Package 'visStatistics'

May 12, 2026

Type Package

Title Automated Selection and Visualisation of Statistical Hypothesis Tests

Version 0.2.0

Description The right test, visualised. 'visStatistics' automatically selects and visualises statistical hypothesis tests comparing two vectors, based on their class, distribution, and sample size.
Visual outputs, including box plots, bar charts, regression lines with confidence bands, mosaic plots, residual plots, and Q-Q plots, are annotated with relevant test statistics, assumption checks, and post-hoc analyses where applicable.
The algorithmic workflow shifts attention from ad-hoc test selection to visual diagnostic assessment and statistical interpretation.
It is particularly suited for server-side R applications, where end users interact solely through a web
interface to select data groups and receive a complete visual statistical analysis automatically. The same automation makes it useful in time-constrained contexts such as statistical consulting, where it reduces effort spent on test selection and leaves more room for interpretation.
The implemented tests cover the most frequently applied inferential methods in biomedical research (Hayat et al. (2017) <doi:10.1371/journal.pone.0179032>).
The test selection algorithm proceeds as follows: Input vectors of class numeric or integer are considered numerical; those of class factor are considered categorical; those of class ordered are considered ordinal. Assumptions of residual normality and homogeneity of variances are considered met if the corresponding test yields a p-value greater than the significance level alpha = 1 - conf.level.
(1) When the response is numerical and the predictor is categorical, a test comparing central tendencies is selected. If every group contains more than 50 observations, the sampling distribution of the group means is assumed approximately normal by the central limit theorem (Lumley et al. (2002) <doi:10.1146/annurev.publhealth.23.100901.140546>); otherwise, residual normality is assessed using shapiro.test() applied to the standardised residuals of lm(). If normality is not met, wilcox.test() is used when the

predictor has two levels and kruskal.test() followed by pairwise.wilcox.test() otherwise. If normality is met, levene.test() assesses variance homogeneity. For two-level predictors, Student's t.test(var.equal = TRUE) is applied if variances are homogeneous and Welch's t.test() otherwise. For predictors with more than two levels, aov() followed by TukeyHSD() is applied if variances are homogeneous, and oneway.test() followed by games.howell() otherwise.
(2) When both vectors are numerical, lm() is fitted by default (correlation = FALSE). If correlation = TRUE, Spearman rank correlation is performed.
(3) When the response is ordinal, it is converted to numeric ranks and the non-parametric path from (1) is followed (Wilcoxon or Kruskal-Wallis). When both variables are ordinal and correlation = TRUE, Kendall's tau_b is used instead.
(4) When both vectors are categorical, Cochran's rule (Cochran (1954) <doi:10.2307/3001666>) is applied to test independence either by chisq.test() or fisher.test().

License MIT + file LICENSE

URL https://github.com/shhschilling/visStatistics, https://shhschilling.github.io/visStatistics/

BugReports https://github.com/shhschilling/visStatistics/issues

Imports Cairo, graphics, grDevices, grid, multcompView, nortest, stats, utils, vcd

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

VignetteBuilder knitr

BuildVignettes true

Config/testthat/edition 3

Encoding UTF-8

NeedsCompilation no

Config/roxygen2/version 8.0.0

Language en-GB

Author Sabine Schilling [cre, aut, cph] (ORCID: <https://orcid.org/0000-0002-8318-9421>, year: 2026), Peter Kauf [ctb]

Maintainer Sabine Schilling <sabineschilling@gmx.ch>

Repository CRAN

Date/Publication 2026-05-12 18:40:02 UTC

Contents

bp.test 3
counts_to_cases 4

games.howell 5
gh_letters 6
levene.test 7
openGraphCairo 8
plot.visstat 10
pooled_normality_test 11
print.visstat 11
saveGraphVisstat 12
summary.visstat 13
visstat 14
visstat_core 17
vis_anova 20
vis_group_normality 22
vis_lm_assumptions 23
vis_numeric 24

Index **26**

`bp.test` *Breusch-Pagan Test for Heteroscedasticity*

Description

Performs the Breusch-Pagan test for heteroscedasticity in linear regression models. Tests the null hypothesis that the error variance is constant (homoscedasticity) against the alternative that the error variance depends on the fitted values.

Usage

```
bp.test(model)
```

Arguments

`model`	A fitted linear model object (from `lm()`).

Details

The Breusch-Pagan test regresses the squared standardized residuals on the fitted values. The test statistic is:

$$BP = n \cdot R^2$$

where:

- n = sample size
- R^2 = coefficient of determination from auxiliary regression of e_i^2 on $\hat{y}_i$

Under the null hypothesis of homoscedasticity, the test statistic follows a chi-squared distribution: $BP \sim \chi^2(p-1)$ where p is the number of parameters in the model (including intercept).

Large values of the test statistic (small p-values) provide evidence against homoscedasticity.

Value

An object of class "htest" with components:

statistic	the value of the chi-squared test statistic.
parameter	degrees of freedom.
p.value	the p-value of the test.
method	a character string indicating the test performed.
data.name	a character string giving the name of the model.

