Hw4

May 9, 2024

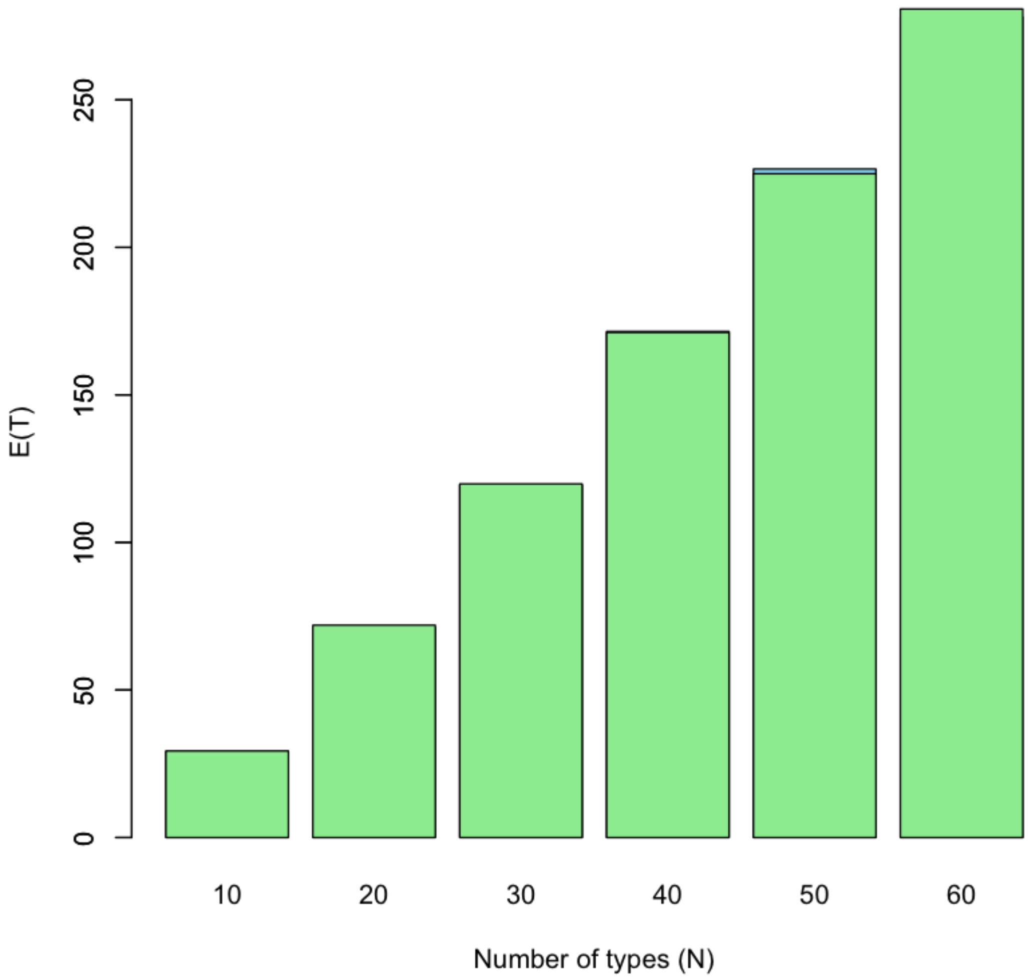
1

[52]:
```
NSim=1500                                       # Number of simulations
# function rep(0,NSim) replicates the value 0, NSim times
x <- c(10, 20, 30, 40, 50, 60)
E = numeric(6) # vector that hold the final E(T) for the plot
Theoretical_value = numeric(6)
a = 1

for(N in x){
    num=rep(0,NSim)                                   # This is  a vector␣
 ↪initialized to 0;
for (i in 1:NSim){
  trials <-rep(0,0)                     # for a simulation intialize trials to␣
 ↪empty
  while (length(unique(as.vector(trials)))<N){     # until all coupons collected
    trials<-cbind(sample(1:N,1),trials)  # withdraw a coupon and add to trials␣
 ↪using cbind function
    num[i]=num[i]+1                      # increment trials
  }
}
    E[a] = mean(num)
    Theoretical_value[a] = N*log(N) + 0.5771*N +0.5
    a = a+1

}
print ("Theoretical_value:")
print(Theoretical_value)
print ("Simulated value:")
print(E)
barplot(E, names.arg = x, ylab="E(T)", xlab="Number of types (N)", col =␣
 ↪"skyblue")
barplot(Theoretical_value, add= TRUE, col = "lightgreen")
```

```
[1] "Theoretical_value:"
[1]  29.29685  71.95665 119.84892 171.13918 224.95615 280.78667
[1] "Simulated value:"
[1]  29.23733  71.19267 119.39467 171.51533 226.56467 278.03067
```



Observation: The over all accuracy is good and close to the theoretical value. The accuracy will be better if doing more simulation.

