Package ‘StatEngine’

January 11, 2021

Type Package

Title An R Package for the UofSC Course STAT 509 ``Statistics for Engineers''

Version 1.0.0

Author Dewei Wang

Maintainer Dewei Wang <deweiwang@stat.sc.edu>

Description This R package provides tools for methods covered in the course STAT 509 ``Statistics for Engineers'' at the University of South Carolina

Depends R (>= 3.1.0),
car (>= 3.0.0)

License MIT

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

R topics documented:

`Continuous Distributions` *Continuous distributions*

Description

Plot pdf/cdf, calculate mean/variance/standard deviation, and compute probabilities for various continuous distributions

Usage

```
# continuous Uniform Distribution
uniform.summary(a,b,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
uniform.prob(a,b,lb,ub)
uniform.quantile(a,b,q)

# Normal distribution
normal.summary(mu,sigma,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
normal.prob(mu,sigma,lb,ub)
normal.quantile(mu,sigma,q)

# Exponential distribution
exponential.summary(lambda,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
exponential.prob(lambda,lb,ub)
exponential.quantile(lambda,q)

# Gamma distribution
gamma.summary(r,lambda,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
gamma.prob(r,lambda,lb,ub)
gamma.quantile(r,lambda,p)

# Weibull distribution
weibull.summary(beta,delta,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
weibull.prob(beta,delta,lb,ub)
weibull.quantile(beta,delta,p)

# Lognormal distribution
lognormal.summary(theta,omega,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
lognormal.prob(theta,omega,lb,ub)
lognormal.quantile(theta,omega,p)

#Beta distribution
beta.summary(alpha,beta,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
beta.prob(alpha,beta,lb,ub)
beta.quantile(alpha,beta,p)
```

Arguments

`plotpdf`	TRUE or FALSE, if TRUE, it plots the pmf
`plotcdf`	TRUE or FALSE, if TRUE, it polts the cdf
`lb,ub`	lower bound (lb) and upper bound (ub) in a probability statement; lb could be -Inf; ub could be Inf; ub cannot be less than lb
`a,b`	lower bound and upper bound of the support of a continuous uniform distribution
`mu,sigma`	mean and standard deviation of a normal distribution
`lambda`	parameter of an exponential distribution
`r,lambda`	shape and scale parameters of a gamma distribution
`beta,delta`	shape and scale parameters of a Weibull distribution
`theta,omega`	shape and scale parameters of a lognormal distribution
`alpba,beta`	parameters of a beta distribution

Details

Plot the probability density function (pdf) and the cumulative distribution function (cdf) and calculate probability, mean, variable and standard deviation, and compute probabilities of various discrete distributions (continuous uniform distribution, normal distribution, exponential distribution, gamma distribution, Weibull distribution, lognormal distribution, beta distribution).

Value

`uniform.summary`
: a list of the mean, variance, and standard deviation of a continuous uniform distribution, plot of the pdf/cdf or not

`uniform.prob`
: probability of X between lb and ub based on a continuous uniform distribution

`uniform.quantile`
: quantile of a continuous uniform distribution

`normal.summary`
: a list of the mean, variance, and standard deviation of a normal distribution, plot of the pdf/cdf or not

`normal.prob`
: probability of X between lb and ub based on a normal distribution

`normal.quantile`
: quantile of a normal distribution

`exponential.summary`
: a list of the mean, variance, and standard deviation of an exponential distribution, plot of the pdf/cdf or not

`exponential.prob`
: probability of X between lb and ub based on an exponential distribution

`exponential.quantile`
: quantile of an exponential distribution

`gamma.summary`
: a list of the mean, variance, and standard deviation of a gamma distribution, plot of the pdf/cdf or not

`gamma.prob`
: probability of X between lb and ub based on a gamma distribution

`gamma.quantile`
: quantile of a gamma distribution

`weibull.summary`
: a list of the mean, variance, and standard deviation of a Weibull distribution, plot of the pdf/cdf or not

`weibull.prob`
: probability of X between lb and ub based on a Weibull distribution

`weibull.quantile`
: quantile of a Weibull distribution

`lognormal.summary`
: a list of the mean, variance, and standard deviation of a lognormal distribution, plot of the pdf/cdf or not

`lognormal.prob`
: probability of X between lb and ub based on a lognormal distribution

