

# Package: CarletonStats (via r-universe)

October 25, 2024

**Title** Functions for Statistics Classes at Carleton College

**Version** 2.2

**Description** Includes commands for bootstrapping and permutation tests, a command for created grouped bar plots, and a demo of the quantile-normal plot for data drawn from different distributions.

**License** GPL-2

**URL** <https://github.com/aloy/CarletonStats>

**BugReports** <https://github.com/aloy/CarletonStats/issues>

**Suggests** grDevices, MASS, testthat

**Encoding** UTF-8

**LazyData** TRUE

**NeedsCompilation** no

**RoxygenNote** 7.2.3

**Imports** ggplot2, scales, patchwork

**Repository** <https://aloy.r-universe.dev>

**RemoteUrl** <https://github.com/aloy/carletonstats>

**RemoteRef** HEAD

**RemoteSha** 21f5c65b9fe7ba76bf1917704345a124f44a66ae

## Contents

anovaSummarized . . . . .	2
boot . . . . .	3
bootCor . . . . .	5
bootPaired . . . . .	7
bootSlope . . . . .	9
confint.carlboot . . . . .	11
confIntDemo . . . . .	11
corDemo . . . . .	12

groupedBar . . . . .	13
Icecream . . . . .	14
Milkshakes . . . . .	15
missingLevel . . . . .	16
permTest . . . . .	16
permTestAnova . . . . .	18
permTestCor . . . . .	20
permTestPaired . . . . .	22
permTestSlope . . . . .	24
plot.carlboot . . . . .	26
print.carlboot . . . . .	26
qqPlotDemo . . . . .	27
states03 . . . . .	28
stemPlot . . . . .	29
summary.carlboot . . . . .	30

<b>Index</b>	<b>32</b>
--------------	-----------

---

anovaSummarized	<i>Anova F test</i>
-----------------	---------------------

---

### Description

ANOVA F test when given summarized data (sample sizes, means and standard deviations).

### Usage

```
anovaSummarized(N, mn, stdev)
```

### Arguments

N	a vector with the sample sizes
mn	a vector of means, one for each group in the sample
stdev	a vector of standard deviations, one for each group in the sample

### Details

Perform an ANOVA F test when presented with summarized data: sample sizes, sample means and sample standard deviations.

### Value

Returns invisibly a list

Treatment SS     The treatment sum of squares (also called the "between sum of squares").

Residual SS     Residual sum of squares (also called the "within sum of squares").

Degrees of Freedom  
                     a vector with the numerator and denominator degrees of freedom.

Treatment Mean Square	Treatment SS/numerator DF
Residual Mean Square	Residual SS/denominator DF
Residual Standard Error	Square root of Residual Mean Square
F	the F statistic
P-value	p-value
...	

**Author(s)**

Laura Chihara

**Examples**

```
#use the data set chickwts from base R
head(chickwts)

N <- table(chickwts$feed)
stdev <- tapply(chickwts$weight, chickwts$feed, sd)
mn <- tapply(chickwts$weight, chickwts$feed, mean)

anovaSummarized(N, mn, stdev)
```

---

 boot

*Bootstrap*


---

**Description**

Bootstrap a single variable or a grouped variable

**Usage**

```
boot(x, ...)

## Default S3 method:
boot(
  x,
  group = NULL,
  statistic = mean,
  conf.level = 0.95,
  B = 10000,
  plot.hist = TRUE,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
```

```

    xlab = NULL,
    ylab = NULL,
    title = NULL,
    seed = NULL,
    ...
)

## S3 method for class 'formula'
boot(formula, data, subset, ...)

```

### Arguments

x	a numeric vector
...	further arguments to be passed to or from methods.
group	an optional grouping variable (vector), usually a factor variable. If it is a binary numeric variable, it will be coerced to a factor.
statistic	function that computes the statistic of interest. Default is the mean.
conf.level	confidence level for the bootstrap percentile interval. Default is 95%.
B	number of times to resample (positive integer greater than 2).
plot.hist	logical value. If TRUE, plot the histogram of the bootstrap distribution.
plot.qq	Logical value. If TRUE, create a normal quantile-quantile plot of the bootstrap distribution.
x.name	Label for variable name
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
seed	optional argument to <a href="#">set.seed</a>
formula	a formula $y \sim g$ where $y$ is a numeric vector and $g$ a factor variable with two levels. If $g$ is a binary numeric vector, it will be coerced to a factor variable. For a single numeric variable, formula may also be $\sim y$ .
data	a data frame that contains the variables given in the formula.
subset	an optional expression indicating what observations to use.

### Details

Perform a bootstrap of a statistic applied to a single variable, or to the difference of the statistic computed on two samples (using the grouping variable). If  $x$  is a binary vector of 0's and 1's and the function is the mean, then the statistic of interest is the proportion.

Observations with missing values are removed.

