Assignment 2

Bradley Assaly-Nesrallah

12/02/2020

Question 1

```
# a) Create a vector v and a data.frame df object in R
v = seq(0,1,by=0.1)
df = data.frame(x=1:4, y =2:5)

# b) Use wrote amd write.table to export the two objects into two files
write(v, file = "data", ncolumns = 1, append = FALSE, sep = " ")
write.table(df, file="data1",append = FALSE, quote= TRUE)

# c) Use scan and read.table to import tata sets in the two files into R but to different names

v1 = scan("data")
df1 = read.table("data1")

# d) Verify that they are the same
if (v == v1 && df == df1){
  print("True")
}else{
  print("False")
}
```

```
## [1] "True"
```

Question 2

```
# Create vectors containing the series 2^n and n^3 from n = 1 to n = 15
n = 1:15
s1 = c()
s2 = c()
for ( n_i in n){
  s1 = append(s1, 2^n_i)
  s2 = append(s2, n_i^3)
}

# Check which values and indexes of 2^n are greater than n^3 for 1 <= n <= 15
# Prints isplays index then value
```

```
index = c()
values = c()

for ( n_i in n){
  if (s1[n_i]> s2[n_i]){
    index = append(index,n_i)
    values = append(values, s1[n_i])
  }
}
print(index)
```

```
## [1]  1 10 11 12 13 14 15
```

```
print(values)
```

```
## [1]     2  1024  2048  4096  8192 16384 32768
```

Question 3

```
# We use the R dump and save functions to observe the similarities and differences
# Create some sample objects
obj1 <- c(1:10)
obj2 <- c("dog","fish","cat")
obj3 <- matrix( c(1, 2, 3, 4, 5, 6), nrow=2, ncol=3)

names = c("obj1","obj2","obj3")
save(obj1,obj2,obj3, file= "saved")
dump(names, file= "dumped")

#Reset variables

obj1 <- NA
obj2 <- NA
obj3 <- NA

# compare equivalence after load

load("saved")
if (obj1 == c(1:10) && obj2 == c("dog","fish","cat")
    && obj3 == matrix( c(1, 2, 3, 4, 5, 6), nrow=2, ncol=3)){
  print("Load returns identical objects")
}else{
  print("Load does not return identical objects")
}
```

```
## [1] "Load returns identical objects"
```

```
# reset objects
obj1 <- NA
obj2 <- NA
obj3 <- NA


source("dumped")

if (obj1 == c(1:10) && obj2  == c("dog","fish","cat")
    && obj3 == matrix( c(1, 2, 3, 4, 5, 6), nrow=2, ncol=3)){
  print("Source returns identical objects")
}else{
  print("Source does not return identical objects")
}
```

```
## [1] "Source returns identical objects"
```

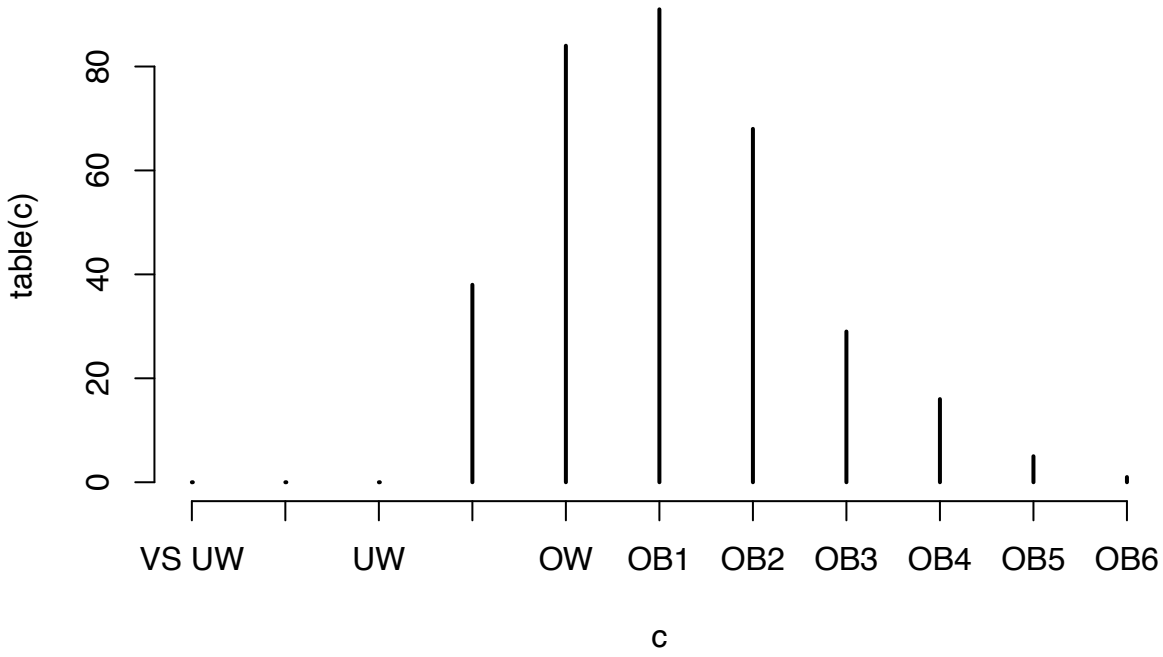
```
# Some of the similarities of Dump and Save are that they take objects
#and store representations to file
# They both can input a vector of object names
# Differences are that dump writes an ASCII representation
#of objects to a file and save writes a binary representation to a file

#are dump can only take a vector of names not object names
# Sourcing does not produce an indentical copy of dumped objects unlike save and load
# Save copies the representations of objects directly
```

