Package 'DELTD'

February 9, 2021

Type Package

Title Kernel Density Estimation using Lifetime Distributions

Version 2.6.7

Author Javaria Ahmad Khan, Atif Akbar.

Maintainer Javaria Ahmad Khan <jakhan0@yahoo.com>

Description A collection of asymmetrical kernels belong to lifetime distributions for kernel density estimation is presented.
Mean Squared Errors (MSE) are calculated for estimated curves. For this purpose, R functions allow the distribution to be Gamma, Exponential or Weibull.
For details see Chen (2000a, b), Jin and Kawczak (2003) and Salha et al. (2014) <doi:10.12988/pms.2014.4616>.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

URL https://CRAN.R-project.org/package=DELTD

NeedsCompilation no

Repository CRAN

Date/Publication 2021-02-09 06:30:02 UTC

R topics documented:

DELTD-package *DELTD*

Description

A collection of asymmetrical kernels belong to lifetime distributions for kernel density estimation is presented. i.e. `plot.BS`, `plot.Beta`, `plot.Erlang`, `plot.Gamma` and `plot.LogN`. Estimated values can also observed by using `Beta`, `BS`, `Erlang`, `Gamma` and `LogN`. For calculating mean squared error by using different kernel functions are `mse` can be used.

Details

Kernel Density Estimation using Lifetime Distributions

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

- Jin, X.; Kawczak, J. 2003. Birnbaum-Saunders & Lognormal kernel estimators for modeling durations in high frequency financial data. *Annals of Economics and Finance* **4**, 103–124.
- Salha, R. B.; Ahmed, E. S.; Alhoubi, I. M. 2014. Hazard rate function estimation using Erlang Kernel. *Pure Mathematical Sciences* **3** (4), 141–152.
- Chen, S. X. 2000. Probability density function estimation using Gamma kernels. *Annals of the Institute of Statistical Mathematics* **52** (3), 471-480.
- Chen, S. X. 2000. Beta kernel smothers for regression curves. *Statistica Sinica* **10**, 73-91.

See Also

Useful links:

- https://CRAN.R-project.org/package=DELTD

Beta *Estimate Density Values by Beta kernel*

Description

This function provide the estimated Kernel density values by using Beta Kernel. The Beta kernel is developed by Chen (2000) by using Beta distribution of first kind. He was first to introduce asymetrical kernels to control boundary Bias. Beta Kernel is

$$K_{Beta(\frac{x}{h}+1,\frac{1-x}{h}+1)}(y) = \frac{y^{\frac{x}{h}}(1-y)^{\frac{1-x}{b}}}{B\{\frac{x}{h}+1,\frac{(1-x)}{h}+1\}}$$

Usage

```
Beta(x = NULL, y, k = NULL, h = NULL)
```

Arguments

x	scheme for generating grid points
y	a vector of positive values
k	number of grid points
h	the bandwidth

Details

In this function, choice of bandwidth, number of grid points and scheme that how these grid points are generated are user based. If any parameter(s) is missing then function used default parameters. But at least x or k should be specified otherwise NA will be produced. If x is missing then function will generate k grid points by using uniform distribution. Similarly, if k is missing then function consider it same to length of main vector y. In case if h is missing then function used normal scale rule bandwidth for non-normal data and described in Silverman (1986). This function can be onlt used if data is between (0, 1). Similarly, x should be also lies between (0, 1).

Value

x	grid points
y	estimated values of density

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Chen, S. X. 2000. Beta kernel smothers for regression curves. *Statistica Sinica* **10**, 73-91. Silverman, B. W. 1986. *Density Estimation*. Chapman & Hall/ CRC, London.

See Also

For further kernels see BS, Gamma, Erlang and LogN. To plot its density see plot.Beta and to calculate MSE mse.

