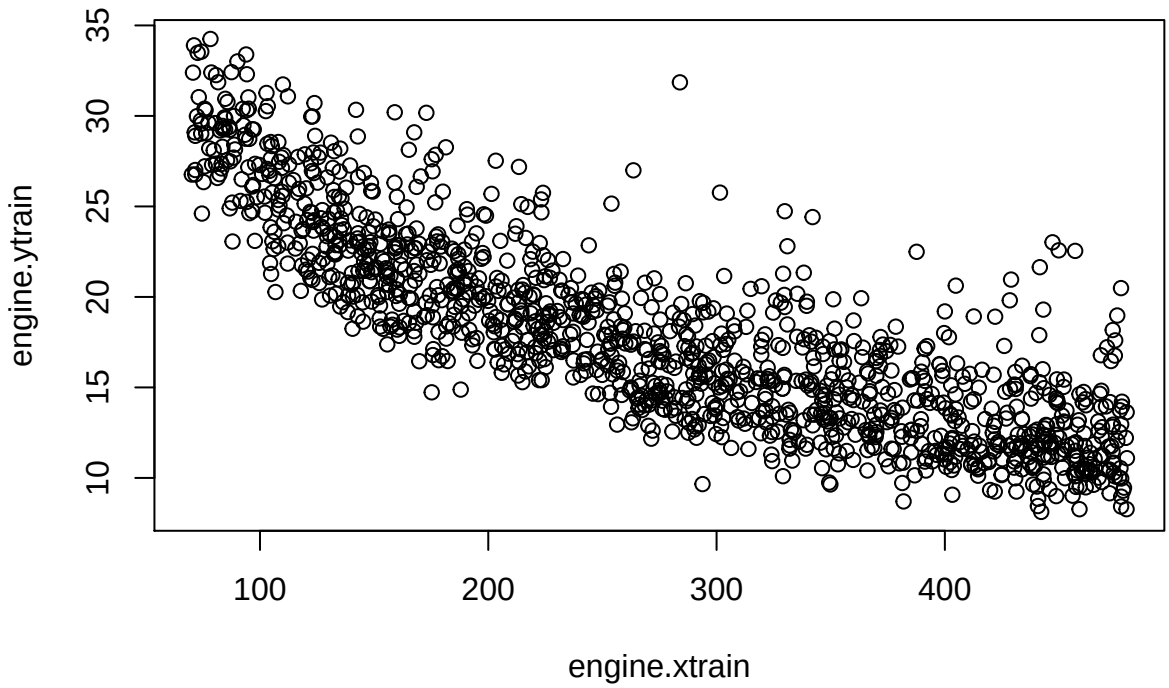
---
title: "36-402 Homework 1"
author: "Jacky Liu"
date: "2/12/21"
output: pdf_document

```
load("engine.Rdata")
```

Problem 1

Problem 1 (a)

```
plot(engine.xtrain, engine.ytrain)
```



Problem 1 (b)

```
library(np)
```

```
## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-10)
## [vignette("np_faq",package="np") provides answers to frequently asked questions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]
```

