Package 'Omisc'

January 20, 2025

Type Package

Title Univariate Bootstrapping and Other Things

Version 0.1.5

Author Patrick O'Keefe

Maintainer Patrick O'Keefe <okeefep@ohsu.edu>

Description Primarily devoted to implementing the Univariate Bootstrap (as well as the Traditional Bootstrap). In addition there are multiple functions for DeFries-Fulker behavioral genetics models. The univariate bootstrapping functions, DeFries-Fulker functions, regression and traditional bootstrapping functions form the original core. Additional features may come online later, however this software is a work in progress. For more information about univariate bootstrapping see: Lee and Rodgers (1998) and Beasley et al (2007) <doi:10.1037/1082-989X.12.4.414>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

Imports MASS, base, stats, psych, copula

Suggests lavaan

NeedsCompilation no

Repository CRAN

Date/Publication 2022-08-09 14:10:02 UTC

Contents

`aboot` *Title*

Description

Title

Usage

```
aboot(boot)
```

Arguments

boot	a vector of bootstrap resample statistics to use to calculate the accelleration parameter.

Value

a vector of accelleration parameters for use in BCa bootstrap intervals

Examples

```
 data<-DFSimulated()
boots<-NaiveBoot(data, groups="Rs", keepgroups=TRUE)
boots<-bootAnalysis(boots, cbind, DFanalysis, 1,2,3, robust=FALSE)
boots<-t(boots)
aboot(boots)
```

---

aCalc *aCalc*

Description

This function calculates the actual "a" estimate from the jackknife approximation of a used in BCa CI's

Usage

```
aCalc(X)
```

Arguments

X	A vector of jackknife results

Value

An estimate of a for use in BCa.

Examples

```
X<-rchisq(100,2)
aCalc(X)
```

add *add*

Description

add

Usage

```
add(x)
```

Arguments

x	a list to be summed. Useful for doing elementwise summation of a list of matrices.

Value

returns a single summed object (e.g., a matrix)

Examples

```
x<-list(matrix(c(1:4),nrow=2),matrix(c(1:4),nrow=2))
add(x)
```

ajack *ajack*

Description

ajack

Usage

```
ajack(data, FUN, ...)
```

Arguments

data	data to get the bias parameter (a) for
FUN	a function to be applied to the data
...	additional arguments passed to FUN

Value

a vector of accelleration parameters for use in BCa bootstrap intervals

Examples

```
data<-DFSimulated()
ajack(data,DFanalysis, betasonly=TRUE, robust=FALSE)
```

`AllBootResults` *AllBootResults*

Description

AllBootResults

Usage

```
AllBootResults(boot, lower = 0.025, upper = 0.975, data, FUN, ...)
```

Arguments

`boot`	A matrix of bootstrap results
`lower`	the lower alpha
`upper`	the upper alpha
`data`	the data used for analysis
`FUN`	the function used for analysis
`...`	additional arguments to pass to FUN

Value

a matrix of results. Includes the baseline results, all output from standardBootIntervals, all results from BCa for both the jackknife and bootstrap accelleration methods. The bootstrap accelleration method is experimental.

Examples

```
data<-DFSimulated()
boots<-NaiveBoot(data, groups="Rs", keepgroups=TRUE)
boots<-bootAnalysis(boots, cbind, DFanalysis, 1,2,3, robust=FALSE)
AllBootResults(boots, .025,.975, data, DFanalysis, 1,2,3, robust=FALSE)
```

BarebonesBetas *BarebonesBetas*

Description

Gives just the beta weights from a linear model.