References

Breusch, T. S., and Pagan, A. R. (1979). A simple test for heteroscedasticity and random coefficient variation. Econometrica, 47(5), 1287-1294. DOI: 10.2307/1911963

Examples

```
# Example with homoscedastic errors
set.seed(123)
x <- runif(100)
y <- 2 + 3*x + rnorm(100, sd = 1)
model1 <- lm(y ~ x)
bp.test(model1)  # Should not reject (p > 0.05)

# Example with heteroscedastic errors (variance increases with x)
set.seed(456)
x <- runif(100)
y <- 2 + 3 *x + rnorm(100, sd = 0.5 + 2*x)
model2 <- lm(y ~ x)
bp.test(model2)  # Should reject (p < 0.05)
```

---

counts_to_cases

Convert data frame of counts to dataframe of cases. The data frame must contain a column with frequencies (counts) as generated by as.data.frame from a contingency table

---

Description

Convert data frame of counts to dataframe of cases. The data frame must contain a column with frequencies (counts) as generated by as.data.frame from a contingency table

Usage

```
counts_to_cases(x, countcol = "Freq")
```

Arguments

x	a data.frame of counts generated from a contingency table.
countcol	character string, name of the column of x containing the counts. Default name of the column is 'Freq'.

Value

data frame of cases of dimension (total number of counts as sum of 'Freq' in x) times 2.

Examples

```
counts_to_cases(as.data.frame(HairEyeColor[, , 1]), countcol = "Freq")
```

---

games.howell *Games-Howell Post-Hoc Test*

Description

Performs pairwise comparisons using the Games-Howell test, which does not assume equal variances or equal sample sizes. This is the appropriate post-hoc test to use after a significant Welch's ANOVA.

Usage

```
games.howell(samples, groups, conf.level = 0.95)
```

Arguments

samples	Numeric vector; the dependent variable.
groups	Factor or vector; the grouping variable.
conf.level	Numeric; confidence level for confidence intervals (default: 0.95).

Details

The Games-Howell test uses the Welch-Satterthwaite approximation for degrees of freedom and does not pool variances. P-values are adjusted using the Holm method to control family-wise error rate.

Value

A data frame with columns:

group1 First group in comparison

group2 Second group in comparison

mean_diff Difference in means (group1 - group2)

se Standard error of the difference

t t-statistic

df Degrees of freedom (Welch-Satterthwaite)

p_value Unadjusted p-value

p_adj Holm-adjusted p-value for multiple comparisons

ci_lower Lower bound of confidence interval

ci_upper Upper bound of confidence interval

significant Logical; TRUE if p_adj < (1 - conf.level)

Examples

```
# Convert dose to factor
ToothGrowth$dose <- as.factor(ToothGrowth$dose)

# Perform Games-Howell test
result <- games.howell(ToothGrowth$len, ToothGrowth$dose)
print(result)
```

`gh_letters` *Get compact letter display from Games-Howell results*

Description

Converts Games-Howell test results into compact letter display using multcompView. Groups sharing a letter are not significantly different.

Usage

```
gh_letters(x, alpha = 0.05)
```

Arguments

`x`	A games.howell object from `games.howell()`
`alpha`	Significance level (default: 0.05)

Value

A named vector with group names and their letter codes

Examples

```
# Convert dose to factor
ToothGrowth$dose <- as.factor(ToothGrowth$dose)
result <- games.howell(ToothGrowth$len, ToothGrowth$dose)
letters <- gh_letters(result)
print(letters)
```

`levene.test` *Levene-Brown-Forsythe Test for Homogeneity of Variance (center = median)*

Description

Performs Levene's test using the Brown-Forsythe modification (median-centred). It tests the null hypothesis that all groups have equal variances by testing whether the absolute deviations from group medians are equal across groups The function reproduces the default behaviour of the leveneTest(y,g,center=median,...) of the car-package.

Usage

```
levene.test(y, g, data = NULL)
```

Arguments

`y`	A numeric response vector.
`g`	A grouping factor.
`data`	Optional data frame containing 'y' and 'g'.

Details

For each observation y_{ij} in group i, compute the absolute deviation from the group median:

$$z_{ij} = |y_{ij} - \tilde{y}_i|$$

where $\tilde{y}_i$ is the median of group i.

The test statistic is the F-statistic from a one-way ANOVA on the z_{ij} values:

$$F = \frac{(N-k)\sum_{i=1}^{k} n_i(\bar{z}_i - \bar{z})^2}{(k-1)\sum_{i=1}^{k}\sum_{j=1}^{n_i}(z_{ij} - \bar{z}_i)^2}$$

where:

- k = number of groups
- N = total sample size
- n_i = sample size of group i
- $\bar{z}_i$ = mean of absolute deviations in group i
- $\bar{z}$ = overall mean of all absolute deviations

Under the null hypothesis of equal variances, the test statistic follows an F-distribution: $F \sim F(k-1, N-k)$.

