R Assignment Script

Kofi Antwi Appiagyei

2025-03-21

Below is my workflow for the R Assignment Load libraries

```
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.4.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.4.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.4     v tibble    3.2.1
## v purrr     1.0.4     v tidyr     1.3.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(tidyr)
library(purrr)
```

DATA INSPECTION

Read the files

```
genotypes <- read.table("fang_et_al_genotypes.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)
snp_position <- read.table("snp_position.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)
```

Inspecting fang_et_al_genotypes.txt 1. File size (in bytes)

```
file_size <- file.info("fang_et_al_genotypes.txt")$size
print(paste("File size (bytes):", file_size))
```

```
## [1] "File size (bytes): 11051939"
```

2. View the first 6 rows (head) of the data

```
head(genotypes[, 1:8])
```

```
##   Sample_ID   JG_OTU Group abph1.20 abph1.22 ae1.3 ae1.4 ae1.5
## 1     SL-15 T-aust-1 TRIPS      ?/?      ?/?   T/T   G/G   T/T
## 2     SL-16 T-aust-2 TRIPS      ?/?      ?/?   T/T   ?/?   T/T
## 3     SL-11 T-brav-1 TRIPS      ?/?      ?/?   T/T   G/G   T/T
## 4     SL-12 T-brav-2 TRIPS      ?/?      ?/?   T/T   G/G   T/T
## 5     SL-18   T-cund TRIPS      ?/?      ?/?   T/T   G/G   T/T
## 6      SL-2 T-dact-1 TRIPS      ?/?      ?/?   T/T   G/G   T/T
```

3. View the last 6 rows (tail) of the data

```
tail(genotypes[, 1:8])
```

```
##      Sample_ID             JG_OTU Group abph1.20 abph1.22 ae1.3 ae1.4 ae1.5
## 2777    SYN262 Zmm-IL-W22-Rrstd_v ZMMIL      C/C      A/A   T/T   G/G   C/C
## 2778     S0398      Zmm-IL-W64A_a ZMMIL      G/G      A/A   T/T   G/G   C/C
## 2779     S1636    Zmm-IL-W64A_b_s ZMMIL      G/G      A/A   T/T   G/G   C/C
## 2780    CU0201        Zmm-IL-WD_f ZMMIL      C/C      A/A   T/T   G/G   C/C
## 2781     S0215         Zmm-IL-Wf9 ZMMIL      G/G      A/A   T/T   ?/?   C/C
## 2782    CU0202  Zmm-IL-Yu796_NS_f ZMMIL      C/C      A/A   T/T   G/G   C/C
```

4. Number of rows and columns in the data

```
num_rows <- nrow(genotypes)
num_cols <- ncol(genotypes)
print(paste("Number of rows:", num_rows))
```

```
## [1] "Number of rows: 2782"
```

```
print(paste("Number of columns:", num_cols))
```

```
## [1] "Number of columns: 986"
```

5. Check for missing data in the data Check for the presence of "?" in the entire dataset

```
missing_data_placeholder <- sum(genotypes == "?/?")
print(paste("Number of '?/?' placeholders:", missing_data_placeholder))
```

```
## [1] "Number of '?/?' placeholders: 135452"
```

6. Most common element in the "Group" column

```
most_common_group <- names(sort(table(genotypes$Group), decreasing = TRUE))[1]
print(paste("Most common element in Group column:", most_common_group))
```

```
## [1] "Most common element in Group column: ZMMLR"
```

7. Most common element in the "Gene" column

```
most_common_gene <- names(sort(table(snp_position$gene), decreasing = TRUE))[1]
print(paste("Most common element in gene column:", most_common_gene))
```

```
## [1] "Most common element in gene column: zmm28"
```

After inspection, I learnt the file size is 10.54 mb The data has 2782 rows and 986 columns There is 135452 missing data encoded by ?/? The most common group is "ZMMLR" The file is ASCII text with very long lines

Inspecting snp_position.txt

1. File size (in bytes)

```
file_size_snp <- file.info("snp_position.txt")$size
print(paste("File size (bytes):", file_size_snp))
```

```
## [1] "File size (bytes): 82763"
```

2. View the first 6 rows (head) of the data

```
head(snp_position)
```

```
##     SNP_ID cdv_marker_id Chromosome  Position alt_pos mult_positions amplicon
## 1 abph1.20          5976          2  27403404                           abph1
## 2 abph1.22          5978          2  27403892                           abph1
## 3    ae1.3          6605          5 167889790                             ae1
## 4    ae1.4          6606          5 167889682                             ae1
## 5    ae1.5          6607          5 167889821                             ae1
## 6    an1.4          5982          1 240498509                             an1
##   cdv_map_feature.name  gene candidate.random Genaissance_daa_id
## 1             AB042260 abph1        candidate               8393
## 2             AB042260 abph1        candidate               8394
## 3                  ae1   ae1        candidate               8395
## 4                  ae1   ae1        candidate               8396
## 5                  ae1   ae1        candidate               8397
```

```
## 6                  an1   an1        candidate               8398
##   Sequenom_daa_id count_amplicons count_cmf count_gene
## 1           10474               1         1          1
## 2           10475               0         0          0
## 3           10477               1         1          1
## 4           10478               0         0          0
## 5           10479               0         0          0
## 6           10481               1         1          1
```

