Name: Thuan Tran

HW_13_1

a)

```
x1 = c(8.9,36.6,36.8,6.1,6.9,6.9,7.3,8.4,6.5,8.0,4.5,9.9,2.9,2.0)
x2= c(31.5,27.0,25.9,39.1,39.2,38.3,33.9,33.8,27.9,33.1,26.3,37,34.6,36.4)
y= c(14.7,48,25.6,10,16,16.8,20.7,38.8,16.9,27,16,24.9,7.3,12.8)
> model = lm (y~ x1 + x2 + I(x1^2) + I(x2^2)+ x1*x2)
> model$coefficients

  (Intercept)            x1            x2       I(x1^2)
-140.22976413  -16.47520736   12.82710366    0.09555181
       I(x2^2)         x1:x2
   -0.24339330    0.49863532
```

b) No, it is not possible to interpret based on the coefficients anymore. Since there is now an interaction term x1*x2. In which the two terms are now affecting each other

c)

```
> summary(model)$r.squared
[1] 0.7561284
```

This $_{R^2}$ can be interpreted as 75.6% of the variability in y can be explained by the predictors in the model. In another word, the model explain 75.6% variabiltiy of y

d)

```
> summary(model)$sigma
[1] 7.02349
```

This standard error shows that on average, the predicted y (hat) value is +- mean of y

e)

```
> resid =  resid(model)
> fittedVar = fitted(model)
> plot(fittedVar,resid)
```

The plot below show that most of the variable is below the horizontal line 0, meaning that most of the predicted value is less than the actual value. It might suggest that there can be bias because of possible under fit

f) and g) and h)

```
> newModel = lm (y ~ x1+x2)
> summary(newModel)$r.squared
[1] 0.4470343
```

Looking at the new smaller R square of the new model, we can say that using the new model, the 44% of the variability in y is explained by the model. While it is true that the model with the higher order term have more R^2 , we can not conclude that the first model is better(or the higher order term is better)

I)

```
> summary(newModel)$sigma
[1] 9.019244
```

Looking at the standard error of the new model, we can say that on average, the new predicted value for y is +- 9.091 of the mean of y. This suggest that the first model is better because it has lower error. However, we can't really conclude which model is better until we test it. The first model might be good for the sample data but might not be good for actual data. Or the sample in the second model is really bad but the second model perform really good on the actual data.

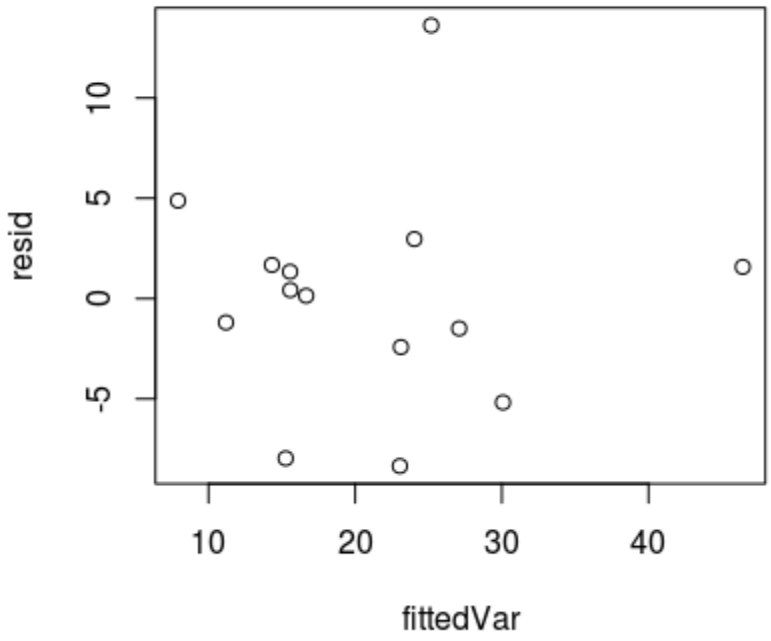

Illustration 1: Residual Plot

13.2

a)

```
> n = 100
> x1 = runif(n,0,1)
> x2 = runif(n,0,1)
> error  = rnorm(n,0,0.5)
> y = 2+3*x1+4*x2+error
> model = lm(y~x1+x2)
> model

Call:
lm(formula = y ~ x1 + x2)

Coefficients:
(Intercept)           x1           x2
      1.964        2.816        4.233

> summary(model)$r.squared
[1] 0.905897
> summary(model)$sigma
[1] 0.4437154
```