Value

An object of class "htest" with components:

statistic	the value of the F-statistic.
parameter	degrees of freedom: df1=k-1, df3=N-k, where k is the number of groups and N the total sample size
p.value	the p-value of the test.
method	a character string indicating the test performed.
data.name	a character string giving the name(s) of the data.

References

Brown, M. B., and Forsythe, A. B. (1974). Robust tests for the equality of variances. Journal of the American Statistical Association, 69(346), 364–367. DOI: 10.1080/01621459.1974.10482955

Examples

```
set.seed(123)
y <- c(rnorm(10), rnorm(10, sd = 2), rnorm(10, sd = 0.5))
g <- factor(rep(1:3, each = 10))
levene.test(y, g)

# Usage with data frame
df <- data.frame(response = y, group = g)
levene.test(response, group, data = df)

# Example with unequal variances (should reject null hypothesis)
set.seed(456)
y_unequal <- c(rnorm(15, sd = 1), rnorm(15, sd = 5), rnorm(15, sd = 0.2))
g_unequal <- factor(rep(c("A", "B", "C"), each = 15))
levene.test(y_unequal, g_unequal)
```

---

openGraphCairo *Cairo wrapper function with plot capture capability*

Description

Cairo wrapper function returning NULL if not type is specified. Enhanced version that can capture plots for later replay.

Usage

```
openGraphCairo(
  width = 640,
  height = 480,
  fileName = NULL,
  type = NULL,
  fileDirectory = getwd(),
  pointsize = 12,
  bg = "transparent",
  canvas = "white",
  units = "px",
  dpi = 150
)
```

Arguments

width	see Cairo()
height	see Cairo()
fileName	name of file to be created. Does not include both file extension '.type' and file filedirectory. Default file name 'visstat_plot'.
type	Supported output types are 'png', 'jpeg', 'pdf', 'svg', 'ps' and 'tiff'. See Cairo()
fileDirectory	path of directory, where plot is stored. Default current working directory.
pointsize	see Cairo()
bg	see Cairo()
canvas	see Cairo()
units	see Cairo()
dpi	DPI used for the conversion of units to pixels. Default value 150.

Details

openGraphCairo() Cairo() wrapper function. Differences to Cairo: a) prematurely ends the function call to Cairo() returning NULL, if no output type of types 'png', 'jpeg', 'pdf', 'svg', 'ps' or 'tiff' is provided. b) The file argument of the underlying Cairo function is generated by file.path(fileDirectory,paste(fileName,'.', type, sep = '')). c) Can set up plot capture when capture_env is provided.

Value

NULL, if no type is specified. Otherwise see Cairo()

Examples

```
##  adapted from example in \code{Cairo()}
openGraphCairo(fileName = "normal_dist", type = "pdf", fileDirectory = tempdir())
plot(rnorm(4000), rnorm(4000), col = "#ff000018", pch = 19, cex = 2)
dev.off() # creates a file 'normal_dist.pdf' in the directory specified in fileDirectory
# ## remove the plot from fileDirectory
file.remove(file.path(tempdir(), "normal_dist.pdf"))
```

`plot.visstat` *Plot method for visstat objects*

Description

Replays captured plots or reports saved plot file paths from a `visstat` object.

Usage

```
## S3 method for class 'visstat'
plot(x, which = NULL, ...)
```

Arguments

`x`	An object of class `"visstat"`, returned by `visstat()`.
`which`	Integer selecting a single plot to display (1, 2, ...). If `NULL` (the default), all available plots are listed without being drawn.
`...`	Currently unused. Included for S3 method compatibility.

Details

When called without `which`, the method lists all available plots (either as file paths or as indices of captured plots). When called with `which`, the selected plot is displayed: for file-based output the stored path is printed, for captured plots the plot is replayed via `replayPlot()`.

Value

The object `x`, invisibly. Called for its side effect.

See Also

`print.visstat`, `summary.visstat`, `visstat`

Examples

```
# File-based output: plot() lists stored paths
anova_path <- visstat(
  npk$block,
  npk$yield,
  graphicsoutput = "png",
  plotDirectory = tempdir()
)

plot(anova_path)

# Interactive output: plot() lists available plots,
# plot(obj, which = n) replays a specific one
linreg <- visstat(trees$Height, trees$Girth)
```

```
plot(linreg)
plot(linreg, which = 2)
```

`pooled_normality_test` *Test normality using pooled standardized residuals*

Description

Standardizes values within each group and applies a single normality test to the pooled standardized values. This avoids multiple testing issues when deciding between parametric and non-parametric methods.

Usage

```
pooled_normality_test(y, g, test = c("shapiro", "ad"), min_n = 3L)
```

Arguments

`y`	Numeric vector of response values
`g`	Factor or vector defining groups (must have at least 2 levels)
`test`	Character, either "shapiro" or "ad" for Anderson-Darling
`min_n`	Minimum sample size per group required (default 3)

Value

A list with test results (statistic, p.value, method, data.name)

`print.visstat` *Print method for visstat objects*

Description

Displays a brief summary of the statistical test results and, if available, assumption tests and post hoc comparisons.

