Statistics 1 Unit 5

Group 8

January 16, 2018

Contents

1 Task 76

```
# Task 76

find_default_correlation <- function (
   asset_correl_coefficient)
{
  d <- qnorm(0.025)
  number_of_obligors <- 100
  number_of_simulations <- 1000
  default_scenar_matrix <- matrix(nrow =
     number_of_simulations,
                                  ncol =
                                     number_of_obligors
                                     )
  for ( i in 1:number_of_simulations)
  {
    Z0 <- rnorm(n = 1, mean = 0, sd = 1)
    for ( j in 1:number_of_obligors)
    {
      Z <- rnorm(n = 1, mean = 0, sd = 1)
      X <- sqrt(asset_correl_coefficient)*Z0 +
        sqrt(1-asset_correl_coefficient)*Z
      if (X < d )
        default_scenar_matrix[i,j] <- 1
      else
        default_scenar_matrix[i,j] <- 0
    }
  }

  default_correl_matrix <- cor(default_scenar_matrix)
  return(default_correl_matrix)
}


matrix <- find_default_correlation (
  asset_correl_coefficient = 0.3)
```

Let's define $p_i = P(Y_i = 1)$ and

$$Var(Y_i) = E(D_i^2) - p_i^2 = E(D_i) - p_i^2 = p_i - p_i^2$$

This gives us the next correlation formula:

$$\rho(D_i, D_j) = \frac{E(D_i D_j) - p_i p_j}{\sqrt{(p_i - p_i^2)(p_j - p_j^2)}}$$

Define: $\pi = P(D_i = 1), i \in (1, 100)$
$\pi_k = P(D_{i_1} = 1 ... D_{i_k} = 1), i \in (1, 100)$

$$E(D_i) = E(D_i^2) = \pi$$
$$E(D_i D_j) = P(D_i = 1, D_j = 1) = \pi_2$$

Now:

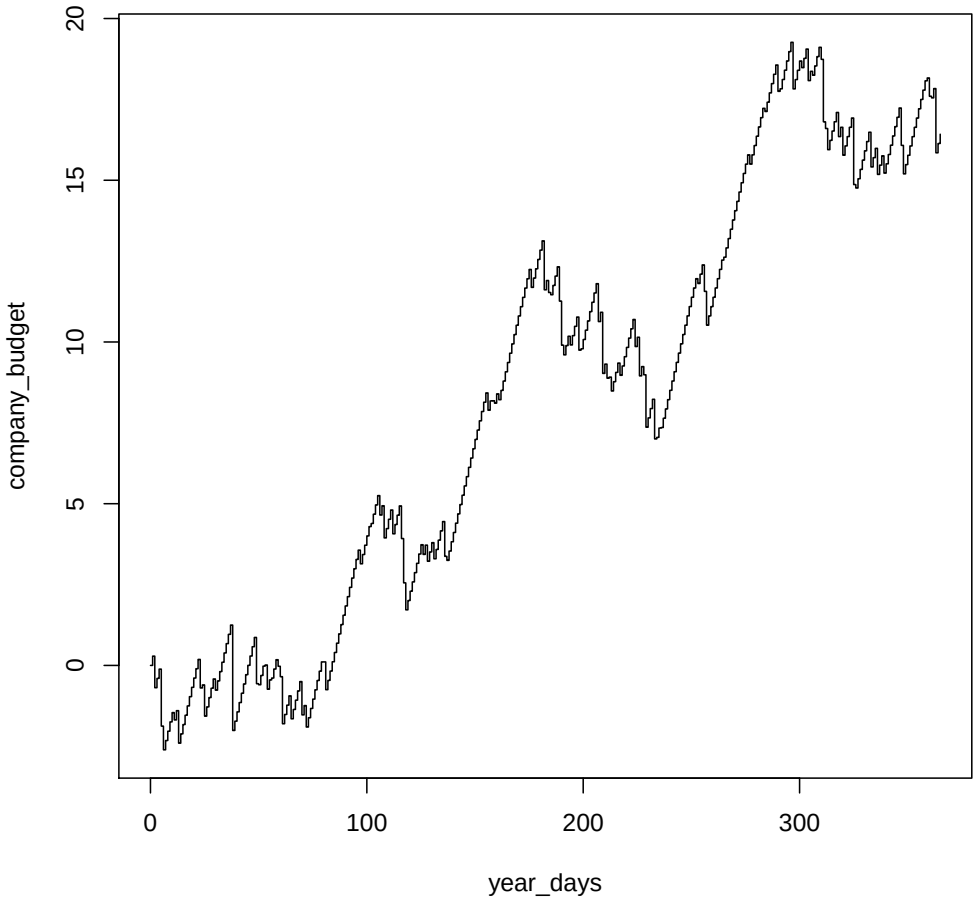
$$\rho(D_i D_j) = \frac{\pi_2 - \pi^2}{\pi - \pi^2}$$

, i $\neq$ j

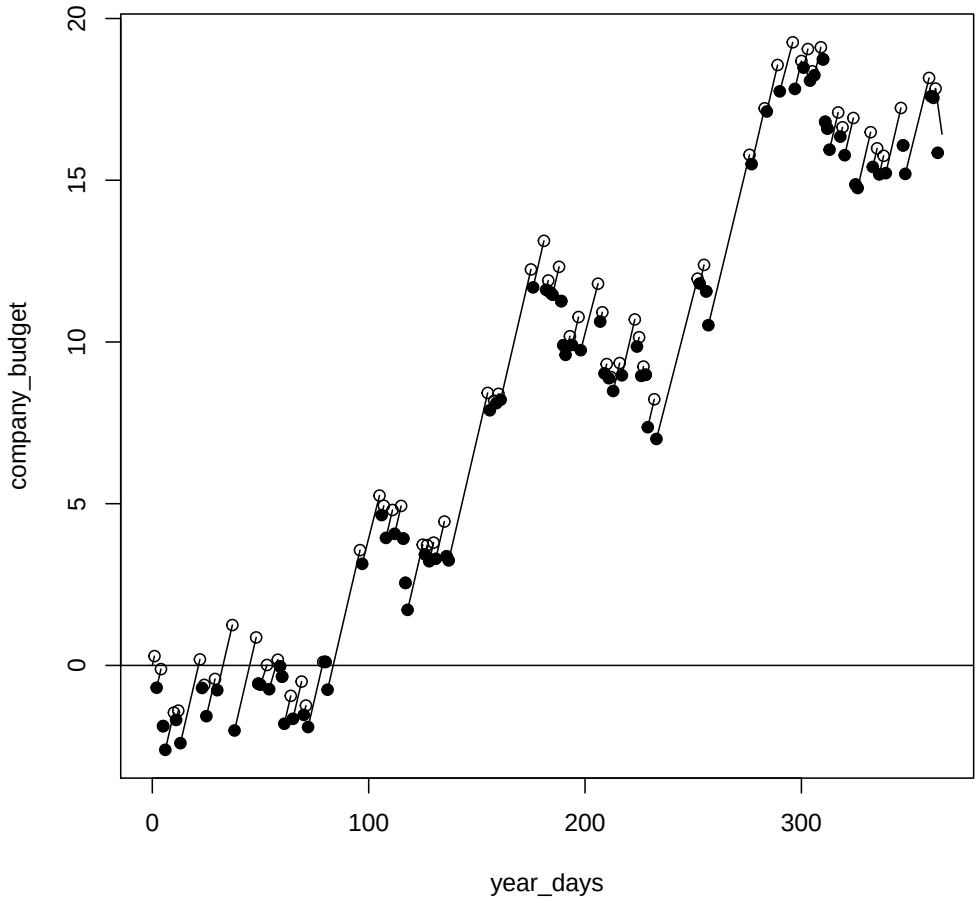
Using this formula, the theoretical value of the default correlation can be determined.

2 Task 79 and Task 88

```
# Task 79


# A

number_of_claims <- rpois(n = 1,lambda = 100)

year_days <- 0:365

days_of_claims <- sort(floor(runif(number_of_claims,
   min = 1, max = 366)))

claims_amount <- rgamma(n = number_of_claims, shape =
   2, rate = 2)

earnings <- year_days * 105/365

company_budget <- c(rep(0,times = 366))


claim_number <- 0

for(i in 1:366){
  if(i == 1){
    # initial value
    company_budget[i] <- 0
    next
  }
  if( (i-1) %in% days_of_claims){
    claim_number <- claim_number + 1
    payment <- claims_amount[claim_number]
    company_budget[i] <- earnings[i] -  payment
    earnings = earnings - payment
  }
  else{
    company_budget[i] <- earnings[i]
  }
}


plot(year_days, company_budget, type = "s")
```


company_budget
0
5
10
15
20
0
100
200
300
year_days

```
# Task 88

black_points <- c()

white_points <- c()


for (i in 2:length(company_budget)) {
  if (company_budget[i] < company_budget[i-1]) {
    white_points <- c(white_points, i-1)
    black_points <- c(black_points, i)
  }
}


lw <- length(white_points)

lt <- length(year_days)


plot(year_days, company_budget, type = "n")

points(year_days[white_points],company_budget[
   white_points])

points(year_days[black_points],company_budget[
   black_points], col = "black", pch = 19)

abline(h=0)


lines(x = c(year_days[1], year_days[white_points[1]]),
   c(company_budget[1],company_budget[white_points
   [1]]))

lines(c(year_days[white_points[lw]],year_days[lt]),c(
   company_budget[white_points[lw]],company_budget[lt
   ]))


for (i in 2:lw) {
  lines(c(year_days[black_points[i-1]], year_days[
     white_points[i]]),
        c(company_budget[black_points[i-1]],
          company_budget[white_points[i]]))
}
```




```
# B

number_of_simulations <- 1000

final_amount <- NULL

min_current_amount <- NULL


for(j in 1:number_of_simulations){
  number_of_claims <- rpois(n = 1,lambda = 100)
  year_days <- 0:365
  days_of_claims <- sort(floor(runif(number_of_claims,
      min = 1, max = 366)))
```

```
  claims_amount <- rgamma(n = number_of_claims, shape
     = 2, rate = 2)
  earnings <- year_days * 105/365
  company_budget <- c(rep(0,times = 366))

  claim_number <- 0
  for(i in 1:366){
    if(i == 1){
      # initial value
      company_budget[i] <- 0
      next
    }
    if( (i-1) %in% days_of_claims){
      claim_number <- claim_number + 1
      payment <- claims_amount[claim_number]
      company_budget[i] <- earnings[i] -  payment
      earnings = earnings - payment
    }
    else{
      company_budget[i] <- earnings[i]
    }
  }

  final_amount[j] <- company_budget[length(
     company_budget)]
  min_current_amount[j] <- min(company_budget)

}


expected_final_amount <- mean(final_amount)

expected_min_current_amount <- mean(min_current_amount
   )
```