`lognormal.quantile`
: quantile of a lognormal distribution

`beta.summary`
: a list of the mean, variance, and standard deviation of a beta distribution, plot of the pdf/cdf or not

`beta.prob`
: probability of X between lb and ub based on a beta distribution

`beta.quantile`
: quantile of a beta distribution

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapter 4 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
# Continuous uniform distribution
a=4.9;b=5.1;uniform.summary(a,b)
uniform.prob(a,b,4.95,5) # P(4.95<X<5)
uniform.quantile(a,b,0.9) # x such that P(X>x)=0.1

#Normal distribution
normal.summary(10,2) #mu=10,sigma=2,variance=4
normal.prob(10,2,9,11)
normal.quantile(10,2,0.98)

#Exponential distribution
exponential.summary(25) #lambda=25
exponential.prob(25,0.1,Inf) #P(X>0.1)
exponential.quantile(25,0.1) # x such that P(X>x)=0.9

#Gamma distribution
gamma.summary(10,.5) #r=10,lambda=0.5
gamma.prob(10,0.5,25,Inf) #P(X>25)
gamma.quantile(10,0.5,0.95) # x such that P(X<x)=0.95

#Weibull distribution
weibull.summary(2,5000) #beta=2,delta=5000
weibull.prob(2,5000,6000,Inf) #P(X>6000)
weibull.quantile(2,5000,1-0.05) # x such that P(X>x)=0.05

#Lognormal distribution
lognormal.summary(10,1.5) #theta=10,omega=1.5
lognormal.prob(10,1.5,10000,Inf) #P(X>10000)
lognormal.quantile(10,1.5,1-0.99) # x such that P(X>x)=0.99

#Beta distribution
beta.summary(2.5,1) #alpha=2.5, beta=1
beta.prob(2.5,1,0.7,Inf) # P(X>0.7)
beta.quantile(2.5,1,0.99) # x such that P(X<x)=0.99
```

`Counting Techniques` *Sample Space Characterization*

Description

Use permutations and combinations to count outcomes

Usage

```
nPr(n,r): permute r items from a set of n distinct items

nCr(n,r): select r items from a set of n items where order does not matter

SimPerm(n_vec): permute n items when they are not totally distinct
```

Arguments

`n`	total n itmes
`r`	select r
`n_vec`	a vector of (n_1, n_2,...,n_r)

Details

Permutation, Combination, and Permutation of similar items

Value

nPr(n,r) provides the number of different ways to permute r items from a set of n distinct items.

nCr(n,r) provides the number of different ways to select r items from a set of n items where order does not matter.

SimPerm(n_vec) provides the number of different ways to permute n items when they are not totally distinct.

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Section 2.2 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
nPr(8,4)
nCr(3,2)
nCr(47,4)
SimPerm(c(2,3,2))
```

`Descriptive Statistics` *Descriptive Statistics*

Description

Calculate and plot descriptive statistics

Usage

```
data.summary(x)
```

Arguments

`x`	the sample
`plot`	True of False; plot or not the stem and leaf diagram, scatterplot, histogram, boxplot, QQ-plot

Details

Calculate various numerical descpritive statistics and plot the stem and leaf diagram, scatterplot, histogram, boxplot, QQ-plot

Value

`data.summary`	a table of numerical descpritive statistics (std stands for standard deviation), a stem and leaf diagram, and a figure consisting of scatterplot, histogram, boxplot, QQ-plot of the sample

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapter 6 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
x=scan("https://raw.githubusercontent.com/Harrindy/StatEngine/master/Data/CompressiveStrength.csv")
data.summary(x)
```

Discrete Distributions

Discrete distributions

Description

Plot pmf/cdf, calculate mean/variance/standard deviation, and compute probabilities for various discrete distributions

Usage

```
# For use-defined discrete distribution
discrete.plotpdf(x,fx)
discrete.plotcdf(x,fx)
discrete.summary(x,fx,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE"))
discrete.prob(x,fx,lb)
discrete.prob(x,fx,lb,ub,inclusive=c("none","left","right","both"))

# Discrete Uniform Distribution
duniform.summary(range,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
duniform.prob(range,lb)
duniform.prob(range,lb,ub,inclusive=c("none","left","right","both"))

# Binomial distribution
binomial.summary(n,p,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
binomial.prob(n,p,lb)
binomial.prob(n,p,lb,ub,inclusive=c("none","left","right","both"))

