Package ‘boodd’

March 3, 2022

Type Package

Title Functions for the book ``Bootstrap for Dependent Data''

Version 0.1

Date 2017-05-30

Author Patrice Bertail [aut], Bernard Desgraupes [aut, cre], Anna Dudek [ctb]

Maintainer Bernard Desgraupes <bernard.desgraupes@parisnanterre.fr>

Description Companion package, functions, data sets, examples for the book
Patrice Bertail, Bernard Desgraupes and Anna Dudek (2020), Bootstrap for Dependent Data.

License GPL (>= 2)

URL http://www.r-project.org

Collate main.R periodic.R utils.R jackknife.R ppw.R

Encoding utf8

Depends stats, tseries

R topics documented:

aidedboot 2
blockboot 3
boodd 5
bootglm 6
boots 7
bootsemi 9
class.boodd 11
confint.boodd 12
embb 14
embb.sample 16
fieldboot 17
freqboot 19
jackFunc 20
jackVar 22
meanCoeff 24
plot.boodd 26
qVar 27
regenboot 28
seasonalMean 31
sieveboot 32
summary.boodd 34

Index **35**

aidedboot *Aided Frequency Bootstrap*

Description

The Aided Frequency Bootstrap (AFB) estimates functionals of the spectral density by bootstrapping the periodograms using the quotient of two spectral densities.

The idea underlying the Aided Frequency Bootstrap is importance sampling. It was introduced by Kreiss and Paparoditis (2003) and allows to better mimic the asymptotic covariance structure of the periodogram in the bootstrap world. Kreiss and Paparoditis (*KP2003*) considered a spectral density which is easy to estimate (typically based on a sieve AR representation of the time series), say $f_{AR}(\omega)$. Then putting $q(\omega) = \frac{f(\omega)}{f_{AR}(\omega)}$, the problem is to estimate this quantity to generate bootstrap values of the periodogram.

Usage

```
aidedboot(x,XI,g,B,order=NULL,kernel="normal",bandwidth)
```

Arguments

x	a vector or time series.
XI	a list of functions defined on the interval $[0, \pi]$.
g	a numeric function taking length(XI) arguments.
B	the number of bootstrap samples.
order	(integer) the order of the autoregressive sieve process.
kernel	a character string which determines the smoothing kernel.
bandwidth	the kernel bandwidth smoothing parameter.

Details

The argument x is supposed to be a sample of a real valued zero-mean stationary time series.

The autoregressive sieve process of order $l = l(n)$ is modelled as

$$X_t = \sum_{k=1}^{l} \psi_k X_{t-k} + \epsilon_t$$

with $E(\epsilon_t) = 0, Var(\epsilon_t) = \sigma^2(l)$.

We estimate functionals of the spectral density $T(f)$ of the form

$$T(f) = g(A(\xi, f))$$

where g is a third order differentiable function,\

$$A(\xi, f) = (\int_0^{\pi} \xi_1(\omega) f(\omega) d\omega, \int_0^{\pi} \xi_2(\omega) f(\omega) d\omega, ..., \int_0^{\pi} \xi_p(\omega) f(\omega) d\omega)$$

and

$$\xi = (\xi_1, \ldots, \xi_p) : [0, \pi] - > R^p.$$

If the order argument is not specified, its default value is $floor(4 * (n/log(n))^(1/4))$.

The kernel argument has the same meaning as with the freqboot function.

Value

aidedboot returns an object of class boodd (see class.boodd).

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

[KP2003] Kreiss, J.-P. and Paparoditis, E. (2003). Autoregressive aided periodogram bootstrap for time series. *Ann. Stat.* **31** 1923–1955.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

freqboot

Examples

```
n <- 200
x <- arima.sim(list(order=c(4,0,0),ar=c(0.7,0.4,-0.3,-0.1)),n=n)
B <- 299
one <- function(x) {1}
XI <- list(cos,one)
g <- function(x,y) {return(x/y)}
ord <- floor(4*n^0.25/log(n)^0.5)
boo <- aidedboot(x,XI,g,B,order=ord)
```

---

blockboot *Block Bootstrap*

Description

The function applies block bootstrap methods to a time series.

This function allows the following block bootstrap methods to be used: the Moving Block Bootstrap (*kunsch* and *LiuSi92*), the Nonoverlapping Block Bootstrap (*Carlstein1986*), the Circular Block Bootstrap (*PolRom92*), the Stationary Bootstrap (*PolRom94*) and the Generalized Seasonal Block Bootstrap (*DLPP2014*).

Usage

```
blockboot(x,func,B,length.block,...,
    method=c("movingblock","nonoverlapping","circular","stationary","seasonal"),
 period)
```

Arguments

x	a time series.
func	the function to apply to each sample.
B	the number of bootstrap samples.
length.block	length of blocks.
...	optional additional arguments for the func function.
method	the block bootstrap method.
period	an integer specifying the length of the period.

Details

The possible values of the method argument are: *"movingblock"*, *"nonoverlapping"*, *"circular"*, *"stationary"* or *"seasonal"*. If it is not specified, the default method is *"movingblock"*. Method names may be abbreviated.

Value

blockboot returns an object of class boodd.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

[kunsch] Künsch, H. (1989). The jackknife and the bootstrap for general stationary observations. *Ann. Statist.*, 17, 1217-1241.

[LiuSi92] Liu, R. and Singh, K. (1992). Moving block jackknife and bootstrap capture weak dependence. Exploring the Limits of Bootstrap, Wiley Ser. Probab. Math. Statist. Probab. Math. Statist. Wiley, New York, pp 225-248.

[PolRom94] Politis, D.N. and Romano, J.P. (1994). The stationary bootstrap. *J. Amer. Statist. Assoc.*, 89, 1303–1313.

[PolRom92] Politis, D.N. and Romano, J.P. (1992). A circular block-resampling procedure for stationary data. *Exploring the Limits of Bootstrap*, Wiley Ser. Probab. Math. Statist. Probab. Math. Statist. Wiley, New York, pp 263-270.

[DLPP2014] Dudek, A.E., Le\'skow, J., Paparoditis, E. and Politis, D. (2014a). A generalized block bootstrap for seasonal time series. *J. Time Ser. Anal.*, 35, 89-114.

[Carlstein1986] Carlstein E. (1986). The use of subseries methods for estimating the variance of a general statistic from a stationary time series. *Annals of Statist.*, 14, 1171-1179.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

boots, bootsemi, plot.boodd, confint.boodd, fieldboot, jackVarBlock.

