Assignment 1 3550

R Markdown

2.9 a)

```
carData<-data.frame(x1=c(14.620,15.630,14.620,15.000,14.500,15.250,16.120,15.130,15.500,15.130,15.500,1
mod<-lm(y~x1+x2+x3,data=carData)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2 + x3, data = carData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.1859 -1.8209  0.0694  2.3886  5.0004
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -69.14649   39.76675  -1.739   0.1025
## x1            0.08147    0.02901   2.808   0.0132 *
## x2            0.21239    0.18483   1.149   0.2685
## x3           20.85564    0.55987  37.251 3.37e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.455 on 15 degrees of freedom
## Multiple R-squared:  0.9942, Adjusted R-squared:  0.993
## F-statistic:   851 on 3 and 15 DF,  p-value: < 2.2e-16
```

2.9 b)

```
params<-mod$coefficients
hFE<-params[1]+14.5*params[2]+220*params[3]+5.0*params[4]
hFE
```

```
## (Intercept)
##    83.03883
```

2.9 c)

```
summary(aov(mod))
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## x1           1   2174    2174   182.1 8.55e-10 ***
## x2           1  11736   11736   983.1 4.33e-15 ***
## x3           1  16565   16565  1387.6 3.37e-16 ***
## Residuals   15    179      12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As you can see at the alpha level 0.05 we reject the null hypothesis 2.9 d) this is just the MSE (i.e. mean of the squared residuals)

```
mean(mod$residuals^2)
```

```
## [1] 9.424418
```

2.9 e)

```
sqrt(diag(vcov(mod)))
```

```
## (Intercept)          x1          x2          x3
##  39.7667489   0.0290112   0.1848257   0.5598695
```

2.9 f) if we look at the t Value column we get the t-values for each statistic. The next column are the p-values for the t-test and at a 0.05 significance level we would reject the null hypothesis for x1 and x2 while we would fail to reject for x3.

```
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2 + x3, data = carData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.1859 -1.8209  0.0694  2.3886  5.0004
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -69.14649   39.76675  -1.739   0.1025
## x1            0.08147    0.02901   2.808   0.0132 *
## x2            0.21239    0.18483   1.149   0.2685
## x3           20.85564    0.55987  37.251 3.37e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.455 on 15 degrees of freedom
## Multiple R-squared:  0.9942, Adjusted R-squared:  0.993
## F-statistic:   851 on 3 and 15 DF,  p-value: < 2.2e-16
```

2.9 g)

```
confint(mod, level =0.99)
```

```
##                     0.5 %     99.5 %
## (Intercept) -1.863277e+02 48.0347044
## x1          -4.014807e-03  0.1669606
## x2          -3.322384e-01  0.7570179
## x3           1.920587e+01 22.5054179
```

2.9 h)

```
predictionData<-data.frame(x1=14.5,x2=220,x3=5.0)
predict(mod, predictionData, interval="predict",level = 0.99)
```

```
##        fit      lwr      upr
## 1 83.03883 72.30447 93.77319
```

The prediction interval is (72.30447,93.77319)

2.9 i)

```
predictionData<-data.frame(x1=14.5,x2=220,x3=5.0)
predict(mod, predictionData, interval="confidence",level = 0.99)
```

```
##        fit      lwr      upr
## 1 83.03883 79.63722 86.44044
```

The mean response confidence interval is (79.63722,86.44044)

2.15)

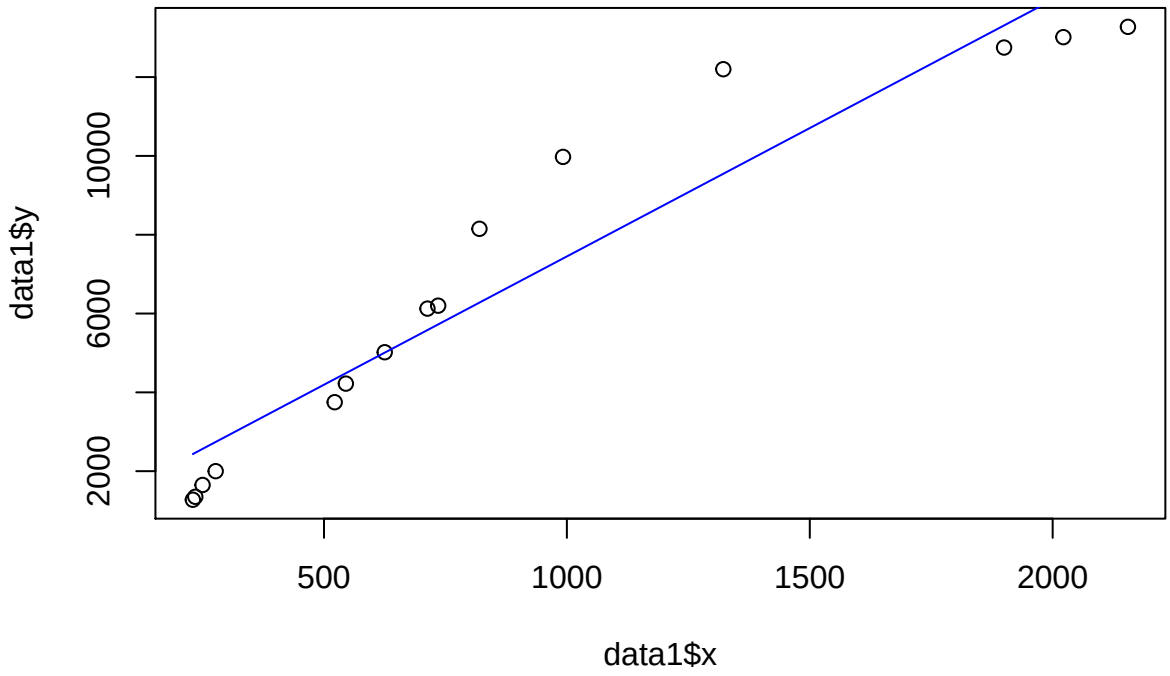
```
data1<-data.frame(y=c(1275,1350,1650,2000,3750,4222,5018,6125,6200,8150,9975,12200,12750,13014,13275),x=
```

2.15 a) using the first data values to form an estimate:

```
nls1<-nls(y~a+b*x,data1,start = list(a=100,b=5))
summary(nls1)
```

```
##
## Formula: y ~ a + b * x
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a 936.9261   631.4785   1.484    0.162
## b   6.5128     0.5764  11.299 4.29e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1428 on 13 degrees of freedom
##
## Number of iterations to convergence: 1
## Achieved convergence tolerance: 1.9e-08
```

```r
plot(data1$x,data1$y)
lines(data1$x,predict(nls1),col="blue")
```

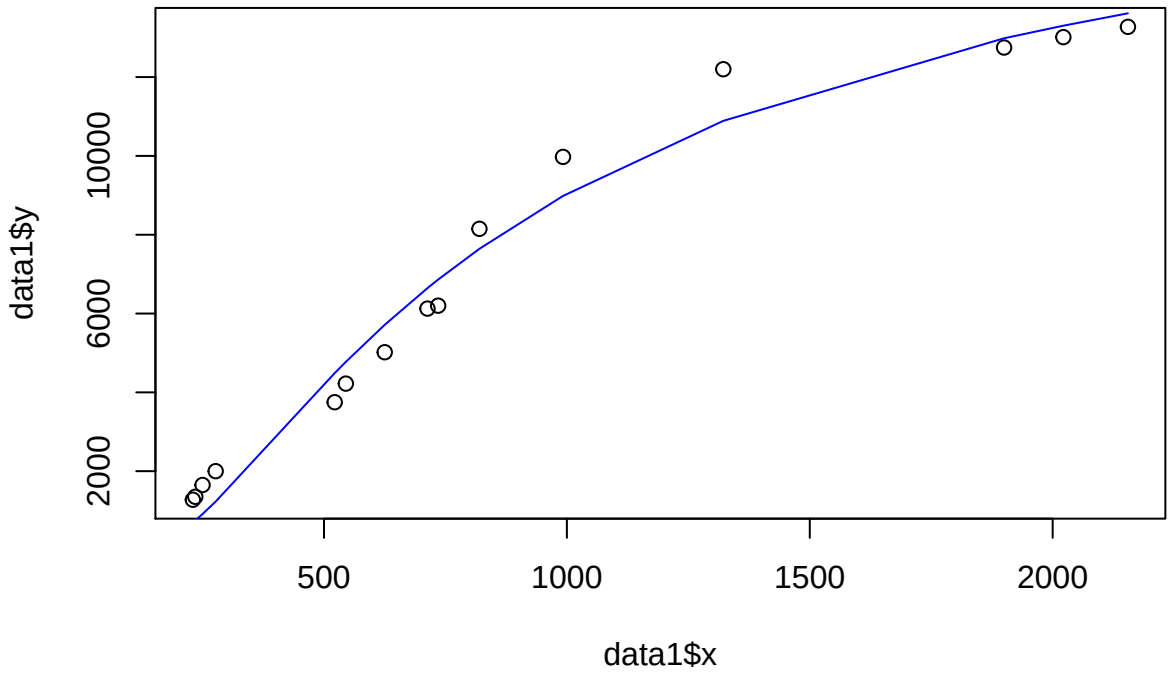


2.15 b) using the first data values to form an estimate:

```r
nls1<-nls(y~exp(a+b*(1/x)),data1,start = list(a=7,b=230))
summary(nls1)
```

```
##
## Formula: y ~ exp(a + b * (1/x))
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## a    9.87394    0.04867  202.87  < 2e-16 ***
## b -764.55009   52.39906  -14.59 1.94e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 738.3 on 13 degrees of freedom
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 2.038e-06
```