Question 4

```
# We use the cut function to cut the bmi dataset in Pima.3 into three categories
# import Pima.te
library(MASS)
X<- Pima.te[,5]
# Using the BMI Labels from Wiki we cut the data and plot it using frequency
#Define Cuts and labels
cts = c(0,15,16,18.5,25,30,35,40,45,50,60,100)
lbls = c("VS UW","S UW","UW","Normal","OW","OB1","OB2","OB3","OB4","OB5","OB6")
# cut data
c<-cut(X,breaks=cts,labels=lbls)
table(c)
```

```
## c
##  VS UW   S UW     UW Normal     OW    OB1    OB2    OB3    OB4    OB5    OB6
##      0      0      0     38     84     91     68     29     16      5      1
```

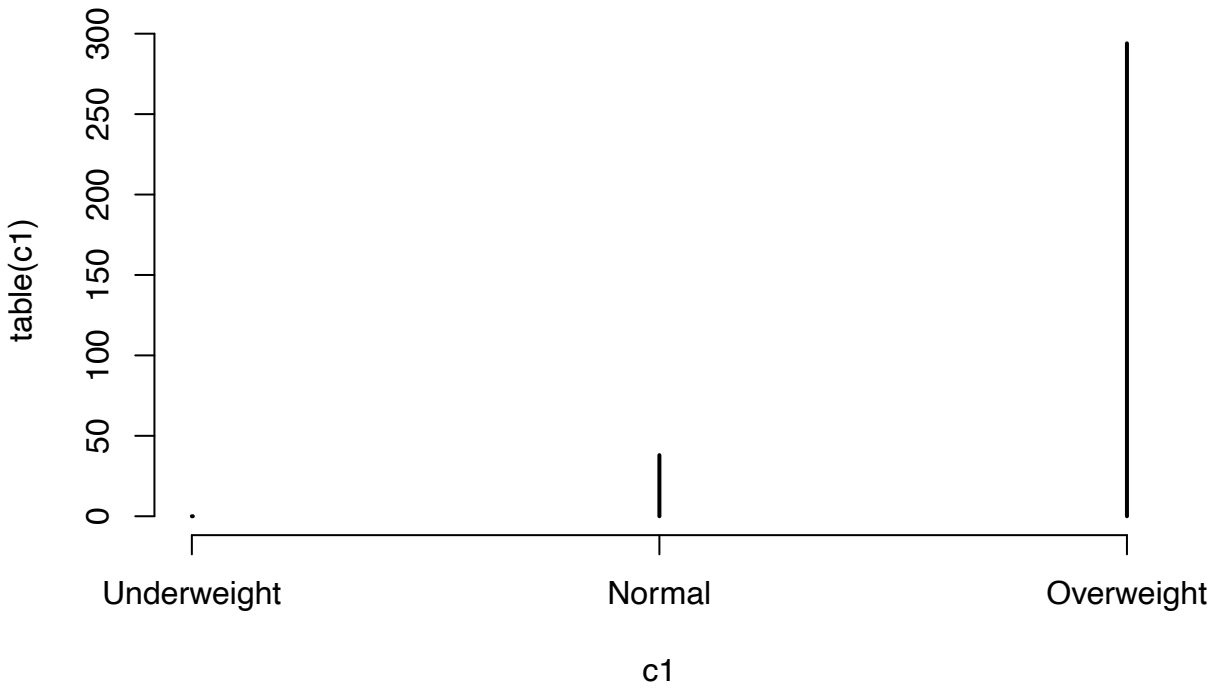
```
# plot frequencies
plot(table(c))
```




```
# We redo the cut with only three BMI underweight, normal and overweight and redo the frequency plot
# define cuts and labels
cts1 = c(0,18.5,25,100)
lbls1 = c("Underweight","Normal","Overweight")
# cut data again
c1<-cut(X,breaks=cts1,labels=lbls1)
# plot frequencies with only three categories
table(c1)
```