3. View the last 6 rows (tail) of the data

```
tail(snp_position)
```

```
##     SNP_ID cdv_marker_id Chromosome  Position alt_pos mult_positions amplicon
## 978 zap1.2          3514          2 233128584                           zap1
## 979 zen1.1          3519    unknown   unknown                           zen1
## 980 zen1.2          3520    unknown   unknown                           zen1
## 981 zen1.4          3521    unknown   unknown                           zen1
## 982 zfl2.6          6463          2  12543294                           zfl2
## 983 zmm3.4          3527          9  16966348                           zmm3
##     cdv_map_feature.name gene candidate.random Genaissance_daa_id
## 978               L46400 zap1        candidate               8434
## 979             CF649098 zen1        candidate               8435
## 980             CF649098 zen1        candidate               8436
## 981             CF649098 zen1        candidate               8437
## 982                 zfl2 zfl2        candidate               8438
## 983               Y09301 zmm3        candidate              10104
##     Sequenom_daa_id count_amplicons count_cmf count_gene
## 978           11823               1         0          1
## 979           11824               1         1          1
## 980           11826               0         0          0
## 981           11827               0         0          0
## 982           11828               1         1          1
## 983           11829               1         0          1
```

4. Number of rows and columns in the data

```
num_rows_snp <- nrow(snp_position)
num_cols_snp <- ncol(snp_position)
print(paste("Number of rows:", num_rows_snp))
```

```
## [1] "Number of rows: 983"
```

```
print(paste("Number of columns:", num_cols_snp))
```

```
## [1] "Number of columns: 15"
```

5. Check for missing data in the data Check for the presence of "?" in the entire dataset

```r
missing_data_placeholder <- sum(snp_position == "?/?")
print(paste("Number of '?/?' placeholders:", missing_data_placeholder))
```

```
## [1] "Number of '?/?' placeholders: 0"
```

6. Most common element in the "Chromosome" column

```r
most_common_chromosome <- names(sort(table(snp_position$Chromosome), decreasing = TRUE))[1]
print(paste("Most common element in Chromosome column:", most_common_chromosome))
```

```
## [1] "Most common element in Chromosome column: 1"
```

After inspection, I learnt the file size is 79 kb The data has 983 rows and 15 columns There is no missing data The most common chromosome number is 1 The most common gene type is "Zmm28" The file is ASCII text with very long lines

DATA PROCESSING

Transpose the data

```r
transposed_genotypes <- as.data.frame(t(genotypes), stringsAsFactors = FALSE)
```

Convert first row to column names

```r
colnames(transposed_genotypes) <- transposed_genotypes[3, ]
```

Remove the first row as it's now the column names

```r
transposed_genotypes <- transposed_genotypes[-c(1,2) ]
```

Add original column names as a new first column

```r
transposed_genotypes <- cbind(Original_Colnames = rownames(transposed_genotypes), transposed_genotypes)
transposed_genotypes <- transposed_genotypes[-c(1:3), ]
colnames(transposed_genotypes)[1] <- "SNP_ID"
```

Reset row names

```r
rownames(transposed_genotypes) <- NULL
```

Extract needed columns for merging

```r
snp_extract <- select(snp_position, SNP_ID, Chromosome, Position)
```

Merge snp_position and transposed_genotypes by "SNP_ID"

```r
colnames(transposed_genotypes) <- make.unique(colnames(transposed_genotypes))
merged <- left_join(snp_extract, transposed_genotypes, by = "SNP_ID")
```

Extract maize columns

```r
maize <- merged %>% select(SNP_ID, Chromosome = Chromosome, Position = Position,
                                  starts_with("ZMMIL"), starts_with("ZMMLR"), starts_with("ZMMMR"))
```

Subset data by specific values in the Chromosome column and sort by Position

```r
subset_data <- function(data, value, filename) {

# Subset the data for the given Chromosome value

  selected <- data[data$Chromosome == value, ]

  # Convert the Position column to numeric (ensuring it is numeric before sorting)
  selected$Position <- suppressWarnings(as.numeric(as.character(selected$Position)))

  # Sort the selected data by the Position column in ascending order
  selected_sorted <- selected[order(selected$Position), ]

  # Write the sorted data to the file
  write.table(selected_sorted, filename, sep = "\t", row.names = FALSE, quote = FALSE)
}
```