Usage

```
BarebonesBetas(data, Y = NULL, RHS = NULL)
```

Arguments

data	Data to be analyzed. Dependent variable MUST BE THE FIRST VARIABLE.
Y	optional. The dependent variable
RHS	option. The right hand side of the model, in R's model formulation (i.e., ~ X1+X2+etc)

Value

A vector of beta coefficients

Examples

```
Data<-TestData()
BarebonesBetas(Data)
```

BCa *BCa*

Description

BCa

Usage

```
BCa(
  boot,
  data,
  alphalower = 0.025,
  alphaupper = 0.975,
  accelleration = "jack",
  FUN,
  ...
)
```

Arguments

boot	A vector of bootstrap estimates of Theta
data	The data that was analyzed via the bootstrap
alphalower	The lower alpha for CI creation
alphaupper	The upper alpha for CI creation
accelleration	can currently take two values, "jack" and "bootstrap". "jack" returns the jackknife estimate of the accelleration parameter. "boot" is an experimental function that uses the bootstrap estimates in the calculation of the accelleration parameter. "boot" is many times faster (approximately n times faster where n is the number of observations).
FUN	The function used to get estimates of Theta
...	Additional arguments to FUN

Value

A matrix of BCa bootstrap CI's, the bias parameter and the accellation parameter

Examples

```
data<-DFSimulated()
boot<-NaiveBoot(data, groups="Rs", keepgroups=TRUE)
boot<-bootAnalysis(boot, cbind, DFanalysis, 1,2,3, robust=FALSE)
BCa(boot, data, .025,.975, accelleration="bootstrap", DFanalysis, 1,2,3, robust=FALSE)
```

---

bias *Title*

Description

Title

Usage

```
bias(boot, theta)
```

Arguments

boot	A vector of bootstrap estimates of theta
theta	the sample estimate of theta

Value

z0 the bias parameter for BCa CI

Examples

```
X<-data.frame(rnorm(1000))
theta<-mean(X)
boot<-NaiveBoot(X)
boot<-lapply(boot, mean)
boot<-do.call(rbind, boot)
bias(boot, theta)
```

---

bootAnalysis *bootAnalysis*

Description

bootAnalysis

Usage

```
bootAnalysis(boot, collapse = cbind, FUN, ...)
```

Arguments

boot	A list of bootstrap resamples from NaiveBoot or uniboot.
collapse	Should the results be collapsed from list form. Can take values of NULL, cbind or rbind
FUN	The function to apply to the bootstrap resamples
...	additional arguments to be passed to FUN

Value

A list or matrix of results

Examples

```
data<-DFSimulated()
data<-doubleEnter(data[,1],data[,2],data[,3])
boots<-uniboot(data, 1000, "Rs", TRUE, .5, NULL)
results<-bootAnalysis(boots, cbind, FUN=DFanalysis, 1,2,3,TRUE,FALSE,FALSE,TRUE,FALSE)
```

bootsample *bootsample*

Description

bootsample

Usage

```
bootsample(data, size = 1)
```

Arguments

data	a dataset to be bootstrapped
size	the size of the bootstrap sample relative to the original sample

Value

a dataset

Examples

```
X<-TestData()
Y<-bootsample(X)
```

cent *cent*

Description

cent

Usage

```
cent(X)
```

Arguments

X	vector to be centered

Value

Returns a centered vector

Examples

```
X<-c(1:10)
cent(X)
```

centerData *centerData*

Description

centerData

Usage

```
centerData(data)
```

Arguments

data	The data to be centered

Value

The centered data

Examples

```
X<-data.frame(X=c(1:4),Y=c(6:9))
centerData(X)
```

cholcors *cholcors*

Description

cholcors

Usage

```
cholcors(X, use = "everything")
```

Arguments

X	A matrix of data.
use	the missing data type to use for the correlation. Default is R's default "everything".

Value

This function returns the cholesky decomposition of the correlation matrix of the data

Examples

```
X<-stats::rnorm(100)
Y<-stats::rnorm(100)+X
Z<-cbind(X,Y)
cholcors(Z)
```

---

cholcovs *cholcovs*

Description

cholcovs

Usage

```
cholcovs(X, use = "everything")
```

Arguments

X	A matrix of data.
use	the missing data type to use for the correlation. Default is R's default "everything".

