Project 1 allele frequency analysis

Ovarian cancer allele frequency analysis

First SNP:

```
# Read the CSV file
data_rs1799966 <- read_csv("1000genomesprojectphase3-PopulationGenotypes-Homo_sapiens_Variation_Populat
```

```
## New names:
## Rows: 31 Columns: 4
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (2): Population, Allele: frequency (count) lgl (2): ...3, Genotypes
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...3`
```

```
# View the first few rows of the data
head(data_rs1799966)
```

```
## # A tibble: 6 x 4
##   Population `Allele: frequency (count)`    ...3  Genotypes
##   <chr>      <chr>                          <lgl> <lgl>
## 1 AFR        T: 0.775 (1024) C: 0.225 (298) NA    NA
## 2 ACB        T: 0.745 (143) C: 0.255 (49)   NA    NA
## 3 ASW        T: 0.754 (92) C: 0.246 (30)    NA    NA
## 4 ESN        T: 0.828 (164) C: 0.172 (34)   NA    NA
## 5 GWD        T: 0.743 (168) C: 0.257 (58)   NA    NA
## 6 LWK        T: 0.773 (153) C: 0.227 (45)   NA    NA
```

Now we clean the data to be able to analyze and plot it

```
# Split the "Allele: frequency (count)" column into separate columns
data_clean_rs1799966 <- data_rs1799966 %>%
  separate(`Allele: frequency (count)`,
           into = c("T_allele", "C_allele"),
           sep = "C:") %>%  # Split at "C:" to separate T and C alleles
  mutate(
    T_allele = str_trim(T_allele),  # Remove extra spaces
    C_allele = str_trim(C_allele)   # Remove extra spaces
  ) %>%
  select(-`...3`, -Genotypes)  # Remove unnecessary columns
# Extract T allele frequency and count
data_clean_rs1799966 <- data_clean_rs1799966 %>%
  mutate(
```

```
    T_freq = as.numeric(str_extract(T_allele, "\\d+\\.\\d+")),  # Extract frequency
    T_count = as.numeric(str_extract(T_allele, "\\d+(?=\\))")), # Extract count
    C_freq = as.numeric(str_extract(C_allele, "\\d+\\.\\d+")),  # Extract frequency
    C_count = as.numeric(str_extract(C_allele, "\\d+(?=\\))"))  # Extract count
  )
#We only need allele frequencies so lets simplify
allele_freq_rs1799966 <- data_clean_rs1799966 %>%
  select(Population, T_freq, C_freq)
# View the simplified data
head(allele_freq_rs1799966)
```

```
## # A tibble: 6 x 3
##   Population T_freq C_freq
##   <chr>       <dbl>  <dbl>
## 1 AFR         0.775  0.225
## 2 ACB         0.745  0.255
## 3 ASW         0.754  0.246
## 4 ESN         0.828  0.172
## 5 GWD         0.743  0.257
## 6 LWK         0.773  0.227
```

```
#we can also take the Population column to replace it for the next allele frequencies
populations <- allele_freq_rs1799966$Population
rm(data_clean_rs1799966)
rm(data_rs1799966)
```

Now we input the other two alleles that we need using the same code but creating different objects

```
# Read the CSV file
data_rs1045485 <- read_csv("1000genomesprojectphase3-PopulationGenotypes-Homo_sapiens_Variation_Populat
```

```
## New names:
## Rows: 31 Columns: 5
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (3): Population, Allele: frequency (count), Genotype: frequency (count) lgl
## (2): ...2, Genotypes
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...2`
```

```
# Replace the 'Population' column with the 'populations' object
data_rs1045485$Population <- populations
# Split the "Allele: frequency (count)" column into separate columns
data_clean_rs1045485 <- data_rs1045485 %>%
  separate(`Allele: frequency (count)`,
           into = c("G_allele", "C_allele"),
           sep = "C:") %>%  # Split at "C:" to separate G and C alleles
  mutate(
    G_allele = str_trim(G_allele),  # Remove extra spaces
    C_allele = str_trim(C_allele)   # Remove extra spaces
  )
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 4 rows [15, 16,
## 17, 18].
```

```
# Remove unnecessary columns if they exist
if (any(grepl("^...3$", colnames(data_clean_rs1045485)))) {
  data_clean_rs1045485 <- data_clean_rs1045485 %>% select(-`...3`)
}
if (any(grepl("Genotypes", colnames(data_clean_rs1045485)))) {
  data_clean_rs1045485 <- data_clean_rs1045485 %>% select(-Genotypes)
}

# Extract G and C allele frequencies and counts
data_clean_rs1045485 <- data_clean_rs1045485 %>%
  mutate(
    G_freq = as.numeric(str_extract(G_allele, "\\d+\\.\\d+")),  # Extract G allele frequency
    G_count = as.numeric(str_extract(G_allele, "\\d+(?=\\))")), # Extract G allele count
    C_freq = as.numeric(str_extract(C_allele, "\\d+\\.\\d+")),  # Extract C allele frequency
    C_count = as.numeric(str_extract(C_allele, "\\d+(?=\\))"))  # Extract C allele count
  )

# Select only the necessary columns (Population, G_freq, C_freq)
allele_freq_rs1045485 <- data_clean_rs1045485 %>%
  select(Population, G_freq, C_freq)

# View the final dataframe with allele frequencies
head(allele_freq_rs1045485)
```

```
## # A tibble: 6 x 3
##   Population G_freq C_freq
##   <chr>       <dbl>  <dbl>
## 1 AFR         0.95   0.05
## 2 ACB         0.927  0.073
## 3 ASW         0.926  0.074
## 4 ESN         0.975  0.025
## 5 GWD         0.96   0.04
## 6 LWK         0.924  0.076
```

```
rm(data_rs1045485)
rm(data_clean_rs1045485)
```

```
# Read the CSV file
data_rs144848 <- read_csv("1000genomesprojectphase3-PopulationGenotypes-Homo_sapiens_Variation_Populati
```

```
## New names:
## * `` -> `...2`
```

```
data_rs144848$Population <- populations
# Split the "Allele: frequency (count)" column into separate columns
data_clean_rs144848 <- data_rs144848 %>%
  separate(`Allele: frequency (count)`,
           into = c("A_allele", "C_allele"),
           sep = "C:") %>%  # Split at "C:" to separate A and C alleles
```

```
  mutate(
    A_allele = str_trim(A_allele),  # Remove extra spaces
    C_allele = str_trim(C_allele)   # Remove extra spaces
  )

# Remove unnecessary columns if they exist
if (any(grepl("^...3$", colnames(data_clean_rs144848)))) {
  data_clean_rs144848 <- data_clean_rs144848 %>% select(-`...3`)
}
if (any(grepl("Genotypes", colnames(data_clean_rs144848)))) {
  data_clean_rs144848 <- data_clean_rs144848 %>% select(-Genotypes)
}

# Extract A and C allele frequencies and counts
data_clean_rs144848 <- data_clean_rs144848 %>%
  mutate(
    A_freq = as.numeric(str_extract(A_allele, "\\d+\\.\\d+")),  # Extract A allele frequency
    A_count = as.numeric(str_extract(A_allele, "\\d+(?=\\))")), # Extract A allele count
    C_freq = as.numeric(str_extract(C_allele, "\\d+\\.\\d+")),  # Extract C allele frequency
    C_count = as.numeric(str_extract(C_allele, "\\d+(?=\\))"))  # Extract C allele count
  )

# Select only the necessary columns (Population, A_freq, C_freq)
allele_freq_rs144848 <- data_clean_rs144848 %>%
  select(Population, A_freq, C_freq)

# View the final dataframe with allele frequencies
head(allele_freq_rs144848)
```

