Package 'scaledescr'

November 6, 2026

Type Package

Title Descriptive, Reliability, and Inferential Tables for Psychometric Scales and Demographic Data

Version 0.2.9

Description Wraps and formats outputs from established statistical and psychometric functions into compact tables and summaries designed for straightforward export to CSV, Word, and Excel formats. Core utilities support psychometric scale preparation, reliability assessment, factorability checks, structural equation and mediation analyses, inferential tests, and demographic summaries. The package does not introduce new statistical methods or estimation procedures; instead, it serves as a reporting wrapper around packages such as 'psych', 'stats', 'lavaan', and 'gtsummary' to facilitate everyday data export workflows in clinical and psychological research. It is intended to lower the barrier to entry for researchers transitioning into the R ecosystem, with function names structured to be intuitive and self-descriptive.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports dplyr, gtsummary, purrr, rlang, stats, officer, openxlsx, utils, psych, stringr, flextable, lavaan, performance

Suggests testthat (>= 3.0.0), webshot2

Config/testthat/edition 3

NeedsCompilation no

Author Darshankumar Dharaiya [aut, cre] (ORCID: <https://orcid.org/0009-0009-9351-8602>)

Maintainer Darshankumar Dharaiya <dharaiya.darshan@gmail.com>

LazyData true

Config/roxygen2/version 8.1.0

Repository CRAN

Date/Publication 2026-11-06 17:40:02 UTC

Contents

compute_ICC . . . 2
counterbalance . . . 4
make_alphanumeric_conversion . . . 6
make_alpha_table . . . 7
make_chisq_test_table . . . 8
make_citation_to_output . . . 9
make_correlation_table . . . 10
make_dataframe_to_output . . . 11
make_demographic_table . . . 13
make_demographic_table_to_output . . . 14
make_EFA_factorability_table . . . 15
make_independent_t_test_table . . . 16
make_item_total_correlation_table . . . 17
make_lavaan_analysis_tables . . . 18
make_lavaan_mediation_tables . . . 19
make_normality_table . . . 20
make_one_sample_t_test_table . . . 20
make_paired_t_test_table . . . 21
make_reverse_score . . . 23
make_scale_description_table . . . 23
make_scale_total . . . 24
merge_counterbalanced_columns . . . 25
scaledescr . . . 26
scaledescr_remove_column . . . 27

Index **28**

`compute_ICC` *Compute Intraclass Correlation Coefficient (ICC) for Selected Columns*

Description

Computes Intraclass Correlation Coefficients (ICC) for a specified set of numeric variables within a data frame using `psych::ICC()`.

This function standardizes column selection and basic validation while delegating all statistical estimation to `psych::ICC()`.

Usage

```
compute_ICC(data, items, check_numeric = TRUE, lmer = FALSE, ...)
```

Arguments

`data`	A data frame containing the ratings or measurements.
`items`	A character vector specifying column names for which ICC should be calculated (e.g., raters or repeated measurements).
`check_numeric`	Logical. If `TRUE` (default), ensures all selected columns are numeric.
`lmer`	Logical. Passed to `psych::ICC()`. Default is FALSE.
`...`	Additional arguments passed to `psych::ICC()`.

Details

The function does not choose or filter specific ICC models. All available ICC types (e.g., ICC1, ICC2, ICC3, single and average) are returned exactly as produced by `psych::ICC()`.

Users are responsible for:

- Choosing the appropriate ICC model for their study design.
- Ensuring assumptions (e.g., continuous ratings, independence).
- Handling missing data appropriately.

Value

An object returned by `psych::ICC()`, typically a list containing a results table with ICC estimates, F statistics, confidence intervals, and model details.

Warning

Do not pass all items of a multi-item scale simultaneously for test-retest or cross-language equivalence. Passing 18 columns (e.g., 9 English and 9 Gujarati items) treats every item as an interchangeable rater (`Judges = 18`). This confounds item-specific variance with language differences, yielding misleading results. For pairwise equivalence, evaluate items individually (`Judges = 2`) or sum/average them into composite scores first.

Note

For test-retest, cross-language, or rater-agreement designs across multi-item scales, do **not** pass all items simultaneously (e.g., all English and Gujarati items together). Doing so treats every individual item as an interchangeable rater/judge, pooling item-level variance with rater variance.

Instead:

- For **item-level equivalence**: compute ICC separately for each matched pair (2 judges per call; e.g., `phq_e_1` vs. `phq_g_1`).
- For **scale-level equivalence**: compute the composite or total scores first, then pass the two total score columns (2 judges; e.g., `phq_total_e` vs. `phq_total_g`).

