Lecture 11: Kernel Density Estimator

D. Jason Koskinen
koskinen@nbi.ku.dk

Advanced Methods in Applied Statistics
Feb - Apr 2017
Quick

• Extra credit, +2% towards final cumulative percentage, is now available on course webpage w/ a due date of March 16 by 17:00 CET

• Problem set #2 is also online w/ a due date of March 20 by 16:00 CET

• Class lecture, especially the next 3 Thursdays, will be shorter than usual so that you can focus on the project assignment

  • Here’s what we will be covering between now and the exam: kernel density estimators (today), uniform confidence intervals & Feldman-Cousins, wavelets, MultiNest bayesian inference, on-going research (maybe), random stuff, and a review lecture
Important

- The exam will begin on **March 30, 2017** with a submission deadline on **March 31, 2017**
• Much of what we have covered has been parameter estimation, but using analytic or defined density expressions
• Today we cover density estimates from the data itself
• The methods are regularly employed on finite data samples that need smoothing or require non-parametric methods to get a PDF

• Last few slides of this lecture contain extended literature options
Histogram

• The histogram is one of the most simple forms of a data-driven non-parametric density estimator.
• But, the only two histogram parameters (bin width and bin position(s)) are arbitrary.

*From Lecture 1*
Histograms

• The histogram is one of the most simple forms of a data-driven non-parametric density estimator

• But, the only two histogram parameters (bin width and bin position(s)) are arbitrary
  • Histograms are not smooth, but ideally our density estimator/function is smooth
  • More dimensions requires more data in order to have a multi-dimensional histogram which can match the true PDF

• We can avoid some of these issues, and others, with density estimates by using something more sensible
Wish List

• For density estimates what do we want?
  • non-parametric, i.e. no explicit requirement for the form of the PDF
  • (Easily) extendable to higher dimensions
  • Use data to get local point-wise density estimates which can be combined to get an overall density estimate
  • Smooth
    • At least more smooth than a ‘jagged’ histogram
  • Preserves real probabilities, i.e. any transformation has to give PDFs which integrate to 1 and don’t ever go negative

• The answer… Kernel Density Estimation (KDE)
  • Sometimes it is “Estimator” too for KDE
Basics

• Fixed regions in parameter space are expected to have approximately equal probabilities
  • The smaller the region(s) the more supported our assumption that probability is constant
  • The more data in each region the more accurate the density estimate

• We will keep the region fixed and find some compromise; large enough to collect some data, but small enough that our probability assumption is reasonable
  • For more thorough treatment of the original idea see the articles by Parzen and Rosenblatt in the last slides of this lecture
Hyper-Cube

- Our first ‘region’ definition is an n-dimensional hyper-cube with lengths equal to h on each side
  - In 3D this is a normal cube with volume $h^3$
  - In 2D, when there are two parameters, the hyper-cube is a square with area $h^2$
- We include all points within the hyper-cube volume via some weighting scheme. This is known as the kernel (K) which is for this KDE:

$$K(x_i) = \begin{cases} 1, & x_i \text{ in region } R_d \\ 0, & x_i \text{ outside region } R_d \end{cases}$$

for some $R$ centered at point $x_d$

- Sometimes you will see the kernel as $K(u)$ where $u$ is the ‘distance’ from $x_i$ to $x_d$
Visual Region

- Everything within the hyper-cube is included with a weight of 1
- For the density estimator we need:
  - To normalize by the total number of events in the sample (N)
  - To normalize for the number of dimensions (D) and the ‘volume’
- The PDF estimator \( P_{KDE} \) is now constructed from the individual data points
- The estimation point is \( x_d \)

\[
P_{KDE}(\vec{x}) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left( \frac{\vec{x} - \vec{x}_n}{h} \right)
\]
Exercise 1

- Take the fixed length hyper-cube KDE out for a spin in 1D

- Using the following data [1,2,5,6,12,15,16,16,22,22,22,23] for the finite data sample and h=3

- Because the length is h, to be in the cube each data point $x_i$ only needs to be $h/2$ in each dimension from $x_d$

- Calculate $P_{KDE}(x=6)$, $P_{KDE}(x=10.1)$, $P_{KDE}(x=20.499)$, and $P_{KDE}(x=20.501)$

Code this by-hand, i.e. no external packages
Exercise 1 Example

• Calculate the $P_{KDE}(x=6)$ by taking all 12 data points and seeing if they are within $\pm h/2$ of $x=6$, i.e. in the range 4.5 to 7.5. Here 6 is our $x_d$ value from the picture on slide 8.

• We include the data point at $x=6$ in the KDE

$$P_{KDE}(x = 6) = \frac{1}{NhD} \sum_{n=1}^{N} K\left(\frac{\bar{x} - \bar{x}_n}{h}\right)$$

$$= \frac{1}{12 \times 3^1} \left[ K\left(\frac{1 - 6}{3}\right) + K\left(\frac{2 - 6}{3}\right) + K\left(\frac{5 - 6}{3}\right) + K\left(\frac{6 - 6}{3}\right) + \ldots K\left(\frac{23 - 6}{3}\right) \right]$$

$$= \frac{1}{12 \times 3^1} \left[ 0 + 0 + 1 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 \right]$$

*The $x$ and $x_n$ are switched from the first line to the second. But, it doesn’t make any difference.*
KDE Comments

- The function $K()$ is known as the **kernel**, and $h$ is the **bandwidth**
- Larger bandwidths mean more smoothing, but it can remove real features
- Smaller bandwidths will approach the true PDF better, but need lots of data points otherwise they’re ‘bumpy’
- The fixed window KDE is similar to a histogram, but has better support for local densities
Hyper-Cube

• We could use the hyper-cube kernel to construct a density estimator, but there are a few drawbacks to this kernel
  • We have discrete jumps in density and limited smoothness
  • Nearby points in \( x \) have some sharp differences in probability, e.g. \( P_{KDE}(x=20.499)=0 \) but \( P_{KDE}(x=20.501)=0.08333 \)
  • All data have equal weighting and contribution regardless of distance to the estimation point

• So let’s switch to a different kernel with weights that decrease smoothly as a function of distance from the estimation point
Gaussian Kernel

• The generic KDE expression remains similar, i.e.

\[ P_{KDE}(\vec{x}) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left(\frac{\vec{x} - \vec{x}_n}{h}\right) \]

• The kernel is now:

\[ K(\vec{x}, \sigma) = \frac{1}{(\sqrt{2\pi}\sigma)^D} e^{-\frac{||\vec{x} - \vec{x}_n||^2}{2\sigma^2}} \]

• The kernel at each data point now contributes a non-zero probability from \([-\infty, +\infty]\) smoothly with decreasing weight as a function of distance

  • Each data point and corresponding kernel integrate to 1 over the whole parameter space
Exercise 2

• Redo exercise 1 using the new Gaussian kernel
• For the gaussian width use $\sigma=3$
• Calculate the KDE two ways:
  • By hand
  • Using an external package
• Plot the density estimate $P_{KDE}(x)$ from $-10 < x < 35$
• If you have time, plot the individual kernel contributions too
Exercise 2 KDE plot

Gaussian Kernels ($\sigma=3.00$)
Compact Kernel

- The gaussian kernel contributes across the whole space (infinite support), but sometimes we want compact support, i.e. zero outside of a specific range
  - Maybe some parameters are constrained to be non-negative
  - We know the physical system has either boundaries or effective cut-offs
- A common compact support kernel is the Epanechnikov kernel

\[
K(u) = \begin{cases} 
\frac{3}{4}(1 - u^2) & \text{for } |u| \leq 1 \\
0 & \text{for } |u| > 1
\end{cases}
\]

Exercise 3

• Redo exercise 2 using the Epanechnikov kernel with a bandwidth that you choose

• In a nicely formatted table compare Calculate $P_{KDE}(x=6)$, $P_{KDE}(x=10.1)$, $P_{KDE}(x=20.499)$, and $P_{KDE}(x=20.501)$ between the 3 different kernels; Parzen-Rosenblatt, gaussian, and Epanechnikov
  • Use either your by-hand(s) version or external package
Kernel Bandwidth

- Every KDE is, unfortunately, strongly influenced by the kernel bandwidth, which is a user defined free parameter.
Bandwidth Selection

- An analytic approach to bandwidth selection is to choose a bandwidth which minimizes the mean integrated square error (MISE)

\[ MISE(h) = E \left[ \int (P_{KDE}(\tilde{x}) - P(\tilde{x}))^2 d\tilde{x} \right] \]

- But analytically this requires some known form of the underlying distribution

- Assuming that the underlying distribution is gaussian, the optimal bandwidth is

\[ h \approx 1.06\hat{\sigma}N^{-1/5} \]

\[ \hat{\sigma} \] standard deviation from data

\[ N \] number of data points
Bandwidth Selection Non-parametric

• Instead of using a known function we can use subsets of the data as cross-validation of the kernel bandwidth

\[ ISE(\hat{f}_h) = \int (\hat{f}_h(y) - f(y))^2 dy \]

\[ = \int (\hat{f}_h(y))^2 dy - 2 \int \hat{f}_h(y) f(y) dy + \int f(y)^2 dy \]

• This can be shown to converge to a least squares cross-validation (LSCV) expression as

\[ LSCV(h) = \int (\hat{f}_h(y))^2 dy - \frac{2}{N} \sum_{i=1}^{n} \hat{f}_{-i}(y_i) \]

• The expression \( \hat{f}_{-i}(y_i) \) is the kernel estimator from the data omitting the data point \( y_i \) which is also known as the “leave-one-out” density estimator

*https://projecteuclid.org/euclid.ss/1113832723
KDE in 2D, and more

• While the previous examples and work were for 1-dimension, the kernels work just fine in additional dimensions
  • No escaping the curse of dimensionality :-(
  • Similar to all other multi-dimensional problems, anything beyond 3D is difficult to visualize

• Kernel bandwidths do not have to be the same in each dimension
  • Either specify the bandwidth in each dimension, or
  • Transform the parameter space(s) to be uniform for a given bandwidth
More KDE comments

• The kernel is symmetric about each data point
  • Makes sense, because the region near the data point should have a similar probability for a narrow (enough) bandwidth
  • Kernel symmetry is not technically a requirement, but in practice it is

• The kernel density estimator PDF is often used for Monte Carlo sampling
  • E.g. N-body simulations (galaxy formation, astrophysical large scale structure, disease propagation in an ecosystem, etc.) take 2 months to generate 200 data points across 3 dimensions or parameters. Real data is much, much larger. In order to use our N-body PDF, we can sample from a smoothed PDF from a KDE.

• Because KDEs require ‘subjective’ input, clearly state the kernel, bandwidth, and any optimization from an analysis
Exercise 4

- There are a host of online tutorials covering different 2D density estimation problems in R, python, MatLab, etc.
  - Eruption of “Old Faithful” geyser
  - Rendering of text and numerals
  - Spread of diseases
  - Geographical population densities

- After working through your own particular choice, use the 500 pseudo-experiment bootstrap from Lecture 7 exercise 1b to produce a 2D KDE
  - Because the LLH method gives precise contours, you can compare the contours from the KDE
Further Info

• Fixed kernel width window, Parzen-Rosenblatt window
  • Parzen (http://www.jstor.org/stable/2237880)
  • Rosenblatt (http://projecteuclid.org/euclid.aoms/1177728190)

• Nice list of various kernels at https://en.wikipedia.org/wiki/Kernel_(statistics)

• Very nice article on kernel bandwidth selection review
  https://projecteuclid.org/euclid.ss/1113832723

• Collection of other cross-validation techniques https://cran.r-project.org/web/packages/kedd/vignettes/kedd.pdf
Further Info (cont.)

• Variable bandwidth kernels

• Any other suggestions?