Sequence_Aglinment

Truc Huynh

3/12/2021

Objectives

- Query/read/Analyze DNA sequence data.
- Create and use score matrices.
- Score and analyze sequence alignments

Question 1:

Retrieve the 2 sequences “AY884001” and “MH940245” from “genbank”.

```
# Choose the ACNUC
choosebank("genbank")

# Retrieve sequence AY884001
My_Que1 <- query("My_Que1", "AC=AY884001")

# write  sequence AY884001 to fasta file
write.fasta(getSequence(My_Que1[['req']][[1]]),getName(My_Que1[['req']][[1]])
,"AY88.fasta")

Seq1 <- (getSequence(read.fasta("AY88.fasta")))

# Retrieve sequence "MH940245"
My_Que2 <- query("My_Que2", "AC=MH940245")

# write sequence MH940245 to fasta
write.fasta(getSequence(My_Que2[['req']][[1]]),
getName(My_Que2[['req']][[1]]),"MH94.fasta")

Seq2 <- getSequence(read.fasta("MH94.fasta"))

closebank()
```

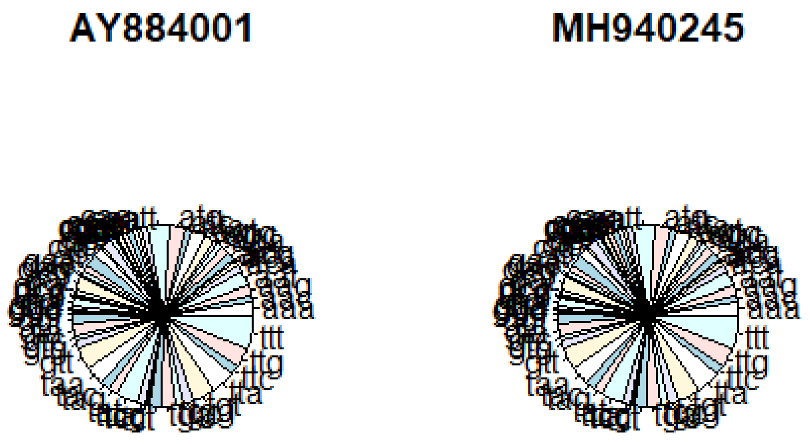
Question 2:

for each sequence, compute the frequency of each amino acid and plot them as a pie chart. Generate only one figure for both sequences.

```
# sequences AY884001
t1 = count(getSequence(Seq1[[1]]), wordsize = 3)
print("frequency of each amino acid in sequences AY884001")
```

```
## [1] "frequency of each amino acid in sequences AY884001"

t1

##
##   aaa   aac   aag   aat   aca   acc   acg   act   aga   agc   agg   agt   ata   atc   atg 
att
##   747   311   503   919   354   199   100   491   478   230   260   553   738   320   891 
1166
##   caa   cac   cag   cat   cca   ccc   ccg   cct   cga   cgc   cgg   cgt   cta   ctc   ctg 
ctt
##   364   170   264   346   172    75    47   277    57    54    49   162   543   209   431 
626
##   gaa   gac   gag   gat   gca   gcc   gcg   gct   gga   ggc   ggg   ggt   gta   gtc   gtg 
gtt
##   391   174   203   639   240   121    68   447   176   155   104   564   599   229   491 
1100
##   taa   tac   tag   tat   tca   tcc   tcg   tct   tga   tgc   tgg   tgt   tta   ttc   ttg 
ttt
##   978   490   551  1211   378   176   107   594   695   437   586  1140  1350   497  1045 
1771

# sequence MH940245
t2= count(getSequence(Seq2[[1]]), wordsize = 3)
print("frequency of each amino acid in sequences AY884001")

## [1] "frequency of each amino acid in sequences AY884001"

t2

##
##   aaa   aac   aag   aat   aca   acc   acg   act   aga   agc   agg   agt   ata   atc   atg 
att
##   747   311   503   919   354   199   100   490   478   229   260   553   738   320   891 
1167
##   caa   cac   cag   cat   cca   ccc   ccg   cct   cga   cgc   cgg   cgt   cta   ctc   ctg 
ctt
##   364   169   264   347   172    75    47   277    56    54    49   162   543   209   431 
625
##   gaa   gac   gag   gat   gca   gcc   gcg   gct   gga   ggc   ggg   ggt   gta   gtc   gtg 
gtt
##   391   174   202   639   240   121    67   447   176   155   104   564   599   229   491 
1100
##   taa   tac   tag   tat   tca   tcc   tcg   tct   tga   tgc   tgg   tgt   tta   ttc   ttg 
ttt
##   978   490   551  1211   378   176   107   594   695   437   586  1140  1350   497  1045 
1772

# Check if they are equal
t1==t2
```

```
##
##   aaa   aac   aag   aat   aca   acc   acg   act   aga   agc   agg   agt
ata
##  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
TRUE
##   atc   atg   att   caa   cac   cag   cat   cca   ccc   ccg   cct   cga
cgc
##  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE
TRUE
##   cgg   cgt   cta   ctc   ctg   ctt   gaa   gac   gag   gat   gca   gcc
gcg
##  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
FALSE
##   gct   gga   ggc   ggg   ggt   gta   gtc   gtg   gtt   taa   tac   tag
tat
##  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE
##   tca   tcc   tcg   tct   tga   tgc   tgg   tgt   tta   ttc   ttg   ttt
##  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE

# Draw Pie Chart
old.par <- par(mfrow=c(1, 2))
pie(t1,main="AY884001")
pie(t2,main="MH940245")
```

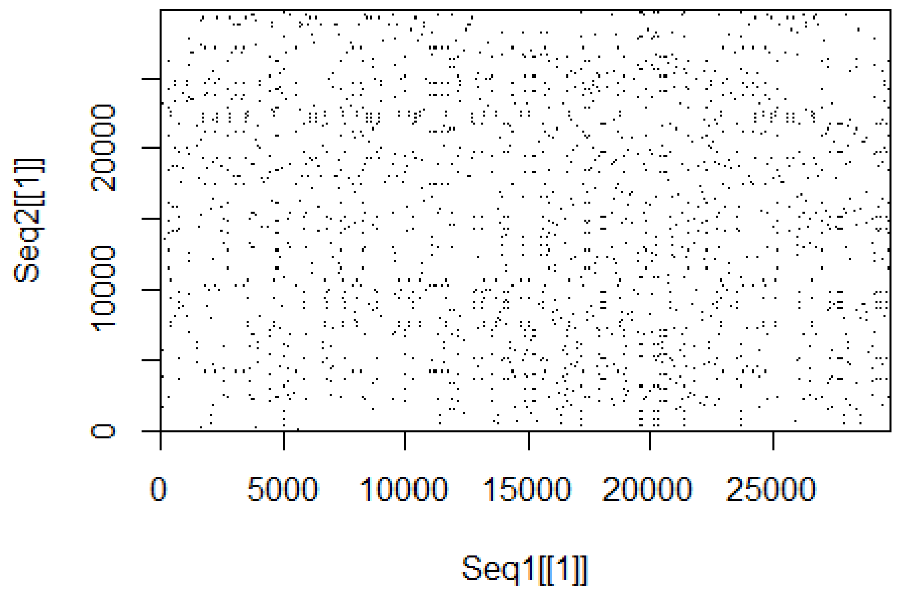


```
par(old.par)
```

Question 3:

Create a dot plot for the first ORF in each of the sequences. Comment on the result.

```
dotPlot(Seq1[[1]], Seq2[[1]], wsize = 3, wstep = 3, nmatch = 3)
```



In this case there is no dots along a diagonal line, which indicates that the two protein sequences don’t contain the same identical amino acids.

Question 4:

Find the optimal global alignment between the two sequences and print the alignment for the first 20 nucleotides. Use a score +2 for a match, -1 for a mismatch, and the gap penalty = 2.

```
#Transform to Upper Case
Seq1 <- toupper(c2s(Seq1[[1]]))
Seq2 <- toupper(c2s(Seq2[[1]]))

sigma <- nucleotideSubstitutionMatrix(match = 2, mismatch = -1, baseOnly =
TRUE)
sigma # Print out the matrix

##    A  C  G  T
## A  2 -1 -1 -1
## C -1  2 -1 -1
## G -1 -1  2 -1
## T -1 -1 -1  2

# Optimal Global Aglinment
pairwiseSeq1Seq2<-pairwiseAlignment(Seq1, Seq2, substitutionMatrix = sigma,
gapOpening = 2, scoreOnly = FALSE)

pairwiseSeq1Seq2

## Global PairwiseAlignmentsSingleSubject (1 of 1)
## pattern:
GAGCGATTGACGTTCGTACCGTCTATCAGCTTAC...ATTGAAATTAATTATAGCCTTTTGGAGGAATTAC
## subject: GA----
TTGACGTTCGTACCGTCTATCAGCTTAC...ATTGAAATTAATTATAGCCTTTTGGAGGAATTAC
## score: 59601

pairwiseSeq1Seq2@score

## [1] 59601
```

Question 5:

Is the global alignment statistically significant? Explain your answer

```
generateSeqsWithMultinomialModel <- function(inputsequence, n)
{
  # Change the input sequence into a vector of letters
    require("seqinr") # This function requires the SeqinR package.
    inputsequencevector <- s2c(inputsequence)
    #inputsequencevector <- inputsequence
    # Find the frequencies of the letters in the input sequence
"inputsequencevector":
    mylength <- length(inputsequencevector)
    mytable <- table(inputsequencevector)
    # Find the names of the letters in the sequence
    letters <- rownames(mytable)
    numletters <- length(letters)
    probabilities <- numeric() # Make a vector to store the probabilities of
letters
    for (i in 1:numletters)
    {
      letter <- letters[i]
      count <- mytable[[i]]
      probabilities[i] <- count/mylength
    }
    # Make n random sequences using the multinomial model with probabilities
"probabilities"
    seqs <- vector("list", n)
    for (j in 1:n)
    {
      seq <- sample(letters, mylength, rep=TRUE, prob=probabilities) # Sample
      seq <- c2s(seq)
      seqs[[j]] <- seq
    }


# Return the vector of random sequences
return(seqs)
}
```

- Create a vector of 1000 random sequences.

```
randomSeq <- generateSeqsWithMultinomialModel(Seq2,1000)
```

- Use PairwiseAlignment to get the score vector of 1000 random vector and store in randomScore

```
randomscores <- double(1000)

for (i in 1:1000)
{
  score <- pairwiseAlignment(Seq1, randomSeq[[i]], substitutionMatrix =
sigma, gapOpening = 2, scoreOnly = TRUE)
  randomscores[i] <- score
}

Pvalue<- sum(randomscores >= pairwiseSeq1Seq2@score)/1000
```

[1] 0

The P value is 0, so that they are probably not related sequences

Question 6:

What is the score of the optimal local alignment between the two sequences? What is the length of the aligned segments? Use a score +3 for a match,-2 for a mismatch, the gap penalty = 4, and gap extension = 2.

```
sigma <- nucleotideSubstitutionMatrix(match = 3, mismatch = -2, baseOnly =
TRUE)
sigma # Print out the matrix

##    A  C  G  T
## A  3 -2 -2 -2
## C -2  3 -2 -2
## G -2 -2  3 -2
## T -2 -2 -2  3

pairwiseAlignment(Seq1, Seq2, substitutionMatrix = sigma, gapOpening = 4,
gapExtension= 2,  scoreOnly = FALSE)

## Global PairwiseAlignmentsSingleSubject (1 of 1)
## pattern:
GAGCGATTGACGTTCGTACCGTCTATCAGCTTAC...ATTGAAATTAATTATAGCCTTTTGGAGGAATTAC
```

```
## subject: GA----
TTGACGTTCGTACCGTCTATCAGCTTAC...ATTGAAATTAATTATAGCCTTTTGGAGGAATTAC
## score: 89416
```