Calculating the Nonparametric Regression Estimators Using the Kernsmooth Library

Lydia Kim

```
# necessary packages
library(ggplot2)
library(KernSmooth)
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2
## --
```

```
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## v purrr   1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

The file andro2.txt contains the record of biological signal taken on a mouse subject. We are going to model this data as noisy measurements around an unknown mean function.

First, we we will estimate the value of the mean function at time 200 by first averaging the 35 points closest to that time. Then, we will give a 95% confidence interval for this estimates using a t statistic and estimating the variance from only the data used in the mean.

```
andro2 <- read.delim('andro2.txt', sep=' ')
names(andro2) <- c('Time', 'Signal')
indx <- which(andro2$Time == 200)
mu.hat.35 <- print(mean(andro2$Signal[(indx-17): (indx+17)]))
```

```
## [1] 0.02257143
```

```
se.35 <- print(sd(andro2$Signal[indx+(-17:17)])/sqrt(35))
```

```
## [1] 0.01720046
```

```
mu.hat.35 + qt(c(0.025, 0.975), df=34)*se.35
```

```
## [1] -0.01238412  0.05752698
```

We will repeat the estimation again but use 105 points closest to time 200. Then, we will find the corresponding 95% confidence interval when using 105 observations in the mean, and estimating the variance from those 105 observations.

```
mu.hat.105 <- print(mean(andro2$Signal[(indx-17): (indx+17)]))
```

```
## [1] 0.02257143
```

```
se.105 <- print(sd(andro2$Signal[indx+(-17:17)])/sqrt(105))
```

```
## [1] 0.009930692
```

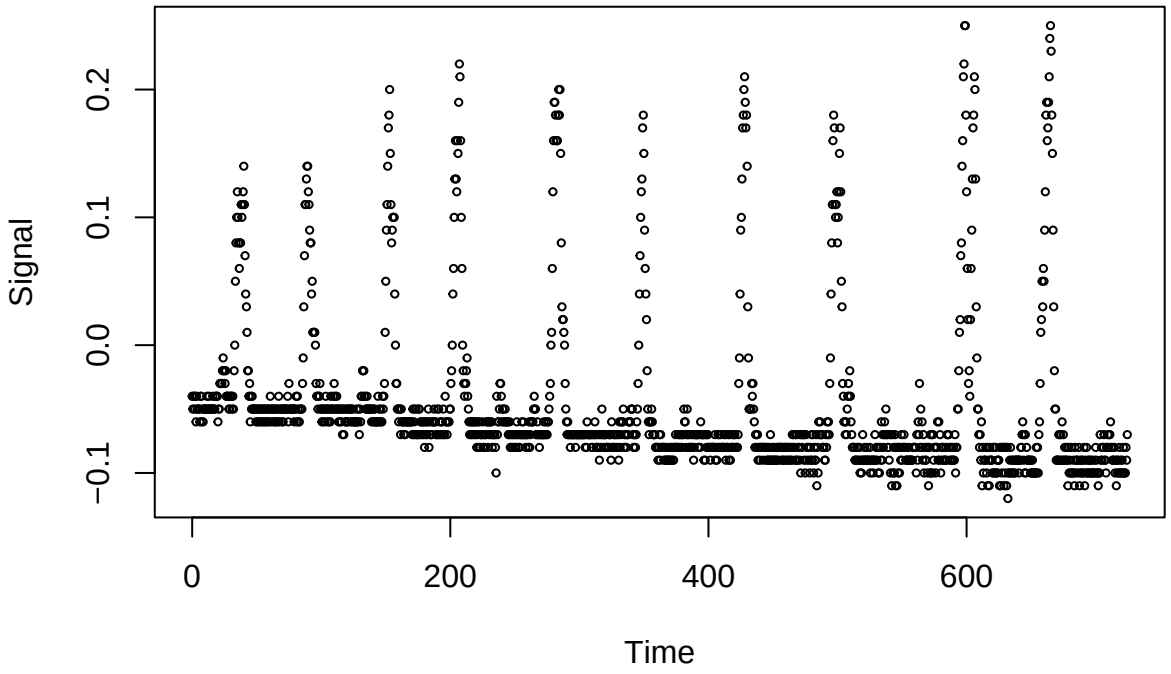
```
mu.hat.105 + qt(c(0.025, 0.975), df=34)*se.105
```