Loop through chromosomes 1 to 10 and subset

```r
for (i in 1:10) {
  subset_data(maize, i, paste0("Maize_chr", i, ".txt"))
}
```

Create a directory for ascending files and move them

```r
dir.create("Maize_ascend")
```

```
## Warning in dir.create("Maize_ascend"): 'Maize_ascend' already exists
```

```r
file.rename(list.files(pattern = "Maize_chr[0-9]+.txt"), file.path("Maize_ascend", list.files(pattern =
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Extract the first line (header) from maize

```r
header <- head(maize, 1)
```

Creating multiple and unknown chromosome files

```r
# Filter rows where the second column is "multiple" (for Maize_chrm.txt)
maize_chrm <- maize[maize$Chromosome == "multiple", ]

# Combine header with the filtered data and write to Maize_chrm.txt
maize_chrm_final <- rbind(header, maize_chrm)
write.table(maize_chrm_final, "Maize_chrm.txt", sep = "\t", row.names = FALSE, col.names = TRUE, quote =
```

```
# Filter rows where the second column is "unknown" (for Maize_chru.txt)
maize_chru <- maize[maize$Chromosome == "unknown", ]

# Combine header with the filtered data and write to Maize_chru.txt
maize_chru_final <- rbind(header, maize_chru)
write.table(maize_chru_final, "Maize_chru.txt", sep = "\t", row.names = FALSE, col.names = TRUE, quote =
```

replace ? with -

```
maize_hyphen <- maize
maize_hyphen[maize_hyphen == "?/?"] <- "-/-"
```

Subset data by specific values in the Chromosome column and sort by Position in descending order

```
subset_data <- function(data, value, filename) {
  # Subset the data for the given Chromosome value
  selected <- data[data$Chromosome == value, ]

  # Convert the Position column to numeric (ensure it is numeric before sorting)
  selected$Position <- suppressWarnings(as.numeric(as.character(selected$Position)))

  # Sort the selected data by the Position column in descending order
  selected_sorted <- selected[order(selected$Position, decreasing = TRUE), ]

  # Write the sorted data to the file
  write.table(selected_sorted, filename, sep = "\t", row.names = FALSE, quote = FALSE)
}
```

Loop through chromosomes 1 to 10 and subset

```
for (i in 1:10) {
  subset_data(maize_hyphen, i, paste0("Maize_chrd", i, ".txt"))
}
```

Create a directory for descending files and move them

```
dir.create("Maize_descend")
```

```
## Warning in dir.create("Maize_descend"): 'Maize_descend' already exists
```

```
file.rename(list.files(pattern = "Maize_chrd[0-9]+.txt"), file.path("Maize_descend", list.files(pattern
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

FOR TEOSINTE

Extract teosinte columns

```
teosinte <- merged %>% select(SNP_ID, Chromosome = Chromosome, Position = Position,
                              starts_with("ZMPBA"), starts_with("ZMPIL"), starts_with("ZMPJA"))
```

Subset data by specific values in the Chromosome column and sort by Position

```
subset_data <- function(data, value, filename) {
  # Subset the data for the given Chromosome value
  selected <- data[data$Chromosome == value, ]

  # Convert the Position column to numeric (ensuring it is numeric before sorting)
  selected$Position <- suppressWarnings(as.numeric(as.character(selected$Position)))

  # Sort the selected data by the Position column in ascending order
  selected_sorted <- selected[order(selected$Position), ]

  # Write the sorted data to the file
  write.table(selected_sorted, filename, sep = "\t", row.names = FALSE, quote = FALSE)
}
```

Loop through chromosomes 1 to 10 and subset

```
for (i in 1:10) {
  subset_data(teosinte, i, paste0("Teosinte_chr", i, ".txt"))
}
```

Create a directory for ascending files and move them

```
dir.create("Teosinte_ascend")
```

```
## Warning in dir.create("Teosinte_ascend"): 'Teosinte_ascend' already exists
```

```
file.rename(list.files(pattern = "Teosinte_chr[0-9]+.txt"), file.path("Teosinte_ascend", list.files(pat
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Creating multiple and unknown chromosome files

```
#Extract the first line (header) from teosinte
header <- head(teosinte, 1)

# Filter rows where the second column is "multiple"
teosinte_chrm <- teosinte[teosinte$Chromosome == "multiple", ]

# Combine header with the filtered data and write to Teosinte_chrm.txt
teosinte_chrm_final <- rbind(header, teosinte_chrm)
write.table(teosinte_chrm_final, "Teosinte_chrm.txt", sep = "\t", row.names = FALSE, col.names = TRUE, 

# Filter rows where the second column is "unknown"
teosinte_chru <- teosinte[teosinte$Chromosome == "unknown", ]

# Combine header with the filtered data and write to Teosinte_chru.txt
teosinte_chru_final <- rbind(header, teosinte_chru)
write.table(teosinte_chru_final, "Teosinte_chru.txt", sep = "\t", row.names = FALSE, col.names = TRUE, 
```

replace ? with -