b)

```
> y = 2+3*x1+4*x2+50*x1*x2 + error
> model = lm(y~ x1 + x2 )
> summary(model)$r.squared
[1] 0.8706943
> summary(model)$sigma
```

```
[1] 4.661374
```
c)
```
> model = lm(y~ x1 + x2 + x1*x2)
> summary(model)$r.squared
[1] 0.9099118
> summary(model)$sigma
[1] 0.4709719
```
d) I was having trouble installing the package. It throws some errors and I have been trying to fix it but unsuccessfully

HW 13.3

a) For the data set, hw_dat1.txt, I decided to use the a model with interaction in it and only use x1, x2 and x1*x2. Here are the steps that I did to reach the conclusion

- I used cor(x1,x2) and plot them on a graph to see that if there are any correlation. And the plot show that there is no correlation
- I plotted x1 vs y and x2 vs y. The data seems to be clustered in the middle and there seems to be a straight line that can go through it
- I plotted (x1, x1-x2) and (x2,x1-x2) to see that as 1 variable change, the difference between those 2 variable seems to follow a correlation. Meaning that 1 variable might interact another

```
> data = read.table("http://www.stat.washington.edu/marzban/390/summer17/hw_3_dat1.txt")
> x1 = data[2:333, 1]
> x2 = data[2:333,2]
> y = data[2:333,3]
> x1 = as.numeric(levels(x1))[x1]
> x2 = as.numeric(levels(x2))[x2]
> plot(x1,x2)
> cor(x1,x2)
> y = as.numeric(levels(y))[y]
> model = lm(y~ x1+x2)
> plot(x1-x2,x2)
> plot(x2-x1,x1)
> model

Call:
lm(formula = y ~ x1 + x2 + x1 * x2)

Coefficients:
(Intercept)           x1           x2        x1:x2
      1.019        1.919        3.037        6.059



> summary(model)$r.squared
[1] 0.918664
> summary(model)$sigma
[1] 2.064917
```

b)
For this model, I detected that there is colinearality between x1 and x2 through the correlation coefficient. After that, I plotted x1 vs y and x2 vs y and found out that the data seems to be follow a curve hence a non linear relationship. That is why I decided to add x1^2 and x2^2. Looking at R^2 = 0.944 indicated that this model is really good

```
> data = read.table("http://www.stat.washington.edu/marzban/390/summer17/hw_3_dat2.txt")

> x1 = data[2:333,1]
> x2 = data[2:333,2]
```

```
> y = data[2:333,3]
> y = as.numeric(levels(y))[y]
> x1 = as.numeric(levels(x1))[x1]
> x2 = as.numeric(levels(x2))[x2]
> cor(x1,x2)
[1] 0.8810057
> plot(x1,x2)
> plot(x1,x1-x2)
> plot(x2,x1-x2)
> plot(x1,y)
> plot(x2,y)
> model = lm(y~ x1 + x2 +I(x1^2) + I(x2^2))
> model

Call:
lm(formula = y ~ x1 + x2 + I(x1^2) + I(x2^2))

Coefficients:
(Intercept)           x1           x2      I(x1^2)
      0.4337       2.1459       2.8518       2.8715
     I(x2^2)
      3.0379

> summary(model)$r.squared
[1] 0.9412127
> summary(model)$sigma
[1] 2.267764
```

3.39

Available in the attached paper

HW.14.1

```
> set.seed(123)
> n= 10
> x1 = runif(n,-1,1)
> y = 1 + 2*x1 + rnorm(n,0,1)
> plot(x1,y)
> model = lm(y~ x1)
> summary(model)$r.squared
[1] 0.4827701
> x2 = runif(n,-1,1)
> x3 = runif(n,-1,1)
> x4 = runif(n,-1,1)
> x5 = runif(n,-1,1)
> newModel = lm(y~ x1+x2+x3+x4+x5)
> summary(newModel)$r.squared
[1] 0.6700543


Looking at the new R^2, seems like the new model performed a better job than the old model.
This is because it gets to use x2, x3,x4 and x5 coming from the same distribution
```