```
## [1] 0.002389834 0.042753023
```

In measuring the performance of a nonparametric regression estimator, we are looking at the variance and the bias of the estimate. Out of these two, the bias of the estimate is not accounted for in the 95% confidence intervals because we're looking at a specific number of points closest to time 200. We estimate the mean function, and thus find the variance using our results.

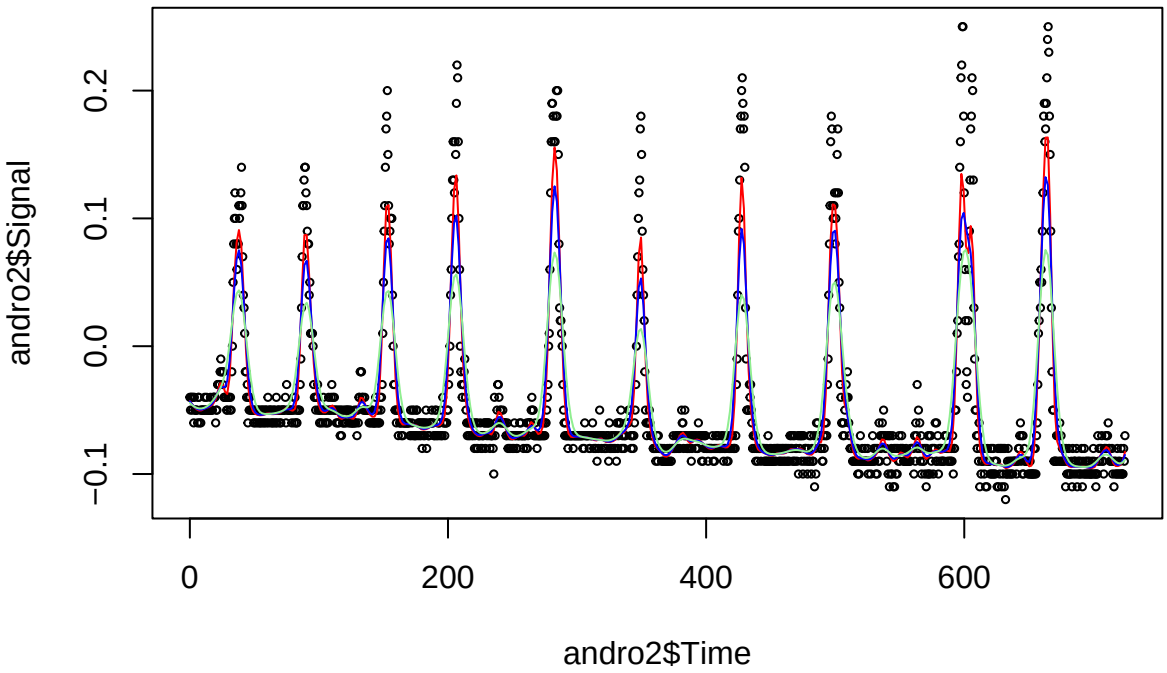
We will now plot the data with time as the x axis and signal as the y axis, calculate the Nadaraya-Watson kernel estimate using a Gaussian kernel, and plot the resulting estimate of a smooth mean function.

```
NWfit <- function(t,x,y,bw) {
  GausK <- dnorm(x, mean=t, sd=bw)
  sum(y+GausK)/sum(GausK)
}
NW.hat <- sapply(andro2$Time, NWfit, x=andro2$Time, y=andro2$Signal, bw=2)
plot(andro2$Time, andro2$Signal, cex=.5, xlab='Time', ylab='Signal')
lines(andro2$Time, NW.hat, type='l', col='red', ylab='NW')
```



Here is a new scatter plot with a local linear regression estimate.

```
plot(andro2$Time, andro2$Signal, cex=.5)
loc.fit1 <- locpoly(andro2$Time, andro2$Signal, bandwidth=2, kernel='normal', degree=1)
loc.fit2 <- locpoly(andro2$Time, andro2$Signal, bandwidth=3, kernel='normal', degree=1)
loc.fit3 <- locpoly(andro2$Time, andro2$Signal, bandwidth=5, kernel='normal', degree=1)

lines(loc.fit1$x, loc.fit1$y, type='l', col='red', ylab='Local Regression')
lines(loc.fit2$x, loc.fit2$y, type='l', col='blue', ylab='Local Regression')
lines(loc.fit3$x, loc.fit3$y, type='l', col='light green', ylab='Local Regression')
```



The scientists were looking at these measurements for evidence of a sequence of peaks in the levels happening regularly over time. We will use our estimates of the mean functions to come up with our best guess as to the number of peaks in this sequence of data.