```
teosinte_hyphen <- teosinte
teosinte_hyphen[teosinte_hyphen == "?/?"] <- "-/-"
```

Subset data by specific values in the Chromosome column and sort by Position in descending order

```
subset_data <- function(data, value, filename) {
  # Subset the data for the given Chromosome value
  selected <- data[data$Chromosome == value, ]

  # Convert the Position column to numeric
  selected$Position <- suppressWarnings(as.numeric(as.character(selected$Position)))

  # Sort the selected data by the Position column in descending order
  selected_sorted <- selected[order(selected$Position, decreasing = TRUE), ]

  # Write the sorted data to the file
  write.table(selected_sorted, filename, sep = "\t", row.names = FALSE, quote = FALSE)
}
```

Loop through chromosomes 1 to 10 and subset

```
for (i in 1:10) {
  subset_data(teosinte_hyphen, i, paste0("Teosinte_chrd", i, ".txt"))
}
```

Create a directory for descending files and move them

```
dir.create("Teosinte_descend")
```

```
## Warning in dir.create("Teosinte_descend"): 'Teosinte_descend' already exists
```

```
file.rename(list.files(pattern = "Teosinte_chrd[0-9]+.txt"), file.path("Teosinte_descend", list.files(pa
```

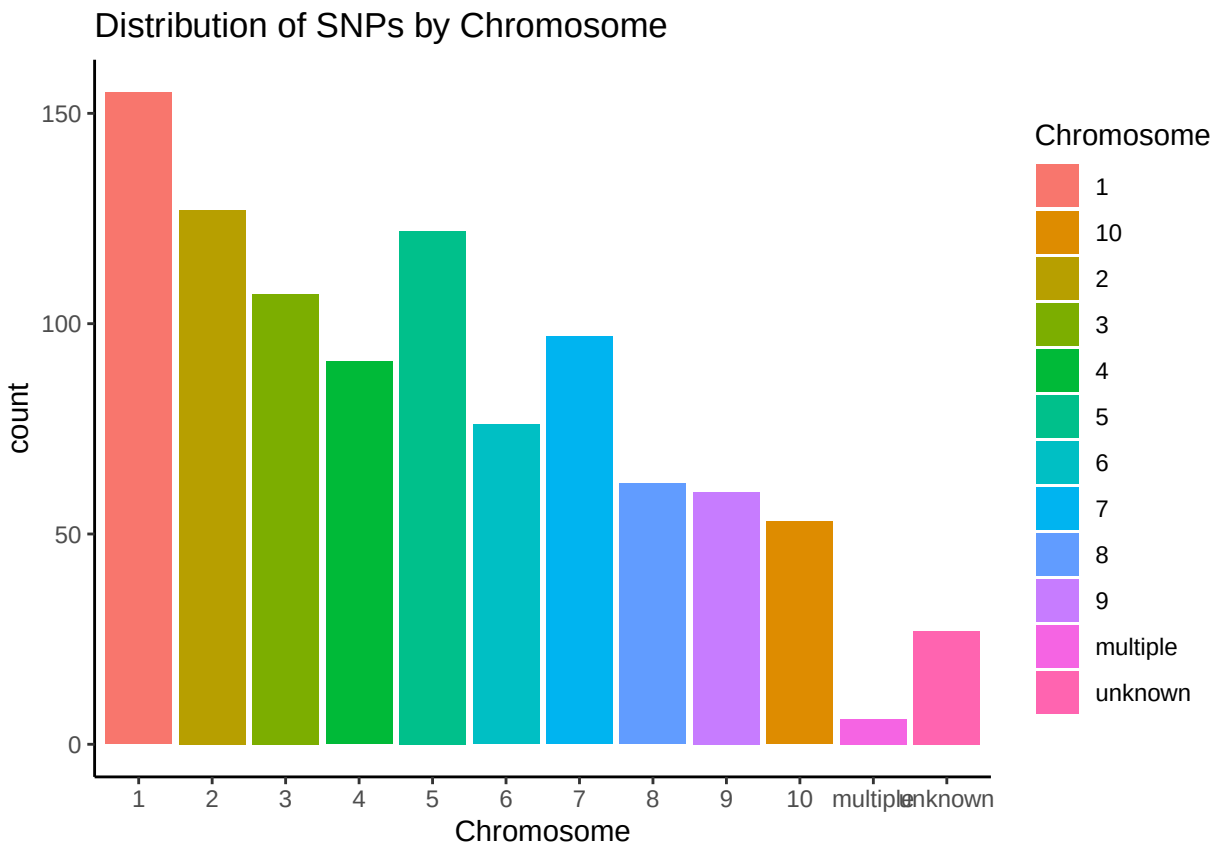
```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

