---
title: "36-402 Homework 5"
author: "Jacky Liu"
date: "2/12/21"
output: pdf_document
---

# Problem 1

## Problem 1 (a)

```
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]
```

```
housetrain = read.csv("housetrain.csv", sep=",")
housetest = read.csv("housetest.csv", sep=",")
housedata = rbind(housetrain, housetest)
samp <- sample(rep(1:5, 2121), replace = FALSE)
model3_errors = vector(length=5)
model4_errors = vector(length=5)
model5_errors = vector(length=5)
model6_errors = vector(length=5)


for (k in 1:5) {
  testd <- housedata[samp == k, ]
  traind <- housedata[!(samp == k), ]
  n=length(housedata)

  # model 3
  model3 = lm(Median_house_value ~ Median_household_income + Mean_household_income, data=traind)
  model3_preds_train = predict(model3, traind)
  model3_preds = predict(model3, testd)
  model3_err = mean((model3_preds - model3_preds_train)^2)
  model3_errors[k] = model3_err

  # model 4
  model4 = lm(Median_house_value ~ Median_household_income + Mean_household_income + Latitude + Longitu
  model4_preds_train = predict(model4, traind)
  model4_preds = predict(model4, testd)
  model4_err = mean((model4_preds - model4_preds_train)^2)
  model4_errors[k] = model4_err
  # model 5
  model5 <- npreg(Median_house_value ~ Median_household_income + Mean_household_income,
                  data = traind, newdata = testd,
                  bws = apply(traind[, c(5,6)], 2, sd) / n^(0.2))
  model5_err = (sum(model5$mean) / length(model5$mean))^2
  model5_errors[k] = model5_err
```

```
    # model 6
  model6<-npreg(Median_house_value ~ Mean_household_income + Median_household_income + Latitude + Longi
                data= traind, newdata = testd,
                bws = apply(traind[, c(2,3,5,6)], 2, sd) / n^(0.2))
  model6_err = (sum(model6$mean) / length(model6$mean))^2
  model6_errors[k] = model6_err

}

values = t(matrix(c(model3_errors,model4_errors,model5_errors,model6_errors), nrow=4, ncol=5))
values
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,]  42196.11  42837.20  42757.32  44128.74
## [2,]  41565.00  57419.94  57487.65  57999.90
## [3,]  60127.75  57279.16 117182.21 114007.70
## [4,] 114063.69 115398.92 112187.19  10802.71
## [5,]  10700.18  11253.20  10573.16  10454.75
```

### Problem 1 (b)

```
apply(values, 2, mean)
```

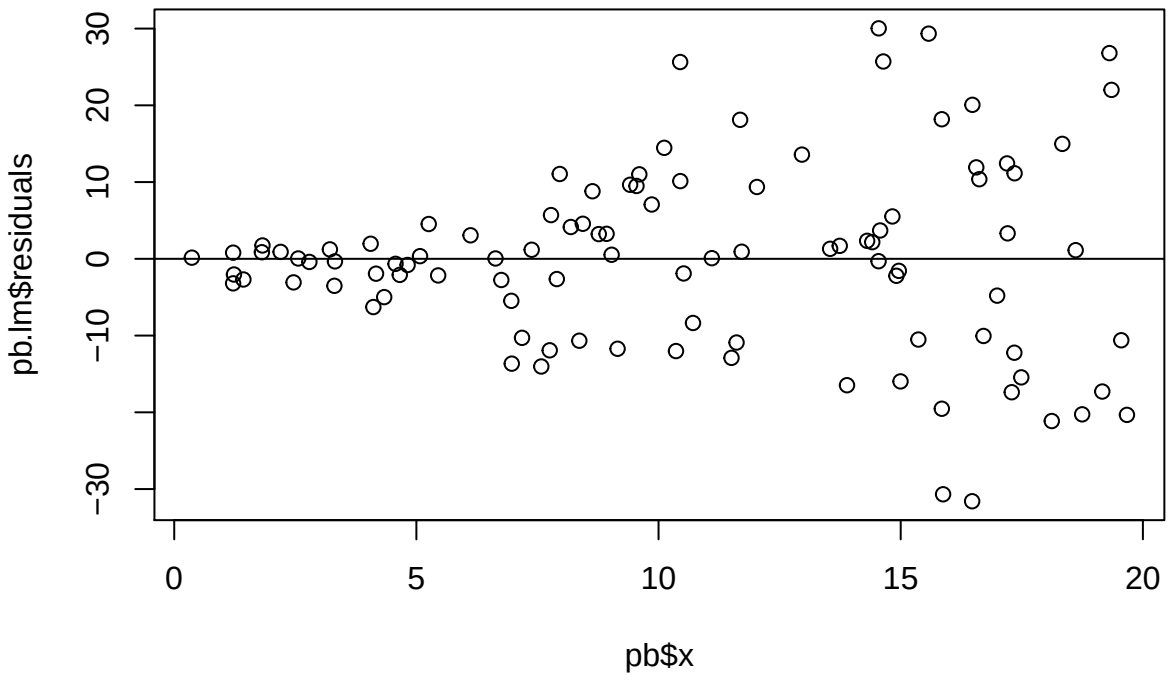
```
## [1] 53730.55 56837.68 68037.51 47478.76
```

It seems that model 6 is the best as it has the lowest error. I would say it is moderately better than the 2nd best one. The models are not extremely better than each other but some are noticeably better, such as model 6 vs model 4.

