Package 'valueSetCompare'

June 24, 2025

Type Package

Title Comparing HRQoL Instrument Value Sets

Version 1.0.1

Maintainer Kim Rand <krand@mathsinhealth.com>

Description The number of countries with multiple Health Related Quality of Life (HRQL) value sets is growing, and this trend is expected to continue. Each instrument and value set characterizes and values health differently. Identical health states can yield different utility values when valued using different value sets. The 'valueSetCompare' package facilitates comparisons of HRQoL value sets, enabling both theoretical and empirical comparisons. For empirical comparisons, it employs a novel simulation-based method by Jiang et al. (2022) <doi:10.1186/s12955-022-02031-8>, allowing users to investigate the responsiveness of HRQoL instruments across the entire health spectrum using cross-sectional data with external health anchors.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Depends R (>= 3.5)

Imports dplyr, ggplot2, rlang, eq5dsuite

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Kim Rand [aut, cre] (<https://orcid.org/0000-0001-7692-4099>), Anabel Estévez-Carrillo [aut] (<https://orcid.org/0000-0001-8778-5055>)

Contents

`.add_EQ5D_utilities` *.add_EQ5D_utilities*

Description

This function calculates utilities for the provided EQ5D version and value sets.

Usage

```
.add_EQ5D_utilities(
  df,
  value_sets,
  version,
  dim.names = c("mo", "sc", "ua", "pd", "ad"),
  colnames = NULL
)
```

Arguments

df	A data frame to which the utilities will be added.
value_sets	A character vector specifying the country value sets for the given EQ5D version.
version	A character string specifying the EQ5D version. Valid versions are "5L", "3L", "XW", and "XWR".
dim.names	A vector of dimension names to identify dimension columns in the df data frame
colnames	(Optional) A character vector specifying the column names for the added utilities.

Value

The df data frame with the added utility columns.

.calculate_mean_transition *calculate_mean_transition*

Description

This function calculates the mean transitions for EQ-5D health states based on the version specified.

Usage

```
.calculate_mean_transition(df, version, utilityColumn, stateColumn)
```

Arguments

df	Data frame containing the health state and utility data.
version	Character string specifying the EQ-5D version ("5L" or "3L").
utilityColumn	Character string specifying the column name in ‘df‘ containing utility values.
stateColumn	Character string specifying the column name in ‘df‘ containing health state values.

Value

A data frame containing the following columns: - Columns for each transition (e.g., "MOb", "MOw", "SCb", "SCw", etc.) - ‘baseline_HS‘: Baseline health state for each transition. - ‘LSS‘: Level Sum Score for the baseline health state. - ‘baseline_utility‘: Baseline utility value for each transition. - ‘mean_transition‘: Mean of the transitions for each row.

.calculate_quantiles *.calculate_quantiles*

Description

This function calculates the mean, standard deviation, and specified quantiles for each column in a provided data array.

Usage

```
.calculate_quantiles(
  data_array,
  data_margin = 2,
  quantile_levels = c(min = 0, `2.5%` = 0.025, `25%` = 0.25, median = 0.5, `75%` =
    0.75, `97.5%` = 0.975, max = 1)
)
```

Arguments

data_array	A numeric array where calculations will be performed on each column. The array can be 2D or 3D.
data_margin	An integer that indicates the margin on which to apply the function. Default is 2.
quantile_levels	A named numeric vector of probabilities for which quantiles are required.

Value

A data frame with columns for each calculated statistic, including mean, standard deviation, and user-specified quantiles.

.calculate_weighted_statistics *.calculate_weighted_statistics*

Description

This function calculate the mean, standard deviation, and quantiles for each column in a provided data array.

Usage

```
.calculate_weighted_statistics(
  data_array,
  data_margin = 2,
  quantile_levels = c(min = 0, `2.5%` = 0.025, `25%` = 0.25, median = 0.5, `75%` =
    0.75, `97.5%` = 0.975, max = 1)
)
```

Arguments

`data_array`	A data array where calculations will be performed on each column.
`quantile_levels`	A named vector of probabilities for which quantiles are required.

Value

A data frame with columns for each calculated statistic, including mean, standard deviation, upper and lower bounds, and user-specified quantiles.

`.compute_ratios` *.compute_ratios*

Description

The function takes in a data frame and a set of utility variables. For each pair of utility variables, it computes their ratio and adds a new column to the data frame. The new columns are named in the format "F_ratio_Var1_Var2".

Usage

```
.compute_ratios(df, combinations, utility_columns)
```

Arguments

`df`	A data frame containing the utility variables for which ratios are to be computed.
`combinations`	A matrix variable specifying the utility_columns combinations.
`utility_columns`	A character vector specifying the names of the utility variables in the data frame.

