Practical 1

Statistical Genetics

Anna Putina Marine Mauzeau

2023-11-14

```
library(dplyr)
library(tidyr)
library(stringr)
library(ggplot2)
library(tidyverse)
```

SNP dataset

The file TSICHR22RAW.raw contains genotype information of individuals from Tuscany in Italy, taken from the 1,000 Genomes project. The datafile contains all single nucleotide polymorphisms on chromosome 22 for which complete information is available. Load this data into the R environment, with the read.table instruction. The first six columns contain non-genetical information. Create a dataframe that only contains the genetic information that is in and beyond the 7th column. Notice that the genetic variants are identifed by an "rs" identifier. The genetic data is coded in the (0, 1, 2) format with 0=AA, 1=AB, 2=BB.

```
data <- read.table("TSICHR22RAW.raw")
colnames(data) <- data[1,]
data <- data[-1, -c(1:6)]
data <- as.data.frame(sapply(data, as.numeric))
```

Question 1

How many variants are there in this database? What percentage of the data is missing?

```
ncol(data)
```

```
## [1] 20649
```

```
NA.counter <- sum(is.na.data.frame(data))
NA.counter
```

```
## [1] 4184
```

```
NA.percentage <- NA.counter*100/prod(dim(data))
NA.percentage
```

```
## [1] 0.1986518
```

There are 20649 columns in this dataframe, i.e. 20649 variants. There are 4184 missing values in this data, i.e. 0.199% of the data. We could remove rows containing NAs but as we can see in the code below, there is at least 1 NA per row.

```
rows.has.na <- function(data){
  for (i in 1:nrow(data)) {
  if (anyNA(data[i,]) == F){
    print(i)
    }
  }
  print("All rows contain at least one NA")
}

rows.has.na(data)
```

```
## [1] "All rows contain at least one NA"
```

We will then replace the NAs by the most frequent value (0, 1 or 2) they belong to.

```
# computes and returns the most frequent value of a column
mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

# replace na
data.without.na <- data %>%
  mutate_all(funs(ifelse(is.na(.), mode(.), .)))
```

```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## i Please use a list of either functions or lambdas:
##
## # Simple named list: list(mean = mean, median = median)
##
## # Auto named with `tibble::lst()`: tibble::lst(mean, median)
##
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
data <- data.without.na
```

Question 2

Calculate the percentage of monomorphic variants. Exclude all monomorphics from the database for all posterior computations of the practical. How many variants do remain in your database?

Monomorphic variants are variants that only have one allele, i.e. either only 0 or only 2 in a column of the dataframe means that the variant is monomorphic.

```
is.not.monomorphic <- function(col){
  return(!(sum(col) %in% c(0, 2*(nrow(data)))))
}

data <- data[, sapply(data, is.not.monomorphic)]

nb.variants <- ncol(data)
nb.variants
```

```
## [1] 18283
```

After removing all monomorphic variants, there are 18283 variants.

Question 3

Report the genotype counts and the minor allele count of polymorphism rs8138488_C, and calculate the MAF of this variant.

```r
rs8138488_C <- data$rs8138488_C

genotype.counts <- table(rs8138488_C)
genotype.counts
```

```
## rs8138488_C
##  0  1  2
## 41 47 14
```

```r
count.B <- genotype.counts[2] + 2*genotype.counts[3]
count.B <- count.B[["1"]]
count.B
```

```
## [1] 75
```

```r
tot.nb.alleles <- 2*nrow(data)

MAF.B <- count.B / tot.nb.alleles
MAF.B
```

```
## [1] 0.3676471
```

Genotype counts : - 0=AA : 47 - 1=AB : 47 - 2=BB : 14

The minor allele is B so the minor allele count of polymorphism rs8138488_C is 75. The MAF is equal to 0.368.

