Carolyn Cotterman
Stat 244
March 16, 2010

# Assignment 2: Dynamic Graphics

## 1 Introduction

For this assignment, I used the tcltk package to develop a dynamic graphics program in R for the kernel density estimator. I then used the resulting graphical user interface (GUI) to study the effects of window width on smoothing. My code for this assignment is contained in the file called, "Stat244_hw2_R_smoother."

## 2 The Kernel Density Estimator

The nonparametric estimation of probability density functions and regression functions is known as "curve estimation" or "smoothing." By visually inspecting the smoothed density function, or the relationship between variables, we can gain insight on whether the assumptions regarding our data's functional form are satisfied. If our assumed functional form is incorrect, then the corresponding statistical inference is invalid. Thus, curve estimation is an important statistical procedure.

Many smoothing techniques exist. Here, I focus on the kernel density estimator, which is one of the most popular density estimators. Relative to histograms (the oldest form of nonparametric density estimation), kernel density estimators are smoother and converge faster to the true density, f. This estimator essentially puts a smoothed-out lump of mass 1/n over each data point, $X_1$,...$X_n$.

A kernel is defined to be any smooth function K such that the following are true:

$$K(x) \geq 0, \quad \int K(x)dx = 1, \quad \int xK(x)dx = 0 \tag{1}$$

The kernel density estimator is defined to be

$$\hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K(\frac{x - X_i}{h}) \tag{2}$$

where h is called the bandwidth, and is a positive number that controls the amount of smoothing – a topic to which we will return in the results section. A common function to use for K is the Gaussian normal. With this kernel, the density estimator becomes

$$\hat{f}_n(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h\sqrt{2\pi}} e^{-(\frac{x-X_i}{h})^2/2} \tag{3}$$

Other reasonable kernel functions include the epanechnikov, triangular, rectangular, triangular, biweight, cosine, and optcosine. It is difficult to produce empirical evidence favoring one kernel over another. In practice, the kernel function has less influence over the resulting estimator than does bandwidth, though we will see in the results section that the kernel does command some influence.

# 3 Dynamic Graphics Program

I developed a user interface that will produce a graph of the kernel density for a user-specified variable and user-supplied kernel function. My GUI asks the user to provide:

1. The name of a dataframe
2. The variable within this dataframe to analyze
3. The kernel function to use

The density of the user-specified variable will initially be displayed using a bandwidth of zero. The user can adjust the bandwidth to his liking using a slider widget, and the graph will be automatically updated as he does so. Bandwidth is defined in accordance with R's density function. Namely, it is the standard deviation of the smoothing kernel.

# 4 Results

To test and demonstrate my GUI's functionalities, I used a dataset containing health information for 462 South Africans (see filed called "SAheart.txt"). Figures 1 and 2, below, depict the age distribution in this data using the kernel density estimator with various parameter selections.

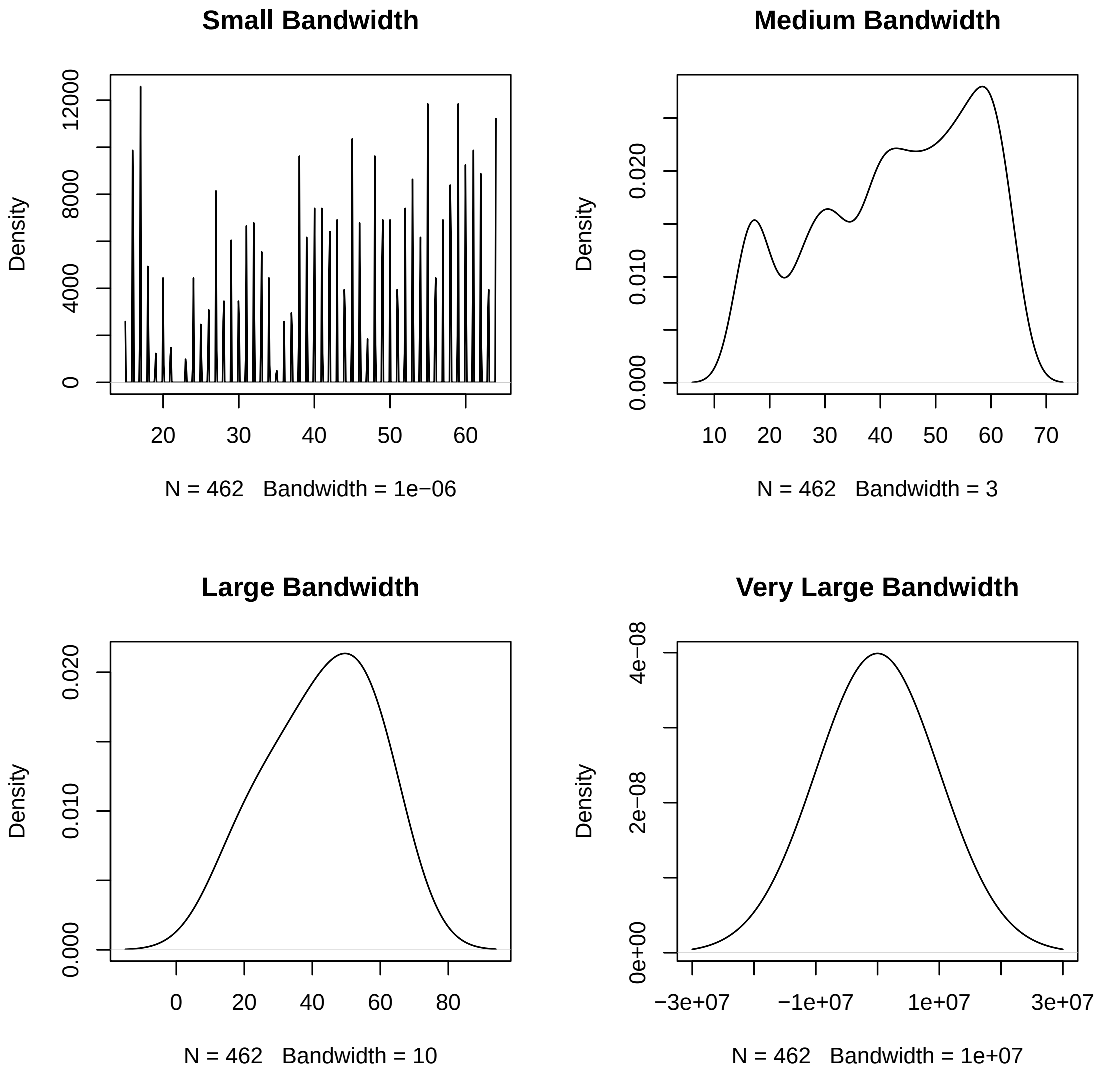
The first set of graphs reveal that the density function becomes increasingly smooth as bandwidth increases. The first graph of this set, for which bandwidth is close to zero, essentially contains a set of spikes placed at each data point (as bandwidth → zero, the height of these spikes tends to infinity). At the other extreme, the density function converges to a uniform distribution as bandwidth $\to \infty$. We can see this beginning to happen in the fourth panel — for a bandwitdh of ten million, we are effectively giving the same weight to all observations, regardless of their distance to the point at which we are estimating density. (Note that the changing x-axis scale on these graphs obscures the degree to which the density function using a very large bandwidth resembles the uniform distribution.)

The second set of graphs display the kernel density function for different kernel funtions while keeping bandwidth constant. These graphs are displayed in descreasing order of smoothness: we see that the cosine and epanechnikov functions are approximately as smooth as the Gaussian while the rectangular kernel results in a rougher density function than do the others. This is unsurprising, as the rectangular kernel has a binary inclusion rule, rather than a smooth one.

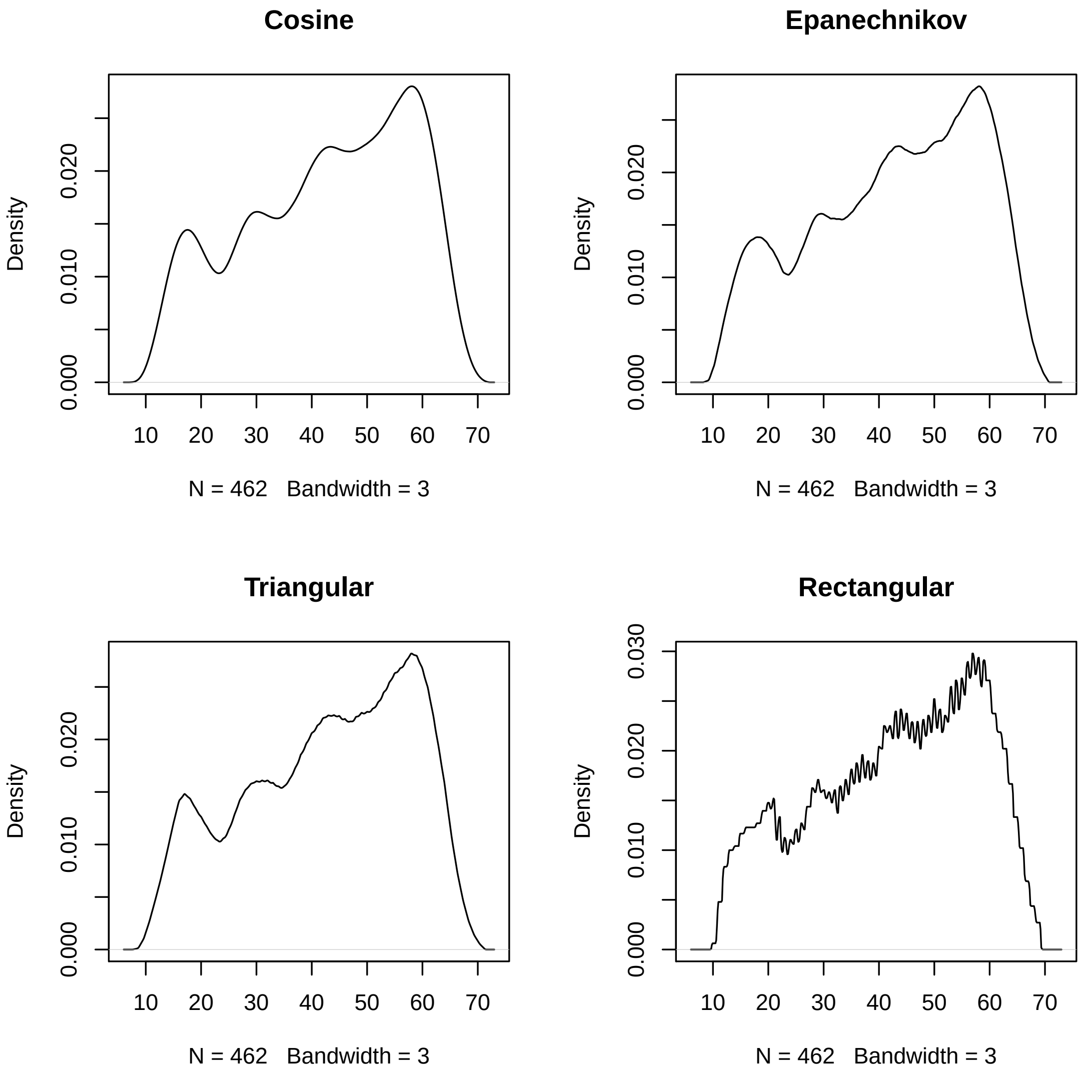
# 5 Conclusion

This assignment has provided me with both an introduction to and an appreciation for the tcltk R library. The graphical user interfaces made possible through this library will allow me to devise a simple way for another user (who might not know R) to run a complex analysis without having to alter any code him or herself. I additionally learned that the slider widget is a useful tool for learning about the effect of a parameter's value on the outcome of a procedure – I used this widget to explore the effect of bandwidth on the shape of the kernel density estimator. If in the future I am ever interested in providing a GUI to another user, or in visually exploring the effect of changing a parameter, I will consider using R's tcltk library.

**Figure 1: Effect of Bandwidth on Density Function using the Gaussian Kernel**



## Figure 2: Effect of Kernel on Density Function for Fixed Bandwidth


Cosine
Density
N = 462   Bandwidth = 3
Epanechnikov
Density
N = 462   Bandwidth = 3
Triangular
Density
N = 462   Bandwidth = 3
Rectangular
Density
N = 462   Bandwidth = 3