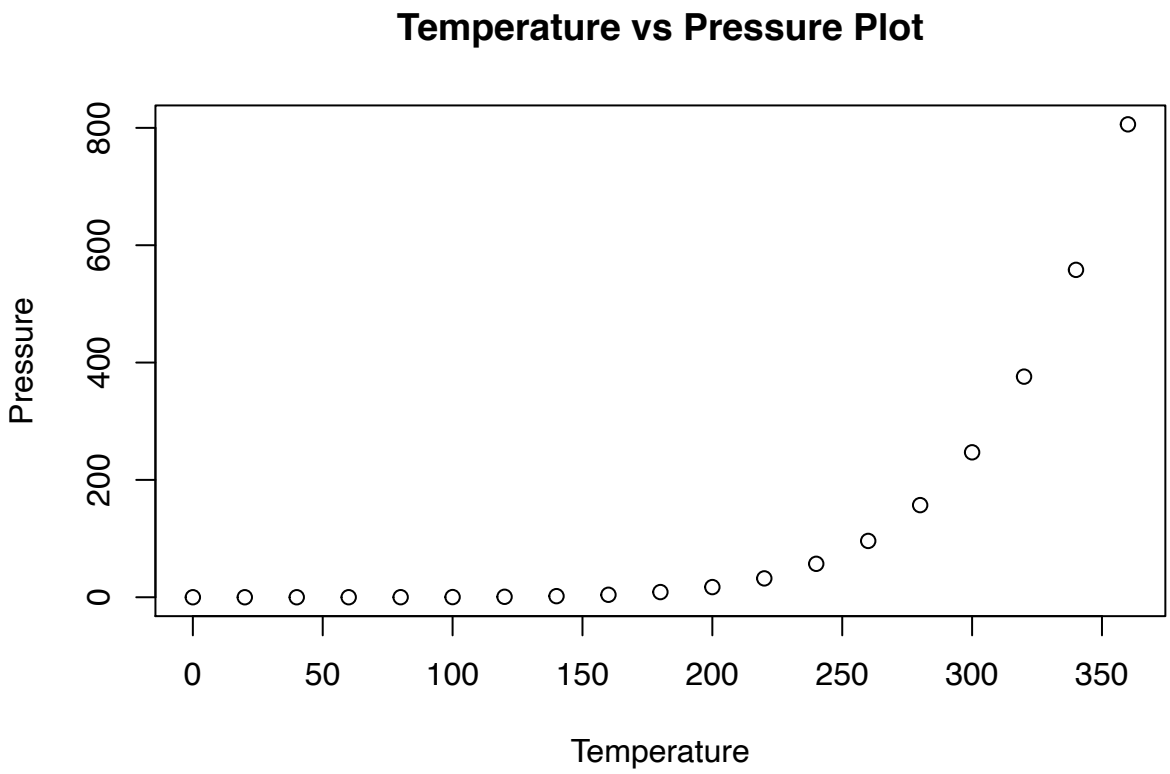
```
## c1
## Underweight      Normal  Overweight
##           0          38         294
```

```
plot(table(c1))
```