### Value

A vector with the resampled statistics is returned invisibly.

**Methods (by class)**

- `boot(default)`: Bootstrap a single variable or a grouped variable
- `boot(formula)`: Bootstrap a single variable or a grouped variable

**Author(s)**

Laura Chihara

**References**

Tim Hesterberg's website <https://www.timhesterberg.net/bootstrap-and-resampling>

**Examples**

```
#ToothGrowth data (supplied by R)
#bootstrap mean of a single numeric variable
boot(ToothGrowth$len)

#bootstrap difference in mean of tooth length for two groups.
boot(ToothGrowth$len, ToothGrowth$supp, B = 1000)

#same as above using formula syntax
boot(len ~ supp, data = ToothGrowth, B = 1000)
```

---

bootCor

*Bootstrap the correlation*

---

**Description**

Bootstrap the correlation of two numeric variables.

**Usage**

```
bootCor(x, ...)

## Default S3 method:
bootCor(
  x,
  y,
  conf.level = 0.95,
  B = 10000,
  plot.hist = TRUE,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
```

```

    y.name = deparse(substitute(y)),
    seed = NULL,
    ...
)

## S3 method for class 'formula'
bootCor(formula, data, subset, ...)

```

### Arguments

<code>x</code>	a numeric vector.
<code>...</code>	further arguments to be passed to or from methods.
<code>y</code>	a numeric vector.
<code>conf.level</code>	confidence level for the bootstrap ercentile interval.
<code>B</code>	number of times to resample (positive integer greater than 2).
<code>plot.hist</code>	a logical value. If TRUE, plot the bootstrap distribution of the resampled correlation.
<code>xlab</code>	an optional character string for the x-axis label
<code>ylab</code>	an optional character string for the y-axis label
<code>title</code>	an optional character string giving the plot title
<code>plot.qq</code>	a logical value. If TRUE a normal quantile-quantile plot of the bootstrapped values is created.
<code>x.name</code>	Label for variable x
<code>y.name</code>	Label for variable y
<code>seed</code>	optional argument to <a href="#">set.seed</a>
<code>formula</code>	a formula of the form lhs ~ rhs where lhs is a numeric variable giving the data values and rhs a factor with two levels giving the corresponding groups.
<code>data</code>	an optional data frame containing the variables in the formula formula. By default the variables are taken from environment(formula).
<code>subset</code>	an optional vector specifying a subset of observations to be used.

### Details

Bootstrap the correlation of two numeric variables. The bootstrap mean and standard error are printed as well as a bootstrap percentile confidence interval.

Observations with missing values are removed.

### Value

The command returns the correlations of the resampled observations.

### Methods (by class)

- `bootCor(default)`: Bootstrap the correlation of two numeric variables.
- `bootCor(formula)`: Bootstrap the correlation of two numeric variables.

**Author(s)**

Laura Chihara

**References**Tim Hesterberg's website <https://www.timhesterberg.net/bootstrap-and-resampling>**Examples**

```
plot(states03$ColGrad, states03$InfMortality)
bootCor(InfMortality ~ ColGrad, data = states03, B = 1000)
bootCor(states03$ColGrad, states03$InfMortality, B = 1000)
```

---

`bootPaired`*Bootstrap paired data*

---

**Description**

Perform a bootstrap of two paired variables.

**Usage**

```
bootPaired(x, ...)

## Default S3 method:
bootPaired(
  x,
  y,
  conf.level = 0.95,
  B = 10000,
  plot.hist = TRUE,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
  y.name = deparse(substitute(y)),
  seed = NULL,
  ...
)

## S3 method for class 'formula'
bootPaired(formula, data, subset, ...)
```

**Arguments**

x	a numeric vector.
...	further arguments to be passed to or from methods.
y	a numeric vector.
conf.level	confidence level for the bootstrap percentile interval.
B	number of resamples (positive integer greater than 2).
plot.hist	logical. If TRUE, plot the histogram of the bootstrap distribution.
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
plot.qq	logical. If TRUE, a normal quantile-quantile plot of the replicates will be created.
x.name	Label for variable x
y.name	Label for variable y
seed	optional argument to <a href="#">set.seed</a>
formula	a formula $y \sim x$ where x, y are both numeric vectors
data	a data frame that contains the variables given in the formula.
subset	an optional expression indicating what observations to use.

**Details**

The command will compute the difference of x and y and bootstrap the difference. The mean and standard error of the bootstrap distribution will be printed as well as a bootstrap percentile interval.

Observations with missing values are removed.

**Value**

The command returns a vector with the replicates of the statistic being bootstrapped.

**Methods (by class)**

- `bootPaired(default)`: Perform a bootstrap of two paired variables.
- `bootPaired(formula)`: Perform a bootstrap of two paired variables.

