MATH 3070 Lab Fall Project 2 Solutions

Pranav Rajan

September 10, 2020

Contents

Remember: I expect to see commentary either in the text, in the code with comments created using `#`*, or (preferably) both!* ***Failing to do so may result in lost points!***

Problem 1 (Verzani problem 2.4)

Create the following sequences, using `:`*,* `seq()`*, or* `rep()` *as appropriate:*

1. `"a" "a" "a" "a" "a" "a"`

```
# Your solution here
rep("a", 6)
```

```
## [1] "a" "a" "a" "a" "a" "a"
```

2. `1 3 ... 99` *(the odd numbers)*

```
# Your solution here
seq(1, 100, by=2)
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
## [26] 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

3. `1 1 1 2 2 2 3 3 3`

```
# Your solution here
rep(c(1, 2, 3), times=c(3, 3, 3))
```

```
## [1] 1 1 1 2 2 2 3 3 3
```

Problem 2 (Verzani problem 2.3)

Let our small data set be `2 5 4 10 8`.

1. *Enter this data into a data vector* **`x`**.

```
# Your code here
x <- c(2, 5, 4, 10, 8)
```

2. *Find the square of each number.*

```
# Your code here
x^2
```

```
## [1]   4  25  16 100  64
```

3. *Subtract 6 from each number.*

```
# Your code here
x - 6
```

```
## [1] -4 -1 -2  4  2
```

4. *Subtract 9 from each number and then square the answer.*

```
# Your code here
(x - 9)^2
```

```
## [1] 49 16 25  1  1
```

Problem 3 (Verzani problem 2.25)

Write a function ***`isprime()`*** *that checks if a number x is prime by dividing x by all the values in $2, ..., x-1$ then checking to see if there is a remainder of 0. The expression* **`a %% b`** *returns the remainder of* **`a`** *divided by* **`b`**.

```
# Your code here
isprime <- function(x) {
  y <- seq(2, x-1)
  for (value in y) {
    if (x %% value == 0)
      return("False")
  }
  return("True")
}
isprime(7)
```

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

BONUS: *You do not need to check all numbers from 2 to $x-1$ to see if a number is prime. What is the largest you would need to go for an arbitrary x? Create a new function,* ***`isprime2()`****, that implements this better (yet still slow) method.*

```
# Your code here
```

Problem 4 (Verzani problem 2.32)

Fit a density estimate to the data set `pi2000` *(**UsingR**). Compare with the appropriate histogram. Why might you want to add an argument like* `breaks = 0:10-.5` *to* `hist()`*? (Hint: read the documentation of* `hist()` *to see what setting this argument does.)*

```
# Your code here
require("UsingR")
```

```
## Loading required package: UsingR
```

```
## Loading required package: MASS
```

```
## Loading required package: HistData
```

```
## Loading required package: Hmisc
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