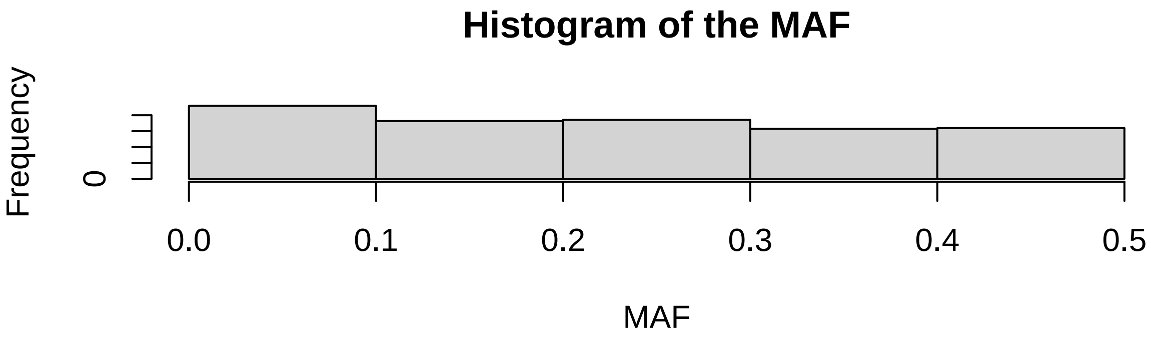
Question 4

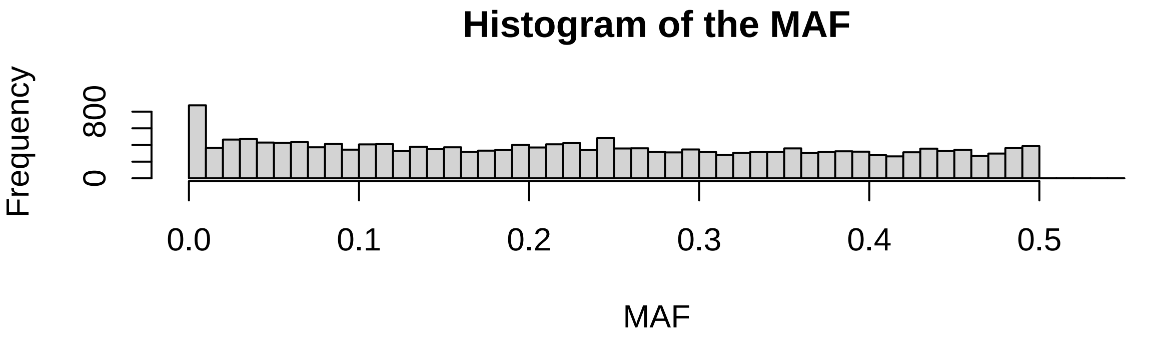
Compute the minor allele frequencies (MAF) for all markers, and make a histogram of it. Does the MAF follow a uniform distribution? What percentage of the markers have a MAF below 0.05? And below 0.01? Can you explain the observed pattern?

```r
compute.MAFs <- function(data){

  MAFs <- c()
  tot.nb.alleles <- 2*nrow(data)

  for (col_name in colnames(data)) {
    col <- data[[col_name]]

    count.A <- sum(col == 1) + 2*sum(col == 0)
    count.B <- 204 - count.A

    MAFs <- c(MAFs, min(count.A, count.B) / tot.nb.alleles)

  }
  return(MAFs)
}
```

```r
MAFs <- compute.MAFs(data)
```

```r
par(mfrow = c(2, 1))
hist(MAFs, main = "Histogram of the MAF", xlab = "MAF", breaks = seq(0, 0.55 , by = 0.1))
hist(MAFs, main = "Histogram of the MAF", xlab = "MAF", breaks = seq(0, 0.55 , by = 0.01))
```





If we look at the 1st histogram, it looks like a uniform distribution. However, if we look at it more precisely by increasing the number of breaks (2nd histogram), there are lots of very low values, so we conclude that the MAF doesn't follow a normal distribution.

```r
below.0.05 <- sum(MAFs < 0.05)*100/nb.variants
below.0.05
```

```
## [1] 14.25368
```

```r
below.0.01 <- sum(MAFs < 0.01)*100/nb.variants
below.0.01
```