See Also

`ICC`

Examples

```
data(counterbalance)

# Step 1: Prepare merged columns across counterbalanced orders
df <- merge_counterbalanced_columns(
  data        = counterbalance,
  col_1       = paste0("phq_e_1_", 1:9),
  col_2       = paste0("phq_e_2_", 1:9),
  new_columns = paste0("phq_e_", 1:9)
)

df <- merge_counterbalanced_columns(
  data        = df,
  col_1       = paste0("phq_g_1_", 1:9),
  col_2       = paste0("phq_g_2_", 1:9),
  new_columns = paste0("phq_g_", 1:9)
)

# ------------------------------------------------------------
# CORRECT USE 1: Item-level cross-language equivalence (Judges = 2)
# ------------------------------------------------------------
compute_ICC(df, items = c("phq_e_1", "phq_g_1"))

# ------------------------------------------------------------
# CORRECT USE 2: Scale-level cross-language equivalence (Judges = 2)
# ------------------------------------------------------------
df$phq_total_e <- rowSums(df[, paste0("phq_e_", 1:9)])
df$phq_total_g <- rowSums(df[, paste0("phq_g_", 1:9)])
compute_ICC(df, items = c("phq_total_e", "phq_total_g"))

# ------------------------------------------------------------
# INCORRECT USE: What NOT to do for cross-language equivalence using `compute_ICC()` function!
# ------------------------------------------------------------
# Passing all 18 columns simultaneously treats each question as an
# interchangeable judge (Judges = 18), conflating internal consistency
# with cross-language agreement and producing misleading ICC estimates:
## Not run:
compute_ICC(df, items = c(paste0("phq_e_", 1:9), paste0("phq_g_", 1:9)))

## End(Not run)
```

`counterbalance` *Counterbalanced PHQ-9 Example Data*

Description

A small dummy dataset simulating a counterbalanced administration of the PHQ-9 in English and Gujarati, as exported from platforms such as PsyToolkit. Each participant completed the English and Gujarati versions in one of two orders, so the unused branch for each language is `NA`.

Usage

```
counterbalance
```

Format

A data frame with 8 rows and 39 variables:

psy_group Counterbalancing group: 1 = English first, then Gujarati; 2 = Gujarati first, then English.

age Age in years.

gender Gender: 1 = male, 2 = female.

phq_e_1_1 English PHQ-9 item 1, Order 1 (group 1), scored 0 to 3.

phq_e_1_2 English PHQ-9 item 2, Order 1 (group 1), scored 0 to 3.

phq_e_1_3 English PHQ-9 item 3, Order 1 (group 1), scored 0 to 3.

phq_e_1_4 English PHQ-9 item 4, Order 1 (group 1), scored 0 to 3.

phq_e_1_5 English PHQ-9 item 5, Order 1 (group 1), scored 0 to 3.

phq_e_1_6 English PHQ-9 item 6, Order 1 (group 1), scored 0 to 3.

phq_e_1_7 English PHQ-9 item 7, Order 1 (group 1), scored 0 to 3.

phq_e_1_8 English PHQ-9 item 8, Order 1 (group 1), scored 0 to 3.

phq_e_1_9 English PHQ-9 item 9, Order 1 (group 1), scored 0 to 3.

phq_g_1_1 Gujarati PHQ-9 item 1, Order 1 (group 2), scored 0 to 3.

phq_g_1_2 Gujarati PHQ-9 item 2, Order 1 (group 2), scored 0 to 3.

phq_g_1_3 Gujarati PHQ-9 item 3, Order 1 (group 2), scored 0 to 3.

phq_g_1_4 Gujarati PHQ-9 item 4, Order 1 (group 2), scored 0 to 3.

phq_g_1_5 Gujarati PHQ-9 item 5, Order 1 (group 2), scored 0 to 3.

phq_g_1_6 Gujarati PHQ-9 item 6, Order 1 (group 2), scored 0 to 3.

phq_g_1_7 Gujarati PHQ-9 item 7, Order 1 (group 2), scored 0 to 3.

phq_g_1_8 Gujarati PHQ-9 item 8, Order 1 (group 2), scored 0 to 3.

phq_g_1_9 Gujarati PHQ-9 item 9, Order 1 (group 2), scored 0 to 3.

phq_e_2_1 English PHQ-9 item 1, Order 2 (group 2), scored 0 to 3.

phq_e_2_2 English PHQ-9 item 2, Order 2 (group 2), scored 0 to 3.

phq_e_2_3 English PHQ-9 item 3, Order 2 (group 2), scored 0 to 3.

phq_e_2_4 English PHQ-9 item 4, Order 2 (group 2), scored 0 to 3.

phq_e_2_5 English PHQ-9 item 5, Order 2 (group 2), scored 0 to 3.

phq_e_2_6 English PHQ-9 item 6, Order 2 (group 2), scored 0 to 3.

phq_e_2_7 English PHQ-9 item 7, Order 2 (group 2), scored 0 to 3.

phq_e_2_8 English PHQ-9 item 8, Order 2 (group 2), scored 0 to 3.

phq_e_2_9 English PHQ-9 item 9, Order 2 (group 2), scored 0 to 3.