```r
plot(data1$x,data1$y)
lines(data1$x,predict(nls1),col="blue")
```

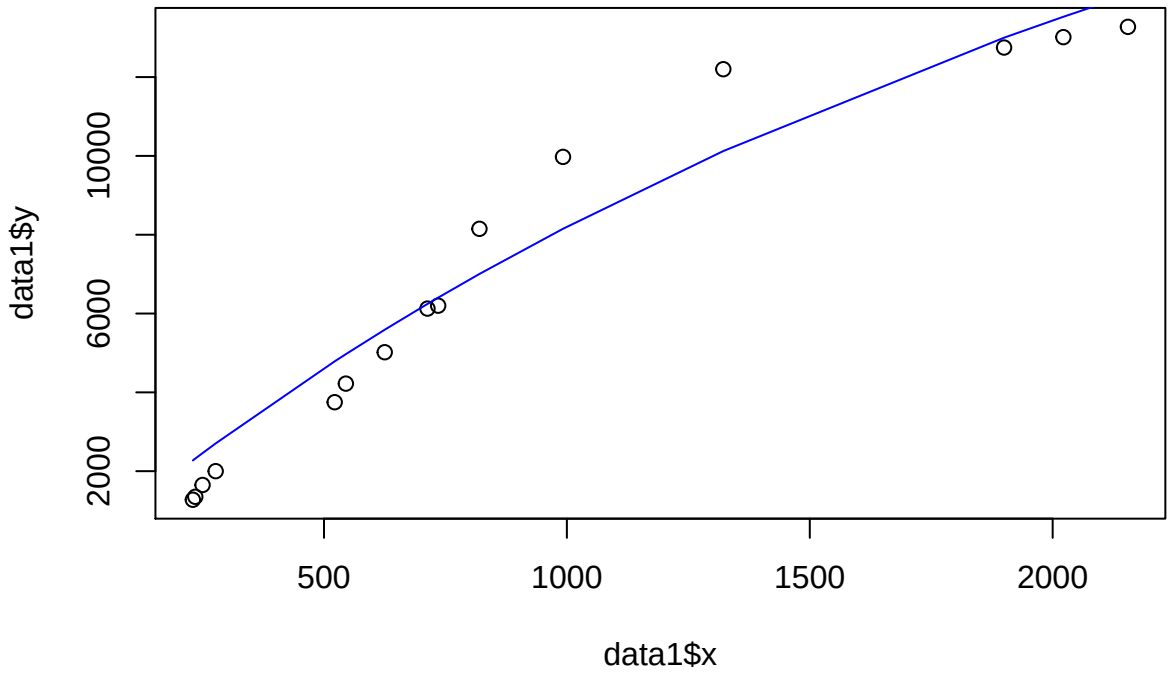


2.15 c) using the first data values to form an estimate:

```r
nls1<-nls(y~1/(a+b*(1/x)),data1,start = list(a=.0004,b=.00230))
summary(nls1)
```

```
##
## Formula: y ~ 1/(a + b * (1/x))
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## a 2.693e-05  7.207e-06   3.737  0.00249 **
## b 9.499e-02  1.098e-02   8.655 9.35e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1075 on 13 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 7.201e-06
```

```r
plot(data1$x,data1$y)
lines(data1$x,predict(nls1),col="blue")
```

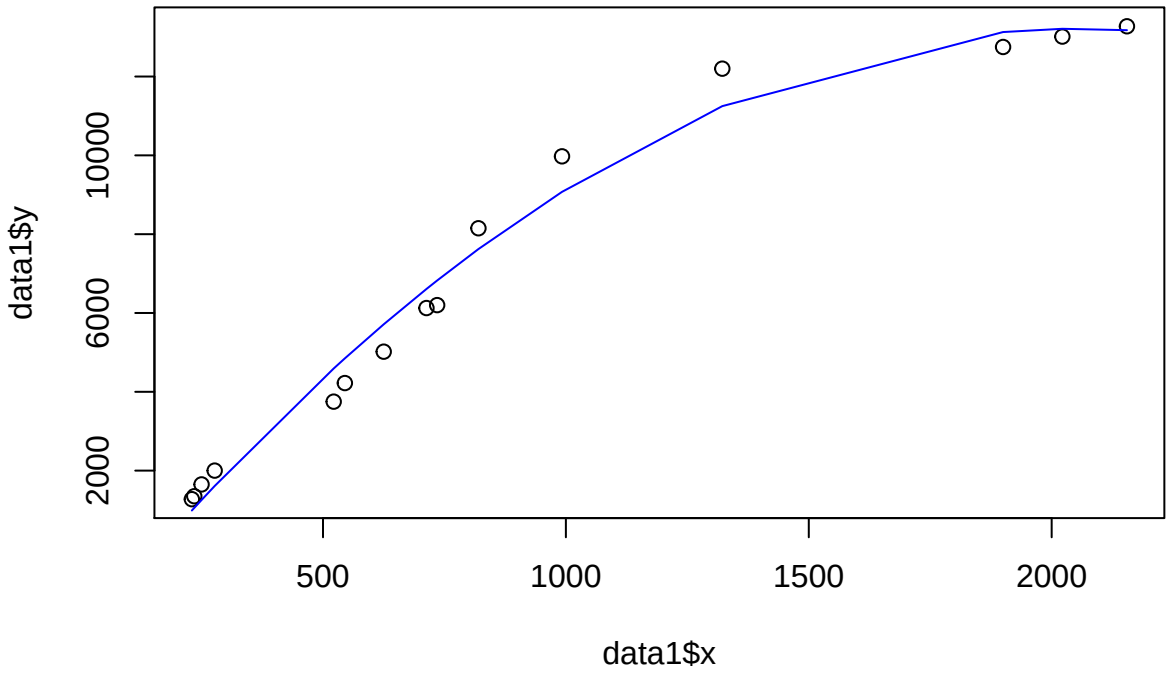


2.15 d) using the first data values to form an estimate:

```r
nls1<-nls(y~a+b*x+c*(x^2),data1,start = list(a=50,b=3,c=.01))
summary(nls1)
```