3 Task 83

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(gridExtra)
```

```
## Loading required package: gridExtra
```

3.1 a)

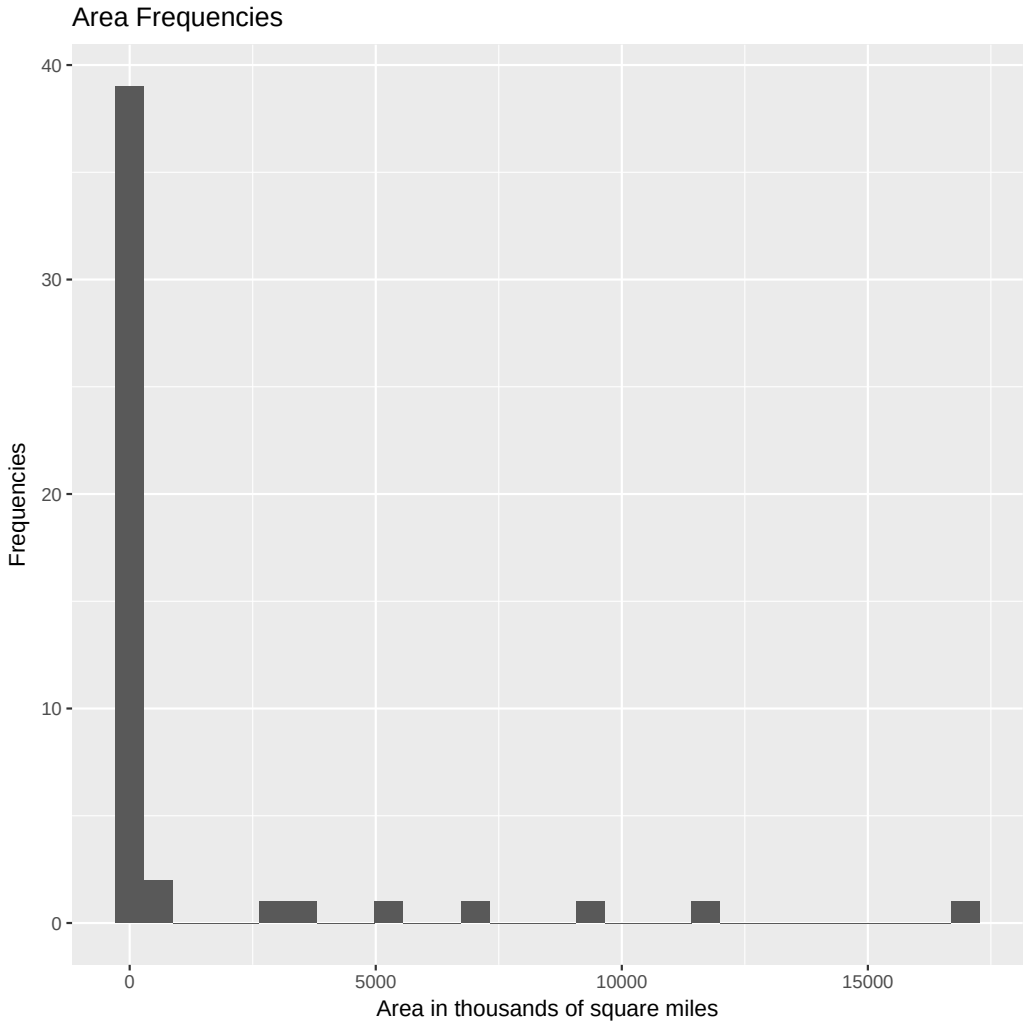
```
summary(islands)
```

```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     12.0    20.5    41.0  1252.7   183.2 16988.0
```

We see that our data has a large maximum compared to the quartiles or the minimum. So a simple histogram will definitely be feebly informative
We plot the histogram with the frequencies on the y-axis and the areas in thousands of square miles of the 48 land masses on the x-axis:

```
qplot(islands,
      geom="histogram",
      main="Area Frequencies",
      ylab="Frequencies",
      xlab="Area in thousands of square miles")
```

```
  ## `stat_bin()` using `bins = 30`. Pick better value with##
                          `binwidth`.
```

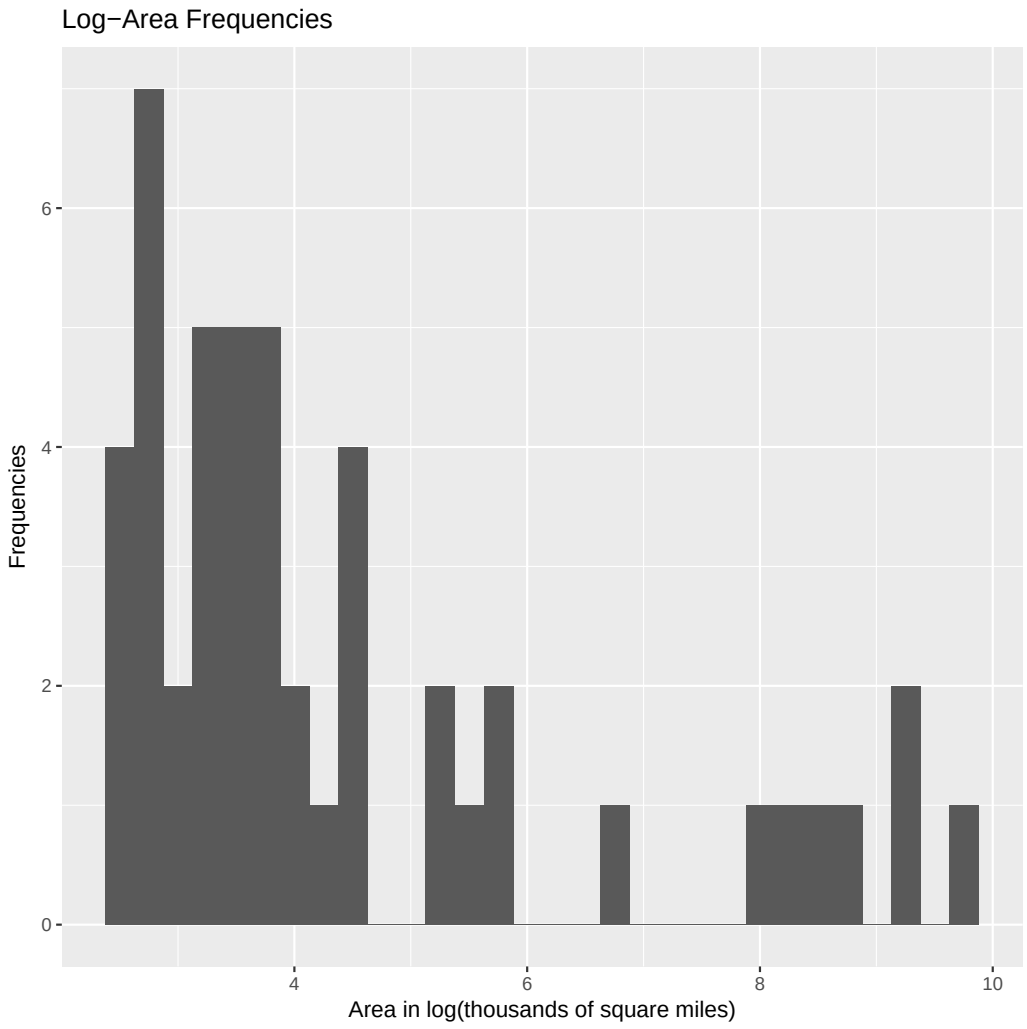


3.2 b)

Taking logarithm will improve our result, because large values will be truncated.

```
log_isl <- log(islands)