**Author(s)**

Laura Chihara

**References**

Tim Hesterberg's website <https://www.timhesterberg.net/bootstrap-and-resampling>

**Examples**

```
#Bootstrap the mean difference of fat content in vanilla and chocolate ice
#cream. Data are paired because ice cream from the same manufacturer will
#have similar content.
Icecream
bootPaired(ChocFat ~ VanillaFat, data = Icecream)
bootPaired(Icecream$VanillaFat, Icecream$ChocFat)
```

---

bootSlope

*Bootstrap the slope of a simple linear regression line*


---

**Description**

Bootstrap the slope of a simple linear regression line. The bootstrap mean and standard error are printed as well as a bootstrap percentile confidence interval.

**Usage**

```
bootSlope(x, ...)
```

## Default S3 method:

```
bootSlope(
  x,
  y,
  conf.level = 0.95,
  B = 10000,
  plot.hist = TRUE,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
  y.name = deparse(substitute(y)),
  seed = NULL,
  ...
)
```

## S3 method for class 'formula'

```
bootSlope(formula, data, subset, ...)
```

**Arguments**

x                    a numeric vector.

...                  further arguments to be passed to or from methods.

y                    a numeric vector.

<code>conf.level</code>	confidence level for the bootstrap percentile interval.
<code>B</code>	number of times to resample (positive integer greater than 2).
<code>plot.hist</code>	a logical value. If TRUE, plot the bootstrap distribution of the resampled slope.
<code>xlab</code>	an optional character string for the x-axis label
<code>ylab</code>	an optional character string for the y-axis label
<code>title</code>	an optional character string giving the plot title
<code>plot.qq</code>	a logical value. If TRUE a normal quantile-quantile plot of the bootstrapped values is created.
<code>x.name</code>	Label for variable x
<code>y.name</code>	Label for variable y
<code>seed</code>	optional argument to <code>set.seed</code>
<code>formula</code>	a formula of the form <code>lhs ~ rhs</code> where <code>lhs</code> is a numeric variable giving the data values and <code>rhs</code> a factor with two levels giving the corresponding groups.
<code>data</code>	an optional data frame containing the variables in the formula <code>formula</code> . By default the variables are taken from <code>environment(formula)</code> .
<code>subset</code>	an optional vector specifying a subset of observations to be used.

### Details

Observations with missing values are removed.

### Value

The command returns the slopes of the resampled observations.

### Methods (by class)

- `bootSlope(default)`: Bootstrap the slope of a simple linear regression line
- `bootSlope(formula)`: Bootstrap the slope of a simple linear regression line

### Author(s)

Adam Loy, Laura Chihara

### References

Tim Hesterberg's website <https://www.timhesterberg.net/bootstrap-and-resampling>

### Examples

```
plot(states03$ColGrad, states03$InfMortality)
bootSlope(InfMortality ~ ColGrad, data = states03, B = 1000)
bootSlope(states03$ColGrad, states03$InfMortality, B = 1000)
```

---

confint.carlboot	<i>Calculate a CI from a carlboot object</i>
------------------	--

---

**Description**

Calculate percentile confidence intervals for a carlboot object.

**Usage**

```
## S3 method for class 'carlboot'
confint(object, parm = NULL, level = 0.95, ...)
```

**Arguments**

object	The carlboot object to print.
parm	not used in CarletonStats, just for generic consistency
level	the confidence level
...	not used

---

confIntDemo	<i>Confidence Interval Demonstration</i>
-------------	--

---

**Description**

Draw many random samples and compute confidence interval. How many intervals capture the true mean?

**Usage**

```
confIntDemo(distr = "normal", size = 20, conf.level = 0.95)
```

**Arguments**

distr	distribution of the population to be sampled. Options include "normal", "exponential", "uniform" and "binary" (partial match allowed).
size	sample size
conf.level	confidence level.

**Details**

This simulation will draw 100 random samples from a given population distribution and compute the corresponding confidence intervals. The 100 intervals will be drawn with an indication of the ones that missed the true mean. A histogram of the population will also be created.

**Value**

The command invisibly returns the fraction of intervals that capture the true mean.

**Author(s)**

Laura Chihara

**Examples**

```
confIntDemo()  
  
confIntDemo(distr = "exponential", size = 40)
```

---

corDemo

*Correlation demonstration*

---

**Description**

For a given  $r$ , create a scatterplot of two variables with that correlation.

**Usage**

```
corDemo(r = 0)
```

**Arguments**

$r$  a number between -1 and 1. Enter any number  $r$ , *latex*, to exit the interactive session[

**Details**

Demonstrate the concept of correlation by inputting a number between -1 and 1 and seeing a scatter plot of two variables with that correlation. Once you invoke this command, you can continue to enter values for  $r$ . Type any number *latex* to exit.

**Author(s)**

Laura Chihara

**Examples**

```
## Not run:  
corDemo()  
  
## End(Not run)
```

---

groupedBar	<i>Grouped bar chart</i>
------------	--------------------------

---

### Description

Create a bar chart of a single categorical variable or a grouped bar chart of two categorical variables.