```
m <- length(loc.fit2$y)
peaks <- (loc.fit2$y[2:(m-1)] > loc.fit2$y[1:(m-2)]) & (loc.fit2$y[2:(m-1)] > loc.fit2$y[3:m])

peaks.big <- peaks & (loc.fit2$y[2:(m-1)] > 0)
sum(peaks.big)
```

```
## [1] 10
```

```
spaces <- diff(loc.fit2$x[peaks.big])
spaces
```

```
## [1] 52.52625 63.39375 52.52625 76.07250 67.01625 77.88375 72.45000 99.61875
## [9] 63.39375
```

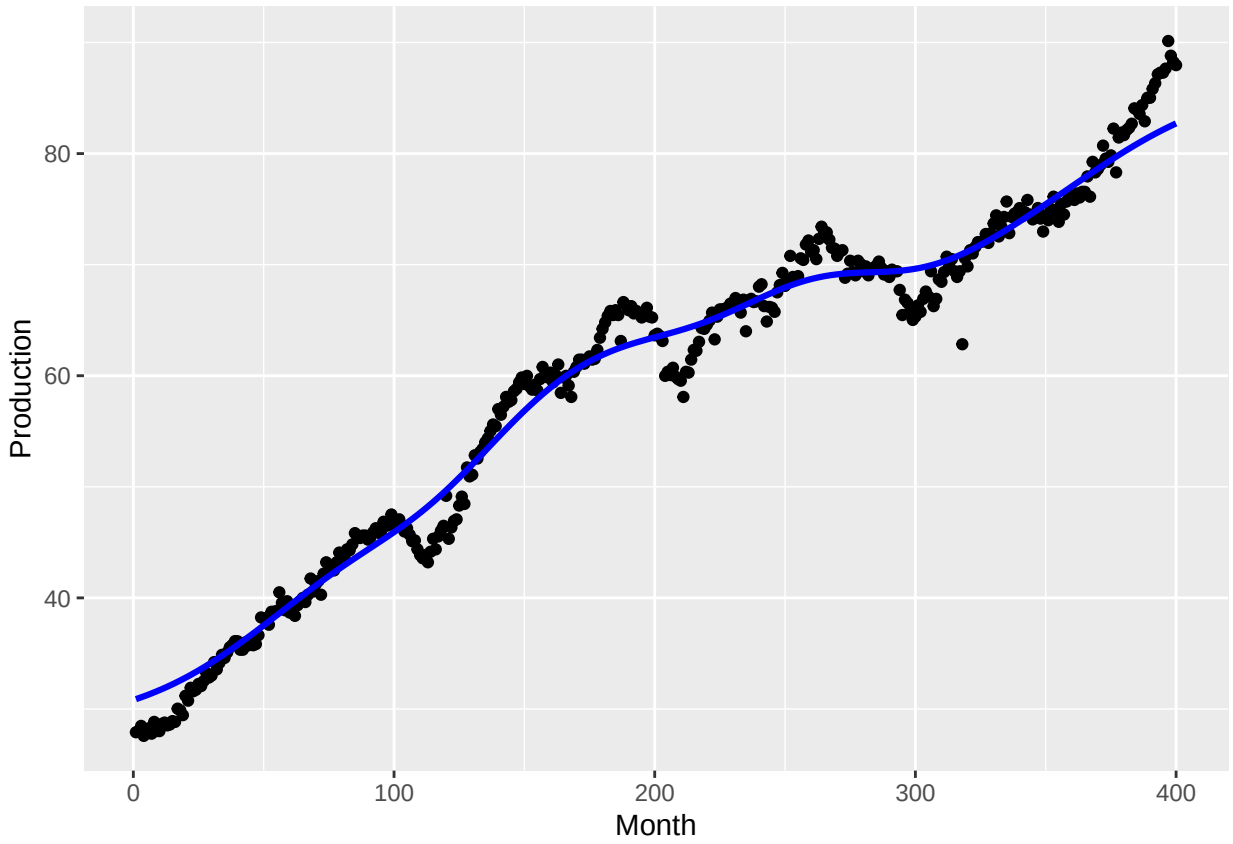
Moving onto our next data set – GermProd.txt contains monthly measurements of industrial production in Germany.

```
GermanProduction <- read.delim('GermProd.txt', sep=' ')
```

Using a Nadaraya–Watson estimator, we will fit a smooth mean to this data and plot our resulting estimate over a scatter plot of the original data series.

