Rosalind solutions

Kevin Ryan

6/13/2022

Some R code to solve a few Rosalind challenges

Counting DNA nucleotides

```
dna <- "AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAAAAGAGTGTCTGATAGCAGC"
```

```
count_nucelotides <- function(dna){
  require(stringr)
  dna_split <- str_split(dna, pattern = "")
  a <- as.integer(0)
  c <- as.integer(0)
  g <- as.integer(0)
  t <- as.integer(0)
  for (i in 1:length(dna_split[[1]])){
    if (dna_split[[1]][i] == "A"){
      a <- a + 1
    } else if (dna_split[[1]][i] == "C") {
      c <- c + 1
  } else if (dna_split[[1]][i] == "G"){
      g <- g + 1
  } else if (dna_split[[1]][i] == "T"){
    t <- t + 1
  }
  }
 answer <- paste(a, c, g, t, sep = " ")
 return(answer)
}
```

```
count_nucelotides(dna)
```

```
## Loading required package: stringr
```

```
## [1] "20 12 17 21"
```

```
test_data <- "CTGCGACATGCCGCGGCCAGTCACGTAACACGTGTATGACCCGACACTCCCTTATAATTTGAACAGTACAAAGTGTAAATTTATCCAAT
ans <- count_nucelotides(test_data)
ans
```

```
## [1] "184 211 192 219"
```

Transcribing DNA to RNA

```
dna_to_rna <- function(dna){
  require(stringr)
  dna_split <- str_split(dna, pattern = "")
  answer <- ""
  for (i in 1:length(dna_split[[1]])){
    if (dna_split[[1]][i] == "T"){
      answer <- paste(answer, "U", sep = "")
    } else {
      answer <- paste(answer, dna_split[[1]][i], sep = "")
    }
  }
 return(answer)
}
```

```
sam_data <- "ATAACACTCCAATGGCCTGAGCTCGGGACACTAGTGGGGAAGACGATCTTTCTGCTCATGTGGCGAAGACCAAATTGACACGTCAAAAAT
ans <- dna_to_rna(sam_data)
ans
```

```
## [1] "AUAACACUCCAAUGGCCUGAGCUCGGGACACUAGUGGGGAAGACGAUCUUUCUGCUCAUGUGGCGAAGACCAAAUUGACACGUCAAAAAUAAAAA
```

Reverse complement

```
reverse_complement <- function(dna){
  require(stringr)
  # reverse string
  string_split <- strsplit(dna, NULL)[[1]]
  reversed_string <- paste(rev(string_split), collapse="")
  dna_split <- str_split(reversed_string, pattern = "")
  answer <- ""
  for (i in 1:length(dna_split[[1]])){
    if (dna_split[[1]][i] == "T"){
      answer <- paste(answer, "A", sep = "")
    } else if (dna_split[[1]][i] == "A"){
      answer <- paste(answer, "T", sep = "")
    } else if (dna_split[[1]][i] == "C"){
      answer <- paste(answer, "G", sep = "")
    } else if (dna_split[[1]][i] == "G"){
      answer <- paste(answer, "C", sep = "")
    }
  }
 return(answer)
}
```

```
test_data <- "TCGAAAAACTCGGCCGTATGGCTAGGCCTAGCGAATACCGGGGCCGTCGGAAGTATACCTTCGGTACCCACACGGATTGCTAGGTTTCGT
ans <- reverse_complement(test_data)
ans
```

```
## [1] "AATGTATAGGGGAGTTGAAGTGTAGCCTATAAGAAGTGGGACCAAGGCTCGCCCGCGGGGGGTCCAGCATGTGAAATTTTAAGCATCTCTCATAGT
```

Computing GC Content

```
gc_content <- function(path_to_fasta){
  require(seqinr)
  require(stringr)
  seq_object <- read.fasta(path_to_fasta, seqtype = "DNA", as.string = TRUE,set.attributes = FALSE)
  gc <- c()
  for (i in 1:length(seq_object)){
    nuc_sequence <- seq_object[[i]]
    seq_length <- nchar(nuc_sequence)
    seq_name <- names(seq_object[i])
    dna_split <- str_split(nuc_sequence, pattern = "")
    gcs <- 0
    for (j in 1:seq_length){
      if (dna_split[[1]][j] == "g" || dna_split[[1]][j] == "c"){
        gcs <- gcs + 1
      }
    }
    gc_percentage <- c((gcs/seq_length)*100)
    names(gc_percentage) <- seq_name
    gc <- c(gc, gc_percentage)
  }
  result <- gc[which.max(gc)]
  output <- data.frame(as.list(result))
  return(output)
}
```

```
path_to_fasta <- "/home/kevin/Downloads/rosalind_gc.txt"
result <- gc_content(path_to_fasta = path_to_fasta)
```

```
## Loading required package: seqinr
```

```
write.table(result, file = "/home/kevin/Documents/PhD/hackathon/gc_content_result.txt",
            quote = F, row.names = F)
```

Hamming distance

```
hamming_distance <- function(data_frame_seqs){
  require(stringr)
  seq1 <- data_frame_seqs[1,]
  seq2 <- data_frame_seqs[2,]
  stopifnot("sequences must be of equal length"=length(seq1) == length(seq2))
  seq1_split <- str_split(seq1, pattern = "")
  seq2_split <- str_split(seq2, pattern = "")
  len_seqs <- nchar(seq1)
  hamming <- 0
  for (i in 1:len_seqs){
    if (seq1_split[[1]][i] != seq2_split[[1]][i]){
      hamming <- hamming + 1
    }
```

```
  }
  return(hamming)
}
```

```
seqs <- read.table("/home/kevin/Downloads/rosalind_hamm.txt")
hamming_distance(data_frame_seqs = seqs)
```

```
## [1] 489
```