Examples

```
## Data: Simulated or real data can be used
## Number of grid points "k" should be at least equal to the data size.
## If user defines the generating scheme of grid points then length
## of grid points should be equal or greater than "k", Otherwise NA will be produced.
y <- runif(50)
xx <- sample(0.00001:900, 500, replace = FALSE)/1000
h <- 0.9
Beta(x = xx, y = y, k = 500, h = h)

## If scheme for generating grid points is unknown
y <- runif(500)
h <- 0.9
Beta(x = xx, y = y, k = 500, h = h)

## Not run:
## If user do not mention the number of grid points
y <- runif(1000)
xx <- seq(0.001, 1000, length = 2000)

## any bandwidth can be used
require(kedd)
h <- h.bcv(y) ## Biased cross validation
Beta(x = xx, y = y, h = h)

## End(Not run)

## Not run:
##if both generating scheme and number of grid points are missing then function generate NA
y <- runif(1000)
band = 0.8
Beta(y = y, h = band)

## End(Not run)

## if bandwidth is missing
y <- runif(100)
xx <- seq(0.001, 100, length = 300)
Beta(x = xx, y = y, k = 200)
```

Description

This function calculates the estimated Values by using Birnbaum-Saunders Kernel. The Birnbaum-Saunders kernel is developed by Jin and Kawczak (2003). They claimed that performance of their developed kernel is better near the boundary points in terms of boundary reduction.

$$K_{BS(h^{\frac{1}{2}},x)}(y) = \frac{1}{2\sqrt{2}\pi h}\left(\sqrt{\frac{1}{xy}} + \sqrt{\frac{x}{y^3}}\right) exp\left(-\frac{1}{2h}\left(\frac{y}{x} - 2 + \frac{x}{y}\right)\right)$$

Usage

```
BS(x = NULL, y, k = NULL, h = NULL)
```

Arguments

x	scheme for generating grid points
y	a vector of positive values.
k	grid points
h	the bandwidth

Details

In this function, choice of bandwidth, number of grid points and scheme that how these grid points are generated are user based. If any parameter(s) is missing then function used default parameters. But at least x or k should be specified otherwise NA will be produced. If x is missing then function will generate k grid points between minimum and maximum values of vector. Similarly, if k is missing then function consider it same to length of main vector y. In case if h is missing then function used normal scale rule bandwidth for non-normal data and described in Silverman (1986).

Value

x	grid points
y	estimated values of density

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Jin, X.; Kawczak, J. 2003. Birnbaum-Saunders & Lognormal kernel estimators for modeling durations in high frequency financial data. *Annals of Economics and Finance* **4**, 103-124.

See Also

For further kernels see Beta, Erlang, Gamma and LogN. To plot the density by using BS kernel plot.BS and to calculate MSE by mse.

Examples

```
## Data: Simulated or real data can be used
## Number of grid points "k" should be at least equal to the data size.
## If user defines the generating scheme of grid points then length
## of grid points should be equal or greater than "k", Otherwise NA will be produced.
alpha = 10
theta = 15 / 60

y <- rgamma(n = 1000, shape = alpha, scale = theta)
xx <- seq(min(y) + 0.05, max(y), length =200)
h <- 1.1
den <- BS(x = xx, y = y, k = 200, h = h)

##If scheme for generating grid points is unknown
y <- rgamma(n = 1000, shape = alpha, scale = theta)
h <- 3
BS(y = y, k = 90, h = h)

## Not run:
##If user do not mention the number of grid points
y <- rgamma(n = 1000, shape = alpha, scale = theta)
xx <- seq(0.001, 1000, length = 1000)

#any bandwidth can be used
require(KernSmooth)
h <- dpik(y)     #Direct Plug-In Bandwidth
BS(x = xx, y = y, h = h)

## End(Not run)

## Not run:
#if both generating scheme and number of grid points are missing then function generate NA
y <- rgamma(n = 1000, shape = alpha, scale = theta)
band = 3
BS(y = y, h = band)

## End(Not run)

#if bandwidth is missing
y <- rgamma(n = 1000, shape = alpha, scale = theta)
xx <- seq(0.001, 100, length = 1000)
BS(x = xx, y = y, k = 900)
```