Usage

```
## S3 method for class 'visstat'
print(x, ...)
```

Arguments

`x`	An object of class `"visstat"`, returned by `visstat()`.
`...`	Currently unused. Included for S3 method compatibility.

Details

Quick overview of the statistical analysis results.

Value

The object x, invisibly.

See Also

summary.visstat, plot.visstat, visstat

Examples

```
anova <- visstat(npk$block, npk$yield)
print(anova)
```

---

saveGraphVisstat *Saves Graphical Output with plot capture capability*

Description

Closes all graphical devices with dev.off() and saves the output only if both fileName and type are provided. Enhanced version that can capture plots before closing devices.

Usage

```
saveGraphVisstat(
  fileName = NULL,
  type = NULL,
  fileDirectory = getwd(),
  oldfile = NULL,
  capture_env = NULL
)
```

Arguments

fileName	name of file to be created in directory fileDirectory without file extension '.type'.
type	see Cairo().
fileDirectory	path of directory, where graphic is stored. Default setting current working directory.
oldfile	old file of same name to be overwritten
capture_env	Environment to store captured plots. If NULL, no capture occurs.

Value

NULL, if no `type` or `fileName` is provided, file path if graph is created

Examples

```
# very simple KDE (adapted from example in Cairo())
openGraphCairo(type = "png", fileDirectory = tempdir())
plot(rnorm(4000), rnorm(4000), col = "#ff000018", pch = 19, cex = 2)
# save file 'norm.png' in directory specified in fileDirectory
saveGraphVisstat("norm", type = "png", fileDirectory = tempdir())
file.remove(file.path(tempdir(), "norm.png")) # remove file 'norm.png'
```

---

summary.visstat *Summary method for visstat objects*

Description

Displays the full statistical test results and, if available, assumption tests and post hoc comparisons.

Usage

```
## S3 method for class 'visstat'
summary(object, ...)
```

Arguments

`object`	An object of class `"visstat"`, returned by `visstat()`.
`...`	Currently unused. Included for S3 method compatibility.

Details

This method provides a full textual report of the statistical test results returned by `visstat()`, and prints the contents of `posthoc_summary` if present.

Value

The object `object`, invisibly.

See Also

`print.visstat`, `visstat`

Examples

```
anova <- visstat(npk$block, npk$yield)
summary(anova)
```

visstat *Wrapper for visstat_core Allowing Three Different Input Styles*

Description

visstat() is a wrapper around visstat_core that provides three alternative input styles: a formula interface, a standardised vector interface, and a backward-compatible data frame interface. visstat_core defines the decision logic for statistical hypothesis testing and visualisation between two variables of class "numeric", "integer", or "factor".

Usage

```
visstat(
  x,
  y,
  ...,
  data = NULL,
  conf.level = 0.95,
  correlation = FALSE,
  numbers = TRUE,
  minpercent = 0.05,
  graphicsoutput = NULL,
  plotName = NULL,
  plotDirectory = getwd()
)
```

Arguments

x	For the formula interface: a formula of the form y ~ x, where y is the response variable and x is the predictor or grouping variable (requires data argument). For the standardised form: a vector of class "numeric", "integer", or "factor" representing the predictor or grouping variable. For the backward-compatible form: a data.frame containing the relevant columns.
y	For the formula interface: not used (variables are extracted from the formula). For the standardised form: a vector of class "numeric", "integer", or "factor" representing the response variable. For the backward-compatible form: a character string specifying the name of the response variable column in x.
...	For the backward-compatible form only: a character string specifying the name of the predictor or grouping variable column in x. Ignored for formula and standardised input styles.
data	A data.frame containing the variables specified in the formula. Required when using the formula interface. Ignored for other input styles.
conf.level	Confidence level for statistical inference; default is 0.95.
correlation	Logical. If FALSE (default), performs simple linear regression analysis with confidence and prediction bands when both variables are numeric. If TRUE, performs Spearman correlation analysis with trend line only (no regression interpretation).

numbers	Logical. Whether to annotate plots with numeric values.
minpercent	Number between 0 and 1 indicating minimal fraction of total count data of a category to be displayed in mosaic count plots.
graphicsoutput	Saves plot(s) of type "png", "jpg", "tiff" or "bmp" in directory specified in plotDirectory. If NULL, no plots are saved.
plotName	Graphical output is stored following the naming convention "plotName.graphicsoutput" in plotDirectory. Without specifying this parameter, plotName is automatically generated following the convention "statisticalTestName_varsample_varfactor".
plotDirectory	Specifies directory where generated plots are stored. Default is current working directory.

Details

This wrapper supports three input formats:

(1) Formula interface: `visstat(y ~ x, data = df)`, where the formula specifies the response (y) and predictor (x) variables, and data is a data frame containing these variables.

(2) Standardised form: `visstat(x, y)`, where both x and y are vectors of class "numeric", "integer", or "factor". Here x is the predictor or grouping variable and y is the response variable.

(3) Backward-compatible form: `visstat(dataframe, "name_of_y", "name_of_x")`, where the first character string refers to the response variable and the second to the predictor or grouping variable. Both must be column names in dataframe.