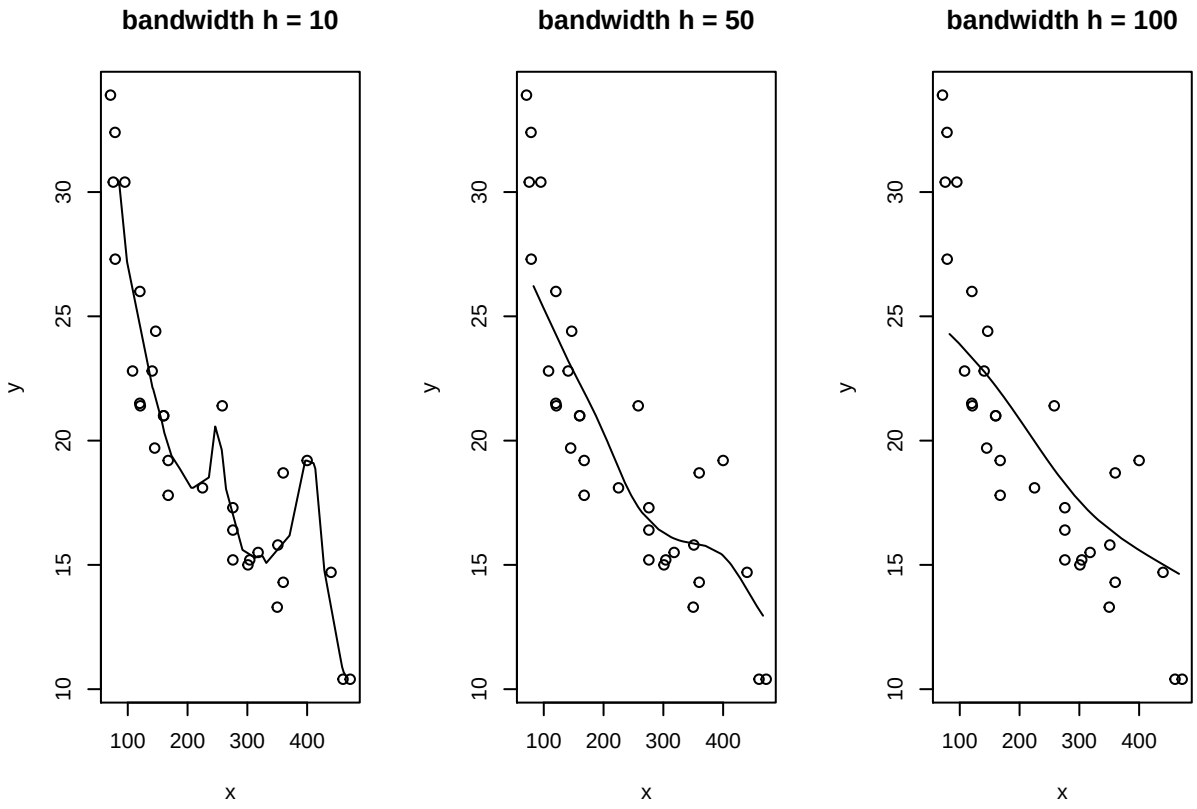
```
bws = c(10,50,100)
first = data.frame(x = engine.xtrain[,1], y = engine.ytrain[,1])
x = engine.xtrain[,1]
y = engine.ytrain[,1]
kregobj10 <- npreg(y ~ x, data=first, bws=bws[1])
kregobj50 <- npreg(y ~ x, data=first, bws=bws[2])
kregobj100 <- npreg(y ~ x, data=first, bws=bws[3])

par(mfrow = c(1,3))

plot(x,y, main = "bandwidth h = 10")
pred10 = predict(kregobj10, newdata = data.frame(x = engine.xtest[,1]))
lines(engine.xtest[,1], pred10)

plot(x,y, main = "bandwidth h = 50")
pred50 = predict(kregobj50, newdata = data.frame(x = engine.xtest[,1]))
lines(engine.xtest[,1], pred50)

plot(x,y, main = "bandwidth h = 100")
pred100 = predict(kregobj100, newdata = data.frame(x = engine.xtest[,1]))
lines(engine.xtest[,1], pred100)
```



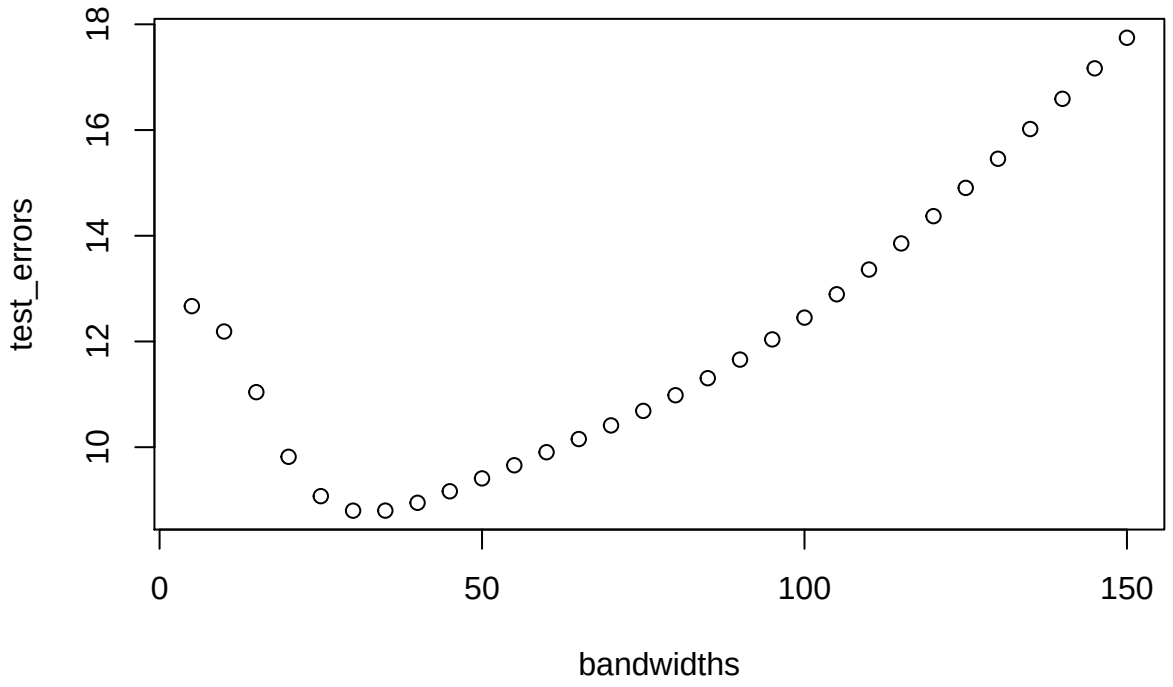
Problem 1 (c)

