STAT 680 Assignment 2

Sukhdeep Singh

May 27, 2018

Soulution 1) Reading the file

```
ecg=read.table('ECG.dat', header = T)
library(KernSmooth)
```

```
## Warning: package 'KernSmooth' was built under R version 3.4.4
```

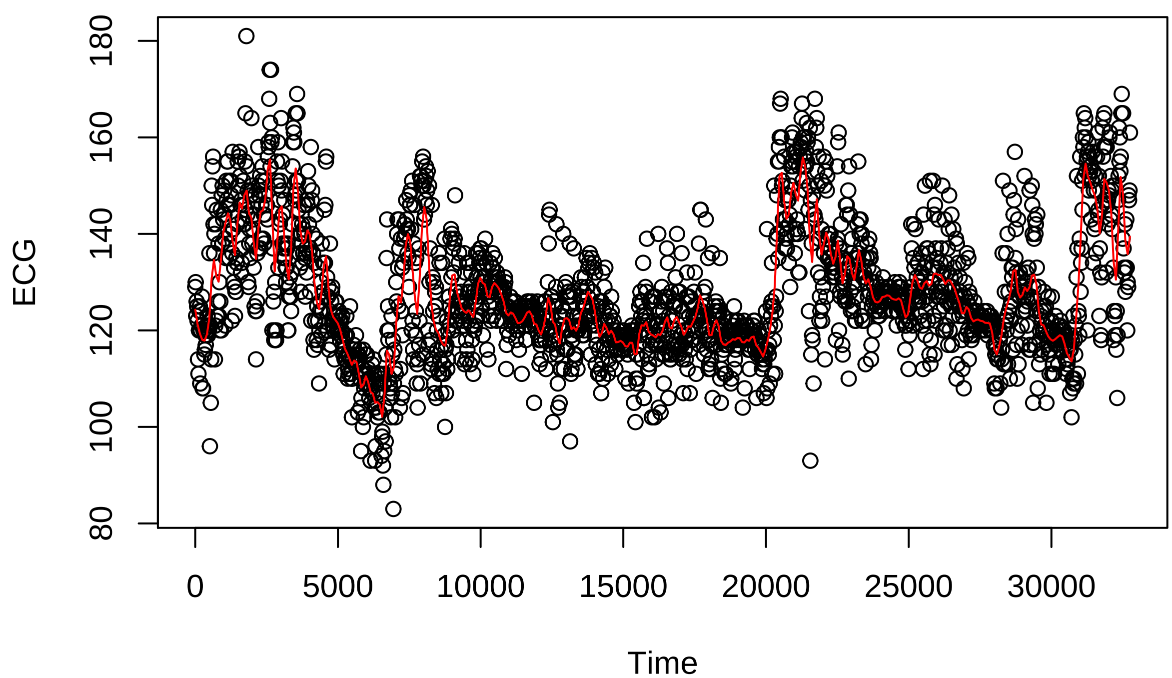
```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

Nadaraya-Watson local constant estimator, with bandwidth h_1=64

