Package 'jmuOutlier'

May 8, 2026

Type Package

Title Permutation Tests for Nonparametric Statistics

Version 2.2

Author Steven T. Garren

Maintainer Steven T. Garren <GARRENST@JMU.EDU>

Date 2019-08-05

Description Performs a permutation test on the difference between two location parameters, a permutation correlation test, a permutation F-test, the Siegel-Tukey test, a ratio mean deviance test. Also performs some graphing techniques, such as for confidence intervals, vector addition, and Fourier analysis; and includes functions related to the Laplace (double exponential) and triangular distributions. Performs power calculations for the binomial test.

License GPL-3

Depends R (>= 2.0)

Suggests agricolae, coin, fastGraph

NeedsCompilation no

Repository CRAN

Date/Publication 2019-08-05 20:40:02 UTC

Contents

`jmuOutlier-package` *Permutation Tests for Nonparametric Statistics*

Description

Performs a permutation test on the difference between two location parameters, a permutation correlation test, a permutation F-test, the Siegel-Tukey test, a ratio mean deviance test. Also performs some graphing techniques, such as for confidence intervals, vector addition, and Fourier analysis; and includes functions related to the Laplace (double exponential) and triangular distributions. Performs power calculations for the binomial test.

Details

(I) Permutation tests

- `perm.cor.test` performs a permutation test based on Pearson and Spearman correlations.
- `perm.f.test` performs a permutation F-test and a one-way analysis of variance F-test.
- `perm.test` performs one-sample and two-sample permutation tests on vectors of data.
- `rmd.test` performs a permutation test based on the estimated RMD, the ratio of the mean of the absolute value of the deviances, using two datasets.
- `siegel.test` performs the Siegel-Tukey test using two datasets.

(II) Confidence intervals

- `CI.t.test` produces two-sided confidence intervals on population mean, allowing for a finite population correction.
- `quantileCI` produces exact confidence intervals on quantiles corresponding to the stated probabilities, based on the binomial test.

(III) Graphs

- `coin.toss` illustrates the Law of Large Numbers for proportions.
- `fourier` determines the Fourier approximation for any function on domain $(0, 2\pi)$ and then graphs both the function and the approximation.
- `lineGraph` constructs a line graph on a vector of numerical observations.
- `plotCI` plots multiple confidence intervals on the same graph, and determines the proportion of confidence intervals containing the true population mean.
- `plotEcdf` graphs one or two empirical cumulative distribution functions on the same plot.
- `plotVector` plots one or two 2-dimensional vectors along with their vector sum.
- `truncHist` produces a truncated histogram, which may be useful if data contain some extreme outliers.

(IV) Laplace (double exponential) and symmetric triangular distributions

- `dlaplace`, `plaplace`, `qlaplace`, and `rlaplace` give the density, the distribution function, the quantile function, and random deviates, respectively, of the Laplace distribution.
- `dtriang`, `ptriang`, `qtriang`, and `rtriang` give the density, the distribution function, the quantile function, and random deviates, respectively, of the triangular distribution.

(V) Reading datasets

- `read.table2` reads table of data from author's website.
- `scan2` scans data from author's website.

(VI) Additional functions

- `abbreviation` determines if one character variable is an abbreviation among a selection of other character variables.
- `latin` generates a Latin square.
- `power.binom.test` computes the power of the binomial test of a simple null hypothesis about a population median.
- `score` generates van der Waerden scores (i.e., normal quantiles) and exponential (similar to Savage) scores.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

R-package `coin` for additional permutation tests, and *R*-package `fastGraph`.

Examples

```
print( x <- rtriang(20,50) )

perm.test( x, mu=25, stat=median )

quantileCI( x, c(0.25, 0.5, 0.75)  )

power.binom.test( 20, 0.05, "less", 47, plaplace, 45.2, 3.7 )

fourier (function(x){ (x-pi)^3 }, 4 )
```

---

abbreviation *Allows Abbreviations of Character Data*

Description

Determines if one character variable is an abbreviation among a section of other character variables.

Usage

```
abbreviation(x, choices)
```

Arguments

x	A character string, and consists of some or all letters in a value in choices or may equal choices.
choices	A vector of character strings.

Details

The function abbreviation returns a value in choices specified by x, which may be an abbreviation. If no such abbreviation exists, then the original value of x is returned.

Value

The value in choices, which can be abbreviated by x.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

Examples

```
choices = c("two.sided", "less", "greater")

abbreviation( "two", choices )

abbreviation( "l", choices )

abbreviation( "gr", choices )

abbreviation( "greater", choices )

abbreviation( "Not in choices", choices )
```

CI.t.test *Student's t-Confidence Interval with Finite Population Correction*

Description

Performs two-sided confidence interval on population mean, allowing for a finite population correction.

