estimate(tg)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_gamma_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_gamma_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Gamma: tidy_gamma(), tidy_inverse_gamma(), util_gamma_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl(), util_logistic_stats_tb util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_gamma() |>
  util_gamma_stats_tbl() |>
  glimpse()
```

---

util_geometric_aic *Calculate Akaike Information Criterion (AIC) for Geometric Distribution*

Description

This function estimates the probability parameter of a geometric distribution from the provided data and then calculates the AIC value based on the fitted distribution.

Usage

```
util_geometric_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a geometric distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a geometric distribution fitted to the provided data.

This function fits a geometric distribution to the provided data. It estimates the probability parameter of the geometric distribution from the data. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimate as a starting point for the probability parameter of the geometric distribution.

Optimization method: Since the parameter is directly calculated from the data, no optimization is needed.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted geometric distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), convert_to_ts(), quantile_normalize(), tidy_mcmc_sampling(), util_beta_aic(), util_binomial_aic(), util_cauchy_aic(), util_chisq_aic(), util_exponential_aic(), util_gamma_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_pareto_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rgeom(100, prob = 0.2)
util_geometric_aic(x)
```

`util_geometric_param_estimate` *Estimate Geometric Parameters*

Description

This function will attempt to estimate the geometric prob parameter given some vector of values `.x`. The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated geometric data.

Usage

```
util_geometric_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be non-negative integers.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will see if the given vector `.x` is a numeric vector. It will attempt to estimate the prob parameter of a geometric distribution.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Geometric: `tidy_geometric()`, `tidy_zero_truncated_geometric()`, `util_geometric_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

tg <- tidy_geometric(.prob = .1) |> pull(y)
output <- util_geometric_param_estimate(tg)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_geometric_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_geometric_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Geometric: tidy_geometric(), tidy_zero_truncated_geometric(), util_geometric_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_hypergeometric_stats_tbl(), util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_geometric() |>
  util_geometric_stats_tbl() |>
  glimpse()
```

---

util_hypergeometric_aic

Calculate Akaike Information Criterion (AIC) for Hypergeometric Distribution

---

Description

This function estimates the parameters m, n, and k of a hypergeometric distribution from the provided data and then calculates the AIC value based on the fitted distribution.

Usage

```
util_hypergeometric_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a hypergeometric distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a hypergeometric distribution fitted to the provided data.

This function fits a hypergeometric distribution to the provided data. It estimates the parameters m, n, and k of the hypergeometric distribution from the data. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function does not estimate parameters; they are directly calculated from the data.

Optimization method: Since the parameters are directly calculated from the data, no optimization is needed.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted hypergeometric distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rhyper(100, m = 10, n = 10, k = 5)
util_hypergeometric_aic(x)
```

`util_hypergeometric_param_estimate`

Estimate Hypergeometric Parameters

Description

This function will attempt to estimate the geometric prob parameter given some vector of values `.x`. Estimate m, the number of white balls in the urn, or m+n, the total number of balls in the urn, for a hypergeometric distribution.

Usage

```
util_hypergeometric_param_estimate(
  .x,
  .m = NULL,
  .total = NULL,
  .k,
  .auto_gen_empirical = TRUE
)
```

Arguments

.x	A non-negative integer indicating the number of white balls out of a sample of size .k drawn without replacement from the urn. You cannot have missing, undefined or infinite values.
.m	Non-negative integer indicating the number of white balls in the urn. You must supply .m or .total, but not both. You cannot have missing values.
.total	A positive integer indicating the total number of balls in the urn (i.e., m+n). You must supply .m or .total, but not both. You cannot have missing values.
.k	A positive integer indicating the number of balls drawn without replacement from the urn. You cannot have missing values.
.auto_gen_empirical	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the tidy_empirical() output for the .x parameter and use the tidy_combine_distributions(). The user can then plot out the data using $combined_data_tbl from the function output.

Details

This function will see if the given vector .x is a numeric integer. It will attempt to estimate the prob parameter of a geometric distribution. Missing (NA), undefined (NaN), and infinite (Inf, -Inf) values are not allowed. Let .x be an observation from a hypergeometric distribution with parameters .m = M, .n = N, and .k = K. In R nomenclature, .x represents the number of white balls drawn out of a sample of .k balls drawn without replacement from an urn containing .m white balls and .n black balls. The total number of balls in the urn is thus .m + .n. Denote the total number of balls by T = .m + .n

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Hypergeometric: tidy_hypergeometric(), util_hypergeometric_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

th <- rhyper(10, 20, 30, 5)
output <- util_hypergeometric_param_estimate(th, .total = 50, .k = 5)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_hypergeometric_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_hypergeometric_stats_tbl(.data)
```