The interpretation of x and y depends on the variable classes. Throughout, data of class numeric or integer are referred to as numeric, while data of class factor are referred to as categorical:

If one variable is numeric and the other a factor, the numeric vector is the response (y) and the factor is the grouping variable (x). This supports tests of central tendencies (e.g., t-test, Welch's ANOVA, Wilcoxon, Kruskal-Wallis).

If both variables are numeric, a linear model is fitted with y as the response and x as the predictor.

If both variables are factors, an association test (Chi-squared or Fisher's exact) is used. The test result is invariant to variable order, but visualisations (e.g., axis layout, bar orientation) depend on the roles of x and y.

This wrapper standardises the input and calls visstat_core, which selects and executes the appropriate test with visual output and assumption diagnostics.

Value

An object of class "visstat" containing the results of the automatically selected statistical test. The specific contents depend on which test was performed. Additionally, the returned object includes two attributes:

- plot_paths: Character vector of file paths where plots were saved (if graphicsoutput was specified)
- captured_plots: List of captured plot objects for programmatic access

In case of insufficient data, returns a list with an error element and basic input summary information.

Note

For best visualization, ensure that the RStudio Plots pane is adequately sized. If you get "figure margins too large" errors, try expanding the Plots pane in RStudio, using `dev.new(width=10, height=6)` for a larger plot window, or reducing the `cex` parameter.

See Also

`visstat_core` defining the decision logic, the package's vignette `vignette("visStatistics")` explaining the decision logic accompanied by illustrative examples, and the accompanying webpage https://shhschilling.github.io/visStatistics/.

Examples

```
# Formula interface
mtcars$am <- as.factor(mtcars$am)
visstat(mpg ~ am, data = mtcars)

# Standardised usage
visstat(mtcars$am, mtcars$mpg)

# Backward-compatible usage (same result as above)
visstat(mtcars, "mpg", "am")

## Student's t-test (equal variances, two groups)
# When residuals are normally distributed and Levene's test indicates
# homoscedasticity, the classic Student's t-test with pooled variance is used
df <- droplevels(subset(PlantGrowth, group %in% c("ctrl", "trt1")))
visstat(df$group,df$weight)

## Welch's t-test (unequal variances, two groups)
# When residuals are normally distributed but Levene's test indicates
# heteroscedasticity, Welch's t-test is used
visstat(mtcars$am, mtcars$mpg)

## Wilcoxon rank sum test (non-normal, two groups)
# When residuals are not normally distributed
grades_gender <- data.frame(
  Sex = as.factor(c(rep("Girl", 20), rep("Boy", 20))),
  Grade = c(
    19.3, 18.1, 15.2, 18.3, 7.9, 6.2, 19.4, 20.3, 9.3, 11.3,
    18.2, 17.5, 10.2, 20.1, 13.3, 17.2, 15.1, 16.2, 17.3, 16.5,
    5.1, 15.3, 17.1, 14.8, 15.4, 14.4, 7.5, 15.5, 6.0, 17.4,
    7.3, 14.3, 13.5, 8.0, 19.5, 13.4, 17.9, 17.7, 16.4, 15.6
  )
)
visstat(grades_gender$Sex, grades_gender$Grade)

## Fisher's ANOVA (equal variances, >2 groups)
# When residuals are normally distributed and Levene's test indicates
# homoscedasticity, classic Fisher's ANOVA with TukeyHSD post-hoc is used.
# Different green letters indicate significant differences between groups.
visstat(PlantGrowth$group, PlantGrowth$weight)
```

```
## Welch's one-way ANOVA (unequal variances, >2 groups)
set.seed(123)
values <- c(rnorm(20, 10, 1),rnorm(20, 15, 5),rnorm(20, 12, 2))
groups <- factor(rep(c("A", "B", "C"), each = 20))
visstat(groups, values)
## Kruskal-Wallis (non-normal, >2 groups)
# When residuals are not normally distributed, kruskal.test() is followed by
# pairwise.wilcox.test.
visstat(iris$Species, iris$Petal.Width)

## Simple linear regression (both numeric)
visstat(trees$Height, trees$Girth, conf.level = 0.99)

## Pearson's Chi-squared test (both factors, large expected counts)
HairEyeColorDataFrame <- counts_to_cases(as.data.frame(HairEyeColor))
visstat(HairEyeColorDataFrame$Eye, HairEyeColorDataFrame$Hair)

## Fisher's exact test (both factors, small expected counts)
HairEyeColorMaleFisher <- HairEyeColor[, , 1]
blackBrownHazelGreen <- HairEyeColorMaleFisher[1:2, 3:4]
blackBrownHazelGreen <- counts_to_cases(as.data.frame(blackBrownHazelGreen))
visstat(blackBrownHazelGreen$Eye, blackBrownHazelGreen$Hair)

## Save PNG
visstat(blackBrownHazelGreen$Hair, blackBrownHazelGreen$Eye,
        graphicsoutput = "png", plotDirectory = tempdir())

## Custom plot name
visstat(iris$Species, iris$Petal.Width,
        graphicsoutput = "pdf", plotName = "kruskal_iris", plotDirectory = tempdir())
```

`visstat_core` *Automated Visualization of Statistical Hypothesis Testing*

Description

`visstat_core()` provides automated selection and visualization of a statistical hypothesis test between a two vectors in a given `data.frame` named `dataframe` based on the data's type, distribution, sample size, and the specified `conf.level`. `visstat_core()` is called by the main wrapper function `visstat()`. `varsample` and `varfactor` are `character` strings corresponding to the column names of the chosen vectors in `dataframe`. These vectors must be of type `integer`, `numeric` or `factor`. The automatically generated output figures illustrate the selected statistical hypothesis test, display the main test statistics, and include assumption checks and post hoc comparisons when applicable. The primary test results are returned as a list object.

