Package 'trendtestR'

September 2, 2025

Type Package

Title Exploratory Trend Analysis and Visualization for Time-Series and Grouped Data

Version 1.0.1

Description Provides a set of exploratory data analysis (EDA) tools for visualizing trends, diagnosing data types for beginner-friendly workflows, and automatically routing to suitable statistical tests or trend exploration models. Includes unified plotting functions for trend lines, grouped boxplots, and comparative scatterplots; automated statistical testing (e.g., t-test, Wilcoxon, ANOVA, Kruskal-Wallis, Tukey, Dunn) with optional effect size calculation; and model-based trend analysis using generalized additive models (GAM) for count data, generalized linear models (GLM) for continuous data, and zero-inflated models (ZIP/ZINB) for count data with potential zero-inflation.
Also supports time-window continuity checks, cross-year handling in compare_monthly_cases(), and ARIMA-ready preparation with stationarity diagnostics, ensuring consistent parameter styles for reproducible research and user-friendly workflows.Methods are based on R Core Team (2024) <https://www.R-project.org/>, Wood, S.N.(2017, ISBN:978-1498728331), Hyndman RJ, Khandakar Y (2008) <doi:10.18637/jss.v027.i03>, Simon Jackman (2024) <https://github.com/atahk/pscl/>, Achim Zeileis, Christian Kleiber, Simon Jackman (2008) <doi:10.18637/jss.v027.i08>.

License GPL (>= 3)

Encoding UTF-8

Imports dplyr, ggplot2 (>= 3.3.0), lubridate, emmeans, e1071, forecast, MASS, multcomp, tidyselect, tidyr, tseries, car, FSA, ggpubr, rlang, splines, pscl, mgcv

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0), mockr, knitr, rmarkdown

Config/testthat/edition 3

URL https://github.com/GrahnH/trendtestR

BugReports https://github.com/GrahnH/trendtestR/issues

VignetteBuilder knitr

NeedsCompilation no

Author Gelan Huang [aut, cre]

Maintainer Gelan Huang <huanggelan97@icloud.com>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2025-09-02 05:40:02 UTC

Contents

`check_continuity_by_window` *Check Time Series Continuity within Defined Window / Pruefung der Zeitreihen-Kontinuitaet*

Description

This function checks whether a date vector contains all expected time points within a specified window. Users can define the time unit (day, week, or month), granularity step, and whether ISO week starts (Monday) should be used. Returns a list indicating whether the data are continuous and reports any missing dates.

Usage

```
check_continuity_by_window(
  date_vec,
  years,
  months,
  window_unit = c("week", "day", "month"),
  step = 1,
  use_isoweek = FALSE,
  start_date = NULL,
  allow_leading_gap = FALSE
)
```

Arguments

date_vec	A vector of dates. / Ein Datumsvektor
years	Numeric vector indicating year range (e.g., c(2021, 2022)). / Jahr(e)
months	Numeric vector of months (1:12). / Monate (1:12)
window_unit	Time unit for continuity check: "day", "week", or "month". / Zeiteinheit fuer Pruefung
step	Step size for the sequence. Default is 1. / Schrittweite
use_isoweek	Logical. If TRUE, weeks start on Monday. / ISO-Woche (Montag)?
start_date	Optional. Override default start date (must be in "YYYY-MM-DD" format). / Optionales Startdatum
allow_leading_gap	Logical. If TRUE, allows first date to be missing but considers rest as continuous. / Erlaubt Anfangsluecke?

Details

Diese Funktion prueft, ob ein Datumsvektor alle erwarteten Zeitpunkte innerhalb eines definierten Fensters enthaelt. Die Zeitgranularitaet (Tag/Woche/Monat), Schrittweite und ISO-Wochenstart (Montag) koennen angepasst werden. Gibt zurueck, ob die Zeitreihe vollstaendig ist, und listet fehlende Zeitpunkte auf.

Value

A list with the following elements:

continuous Logical. Whether the time series is complete

gaps Data frame of missing expected dates

datum Vector of available dates within the window

range Start and end of expected time window

Examples

```
vec <- seq(as.Date("2021-01-01"), as.Date("2021-03-31"), by = "day")
check_continuity_by_window(vec, years = 2021, months = 1:3, window_unit = "day")
```

`check_input_validity` *Validate Time and Group Inputs for Case Comparison / Eingabepruefung fuer Zeit- und Gruppierungsvariablen*

Description

This function checks the validity of time-based and grouping arguments passed to functions like compare_monthly_cases(). It validates month/year ranges, aggregation settings, and optionally the presence and structure of group_col, returning standardized values and user-friendly messages for potential issues (e.g., non-factors or too many levels).

Usage

```
check_input_validity(
  months,
  years,
  shift_month,
  granularity,
  agg_fun,
  df,
  group_col = NULL
)
```

Arguments

`months`	Integer vector of months (1:12). / Vektor der Monate (1:12).
`years`	Integer vector of years (must be strictly increasing). / Vektor der Jahre (streng aufsteigend).
`shift_month`	One of "none", "mth_to_next", "mth_to_prev"; defines cross-year logic. / Jahreslogik.
`granularity`	"day" or "week". / Aggregationsebene.
`agg_fun`	Aggregation function: "sum", "mean", or "median". / Aggregationsfunktion.
`df`	Data frame used to validate group_col. / Datensatz zur Validierung von group_col.
`group_col`	Optional grouping column(s) to validate. / Optionale Gruppierungsvariable(n).

Details

Diese Funktion prueft Zeit- und Gruppierungsparameter, wie sie z.B. in compare_monthly_cases() verwendet werden. Sie validiert Monats- und Jahresangaben, Aggregationseinstellungen und (optional) die Struktur von group_col, und gibt standardisierte Werte sowie Hinweise bei potenziellen Problemen (z.B. fehlende Faktoren, zu viele Gruppen) zurueck.

Function Behavior and Messages: - Issues stop() for invalid months/years or aggregation settings. - Warns if group_col is missing in df. - Gives messages if group variables are not factors, or if too many levels (>8) are detected.

Funktionsverhalten und Hinweise: - Bei ungueltigen Zeitangaben erfolgt ein Abbruch (stop()). - Warnung bei nicht vorhandenen Gruppierungsvariablen. - Hinweis, falls Gruppenvariablen keine Faktoren sind oder zu viele Auspraegungen (>8) besitzen.

Value

A list with standardized values for:

months Validated months vector

years Validated and sorted years

granularity One of "day" or "week"

agg_fun One of "sum", "mean", or "median"

shift_month Cross-year setting

See Also

[compare_monthly_cases()], [run_group_tests()]

Diese Funktion wird typischerweise zusammen mit [compare_monthly_cases()], [run_group_tests()] verwendet.