### Usage

```
groupedBar(resp, ...)

## Default S3 method:
groupedBar(
  resp,
  condvar = NULL,
  percent = TRUE,
  print = TRUE,
  cond.name = deparse(substitute(condvar)),
  resp.name = deparse(substitute(resp)),
  ...
)

## S3 method for class 'formula'
groupedBar(formula, data = parent.frame(), subset, ...)
```

### Arguments

resp	a factor variable. If resp is numeric, it will be coerced to a factor variable.
...	further arguments to be passed to or from methods.
condvar	a factor variable to condition on. If NULL, then a bar plot of just the resp variable will be created. If condvar is numeric, it will be coerced to a factor variable.
percent	a logical value. Should the y-axis give percent or counts?
print	a logical value. If TRUE, print out the table.
cond.name	Label for variable condvar.
resp.name	Label for variable resp.
formula	a formula of the form $x \sim \text{condvar}$ . If x or condvar is (are) not a factor variable, then it (they) will be coerced into one. Formula can also be $\sim x$ for a single factor variable.
data	a data frame that contains the variables in the formula.
subset	an optional vector specifying a subset of observations to be used.

### Details

For a single factor variable, a bar plot. If two factor variables are given, then a bar plot of x conditioned by condvar. This command uses R's table command so missing values are automatically removed.

**Value**

Returns invisibly a table of the variable(s).

**Methods (by class)**

- `groupedBar(default)`: Grouped bar chart
- `groupedBar(formula)`: Grouped bar chart

**Author(s)**

Laura Chihara

**Examples**

```
groupedBar(states03$Region)

## Not run:
groupedBar(states03$DeathPenalty, states03$Region, legend.loc = "topleft")

#Using a formula syntax:

groupedBar(~Region, data = states03)
groupedBar(DeathPenalty ~ Region, data = states03, legend.loc = "topleft")

## End(Not run)
```

---

Icecream

*Ice cream data*

---

**Description**

Nutritional information on vanilla and chocolate ice cream from a sample of companies.

**Format**

A data frame with 39 observations on the following 7 variables.

**Brand** Brand name

**VanillaCalories** Calories per serving in vanilla

**VanillaFat** Fat per serving (g) in vanilla

**VanillaSugar** Sugar per serving (g) in vanilla

**ChocCalories** Calories per serving in chocolate

**ChocFat** Fat per serving (g) in chocolate

**ChocSugar** Sugar per serving (g) in chocolate

**Source**

Data collected by Carleton student Ann Butkowski (2008).

**Examples**

```
head(Icecream)
t.test(Icecream$VanillaCalories, Icecream$ChocCalories, paired = TRUE)
```

---

Milkshakes	<i>Milkshakes (chocolate) Nutritional information on chocolate milkshakes from a sample of restaurants.</i>
------------	---

---

**Description**

Milkshakes (chocolate) Nutritional information on chocolate milkshakes from a sample of restaurants.

**Format**

A data frame with 29 observations on the following 11 variables.

**Restaurant** Names of restaurants

**Type** Type of restaurant, Dine In Fast Food

**Calories** Calories per serving

**Fat** Fat per serving (g)

**Sodium** Sodium per serving (mg)

**Carbs** Carbohydrates per serving (g)

**SizeOunces** Size of milkshake (ounces)

**CalPerOunce** Calories per ounce

**FatPerOunce** Fat per ounce

**CarbsPerOunce** Carbohydrates per ounce

**Source**

Data collected by Carleton students Yoni Blumberg (2013) and Lindsay Guthrie (2013).

---

missingLevel	<i>Missing observations in factors</i>
--------------	--

---

**Description**

In data frames with factor variables, convert any observation with "" into <NA>.

**Usage**

```
missingLevel(data)
```

**Arguments**

data            a data frame with factor variables.

**Details**

In a factor variable with the level "" "", this command will convert this to an <NA>.

**Value**

Returns the same data frame with "" "" replaced by <NA> in factor variables.

**Note**

When importing data from comma separated files (for example), missing values in a categorical variable are often denoted by "". We often do not want to treat this as a level of a factor variable in R.

**Author(s)**

Laura Chihara

---

permTest	<i>Permutation test</i>
----------	-------------------------

---

**Description**

Permutation test to test a hypothesis involving two samples.