# Geometric distribution
geometric.summary(p,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
geometric.prob(p,lb)
geometric.prob(p,lb,ub,inclusive=c("none","left","right","both"))

# Negative Binomial distribution
negbinom.summary(r,p,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
negbinom.prob(r,p,lb)
negbinom.prob(r,p,lb,ub,inclusive=c("none","left","right","both"))

# Hypergeometric distribution
hypergeo.summary(N,K,n,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
hypergeo.prob(N,K,n,lb)
hypergeo.prob(N,K,n,lb,ub,inclusive=c("none","left","right","both"))

# Poisson distribution
poisson.summary(lambda,L,plotpdf=c("TRUE","FALSE"), plotcdf=c("TRUE","FALSE")))
poisson.prob(lambda,L,lb)
poisson.prob(lambda,L,lb,ub,inclusive=c("none","left","right","both"))
```

Arguments

x	possible values of a user defined discrete random variable

`fx`	probabilities of X=x, the order of entries in fx must matches the order of entries in x.
`plotpdf`	TRUE or FALSE, if TRUE, it plots the pmf
`plotcdf`	TRUE or FALSE, if TRUE, it polts the cdf
`lb,ub`	lower bound (lb) and upper bound (ub) in a probability statement; lb could be -Inf; ub could be Inf; ub cannot be less than lb
`inclusive`	"none": lb<X<ub; "left": lb<=X<ub; "right": lb<X<=ub; "both": lb<=X<=ub
`Range`	contains all possible values of a discrete uniform random variable
`p`	parameter p of a Binomial distribution/Geometric distribution/Negtive Binomial distribution
`n`	parameter n of a Binomial distribution
`r`	parameter r of a negative Binomial distribution
`N,K,n`	parameters N, K, and n of a hypergeometric distribution
`lambda,L`	parameters lambda and L of a Poisson distribution. Default: L=1

Details

Plot the probability mass function (pmf) and the cumulative distribution function (cdf) and calculate probability, mean, variable and standard deviation, and compute probabilities of various discrete distributions (a self-defined discrete distribution, discrete uniform distribution, binomial distribution, geometric distribution, negative binomial distribution, hypergeometric distribution, and Poisson distribution).

Value

`discrete.plotpdf`	a figure of the pmf of the user-defined discrete distribution
`discrete.plotcdf`	a figure of the cdf of the user-defined discrete distribution
`discrete.summary`	a list of the mean, variance, and standard deviation of the user-defined discrete distribution, plot of the pmf/cdf or not
`discrete.prob`	probability of X between lb and ub based on the user-defined discrete distribution
`duniform.summary`	a list of the mean, variance, and standard deviation of a discrete uniform distribution, plot of the pmf/cdf or not
`duniform.prob`	probability of X between lb and ub based on a uniform distribution
`binomial.summary`	a list of the mean, variance, and standard deviation of a binomial distribution, plot of the pmf/cdf or not
`binomial.prob`	probability of X between lb and ub based on a binomial distribution
`geometric.summary`	a list of the mean, variance, and standard deviation of a geometric distribution, plot of the pmf/cdf or not
`geometric.prob`	probability of X between lb and ub based on a geometric distribution
`negbinom.summary`	a list of the mean, variance, and standard deviation of a negative binomial distribution, plot of the pmf/cdf or not

`negbinom.prob` probability of X between lb and ub based on a negative binomial distribution

`hypergeo.summary`
: a list of the mean, variance, and standard deviation of a hypergeometric distribution, plot of the pmf/cdf or not

`hypergeo.prob` probability of X between lb and ub based on a hypergeometric distribution

`poisson.summary`
: a list of the mean, variance, and standard deviation of a Poisson distribution, plot of the pmf/cdf or not

`poisson.prob` probability of X between lb and ub based on a Poisson distribution

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapter 3 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
x=c(0,1,2,3,4);fx=c(0.6561,0.2916,0.0486,0.0036,0.0001);

discrete.summary(x,fx)
discrete.summary(x,fx,plotpdf=FALSE,plotcdf=TRUE)
discrete.summary(x,fx,plotpdf=TRUE,plotcdf=FALSE)

discrete.prob(x,fx,2) #P(X=2)

discrete.prob(x,fx,2,4,inclusive="none") #P(2<X<4)
discrete.prob(x,fx,2,4,inclusive="left") #P(2<=X<4)
discrete.prob(x,fx,2,4,inclusive="right") #P(2<X<=4)
discrete.prob(x,fx,2,4,inclusive="both") #P(2<=X<=4)

discrete.prob(x,fx,2,Inf,inclusive="none") #P(2<X)
discrete.prob(x,fx,2,Inf,inclusive="left") #P(2<=X)
discrete.prob(x,fx,2,Inf,inclusive="right") #P(2<X)
discrete.prob(x,fx,2,Inf,inclusive="both") #P(2<=X)

discrete.prob(x,fx,-Inf,4,inclusive="none") #P(X<4)
discrete.prob(x,fx,-Inf,4,inclusive="left") #P(X<4)
discrete.prob(x,fx,-Inf,4,inclusive="right") #P(X<=4)
discrete.prob(x,fx,-Inf,4,inclusive="both") #P(X<=4)

#X~Discrete Uniform(1,2,4,5,8,10)
range=c(1,2,4,5,8,10)
duniform.summary(range)
duniform.prob(range,2) #P(X=2)
duniform.prob(range,2,4,inclusive="left") #P(2<=X<4)

#X~Binomial(n=5,p=0.3)
binomial.summary(5,0.3)
```

```
binomial.prob(5,0.3,2) #P(X=2)
binomial.prob(5,0.3,2,4,inclusive="left") #P(2<=X<4)

#X~Geometric(p=0.3)
geometric.summary(0.3)
geometric.prob(0.3,2) #P(X=2)
geometric.prob(0.3,2,4,inclusive="left") #P(2<=X<4)

#X~Negative Binomial(r=3,p=0.3)
negbinom.summary(3,0.3)
negbinom.prob(3,0.3,5) #P(X=5)
negbinom.prob(3,0.3,5,7,inclusive="left") #P(5<=X<7)

#X~Hypergeometric(N=300,K=100,n=4)
hypergeo.summary(300,100,4)
hypergeo.prob(300,100,4,2) #P(X=2)
hypergeo.prob(300,100,4,2,4,inclusive="left") #P(2<=X<4)

#X~Poisson(lambda=2.3,L=5)
poisson.summary(2.3,5)
poisson.prob(2.3,5,10) #P(X=10)
poisson.prob(2.3,5,1,Inf,inclusive="left") #P(1<=X)
```