Examples

```
# Example 1: Valid input without group_col
# Beispiel 1: Gueltige Eingabe ohne Gruppenvariable
df <- data.frame(
  datum = seq.Date(from = as.Date("2023-12-01"), to = as.Date("2025-02-28"), by = "day"),
  neue_faelle = sample(0:100, 456, replace = TRUE)
)
check_input_validity(
  months = c(12, 1, 2),
  years = c(2024, 2025),
  shift_month = "mth_to_next",
  granularity = "day",
  agg_fun = "sum",
  df = df
)

# Example 2: group_col exists but is not a factor
# Beispiel 2: group_col ist kein Faktor: Hinweis wird ausgegeben
df$region <- sample(c("Nord", "Sued", "West"), size = nrow(df), replace = TRUE)
check_input_validity(
  months = c(12, 1, 2),
  years = c(2024, 2025),
  shift_month = "mth_to_next",
  granularity = "day",
  agg_fun = "mean",
  df = df,
  group_col = "region"
)
```

```
# Example 3: Too many group levels triggers a message
# Beispiel 3: Zu viele Gruppenauspraegungen (>8): Warnung zur Plot-Lesbarkeit
df$gruppe <- factor(paste0("G", sample(1:12, size = nrow(df), replace = TRUE)))
check_input_validity(
  months = c(12, 1, 2),
  years = c(2024, 2025),
  shift_month = "mth_to_next",
  granularity = "week",
  agg_fun = "median",
  df = df,
  group_col = "gruppe"
)
```

`check_rate_diff_arima_ready` *Assess Time Series Readiness for ARIMA Modeling / Pruefung der Eignung fuer ARIMA-Zeitreihenmodellierung*

Description

This function performs diagnostics on a numeric time series (e.g., rate difference) to evaluate whether ARIMA modeling is appropriate. It runs tests for autocorrelation (Ljung-Box), trend presence, and stationarity (ADF & KPSS), and gives modeling recommendations. Optional visualizations include line plot, ACF/PACF, and STL decomposition.

Usage

```
check_rate_diff_arima_ready(
  rate_diff_vec,
  date_vec = NULL,
  frequency = 52,
  plot_acf = TRUE,
  do_stl = TRUE,
  verbose = TRUE,
  max_lag_acf = min(3 * frequency, floor(length(rate_diff_vec)/4))
)
```

Arguments

`rate_diff_vec`	Numeric vector of rate differences. / Numerischer Vektor (z.B. Rate)
`date_vec`	Optional. Corresponding date vector (used for plotting). / Optionaler Datumsvektor
`frequency`	Time series frequency (e.g., 52 for weekly). Default is 52. / Frequenz der Zeitreihe
`plot_acf`	Logical. Whether to compute and plot ACF/PACF. Default is TRUE. / ACF/PACF anzeigen?

do_stl	Logical. Whether to perform and plot STL decomposition. Default is TRUE. / STL-Dekomposition durchfuehren?
verbose	Whether to print standardization info /Ob Statusinformationen ausgegeben werden sollen
max_lag_acf	Max lag to use for ACF plots. Default is min(3 * frequency, floor(length(rate_diff_vec) / 4)). / Max. Verzoegerung fuer ACF

Details

Diese Funktion prueft, ob eine Zeitreihe (z.B. Differenz von Raten) fuer ARIMA-Modelle geeignet ist. Sie fuehrt Autokorrelationspruefung (Ljung-Box), Trendtest, sowie Stationaritaetstests (ADF & KPSS) durch und gibt Modellierungsempfehlungen. Optional werden Zeitreihengrafiken wie Linienplot, ACF/PACF und STL-Dekomposition erstellt.

Value

A list containing:

ts_data The cleaned numeric time series

assessment Overall diagnostic and modeling recommendation

adf ADF test result (stationarity)

kpss KPSS test result (stationarity)

plots Optional ggplot objects (e.g., time series plot)

stationarity_assessment Summary of stationarity status and differencing recommendation

Examples

```
vec <- c(NA, rnorm(60, 0.1, 1))
check_rate_diff_arima_ready(vec, frequency = 12)
```

---

compare_distribution_by_granularity

Compare Normality across Granularity Levels / Vergleich der Normalverteilung je Granularitaet

---

Description

This function compares Shapiro-Wilk normality results between two granularity levels (e.g., daily vs. weekly data). It extracts diagnostics from test result objects (from [run_paired_tests()], [run_group_tests()] or [run_multi_group_tests()]) and displays them side-by-side. Optionally, QQ plots are generated to visualize distributional properties.

Usage

```
compare_distribution_by_granularity(
  res_day,
  res_week,
  plot = TRUE,
  save_plot = FALSE,
  save_path = "."
)
```

Arguments

res_day	Result object from daily-level analysis. / Ergebnisobjekt der Tagesebene
res_week	Result object from weekly-level analysis. / Ergebnisobjekt der Wochenebene
plot	Logical. Whether to display QQ plots. / QQ-Plots anzeigen?
save_plot	Logical. Whether to save the plot as PNG. / Soll der Plot gespeichert werden?
save_path	Folder to save plot. Default is ".". / Speicherpfad fuer den Plot

Details

Diese Funktion vergleicht die Shapiro-Wilk-Ergebnisse der Normalverteilung zwischen zwei Granularitaetsebenen (z.B. Tages- vs. Wochen-Daten). Diagnosen aus Testergebnissen (z.B. von [run_paired_tests()], [run_group_tests()] oder [run_multi_group_tests()]) werden extrahiert und nebeneinander dargestellt. Optional werden QQ-Plots erzeugt, um Verteilungen zu visualisieren.

Value

A data frame with normality diagnostics for each granularity level.

factor Group name(s)

granularity Data granularity ("day" or "week")

shapiro_W Shapiro-Wilk W statistic

shapiro_p p-value of Shapiro-Wilk test

normal Whether data are considered normal ("ja"/"nein")

levene_p Levene's test p-value (if available)

bartlett_p Bartlett's test p-value (if available)

See Also

[run_paired_tests()], [run_group_tests()], [run_multi_group_tests()], [compare_monthly_cases]

Examples

```
df <- data.frame(
  datum = seq.Date(from = as.Date("2024-01-01"), by = "day", length.out = 400),
  neue_faelle = rpois(400, lambda = 20)
)
res_day <- compare_monthly_cases(
```

```
  df = df,
  datum_col = "datum",
  value_col = "neue_faelle",
  years = c(2024,2025),
  months = 1:2,
  granularity = "day",
  shift_month = "none",
  agg_fun = "sum",
  save_plot = FALSE
)
res_week <- compare_monthly_cases(
  df = df,
  datum_col = "datum",
  value_col = "neue_faelle",
  years = c(2024,2025),
  months = 1:2,
  granularity = "week",
  shift_month = "none",
  agg_fun = "sum",
  save_plot = FALSE
)
compare_distribution_by_granularity(res_day, res_week)
```

`compare_monthly_cases` *Compare Monthly Case Trends across Years / Vergleich monatlicher Falltrends zwischen Jahren*

Description

This function compares numeric variables (e.g., new case numbers) across specified months and years. It supports aggregation by day or ISO week, optional cross-year logic (e.g., combining Dec:Jan), automated visualization (trend line, dot plot, boxplot), and group-wise faceting. Statistical tests (e.g., t-test, ANOVA) are automatically selected and executed.