Value

A data frame with additional columns corresponding to the computed ratios.

`.create_severity_ribbon_plot` *.create_severity_ribbon_plot*

Description

This function creates a ggplot2 plot with confidence intervals and ribbons for the given data frame.

Usage

```
.create_severity_ribbon_plot(
  df,
  graph_title = "",
  x_axis_title = "",
  y_axis_title = "",
  legend_name = "Type",
  legend_labels = NULL,
  y_axis_limits = c(0.15, 0.95),
  y_min_value = "2.5%",
  y_max_value = "97.5%",
  alpha_1 = 0.5,
  alpha_2 = 0.3,
  linetype_1 = 5,
  linetype_2 = 2,
 color_palette = c("#bebada", "#fb8072", "#8dd3c7", "#80b1d3", "#ffff67", "#fdb462",
    "#b3de69", "#fccde5", "#d9d9d9", "#bc80bd")
)
```

Arguments

df	A data frame containing the data to be plotted. The data frame should have columns for 'MEAN', 'lb', 'ub', 'topval', and 'type'.
graph_title	A string specifying the title of the graph. Default is an empty string.
x_axis_title	A string specifying the title for the x-axis. Default is an empty string.
y_axis_title	A string specifying the title for the y-axis. Default is an empty string.
legend_name	A string specifying the name of the legend. Default is "Type".
legend_labels	A named vector specifying custom labels for the legend. Default is NULL.
y_axis_limits	A numeric vector specifying the limits for the y-axis. Default is c(0.15, 0.95).
y_min_value	A string specifying the column name for the lower bound of the ribbon. Default is "2.5".
y_max_value	A string specifying the column name for the upper bound of the ribbon. Default is "97.5".
alpha_1	A numeric value between 0 and 1 to define the transparency of the interquartile range. Default is 0.15.
alpha_2	A numeric value between 0 and 1 to define the transparency of the confidence interval range. Default is 0.05.
linetype_1	A numeric value between 0 and 1 to define the linetype of the interquartile range. Default is 1 "solid".
linetype_2	A numeric value between 0 and 1 to define the linetype of the confidence interval range. Default 2 "dashed".
color_palette	A character vector specifying the color palette to use for the plot. Default is a set of 10 colors.

Value

A ggplot2 object representing the plot.

.cut_variable .cut_variable

Description

This function cuts a numeric variable into intervals based on the provided breaks.

Usage

```
.cut_variable(variable, breaks)
```

Arguments

variable	A numeric vector that you want to cut into intervals.
breaks	A numeric vector specifying the breakpoints for cutting the variable.

Value

A factor vector representing the intervals into which the variable has been cut.

.extract_columns .extract_columns

Description

This function extracts the specified columns from a data frame for the given samples.

Usage

```
.extract_columns(
  df,
  column_names = c("VAS", "utility_3L", "utility_5L", "utility_xw"),
  sample_indices
)
```

Arguments

df	A data frame from which to extract columns.
column_names	A character vector of the names of the columns to be extracted from df.
sample_indices	An array of sampled indices, each element of the array represents the index of a row in df.

Value

A list of arrays, with each array containing the values of the respective column for the sampled indices. Each array has the same dimensions as sampledIndices. The names of the list elements are the names of column_names.

.factorize_variable *.factorize_variable*

Description

This function factorizes a numeric variable in a data frame based on a specified variant function and optional breaks.

Usage

```
.factorize_variable(df, weight_column, variant_fun, breaks = NULL)
```

Arguments

df	A data frame containing the variable to be factorized.
weight_column	A string specifying the name of the column in the data frame that contains the variable to be factorized.
variant_fun	A function that will be applied to the variable for factorization.
breaks	An optional numeric vector specifying the breakpoints for cutting the variable, if applicable.

Value

A factor or numeric vector representing the factorized variable.

.flatten_group_to_df *.flatten_group_to_df*

Description

This function takes a matrix with an attribute "gr" representing group sizes and flattens it into a data frame, adding a 'gr' column to indicate the group each row belongs to.

Usage

```
.flatten_group_to_df(x)
```

Arguments

x	A matrix containing the data to be flattened. The matrix should have an attribute "gr" that contains the size for each group.

Value

A data frame containing the flattened data with an additional 'gr' column indicating the group each row belongs to.

.f_stat_from_df .f_stat_from_df

Description

The function is designed to compute F-statistic(s) using one-way ANOVA for each numeric column in the provided data frame against a reference column named 'gr'.

Usage

```
.f_stat_from_df(df, include_p = FALSE)
```

Arguments

df	A data frame containing one or more numeric columns and a reference column named 'gr'.
include_p	A logical value indicating whether to include the p-values in the result.