`Linear Regression` *Linear Regression*

Description

Inferences for linear regression: confidence intervals and hypthesis testing on regression coefficients, confidence interval on the mean response at a given x and prediction interval on a future observation at a given x, Partial F-test, model adequacy checking.

Usage

```
# fit must be an lm object; e.g., fit=lm(y~x).
lm.est(fit)

# based on the fit, compute CIs and tests on betas
lm.coef.CI(fit,level=0.95)
# The default value of hypo.beta is zero
lm.coef.test(fit,alpha=0.05,H1="two",hypo.beta=?)

#Partial F-test,
#fit.H0 is the lm object using the null model
#fit.ALL is the lm object using the full model
lm.partialFtest(fit.H0=lmH0,fit.ALL=lmALL,alpha=0.05)

# Model Adequacy checking
lm.modelcheck(fit)
# fit must be an lm object. VIFs are also printed.
```

Arguments

`fit`	an lm object
`level`	the confidence level
`alpha`	the significance level
`hypo.beta`	the hypothesized beta value that about to be tested
`fit.H0, fit.ALL`	the null model and full model in the partial F-test

Details

Inferences for linear regression: confidence intervals and hypthesis testing on regression coefficients, confidence interval on the mean response at a given x and prediction interval on a future observation at a given x, Partial F-test, model adequacy checking.

Value

`lm.est`	the least squares estimates
`interval`	As long as the function has "interval", the outcome are confidence intervals.
`test`	As long as the function has "test", it conduct the hypothesis testing.
`lm.modelcheck`	residual analysis and VIFs

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapters 11-12 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
Example1=read.csv("https://raw.githubusercontent.com/Harrindy/StatEngine/master/Data/HydrocarbonPurity.csv")
x=Example1$HydrocarbonLevels
y=Example1$Purity
fit=lm(y~x)
lm.est(fit)
summary(fit)
lm.coef.interval(fit,level=0.95)
predict.lm(fit,new=data.frame(x=1),interval="confidence")
predict.lm(fit,new=data.frame(x=1),interval="prediction")