Arguments

`.data`	The data being passed from a `tidy_` distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of `tidy_` distribution. It is required that data be passed from a `tidy_` distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Hypergeometric: tidy_hypergeometric(), util_hypergeometric_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_hypergeometric() |>
  util_hypergeometric_stats_tbl() |>
  glimpse()
```

---

util_logistic_aic *Calculate Akaike Information Criterion (AIC) for Logistic Distribution*

Description

This function estimates the location and scale parameters of a logistic distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_logistic_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a logistic distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a logistic distribution fitted to the provided data.

This function fits a logistic distribution to the provided data using maximum likelihood estimation. It estimates the location and scale parameters of the logistic distribution using maximum likelihood estimation. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates as starting points for the location and scale parameters of the logistic distribution.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted logistic distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rlogis(30)
util_logistic_aic(x)
```

---

util_logistic_param_estimate

Estimate Logistic Parameters

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated logistic data.

Three different methods of shape parameters are supplied:

- MLE
- MME
- MMUE

Usage

```
util_logistic_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the logistic location and scale parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Logistic: `tidy_logistic()`, `tidy_paralogistic()`, `util_logistic_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

x <- mtcars$mpg
output <- util_logistic_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

t <- rlogis(50, 2.5, 1.4)
util_logistic_param_estimate(t)$parameter_tbl
```

`util_logistic_stats_tbl` *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_logistic_stats_tbl(.data)
```

Arguments

`.data`	The data being passed from a `tidy_` distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of `tidy_` distribution. It is required that data be passed from a `tidy_` distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Logistic: `tidy_logistic()`, `tidy_paralogistic()`, `util_logistic_param_estimate()`

Other Distribution Statistics: `util_bernoulli_stats_tbl()`, `util_beta_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_chisquare_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_lognormal_stats_tbl()`, `util_negative_binomial_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_t_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_logistic() |>
  util_logistic_stats_tbl() |>
  glimpse()
```

`util_lognormal_aic` *Calculate Akaike Information Criterion (AIC) for Log-Normal Distribution*

Description

This function estimates the meanlog and sdlog parameters of a log-normal distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_lognormal_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a log-normal distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a log-normal distribution fitted to the provided data.

This function fits a log-normal distribution to the provided data using maximum likelihood estimation. It estimates the meanlog and sdlog parameters of the log-normal distribution using maximum likelihood estimation. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates as starting points for the meanlog and sdlog parameters of the log-normal distribution.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted log-normal distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rlnorm(100, meanlog = 0, sdlog = 1)
util_lognormal_aic(x)
```

`util_lognormal_param_estimate`

Estimate Lognormal Parameters

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated lognormal data.

Three different methods of shape parameters are supplied:

- mme, see `EnvStats::elnorm()`
- mle, see `EnvStats::elnorm()`

Usage

```
util_lognormal_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the lognormal meanlog and log sd parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Lognormal: tidy_lognormal(), util_lognormal_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

x <- mtcars$mpg
output <- util_lognormal_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

tb <- tidy_lognormal(.meanlog = 2, .sdlog = 1) |> pull(y)
util_lognormal_param_estimate(tb)$parameter_tbl
```

---

util_lognormal_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_lognormal_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Lognormal: `tidy_lognormal()`, `util_lognormal_param_estimate()`

Other Distribution Statistics: `util_bernoulli_stats_tbl()`, `util_beta_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_chisquare_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_logistic_stats_tbl()`, `util_negative_binomial_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_t_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_lognormal() |>
  util_lognormal_stats_tbl() |>
  glimpse()
```

---

util_negative_binomial_param_estimate

Estimate Negative Binomial Parameters

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated negative binomial data.

