

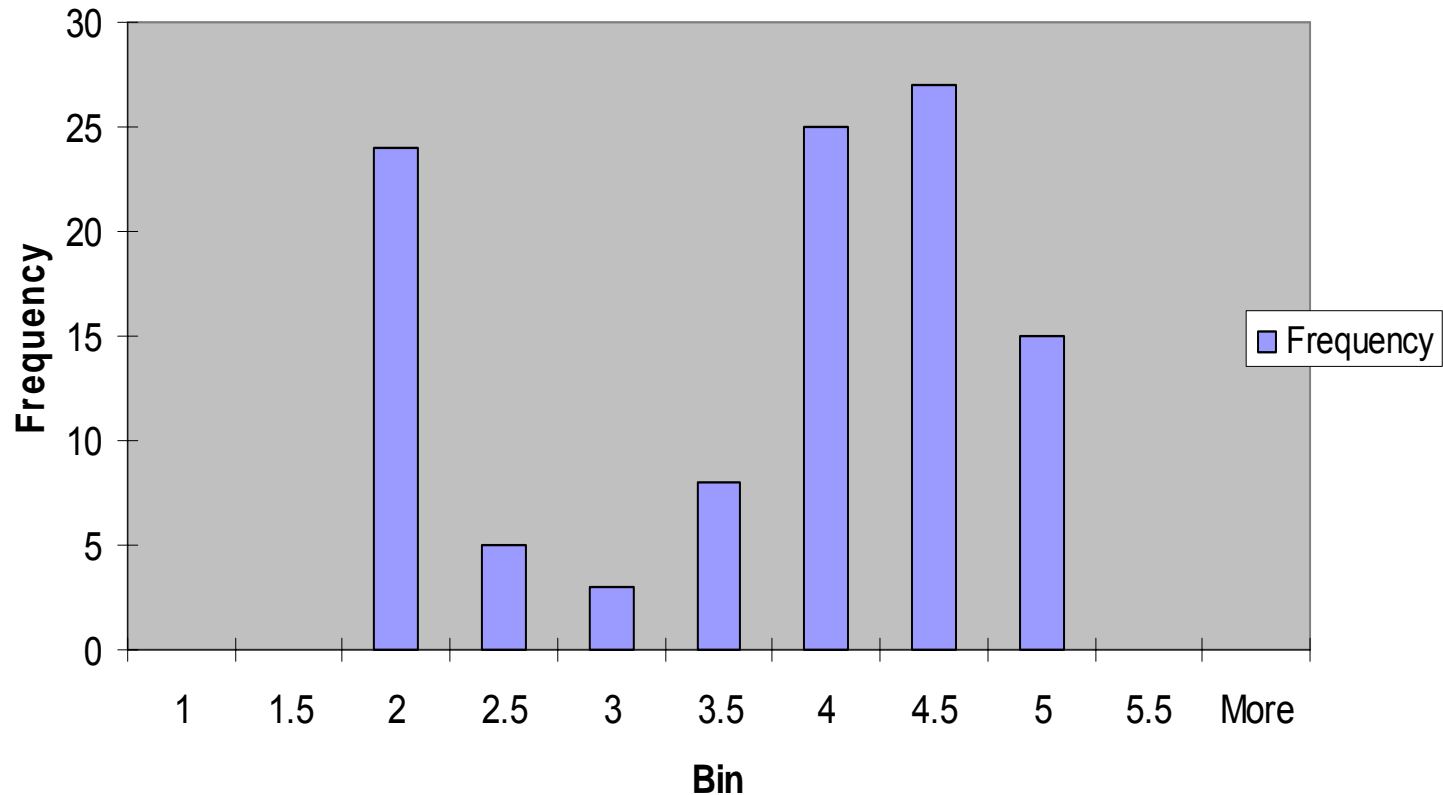
Kernel Density Estimation

Histograms have a problem

- The following few slides show the histograms of the eruption times for Old Faithful, a geyser in the Yellowstone National Park, USA (Source: Weisberg 1980)