HW14.2

```
n = 50
numberTrials = 5000
i = 1
```

```
maxSample = matrix(nrow = 1, ncol = 5000)
minSample = matrix(nrow = 1, ncol =5000)
while(i <= numberTrials)
{
  sample = rnorm(n, 0,1)
  maxSample[i] = max(sample)
  minSample[i] = min(sample)
  i = i + 1
}
hist(maxSample)
hist(minSample)
```

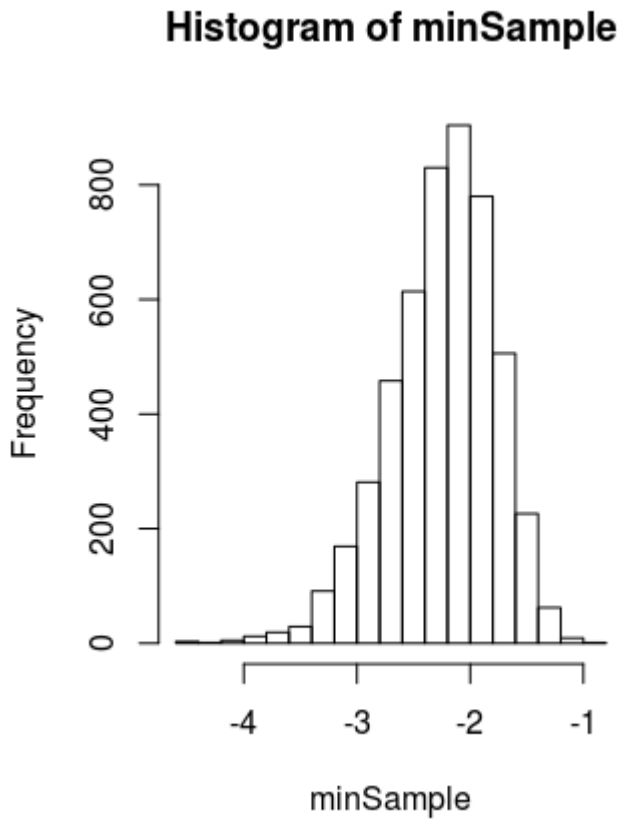

Illustration 2: Histogram of sample minimum

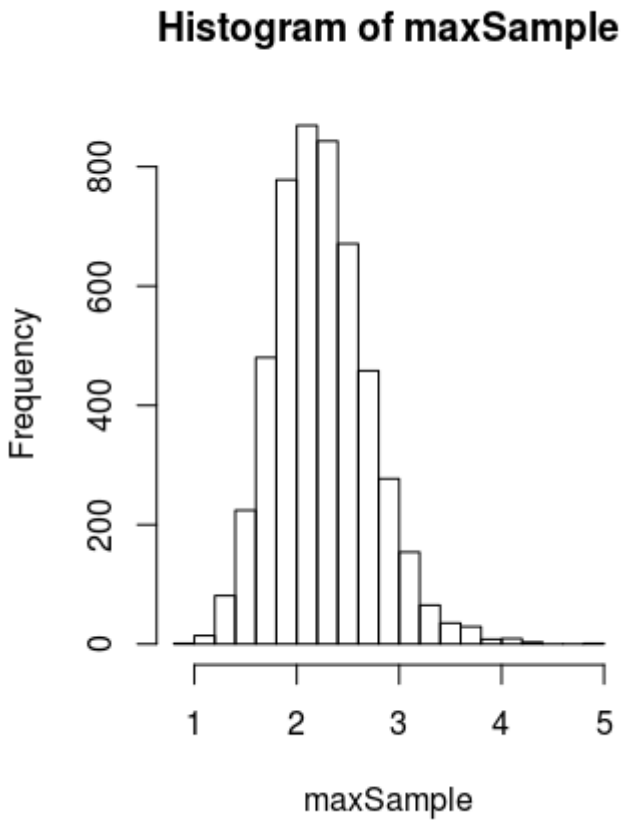

Illustration 3: Histogram of Sample Maximum

HW_14_3

```
n = 100
numberTrials = 5000
i = 1
meanSample = matrix(nrow
= 1, ncol = 5000)
while(i <= numberTrials)
{
  sample = rexp(n, 2)
  meanSample[i] =
mean(sample)
  i = i + 1
}
qqnorm(meanSample)
```