`Erlang` *Estimate Density Values by Erlang kernel*

Description

This function provide the estimated values for density by using Erlang Kernel. Erlang kernel is developed by Salha et al. (2014). They developed this asymmetrical kernal with its hazard function

and also proved its asymtotic normality.

$$K_{E(x,\frac{1}{h})}(y) = \frac{1}{\Gamma(1+\frac{1}{h})}\left[\frac{1}{x}(1+\frac{1}{h})\right]^{\frac{h+1}{h}} y^{\frac{1}{h}} exp\left(-\frac{y}{x}(1+\frac{1}{h})\right)$$

Usage

```
Erlang(x = NULL, y, k = NULL, h = NULL)
```

Arguments

x	scheme for generating grid points
y	a vector of positive values.
k	grid points.
h	the bandwidth

Details

see the details in the BS.

Value

x	grid points
y	estimated values of density

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Salha, R. B.; Ahmed, E. S.; Alhoubi, I. M. 2014. Hazard rate function estimation using Erlang Kernel. *Pure Mathematical Sciences* **3** (4), 141-152.

See Also

For further MSE by using other kernels see Beta, BS, Gamma and LogN. For plotting these estimated values plot.Erlang and for calculating MSE use mse.

Examples

```
## Data: Simulated or real data can be used
## Number of grid points "k" should be at least equal to the data size.
## If user defines the generating scheme of grid points then length
## of grid points should be equal or greater than "k", Otherwise NA will be produced.
y <- rlnorm(100, meanlog = 0, sdlog = 1)
xx <- seq(min(y) + 0.05, max(y), length = 500)
h <-2
den <- Erlang(x = xx, y = y, k = 200, h = h)
```

```
##If scheme for generating grid points is unknown
y <- rlnorm(1000, meanlog = 0, sdlog = 1)
h <- 3
Erlang(y = y, k = 90, h = h)

## Not run:
##If user do not mention the number of grid points
 y <- rlnorm(100, meanlog = 0, sdlog = 1)
xx <- seq(0.001, 1000, length = 1000)

#any bandwidth can be used
require(kedd)
h <- h.ucv(y)     #Unbaised cross validation bandwidth
Erlang(x = xx, y = y, h = h)

## End(Not run)

## Not run:
#if generating scheme and number of grid points are missing then function generate NA
y <- rlnorm(100, meanlog = 0, sdlog = 1)
band = 3
Erlang(y = y, h = band)

## End(Not run)

#if bandwidth is missing
 y <- rlnorm(100, meanlog = 0, sdlog = 1)
xx <- seq(0.001, 100, length = 100)
Erlang(x = xx, y = y, k = 90)
```

Gamma *Estimate Density Values by Gamma kernel*

Description

This function provide the estimated Kernel density values by using Gamma Kernel.The Gamma kernel is developed by Chen (2000). He was first to introduce asymetrical kernels to control boundary Bias. Gamma Kernel is

$$K_{Gam1(\frac{x}{h}+1,h)}(y) = \frac{y^{\frac{x}{h}} exp(-\frac{y}{h})}{\Gamma(\frac{x}{h}+1)h^{\frac{x}{h}+1}}$$

Usage

```
Gamma(x = NULL, y, k = NULL, h = NULL)
```

Arguments

x	scheme for generating grid points
y	a vector of positive values

k	number of grid points
h	the bandwidth

Details

see the details in the BS.

Value

x	grid points
y	estimated values of density

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

- Chen, S. X. 2000. Probability density function estimation using Gamma kernels. *Annals of the Institute of Statistical Mathematics* **52** (3), 471-480.
- Silverman, B. W. 1986. *Density Estimation*. Chapman & Hall/ CRC, London.

See Also

For further kernels see Beta, BS, Erlang and LogN. To plot its density see plot.Gamma and to calculate MSE mse.