qplot(log_isl,
      geom="histogram",
      main="Log-Area Frequencies",
      ylab="Frequencies",
      xlab="Area in log(thousands of square miles)")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with##
                          `binwidth`.
```

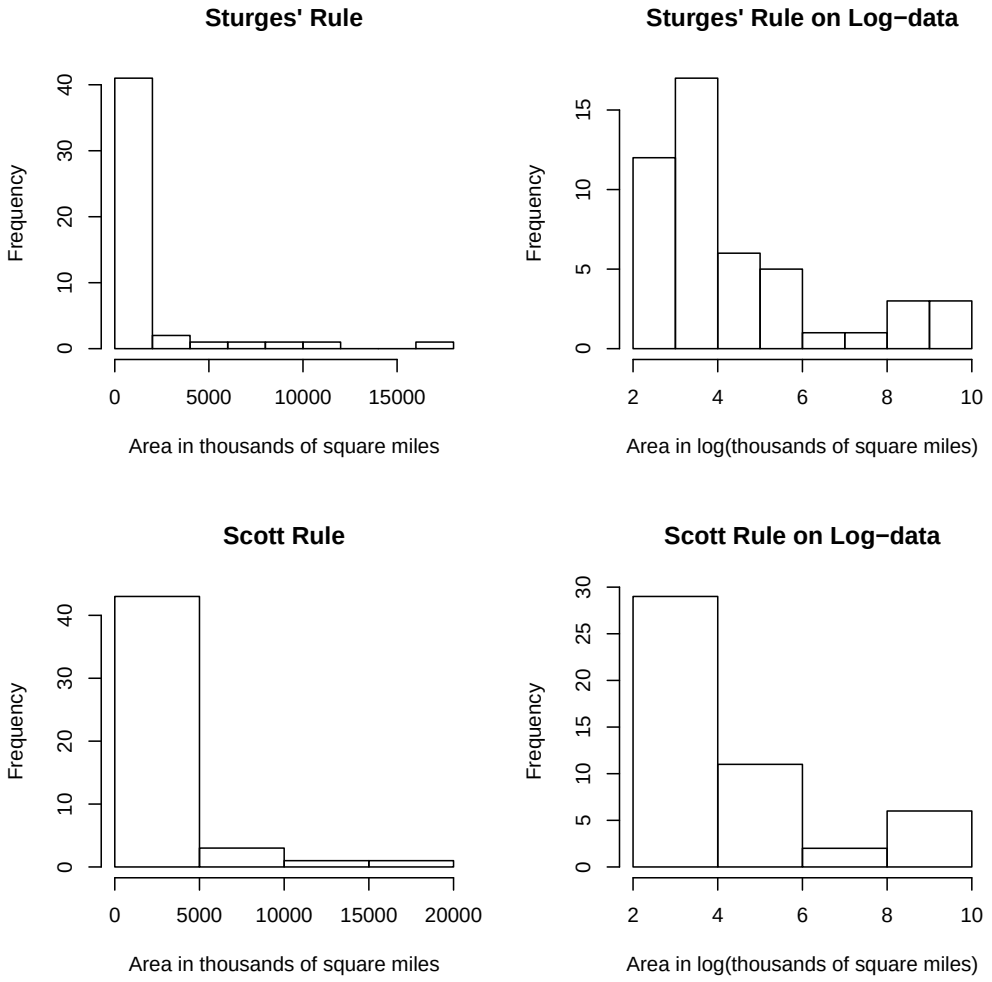


Now, the breaks between the bins are now more reasonable. However, the essence of the x-axis becomes vague.

3.3 c)

```
par(mfrow = c(2, 2))
```

```
plot1 <- hist(islands,
              main="Sturges' Rule",
              breaks = "Sturges",
              ylab="Frequency",
              xlab="Area in thousands of square miles
                 ")

plot2 <- hist(log_isl,
              main="Sturges' Rule on Log-data",
              breaks = "Sturges",
              ylab="Frequency",
              xlab="Area in log(thousands of square
                 miles)")

plot3 <-  hist(islands,
              main="Scott Rule",
              breaks = "Scott",
              ylab="Frequency",
              xlab="Area in thousands of square miles
                 ")

plot4 <- hist(log_isl,
              main="Scott Rule on Log-data",
              breaks = "Scott",
              ylab="Frequency",
              xlab="Area in log(thousands of square
                 miles)")
```

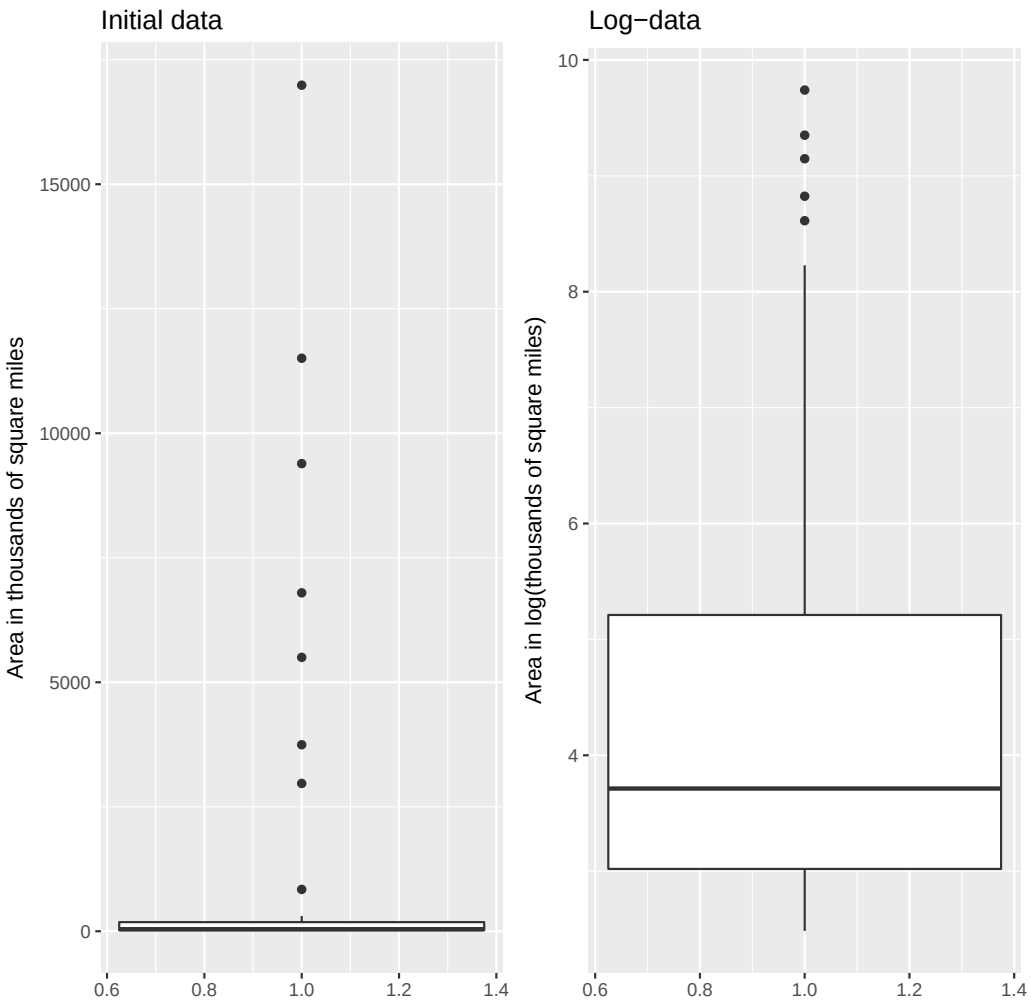


For the non-log data we get almost the same results in both methods, but log-data is more informative. For that data we would suggest using Sturges' Rule, since Scott's Rule only comes up with 4 different bins which is not very informative.

3.4 d)

```
par(mfcol=c(1, 2))

plot5 <- qplot(islands,
               x=1,
```

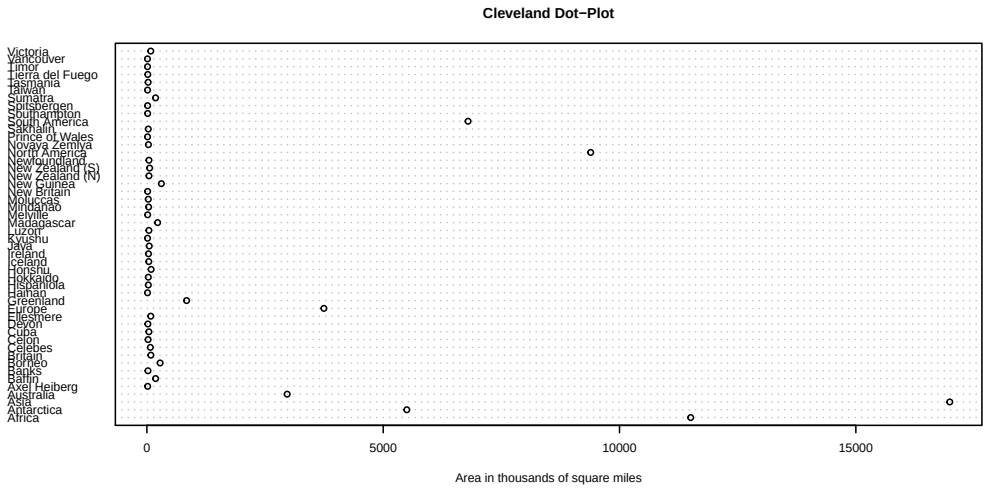
```
               geom = "boxplot",
               main="Initial data",
               ylab="Area in thousands of square miles
                  ",
               xlab="")

plot6 <- qplot(log_isl,
               x=1,
               geom = "boxplot",
               main="Log-data",
               ylab="Area in log(thousands of square
                  miles)",
               xlab="")

grid.arrange(plot5, plot6, ncol=2)
```

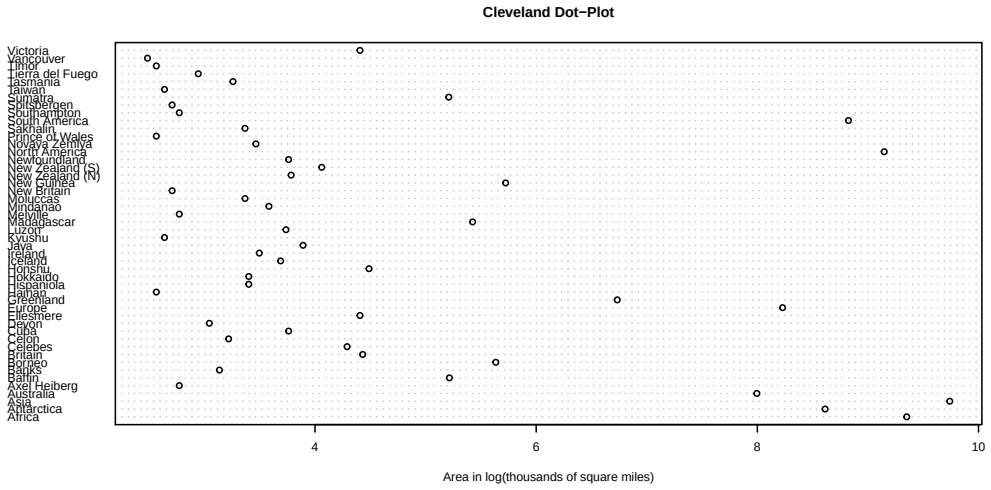


Since the box-plot has big outliers, the non-log data does not make any sense.

3.5 e)