```
p <- ggplot(data=GermanProduction, aes(x=month, y=production)) + geom_point() + labs(x='Month', y='Produ
NW.hat <- locpoly(GermanProduction$month, GermanProduction$production, bandwidth=24, degree=0)
p + annotate(geom='line', NW.hat$x, NW.hat$y, color='blue', size=1.1)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
```

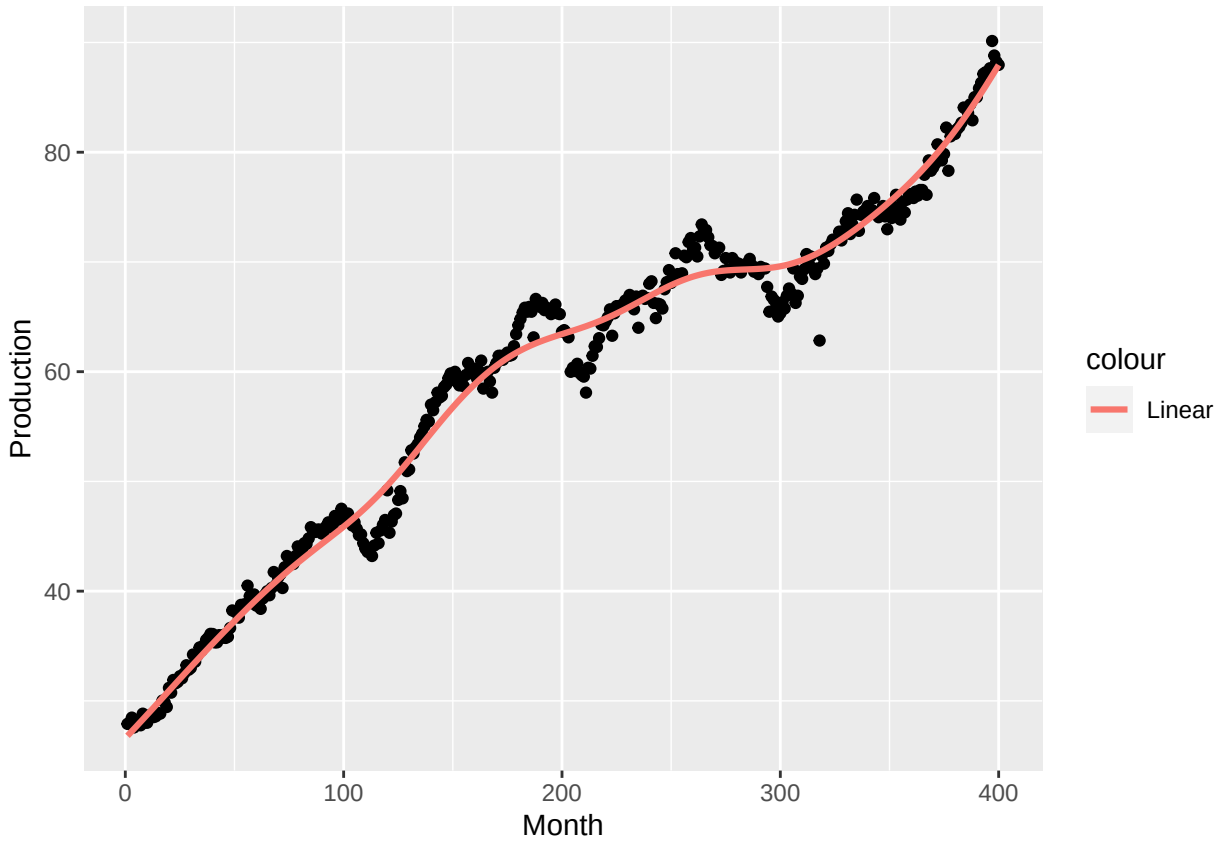


At the boundary, the plots continue to increase at a faster rate, but the line does not depict that as accurately.