**Usage**

```
permTest(x, ...)

## Default S3 method:
permTest(
  x,
  group,
  statistic = mean,
  B = 9999,
  alternative = "two.sided",
  plot.hist = TRUE,
  plot.qq = FALSE,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  seed = NULL,
  ...
)

## S3 method for class 'formula'
permTest(formula, data = parent.frame(), subset, ...)
```

**Arguments**

x	a numeric vector. If the function is the mean (fun = mean) and x is a binary numeric vector of 0's and 1's, then the test is between proportions.
...	further arguments to be passed to or from methods.
group	a factor variable with two levels. If group is a binary numeric vector, it will be coerced into a factor variable.
statistic	the statistic of interest.
B	the number of resamples (positive integer greater than 2).
alternative	the alternative hypothesis. Options are "two.sided", "less" or "greater".
plot.hist	a logical value. If TRUE, the permutation distribution of the statistic is plotted.
plot.qq	a logical value. If TRUE, then a normal quantile-quantile plot of the resampled test statistic is created.
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
seed	optional argument to <a href="#">set.seed</a>
formula	a formula of the form $y \sim \text{group}$ where y is numeric and group is a factor variable.
data	a data frame with the variables in the formula.
subset	an optional expression specifying which observations to keep.

**Details**

Permutation test to see if a population parameter is the same for two populations. For instance, test *latex* where *latex* denotes the population mean. The values of the numeric variable are randomly assigned to the two groups and the difference of the statistic for each group is calculated. The command will print the mean and standard error of the distribution of the test statistic as well as a P-value.

Observations with missing values are removed.

**Value**

Returns invisibly a vector of the replicates of the test statistic.

**Methods (by class)**

- permTest(default): Permutation test
- permTest(formula): Permutation test

**Author(s)**

Laura Chihara

**References**

Tim Hesterberg's website: <https://www.timhesterberg.net/bootstrap-and-resampling>

**Examples**

```
permTest(states03$ViolentCrime, states03$DeathPenalty)

#using formula syntax
permTest(ViolentCrime ~ DeathPenalty, data = states03, alt = "less")
```

---

permTestAnova

*Permutation test for ANOVA F-test*

---

**Description**

Permutation test to see if the population mean is the same for two or more populations. For instance, test *latex* where *latex* denotes the population mean. The values of the numeric variable are randomly assigned to the groups and the ANOVA F statistic is calculated. The command will print the mean and standard error of the distribution of the test statistic as well as a P-value.

**Usage**

```
permTestAnova(x, ...)

## Default S3 method:
permTestAnova(
  x,
  group,
  B = 9999,
  plot.hist = TRUE,
  plot.qq = FALSE,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  seed = NULL,
  ...
)

## S3 method for class 'formula'
permTestAnova(formula, data = parent.frame(), subset, ...)
```

**Arguments**

x	a numeric vector.
...	further arguments to be passed to or from methods.
group	a factor variable with two or more levels. If group is a numeric vector, it will be coerced into a factor variable.
B	the number of resamples (positive integer greater than 2).
plot.hist	a logical value. If TRUE, the permutation distribution of the statistic is plotted.
plot.qq	a logical value. If TRUE, then a normal quantile-quantile plot of the resampled test statistic is created.
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
seed	optional argument to <a href="#">set.seed</a>
formula	a formula of the form $y \sim \text{group}$ where $y$ is numeric and group is a factor variable.
data	a data frame with the variables in the formula.
subset	an optional expression specifying which observations to keep.

**Details**

Observations with missing values are removed.

**Value**

Returns invisibly a vector of the replicates of the test statistic.

**Methods (by class)**

- permTestAnova(default): Permutation test for ANOVA F-test
- permTestAnova(formula): Permutation test for ANOVA F-test

**Author(s)**

Adam Loy, Laura Chihara

**References**

Tim Hesterberg's website: <https://www.timhesterberg.net/bootstrap-and-resampling>

**Examples**

```
permTestAnova(states03$ViolentCrime, states03$Region, B = 499)

#using formula syntax
## Not run:
permTestAnova(ViolentCrime ~ Region, data = states03, B = 9999)

## End(Not run)
```

---

permTestCor

*Permutation test for the correlation of two variables.*

---

**Description**

Hypothesis test for a correlation of two variables. The null hypothesis is that the population correlation is 0.

**Usage**

```
permTestCor(x, ...)

## Default S3 method:
permTestCor(
  x,
  y,
  B = 999,
  alternative = "two.sided",
  plot.hist = TRUE,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
  y.name = deparse(substitute(y)),
  xlab = NULL,
  ylab = NULL,
```

```

    title = NULL,
    seed = NULL,
    ...
)

## S3 method for class 'formula'
permTestCor(formula, data, subset, ...)

```

### Arguments

x	a numeric vector.
...	further arguments to be passed to or from methods.
y	a numeric vector.
B	the number of resamples to draw (positive integer greater than 2).
alternative	alternative hypothesis. Options are "two.sided", "less" or "greater".
plot.hist	a logical value. If TRUE, plot the distribution of the correlations obtained from each resample.
plot.qq	a logical value. If TRUE, plot the normal quantile-quantile plot of the correlations obtained from each resample.
x.name	Label for variable x
y.name	Label for variable y
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
seed	optional argument to <a href="#">set.seed</a>
formula	a formula $y \sim x$ where x, y are numeric vectors.
data	a data frame that contains the variables given in the formula.
subset	an optional expression indicating what observations to use.