Usage

```
compare_monthly_cases(
  df,
  datum_col,
  value_col,
  group_col = NULL,
  years,
  months,
  granularity = "day",
  agg_fun = "sum",
  shift_month = "none",
  save_plot = FALSE,
  save_path = "."
)
```

Arguments

df	Data frame with at least a date and value column. / Data Frame mit Datum und Wert
datum_col	Name of the date column.. / Name der Datums-Spalte
value_col	Name of the value column. / Name der Wertespalte
group_col	Optional grouping variable(s) for faceting. / Optionale Gruppierung
years	Vector of years to include. E.g., c(2023, 2024). / Zu vergleichende Jahre
months	Vector of months to include (1:12). / Zu vergleichende Monate
granularity	Aggregation level: "day" or "week". / Aggregationsebene
agg_fun	Aggregation function: "sum", "mean", or "median". / Aggregationsfunktion
shift_month	Cross-year adjustment for Dec/Jan: "none", "mth_to_next", "mth_to_prev". / Jahreswechsel-Logik
save_plot	Logical. Whether to save the plots as PNG files. / Plots speichern?
save_path	Path to folder where plots should be saved. / Speicherpfad

Details

Diese Funktion vergleicht numerische Variablen (z.B. Fallzahlen) ueber ausgewaehlte Monate und Jahre hinweg. Sie unterstuetzt Aggregation nach Tag oder ISO-Woche, optionale Jahreswechsel-Logik (z.B. Dezember:Januar), automatische Visualisierung (Linien-, Punkt- und Boxplot) sowie Facetierung nach Gruppenvariablen. Die passenden statistischen Tests (z.B. t-Test, ANOVA) werden automatisch durchgefuehrt.

Function Behavior and Notes: - The function compares a numeric variable (e.g., case counts) across selected months and years. - Aggregation can be done at the "day" or "week" level (ISO week, Monday start). - When shift_month is set to "mth_to_next" or "mth_to_prev", months like December and January can be merged across year boundaries: - "mth_to_next": assigns months to the *next* year group (e.g., Dec 2023 to 2024). - "mth_to_prev": assigns monthd to the *previous* year group (e.g., Jan 2024 to 2023). - All plots (trend_plot, monthly_trend_plot, box_plot) are automatically colored by year and faceted if group_col is provided. - Statistical tests are performed automatically based on the number of groups (e.g., t.test, Wilcoxon, ANOVA, Kruskal-Wallis).

—

Funktionsverhalten und Hinweise: - Die Funktion vergleicht eine numerische Variable (z.B. Fallzahlen) ueber Monate und Jahre hinweg. - Die Aggregation erfolgt auf "day"- oder "week"-Ebene (ISO-Woche, Montag-basiert). - Mit shift_month = "mth_to_next" oder "mth_to_prev" koennen Monate ueber Jahresgrenzen hinweg zugeordnet werden: - "mth_to_next": Monat zum Folgejahr (z.B. Dez. 2023 → 2024) - "mth_to_prev": Monat zum Vorjahr (z.B. Jan. 2024 → 2023) - Alle Plots sind nach Jahr eingefaerbt; bei Angabe von group_col erfolgt eine Facetierung. - Die geeigneten statistischen Tests werden automatisch ausgewaehlt und durchgefuehrt.

—

Value

A list with the following elements:

data Aggregated and annotated data frame

trend_plot Line plot showing daily/weekly trends

monthly_trend_plot Dot plot by year and month

box_plot Boxplot comparing distributions across months and years

tests Result of statistical test (from [run_group_tests()])

table Frequency table of observations per year

See Also

[run_group_tests()], [check_continuity_by_window()], [standardize_case_columns()], [infer_value_type()]

Examples

```
set.seed(123)
test_df <- data.frame(
  datum = seq.Date(from = as.Date("2024-12-15"), to = as.Date("2025-01-20"), by = "day"),
  value = sample(0:50, size = 37, replace = TRUE)
)

compare_monthly_cases(
  df = test_df,
  datum_col = "datum",
  value_col = "value",
  years = c(2024, 2025),
  months = c(12, 1),
  granularity = "day",
  agg_fun = "sum",
  shift_month = "mth_to_next",
  save_plot = FALSE
)
```

---

diagnose_model_trend *Diagnose a fitted model using residual plots and statistical tests (ggplot2 only) / Modell-Diagnose mittels Residuenplots und statistischen Tests (nur ggplot2)*

Description

Diagnose model fit for lm, glm, gam (mgcv), and zeroinfl (pscl) models using residual plots and tests. / Diagnose lineare Modelle (lm), generalisierte lineare Modelle (glm), GAMs von mgcv und Zero-Inflated-Modelle von pscl mit ggplot2.

Usage

```
diagnose_model_trend(
  model,
  value_col = "value",
  residual_type = "deviance",
```

```
  group_col = NULL,
  verbose = TRUE
)
```

Arguments

model	A fitted model object (lm, glm, gam, or zeroinfl). / Ein angepasstes Modellobjekt (lm, glm, gam oder zeroinfl).
value_col	Name of the response variable (used in axis labels). / Name der Zielvariable (verwendet fuer Achsenbeschriftungen).
residual_type	Type of residuals to use ("deviance", "pearson", "response", etc.). / Art der Residuen ("deviance", "pearson", "response" usw.).
group_col	Optional. Grouping variable to color residual plots. / Optional. Gruppierungsvariable fuer Farbgebung in den Residuenplots.
verbose	Logical; whether to print diagnostic messages. / Logisch; ob Diagnosenachrichten ausgegeben werden sollen.

Value

A list with ggplot2 plots and diagnostic test results. / Eine Liste mit ggplot2-Plots und diagnostischen Testergebnissen:

plots A named list with residual plots ("residuals_vs_fitted", "qq", "scale_location"). / Eine Liste mit Residuenplots ("residuals_vs_fitted", "qq", "scale_location").

diagnostics A named list of statistical test results ("Shapiro", "KS", "Levene", "GAM check"). / Eine Liste mit Ergebnissen statistischer Tests ("Shapiro", "KS", "Levene", "GAM check").

See Also

[explore_poisson_trend], [explore_continuous_trend], [explore_zinb_trend], [explore_trend_auto]

Examples

```
# Example for a linear model
set.seed(123)
df <- data.frame(x = rnorm(100), y = rnorm(100))
model_lm <- lm(y ~ x, data = df)
diagnose_model_trend(model_lm)

# Beispiel fuer ein GLM
df_glm <- data.frame(x = rnorm(100), y = rpois(100, lambda = 2))
model_glm <- glm(y ~ x, data = df_glm, family = poisson())
diagnose_model_trend(model_glm)
```

`explore_continuous_trend` *Explore linear and GLM trends for continuous data with automatic model selection / Analyse linearer und GLM-Trends fuer kontinuierliche Daten mit automatischer Modellauswahl*

Description

Fits linear models or GLMs (Gaussian or Gamma) to continuous time series data, optionally using natural cubic splines. / Passt lineare Modelle oder GLMs (Gaussian oder Gamma) an kontinuierliche Zeitreihendaten an, optional mit natuerlichen kubischen Splines.