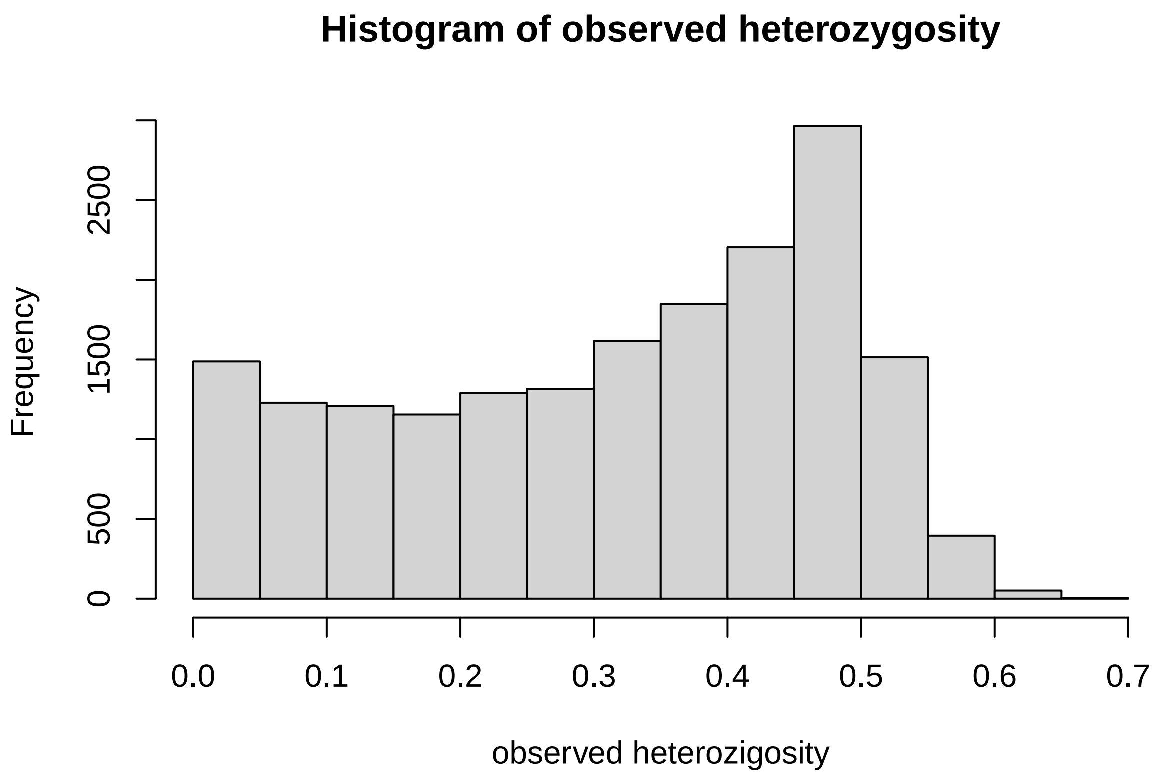
```
## [1] 4.791336
```

There are approximately 14.3% of the variants that have a MAF below 0.05. There are approximately 4.8% of the variants that have a MAF below 0.01. It looks like the most frequent values for MAF are the closest to 0. It is normal because we are looking at the proportion of minor allele, which are rare. This can be explained by several factors : - Mutations : Genetic mutations introduce new alleles into a population. If a mutation is rare, it will have a low allele frequency. - Natural Selection : if the minor allele is disadvantageous, it may have low allele frequency due to natural selection because the individuals that hold this allele reproduce themselves less than others.

Question 5

Calculate the observed heterozygosity H0, and make a histogram of it. What is, theoretically, the range of variation of this statistic?

```
compute.all.H0 <- function(data){
  all.H0 <- c()
  for (col_name in colnames(data)) {
  col <- data[[col_name]]
  all.H0 <- c(all.H0, sum(col == 1)/nrow(data))
}
  return(all.H0)
}

H0 <- compute.all.H0(data)

hist(H0, main = "Histogram of observed heterozygosity", xlab = "observed heterozigosity")
```