### Details

Perform a permutation test to test  $\rho$ , where  $\rho$  is the population correlation. The rows of the second variable are permuted and the correlation is re-computed.

The mean and standard error of the permutation distribution is printed as well as a P-value.

Observations with missing values are removed.

### Value

Returns invisibly a vector of the correlations obtained by the randomization.

### Methods (by class)

- permTestCor(default): Permutation test for the correlation of two variables.
- permTestCor(formula): Permutation test for the correlation of two variables.

**Author(s)**

Laura Chihara

**References**

Tim Hesterberg's website: <https://www.timhesterberg.net/bootstrap-and-resampling>

**Examples**

```
plot(states03$HSGrad, states03$TeenBirths)
cor(states03$HSGrad, states03$TeenBirths)

permTestCor(states03$HSGrad, states03$TeenBirths)
permTestCor(TeenBirths ~ HSGrad, data = states03)
```

---

permTestPaired	<i>Permutation test for paired data.</i>
----------------	--

---

**Description**

Permutation test for paired data.

**Usage**

```
permTestPaired(x, ...)

## Default S3 method:
permTestPaired(
  x,
  y,
  B = 9999,
  alternative = "two.sided",
  plot.hist = TRUE,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
  y.name = deparse(substitute(y)),
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  seed = NULL,
  ...
)

## S3 method for class 'formula'
permTestPaired(formula, data, subset, ...)
```

**Arguments**

x	a numeric vector.
...	further arguments to be passed to or from methods.
y	a numeric vector.
B	the number of resamples.
alternative	the alternative hypothesis. Options are "two.sided", "less" and "greater".
plot.hist	a logical value. If TRUE, create a histogram displaying the permutation distribution of the statistic.
plot.qq	a logical value. If TRUE, include a quantile-normal plot of the permutation distribution.
x.name	Label for x variable
y.name	Label for y variable
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
seed	optional argument to <code>set.seed</code>
formula	a formula of the form $y \sim x$ , where $x$ , $y$ are both numeric variables.
data	an optional data frame containing the variables in the formula. By default the variables are taken from environment(formula).
subset	an optional vector specifying a subset of observations to be used.

**Details**

For two paired numeric variables with  $n$  rows, randomly select  $k$  of the  $n$  rows ( $k$  also is randm) and switch the entries *latex* and then compute the mean of the difference of the two variables ( $y-x$ ).

Observations with missing values are removed.

**Value**

Returns invisibly a vector of the replicates of the test statistic (ex. mean of the difference of the resampled variables).

**Methods (by class)**

- `permTestPaired(default)`: Permutation test for paired data.
- `permTestPaired(formula)`: Permutation test for paired data.

**Author(s)**

Laura Chihara

**References**

Tim Hesterberg's website: <https://www.timhesterberg.net/bootstrap-and-resampling>

**Examples**

```
#Does chocolate ice cream have more calories than vanilla ice cream, on average?
#H0: mean number of calories is the same
#HA: mean number of calories is greater in chocolate ice cream

permTestPaired(Icecream$VanillaCalories, Icecream$ChocCalories, alternative = "less")
permTestPaired(ChocCalories ~ VanillaCalories, data = Icecream, alternative = "greater")
```

---

permTestSlope	<i>Permutation test for the Slope</i>
---------------	---------------------------------------

---

**Description**

Hypothesis test for a slope of a simple linear regression model. The null hypothesis is that the population slope is 0.

**Usage**

```
permTestSlope(x, ...)

## Default S3 method:
permTestSlope(
  x,
  y,
  B = 999,
  alternative = "two.sided",
  plot.hist = TRUE,
  plot.qq = FALSE,
  x.name = deparse(substitute(x)),
  y.name = deparse(substitute(y)),
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  seed = NULL,
  ...
)

## S3 method for class 'formula'
permTestSlope(formula, data, subset, ...)
```

**Arguments**

x	a numeric vector.
...	further arguments to be passed to or from methods.
y	a numeric vector.

B	the number of resamples to draw (positive integer greater than 2).
alternative	alternative hypothesis. Options are "two.sided", "less" or "greater".
plot.hist	a logical value. If TRUE, plot the distribution of the slopes obtained from each resample.
plot.qq	a logical value. If TRUE, plot the normal quantile-quantile plot of the slopes obtained from each resample.
x.name	Label for variable x
y.name	Label for variable y
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
seed	optional argument to <code>set.seed</code>
formula	a formula $y \sim x$ where $x$ , $y$ are numeric vectors.
data	a data frame that contains the variables given in the formula.
subset	an optional expression indicating what observations to use.

### Details

Perform a permutation test to test  $latex$ , where  $latex$  is the population slope. The rows of the second variable are permuted and the slope is re-computed.