```
##
## Formula: y ~ a + b * x + c * (x^2)
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## a -2.293e+03  5.247e+02  -4.370 0.000913 ***
## b  1.511e+01  1.206e+00  12.531 2.98e-08 ***
## c -3.680e-03  5.043e-04  -7.298 9.51e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 637.2 on 12 degrees of freedom
##
## Number of iterations to convergence: 1
## Achieved convergence tolerance: 7.753e-08
```

```
plot(data1$x,data1$y)
lines(data1$x,predict(nls1),col="blue")
```



2.15 e) For a) we see that the p-value for b is very low however the p-value for a is quite high in comparison. If we look at the plot the points seem to not fit the line at all.

For b) we see that both paramters have a very very low p-value which indicates they are most likely very accurate. The plot seems to also fit the shape of the data well.

For c) the second parameter seems to be pretty accurate and the first parameter's p-value would likely pass most hypothesis tests however it does not have the same accuracy as b). Looking at the plot we can see it doesn't curve as well as b) does.

for d) the second and third parameters have low p-values so they are likely strongly accurate and the first parameter also has a decently low value meaning this model has overall good accuracy. Looking at c) and d) we can see that the more accurate a model gets, the closer the plot looks like the plot in b). This indicates, along with the p-values of the parameters, that b) is likely the most accurate model.

```
data2<- data.frame(y=c(2.240,3.581,5.131,5.715,0.889,2.845,6.147,7.016,7.747,8.286,9.321,9.195,1.895,3.
```

2.16 a)

```
mlrMod<-lm(y~x1+x2+x3,data=data2)
summary(mlrMod)
```

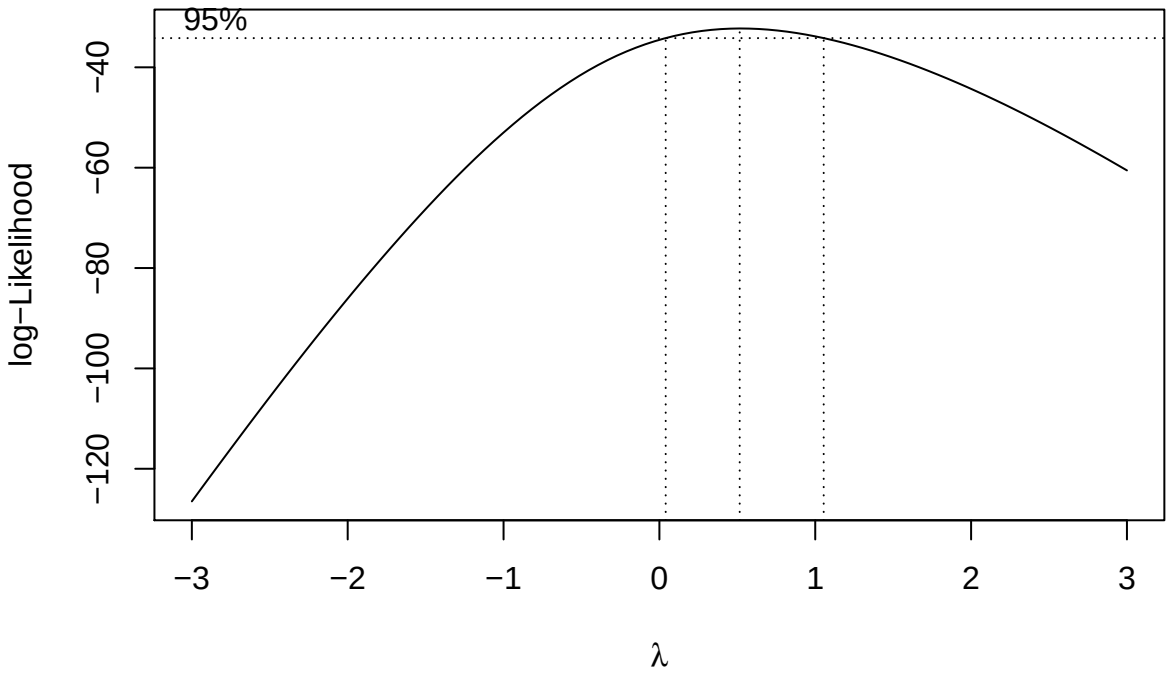
```
##
```

```
## Call:
## lm(formula = y ~ x1 + x2 + x3, data = data2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.8686 -1.8980  0.0018  1.0706  6.1197
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.4349794  3.9019697   1.137  0.26534
## x1           0.0460049  0.0155614   2.956  0.00626 **
## x2           0.0004249  0.0023629   0.180  0.85860
## x3          -0.1850077  0.1994489  -0.928  0.36155
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.616 on 28 degrees of freedom
## Multiple R-squared:  0.283,  Adjusted R-squared:  0.2062
## F-statistic: 3.684 on 3 and 28 DF,  p-value: 0.0236
```

We can see from the p-values that most of these parameters would not be accepted in t tests by a large margin at any reasonable significance level. This hints that this model is not very adequate.

2.16 b)

```
bc<-boxcox(mlrMod,lambda = seq(-3, 3, length = 10))
```



The max (lambda) is