DATA VISUALIZATION

Distribution of SNPs between chromosomes

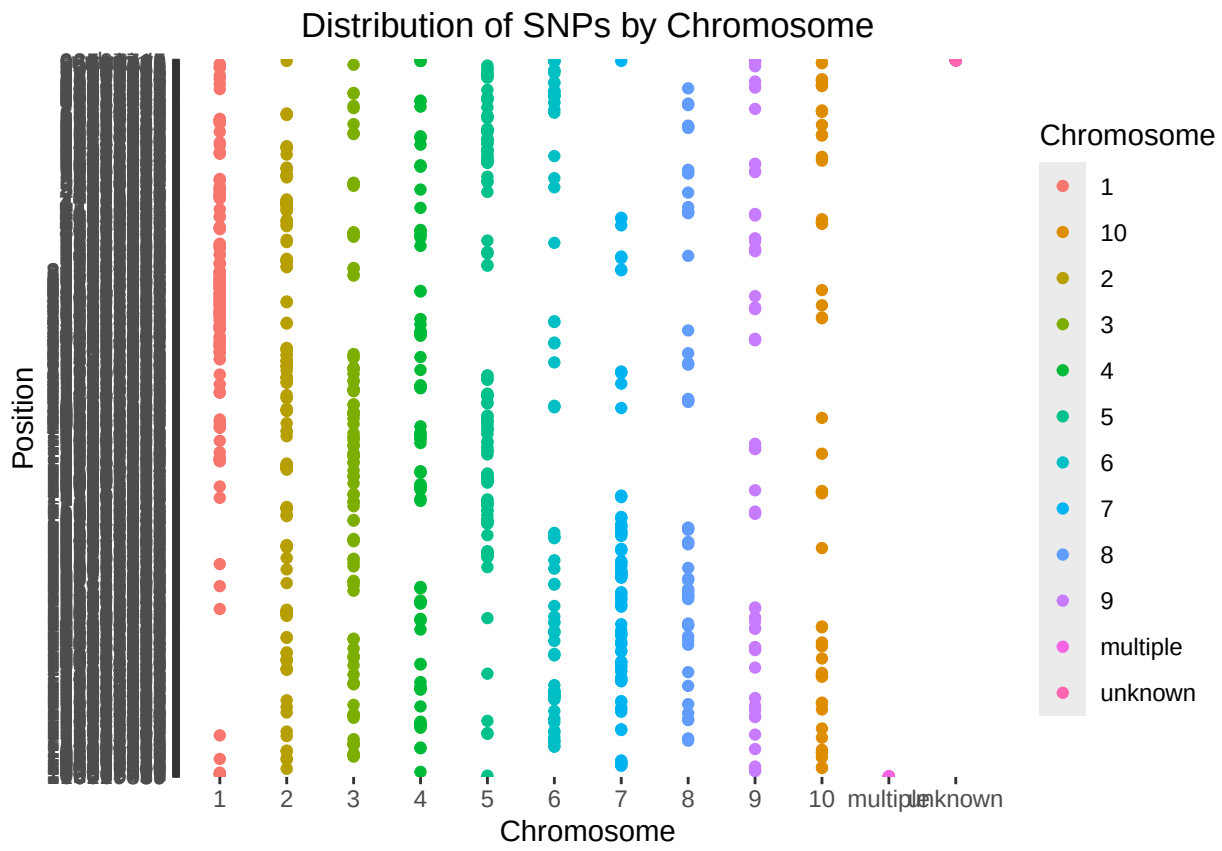
Bar chart of the distribution of SNPs by Chromosome

```
ggplot(data = maize) +
    geom_bar(mapping = aes(x = Chromosome, fill = Chromosome)) +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown"))
  ggtitle("Distribution of SNPs by Chromosome") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme_classic()
```