Usage

```
CI.t.test(x, conf.level = 0.95, fpc = 1)
```

Arguments

x	A nonempty numeric vector of data values.
conf.level	Confidence level of the interval, and should be between 0 and 1.
fpc	The finite population correction, and should be between 0 and 1.

Details

The fpc is typically defined as $1 - n/N$, where n is the sample size, and N is the population size, for simple random sampling without replacement. When sampling with replacement, set fpc=1 (default).

Value

A confidence interval for the population mean.

Note

The definition of fpc is based on the textbook by Scheaffer, Mendenhall, Ott, Gerow (2012), chapter 4.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Scheaffer, R. L., Mendenhall, W., Ott, R. L., Gerow, K. G. (2012) *Elementary Survey Sampling,* 7th edition.

See Also

`t.test` and `plotCI`.

Examples

```
# Sample 43 observations from a population of 200 numbers, and compute the 95% confidence interval.
pop = sqrt(1:200) ; x1 = sample( pop, 43 ) ; print(sort(x1))

CI.t.test( x1, fpc = 1-length(x1)/length(pop) )

# Sample 14 observations from a Normal(mean=50, sd=5) distribution,
#    and compute the 90% confidence interval.
x2 = rnorm( 14, 50, 5 ) ; print(sort(x2))

CI.t.test( x2, 0.9 )
```

`coin.toss` *Coin Toss*

Description

Graphs a simulation of the sample proportion of heads.

Usage

```
coin.toss(n, p=0.5, burn.in=0, log.scale=FALSE, col=c("black","red"), ...)
```

Arguments

`n`	An integer denoting the number of times the coin is tossed.
`p`	The probability of heads, which must be between 0 and 1.
`burn.in`	An integer denoting the number of initial coin tosses which should be omitted from the graph.
`log.scale`	Logical; indicating whether or not the x-axis should have a logarithmic scale.
`col`	A vector of two colors, where the first color is used for the graph of the sample proportions, and the second color is used for the horizontal line occurring at the value p.
`...`	Optional arguments to be passed to the `plot` function (see `par`).

Details

This function `coin.toss` illustrates the Law of Large Numbers for proportions, by simulating cumulative sample proportions. Using nonzero `burn.in` typically reveals greater precision in the graph as the number of coin tosses increases.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

Examples

```
par( mfrow=c(2,2) )
coin.toss( 600, 0.5 )
coin.toss( 3e4, 0.4, )
coin.toss( 3e4, 0.7, 1000, col=c("hotpink","turquoise") )
coin.toss( 7e4, 0.3, 1000, TRUE, col=c("purple","green") )
par( mfrow=c(1,1) )
```

---

dlaplace *Laplace (Double Exponential) Density Function*

Description

Laplace (double exponential) density with mean equal to `mean` and standard deviation equal to `sd`.

Usage

```
dlaplace(x, mean = 0, sd = 1)
```

Arguments

`x`	Vector of quantiles.
`mean`	Population mean.
`sd`	Population standard deviation.

Details

The Laplace distribution has density $e^{-|x-\mu|\sqrt{2}/\sigma}/(\sigma\sqrt{2})$, where μ is the mean of the distribution and σ is the standard deviation.

Value

`dlaplace` gives the density.

Note

The formulas computed within `dlaplace` are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

plaplace, qlaplace, and rlaplace.

Examples

```
dlaplace( seq( 20, 80, length.out=11 ), 50, 10 )
```

---

dtriang *Triangular Density Function*

Description

Symmetric triangular density with endpoints equal to min and max.

Usage

```
dtriang(x, min = 0, max = 1)
```

Arguments

x	Vector of quantiles.
min	Left endpoint of the triangular distribution.
max	Right endpoint of the triangular distribution.

Details

The triangular distribution has density $4(x-a)/(b-a)^2$ for $a \leq x \leq \mu$, and $4(b-x)/(b-a)^2$ for $\mu < x \leq b$, where a and b are the endpoints, and the mean of the distribution is $\mu = (a+b)/2$.

Value

dtriang gives the density.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

ptriang, qtriang, and rtriang.

Examples

```
dtriang( seq( 100, 200, length.out=11 ), 100, 200 )
```

`fourier` *Determining and Graphing Fourier Approximation*

Description

The Fourier approximation is determined for any function on domain $(0, 2\pi)$ and then graphed.