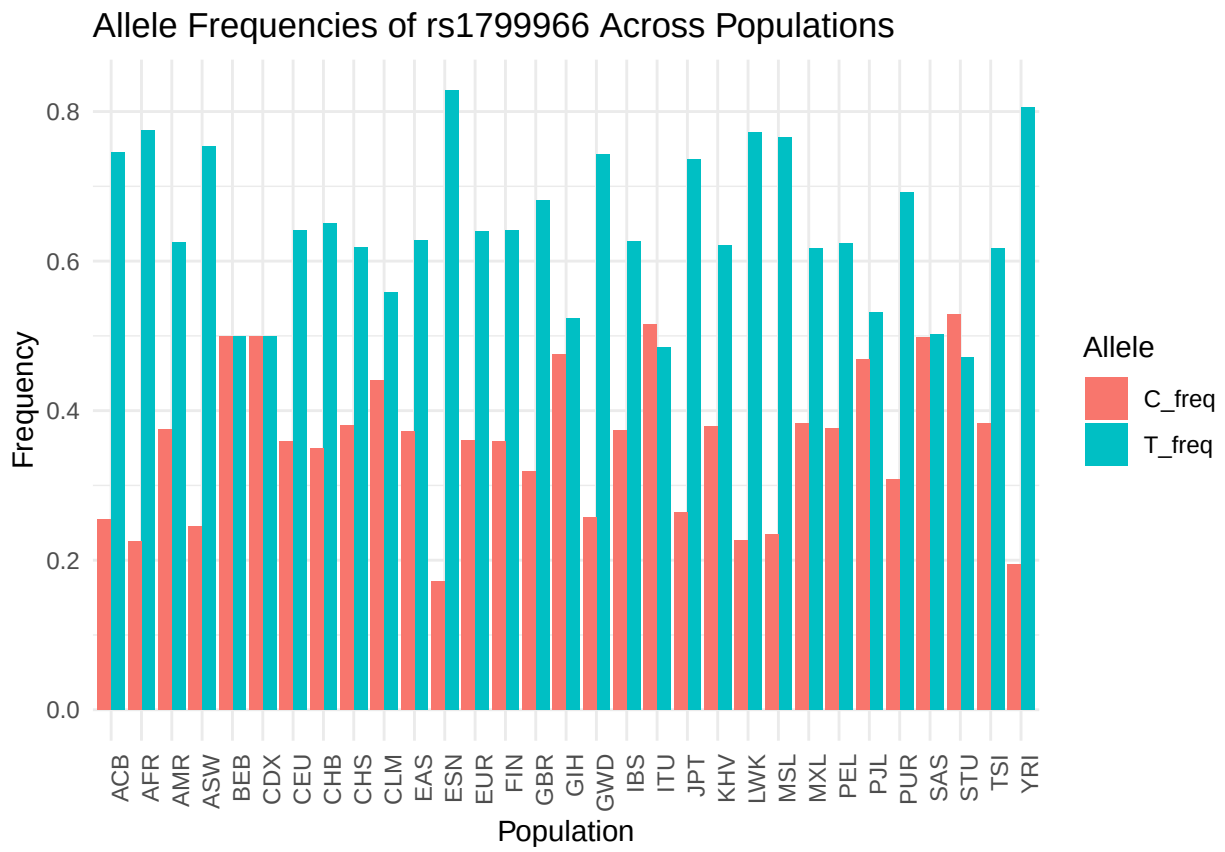
```
## # A tibble: 6 x 3
##   Population A_freq C_freq
##   <chr>       <dbl>  <dbl>
## 1 AFR         0.916  0.084
## 2 ACB         0.865  0.135
## 3 ASW         0.902  0.098
## 4 ESN         0.934  0.066
## 5 GWD         0.956  0.044
## 6 LWK         0.939  0.061
```

```
rm(data_rs144848)
rm(data_clean_rs144848)
```

##visualize the data We can start with a simple barplot to see the allele frequency differences between populations