Examples

```
B <- 299
data(airquality)
x <- airquality$Wind
n <- length(x)
b <- floor(sqrt(n))
boo1 <- blockboot(x,mean,B,b,method="moving")
plot(boo1,main="MBB")


x <- nottem
boo2 <- blockboot(x,mean,B,b,period=12,method="seasonal")
plot(boo2,main="GSBB")
```

---

boodd *Package overview: bootstrap for dependent data*

Description

The package boodd contains functions, datasets and examples to accompany the text book *Bootstrap for Dependent Data* by Patrice Bertail and Anna Dudek.

Details

Version:	0.1
Date:	2022-03-03
License:	GPL (>= 2)

A list of functions:

boots - Bootstrap in the i.i.d. case.

bootsemi - Semi-parametric bootstrap for time series.

blockboot - Block bootstrap(s) for stationary time series.

regenboot - Regenerative Bootstrap.

Author

Patrice Bertail
<patrice.bertail@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X
Chaire Big Data, TéléComParis-Tech

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

Anna Dudek
Department of Applied Mathematics
AGH University of Science and Technology - Krakow, Poland

References

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

---

bootglm *Bootstrap for Generalized Linear Model*

Description

Parametric bootstrap for generalized linear model.

Usage

```
bootglm(model,data,func,B,...)
```

Arguments

model	an object of class lm or glm
data	the dataframe used to fit the model
B	the number of bootstrap samples.
func	the function to apply to each sample.
...	optional additional arguments for the func function

Details

The parametric bootstrap simply consists in resampling data in the model with estimated parameters. bootglm uses this principle for generalized linear models conditionally to the explanatory variables (see *beran97*)

The model argument must be a model fitted with the lm or glm functions.

Value

bootglm returns an object of class boodd (see class.boodd).

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

[beran97] Beran, R. (1997). *Diagnosing Bootstrap Success*, Annals of the Institute of Statistical Mathematics 49, 1-24.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

bootsemi.

Examples

```
B <- 299

# Family binomial
x <- runif(100)
e <- 0.5*rnorm(100)
y <- (x + e>0)
data <- data.frame(x,y)
glm_probit <- glm(y ~ x, family=binomial(link="probit"),data=data)
coeff <- function(data){gg=glm(y ~ x, family=binomial(link="probit"),data=data)$coeff}

boo1 <- bootglm(glm_probit,data,coeff,B) # parametric bootstrap of the coeff of a probit model
plot(boo1,main=c("Boostrap of GLM : probit model","Coefficient"))

# coeffv : a function to return coeff and variance of coefficients
coeffv <- function(data){
  gg <- glm(y~ x, family=binomial(link="probit"),data=data)
  var <- diag(summary(gg)$cov.u)
  c(gg$coeff,var)
}
# Parametric bootstrap of all coeff and variances
boo2 <- bootglm(glm_probit,data,coeffv,B)
# Construct all type of confidence intervals including bootstrap-t
# and symmetric bootstrap-t
confint(boo2,method="all")

# Family Poisson (Poisson regression)
counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
data <- data.frame(treatment,outcome,counts)
gl <- glm(counts ~ outcome + treatment,family=poisson())
meancounts <- function(data) {mean(data$counts)}
boo3 <- bootglm(gl,data,meancounts,B)
```

---

boots — *Bootstrap for the iid case*

Description

Bootstrap for the iid case as described in chapter 2 of [3]. See also [1] and [2] for details.

Usage

```
boots(x,func,B,smooth=FALSE,moonsize=NULL,mreplace=TRUE,...)
```

Arguments

x	a vector or a matrix.
func	the function to apply to each sample.
B	the number of bootstrap samples.
smooth	(logical) specify smooth bootstrap.
moonsize	size for 'm out of n' bootstrap.
mreplace	(logical) TRUE if bootstrap is done with replacement in 'm out of n'.
...	optional additional arguments for the func function.

Details

The function boots performs a naive bootstrap in the iid case. The func argument must be a function whose first argument is a vector and which returns either a single value or a vector.

The x argument can be a vector or a matrix. In the case of a matrix, the *rows* of the matrix are bootstrapped.

The moonsize and mreplace arguments concern *m out of n* bootstrap (aka *moon bootstrap*). The moonsize argument is an integer less than the length of x (or the number of rows if x is a matrix). The mreplace argument is a logical that indicates whether the bootstrap samples are drawn with or without replacement.

Value

An object of class boodd containing either a vector or a matrix, depending on whether the func function returns a single value or a vector. If the func function returns a vector of size n, then boots returns a matrix of size B x n.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

[1] B. Efron, R. Tibshirani(1993). *An Introduction to the Bootstrap*, Chapman and Hall.

[2] A. C. Davison, D. Hinkley (1997). *Bootstrap Methods and Their Application*, Cambridge Series in Statistical and Probabilistic Mathematics.

[3] P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

plot.boodd, confint.boodd.

Examples

```
B <- 299
n <- 200
x <- rnorm(n)
boo1 <- boots(x,mean,B)
summary(boo1)
plot(boo1)
confint(boo1)
confint(boo1,method="bperc")

# bootstrap of several statistics
mv <- function(data) {c(mean(data),var(data))} # compute both mean and variance
boo2 <- boots(x,mv,B)
# Compute both percentile and t-percentile confidence intervals when variance is bootstrapped
confint(boo2,method="all")

# Naive Bootstrap of all the output parameters of lm (linear regression) function
sigma <- 0.2
y <- x+rnorm(n)
data <- as.matrix(data.frame(x,y))
# the function of interest is here the coeff of the linear model
nlm <- function(dat){lm(dat[,2]~dat[,1])$coefficient}
boo3 <- boots(data,nlm,B)

# Smoothed bootstrap for quantiles
boo4 <- boots(x,median,B) # without smoothing
plot(boo4)
boo5 <- boots(x,median,B,smooth=TRUE) # with smoothing using a cross-validation estimator of the window
plot(boo5)

# Moon bootstrap
n <- 10000
x <- rnorm(n)
boo6 <- boots(x,max,B) # i.i.d bootstrap is not consistant for the max
boo7 <- boots(x,max,B,moonsize=sqrt(n)) # a reasonable approximation with moon bootstrap
boo8 <- boots(x,max,B,moonsize=sqrt(n),mreplace=TRUE) # quite similar with or without
# due to the fact that the probability to observe ties in a bootstrap sample  when moonsize=sqrt(n) is very small.
```

---

bootsemi *Semiparametric Bootstrap*

Description

The function performs a semiparametric bootstrap for a general statistics, using a time-series model.

Usage

```
bootsemi(x,func,B,...,model=c("ARIMA","GARCH"),params,
          model.fit=NULL,model.sim=NULL)
```

Arguments

x	a time series.

func	the function to apply to each sample.
B	the number of bootstrap samples.
...	optional additional arguments for the func function.
model	the chosen model to fit the time series.
params	the model parameters (see below).
model.fit	fitting function for generic model (see below).
model.sim	simulation function for generic model(see below).