Now, we will refit the data using a local-linear model and produce a plot which compares the estimator from the local-linear estimator and the N–W estimator.

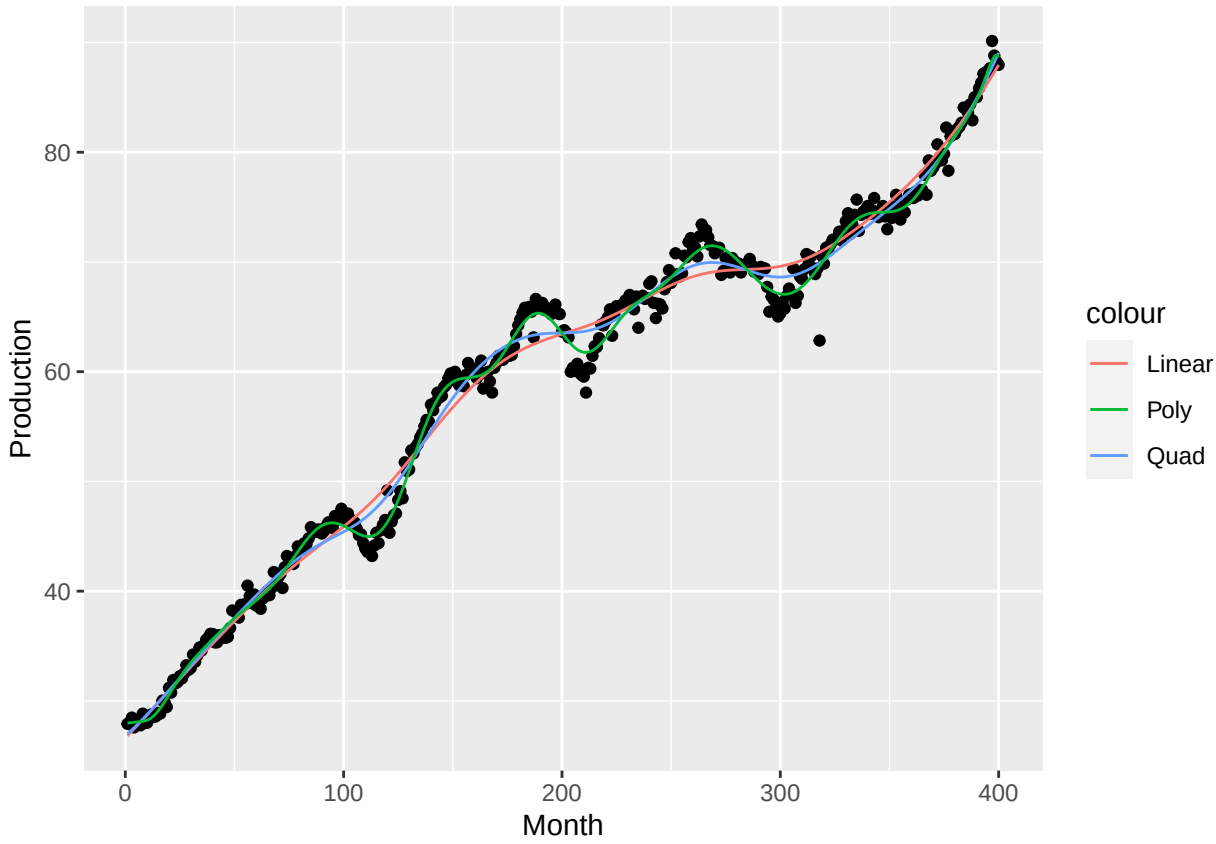
```
loc.fit1 <- locpoly(GermanProduction$month, GermanProduction$production, bandwidth=24, kernel='normal',
loc.fit2 <- locpoly(GermanProduction$month, GermanProduction$production, bandwidth=24, kernel='normal',
p + geom_line(aes(GermanProduction$month, loc.fit1$y[1:400], color='Linear'), size=1.1)
```

