Package ‘ifs’

December 5, 2022

Version 0.1.10

Title Iterated Function Systems

Author S. M. Iacus

Maintainer Stefano M. Iacus <siacus@iq.harvard.edu>

Description Iterated Function Systems Estimator as in Iacus and La Torre (2005) <doi:10.1155/JAMDS.2005.33>.

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-12-05 08:30:05 UTC

R topics documented:

`ifs` *IFS estimator*

Description

Distribution function estimator based on sample quantiles.

Usage

```
ifs(x, p, s, a, k = 5)
ifs.flex(x, p, s, a, k = 5, f = NULL)
IFS(y, k = 5, q = 0.5, f = NULL, n = 512, maps = c("quantile",
    "wl1", "wl2"))
```

Arguments

x	where to estimate the distribution function
p	the vector of coefficients p_i
s	the vector of coefficients s_i in: $w_i = s_i * x + a_i$
a	the vector of coefficients a_i in: $w_i = s_i * x + a_i$
k	number of iterations, default = 5
y	a vector of sample observations
q	the proportion of quantiles to use in the construction of the estimator, default = 0.5. The number of quantiles is the `q * length(y)`.
f	the starting point in the space of distribution functions
n	the number of points in which to calculate the IFS
maps	type of affine maps

Details

This estimator is intended to estimate the continuous distribution function of a random variable on [0,1]. The estimator is a continuous function not everywhere differentiable.

Value

The estimated value of the distribution function for `ifs` and `ifs.flex` or a list of 'x' and 'y' coordinates of the IFS(x) graph for `IFS`.

Note

It is asymptotically as good as the empirical distribution function (see Iacus and La Torre, 2001). This function is called by IFS. If you need to call the function several times, you should better use `ifs` providing the points and coefficients once instead of `IFS`. Empirical evidence shows that the IFS-estimator is better than the edf (even for very small samples) in the sup-norm metric. It is also better in the MSE sense outside of the distribution's tails if the sample quantiles are used as points.

Author(s)

S. M. Iacus

References

Iacus, S.M, La Torre, D. (2005) Approximating distribution functions by iterated function systems, *Journal of Applied Mathematics and Decision Sciences*, 1, 33-46.

See Also

ecdf

Examples

```
require(ifs)


y<-rbeta(50,.5,.1)

# uncomment if you want to test the normal distribution
# y<-sort(rnorm(50,3,1))/6


IFS.est <- IFS(y)
xx <- IFS.est$x
tt <- IFS.est$y

ss <- pbeta(xx,.5,.1)

# uncomment if you want to test the normal distribution
# ss <- pnorm(6*xx-3)

par(mfrow=c(2,1))

plot(ecdf(y),xlim=c(0,1),main="IFS estimator versus EDF")
lines(xx,ss,col="blue")
lines(xx,tt,col="red")


# calculates MSE


ww <- ecdf(y)(xx)
mean((ww-ss)^2)
mean((tt-ss)^2)

plot(xx,(ww-ss)^2,main="MSE",type="l",xlab="x",ylab="MSE(x)")
lines(xx,(tt-ss)^2,col="red")
```

---

ifs.FT *IFS estimator*

Description

Distribution function estimator based on inverse Fourier transform of ans IFSs.

Usage

```
ifs.FT(x, p, s, a, k = 2)
ifs.setup.FT(m, p, s, a, k = 2, cutoff)
ifs.pf.FT(x,b,nterms)
ifs.df.FT(x,b,nterms)
IFS.pf.FT(y, k = 2, n = 512, maps=c("quantile","wl1","wl2"))
IFS.df.FT(y, k = 2, n = 512, maps=c("quantile","wl1","wl2"))
```

Arguments

`x`	where to estimate the function
`p`	the vector of coefficients p_i
`s`	the vector of coefficients s_i in: $w_i = s_i * x + a_i$
`a`	the vector of coefficients a_i in: $w_i = s_i * x + a_i$
`m`	the vector of sample moments
`k`	number of iterations, default = 2
`y`	a vector of sample observations
`n`	the number of points in which to calculate the estimator
`maps`	type of affine maps
`b`	the Fourier coefficients
`nterms`	the number of significant Fourier coefficients after the cutoff
`cutoff`	cutoff used to determine how many Fourier coefficients are needed

Details

This estimator is intended to estimate the continuous distribution function, the charateristic function (Fourier transform) and the density function of a random variable on [0,1].