phq_g_2_1 Gujarati PHQ-9 item 1, Order 2 (group 1), scored 0 to 3.

phq_g_2_2 Gujarati PHQ-9 item 2, Order 2 (group 1), scored 0 to 3.

phq_g_2_3 Gujarati PHQ-9 item 3, Order 2 (group 1), scored 0 to 3.

phq_g_2_4 Gujarati PHQ-9 item 4, Order 2 (group 1), scored 0 to 3.

phq_g_2_5 Gujarati PHQ-9 item 5, Order 2 (group 1), scored 0 to 3.

phq_g_2_6 Gujarati PHQ-9 item 6, Order 2 (group 1), scored 0 to 3.

phq_g_2_7 Gujarati PHQ-9 item 7, Order 2 (group 1), scored 0 to 3.

phq_g_2_8 Gujarati PHQ-9 item 8, Order 2 (group 1), scored 0 to 3.

phq_g_2_9 Gujarati PHQ-9 item 9, Order 2 (group 1), scored 0 to 3.

Source

Simulated dummy data for demonstrating `merge_counterbalanced_columns`.

`make_alphanumeric_conversion`

Convert Between Alphabetical and Numeric Responses

Description

Converts specified columns between alphabetical (text) and numeric values. Can either overwrite existing columns or create new ones.

Usage

```
make_alphanumeric_conversion(
  data,
  column_vars,
  from_values,
  to_values,
  new_names = NULL,
  case_sensitive = FALSE
)
```

Arguments

`data`	A data frame.
`column_vars`	Character vector of column names to convert.
`from_values`	Vector of values to replace (character or numeric).
`to_values`	Vector of replacement values (must be same length as from_values).
`new_names`	Optional character vector of new column names. If NULL (default), original columns are overwritten.
`case_sensitive`	Logical. Only relevant when converting character values. Default is FALSE.

Details

Factors are automatically converted to character before matching. Unmatched values become NA with a warning.

Value

A data frame with converted values.

`make_alpha_table` *Wrap a pre computed psych::alpha object into a single row table with N and alpha only.*

Description

Extracts only the Scale Name, Sample Size (N), and Cronbach's Alpha from a 'psych::alpha' object.

Usage

```
make_alpha_table(alpha_res, scale_name = "Scale")
```

Arguments

`alpha_res`	A psych::alpha object (already computed)
`scale_name`	Name of the scale (default: "Scale")

Value

A data frame with three columns: Scale,N, and Alpha.

Examples

```
## Not run:
library(psych)
res <- psych::alpha(mtcars[,1:3])
make_alpha_table(res, scale_name = "Car Scale")

## End(Not run)
```

`make_chisq_test_table` *Create a one-row summary table for a chi-square test*

Description

This function formats the result of a pre-computed `chisq.test()` into a single-row data frame. It supports goodness-of-fit, independence, and homogeneity chi-square tests and includes an appropriate effect size with a qualitative interpretation.

Usage

```
make_chisq_test_table(
  chisq_object,
  test_type = c("independence", "goodness-of-fit", "homogeneity"),
  digits = 3
)
```

Arguments

`chisq_object`	An object or list of objects of class `"htest"` produced by `stats::chisq.test()`.
`test_type`	Character string specifying the type of chi-square test. One of `"goodness-of-fit"`, `"independence"`, or `"homogeneity"`. for multiple object, if separate test_type are not provided than given test type will be uesd for all chi square test./
`digits`	Integer indicating the number of decimal places to round to.

Details

The function does not perform the chi-square test itself and does not introduce new statistical methods. All test statistics are extracted directly from the supplied `chisq.test()` object.

For goodness-of-fit tests, Cohen's w is reported. For tests of independence and homogeneity, Cramér's V is reported. Effect size interpretations follow conventional benchmarks (0.10 = small, 0.30 = medium, 0.50 = large).

Value

A single-row data frame with the following columns:

- `test`: Type of chi-square test
- `chi_square`: Chi-square statistic
- `df`: Degrees of freedom
- `p_value`: p-value
- `N`: Total sample size
- `effect_size`: Effect size (Cohen's w or Cramér's V)
- `effect_type`: Type of effect size reported
- `effect_interpretation`: Qualitative interpretation of effect size

Examples

```
# Goodness-of-fit example
observed <- c(40, 30, 50)
chisq_gof <- chisq.test(observed)

make_chisq_test_table(
  chisq_object = chisq_gof,
  test_type = "goodness-of-fit"
)

# Independence test example
tbl <- matrix(c(20, 30, 10, 40), nrow = 2)
chisq_ind <- chisq.test(tbl)

make_chisq_test_table(
  chisq_object = chisq_ind,
  test_type = "independence"
)
```

`make_citation_to_output` *Export Session Package Citations to Multiple Formats*

Description

This function identifies all currently loaded external packages in the current R session, along with base R, and extracts their citation information. It generates both plain text and BibTeX formats and exports them using the package's internal export utility.

