HW1 Computing in Statistics

Elias Washor

2024-09-05

Question 1

a)

```
df <- data.frame('radius' = 3:20, 'volume'= rep(0, 18) )
Vols <- df$radius^3 * 4 * pi / 3
df['volume'] <- Vols
(df)
```

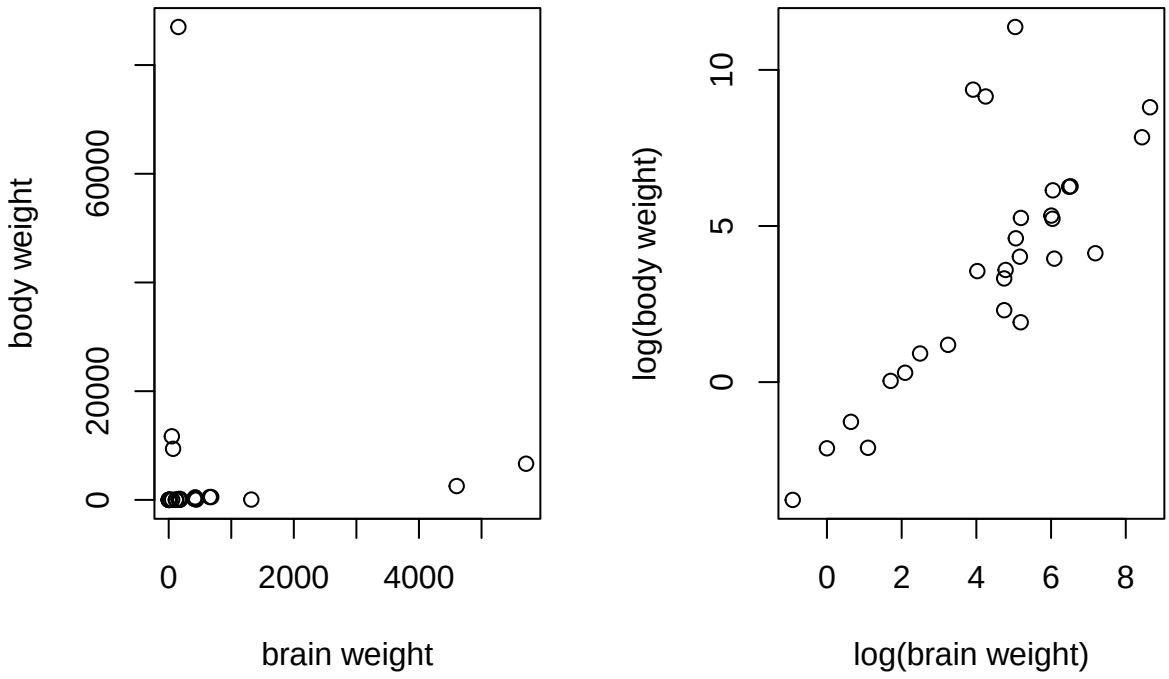
```
##    radius     volume
## 1       3   113.0973
## 2       4   268.0826
## 3       5   523.5988
## 4       6   904.7787
## 5       7  1436.7550
## 6       8  2144.6606
## 7       9  3053.6281
## 8      10  4188.7902
## 9      11  5575.2798
## 10     12  7238.2295
## 11     13  9202.7721
## 12     14 11494.0403
## 13     15 14137.1669
## 14     16 17157.2847
## 15     17 20579.5263
## 16     18 24429.0245
## 17     19 28730.9120
## 18     20 33510.3216
```

b)

```
library(MASS)
par(mfcol = c(1,2) )
attach(Animals)
plot(brain, body, xlab = 'brain weight', ylab = 'body weight')
identify(brain, body, labels = rownames(Animals))
```