```
par(mfcol=c(2,1))

plot7 <- dotchart(islands,
                  cex=0.5,
                  main="Cleveland Dot-Plot",
                  xlab="Area in thousands of square
                     miles")

plot8 <- dotchart(log_isl,
                  cex=0.5,
                  main="Cleveland Dot-Plot",
                  xlab="Area in log(thousands of
                     square miles)")
```





We see that on the first dot-plot on initial data, there is a little difference in the areas of many variables, since our plot extends to over 15000 on the x-axis and the most land masses appears to be small. Thus this is not the best way to represent the data.On the other hand the log-data looks better in terms of representation of the different areas of the land masses.

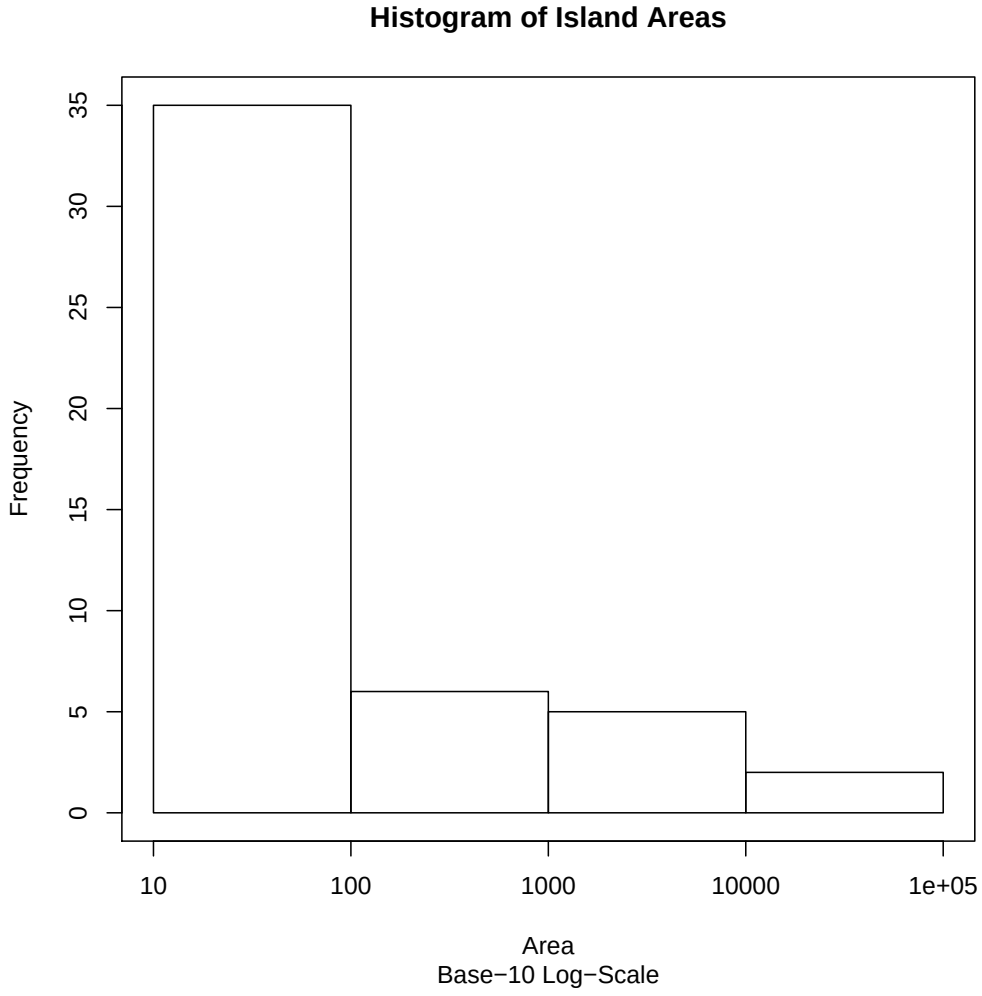
3.6 f)

As all the graphs on initial data gave us the worst results in terms of graphical representation, we would suggest to use the log-data inspite of it is unclear of what is log of square miles.
We supose that the boxplot and dot-plot on the log-data are the ones which are the most informative. The dot-plot reveals information about the relative

differences within variables and the boxplot is able to capture the location of the distribution.

4 Task 84

```
par(mfcol = c(1, 1))

hist(log(islands, 10),
     breaks = "Scott",
     axes = FALSE,
     xlab = "Area",
     main = "Histogram of Island Areas")

axis(1, at = 1 : 5, labels = 10 ^ (1 : 5))

axis(2)

box()

title(sub = "Base-10 Log-Scale")
```

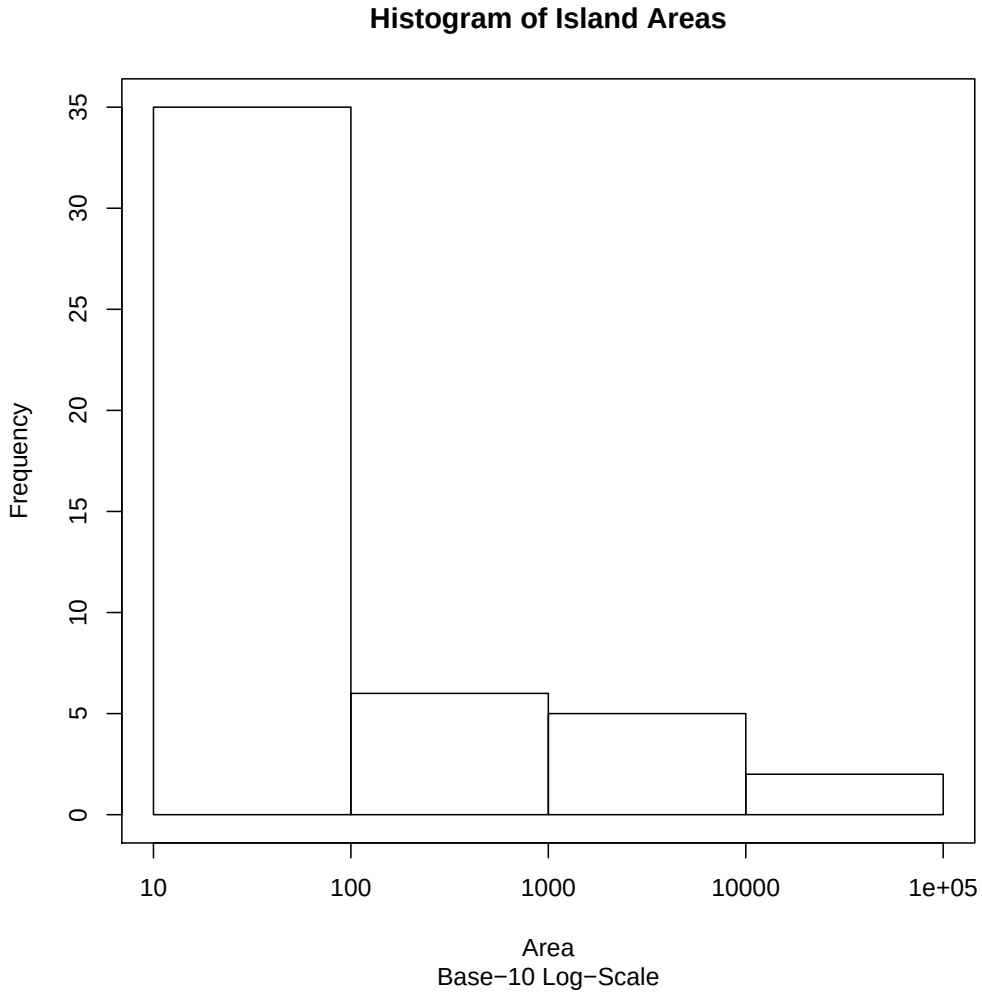


4.1 a)

First, the code draws a histogram using the Scott's Rule, with the log10-data (base-10). But the first part does not draw any axes, only labels the x-axis and writes a main title.
Second, the x-axis is added, where at every boundary of the bins, the actual areas written (and not the logarithmized ones on which the graph is based). Afterwards the y-axis is added and a box is drawn around the plot.
Here, the problem with taking the log of the data is solved. The log x-axis is aligned to the actual data.

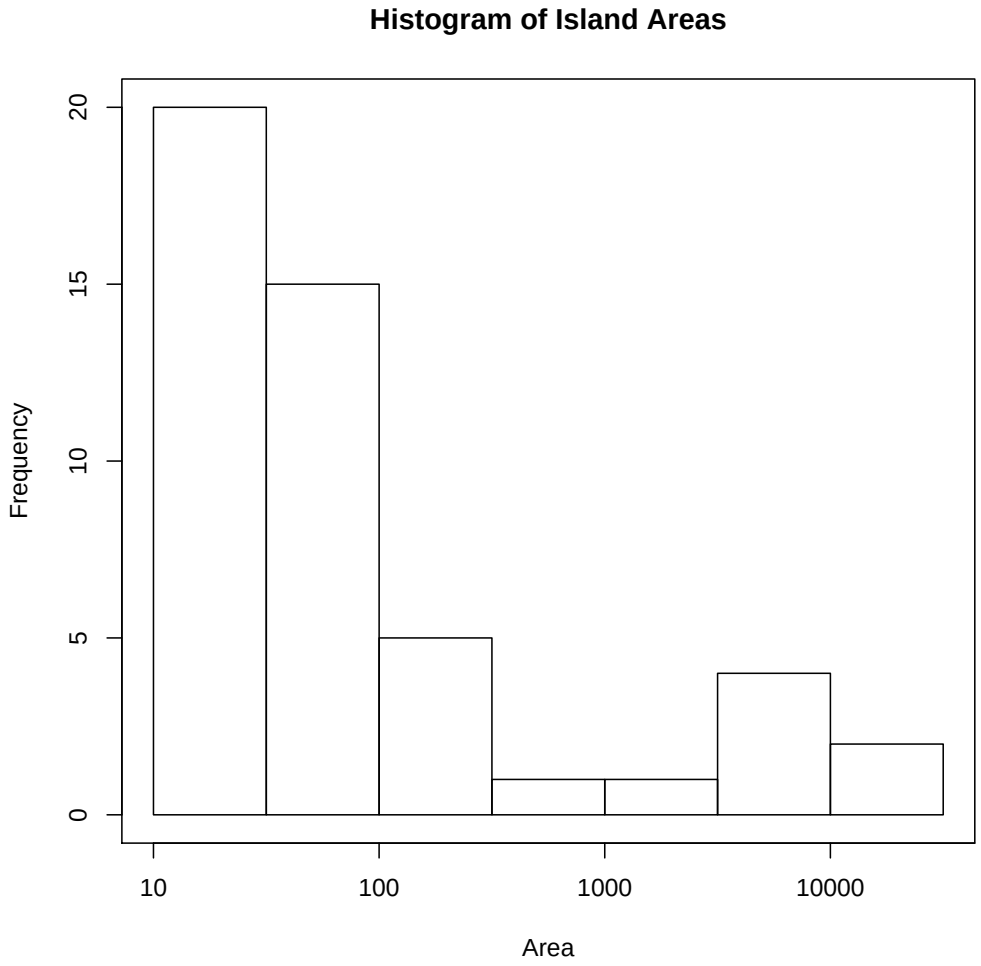
4.2 b)