```
localpoint_64=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 0,kernel = "normal",bandwidth = 64))
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_64,lines(x,y,col = 2))
```



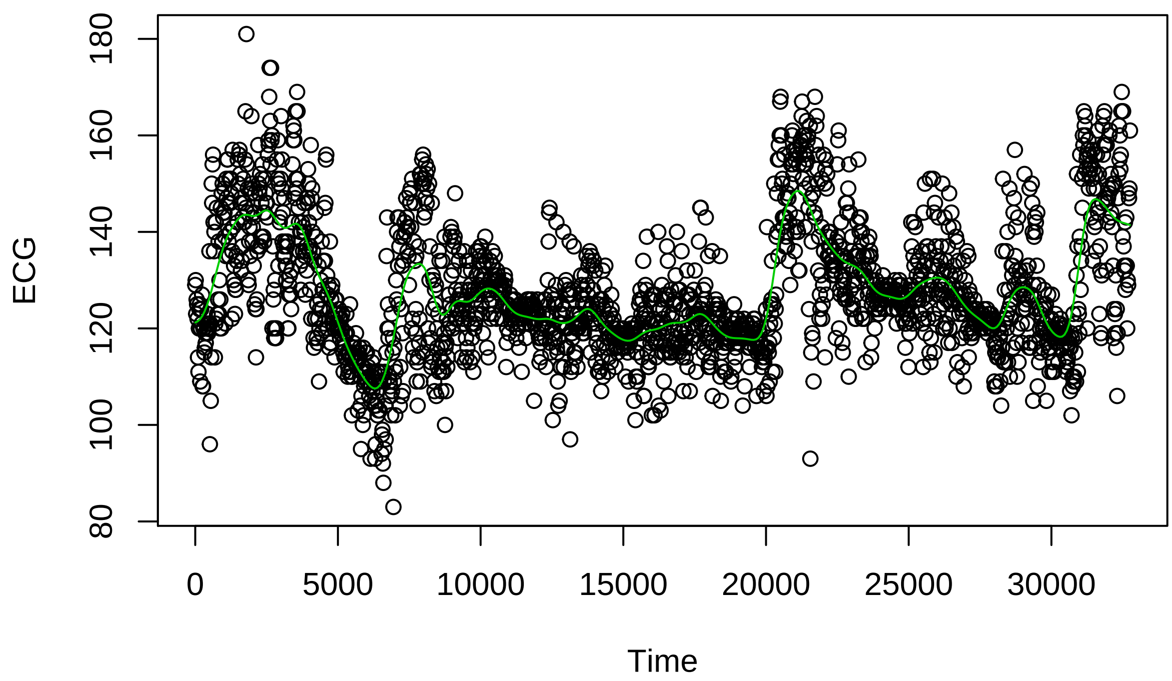
The estimate is very rough as we see in above plot, jagged plot lines indicate the bandwidth size of H_1=64 is too small

Nadaraya-Watson local constant estimator, with bandwidth h_2=304

```
localpoint_304=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 0,kernel = "normal",bandwidth = 304))
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_304,lines(x,y,col = 3))
```



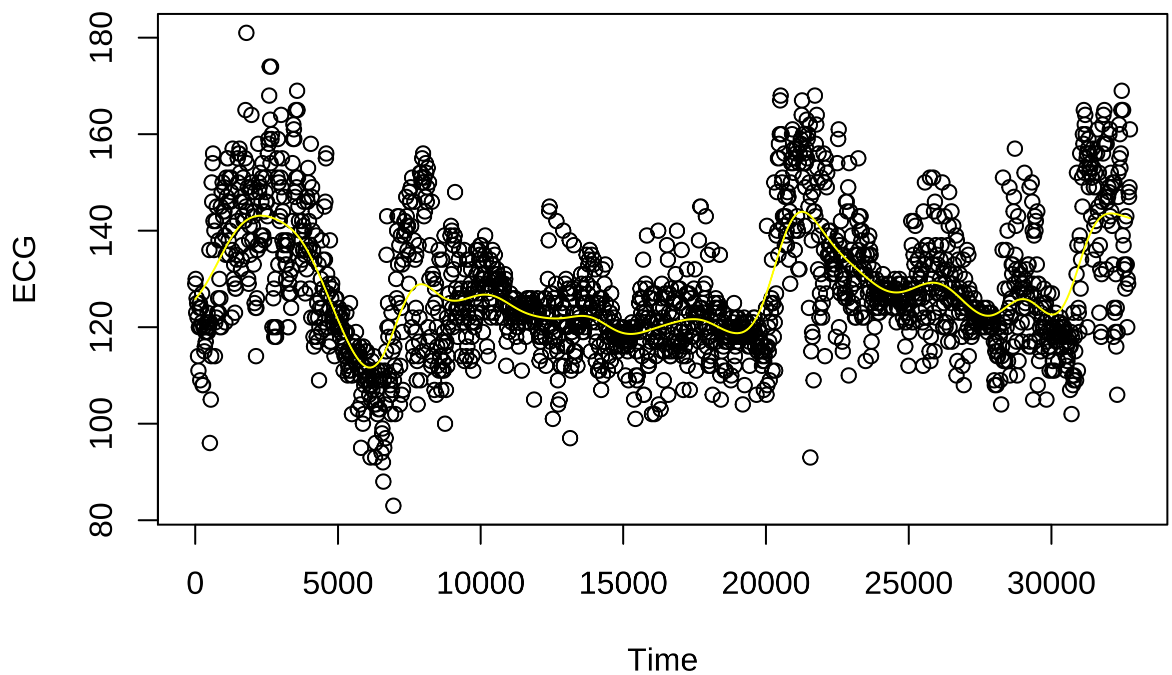
We can see that increasing the bandwidth to 304 has smoothed out the jagged lines seen in previous estimate, therefore the bandwidth selection of h_2=304 has improved the estimate.

Nadaraya-Watson local constant estimator, with bandwidth h_3=608