Usage

```
explore_continuous_trend(
  data,
  datum_col,
  value_col,
  group_col = NULL,
  df_spline = 2,
  family = c("auto", "gaussian", "gamma"),
  return_formula = FALSE,
  verbose = FALSE
)
```

Arguments

`data`	Dataframe with time series continuous data. / Dataframe mit Zeitreihen-kontinuierlichen Daten.
`datum_col`	Name of the time column (usually Date). / Name der Zeitspalte (normalerweise Date).
`value_col`	Name of the continuous value column (dependent variable). / Name der Spalte mit kontinuierlichen Werten (abhaengige Variable).
`group_col`	Optional. Name of grouping column for interaction. / Optional. Name der Gruppierungsspalte fuer Interaktion.
`df_spline`	Degrees of freedom for spline (default = 2). Set to 1 for linear trend. / Freiheitsgrade fuer den Spline (Standard = 2). Setze auf 1 fuer linearen Trend.
`family`	Specifies the GLM family: "auto" (default), "gaussian", or "gamma". / Gibt die GLM-Familie an: "auto" (Standard), "gaussian" oder "gamma".
`return_formula`	If TRUE, returns the model formula instead of fitting. / Wenn TRUE, wird nur die Modellformel zurueckgegeben.
`verbose`	Logical. Whether to print model fitting messages. / Ob Anpassungsmeldungen ausgegeben werden sollen.

Value

A list containing fitted model, formula, summary, plot, and model diagnostics. / Eine Liste mit Modell, Formel, Zusammenfassung, Plot und Diagnosen:

model The fitted GLM object. / Das angepasste GLM-Objekt.

summary Model summary. / Zusammenfassung des Modells.

plot ggplot2 visualization of the trend. / ggplot2-Visualisierung des Trends.

dispersion_parameter Estimated dispersion. / Geschaetzter Dispersionsparameter.

model_family_used Family used for final model. / Verwendete Modellfamilie.

model_selection_info Information about family selection (if auto). / Hinweise zur Modellauswahl (bei auto).

aic_comparison Optional AIC comparison table (if auto and gamma used). / Optionale AIC-Vergleichstabelle (bei auto mit Gamma).

messages Concatenated messages from model fitting. / Konsolidierte Anpassungsmeldungen.

See Also

[explore_trend_auto()]

Examples

```
# Basic Gaussian GLM on continuous data
df <- data.frame(
  datum = seq.Date(from = as.Date("2023-01-01"), by = "day", length.out = 100),
  value = 5 + sin(1:100 / 10) + rnorm(100)
)
explore_continuous_trend(df, datum_col = "datum", value_col = "value", df_spline = 1)

# Automatische Auswahl zwischen Gaussian und Gamma
df2 <- data.frame(
  datum = seq.Date(from = as.Date("2023-01-01"), by = "day", length.out = 100),
  value = exp(seq(-1, 1, length.out = 100)) + rnorm(100, sd = 0.2)
)
explore_continuous_trend(df2,
                         datum_col = "datum",
                         value_col = "value",
                         df_spline = 2, family = "auto")
```

`explore_poisson_trend` *Explore time-based GAM for count data trend with automatic model selection / Zeitbasierte GAM-Trendanalyse fuer Zaehldaten mit automatischer Modellauswahl*

Description

Fits a Generalized Additive Model (GAM) with time-based splines to count data, supporting automatic selection between Poisson and Negative Binomial families. / Passt ein Generalisiertes Additives Modell (GAM) mit zeitbasierten Splines an Zaehldaten an, inklusive automatischer Auswahl zwischen Poisson und Negativer Binomialverteilung.

Usage

```
explore_poisson_trend(
  data,
  datum_col,
  value_col,
  group_col = NULL,
  k_spline = 4,
  family = c("auto", "poisson", "negbin"),
  phi = 1.5,
  return_formula = FALSE,
  verbose = FALSE
)
```

Arguments

`data`	Data frame with time series count data. / Data Frame mit Zeitreihen-Zaehldaten.
`datum_col`	Name of the time column (usually Date). / Name der Zeit-Spalte (normalerweise Date).
`value_col`	Name of the count column (dependent variable). / Name der Zaehldaten-Spalte (abhaengige Variable).
`group_col`	Optional. Name of grouping column for interaction. / Optional. Name der Gruppierungs-Spalte fuer Interaktion.
`k_spline`	Basis dimension for smooth terms (default = 4). Larger k allows more complex curves. / Basisdimension fuer glatte Terme (Standard = 4). Hoeheres k erlaubt komplexere Kurven.
`family`	Specifies the GAM family: "auto" (default, chooses based on overdispersion), "poisson", or "negbin". / Gibt die GAM-Familie an: "auto" (Standard, Auswahl basierend auf Overdispersion), "poisson" oder "negbin".
`phi`	Overdispersion parameter threshold for model selection (default = 1.5). / Schwellwert fuer Overdispersion zur Modellauswahl (Standard = 1.5).
`return_formula`	If TRUE, returns the model formula instead of fitting. / Wenn TRUE, wird die Modellformel zurueckgegeben.
`verbose`	Logical. Whether to print model fitting messages. / Ob Modellanpassungsmeldungen ausgegeben werden sollen.

Value

A list containing: / Eine Liste mit:

model Fitted GAM model object / Angepasstes GAM-Modellobjekt

summary Summary of the model / Modellzusammenfassung

plot ggplot of observed vs fitted trend / ggplot mit beobachteten und geschaetzten Trends

dispersion_parameter List with overdispersion info (phi or theta) / Liste mit Overdispersion-Informationen (phi oder theta)

model_family_used Model family used / Verwendete Modellfamilie

model_selection_info Explanation of model selection / Beschreibung der Modellauswahl

effective_df Effective degrees of freedom of smooth term(s) / Effektive Freiheitsgrade der glatten Terme

aic_comparison AIC comparison if applicable / AIC-Vergleich, falls zutreffend

messages Vector of fitting messages and warnings / Meldungen und Warnungen zur Modellanpassung

formula Model formula used / Verwendete Modellformel

See Also

[mgcv::gam()], [mgcv::nb()], [explore_zinb_trend()]

Examples

```
# Simulierte Zeitreihen-Zaehldaten
set.seed(123)
df <- data.frame(
  datum = seq.Date(from = as.Date("2023-01-01"), by = "day", length.out = 100),
  value = rpois(100, lambda = 5)
)

# Automatische Familienauswahl basierend auf Overdispersion
explore_poisson_trend(df, datum_col = "datum", value_col = "value", family = "auto")

# Negative Binomial-GAM erzwingen
explore_poisson_trend(df, datum_col = "datum", value_col = "value", family = "negbin")
```

---

explore_trend_auto *Main dispatcher for trend analysis based on data type / Hauptverzweiger fuer Trendanalyse basierend auf Datentyp*

Description

Automatically selects and calls the appropriate trend analysis function depending on whether the data is count-based or continuous. / Waehlt automatisch die passende Trendanalyse-Funktion basierend auf Zaehldaten oder stetigen Daten.