```
hist(log(islands, 10), breaks = "Scott", axes=F,
     xlab="Area", main="Histogram of Island Areas")

axis(1, at=1:5, labels = 10^(1:5))

axis(2)

box()

title(sub = "Base-10 Log-Scale")
```



4.3 c)

```
hist(log(islands, 10),
     breaks = "Sturges",
     axes=FALSE,
     xlab="Area",
     main="Histogram of Island Areas")

axis(1, at=1:8, labels = 10^(1:8))

axis(2)

box()
```

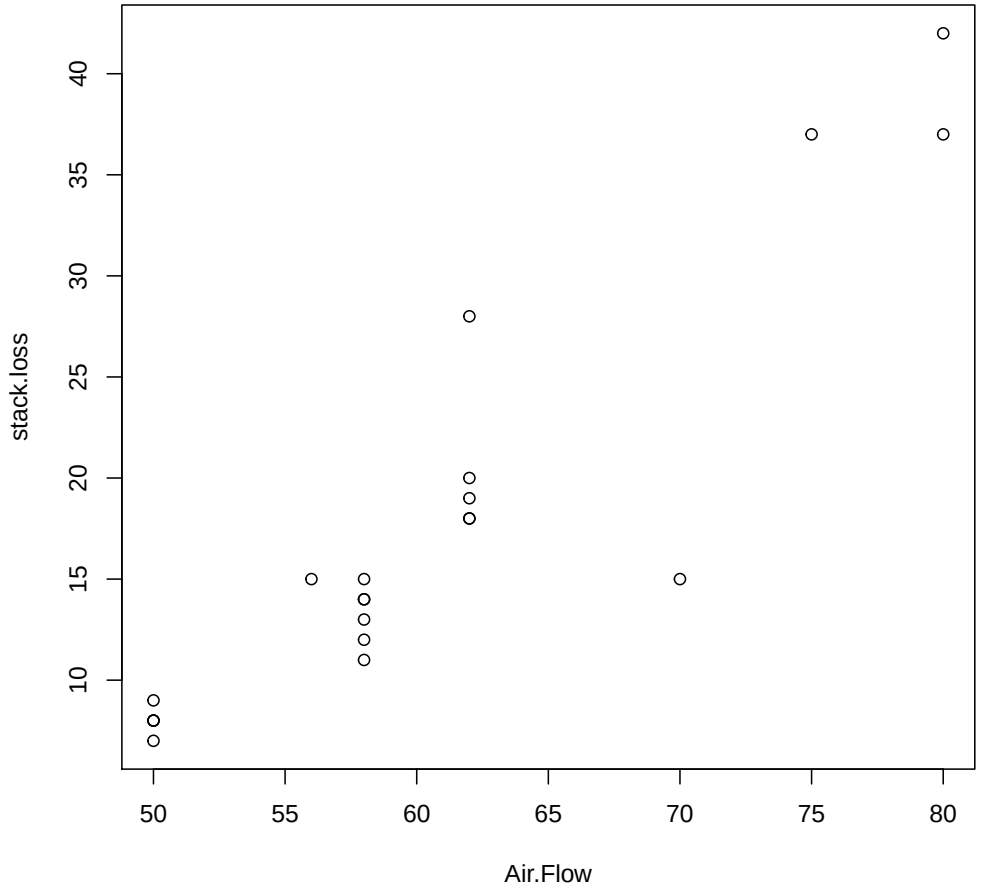


The graph seems to be quite clear without "round" function, so we didn't use it, but there is no problem to insert it in the code if necessary.

5 Task 85

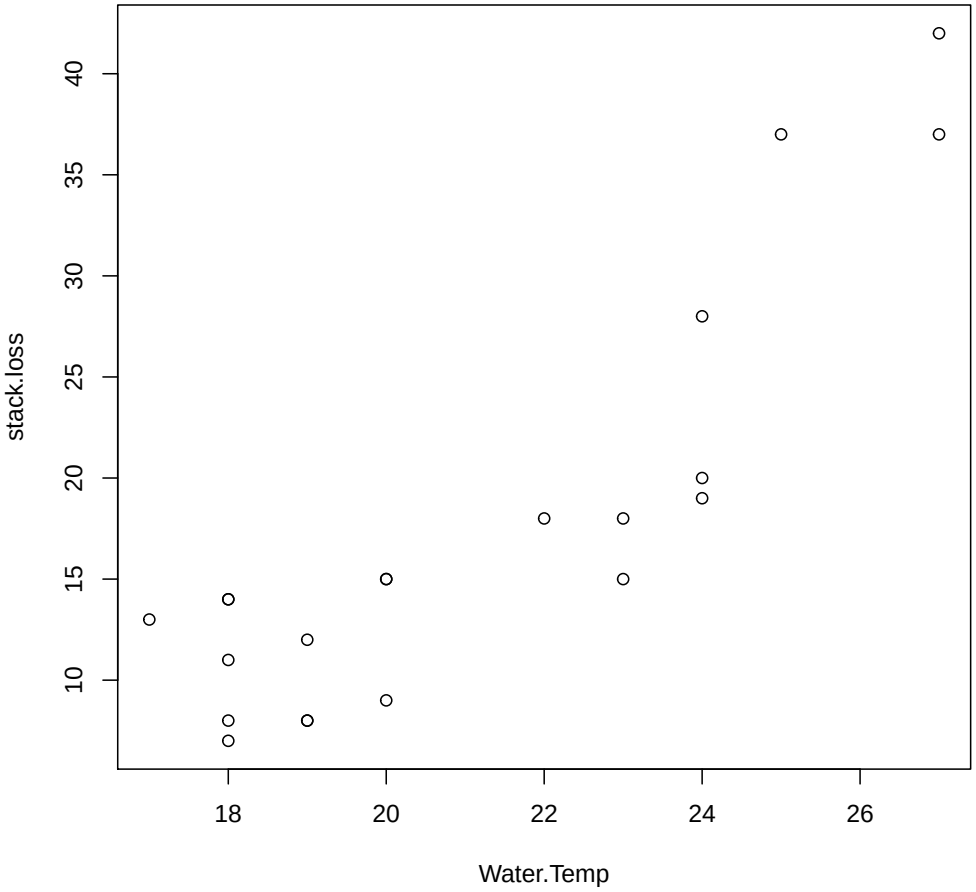
5.1 a)

```
with(stackloss, plot(Air.Flow, stack.loss))
```



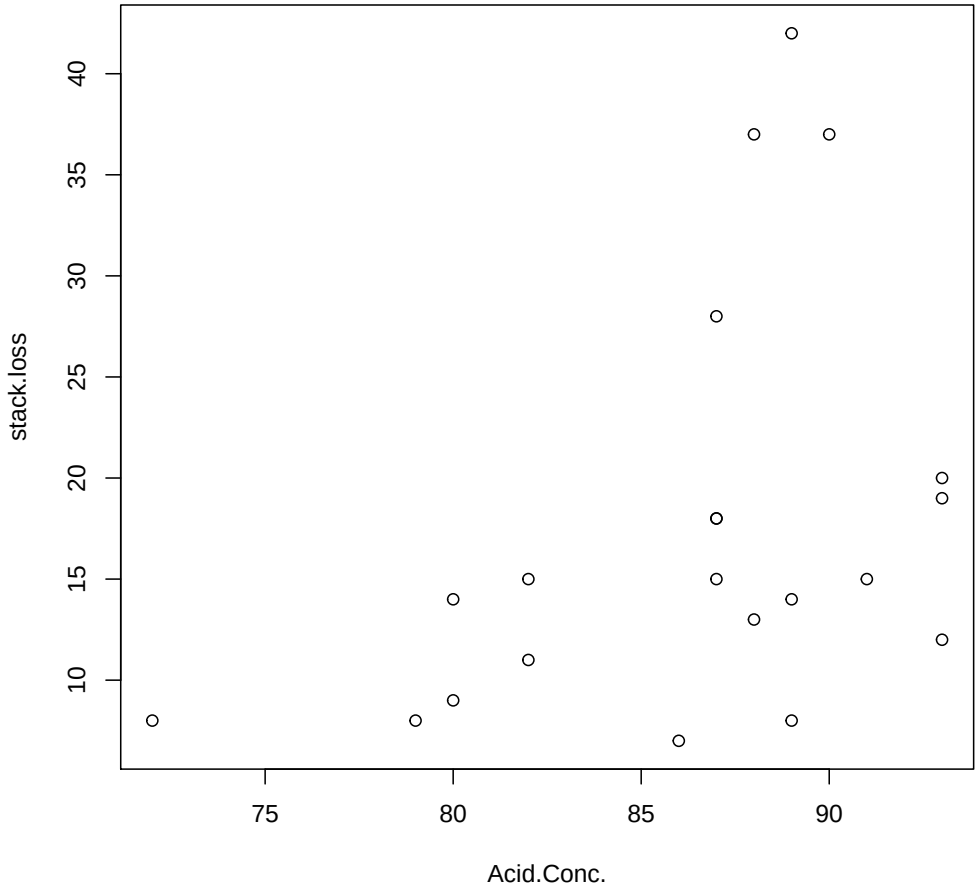
Here we can notice, that at the same value of Air Flow we can have different stack loss. But As Air Flow increases, the stack loss also increases. Thus, the Air Flow has (weak) linear (positive) influence on the stack loss.