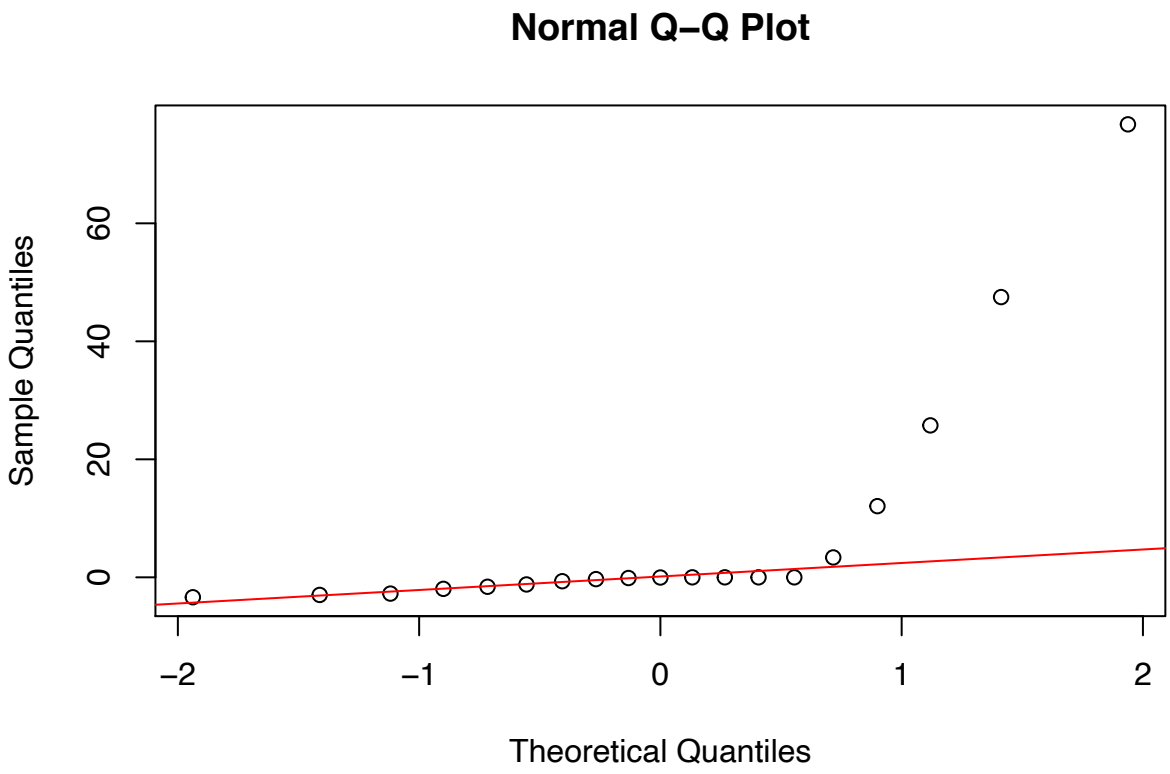


Question 5 ==========

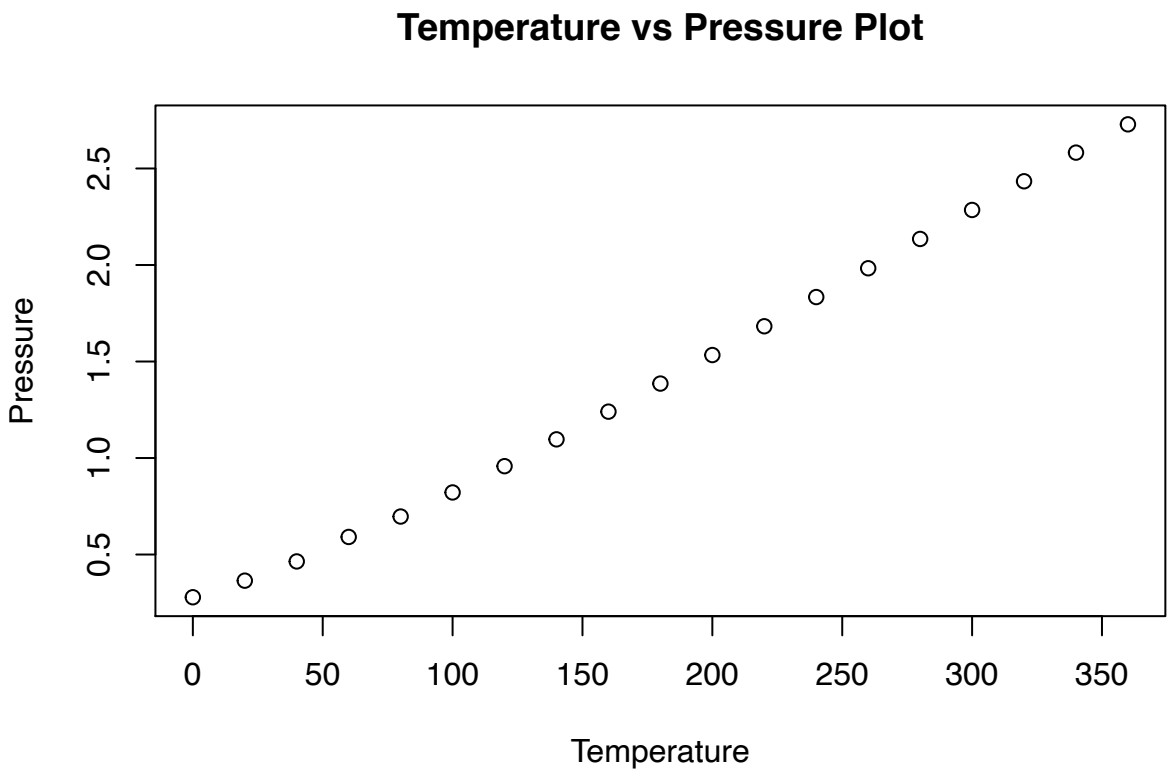
```
# a) We construct a scatterplot for pressure with pressure
# on the vertical axis and temperature on the horizontal axis

X = pressure[,1]
Y = pressure[,2]

plot(X,Y,xlab="Temperature", ylab="Pressure", main = "Temperature vs Pressure Plot")
```