Usage

```
visstat_core(
```

```
  dataframe,
  varsample,
  varfactor,
  conf.level = 0.95,
  correlation = FALSE,
  numbers = TRUE,
  minpercent = 0.05,
  graphicsoutput = NULL,
  plotName = NULL,
  plotDirectory = getwd()
)
```

Arguments

dataframe	data.frame with at least two columns.
varsample	character string matching a column name in dataframe. Interpreted as the response if the referenced column is of class numeric or integer and the column named by varfactor is of class factor.
varfactor	character string matching a column name in dataframe. Interpreted as the grouping variable if the referenced column is of class factor and the column named by varsample is of class numeric or integer.
conf.level	Confidence level
correlation	Logical. If FALSE (default), performs simple linear regression analysis with confidence and prediction bands. If TRUE, performs Spearman correlation analysis with trend line only (no regression interpretation).
numbers	a logical indicating whether to show numbers in mosaic count plots.
minpercent	number between 0 and 1 indicating minimal fraction of total count data of a category to be displayed in mosaic count plots.
graphicsoutput	saves plot(s) of type "png", "jpg", "tiff" or "bmp" in directory specified in plotDirectory. If graphicsoutput=NULL, no plots are saved.
plotName	graphical output is stored following the naming convention "plotName.graphicsoutput" in plotDirectory. Without specifying this parameter, plotName is automatically generated following the convention "statisticalTestName_varsample_varfactor".
plotDirectory	specifies directory, where generated plots are stored. Default is current working directory.

Details

The decision logic for selecting a statistical test is described below. For more details, please refer to the package's vignette("visStatistics"). Throughout, data of class numeric or integer are referred to as numeric, while data of class factor are referred to as categorical. The significance level alpha is defined as one minus the confidence level, given by the argument conf.level. Assumptions of normality and homoscedasticity are considered met when the corresponding test yields a p-value greater than alpha = 1 - conf.level. The choice of statistical tests performed by visstat_core() depends on whether the data are numeric or categorical, the number of levels in the categorical variable, the distribution of the data, and the chosen conf.level. The function

prioritises interpretable visual output and tests that remain valid under their assumptions, following the logic below:

(1) When the response is numerical and the predictor is categorical, tests of central tendencies are performed. For the decision logic, please refer to the packages vignette `vignette("visStatistics")`

(2): When both the response and predictor are numerical, a linear model `lm()` is fitted, with residual diagnostics and a confidence band plot.

(3): When both variables are categorical, `visstat_core()` uses `chisq.test()` or `fisher.test()` depending on expected counts, following Cochran's rule (Cochran (1954) <doi:10.2307/3001666>).

Implemented main tests:

`t.test()`, `wilcox.test()`, `aov()`, `oneway.test()`, `lm()`, `kruskal.test()`, `fisher.test()`, `chisq.test()`.

Implemented tests for assumptions:

- Normality: `shapiro.test()` and `ad.test()`
- Heteroscedasticity: `bartlett.test()` and `levene.test()` and `bp_test()`

Implemented post hoc tests:

- `TukeyHSD()` for `aov()`
- `games.howell` for `oneway.test()`
- `pairwise.wilcox.test()` for `kruskal.test()`

Value

An object of class `"visstat"` containing the results of the automatically selected statistical test. The specific contents depend on which test was performed. Additionally, the returned object includes two attributes:

- `plot_paths`: Character vector of file paths where plots were saved (if `graphicsoutput` was specified)
- `captured_plots`: List of captured plot objects for programmatic access

See Also

The package's vignette `vignette("visStatistics")` for a description of the decision logic, illustrated with numerous examples. The package is accompanied by its webpage https://shhschilling.github.io/visStatistics/. The main function `visstat` for a detailed description of the return value.