```
localpoint_608=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 0,kernel = "normal",bandwidth = 608))
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_608,lines(x,y,col = 7))
```



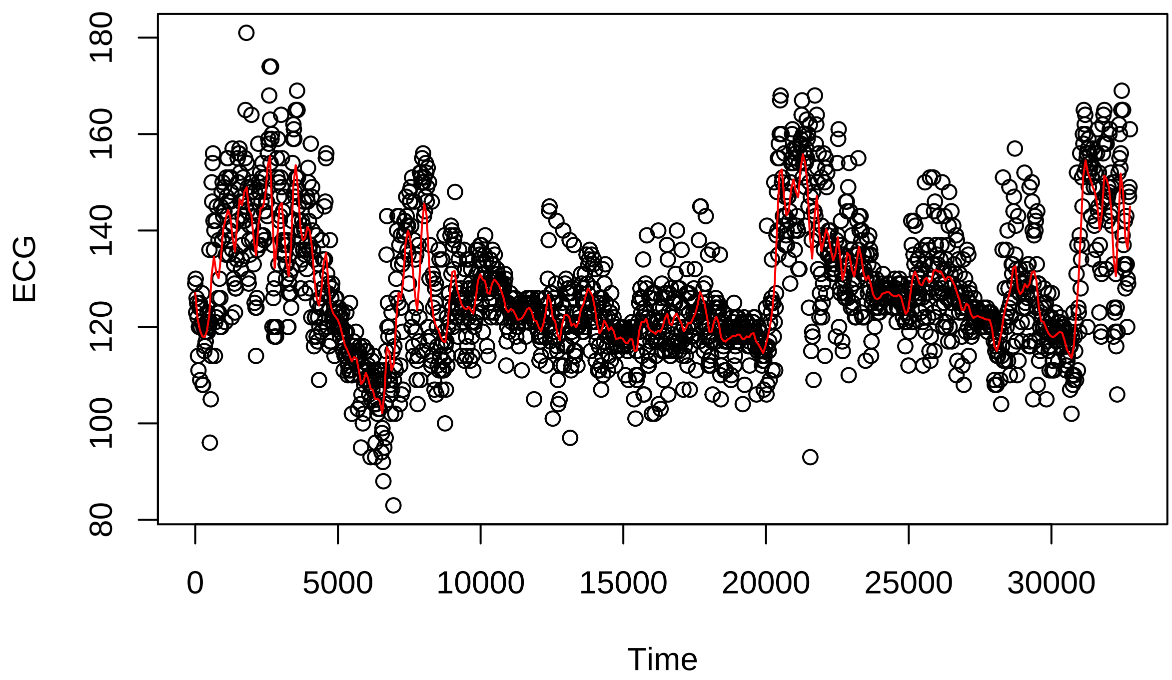
We can infer from the above plot that increasing the bandwidth size to 608 causes oversmoothning of the linear estimate and information is being lost.

Nadaraya-Watson local linear estimator, with bandwidth h_1=64