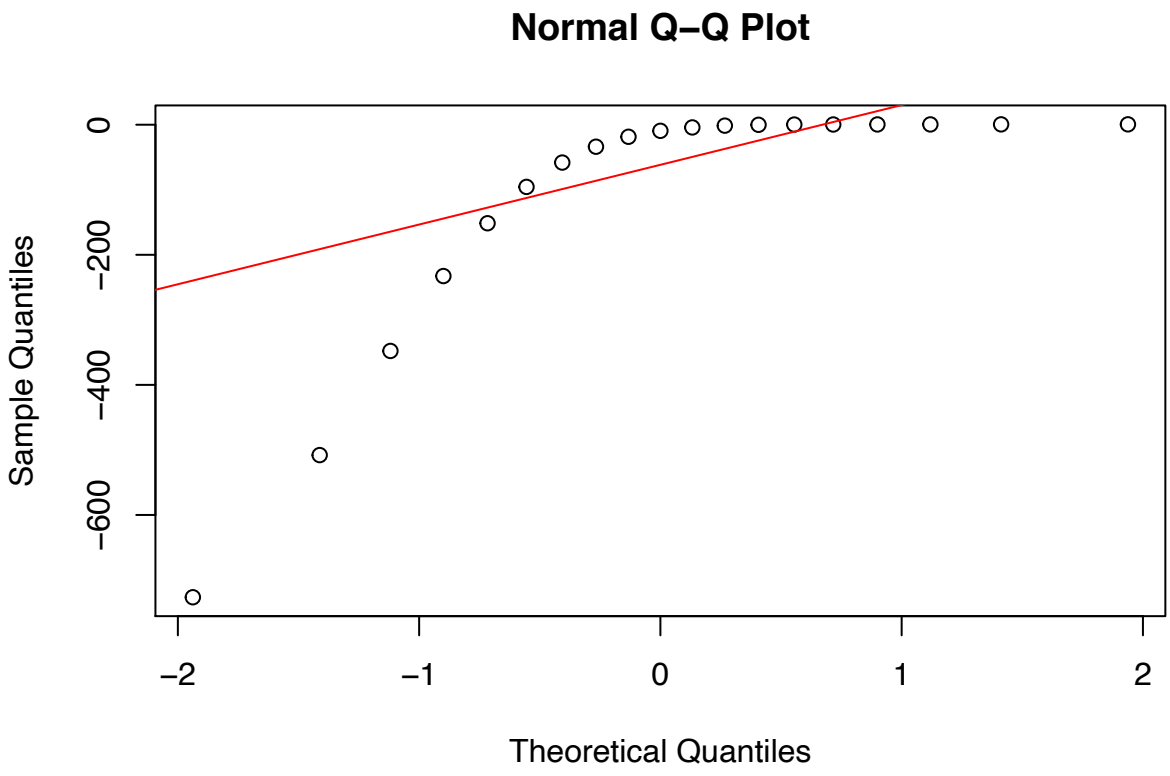

```
# The variables appear to be related nonlinearly based on the plot

# b) We create a QQ plot with the residuals and determine
#whether they are normally distributed
# or follow a skew distribution
# Firt compute the residuals
residuals <- with(pressure, pressure-(0.168+0.007*temperature)^(20/3))
qqnorm(residuals); qqline(residuals, col=2)
```




```
# There apears to be a right skew to the distribution

# c) Now we apply a power transformation of y^(3/20) to the pressure data values

Yt = Y ^(3/20)
plot(X,Yt,xlab="Temperature", ylab="Pressure", main = "Temperature vs Pressure Plot")
```