```r
bc$x[which(bc$y==max(bc$y))]
```

```
## [1] 0.5151515
```

```r
qchisq(.95,1)
```

```
## [1] 3.841459
```

2.16 c) We can easily see from the plot that the value lambda=1 is present in the 95% confidence interval so it is possible that this transformation is unnecessary.

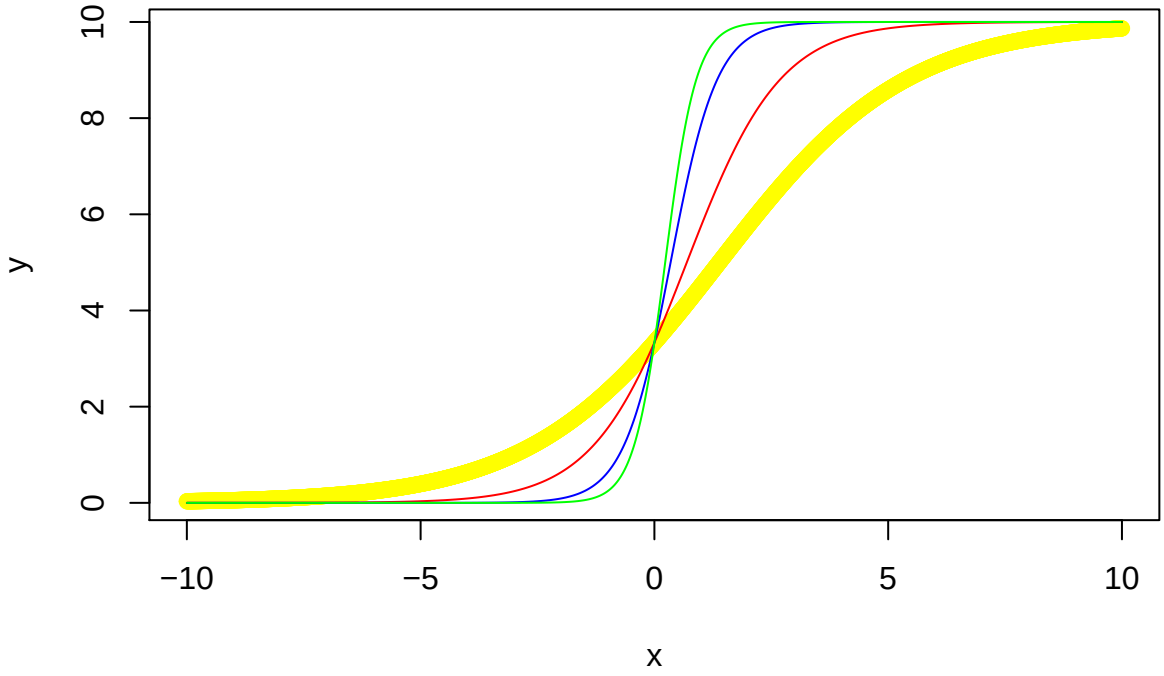
2.16 d) Transformations is likely not needed

3.3

```r
b1=10
b2=2
b3=c(.5,1,2,3)
x=seq(-10,10,by=.01)

y<-b1/(1+b2*exp(-b3[1]*x))
y2<-b1/(1+b2*exp(-b3[2]*x))
y3<-b1/(1+b2*exp(-b3[3]*x))
y4<-b1/(1+b2*exp(-b3[4]*x))

plot(x,y,col="yellow",lwd=1)
lines(x,y2,col="red",lwd=1)
lines(x,y3,col="blue",lwd=1)
lines(x,y4,col="green",lwd=1)
```



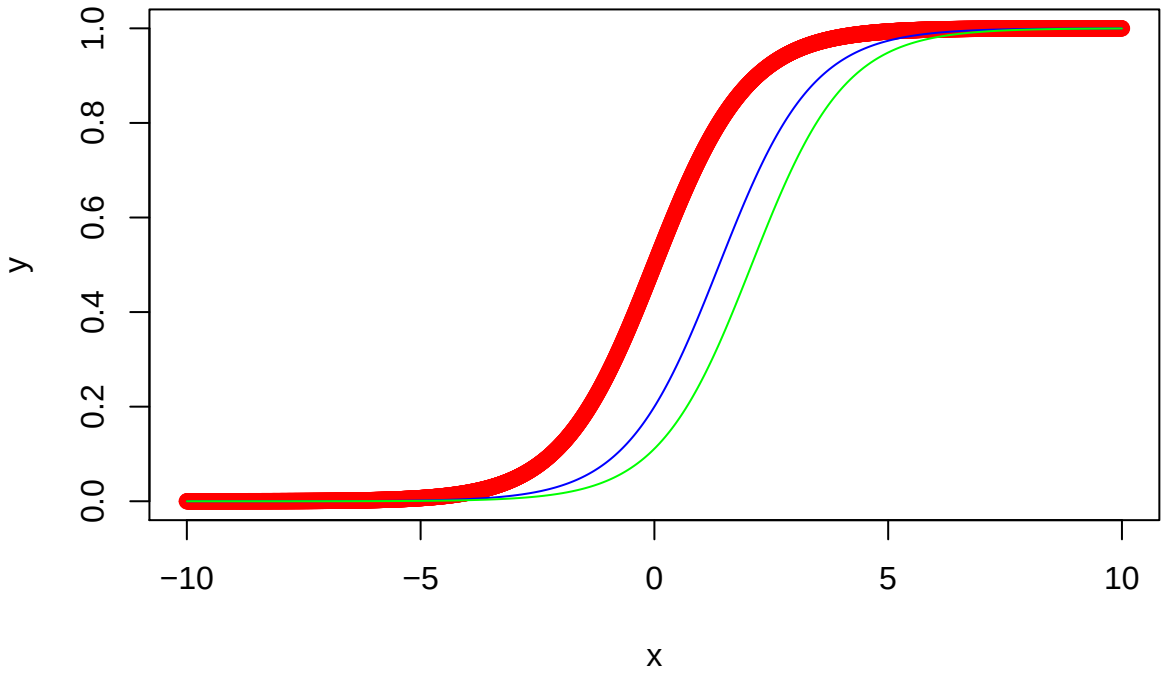
We can see that as b3 increases, the response variable y grows quicker and thus the slopes are steeper and the graph plateaus quicker.

3.4