Usage

```
fourier(f, order = 3, ...)
```

Arguments

`f`	The function to be approximated by Fourier analysis.
`order`	Integer; the order of the Fourier transformation.
`...`	Optional arguments to be passed to the `plot` function (see `par`).

Details

The numerical output consists of $a_0/2, a_1, ..., a_n, b_1, ..., b_2$. The equation is (constant) $+a_1 cos(x) + ... + a_n cos(nx) + b_1 sin(x) + ... + b_n sin(nx)$.

Value

`constant`	The constant term.
`cosine.coefficients`	The coefficients for the cosine terms.
`sine.coefficients`	The coefficients for the sine terms.

Note

The formulas computed within `fourier` are based on the textbook by Larson (2013).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Larson, R. (2013) *Elementary Linear Algebra*, 7th edition.

Examples

```
par( mfrow=c(2,2) )
fourier( function(x){ exp(-x)*(x-pi) }, 4 )
fourier( function(x){ exp(-x) }, 7 )
fourier( function(x){ (x-pi) }, 5 )
fourier( function(x){ (x-pi)^2 } )
par( mfrow=c(1,1) )
```

`latin` *Latin Square*

Description

Generates a Latin square, which is either standard or based on randomized rows and columns.

Usage

```
latin(n, random = TRUE)
```

Arguments

`n`	An integer between 2 and 26, inclusively, denoting the number of treatment groups.
`random`	Logical; if `TRUE` (default), a Latin square with randomized rows and columns is produced. If `FALSE`, a standard non-random Latin square is produced.

Details

The Latin square is produced in matrix format with treatments labeled as *A*, *B*, *C*, etc.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

`design.lsd` in *R*-package `agricolae`

Examples

```
latin( 5, random=FALSE )
latin( 6 ) # Default is random=TRUE
```

lineGraph *Line Graph Plotting*

Description

Constructs a line graph.

Usage

```
lineGraph(x, freq = TRUE, prob = NULL, col = "red", ...)
```

Arguments

x	Vector of numerical observations to be graphed.
freq	Logical; if freq is FALSE or prob sums to 1, then relative frequencies are graphed; otherwise, frequencies are graphed.
prob	Vector of the probabilities or weights on x, and does not need to sum to one. If prob is NULL, then all x values are equally weighted. Also, see freq.
col	The color of the plotted lines. Type colors() for selections.
...	Optional arguments to plot.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

hist

Examples

```
par( mfrow=c(2,2) )
lineGraph( c( rep(6,4), rep(9,7), rep(3,5), 5, 8, 8 ) )
lineGraph( c( rep(6,4), rep(9,7), rep(3,5), 5, 8, 8 ), FALSE, col="purple" )
lineGraph( 11:14, , c( 12, 9, 17, 5 ), col="blue" )
lineGraph( 0:10, FALSE, dbinom(0:10,10,0.4), col="darkgreen",
   main="Binomial(n=10,p=0.4) probabilities" )
par( mfrow=c(1,1) )
```

perm.cor.test *Permutation Test on Correlation*

Description

A permutation test is performed based on Pearson and Spearman correlations.

Usage

```
perm.cor.test(x, y = NULL, alternative = c("two.sided", "less", "greater"),
              method = c("pearson", "spearman"), num.sim = 20000)
```

Arguments

x	Numeric vector of design variable if y is not NULL, or N by 2 data frame or matrix of design and response variables if y is NULL.
y	Numeric vector of response variable, and should be NULL if x is an N by 2 data frame or matrix.
alternative	A character string specifying the alternative hypothesis, and must be one of "two.sided" (default), "greater" or "less". Only the initial letter needs to be specified.
method	A character string specifying the type of correlation, and must be one of "pearson" (default) or "spearman". Only the initial letter needs to be specified.
num.sim	The number of simulations generated.

Details

The p-value is estimated by randomly generating the permutations, and is hence not exact. The larger the value of num.sim the more precise the estimate of the p-value, but also the greater the computing time. Thus, the p-value is not based on asymptotic approximation. The output states more details about the permutation test, such as the values of method and num.sim.

Value

alternative	Same as the input.
p.value	The p-value of the permutation test.

Note

The formulas computed within perm.cor.test are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

cor, cor.test, and perm.test.

Examples

```
perm.cor.test( c( 4, 6, 8, 11 ), c( 19, 44, 15, 13 ), "less", "pearson" )
perm.cor.test( c( 4, 6, 8, 11 ), c( 19, 44, 15, 13 ), "less", "spearman" )
```

---

perm.f.test *Permutation Test on the F-statistic*

Description

A permutation F-test is performed, and a one-way analysis of variance F-test is performed.