Value

The estimated value of the Fourier transform for `ifs.FT`, the estimated value of the distribution function for `ifs.pf.FT` and the estimated value of the density function for `ifs.df.FT`. A list of 'x' and 'y' coordinates plus the Fourier coefficients and the number of significant coefficients of the distribution function estimator for `IFS.pf.FT` and the density function for `IFS.df.FT`. The function `ifs.setup.FT` return a list of Fourier coefficients and the number of significant coefficients.

Note

Details of this tecnique can be found in Iacus and La Torre, 2002.

Author(s)

S. M. Iacus

References

Iacus, S.M, La Torre, D. (2005) Approximating distribution functions by iterated function systems, *Journal of Applied Mathematics and Decision Sciences*, 1, 33-46.

See Also

ecdf

Examples

```
require(ifs)


nobs <- 100
y<-rbeta(nobs,2,4)

# uncomment if you want to test the normal distribution
# y<-sort(rnorm(nobs,3,1))/6


IFS.est <- IFS(y)
xx <- IFS.est$x
tt <- IFS.est$y

ss <- pbeta(xx,2,4)

# uncomment if you want to test the normal distribution
# ss <- pnorm(6*xx-3)

par(mfrow=c(3,1))

plot(ecdf(y),xlim=c(0,1),main="IFS estimator versus EDF")
lines(xx,ss,col="blue")
lines(IFS.est,col="red")
IFS.FT.est <- IFS.pf.FT(y)
xxx <- IFS.FT.est$x
uuu <- IFS.FT.est$y
sss <- pbeta(xxx,2,4)
# uncomment if you want to test the normal distribution
# sss <- pnorm(6*xxx-3)

lines(IFS.FT.est,col="green")


# calculates MSE


ww <- ecdf(y)(xx)
mean((ww-ss)^2)
mean((tt-ss)^2)
mean((uuu-sss)^2)
```

```
plot(xx,(ww-ss)^2,main="MSE",type="l",xlab="x",ylab="MSE(x)")
lines(xx,(tt-ss)^2,col="red")
lines(xxx,(uuu-sss)^2,col="green")

plot(IFS.df.FT(y),type="l",col="green",ylim=c(0,3),main="IFS vs Kernel")
lines(density(y),col="blue")
curve(dbeta(x,2,4),0,1,add=TRUE)
# uncomment if you want to test the normal distribution
# curve(6*dnorm(x*6-3,0,1),0,1,add=TRUE)
```

IFSM *IFSM operator*

Description

IFSM operator

Usage

```
IFSM(x, cf, a, s, k = 2)
```

Arguments

x	where to approximate the function
cf	the vector of coefficients phi_i
s	the vector of coefficients s_i in: $w_i = s_i * x + a_i$
a	the vector of coefficients a_i in: $w_i = s_i * x + a_i$
k	number of iterations, default = 2

Details

This operator is intended to approximate a function on L2[0,1]. If 'u' is simulated, then the IFSM can be used to simulate a IFSM version of 'u'.

Value

The value of the approximate target function.

Author(s)

S. M. Iacus

References

Iacus, S.M, La Torre, D. (2005) IFSM representation of Brownian motion with applications to simulation, *forthcoming*.

Examples

```
require(ifs)

set.seed(123)
n <- 50
dt <- 1/n
t <- (1:n)*dt
Z <- rnorm(n)
B <- sqrt(dt)*cumsum(Z)

ifsm.w.maps() -> maps
a <- maps$a
s <- maps$s

ifsm.setQF(B, s, a) -> QF
ifsm.cf(QF$Q,QF$b,QF$L1,QF$L2,s)-> SOL
psi <- SOL$psi

t1 <- seq(0,1,length=250)
as.numeric(sapply(t1, function(x) IFSM(x,psi,a,s,k=5))) -> B.ifsm
old.mar <- par()$mar
old.mfrow <- par()$mfrow
par(mfrow=c(2,1))
par(mar=c(4,4,1,1))
plot(t1,B.ifsm,type="l",xlab="time",ylab="IFSM")
plot(t,B,col="red",type="l",xlab="time",ylab="Euler scheme")
par(mar=old.mar)
par(mfrow=old.mfrow)
```

`ifsm.cf` *Calculates the main parameters of the IFSM operator*

Description

Tool function to construct and find the solution of the minimization problem involving the quadratic form $x'Qx + b'x$. Not an optimal one. You can provide one better then this.