```
localpoint_64=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 1,kernel = "normal",bandwidth = 64))
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_64,lines(x,y,col = 2))
```



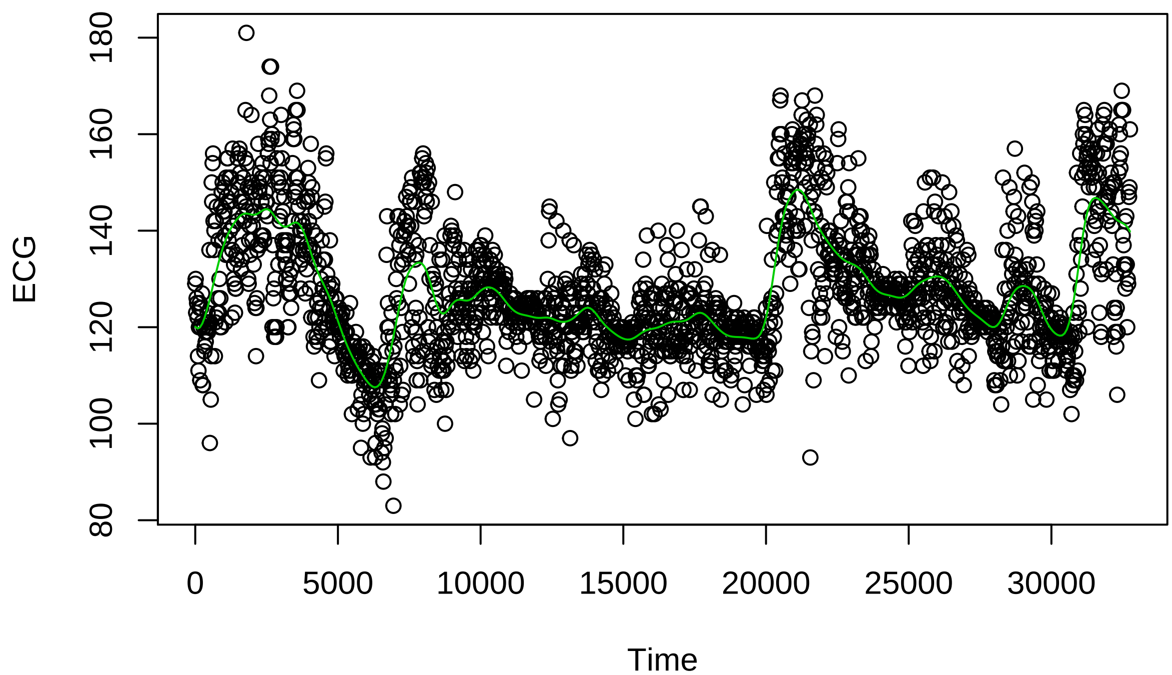
The linear estimate is very rough as we see in above plot, jagged plot lines indicate the bandwidth size of H_1=64 is too small

Nadaraya-Watson linear linear estimator, with bandwidth h_2=304

```
localpoint_304=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 1,kernel = "normal",bandwidth = 304))
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_304,lines(x,y,col = 3))
```



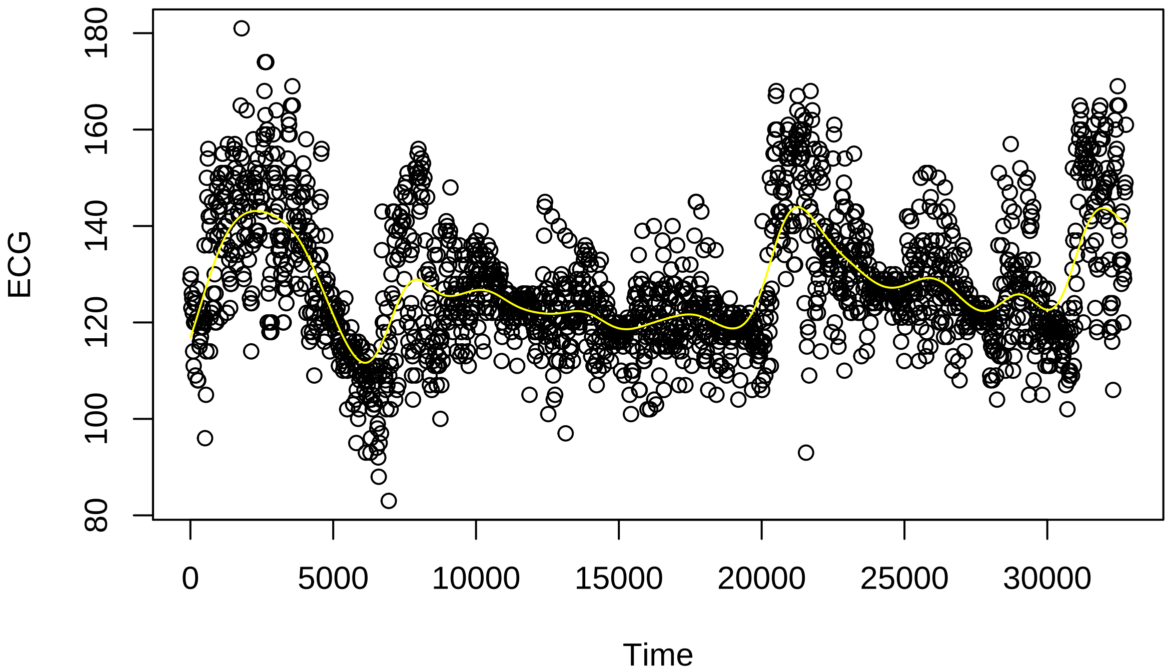
We can see that increasing the bandwidth to 304 has smoothed out the jagged lines seen in previous estimate, therefore the bandwidth selection of h_2=304 has improved the estimate.

Nadaraya-Watson local linear estimator, with bandwidth h_3=608

