Package 'RcmdrPlugin.TeachStat'

December 21, 2018

Type Package

Title RCommander Plugin for Teaching Statistical Methods

Version 1.0.10

Description RCommander plugin for teaching statistical methods.
It adds a new menu for making easier the teaching of the main concepts about the main statistical methods.

License GPL (>= 2)

Imports Rcmdr (>= 2.3-0), Hmisc, tcltk2, randtests, tseries

Suggests tkrplot

LazyData TRUE

Encoding latin1

NeedsCompilation no

Author Tomás R. Cotos Yañez [aut] (<https://orcid.org/0000-0002-7732-6565>),
Manuel A. Mosquera Rodríguez [aut, cre]
(<https://orcid.org/0000-0002-4769-6119>),
Ana Pérez González [aut] (<https://orcid.org/0000-0003-4706-7125>),
Benigno Reguengo Lareo [aut]

Maintainer Manuel A. Mosquera Rodríguez <mamrguez@uvigo.es>

Repository CRAN

Date/Publication 2018-12-21 14:00:17 UTC

R topics documented:

`RcmdrPlugin.TeachStat-package`

RCommander plugin for teaching statistical methods.

Description

It adds a new menu for making easier the teaching of the main concepts about the main statistical methods.

Details

Package:	RcmdrPlugin.TeachStat
Type:	Package
Version:	1.0.10-0
Date:	2018-12-11
License:	GPL version 2 or newer

Author(s)

Tomás R. Cotos Yañez <cotos@uvigo.es>
Manuel A. Mosquera Rodríguez <mamrguez@uvigo.es>
Ana Pérez González <anapg@uvigo.es>
Benigno Reguengo Lareo <benireguengo@gmail.com>

See Also

`Rcmdr`.

Agrupadas *Grouped or tabulated data set*

Description

Grouped or tabulated data set, given by lower and upper limits and frequency. It is used as an example for the use of the *Numerical Summaries - Tabulated data* window of the RcmdrPlugin.TeachStat package

Usage

```
data("Agrupadas")
```

Format

Data frame with 4 cases (rows) and 3 variables (columns).

`Linf` Numeric value, the lower limit of the tabulated data.

`Lsup` Numeric value, the upper limit of the tabulated data.

`ni` Numeric value, the frecuency of the tabulated data.

Examples

```
data(Agrupadas)
calcularResumenDatosTabulados(l_inf=Agrupadas$Linf, l_sup=Agrupadas$Lsup,
  ni=Agrupadas$ni, statistics =c("mean", "sd", "IQR", "quantiles"), quantiles
  = c(0,0.25,0.5,0.75,1), tablaFrecuencia=FALSE)
```

calcularResumenDatosTabulados *Summary statistics for tabulated data*

Description

`calcularResumenDatosTabulados` performs the main statistical summary for tabulated data (mean, standard deviation, coefficient of variation, skewness, kurtosis, quantile and mode) are calculated. Also it allows to obtain the frequency table (with classmark, amplitude and density).

Usage

```
calcularResumenDatosTabulados(l_inf, l_sup, ni,
                              statistics = c("mean", "sd", "se(mean)", "IQR",
                              "quantiles", "cv", "skewness", "kurtosis"),
                              quantiles = c(0, 0.25, 0.5, 0.75, 1),
                              tablaFrecuencia = FALSE)
```

Arguments

l_inf	numeric vector with the lower limit of each interval.
l_sup	numeric vector with the upper limit of each interval.
ni	numeric vector with the frequency of occurrence of values in the range between the lower limit and upper limit [l_inf [i-1], l_sup [i]).
statistics	any of "mean", "sd", "se(mean)", "quantiles", "cv" (coefficient of variation - sd/mean), "skewness", "kurtosis" or "mode"; defaulting to c("mean", "sd", "quantiles", "IQR"
quantiles	quantiles to report; by default is c(0, 0.25, 0.5, 0.75, 1).
tablaFrecuencia	logical value indicating whether or not to display the frequency table, by default is FALSE.