```
## Warning: Use of `GermanProduction$month` is discouraged.
## i Use `month` instead.
```




```
loc.fit10 <- locpoly(GermanProduction$month, GermanProduction$production, bandwidth=24, kernel='normal'
p + geom_line(aes(GermanProduction$month, loc.fit1$y[1:400], color='Linear')) +
  geom_line(aes(GermanProduction$month, loc.fit2$y[1:400], color='Quad')) +
  geom_line(aes(GermanProduction$month, loc.fit10$y[1:400], color='Poly'))
```

```
## Warning: Use of `GermanProduction$month` is discouraged.
## i Use `month` instead.
## Use of `GermanProduction$month` is discouraged.
## i Use `month` instead.
## Use of `GermanProduction$month` is discouraged.
## i Use `month` instead.
```



This adjusts for the boundary problem by adding different types of lines that follow the overall pattern of the plots.

Using the approval ratings data for George W. Bush, we would like to estimate his approval rating on June 15, 2005 and October 1, 2001.

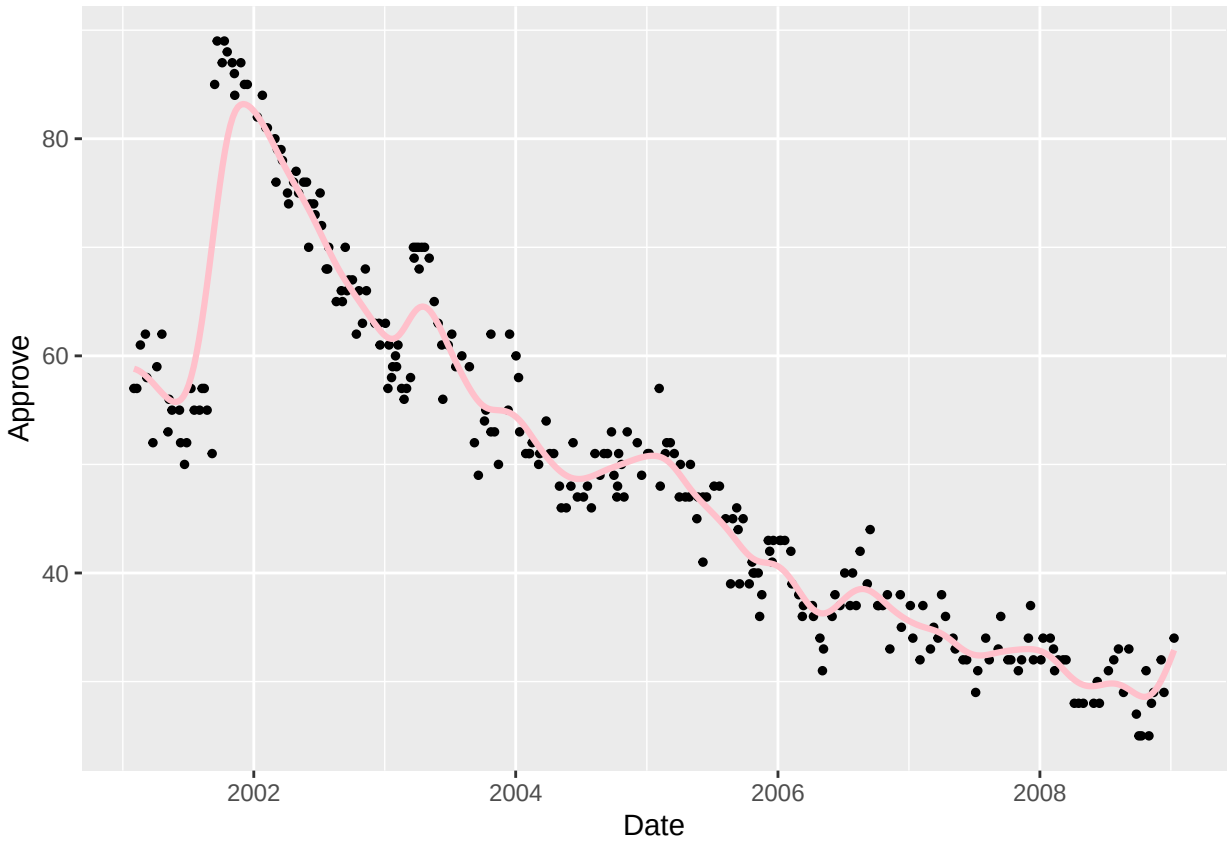
```
WBushApproval <- read.csv('WBushApproval.csv')
```

Here, we will plot the approval ratings and a local-linear regression estimator.

```
WBushApproval$Date <- as.Date(WBushApproval$Start, format='%m/%d/%Y')
WBushApproval <- mutate(WBushApproval, Real.Date=as.numeric(WBushApproval$Date-WBushApproval$Date[1]))

WBushApproval$Real.Date[1:5]
```

```
## [1]  0  8 18 32 36
```

```
loc.linear <- locpoly(WBushApproval$Real.Date, WBushApproval$Approve, bandwidth=50, degree=1)
loc.linear$dates <- WBushApproval$Date[1] + loc.linear$x

bush.plot <- ggplot(WBushApproval, aes(x=Date, y=Approve)) + geom_point(size=1)

bush.plot + annotate(geom='line', x=loc.linear$dates, y=loc.linear$y, color='pink', size=1.1)
```



Now, we will use the results to find the estimated value on June 15, 2005.