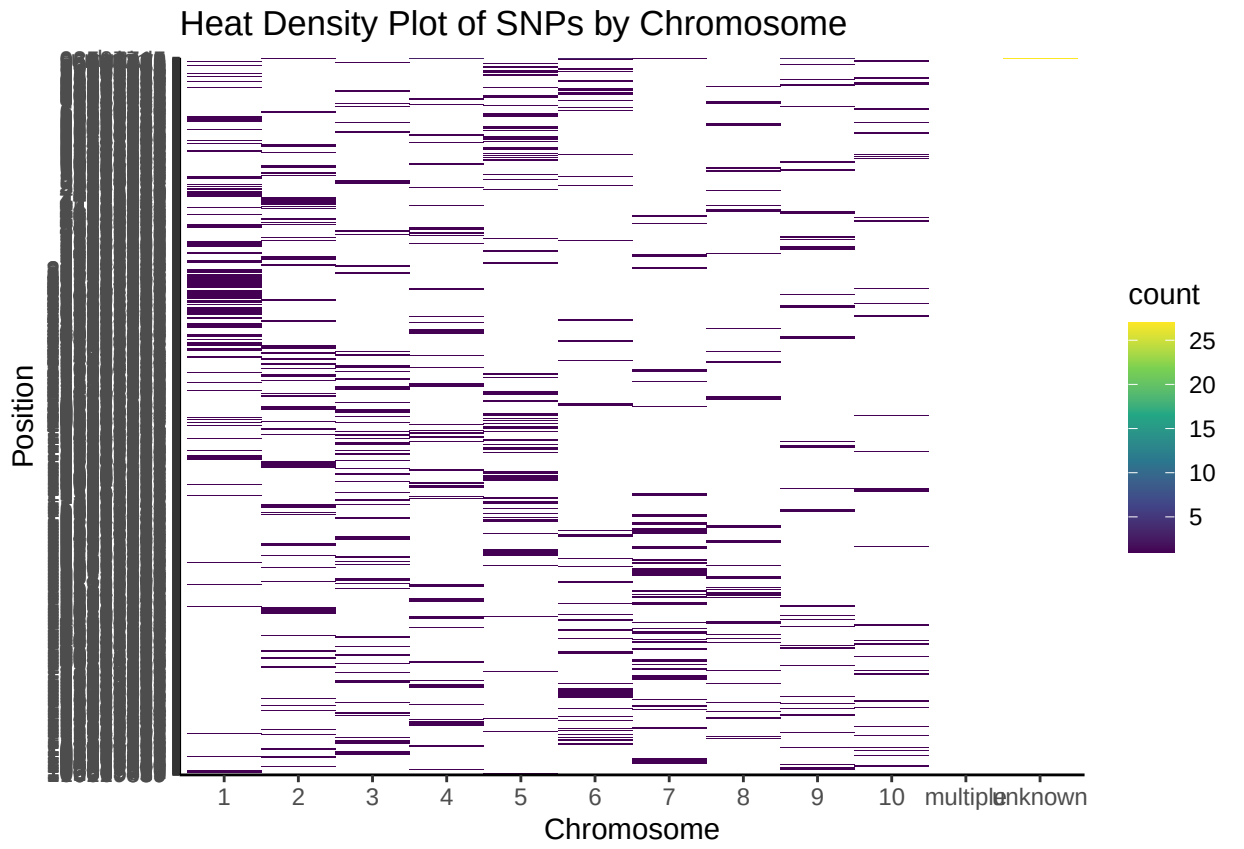
Scatter plot of the distribution of SNPs by Chromosome

```
ggplot(data = maize) +
  geom_point(mapping = aes(x = Chromosome, y = Position, color = Chromosome)) +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown")) +
  ggtitle("Distribution of SNPs by Chromosome") +
  theme(plot.title = element_text(hjust = 0.5))
```



Heat map of SNPs by Chromosome

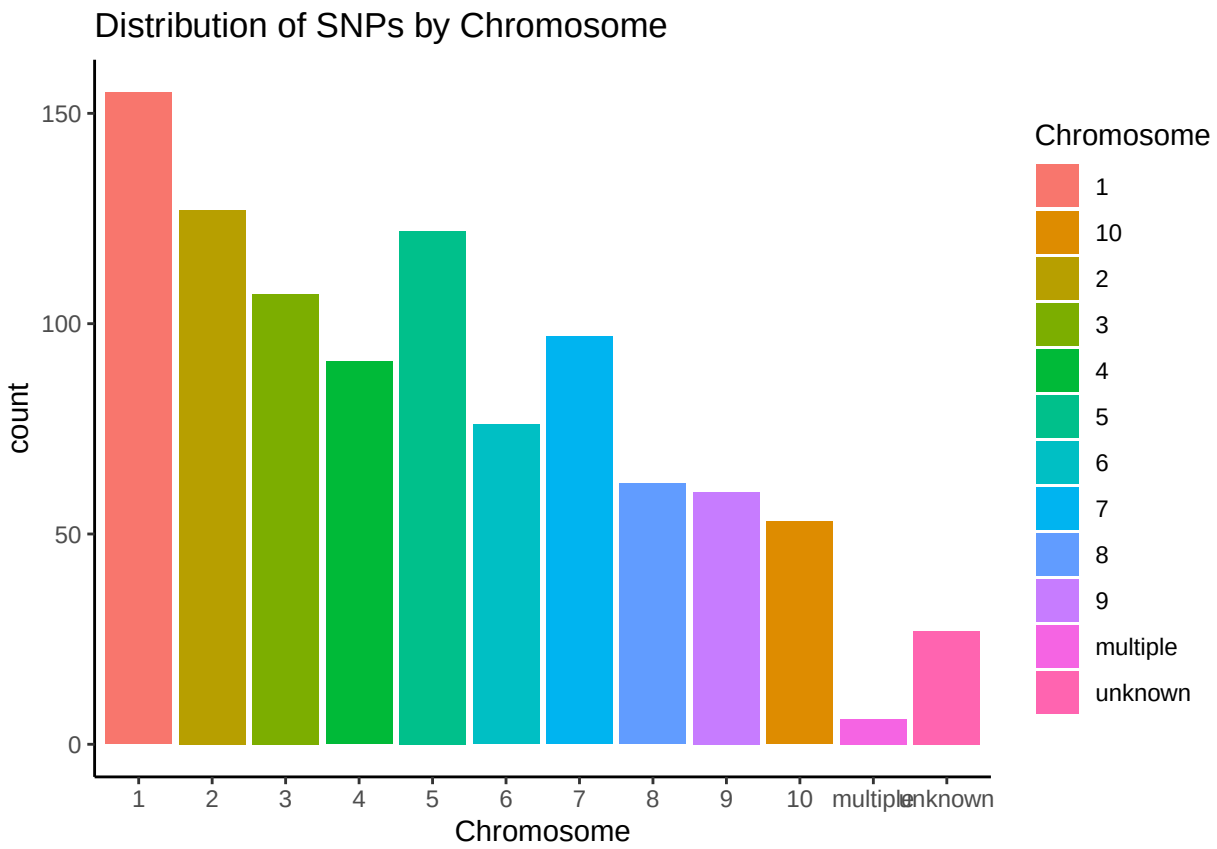
```
ggplot(data = maize) +
    geom_bin2d(mapping = aes(x = Chromosome, y = Position), bins = 30) +  # Use geom_bin2d for 2D binni
    scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown
    ggtitle("Heat Density Plot of SNPs by Chromosome") +
    theme(plot.title = element_text(hjust = 0.5)) +
    theme_classic() +
    scale_fill_viridis_c()
```