Details

calcularResumenDatosTabulados performs an analysis of **tabulated data** (frequently used in statistics when the number of distinct values is large or when dealing with continuous quantitative variables), represented by a table of statistics (arithmetic mean, standard deviation, interquartile range, coefficient of variation, asymmetry, kurtosis, and quantile).

It also allows to show the frequency table of the tabulated variable by selecting tablaFrecuencia=TRUE. The class mark, amplitude and density are added to the frequency table.

The LOWER LIMIT or L[i-1] and UPPER LIMIT or L[i] vectors, represent the data of continuous quantitative variables in class intervals of the form [L[i-1], L[i]) where i = 1, .. ., k.

Value

calcularResumenDatosTabulados() returns a list of two elements:

.numsummary	an object of class "numSummary" containing the numerical summary of the tabulated variable.
.table	a matrix containing the values of the frequency table.

See Also

cut

Examples

```
data(cars93)
cortes <- seq(from=1500, to=4250, by=250)
aa <- cut( cars93$Weight, breaks=cortes, dig.lab=4)
ni <- table(aa)
l_inf <- cortes[-length(cortes)]
l_sup <- cortes[-1]
agrup <- data.frame(l_inf,l_sup,ni)
head(agrup)

calcularResumenDatosTabulados(agrup$l_inf, agrup$l_sup, agrup$Freq)
```

```
calcularResumenDatosTabulados(agrup$l_inf, agrup$l_sup, agrup$Freq, tabla=TRUE)

bb <- calcularResumenDatosTabulados(agrup$l_inf, agrup$l_sup, agrup$Freq,
                                    statistics=c("mean","mode") )
bb
str(bb)
class(bb$.summary)
class(bb$.table)
```

`calcularResumenVariablesDiscretas` *Summary statistics for discrete variables*

Description

`calcularResumenVariablesDiscretas` gives the main statistical summary for discrete variables (mean, standard deviation, coefficient of variation, skewness, kurtosis and quantiles). Also builds the frequency table (with classmark, amplitude and density).

Usage

```
calcularResumenVariablesDiscretas(data,
                                  statistics = c("mean", "sd", "se(mean)", "IQR",
                                    "quantiles", "cv", "skewness", "kurtosis"),
                                quantiles = c(0, 0.25, 0.5, 0.75, 1), groups = NULL,
                                     tablaFrecuencia = FALSE, cortes=NULL)
```

Arguments

`data`	`data.frame` with the discrete variables.
`statistics`	any of `"mean"`, `"sd"`, `"se(mean)"`, `"quantiles"`, `"cv"` (coefficient of variation - sd/mean), `"skewness"` or `"kurtosis"`; defaulting to `c("mean", "sd", "quantiles", "IQR")`.
`quantiles`	quantiles to report; by default is `c(0, 0.25, 0.5, 0.75, 1)`.
`groups`	optional variable, typically a factor, to be used to partition the data. By deafault is `NULL`.
`tablaFrecuencia`	logical value indicating whether or not to display the frequency table, by default is `FALSE`.
`cortes`	either a numeric vector of two or more unique cut points or a single number (greater than or equal to 2) giving the number of intervals into which data is to be cut, by default is `NULL`.

Details

calcularResumenVariablesDiscretas performs a descriptive analysis of discrete variables (quantitative variables that take as a finite or infinite numerable distinct values), generating a table of statistics (arithmetic mean, standard deviation, interquartile range, coefficient of variation, skewness, kurtosis, and quantiles) optionally allowing the partition of the data by a factor variable (groups).

It also allows to show the frequency table of selected discrete variables by selecting tablaFrecuencia=TRUE. Moreover it also allows to divide the range of the variables into intervals given by the argument cortes (breaks). See more info in cut.

Value

calcularResumenVariablesDiscretas returns a list of two elements:

.numsummary	an object of class "numSummary" containing the numerical summary of the discrete variables.
.table	a matrix containing the values of the frequency table.