The mean and standard error of the permutation distribution is printed as well as a P-value.

Observations with missing values are removed.

### Value

Returns invisibly a vector of the slopes obtained by the randomization.

### Methods (by class)

- `permTestSlope(default)`: Permutation test for the slope
- `permTestSlope(formula)`: Permutation test for the slope

### Author(s)

Adam Loy, Laura Chihara

### References

Tim Hesterberg's website: <https://www.timhesterberg.net/bootstrap-and-resampling>

### Examples

```
plot(states03$HSGrad, states03$TeenBirths)
lm(HSGrad ~ TeenBirths, data = states03)

permTestSlope(states03$HSGrad, states03$TeenBirths)
permTestSlope(TeenBirths ~ HSGrad, data = states03)
```

---

```
plot.carlboot          Plot the bootstrap distribution in carlboot object
```

---

### Description

Plot the bootstrap distribution returned as a carlboot object.

### Usage

```
## S3 method for class 'carlboot'
plot(x, bins = 15, size = 5, xlab = NULL, ylab = NULL, title = NULL, ...)

## S3 method for class 'carlperm'
plot(x, bins = 15, size = 5, xlab = NULL, ylab = NULL, title = NULL, ...)
```

### Arguments

x	The carlboot object to print.
bins	number of bins in histogram.
size	size of points.
xlab	an optional character string for the x-axis label
ylab	an optional character string for the y-axis label
title	an optional character string giving the plot title
...	not used

### Examples

```
boot_dist <- boot(ToothGrowth$len, ToothGrowth$supp, B = 1000)
plot(boot_dist)

perm_dist <- permTest(states03$ViolentCrime, states03$DeathPenalty, B = 999)
plot(perm_dist)
```

---

```
print.carlboot        Print a summary of an carlboot object
```

---

### Description

Print summary statistics and confidence intervals for an carlboot object.

**Usage**

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

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

**Arguments**

x	The carlboot object to print.
...	not used

---

qqPlotDemo

*Demonstration of the normal qq-plot.*


---

**Description**

Demonstrate the normal quantile-quantile plot for samples drawn from different populations.

**Usage**

```
qqPlotDemo(
  n = 25,
  distribution = "normal",
  mu = 0,
  sigma = 1,
  df = 10,
  lambda = 10,
  numdf = 10,
  dendf = 16,
  shape1 = 40,
  shape2 = 5
)
```

**Arguments**

n	sample size
distribution	population distribution. Options are "normal", "t", "exponential", "chi.square", "F" or "beta" (partial matches are accepted).
mu	mean for the normal distribution.
sigma	(positive) standard deviation for the normal distribution.
df	(positive) degrees of freedom for the t-distribution.
lambda	positive rate for the exponential distribution.
numdf	(positive) numerator degrees of freedom for the chi-square distribution.
dendf	(positive) denominator degrees of freedom for the chi-square distribution.
shape1	positive parameter for the beta distribution (shape1 = a).
shape2	positive parameter for the beta distribution (shape2 = b).

**Details**

Draw a random sample from the chosen sample and display the normal qq-plot as well as the histogram of its distribution.

**Value**

Returns invisibly the random sample.

**Author(s)**

Laura Chihara

**Examples**

```
qqPlotDemo(n = 30, distr = "exponential", lambda = 1/3)
```

---

states03

*US government data, 2003*

---

**Description**

Census data on the 50 states from 2003.

**Format**

A data frame with 50 observations on the following 24 variables.

**State** the 50 states

**Region** a factor with levels Midwest, Northeast, South, West

**Pop** Population in 1000

**Births** Number of births

**Deaths** Number of deaths

**Pop18** Percent of population 18 years of age or younger

**Pop65** Percent of population 65 years of age or older

**HSGrad** Percent of population 25 years of age or older with a high school degree

**ColGrad** Percent of population 25 years of age or older with a college degree

**TeacherPay** Average teachers salary in dollars

**InfMortality** Infant mortality per 1000 live births

**TeenBirths** Live births per 1000 15-19 year old females

**ViolentCrime** Violent crime per 100000 population

**PropertyCrime** Property crime per 100000 population

**DeathPenalty** State has death penalty?  
**Executions** Number of executions 1977-2003  
**Poverty** Percent of populaton below the poverty level  
**Unemp** Percent unemployed (of population 16 years or older)  
**Uninsured** Percent uninsured (3 year aveage)  
**Income** Median household income in 1998 dollars  
**Earnings** Average hourly earnings of production workers in manufacturing  
**Heart** Deaths by heart disease per 100000 population  
**Vehicles** Deaths by motor vehicle accidents per 100000 population  
**Homeowners** Home ownership rate

### Source

United States Census Bureau <https://www.census.gov/>

---

stemPlot	<i>Stem and leaf plot</i>
----------	---------------------------

---

### Description

Stem and leaf plot. Will accept a factor variable as a second argument to create stem plots for each of the levels.