```
with(stackloss, plot(Water.Temp, stack.loss))
```



Here, there is a relationship between Water temperature and stack loss, but it seems not that much linear (more like exponential relationship)

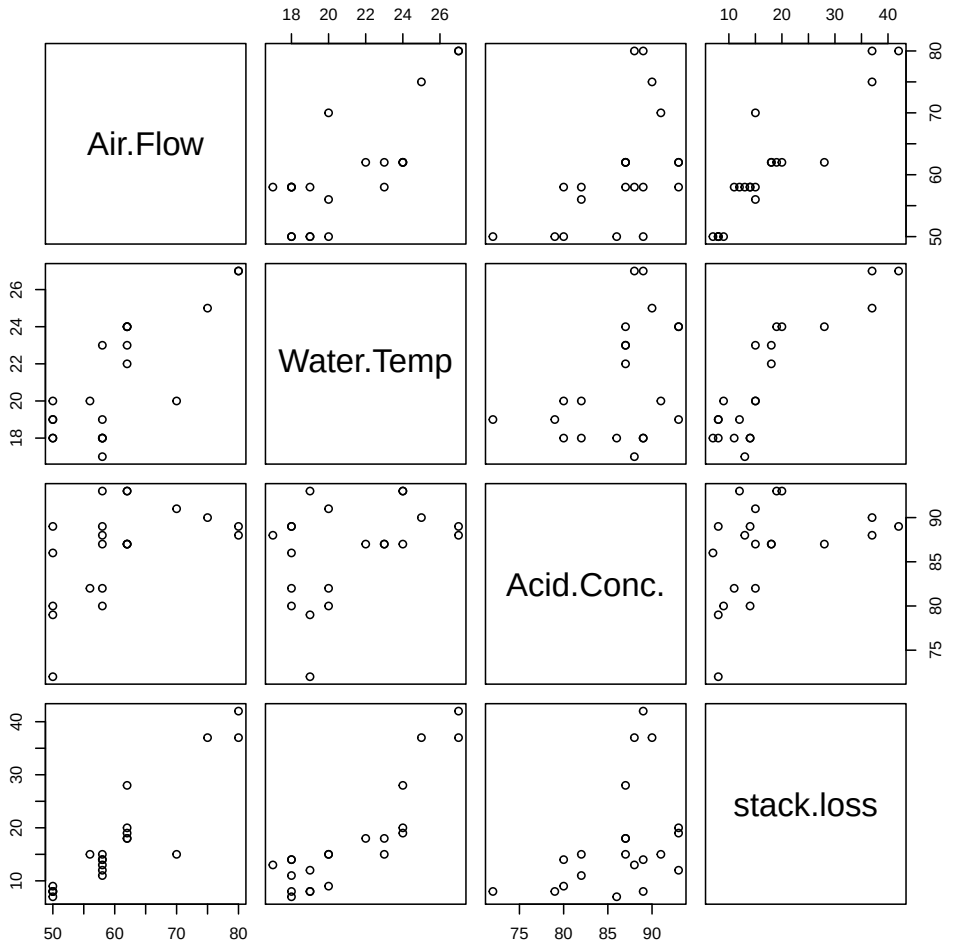
```
with(stackloss, plot(Acid.Conc., stack.loss))
```



The Acid concentration mostly varies from 80 to 95. We can see the positive relationship, but if we exclude outliers (values of stack loss are around 5, 28, 36, 42), the trend will be roughly linear.

5.2 b)

```
pairs(stackloss)
```

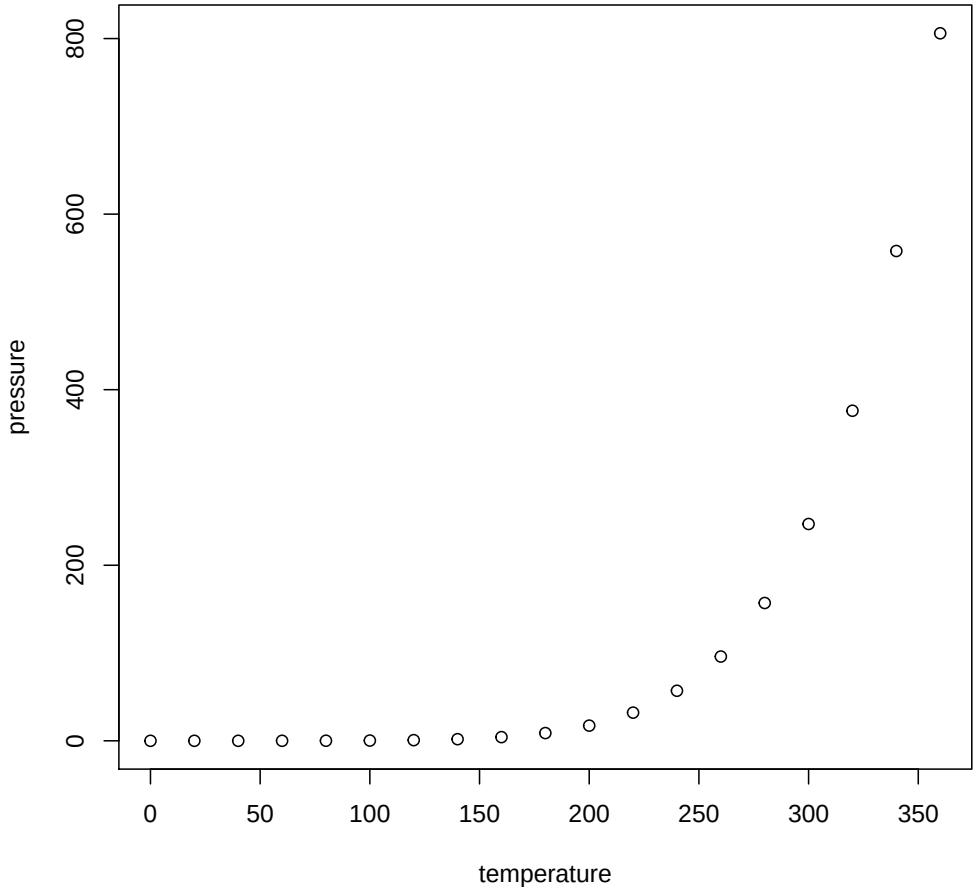


According to this plots, the relationships might be between air flow and water temperature; air flow and stack loss; water temperature and stack loss; acid concentration and stack loss; acid concentration and air flow.

6 Task 86

6.1 a)

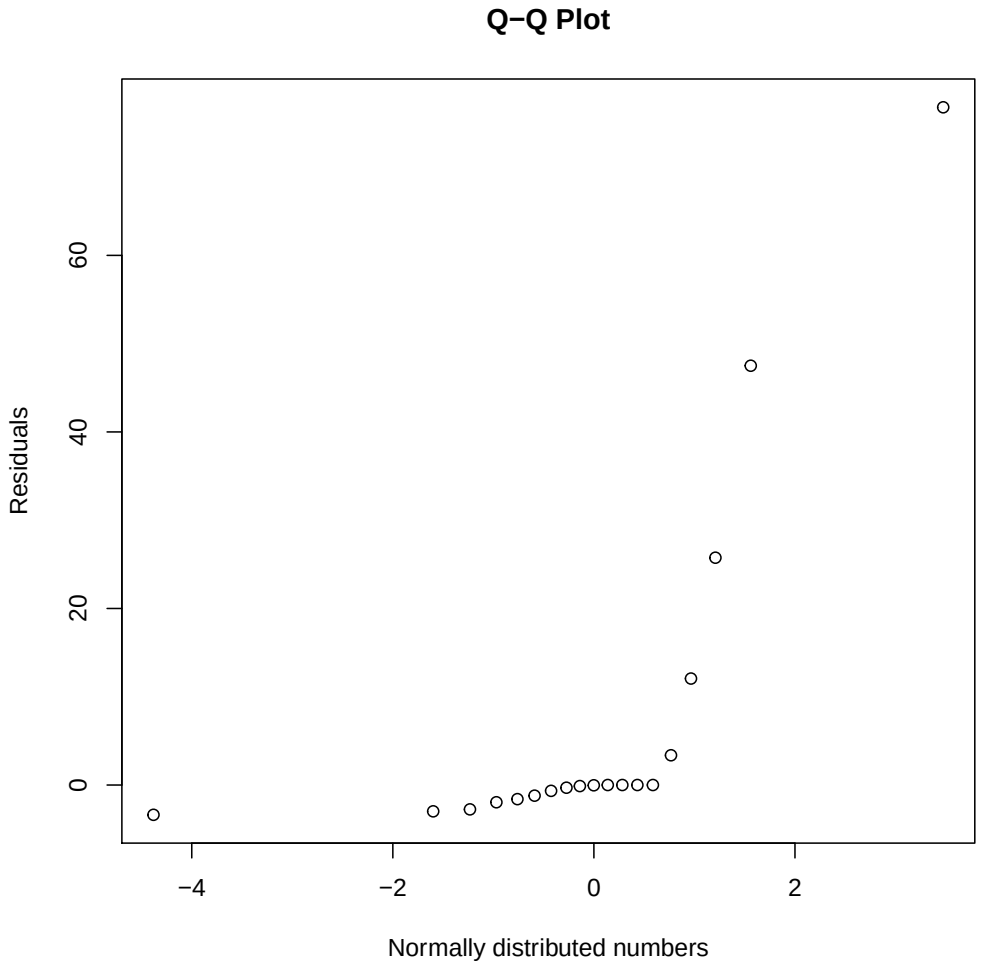
```
plot(pressure~temperature,data=pressure)
```



Variables are related non linearly.

6.2 b)

