Package ‘WPKDE’

October 12, 2022

Type Package

Title Weighted Piecewise Kernel Density Estimation

Version 0.1

Date 2017-02-04

Author Kunyu Ye, Siyao Wang, Xudong Liu, Tianwei Yu

Maintainer Kunyu Ye <kunyuye@163.com>

Suggests mvtnorm(>= 1.0-0)

Description Weighted Piecewise Kernel Density Estimation for large data.

License GPL

LazyData TRUE

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-03-02 14:29:46

R topics documented:

findPeak 1
kdeC 3
plot2d 4

Index **6**

`findPeak` *find peaks*

Description

using the result of `kdeC` to find peaks

Usage

```
findPeak(estimate,filter)
```

Arguments

`estimate`	matrix returned by the `kdeC` function
`filter`	a num value, filter the result less than argument value `filter` and set 0 as default

Details

the function `findPeak` can be executed after `kdeC` to find peaks

Value

The returned value is a matrix corresponding to input argument `estimate`, the value in the returned matrix larger than 0 means it is a peak

Author(s)

Kunyu Ye

Examples

```
data.gen<-function(n.peaks=100, N=1e5, max.var=0.001, max.corr=0.5)
{
  library(mvtnorm)

  dat<-matrix(0, nrow=N, ncol=2)
  all.m<-c(NA,NA)

  for(i in 1:n.peaks)
  {
    this.m<-runif(2)
    this.var<-runif(2, min=0.1*max.var, max=max.var)
   this.cov<-runif(1, min=-1*max.corr, max=max.corr) * sqrt(this.var[1])* sqrt(this.var[2])
    this.s<-matrix(c(this.var[1], this.cov, this.cov, this.var[2]),ncol=2)

   dat[((i-1)*N/n.peaks+1):(i*N/n.peaks),]<-rmvnorm(N/n.peaks, mean=this.m, sigma=this.s)
    all.m<-rbind(all.m, this.m)
  }

  all.m[,1]<-(all.m[,1]-min(dat[,1]))/diff(range(dat[,1]))
  all.m[,2]<-(all.m[,2]-min(dat[,2]))/diff(range(dat[,2]))
  dat[,1]<-(dat[,1]-min(dat[,1]))/diff(range(dat[,1]))
  dat[,2]<-(dat[,2]-min(dat[,2]))/diff(range(dat[,2]))

  all.m<-all.m[-1,]
  return(list(dat=dat,m=all.m))
}


r<-data.gen(n.peaks=100, N=1e5, max.var=0.001, max.corr=0.5)

k1<-kdeC(r$dat, H=c(0.005,0.005),  gridsize = c(501,501), cutNum=c(1,1))
```

```
matPeaks<-findPeak(estimate=k1$estimate,filter=0)
```

`kdeC` *weighted kernel density estimation*

Description

fast weighted kernel density estimation for 2-dimension and calling C function to implement the calculation procedure

Usage

```
kdeC(x,H,gridsize,cutNum,w)
```

Arguments

`x`	data points in the format n*2 matrix
`H`	bandwidth, a vector containing 2 num values and set c(0.01,0.01) as default
`gridsize`	number of points for each direction, a vector containing 2 int values and set c(200,50) as default
`cutNum`	number of pieces to be cutted for each direction, a vector containing 2 int values and set c(1,1) as default
`w`	weight, a vector corresponding to parameter `x` and set rep(1,length(x)/2) as default

Details

The function `kdeC` is only suitable for 2-dimension data. The advantage of `kdeC` is that it can get the result quickly because the calculation procedure is implemented in C code.

Value

the returned value is a list

`estimate`	density estimate at points `evalpointsX` and `evalpointsY`
`evalpointsX`	points at which the `estimate` is evaluated at x-axis direction
`evalpointsY`	points at which the `estimate` is evaluated at y-axis direction

Author(s)

Kunyu Ye

References

R package 'ks'

Examples

```
data.gen<-function(n.peaks=100, N=1e5, max.var=0.001, max.corr=0.5)
{
  library(mvtnorm)

  dat<-matrix(0, nrow=N, ncol=2)
  all.m<-c(NA,NA)

  for(i in 1:n.peaks)
  {
    this.m<-runif(2)
    this.var<-runif(2, min=0.1*max.var, max=max.var)
   this.cov<-runif(1, min=-1*max.corr, max=max.corr) * sqrt(this.var[1])* sqrt(this.var[2])
    this.s<-matrix(c(this.var[1], this.cov, this.cov, this.var[2]),ncol=2)

   dat[((i-1)*N/n.peaks+1):(i*N/n.peaks),]<-rmvnorm(N/n.peaks, mean=this.m, sigma=this.s)
    all.m<-rbind(all.m, this.m)
  }

  all.m[,1]<-(all.m[,1]-min(dat[,1]))/diff(range(dat[,1]))
  all.m[,2]<-(all.m[,2]-min(dat[,2]))/diff(range(dat[,2]))
  dat[,1]<-(dat[,1]-min(dat[,1]))/diff(range(dat[,1]))
  dat[,2]<-(dat[,2]-min(dat[,2]))/diff(range(dat[,2]))

  all.m<-all.m[-1,]
  return(list(dat=dat,m=all.m))
}

r<-data.gen(n.peaks=100, N=1e5, max.var=0.001, max.corr=0.5)

k1<-kdeC(r$dat, H=c(0.005,0.005),  gridsize = c(501,501), cutNum=c(1,1))
k2<-kdeC(r$dat, H=c(0.005,0.005),  gridsize = c(101,101), cutNum=c(5,5))
```

`plot2d` *plot function*

Description

plot all the data points(black spots in the plot) and peaks(red spots in the plot) in one coordinate system

Usage

```
plot2d(x,matPeaks,evalpointsX,evalpointsY)
```

Arguments

`x`	data points in the format n*2 matrix
`matPeaks`	matrix returned by the `findPeak` function

`evalpointsX`	points at which the `matPeaks` is evaluated at x-axis direction
`evalpointsY`	points at which the `matPeaks` is evaluated at y-axis direction

Details

The function `plot2d` is mainly designed to make the result of functions `kdeC` and `findPeak` visual

Author(s)

Kunyu Ye

Examples

```
data.gen<-function(n.peaks=100, N=1e5, max.var=0.001, max.corr=0.5)
{
  library(mvtnorm)

  dat<-matrix(0, nrow=N, ncol=2)
  all.m<-c(NA,NA)

  for(i in 1:n.peaks)
  {
    this.m<-runif(2)
    this.var<-runif(2, min=0.1*max.var, max=max.var)
   this.cov<-runif(1, min=-1*max.corr, max=max.corr) * sqrt(this.var[1])* sqrt(this.var[2])
    this.s<-matrix(c(this.var[1], this.cov, this.cov, this.var[2]),ncol=2)

   dat[((i-1)*N/n.peaks+1):(i*N/n.peaks),]<-rmvnorm(N/n.peaks, mean=this.m, sigma=this.s)
    all.m<-rbind(all.m, this.m)
  }

  all.m[,1]<-(all.m[,1]-min(dat[,1]))/diff(range(dat[,1]))
  all.m[,2]<-(all.m[,2]-min(dat[,2]))/diff(range(dat[,2]))
  dat[,1]<-(dat[,1]-min(dat[,1]))/diff(range(dat[,1]))
  dat[,2]<-(dat[,2]-min(dat[,2]))/diff(range(dat[,2]))

  all.m<-all.m[-1,]
  return(list(dat=dat,m=all.m))
}

r<-data.gen(n.peaks=100, N=1e5, max.var=0.001, max.corr=0.5)

k1<-kdeC(r$dat, H=c(0.005,0.005),  gridsize = c(501,501), cutNum=c(1,1))

matPeaks<-findPeak(estimate=k1$estimate,filter=0)

plot2d(x=r$dat,matPeaks=matPeaks,evalpointsX=k1$evalpointsX,evalpointsY=k1$evalpointsY)
```

Index

∗ **kernel**
    kdeC, 3
∗ **peak**
    findPeak, 1
∗ **plot**
    plot2d, 4
∗ **smooth**
    kdeC, 3

findPeak, 1

kdeC, 3

plot2d, 4