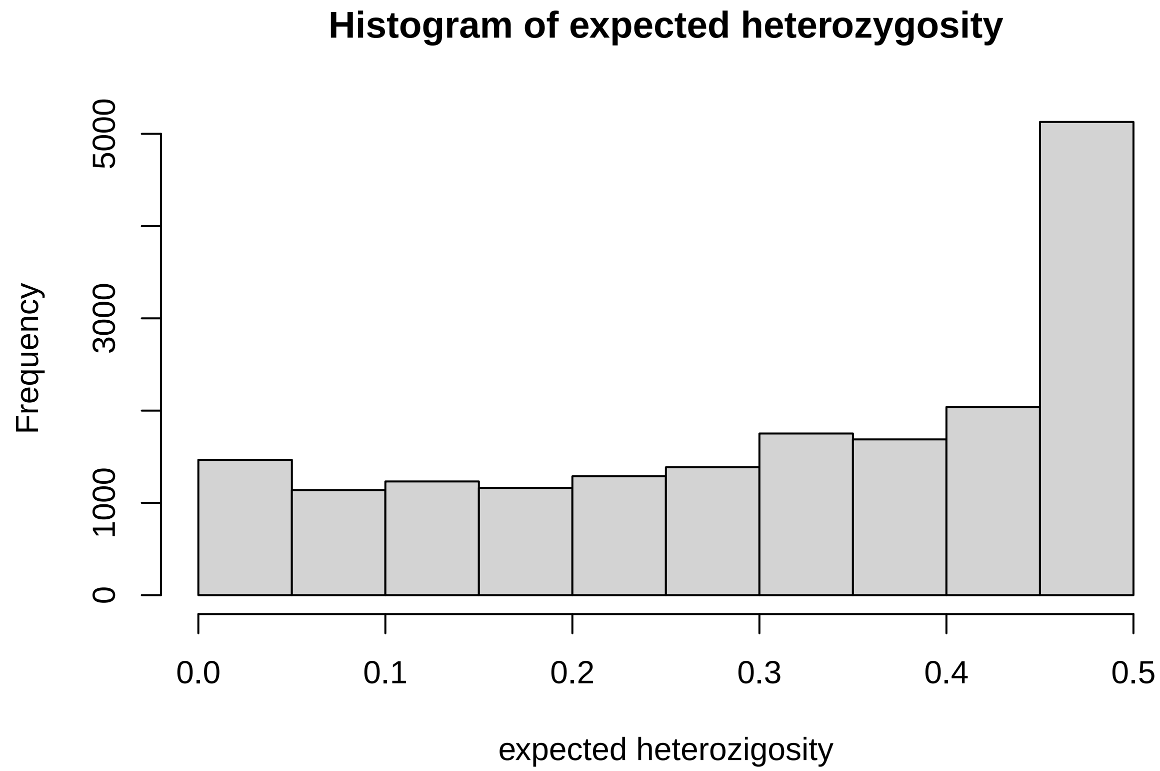


Theoretically, H_0 lies between 0 and 1 : - Minimum value (0): If all individuals at a particular locus are homozygous (i.e., have identical alleles), the observed heterozygosity would be 0. - Maximum value (1): If all individuals at a particular locus are heterozygous (i.e., have different alleles), the observed heterozygosity would be 1. However, this statistic is influenced by many different factors such as the population's history, size, mating patterns, and natural selection, which is why it is more complex than H_0's values following a uniform distribution.