```
points <- rnorm(10000)

residuals <- with(pressure, pressure - (0.168 + 0.007*
   temperature)^(20/3))

qqplot(points,residuals,xlab="Normally distributed
   numbers",ylab="Residuals",main="Q-Q Plot")
```

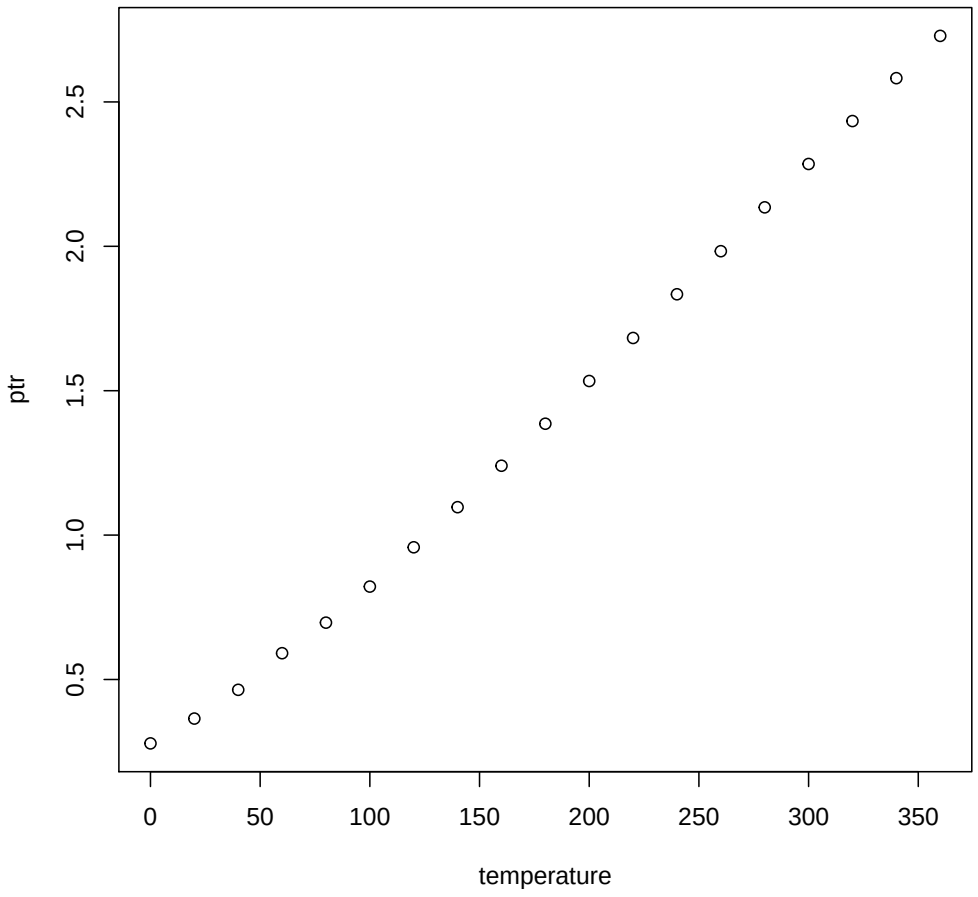


Residuals follow a right-skewed distribution.

6.3 c

```
head(pressure<-transform(pressure,ptr=pressure^(3/20))
   )

##   temperature pressure       ptr
## 1           0   0.0002 0.2787113
## 2          20   0.0012 0.3646508
## 3          40   0.0060 0.4642188
## 4          60   0.0300 0.5909737
```

```
## 5           80    0.0900 0.6968453
## 6          100    0.2700 0.8216835

plot(ptr~temperature,data=pressure)
```

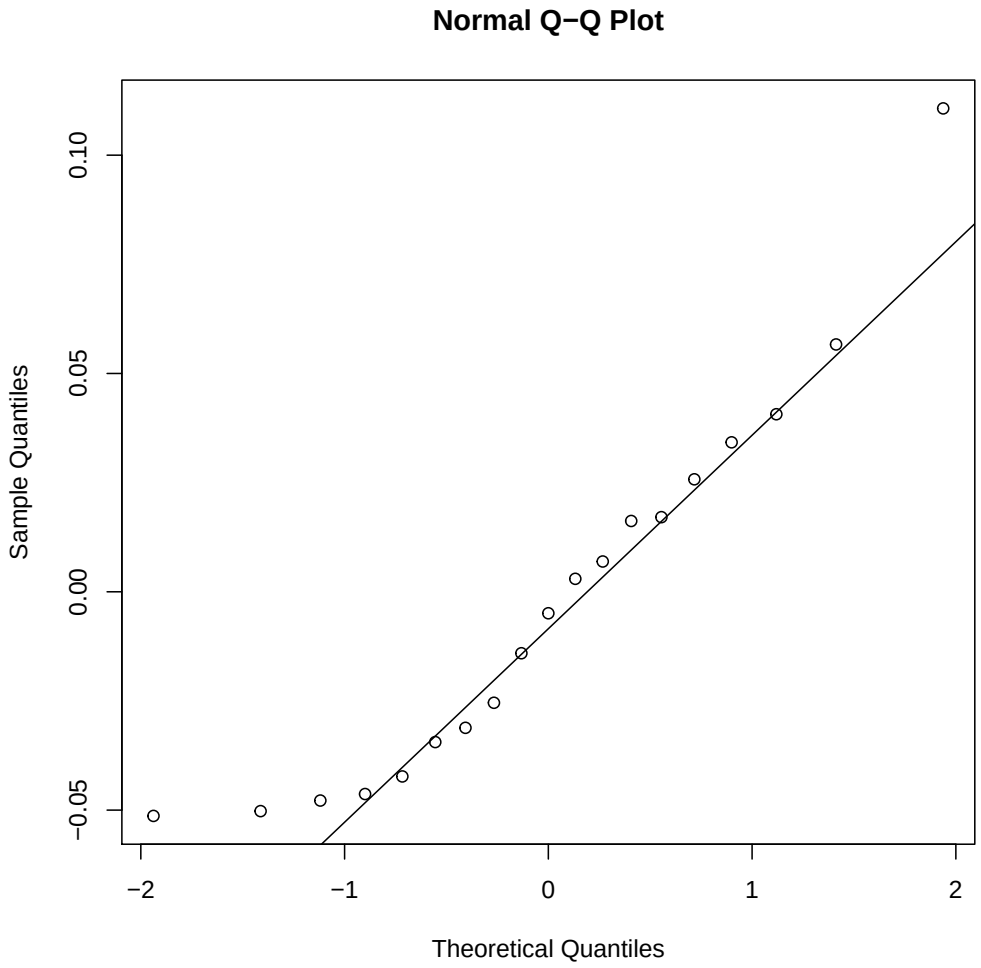


Linear relationship is evident now.

6.4 d)

```
ptr<-pressure^(3/20)
```

```
residuals <- with(pressure, ptr - (0.168 + 0.007*
   temperature))

qqnorm(residuals)

qqline(residuals)
```

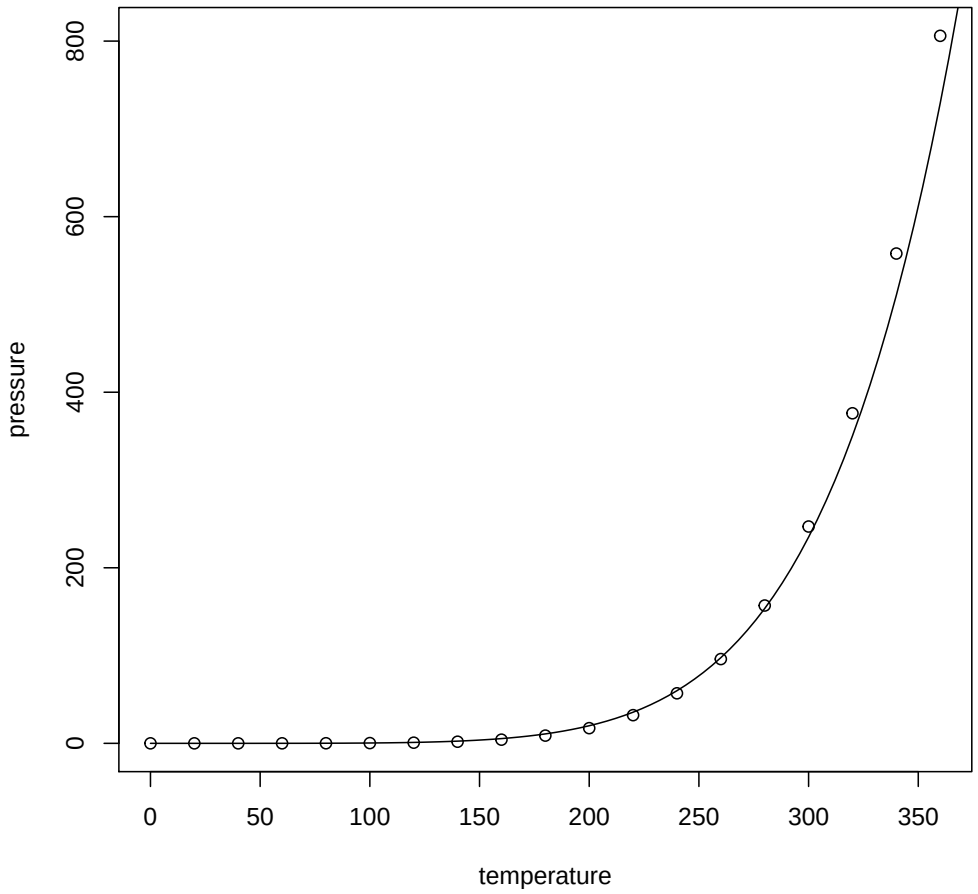


Residuals seem to follow normal distribution.

7 Task 87

7.1 a)