Usage

```
perm.f.test(response, treatment = NULL, num.sim = 20000)
```

Arguments

response	Numeric vector of responses if treatment is not NULL. If treatment is NULL, then response must be an N by 2 data frame or matrix, such that the first column represents response and the second column represents treatment.
treatment	Vector of treatments, which need not be numerical. If response is an N by 2 data frame or matrix, then treatment should be set to NULL.
num.sim	The number of simulations performed. If num.sim is smaller than one, then the permutation test is not performed.

Details

The one-way analysis of variance F-test is performed, regardless of the value of num.sim. The permutation F-test is performed whenever num.sim is at least 1. The p-value of the permutation F-test is estimated by randomly generating the permutations, and is hence not exact. The larger the value of num.sim the more precise the estimate of the p-value of the permutation F-test, but also the greater the computing time. Thus, the p-value of the permutation F-test is not based on asymptotic approximation.

Value

The output consists of results from calling aov and from the permutation F-test.

Note

The formulas computed within `perm.f.test` are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

`aov` and `perm.test`.

Examples

```
perm.f.test( c( 14,6,5,2,54,7,9,15,11,13,12 ), rep( c("I","II","III"), c(4,4,3) ) )
```

`perm.test` *Permutation Test*

Description

Performs one-sample and two-sample permutation tests on vectors of data.

Usage

```
perm.test(x, y = NULL, alternative = c("two.sided", "less", "greater"), mu = 0,
    paired = FALSE, all.perms = TRUE, num.sim = 20000, plot = FALSE, stat = mean, ...)
```

Arguments

`x`	A (non-empty) numeric vector of data values.
`y`	An optional numeric vector data values.
`alternative`	A character string specifying the alternative hypothesis, and must be one of `"two.sided"` (default), `"greater"` or `"less"`. Only the initial letter needs to be specified.
`mu`	A number indicating the null value of the location parameter (or the difference in location parameters if performing a two-sample test).
`paired`	Logical, indicating whether or not a two-sample test should be paired, and is ignored for a one-sample test.
`all.perms`	Logical. The exact p-value is attempted when `all.perms` (i.e., all permutations) is `TRUE` (default), and is simulated when `all.perms` is `FALSE` or when computing an exact p-value requires more than `num.sim` calculations.
`num.sim`	The upper limit on the number of permutations generated.

plot	Logical. If TRUE, then plot the histogram of the permutation distribution; otherwise, list the p-value.
stat	Function, naming the test statistic, such as mean and median.
...	Optional arguments to stat; and is the second argument to stat when unspecified. For example, if stat equals mean, then the second argument trim denotes the fraction (0 to 0.5) of observations to be trimmed from each end of x and y before the mean is computed.

Details

A paired test using data x and nonNULL y is equivalent to a one-sample test using data x-y. The output states more details about the permutation test, such as one-sample or two-sample, and whether or not the p.value calculated was based on all permutations.

Value

alternative	Same as the input.
mu	Same as the input.
p.value	The p-value of the permutation test.

Note

The formulas computed within perm.test are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

Examples

```
# One-sample test

print( x <- rnorm(10,0.5) )

perm.test( x, stat=median )

# Two-sample unpaired test

print( y <- rnorm(13,1) )

perm.test( x, y )
```

plaplace *Laplace (Double Exponential) Cumulative Distribution Function*

Description

Laplace (double exponential) cumulative distribution function with mean equal to `mean` and standard deviation equal to `sd`.

Usage

```
plaplace(q, mean = 0, sd = 1, lower.tail = TRUE)
```

Arguments

`q`	Vector of quantiles.
`mean`	Population mean.
`sd`	Population standard deviation.
`lower.tail`	Logical; if `TRUE` (default), probabilities are $P[X \leq x]$; otherwise, $P[X > x]$.

Details

The Laplace distribution has density $e^{-|x-\mu|\sqrt{2}/\sigma}/(\sigma\sqrt{2})$, where μ is the mean of the distribution and σ is the standard deviation.

Value

`plaplace` gives the distribution function.

Note

The formulas computed within `plaplace` are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

`dlaplace`, `qlaplace`, and `rlaplace`.

Examples

```
plaplace( seq( 20, 80, length.out=11 ), 50, 10 )

plaplace( seq( 20, 80, length.out=11 ), 50, 10, FALSE )
```

plotCI *Confidence Interval Plot*

Description

Plots multiple confidence intervals on the same graph, and determines the proportion of confidence intervals containing the true population mean.