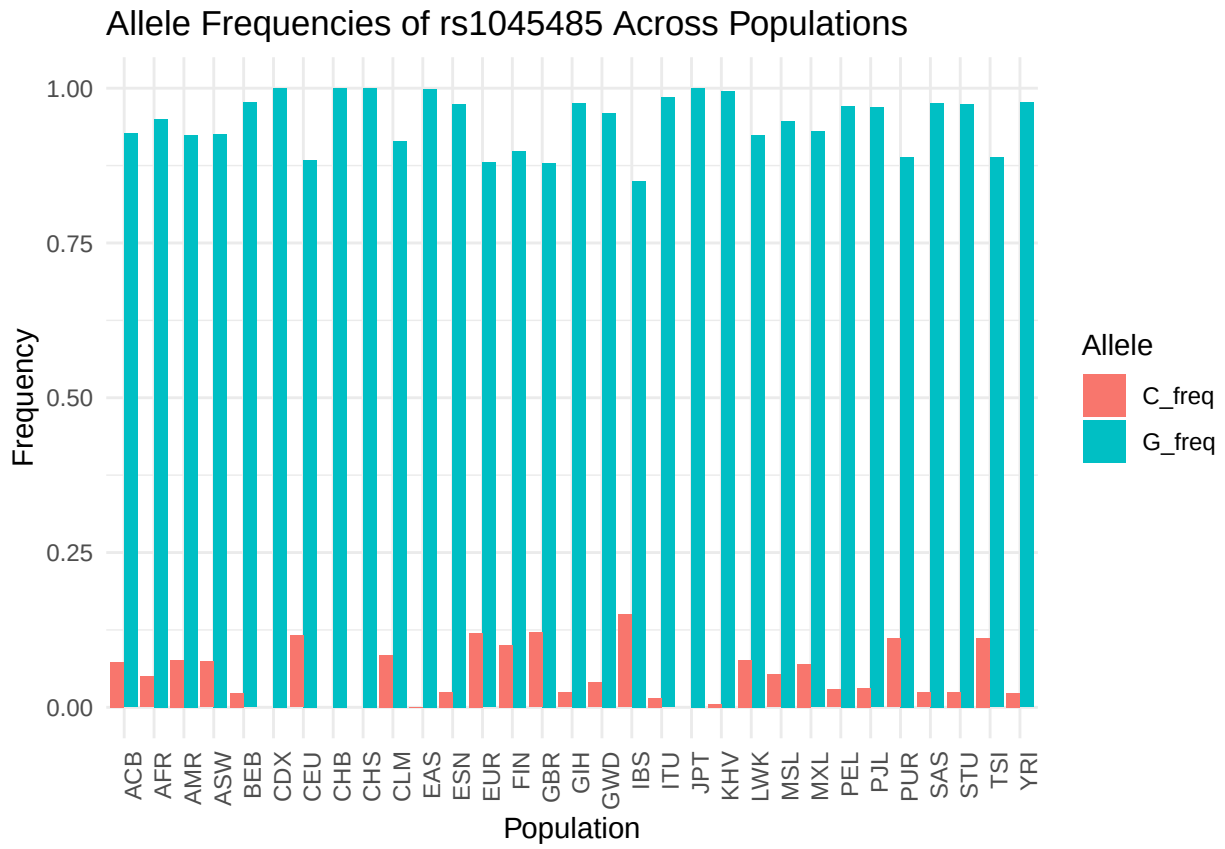
```
# Reshape the data for plotting
data_rs1799966 <- allele_freq_rs1799966 %>%
  pivot_longer(cols = c(T_freq, C_freq),
               names_to = "Allele",
               values_to = "Frequency")

# Create a bar plot
```

```r
ggplot(data_rs1799966, aes(x = Population, y = Frequency, fill = Allele)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Allele Frequencies of rs1799966 Across Populations",
       x = "Population",
       y = "Frequency",
       fill = "Allele") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

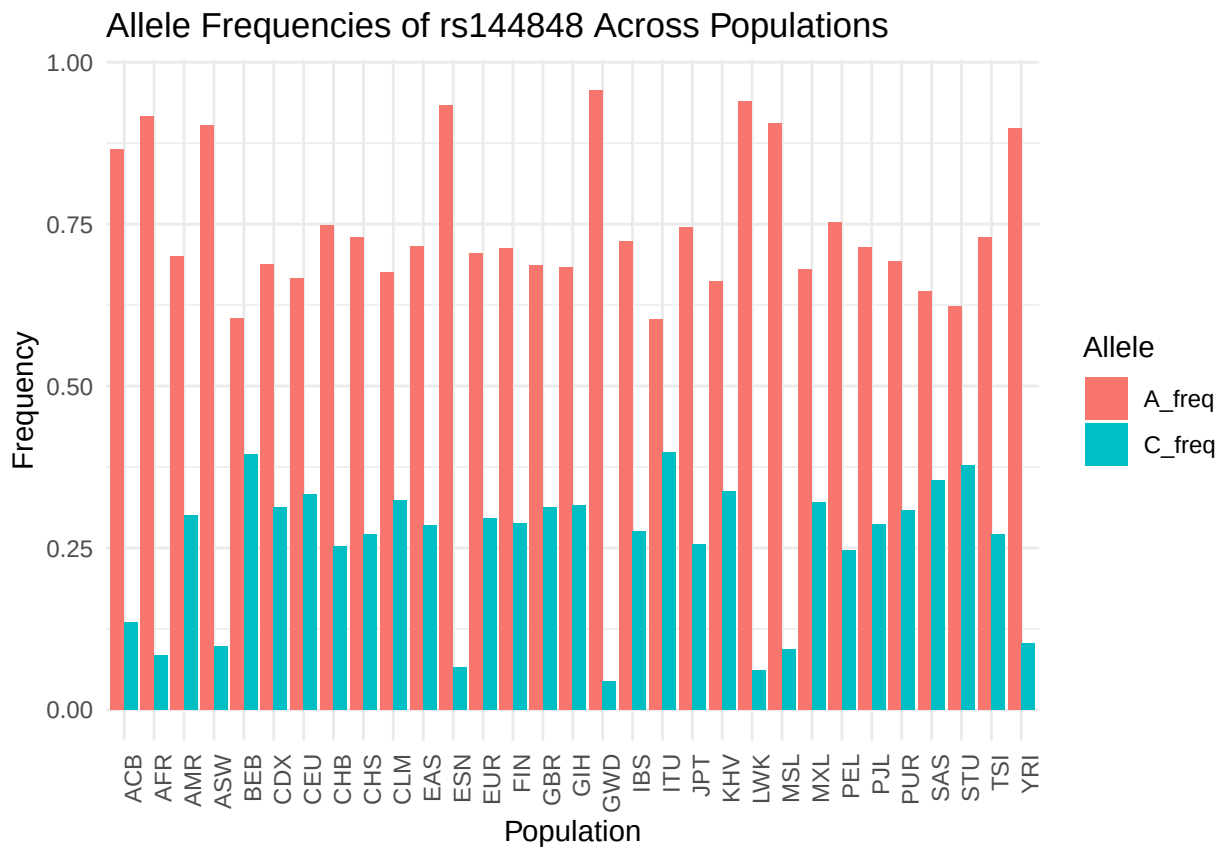