```
plot(pressure~temperature,data=pressure)

curve((0.168 + 0.007*x)^(20/3), from=0, to=400, add=
   TRUE)
```

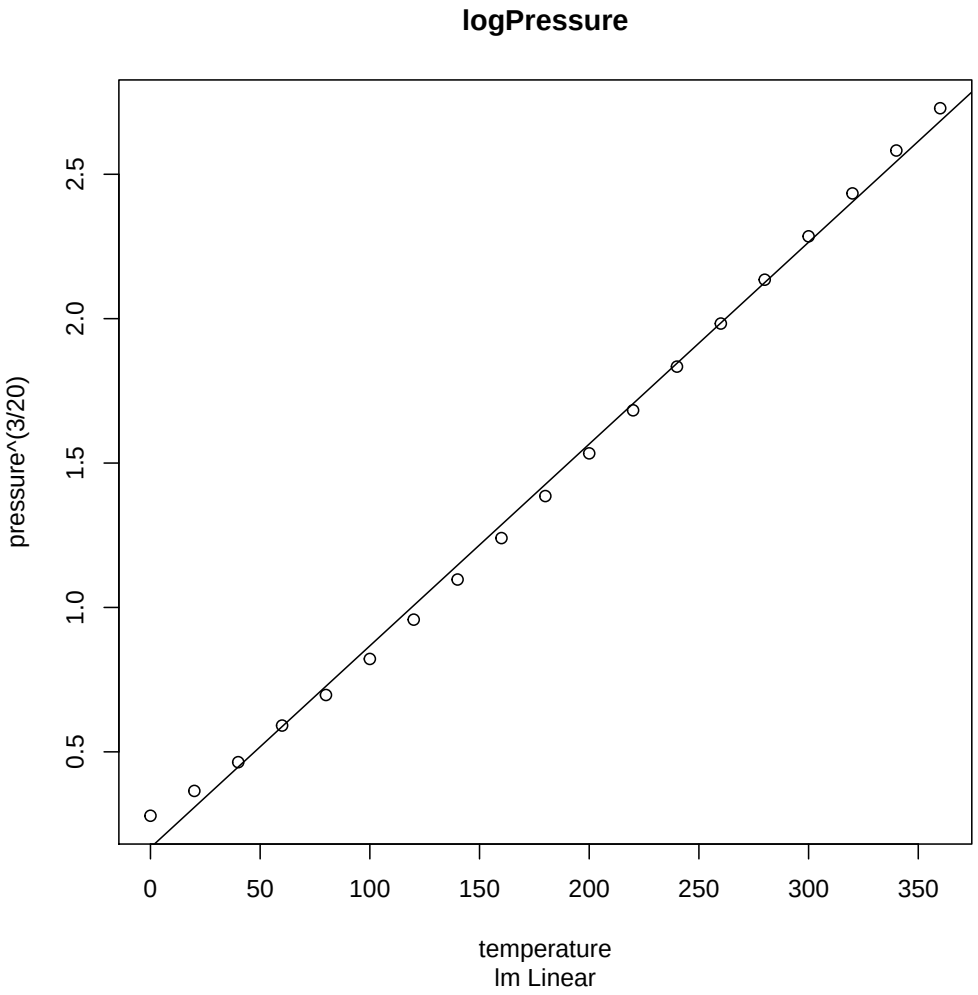


It is not linear.

7.2 b) and c)

```
plot(pressure^(3/20)~temperature,data=pressure)
```

```
abline(lm(pressure^(3/20)~temperature,data=pressure))

title(main="logPressure", sub= "lm Linear")
```

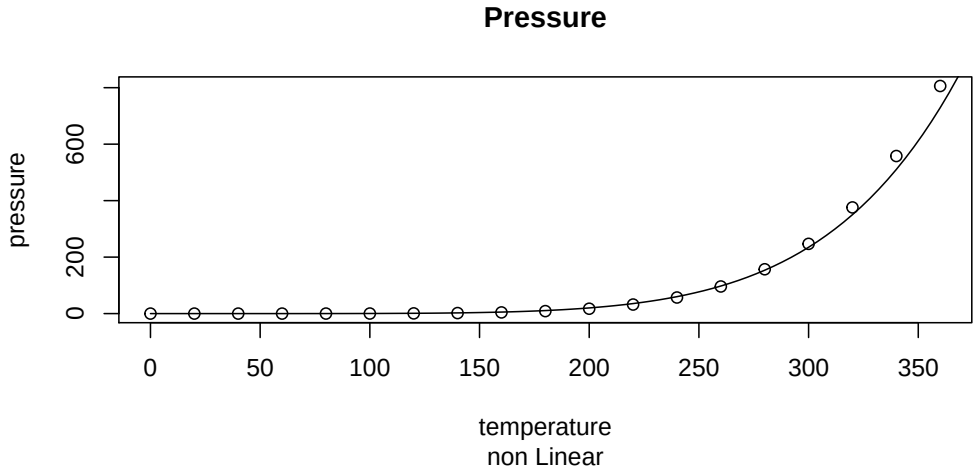


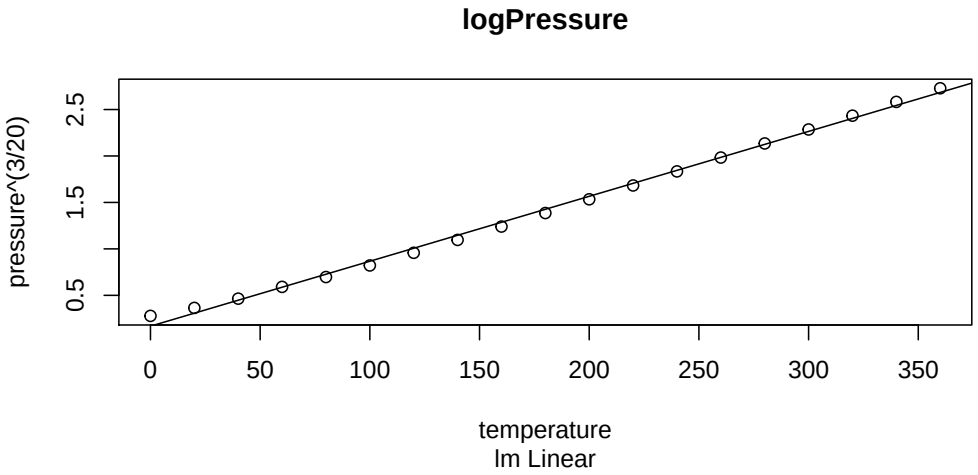
The plot is now linear.

7.3 d)

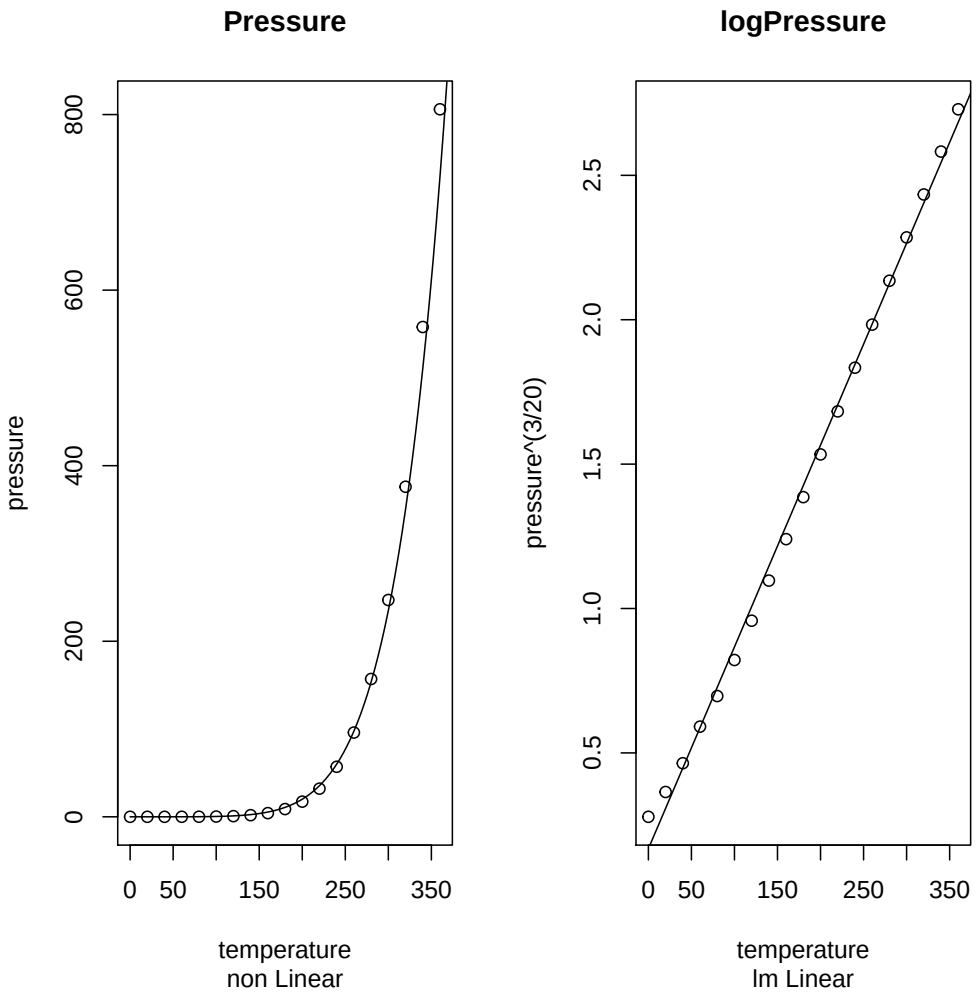
```
par(mfrow=c(2,1))

plot(pressure~temperature,data=pressure)
```

```
curve((0.168 + 0.007*x)^(20/3), from=0, to=400, add=
   TRUE)

title(main="Pressure", sub= "non Linear")

plot(pressure^(3/20)~temperature,data=pressure)

abline(lm(pressure^(3/20)~temperature,data=pressure))

title(main="logPressure", sub= "lm Linear")
```





```
par(mfrow=c(1,2))

plot(pressure~temperature,data=pressure)

curve((0.168 + 0.007*x)^(20/3), from=0, to=400, add=
   TRUE)

title(main="Pressure", sub= "non Linear")

plot(pressure^(3/20)~temperature,data=pressure)

abline(lm(pressure^(3/20)~temperature,data=pressure))

title(main="logPressure", sub= "lm Linear")
```


Pressure
pressure
0
200
400
600
800
0
50
150
250
350
temperature
non Linear
logPressure
pressure^(3/20)
0.5
1.0
1.5
2.0
2.5
0
50
150
250
350
temperature
lm Linear