Usage

```
plotCI(CI, mu = NULL, plot.midpoints = TRUE,
       col = c("black", "red", "darkgreen", "purple"))
```

Arguments

CI	N by 2 matrix or 2 by N matrix consisting of N two-sided confidence intervals.
mu	Numeric; the population mean, and is NULL if unknown.
plot.midpoints	Logical; plots the midpoints of the confidence intervals if TRUE (default); otherwise, does not plot the midpoints.
col	A vector of size four, specifying the colors of the line representing population mean, confidence intervals not containing the population mean, confidence intervals containing the population mean, and the sample means, respectively.

Details

The title of the graph states the proportion of the confidence intervals containing the true population mean, when the population mean is not NULL.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

CI.t.test

Examples

```
# Plot fifty 90% confidence intervals, each based on 13 observations from a
#    Normal( mean=70, sd=10 ) distribution.

plotCI( replicate( 50, t.test( rnorm( 13, 70, 10 ), conf.level=0.9 )$conf.int ), 70 )
```

plotEcdf *Plotting Two Empirical Cumulative Distribution Functions*

Description

Graphs one or two empirical cumulative distribution functions on the same plot.

Usage

```
plotEcdf(x, y = NULL, col = c("black", "red"))
```

Arguments

x	Vector of numerical observations whose empirical cdf is to be graphed.
y	Optional vector of observations whose empirical cdf is to be graphed.
col	Scalar or vector of length two, specifying the colors of the two empirical distribution functions. The two colors correspond to x and y, respectively, and preferably should differ. Type colors() for selections.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

plot.ecdf

Examples

```
par( mfrow=c(2,2) )
plotEcdf( c(2,4,9,6), c(1,7,11,3,8) )
plotEcdf( c(2,4,9,6), c(1,7,11,3), col=c("navyblue", "orange") )
plotEcdf( c(11,5,3), c(3,7,9), col=c("tomato","darkgreen") )
plotEcdf( c(15,19,11,4,6), col="purple" )
par( mfrow=c(1,1) )
```

plotVector *Plotting Vector Addition*

Description

Plots one or two 2-dimensional vectors along with their vector sum.

Usage

```
plotVector(x1, y1, x2 = NULL, y2 = NULL, add.vectors = TRUE,
           col = c("black", "red", "darkgreen", "purple"), lwd = 8, font = 2,
           font.lab = 2, las = 1, cex.lab = 1.3, cex.axis = 2, usr = NULL, ...)
```

Arguments

x1	Value on the x-axis of the first vector.
y1	Value on the y-axis of the first vector.
x2	Value on the x-axis of the second vector.
y2	Value on the y-axis of the second vector.
add.vectors	Logical; if TRUE (default), display the vector sum.
col	A vector of size four, specifying the colors of the first vector, the second vector, the vector sum, and parallel lines, respectively. Type colors() for selections.
lwd	The line width of the vectors.
font	An integer specifying which font to use for text.
font.lab	The font to be used for x and y labels.
las	Numeric in (0,1,2,3); the style of axis labels.
cex.lab	The magnification to be used for x and y labels.
cex.axis	The magnification to be used for axis annotation.
usr	A vector of the form c(x1, x2, y1, y2) giving the extremes of the user coordinates of the plotting region.
...	Optional arguments to be passed to the plot function (see par).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

plot and curve.

Examples

```
par( mfrow=c(2,2) )

# Vectors (2,8) and (4,-3) and their vector sum.
     plotVector( 2, 8, 4, -3 )

# Colinear vectors (-3,6) and (-1,2).
     plotVector( -3, 6, -1, 2, add=FALSE, col=c("red","black") )

# Colinear vectors (-1,2) and (3,-6).
     plotVector( -1, 2, 3, -6, add=FALSE )

# Vectors (2,3) and (5,-4)
```

```
    plotVector( 2, 3, 5, -4, add=FALSE, usr=c( -5, 5, -4, 7) )

par( mfrow=c(1,1) )
```

---

power.binom.test *Power Calculations for Exact Binomial Test*

Description

Compute the power of the binomial test of a simple null hypothesis about a population median.

Usage

```
power.binom.test(n, alpha = 0.05, alternative = c("two.sided", "less", "greater"),
                 null.median, alt.pdist, ...)
```

Arguments

n	The sample size.
alpha	Probability of Type I error.
alternative	A character string specifying the alternative hypothesis, and must be one of "two.sided" (default), "greater" or "less". Only the initial letter needs to be specified.
null.median	The population median under the null hypothesis.
alt.pdist	Name of the cumulative distribution function under the alternative distribution. Some options include pnorm, pexp, pcauchy, plaplace, pt, pchisq, pf, ptriang, punif, pbinom, pgeom, ppois.
...	Optional arguments to alt.pdist, excluding the first argument of alt.pdist. See the examples below.