See Also

cut

Examples

```
## Not run:
data(cars93)
calcularResumenVariablesDiscretas(data=cars93["Cylinders"],group=NULL)
calcularResumenVariablesDiscretas(data=cars93["Cylinders"],group=cars93$Airbags)
bb <- calcularResumenVariablesDiscretas(data=cars93["Cylinders"],group=cars93$Airbags,
                                        tablaFrecuencia=TRUE)
str(bb)
bb
bb$.summary
class(bb$.summary)

calcularResumenVariablesDiscretas(data=cars93["Horsepower"], tablaFrecuencia=TRUE)
calcularResumenVariablesDiscretas(data=cars93["Horsepower"], tablaFrecuencia=TRUE, cortes=5)
calcularResumenVariablesDiscretas(data=cars93["Horsepower"], tablaFrecuencia=TRUE,
                                  cortes=c(50,100,200,250,300))
calcularResumenVariablesDiscretas(data=cars93["Horsepower"], groups=cars93$Airbags,
                                  tablaFrecuencia=TRUE, cortes=5)

## End(Not run)
```

`calcular_frecuencia` *Frequency distributions for qualitative variables*

Description

Performs the frequency distribution for qualitative variables, nominals and/or ordinals. For ordinal variables requested quantile is calculated.

Usage

```
calcular_frecuencia(df.nominal, ordenado.frec = FALSE,df.ordinal,
                    cuantil.p = 0.5,iprint = TRUE, ...)
```

Arguments

`df.nominal`	`data.frame` with factor type components (including character type) that are interpreted as nominal variables.
`ordenado.frec`	table ordered frequencies depending on their frequency (only used for nominal variables).
`df.ordinal`	`data.frame` with factor type components (including character type) that are interpreted as ordinal variables.
`cuantil.p`	requested quantile value (only used for ordinal variables).
`iprint`	logical value indicating whether or not to display the frequency table.
`...`	further arguments to be passed to or from methods.

Value

`calcular_frecuencia` returns a list of three elements:

`.nominal`	a matrix containing the table of frequency distribution for nominal variables (`ni` = absolute frequencies and `fi` = relative frequencies).
`.ordinal`	a matrix containing the table of frequency distribution for ordinal variables (`ni` = absolute frequencies, `fi` = relative frequencies, `Ni` = absolute cumulative frequency, `Fi` = cumulative absolute frequencies).
`df.cuantil`	data frame containing the quantiles.

See Also

`table` , `cumsum`

Examples

```
data(cars93)
aa <- calcular_frecuencia(df.nominal=cars93["Type"], ordenado.frec=TRUE, df.ordinal=NULL,
                          cuantil.p=0.5, iprint = TRUE)
calcular_frecuencia(df.nominal=NULL, ordenado.frec=TRUE, df.ordinal=cars93["Airbags"],
                    cuantil.p=0.25, iprint = TRUE)
bb <- calcular_frecuencia(df.nominal=cars93["Type"], ordenado.frec=TRUE,
                          df.ordinal=cars93["Airbags"], cuantil.p=0.25, iprint = FALSE)
str(bb)
bb
```

---

cars93 *Data from 93 Cars on Sale in the USA in 1993*

Description

The cars93 data frame has 93 rows and 26 columns.

Usage

```
cars93
```

Format

This data frame contains the following columns:

`Manufacturer` Manufacturer.

`Model` Model.

`Type` Type: a factor with levels `"Small"`, `"Sporty"`, `"Compact"`, `"Midsize"`, `"Large"` and `"Van"`.

`MinPrice` Minimum Price (in \$1,000): price for a basic version.

`MidPrice` Midrange Price (in \$1,000): average of `Min.Price` and `Max.Price`.

`MaxPrice` Maximum Price (in \$1,000): price for "a premium version".

`CityMPG` City MPG (miles per US gallon by EPA rating).

`HighwayMPG` Highway MPG.

`Airbags` Air Bags standard. Factor: none, driver only, or driver & passenger.

`DriveTrain` Drive train type: rear wheel, front wheel or 4WD; (factor).

`Cylinders` Number of cylinders (missing for Mazda RX-7, which has a rotary engine).

`EngineSize` Engine size (litres).

`Horsepower` Horsepower (maximum).

`RPM` RPM (revs per minute at maximum horsepower).

`EngineRevol` Engine revolutions per mile (in highest gear).

`Manual` Is a manual transmission version available? (yes or no, Factor).

`FuelCapacity` Fuel tank capacity (US gallons).