Details

The default basic models currently supported are : *ARIMA* and *GARCH*.

The argument params specifies the parameters of the chosen model. In the case of *ARIMA*, it is a vector of the form *c(p,q)* or *c(p,d,q)*. In the case of *GARCH*, it is a vector of the form *c(q)* or *c(p,q)* corresponding to an *ARCH(q)* or *GARCH(p,q)* model respectively.

Alternatively, one can specify two functions in the model.fit and model.sim arguments. They are used to implement a generic bootstrap. The model.fit function has the following prototype:

```
model.fit(x,params)
```

It receives the params argument specified in the bootsemi function. It should return an object describing the model (typically a list containing all the necessary components for the model). The model.sim function has the following prototype:

```
model.sim(model,innovations,params)
```

The innovations argument is a resampled vector of centered residuals. It should build a new trajectory of the original process using the data contained in the model object provided by the model.fit function.

The Examples section below shows how this can be done in the case of a Threshold Autoregressive (*TAR*) process.

Value

bootsemi returns an object of class boodd (see class.boodd).

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

boots, blockboot, bootglm, plot.boodd, confint.boodd.

Examples

```
# An ARIMA(2,1) process
library(stats)
B <- 299
n <- 200
x <- arima.sim(model=list(ar=c(0.8,-0.4),ma=c(0.2)),n=n)
boo1 <- bootsemi(x,mean,B,model="ARIMA",params=c(2,1))
plot(boo1)

## Not run:
# A GARCH(1,1) process (needs the TSA package)
library(TSA)
x <- garch.sim(alpha=c(1.5,0.4),beta=0.2,n=n)
# bootstrap of several statistics
mv <- function(data) {c(mean(data),var(data))} # compute both mean and variance
boo2 <- bootsemi(x,mv,B,model="GARCH",params=c(1,1))

# A TAR(1,1,1) process using the generic implementation.
# This example needs the TSA package for the tar and tar.sim functions.
library(TSA)
x <- tar.sim(n=n,Phi1=c(0,0.5),Phi2=c(0,-1.8),thd=-1,d=1,p=1,sigma1=1,sigma2=2)$y
# Define the fitting function. The 'params' argument is of the form c(p1, p2, d).
myfit <- function(x,params) {
   res <- tar(x, p1=params[1], p2=params[2], d=params[3], order.select=FALSE)
   return(res)
}
# Define the simulation function. The 'fit' argument is a TAR model fitted by the tar function.
mysim <- function(fit,innov,params) {
   n <- length(fit$y)
   nx <- tar.sim(fit,ntransient=0,n=n)$y
   return(nx)
}
boo3 <- bootsemi(x,mean,B,params=c(1,1,1),model.fit=myfit,model.sim=mysim)
print(boo3)

## End(Not run)
```

---

class.boodd *Objects of class boodd*

Description

The bootstrap functions return an object of class boodd containing the generated data. Some generic functions may be applied to these objects. See their documentation for more details.

Details

The functions plot, confint and summary can be applied directly to all the objects of class boodd.

Value

An object of class boodd is a list containing at least two components :

$s is a vector or matrix of the values of the statistic for all the bootstrap samples. $Tn is the value of the statistic for the initial series.

References

Efron, B., Tibshirani, R. (1993). *An Introduction to the Bootstrap*, Chapman and Hall.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

confint.boodd, plot.boodd, summary.boodd, boots, blockboot, regenboot, bootsemi.

Examples

```
B <- 299
n <- 200
x <- rnorm(n)
boo1 <- boots(x,mean,B)
print(boo1)
names(boo1)

# bootstrap of several statistics
mv <- function(data) {c(mean(data),var(data))}
boo2 <- boots(x,mv,B)
print(boo2)
```

---

confint.boodd   *Confidence intervals for objects of class boodd*

Description

The bootstrap functions return an object of class boodd containing the generated data. The generic function confint may be applied to these objects.

Usage

```
## S3 method for class 'boodd'
confint(object,alpha=0.05,method=c("perc","bperc","aboot","tboot","tsymboot","all"),recenter,...)
```

Arguments

object	an object of class boodd.
alpha	error level for confidence interval.
method	method used to build the confidence interval. The default is *perc*.
recenter	logical. If TRUE, indicates whether to center the intervals on the mean value of the bootstrap samples (only for *tboot* or *tsymboot* methods).
...	additional arguments.

Details

The function confint.boodd provides a confidence interval. Several methods are available (see *EfronTibshirani1993*):

- *perc*: percentile
- *bperc*: basic percentile
- *aboot*: asymptotic bootstrap
- *tboot*: bootstrap-t
- *tsymboot*: symmetric bootstrap-t
- *all*: all the previous methods

The *tboot* and *tsymboot* methods require the function func to which the bootstrap method is applied to return an even number of values corresponding to estimates of the parameters and estimates of their variances.

The *recenter* argument is used only by the *tboot* and *tsymboot* methods and is *TRUE* by default (unless the object was obtained by *Circular Block Bootstrap*, in which case it is *FALSE* by default).

Value

If the method argument is not "all", the function confint.boodd returns a two-column matrix representing the lower and upper bounds of the interval. Each row of the matrix corresponds to the variable to which the interval applies. The default value of the method argument is "perc". If the method argument is "all", the function confint.boodd returns a list with the confidence intervals for all supported methods.

References

[EfronTibshirani1993] Efron, B., Tibshirani, R. (1993). *An Introduction to the Bootstrap*, Chapman and Hall.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

plot.boodd, summary.boodd, boodd-class.

Examples

```
B <- 299
x <- round(rnorm(15),3)
boo1 <- boots(x,mean,B)
confint(boo1)
confint(boo1,method="bperc")

# bootstrap of several statistics
mv <- function(data) {c(mean(data),var(data))} # compute both mean and variance
boo2 <- boots(x,mv,B)
# Compute both percentile and t-percentile confidence intervals when variance is bootstrapped
confint(boo2,method="all")
```

embb *Characteristics for Extended Moving Block Bootstrap class*

Description

The class of functions that uses the sample obtained with the Extension of Moving Block Bootstrap (EMBB) method or its circular version (CEMBB). The functions calculate the seasonal means, seasonal variances and seasonal autocovariances when a PC time series with period length d is considered. For both PC and APC time series, the functions calculate the Fourier coefficients of the mean and autocovariance functions.