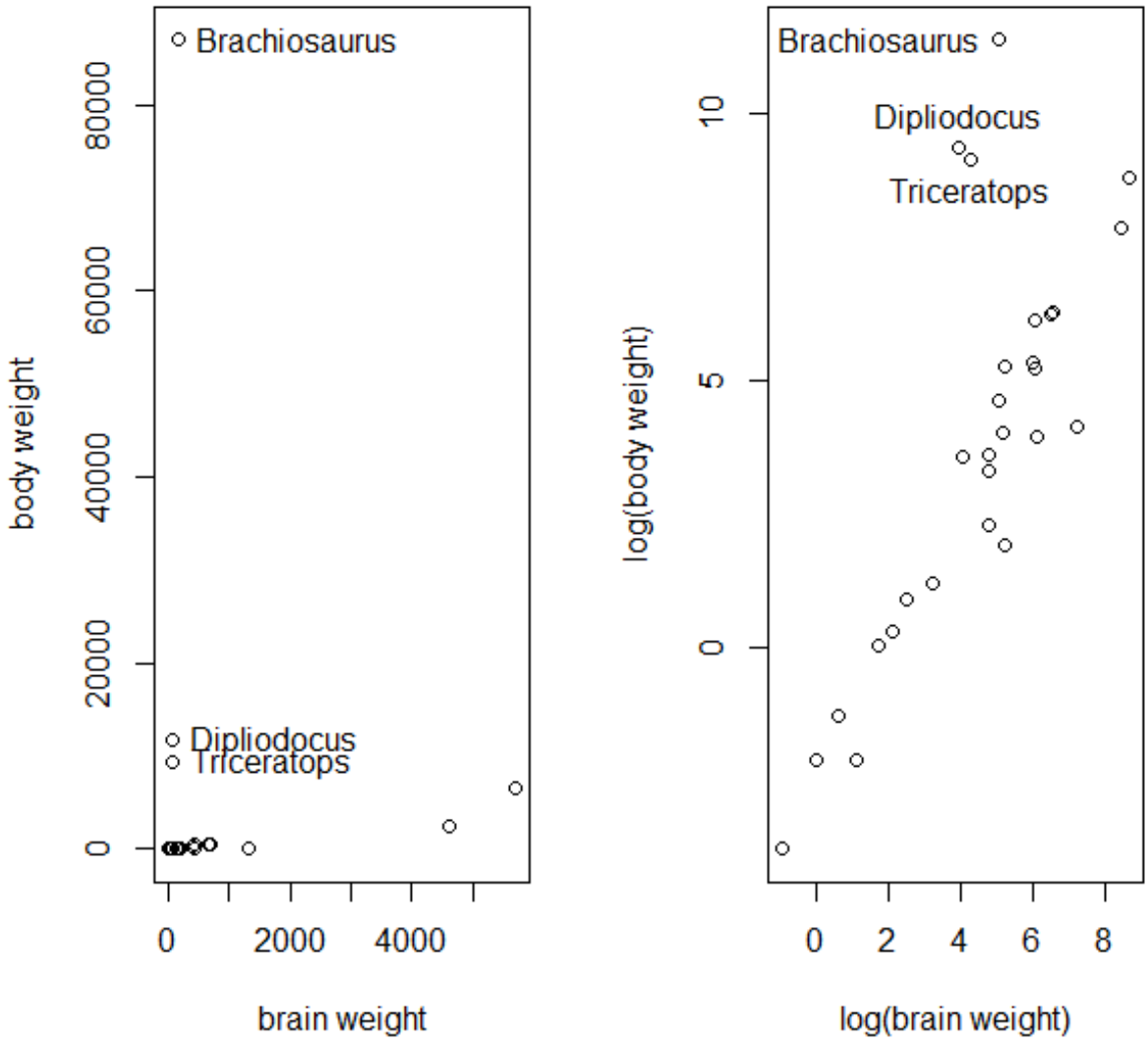
```
## integer(0)
```

```
plot(log(brain), log(body), xlab = 'log(brain weight)', ylab = 'log(body weight)')
identify(log(brain), log(body), labels = rownames(Animals) )
```




```
## integer(0)
```

```
detach(Animals)
```



