Package ‘GoFKernel’

May 27, 2018

Type Package

Title Testing Goodness-of-Fit with the Kernel Density Estimator

Version 2.1-1

Date 2018-05-26

Author Jose M. Pavia

Maintainer Jose M. Pavia <Jose.M.Pavia@uv.es>

Description Tests of goodness-of-fit based on a kernel smoothing of the data.

Depends stats, KernSmooth

License GPL

NeedsCompilation no

Repository CRAN

Date/Publication 2018-05-26 22:01:54 UTC

R topics documented:

GoFKernel-package ................................................. 2
area.between .................................................. 2
density.reflected ............................................. 4
dgeometric.test ................................................ 5
fan.test ......................................................... 7
inverse .......................................................... 9
random.function ................................................. 10
risk76.1929 ...................................................... 11
support.facto .................................................. 12

Index 13
GoFKernel-package  Testing Goodness-of-fit with the Kernel Density Estimator

Description
Tests of goodness-of-fit based on kernel smoothing of the data.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>GoFKernel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depends</td>
<td>R (&gt;=2.17.3), stats, KernSmooth (&gt;=2.23-8)</td>
</tr>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>2.1-1</td>
</tr>
<tr>
<td>Date</td>
<td>2018-05-26</td>
</tr>
<tr>
<td>License</td>
<td>GPL</td>
</tr>
</tbody>
</table>

The most important functions in GoFKernel are dgeometric.test and fan.test.

Author(s)

Jose M. Pavia
Maintainer: Jose M. Pavia <Jose.M.Pavia@uv.es>

References


area.between  Area between a Density Function and a Kernel Estimate

Description
The function area.between is an (internal) function of the GoFKernel package that calculates the area, in a given interval, between a theoretical density function and an empirical kernel estimate. area.between is called by dgeometric.test of the GoFKernel package.
Usage

area.between(f, kernel.density, lower = -Inf, upper = Inf)

Arguments

- `f`: a density function.
- `kernel.density`: an empirical kernel estimate, an object of the class `density`.
- `lower`: lower limit of the support of `f`, default -Inf.
- `upper`: upper limit of the support of `f`, default Inf.

Details

area.between is called by `dgeom.test` and numerically calculates the area between the density function of the null hypothesis and the kernel density estimate of either the observed sample or a simulated sample from `f`.

Value

A number corresponding to the numerical value of the area between a density function and a kernel estimate.

Author(s)

Jose M. Pavia

See Also

density.reflected, dgeom.test, inverse random.function, support.facto and density

Examples

```r
## Unbounded example
x <- rnorm(100)
dx <- density(x)
area.between(dnorm, dx)

## Bounded example
x <- rbeta(100, 1.3, 2)
dx <- density.reflected(x, lower=0, upper=1)
area.between(dunif, dx)
```
Description

The function `density.reflected` computes kernel density estimates for univariate observations using reflection in the borders.

Usage

```r
## S3 method for class 'reflected'
density(x, lower = -Inf, upper = Inf, weights = NULL, ...)
```

Arguments

- `x`: a numeric vector of data from which the estimate is to be computed.
- `lower`: the lower limit of the interval to which x is theoretically constrained, default `-Inf`.
- `upper`: the upper limit of the interval to which x is theoretically constrained, default `Inf`.
- `weights`: numeric vector of non-negative observation weights, hence of same length as x. The default NULL is equivalent to `weights = rep(1/length(x), length(x))`.
- `...`: further `density` arguments.

Details

density.reflected is called by `dgeometry.test` and computes the density kernel estimate of a univariate random sample `x` of a random variable defined in the interval `(lower, upper)` using the default options of `density` and reflection in the borders. This avoids the density kernel estimate being underestimated in the proximity of `lower` or `upper`. For unbounded variables, `density.reflected` generates the same output as `density` with its default options.

Value

An object of the class `density` with borders correction, whose underlying structure is a list containing the following components.

- `x`: the `n` coordinates of the points where the density is estimated.
- `y`: the estimated density values. These will be non-negative.
- `bw`: the bandwidth used.
- `n`: the sample size after elimination of missing values.
- `call`: the call which produced the result.
- `data.name`: the deparsed name of the `x` argument.
- `has.na`: logical, for compatibility (always `FALSE`).

The `print` method reports `summary` values on the `x` and `y` components.
dgeomtest

Note
The function is based on density.

Author(s)
Jose M. Pavia

References

See Also
dgeomtest and density

Examples
set.seed(234)
x <- runif(2000)
dx <- density.reflected(x,0,1)

## Plot of the density estimate with and without reflection
par(mfcol=c(1,2))
plot(dx, xlim=c(-0.1,1.1), ylim=c(0,1.1))
abline(h=1, col="red")
plot(density(x), xlim=c(-0.1,1.1), ylim=c(0,1.1))
abline(h=1, col="blue")

---

dgeomtest Geometric Goodness-of-fit Test

Description
Implementation of the goodness-of-fit test based on assessing the size of the area between the null hypothesis density function and a kernel density estimate of a sample.