Usage

```
explore_trend_auto(
  df,
  datum_col,
  value_col,
  group_col = NULL,
  family = "auto",
  kdf = 3,
  return_formula = FALSE,
  verbose = FALSE,
  control = NULL
)
```

Arguments

df	Data frame with time series data. / Data Frame mit Zeitreihendaten.
datum_col	Name of the date/time column. / Name der Datums- oder Zeitspalte.
value_col	Name of the dependent variable column. / Name der abhaengigen Variablen.
group_col	Optional. Name of the grouping column. / Optional. Name der Gruppierungsvariable.
family	Model family to use: "auto", "poisson", "negbin", "zip", "zinb", "gaussian", etc. Passed to sub-functions. / Modellfamilie: "auto", "poisson", "negbin", "zip", "zinb", "gaussian" usw. Wird an Unterfunktionen weitergegeben.
kdf	Basis dimension for spline terms (k for GAM or ZI models). / Basisdimension fuer Splines (k bei GAM oder ZI-Modellen).
return_formula	If TRUE, return only the model formula without fitting. / Bei TRUE wird nur die Modellformel zurueckgegeben.
verbose	If TRUE, print detailed messages. / Bei TRUE werden Diagnosemeldungen ausgegeben.
control	Optional control parameters for model fitting (e.g., maxit). / Optionale Steuerparameter fuer die Modellanpassung.

Value

Result from the appropriate trend analysis function. / Rueckgabe des Ergebnisses der ausgewaehlten Trendanalysefunktion (z.B. explore_poisson_trend()).

See Also

[explore_poisson_trend()], [explore_zinb_trend()], [explore_continuous_trend()], [infer_value_type()], [prepare_group_data()]

Examples

```
# Simulated count data (Poisson)
df <- data.frame(
  datum = seq.Date(from = as.Date("2023-01-01"), by = "day", length.out = 100),
```

```
  value = rpois(100, lambda = 5)
)
explore_trend_auto(df, datum_col = "datum", value_col = "value")

# Beispiel mit kontinuierlichen Werten
df2 <- data.frame(
  datum = seq.Date(from = as.Date("2023-01-01"), by = "day", length.out = 100),
  value = sin(1:100 / 10) + rnorm(100)
)
explore_trend_auto(df2, datum_col = "datum", value_col = "value")
```

`explore_zinb_trend` *Explore zero-inflated models (ZIP/ZINB) for count data trends / Analyse von Zero-Inflated-Modellen (ZIP/ZINB) fuer Zeitreihen mit Zaehldaten*

Description

Fits zero-inflated Poisson (ZIP) or negative binomial (ZINB) models to time series count data using splines. / Passt Zero-Inflated Poisson- oder Negativ-Binomial-Modelle mit Splines an Zeitreihen mit Zaehldaten an.

Usage

```
explore_zinb_trend(
  data,
  datum_col,
  value_col,
  group_col = NULL,
  k_spline = 4,
  family = c("auto", "zip", "zinb"),
  run_vuong = FALSE,
  return_formula = FALSE,
  verbose = FALSE,
  control = NULL
)
```

Arguments

`data`	Dataframe with time series count data. / Dataframe mit Zeitreihenzaehldaten.
`datum_col`	Name of the time column (usually Date). / Name der Zeitspalte (normalerweise Date).
`value_col`	Name of the count column (dependent variable). / Name der Zaehlspalte (abhaengige Variable).
`group_col`	Optional. Name of grouping column for interaction. / Optional. Name der Gruppierungsspalte fuer Interaktion.

k_spline	Basis dimension for spline terms (default = 4). / Basisdimension fuer Spline-Terme (Standard = 4).
family	One of "zip", "zinb", or "auto". If "auto", selects model based on AIC. / Einer von "zip", "zinb" oder "auto". Bei "auto" erfolgt die Auswahl basierend auf AIC.
run_vuong	Logical. If TRUE, run Vuong test for model comparison (default = FALSE). / Wenn TRUE, wird Vuong-Test fuer Modellvergleich durchgefuehrt (Standard = FALSE).
return_formula	If TRUE, return model formula instead of fitting. / Wenn TRUE, wird nur die Modellformel zurueckgegeben.
verbose	Logical. Whether to print model fitting messages. / Ob Meldungen zur Modellanpassung gedruckt werden.
control	Optional. List for pscl::zeroinfl.control (e.g., list(maxit = 200)). Default: maxit = 100. / Steuerparameter fuer pscl::zeroinfl.control (z.B. list(maxit = 200)). Standard: maxit = 100.

Details

Supports automatic model selection based on AIC, optional Vuong test, flexible optimizer control, and visualization. / Unterstuetzt automatische Modellauswahl basierend auf AIC, optionalen Vuong-Test, flexible Optimierungssteuerung und Visualisierung.

Value

A list containing model object and diagnostics. / Eine Liste mit Modellobjekt und Diagnoseergebnissen:

model The fitted zeroinfl model. / Das angepasste zeroinfl-Modell.

summary Model summary (if available). / Modellzusammenfassung (wenn verfuegbar).

plot Trend plot with fitted values (ggplot2). / Trendplot mit geschaetzten Werten (ggplot2).

model_family_used Used model type: "ZIP" or "ZINB". / Verwendeter Modelltyp: "ZIP" oder "ZINB".

model_selection_info Information about model selection logic. / Hinweise zur Modellauswahl.

aic_comparison Data frame with AIC values for both models. / Data Frame mit AIC-Werten fuer beide Modelle.

vuong_test Vuong test result (if computed). / Vuong-Testergebnis (falls berechnet).

messages Messages from the fitting process. / Meldungen aus dem Anpassungsprozess.

See Also

[pscl::zeroinfl()], [pscl::vuong()], [explore_poisson_trend()]

Examples

```
# Simulierte Zero-Inflated Zaehldaten
set.seed(123)
df <- data.frame(
  datum = seq.Date(from = as.Date("2023-01-01"), by = "day", length.out = 100),
  value = rbinom(100, 1, 0.3) * rpois(100, lambda = 4)
)

# Automatische Auswahl zwischen ZIP und ZINB
explore_zinb_trend(df, datum_col = "datum", value_col = "value", family = "auto")

# Nur ZIP-Modell erzwingen
explore_zinb_trend(df, datum_col = "datum", value_col = "value", family = "zip", k_spline = 3)
```

---

filter_by_groupcol *Filter and optionally reshape a data frame by group column / Nach Gruppenspalte filtern und optional umstrukturieren*

Description

This function filters a data frame based on specified grouping levels and optionally transforms it into a wide format for further analysis. It supports retaining extra columns and provides robust error checking.