`Passengers` Passenger capacity (persons)

`Length` Length (inches).

`Wheelbase` Wheelbase (inches).

`Width` Width (inches).

`UTurnSpace` U-turn space (feet).

`RearSeatRoom` Rear seat room (inches) (missing for 2-seater vehicles).

`LuggageCapacity` Luggage capacity (cubic feet) (missing for vans).

`Weight` Weight (pounds).

`USA` Of non-USA or USA company origins? (factor).

Details

Cars were selected at random from among 1993 passenger car models that were listed in both the *Consumer Reports* issue and the *PACE Buying Guide*. Pickup trucks and Sport/Utility vehicles were eliminated due to incomplete information in the *Consumer Reports* source. Duplicate models (e.g., Dodge Shadow and Plymouth Sundance) were listed at most once.

Further description can be found in Lock (1993).

Source

Lock, R. H. (1993) 1993 New Car Data. *Journal of Statistics Education* **1**(1). https://doi.org/10.1080/10691898.1993.11910459.

References

Venables, W. N. and Ripley, B. D. (1999) *Modern Applied Statistics with S-PLUS.* Third Edition. Springer.

`ConvertVariables` *Modify Variable Types*

Description

In this graphical interface, the user can modify the variable type into nominal (factor), ordinal (ordered factor) or numeric type.

`Cprop.test` *Test for proportions of one or two samples*

Description

Performs hypothesis testing and confidence interval for a proportion or difference of two proportions. The values of the samples necessary to perform the function are the number of successes and the number of trails.

Usage

```
Cprop.test(ex, nx, ey = NULL, ny = NULL, p.null = 0.5,
           alternative = c("two.sided", "less", "greater"), conf.level = 0.95,
           ...)
```

Arguments

`ex`	numeric value that represents the number of successes of the first sample (see Details).
`nx`	numerical value representing the total number of trails of the first sample.
`ey`	(optional) numerical value representing the number of success of the second sample (see Details).
`ny`	(optional) numerical value representing the total number of trails of the second sample.
`p.null`	numeric value that represents the value of the population proportion or the difference between the two population proportions, depending on whether there are one or two samples (see Details).
`alternative`	a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.
`conf.level`	confidence level of the interval.
`...`	further arguments to be passed to or from methods.

Details

So that the contrast can be made must be fulfilled that at least 1 hit. That is, in the case of a sample `ex` must be greater than or equal to 1 and in the case of two samples, `ex` or `ey` must be greater than or equal to 1.

Furthermore, for the case of a sample value p.null must be strictly positive.

Value

A list with class "htest" containing the following components:

`statistic`	the value of the test statistic.

parameter	number of trails and value of the population proportion or the difference in population proportions.
p.value	the p-value for the test.
conf.int	a confidence interval for the proportion or for the difference in proportions, appropriate to the specified alternative hypothesis.
estimate	a value with the sample proportions.
null.value	the value of the null hypothesis.
alternative	a character string describing the alternative.
method	a character string indicating the method used, and whether Yates' continuity correction was applied.
data.name	a character string giving the names of the data.

See Also

prop.test

Examples

```
## Proportion for a sample
Cprop.test(1,6) # 1 success in 6 attempts

#### With a data set: proportion of cars not manufactured in US
data(cars93) #data set provided with the package
exitos<-sum(cars93$USA == "nonUS")
total<-length(cars93$USA)
Cprop.test(ex=exitos, nx=total)

## Difference of proportions
Cprop.test(1,6,3,15)
  # Sample 1: 1 success in 6 attempts
  # Sample 2: 3 success in 15 attempts

#### With a data set: difference of proportions of cars not manufactured in US
#### between manual and automatic
exitosx<-sum(cars93$USA == "nonUS" & cars93$Manual == "Yes" )
totalx<-sum(cars93$Manual == "Yes")
exitosy<-sum(cars93$USA == "nonUS" & cars93$Manual == "No" )
totaly<-sum(cars93$Manual == "No")
Cprop.test(ex=exitosx, nx=totalx,ey=exitosy, ny=totaly)
```

DMKV.test *Z-test for the difference of means of two independent Normal variables with known population variances.*

Description

Under the assumption that the data come from two independent Normal distributions, it performs the hypothesis test and the confidence interval for the difference of means with known population variances.