Value

If include_p is FALSE, a named numeric vector of F-statistics for each column against the 'gr' column. If include_p is TRUE, a matrix with two rows containing F-statistics and p-values, respectively.

.gen_samples .gen_samples

Description

This function generates stratified samples from a specified dataframe.

Usage

```
.gen_samples(
  df,
  weight_column = "VAS",
  weight_range = c(0:100),
  weight_values = NULL,
  weight_function = .makeWeightsMixed,
  sample_size = 1000,
  number_of_samples = 1000
)
```

Arguments

df	A data frame from which the samples are to be generated.
weight_column	A string specifying the name of the column to use for weighting.Defaults to "VAS".
weight_range	A numeric vector indicating the range of weight_column, default is 0 to 100.
weight_values	A numeric vector specifying the weight values of interest. Defaults to c(0, 25, 50, 75, 100).
weight_function	A function used to compute weights. Defaults to "makeWeightsTriangular".
sample_size	An integer specifying the size of each sample. Defaults to 1000.
number_of_samples	An integer specifying the number of samples to generate. Defaults to 1000.

Value

An array containing the indices of the sampled rows in the original data frame. The dimensions of the array are determined by the number of weight values and the number of samples.

.gen_samples_proportional

.gen_samples_proportional

Description

The function takes in a dataframe and a column (factor_column) that dictates the groupings. It then generates bootstrap samples ensuring that each sample is proportionally representative of the original dataset based on the given groupings.

Usage

```
.gen_samples_proportional(
  df,
  factor_column = "vasdecile",
  sample_size = 1000,
  number_of_samples = 1000
)
```

Arguments

df	A dataframe containing the dataset.
factor_column	A string specifying the column name in the dataframe that contains the groupings or factors.
sample_size	An integer indicating the size of each bootstrap sample. Default is 1000.
number_of_samples	An integer indicating the number of bootstrap samples to generate. Default is 1.

Value

A matrix containing bootstrap samples with rows corresponding to individual samples and columns corresponding to observations in each sample. The matrix has an attribute "gr" that contains the calculated size for each group to ensure proportional representation.

.get_EQ5D_value_sets *.get_EQ5D_value_sets*

Description

This function generates all EQ states based on the provided EQ5D version and adds utilities.

Usage

```
.get_EQ5D_value_sets(version, value_sets, value_sets_XW)
```

Arguments

version	A character string specifying the EQ5D version.
value_sets	A character vector specifying the country value sets for the given EQ5D version.
value_sets_XW	A character vector specifying the country value sets for the crosswalk EQ5D version.

Value

A list containing:

- utilityColumn: A character vector with the names of the added utility columns.
- df: A data frame containing all generated EQ states with their utilities.

.get_EQ5D_version *.get_EQ5D_version*

Description

This function determines the version of EQ5D used based on the provided stateColumn of a data frame.

Usage

```
.get_EQ5D_version(df, stateColumn)
```

Arguments

df	A data frame containing EQ5D health states.
stateColumn	A character string specifying the name of the column in the df data frame that contains the EQ5D health states.

Value

A character string indicating the version of EQ5D used. Valid return values are "3L" for EQ5D-3L, "5L" for EQ5D-5L, or NULL if the version cannot be determined.

.get_Fstatistics_interpretation *.get_Fstatistics_interpretation*

Description

This function extracts and interprets the results from a given F-statistics error bar plot.

Usage

```
.get_Fstatistics_interpretation(errorbar_plot, utility_combinations = NULL)
```

Arguments

errorbar_plot	A ggplot object containing an error bar plot.
utility_combinations	A matrix or data frame of utility combinations corresponding to the x-axis labels (default is NULL).

Value

A list containing the interpretation results for each combination of types in the F-statistic plot. Each list element contains details about significance, mean, confidence interval.

.get_severity_interpretation *.get_severity_interpretation*

Description

This function extracts and interprets the results from a given severity ribbon plot.

Usage

```
.get_severity_interpretation(
  ribbon_plot,
  quartiles = c(0, 0.25, 0.5, 0.75, 1),
  elevation_threshold = 0.05,
  slope_threshold = 0.02
)
```

Arguments

ribbon_plot	A severity ribbon plot to be interpreted.
quartiles	A numeric vector specifying the quartiles for interpretation. Default is c(0, 0.25, 0.5, 0.75, 1).
elevation_threshold	A numeric value specifying the threshold for elevation differences. Default is 0.05.
slope_threshold	A numeric value specifying the threshold for slope differences. Default is 0.02.

Value

A list containing the interpretation results for each combination of types in the ribbon plot. Each list element contains details about crossing, elevation differences, and slope changes.

`.get_VS_input_list` *.get_VS_input_list*

Description

This function generates a list of input specifications based on provided EQ5D versions and value sets.