Usage

```
filter_by_groupcol(
  df,
  group_col,
  value_col,
  datum_col,
  keep_levels = NULL,
  to_wide = FALSE,
  keep_other_cols = FALSE
)
```

Arguments

df	A data.frame or tibble containing the data.
group_col	A string specifying the grouping column (e.g., "region", "age_group").
value_col	A string for the value column (default: "neue_faelle").
datum_col	A string for the date column (default: "datum").
keep_levels	Optional vector of levels to retain in group_col. Default = NULL (keep all).
to_wide	Logical, if TRUE returns a wide-format table (each level a column).
keep_other_cols	Logical, if TRUE keeps all other original columns.

Details

Diese Funktion filtert einen Data Frame basierend auf bestimmten Gruppenwerten und kann ihn optional in ein Wide-Format umwandeln. Es koennen weitere Spalten beibehalten werden, und die Funktion enthaelt robuste Fehlerpruefungen.

This function is particularly useful for preparing time series grouped by categories, such as cases per region or age group.

Diese Funktion eignet sich besonders zur Vorbereitung gruppierter Zeitreihen, z.B. nach Region oder Altersgruppe.

Value

A filtered and optionally reshaped tibble.

Examples

```
# English / Deutsch
df <- data.frame(
  datum = as.Date("2024-01-01") + 0:9,
  gruppe = rep(c("A", "B"), each = 5),
  neue_faelle = c(10, 12, 13, 15, 11, 20, 21, 22, 19, 18),
  region = rep("Berlin", 10)
)

filter_by_groupcol(
  df,
  group_col = "gruppe",
  value_col = "neue_faelle",
  datum_col = "datum",
  keep_levels = "A"
)

filter_by_groupcol(
  df,
  group_col = "gruppe",
  value_col = "neue_faelle",
  datum_col = "datum",
  to_wide = TRUE
)
```

---

infer_value_type *Infer variable type from numeric vector / Typ-Erkennung numerischer Vektoren*

Description

This function analyzes a numeric vector and infers the underlying variable type: "binary", "proportion", "count", "discrete", or "continuous". Useful for selecting statistical tests or visualization strategies.

Usage

```
infer_value_type(values, verbose = TRUE, thresholds = NULL)
```

Arguments

values	A numeric vector Ein numerischer Vektor
verbose	Logical, whether to show warnings (default TRUE) Gibt an, ob Warnmeldungen angezeigt werden (Standard: TRUE)
thresholds	Optional list of numeric thresholds for type detection (for internal use only) Optionale Liste von Schwellwerten zur Typ-Erkennung (nur intern verwendet)

Details

Diese Funktion analysiert einen numerischen Vektor und erkennt den zugrunde liegenden Typ: "binary", "proportion", "count", "discrete" oder "continuous". Nützlich zur Auswahl geeigneter statistischer Tests oder Visualisierungen.

Value

A character string indicating the inferred type: "binary", "proportion", "count", "discrete", or "continuous".
 Ein Zeichenstring mit dem erkannten Typ: "binary", "proportion", "count", "discrete" oder "continuous".

See Also

[standardize_case_columns()]

Examples

```
infer_value_type(c(1, 0, 1, 1))
infer_value_type(c(0.2, 0.5, 0.8))
infer_value_type(c(3, 4, 6, 1000000))
```

---

plot_weekly_cases *Visualize Weekly Aggregated Values / Woechentliche aggregierte Werte visualisieren*

Description

This function aggregates time series data by calendar week and generates three plots trend line, histogram, and boxplot based on a specified retrospective period (either a number of weeks or a date range). It also shows a 95

Usage

```
plot_weekly_cases(
  df,
  datum_col = "datum",
  value_col = NULL,
  weeks_back = 51,
  agg_fun = "sum",
  plottype = NULL,
  save_plot = FALSE,
  save_path = "."
)
```

Arguments

df	A data.frame with date and value columns. / Ein Data Frame mit Datums- und Wertespalten
datum_col	Name of the date column, default is "datum". / Name der Datumsspalte, Standard: "datum"
value_col	Name of the value column. / Name der Wertespalte /
weeks_back	Number of recent weeks or a length-2 numeric vector. / Anzahl der zurueckliegenden Wochen oder ein Vektor mit zwei Werten
agg_fun	Aggregation function (e.g., "sum", "mean"). / Aggregationsfunktion, z.B. "sum", "mean"
plottype	Optional plot type: 1 for all, 2 for hist+box, 3 for trend+box. / Optionaler Plottyp: 1 fuer alle, 2 fuer hist+box, 3 fuer trend+box
save_plot	Logical, whether to save the plots. / Logisch, ob die Plots gespeichert werden sollen
save_path	Folder to save the plots. / Zielpfad zum Speichern der Plots

Details

Diese Funktion aggregiert Zeitreihendaten nach Kalenderwochen und erstellt fuer den angegebenen Rueckblickzeitraum (als Anzahl der Wochen oder Zeitfenster) drei Diagramme: Trendverlauf, Histogramm und Boxplot. Zusaetzlich wird ein 95

Value

A list containing:

data Aggregated weekly data

trend_plot Trend plot

hist_plot Histogram

box_plot Boxplot

Examples

```
df <- data.frame(
  datum = as.Date("2022-01-01") + 0:100,
  neue_faelle = rpois(101, lambda = 20)
)
result <- plot_weekly_cases(df, value_col = "neue_faelle", weeks_back = 20)
```

`prepare_group_data` *Prepare Grouped Data for Statistical Testing*

Description

This function prepares a dataset for grouped statistical tests by: - Filtering out NA values in the target variable; - Dropping empty groups and reporting excluded levels; - Splitting the values by group and computing sample sizes.

Diese Funktion bereitet Daten fuer gruppierte Tests vor: - Entfernt fehlende Werte (NA) in der Zielvariablen; - Entfernt leere Gruppen und gibt eine Warnung aus; - Teilt die Werte nach Gruppen und berechnet Stichprobengroessen.

Usage

```
prepare_group_data(df, value_col = ".value", group_col = "jahr")
```

Arguments

`df`	A data.frame or tibble containing the data.
`value_col`	A string indicating the name of the column with values to test.
`group_col`	A string indicating the name of the grouping column.

Details

Cleans and splits the input data by group, removing missing values and empty groups.

Value

A list with the following elements:

df The filtered data frame with updated grouping column.

vals A list of vectors, one per group, containing values.

sample_sizes A named vector with sample sizes per group.

n_groups The number of groups remaining after filtering.

group_names The names of the groups.

`run_count_multi_group_tests` *Statistical Test for Count Data (Multi-Groups) / Statistischer Test fuer Zaehldaten (Mehrere Gruppen)*

Description

This function performs a simple comparison of count data across three or more groups. It uses Poisson or Negative Binomial regression, considering overdispersion, followed by an ANOVA-like test for the overall group effect and optional post-hoc tests. Focus is on overall p-value and identifying differing groups, without complex model interpretation.