By inspection, the kernel regression fit changes as we vary the bandwidth. The procedure this reminds me of is knn regression. As we increase the badnwidth parameter, the line becomes smoother and approaches underfitting. Again, this reminds me of knn regression, where a larger k value creates a smoother line that

captures less points.

Problem 1 (d)

```
first = data.frame(x = engine.xtrain[,1], y = engine.ytrain[,1])
x = engine.xtrain[,1]
y = engine.ytrain[,1]
bandwidths = seq(from=5, to=150, by=5)
test_errors = rep(NA, 30)
counter = 1
for(i in bandwidths){
  kregobj = npreg(y ~ x, data=first, bws=i)
  test_error = mean((engine.ytest[,1] - predict(kregobj, newdata = data.frame(x = engine.xtest[,1])))^2)
  test_errors[counter] = test_error
  counter = counter + 1
}
plot(bandwidths, test_errors)
```



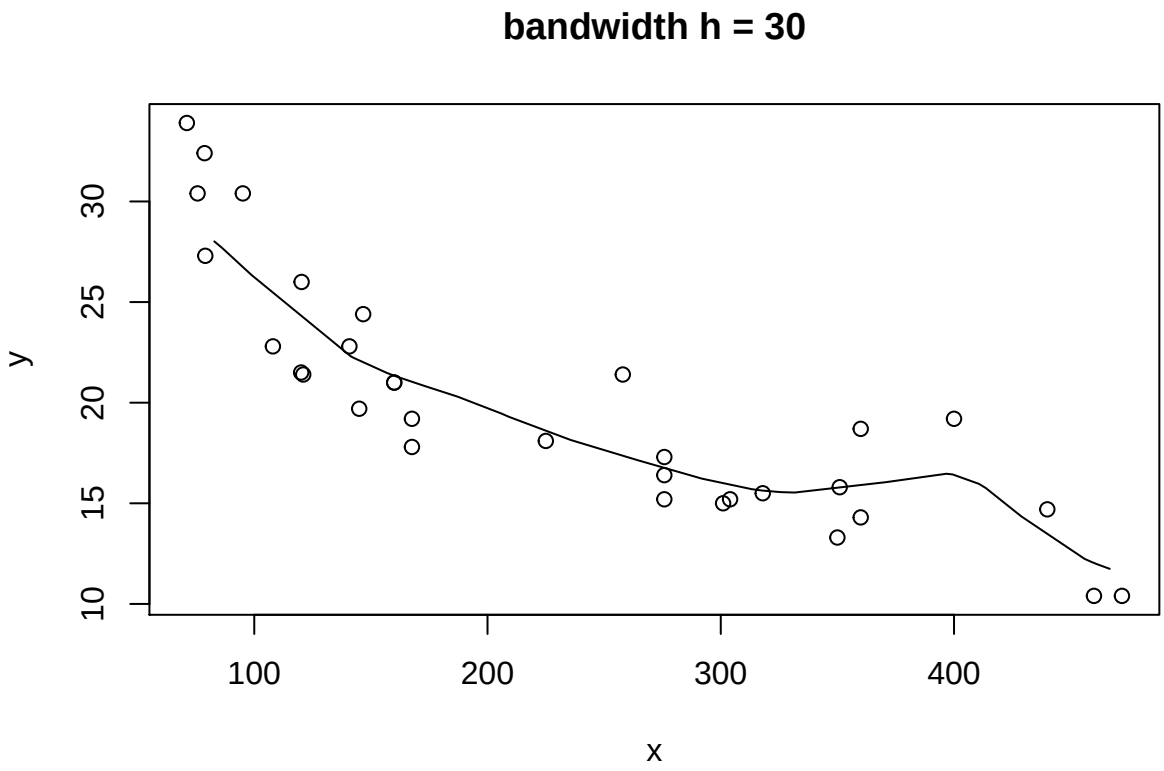

```
print(test_errors[30/5])
```

```
## [1] 8.799357
```