```
june15 <- as.numeric(as.Date('2005-06-15')-WBushApproval$Date[1])

approx(loc.linear$x, loc.linear$y, xout=june15)
```

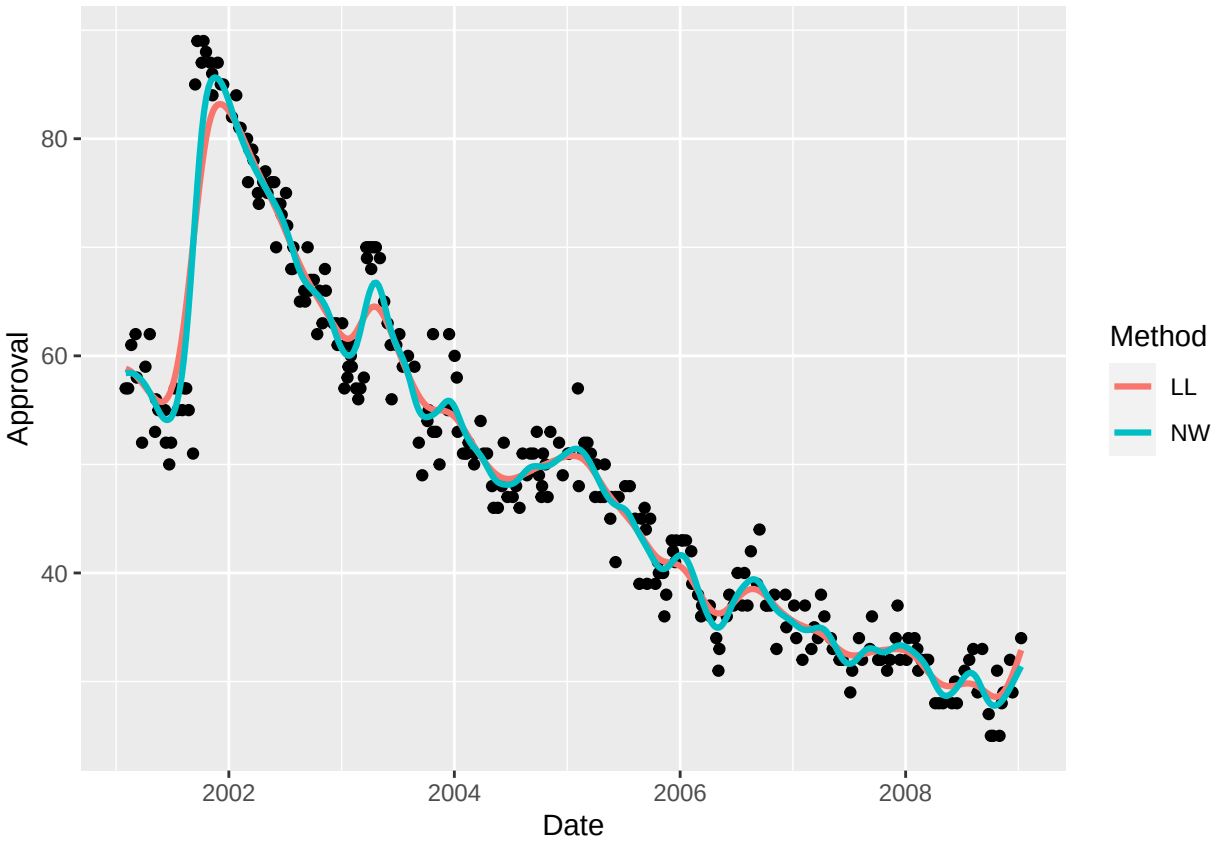
```
## $x
## [1] 1595
##
## $y
## [1] 46.11322
```

We will fit a new Nadaraya–Watson kernel regression estimator and compare the estimate on October 1, 2001 from the kernel and local-linear estimates.