```r
# Reshape the data for plotting
data_rs1045485 <- allele_freq_rs1045485 %>%
  pivot_longer(cols = c(G_freq, C_freq),
               names_to = "Allele",
               values_to = "Frequency")

# Create a bar plot
ggplot(data_rs1045485, aes(x = Population, y = Frequency, fill = Allele)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Allele Frequencies of rs1045485 Across Populations",
       x = "Population",
       y = "Frequency",
       fill = "Allele") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
## Warning: Removed 4 rows containing missing values or values outside the scale range
## (`geom_bar()`).
```

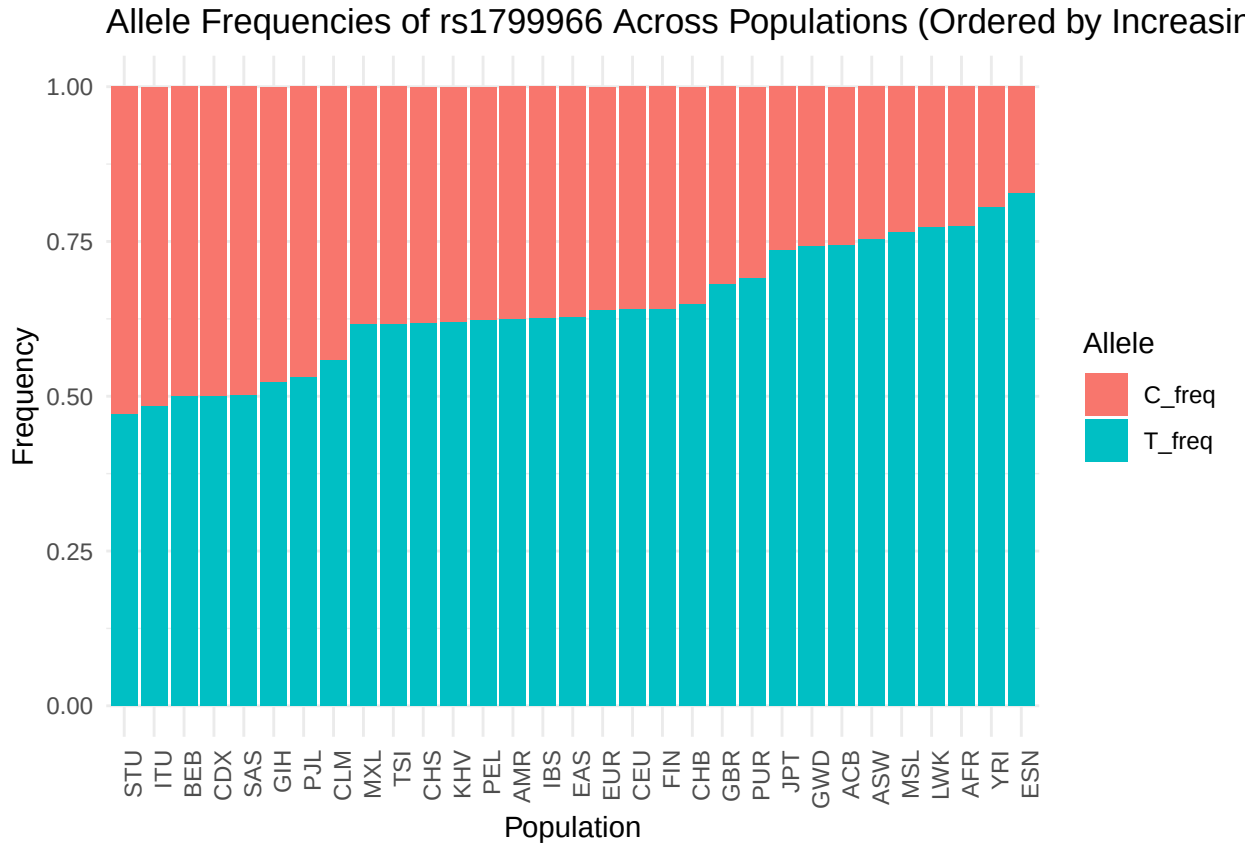



```r
# Reshape the data for plotting
data_rs144848 <- allele_freq_rs144848 %>%
  pivot_longer(cols = c(A_freq, C_freq),
               names_to = "Allele",
               values_to = "Frequency")

# Create a bar plot
ggplot(data_rs144848, aes(x = Population, y = Frequency, fill = Allele)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Allele Frequencies of rs144848 Across Populations",
       x = "Population",
       y = "Frequency",
       fill = "Allele") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



we can make a better plot by using a different approach that sorts the populations by allele frequency differences