Usage

```
make_citation_to_output(
  filename = "Session_Citations",
  format = "csv",
  path = NULL,
  packages = NULL
)
```

Arguments

`filename`	A character string specifying the name of the output file. Defaults to "Session_Citations".
`format`	A character string specifying the output format: "csv", "word", "excel", or "pdf". Defaults to "csv".
`path`	A character string specifying the directory path for the output. If NULL (default), it uses a temporary directory. Use "getwd()" for the current working directory.

packages	A character vector of package names to extract citations for. If NULL (default), citations are extracted for all currently loaded packages in the session, including base R. Example: packages = c("lavaan", "psych").

Value

The file path of the created document (invisibly).

Examples

```
## Not run:
# Export citations for currently loaded packages to a docs
make_citation_to_output(filename = "My_References", format = "word")

# Export citations for selected packages to a docs in current working directory
make_citation_to_output( packages = c("base","psych","ggplot2"),
                         filename = "Selected_References",
                         format = "word",
                         path = getwd())


# Export to Excel in the current working directory
make_citation_to_output(format = "excel", path = "getwd()")


## End(Not run)
```

---

make_correlation_table

Make Correlation Table

Description

Computes a correlation matrix with inline significance stars.

Usage

```
make_correlation_table(data, vars = NULL, method = "pearson")
```

Arguments

data	A data frame.
vars	Character vector of variable names. If NULL, all columns are used.
method	Correlation method: "pearson" (default), "spearman", or "kendall".

Value

A character data frame with correlations and significance stars.

make_dataframe_to_output *Export a data frame to CSV, Word (.docx),Excel (.xlsx) or pdf format.*

Description

If no output path is specified, the file is written to a temporary directory using ‘tempdir()‘. For reproducible workflows, users are encouraged to explicitly specify an output location, either the current working directory or a full file path.

Usage

```
make_dataframe_to_output(data, filename = NULL, format = "csv", path = NULL)
```

Arguments

data	A data frame to export.
filename	Optional base file name (without extension). Defaults to object name. (without path and without extension). If not provided, the name of the input object is used.
format	Character string specifying the output format. One of ‘"csv"‘, ‘"word"‘,‘"excel"‘ or ‘"pdf"‘. Default is ‘"csv"‘.
path	Optional character string specifying the directory where the file should be saved. If ‘NULL‘ (default), the file is written to a temporary directory. Use ‘path = getwd()‘ to save the file in the current working directory, or provide a full directory path.

Details

- CSV files are written using ‘utils::write.csv()‘ - Word files (‘docx‘) are created using the ‘officer‘ package - Excel (‘xlsx‘) files are created using the ‘openxlsx‘ package - PDF files are created using the ’flextable’ package

Value

Invisibly returns the full file path to the generated output file.

Note

Exporting to PDF requires the ’webshot2’ package to be installed.

Examples

```
## Not run:

# Example dataset available in base R
data_df <- head(mtcars)

#-------------------------------------------------
# 1. Simplest use: export to CSV in temp directory
#-------------------------------------------------
make_dataframe_to_output(data_df)

#-------------------------------------------------
# 2. Specify filename
#-------------------------------------------------
make_dataframe_to_output(
  data = data_df,
  filename = "mtcars_sample"
)

#-------------------------------------------------
# 3. Export as Word document
#-------------------------------------------------
make_dataframe_to_output(
  data = data_df,
  filename = "mtcars_word_table",
  format = "word"
)

#-------------------------------------------------
# 4. Export as Excel file
#-------------------------------------------------
make_dataframe_to_output(
  data = data_df,
  filename = "mtcars_excel_table",
  format = "excel"
)

#-------------------------------------------------
# 5. Save to current working directory
#-------------------------------------------------
make_dataframe_to_output(
  data = data_df,
  filename = "mtcars_current_folder",
  format = "csv",
  path = "getwd()"
)

#-------------------------------------------------
# 6. Save Excel file to current working directory
#-------------------------------------------------
make_dataframe_to_output(
  data = data_df,
```

```
  filename = "mtcars_excel_current",
  format = "excel",
  path = "getwd()"
)

#-------------------------------------------------
# 7. Export another base dataset (iris)
#-------------------------------------------------
make_dataframe_to_output(
  data = head(iris),
  filename = "iris_sample",
  format = "word"
)

#-------------------------------------------------
# 8. Using a custom folder path
#-------------------------------------------------
make_dataframe_to_output(
  data = head(airquality),
  filename = "airquality_data",
  format = "excel",
  path = "D:/output_folder"
)


## End(Not run)
```

`make_demographic_table` *Create a demographics summary table*

Description

Create a demographics summary table

Usage

```
make_demographic_table(data, vars, continuous_vars = NULL)
```

Arguments

`data`	A data frame
`vars`	demographic variables to include in the table
`continuous_vars`	Optional subset of vars to be treated as continuous