Usage

```
.get_VS_input_list(
  value_sets_3L = NULL,
  value_sets_5L = NULL,
  value_sets_XW = NULL,
  value_sets_XWR = NULL,
  value_sets_others = NULL
)
```

Arguments

`value_sets_3L`	(Optional) A character vector specifying the country value sets for EQ5D-3L version.
`value_sets_5L`	(Optional) A character vector specifying the country value sets for EQ5D-5L version.
`value_sets_XW`	(Optional) A character vector specifying the country value sets for EQ5D-XW version.
`value_sets_XWR`	(Optional) A character vector specifying the country value sets for EQ5D-XWR version.
`value_sets_others`	(Optional) A list of lists specifying the inputs for other instruments. Each list within the main list should be named and contain a data frame ("df"), a column specifying the health states ("stateColumn"), and a column specifying the utility values ("utilityColumn").

Value

A list containing input specifications for each provided value set. Each element of the list is a list including three elements: "df", which is a dataframe, "stateColumn" which specifies the name of the column in the "df" dataframe that contains health states, and "utilityColumn" which is the name of the column in the "df" dataframe that contains utility values.

.makeWeights .makeWeights

Description

A function to calculate weights with specified variant function.

Usage

```
.makeWeights(
  x,
  pointval = 100,
  rng = c(0:100),
  variant_fun = .makeWeightsMixed
)
```

Arguments

x	A numeric vector of input values.
pointval	A numeric point value within the range of x, default is 100.
rng	A numeric vector indicating the range of x, default is 0 to 100.
variant_fun	A function to calculate weights. It can be makeWeightsTriangular, makeWeightsGradient, makeWeightsMixed, or a custom function name. Default is makeWeightsTriangular.

Value

A numeric vector of calculated and scaled weights.

.makeWeightsGradient .makeWeightsGradient

Description

A function to calculate weights based on a gradient approach.

Usage

```
.makeWeightsGradient(x, pointval, rng)
```

Arguments

x	A numeric vector of input values.
pointval	A numeric point value within the range of x.
rng	A numeric vector indicating the range of x.

Value

A numeric vector of calculated weights.

`.makeWeightsMixed` *.makeWeightsMixed*

Description

A function to calculate weights based on a mixed approach.

Usage

```
.makeWeightsMixed(x, pointval, rng)
```

Arguments

`x`	A numeric vector of input values.
`pointval`	A numeric point value within the range of x.
`rng`	A numeric vector indicating the range of x.

Value

A numeric vector of calculated weights.

`.makeWeightsTriangular` *.makeWeightsTriangular*

Description

A function to calculate weights based on a triangular approach.

Usage

```
.makeWeightsTriangular(x, pointval, rng)
```

Arguments

`x`	A numeric vector of input values.
`pointval`	A numeric point value within the range of x.
`rng`	A numeric vector indicating the range of x.

Value

A numeric vector of calculated weights.

.merge_adjacent_intervals *.merge_adjacent_intervals*

Description

This function takes a character vector of intervals (e.g., "0-25", "25-50") and merges adjacent intervals.

Usage

```
.merge_adjacent_intervals(intervals)
```

Arguments

intervals	A character vector of intervals to be merged.

Value

A character vector of merged intervals.

.plot_F_statistics *.plot_F_statistics*

Description

This function creates a bar plot for F-statistics along with error bars for specified utility columns.

Usage

```
.plot_F_statistics(
  df,
  utility_columns,
  utility_combinations = NULL,
  graph_title = "",
  x_axis_title = "",
  y_axis_title = "",
  y_min_value = NULL,
  y_max_value = NULL
)
```

Arguments

df	A data frame containing the F-statistics.
utility_columns	A character vector specifying the names of the utility columns in the data frame.
graph_title	A string specifying the title of the graph. Default is an empty string.
x_axis_title	A string specifying the title for the x-axis. Default is an empty string.
y_axis_title	A string specifying the title for the y-axis. Default is an empty string.
y_min_value	A numeric value specifying the minimum limit for the y-axis. Default is NULL.
y_max_value	A numeric value specifying the maximum limit for the y-axis. Default is NULL.

Value

A ggplot object representing the bar plot with error bars.

.write_Fstatistics_interpretation *.write_Fstatistics_interpretation*

Description

This function generates the interpretation of the results obtained from a F-statistics plot.

Usage

```
.write_Fstatistics_interpretation(errorbar_plot, utility_combinations = NULL)
```

Arguments

`errorbar_plot`	A ggplot object containing an error bar plot.
`utility_combinations`	A matrix or data frame of utility combinations corresponding to the x-axis labels (default is NULL).

Value

A character vector containing the interpretation paragraphs for each combination of types in the F-statistic plot.