2

2.1

Since $\lambda = 1$,

$$P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Thus, the probability that there are at least 2 mutations on the page:

$$\begin{aligned} P(X \geq 2) &= 1 - (P(X=1) + P(X=0)) \\ &= 1 - (\frac{e^{-1}1^1}{1!} + \frac{e^{-1}1^0}{1!}) \\ &= 0.2642 \end{aligned}$$

2.2

Since there are no more than 2 mutations over all three pages, which mean:

$$0 \leq k \leq 2$$

$$P(X = k) = \frac{e^{-\lambda \times 3}(\lambda \times 3)^k}{k!}$$

Thus:

$$\begin{aligned} P(0 \leq X \leq 2) &= P(X=2) + P(X=1) + P(X=0) \\ &= \frac{e^{-1\times 3}(1 \times 3)^2}{2!} + \frac{e^{-1\times 3}(1 \times 3)^1}{1!} + \frac{e^{-1\times 3}(1 \times 3^0)}{0!} \\ &= 0.4232 \end{aligned}$$

2.3

$$\begin{aligned} P\{X \geq 2 | X \geq 1\} &= \frac{P\{X \geq 2 \cap X \geq 1\}}{P\{X \geq 1\}} \\ &= \frac{P\{X \geq 2\}}{P\{X \geq 1\}} \\ &= \frac{0.2642}{1 - (P\{X = 0\})} \\ &= 0.41796 \end{aligned}$$

3

$$\begin{aligned} \sum_{k=0}^{\infty} p_k &= \frac{e^{-\lambda}\lambda^0}{0!} + \frac{e^{-\lambda}\lambda^1}{1!} + \frac{e^{-\lambda}\lambda^2}{2!} + ... \\ &= e^{-\lambda} \times (\frac{\lambda^0}{0!} + \frac{\lambda^1}{1!} + \frac{\lambda^2}{2!} + ...) \\ &= e^{-\lambda} \times e^x \end{aligned}$$

4

4.1

The total $\lambda = \lambda\text{A} + \lambda\text{B} = 3$ trains/hr + 6 trains/hr = 9 trains/hr. Thus, the probability that exactly 9 trains arrive at the station in any given hour:

$$\begin{aligned} P(X=9) &= \frac{e^{-9}9^9}{9!} \\ &= 0.1318 \end{aligned}$$

4.2

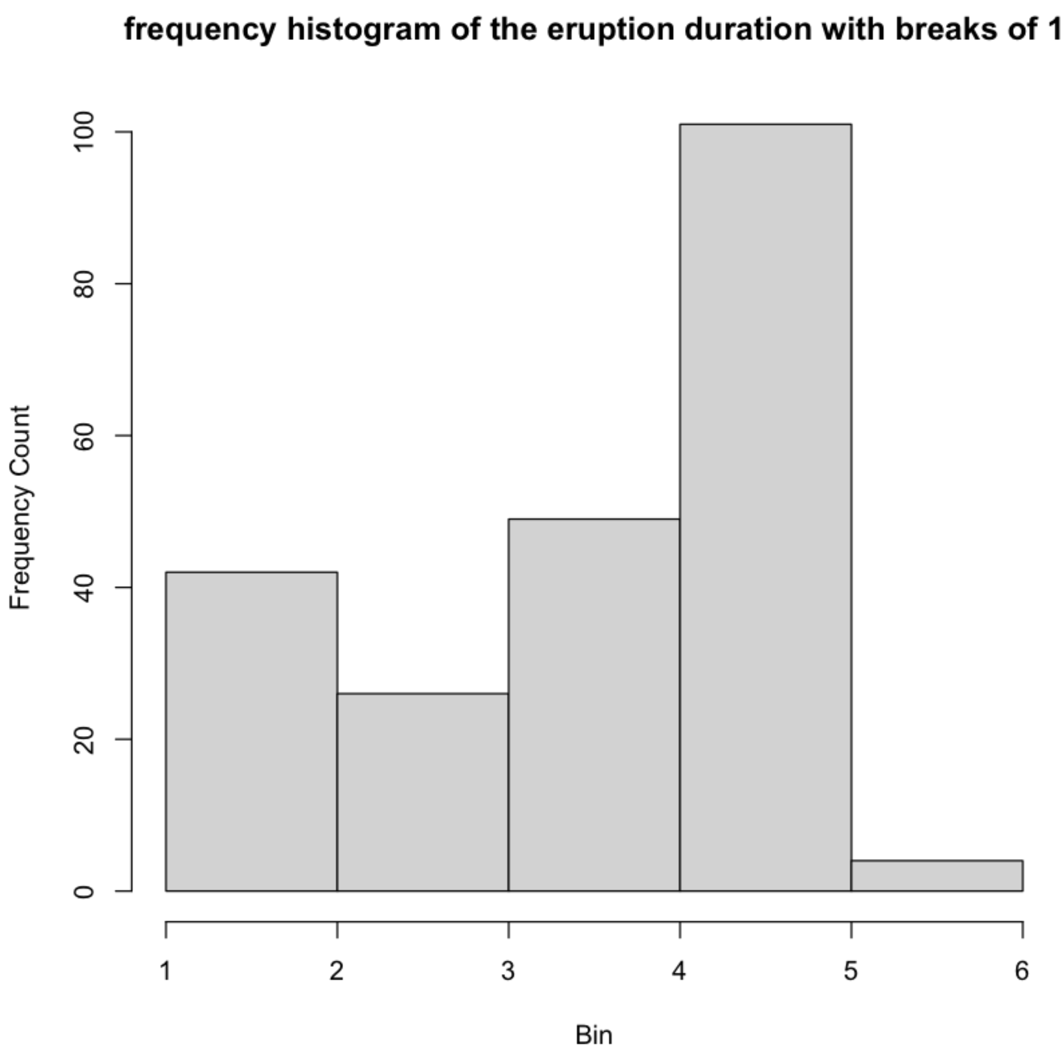
So, in the problem it is asking for the *kth* hour that is the first success, Which is belong to **geometric distribution**. Thus, the expected number of hours they need to wait = the expected number of trials before the first success.

$$\begin{aligned} E(X=9) &= \frac{1}{P(X=9)} \\ &= \frac{1}{0.1318} \\ &= 7.59\text{hours} \end{aligned}$$