Usage

```
## S3 method for class 'embb'
seasonalMean(x,d,...)
## S3 method for class 'embb'
seasonalVar(x,d,...)
## S3 method for class 'embb'
seasonalACF(x,tau,d,...)
## S3 method for class 'embb'
meanCoeff(x,freq,...)
## S3 method for class 'embb'
acfCoeff(x,tau,freq,...)
```

Arguments

x	An object of class embb.
d	A positive integer which is the period length.
tau	Single lag or vector of lags (integers).
freq	Vector of frequencies.
...	Additional arguments.

Details

These methods apply to objects of class embb typically obtained with the embb.sample functions.

Value

The seasonalMean and seasonalVar functions return a vector of length d.

seasonalACF returns either a vector of length d if a single lag tau is specified, or a matrix with length(tau) rows and d columns if tau is a vector.

meanCoeff returns a vector of the same length as freq.

acfCoeff returns either a vector of length length(freq) if a single lag tau is specified, or a matrix with length(tau) rows and length(freq) columns if tau is a vector.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

A.E. Dudek (2015). Circular block bootstrap for coefficients of autocovariance function of almost periodically correlated time series, *Metrika*, 78(3), 313-335.

A.E. Dudek (2018). Block bootstrap for periodic characteristics of periodically correlated time series. *Journal of Nonparametric Statistics*, 30(1), 87-124.

Anna Dudek, Harry Hurd and Wioletta Wojtowicz (2016). perARMA: Periodic Time Series Analysis. https://CRAN.R-project.org/package=perARMA

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

embb.sample, seasonalMean.default, seasonalVar.default, seasonalACF.default.

Examples

```
# Generate a PARMA(2,1) sequence. This requires the perARMA package.
library(perARMA)
T = 12
n = 480
p = 2
a = matrix(0,T,p)
q = 1
b = matrix(0,T,q)

a[1,1] = .8
a[2,1] = .3
phia <- ab2phth(a)
phi0 = phia$phi
phi0 = as.matrix(phi0)

b[1,1] = .7
b[2,1] = .6
thetab <- ab2phth(b)
theta0 = thetab$phi
theta0 = as.matrix(theta0)

del0 = matrix(1,T,1)
PARMA21 <- makeparma(n,phi0,theta0,del0)
x <- PARMA21$y
lb <- 41
# Get an embb object
em <- embb.sample(x,lb,method="movingblock")
d <- 12
seasonalMean(em,d)
seasonalVar(em,d)
```

```
h <- c(1,3,6)
seasonalACF(em,h,d)
freq <- 2*pi*(0:(d-1))/d
meanCoeff(em,freq)
acfCoeff(em,h,freq)
```

---

embb.sample *EMBB method*

Description

The function constructs the bootstrap sample using the Extension of Moving Block Bootstrap (EMBB) method, which is valid for periodically correlated and almost periodically correlated time series.

Usage

```
embb.sample(x,length.block,method=c("movingblock","circular"))
```

Arguments

x	almost periodically correlated time series.
length.block	length of blocks.
method	bootstrap method.

Details

The argument method can be set to *"movingblock"* (in the case of the *EMBB*) or to *"circular"* (in the case of the circular version of the *EMBB*). Method names may be abbreviated.

Value

embb.sample returns a matrix whose first column is the bootstrapped sample and second column contains the original time indices of the chosen observations.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

A.E. Dudek (2015). Circular block bootstrap for coefficients of autocovariance function of almost periodically correlated time series, *Metrika*, 78(3), 313-335.

A.E. Dudek (2018). Block bootstrap for periodic characteristics of periodically correlated time series. *Journal of Nonparametric Statistics*, 30(1), 87-124.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

embb, blockboot.

Examples

```
# Generate a PARMA(2,1) sequence. This requires the perARMA package.
library(perARMA)
T = 12
n = 480
p = 2
a = matrix(0,T,p)
q = 1
b = matrix(0,T,q)
a[1,1] = .8
a[2,1] = .3
phia <- ab2phth(a)
phi0 = phia$phi
phi0 = as.matrix(phi0)
b[1,1] = .7
b[2,1] = .6
thetab <- ab2phth(b)
theta0 = thetab$phi
theta0 = as.matrix(theta0)
del0 = matrix(1,T,1)
x <- makeparma(n,phi0,theta0,del0)$y
lb <- 41

# Moving blocks method
embb.sample(x,length.block=lb,method="movingblock")

# Circular method
embb.sample(x,length.block=lb,method="circular")
```

---

fieldboot *Random Field Bootstrap*

Description

The function bootstraps a data array representing a random field using the Moving Block Bootstrap, Circular Block Bootstrap or Nonoverlapping Block Bootstrap.

Usage

```
fieldboot(arr,func,B,blocklens,...,method=c("movingblock","nonoverlapping","circular"))
```

Arguments

arr	the data array
func	the function to apply to each sample.
B	the number of bootstrap samples.
blocklens	a scalar or a vector of block lengths.
method	method for array reconstruction.
...	optional additional arguments for the func function.

Details

The arr argument is a multidimensional array (with at least two dimensions).

The blocklens argument specifies the block lengths in all dimensions. If it is a scalar integer value, the same size is used for all dimensions, otherwise it must be an integer vector with as many elements as there are dimensions in the array.

Different block bootstrap methods are supported to rebuild an array by bootstrapping its blocks. The possible values of the method argument are: *"movingblock"*, *"nonoverlapping"*, or *"circular"*. If it is not specified, the default method is *"movingblock"*. Method names may be abbreviated.

Value

The fieldboot function returns an object of class boodd (see class.boodd).

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

P. Bertail, D. N. Politis, N. Rhomari (2000). Subsampling continuous parameter random fields and a Bernstein inequality, *Statistics*, 33(4), 367-392.

D.J. Nordman, S.N. Lahiri (2004). On optimal spatial subsample size for variance estimation, *The Annals of Statistics*, 32(5), 1981-2027.

D.N.Politis, J.P.Romano (1993). Nonparametric Resampling for Homogeneous Strong Mixing Random Fields, *J. Multivar. Anal.*, 47(2), 301-328.

D.N.Politis, J.P.Romano (1994). Large Sample Confidence Regions Based on Subsamples under Minimal Assumptions, *Ann. Statist.*, 22(4), 2031-2050.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

blockboot, jackVarBlock, jackVarField

Examples

```
# This example needs the RandomFields package
library(RandomFields)
# Simulate a gaussian random field with exponential covariance
arr <- model <- RMexp(var=1,scale=10)
# Simulate on [0,1]^2 a gid of size 100x100
from <- 0
to <- 10
x.seq <- seq(from, to, length=100)
y.seq <- seq(from, to, length=100)
arr <- RFsimulate(model,x=x.seq,y=y.seq)
arr <- as.array(arr)
# Bootstrap the array
```

```
B <- 299
blens <- c(20,10)
boo1 <- fieldboot(arr,mean,B,blens)
# Function with additional argument
meanpow <- function(arr,p) {mean(abs(arr)^p)^(1/p)}
boo2 <- fieldboot(arr,meanpow,B,blens,3)
# Circular blocks reconstruction
blens <- c(18,8)
boo3 <- fieldboot(arr,mean,B,blens,method="circular")
```

---

freqboot *Frequency Domain Bootstrap*

Description

This function implements the Frequency Domain Bootstrap (FDB).