Value

Power of the test.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

power.t.test

Examples

```
# Alternative distribution is Normal( mean=55.7, sd=2.5 ).
power.binom.test( 30, 0.05, "greater", 55, pnorm, 55.7, 2.5 )

# Alternative distribution is Laplace( mean=55.7, sd=2.5 ).
power.binom.test( 30, 0.05, "greater", 55, plaplace, 55.7, 2.5 )
```

---

ptriang *Triangular Cumulative Distribution Function*

Description

Triangular cumulative distribution function with endpoints equal to `min` and `max`.

Usage

```
ptriang(q, min = 0, max = 1, lower.tail = TRUE)
```

Arguments

`q`	Vector of quantiles.
`min`	Left endpoint of the triangular distribution.
`max`	Right endpoint of the triangular distribution.
`lower.tail`	Logical; if `TRUE` (default), probabilities are $P[X \leq x]$; otherwise, $P[X > x]$.

Details

The triangular distribution has density $4(x-a)/(b-a)^2$ for $a \leq x \leq \mu$, and $4(b-x)/(b-a)^2$ for $\mu < x \leq b$, where a and b are the endpoints, and the mean of the distribution is $\mu = (a+b)/2$.

Value

`ptriang` gives the distribution function.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

`dtriang`, `qtriang`, and `rtriang`.

Examples

```
ptriang( seq( 100, 200, length.out=11 ), 100, 200 )

ptriang( seq( 100, 200, length.out=11 ), 100, 200, FALSE )
```

qlaplace *Laplace (Double Exponential) Quantile Function*

Description

Laplace (double exponential) quantile function with mean equal to mean and standard deviation equal to sd.

Usage

```
qlaplace(p, mean = 0, sd = 1, lower.tail = TRUE)
```

Arguments

p	Vector of probabilities.
mean	Population mean.
sd	Population standard deviation.
lower.tail	Logical; if TRUE (default), probabilities are $P[X \leq x]$; otherwise, $P[X > x]$.

Details

The Laplace distribution has density $e^{-|x-\mu|\sqrt{2}/\sigma}/(\sigma\sqrt{2})$, where μ is the mean of the distribution and σ is the standard deviation.

Value

qlaplace gives the quantile function.

Note

The formulas computed within qlaplace are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

dlaplace, plaplace, and rlaplace.

Examples

```
# 5th, 15th, 25th, ..., 95th percentiles from a Laplace( 50, 10 ) distribution.

qlaplace( seq( 0.05, 0.95, length.out=11 ), 50, 10 )
```

qtriang *Triangular Quantile Function*

Description

Symmetric triangular density with endpoints equal to `min` and `max`.

Usage

```
qtriang(p, min = 0, max = 1)
```

Arguments

`p`	Vector of probabilities.
`min`	Left endpoint of the triangular distribution.
`max`	Right endpoint of the triangular distribution.

Details

The triangular distribution has density $4(x-a)/(b-a)^2$ for $a \le x \le \mu$, and $4(b-x)/(b-a)^2$ for $\mu < x \le b$, where a and b are the endpoints, and the mean of the distribution is $\mu = (a+b)/2$.

Value

`qtriang` gives the quantile function.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

`dtriang`, `ptriang`, and `rtriang`.

Examples

```
# 5th, 15th, 25th, ..., 95th percentiles from a Triangular( 100, 200 ) distribution.

qtriang( seq( 0.05, 0.95, length.out=11 ), 100, 200 )
```

quantileCI *Confidence Intervals on Quantiles*

Description

Produces exact confidence intervals on quantiles corresponding to the stated probabilities, based on the binomial test.

Usage

```
quantileCI(x, probs = 0.5, conf.level = 0.95)
```

Arguments

x	Numeric vector of observations.
probs	Numeric vector of cumulative probabilities between 0 and 1.
conf.level	Confidence level of the interval.

Details

If probs=0.5 (default), then a confidence interval on the population median is produced.

Value

Confidence interval for each quantile based on probs.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

Examples

```
# Sample 20 observations from an Exponential distribution with mean=10.

print( sort( x <- rexp( 20, 0.1 ) ) )

# Construct 90% confidence intervals on the 25th, 50th, and 75th percentiles.

quantileCI( x, c( 0.25, 0.5, 0.75 ), 0.9 )
```

read.table2 *Reads Tables from Author's Website*

Description

Performs read.table of dataset without typing the URL.