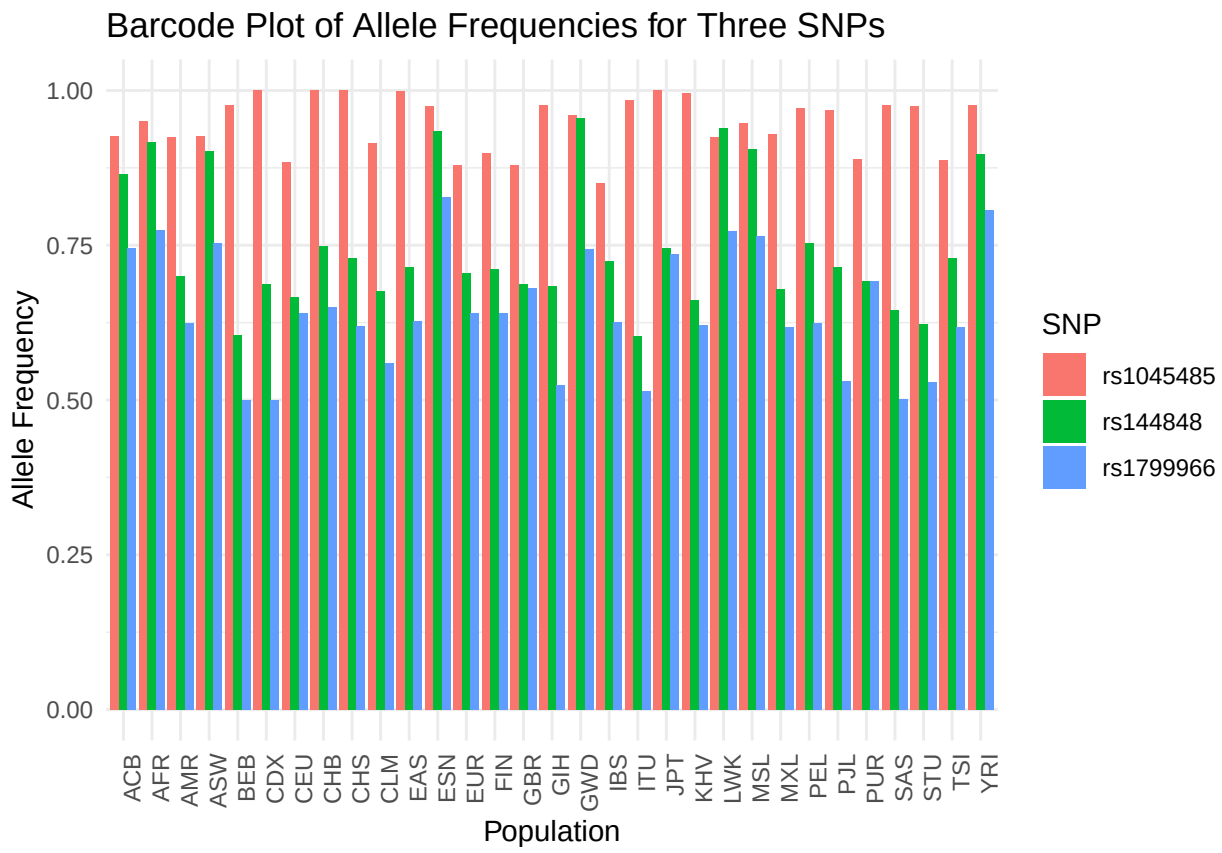
```
# Calculate the difference in allele frequencies
allele_freq_rs1799966 <- allele_freq_rs1799966 %>%
  mutate(Difference = T_freq - C_freq)
# Sort populations by increasing difference
allele_freq <- allele_freq_rs1799966 %>%
  arrange(Difference)
# Reshape the data for plotting
allele_freq_long <- allele_freq %>%
  pivot_longer(cols = c(T_freq, C_freq),
               names_to = "Allele",
               values_to = "Frequency")
#plot
ggplot(allele_freq_long, aes(x = reorder(Population, Difference), y = Frequency, fill = Allele)) +
  geom_bar(stat = "identity", position = "stack") +
  labs(title = "Allele Frequencies of rs1799966 Across Populations (Ordered by Increasing Difference)",
       x = "Population",
       y = "Frequency",
       fill = "Allele") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Now we compare the frequencies of all three snps

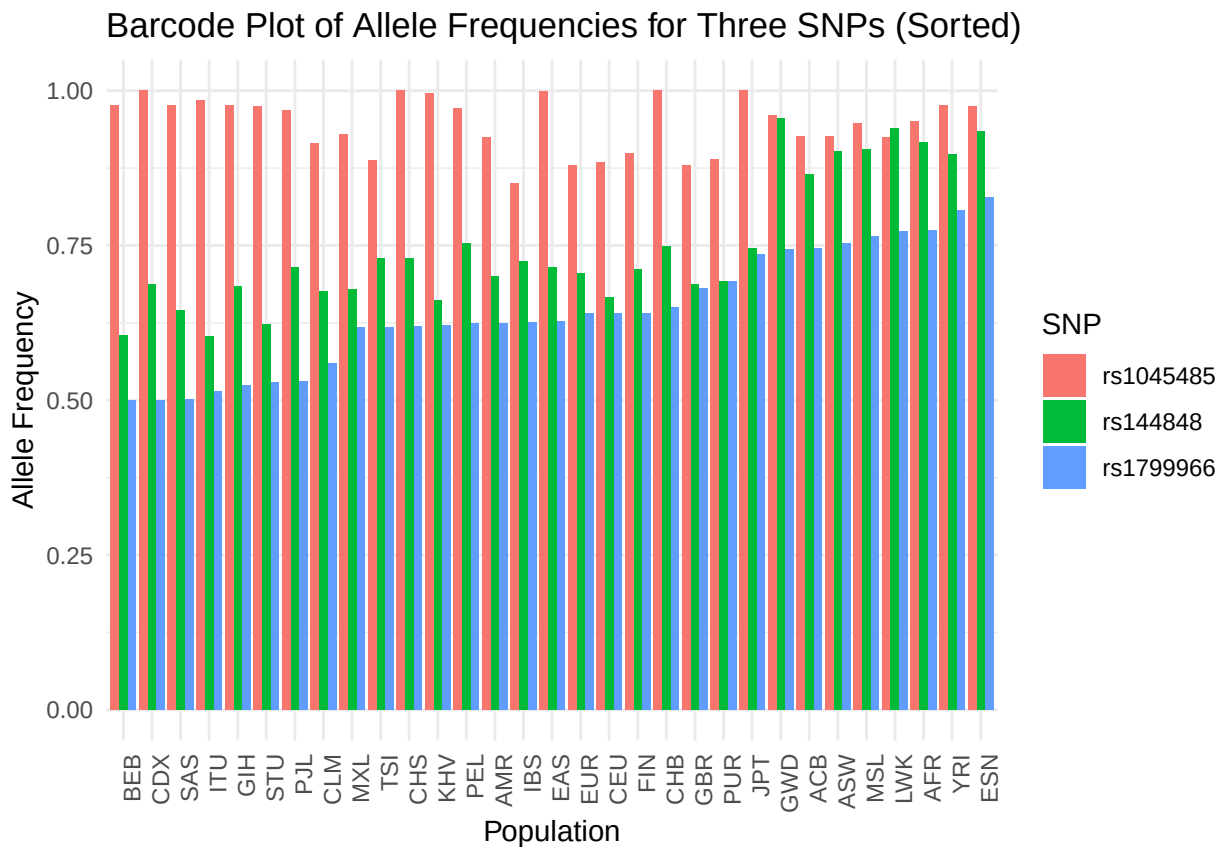
```
# Combine the three dataframes into one
combined_data <- bind_rows(
  allele_freq_rs1045485 %>% mutate(SNP = "rs1045485"),
  allele_freq_rs144848 %>% mutate(SNP = "rs144848"),
  allele_freq_rs1799966 %>% mutate(SNP = "rs1799966")
)