Problem 1 (e)

According to the test error curve above, the optimal bandwidth value is h = 30 with a test error of 8.799357.

```
first = data.frame(x = engine.xtrain[,1], y = engine.ytrain[,1])
x = engine.xtrain[,1]
y = engine.ytrain[,1]
kregobj30 <- npreg(y ~ x, data=first, bws=30)
plot(x,y, main = "bandwidth h = 30")
pred30 = predict(kregobj30, newdata = data.frame(x = engine.xtest[,1]))
lines(engine.xtest[,1], pred30)
```

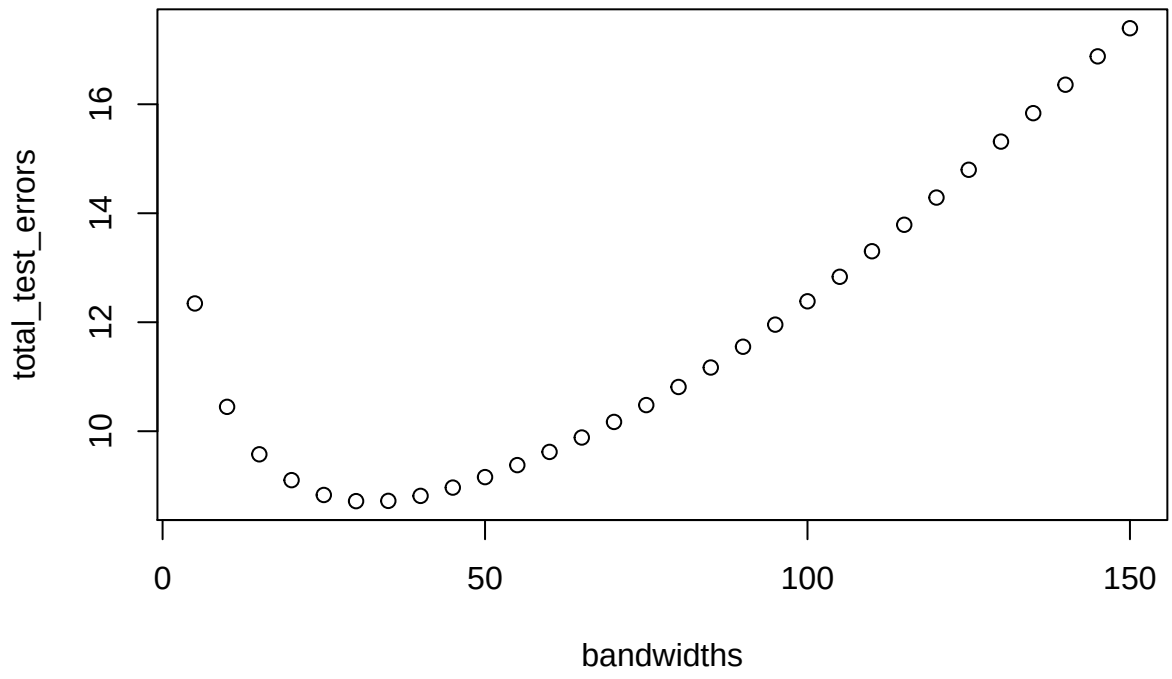


By looking at the plot, this seems to be a good bandwidth value because the line seems to fit the general trend of points without overfitting or underfitting too much.

Problem 1 (f)

```
total_test_errors = rep(0, 30)
bandwidths = seq(from=5, to=150, by=5)
for(j in 1:40) {
  first = data.frame(x = engine.xtrain[,j], y = engine.ytrain[,j])
  x = engine.xtrain[,j]
  y = engine.ytrain[,j]
  test_errors = rep(NA, 30)
```

```
  counter = 1
  for(i in bandwidths){
    kregobj = npreg(y ~ x, data=first, bws=i)
    test_error = mean((engine.ytest[,j] - predict(kregobj, newdata = data.frame(x = engine.xtest[,j])))
    test_errors[counter] = test_error
    counter = counter + 1
  }
  total_test_errors = total_test_errors + test_errors
}
total_test_errors = total_test_errors/40
plot(bandwidths, total_test_errors)
```



Now, I see a curve that is a similar shape to the previous one but more smooth. The optimal value of bandwidth is still 30, but the associated test error at 30 has changed to 8.716932 from 8.799357.