```
localpoint_608=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 1,kernel = "normal",bandwidth = 608))
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_608,lines(x,y,col = 7))
```



We can infer from the above plot that increasing the bandwidth size to 608 causes oversmoothning of the linear estimate and information is being lost.

Selecting appropriate bandwidth size using the inbuilt function

```
idealbandwidth=dpill(ecg$Time, ecg$ECG)

print(idealbandwidth)
```

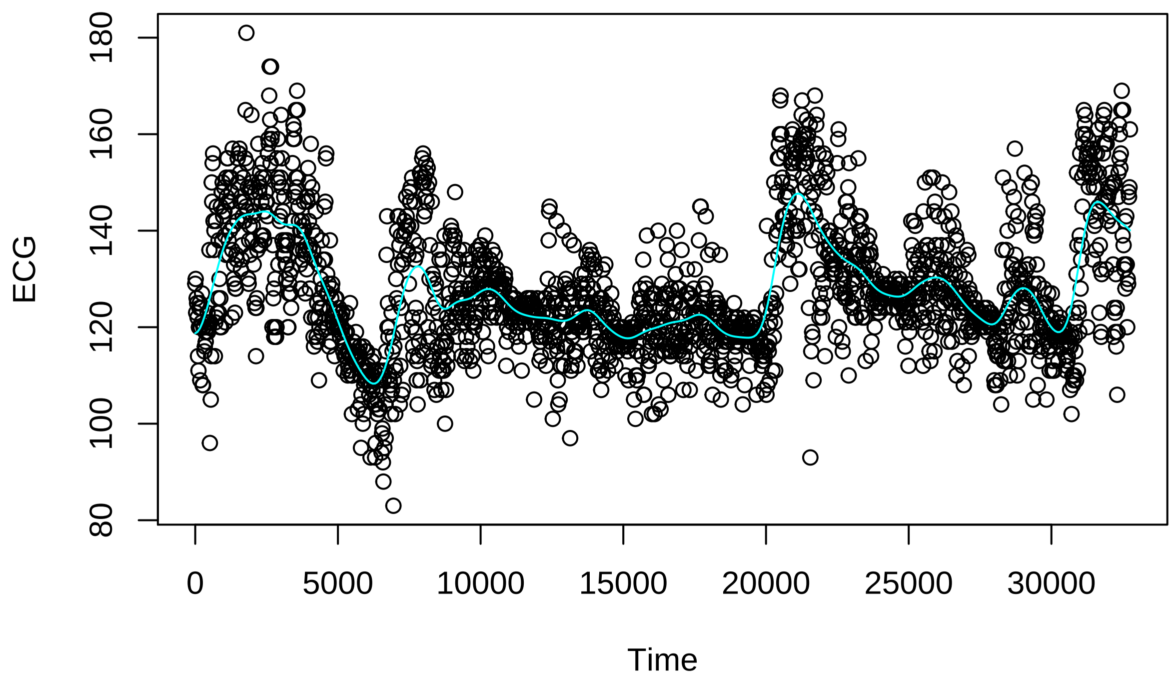
```
## [1] 361.8178
```

Nadaraya-Watson local linear estimator, with ideal bandwidth h_4=361.8178