.write_severity_interpretation *.write_severity_interpretation*

Description

This function generates the interpretation of the results obtained from a severity ribbon plot.

Usage

```
.write_severity_interpretation(
  ribbon_plot,
  quartiles = c(0, 0.25, 0.5, 0.75, 1),
  elevation_threshold = 0.05,
  slope_threshold = 0.02
)
```

Arguments

`ribbon_plot`	A severity ribbon plot to be interpreted.
`quartiles`	A numeric vector specifying the quartiles for interpretation. Default is c(0, 0.25, 0.5, 0.75, 1).
`elevation_threshold`	A numeric value specifying the threshold for elevation differences. Default is 0.05.
`slope_threshold`	A numeric value specifying the threshold for slope differences. Default is 0.02.

Value

A character vector containing the interpretation paragraphs for each combination of types in the ribbon plot.

`cdta` *cdta*

Description

A dataset with 3749 participants from six countries containing EQ-5D-3L and EQ-5D-5L data.

Usage

```
data(cdta)
```

Format

A data frame with 3749 rows and 20 variables:

`conditiongroups` integer An integer variable representing different health conditions. Values range from 1 to 10, where each value corresponds to the following conditions, in order:

1. COPD/Asthma
2. Diabetes
3. Liver disease
4. RA/A
5. CVD
6. Stroke
7. Depression
8. Personality Disorder
9. Other
10. Students

`studyID` integer Identification of the origical study

`profile3L` integer EQ-5D-3L Health state

`profile5L` integer EQ-5D-5L Health state

`respID` integer Participant's ID in the original study

`condition` integer An integer variable representing different health conditions before grouping

`age` integer Participant's age in years

`gender` integer Participant's gender (Female / Male)

`education` integer Participant's educational level

`mobility` integer EQ-5D-3L Mobility dimension

`selfcare` integer EQ-5D-3L Self-Care dimension

`activity` integer EQ-5D-3L Usual activities dimension

`pain` integer EQ-5D-3L Pain/discomfort dimension

`anxiety` integer EQ-5D-3L Anxiety/depression dimension

`VAS` integer Value of the VAS scale measurememnt

`mobility5L` integer EQ-5D-5L Mobility dimension

`selfcare5L` integer EQ-5D-5L Self-Care dimension

`activity5L` integer EQ-5D-5L Usual activities dimension

`pain5L` integer EQ-5D-5L Pain/discomfort dimension

`anxiety5L` integer EQ-5D-5L Anxiety/depression dimension

`compute_F_statistics` *compute_F_statistics*

Description

This function computes F-statistics for specified utility columns in a data frame.

Usage

```
compute_F_statistics(
  df,
  utility_columns,
  utility_combinations = NULL,
  weight_column = "VAS",
  weight_range = c(0:100),
  sample_size = nrow(df),
  number_of_samples = 1000,
  variant_fun = .cut_variable,
  breaks = c(0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100),
  graph_title = "",
  x_axis_title = "",
  y_axis_title = "",
  y_min_value = NULL,
  y_max_value = NULL,
  F_stats_groups = NULL
)
```

Arguments

df	A data frame containing the utility and weight columns.
utility_columns	A character vector specifying the names of utility columns.
utility_combinations	A matrix with two rows indicating the utility columns combinations. Default is all possible combinations of the elements of utility_columns taken 2 at a time.
weight_column	A character string specifying the name of the weight column. Default is "VAS".
weight_range	A numeric vector specifying the range of weights. Default is c(0:100).
sample_size	An integer specifying the sample size for bootstrapping. Default is 1000.
number_of_samples	An integer specifying the number of bootstrap samples. Default is 1000.
variant_fun	A function to be applied for factorizing the weight column. Default is .cut_variable.
breaks	A numeric vector specifying the breaks for the 'cut' method. Default is c(0,10,20,30,40,50,60,70,80,9
graph_title	A character string specifying the title of the plot. Default is an empty string.
x_axis_title	A character string specifying the title for the x-axis. Default is an empty string.
y_axis_title	A character string specifying the title for the y-axis. Default is an empty string.
y_min_value	A numeric value specifying the minimum value for the y-axis. Default is NULL.
y_max_value	A numeric value specifying the maximum value for the y-axis. Default is NULL.
F_stats_groups	An optional data frame of pre-computed group-based F-statistics. Default is NULL.

Value

A list containing two elements: 'df' which is a data frame of weighted statistics, and 'plot' which is the ggplot object representing the ribbon plot.