```
b1=1
b3=1
b2=c(1,4,8)
x=seq(-10,10,by=.01)

y<-b1/(1+b2[1]*exp(-b3*x))
y2<-b1/(1+b2[2]*exp(-b3*x))
y3<-b1/(1+b2[3]*exp(-b3*x))

plot(x,y,col="red",lwd=1)
lines(x,y2,col="blue",lwd=1)
lines(x,y3,col="green",lwd=1)
```



We can see that as b2 increases, the graph shifts forward. This means it increases each response by a constant value, but doesa not change the rate of growth

3.8

```
x<-c(.5,1,2,4,8,9,10)
y1<-c(0.68,0.45,2.50,6.19,56.1,89.8,147.7)
y2<-c(1.58,2.66,2.04,7.85,54.2,90.2,146.3)
y<-(y1+y2)/2
df<-data.frame(x,y)
```

a) I will take a point near the middle and calculate a b1 and b2 that works for that point

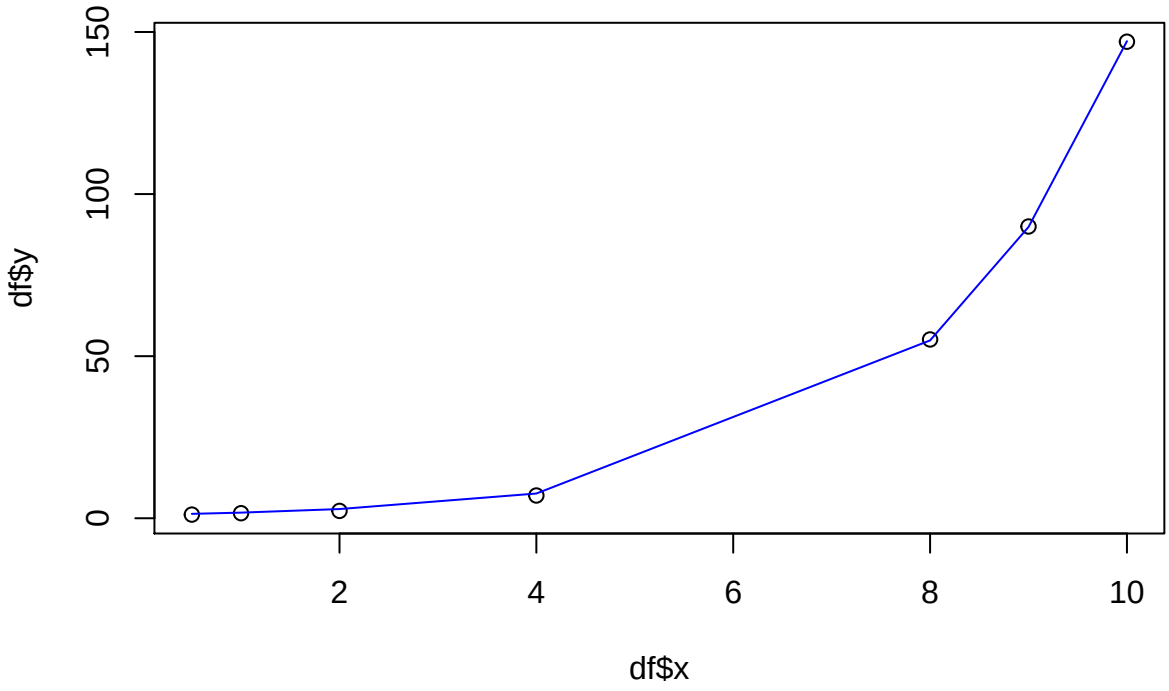
b)

```
nlsModel<-nls(y~a*exp(b*x),df,start = list(a=0.7,b=.25))
summary(nlsModel)
```

```
##
## Formula: y ~ a * exp(b * x)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a  1.05911    0.03349   31.62 5.94e-07 ***
## b  0.49341    0.00330  149.53 2.54e-10 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4261 on 5 degrees of freedom
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 4.437e-06
```