Usage

```
DMKV.test(x, y, difmu = 0, sdx, sdy,
          alternative = c("two.sided", "less", "greater"), conf.level = 0.95,
          ...)
```

Arguments

x	numerical vector (non-empty) that contains the data of the first sample.
y	numerical vector (non-empty) containing the data of the second sample.
difmu	numeric value indicating the value of the difference in population means between the two samples.
sdx	numerical value indicating the population standard deviation of the first sample, which is assumed to be known (mandatory).
sdy	numeric value indicating the population standard deviation of the second sample, which is assumed to be known (mandatory).
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
conf.level	confidence level of the interval.
...	further arguments to be passed to or from methods.

Value

A list with class "htest" containing the following components:

statistic	the value of the test statistic.
parameter	sample lenghts and population standard deviations.
p.value	the p-value for the test.
conf.int	intervalo de confianza para la diferencia de medias con varianzas poblacionales conocidas asociado a la hipótesis alternativa especificada.
estimate	the estimated difference in means.
null.value	the specified hypothesized value of the mean difference.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating what type of statistical method was performed.
data.name	a character string giving the name(s) of the data.

See Also

t.test

Examples

```
data(cars93) # Data set provided with the package
# Maximum price difference (MaxPrice) in means between cars manufactured in the
# US and those manufactured outside, assuming that the variances are known and
# equal to 64 and 169, respectively
var1<-subset(cars93, USA=="nonUS", select=MaxPrice)
var2<-subset(cars93, USA=="US", select=MaxPrice)
DMKV.test(var1, var2, sdx=13, sdy=8, difmu=0,
alternative="greater", conf.level=0.95)
```

`intervaloConfianzaMedia`

Confidence interval or hypothesis testing for the mean of a Normal variable

---

Description

In this graphical interface, the data selection is made to perform the calculation of the confidence interval or the hypothesis testing for the mean of a Normal variable.

This interface will call the statistical functions `MKV.test` and `t.test`, depending, respectively, on whether the population variance is known or not.

`intervaloConfianzaMediasIndependientes`

Confidence interval or hypothesis testing for the difference in means of two independent Normal variables

---

Description

In this graphical interface, the data selection is made to perform the calculation of the confidence interval or the hypothesis testing for the difference in means of two independent Normal variables.

This interface will call the statistical functions `DMKV.test` and `t.test`, dependiendo, respectivamente, depending, respectively, on whether the population variances are known or not.

`intervaloConfianzaVarianza`

Confidence interval or hypothesis testing for the Variance

Description

In this graphical interface, the data selection is made to perform the calculation of the confidence interval or the hypothesis testing for the variance of a Normal variable.

This interface will call the statistical functions `VKM.test` and `VUM.test`, depending, respectively, on whether the population mean is known or not.

listTypesVariables *List of variables and types of a Data Frame*

Description

listTypesVariables returns a vector with the names and types of the variables of a data frame.

Usage

```
listTypesVariables(dataSet)
```

Arguments

dataSet	the quoted name of a data frame in memory.

Value

A character vector

See Also

names

Examples

```
require(datasets)
listTypesVariables("iris")
```

MKV.test *Z-test for the mean of a Normal variable with known population variance.*

Description

Under the assumption that the data come from a Normal distribution, it makes the hypothesis testing and the confidence interval for the mean with known population variance.

Usage

```
MKV.test(x, mu = 0, sd, alternative = c("two.sided", "less", "greater"),
         conf.level = 0.95, ...)
```

Arguments

x	a (non-empty) numeric vector of data values.
mu	a number indicating the true value of the mean - Null hypothesis.
sd	numerical value indicating the population standard deviation assumed to be known (mandatory).
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
conf.level	confidence level of the interval.
...	further arguments to be passed to or from methods.

Value

A list with class "htest" l containing the following components:

statistic	the value of the test statistic.
parameter	sample length, population standard deviation and sample standard deviation.
p.value	the p-value for the test.
conf.int	a confidence interval for the mean appropriate to the specified alternative hypothesis.
estimate	the estimated mean.
null.value	the specified hypothesized value of the mean.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating what type of statistical test was performed.
data.name	a character string giving the name of the data.