Value

This function returns the cholesky decomposition of the correlation matrix of the data

Examples

```
X<-stats::rnorm(100)
Y<-stats::rnorm(100)+X
Z<-cbind(X,Y)
cholcovs(Z)
```

DFanalysis *DFanalysis*

Description

There are three possible models to be fit. The default is the Rodgers and Kohler formulation of the DF model (Rodgers & Kohler, 2005). The non-default (if RK=F), is to fit the original DeFries-Fulker model. The third option is only used when dominance coefficients are provided, and is based on the formulation by Waller (Waller 1994).

Usage

```
DFanalysis(
  data = NULL,
  proband,
  sibling,
  Rs,
  Ds = NULL,
  RK = T,
  robust = T,
  DE = T,
  betasonly = F,
  typicalSE = F
)
```

Arguments

data	A dataframe. This is not necessary as the variables can be passed directly via the other arguments.
proband	Called "proband" for historical reasons this is the variable on the left hand side of the regression.
sibling	The right hand side version of proband. This would be the matched sibling scores.
Rs	This is the vector of relatedness coefficients
Ds	A vector of dominance coefficients. 1 for MZ twins, .25 for DZ twins and full siblings. The default is null, and no value should be provided if using the ACE model. This should only have a non-null value when fitting an ADE model. There is an RK version of this model, however it is not based on published work. The RK version uses double entered (and mean centered) data in order to drop the intercept term and the extraneous regression coefficient (both of which can be constrained to 0 when the phenotypic mean is 0). Initial simulations suggest that this formulation provides accurate parameter estimates, however the original formulation can be used by simply setting RK=F. It is assumed that, if RK=T, that DE=T (i.e., do NOT double enter data prior to analysis if using the ADE model).
RK	Use the Rodgers and Kohler simplified version of the DF model (recommended). Data should not be double entered prior to analysis.
robust	Use the Kohler and Rodgers robust standard errors (recommeneded when using double entered data)
DE	Will the data need to be double entered?
betasonly	If TRUE only the beta weights from the regression analysis will be returned.
typicalSE	Should the typical regression standard errors be used? Default is false.

Value

The results from MyLM

Examples

```
TwinData<-DFSimulated(2000,2000,.3,.3)
p<-TwinData[,1]
s<-TwinData[,2]
r<-TwinData[,3]
DFanalysis(data=NULL, p,s,r)
```

`DFSimulated` *DFSimulated*

Description

DFSimulated

Usage

```
DFSimulated(MZ = 250, DZ = 250, a2 = 0.3, c2 = 0.3)
```

Arguments

`MZ`	Number of MZ twins to simulate
`DZ`	Number of DZ twins to simulate
`a2`	Heritability (proportion of variance)
`c2`	Shared environment (proportion of variance)

Value

A dataframe

Examples

```
TwinData<-DFSimulated(200,200,.3,.3)
```

`DFSimulatedChisq` *DFSimulatedChisq*

Description

DFSimulatedChisq

Usage

```
DFSimulatedChisq(MZ = 250, DZ = 250, a2 = 0.3, c2 = 0.3, df = 10)
```

Arguments

MZ	Number of MZ twins to simulate
DZ	Number of DZ twins to simulate
a2	Heritability (proportion of variance)
c2	Shared environment (proportion of variance)
df	Total degrees of freedom for the Chi-Square variable

Value

A dataframe of Chi-Square distributed outcome observations for MZ and DZ twins

Examples

```
TwinData<-DFSimulatedChisq(200,200,.3,.3, 10)
```

DFSimulatedChisqNew *DFSimulatedChisqNew*

Description

DFSimulatedChisqNew

Usage

```
DFSimulatedChisqNew(MZ = 250, DZ = 250, a2 = 0.3, c2 = 0.3, df = 5)
```

Arguments

MZ	Number of MZ twins to simulate
DZ	Number of DZ twins to simulate
a2	Heritability (proportion of variance)
c2	Shared environment (proportion of variance)
df	Total degrees of freedom for the Chi-Square variable