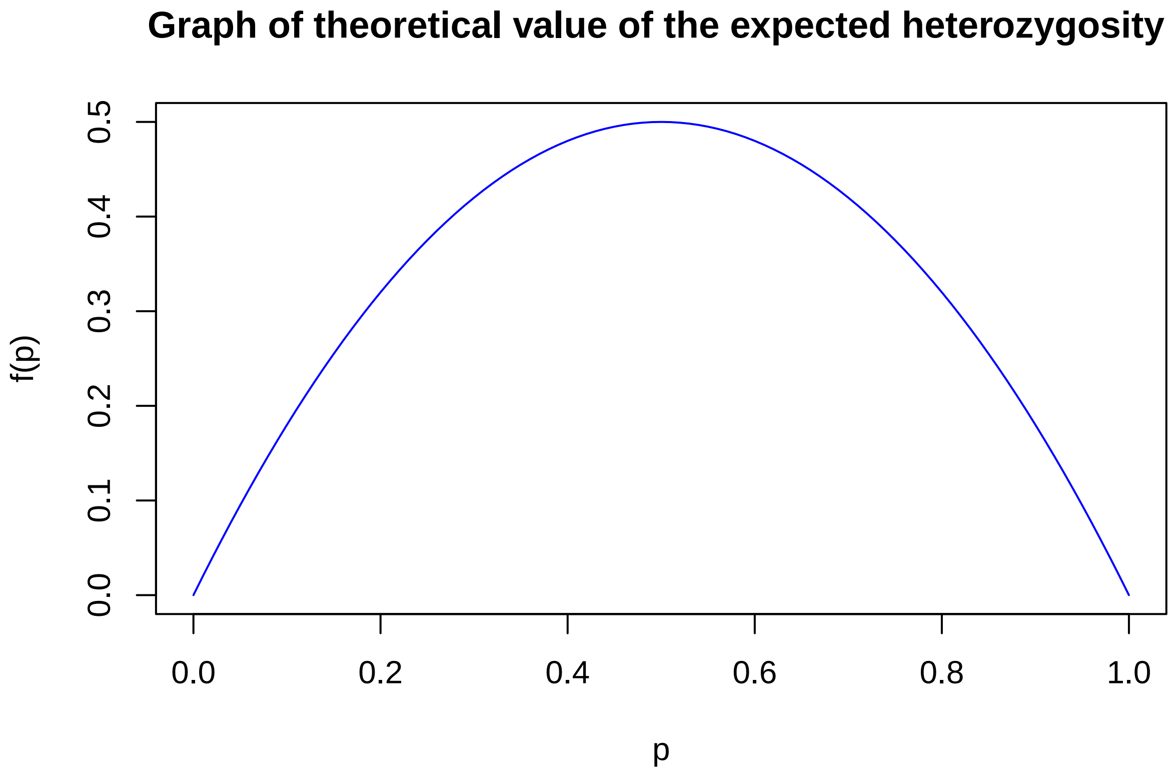
Question 6

Compute for each marker its expected heterozygosity (H_e), where the expected heterozygosity for a bi-allelic marker is defined as $1 - \sum_{i=1}^{k} 1 - p_i^2$, where p_i^2 is the frequency of the i^{th} allele. Make a histogram of the expected heterozygosity. What is, theoretically, the range of variation of this statistic? What is the average of H_e for this database?

```r
compute.all.He <- function(data){
  all.He <- c()
  for (col_name in colnames(data)) {
  col <- data[[col_name]]

  count.A <- sum(col == 1) + 2*sum(col == 0)
  count.B <- 204 - count.A
  tot.nb.alleles <- 2*nrow(data)

  pA <- count.A/tot.nb.alleles
  pB <- count.B/tot.nb.alleles

  He <- 1-pA*pA-pB*pB

  all.He <- c(all.He, He)
}
  return(all.He)
}

He <- compute.all.He(data)

hist(He, main = "Histogram of expected heterozygosity", xlab = "expected heterozigosity")
```



```r
He.theo <- function(x) 1 - x^2 - (1 - x)^2

curve(He.theo, from = 0, to = 1, col = "blue", xlab = "p", ylab = "f(p)", main = "Graph of theoretical
```



Theoretically, the range of variation of the expected heterozygosity is between 0 (there is no minor allele) and 0.5 (both alleles have the same frequency).

```r
mean(He)
```

```
## [1] 0.3113688
```

The average of H_e for this database is 0.311.