Value

A gtsummary table

Examples

```
df <- data.frame(
  age = c("25", "30 years", "35", " 40 ", "22.5", "28+", NA, ""),
  sex = c("M", "F", "m", "f", " M ", "F", "m", NA),
  education = c("HS", "BA", "MA", "ma", "Hs", "Ma", "Ba Bed", "Msc bed ")
)

# Generate a demographic summary table (assign to object to avoid printing)
demo_table <- make_demographic_table(df, vars = c("age", "sex", "education"))
demo_table # optionally inspect the table
```

`make_demographic_table_to_output`
Create and Export Demographic Summary Table

Description

Generates a demographic summary table using ‘make_demographic_table()‘ and exports the resulting table to Word, Excel, or CSV format.

This is a convenience wrapper that combines analysis and output in a single step for users who want immediate file export.

Usage

```
make_demographic_table_to_output(
  data,
  vars,
  continuous_vars = NULL,
  file_name = "demographic_table",
  format = c("word", "excel", "csv"),
  path = NULL
)
```

Arguments

`data`	A data frame containing the dataset.
`vars`	Character vector of variable names to include in the demographic summary.
`continuous_vars`	Optional character vector specifying which variables should be treated as continuous. If ‘NULL‘, numeric variables are automatically detected.
`file_name`	Name of the output file WITHOUT extension (e.g., "demographics").
`format`	Output format. Must be one of: • ‘"word"‘ - exports to a Word document (.docx) • ‘"excel"‘ - exports to an Excel file (.xlsx)

- ‘"csv"‘ - exports to a CSV file (.csv)

`path` Optional character string specifying the directory where the file should be saved. If ‘NULL‘ (default), the file is written to a temporary directory. Provide a full directory path or use getwd() during the function call.

Details

The demographic table includes:

- Categorical variables reported as n (
- Continuous variables reported as Mean (SD)

Word export uses ‘flextable‘ and ‘officer‘ for formatting. Excel export uses ‘openxlsx‘. CSV export uses base R ‘write.csv()‘.

Value

Invisibly returns the full file path to the generated output file.

`make_EFA_factorability_table` *Check EFA Factorability and Return Summary Table*

Description

This function calculates the overall Kaiser-Meyer-Olkin (KMO) MSA, Bartlett's Test of Sphericity, and the Determinant of the correlation matrix.

Usage

```
make_EFA_factorability_table(data, items, scale_name = "Overall")
```

Arguments

`data`	A data frame containing the variables.
`items`	A character vector of column names to be included in the analysis.
`scale_name`	Character. A label for the scale (e.g., "GLPFS"). Defaults to "Overall".

Value

A data frame with one row containing the scale-level factorability diagnostics.

`make_independent_t_test_table` *Create a one-row summary table for an independent-samples test*

Description

This function performs an independent-samples test (Welch's t-test, Student's t-test, or Mann-Whitney U test) between two groups defined by a binary grouping variable and returns a single-row data frame. The output includes group names, sample sizes, mean difference, test statistics, p-value, and effect size (Cohen's d) with a qualitative interpretation.

Usage

```
make_independent_t_test_table(
  data,
  outcome,
  group,
  test_type = c("welch", "student", "mann-whitney")
)
```

Arguments

`data`	A data frame containing the outcome and grouping variables.
`outcome`	Character string specifying the numeric outcome variable.
`group`	Character string specifying the grouping variable. Must have exactly two levels.
`test_type`	Character string specifying the test method: `"welch"` (default), `"student"`, or `"mann-whitney"`.

Details

The function is intended for streamlined reporting and does not introduce new statistical methods. Computations rely on `stats::t.test()` or `stats::wilcox.test()`.

Welch's t-test is used by default, which does not assume equal variances. Cohen's d is computed using the pooled standard deviation for comparability with conventional benchmarks. Group ordering follows the factor level order of the grouping variable.

Value

A single-row data frame with the following columns:

- `test`: Name of the statistical test
- `group1`, `group2`: Group labels
- `mean_diff`: Mean difference between groups (group1 - group2)
- `t_value`: Test statistic (t-value or W-statistic for Mann-Whitney)
- `df`: Degrees of freedom (returns `NA` for Mann-Whitney)

- p_value: p-value
- n_group1, n_group2: Sample sizes per group
- cohens_d: Cohen's d effect size
- interpretation: Qualitative interpretation of effect size

Examples

```
set.seed(123)

data_t <- data.frame(
  group = rep(c("CBT", "Psychodynamic"), each = 30),
  score = c(
    rnorm(30, mean = 18, sd = 4),
    rnorm(30, mean = 21, sd = 4)
  )
)

# Legacy call (defaults to welch)
make_independent_t_test_table(data = data_t, outcome = "score", group = "group")

# New explicitly defined calls
make_independent_t_test_table(data_t, "score", "group", test_type = "student")
make_independent_t_test_table(data_t, "score", "group", test_type = "mann-whitney")
```

---

make_item_total_correlation_table

Generate a Tidy Item-Total Correlation and Scale Reliability Table

---

Description

'make_item_total_correlation_table' extracts a specified subset of items from a dataset, executes an item-level psychometric reliability analysis using the \(performance\) package, and returns a clean, publication-ready data frame. It also prints overall scale reliability metrics to the console.