### Usage

```
stemPlot(x, ...)

## Default S3 method:
stemPlot(x, grpvar = NULL, varname = NULL, grpvarname = NULL, ...)

## S3 method for class 'formula'
stemPlot(formula, data = parent.frame(), subset, ...)
```

### Arguments

x	a numeric variable.
...	further arguments to be passed to or from methods.
grpvar	a factor variable. A stem plot of x will be created for each level of the factor variable.
varname	name of the numeric variable. This is for printing the output only. Change if you want to print out a name different from the actual variable name.
grpvarname	name of the factor variable. This is for printing the output only. Change if you want to print out a name different from the actual variable name.

formula	a formula of the form $x \sim \text{grpvar}$ where $x$ is numeric and $\text{grpvar}$ is a factor variable.
data	a data frame with the variables in the formula.
subset	an optional expression specifying which observations to keep.

### Details

This command is just an enhanced version of R's `stem` command. It allows the user to create the stem plot for a numeric variable grouped by the levels of a factor variable.

### Methods (by class)

- `stemPlot(default)`: Stem and leaf plot
- `stemPlot(formula)`: Stem and leaf plot

### Author(s)

Laura Chihara

### Examples

```
stemPlot(states03$Births, states03$Region)
stemPlot(Births ~ Region, data = states03)
```

---

summary.carlboot      *Print a summary of an carlboot object*

---

### Description

Print summary statistics and confidence intervals, if desired, for an `lmeresamp` object.

### Usage

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

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

### Arguments

object	The carlboot object to print.
...	not used

**Examples**

```
boot_dist <- boot(ToothGrowth$len, ToothGrowth$supp, B = 1000)
summary(boot_dist)
perm_dist <- permTest(states03$ViolentCrime, states03$DeathPenalty, B = 999)
summary(perm_dist)
```

# Index

- \* **ANOVA**
    - anovaSummarized, 2
  - \* **Correlation**
    - corDemo, 12
  - \* **Summarized**
    - anovaSummarized, 2
  - \* **bar**
    - groupedBar, 13
  - \* **bootstrap**
    - boot, 3
    - bootPaired, 7
  - \* **confidence**
    - confIntDemo, 11
  - \* **correlation**
    - permTestCor, 20
    - permTestSlope, 24
  - \* **datasets**
    - Icecream, 14
    - Milkshakes, 15
    - states03, 28
  - \* **data**
    - anovaSummarized, 2
    - permTestPaired, 22
  - \* **grouped**
    - groupedBar, 13
  - \* **interval**
    - confIntDemo, 11
  - \* **missing**
    - missingLevel, 16
  - \* **normal**
    - qqPlotDemo, 27
  - \* **paired**
    - permTestPaired, 22
  - \* **permutation**
    - permTest, 16
    - permTestAnova, 18
    - permTestCor, 20
    - permTestPaired, 22
    - permTestSlope, 24
  - \* **plot**
    - groupedBar, 13
    - qqPlotDemo, 27
    - stemPlot, 29
  - \* **quantile-quantile**
    - qqPlotDemo, 27
  - \* **randomization**
    - boot, 3
    - bootPaired, 7
    - permTest, 16
    - permTestAnova, 18
    - permTestCor, 20
    - permTestPaired, 22
    - permTestSlope, 24
  - \* **randomization**
    - bootCor, 5
    - bootSlope, 9
  - \* **resampling**
    - boot, 3
    - bootCor, 5
    - bootPaired, 7
    - bootSlope, 9
    - permTest, 16
    - permTestAnova, 18
    - permTestCor, 20
    - permTestPaired, 22
    - permTestSlope, 24
  - \* **stem**
    - stemPlot, 29
  - \* **test**
    - permTest, 16
    - permTestAnova, 18
    - permTestCor, 20
    - permTestPaired, 22
    - permTestSlope, 24
  - \* **values**
    - missingLevel, 16
- anovaSummarized, 2

boot, 3  
bootCor, 5  
bootPaired, 7  
bootSlope, 9  
  
confint.carlboot, 11  
confIntDemo, 11  
corDemo, 12  
  
groupedBar, 13  
  
Icecream, 14  
  
Milkshakes, 15  
missingLevel, 16  
  
permTest, 16  
permTestAnova, 18  
permTestCor, 20  
permTestPaired, 22  
permTestSlope, 24  
plot.carlboot, 26  
plot.carlperm(plot.carlboot), 26  
print.carlboot, 26  
print.carlperm(print.carlboot), 26  
  
qqPlotDemo, 27  
  
set.seed, 4, 6, 8, 10, 17, 19, 21, 23, 25  
states03, 28  
stemPlot, 29  
summary.carlboot, 30  
summary.carlperm(summary.carlboot), 30