Two different methods of shape parameters are supplied:

- MLE/MME
- MMUE

Usage

```
util_negative_binomial_param_estimate(.x, .size, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.size`	The size parameter.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the negative binomial size and prob parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_triangular_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Binomial: `tidy_binomial()`, `tidy_negative_binomial()`, `tidy_zero_truncated_binomial()`, `tidy_zero_truncated_negative_binomial()`, `util_binomial_param_estimate()`, `util_binomial_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

x <- as.integer(mtcars$mpg)
output <- util_negative_binomial_param_estimate(x, .size = 1)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

t <- rnbinom(50, 1, .1)
util_negative_binomial_param_estimate(t, .size = 1)$parameter_tbl
```

`util_negative_binomial_stats_tbl` *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_negative_binomial_stats_tbl(.data)
```

Arguments

`.data`	The data being passed from a `tidy_` distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of `tidy_` distribution. It is required that data be passed from a `tidy_` distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Distribution Statistics: `util_bernoulli_stats_tbl()`, `util_beta_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_chisquare_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_logistic_stats_tbl()`, `util_lognormal_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_t_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_negative_binomial() |>
  util_negative_binomial_stats_tbl() |>
  glimpse()
```

`util_normal_aic` *Calculate Akaike Information Criterion (AIC) for Normal Distribution*

Description

This function estimates the parameters of a normal distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_normal_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a normal distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a normal distribution fitted to the provided data.

Value

The AIC value calculated based on the fitted normal distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
data <- rnorm(30)
util_normal_aic(data)
```

util_normal_param_estimate *Estimate Normal Gaussian Parameters*

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated normal data.

Three different methods of shape parameters are supplied:

- MLE/MME
- MVUE

Usage

```
util_normal_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the normal gaussian mean and standard deviation parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_pareto_param_estimate()`

util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Gaussian: tidy_inverse_normal(), tidy_normal(), util_normal_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

x <- mtcars$mpg
output <- util_normal_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

t <- rnorm(50, 0, 1)
util_normal_param_estimate(t)$parameter_tbl
```

---

util_normal_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_normal_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Gaussian: tidy_inverse_normal(), tidy_normal(), util_normal_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_normal() |>
  util_normal_stats_tbl() |>
  glimpse()
```

---

util_pareto_aic *Calculate Akaike Information Criterion (AIC) for Pareto Distribution*

Description

This function estimates the shape and scale parameters of a Pareto distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_pareto_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a Pareto distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a Pareto distribution fitted to the provided data.

This function fits a Pareto distribution to the provided data using maximum likelihood estimation. It estimates the shape and scale parameters of the Pareto distribution using maximum likelihood estimation. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates as starting points for the shape and scale parameters of the Pareto distribution.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted Pareto distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: check_duplicate_rows(), convert_to_ts(), quantile_normalize(), tidy_mcmc_sampling(), util_beta_aic(), util_binomial_aic(), util_cauchy_aic(), util_chisq_aic(), util_exponential_aic(), util_gamma_aic(), util_geometric_aic(), util_hypergeometric_aic(), util_logistic_aic(), util_lognormal_aic(), util_normal_aic(), util_poisson_aic(), util_uniform_aic(), util_weibull_aic()

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- TidyDensity::tidy_pareto()$y
util_pareto_aic(x)
```

---

util_pareto_param_estimate

Estimate Pareto Parameters

Description

The function will return a list output by default, and if the parameter .auto_gen_empirical is set to TRUE then the empirical data given to the parameter .x will be run through the tidy_empirical() function and combined with the estimated pareto data.

Two different methods of shape parameters are supplied:

- LSE
- MLE

Usage

```
util_pareto_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

.x	The vector of data to be passed to the function.
.auto_gen_empirical	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the tidy_empirical() output for the .x parameter and use the tidy_combine_distributions(). The user can then plot out the data using $combined_data_tbl from the function output.

