Genome Functions to Use Later

Brief description of each function

Libraries used

```
library(stringr)
library(stringi)
```

Example sequences:

```
seq1 <- "gtatgggaatcagccgggtctcactatgtgcaaaggagattcggtcgtgtggtacttattcag"
seq2 <- "gtatgggaatcagccgggtctcactatgtgcaaa"
seq3 <- "gtatgggaat"

sequence1 <- toupper(seq1)
sequence1
```

```
[1] "GTATGGGAATCAGCCGGGTCTCACTATGTGCAAAGGAGATTCGGTCGTGTGGTACTTATTCAG"
```

```
sequence2 <- toupper(seq2)
sequence2
```

```
[1] "GTATGGGAATCAGCCGGGTCTCACTATGTGCAAA"
```

```
sequence3 <- toupper(seq3)
sequence3
```

```
[1] "GTATGGGAAT"
```

Counts per Base

```
bases_count <- function(sequence, bases = c("A", "T", "C", "G")) {
  base_counts <- str_count(sequence, bases)
  names(base_counts) <- bases
  return(base_counts)
}

bases_count(sequence1)
```

```
 A  T  C  G
14 18 11 20
```

```
bases_count(sequence2)
```

```
 A  T  C  G
 9  8  7 10
```

```
bases_count(sequence3)
```

```
A T C G
3 3 0 4
```

Percentage per Base

```
bases_percentage <- function(sequence, bases = c("A", "T", "C", "G")) {
  base_percs <- bases_count(sequence, bases) / str_length(sequence)
  names(base_percs) <- bases
  return(base_percs)
}

bases_percentage(sequence1)
```

```
        A         T         C         G
0.2222222 0.2857143 0.1746032 0.3174603
```

```
bases_percentage(sequence2)
```

```
        A         T         C         G
0.2647059 0.2352941 0.2058824 0.2941176
```

```
bases_percentage(sequence3)
```

```
  A   T   C   G
0.3 0.3 0.0 0.4
```

GC Percentage

```
gc_percentage <- function(sequence, bases = c("A", "T", "C", "G")) {
  base_percs <- bases_percentage(sequence, bases)
  return(base_percs[3] + base_percs[4])
}

gc_percentage(sequence1)
```

```
        C
0.4920635
```

```
gc_percentage(sequence2)
```

```
  C
0.5
```

```
gc_percentage(sequence3)
```

```
  C
0.4
```

Base Highlight

```
highlight_base <- function(sequence, base) {
  BASE <- toupper(base)
  base <- tolower(base)
  sequence <- tolower(sequence)
  return(gsub(base, BASE, sequence))
}

highlight_base(sequence1, "a")
```

```
[1] "gtAtgggAAtcAgccgggtctcActAtgtgcAAAggAgAttcggtcgtgtggtActtAttcAg"
```

```
highlight_base(sequence2, "a")
```

```
[1] "gtAtgggAAtcAgccgggtctcActAtgtgcAAA"
```

```
highlight_base(sequence3, "a")
```

```
[1] "gtAtgggAAt"
```

Reverse Complementary

```
rev_complement <- function(sequence, bases = "ATCG", replace_bases = "TAGC") {
  return(stri_reverse(chartr(bases, replace_bases, sequence)))
}

sequence1
```

```
[1] "GTATGGGAATCAGCCGGGTCTCACTATGTGCAAAGGAGATTCGGTCGTGTGGTACTTATTCAG"
```

```
rev_complement(sequence1)
```

```
[1] "CTGAATAAGTACCACACGACCGAATCTCCTTTGCACATAGTGAGACCCGGCTGATTCCCATAC"
```

```
sequence2
```

```
[1] "GTATGGGAATCAGCCGGGTCTCACTATGTGCAAA"
```

```
rev_complement(sequence2)
```

```
[1] "TTTGCACATAGTGAGACCCGGCTGATTCCCATAC"
```

```
sequence3
```

```
[1] "GTATGGGAAT"
```

```
rev_complement(sequence3)
```

```
[1] "ATTCCCATAC"
```