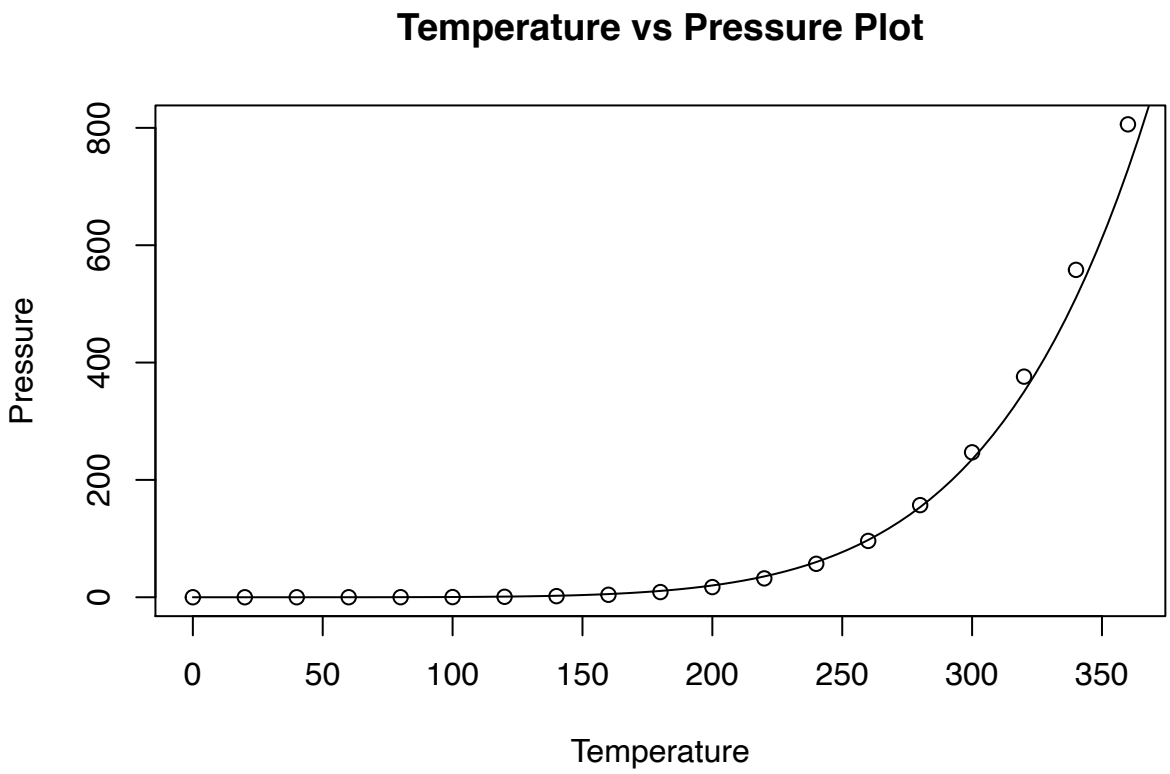

```
# Now there is clearly a linear relationship between the data evident

# d) Now we compute the residuals for the difference b
pressure[,2] = pressure[,2]^(3/20)
residuals1 <- with(pressure, pressure-(0.168+0.007*temperature)^(20/3))
qqnorm(residuals1); qqline(residuals1, col=2)
```



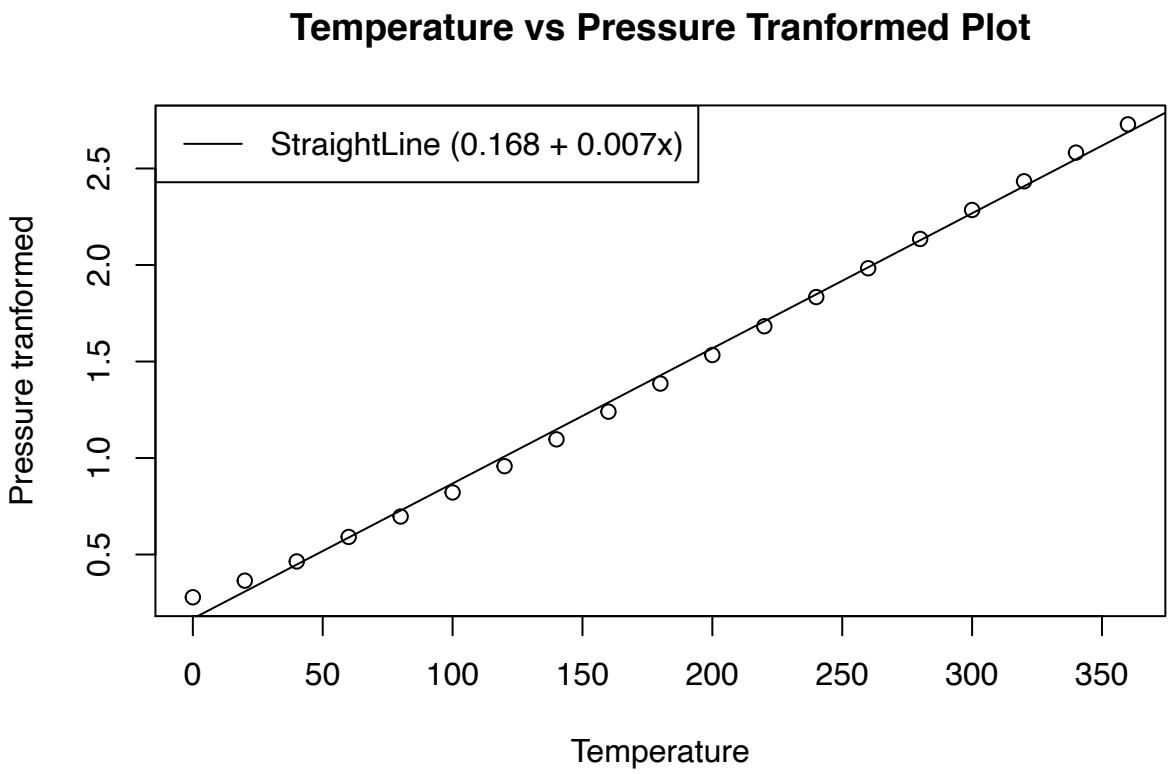

```
pressure[,2] = pressure[,2]^(20/3)

# The residuals do not follow a normal distribution
```