```
localpoint_ideal=with(ecg,locpoly(ecg$Time, y = ecg$ECG,degree = 1,kernel = "normal",bandwidth = 361.81
```

Checking the performance of the model

```
plot(ecg$ECG ~ ecg$Time,xlab="Time",ylab = "ECG")

with(localpoint_ideal,lines(x,y,col = 5))
```



We can see from the above plot tha adjusting the bandwidth to the ideal one calculated using the dpill function results in a model which retains all the information and is smooth at the same time with no jagged or rough lines. Comparing the performance of this model with the previous models with bandwitdth H_1,H_2,H_3 we see a huge improvement.

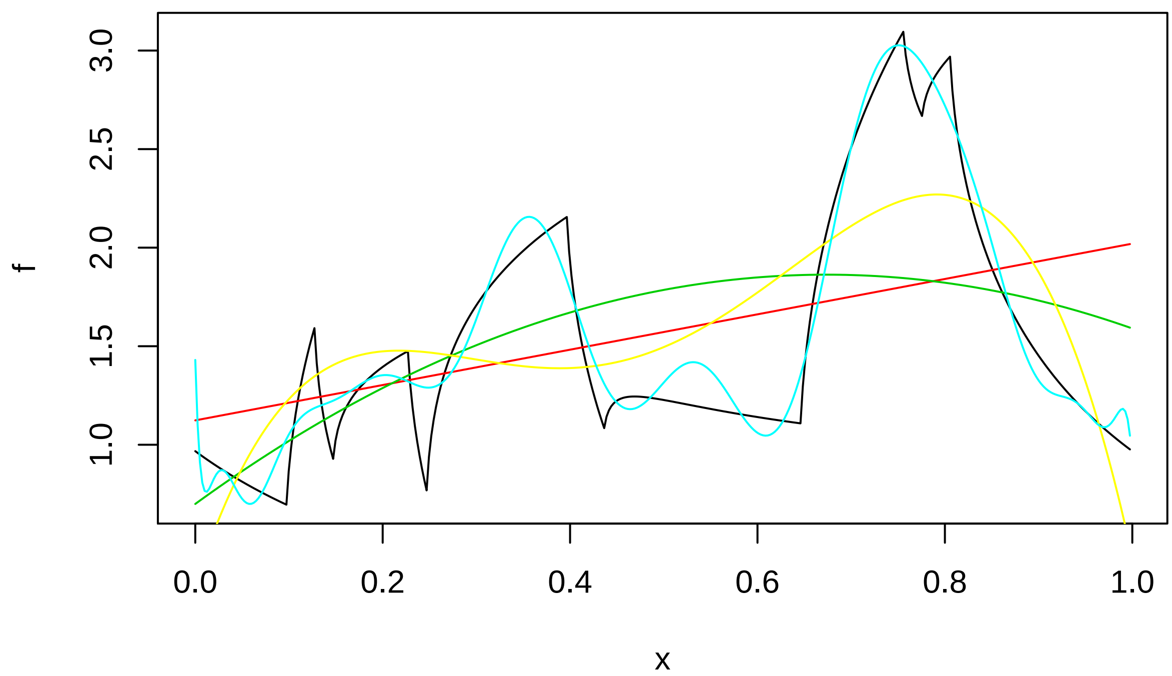
Question 2: Reading the data

```
simdata=read.table('sim.dat',header = T)
```

Fitting all the models to the data

```
linearfit=lm(y ~ x,data = simdata)

quadraticfit=lm(y ~ poly(x,2), data = simdata)

quarticfit = lm(y ~ poly(x,4), data = simdata)

higher_order_polyfit = lm(y ~ poly(x,20), data = simdata)

x = seq(from = min(simdata$x), to = max(simdata$x), length = 401)

predx= data.frame(x = x)


plot(f ~ x, data = simdata,type='l')

lines(x,predict(linearfit), col=2)
```

```
lines(x,predict(quadraticfit), col=3)

lines(x,predict(quarticfit), col=7)