Examples

```
##Number of grid points "k" should be at least equal to the data size.
###If user defines the generating scheme of grid points then length
####of grid points should be equal or greater than "k". Otherwise NA will be produced.
y <- rexp(100, 1)
xx <- seq(min(y) + 0.05, max(y), length = 500)
h <- 2
den <- Gamma(x = xx, y = y, k = 200, h = h)

##If scheme for generating grid points is unknown
y <- rexp(200, 1)
h <- 3
Gamma(y = y, k = 90, h = h)

## Not run:
##If user do not mention the number of grid points
y <- rexp(1000, 1)
xx <- seq(0.001, 1000, length = 1000)

#any bandwidth can be used
require(KernSmooth)
h <- dpik(y)
Gamma(x = xx, y = y, h = h)
```

```
## End(Not run)

## Not run:
#if generating scheme and number of grid points are missing then function generate NA
y <- rexp(1000, 1)
band = 3
Gamma(y = y, h = band)

## End(Not run)

#if bandwidth is missing
y <- rexp(100,1)
xx <- seq(0.001, max(y), length = 100)
Gamma(x = xx, y = y, k = 90)
```

LogN *Estimate Density Values by Lognormal kernel*

Description

The LogN estimate Values of density by using Lognormal Kernel.The Lognomal kernel is developed by Jin and Kawczak (2003). For this too, they claimed that performance of their developed kernel is better near the boundary points in terms of boundary reduction. Lognormal Kernel is

$$K_{LN(\ln(x),4\ln(1+h))} = \frac{1}{\sqrt{(8\pi\ln(1+h))y)}} exp\left[-\frac{(\ln(y)-\ln(x))^2}{(8\ln(1+h))}\right]$$

Usage

```
LogN(x = NULL, y, k = NULL, h = NULL)
```

Arguments

x	scheme for generating grid points
y	a vector of positive values.
k	grid points.
h	the bandwidth

Details

see the details in the BS.

Value

x	grid points
y	estimated values of density

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Jin, X.; Kawczak, J. 2003. Birnbaum-Saunders & Lognormal kernel estimators for modeling durations in high frequency financial data. *Annals of Economics and Finance* **4**, 103-124.

See Also

For further kernels see Beta, BS, Erlang and Gamma. To plot its density see plot.LogN and to calculate MSE use mse.

Examples

```
## Data: Simulated or real data can be used
## Number of grid points "k" should be at least equal to the data size.
## If user defines the generating scheme of grid points then length
## of grid points should be equal or greater than "k", Otherwise NA will be produced.
y <- rweibull(350, 1)
xx <- seq(0.001, max(y), length = 500)
h <- 2
den <- LogN(x = xx, y = y, k = 200, h = h)

##If scheme for generating grid points is unknown
n <- 1000
y <- abs(rlogis(n, location = 0, scale = 1))
h <- 3
LogN(y = y, k = 90, h = h)

## Not run:
##If user do not mention the number of grid points
y <- rweibull(350, 1)
xx <- seq(0.00001, max(y), 500)

#any bandwidth can be used
require(ks)
h <- hscv(y)   #Smooth cross validation bandwidth
LogN(x = xx, y = y, h = h)

## End(Not run)

## Not run:
#if both scheme and number of grid points are missing then function generate NA
n <- 1000
y <- abs(rlogis(n, location = 0, scale = 1))
band = 3
LogN(y = y, h = band)

## End(Not run)

#if bandwidth is missing
```

```
y <- rweibull(350, 1)
xx <- seq(0.001, 100, length = 500)
LogN(x = xx, y = y, k = 90)
```

`mse` *Calculate Mean Squared Error(MSE) by using different Kernels*

Description

This function calculates the mean squared error (MSE) by using user specified kernel. But distribution of vector should be Exponential, Gamma or Weibull. Any other choice of distribution will result `NaN`.

Usage

```
mse(kernel, type)
```

Arguments

`kernel`	type of kernel which is to be used
`type`	mention distribution of vector. If exponential distribution then use `"Exp"`. If use gamma distribution then use `"Gamma"`.If Weibull distribution then use `"Weibull"`.