Question 6

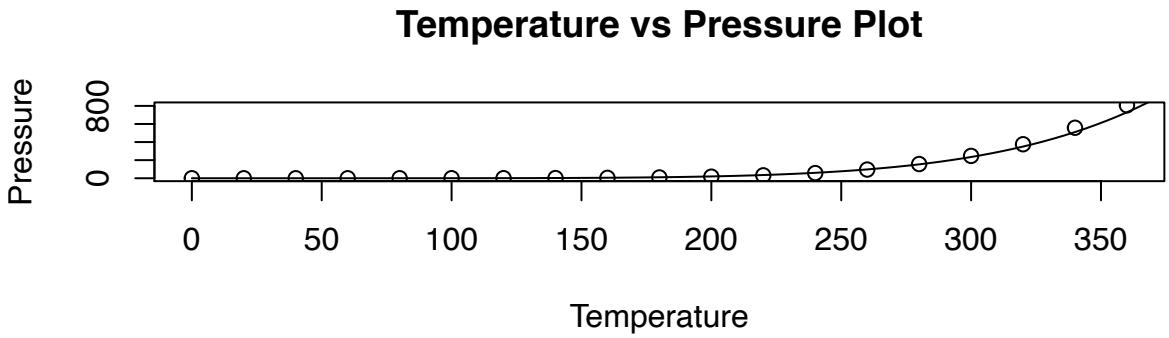
```
# a) We plot pressure against temperature and pass a curve through
# the data using the curve command
X = pressure[,1]
Y = pressure[,2]

plot(X,Y,xlab="Temperature", ylab="Pressure", main = "Temperature vs Pressure Plot")
curve((0.168+0.007*x)^(20/3), from = 0, to = 400, add = TRUE)
```

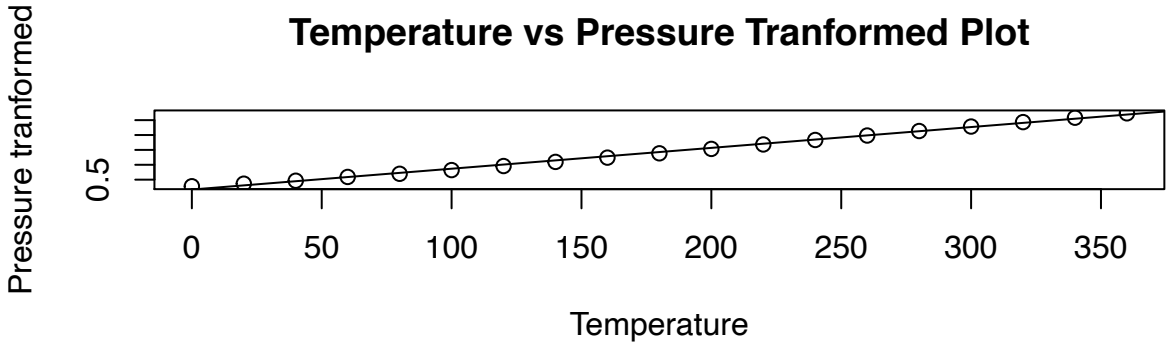



```r
# b) We apply the power tranformation y^3/20 to the pressure data
# We plot this and use abline function to pass a straight line through points

Yt = Y^(3/20)

plot(X,Yt,xlab="Temperature", ylab="Pressure tranformed",
     main = "Temperature vs Pressure Tranformed Plot")
# Use abline function to plot staight line through the points
abline(0.168,0.007)

#c) Add A suitable title to the graph and add a legend to indicate straight line

legend("topleft", legend=c("StraightLine (0.168 + 0.007x)"), col=1, lty=1)
```