Usage

```
freqboot(x,XI,g,B,kernel="normal",bandwidth)
```

Arguments

x	a vector or time series.
XI	a list of functions defined on the interval $[0, \pi]$.
g	a numeric function taking length(XI) arguments.
B	the number of bootstrap samples.
kernel	a character string which determines the smoothing kernel.
bandwidth	the kernel bandwidth smoothing parameter.

Details

The series x is supposed to be a sample of a real valued zero-mean stationary time series.

The XI argument is a list of functions ξ_i (i=1,...,p) used to define a linear functional of the form

$$A(\xi, f) = (\int_0^\pi \xi_1(\omega)f(\omega)d\omega, \int_0^\pi \xi_2(\omega)f(\omega)d\omega, ..., \int_0^\pi \xi_p(\omega)f(\omega)d\omega)$$

The g argument is a numeric function with *p* arguments so that the statistic computed by bootstrap is $T(f) = g(A(\xi, f))$.

An estimate of the spectral density is obtained by smoothing the periodograms of the series. The kernel argument specifies the smoothing kernel. The possible values are (the names can be abbreviated):

"normal" the Gaussian density function (the default).

"epanechnikov" the centered beta(2,2) density.

"box" the uniform density on [-1,1].

If the bandwidth argument is not specified, an optimal value is computed. Refer to the package for the optimal value.

Value

freqboot returns an object of class boodd (see class.boodd).

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

P. Bertail and A.E. Dudek (2021). Consistency of the Frequency Domain Bootstrap for differentiable functionals, *Electron. J. Statist.*, 15(1), 1-36.

Lahiri, S.N. (2003). *Resampling Methods for Dependent Data*. Springer, New York.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

aidedboot

Examples

```
set.seed(123)
n <- 120
x <- arima.sim(list(order=c(1,0,0),ar=0.7),n=n)
B <- 299
one <- function(x) {1}
XI <- list(cos,one)
g <- function(x,y) {return(x/y)}
# This gives an estimate for the autocorrelation of order 1
freqboot(x,XI,g,B,"normal")
```

---

jackFunc — *Create a function that calculates the statistic and the jackknife variance.*

Description

Create a vector-valued function that calculates both the statistics defined by 'func' and the estimated jackknife variance.

Usage

```
jackFunc(func,...)
jackFuncBlock(func,blen=NULL,...)
```

Arguments

func	the function to apply to each sample.
blen	(integer) block length.
...	optional additional arguments for the func function.

Details

The jackFunc function creates a vector-valued function that calculates both the statistics defined by 'func' and the estimated jackknife variance. This function can then be used in generic bootstrap procedure to construct bootstrap-t confidence intervals. The jackFuncBlock function is similar but uses the estimated jackknife variance based on non-overlapping blocks (calculated with the jackVarBlock function). If the blen argument is not specified, the created function uses, as a default, floor(n^(1/3)) where n is the length of the vector it is applied to.

Value

jackFunc and jackFuncBlock return a function object that takes a single vector argument.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

Efron, B. (1979). Bootstrap methods: an other look at the jackknife, *Ann. Statist.*, 7, 1-26.

Gray, H., Schucany, W. and Watkins, T. (1972). *The Generalized Jackknife Statistics*. Marcel Dekker, New-York.

Quenouille, M.H. (1949). Approximate tests of correlation in time-series, *J. Roy. Statist. Soc., Ser. B*, 11, 68-84.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

regenboot, blockboot, fieldboot.

Examples

```
# Create a function to compute both estimation and variance of the empirical skewness
func <- function(x) {mean((x-mean(x))^3)/(mean((x-mean(x))^2)^(3/2))}
x <- arima.sim(list(order=c(1,0,4),ar=0.5,ma=c(0.7,0.4,-0.3,-0.1)),n=101)
jf <- jackFunc(func)
boo1 <- boots(x,jf,299)
# Compute all confidence intervals including bootstrap-t and symmetric
# bootstrap-t methods
confint(boo1,method="all")

jfb <- jackFuncBlock(func,blen=5)
boo2 <- boots(x,jfb,299)
```

jackVar *Jackknife Variance for statistics based on i.i.d data*

Description

Estimate the variance of a statistic applied to a vector or a matrix using a jackknife procedure.

Usage

```
jackVar(x,func,...)
jackVarBlock(x,func,blen,...)
jackVarRegen(x,func,...,atom,small=NULL,s=median(x))
jackVarField(arr,func,blocklens,...)
```

Arguments

x	a vector or a matrix.
func	the function to apply to each sample.
arr	a data array.
blocklens	an integer or a vector of integers for the block lengths.
atom	atom used to cut finite states Markov chains.
s	a real number which is the center for the small set.
small	an object of class smallEnsemble (see regenboot).
...	optional additional arguments for the func function.

Details

If x is a vector of length n or a matrix with n rows, for all i in [[1,n]], the statistic func is calculated on x with its i-th element removed:

$$T_{n-1}^i = func(x[-i])$$

Then, for $T_n = func(x)$:

$$J^i = nT_n - (n-1)T_{n-1}^i,$$

$$\bar{J} = \frac{1}{n}\sum J^i,$$

$$V = \frac{1}{n(n-p)}\sum_{i=1}^{n}(J^i - \bar{J})^t(J^i - \bar{J}),$$

where p is the size of the vector returned by the function func.

The jackVarBlock function computes the same kind of estimator but using blocks of observations of length blen. The sample x is split into

$$T_{n-1}^i = func(x[-B_i]).$$

Then

$$J^i = nT_n - (n - blen)T_{n-1}^i,$$

$$\bar{J} = \frac{1}{bnum} \sum J^i,$$

$$V = \frac{1}{n\ bnum} \sum_{i=1}^{b} num(J^i - \bar{J})^t (J^i - \bar{J}).$$

The jackVarField function is similar to jackVarBlock but applied to random fields with mutidimensional blocks.

The jackVarRegen function is similar to jackVarBlock in the case of regenerative statistics. It handles variable length blocks and supports both the finite states and homogeneous Markov chains. See details in the documentation of the regenboot and smallEnsemble functions.