Examples

```
# Welch Two Sample t-test (t.test())
visstat_core(mtcars, "mpg", "am")

## Wilcoxon rank sum test (wilcox.test())
grades_gender <- data.frame(
  Sex = as.factor(c(rep("Girl", 20), rep("Boy", 20))),
  Grade = c(
```

```
    19.3, 18.1, 15.2, 18.3, 7.9, 6.2, 19.4,
    20.3, 9.3, 11.3, 18.2, 17.5, 10.2, 20.1, 13.3, 17.2, 15.1, 16.2, 17.3,
    16.5, 5.1, 15.3, 17.1, 14.8, 15.4, 14.4, 7.5, 15.5, 6.0, 17.4,
    7.3, 14.3, 13.5, 8.0, 19.5, 13.4, 17.9, 17.7, 16.4, 15.6
  )
)
visstat_core(grades_gender, "Grade", "Sex")

## Welch's oneway ANOVA not assuming equal variances (oneway.test())
anova_npk <- visstat_core(npk, "yield", "block")
anova_npk # prints summary of tests

## Kruskal-Wallis rank sum test (kruskal.test())
visstat_core(iris, "Petal.Width", "Species")
visstat_core(InsectSprays, "count", "spray")

## Simple linear regression  (lm())
visstat_core(trees, "Girth", "Height", conf.level = 0.99)

## Pearson's Chi-squared test (chisq.test())
### Transform array to data.frame
HairEyeColorDataFrame <- counts_to_cases(as.data.frame(HairEyeColor))
visstat_core(HairEyeColorDataFrame, "Hair", "Eye")

## Fisher's exact test (fisher.test())
HairEyeColorMaleFisher <- HairEyeColor[, , 1]
### slicing out a 2 x2 contingency table
blackBrownHazelGreen <- HairEyeColorMaleFisher[1:2, 3:4]
blackBrownHazelGreen <- counts_to_cases(as.data.frame(blackBrownHazelGreen))
fisher_stats <- visstat_core(blackBrownHazelGreen, "Hair", "Eye")
```

---

vis_anova *ANOVA or Welch's ANOVA with appropriate post-hoc tests*

Description

Internal function that performs ANOVA or Welch's one-way test and corresponding post-hoc comparisons. Uses TukeyHSD for equal variances (Fisher's ANOVA) and Games-Howell for unequal variances (Welch's ANOVA).

Usage

```
vis_anova(
  samples,
  fact,
  conf.level = conf.level,
  samplename = "",
  factorname = "",
  cex = 1
)
```

Arguments

samples	Numeric vector; the dependent variable.
fact	Factor; the grouping variable.
conf.level	Numeric; confidence level for tests and intervals (default: 0.95).
samplename	Character; label for y-axis (default: "").
factorname	Character; label for x-axis (default: "").
cex	Numeric; character expansion factor for plot elements (default: 1).

Details

The function first tests for homogeneity of variance using Levene's test. If variances are equal (p > alpha), Fisher's one-way ANOVA with Tukey's HSD post-hoc is performed. If variances are unequal (p <= alpha), Welch's one-way ANOVA with Games-Howell post-hoc is performed.

The function produces a box plot with jittered points and group means (red diamonds for the parametric branches), annotated with a compact letter display showing which groups differ significantly.

Value

A list with components:

summary statistics of ANOVA Summary of Fisher's ANOVA or Welch's oneway test

post-hoc analysis TukeyHSD object or Games-Howell results in compatible format

conf.level The confidence level used

Examples

```
# Example with equal variances (uses Fisher's ANOVA + TukeyHSD)
data(PlantGrowth)
result1 <- vis_anova(PlantGrowth$weight, PlantGrowth$group,
                     samplename = "Weight", factorname = "Group")

# Example with unequal variances (uses Welch's ANOVA + Games-Howell)
# Create data with heterogeneous variances
set.seed(123)
group_a <- rnorm(20, mean = 10, sd = 1)
group_b <- rnorm(20, mean = 15, sd = 5)  # Much larger variance
group_c <- rnorm(20, mean = 12, sd = 2)
values <- c(group_a, group_b, group_c)
groups <- factor(rep(c("A", "B", "C"), each = 20))
result2 <- vis_anova(values, groups,
                     samplename = "Value", factorname = "Group")
```

`vis_group_normality` *Visualisation of the normality assumption for Welch ANOVA/t-test*

Description

`vis_group_normality` checks for normality of each group separately using the Shapiro-Wilk and Anderson-Darling tests. The null hypothesis is that each group is normally distributed. The function generates histograms with normal distribution overlays and Q-Q plots to visually assess normality. The layout is always 2 rows × k columns (histograms on top, Q-Q plots on bottom).

Usage

```
vis_group_normality(
  samples,
  groups,
  conf.level = 0.95,
  samplename = "",
  groupname = "",
  cex = 1
)
```

Arguments

`samples`	Numeric vector; the dependent variable.
`groups`	Factor or vector; the grouping variable (2 to 8 groups for visual display).
`conf.level`	Numeric; confidence level (default: 0.95). Used to determine alpha = 1 - conf.level for normality test interpretation.
`samplename`	Character; label for the y-axis (default: "").
`groupname`	Character; label for the x-axis (default: "").
`cex`	Numeric; scaling factor for plot text and symbols (default: 1).

Details

Layout is always 2 rows × k columns:

- Top row: Histograms with normal overlay for each group
- Bottom row: Q-Q plots for each group

For more than 8 groups, a tabular summary is provided instead of plots.