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

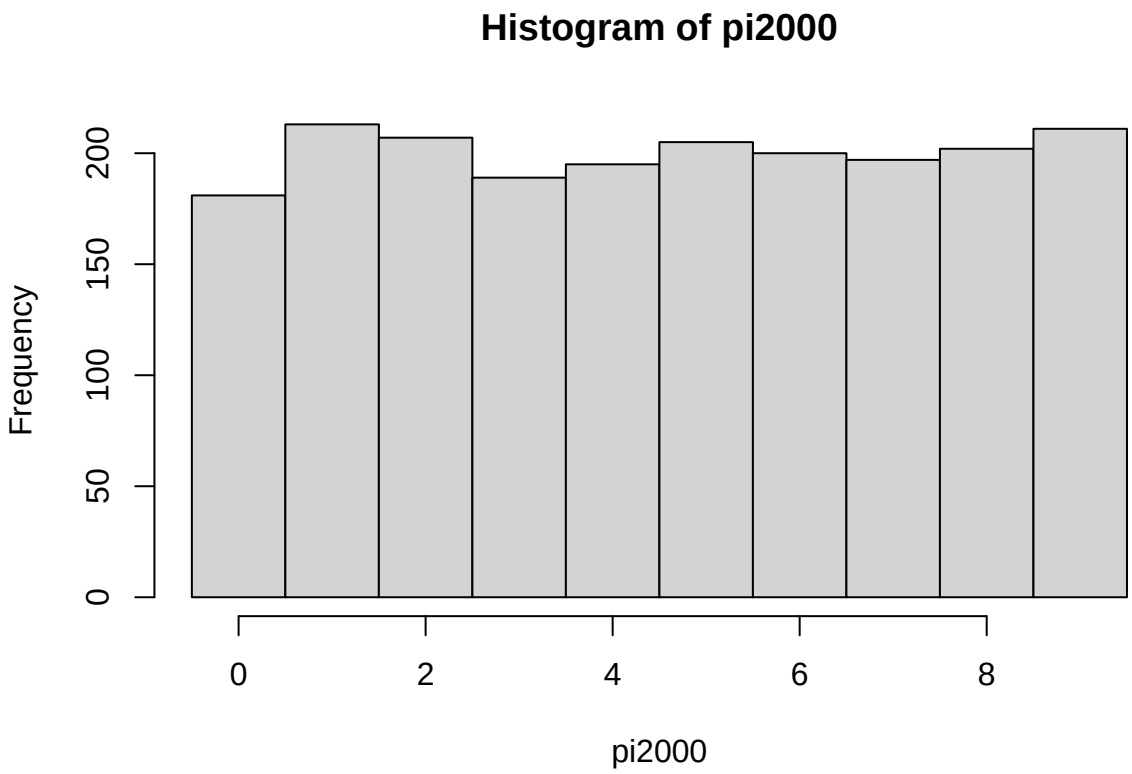
```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

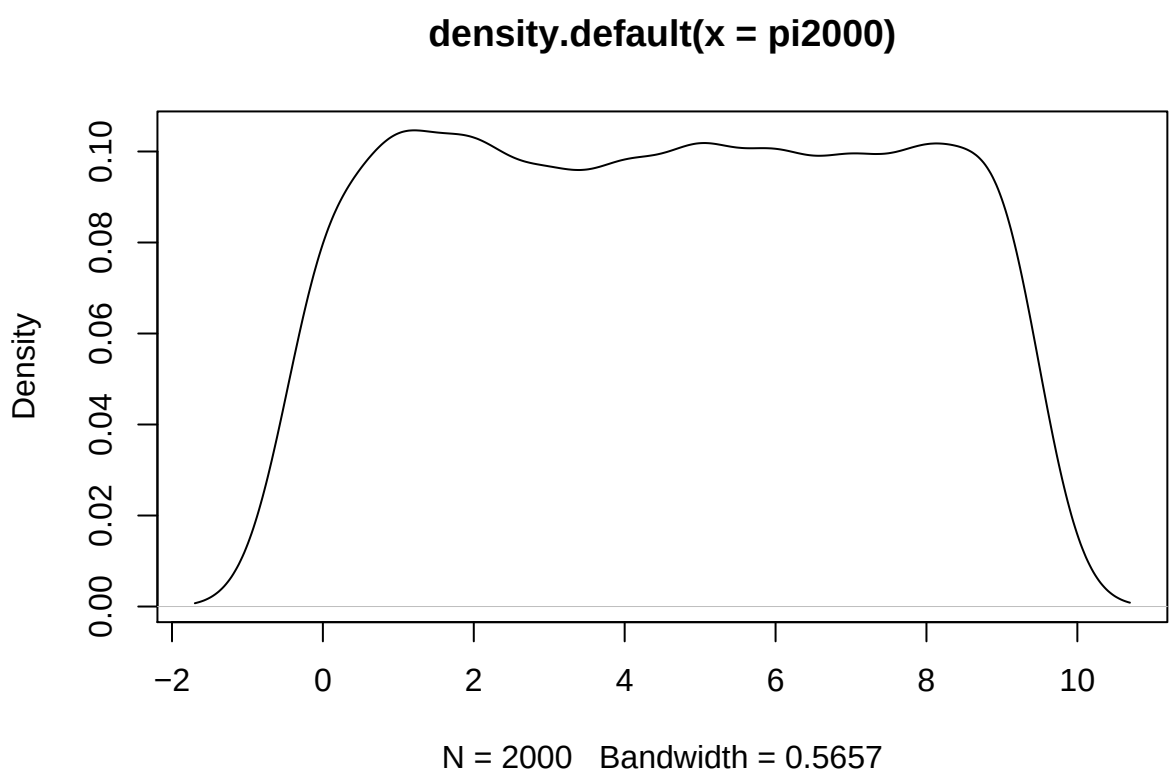
```
##
## Attaching package: 'UsingR'
```