Value

Mean Squared Error (MSE)

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

- Jin, X.; Kawczak, J. 2003. Birnbaum-Saunders & Lognormal kernel estimators for modeling durations in high frequency financial data. *Annals of Economics and Finance* **4**, 103-124.
- Salha, R. B.; Ahmed, E. S.; Alhoubi, I. M. 2014. Hazard rate function estimation using Erlang Kernel. *Pure Mathematical Sciences* **3** (4), 141-152.
- Chen, S. X. 2000. Probability density function estimation using Gamma kernels. *Annals of the Institute of Statistical Mathematics* **52** (3), 471-480.
- Chen, S. X. 2000. Beta kernel smothers for regression curves. *Statistica Sinica* **10**, 73-91.

Examples

```
y <- rexp(100, 1)
xx <- seq(min(y) + 0.05, max(y), length = 500)
h <- 2
gr <- Gamma(x = xx, y = y, k = 200, h = h)
mse(kernel = gr, type = "Exp")
## if distribution is other than mentioned \code{type} is used then NaN will be produced.
## Not run:
mse(kernel = gr, type ="Beta")

## End(Not run)
```

`plot.Beta` *Density Plot by Beta kernel*

Description

Plot density by using Beta Kernel.

Usage

```
## S3 method for class 'Beta'
plot(x, ...)
```

Arguments

x	an object of class "Beta"
...	Not presently used in this implementation

Value

nothing

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Chen, S. X. 2000. Beta kernel smothers for regression curves. *Statistica Sinica* **10**, 73-91.

See Also

For further kernels see `plot.BS`, `plot.Erlang`, `plot.Gamma` and `plot.LogN`. To calculate its estimated values see `Beta` and for MSE see `mse`.

Examples

```
y <- runif(100)
h <- 0.5
xx <- sample(0.00001:900, 50, replace = FALSE)/1000
den <- Beta(x = xx, y = y, k = 50, h = h)
plot(den, type = "p")
##other details can also be added
y <- runif(100)
h <- 0.7
xx <- sample(0.00001:900, 50, replace = FALSE)/1000
den <- Beta(x = xx, y = y, k = 50, h = h)
plot(den, type = "l", ylab = "Density Function", lty = 1, xlab = "Time")
```

---

plot.BS *Density Plot by Birnbaum-Saunders kernel*

Description

Plot Kernel density by using Birnbaum-Saunders Kernel.

Usage

```
## S3 method for class 'BS'
plot(x, ...)
```

Arguments

x	An object of class "BS"
...	Not presently used in this implementation

Value

Nothing

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Jin, X.; Kawczak, J. 2003. Birnbaum-Saunders & Lognormal kernel estimators for modeling durations in high frequency financial data. *Annals of Economics and Finance* **4**, 103-124.

See Also

For further kernels see plot.Beta, plot.Erlang, plot.Gamma and plot.LogN. For estimated values BS and for MSE mse.

Examples

```
alpha = 10
theta = 15 / 60
y <- rgamma(n = 1000, shape = alpha, scale = theta)
h <- 1.5
xx <- seq(min(y) + 0.05, max(y), length = 200)
den <- BS(x = xx, y = y, k = 200, h = h)
plot(den, type = "l")

##other details can also be added
y <- rgamma(n = 1000, shape = alpha, scale = theta)
h <- 0.79 * IQR(y) * length(y) ^ (-1/5)  #Normal Scale Rule Bandwidth
gr <- BS(x = xx, y = y, k = 200, h = h)
plot(gr, type = "s", ylab = "Density Function", lty = 1, xlab = "Time")

## To add true density along with estimated
d1 <- density(y, bw = h)
lines(d1, type = "p", col = "red")
legend("topright", c("Real Density", "Density by Birnbaum-Saunders Kernel"),
col=c("red", "black"), lty = c(1,2))
```

`plot.Erlang` *Density Plot by Erlang kernel*

Description

Plot Kernel density by using Erlang Kernel.

Usage

```
## S3 method for class 'Erlang'
plot(x, ...)
```

Arguments

x	An object of class "Erlang"
...	Not presently used in this implementation

Value

Nothing

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Salha, R. B.; Ahmed, E. S.; Alhoubi, I. M. 2014. Hazard rate function estimation using Erlang Kernel. *Pure Mathematical Sciences* **3** (4), 141-152.