Example2=read.csv("https://raw.githubusercontent.com/Harrindy/StatEngine/master/Data/WireBond.csv")
head(Example2,2)
y=Example2$PullStrength
x1=Example2$WireLength
x2=Example2$DieHeight
fit=lm(y~x1+x2)
summary(fit)
```

```
lm.coef.interval(fit,level=0.95)
lm.coef.test(fit,alpha=0.05,H1="two")
predict.lm(fit,new=data.frame(x1=8,x2=275),interval="confidence")
predict.lm(fit,new=data.frame(x1=8,x2=275),interval="prediction")
data.summary(fit$residuals)
vif(fit)
lm.modelcheck(fit)

x3=x1^2
x4=x2^2
lmH0=lm(y~x1+x2)
lmALL=lm(y~x1+x2+x3+x4)
lm.partialFtest(fit.H0=lmH0,fit.ALL=lmALL,alpha=0.05)
summary(lm(y~x1+x2+x3))
summary(lm(y~x1+x2+x4))
summary(lm(y~x1+x2+x3+x4))

Example3=read.csv("https://raw.githubusercontent.com/Harrindy/StatEngine/master/Data/AirplaneSidewallPanels
head(Example3,2)
y=Example3$cost
x=Example3$lotsize
plot(x,y)
fit=lm(y~x)
lines(x,fit$fitted.values)
x2=x^2
fit.quad=lm(y~x+x2)
lines(x,fit.quad$fitted.values,col="blue")
summary(fit)
summary(fit.quad)


Example4=read.csv("https://raw.githubusercontent.com/Harrindy/StatEngine/master/Data/SurfaceFinishData.csv"
head(Example4,2)
plot(Example4)
y=Example4$SurfaceFinish
x1=Example4$RPM
x2=Example4$TypeofCuttingTool
fit=lm(y~x1+x2)
summary(fit)
par(mar=c(4,4,.1,.1))
par(mfrow=c(1,2))
plot(x1[x2==0],y[x2==0])
lines(x1[x2==0],fit$fitted.values[x2==0])
plot(x1[x2==1],y[x2==1])
lines(x1[x2==1],fit$fitted.values[x2==1])

x3=x1*x2
fit2=lm(y~x1+x2+x3)
summary(fit2)
par(mfrow=c(1,1))
plot(x1,y)
lines(x1[x2==0],fit$fitted.values[x2==0])
lines(x1[x2==1],fit$fitted.values[x2==1])
lines(x1[x2==0],fit2$fitted.values[x2==0],col="blue")
lines(x1[x2==1],fit2$fitted.values[x2==1],col="blue")
```

One-Sample Confidence Intervals
One-Sample Tests

Description

Conduct hypothesis testing on population mean, population variance, and population proportion. Compute power of a test, and calculate the required sample size for a desired power.

Usage

```
#Test on pupulation mean of a normal distribution when the population variance is known:
Ztest(mu0,H1,alpha,sigma,sample) # if sample available
Ztest(mu0,h1,alpha,sigma,n,barx) # if statistics are provided
# Power calculation
Ztest.power(H1,alpha,sigma,n,delta)
# Sample size calculation
sample.size.Ztest(H1,sigma,alpha,beta,delta)

#Test on pupulation mean of a normal distribution when the population variance is unknown:
Ttest(mu0=?,H1=?,alpha=?,sample=?) #If data are available
Ttest(mu0=?,H1=?,alpha=?,n=?,barx=?,s=?) #If statistics are provided s=sd(data)
# Power calculation
Ttest.power(H1=?,est.sigma=sn,alpha=?,n=?,delta=?)
# Sample size calculation
sample.size.Ttest(H1=?,est.sigma=sn, beta=?,delta=?,alpha=?