```
## The following object is masked from 'package:survival':
##
##     cancer
```

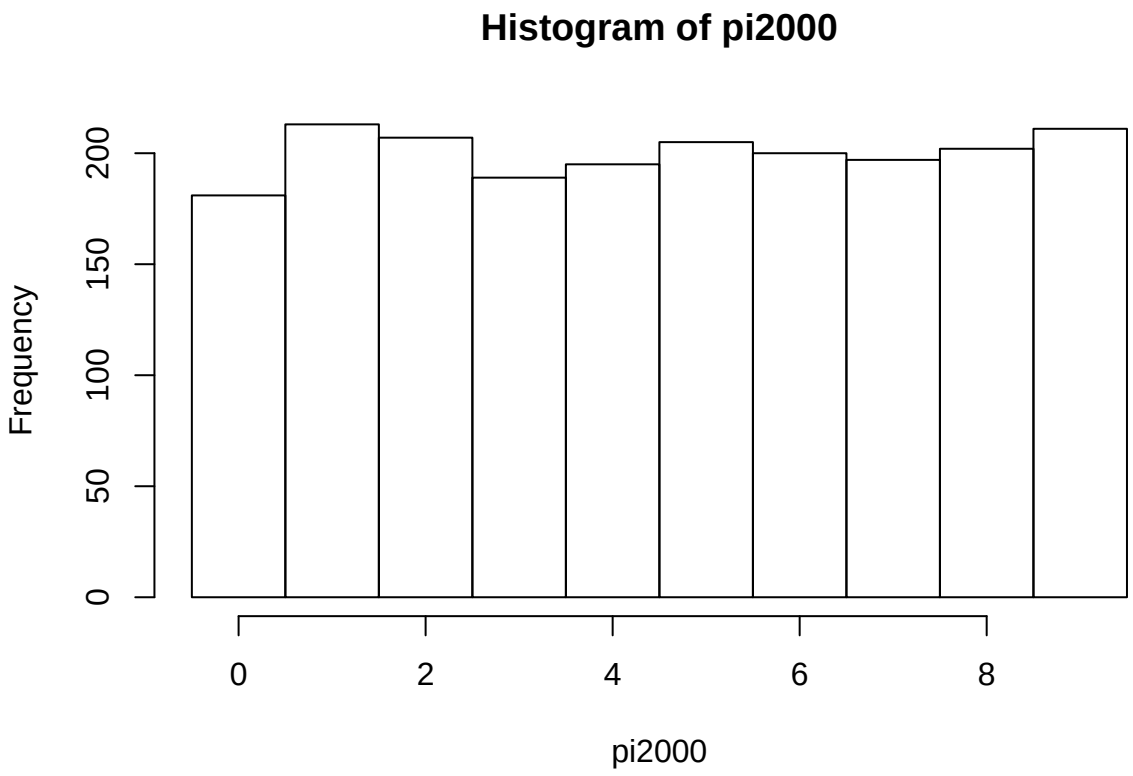
```
# density plot
pi2000density <- density(pi2000)
# histogram plot
# the breaks parameter defines the spacing between bars or the number of bars
pi2000hist <- hist(pi2000, breaks= 0:10 - 0.5)
```