Usage

```
run_count_multi_group_tests(
  df,
  value_col = ".value",
  group_col = "jahr",
  alpha = 0.05,
  phi = 1.5,
  effect_size = FALSE,
  report_assumptions = FALSE
)
```

Arguments

`df`	A data frame containing the data, already prepared (e.g., by prepare_group_data).
`value_col`	Name of the column containing count values. Default is ".value".
`group_col`	Name of the grouping variable. Default is "jahr".
`alpha`	Significance level for hypothesis testing. Default is 0.05.
`phi`	Common heuristic for overdispersion. Default is 1.5.
`effect_size`	Logical. Whether to calculate and return a simple effect size (e.g., Pseudo R-squared).
`report_assumptions`	Logical. Whether to report basic assumption diagnostics (e.g., overdispersion status).

Details

Diese Funktion fuehrt einen einfachen Vergleich von Zaehldaten bei drei oder mehr Gruppen durch. Sie verwendet Poisson- oder Negative Binomial-Regression (abhaengig von Ueberdispersion), gefolgt von einem ANOVA-aehnlichen Test fuer den Gesamtgruppeneffekt und optionalen Post-Hoc-Tests. Der Schwerpunkt liegt auf dem Gesamt-p-Wert und der Identifizierung unterschiedlicher Gruppen, ohne komplexe Modellinterpretation.

Value

A list containing test results (p-value, significant groups, chosen method).

Examples

```
set.seed(123)
data <- data.frame(
.value = c(rpois(50, 3), rpois(50, 5), rpois(50, 4)),
jahr = factor(rep(c("2020", "2021", "2022"), each = 50))
)

result <- run_count_multi_group_tests(
df = data,
value_col = ".value",
group_col = "jahr",
alpha = 0.05,
phi = 1.5,
effect_size = TRUE,
report_assumptions = TRUE
)

print(result$p_value)
print(result$significant_pairwise_differences)
print(result$effect_size)
```

---

run_count_two_group_tests

Statistical Test for Count Data (Two Groups) / Statistischer Test fuer Zaehldaten (Zwei Gruppen)

Description

This function performs a simple comparison of count data between two groups. It uses Poisson or Negative Binomial regression based on overdispersion, focusing on providing a p-value and direction of difference without complex model interpretation. Now includes basic descriptive statistics and confidence intervals.

Usage

```
run_count_two_group_tests(
  df,
  value_col = ".value",
  group_col = "jahr",
  alpha = 0.05,
  phi = 1.5,
  effect_size = FALSE,
  report_assumptions = TRUE
)
```

Arguments

df	A data frame containing the data, already prepared (e.g., by prepare_group_data).
value_col	Name of the column containing count values. Default is ".value".
group_col	Name of the grouping variable. Default is "jahr".
alpha	Significance level for hypothesis testing. Default is 0.05.
phi	Common heuristic for overdispersion. Default is 1.5.
effect_size	Logical. Whether to calculate and return a simple effect size (e.g., Incidence Rate Ratio).
report_assumptions	Logical. Whether to report basic assumption diagnostics (e.g., overdispersion status).

Details

Diese Funktion fuehrt einen einfachen Vergleich von Zaehldaten zwischen zwei Gruppen durch. Sie verwendet Poisson- oder Negative Binomial-Regression (abhaengig von Ueberdispersion), wobei der Schwerpunkt auf der Bereitstellung eines p-Wertes und der Richtung des Unterschieds liegt, ohne komplexe Modellinterpretation. Nun auch mit grundlegenden deskriptiven Statistiken und Konfidenzintervallen.

Value

A list containing test results (p-value, direction, chosen method), basic statistics, and confidence intervals.

---

run_group_tests *Automated Selection of Statistical Group Tests / Automatisierte Auswahl statistischer Gruppentests*

Description

This function automatically determines whether to perform a two-group test (paired or unpaired) or a multi-group test depending on the number of groups in the data. For two groups, both paired t-test (if specified) and Wilcoxon test are run. For three or more groups, the function checks assumptions (normality and homogeneity of variances) and selects either ANOVA with Tukey post-hoc or Kruskal-Wallis with Dunn post-hoc. All tests include assumption checking and optional effect size calculation.

Usage

```
run_group_tests(
  df,
  value_col = ".value",
  group_col = "jahr",
  alpha = 0.05,
```

```
  effect_size = TRUE,
  report_assumptions = TRUE,
  paired = FALSE
)
```

Arguments

df	A data frame with at least two groups. / Ein Data Frame mit mindestens zwei Gruppen
value_col	Name of the column containing values to compare. Default is ".value". / Name der Werte-Spalte, Standard: ".value"
group_col	Name of the grouping variable. Default is "jahr". / Spaltenname der Gruppierungsvariable, Standard: "jahr"
alpha	Significance level for hypothesis testing. Default is 0.05. / Signifikanzniveau fuer Testentscheidungen, Standard: 0.05
effect_size	Logical. Whether to calculate effect sizes. / Logisch, ob Effektgroessen berechnet werden sollen
report_assumptions	Logical. Whether to include assumption check results. / Logisch, ob Vorannahmen ausgegeben werden sollen
paired	Only relevant for two groups: TRUE for paired data. / Nur bei zwei Gruppen relevant: TRUE fuer gepaarte Daten

Details

Diese Funktion erkennt anhand der Anzahl der Gruppen automatisch, ob ein Zwei-Gruppen-Test (gepaart oder ungepaart) oder ein Mehr-Gruppen-Test erforderlich ist. Bei zwei Gruppen werden t-Test (gepaart oder ungepaart) und Wilcoxon-Test durchgefuehrt. Bei drei oder mehr Gruppen erfolgt eine Auswahl zwischen ANOVA mit Tukey oder Kruskal-Wallis mit Dunn, je nach Verteilungsannahmen. Alle Tests beinhalten Vorannahmepruefungen und (optional) Effektgroessenschaetzungen.

Value

A list containing:

type Type of test performed (e.g., "Paired Test", "ANOVA")

sample_sizes Number of observations per group

group_names Group labels

t_test / kruskal / anova Test result object(s)

effect_size Effect size estimates (e.g., Cohen's d, eta-squared)

assumptions Assumption check results

recommendation Recommended test type based on assumptions

See Also

[run_paired_tests()], [run_multi_group_tests()], [run_count_two_group_tests()], [run_count_multi_group_tests()]

Examples

```
df <- data.frame(
  jahr = rep(c("2020", "2021"), each = 10),
  .value = c(rnorm(10, 20, 3), rnorm(10, 22, 3))
)
result <- run_group_tests(df)
```

`run_multi_group_tests` *Multi-Group Test with Assumption Checks / Mehr-Gruppen-Test mit Annahmepruefung*

Description

This function performs multi-group statistical comparisons depending on distribution and variance assumptions. If all groups pass the Shapiro-Wilk test for normality and Levene's test for homogeneity of variances, an ANOVA is performed with post-hoc Tukey test. Otherwise, the Kruskal-Wallis test is used, followed by Dunn's test (Bonferroni-adjusted). Effect size (eta2 or approximate) and assumption diagnostics are returned.