Value

jackVar, jackVarBlock and jackVarRegen return a scalar or a variance-covariance matrix depending on whether the function func is univariate or multivariate. If the function func returns a vector of length p, the variance-covariance matrix has size p x p.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

Efron, B. (1979). Bootstrap methods: an other look at the jackknife, *Ann. Statist.*, 7, 1-26.

Gray, H., Schucany, W. and Watkins, T. (1972). *The Generalized Jackknife Statistics*. Marcel Dekker, New-York.

Quenouille, M.H. (1949). Approximate tests of correlation in time-series, *J. Roy. Statist. Soc., Ser. B*, 11, 68-84.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

regenboot, blockboot, fieldboot.

Examples

```
set.seed(1)
x <- rnorm(101)
# Function returning a single value
func <- function(x) {sum(abs(x))}
jackVar(x,func)
# Function returning a vector with two elements
mfunc <- function(x) {c(mean(abs(x)),sd(x))}
jackVar(x,mfunc)
# Function with additional argument
funca <- function(x,p) {sum(abs(x)^p)}
jackVar(x,funca,3)
```

```
x <- arima.sim(list(order=c(1,0,4),ar=0.5,ma=c(0.7,0.4,-0.3,-0.1)),n=101)
# Jackknife variance estimator of 'func' with blocks of length blen
blen <- 10
V1 <- jackVarBlock(x,func,blen)
V2 <- jackVarBlock(x,mfunc,blen)
V3 <- jackVarBlock(x,funca,blen,2)

# jackVarRegen function in the case of finite state Markov chains
acgt <- c("A","C","G","T")
probs <- c(.3,.1,.3,.3)
n <- 100
atom <- "A"
set.seed(1)
y <- sample(acgt,n,prob=probs,repl=TRUE)
propAtom <- function(x) {
   tbl <- as.vector(table(x))
   prop <- tbl[1]/length(x)
   return(prop)
}
jackVarRegen(y,propAtom,atom=atom)
```

---

meanCoeff *Estimation of the Fourier coefficients.*

Description

For both periodically (PC) and almost periodically correlated (APC) data, the functions calculate the Fourier coefficients of the mean and autocovariance functions. The function can also be used for bootstrap samples obtained with the EMBB, CEMBB, GSBB, CGSBB.

Usage

```
meanCoeff(x,d,freq=NULL,...)
acfCoeff(x,tau,d,freq=NULL,...)

## Default S3 method:
meanCoeff(x,d,freq=NULL,...)
## Default S3 method:
acfCoeff(x,tau,d,freq=NULL,...)

## S3 method for class 'ts'
meanCoeff(x,d=frequency(x),freq=NULL,...)
## S3 method for class 'ts'
acfCoeff(x,tau,d=frequency(x),freq=NULL,...)
```

Arguments

x	Periodically or almost periodically correlated time series.
d	(Integer) Period length. By default, frequency(x).
tau	Single lag or vector of lags (integers).
freq	Vector of frequencies.
...	Additional arguments.

Details

If the freq argument is not specified, the Fourier frequencies are used: $2 * k * pi/d$ for *k=0,1,...,d* where *d* is the frequency of the time series.

The meanCoeff function implements the estimator of the Fourier coefficient of the mean at frequency γ:

$$\widehat{b}(\gamma) = \frac{1}{n} \sum_{t=1}^{n} X_t e^{-i\gamma t}$$

The acfCoeff function implements the estimator of the Fourier coefficient of the autocovariance for given lag tau at frequency λ:

$$\widehat{a}(\lambda, \tau) = \frac{1}{n} \sum_{t=1-\min\{\tau,0\}}^{n-\max\{\tau,0\}} (X_{t+\tau} - \widehat{\mu}_n(t+\tau))(X_t - \widehat{\mu}_n(t))e^{-i\lambda t}$$

Value

meanCoeff returns a vector of the same length as freq.

acfCoeff returns either a vector of length length(freq) if a single lag tau is specified, or a matrix with length(tau) rows and length(freq) columns if tau is a vector.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

A.E. Dudek (2015). Circular block bootstrap for coefficients of autocovariance function of almost periodically correlated time series, *Metrika*, 78(3), 313-335.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

Methods for class embb: meanCoeff.embb, acfCoeff.embb.

Examples

```
# Fourier frequencies for the data nottem (temperatures at Nottingham Castle)
meanCoeff(nottem)
acfCoeff(nottem,5)
# Given frequencies
meanCoeff(nottem,freq=1:6)
acfCoeff(nottem,tau=c(1,2,5),freq=1:6)
```

plot.boodd *Plot objects of class boodd*

Description

The bootstrap functions return an object of class boodd containing the generated data. The generic function plot may be applied to these objects.

Usage

```
## S3 method for class 'boodd'
plot(x,with.density=TRUE,which,byrow=FALSE,...)
```

Arguments

x	an object of class boodd.
with.density	logical. If TRUE, estimated density of the bootstrap distribution is plotted.
which	which columns of the data to plot.
byrow	logical. If TRUE, display the matrix of histograms by row.
...	additional arguments for the hist function.

Details

The function plot.boodd plots a histogram (or a matrix of histograms) of the output data with an estimation of the bootstrap density if the with.density argument is set to TRUE. If the generated data have more than 1 column, the function displays a matrix of histograms. The which argument lets you specify which columns of the boodd object should be plotted. The plot.boodd function cannot display more than 6 columns. The byrow argument indicates if the matrix is organized by rows or by columns. The main and xlab additional parameters may be vectors in order to specify different strings for each histogram.

Value

The function plot.boodd returns an invisible list containing the output of the hist function. In the case of multiple histograms, it is a list of lists.

References

Efron, B., Tibshirani, R. (1993). *An Introduction to the Bootstrap*, Chapman and Hall.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

confint.boodd, summary.boodd, boodd-class.

Examples

```
B <- 299
x <- round(rnorm(15),3)
boo1 <- boots(x,mean,B)
plot(boo1)
confint(boo1)
confint(boo1,method="bperc")

# bootstrap of several statistics
mv <- function(data) {c(mean(data),var(data))} # compute both mean and variance
boo2 <- boots(x,mv,B)
# Compute both percentile and t-percentile confidence intervals when variance is bootstrapped
confint(boo2,method="all")
```

---

qVar *Variance estimator for quantile*

Description

This is a kernel based estimator of the variance for quantile.

Usage

```
qVar(x,alpha,hn=NULL,kernel=c("gaussian","epanechnikov","rectangular"))
```

Arguments

x	a vector.
alpha	a numeric vector of probabilities.
hn	the smoothing bandwith.
kernel	the smoothing kernel to be used. Possible values are "gaussian", "epanechnikov", or "rectangular" and may be abbreviated. The default is "gaussian".