# Select the allele with the highest frequency for each SNP and population
combined_data <- combined_data %>%
  pivot_longer(cols = c(G_freq, C_freq, A_freq, T_freq), names_to = "Allele", values_to = "Frequency") %
  filter(!is.na(Frequency)) %>%  # Remove rows with NA frequencies
  group_by(Population, SNP) %>%
  slice_max(Frequency) %>%  # Select the allele with the highest frequency
  ungroup()
# Plot the barcode of allele frequencies
ggplot(combined_data, aes(x = Population, y = Frequency, fill = SNP)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Barcode Plot of Allele Frequencies for Three SNPs",
    x = "Population",
    y = "Allele Frequency",
    fill = "SNP"
  ) +
  theme_minimal() +
```

```
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Rotate x-axis labels for readability
```



Now we sort them by allele frequency in asending order

```
# Sort the populations based on the smallest to highest allele frequency
combined_data <- combined_data %>%
  arrange(Frequency) %>%  # Sort by frequency in ascending order
  mutate(Population = factor(Population, levels = unique(Population)))  # Reorder factor levels
# Plot the barcode of allele frequencies with sorted bars
ggplot(combined_data, aes(x = Population, y = Frequency, fill = SNP)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Barcode Plot of Allele Frequencies for Three SNPs (Sorted)",
    x = "Population",
    y = "Allele Frequency",
    fill = "SNP"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  # Rotate x-axis labels for readability
```



Now we plot the first snp allele in a world map

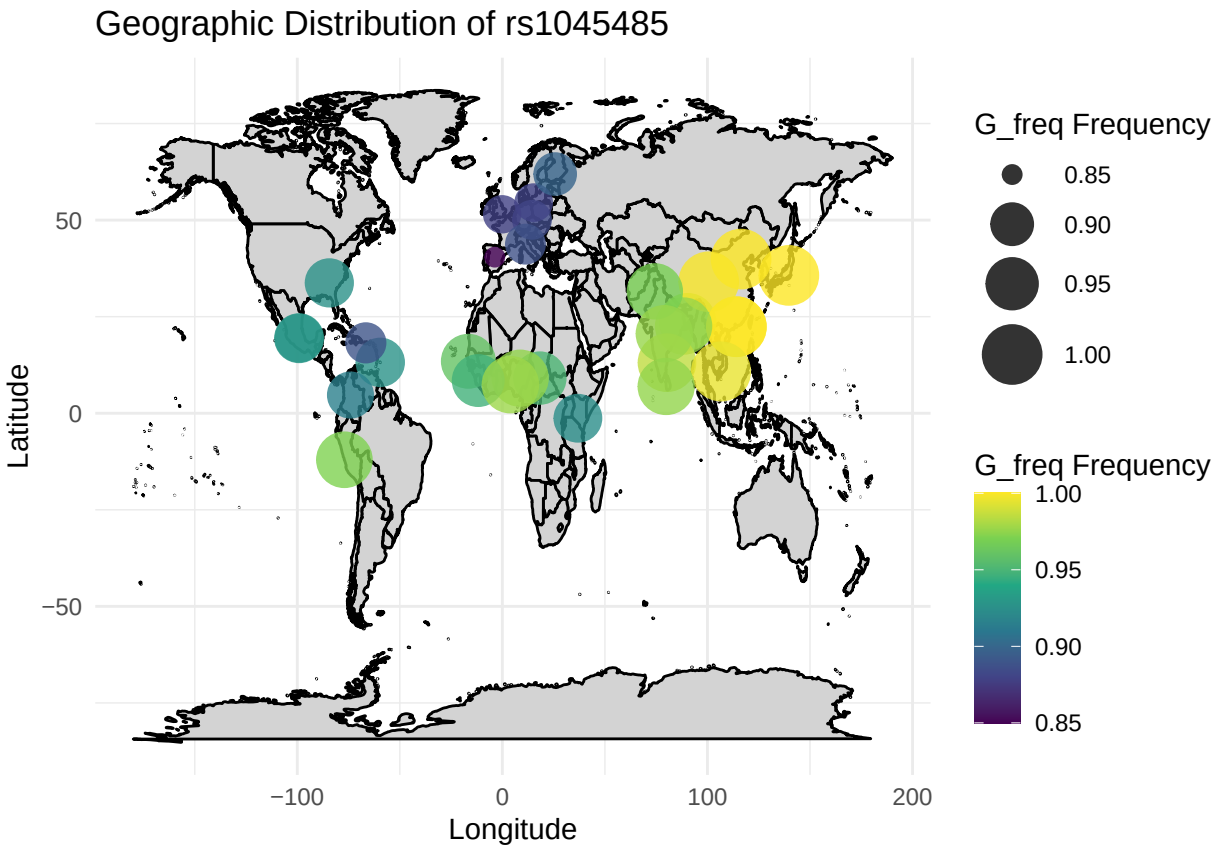
```
#The first step is to give each population the coordinates for it to map into the world
population_coords <- data.frame(
  Population = c("ALL", "AFR", "ACB", "ASW", "ESN", "GWD", "LWK", "MSL", "YRI",
                 "AMR", "CLM", "MXL", "PEL", "PUR", "EAS", "CDX", "CHB", "CHS",
                 "JPT", "KHV", "EUR", "CEU", "FIN", "GBR", "IBS", "TSI", "SAS",
                 "BEB", "GIH", "ITU", "PJL", "STU"),
  Latitude = c(0, 9.1021, 13.1772, 33.7490, 9.0820, 13.4549, -1.2921, 8.4606, 7.3964,
               19.4326, 4.7110, 19.4326, -12.0464, 18.2208, 34.0479, 22.3964, 39.9042,
               22.3964, 35.6895, 10.8231, 54.5260, 50.0755, 61.9241, 51.5074, 40.4168,
               43.7696, 20.5937, 23.6850, 22.5726, 13.0827, 31.5546, 6.9271),
  Longitude = c(0, 18.2812, -59.5432, -84.3880, 8.6753, -16.5790, 36.8219, -11.7799,
                3.9167, -99.1332, -74.0721, -99.1332, -77.0428, -66.5901, 100.6197,
                114.1095, 116.4074, 114.1095, 139.6917, 106.6297, 15.2551, 14.4378,
                25.7482, -0.1278, -3.7038, 11.2558, 78.9629, 90.3563, 88.3639, 80.2707,
                74.3572, 79.8612)
)
#we include this in our dataset
allele_freq_map <- merge(allele_freq_rs1799966, population_coords, by = "Population")