FOR TEOSINTE

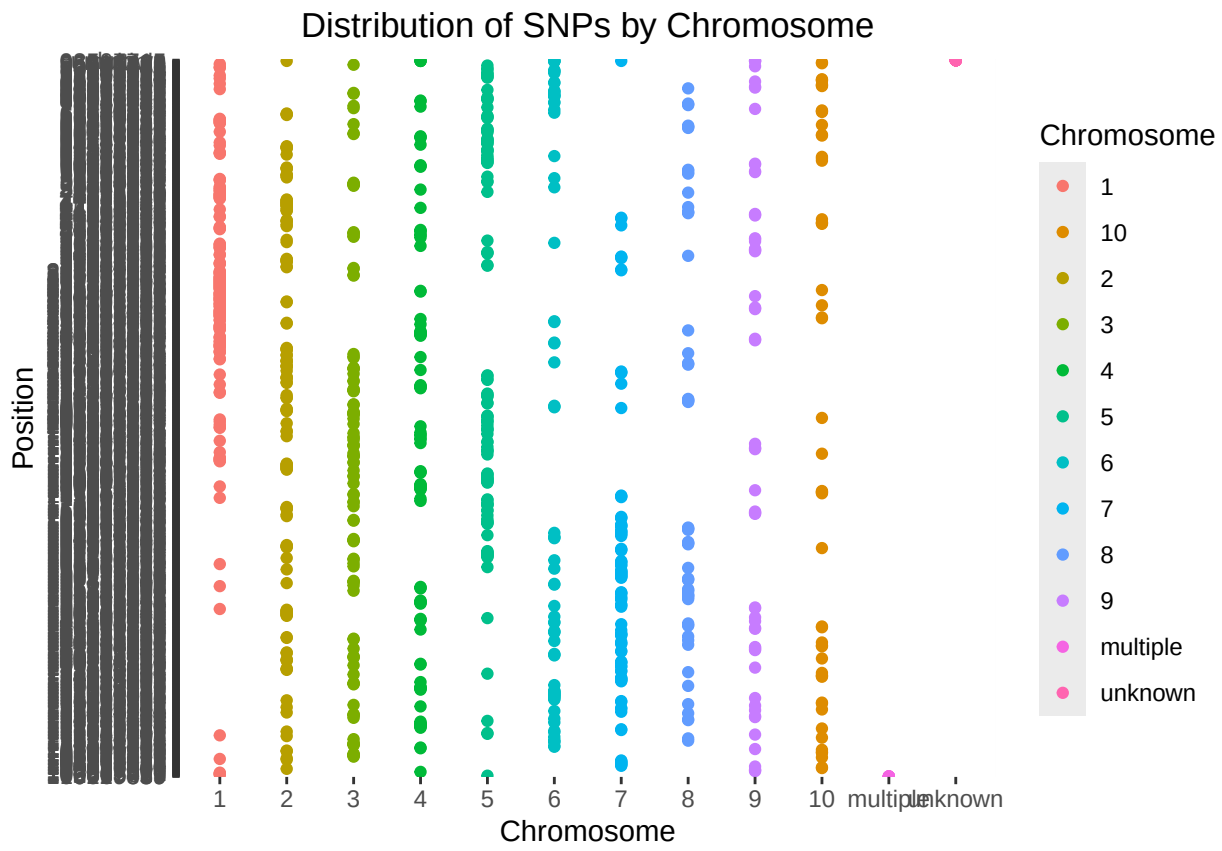
Bar chart of the distribution of SNPs by Chromosome

```
ggplot(data = teosinte) +
    geom_bar(mapping = aes(x = Chromosome, fill = Chromosome)) +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown")) +
  ggtitle("Distribution of SNPs by Chromosome") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme_classic()
```