#One-Sample Proportion Z-tests on a population proportion:
Proptest(p0=?,H1=?,alpha=?,n=?,X=?)
# Power calculation
Proptest.power(H1=?,alpha=?,p0=?,p1=?,n=?)
# Sample size calculation
sample.size.Proptest(H1=?,beta=?,alpha=?,p0=?,p1=?)
```

Arguments

`mu0`	the hypothesized value of the population mean in Z-tests and T-tests
`H1`	type of alternative hypothesis: "two", "left", or "right"
`alpha`	the significance level
`sample`	a vector of the observed sample
`sigma`	the known population standard deviation
`n`	the sample size
`barx`	the observed sample mean
`delta`	the delta value in power/sample size calculation of Z-test/T-test
`beta`	the desired power for sample size calculation
`s`	the observed sample standard deviation
`est.sigma`	an estimate of the population standard deviation in power/sample size calculation of T-test

`sigma0`	the hypothesized value of the population standard deviation in Chi-square-tests
`lambda`	the lambda value in power/sample size calculation of Chi-square-tests
`p0`	the hypothesized value of the population proportion
`X`	number of observations belongs to a class of interest
`p1`	the p1 value in power/sample size calculation of One-sample Prportion Z-tests

Details

Conduct hypothesis testing on population mean, population variance, and population proportion. Compute power of a test, and calculate the required sample size for a desired power.

Value

`test`	As long as the function has "test", it produces the test results of using three approaches.
`power`	As long as the function has "test", it computes the power of the test.
`sample.size`	As long as the function has "sample.size", the outcome is the minimum sample size required to reach the given power.

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapter 9 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
#Ztest
#must include the = sign
Ztest(mu0=3,H1="two",alpha=0.05,sigma=0.9,n=15,barx=2.78)
Ztest.power(H1="two",alpha=0.05,sigma=0.9,n=15,delta=3.25-3)
sample.size.Ztest(H1="two",sigma=0.9,alpha=0.05,beta=1-0.9,delta=3.75-3)

#Ttest
#must include the = sign
x=c(131.15, 130.69, 130.91, 129.54, 129.64, 128.77, 130.72, 128.33,
128.24, 129.65, 130.14, 129.29, 128.71, 129.00, 129.39, 130.42,
129.53, 130.12, 129.78, 130.92)
data.summary(x)
Ttest(mu0=130,H1="two",alpha=0.05,sample=x)
Ttest.power(H1="two",est.sigma=sd(x),alpha=0.05, n=length(x),delta=0.5)
sample.size.Ttest(H1="two",est.sigma =sd(x),beta=0.25,delta=0.1,alpha=0.05)

#Chi-square test
#must include the = sign
Chi2test(sigma0=sqrt(0.01),H1="right",alpha=0.05,n=20,s=sqrt(0.0153))
Chi2test.power(H1="right",alpha=0.05,n=20,lambda=1.25)
sample.size.Chi2test(H1="right",beta=0.2,lambda=1.25,alpha=0.05)
```

```
Chi2test(sigma0=0.01,H1="right",alpha=0.01,n=15,s=0.008)
Chi2test.power(H1="right",alpha=0.01,n=15,lambda=1.5)
sample.size.Chi2test(H1="right",beta=0.2,lambda=1.25,alpha=0.01)

#One-sample proportion Z-test
#must include the = sign
Proptest(p0=0.05, H1="left",alpha=0.05,n=300,X=7)
Proptest.power(H1="left",alpha=0.05,p0=0.05,p1=0.03,n=300)
sample.size.Proptest(H1="left",beta=0.1,alpha=0.05,p0=0.05,p1=0.03)
```

One-Sample Tests *One-Sample Confidence Intervals*

Description

Compute confidence intervals on the population mean, population variance, and population proportion. In addition, it computes prediction interval for a single future observation from a normal distribution.

Usage

```
#CI for pupulation mean of a normal distribution when the population variance is known:
Zinterval(level,sigma,sample) # if sample available
Zinterval(level,sigma,n,barx) # if stats are provided
# Choice of sample size for estimating the population mean when error is specified
sample.size.Zinterval(level,sigma,E)

#CI for pupulation mean when the population variance is unknown and the distribution is normal
#or when the sample size is smaller than 25:
Tinterval(level,sample) # if sample available
Tinterval(level,n,barx,s) # if stats are provided

#Large-sample CI for pupulation mean:
AZinterval(level,sample) # if sample available
AZinterval(level,n,barx,s) # if stats are provided

#CI for pupulation variance (or standard deviation) of a normal distribution:
Chi2interval(level,sample) # if sample available
Chi2interval(level,n,s) # if stats are provided

#Large-sample CI for a pupulation proportion:
Propinterval(level,n,X)
# Choice of sample size for estimating a population proportion when error is specified
sample.size.Propinterval(level,ini.p,E) # using an intial guess
sample.size.Propinterval(level,ini.p=0.5,E) # using the conservative apporach

# Prediction interval of a single future observation form a normal distribution:
Predinterval(level,sample) # if sample available
Predinterval(level,n,barx,s) # if stats are provided
```