Details

Consider the cumulative distribution function F of P, which is supposed to be continuous and differentiable, with a density f. Denote a quantile $T(P) = F^{-1}(\alpha)$ for some $\alpha \in (0,1)$. Assume that

$$f(F^{-1}(\alpha)) \neq 0.$$

and define the quantile as the unique solution of the equation

$$F(T(P)) = \alpha.$$

Put $T(P) = F^{-1}(\alpha)$ and $T(P_n) = F_n^{-1}(a)$. In that case the variance estimator is given by $S(P)^2 = \alpha(1-\alpha)/f(F^{-1}(\alpha))^2$.

A kernel density estimator is defined by

$$\hat{f}_n^{h_{n,1}}(x) = \frac{1}{nh_n}\sum_{i=1}^{n} k(\frac{x - X_i}{h_{n,1}}),$$

where $h_{n,1}$ is the window or smoothing parameter.

Then the estimator of the variance is given by

$$S^2_{n,h_{n,1}} = \frac{\alpha(1-\alpha)}{\hat{f}_n^{h_{n,1}}(F_n^{-1}(\alpha))^2},$$

where the smoothing parameter is such that

$$nh^3_{n,1}/log(n)^2 -> \infty$$

and

$$nh^{2r+1-\tau}_{n,1} -> 0$$

for some $\tau \in (0,1)$.

If the bandwith argument hn is not specified, a default value of order $n^(-1/3)$ is computed by unbiased cross-validation.

Value

qVar returns a vector of variance estimates of the same length as the argument alpha.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

Chapter 8 of R.J. Serfling. *Approximation Theorems of Mathematical Statistics*, Wiley Series in Probability and Statistics.

Chapter 3 of P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

Examples

```
set.seed(123)
x <- rnorm(101)
qVar(x,seq(0, 1, 0.25))
qVar(x,0.25,kernel="epanechnikov")
```

---

regenboot *Regenerative Bootstrap*

Description

Perform regenerative bootstrap on a Markov chain in the atomic case or an approximate regenerative boostrap in the general Harris case.

Usage

```
regenboot(x,func,B,...,atom,small=NULL,s=median(x))
findBestEpsilon(x,s=median(x),plotIt=FALSE)
fastNadaraya(x,h)
smallEnsemble(s,eps,delta,trans)
```

Arguments

x	a vector representing a Markov chain.
func	the function to apply to each sample.
B	integer representing the number of bootstrap samples.
atom	either a real number or a string denoting the atom used to cut finite state Markov chains.
s	a real number which represents the center of the small ensemble.
small	an object of class smallEnsemble (see details below).
plotIt	logical. If TRUE, draw plot of the number of regenerations as a function of the small ensemble radius.
eps	a positive real number, radius for the small ensemble.
delta	a positive real number, lower bound of the transition probability on the small set.
trans	a vector representing the transition density of the Markov chain.
h	a positive real number, bandwith for kernel density estimator of the transition.
...	optional additional arguments for the func function.

Details

This function regenboot implements two different kinds of regenerative bootstrap:

- *A regenerative atomic bootstrap* used for finite state Markov chains.
- *An approximate regenerative bootstrap* used to bootstrap continuous Markov chains based on a given small set of the form *[s-eps,s+eps]* where *s* is the center and *eps* the radius.

One must specify either the atom argument or the small argument. In the first case, atom is the state used to split the Markov chain into blocks ending with the atom. In the second case, small is an object of class smallEnsemble representing the small ensemble (see the *Value* section). Such objects are typically obtained using the findBestEpsilon function but may also be constructed manually using the smallEnsemble function. By default, s is the median of the original vector.

The function fastNadaraya computes the estimated transition densities $p_n(X_i, X_{i+1})$ of the Markov chain. It is used in particular by findBestEpsilon. It is a Nadaraya kernel type estimator of the transition density, with a bandwith h provided by the user. The optimal h is automatically computed inside the function findBestEpsilon.

Value

regenboot returns an object of class boodd (see class.boodd).

findBestEpsilon and smallEnsemble return an object of class smallEnsemble which is a list with the following components:

- *s*: the middle of the small ensemble
- *epsilon*: the estimated best radius of the ensemble
- *delta*: the estimated lower bound of the transition probability
- *trans*: the estimated transition densities $p_n(X_i, X_{i+1})$

fastNadaraya returns a vector of size length(x)-1.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

Bertail, P., Clémençon, S. (2006a).*Regenerative Block Bootstrap for Markov Chains*. Bernoulli, 12(4):689–712.

P. Bertail and S. Clémençon. *Regeneration-based statistics for Harris recurrent Markov chains*, pages 1–54. Number 187 in Lecture notes in Statistics. Springer.

Radulovi\'c, D. Renewal type bootstrap for Markov chains. Test 13, 147-192.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

boots, blockboot, plot.boodd, confint.boodd.

Examples

```
B <- 299
n <- 200

# Atomic Boostrap
acgt <- c("A","C","G","T")
probs <- c(.3,.1,.3,.3)
atom <- "C"
set.seed(1)
x <- sample(acgt,n,prob=probs,repl=TRUE)
propAtom <- function(x) {
    tbl <- as.vector(table(x))
    prop <- tbl[3]/length(x)
    return(prop)
}
boo <- regenboot(x,propAtom,B,atom=atom)
plot(boo)

# Approximate regenerative bootstrap with estimated small set
ar <- arima.sim(list(c(1,0,0),ar=0.6),n=500)
# Find the small ensemble with the largest number of regenerations
sm <- findBestEpsilon(ar,s=0,plotIt=TRUE)
# Approximate regenerative bootstrap of the mean
rboo <- regenboot(ar,mean,small=sm,B=999)
```

```
# Plot the corresponding bootstrap distribution
plot(rboo)
# Compute the bootstrap percentile confidence interval
confint(rboo)
```

---

seasonalMean *Characteristics of periodically correlated time series*

Description

Calculate estimates of the seasonal means, variances and autocovariances of a periodically correlated time series.

Usage

```
seasonalMean(x,d,...)
seasonalVar(x,d,...)
seasonalACF(x,tau,d,...)

## Default S3 method:
seasonalMean(x,d,...)
## Default S3 method:
seasonalVar(x,d,...)
## Default S3 method:
seasonalACF(x,tau,d,...)

## S3 method for class 'ts'
seasonalMean(x,d=frequency(x),...)
## S3 method for class 'ts'
seasonalVar(x,d=frequency(x),...)
## S3 method for class 'ts'
seasonalACF(x,tau,d=frequency(x),...)
```

Arguments

x	A periodically correlated time series or a vector.
d	a positive integer representing period length.
tau	An integer or a vector of integers representing a single lag or a vector of lags (positive integers).
...	Additional arguments.