Details

This function will attempt to estimate the pareto shape and scale parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Pareto: tidy_generalized_pareto(), tidy_inverse_pareto(), tidy_pareto(), tidy_pareto1(), util_pareto_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

x <- mtcars$mpg
output <- util_pareto_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

t <- tidy_pareto(50, 1, 1) |> pull(y)
util_pareto_param_estimate(t)$parameter_tbl
```

---

util_pareto_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_pareto_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Pareto: tidy_generalized_pareto(), tidy_inverse_pareto(), tidy_pareto(), tidy_pareto1(), util_pareto_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_pareto() |>
  util_pareto_stats_tbl() |>
  glimpse()
```

`util_poisson_aic` *Calculate Akaike Information Criterion (AIC) for Poisson Distribution*

Description

This function estimates the lambda parameter of a Poisson distribution from the provided data and then calculates the AIC value based on the fitted distribution.

Usage

```
util_poisson_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a Poisson distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a Poisson distribution fitted to the provided data.

This function fits a Poisson distribution to the provided data. It estimates the lambda parameter of the Poisson distribution from the data. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimate as a starting point for the lambda parameter of the Poisson distribution.

Optimization method: Since the parameter is directly calculated from the data, no optimization is needed.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted Poisson distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_uniform_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rpois(100, lambda = 2)
util_poisson_aic(x)
```

`util_poisson_param_estimate`
Estimate Poisson Parameters

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated poisson data.

Usage

```
util_poisson_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the pareto lambda parameter given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate(), util_weibull_param_estimate()

Other Poisson: tidy_poisson(), tidy_zero_truncated_poisson(), util_poisson_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

x <- as.integer(mtcars$mpg)
output <- util_poisson_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

t <- rpois(50, 5)
util_poisson_param_estimate(t)$parameter_tbl
```

---

util_poisson_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_poisson_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Poisson: tidy_poisson(), tidy_zero_truncated_poisson(), util_poisson_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_poisson() |>
  util_poisson_stats_tbl() |>
  glimpse()
```

---

util_triangular_param_estimate

Estimate Triangular Parameters

Description

This function will attempt to estimate the triangular min, mode, and max parameters given some vector of values.

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated beta data.

Usage

```
util_triangular_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function. Must be numeric, and all values must be 0 <= x <= 1
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the triangular min, mode, and max parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: `util_bernoulli_param_estimate()`, `util_beta_param_estimate()`, `util_binomial_param_estimate()`, `util_burr_param_estimate()`, `util_cauchy_param_estimate()`, `util_chisquare_param_estimate()`, `util_exponential_param_estimate()`, `util_gamma_param_estimate()`, `util_geometric_param_estimate()`, `util_hypergeometric_param_estimate()`, `util_logistic_param_estimate()`, `util_lognormal_param_estimate()`, `util_negative_binomial_param_estimate()`, `util_normal_param_estimate()`, `util_pareto_param_estimate()`, `util_poisson_param_estimate()`, `util_uniform_param_estimate()`, `util_weibull_param_estimate()`

Other Triangular: `tidy_triangular()`, `util_triangular_stats_tbl()`

Examples

```
library(dplyr)
library(ggplot2)

x <- mtcars$mpg
output <- util_triangular_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()

params <- tidy_triangular()$y |>
  util_triangular_param_estimate()
params$parameter_tbl
```

util_triangular_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_triangular_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Triangular: tidy_triangular(), util_triangular_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_uniform_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_triangular() |>
  util_triangular_stats_tbl() |>
  glimpse()
```

`util_t_stats_tbl` *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_t_stats_tbl(.data)
```

Arguments

`.data`	The data being passed from a `tidy_` distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of `tidy_` distribution. It is required that data be passed from a `tidy_` distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other T Distribution: `tidy_t()`