Usage

```
read.table2(file.name, course.num=course.number, na.strings=".", ...)
```

Arguments

file.name	The file name in character format without the URL.
course.num	The course number in character or numeric format, where course.number is a global variable.
na.strings	Character vector. Elements of this vector are to be interpreted as missing NA values.
...	Optional arguments to be passed to the read.table function.

Details

The datasets are available on the author's website, http://educ.jmu.edu/~garrenst. The global variable course.number may be entered as the value of the second argument, course.num, in function read.table2.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

read.table and scan2

Examples

```
# The following two commands, when uncommented, are equivalent.

# read.table2( "ex6.1.txt", 321, header=TRUE )

# read.table( "http://educ.jmu.edu/~garrenst/math321.dir/datasets/ex6.1.txt", header=TRUE )
```

`rlaplace` *Laplace (Double Exponential) Random Generation*

Description

Laplace (double exponential) random generation with mean equal to `mean` and standard deviation equal to `sd`.

Usage

```
rlaplace(n, mean = 0, sd = 1)
```

Arguments

`n`	Number of observations. If `length(n)>1`, the length is taken to be the number required.
`mean`	Population mean.
`sd`	Population standard deviation.

Details

The Laplace distribution has density $e^{-|x-\mu|\sqrt{2}/\sigma}/(\sigma\sqrt{2})$, where μ is the mean of the distribution and σ is the standard deviation.

Value

`rlaplace` generates random deviates.

Note

The formulas computed within `rlaplace` are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

`dlaplace`, `plaplace`, and `qlaplace`.

Examples

```
# 20 random variates from a Laplace( 50, 10 ) distribution.

rlaplace( 20, 50, 10 )
```

`rmd.test` *Ratio Mean Deviance Test*

Description

A permutation test is performed based on the estimated RMD, the ratio of the mean of the absolute value of the deviances, for data `x` and `y`.

Usage

```
rmd.test(x, y, alternative = c("two.sided", "less", "greater"), all.perms = TRUE,
         num.sim = 20000)
```

Arguments

`x`	Numeric vector of data values.
`y`	Numeric vector of data values.
`alternative`	A character string specifying the alternative hypothesis, and must be one of `"two.sided"` (default), `"greater"` or `"less"`. Only the initial letter needs to be specified.
`all.perms`	Logical. The exact p-value is attempted when `all.perms` (i.e., all permutations) is `TRUE` (default), and is simulated when `all.perms` is `FALSE` or when computing an exact p-value requires more than `num.sim` calculations.
`num.sim`	The upper limit on the number of permutations generated.

Value

`alternative`	Same as the input.
`rmd.hat`	The value of the RMD test statistic.
`p.value`	The p-value of the test.

Note

The formulas computed within `rmd.test` are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

`ansari.test`, `siegel.test`, and `perm.test`

Examples

```
rmd.test( c(13, 34, 2, 19, 49, 63), c(17, 29, 22) )
rmd.test( c(13, 34, 2, 19, 49, 63), c(17, 29, 22), "greater" )
```

---

rtriang *Triangular Random Generation*

Description

Symmetric triangular random generation with endpoints equal to `min` and `max`.

Usage

```
rtriang(n, min = 0, max = 1)
```

Arguments

`n`	Number of observations. If `length(n)>1`, the length is taken to be the number required.
`min`	Left endpoint of the triangular distribution.
`max`	Right endpoint of the triangular distribution.

Details

The triangular distribution has density $4(x-a)/(b-a)^2$ for $a \leq x \leq \mu$, and $4(b-x)/(b-a)^2$ for $\mu < x \leq b$, where a and b are the endpoints, and the mean of the distribution is $\mu = (a+b)/2$.

Value

`rtriang` generates random deviates.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

`dtriang`, `ptriang`, and `qtriang`.

Examples

```
# 20 random variates from a Triangular( 100, 200 ) distribution.

rtriang( 20, 100, 200 )
```

scan2 *Scans Data from Author's Website*

Description

Performs scan of dataset without typing the URL.

Usage

```
scan2(file.name, course.num=course.number, na.strings=".", comment.char="#", ...)
```

Arguments

file.name	The file name in character format without the URL.
course.num	The course number in character or numeric format, where course.number is a global variable.
na.strings	Character vector. Elements of this vector are to be interpreted as missing NA values.
comment.char	Single character or empty string, denoting beginning of comment. Use "" to turn off the interpretation of comments altogether.
...	Optional arguments to be passed to the scan function.