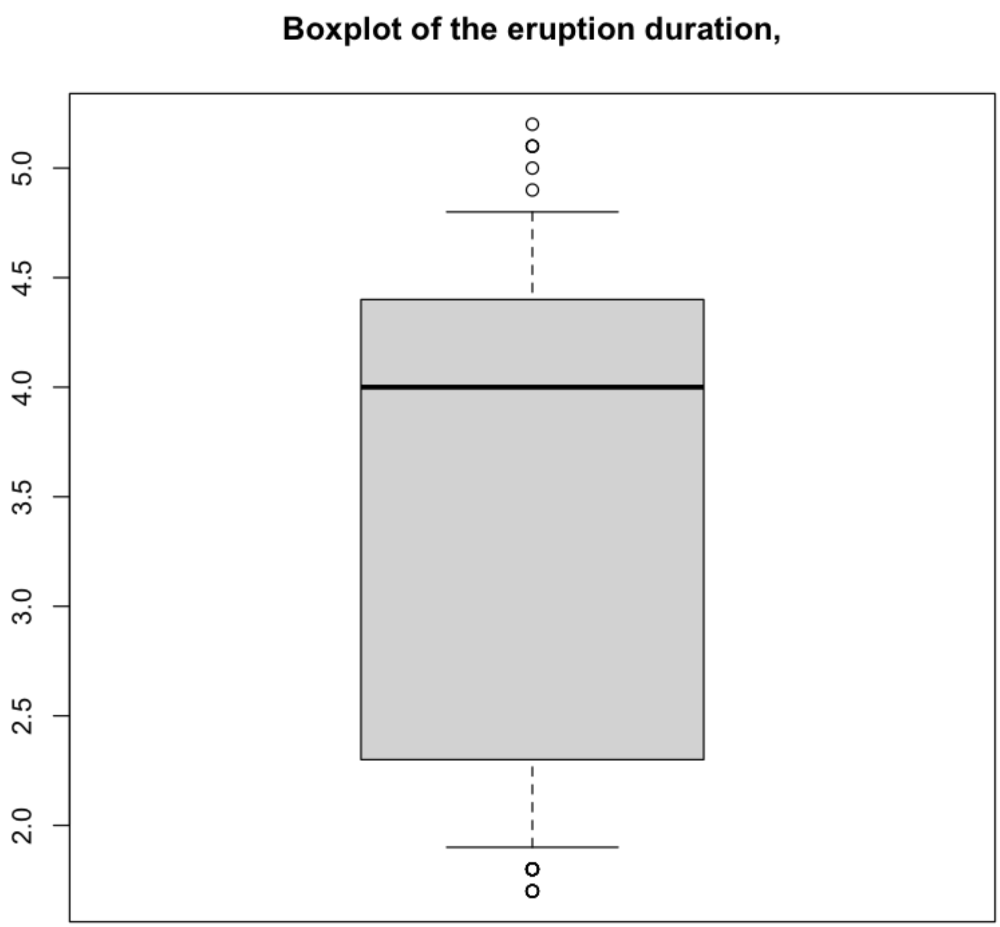
5

[78]:
```
data <- read.table("./Old_Faithful.txt", header=TRUE)
low = min(data[,3])
high = max(data[,3])

hist(data[,3], breaks = seq(1,6,1), xlab = "Bin", ylab = "Frequency Count",␣
↪main = "frequency histogram of the eruption duration with breaks of 1")
```



[79]:
```r
# Boxplot
boxplot(data[,3], range=0.2, main = "Boxplot of the eruption duration,")
```