# Get world map data
world_map <- map_data("world")
# Plot the world map with allele frequencies
map_rs1799966 <- ggplot() +
  geom_polygon(data = world_map, aes(x = long, y = lat, group = group),
               fill = "lightgray", color = "black") +
```

```
  geom_point(data = allele_freq_map,
             aes(x = Longitude, y = Latitude, size = T_freq, color = T_freq),
             alpha = 0.8) +
  scale_size_continuous(range = c(3, 10)) +
  scale_color_viridis_c() +
  labs(title = "Geographic Distribution of T Allele Frequencies for rs1799966",
       x = "Longitude",
       y = "Latitude",
       size = "T Allele Frequency",
       color = "T Allele Frequency") +
  theme_minimal()
```

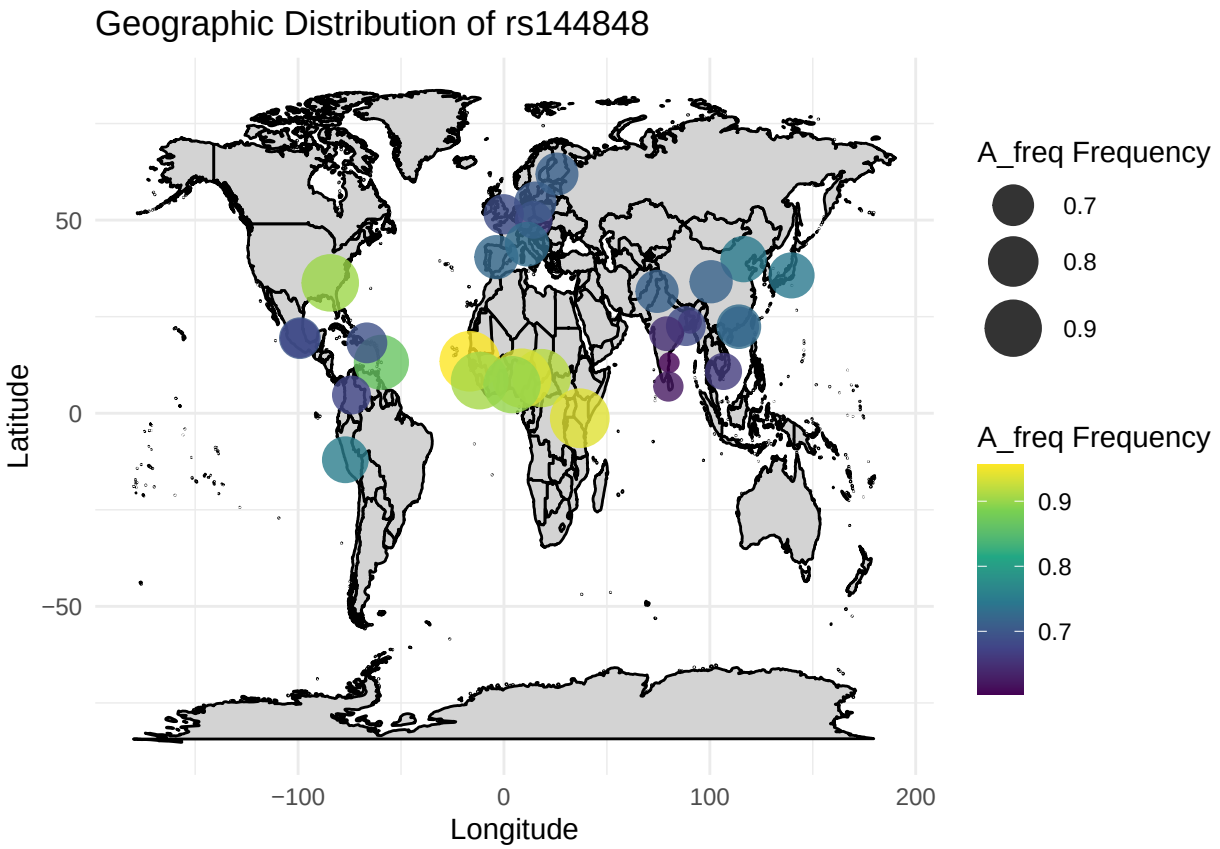
Now we plot the other two and try to plot them together