lines(x,predict(higher_order_polyfit), col=5)
```



- Black line indicates true function (f)
- Red line indicates linear estimated function
- Green line indicates quadratic estimated function
- Yellow line indicates quartic estimated function
- Blue line indicates polynomial degree 20 estimated function

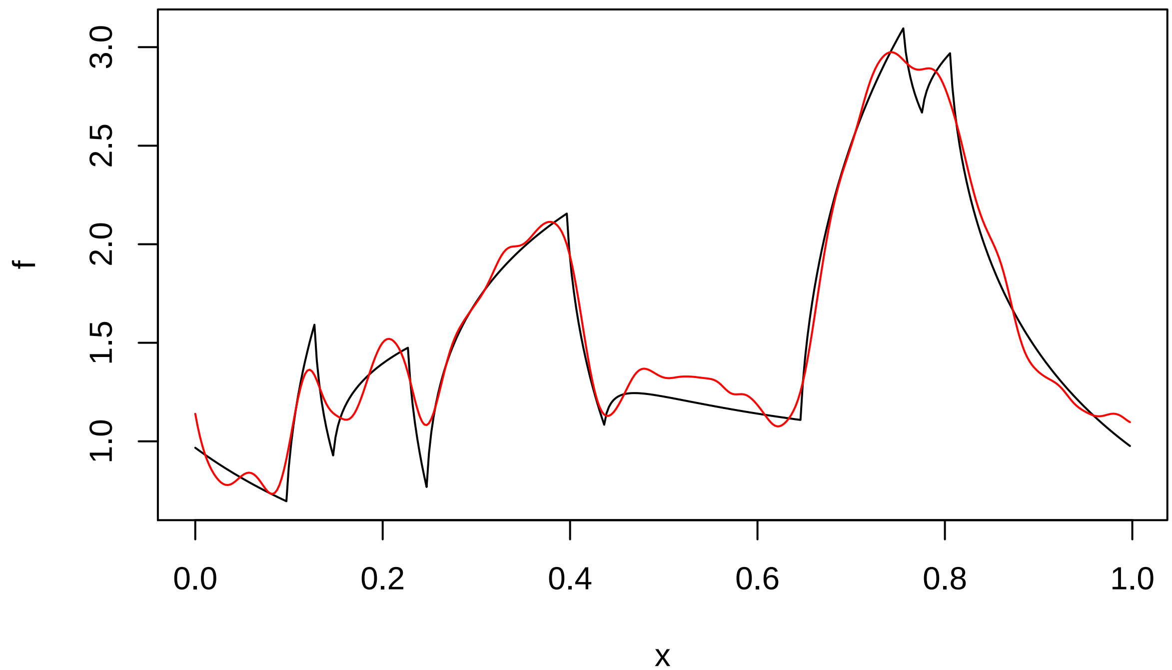
Calculating the ideal windowsize for performing the norparametric regression (local linear) NW estimate

```
ideal=dpill(simdata$x ,simdata$f+simdata$y)
print(ideal)
```

```
## [1] 0.01180527
```

Trying a NW estimator function with bandwidth size of 0.01180527

```
simulation_local_linear = with(simdata,locpoly(x,y,degree = 1,kernel = "normal", bandwidth = 0.01180527)

plot(f~x,data = simdata,type= 'l')

with(simulation_local_linear,lines(x,y,col=2))
```



- Black line indicates true function (f)
- Red line indicates estimated function (f_cap)

As we can see the non-parametric local linear with ideal bandwidth gives a relatively better fit as compared to the linear, quadratic, quartic and higher order polynomial.

C) If we intend to capture the sharp points of the function, we will need a local linear estimate with smaller bandwidth, the resulting estimate would be very rough and not suitable for drawing insights from the plot.

D) ASE for linear estimate

```
linearpredict=predict(linearfit,predx)

mean((simdata$f-linearpredict)^2)
```

```
## [1] 0.2981107
```

ASE for higher order polynomial estimate

```
polyprediction=predict(higher_order_polyfit,predx)
mean((simdata$f- polyprediction)^2)
```

```
## [1] 0.01830507
```

ASE for NW local linear estimate

```
mean((simdata$f - simulation_local_linear$y)^2)
```

```
## [1] 0.007441935
```

We can see that non parametric NW estimate is closest to the true function.