Usage

```
make_item_total_correlation_table(data, columns)
```

Arguments

data	A data frame containing the questionnaire items or scale vectors to analyze.
columns	A character vector specifying the exact column names of the items to include in the scale analysis (e.g.\(, paste0("phq_e_", 1:9)\)).

Details

The function wraps \(performance::item_reliability()\) to isolate relevant metrics for psychometric scale validation (such as translating or adapting existing instruments). It strips away complex package metadata to output a standard, easily exportable data frame.

The returned data frame contains:

- \(Item:\) The name of each questionnaire item.
- \(Alpha_If_Deleted:\) The scale's overall Cronbach's Alpha if that specific item is removed.
- \(Total_Correlation:\) The uncorrected item-total correlation (includes the item itself).
- \(Discrimination:\) The corrected item-total correlation (excludes the target item). Values > 0.30 indicate good item cohesion.

Value

A standard data frame containing item-level statistics (Item, Alpha_If_Deleted, Total_Correlation, and Discrimination).

Examples

```
## Not run:
# Load required package data
data("counterbalance", package = "scaledescr")

# Example A: Passing a pre-defined variable vector
phq_cols <- paste0("phq_e_", 1:9)
make_item_total_correlation_table(data = counterbalance, columns = phq_cols)

# Example B: Directly inserting the column generation pattern
make_item_total_correlation_table(data = counterbalance, columns = paste0("phq_e_", 1:9))

## End(Not run)
```

---

make_lavaan_analysis_tables

Comprehensive Lavaan Regression Analysis

Description

Fits a lavaan model and returns a list containing the regression coefficients table and the global model fit indices.

Usage

```
make_lavaan_analysis_tables(data, dvs, ivs, standardize = FALSE)
```

Arguments

data	A dataframe.
dvs	A character vector of one or more Dependent Variables.
ivs	A character vector of one or more Independent Variables.
standardize	Logical. If TRUE, standardizes data before analysis.

Value

A list with two dataframes: 'Regression' and 'Fit'.

---

make_lavaan_mediation_tables

Comprehensive Lavaan Mediation Analysis with Optional Covariances

---

Description

Fits a lavaan mediation model with one or more mediators and optionally allows covariances between IVs, mediators, and DVs. Returns direct effects, indirect effects, total effects, correlations, and fit indices.

Usage

```
make_lavaan_mediation_tables(
  data,
  dvs,
  ivs,
  mediators,
  bootstrap = 1000,
  standardize = FALSE,
  correlate_ivs = FALSE,
  correlate_mediators = FALSE,
  correlate_dvs = FALSE
)
```

Arguments

data	A dataframe.
dvs	A character vector of one or more Dependent Variables.
ivs	A character vector of one or more Independent Variables.
mediators	A character vector of one or more Mediator Variables.
bootstrap	Integer. Number of bootstrap samples. Default 1000.
standardize	Logical. If TRUE, standardizes data before analysis.
correlate_ivs	Logical. If TRUE, adds covariances between IVs.
correlate_mediators	Logical. If TRUE, adds covariances between mediators.
correlate_dvs	Logical. If TRUE, adds covariances between DVs.

Value

Invisibly returns a list: Direct, Indirect, Total, Correlations, Fit, R2

---

make_normality_table *Make Normality Table*

Description

Computes Shapiro-Wilk and Kolmogorov-Smirnov normality tests for specified variables.

Usage

```
make_normality_table(data, vars = NULL)
```

Arguments

data	A data frame.
vars	Character vector of variable names. If NULL, all columns are used.

Value

A data frame with W/D statistics and p-values for both tests.

---

make_one_sample_t_test_table *Create a Summary Table for a One-Sample t-test*

Description

Create a Summary Table for a One-Sample t-test

Usage

```
make_one_sample_t_test_table(t_res, variable_label = "Variable", digits = 3)
```

Arguments

t_res	An object of class "htest" produced by `t.test()`,or a list of such objects.
variable_label	A character string to label the tested variable(s).
digits	Integer indicating the number of decimal places to round to.

Value

A data frame summarizing the one-sample t-test results.