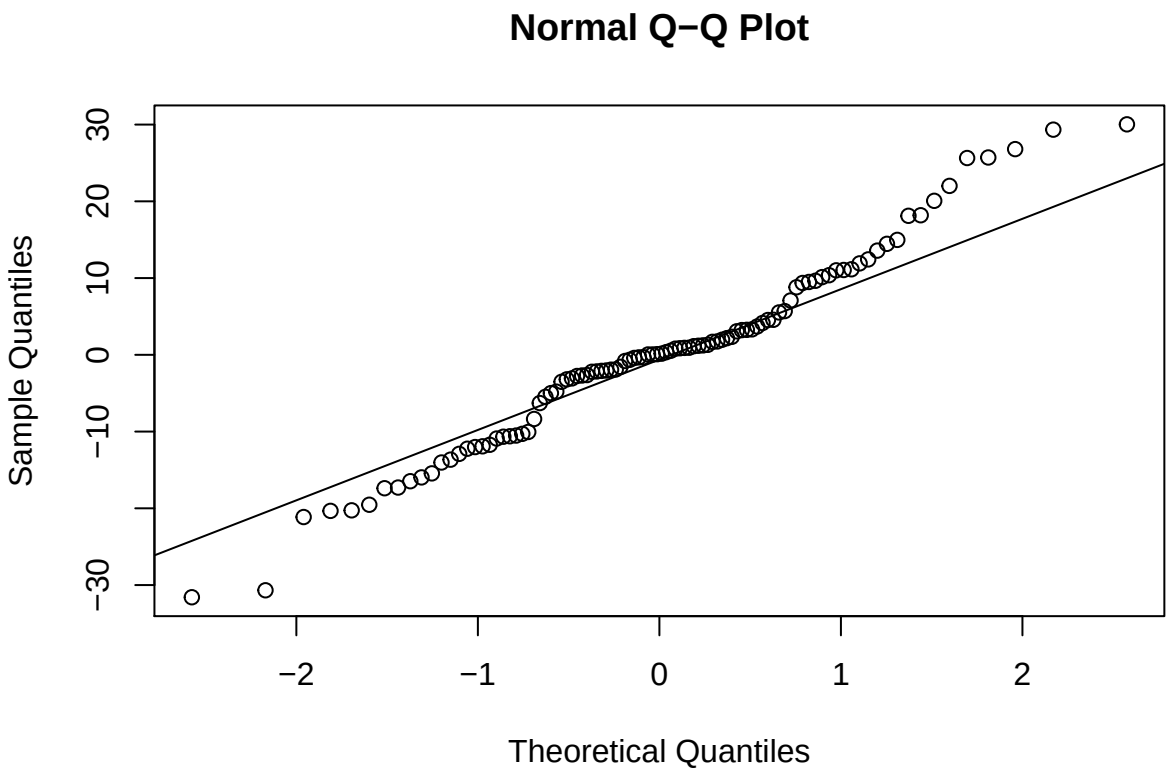
## Problem 3

### Problem 3 (a)

```
pb = read.csv("parametric-bootstrap.csv")
pb.lm = lm(y~x,data=pb)
plot(pb$x, pb.lm$residuals); abline(h=0)
```




```
qqnorm(pb.lm$residuals); qqline(pb.lm$residuals)
```




The residuals plot shows a violation of constant variance.

**Problem 3 (b)**

```
predict(pb.lm, new = data.frame(x=20), se.fit=TRUE)$fit
```

```
##        1
## 13.89667
```

The iid assumptions of linear regression must hold true. These are not all met as you can see above, the assumption of constant variance is violated.

**Problem 3 (c)**

```
#bootstrap = function(x, fit){
#  B0 = coef(fit)[1]
#  B1 = coef(fit)[2]
#  e = predict(pb.lm, new = data.frame(x=20), se.fit=TRUE)$fit
#  return(B0 + B1*x + e)
#}

bootstrap = function(data, m){
```

```
  m = lm(y~x,data)
  B0 = coef(m)[1]
  B1 = coef(m)[2]
  x = data$x
  e = predict(m, new = data.frame(x=20), se.fit=TRUE)$fit
  data$y = B0+B1*data$x+e
  return(data)
}
#pb$y = bootstrap(pb$x, pb.lm)
head(pb)
```

```
##           x          y
## 1 17.345809  0.1684985
## 2 10.711470  0.3118346
## 3 16.560001 23.8770791
## 4 16.709098  1.9945140
## 5  7.899252  4.4541096
## 6 14.910671  8.8255629
```

**Problem 3 (d)**

```
B = 1000
estimates = vector(length=1000)
for(i in 1:B){
  newdata = bootstrap(pb, pb.lm)
  newlm = lm(y~x, pb)
  x = newdata$x
  r20 = predict(newlm, new = data.frame(x=20), se.fit=TRUE)$fit
  estimates[i] = r20
}
sd(estimates)
```

```
## [1] 0
```