Other Distribution Statistics: `util_bernoulli_stats_tbl()`, `util_beta_stats_tbl()`, `util_binomial_stats_tbl()`, `util_burr_stats_tbl()`, `util_cauchy_stats_tbl()`, `util_chisquare_stats_tbl()`, `util_exponential_stats_tbl()`, `util_f_stats_tbl()`, `util_gamma_stats_tbl()`, `util_geometric_stats_tbl()`, `util_hypergeometric_stats_tbl()`, `util_logistic_stats_tbl()`, `util_lognormal_stats_tbl()`, `util_negative_binomial_stats_tbl()`, `util_normal_stats_tbl()`, `util_pareto_stats_tbl()`, `util_poisson_stats_tbl()`, `util_triangular_stats_tbl()`, `util_uniform_stats_tbl()`, `util_weibull_stats_tbl()`

Examples

```
library(dplyr)

tidy_t() |>
  util_t_stats_tbl() |>
  glimpse()
```

`util_uniform_aic` *Calculate Akaike Information Criterion (AIC) for Uniform Distribution*

Description

This function estimates the min and max parameters of a uniform distribution from the provided data and then calculates the AIC value based on the fitted distribution.

Usage

```
util_uniform_aic(.x)
```

Arguments

`.x`	A numeric vector containing the data to be fitted to a uniform distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a uniform distribution fitted to the provided data.

This function fits a uniform distribution to the provided data. It estimates the min and max parameters of the uniform distribution from the range of the data. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the minimum and maximum values of the data as starting points for the min and max parameters of the uniform distribution.

Optimization method: Since the parameters are directly calculated from the data, no optimization is needed.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted uniform distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_weibull_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- runif(30)
util_uniform_aic(x)
```

`util_uniform_param_estimate` *Estimate Uniform Parameters*

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated uniform data.

Usage

```
util_uniform_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

`.x`	The vector of data to be passed to the function.
`.auto_gen_empirical`	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the `tidy_empirical()` output for the `.x` parameter and use the `tidy_combine_distributions()`. The user can then plot out the data using `$combined_data_tbl` from the function output.

Details

This function will attempt to estimate the uniform min and max parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_weibull_param_estimate()

Other Uniform: tidy_uniform(), util_uniform_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

x <- tidy_uniform(.min = 1, .max = 3)$y
output <- util_uniform_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl |>
  tidy_combined_autoplot()
```

---

util_uniform_stats_tbl

Distribution Statistics

Description

Returns distribution statistics in a tibble.

Usage

```
util_uniform_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Uniform: tidy_uniform(), util_uniform_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_weibull_stats_tbl()

Examples

```
library(dplyr)

tidy_uniform() |>
  util_uniform_stats_tbl() |>
  glimpse()
```

---

util_weibull_aic *Calculate Akaike Information Criterion (AIC) for Weibull Distribution*

Description

This function estimates the shape and scale parameters of a Weibull distribution from the provided data using maximum likelihood estimation, and then calculates the AIC value based on the fitted distribution.

Usage

```
util_weibull_aic(.x)
```

Arguments

.x	A numeric vector containing the data to be fitted to a Weibull distribution.

Details

This function calculates the Akaike Information Criterion (AIC) for a Weibull distribution fitted to the provided data.

This function fits a Weibull distribution to the provided data using maximum likelihood estimation. It estimates the shape and scale parameters of the Weibull distribution using maximum likelihood estimation. Then, it calculates the AIC value based on the fitted distribution.

Initial parameter estimates: The function uses the method of moments estimates as starting points for the shape and scale parameters of the Weibull distribution.

Optimization method: The function uses the optim function for optimization. You might explore different optimization methods within optim for potentially better performance.

Goodness-of-fit: While AIC is a useful metric for model comparison, it's recommended to also assess the goodness-of-fit of the chosen model using visualization and other statistical tests.

Value

The AIC value calculated based on the fitted Weibull distribution to the provided data.