STR dataset

```r
library(HardyWeinberg)
data("NistSTRs")
```

```r
str(NistSTRs)
```

```
## 'data.frame':    361 obs. of  58 variables:
##  $ CSF1PO-1  : int  11 10 10 11 11 11 10 10 10 10 ...
##  $ CSF1PO-2  : int  13 12 12 11 11 11 12 11 11 11 ...
##  $ D10S1248-1: int  13 15 15 13 13 14 13 13 16 14 ...
##  $ D10S1248-2: int  15 16 17 16 14 15 14 13 17 14 ...
##  $ D12S391-1 : num  17 19 17.3 18 18 17 18 17 18 16 ...
##  $ D12S391-2 : num  18 19 20 20 19 20 18 18 20 17 ...
##  $ D13S317-1 : int  13 11 11 11 11 11 11 11 11 11 ...
##  $ D13S317-2 : int  13 11 11 12 12 11 13 11 12 14 ...
##  $ D16S539-1 : int  8 12 11 11 11 9 8 8 11 10 ...
##  $ D16S539-2 : int  12 13 12 13 13 11 11 12 13 12 ...
##  $ D18S51-1  : num  13 15 14 13 14 16 13 14 14 15 ...
##  $ D18S51-2  : num  15 17 19 15 17 18 17 15 17 17 ...
##  $ D19S433-1 : num  13 14 13 12 12 13 13 12 13 13 ...
##  $ D19S433-2 : num  13 15 14 14 14 16.2 15 15 13 15 ...
##  $ D1S1656-1 : num  14 12 16 11 11 12 15 12 14 13 ...
##  $ D1S1656-2 : num  16.3 15 17.3 14 18.3 15.3 17.3 13 15 14 ...
##  $ D21S11-1  : num  28 30 31.2 31.2 28 28 30 28 28 29 ...
##  $ D21S11-2  : num  30 31 32.2 32.2 28 30 32.2 30 29 30 ...
##  $ D22S1045-1: int  14 15 11 16 15 11 14 16 15 16 ...
##  $ D22S1045-2: int  16 16 16 17 16 16 15 16 16 16 ...
##  $ D2S1338-1 : int  17 17 19 18 24 18 17 17 23 17 ...
##  $ D2S1338-2 : int  20 20 25 20 25 22 18 24 23 22 ...
##  $ D2S441-1  : num  14 11 10 11 11.3 11 11 13 10 10 ...
##  $ D2S441-2  : num  15 12 14 14 14 12 14 14 11 11.3 ...
##  $ D3S1358-1 : num  14 15 15 16 14 15 15 14 18 15 ...
##  $ D3S1358-2 : int  17 18 16 17 16 18 18 16 18 15 ...
##  $ D5S818-1  : int  10 11 9 11 12 11 9 11 11 12 ...
##  $ D5S818-2  : int  11 12 12 13 13 13 11 12 12 12 ...
##  $ D6S1043-1 : num  12 13 11 10 11 12 11 12 11 11 ...
##  $ D6S1043-2 : num  18 19 19 18 12 14 17 13 11 19 ...
##  $ D7S820-1  : num  10 11 8 10 10 9 9 8 8 10 ...
##  $ D7S820-2  : int  12 12 12 10 11 9 12 10 9 11 ...
##  $ D8S1179-1 : int  13 12 13 10 12 12 11 11 11 13 ...
##  $ D8S1179-2 : int  14 12 13 10 13 15 13 11 13 14 ...
##  $ F13A01-1  : num  5 5 5 6 5 6 6 5 7 5 ...
##  $ F13A01-2  : num  14 6 7 6 6 7 6 7 7 6 ...
##  $ F13B-1    : num  6 6 6 6 9 8 8 6 9 8 ...
##  $ F13B-2    : int  9 9 9 10 10 10 8 8 10 10 ...
##  $ FESFPS-1  : num  10 11 12 11 10 10 10 11 12 11 ...
##  $ FESFPS-2  : num  11 11 12 11 12 11 11 11 12 13 ...
##  $ FGA-1     : num  24 21 19 21 20 20 19 23 19 23 ...
##  $ FGA-2     : num  24 21 21 21 24 26 23 24 21 26 ...
##  $ LPL-1     : int  10 9 10 11 10 10 12 12 10 10 ...
##  $ LPL-2     : int  12 10 11 11 11 11 12 13 12 11 ...
##  $ Penta_C-1 : num  11 13 11 11 11 10 9 9 11 10 ...
##  $ Penta_C-2 : int  12 13 12 13 12 13 11 12 12 13 ...
##  $ Penta_D-1 : num  10 10 9 9 9 12 9 10 12 10 ...
##  $ Penta_D-2 : num  11 11 11 12 15 13 12 12 13 12 ...
##  $ Penta_E-1 : num  7 12 7 12 7 7 7 14 7 13 ...
##  $ Penta_E-2 : num  12 15 12 12 19 7 13 16 12 14 ...
##  $ SE33-1    : num  13 19 22.2 25.2 14 15 19 17 16 16 ...
##  $ SE33-2    : num  14 22.2 28.2 30.2 16 19 28.2 19 29.2 25.2 ...
##  $ TH01-1    : num  6 6 7 6 8 6 9.3 7 6 8 ...
##  $ TH01-2    : num  9.3 7 9.3 9.3 9.3 8 9.3 9.3 7 9.3 ...
##  $ TPOX-1    : int  8 8 9 8 8 8 8 8 8 8 ...
##  $ TPOX-2    : int  8 8 11 8 8 11 11 11 11 11 ...
##  $ vWA-1     : int  17 17 16 16 16 15 16 17 18 15 ...
##  $ vWA-2     : int  17 18 18 17 19 17 19 18 19 19 ...
```