Usage
dgeomtest(x, fun.den, par = NULL, lower = -Inf, upper = Inf, n.sim = 101, bw=NULL)
Arguments

x  a numeric vector of data values for which the null hypothesis is tested.
fun.den  an actual density distribution function, such as dnorm. Only continuous densities are valid.
par  list of (additional) parameters of the density function under the null hypothesis, default NULL.
lower  lower end point of the support of the random variable defined by fun.den, default -Inf.
upper  upper end point of the support of the random variable defined by fun.den, default -Inf.
n.sim  number of iterations performed to calculate the p.value of the test, default 101.
bw  a number indicating the bandwidth to be used in the empirical kernel estimate of the data, default NULL. In its default option, the bandwidth varies in each simulated dataset and is the one estimated by default by density with a Gaussian kernel.

Details
dgeometric.test uses numerical integration and Monte Carlo simulation to implement the test based on assessing the extend of the area between a null hypothesis density function and a density kernel estimation. It works as follows. After computing by numerical integration the area between the density function under the null hypothesis and its sample empirical kernel estimate obtained using density.reflected, the p.value of the test is obtained by simulation as follows: (i) drawing n.sim samples from fun.den with the same size length(x) of our actual sample x; (ii) estimating the kernel density function for each of these new samples; (iii) computing the area between the theoretical density and each of the estimates obtained in (ii); and, (iv) calculating the p-value as the proportion of times the sample n.sim areas computed in (iii) exceed the value of the area computed from the observed sample.

Value

The output is an object of the class htest exactly like for the Kolmogorov-Smirnov test, ks.test. A list containing the following components:

statistic  the value of the test statistic.
p.value  the p-value of the test.
method  the character string "Geometric test".
data.name  a character string giving the name of the data.

Note
dgeometric.test calls density.reflected and area.between (and, in some circumstances, also inverse, random.function and support.facto), which are (internal) functions of the package GoFKernel.
fan.test

Author(s)

Jose M. Pavia

References


See Also

area.between.density.reflected, inverse random.function, support.facto and fan.test.

Examples

set.seed(12)
x <- rlnorm(50, meanlog=1, sdlog=1)
## test if x follows a Gamma distribution with shape .6 and rate .1
dgeometric.test(x, dgamma, par=list(shape=0.6, rate=0.1), lower=0, upper=Inf, n.sim=100)

f0 <- function(x) ifelse(x>=0 & x<=1, 2-2*x, 0)
## test if risk76.1929 follows the distribution characterized by f0
dgeometric.test(risk76.1929, f0, lower=0, upper=1, n.sim=21)

fan.test  Univariate implementation of the test of Fan (1994) in the form proposed by Li and Racine (2007).

Description

Given a sample of a continuous univariate random variable and a density function fun.den with support in the interval (lower, upper), fan.test considers the test whose null hypothesis is that the sample has fun.den as density function and the test statistic and the corresponding p-value of the test based on the integral of the squared difference between the null hypothesis density function and a kernel smoothing approximation. To properly run, the KernSmooth package needs to be installed, as in its default option it depends on the dpik function to estimate the bandwidth.

Usage

fan.test(x, fun.den, par = NULL, lower = -Inf, upper = Inf, kernel = "normal", bw=NULL)
Arguments

- **x**: a numeric vector of data values for which the null hypothesis is tested.
- **fun.den**: an actual density distribution function, such as `dnorm`. Only continuous densities are valid.
- **par**: list of (additional) parameters of the density function under the null hypothesis, default NULL.
- **lower**: lower end point of the support of the random variable defined by `fun.den`, default -Inf.
- **upper**: upper end point of the support of the random variable defined by `fun.den`, default -Inf.
- **kernel**: a character string with the kernel to be used, either "normal" (a N(0,1) density), "box" (a uniform in -1 to 1) or "epanech" (a Epanechnikov quadratic kernel), default "normal".
- **bw**: a number indicating the bandwidth to be used in the empirical kernel estimate of the data, default NULL. In its default option, the bandwidth is estimated using the `dpik` function included in the package `KernSmooth`.

Details

The Fan’s test is based on a normal approximation of the integral of the squared difference between the null hypothesis density function and a kernel smoothing approximation. In Li and Racine’s form it is obtained as the aggregation of (i) a sampling component, (ii) the integrate of the square of the kernel convolution of the density null function and (iii) the sum of the convolution of the density in the sampled values, see Li and Racine (2007, pp.380-1) for details.

Value

The output is an object of the class `htest` exactly like for the Kolmogorov-Smirnov test, `ks.test`. A list containing the following components:

- **statistic**: the value of the test statistic.
- **p.value**: the p-value of the test.
- **method**: the character string "Geometric test".
- **data.name**: a character string giving the name of the data.

Warning

`fan.test` calls the `dpik` function of `KernSmooth`

Note

To properly run the function requires the package `KernSmooth` to be installed to estimate the bandwidth.

Author(s)

Jose M. Pavia
References


See Also
dgeometric.test, integrate and dpik.

Examples

fan.test(runif(100), dunif, lower=0, upper=1)

f0 <- function(x) ifelse(x>=0 & x<=1, 2-2*x, 0)
## testing if risk76.1929 follows the distribution characterized by f0
fan.test(risk76.1929, f0, lower=0, upper=1, kernel="epanech")

### Inverse CDF Function

#### Description

Function to calculate the inverse function of a cumulative distribution function.

#### Usage

inverse(f, lower = -Inf, upper = Inf)

#### Arguments

- **f**: a cdf function for which we want to obtain its inverse.
- **lower**: the lower limit of f domain (support of the random variable), default -Inf.
- **upper**: the upper limit of f domain (support of the random variable), default Inf.

#### Details

inverse is called by random.function and calculates the inverse of a given function f. inverse has been specifically designed to compute the inverse of the cumulative distribution function of an absolutely continuous random variable, therefore it assumes there is only a root for each value in the interval (0,1) between f(lower) and f(upper). It is for internal use in dgeometric.test.

#### Value

A function, the inverse function of a cumulative distribution function f.
Note

This function uses either `optim` with default options `method="L-BFGS-B"` or `uniroot` to derive the inverse function.

The upper endpoint must be strictly larger than the lower endpoint.

Author(s)

Jose M. Pavia

References

See the references in `optim` and `uniroot`.

See Also

dgeometric.test, integrate, optim, random.function, support.facto and unifroot.

Examples

```r
f <- function(x) pbeta(x, shape1=2, shape2=3)
f.inv <- inverse(f,lower=0,upper=1)
f.inv(.2)
```

random.function Random Draw Generator

Description

This function generates random draws of a continuous random variable given either its density or its cumulative distribution function.

Usage

```r
random.function(n = 1, f, lower = -Inf, upper = Inf, kind = "density")
```

Arguments

- `n` number of draws, default 1.
- `f` either a density (default) or cumulative distribution function of the random variable.
- `lower` lower limit of the support of the random variable, default -Inf.
- `upper` upper limit of the support of the random variable, default Inf.
- `kind` character string with the function used to identify the distribution, either "density" (default) or "cumulative", as alternative.
Details

random.function uses the method of the inverse of the cdf to generate random draws from \( f \).

Value

A vector of length \( n \) with \( n \) draws from a random variable with density (or cumulative distribution) function given by \( f \).

Note

random.function is called by dgeometric.test when the corresponding r-function (random generator of \( f \)) is not available in the environment. random.function generates random samples from the null hypothesis density function specified in dgeometric.test.

Author(s)

Jose M. Pavia

See Also

dgeometric.test, integrate, inverse and support.facto.

Examples

```r
f0 <- function(x) ifelse(x>=0 & x<=1, 2-2*x, 0)
random.function(10, f0, lower=0, upper=1, kind="density")
```

---

**risk76.1929**

*Immigrants Exposed to Risk of Death*

Description

Vector containing the time exposed to risk of death with 76 years during 2006 for the 2006 registered Spanish immigrants born in 1929.

Usage

```r
data(risk76.1929)
```

Format

The format is: num [1:362] 0.94 0.885 0.863 0.852 0.797 ...

Note

Under the null hypotheses of uniform distribution of date of birth and date of migration, this time exposed to risk is distributed as \( f(x)=2-2x \text{ for } 0<x<1 \).
Source

Own elaboration from data available in www.ine.es

Examples

plot(density.reflected(risk76.1929, 0, 1))

---

support.facto  "De Facto" Support

Description

support.facto computes the de facto numerical limits of a density function with theoretical infinite support. This function is an (internal) function of the GoFKernel package.

Usage

support.facto(f, lower = -Inf, upper = Inf)

Arguments

- `f`: a density function.
- `lower`: theoretical lower limit of the support of the random variable, default -Inf.
- `upper`: theoretical upper limit of the support of the random variable, default, Inf.

Details

support.facto requires that the two first ordinary moments of `f` exist; otherwise, support.facto returns the introduced limits.

Value

A two components vector with the de facto lower and upper limits of `f`.

Author(s)

Jose M. Pavia

See Also

area.between, dgeometric.test, inverse, random.function and fan.test.

Examples

support.facto(dnorm)
Index

* Topic `\textasciitilde kwd1`
  inverse, 9
* Topic `\textasciitilde kwd2`
  inverse, 9
* Topic `datasets`
  risk76.1929, 11
* Topic `density`
  area.between, 2
  density.reflected, 4
* Topic `htest`
  dgeometric.test, 5
  fan.test, 7
* Topic `package`
  GoFKernel-package, 2
  area.between, 2, 6, 7, 12
  density, 3, 5, 6
  density (density.reflected), 4
  density.reflected, 3, 4, 6, 7
  dgeometric.test, 2, 3, 5, 5, 9–12
  dpik, 8, 9
  fan.test, 2, 7, 7, 12
  GoFKernel (GoFKernel-package), 2
  GoFKernel-package, 2
  integrate, 9–11
  inverse, 3, 6, 7, 9, 11, 12
  ks.test, 6, 8
  optim, 10
  random.function, 3, 6, 7, 10, 10, 12
  risk76.1929, 11
  summary, 4
  support.facto, 3, 6, 7, 10, 11, 12
  uniroot, 10