Value

A list containing:

shapiro_tests List of Shapiro-Wilk test results for each group

ad_tests List of Anderson-Darling test results for each group

n_groups Number of groups

group_names Names of the groups

Examples

```
# Two groups (like t-test)
vis_group_normality(ToothGrowth$len, ToothGrowth$supp)

# Three groups
ToothGrowth$dose <- as.factor(ToothGrowth$dose)
vis_group_normality(ToothGrowth$len, ToothGrowth$dose)
```

`vis_lm_assumptions` *Visualisation of the normality distribution of the standardised residuals*

Description

Checks for normality of the standardised residuals in the general linear model Student's t-test (t.test,var=EQUAL) Fisher oneway ANOVA (aov) or simple linear regression. Performs the Shapiro-Wilk test and Anderson-Darling test for normality and, if not a regression, also the Levene-Brown-Forsythe and the Bartlett's test for homogeneity of variances. It produces a histogram with normal overlay, a residuals vs fitted plot, and a normal Q-Q plot.

Usage

```
vis_lm_assumptions(samples, fact, cex = 1, regression = FALSE)

vis_anova_assumptions(...)
```

Arguments

`samples`	Numeric vector; the dependent variable.
`fact`	Factor; the independent variable.
`cex`	Numeric; scaling factor for plot text and symbols (default: 1).
`regression`	Logical; if TRUE, skips Bartlett's test (for regression diagnostics). Default is FALSE.
`...`	Arguments passed to `vis_lm_assumptions()`.

Value

A list with elements:

summary_anova Summary of the ANOVA model.

shapiro_test Result from `shapiro.test()`.

ad_test Result from `nortest::ad.test()` or a character message if n < 7.

levene_test Result from `levene.test()` (only if `regression = FALSE`).

bartlett_test Result from `bartlett.test()` (only if `regression = FALSE`).

bp_test Result from `bp.test()` (only if `regression = TRUE`).

Examples

```
ToothGrowth$dose <- as.factor(ToothGrowth$dose)
vis_lm_assumptions(ToothGrowth$len, ToothGrowth$dose)
```

`vis_numeric` *Visualize Numeric Relationships: Regression or Correlation Analysis*

Description

This function provides unified visualization for numeric relationships between two continuous variables. It can perform either regression analysis (with confidence and prediction bands) or Spearman rank correlation analysis with appropriate visualizations and statistical output. For regression, statistical assumptions are checked and warnings are issued if violated, but analysis proceeds.

Usage

```
vis_numeric(
  y,
  x,
  correlation = FALSE,
  conf.level = 0.95,
  name_of_factor = character(),
  name_of_sample = character()
)
```

Arguments

`y`	Numeric vector. The response variable (dependent variable) for regression analysis, or the y-axis variable for correlation analysis.
`x`	Numeric vector. The predictor variable (independent variable) for regression analysis, or the x-axis variable for correlation analysis. Must have the same length as y.
`correlation`	Logical. If FALSE (default), performs regression analysis with confidence and prediction bands. If TRUE, performs Spearman rank correlation analysis.
`conf.level`	Numeric. Confidence level for statistical tests and intervals. Must be between 0 and 1. Default is 0.95 (95 percent confidence level).
`name_of_factor`	Character string. Label for the x-axis (independent variable). If empty, defaults to the variable name.
`name_of_sample`	Character string. Label for the y-axis (dependent variable). If empty, defaults to the variable name.

Details

Statistical Assumptions Checked: Regression: Normality of residuals (Shapiro-Wilk test) and homoscedasticity (Breusch-Pagan test). All regression analyses proceed even if assumptions are violated, but appropriate warnings are issued. Correlation: Spearman rank correlation requires no distributional assumptions.

Value

A list containing analysis results and assumption checks. Content depends on analysis type. For regression analysis: analysis_type, summary_regression, assumptions, warnings, r_squared, adj_r_squared. For correlation analysis: analysis_type, correlation_test, correlation_coefficient, assumptions, warnings, method_used.

Author(s)

Sabine Schilling

See Also

cor.test, lm, shapiro.test

Examples

```
## Not run:
# Generate sample data
set.seed(123)
x <- rnorm(50, mean = 10, sd = 2)
y <- 2 * x + rnorm(50, mean = 0, sd = 1)

# Regression analysis (default)
result1 <- vis_numeric(y, x,
                       name_of_factor = "Predictor",
                       name_of_sample = "Response")

# Spearman rank correlation
result2 <- vis_numeric(y, x, correlation = TRUE)

## End(Not run)
```

Index

bp.test, 3

cor.test, *25*
counts_to_cases, 4

games.howell, 5
gh_letters, 6

levene.test, 7
lm, *25*

openGraphCairo, 8

plot.visstat, 10, *12*
pooled_normality_test, 11
print.visstat, *10*, 11, *13*

saveGraphVisstat, 12
shapiro.test, *25*
summary.visstat, *10*, *12*, 13

vis_anova, 20
vis_anova_assumptions
        *(*vis_lm_assumptions*)*, 23
vis_group_normality, 22
vis_lm_assumptions, 23
vis_numeric, 24
visstat, *10*, *12, 13*, 14, *19*
visstat_core, *14–16*, 17