Exam2

Whitney Domico

2022-11-06

## Question 1

I used the command diff -u Sample2Data\_1.txt ‘Sample2Data (1).txt’ and it showed that the files do differ. Sample2Data\_1.txt has ‘Pikachurin’ at line 126 while the ‘Sample2Data (1).txt’ has TAM410.

## Question 2

I used the command wc -l for each of the text files and found that only 3 and 4 had the correct number of lines. Then I used the command /carolina to search for information on the box turtle, and I found it in the 4th file (SpeciesData4.txt), so this is the correct file.

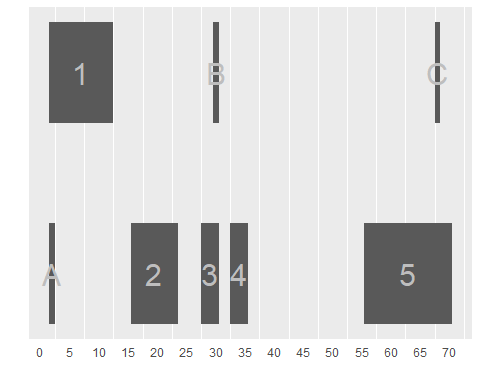
## Question 3

DNA contains genetic information in the form of a specific sequence of base pairs (A,C,G,T). This is an organism’s ‘genotype’; it is what is directly coded for in the DNA. RNA is then transcribed from the coding strand of DNA and is identical to the non-coding strand of DNA, except that it has U instead of T. For mRNA in particular, ribosomes in the cytoplasm translate these series of bases (in groups of three, called ‘codons’) into polypeptide chains where each codon codes for a particular amino acid. This newly formed protein – whether just the expression/nonexpression or what the sequence of amino acids is – is the ‘phenotype’; in other words, it is how the genotype’s coding is expressed in the organism. So, for example, perhaps a particular genotype is – ACCTGT. This would code for an mRNA that is UGGACA and the protein expression would be Trp, Thr. For simplification, maybe this codes for brown eyes (phenotype). If a mutation occurred in the DNA, maybe the genotype would be changed to ACCTGG, which would give the mRNA UGGACC. This still codes for Trp, Thr, and so the phenotype would be identical (in this case, brown eyes), even though the genotype has changed.

## Question 4

This code has been modified from RangeAndSequence.Rmd

#given IRange data  
x <- IRanges(start=c(2, 16, 28, 33, 56, 2, 30, 68), end=c(12, 23, 30, 35, 70, 2, 30, 68))  
names(x) <- c('1','2','3','4','5','A','B','C')  
  
#setting up parameters for ggplot; saved y, ymin, ymax, start, end, width into df 'out' for simplicity  
sep=0.5  
height=0.5  
out <- as.data.frame(x)  
out$y <- disjointBins(x)  
out$ymin <- out$y \* (sep + height) - height  
out$ymax <- out$ymin + height  
xmin <- min(out$start)  
xmax <- max(out$end)  
xbreaks <- seq(0, xmax, by=5)  
xlabels <- seq(0, xmax, by=5)  
  
#plot  
ggplot(out) + geom\_rect(aes(ymin=ymin, ymax=ymax, xmin=start-0.5, xmax=end+0.5)) + geom\_text(aes(x=start + width/2 - 0.5, y=ymin+(ymax-ymin)/2, label=names), size=8, color="grey") + scale\_x\_continuous(breaks=xbreaks, labels=xlabels) + xlab("") + ylab("") + theme(axis.text.y=element\_blank()) + theme(panel.grid.major=element\_blank(), panel.grid.minor.y=element\_blank(), axis.ticks=element\_blank())



#SNP A overlaps with gene 1, SNP B overlaps with gene 3, SNP C overlaps with gene 5

## Question 5

#given GRange data  
gr <- GRanges(seqname=c("chr1", "chr1", "chr2", "chr3","chr1", "chr6", "chr2", "chr3","chr1", "chr4", "chr2", "chr4","chr10", "chr1", "chr2", "chr6","chr1", "chr", "chr6", "chr3"),  
ranges=IRanges(start=c(2, 3, 5, 7, 9, 12, 13, 16, 18, 22, 23, 26, 30, 36, 38, 41, 46, 56, 57, 59), end=c(19,19,18,26,13,24,21,31,35,25,26,44,33,39,54,50,57,60,74,74)),  
strand=c("+", "-", "-", "+","+", "-", "-", "+","+", "-", "-", "+","+", "-", "-", "+","+", "-", "-", "+"),identifier=letters[1:20])  
  
#finding which width is the largest in gr  
max(width(gr))

## [1] 20

#because max is largest, there will be only one item that is >=max, so I can find the other answers using this  
  
gr[width(gr)>=max(width(gr))]

## GRanges object with 1 range and 1 metadata column:  
## seqnames ranges strand | identifier  
## <Rle> <IRanges> <Rle> | <character>  
## [1] chr3 7-26 + | d  
## -------  
## seqinfo: 7 sequences from an unspecified genome; no seqlengths

#this shows chr3, range 7-26, +, d  
  
#Therefore the answer is that it is 20 bp width, chr3, with identifier 'd'

## Question 6

This code has been modified from RangeandSequence.Rmd

#uploading fasta file as "forfasta" and saving as dataframe "fordf"  
forfasta <- readDNAStringSet('C:/Users/Whitn/OneDrive/Desktop/BIO247/BIO247Lecture/BIO247Exams/BIO247Exam2/Exam 2/Forest\_Seqs.fasta')  
seq\_name = names(forfasta)  
sequence = paste(forfasta)  
fordf = data.frame(seq\_name,sequence)  
  
#making a vector of all seq names that begin with "MPM" because those are his initials  
MPM <- c()  
row <- 1:length(fordf$seq\_name)  
for (each in fordf$seq\_name){  
 if (grepl("MPM", each)){  
 MPM <- c(MPM, each)  
 }  
}  
#putting MPM in order for ease  
MPM <- sort(MPM)  
  
#using a loop to find corresponding sequences to seq names found above  
row <- c(1:length(fordf$seq\_name))  
MPMseq <- c()  
for (each in MPM){  
 for (each2 in row){  
 if (each == fordf$seq\_name[each2]){  
 MPMseq <- c(MPMseq, fordf$sequence[each2])  
 }  
 }  
}  
  
  
#making a dataframe and saving as an excel file to send  
MPMdf <- data.frame("seqname"=MPM, "sequence"=MPMseq)  
  
write\_xlsx(MPMdf, "C:/Users/Whitn/OneDrive/Desktop/BIO247/BIO247Lecture/BIO247Exams/BIO247Exam2/Exam 2/DomicoMPMdf.xlsx")

## Question 7

This code has been modified from RangeAndSequence.Rmd

# uploading FASTQ file  
fastq <- read.table('C:/Users/Whitn/OneDrive/Desktop/BIO247/BIO247Lecture/BIO247Exams/BIO247Exam2/Exam 2/Gene\_compare.fastq', header=FALSE,sep="|",fill=TRUE)  
  
  
#making vectors with indices since the file is "name, seq, +, score"  
name <- seq(1, length(fastq$V1), by=4)  
seq <- seq(2, length(fastq$V1), by=4)  
score <- seq(4, length(fastq$V1), by=4)  
  
#using vectors from above to make vectors of name, sequence, and scorechar  
Name <- fastq$V1[name]  
Sequence <- fastq$V1[seq]  
Score\_char <- fastq$V1[score]  
Score\_val = c()  
  
#creating initial data frame  
fastq\_data <- data.frame(Name, Sequence, Score\_char)  
  
#finding corresponding ASCII-Sanger-PHRED scores for both samples  
for (x in 1:length(fastq\_data$Name)){  
 score\_num = c()  
 char <- unlist(strsplit(fastq\_data$Score\_char[x],""))  
 for (a in char){  
 score\_num <- append(score\_num,utf8ToInt(a)-33)  
 }  
 Score\_val <- append(Score\_val,list(score\_num))  
}  
  
# adding each list of score values to a column in the dataframe  
for (i in 1:(length(Score\_val))){  
 fastq\_data$Score\_val[i] <- list(Score\_val[i])  
}  
  
#making each sample's score value into a vector (not list)and definding a vector as the difference in scores for each bp  
samp1 <- unlist(fastq\_data$Score\_val[1])  
samp2 <- unlist(fastq\_data$Score\_val[2])  
sampdiff <- samp1-samp2  
  
  
#finding which sample has more "wins" for each bp (if sampdiff[each] is positive, that means the score in samp1 is higher; if sampdiff[each] is negative, that meanst the score in samp2 is higher)  
samp1freq <- c(0)  
samp2freq <- c(0)  
eqfreq <- c(0)  
for (each in sampdiff){  
 if (each>0){  
 samp1freq <- samp1freq+1  
 } else if (each<0){  
 samp2freq <- samp2freq+1  
 } else  
 eqfreq <- eqfreq+1  
}  
  
samp1freq

## [1] 29

samp2freq

## [1] 13

eqfreq

## [1] 10

#samp1freq > samp2freq, so sample1 probably has better quality  
  
#quick check of average score for each sample to see if it agrees  
mean(samp1)

## [1] 35.71154

mean(samp2)

## [1] 34.34615

#sample 1 mean > sample 2 mean, so this supports that sample1 has better quality