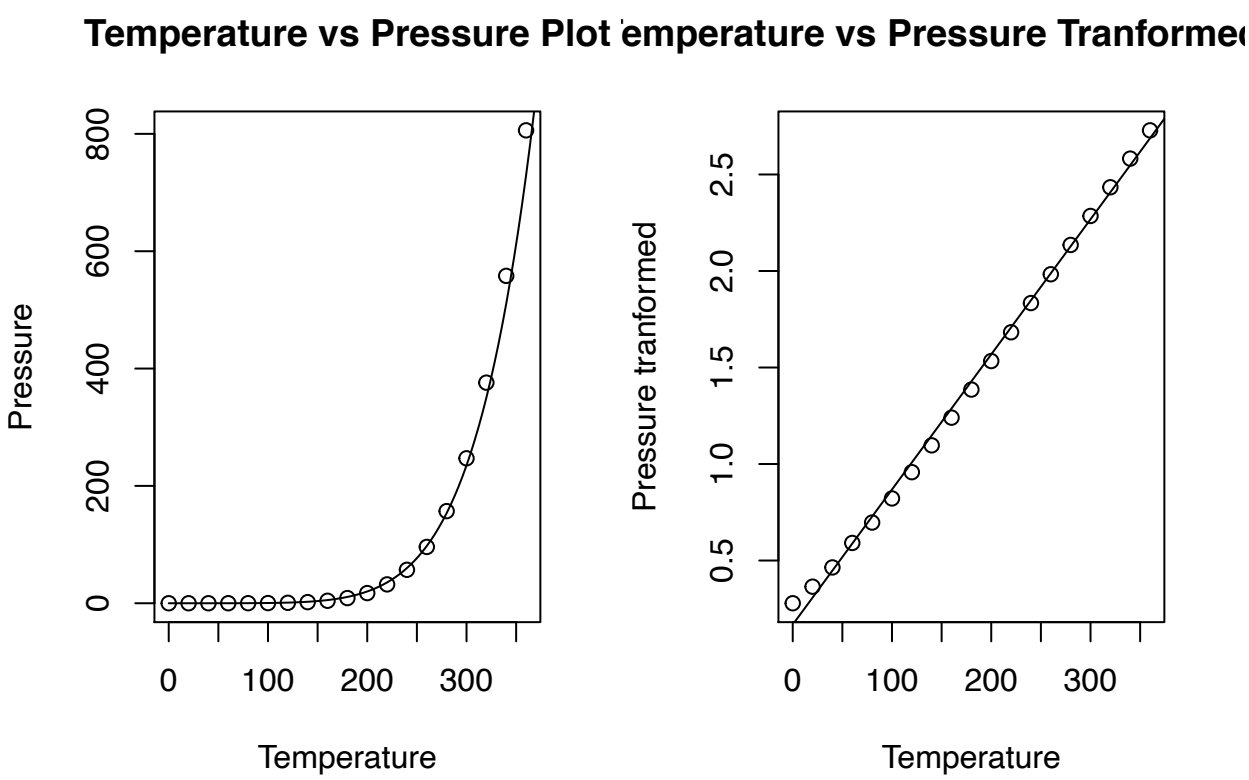



```
# d) we redo the above plots using the mf function to display
#them in a 2x1 layout then in a 1x2 layout

# use mfrow function to plot graphs in 2x1 layout
par(mfrow = c(2,1))

plot(X,Y,xlab="Temperature", ylab="Pressure", main = "Temperature vs Pressure Plot")
curve((0.168+0.007*x)^(20/3), from = 0, to = 400, add = TRUE)

plot(X,Yt,xlab="Temperature", ylab="Pressure tranformed",
     main = "Temperature vs Pressure Tranformed Plot")
abline(0.168,0.007)
```






```r
# use mfrow function to plot graphs in 1x2 layout
par(mfrow = c(1,2))

plot(X,Y,xlab="Temperature", ylab="Pressure",
     main = "Temperature vs Pressure Plot")
curve((0.168+0.007*x)^(20/3), from = 0, to = 400, add = TRUE)

plot(X,Yt,xlab="Temperature", ylab="Pressure tranformed",
     main = "Temperature vs Pressure Tranformed Plot")
abline(0.168,0.007)
```


Temperature vs Pressure Plot
emperature vs Pressure Tranforme
Pressure
0
200
400
600
800
0
100
200
300
Temperature
Pressure tranformed
0.5
1.0
1.5
2.0
2.5
0
100
200
300
Temperature