Value

A dataframe of Chi-Square distributed outcome observations for MZ and DZ twins

Examples

```
TwinData<-DFSimulatedChisqNew(200,200,.3,.3, 10)
```

doubleEnter *DoubleEnter*

Description

DoubleEnter

Usage

```
doubleEnter(proband, sibling, Rs)
```

Arguments

proband	The proband scores
sibling	The matched sibling scores
Rs	The relatedness coefficients

Value

A dataframe

Examples

```
X<-DFSimulated(10,10,.2,.2)
Y<-doubleEnter(X[,"proband"], X[,"sibling"], X[,"Rs"])
```

findSa *findSa*

Description

This is an implementation of the YHY bootstrap covariance matrix.

Usage

```
findSa(S, fitted, p, a = 0.5, df, n, tau = NULL, tol = 1e-07)
```

Arguments

S	Sample covariance matrix
fitted	The fitted covariance matrix
p	the number of columns in the covariance matrix
a	the starting value for the a parameter
df	the degrees of freedom in the model

n	the number of participants in the model
tau	the population tau. If no tau is provided, the estimated tau from the model will be used
tol	the difference between ga and tau at which the function will converge

Value

a list of the "a" adjusted covariance matrix, Sa, the tau, ga, and the number of interations.

Examples

```
require(Omisc)
require(lavaan)
set.seed(2^7-1)
modelTest<-'
LV1=~ .7*x1+.8*x2+.75*x3+.6*x4
LV2=~ .7*y1+.8*y2+.75*y3+.6*y4
LV1~~.3*LV2
LV1~~1*LV1
LV2~~1*LV2
'
modelFit<-'
LV1=~ x1+x2+x3+x4
LV2=~ y1+y2+y3+y4
LV1~~start(.5)*LV2
LV1~~1*LV1
LV2~~1*LV2
'

testdata<-simulateData(modelTest, sample.nobs = 250)
fit<-cfa(modelFit, testdata)

fitted<-fitted(fit)$cov
fitted<-fitted[,1:ncol(fitted)]
S<-cov(testdata)
p<-8
a<-.5
n<-250
df<-21
findSa(S, fitted, p, .5, df, n)
```

Group_function *Grouping_function*

Description

originally from the ParallelTree package. If data argument is Null, takes a variable "x" and a matrix or dataframe of level identifiers (e.g., mother and then child IDs). Level variables should be included in order from highest level to the lowest. Listwise deletes missing data. Otherwise grabs variables from entered dataframe Group_function

Usage

```
Group_function(
  data = NULL,
  x,
  levels,
  func = mean,
  center = FALSE,
  nested = TRUE,
  append = FALSE,
  funcName = "Mean"
)
```

Arguments

data	a data frame with the x and level variables included. Default is NULL.
x	If data = NULL a dataframe of scores to have the function applied to. If data != NULL, a vector of string(s) naming the variable(s) in data to use.
levels	If data = NULL, a dataframe of grouping variables. If data != NULL, a vector of strings naming the variables in data to use. levels should be ordered from the highest level to the lowest. Group and case identifiers should be unique, if they are not unique, cases with non-unique identifiers will be grouped together.
func	A function to apply at each group. Default is mean.
center	If set to true variables will be group/person mean centered. Note that the grand mean remains unchanged by this operation. If this output is to be passed directly to Parallel_Tree the grand mean should be set to 0.
nested	Are level variables nested? Default is TRUE. If set to FALSE means will be calculated for level variable independently. FALSE may be useful in cases of crossed designs. Note that if data are nested but all identifiers are unique both within and across groups nested = FALSE and nested = TRUE will return the same result.
append	If set to true, the original data will be returned along with all created variables.
funcName	Provides way to name function used. This is used when creating names for created variables. Default is "Mean".