See Also

For further MSE by using other kernels see `plot.Beta`, `plot.BS`, `plot.Gamma` and `plot.LogN`. For estimated values `Erlang` and for calculating MSE see `mse`.

Examples

```
y <- rlnorm(100, meanlog = 0, sdlog = 1)
h <- 1.5
xx <- seq(min(y) + 0.05, max(y), length = 200)
den <- Erlang(x = xx, y = y, k = 200, h = h)
plot(den, type = "l")

##other details can also be added
y <- rlnorm(100, meanlog = 0, sdlog = 1)
grid <- seq(min(y) + 0.05, max(y), length = 200)
h <- 0.79 * IQR(y) * length(y) ^ (-1/5)
gr <- Erlang(x = grid, y = y, k = 200, h = h)
plot(gr, type = "s", ylab = "Density Function", lty = 1, xlab = "Time")

## To add true density along with estimated
d1 <- density(y, bw = h)
lines(d1, type = "p", col = "red")
legend("topright", c("Real Density", "Density by Erlang Kernel"),
col=c("red", "black"), lty=c(1,2))
```

---

plot.Gamma *Density Plot by Gamma kernel*

Description

Plot density by using Gamma Kernel.

Usage

```
## S3 method for class 'Gamma'
plot(x, ...)
```

Arguments

x	an object of class "Gamma"
...	Not presently used in this implementation

Value

nothing

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Chen, S. X. 2000. Probability density function estimation using Gamma kernels. *Annals of the Institute of Statistical Mathematics* **52** (3), 471-480.

See Also

For further kernels see plot.Beta, plot.BS, plot.Erlang and plot.LogN. To calculate its estimated values see Gamma and for MSE mse.

Examples

```
y <- rexp(100, 1)
h <- 1.5
xx <- seq(min(y) + 0.05, max(y), length =200)
den <- Gamma(x=xx, y=y, k=200, h=h)
plot(den, type = "l")

##other details can also be added
y <- rexp(100, 2)
h <- 0.79 * IQR(y) * length(y) ^ (-1/5)
gr <- Gamma(x=xx, y=y, k=200, h=h)
plot(gr, type = "s", ylab = "Density Function", lty = 1, xlab = "Time")

## To add true density along with estimated
d1 <- density(y, bw=h)
lines(d1, type="p", col="red")
legend("topright", c("Real Density", "Density by Gamma Kernel"),
col=c("red", "black"), lty=c(1,2))
```

---

plot.LogN *Density Plot by Lognormal kernel*

Description

Plot Kernel density by using Lognormal Kernel.

Usage

```
## S3 method for class 'LogN'
plot(x, ...)
```

Arguments

x	An object of class "LogN"
...	Not presently used in this implementation

Value

Nothing

Author(s)

Javaria Ahmad Khan, Atif Akbar.

References

Jin, X.; Kawczak, J. 2003. Birnbaum-Saunders & Lognormal kernel estimators for modeling durations in high frequency financial data. *Annals of Economics and Finance* **4**, 103-124.

See Also

For further kernels see plot.Beta, plot.BS, plot.Erlang and plot.Gamma. To calculate MSE use mse and for estimated values for density estimation see LogN.

Examples

```
n <- 1000
y <- abs(rlogis(n, location = 0, scale = 1))
xx <- seq(min(y) + 0.05, max(y), length =90)
h <- 0.00003
den <- LogN(x = xx, y = y, k = 90, h = h)
plot(den, type = "l")

##other details can also be added
y <- abs(rlogis(n, location = 0, scale = 1))
h <- 3
gr <- LogN(x = xx, y = y, k = 90, h = h)
plot(gr, type = "s", ylab = "Density Function", lty = 1, xlab = "Time")

## To add true density along with estimated
d1 <- density(y, bw = h)
lines(d1, type = "p", col = "green")
legend("topleft", c("Real Density", "Density by Lognormal Kernel"),
col = c("green", "black"), lty = c(1,2))
```

Index