```
oct01 <- as.numeric(as.Date('2001-10-01') - WBushApproval$Date[1])
approx(loc.linear$x, loc.linear$y, xout=oct01)$y
```

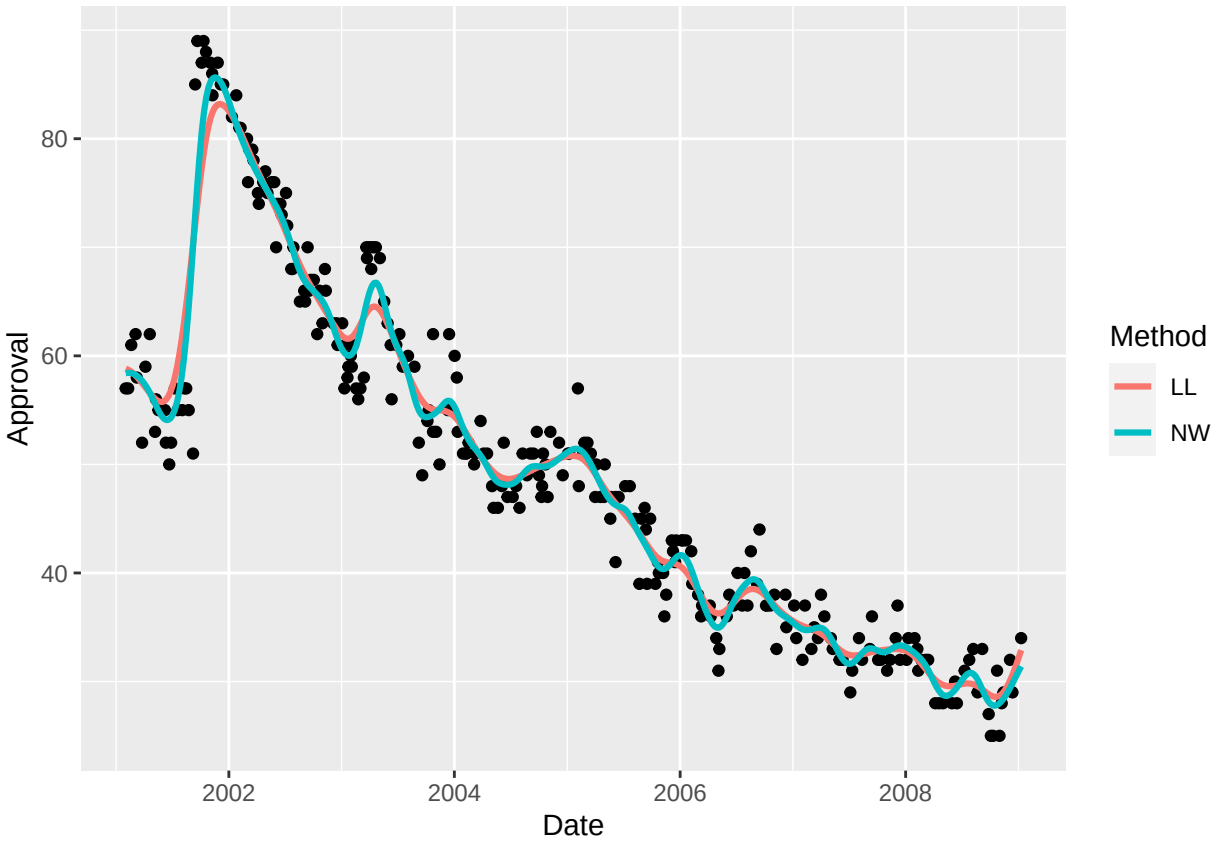
```
## [1] 76.37505
```

```
bush.NW <- locpoly(x=WBushApproval$Real.Date, y=WBushApproval$Approve, bandwidth=30,
                   degree=0, range.x=c(0,max(WBushApproval$Real.Date)))

bush.LL.NW <- bind_rows(LL = loc.linear, NW = bush.NW, .id='Method')
bush.LL.NW$Days <- WBushApproval$Date[1] + bush.LL.NW$x
ggplot(bush.LL.NW, aes(x=Days, y=y)) + annotate(geom = 'point', x=WBushApproval$Date,
                                                y=WBushApproval$Approve, color='black') +
  geom_line(aes(color=Method), size=1.1) +labs(x='Date', y='Approval')
```



Now, we plot the kernel regression estimator over the scatter plot of the approval ratings.

```
ggplot(bush.LL.NW, aes(x=Days, y=y)) + annotate(geom='point', x=WBushApproval$Date,
  y=WBushApproval$Approve, color='black') + geom_line(aes(color=Method), size=1.1) +
  labs(x='Date', y='Approval')
```



Potential issues and solutions:

The discontinuity in this data at Sept 11, 2001 does not align with a typical linear regression because our estimators are meant to estimate regression. Therefore, the estimators underestimate the true peak of approval in 2002, and thus we can see a bias. To mitigate this effect, we can decrease the bandwidth so that the estimator can closely follow the data set.