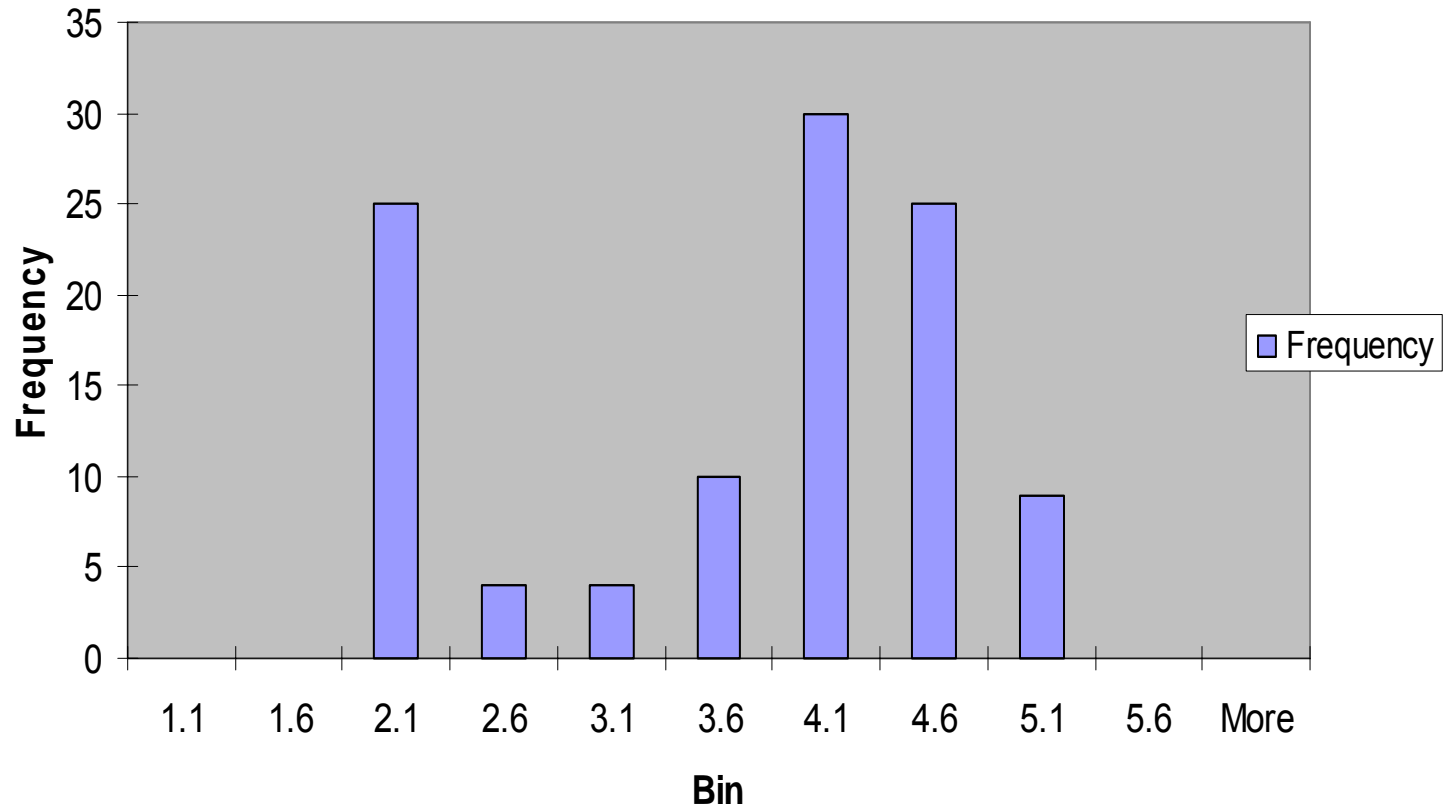
Histograms have a problem

Histogram of Geyser Data
Bins Starting at 1, Bin width 0.5



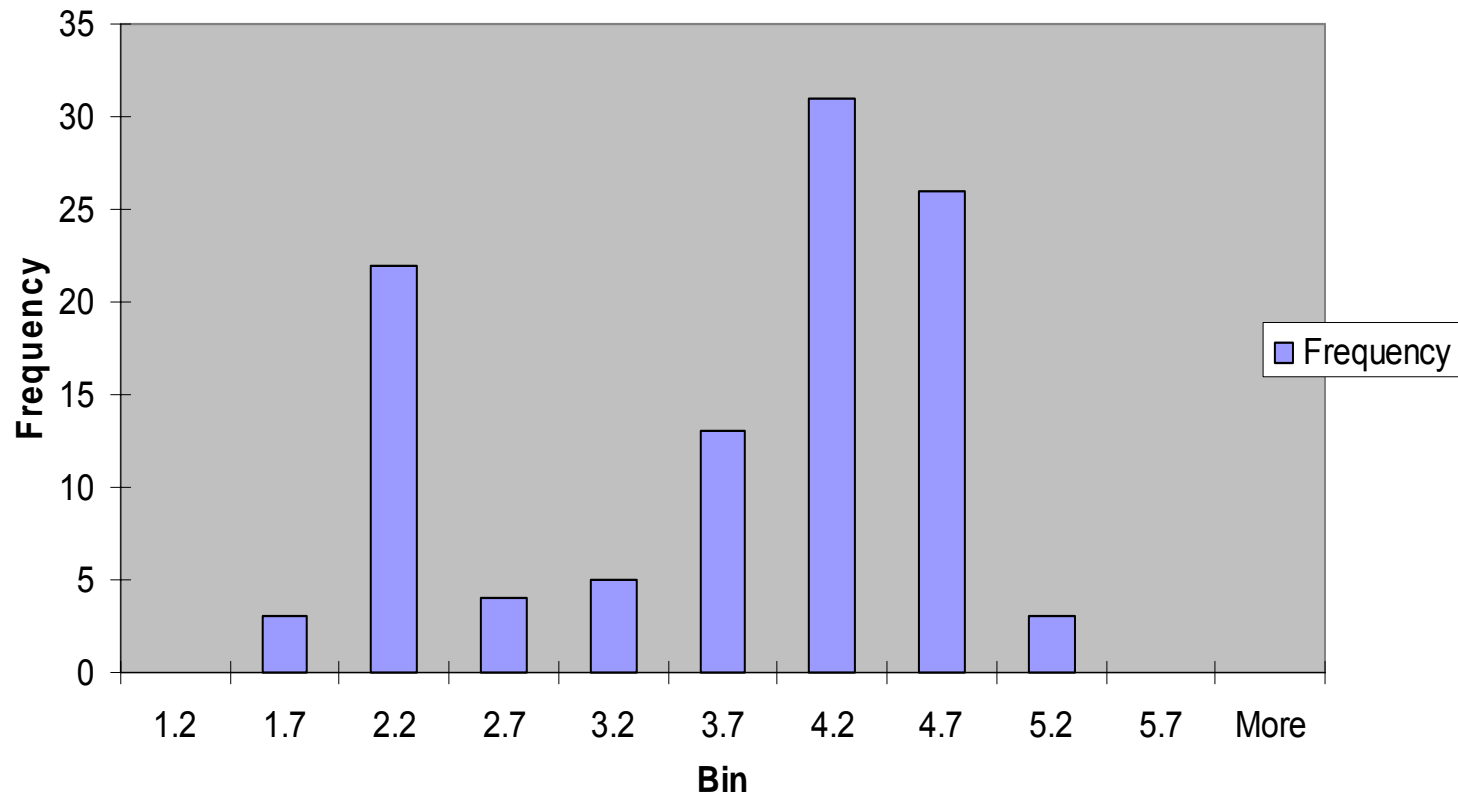
Histograms have a problem

Histogram of Geyser Data
Bins Starting at 1.1, Bin width 0.5



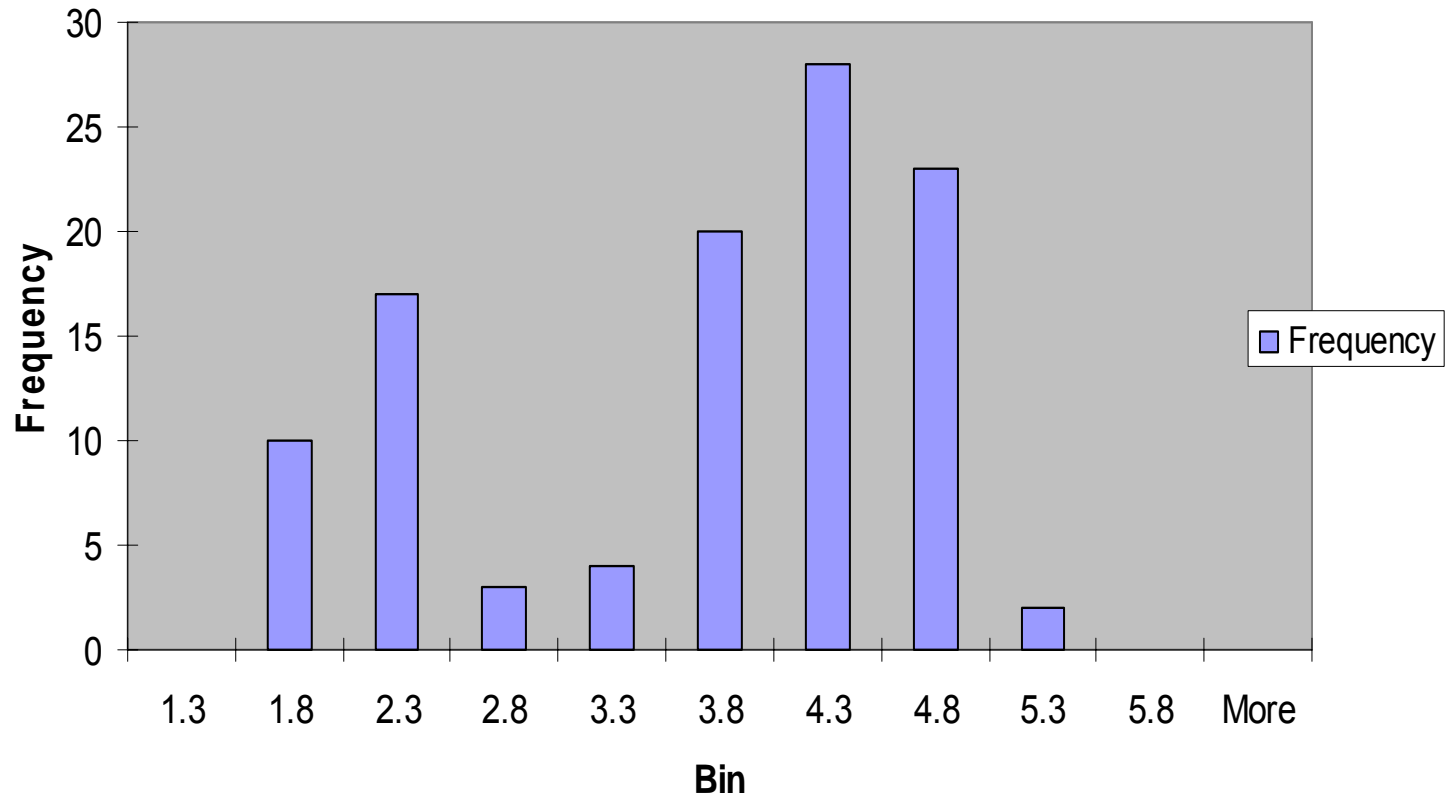
Histograms have a problem

Histogram of Geyser Data
Bins Starting at 1.2, Bin width 0.5



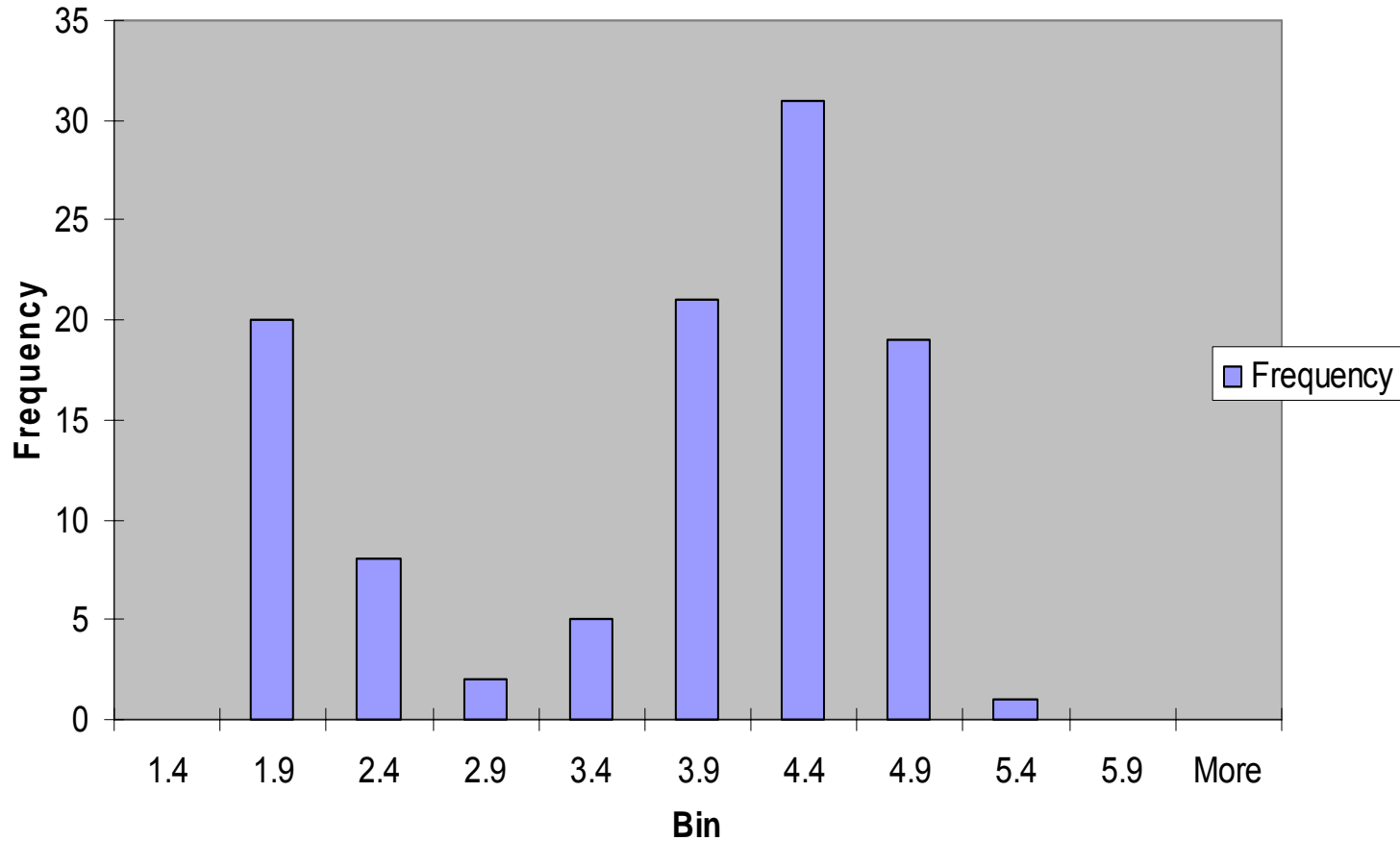
Histograms have a problem

Histogram of Geyser Data
Bins Starting at 1.3, Bin width 0.5



Histograms have a problem

Histogram of Geyser Data
Bins Starting at 1.4, Bin width 0.5



Histograms have a problem

- These very different pictures were got just by changing the starting points for the bins
- The effect of changing the width of the bins could be even more striking
- Problems will be even worse in more than one dimension

Naïve Estimator

- From the definition of a probability density f :

$$f(x) = \lim_{h \rightarrow 0} \frac{1}{2h} P(x - h < X < x + h)$$

So we can choose to estimate this by counting the proportion of the sample that fall in $x-h$ to $x+h$.

$$\hat{f}(x) = \frac{1}{2hn} [\text{Number of } X_1, \dots, X_n \text{ falling inside } (x - h, x + h)]$$