[82]:
```
# Specified quantiles
q = c(.95,.97,.99)
quantile(data[,3], q)
```

95\% 4.8 **97\%** 4.8 **99\%** 5.1

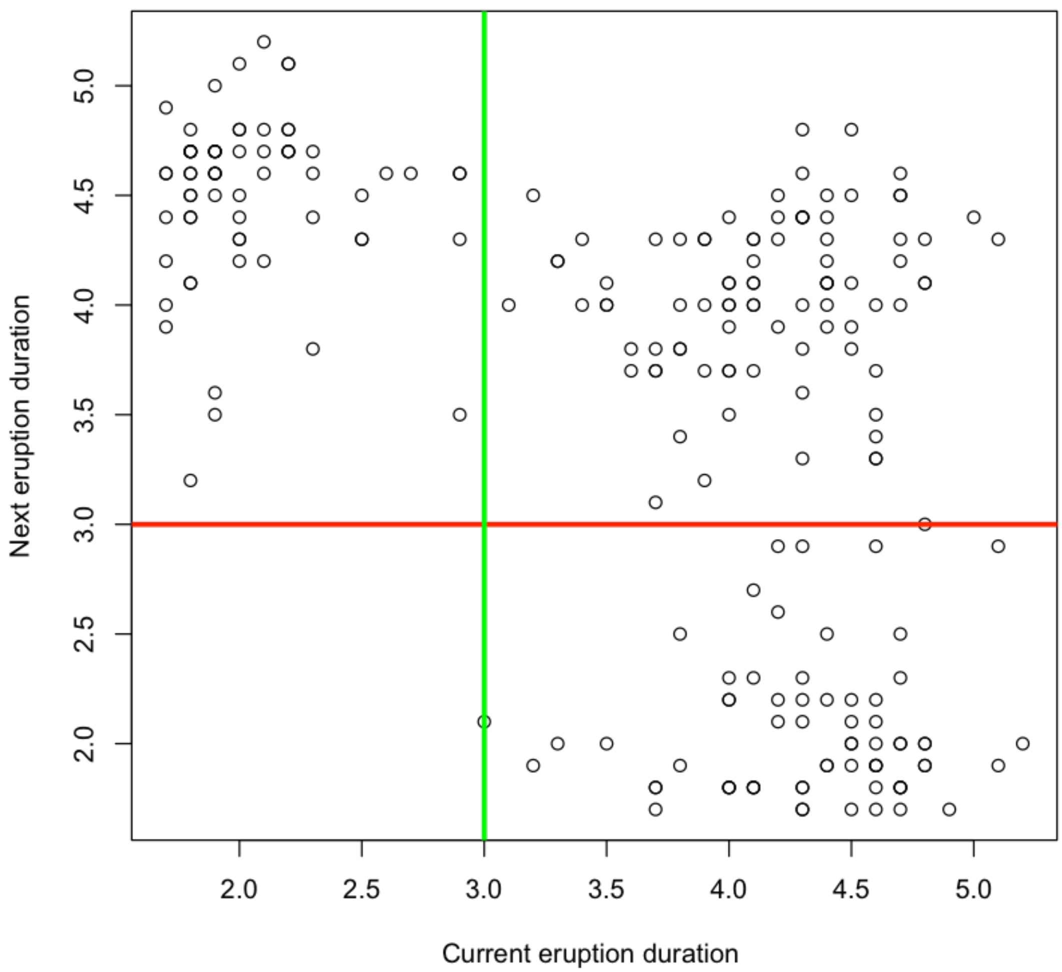
[96]:
```
data <- read.table("./Old_Faithful.txt", header=TRUE)
N = length(data[,3])

e = data[,3]

x = numeric(N/2)
y = numeric(N/2)
```

```
for (i in 1:N-1){
    x[i] = e[i]
    y[i] = e[i+1]
}

plot(x, y, ylab="Next eruption duration", xlab="Current eruption duration ")
abline(h = 3 , col = "red", lwd = 3)
abline(v = 3 , col = "green", lwd = 3)
```



[113]:
```
LandL = 0
LandS = 0
SandL = 0
SandS = 0
```

```r
for(i in 1:(N-1)){
    if((e[i] > 3) && (e[i+1] > 3)){
        LandL = LandL + 1
    }
    else if((e[i] > 3) && (e[i+1] <= 3)){
        LandS = LandS + 1
        }
    else if((e[i] <= 3) && (e[i+1] > 3)){
        SandL = SandL + 1
        }
    else{
        SandS = SandS + 1
        }
}

P_LL = LandL / N
P_LS = LandS / N
P_SL = SandL / N
P_SS = SandS / N
print("a long eruption is followed by a long eruption:")
print(P_LL)
print("a long eruption is followed by a short eruption:")
print(P_LS)
print("a short eruption is followed by a long eruption:")
print(P_SL)
print("a short eruption is followed by a short eruption:")
print(P_SS)
```

```
[1] "a long eruption is followed by a long eruption:"
[1] 0.3918919
[1] "a long eruption is followed by a short eruption:"
[1] 0.3018018
[1] "a short eruption is followed by a long eruption:"
[1] 0.2972973
[1] "a short eruption is followed by a short eruption:"
[1] 0.004504505
```

[ ]: