kernel_estimator_selection_bootstrapping

1 - Cross-Validation For Kernel Estimators

A – Let's implement a function that computes, for a given vector u and a given window h, the value $\hat{f}_h(.)$ for each coordinate of the vector u. Then let's simulate exponential variables and apply the kernel estimator with different values for h.

B – Choosing h can be done with cross-validation: Let's rely on the least-square contrast:

$$C(g) = -\frac{2}{n}\sum_{i=1}^{n} g(X_i) + \int g(x)^2 dx$$

Such that $\mathbb{E}[C(g)]$ is minimal when $f = g$ where g is a candidate density. Let's demonstrate that state.

C – Using the function `integrate` , let's compute the integral of the kernel estimator f to the square and implement a function that computes the least square contrast on a different sample than the one previously simulated.

D – Let's implement the hold-out estimator: The data is cut in half at random, half to be used for estimation, and the other half for the selection of the bandwidth parameter h. Then let's plot the corresponding estimator.

A

Your code is reproduced here with comments below.

```
f_h=rep(1,100)
h=0.15
x=rexp(100,2)
u=c(h,x)
for(sim in 1:100){
  n=1:100
  #K=(exp*(((-x)^2)*2)/(sqrt(2*pi))) <- declaration error (see comments),
  #                                     exp() is a standard library function in
  #                                     R and not a variable.
  #f_h[sim]=(1/n*h)*sum(K*((u-X_i)/h)) <- X_i is not declared which breaks the
  #                                       code
}

#plot(density(f_h),bty='n') <- because the loop computation fails, it outputs
#                              a faux-normal distribution
```

1. `K=(exp*(((-x)^2)*2)/(sqrt(2*pi)))` is erroneous because of the order of operations. Exponents have priority over multiplication so $K(x) = exp(-x^2/2)/\sqrt{2\pi}$ should be `K = function(x) {exp(-(x^2)/2)/sqrt(2*pi)}` .
2. $u, \forall i \in \{1, \ldots, n\}, X_i$ are unknown vectors that are input to a function (requested by the instruction, along with a real-valued bandwidth h).
3. The idea behind the question is to create a convolution operation (the kernel estimator (https://en.wikipedia.org/wiki/Kernel_density_estimation)) that performs an operation over slices of input signals (here the interspike intervals X_i) to approximate an underlying distribution law (here exponential):
    - The use of a bandwidth (like in signal processing, think radio signals) is meant to extract details from an input at different levels of details (think zooming in and out with a microscope).
    - Since we only have one law underlying the data X_i, a single bandwidth h should be necessary in the case of the exercise. This is why the latter questions wanted to implement differnt model selections, to find the best bandwidth h to yield the actual exponential function that characterizes the ISI generated in this question.

B

The demonstration is correct and exhaustive.

C

The process described on paper looks about right, though the instruction requested it to be implemented in R.

D

It seems that you mentioned that the question D was also covered by your answer in B. I did not understand what you meant by writing `1) b)d)`

2 - Parametric Bootstrap

Overview

Let's implement parametric bootstrap.

A – Let's simulate $n = 50$ exponential variables with parameter $\theta_0 = 2$. This will be considered as observations.

A

Completed as requested.