Naïve Estimator

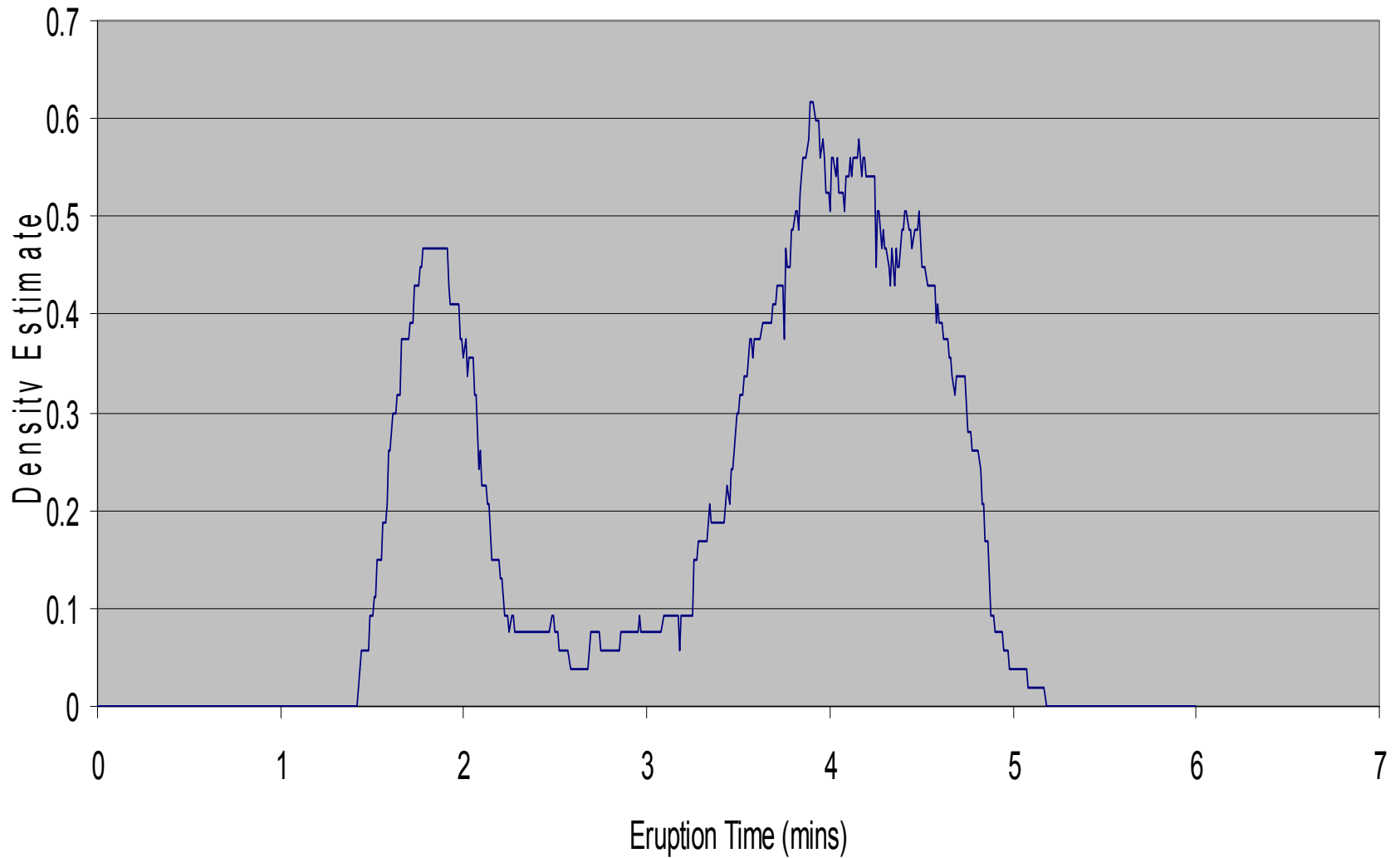
- This is better done with weights, where

$$w(x) = \begin{cases} 1/2 & \text{if } |x| < 1 \\ 0 & \text{otherwise} \end{cases}$$

This enables us to write the naive estimator as:

$$\hat{f}(x) = \frac{1}{n} \sum_{i=1}^n \frac{1}{h} w\left(\frac{x - X_i}{h}\right)$$

Naive Estimate for Density Estimate for Geyser Data



Problems with Naïve Estimators

- Not a nice smooth function, both aesthetically undesirable, and possibly misleading
- Hence we replace the weight function with a *kernel function*, designed to give a smoother density estimate. This will often be a symmetric probability density function like the normal density function

Kernel Density Estimator

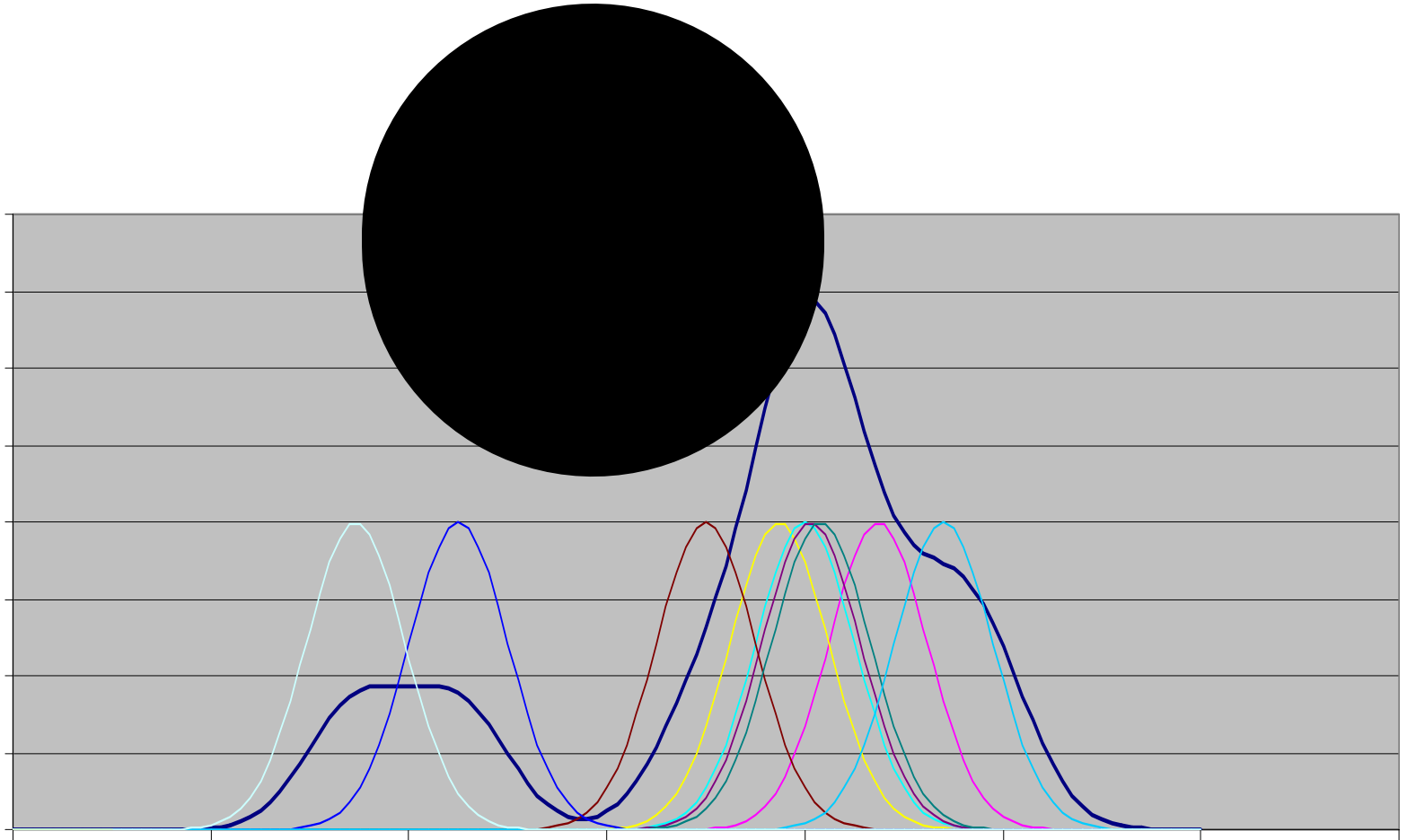
- We use a kernel function K which satisfies:

$$\int_{-\infty}^{\infty} K(x)dx = 1$$

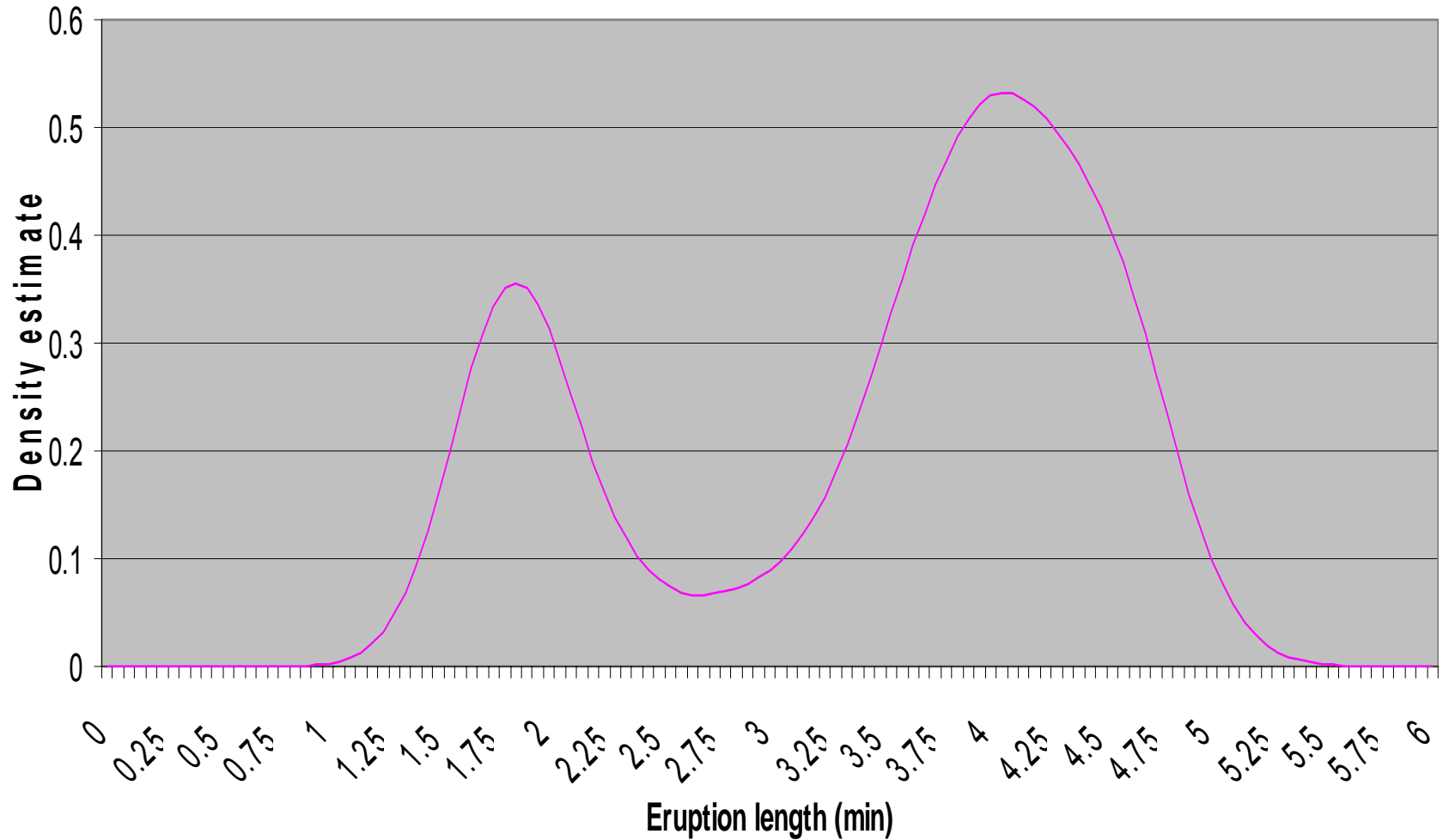
This kernel takes the place of the weight function in our naïve estimator, hence the kernel estimator is:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - X_i}{h}\right)$$

Kernel Estimator



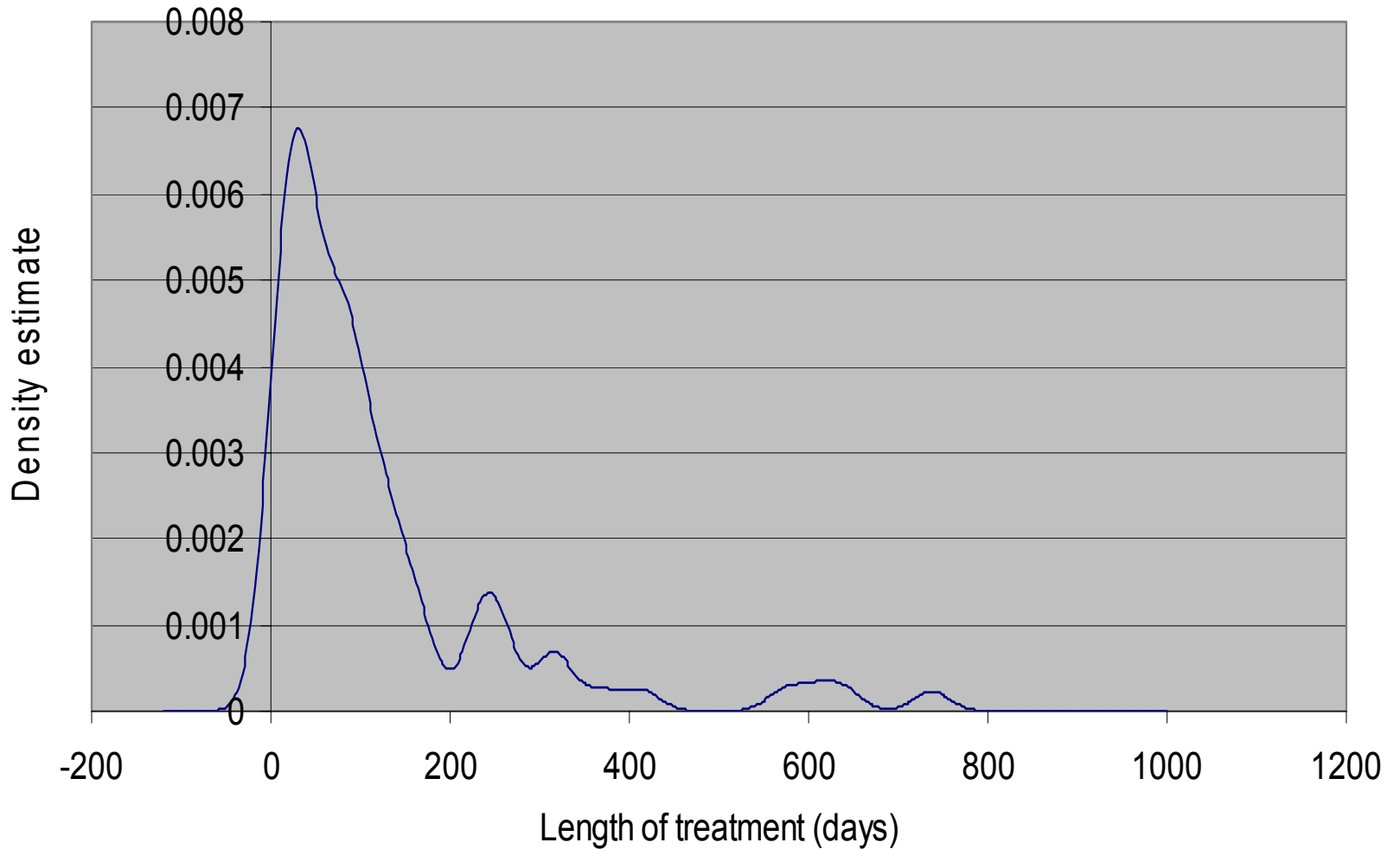
Kernel Density Estimate for Old Faithful geyser data