Examples

```
# Define dimension names for EQ-5D-3L and EQ-5D-5L
dim.names.3L <- c("mobility", "selfcare", "activity", "pain", "anxiety")
dim.names.5L <- c("mobility5L", "selfcare5L", "activity5L", "pain5L", "anxiety5L")
# Compute EQ-5D scores using the eq5dsuite package
cdta$EQ5D3L <- eq5dsuite::eq5d3l(x = cdta,
                                 country = "US",
                                 dim.names = dim.names.3L)
cdta$EQ5D5L <- eq5dsuite::eq5d5l(x = cdta,
                                 country = "US",
                                 dim.names = dim.names.5L)
cdta$EQXW <- eq5dsuite::eqxw(x = cdta,
                             country = "US",
                             dim.names = dim.names.5L)
# Define combinations of utility columns for F-statistics calculation
utility_combinations <- matrix(c("EQ5D5L", "EQ5D3L", "EQ5D5L", "EQXW"), nrow = 2)
# Compute F-statistics for the utility columns
result <- compute_F_statistics(df = cdta,
                               utility_columns = c("EQ5D3L", "EQ5D5L", "EQXW"),
                               utility_combinations = utility_combinations)
# Plot the results
print(result$plot)
```

compute_utility_stats *compute_utility_stats*

Description

Compute utility statistics for given EQ5D versions and value sets.

Usage

```
compute_utility_stats(
  value_sets_3L = NULL,
  value_sets_5L = NULL,
  value_sets_XW = NULL,
  value_sets_XWR = NULL,
  value_sets_others = NULL,
  format_results = FALSE
)
```

Arguments

value_sets_3L	(Optional) A character vector specifying the country value sets for EQ5D-3L version.
value_sets_5L	(Optional) A character vector specifying the country value sets for EQ5D-5L version.
value_sets_XW	(Optional) A character vector specifying the country value sets for EQ5D-XW version.
value_sets_XWR	(Optional) A character vector specifying the country value sets for EQ5D-XWR version.
value_sets_others	(Optional) A list of lists specifying the inputs for other instruments. Each list within the main list should be named and contain a data frame ('df'), a column specifying the health states ('stateColumn'), and a column specifying the utility values ('utilityColumn').
format_results	(Optional) A logical indicating whether the result should be formatted. Default is FALSE.

Value

A data frame containing utility statistics for each provided value set.

Note

Mean single level transitions are calculated only for EQ-5D.

Examples

```
compute_utility_stats(value_sets_3L = "ES", value_sets_5L = "ES")
value_set_other <- list(instrument1 = list(df = data.frame(state=c(1, 2, 3), utility = c(-1, 0, 1)),
                        stateColumn = "state",
                        utilityColumn = "utility"))
compute_utility_stats(value_sets_3L = "AU", value_sets_others = value_set_other)
```

density_plot_empirical *density_plot_empirical*

Description

This function creates a smoothed kernel density plot of the empirical distribution of utility values in a given data frame.

Usage

```
density_plot_empirical(
  df,
  utility_columns,
  graph_title = "",
  x_axis_title = "Index Value",
  x_min_value = NULL,
  x_max_value = NULL,
  y_axis_title = "Density",
  y_min_value = NULL,
  y_max_value = NULL,
  legend_name = "",
  color_palette = NULL,
  line_types = NULL
)
```

Arguments

df	A data frame containing the utility and weight columns.
utility_columns	A character vector specifying the names of utility columns.
graph_title	A character string specifying the title of the graph. Default is an empty string.
x_axis_title	A character string specifying the title of the x-axis. Default is "Index Value".
x_min_value	A numeric specifying the minimum value for the x-axis. Default is NULL.
x_max_value	A numeric specifying the maximum value for the x-axis. Default is NULL.
y_axis_title	A character string specifying the title of the y-axis. Default is "Density".
y_min_value	A numeric specifying the minimum value for the y-axis. Default is NULL.
y_max_value	A numeric specifying the maximum value for the y-axis. Default is NULL.
legend_name	A character string specifying the name of the legend. Default is "".
color_palette	A character vector specifying the colors for the density lines. Default is a predefined color palette.
line_types	A character vector specifying the line types for the density lines. Default is solid.

Value

A ggplot object visualizing the density of utilities for the specified EQ5D versions and other instruments value sets.

Examples

```
dim.names.3L <- c("mobility", "selfcare", "activity", "pain", "anxiety")
cdta$EQ5D3L <- eq5dsuite::eq5d3l(x = cdta,
                                 country = "US",
                                 dim.names = dim.names.3L)
dim.names.5L <- c("mobility5L", "selfcare5L", "activity5L", "pain5L", "anxiety5L")
cdta$EQ5D5L <- eq5dsuite::eq5d5l(x = cdta,
                                 country = "US",
                                 dim.names = dim.names.5L)
cdta$EQXW <- eq5dsuite::eqxw(x = cdta,
                             country = "US",
                             dim.names = dim.names.5L)
density_plot_empirical(df = cdta, utility_columns = c("EQ5D3L", "EQ5D5L", "EQXW"))
```

---

density_plot_theorical

density_plot_theorical

Description

This function creates a smoothed kernel density plot of utilities for different EQ5D versions and specified value sets.