Usage

```
run_multi_group_tests(
  df,
  value_col = ".value",
  group_col = "jahr",
  alpha = 0.05,
  effect_size = TRUE,
  report_assumptions = TRUE
)
```

Arguments

`df`	A data frame with three or more groups. / Ein Data Frame mit drei oder mehr Gruppen
`value_col`	Name of the column containing values to compare. Default is ".value". / Name der Werte-Spalte, Standard: ".value"
`group_col`	Name of the grouping variable. Default is "jahr". / Spaltenname der Gruppierungsvariable, Standard: "jahr"
`alpha`	Significance level for hypothesis testing. Default is 0.05. / Signifikanzniveau fuer Testentscheidungen, Standard: 0.05
`effect_size`	Logical. Whether to calculate eta2 or its approximation. / Logisch, ob eta2 berechnet werden soll
`report_assumptions`	Logical. Whether to include assumption checks. / Logisch, ob Vorannahmen ausgegeben werden sollen

Details

Diese Funktion fuehrt Mehr-Gruppen-Vergleiche durch, abhaengig von Verteilungs- und Varianzannahmen. Wenn alle Gruppen normalverteilt sind (Shapiro-Wilk) und die Varianz homogen ist (Levene-Test), wird eine ANOVA mit Tukey-Post-Hoc-Test durchgefuehrt. Andernfalls wird ein Kruskal-Wallis-Test mit anschliessender Dunn-Analyse (Bonferroni-korrigiert) verwendet. Effektgroessen (eta2 oder Annaeherung) und Annahmepruefungen werden zurueckgegeben.

Value

A list containing:

type Type of test performed ("ANOVA" or "Kruskal-Wallis")

sample_sizes Sample size per group

assumptions List of assumption test results: Shapiro-Wilk, Levene, Bartlett

anova / kruskal Test result object

eta_squared / eta_squared_approx Effect size

interpretation Interpretation of eta2 magnitude

posthoc / dunn Post-hoc test result (Tukey or Dunn)

recommendation Recommended method based on assumption checks

See Also

[run_group_tests()], [run_paired_tests()]

Examples

```
df <- data.frame(
  jahr = rep(c("2020", "2021", "2022"), each = 10),
  .value = c(rnorm(10, 20), rnorm(10, 23), rnorm(10, 22))
)
run_multi_group_tests(df)
```

---

run_paired_tests *Paired / Unpaired Two-Group Tests with Assumption Checks / Zwei-Gruppen-Test mit Vorannahmepruefung*

Description

This function performs both parametric (t-test) and non-parametric (Wilcoxon test) comparisons between two groups. For paired data, it calculates the difference and performs Shapiro-Wilk normality test on the difference. Based on this, it recommends either a paired t-test or a Wilcoxon signed-rank test. Optionally, it calculates the effect size (Cohen's d) and returns assumption diagnostics.

Usage

```
run_paired_tests(
  df,
  value_col = ".value",
  group_col = "jahr",
  alpha = 0.05,
  effect_size = TRUE,
  report_assumptions = TRUE,
  paired = TRUE
)
```

Arguments

df	A data frame with exactly two groups. / Ein Data Frame mit genau zwei Gruppen
value_col	Name of the column containing values to compare. Default is ".value". / Name der Werte-Spalte, Standard: ".value"
group_col	Name of the grouping variable. Default is "jahr". / Spaltenname der Gruppierungsvariable, Standard: "jahr"
alpha	Significance level for hypothesis testing. Default is 0.05. / Signifikanzniveau fuer Testentscheidungen, Standard: 0.05
effect_size	Logical. Whether to calculate Cohen's d. / Logisch, ob Cohen's d berechnet werden soll
report_assumptions	Logical. Whether to include normality test results. / Logisch, ob Shapiro-Test zurueckgegeben wird
paired	Logical. Whether the data are paired. / Logisch: gepaarte Daten?

Details

Diese Funktion fuehrt sowohl parametrische (t-Test) als auch nicht-parametrische (Wilcoxon-Test) Vergleiche zwischen zwei Gruppen durch. Bei gepaarten Daten wird die Differenz gebildet und auf Normalverteilung geprueft(Shapiro-Test). Je nach Ergebnis wird ein gepaarter t-Test oder ein Wilcoxon-Vorzeichen-Rang-Test empfohlen. Optional wird die Effektgroesse (Cohens d) berechnet und die Vorannahmen zurueckgegeben.

Value

A list containing:

type Test type performed ("Paired Test" or "Unpaired Test")

sample_sizes Number of observations per group

group_names Names of the two groups

t_test Result of the t-test (paired or unpaired)

wilcox_test Result of the Wilcoxon test

effect_size Cohen's d (if enabled)

assumptions Shapiro-Wilk normality test result(s)

recommendation Recommended test based on normality

See Also

[run_group_tests()], [run_multi_group_tests()], [prepare_group_data()]

Examples

```
df <- data.frame(
  jahr = rep(c("2020", "2021"), each = 10),
  .value = c(rnorm(10, 30), rnorm(10, 32))
)
run_paired_tests(df, paired = TRUE)
```

---

standardize_case_columns

Standardize date and value columns / Standardisierung von Datum und Werten

---

Description

This function converts a date column to "Date" format and ensures the value column is numeric. If a "monat" column exists, it will be converted to an ordered factor. Useful for preprocessing time series data (e.g., daily cases).

Usage

```
standardize_case_columns(df, datum_col = NULL, value_col, verbose = TRUE)
```

Arguments

df	A data.frame / Ein Data Frame
datum_col	Name of the date column, default is "NULL" / Spaltenname des Datums, default value is NULL
value_col	Name of the value column / Spaltenname der Werte
verbose	Ob Statusinformationen ausgegeben werden sollen / Whether to print standardization info

Details

Diese Funktion konvertiert eine Datumsspalte in das "Date"-Format und stellt sicher, dass die Wertespalte numerisch ist. Falls eine "monat"-Spalte vorhanden ist, wird sie als geordneter Faktor umkodiert. Nuetzlich fuer die Vorverarbeitung von Zeitreihendaten (z.B. Fallzahlen).

Value

A cleaned data.frame with a ".value" column, standardized Date column, and possibly ordered "monat" factor.
 Ein aufbereiteter Data Frame mit ".value"-Spalte, konvertiertem Datum und ggf. geordnetem "monat"-Faktor.

See Also

[infer_value_type()]

Examples

```
df <- data.frame(
  datum = c("2021-01-01", "2021-01-02"),
  neue_faelle = c("12", "15"),
  monat = c("Jan", "Jan")
)
df_clean <- standardize_case_columns(df, datum_col = "datum", value_col = "neue_faelle" )
head(df_clean)
```

Index