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Utility: `check_duplicate_rows()`, `convert_to_ts()`, `quantile_normalize()`, `tidy_mcmc_sampling()`, `util_beta_aic()`, `util_binomial_aic()`, `util_cauchy_aic()`, `util_chisq_aic()`, `util_exponential_aic()`, `util_gamma_aic()`, `util_geometric_aic()`, `util_hypergeometric_aic()`, `util_logistic_aic()`, `util_lognormal_aic()`, `util_normal_aic()`, `util_pareto_aic()`, `util_poisson_aic()`, `util_uniform_aic()`

Examples

```
# Example 1: Calculate AIC for a sample dataset
set.seed(123)
x <- rweibull(100, shape = 2, scale = 1)
util_weibull_aic(x)
```

`util_weibull_param_estimate`

Estimate Weibull Parameters

Description

The function will return a list output by default, and if the parameter `.auto_gen_empirical` is set to `TRUE` then the empirical data given to the parameter `.x` will be run through the `tidy_empirical()` function and combined with the estimated weibull data.

Usage

```
util_weibull_param_estimate(.x, .auto_gen_empirical = TRUE)
```

Arguments

.x	The vector of data to be passed to the function.
.auto_gen_empirical	This is a boolean value of TRUE/FALSE with default set to TRUE. This will automatically create the tidy_empirical() output for the .x parameter and use the tidy_combine_distributions(). The user can then plot out the data using $combined_data_tbl from the function output.

Details

This function will attempt to estimate the weibull shape and scale parameters given some vector of values.

Value

A tibble/list

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Parameter Estimation: util_bernoulli_param_estimate(), util_beta_param_estimate(), util_binomial_param_estimate(), util_burr_param_estimate(), util_cauchy_param_estimate(), util_chisquare_param_estimate(), util_exponential_param_estimate(), util_gamma_param_estimate(), util_geometric_param_estimate(), util_hypergeometric_param_estimate(), util_logistic_param_estimate(), util_lognormal_param_estimate(), util_negative_binomial_param_estimate(), util_normal_param_estimate(), util_pareto_param_estimate(), util_poisson_param_estimate(), util_triangular_param_estimate(), util_uniform_param_estimate()

Other Weibull: tidy_inverse_weibull(), tidy_weibull(), util_weibull_stats_tbl()

Examples

```
library(dplyr)
library(ggplot2)

x <- tidy_weibull(.shape = 1, .scale = 2)$y
output <- util_weibull_param_estimate(x)

output$parameter_tbl

output$combined_data_tbl %>%
  tidy_combined_autoplot()
```

util_weibull_stats_tbl *Distribution Statistics*

Description

Returns distribution statistics in a tibble.

Usage

```
util_weibull_stats_tbl(.data)
```

Arguments

.data	The data being passed from a tidy_ distribution function.

Details

This function will take in a tibble and returns the statistics of the given type of tidy_ distribution. It is required that data be passed from a tidy_ distribution function.

Value

A tibble

Author(s)

Steven P. Sanderson II, MPH

See Also

Other Weibull: tidy_inverse_weibull(), tidy_weibull(), util_weibull_param_estimate()

Other Distribution Statistics: util_bernoulli_stats_tbl(), util_beta_stats_tbl(), util_binomial_stats_tbl(), util_burr_stats_tbl(), util_cauchy_stats_tbl(), util_chisquare_stats_tbl(), util_exponential_stats_tbl(util_f_stats_tbl(), util_gamma_stats_tbl(), util_geometric_stats_tbl(), util_hypergeometric_stats_tbl() util_logistic_stats_tbl(), util_lognormal_stats_tbl(), util_negative_binomial_stats_tbl(), util_normal_stats_tbl(), util_pareto_stats_tbl(), util_poisson_stats_tbl(), util_t_stats_tbl(), util_triangular_stats_tbl(), util_uniform_stats_tbl()

Examples

```
library(dplyr)

tidy_weibull() |>
  util_weibull_stats_tbl() |>
  glimpse()
```

Index