Lab 4 exercises

23847

2024-03-31

Gen AI has not been used within the creation of this document

# Problem 1

## a)

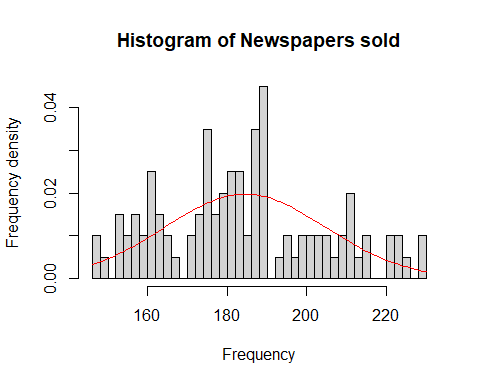
First we calculate the first moment using the formula

then we calculate the second moment using the formula

where x\_i represent the elements of the npsold vector. We can then rearrange to get

, denoted as **sdev**. The data does not appear to follow the normal distribution however, this could be expected as the standard deviation is quite large.

# npsold not included to keep the code clean  
mu = sum(npsold)/length(npsold)  
m2 = sum(npsold^2)/length(npsold)  
sigma = sqrt(m2 - mu^2)  
hist(npsold, breaks = 40, xlab = "Frequency", ylab = "Frequency density", main = "Histogram of Newspapers sold", prob = TRUE)  
curve(dnorm(x, mu, sigma), add = TRUE, col = "red")

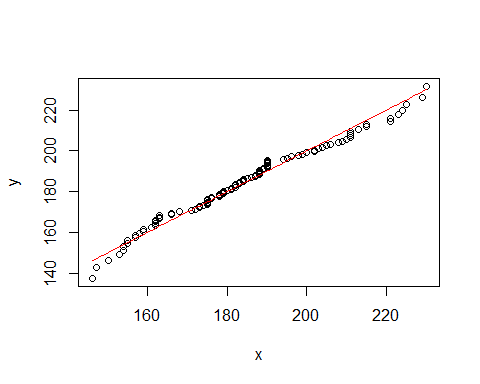


## b)

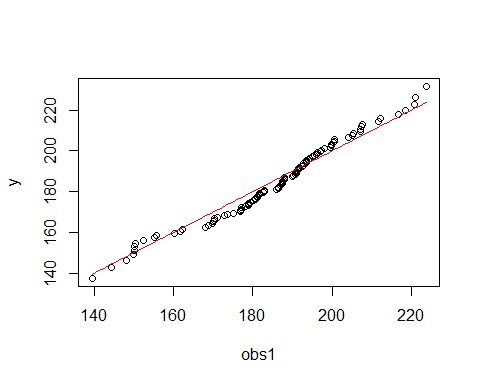
When making a Q-Q plot you must sort the observed values and compare these to

. When the data is a good fit it will be observed to be close to the line y = x. Here we can see the data very weakly follows a normal distribution, if at all as the tails of the data trail off.

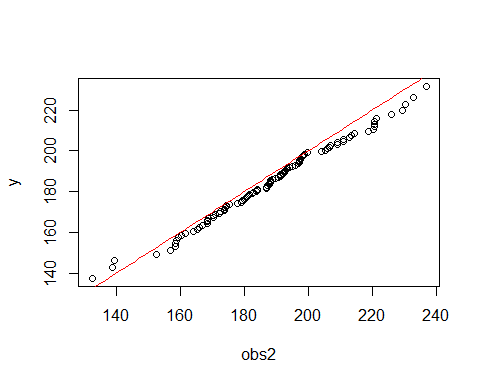
x= sort(npsold)  
i = c(1:100)  
y = qnorm(i/101, mu, sigma)  
plot(x,y)  
curve(x^1, add = TRUE, col = "red")