Kmer Combinations

```
combi_kmers <- function(bases = c("A", "T", "C", "G"), k = 2) {
  return(do.call(paste0, expand.grid(rep(list(bases), k))))
}

combi_kmers()
```

```
 [1] "AA" "TA" "CA" "GA" "AT" "TT" "CT" "GT" "AC" "TC" "CC" "GC" "AG" "TG" "CG"
[16] "GG"
```

```
combi_kmers(k = 3)
```

```
 [1] "AAA" "TAA" "CAA" "GAA" "ATA" "TTA" "CTA" "GTA" "ACA" "TCA" "CCA" "GCA"
[13] "AGA" "TGA" "CGA" "GGA" "AAT" "TAT" "CAT" "GAT" "ATT" "TTT" "CTT" "GTT"
[25] "ACT" "TCT" "CCT" "GCT" "AGT" "TGT" "CGT" "GGT" "AAC" "TAC" "CAC" "GAC"
[37] "ATC" "TTC" "CTC" "GTC" "ACC" "TCC" "CCC" "GCC" "AGC" "TGC" "CGC" "GGC"
[49] "AAG" "TAG" "CAG" "GAG" "ATG" "TTG" "CTG" "GTG" "ACG" "TCG" "CCG" "GCG"
[61] "AGG" "TGG" "CGG" "GGG"
```

Counts per Kmer

```
count_kmers <- function(sequence, vector_kmers = combi_kmers()) {
  kmer_counts <- stri_count_fixed(sequence, vector_kmers, overlap = TRUE)
  names(kmer_counts) <- vector_kmers
  return(kmer_counts)
}

count_kmers(sequence3)
```

```
AA TA CA GA AT TT CT GT AC TC CC GC AG TG CG GG
 1  1  0  1  2  0  0  1  0  0  0  0  0  1  0  2
```

```
count_kmers(sequence3, combi_kmers(k = 3))
```

```
AAA TAA CAA GAA ATA TTA CTA GTA ACA TCA CCA GCA AGA TGA CGA GGA AAT TAT CAT GAT
  0   0   0   1   0   0   0   1   0   0   0   0   0   0   0   1   1   1   0   0
ATT TTT CTT GTT ACT TCT CCT GCT AGT TGT CGT GGT AAC TAC CAC GAC ATC TTC CTC GTC
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
ACC TCC CCC GCC AGC TGC CGC GGC AAG TAG CAG GAG ATG TTG CTG GTG ACG TCG CCG GCG
  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0
AGG TGG CGG GGG
  0   1   0   1
```

Kmer Windows

```
kmer_windws <- function(sequence, k = 2) {
  seq_len <- str_length(sequence)
  return(str_sub(sequence, seq(1, seq_len + 1 - k), seq(k, seq_len)))
}

sequence3
```

```
[1] "GTATGGGAAT"
```

```
kmer_windws(sequence3)
```

```
[1] "GT" "TA" "AT" "TG" "GG" "GG" "GA" "AA" "AT"
```

```
kmer_windws(sequence3, k = 3)
```

```
[1] "GTA" "TAT" "ATG" "TGG" "GGG" "GGA" "GAA" "AAT"
```

TM Calculation

```
tm_calc <- function(sequence, bases = c("A", "T", "C", "G")) {
  base_counts <- bases_count(sequence, bases)
  seq_len <- str_length(sequence)
  countA <- base_counts[1]
  countT <- base_counts[2]
  countC <- base_counts[3]
  countG <- base_counts[4]
  if (seq_len <= 13) {
    temp <- ((countA + countT) * 2) + ((countC + countG) * 4)
  } else {
    temp <- 64.9 + (41 * (countG + countC - 16.4) / (countA + countT + countC + countG))
  }
  names(temp) <- "TM in Celcius"
  return(temp)
}

tm_calc(sequence3)
```

```
TM in Celcius
           28
```

```
tm_calc(sequence2)
```

```
TM in Celcius
     65.62353
```

TM Calculation (Sequence Length less than 14 bp)

```
tm_len_lt14 <- function(sequence, bases = c("A", "T", "C", "G")) {
  base_counts <- bases_count(sequence, bases)
  countA <- base_counts[1]
  countT <- base_counts[2]
  countC <- base_counts[3]
  countG <- base_counts[4]
  temp <- ((countA + countT) * 2) + ((countC + countG) * 4)
  names(temp) <- "TM in Celcius"
  return(temp)
}

tm_len_lt14(sequence3)
```

```
TM in Celcius
           28
```

TM Calculation (Sequence Length more than 13 bp)

```
tm_len_mt13 <- function(sequence, bases = c("A", "T", "C", "G")) {
  base_counts <- bases_count(sequence, bases)
  countA <- base_counts[1]
  countT <- base_counts[2]
  countC <- base_counts[3]
  countG <- base_counts[4]
  temp <- 64.9 + (41 * (countG + countC - 16.4) / (countA + countT + countC + countG))
  names(temp) <- "TM in Celcius"
  return(temp)
}

tm_len_mt13(sequence2)
```

```
TM in Celcius
     65.62353
```