```
plot(df$x,df$y)
lines(df$x,predict(nlsModel),col="blue")
```



c) We can easily see from the p-values that both parameter estimates have tiny tiny p-values. This indicates that we should reject the null hypothesis that both of them are 0 as at least 1 of them is likely not 0.

d) We will use the Residual standard error squared as RSE is an estimate for sigma

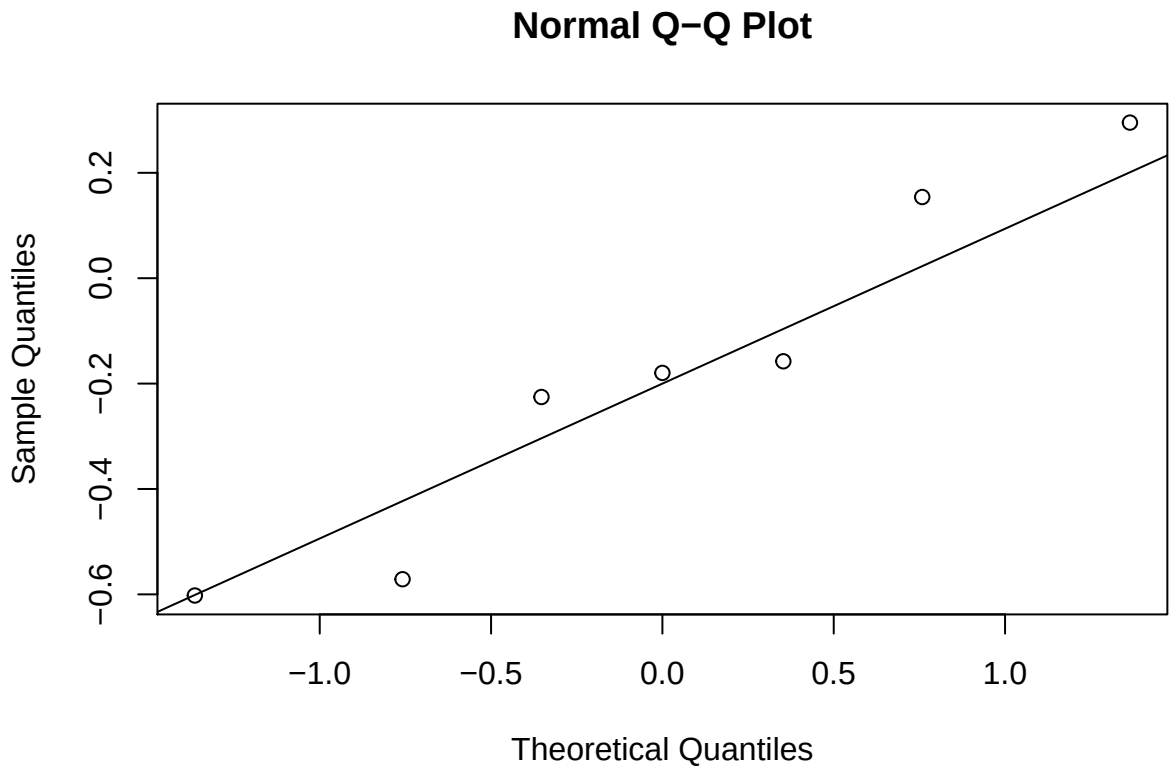
```
sigma(nlsModel)^2
```

```
## [1] 0.1815533
```

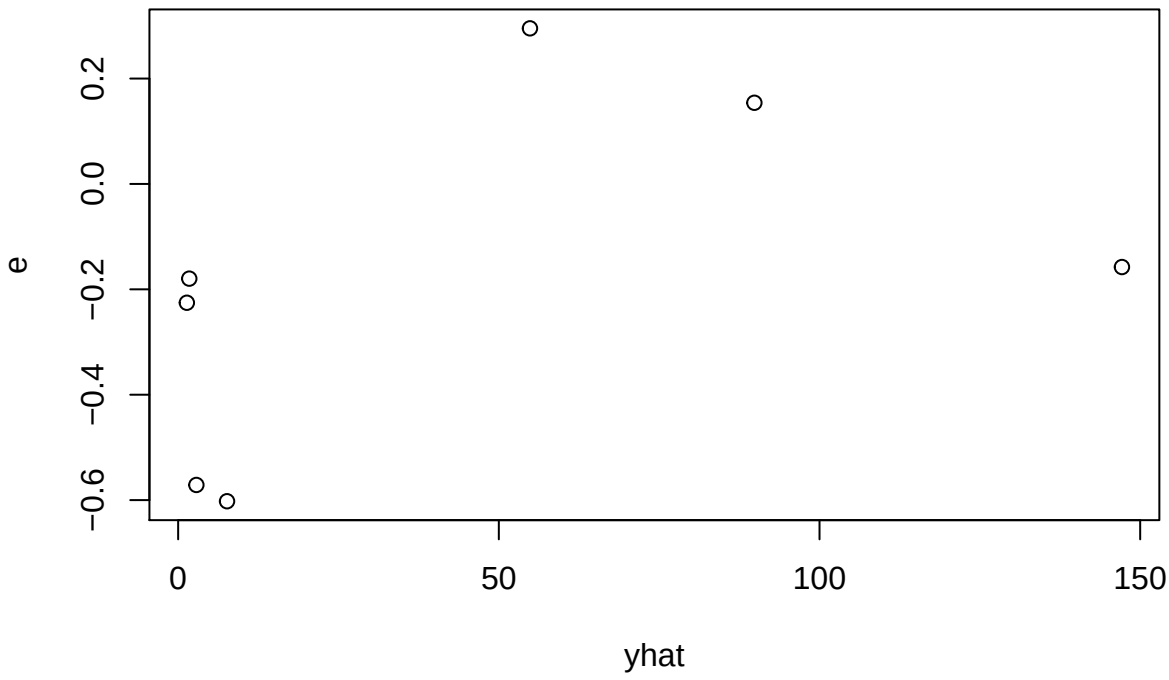
e) We see that the p-values for the parameters are tiny, less than any reasonable significance level so we can say we reject the null hypothesis and conclude that the two parameters are different from 0.

f)