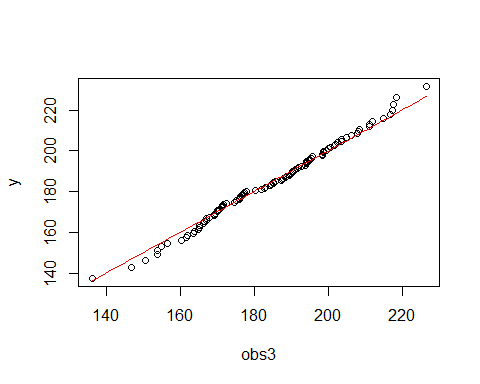
obs1 = sort(rnorm(100, mu, sigma))  
plot(obs1, y)  
curve(x^1, add = TRUE, col = "red")



obs2 = sort(rnorm(100, mu, sigma))  
plot(obs2, y)  
curve(x^1, add = TRUE, col = "red")



obs3 = sort(rnorm(100, mu, sigma))  
plot(obs3, y)  
curve(x^1, add = TRUE, col = "red")



## c)

### i.

For the case n>200 we may simplify the return, expanding from 200 \* 0.3 + (n-200) \* 0.1 to 40 + n \* 0.1.

### ii.

If Erik buys less papers than are in demand (d > n) then he will sell every paper. Otherwise ( d <= n ) he will sell d papers. In both cases we multiply n or d by 100 as the sell price is £1, and subtract the cost of buying n papers, since this is independent of how many he sells.

### iii.

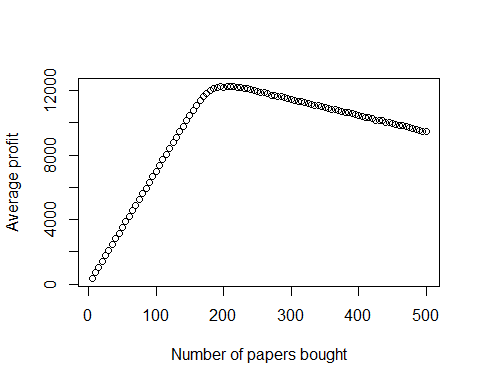
For the average profit we will calculate the mean value by creating a vector **d** of random normal values, then we use the sapply function to apply the function **profit(n,d)** to all of the values within **d**, and then we may calculate the mean of the output

# i  
cost = function(n){  
 if((0 <= n) && (n <= 200)){  
 return(n \* 30)  
 }   
 else if(n > 200){  
 return(4000 + n \* 10)  
 }  
}  
  
  
# ii  
profit = function(n, d){  
 return (100\*(if(d > n) n else d) - cost(n))  
}  
  
# iii  
averageProfit = function(n, mu, sigma, nreps){  
 d = rnorm(nreps, mu, sigma)  
 prof = sapply(d, function(d) profit(n, d))  
 return (mean(prof))  
}  
  
# Tests  
test\_1 = cost(100) # Expected output of 0.3 \* 100 = 30  
test\_2 = profit(100, 50) # Expected output of 50 - 30 = 20

## d)

To find the optimal number of papers we may use mu and sigma from **a)**, then entering , to see which value has the highest average profit. Firstly it was run with nreps = 10^3, which was then increased to 10^4 for a greater accuracy. Upon plotting the vector *avg* we can see the maximum of the graph is around n = 200 or n = 210

n = 100  
avg = numeric(n)  
for(i in 1:n){  
 avg[i] = averageProfit(5\*i, mu, sigma, 10^4)  
}  
plot(seq(from = 5, to = 5\*n, by = 5), avg, xlab = "Number of papers bought", ylab = "Average profit")



## e)

We would need to know the time it takes Erik to sell his papers, whether it is linear, so each paper takes the same amount of time, or if the time follows its own distribution. Once this information is obtained we could set up a relation between profits and sell time, a relation we would then be able to minimize the time for a given profit. Under the assumption the time taken is linear ( although highly unlikely in a real world scenario ) then we can observe the plot from **d)** and suggest he buys n = 200 or n = 210 papers as these two give the best profits for the number of papers. There is a steady decrease on either side of the function so it is not advised he goes any lower than 170 papers.