**** Plot was labeled with identify() using R in windows cmd. RStudio won't label points with identify**

Question 3

```
### sapply() double cols. in data frame\n')
doubled <- function(x) x * 2
df <- data.frame(i = 5:8, ii = 19:22, iii= 42:45)
new_df <- sapply(df, doubled)
as.matrix(df)
```

```
##      i ii iii
```

```
## [1,] 5 19  42
## [2,] 6 20  43
## [3,] 7 21  44
## [4,] 8 22  45
```

```
(new_df)
```

```
##       i ii iii
## [1,] 10 38  84
## [2,] 12 40  86
## [3,] 14 42  88
## [4,] 16 44  90
```

```
### invisible(): cubed(5) does not print; cubed(5) * 10 works bc/ function still ### returns value
cubed <- function(x) invisible(x ** 3)
cubed(5) ## does not print
cubed(5) * 10 ### function still returns value
```

```
## [1] 1250
```

```
### unique() 8th row is a duplicate and removed
my_df <- data.frame(w = c(3,20:25,3) , x = c(3,1:6,3), y = c(3,10:15,3), z = c(3,1:6,3), a = c(3, 15:20
my_df2 <- unique(my_df)
as.matrix(my_df)
```

```
##       w x  y z  a
## [1,]  3 3  3 3  3
## [2,] 20 1 10 1 15
## [3,] 21 2 11 2 16
## [4,] 22 3 12 3 17
## [5,] 23 4 13 4 18
## [6,] 24 5 14 5 19
## [7,] 25 6 15 6 20
## [8,]  3 3  3 3  3
```

```
as.matrix(my_df2)
```

```
##    w x  y z  a
## 1  3 3  3 3  3
## 2 20 1 10 1 15
## 3 21 2 11 2 16
## 4 22 3 12 3 17
## 5 23 4 13 4 18
## 6 24 5 14 5 19
## 7 25 6 15 6 20
```

Question 4: Factorial Function

```
#install.packages("microbenchmark")
```

```
library(purrr)

## product function to pass through reduce
product <- function(x,y) x*y

fact <- function(x) {
  return (reduce(c(1:x), product))
}
fact(5)
```

```
## [1] 120
```

```
fact(4)
```

```
## [1] 24
```

```
fact(7)
```

```
## [1] 5040
```

```
#install.packages("microbebenchmark")
library(microbenchmark)

microbenchmark(
  fact(10),
  factorial(10)
)
```

```
## Unit: nanoseconds
##           expr   min    lq     mean median    uq    max neval cld
##       fact(10) 40002 40501 41665.98  40701 41102 104400   100  a
##  factorial(10)   301   401   613.06    501   601  12402   100   b
```

** my factorial function runs slower than the built-in factorial function. My median time was 43300 ns and built in function runs with a median time of 600 ns.

Question 5: Unique element, odd times in a sequence

```
set.seed(1)
n <- 5
a <- sample(n, n, replace=TRUE)
(myseq <- sample(c(a, a))[seq(2*n-1)])
```

```
## [1] 4 4 1 1 1 5 1 2 5
```

```r
### function
uniqueOdd <- function(s){
  N <- length(unique(s))
  mymatrix <- matrix(rep(0, N) ,nrow = N)
  rownames(mymatrix) <- unique(s)

  ## count how many times each number appears
  for (n in s) {
    m <- as.character(n)
    mymatrix[m,] = mymatrix[m,] + 1
  }
  ## look at counts, find which is odd
  counts <- as.numeric(mymatrix[,1])
  i <- which((counts %% 2 ) == 1)
  return (as.numeric(row.names(mymatrix)[i]))
}

cat("uniqueOdd(myseq): ", uniqueOdd(myseq))
```

```
## uniqueOdd(myseq):  2
```