Examples

```
data("attitude")
A <- as.data.frame(attitude)

# Example 1: One-sample t-test with default mu = 0
t1 <- t.test(A$rating)
make_one_sample_t_test_table(
  t_res = t1,
  variable_label = "Rating"
)

# Example 2: One-sample t-test with specified mu
t2 <- t.test(A$rating, mu = 60)
make_one_sample_t_test_table(
  t_res = t2,
  variable_label = "Rating"
)

# Example 3: Multiple one-sample t-tests combined into one table
t1 <- t.test(A$rating)
t2 <- t.test(A$learning)
t3 <- t.test(A$raises)

make_one_sample_t_test_table(
  t_res = list(t1, t2, t3),
  variable_label = c("Rating", "Learning", "Raises")
)

# Example 4: Multiple tests with different mu values
t1 <- t.test(A$rating, mu = 60)
t2 <- t.test(A$learning, mu = 65)
t3 <- t.test(A$raises, mu = 50)

make_one_sample_t_test_table(
  t_res = list(t1, t2, t3),
  variable_label = c("Rating", "Learning", "Raises (mu = 50)")
)
```

`make_paired_t_test_table` *Create a one-row summary table of a paired t-test*

Description

This function performs a paired t-test between two numeric variables in a data frame and returns a one-row summary table including means, mean difference, t-value, degrees of freedom, p-value, and confidence interval.

Usage

```
make_paired_t_test_table(
  data,
  var1,
  var2,
  var_name = NULL,
  alternative = "two.sided",
  conf.level = 0.95
)
```

Arguments

data	A data frame containing the two numeric variables.
var1	Character string. Name of the first variable (observation 1) in 'data'.
var2	Character string. Name of the second variable (observation 2) in 'data'.
var_name	Optional character string. Custom name for the variable to display in the table. Default is 'var1 vs var2'.
alternative	Character string specifying the alternative hypothesis. One of '"two.sided"', '"less"', or '"greater"'. Default is '"two.sided"'.
conf.level	Confidence level for the interval. Default is 0.95.

Value

A one-row data frame with columns:

- 'Variable' - variable name
- 'Mean_obs1' - mean of observation 1
- 'Mean_obs2' - mean of observation 2
- 'Mean_diff' - mean difference (obs1 - obs2)
- 't_value' - t statistic
- 'df' - degrees of freedom
- 'p_value' - p-value
- 'CI_lower' - lower bound of confidence interval
- 'CI_upper' - upper bound of confidence interval

Examples

```
# example data
df <- data.frame(
  before = c(10, 12, 14, 15, 11),
  after  = c(11, 13, 13, 16, 12)
)

# Run the paired t-test summary
make_paired_t_test_table(df, var1 = "before", var2 = "after")
```

make_reverse_score *Create Reverse Scores as New Columns*

Description

Create Reverse Scores as New Columns

Usage

```
make_reverse_score(data, vars, min_val, max_val, suffix = "_rev")
```

Arguments

data	A data frame.
vars	A character vector of column names to reverse.
min_val	The minimum possible score of the scale.
max_val	The maximum possible score of the scale.
suffix	A character string to append to the new column names. Defaults to "_rev".

Value

A data frame containing the original data plus the new reversed columns.

make_scale_description_table *Create Descriptive Table for Multiple Scale Columns*

Description

Computes descriptive statistics for one or more scale total columns. Accepts either a numeric vector (single column) or a data frame with column names.

Usage

```
make_scale_description_table(
  x = NULL,
  data = NULL,
  columns = NULL,
  scale_names = NULL,
  type = NULL
)
```

Arguments

x	Optional numeric vector of total scores (single column, old-style).
data	Optional data frame containing one or more scale columns.
columns	Optional character vector of column names in ‘data‘ (new-style).
scale_names	Optional character vector of names for each scale. Defaults to column names.
type	"summary" for base::summary(), NULL (default) uses psych::describe().

Value

A data frame with one row per column, containing descriptive statistics.

---

make_scale_total *Compute Scale Total Score*

Description

Computes the total score for a psychometric scale by summing specified numeric variables. The resulting total score is appended as a new column to the input data frame. Missing values are handled using ‘na.rm = TRUE‘.

Usage

```
make_scale_total(data, vars, new_var)
```

Arguments

data	A data frame.
vars	A character vector of column names to be summed.
new_var	A single character string specifying the name of the new total score column.

Value

The input data frame with one additional numeric column.

Examples

```
# Example using the standard 'bfi' (Big Five Inventory) dataset from psych
if (requireNamespace("psych", quietly = TRUE)) {
  data(bfi, package = "psych")

  # Take a small subset for demonstration
  sample_bfi <- head(bfi, 10)

  # Compute the Agreeableness total score (items A1 to A5)
  sample_bfi <- make_scale_total(
    data = sample_bfi,
```

```
  vars = paste0("A", 1:5),
  new_var = "Agreeableness_Total"
)

# Compute the Extraversion total score (items E1 to E5)
sample_bfi <- make_scale_total(
  data = sample_bfi,
  vars = paste0("E", 1:5),
  new_var = "Extraversion_Total"
)

# View the newly appended score columns alongside the items
sample_bfi[, c("A1", "Agreeableness_Total", "E1", "Extraversion_Total")]
}
```

`merge_counterbalanced_columns` *Merge Counterbalanced Item Pairs of Scale Administration Columns*

Description

Merges mutually exclusive item pairs resulting from counterbalanced survey administrations,generated during counterbalanced psychometric testing (e.g., online survey exports from PsyToolkit, Google Forms, or Qualtrics),and also handling blank strings, checking for missing columns, and flagging responses containing data in both versions.

Purpose: In counterbalanced designs (e.g., English vs. Gujarati version, or Order 1 vs. Order 2), each participant answers only one branch, so the export contains two sparse columns per item, each empty (‘NA‘ or ‘""‘) for half the sample.Such data is generated when same scales area administered in counterbalanced order using platforms like PsyToolkit

This function coalesces paired administration vectors into a unified set of item columns (e.g., merging ‘phq_e_1_1‘ and ‘phq_e_2_1‘ into ‘phq_e_1‘), checks for data-entry or branching overlap errors, and prepares the dataset for downstream psychometric evaluation

Usage

```
merge_counterbalanced_columns(
  data,
  col_1,
  col_2,
  new_columns,
  on_conflict = c("warning", "error", "silent")
)
```

Arguments

`data`	A data frame.
`col_1`	Character vector of column names for version 1.

col_2	Character vector of column names for version 2.
new_columns	Character vector of names for the merged output columns.
on_conflict	Character string specifying action if a participant answered both columns: "warning" (default; logs warning, keeps col_1), "error" (stops execution), or "silent" (silently keeps col_1).

Value

A data frame with merged columns appended.

Examples

```
## Not run:
# PHQ-9 administered in English and Gujarati, each across two orders

# Merge English items across orders -> phq_e_1 ... phq_e_9
counterbalance <- merge_counterbalanced_columns(
  data        = counterbalance,
  col_1       = paste0("phq_e_1_", 1:9),
  col_2       = paste0("phq_e_2_", 1:9),
  new_columns = paste0("phq_e_", 1:9),
  on_conflict = "warning"
)

# Merge Gujarati items across orders -> phq_g_1 ... phq_g_9
counterbalance <- merge_counterbalanced_columns(
  data        = counterbalance,
  col_1       = paste0("phq_g_1_", 1:9),
  col_2       = paste0("phq_g_2_", 1:9),
  new_columns = paste0("phq_g_", 1:9)
)

# Inspect the merged items
head(counterbalance[c("age", "gender", paste0("phq_g_", 1:9))])

## End(Not run)
```

---

scaledescr *scaledescr*

Description

Provides helper functions to format and summarise already computed outputs from commonly used statistical and psychometric functions into compact, single-row tables and simple graphs. Functions such as make_scale_description_table(), make_demographic_table(), make_alpha_table(), make_paired_t_test_table(), and make_dataframe_to_output() organise results obtained from existing functions including psych::describe(), psych::alpha(), stats::t.test(), and gtsummary::tbl_summary() for streamlined reporting and export to CSV, Word, and Excel formats. The package does not implement new statistical methods or perform additional estimation.

scaledescr_remove_column *Remove Columns from Data Frame*

Description

Removes one or more specified columns from a data frame by name. The function accepts either a single column name or a character vector of multiple column names. Missing columns are reported without crashing the script.

Usage

```
scaledescr_remove_column(data, column)
```

Arguments

data	A data frame.
column	A character vector of column names to be removed.

Value

The input data frame with the specified columns removed.

Examples

```
# Example using the standard 'bfi' (Big Five Inventory) data set from psych
if (requireNamespace("psych", quietly = TRUE)) {
  data(bfi, package = "psych")

  # Take a small subset for demonstration
  sample_bfi <- head(bfi, 10)

  # Remove a single column by name
  sample_bfi <- scaledescr_remove_column(
    data = sample_bfi,
    column = "education"
  )

  # Remove multiple columns using paste0()
  sample_bfi <- scaledescr_remove_column(
    data = sample_bfi,
    column = paste0("E", 1:5)
  )
}
```

Index

∗ **datasets**
counterbalance, 4

compute_ICC, 2
counterbalance, 4

ICC, *3*

make_alpha_table, 7
make_alphanumeric_conversion, 6
make_chisq_test_table, 8
make_citation_to_output, 9
make_correlation_table, 10
make_dataframe_to_output, 11
make_demographic_table, 13
make_demographic_table_to_output, 14
make_EFA_factorability_table, 15
make_independent_t_test_table, 16
make_item_total_correlation_table, 17
make_lavaan_analysis_tables, 18
make_lavaan_mediation_tables, 19
make_normality_table, 20
make_one_sample_t_test_table, 20
make_paired_t_test_table, 21
make_reverse_score, 23
make_scale_description_table, 23
make_scale_total, 24
merge_counterbalanced_columns, *6*, 25

scaledescr, 26
scaledescr_remove_column, 27