Details

The functions seasonalMean and seasonalVar calculate estimates of the seasonal means and variances respectively. The function seasonalACF calculates an estimator of the autocovariance for the given lags. Lags should be positive integers.

Value

The seasonalMean and seasonalVar functions return a vector of length d.

seasonalACF returns either a vector of length d if a single lag tau is specified, or a matrix with length(tau) rows and d columns if tau is a vector.

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

References

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

Hurd, H.L., Miamee, A.G. (2007). *Periodically Correlated Random Sequences: Spectral. Theory and Practice*. Wiley.

See Also

blockboot, seasonalMean.embb, seasonalVar.embb, seasonalACF.embb.

Examples

```
# Means
seasonalMean(nottem)  # The period is already in the time-series object nottem
# Variances
seasonalVar(nottem)
# Autocovariances
seasonalACF(nottem,0)
seasonalACF(nottem,1)
seasonalACF(nottem,c(1,3,6))
```

---

sieveboot *Autoregressive Sieve Bootstrap*

Description

The function applies autoregressive sieve bootstrap to stationary time series. The idea is to estimate an *AR(p)* model, with p large and to resample the centered estimated residuals to reconstruct an *AR(p)* bootstrap times series on which a given statistics is computed.

Usage

```
sieveboot(x,func,B,order=NULL,...)
```

Arguments

x	a vector of observations.
func	the function to apply to each sample.
B	a positive integer: the number of bootstrap samples.
order	a positive integer: the order of the autoregressive process.
...	optional additional arguments for the func function.

Details

The sieve bootstrap procedure approximates the given data by a large order autoregressive process.

The validity of the sieve bootstrap was obtained for $p = o((n/log(n))^{(}1/4))$. When the order is not specified, we set $p = 4 * (n^1/4)/log(n)^{(}1/2)$.

Value

sieveboot returns an object of class boodd (see class.boodd).

Author(s)

Bernard Desgraupes
<bernard.desgraupes@parisnanterre.fr>
University of Paris Ouest - Nanterre - Lab Modal'X

Source

Bühlmann, P. (1997). Sieve Bootstrap for time series. Bernoulli, 3, 123-148.

References

Bühlmann, P. (1997). Sieve Bootstrap for time series. *Bernoulli,* **3**, 123-148.

Choi, E., Hall, P. (2000). Bootstrap confidence regions computed from autoregression of abitrary order, *Journal of the Statistical Royal Society*, ser. B, **62**, 461-477.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

freqboot, aidedboot

Examples

```
n <- 200
B <- 299
x <- arima.sim(list(order=c(0,0,4),ma=c(0.7,0.4,-0.3,-0.1)),n=n)
sieveboot(x,mean,B,order=10)
```

summary.boodd *Summary for objects of class boodd*

Description

The bootstrap functions return an object of class boodd containing the generated data. The generic function summary may be applied to these objects.

Usage

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

Arguments

object	an object of class boodd.
...	additional arguments.

Details

The function summary.boodd displays basic information about the values of the samples computed by the bootsrap functions.

Value

The function summary.boodd returns the kind attribute of the function that produced the boodd object and a (possibly multicolumn) table with the quartiles, mean, min and max of the computed values of the statistic. The number of columns is the size of the return value of the function func.

References

Efron, B., Tibshirani, R. (1993). *An Introduction to the Bootstrap*, Chapman and Hall.

P. Bertail, B. Desgraupes and A. Dudek (2023), *Bootstrap for Dependent Data with "R package"*, Springer, N-Y.

See Also

confint.boodd, plot.boodd, boodd-class.

Examples

```
B <- 299
x <- round(rnorm(15),3)
boo1 <- boots(x,mean,B)
summary(boo1)

# bootstrap of several statistics
mv <- function(data) {c(mean(data),var(data))} # compute both mean and variance
boo2 <- boots(x,mv,B)
summary(boo2)
```

Index

∗ **bootstrap, frequency domain, periodogram**
aidedboot, 2
∗ **bootstrap**
blockboot, 3
boots, 7
∗ **frequency domain, estimators**
meanCoeff, 24
∗ **frequency domain, fft, periodogram**
freqboot, 19
∗ **generalized linear model**
bootglm, 6
∗ **package**
boodd, 5
∗ **random fields**
fieldboot, 17
∗ **regenerative,bootstrap,Markov chains**
regenboot, 28
∗ **seasonal, mean, variance, autocovariance**
seasonalMean, 31
∗ **semiparametric,bootstrap**
bootsemi, 9

acfCoeff (meanCoeff), 24
acfCoeff.embb, *25*
acfCoeff.embb (embb), 14
aidedboot, 2, *20*, *33*

blockboot, 3, *5*, *10*, *12*, *17*, *18*, *21*, *23*, *30*, *32*
boodd, 5
boodd-package (boodd), 5
bootglm, 6, *10*
boots, *4*, *5*, 7, *10*, *12*, *30*
bootsemi, *4*, *5*, *7*, 9, *12*

class.boodd, *3*, *6*, *10*, 11, *18*, *20*, *29*, *33*
confint.boodd, *4*, *8*, *10*, *12*, 12, *26*, *30*, *34*

embb, 14, *17*
embb.sample, *14*, *15*, 16

fastNadaraya (regenboot), 28
fieldboot, *4*, 17, *21*, *23*
findBestEpsilon (regenboot), 28
freqboot, *2*, *3*, 19, *33*

glm, *6*

hist, *26*

jackFunc, 20
jackFuncBlock (jackFunc), 20
jackVar, 22
jackVarBlock, *4*, *18*
jackVarBlock (jackVar), 22
jackVarField, *18*
jackVarField (jackVar), 22
jackVarRegen (jackVar), 22

lm, *6*

meanCoeff, 24
meanCoeff.embb, *25*
meanCoeff.embb (embb), 14

plot.boodd, *4*, *8*, *10*, *12*, *13*, 26, *30*, *34*

qVar, 27

regenboot, *5*, *12*, *21*–*23*, 28

seasonalACF (seasonalMean), 31
seasonalACF.default, *15*
seasonalACF.embb, *32*
seasonalACF.embb (embb), 14
seasonalMean, 31
seasonalMean.default, *15*
seasonalMean.embb, *32*
seasonalMean.embb (embb), 14
seasonalVar (seasonalMean), 31
seasonalVar.default, *15*
seasonalVar.embb, *32*
seasonalVar.embb (embb), 14
sieveboot, 32
smallEnsemble, *23*
smallEnsemble (regenboot), 28
summary.boodd, *12*, *13*, *26*, 34