Details

The datasets are available on the author's website, http://educ.jmu.edu/~garrenst. The global variable course.number may be entered as the value of the second argument, course.num, in function scan2.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

read.table2 and scan

Examples

```
# The following two commands, when uncommented, are equivalent.

# scan2( "exercise2.7.txt", 324 )

# scan( "http://educ.jmu.edu/~garrenst/math324.dir/datasets/exercise2.7.txt", comment.char="#" )
```

score *Generating van der Waerden and Exponential Scores*

Description

Generates van der Waerden scores (i.e., normal quantiles) and exponential (similar to Savage) scores, for combined data x and y.

Usage

```
score(x, y = NULL, expon = FALSE)
```

Arguments

x	A positive integer equal to the number of desired scores when y is NULL, or x is a vector of observations.
y	An optional vector of observations, typically used with two-sample tests.
expon	Logical; if FALSE (default), van der Waerden scores are computed, even for ties. If TRUE, Exponential scores are computed, and interpolation is used for ties.

Details

The scored values for x are the output, when y is NULL.

Value

x	Scored values for x, when y is not NULL.
y	Scored values for y, when y is not NULL.

Note

The formulas computed within score are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

Examples

```
score( 10 )

score( 15, expon=TRUE )

score( c(4,7,6,22,13), c(15,16,7) )  # Two samples, including a tie.
```

siegel.test *Siegel-Tukey Test*

Description

Performs the Siegel-Tukey test on data x and y, where ties are handled by averaging ranks, not by asymptotic approximations.

Usage

```
siegel.test(x, y, alternative = c("two.sided", "less", "greater"), reverse = FALSE,
            all.perms = TRUE, num.sim = 20000)
```

Arguments

x	Numeric vector of data values.
y	Numeric vector of data values.
alternative	A character string specifying the alternative hypothesis, and must be one of "two.sided" (default), "greater" or "less". Only the initial letter needs to be specified.
reverse	Logical; If FALSE (default), then assign rank 1 to the smallest observation. If TRUE, then assign rank 1 to the largest observation.
all.perms	Logical. The exact p-value is attempted when all.perms (i.e., all permutations) is TRUE (default), and is simulated when all.perms is FALSE or when computing an exact p-value requires more than num.sim calculations.
num.sim	The upper limit on the number of permutations generated.

Details

Since the logical value of reverse may affect the p-value, yet neither logical value of reverse is preferred over the other, one should consider using ansari.test instead.

Value

alternative	Same as the input.
rank.x	The Siegel-Tukey ranks of the data x.
rank.y	The Siegel-Tukey ranks of the data y.
p.value	The p-value of the test.

Note

The formulas computed within siegel.test are based on the textbook by Higgins (2004).

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

References

Higgins, J. J. (2004) *Introduction to Modern Nonparametric Statistics*.

See Also

ansari.test, rmd.test, and perm.test

Examples

```
# The same data are used in the following two commands.

siegel.test( c(13, 34, 2, 19, 49, 63), c(17, 29, 22) )
siegel.test( c(13, 34, 2, 19, 49, 63), c(17, 29, 22), reverse=TRUE )
```

---

truncHist *Truncated Histograms*

Description

Produces a truncated histogram.

Usage

```
truncHist(x, xmin = NULL, xmax = NULL, trim = 0.025, main = NULL, xlab = "x", ...)
```

Arguments

x	Vector of numerical observations.
xmin	Minimum numerical value to be shown in graph.
xmax	Maximum numerical value to be shown in graph.
trim	The fraction (0 to 0.5) of observations to be trimmed from each end of x before the histogram is constructed.
main	An overall title for the histogram.
xlab	A title for the x-axis.
...	Optional arguments to hist.

Details

truncHist may be useful if data contain some extreme outliers.

Author(s)

Steven T. Garren, James Madison University, Harrisonburg, Virginia, USA

See Also

hist

Examples

```
x1 = sort(rnorm(1000)) ;   c( head(x1), tail(x1))

x2 = sort(rnorm(1000)) ;   c( head(x2), tail(x2))

y1 = sort(rcauchy(1000)) ; c( head(y1), tail(y1))

y2 = sort(rcauchy(1000)) ; c( head(y2), tail(y2))

par( mfrow=c(2,2) )
truncHist(x1, main="Normal data; first simulation",  xlab="x1")
truncHist(x2, main="Normal data; second simulation", xlab="x2")
truncHist(y1, main="Cauchy data; first simulation",  xlab="y1")
truncHist(y2, main="Cauchy data; second simulation", xlab="y2")
par( mfrow=c(1,1) )
```

Index