Value

This function returns a dataframe with variables labeled according to the level at which the function was applied. Assumed function is mean, and all variables are labeled accordingly. If an alternative function is used labels should be manually changed to reflect function used.

Examples

```
#the ChickWeight data is from base R
#nested is set to false because Chick and Time are crossed
Means_Chick<-Group_function(data=ChickWeight,x="weight", levels =c("Diet","Chick","Time"),
nested = FALSE, append=TRUE)
```

HoffPseudoStandard *HoffPseudoStandard*

Description

HoffPseudoStandard

Usage

```
HoffPseudoStandard(betas, SDX, interceptvar)
```

Arguments

betas	A vector of betas from a multilevel model
SDX	A vector of the standard deviations of the X value for each of the X's associated with the bets
interceptvar	A vector of the intercept variances at the level associated with the betas

Value

A vector of pseudostandardized coefficients

Examples

```
print("none")
```

jackknife *jackknife*

Description

jackknife

Usage

```
jackknife(data)
```

Arguments

data	The data to jackknife

Value

a list of jackknife datasets

Examples

```
data<-cbind(1:10,1:10)
result<-jackknife(data)
lapply(result,mean)
```

---

justBetas *justBetas*

Description

justBetas

Usage

```
justBetas(data, Y, X)
```

Arguments

data	A data frame
Y	The name or column number of the Y variable
X	The name(s) or column number(s) of the X variables

Value

A vector of unstandardized beta weights

Examples

```
X<-stats::rnorm(100)
Y<-stats::rnorm(100)+5*(X)
data<-cbind(Y,X)
justBetas(data,1,2)
#if you want an intercept
Y<-stats::rnorm(100)+5*(X)+5
data<-cbind(Y,X,1)
justBetas(data,1,c(2:3))
```

lbind *lbind*

Description

lbind is meant to be used in conjuction with lapply to combine elements of lists using rbind.

Usage

```
lbind(index, alist, n)
```

Arguments

index	a list of indexes. This should count the number of items to return in the final list
alist	a list of objects to be passed to rbind. They should be grouped according to which objects will be combined (e.g., if 1,2,3 are to be passed to cbind then they should be adjacent to eachother).
n	The number of objects in each group. Currently each group must consist of the same number of objects.

Value

a list

Examples

```
alist<-list(c(1,1),c(2,2),c(3,3))
index<-list(1)
n<-3
lapply(index,lbind,alist,3)
```

leave1out *leave1out*

Description

leave1out

Usage

```
leave1out(x, data)
```

Arguments

x	Which row(s) of data to leave out
data	A dataframe or matrix.

Value

The reduced dataframe or matrix

Examples

```
data<-cbind(1:10,1:10)
leave1out(5,data)
```

`MyLM` *MyLM*

Description

MyLM

Usage

```
MyLM(Y, X, robust = F, betasonly = F, typicalSE = T)
```

Arguments

`Y`	The Y variable
`X`	A matrix of X variables
`robust`	Should robust standard errors be calculated? Assumes a double entered twin dataset with twins evenly spaced in the dataset.
`betasonly`	Should only the betas be returned? Good for bootstrapping
`typicalSE`	Should the typical standard errors be included? Default is true. Can be true when robust is True.

Value

Returns a matrix of betas and standard errors

Examples

```
X<-DFSimulated(100,100,.4,.4)
Y<-RK(X[,1],X[,2],X[,3])
MyLM(Y[,1],Y[,c(2:3)],TRUE)
```

NaiveBoot *The Naive Bootstrap*

Description

The Naive Bootstrap

Usage

```
NaiveBoot(data, B = 1000, groups = NULL, keepgroups = F, size = 1)
```

Arguments

data	data to be bootstrapped
B	number of bootstrap samples to take
groups	grouping variable if there is one
keepgroups	keep the grouping variable?
size	size of the bootstrap resamples relative to the original sample