Usage

```
density_plot_theorical(
  value_sets_3L = NULL,
  value_sets_5L = NULL,
  value_sets_XW = NULL,
  value_sets_XWR = NULL,
  value_sets_others = NULL,
  graph_title = "",
  x_axis_title = "Index Value",
  x_min_value = NULL,
  x_max_value = NULL,
  y_axis_title = "Density",
  y_min_value = NULL,
  y_max_value = NULL,
  legend_name = "",
  color_palette = NULL,
  line_types = NULL
)
```

Arguments

value_sets_3L	A character vector specifying the country value sets for the EQ5D-3L version.
value_sets_5L	A character vector specifying the country value sets for the EQ5D-5L version.
value_sets_XW	A character vector specifying the country value sets for the EQ5D-XW version.
value_sets_XWR	A character vector specifying the country value sets for the EQ5D-XWR version.

value_sets_others
: A list of lists specifying the inputs for other instruments. Each list within the main list should be named and contain a data frame (‘df‘), a column specifying the health states (‘stateColumn‘), and a column specifying the utility values (‘utilityColumn‘).

graph_title
: A character string specifying the title of the graph. Default is an empty string.

x_axis_title
: A character string specifying the title of the x-axis. Default is "Index Value".

x_min_value
: A numeric specifying the minimum value for the x-axis. Default is NULL.

x_max_value
: A numeric specifying the maximum value for the x-axis. Default is NULL.

y_axis_title
: A character string specifying the title of the y-axis. Default is "Density".

y_min_value
: A numeric specifying the minimum value for the y-axis. Default is NULL.

y_max_value
: A numeric specifying the maximum value for the y-axis. Default is NULL.

legend_name
: A character string specifying the name of the legend. Default is "".

color_palette
: A character vector specifying the colors for the density lines. Default is a predefined color palette.

line_types
: A character vector specifying the line types for the density lines. Default is solid.

Value

A ggplot object visualizing the density of utilities for the specified EQ5D versions and other instruments value sets.

Examples

```
density_plot_theorical(value_sets_3L = "NL", value_sets_5L = "NL")
instrument <- data.frame(HS=c(123, 456, 789), val = c(-0.3, 0.1, 0.75))
value_set_other <- list(test_instrument = list(df = instrument,
                        stateColumn = "HS",
                        utilityColumn = "val"))
density_plot_theorical(value_sets_3L = "HU", value_sets_others = value_set_other)
```

---

severity_ribbon_plot *severity_ribbon_plot*

Description

This function generates a ribbon plot for given utility columns, based on weighted statistics.

Usage

```
severity_ribbon_plot(
  df,
  utility_columns,
  weight_column = "VAS",
  weight_range = c(0:100),
  weight_values = NULL,
  weight_function = .makeWeightsMixed,
  sample_size = 1000,
```

```
  number_of_samples = 1000,
 probability_levels = c(min = 0, `2.5%` = 0.025, `25%` = 0.25, median = 0.5, `75%` =
    0.75, `97.5%` = 0.975, max = 1),
  graph_title = "",
  x_axis_title = "",
  y_axis_title = "",
  legend_name = "Type",
  legend_labels = NULL,
  y_axis_limits = c(0.15, 0.95),
  y_min_value = "2.5%",
  y_max_value = "97.5%",
  alpha_1 = 0.15,
  alpha_2 = 0.05,
  linetype_1 = 1,
  linetype_2 = 2,
  interpretation_quartiles = c(0, 0.25, 0.5, 0.75, 1),
  elevation_threshold = 0.05,
  slope_threshold = 0.02,
  color_palette = NULL,
  weighted_statistics = NULL
)
```

Arguments

df	A data frame containing the utility and weight columns.
utility_columns	A character vector specifying the utility columns for which the ribbon plot will be generated.
weight_column	A string specifying the column that contains the weights. Default is "VAS".
weight_range	A numeric vector specifying the range of weights. Default is c(0:100).
weight_values	A numeric vector specifying the weight values to be used. Default is NULL, in which case the weight_range will be used.
weight_function	A function to generate weights. Default is .makeWeightsMixed.
sample_size	An integer specifying the sample size for bootstrapping. Default is 1000.
number_of_samples	An integer specifying the number of bootstrap samples. Default is 1000.
probability_levels	A named vector specifying the probability levels for quantile.
graph_title	A string specifying the title of the graph. Default is an empty string.
x_axis_title	A string specifying the title for the x-axis. Default is an empty string.
y_axis_title	A string specifying the title for the y-axis. Default is an empty string.
legend_name	A string specifying the name for the legend. Default is "Type".
legend_labels	A character vector specifying the labels for the legend. Default is NULL.
y_axis_limits	A numeric vector specifying the limits for the y-axis. Default is c(0.15, 0.95).
y_min_value	A string specifying the minimum value for the y-axis.
y_max_value	A string specifying the maximum value for the y-axis.