See Also

t.test

Examples

```
data(cars93) # Dataset provided with the package
# Mean maximum price (MaxPrice) less than 20 thousand $ assuming that the
# variance is known and equal to 11
MKV.test(cars93$MaxPrice, sd=11, alternative="less", mu=20, conf.level=0.95)
```

randomnessMenu *Randomness test*

Description

In the "Nonparametric Tests" menu, two new entries are provided to perform the randomness test.

The first "Randomness test for two level factor..." can be used to contrast the randomness of a factor with two levels. This option use the function `runs.test` from `tseries` package. For more information see `runs.test`.

The second entry in the menu "Randomness test for numeric variable..." is used to test the randomness of a numerical variable. This option use the function `runs.test` from `randtest` package. For more information see `runs.test`.

Details

Here is an example of "Randomness test for a two level factor..." menu entry.

Load data "AMSsurvey" selecting from Rcmdr menu: "Data" -> "Data in packages" -> "Read data set from an attached package..." then double-click on "car", click on "AMSsurvey" and on "OK". Rcmdr reply with the following command in source pane (R Script)

```
data(AMSsurvey, package="car")
```

To make randomness test on variable "sex", select from Rcmdr menu: "Statistics" -> "Nonparametric tests" -> "Randomness test for two level factor..." select "sex" and "OK". Rcmdr reply with the following command in source pane (R Script)

```
with(AMSsurvey, twolevelfactor.runs.test(sex))
```

Here is an example of "Randomness test for a numeric variable..." menu entry.

Load data "sweetpotato" selecting from Rcmdr menu: "Data" -> "Data in packages" -> "Read data set from an attached package..." then double-click on "randtests", click on "sweetpotato" and on "OK". Rcmdr reply with the following command in source pane (R Script)

```
data(sweetpotato, package="randtests")
```

```
sweetpotato <- as.data.frame(sweetpotato)
```

To make randomness test on variable "yield", select from Rcmdr menu: "Statistics" -> "Nonparametric tests" -> "Randomness test for numeric variable..." select "yield" and "OK". Rcmdr reply with the following command in source pane (R Script)

```
with(sweetpotato, numeric.runs.test(yield))
```

Author(s)

Manuel Munoz-Marquez <manuel.munoz@uca.es>

See Also

For more information see `Rcmdr-package`.

twoOrMoreLevelFactors *RcmdrPlugin.TeachStat Utility Functions*

Description

`twoOrMoreLevelFactorsP()` returns TRUE if there is at least one factor in the active dataset that has two or more levels.

`twoOrMoreLevelFactors()` returns the object name of those factors that are active in the dataset that have at least two levels.

VKM.test *Chi-square test for the variance of a Normal variable with known population mean.*

Description

Under the assumption that the data come from a Normal distribution, it performs the hypothesis testing and the confidence interval for the variance with known population mean.

Usage

```
VKM.test(x, sigma = 1, sigmasq = sigma^2, mu,
         alternative = c("two.sided", "less", "greater"), conf.level = 0.95,
         ...)
```

Arguments

`x`	a (non-empty) numeric vector of data values.
`sigma`	a number indicating the true value of the population standar deviation - Null hypothesis.
`sigmasq`	control argument.
`mu`	numerical value indicating the population mean assumed to be known (mandatory).
`alternative`	a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.
`conf.level`	confidence level of the interval.
`...`	further arguments to be passed to or from methods.

Value

A list with class "htest" containing the following components:

statistic	the value of the ctest statistic.
parameter	the degrees of freedom for the test statistic.
p.value	the p-value for the test.
conf.int	confidence interval for variance with known population mean associated with the specified alternative hypothesis.
estimate	the estimated variance.
null.value	the specified hypothesized value of the variance.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating what type of statistical test was performed.
data.name	a character string giving the name of the data.