```r
yhat = fitted(nlsModel)
e = residuals(nlsModel)
qqnorm(e)
qqline(e)
```




```r
plot(yhat, e)
```



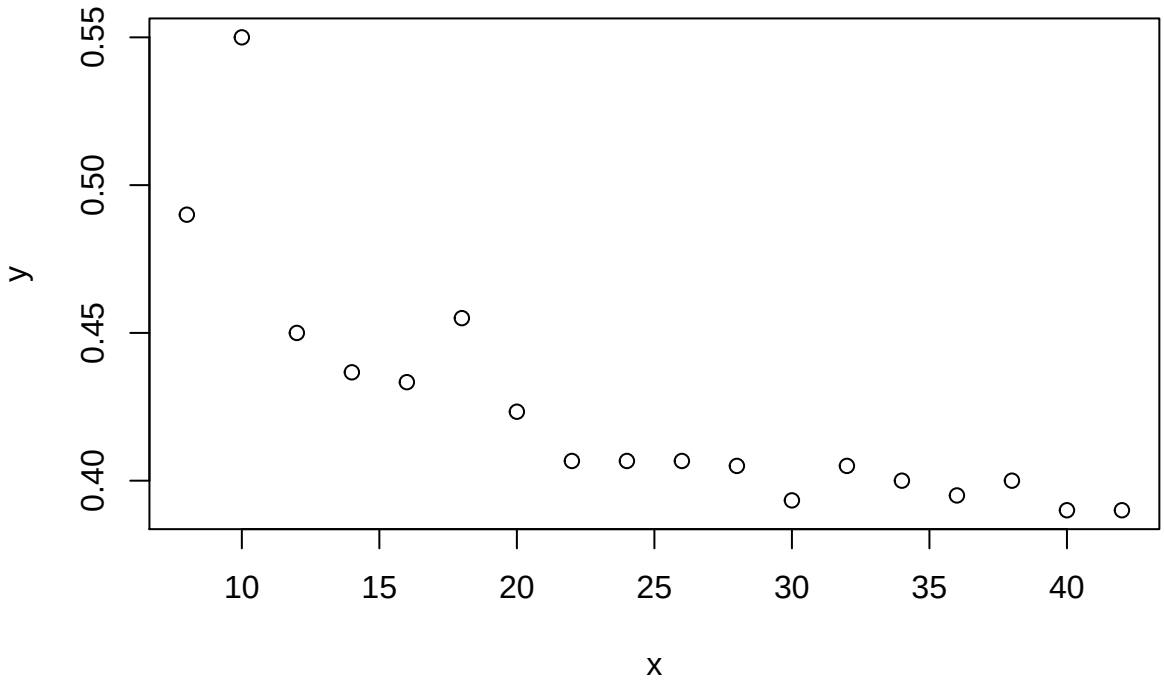
We can see from the plot that residuals seem to be randomly distributed and from the qq plot that there seems to be a linear relationship between sample and theoretical quantiles with a slope of 1. These both indicate that the sample quantiles are likely from the same distribution as the theoretical quantiles.

3.11

```
y1<-c(0.49,0.48,0.46,0.45,0.44,0.46,0.42,0.41,0.42,0.41,0.41,0.40,0.41,0.40,0.41,0.40,0.39,0.39)
y2<-c(0.49,0.47,0.46,0.43,0.43,0.45,0.42,0.41,0.4,0.4,0.4,0.4,0.4,0.4,0.38,0.4,0.39,0.39)
y3<-c(0.490,0.480,0.450,0.430,0.430,0.455,0.430,0.400,0.400,0.410,0.405,0.380,0.405,0.400,0.395,0.400,0
y4<-c(0.49,0.77,0.43,0.43667,0.43333,0.45500,0.42333,0.40667,0.40667,0.40667,0.40500,0.39333,0.40500,0.4
y<-(y1+y2+y3+y4)/4
x<-c(8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38,40,42)
df<-data.frame(x,y)
```

a)

```
plot(x,y)
```



b) using x=22 and y=0.4066 we can estimate the parameters because the 22nd root of 0.4066 is about 0.96 so plugging in .959^22 and then fixing that value to equal around .6 by multiplying by b and then doing 1 minus that value we get

```
mitchModel<-nls(y~a-b*(k^x),df,start=list(a=1,b=1.5,k=0.959))
summary(mitchModel)
```

```
##
## Formula: y ~ a - b * (k^x)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a  0.38907    0.01197  32.499 2.55e-15 ***
## b -0.30448    0.09990  -3.048  0.00814 **
## k  0.89639    0.03403  26.342 5.65e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01971 on 15 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 3.386e-06
```

c) From the p-values of the parameters we can see that at least one is very very tiny and so we can conclude that not all parameters are 0. This means we reject the null hypothesis for the significance of regression test.

d)

```
confint(mitchModel)
```

```
## Waiting for profiling to be done...
```

```
##         2.5%      97.5%
## a  0.3367069  0.4083855
## b -0.6885500 -0.1739965
## k  0.8145864  0.9609880
```

From these intervals we can see that it is likely all the model parameters differ from 0 as the value 0 is not contained in any of the intervals.

e) The estimate for sigma is residual standard error so we will simply square it

```
sigma(nlsModel)^2
```

```
## [1] 0.1815533
```