`alpha_1` A numeric value between 0 and 1 to define the transparency of the interquartile range. Default is 0.15.

`alpha_2` A numeric value between 0 and 1 to define the transparency of the confidence interval range. Default is 0.05.

`linetype_1` A numeric value between 0 and 1 to define the line type of the interquartile range. Default is 1 "solid".

`linetype_2` A numeric value between 0 and 1 to define the line type of the confidence interval range. Default 2 "dashed".

`interpretation_quartiles`
A numeric vector of values between 0 and 1 to define the quantile ranges for the figure interpretation. Default is c(0, 0.25, 0.5, 0.75, 1)

`elevation_threshold`
A numeric value specifying the threshold for elevation differences. Default is 0.05.

`slope_threshold`
A numeric value specifying the threshold for slope differences. Default is 0.02.

`color_palette` A character vector specifying the color palette for the plot. Default is c("#8dd3c7", "#bebada", "#80b1d3", "#fb8072", "#ffff67", "#fdb462", "#b3de69", "#fccde5", "#d9d9d9", "#bc80bd").

`weighted_statistics`
An optional data frame of pre-computed weighted statistics. Default is NULL.

Value

A list containing three elements: 'df' which is a data frame of weighted statistics, 'plot' which is the ggplot object representing the ribbon plot and 'interpretation' with the automatic interpretation of the ribbon plot.

Examples

```
# Define dimension names for EQ-5D-3L and EQ-5D-5L
dim.names.3L <- c("mobility", "selfcare", "activity", "pain", "anxiety")
dim.names.5L <- c("mobility5L", "selfcare5L", "activity5L", "pain5L", "anxiety5L")
# Compute EQ-5D scores using the eq5dsuite package
cdta$EQ5D3L <- eq5dsuite::eq5d3l(x = cdta,
                                 country = "US",
                                 dim.names = dim.names.3L)
cdta$EQ5D5L <- eq5dsuite::eq5d5l(x = cdta,
                                 country = "US",
                                 dim.names = dim.names.5L)
cdta$EQXW <- eq5dsuite::eqxw(x = cdta,
                             country = "US",
                             dim.names = dim.names.5L)
# Get severity ribbon plot
result <- severity_ribbon_plot(df = cdta, utility_columns = c("EQ5D3L", "EQ5D5L", "EQXW"))
```

single_transition_plots *single_transition_plots*

Description

This function creates a scatter plot of mean one-level transitions for different EQ5D versions and specified value sets.

Usage

```
single_transition_plots(
  value_sets_3L = NULL,
  value_sets_5L = NULL,
  value_sets_XW = NULL,
  value_sets_XWR = NULL,
  graph_title = "",
  x_axis_title = NULL,
  x_min_value = -1,
  x_max_value = 1,
  y_axis_title = "Mean 1-level transition",
  y_min_value = 0,
  y_max_value = 0.5,
  legend_name = NULL
)
```

Arguments

value_sets_3L	A character vector specifying the country value sets for the EQ5D-3L version.
value_sets_5L	A character vector specifying the country value sets for the EQ5D-5L version.
value_sets_XW	A character vector specifying the country value sets for the EQ5D-XW version.
value_sets_XWR	A character vector specifying the country value sets for the EQ5D-XWR version.
graph_title	A character string specifying the title of the graph. Default is an empty string.
x_axis_title	A character string specifying the title of the x-axis. Default is "Utility".
x_min_value	A numeric specifying the minimum value for the x-axis. Default is -1.
x_max_value	A numeric specifying the maximum value for the x-axis. Default is 1.
y_axis_title	A character string specifying the title of the y-axis. Default is "Mean 1-level transition".
y_min_value	A numeric specifying the minimum value for the y-axis. Default is 0.
y_max_value	A numeric specifying the maximum value for the y-axis. Default is 0.5.
legend_name	A character string specifying the name of the legend. Default is NULL.

Value

A list of ggplot objects visualizing the mean one-level transitions for the specified EQ5D versions and value sets.

Note

This function is primarily intended to work with EQ5D data and it is not be applicable to other instruments.

Examples

```
single_transition_plots(value_sets_5L = "IT")
single_transition_plots(value_sets_3L = c("JP", "US"))
single_transition_plots(value_sets_3L ="ES",
                        value_sets_5L = "ES",
                        value_sets_XW = "ES",
                        value_sets_XWR = "ES")
```

Index