Usage

```
ifsm.cf(Q, b, d, l2, s, mu=1e-4)
```

Arguments

`Q`	the matrix Q of $x'Qx + b'x$
`b`	the vector b of $x'Qx + b'x$
`d`	the L1 norm of the target function
`l2`	the L2 norm of the target function
`s`	the vector s in: $w_i = s_i * x + a_i$
`mu`	tolerance

Value

A list

`cf`	the vector of the coefficients to be plugged into the IFSM
`delta`	the collage distance at the solution

References

Iacus, S.M, La Torre, D. (2005) IFSM representation of Brownian motion with applications to simulation, *forthcoming*.

See Also

IFSM

`ifsm.setQF` *Sets up the quadratic form for the IFSM*

Description

Tool function to construct the quadratic form $x'Qx+b'x+l2$ to be minimized under some constraint depending on l1. This is used to construct the IFSM operator.

Usage

```
ifsm.setQF(u, s, a)
```

Arguments

`u`	the vector of values of the target function u
`s`	the vector of coefficients s_i in: $w_i = s_i * x + a_i$
`a`	the vector of coefficients a_i in: $w_i = s_i * x + a_i$

Details

This operator is intended to approximate a function on L2[0,1]. If 'u' is simulated, then the IFSM can be used to simulate a IFSM version of 'u'.

Value

List of elements

`Q`	the matrix of the quadratic form
`b`	the matrix of the quadratic form
`L1`	the L1 norm of the target function
`L2`	the L2 norm of the target function
`M1`	the integral of the target function

Author(s)

S. M. Iacus

References

Iacus, S.M, La Torre, D. (2005) IFSM representation of Brownian motion with applications to simulation, *forthcoming*.

See Also

`IFSM`

`ifsm.w.maps` *Set up the parameters for the maps of the IFSM operator*

Description

This is called before calling `ifsm.setQF` to prepare the parameters to be passed in `ifsm.setQF`.

Usage

```
ifsm.w.maps(M=8)
```

Arguments

`M`	M is such that `sum(2^(1:M))` maps are created

Value

A list of

`a`	the vector of the coefficents 'a' in the maps
`s`	the vector of the coefficents 's' in the maps

Author(s)

S. M. Iacus

See Also

`IFSM`

ifsp.cf *Calculates the main parameters of the IFS estimators*

Description

Tool function to construct and find the solution of the minimization problem involving the quadratic form $x'Qx + b'x$. Not an optimal one. You can provide one better then this.

Usage

```
ifsp.cf(Q,b)
```

Arguments

`Q`	the matrix Q of $x'Qx + b'x$
`b`	the vector b of $x'Qx + b'x$

Value

`p`	the vector of the coefficients to be plugged into the IFS

References

Iacus, S.M, La Torre, D. (2005) Approximating distribution functions by iterated function systems, *Journal of Applied Mathematics and Decision Sciences*, 1, 33-46.

See Also

`ifs`

ifsp.setQF *Sets up the quadratic form for the IFSP*

Description

Tool function to construct the quadratic form $x'Qx + b'x$ to be minimized to construct the IFSP operator.

Usage

```
ifsp.setQF(m, s, a, n = 10)
```

Arguments

m	the vector of the sample or true moments of the target function
s	the vector of coefficients s_i in: $w_i = s_i * x + a_i$
a	the vector of coefficients a_i in: $w_i = s_i * x + a_i$
n	number of parameter to use in the IFSP operator, default = 10

Details

This operator is intended to approximate a continuous distribution function of a random variable on [0,1]. If moments are estimated on a random sample, then the IFSP operator is an estimator of the distribution function of the data.

Value

Q	the matrix of the quadratic form
b	the matrix of the quadratic form

Author(s)

S. M. Iacus

References

Iacus, S.M, La Torre, D. (2005) Approximating distribution functions by iterated function systems, *Journal of Applied Mathematics and Decision Sciences*, 1, 33-46.

See Also

ifs

---

ifsp.w.maps — *Set up the parameters for the maps of the IFSP operator*

Description

This is called before calling `ifsp.setQF` to prepare the parameters to be passed in `ifsp.setQF`.

Usage

```
ifsp.w.maps(y, maps = c("quantile","wl1","wl2"), qtl)
```

Arguments

y	the vector of the sample observations
maps	type of maps: quantile, wl1 or wl2
qtl	instead of passing the data y you can pass a vector of quantiles

Value

`m`	the vector of the empirical moments
`a`	the vector of the coefficents ‘a’ in the maps
`s`	the vector of the coefficents ‘s’ in the maps
`n`	the number of maps

Author(s)

S. M. Iacus

See Also

`ifs`

Index