See Also

VUM.test , var.test

Examples

```
data(cars93) # Dataset provided with the package
# Variance of the maximum price (MaxPrice) assuming that the population mean
# price is known and equal to 22
VKM.test(cars93$MaxPrice, alternative="two.sided", sigma=11, mu=22, conf.level=0.95)
```

VUM.test *Chi-square test for the variance of a Normal variable with unknown population mean.*

Description

Under the assumption that the data come from a Normal distribution, it performs the hypothesis testing and the confidence interval for the variance with unknown population mean.

Usage

```
VUM.test(x, sigma = 1, sigmasq = sigma^2,
         alternative = c("two.sided", "less", "greater"), conf.level = 0.95,
         ...)
```

Arguments

x	a (non-empty) numeric vector of data values.
sigma	a number indicating the true value of the population standar deviation - Null hypothesis.
sigmasq	control argument.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided", "greater" o "less". You can specify just the initial letter.
conf.level	confidence level of the interval.
...	further arguments to be passed to or from methods.

Value

A list with class "htest" containing the following components:

statistic	the value of the test statistic
parameter	the degrees of freedom for the test statistic
p.value	the p-value for the test.
conf.int	confidence interval for variance with unknown population mean associated with the specified alternative hypothesis.
estimate	vthe estimated variance.
null.value	the specified hypothesized value of the variance.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating what type of statistical test was performed.
data.name	a character string giving the name of the data.

See Also

VKM.test , var.test

Examples

```
data(cars93) # Dataset provided with the package
# Variance of the maximum price (MaxPrice) assuming that the population mean
# price is unknown
VUM.test(cars93$MaxPrice, alternative="two.sided", sigma=11, conf.level=0.95)
```

W.numSummary *Summary statistics for weighted variables*

Description

W.numSummary gives the main statistical summary for weighted variables (mean, standard deviation, coefficient of variation, skewness, kurtosis and quantiles). It also allows the partition of the data by a factor variable.

Usage

```
W.numSummary(data,
             statistics = c("mean", "sd", "se(mean)", "IQR",
            "quantiles", "cv", "skewness", "kurtosis"),type = c("2", "1", "3"),
             quantiles = c(0, 0.25, 0.5, 0.75, 1),groups = NULL, weights)
```

Arguments

data	data.frame with the variables.
statistics	any of "mean", "sd", "se(mean)", "quantiles", "cv" (coefficient of variation - sd/mean), "skewness" or "kurtosis"; defaulting to c("mean", "sd", "quantiles", "IQR").
type	definition to use in computing skewness and kurtosis; see the skewness and kurtosis functions in the **e1071** package. The default is "2".
quantiles	quantiles to report; by default is c(0, 0.25, 0.5, 0.75, 1).
groups	optional variable, typically a factor, to be used to partition the data. By deafault is NULL.
weights	numeric vector of weights. Zero values are allowed.

Details

W.numSummary performs a descriptive analisys of quantitative variables weighted (or not) by a numeric variable which determines the importance of each subject in the data frame. Optionally it allows the partition of the data by a factor variable (groups).

Note that, unlike the numSummary function, the sample standard deviation is calculated instead of the sample standard quasideviation.

Value

An object with class "numSummary".

See Also

numSummary, skewness, kurtosis.

Examples

```
data(cars93)

# no weighted
W.numSummary(data=cars93[,c("CityMPG")], statistics =c("mean", "sd", "IQR", "quantiles"),
   quantiles = c(0,0.25,0.5,0.75,1), weights=NULL, groups=NULL)
# weighted
W.numSummary(data=cars93[,c("CityMPG")], statistics =c("mean", "sd", "IQR", "quantiles"),
   quantiles = c(0,0.25,0.5,0.75,1), weights=cars93$FuelCapacity, groups=NULL)
# no weighted
W.numSummary(data=cars93[,c("CityMPG")], statistics =c("mean", "sd", "IQR", "quantiles"),
   quantiles = c(0,0.25,0.5,0.75,1), weights=NULL, groups=cars93$Manual)
# weighted
bb <- W.numSummary(data=cars93[,c("CityMPG")], statistics =c("mean", "sd", "IQR", "quantiles"),
   quantiles = c(0,0.25,0.5,0.75,1), weights=cars93$FuelCapacity, groups=cars93$Manual)

bb
str(bb)
class(bb)
```

Index