Problems With Kernel Estimator

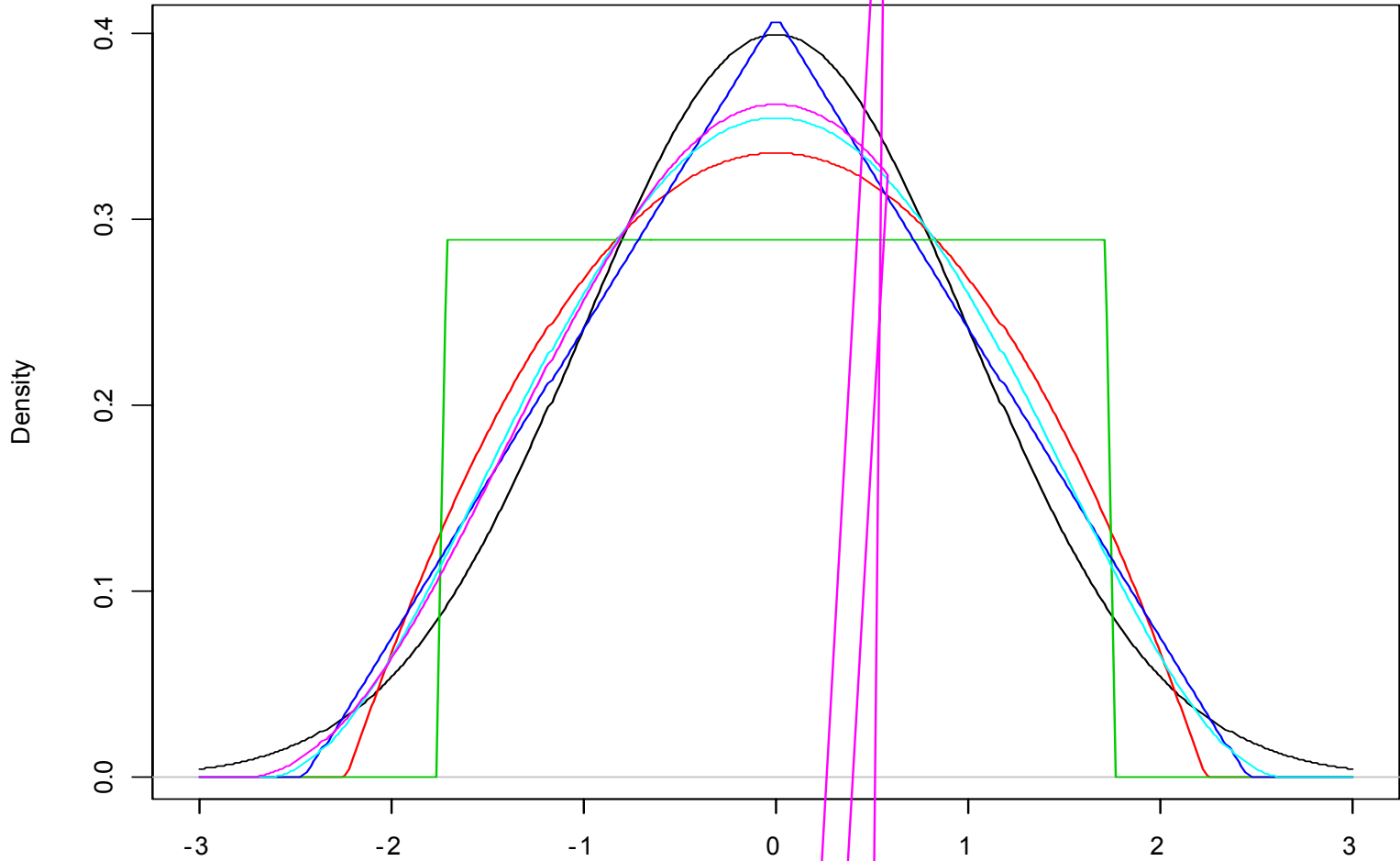
- Need to decide on a suitable value for h , the “bandwidth”
- What to use as a kernel
- Can have problems when the data is restricted to a certain range of values in that the kernel estimator will have a positive probability of providing estimates outside of this range.

Kernel Density Estimate for Suicide Study data



Possible Kernels

R's density() kernels with bw = 1



Deciding on the bandwidth

- There are complicated formulae to help you decide, but this is perhaps best done by eye.
- For those who want a better answer see Jones, Marron and Sheather (1996)“A brief survey of bandwidth selection for density estimation”