Value

a list of bootstrap resamples

Examples

```
X<-TestData()
Y<-NaiveBoot(X)
```

NaiveBoot_dep *The Naive Bootstrap*

Description

The Naive Bootstrap

Usage

```
NaiveBoot_dep(data, B = 1000, groups = NULL, keepgroups = F, size = 1)
```

Arguments

data	data to be bootstrapped
B	number of bootstrap samples to take
groups	grouping variable if there is one
keepgroups	keep the grouping variable?
size	size of the bootstrap resamples relative to the original sample

Value

a list of bootstrap resamples

Examples

```
X<-TestData()
Y<-NaiveBoot(X)
```

`resample` *resample*

Description

resample

Usage

```
resample(X, size)
```

Arguments

`X`	A vector to be resamples
`size`	The size of the resulting vector. Should be a number such that size*nrow(X) is a whole number

Value

A vector of resampled X values

Examples

```
X<-c(1:10)
resample(X,.5)
```

`RK` *RK*

Description

RK

Usage

```
RK(proband, sibling, Rs, DE = T)
```

Arguments

proband	column name or number of the proband
sibling	column name or number of the siblings
Rs	column name or number of the relatedness coefficients
DE	Should the data be double entered?

Value

A dataframe

Examples

```
X<-DFSimulated(100,100,.3,.3)
Y<-RK(X[,1],X[,2],X[,3])
```

Sfunc *Sfunc*

Description

function for calculating the matrices for the Kohler Rodgers SE

Usage

```
Sfunc(X, e)
```

Arguments

X	A matrix of X variables
e	A matrix of error terms

Value

A matrix

Examples

```
print("Nah")
```

standardBootIntervals *standardBootIntervals*

Description

This returns the quantiles of the bootstrap samples specified by the user. The quantiles uses the type=4 argument of the quantile function, which appears to function best.

Usage

```
standardBootIntervals(boot, lower = 0.025, upper = 0.975)
```

Arguments

boot	A vector of bootstrap results
lower	the lower alpha
upper	the upper alpha

Value

A matrix of the mean, median, min, max, lower and upper CI values

Examples

```
data<-DFSimulated()
boots<-NaiveBoot(data, groups="Rs", keepgroups=TRUE, B=100)
boots<-bootAnalysis(boots, cbind, DFanalysis,1,2,3,TRUE,FALSE,TRUE,TRUE,FALSE)
apply(boots,1, standardBootIntervals)
DFanalysis(data,1,2,3)
```

TestData *TestData*

Description

Simple function for creating a dataset of two related variables.

Usage

```
TestData(nobs = 1000, intercept = 0, beta = 5, meanX = 0, sdX = 1, sdYerr = 1)
```

Arguments

nobs	Number of observations, defaults to 1000
intercept	Intercept of the regression. Defaults to 0
beta	Beta for the regression equation, defaults to 5
meanX	Mean of X, defaults to 0
sdX	Standard deviation of X, defaults to 1
sdYerr	Variance of the error term of Y, defaults to 1

Value

A dataframe with an X and Y variable produced by the entered parameters

Examples

```
X<-TestData()
```

---

uniboot *Univariate Bootstrap*

Description

WARNING: This function can't be used with data that is already fed through the RK function. The correlation matrix will not be positive definite.

Usage

```
uniboot(
  data,
  B = 1000,
  groups = NULL,
  keepgroups = F,
  size = 1,
  HIcor = NULL,
  samplefrom = "group",
  use = "everything",
  standardized = T
)
```

Arguments

data	The data frame to be resampled
B	The number of bootstrap samples.
groups	A grouping variable name
keepgroups	Should the grouping variable be kept in the final datasets?