Arguments

`level`	the confidence level
`sample`	a vector of the observed sample
`sigma`	the known population standard deviation
`s`	the observed sample standard deviation
`barx`	the observed sample mean
`n`	the sample size
`X`	number of observations belongs to a class of interest
`E`	specified error in sample size calculation
`ini.p`	A initial estimate of the populatin proportion. Default is 0.5 which corresponds to the conservative approach
`df`	the degrees of freedom of a t or chi.square distribution
`q`	a quantile value

Details

Compute CIs for the population mean, population variance, and population proportaion and PI for a single future observation from a normal distribution.

Value

`interval`	As long as the function has "interval", the outcome contains a two-sided CI (or PI) and the two one-sided confidence bounds.
`sample.size`	As long as the function has "sample.size", the outcome is the minimum sample size required to control the error to be no larger than E.
`t.quantile`	quantile of a t distribution
`Chi2.quantile`	quantile of a chi.square distribution

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapter 8 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
#Zinterval
#must include the = sign
x=c(64.1, 64.7, 64.5, 64.6, 64.5, 64.3, 64.6, 64.8, 64.2, 64.3)
Zinterval(level=0.95,sigma=1,sample=x)
Zinterval(level=0.99,sigma=1,sample=x)
sample.size.Zinterval(E=0.5,sigma=1,level=0.95)
# Using stats, must include the = sign
Zinterval(level=0.95,sigma=2,n=9,barx=98)
```

```
#Tinterval
#must include the = sign
Tinterval(level=0.95,n=10,barx=1000,s=20)
Tinterval(level=0.95,n=25,barx=1000,s=20)
Tinterval(level=0.99,n=10,barx=1000,s=20)
Tinterval(level=0.99,n=25,barx=1000,s=20)

#Large-sample Zinterval
#must include the = sign
x=scan("https://raw.githubusercontent.com/Harrindy/StatEngine/master/Data/Mercury.csv")
AZinterval(level=0.95,sample=x)

#Chi.square interval for variance/standard deviation
#must include the = sign
Chi2interval(level=0.95,n=20,s=0.01532)

#CIs for a porpulation proportion
#must include the = sign
Propinterval(level=0.95,n=85,X=10)
sample.size.Propinterval(level=0.95,ini.p=0.12,E=0.05)
sample.size.Propinterval(level=0.95,ini.p=0.5,E=0.05)

#Prediction interval for normal distribution
#must include the = sign
x=c(19.8, 10.1, 14.9, 7.5, 15.4, 15.4, 15.4, 18.5, 7.9, 12.7, 11.9, 11.4, 11.4,
14.1, 17.6, 16.7, 15.8, 19.5, 8.8, 13.6, 11.9, 11.4)
Tinterval(level=0.95,sample=x)
Predinterval(level=0.95,sample=x)
```

`Two-Sample Inferences` *Two-Sample Inferences (Confidence Intervals and Tests)*

Description

Compute confidence intervals and conduct hypothesis testing on difference between two population means, population variances, and population proportions.

Usage

```
#CI for the difference in two pupulation means of a normal distribution
#when the population variances are known:
# if sample available
twosample.Zinterval(level,sigma1,sigma2,sample1,sample2)
# if stats are provided
twosample.Zinterval(level,sigma1,sigma2,barx1,barx2,n1,n2)

#Test on the difference in two pupulation means of a normal distribution
#when the population variances are known:
# if sample available
twosample.Ztest(Delta0,H1,alpha,sigma1,sigma2,sample1,sample2)
# if stats are provided
twosample.Ztest(Delta0,H1,alpha,sigma1,sigma2,barx1,barx2,n1,n2)
```

```
#CI for the difference in two pupulation means of a normal distribution
#when the population variances are unknown (pooled=yes or no)
# if sample available
twosample.Tinterval(level, pooled,sample1,sample2)
# if stats are provided
twosample.Tinterval(level, pooled,barx1,barx2,n1,n2,s1,s2)

#Test on the difference in two pupulation means of a normal distribution
#when the population variances are unknown:
# if sample available
twosample.Ttest(Delta0,H1,alpha,pooled=yes,sample1,sample2)
# if stats are provided
twosample.Ttest(Delta0,H1,alpha,pooled=yes,barx1,barx2,n1,n2,s1,s2)

