Computer Assignment 1

781552

9/24/2021

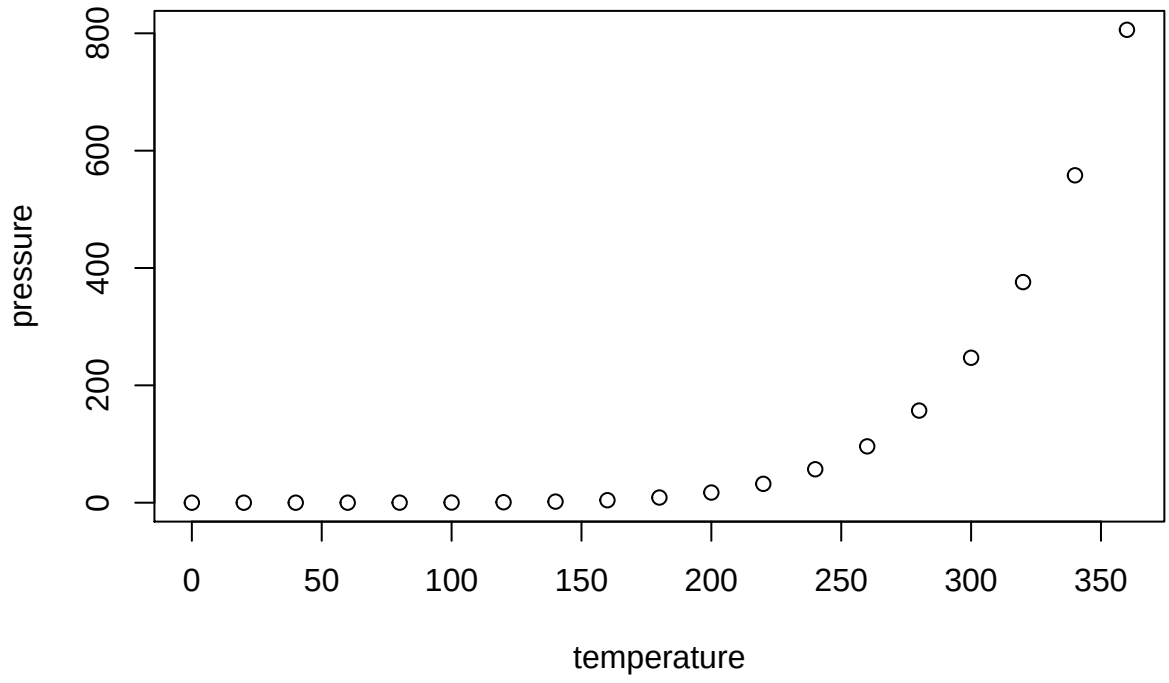
R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Question 1

Subtask (a)

```
seq <- read.GenBank("NC_001643", as.character = TRUE)[[1]]
seqStr <- paste(seq, sep ="", collapse = "")
DNAStr <- DNAString(seqStr)
```

Subtask (b)

```
freq <- alphabetFrequency(DNAStr)[DNA_BASES]
freq
```

```
##    A    C    G    T
## 5154 5099 2133 4168
```

```
comp <- complement(DNAStr)

freqComp <- alphabetFrequency(comp)[DNA_BASES]
freqComp
```

```
##    A    C    G    T
## 4168 2133 5099 5154
```

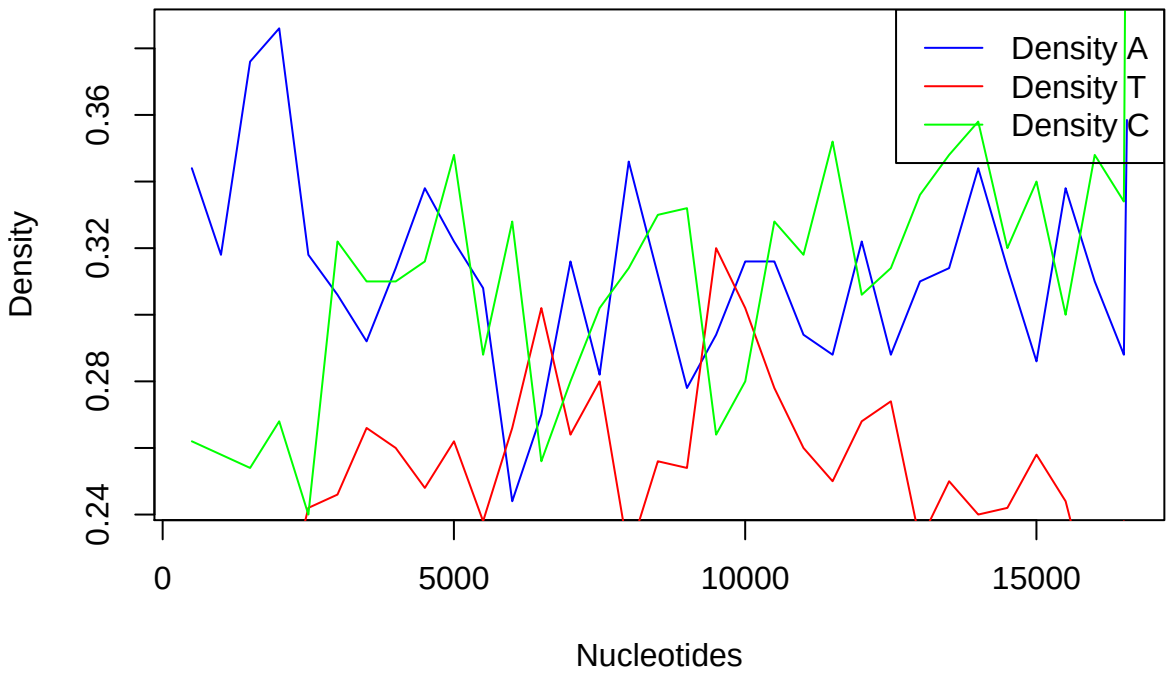
Subtask (c)

```
t <- seq(1, length(DNAStr), 500)
t
```

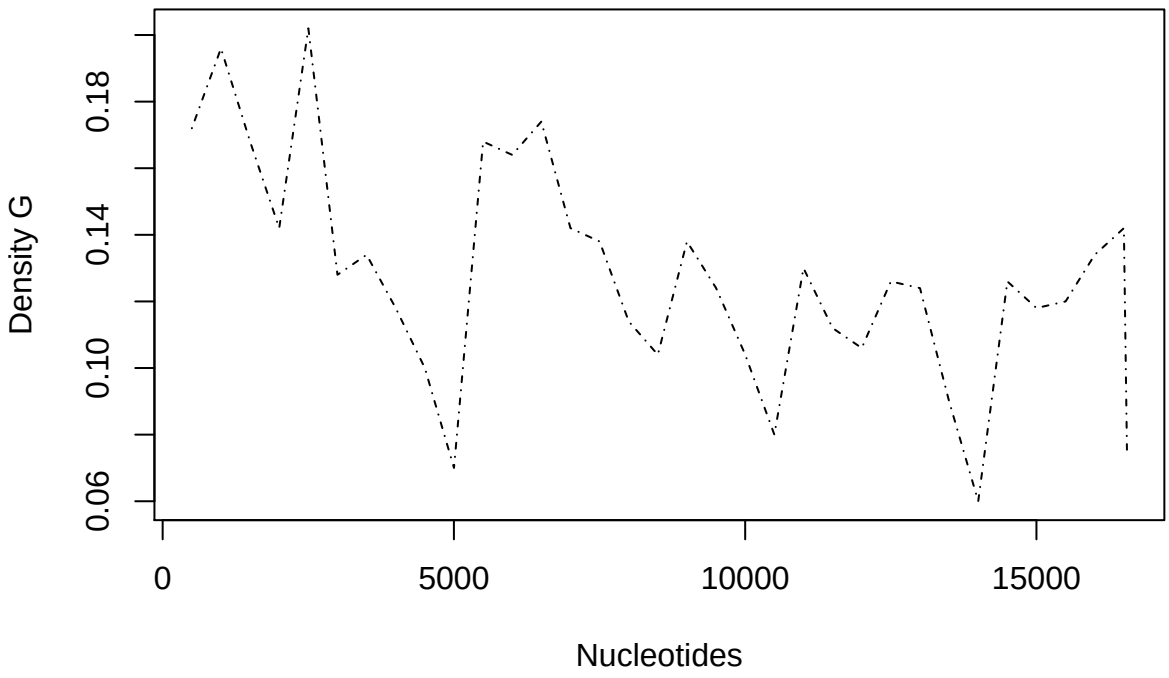
```
##  [1]     1   501  1001  1501  2001  2501  3001  3501  4001  4501  5001  5501
## [13]  6001  6501  7001  7501  8001  8501  9001  9501 10001 10501 11001 11501
## [25] 12001 12501 13001 13501 14001 14501 15001 15501 16001 16501
```

```
len <- length(t)

t[[len+1]] <-  16554

t1 <- t[2:35]
last <- seq(length(DNAStr) - 500, length(DNAStr))

densityA <- list()
densityT <- list()
densityC <- list()
densityG <- list()
densityAT <- list()
densityCG <- list()
```

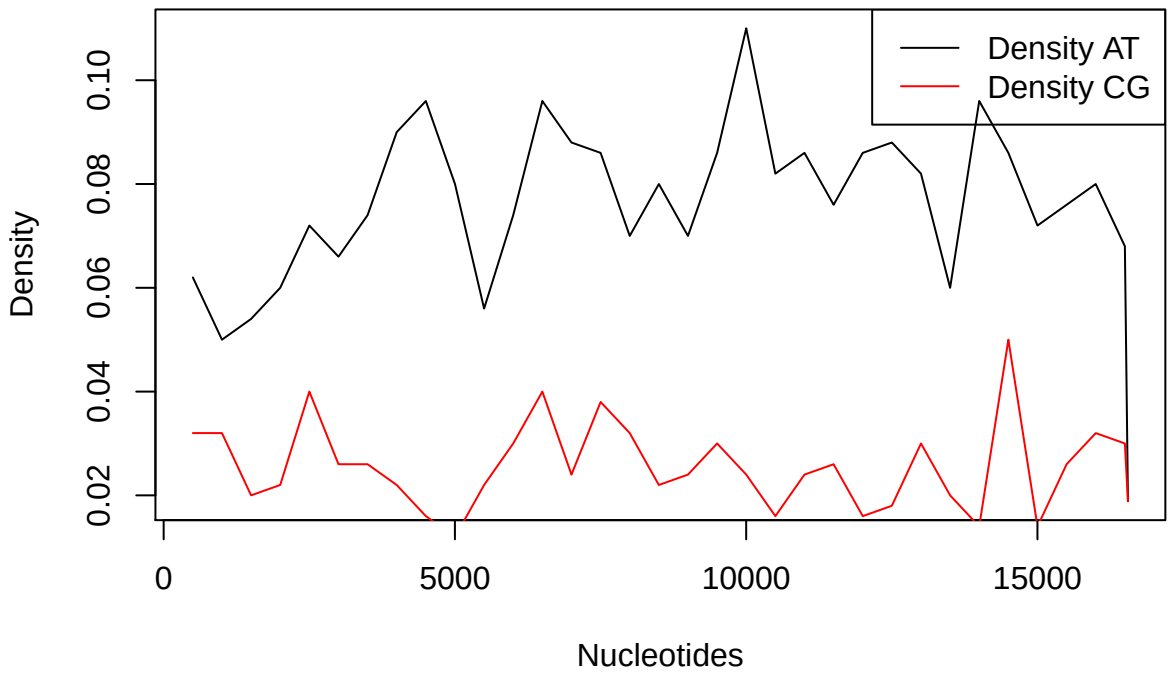
```
name <- c("1:500", "501:100", "1001:1500", "1501:2000")

for (i in 1:(length(t)-1)) {
  x <- t[i]
  y <- t[i+1]
  freqA <- alphabetFrequency(DNAStr[x:y])['A']
  freqT <- alphabetFrequency(DNAStr[x:y])['T']
  freqC <- alphabetFrequency(DNAStr[x:y])['C']
  freqG <- alphabetFrequency(DNAStr[x:y])['G']

  densityA <- c(densityA, (freqA / (y - x)) )
  densityT <- c(densityT, (freqT / (y - x)) )
  densityC <- c(densityC, (freqC / (y - x)) )
  densityG <- c(densityG, (freqG / (y - x)) )

  freqAT <- dinucleotideFrequency(DNAStr[x:y])["AT"]
  freqCG <- dinucleotideFrequency(DNAStr[x:y])["CG"]

  densityAT <- c(densityAT, (freqAT / (y - x)) )
  densityCG <- c(densityCG, (freqCG / (y - x)) )
}
plot(t1, densityA, type = "l", col = 'blue', lty = 1, xlab ='Nucleotides', ylab = 'Density')

lines(t1, densityT,  type = "l", col = 'red', lty = 1 )

lines(t1, densityC, type = "l", col = 'green', lty = 1 )

legend( x = "topright", legend=c('Density A', 'Density T', 'Density C'), col=c('blue', 'red', 'green'),
```




```
plot(t1, densityG, type = "l", col = 'black', lty = 4, xlab ='Nucleotides', ylab = 'Density G' )
```




```
plot(t1, densityAT, type = "l", col = 'black', lty = 1, xlab ='Nucleotides', ylab = 'Density' )

lines(t1, densityCG,  type = "l", col = 'red', lty = 1 )

legend( x = "topright", legend=c('Density AT', 'Density CG'), col=c('black', 'red'), lty=1:1)
```



Subtask (d)

```
dimers <-  dinucleotideFrequency(DNAStr, as.prob = TRUE, as.matrix = TRUE)
dimers
```

```
##            A          C          G          T
## A 0.09804869 0.08892648 0.04700054 0.07732737
## C 0.08916813 0.10620431 0.02573552 0.08693288
## G 0.03769709 0.04132181 0.02488975 0.02495016
## T 0.08644959 0.07158823 0.03117260 0.06258684
```

Subtask (e)

```
fourmers <- oligonucleotideFrequency(DNAStr, width = 4)

sort(fourmers, decreasing = TRUE)[1:10]
```

```
## CCTA CCCC CCCT ACCC AAAA AACC CTAC AACA CCTC ACTA
##  229  225  199  194  187  182  162  156  155  152
```

#Question 2 ## Subtask (a)

```
seq2 <- read.GenBank("L43967", as.character = TRUE)[[1]]
seqStr2 <- paste(seq, sep ="", collapse = "")
DNAStr2 <- DNAString(seqStr2)
```

Subtask (b)

```
dimers2 <-  dinucleotideFrequency(DNAStr2, as.prob = TRUE, as.matrix = TRUE)
dimers2
```

```
##            A          C          G          T
## A 0.09804869 0.08892648 0.04700054 0.07732737
## C 0.08916813 0.10620431 0.02573552 0.08693288
## G 0.03769709 0.04132181 0.02488975 0.02495016
## T 0.08644959 0.07158823 0.03117260 0.06258684
```

Subtask (c)

```
a <- dimers2[1,]
c <- dimers2[2,]
g <- dimers2[3,]
t <- dimers2[4,]


pi <- c(sum(a), sum(c), sum(g), sum(t) )
pi
```

```
## [1] 0.3113031 0.3080408 0.1288588 0.2517973
```

```
sum(pi)
```

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

Subtask (d)

```
dimers2[1,] <-  dimers2[1,] / sum(a)
dimers2[2,] <-  dimers2[2,] / sum(c)
dimers2[3,] <-  dimers2[3,] / sum(g)
dimers2[4,] <-  dimers2[4,] / sum(t)
dimers2
```

```
##           A         C          G         T
## A 0.3149622 0.2856588 0.15098001 0.2483990
## C 0.2894685 0.3447735 0.08354579 0.2822122
## G 0.2925457 0.3206751 0.19315518 0.1936240
## T 0.3433301 0.2843090 0.12380038 0.2485605
```

```r
sum(dimers2[1,])
```

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

Subtask (e)

```r
markovChainSim <- function(x, n, pi, P){
  seq <- vector(mode = "list", length = n)
  a <- sample(x, size = 1, replace = TRUE, prob = pi )
  seq[i]


for (i in 1:length(seq)) {
  b <- seq[i]


  if (b == "A"){
    c <- sample(x, size = 1, replace = TRUE, prob = P[1,])
  } else if(b == "C") {
     c <- sample(x, size = 1, replace = TRUE, prob = P[2,])
  } else if(b == "G") {
    c <- sample(x, size = 1, replace = TRUE, prob = P[3,])
  } else {
    c <- sample(x, size = 1, replace = TRUE, prob = P[4,] )
  }
  seq[i] <- c
}

  return(seq)

}
```

Subtask(f)

```r
testSim <-  markovChainSim(x = c("A", "C", "G", "T" ), n = 50000, pi = pi, P = dimers2)
testSimseq <- paste(testSim, sep ="", collapse = "")
testSimDNAString <- DNAString(testSimseq)

dimersTest <-  dinucleotideFrequency(testSimDNAString, as.prob = TRUE, as.matrix = TRUE)
dimersTest
```

```
##            A          C          G          T
## A 0.11800236 0.09938199 0.04162083 0.08618172
## C 0.09802196 0.08086162 0.03638073 0.06954139
## G 0.04480090 0.03434069 0.01530031 0.02956059
## T 0.08436169 0.07020140 0.03072061 0.06072121
```

```
dimers
```

```
##            A          C          G          T
## A 0.09804869 0.08892648 0.04700054 0.07732737
## C 0.08916813 0.10620431 0.02573552 0.08693288
## G 0.03769709 0.04132181 0.02488975 0.02495016
## T 0.08644959 0.07158823 0.03117260 0.06258684
```

```
pi
```

```
## [1] 0.3113031 0.3080408 0.1288588 0.2517973
```

```
a <- dimersTest[1,]
c <- dimersTest[2,]
g <- dimersTest[3,]
t <- dimersTest[4,]

piTest <- c(sum(a), sum(c), sum(g), sum(t))
piTest
```

```
## [1] 0.3451869 0.2848057 0.1240025 0.2460049
```

#The simulated frequencies vary only a little bit from the original frequencies, so the function's performance is quite decent.