Question 1

How many individuals and how many STRs contains the database?

```
dimensions <- dim(NistSTRs)

num_individuals <- dimensions[1]

#two alleles of an individual each STR
num_strs <- dimensions[2] / 2

cat("Number of individuals:", num_individuals, "\n")
```

```
## Number of individuals: 361
```

```
cat("Number of STRs:", num_strs, "\n")
```

```
## Number of STRs: 29
```

Question 2

Write a function that determines the number of alleles for a STR. Determine the number of alleles for each STR in the database. Compute basic descriptive statistics of the number of alleles (mean, standard deviation, median, minimum, maximum).

```
all_columns <- names(NistSTRs)

odd_columns <- all_columns[c(TRUE, FALSE)]
even_columns <- all_columns[c(FALSE, TRUE)]

first <- NistSTRs %>% select(all_of(odd_columns)) %>%
                    rename_with(~str_sub(., end = -3), everything())

second <- NistSTRs %>% select(all_of(even_columns)) %>%
                    rename_with(~str_sub(., end = -3), everything())

combined_str <- bind_rows(list(first, second))

allele_counts <- combined_str %>%
  summarise_all(n_distinct)


cat("Number of alleles for every STR:\n")
```

```
## Number of alleles for every STR:
```

```
print(allele_counts)
```

```
##   CSF1PO D10S1248 D12S391 D13S317 D16S539 D18S51 D19S433 D1S1656 D21S11
## 1      7        9      16       8       7     15      15      15     16
##   D22S1045 D2S1338 D2S441 D3S1358 D5S818 D6S1043 D7S820 D8S1179 F13A01 F13B
## 1        8      12     11       9      9      14      9      10     12    6
##   FESFPS FGA LPL Penta_C Penta_D Penta_E SE33 TH01 TPOX vWA
## 1      7  14   8      10      13      19   39    8    8  10
```

```
num_alleles <- allele_counts %>%
                slice(1) %>%
                unlist()

mean_num_alleles <- mean(num_alleles)
sd_num_alleles <- sd(num_alleles)
```

```
median_num_alleles <- median(num_alleles)
min_num_alleles <- min(num_alleles)
max_num_alleles <- max(num_alleles)

cat("Descriptive Statistics of the Number of Alleles:\n")
```

```
## Descriptive Statistics of the Number of Alleles:
```

```
cat("Mean:", mean_num_alleles, "\n")
```

```
## Mean: 11.86207
```

```
cat("Standard Deviation:", sd_num_alleles, "\n")
```

```
## Standard Deviation: 6.226236
```

```
cat("Median:", median_num_alleles, "\n")
```

```
## Median: 10
```

```
cat("Minimum:", min_num_alleles, "\n")
```

```
## Minimum: 6
```

```
cat("Maximum:", max_num_alleles, "\n")
```

```
## Maximum: 39
```