Journal of the American Statistical Association 91, p401-7

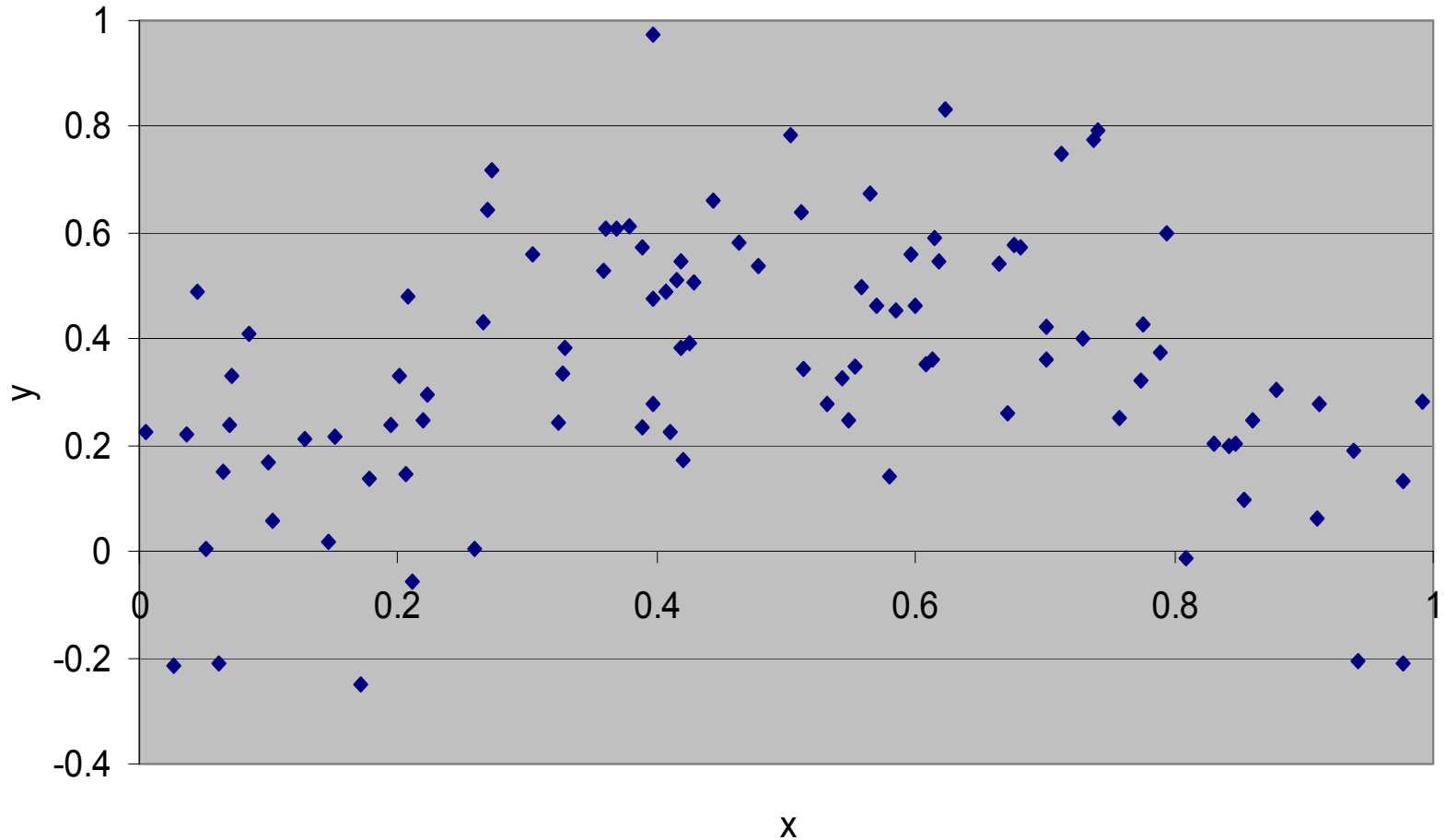
Kernel Smoothers

- We can use a similar idea to fit a *local regression line*
- This will avoid the discontinuities of the nearest neighbour method, as now the weights will continuously rise up from zero and then gradually sink back to zero again.

$$\hat{f}(x) = \frac{\sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) y_i}{\sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)}$$