size	The size of the bootstrap sample to be returned. Should be as a proportion and must be evenly divided into nrow(data).
HIcor	If a hypothesis imposed correlation matrix is to be used, this argument takes a list of hypothesized correlation matrices. IT MUST BE A LIST OF ONE OR MORE MATRICES. Multiple matrices can be entered in the case of grouped data (one for each group). If the nil-null correlation is to be used an identity matrix can be entered here (the same size as the appropriate correlation matrix).
samplefrom	Takes one of either "group" or "whole". When doing bootstrapping of grouped data this tells uniboot if the whole sample should be used as the sampling frame for each group (whole), or not (group). "group" should be used unless it is believed that all groups share the same underlying marginal distribution for each variable (e.g., the same mean and variance in the case of normally distributed data).
use	The missing data method for cor. Default is R's default "everything".
standardized	should the resampled data be standardized? The default is TRUE. This is computationally more efficient (the data are standardized as a step during the diagonalization procedure).

Value

A list of bootstrap samples

Examples

```
data<-TestData()
X<-uniboot(data,1000)
```

`unibootsample` *unibootsample*

Description

unibootsample

Usage

```
unibootsample(data, size)
```

Arguments

data	A dataframe or matrix to be univariately bootstrapped
size	size of each bootstrap sample as a fraction of the total sample size. Total sample size must be evenly divisible by "size".

Value

A matrix or dataframe with nrow=nrow(X)*size

Examples

```
X<-c(0:9)
Y<-c(20:29)
Z<-cbind(X,Y)
unibootsample(Z,1)
```

`unibootVar` *unibootVar*

Description

unibootVar

Usage

```
unibootVar(X, times)
```

Arguments

`X`	The variable
`times`	The number of times the variable is repeated in the univariate sampling frame. This is going to be equal to the number of variables being univariately sampled

Value

The variance of the variable in the univariate sampling frame

Examples

```
X<-c(1,2)
times<-100
unibootVar(X,times)
var(X)
```

`uniboot_dep` *Univariate Bootstrap*

Description

WARNING: This function can't be used with data that is already fed through the RK function. The correlation matrix will not be positive definite.

Usage

```
uniboot_dep(
  data,
  B = 1000,
  groups = NULL,
  keepgroups = F,
  size = 1,
  HIcor = NULL,
  samplefrom = "group",
  use = "everything"
)
```

Arguments

data	The data frame to be resampled
B	The number of bootstrap samples. Alternatively "sampleframe" which will return the univariate sampling frame. "samplefrom" is not advised when there are many observations and/or many variables as the returned dataframe will be quite large.
groups	A grouping variable name
keepgroups	Should the grouping variable be kept in the final datasets?
size	The size of the bootstrap sample to be returned. Should be as a proportion and must be evenly divided into nrow(data).
HIcor	If a hypothesis imposed correlation matrix is to be used, this argument takes a list of hypothesized correlation matrices. IT MUST BE A LIST OF ONE OR MORE MATRICES. Multiple matrices can be entered in the case of grouped data (one for each group). If the nil-null correlation is to be used an identity matrix can be entered here (the same size as the appropriate correlation matrix).
samplefrom	Takes one of either "group" or "whole". When doing bootstrapping of grouped data this tells uniboot if the whole sample should be used as the sampling frame for each group (whole), or not (group). "group" should be used unless it is believed that all groups share the same underlying marginal distribution for each variable (e.g., the same mean and variance in the case of normally distributed data).
use	The missing data method for cor. Default is R's default "everything".

Value

A list of bootstrap samples

Examples

```
data<-TestData()
X<-uniboot(data,1000)
```

zScore *Title*

Description

Title

Usage

```
zScore(X, reps = 1)
```

Arguments

X	The vector to be turned into z scores
reps	The number of reps the vector is to be repeated. This will only be used in univariate bootstrapping. The default is 1.

Value

A vector of z scores.

Examples

```
X<-c(1:10)
zScore(X)
```

zScoreData *centerData*

Description

centerData

Usage

```
zScoreData(data)
```

Arguments

data	The data to be converted to z scores

Value

Data converted to z scores

Examples

```
X<-data.frame(X=c(1:4),Y=c(6:9))
zScoreData(X)
```

Index