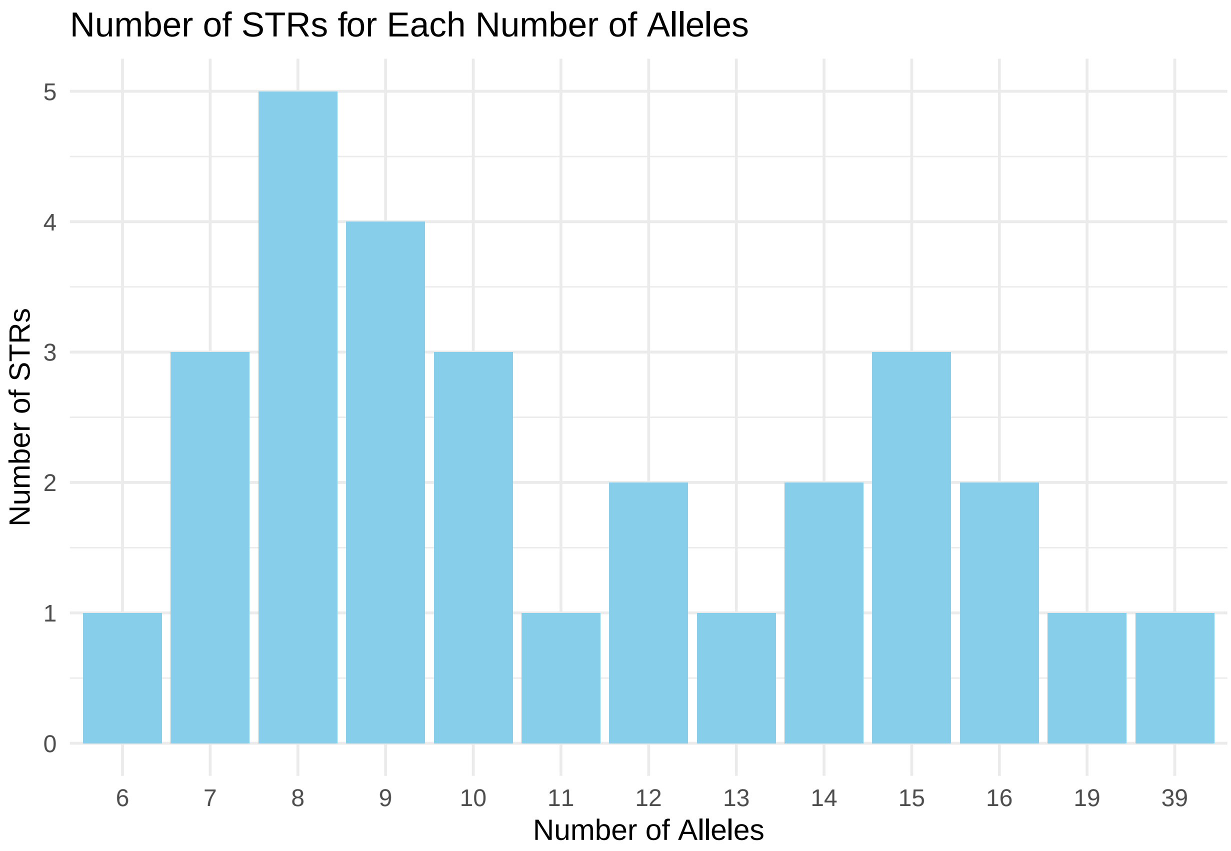
Question 3

Make a table with the number of STRs for a given number of alleles and present a barplot of the number STRs in each category. What is the most common number of alleles for an STR?

```
allele_counts <- allele_counts %>%
  pivot_longer(everything(), names_to = "STR", values_to = "NumAlleles")

# Table with the count of STRs for each number of alleles
allele_count_table <- allele_counts %>%
  group_by(NumAlleles) %>%
  summarise(Count = n())

# Print the table
print(allele_count_table)
```

```
## # A tibble: 13 x 2
##    NumAlleles Count
##         <int> <int>
##  1          6     1
##  2          7     3
##  3          8     5
##  4          9     4
##  5         10     3
##  6         11     1
##  7         12     2
##  8         13     1
##  9         14     2
## 10         15     3
## 11         16     2
## 12          19     1
## 13          39     1
```

```r
# Barplot with ggplot2
ggplot(allele_count_table, aes(x = factor(NumAlleles), y = Count)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(x = "Number of Alleles", y = "Number of STRs",
       title = "Number of STRs for Each Number of Alleles") +
  theme_minimal()
```



```r
# Find the most common number of alleles
most_common_alleles <- allele_count_table %>%
  filter(Count == max(Count))

# Print the most common number of alleles
cat("Most common number of alleles for an STR:", most_common_alleles$NumAlleles, "\n")
```

```
## Most common number of alleles for an STR: 8
```

Question 4

Compute the expected heterozygosity for each STR. Make a histogram of the expected heterozygosity over all STRS. Compute the average expected heterozygosity over all STRs.

```r
sum_str <- NistSTRs %>%
  summarise(across(everything(), sum))
```