```
plot(pi2000density)
```



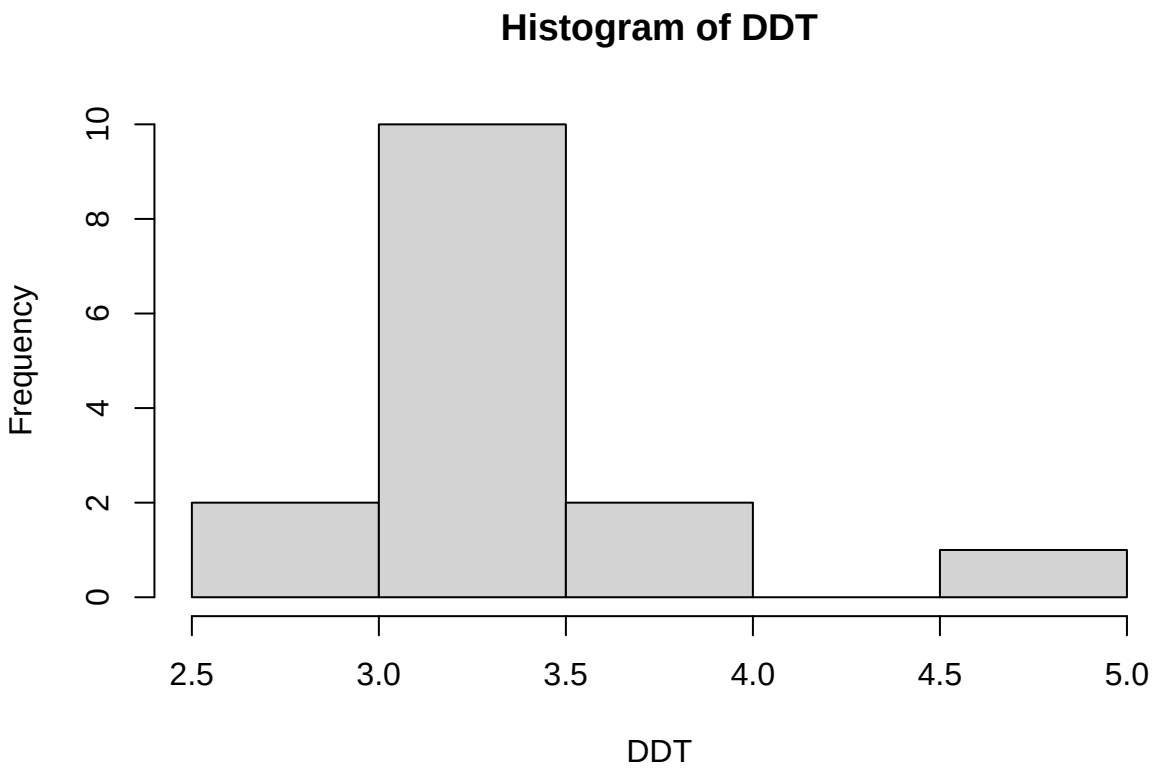

```
plot(pi2000hist)
```



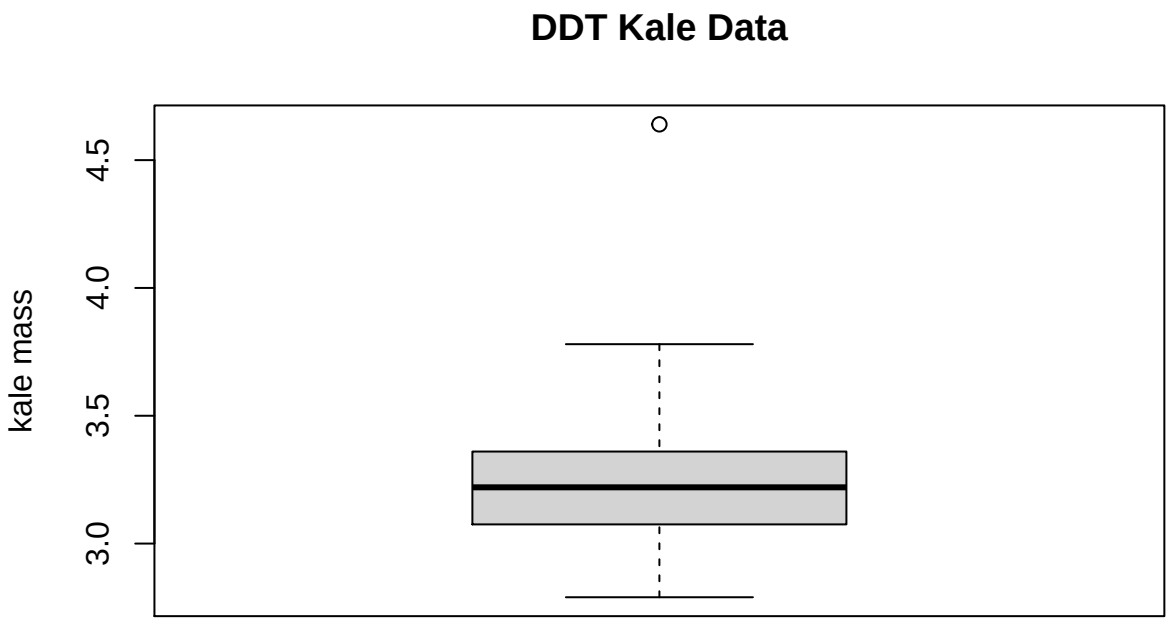
Problem 5 (Verzani problem 2.34)

The data set `DDT` *(**MASS**) contains independent measurements of the pesticide DDT on kale. Make a histogram and a boxplot of the data. From these, estimate the mean and standard deviation. Check your answers with the appropriate functions.*

```
# Your code here
require("UsingR")
histogramChart <- hist(DDT)
```




```
#plot(histogramChart)
boxplotChart <- boxplot(DDT, main="DDT Kale Data", ylab="kale mass")
```




```
# Estimates
# Mean: 3.25 based on the boxplot
# SD: roughly 0.5 based on sorting the data since the points seem really close
# print(sort(DDT))

# Actual data summary
mean(DDT)
```

```
## [1] 3.328
```

```
sd(DDT)
```

```
## [1] 0.4371531
```