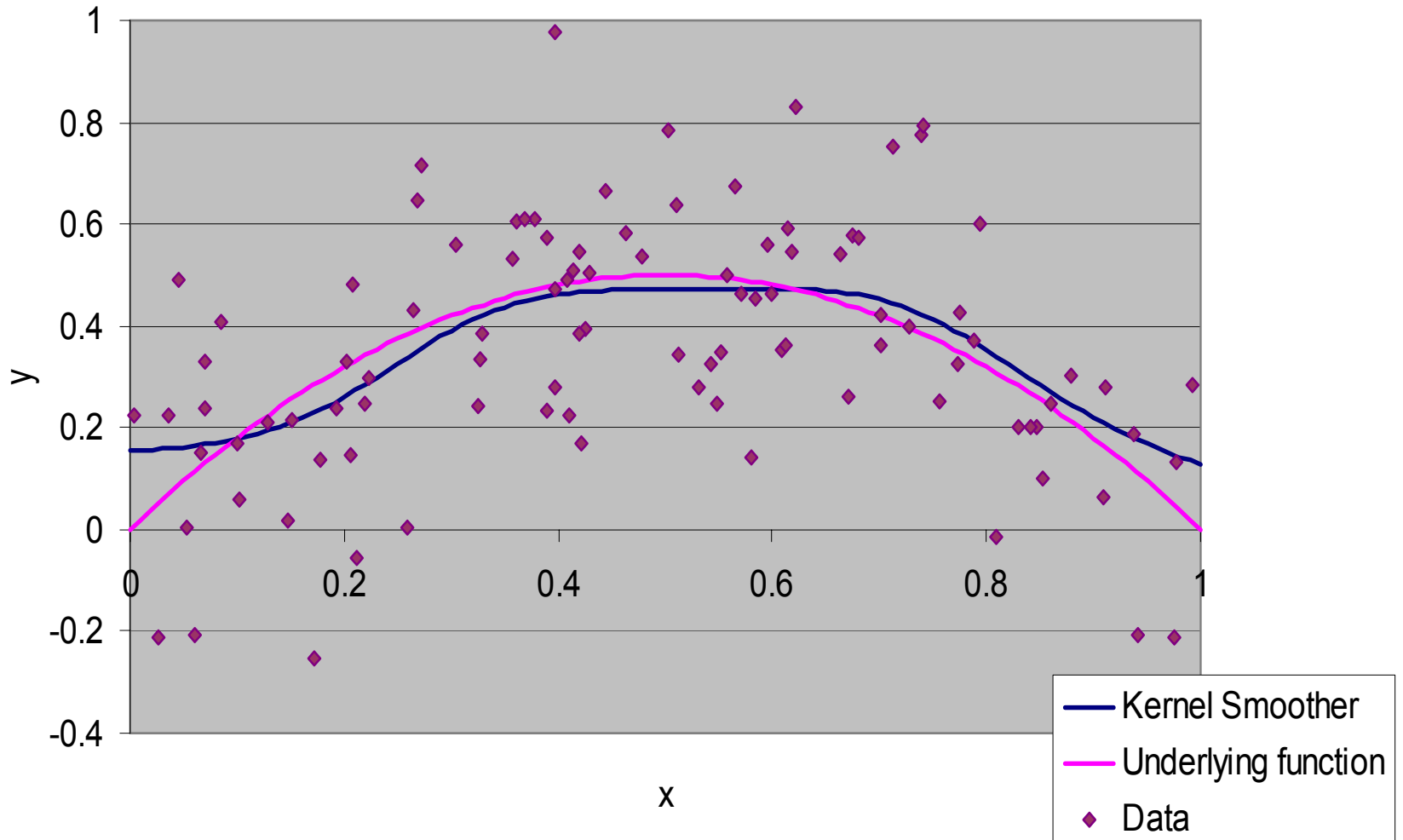
Simulated data for Kernel Smoothing

Underlying Data



Kernel Smoothing with $h=0.01$

Kernel Smoothing

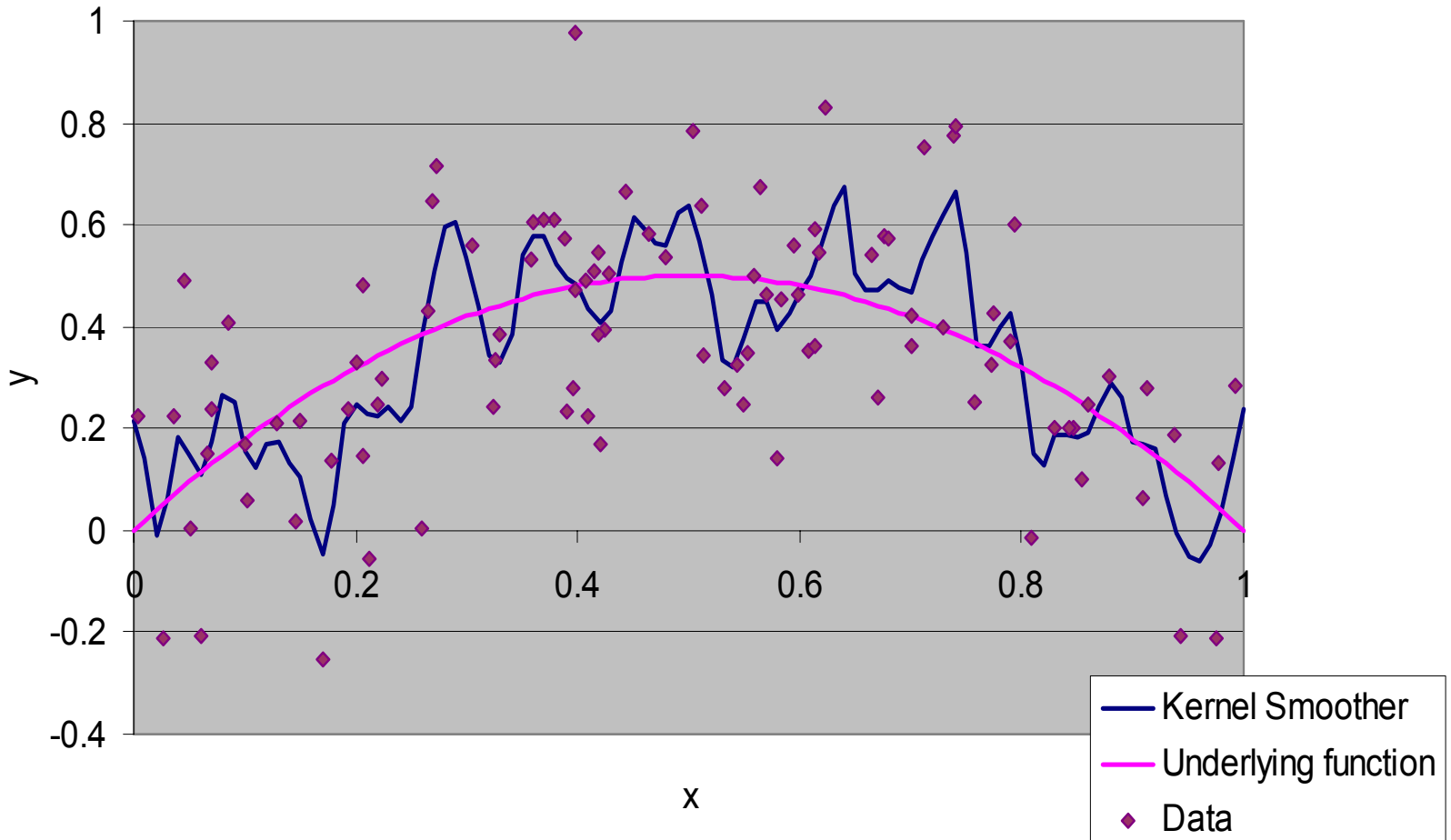


Problems with Kernel Smoothing

- Deciding on the bandwidth
- Deciding on the kernel
- The method effectively fits a local level and so does not do well at the edges, where it will tend to fit a level rather than the slope. The method has been extended to fit local polynomials, which avoids this.

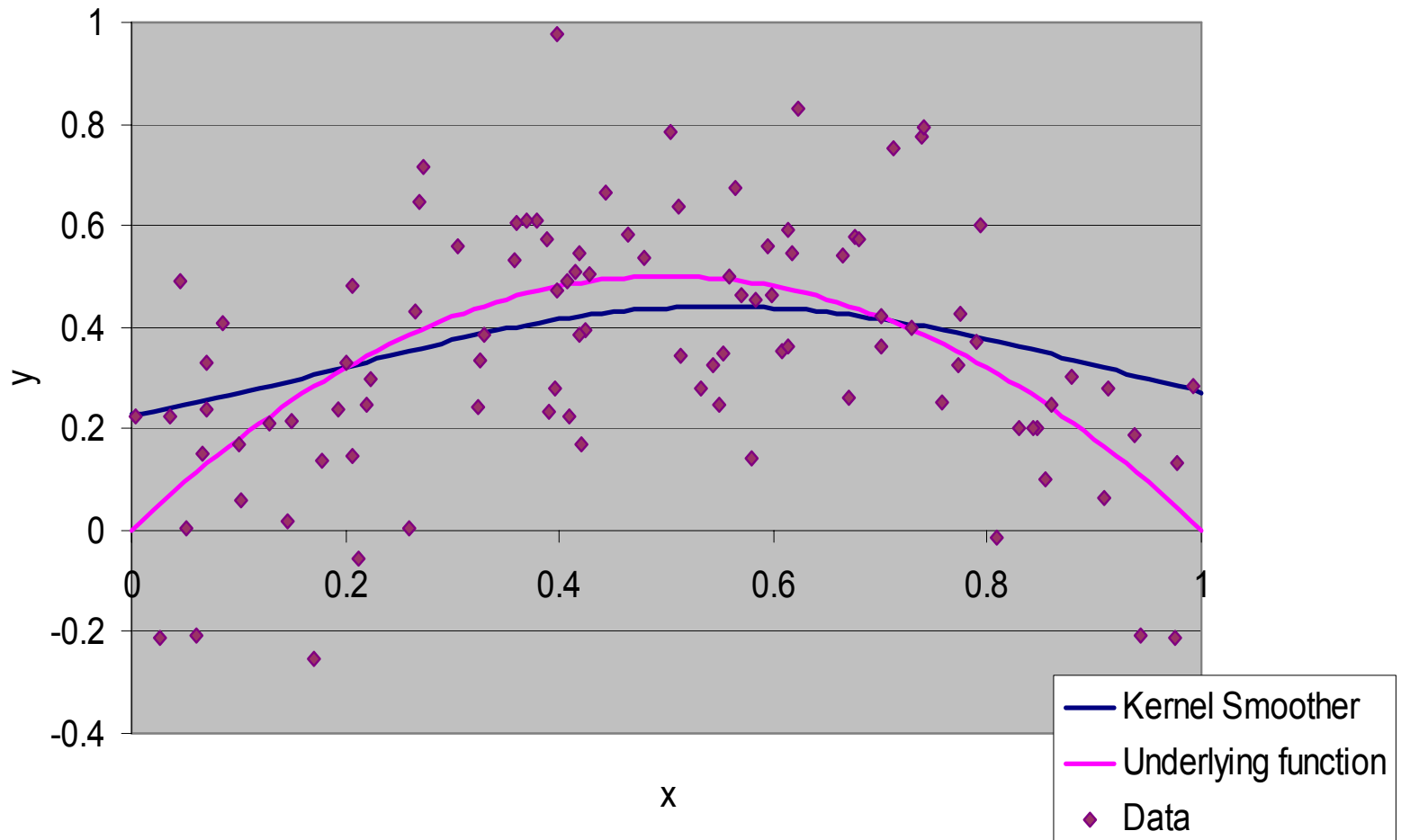
Kernel smoothing with $h=0.01$

Kernel Smoothing



Kernel smoothing with $h=0.2$

Kernel Smoothing



Exercises

- Implement Kernel Estimation on a spreadsheet. (The data set Geyser is available on the website)
- Implement Kernel Smoothing on a spreadsheet. (There is a data set called motorcycle.xls available on the website.)
- Try doing both in R

Relevant R functions

- **ksmooth**(x, y, kernel = c("box", "normal"), bandwidth = 0.5, range.x = range(x), n.points = max(100, length(x)), x.points)
- E.g. **ksmooth**(speed, dist, "normal", bandwidth=2)

- **density**(x, bw = "nrd0", adjust = 1, kernel = c("gaussian", "epanechnikov", "rectangular", "triangular", "biweight", "cosine", "optcosine"))
- E.g. **density**(faithful\$eruptions, bw = "sj")

(Note: bw (or bandwidth) = "sj" uses the Sheather-Jones method to determine bandwidth.)

R function for local polynomial fitting

- **Locpoly** which has a variety of subcommands to deal with a range of options but at it's simplest looks like:
- E.g. **locpoly(x,y,bandwidth=0.25)**