```
# Function to create a map for a given SNP
create_map <- function(allele_freq, snp_name, allele_col) {
  # Merge allele frequencies with population coordinates
  allele_freq_map <- merge(allele_freq, population_coords, by = "Population")

  # Get world map data
  world_map <- map_data("world")

  # Plot the world map with allele frequencies
  ggplot() +
    geom_polygon(data = world_map, aes(x = long, y = lat, group = group),
                 fill = "lightgray", color = "black") +
    geom_point(data = allele_freq_map,
               aes(x = Longitude, y = Latitude, size = !!sym(allele_col), color = !!sym(allele_col)),
               alpha = 0.8) +
    scale_size_continuous(range = c(3, 10)) +
    scale_color_viridis_c() +
    labs(title = paste("Geographic Distribution of", snp_name),
         x = "Longitude",
         y = "Latitude",
         size = paste(allele_col, "Frequency"),
         color = paste(allele_col, "Frequency")) +
    theme_minimal()
}

# Create maps for each SNP
map_rs1045485 <- create_map(allele_freq_rs1045485, "rs1045485", "G_freq")
map_rs144848 <- create_map(allele_freq_rs144848, "rs144848", "A_freq")

# Combine the maps into a 1x3 grid using patchwork
combined_maps <- (map_rs1045485 | map_rs144848) +
  plot_layout(nrow = 1, ncol = 2)  # Arrange in 1 row and 3 columns
```

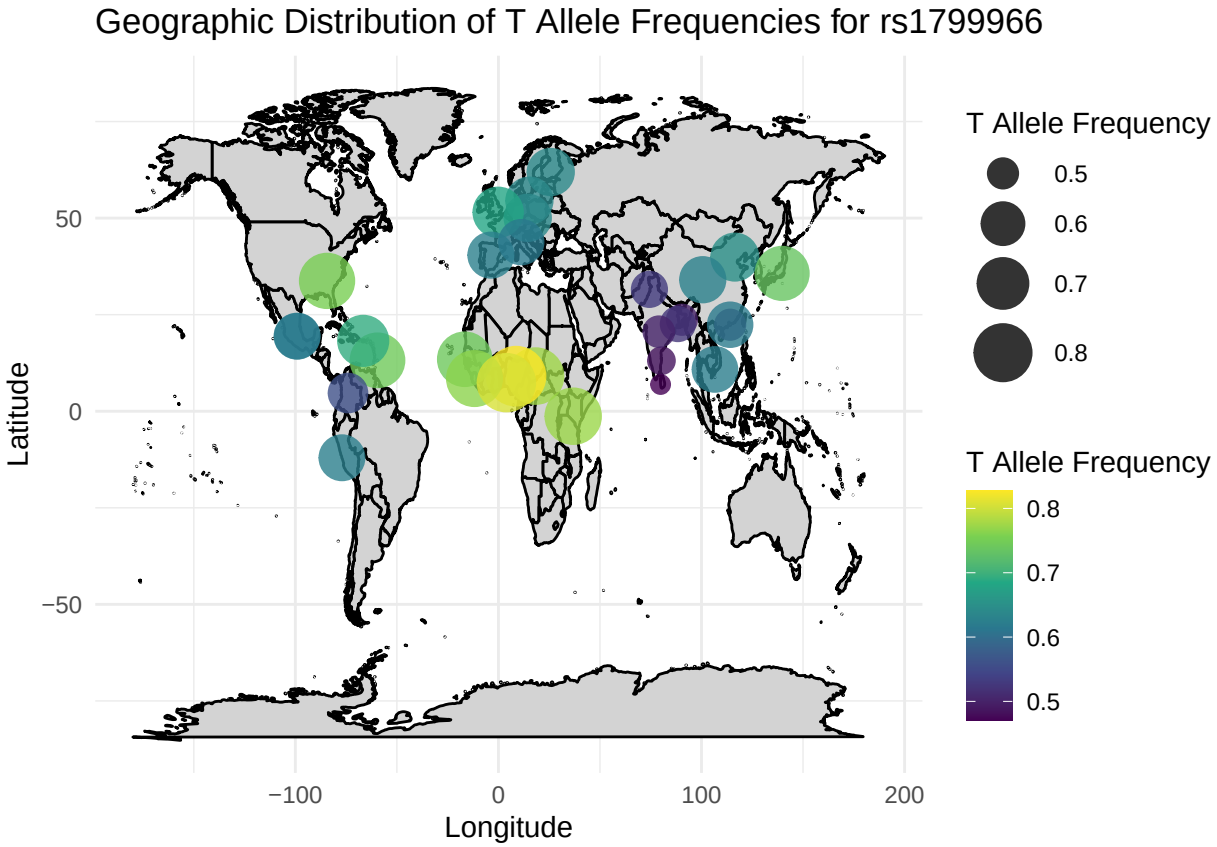
```
#
map_rs1045485
```




```
map_rs144848
```




```
map_rs1799966
```


Geographic Distribution of T Allele Frequencies for rs1799966
Latitude
50
0
−50
−100
0
100
200
Longitude
T Allele Frequency
0.5
0.6
0.7
0.8
T Allele Frequency
0.8
0.7
0.6
0.5