#CI for the ratio between two pupulation variances of a normal distribution
# if sample available
Finterval(level, sample1,sample2)
# if stats are provided
Finterval(level, n1,n2,s1,s2)

#Test on the ratio between two pupulation variances of a normal distribution
# if sample available
Ftest(H1,alpha,sample1,sample2)
# if stats are provided
Ftest(H1,alpha,n1,n2,s1,s2)
```

Arguments

`level`	the confidence level
`sample1`	a vector of the observed sample from the first population
`sample2`	a vector of the observed sample from the second population
`sigma1`	the known population standard deviation of the first population
`sigma2`	the known population standard deviation of the second population
`s1,s2`	the sample standard deviations
`barx1,barx2`	the sample means
`n1,n2`	the sample sizes
`X1,X2`	number of observations belongs to a class of interest
`Delta0`	the hypothesized value of mu1-mu2
`H1`	type of alternative: "two","left", or "right"
`alpha`	the significance level
`pooled`	"yes" or "no"

Details

Compute confidence intervals and conduct hypothesis testing on difference between two population means, population variances, and population proportions.

Value

`interval`	As long as the function has "interval", the outcome contains a two-sided CI and the two one-sided confidence bounds.
`test`	As long as the function has "test", it produces the test results of using three approaches.

Note

deweiwang@stat.sc.edu

Author(s)

Dewei Wang

References

Chapter 10 of the textbook "Applied Statistics and Probability for Engineers" 7th edition

Examples

```
#two-sample Zinterval
#must include the = sign
twosample.Zinterval(level=0.9,sigma1=1,sigma2=1.5,barx1=87.6,barx2=74.5,n1=10,n2=12)

#two-sample Ztest
twosample.Ztest(Delta0=0,H1="right",alpha=0.05, sigma1=8,sigma2=8,
                barx1=121,barx2=112,n1=10,n2=10)

#two-sample Tinterval
#must include the = sign
twosample.Tinterval(level=0.95,pooled="no",s1=5,s2=4,barx1=90,barx2=87,n1=10,n2=15)

#two-sample Ttest
catalyst1=c(91.50,94.18,92.18,95.39,91.79,89.07,94.72,89.21)
catalyst2=c(89.19,90.95,90.46,93.21,97.19,97.04,91.07,92.75)
data.summary(catalyst1)
data.summary(catalyst2)
twosample.Tinterval(level=0.95,pooled="yes",
                    sample1=catalyst1,sample2=catalyst2)
twosample.Ttest(Delta0=0,H1="two",alpha=0.05, pooled="yes",
                sample1=catalyst1,sample2=catalyst2)


C50=c(0.047, 0.060, 0.061, 0.064, 0.080, 0.090, 0.118, 0.165, 0.183)
C60=c(0.062, 0.105, 0.118, 0.137, 0.153, 0.197, 0.210, 0.250, 0.335)
data.summary(C50)
data.summary(C60)
twosample.Tinterval(level=0.95,pooled="no",
                    sample1=C50,sample2=C60)
twosample.Ttest(Delta0=0,H1="left",alpha=0.05, pooled="no",
                sample1=C50,sample2=C60)


#Paired T-test
Karlsrube=c(1.186,1.151,1.322,1.229,1.200,1.402,1.365,1.537,1.559)
```

```
   Lehigh=c(1.061,0.992,1.063,1.062,1.065,1.178,1.037,1.086,1.052)
   data.summary(Karlsrube-Lehigh)
Tinterval(level=0.95,sample=Karlsrube-Lehigh)
Ttest(mu0=0, H1="two",alpha=0.05,sample=Karlsrube-Lehigh)

#F-interval
Finterval(level=0.9,n1=11,n2=16,s1=5.1,s2=4.7)
Ftest(H1="two",alpha=0.1,n1=11,n2=16,s1=5.1,s2=4.7)

#two-sample proportion Z-interval
twosample.Propinterval(level=0.95, n1=85,n2=85,X1=10,X2=8)
twosample.Propinterval(level=0.95, n1=100,n2=100,X1=27,X2=19)
twosample.Proptest(H1="right",alpha=0.05,n1=100,n2=100,X1=27,X2=19)
```

Index