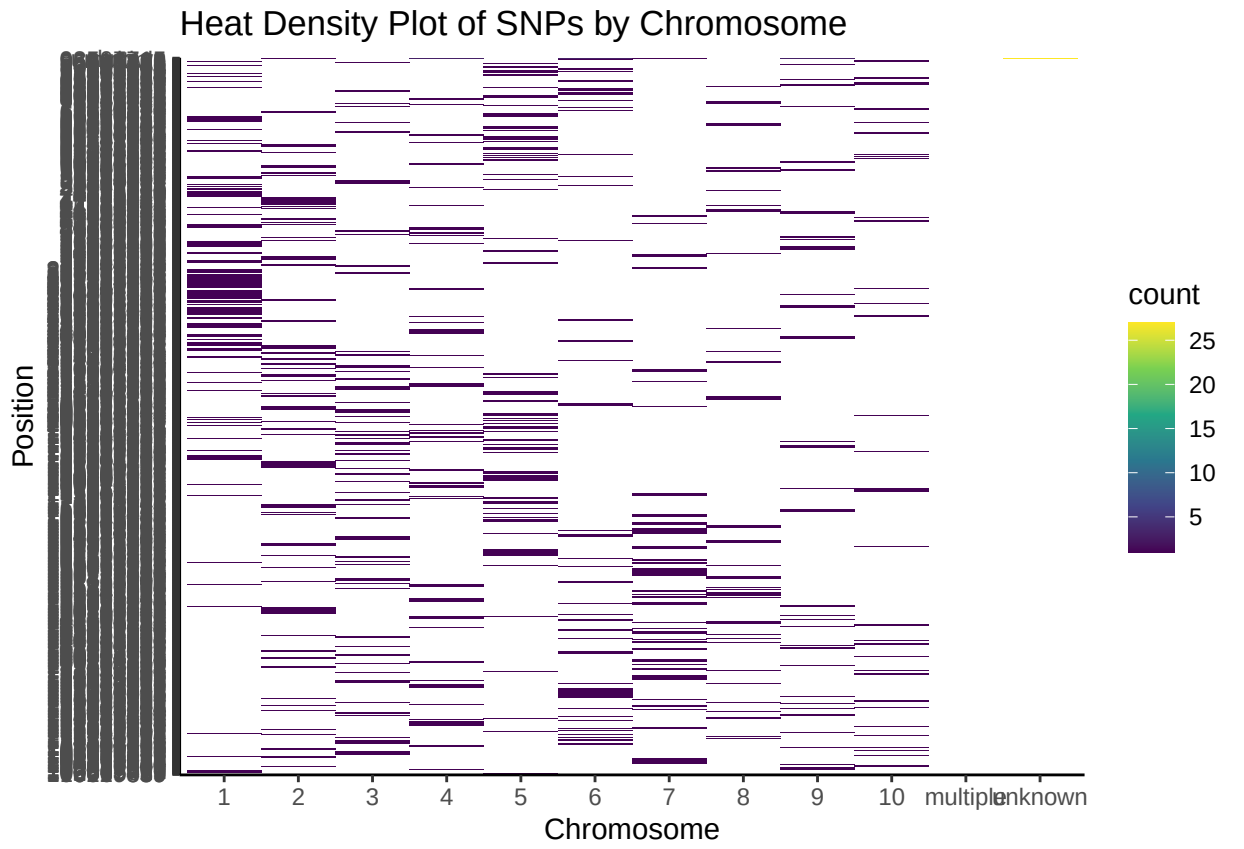
Scatter plot of the distribution of SNPs by Chromosome

```
ggplot(data = teosinte) +
  geom_point(mapping = aes(x = Chromosome, y = Position, color = Chromosome)) +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown")) +
  ggtitle("Distribution of SNPs by Chromosome") +
  theme(plot.title = element_text(hjust = 0.5))
```



Heat map of SNPs by Chromosome

```
ggplot(data = teosinte) +
    geom_bin2d(mapping = aes(x = Chromosome, y = Position), bins = 30) +  # Use geom_bin2d for 2D binni
    scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown
    ggtitle("Heat Density Plot of SNPs by Chromosome") +
    theme(plot.title = element_text(hjust = 0.5)) +
    theme_classic() +
    scale_fill_viridis_c()
```



Merged data

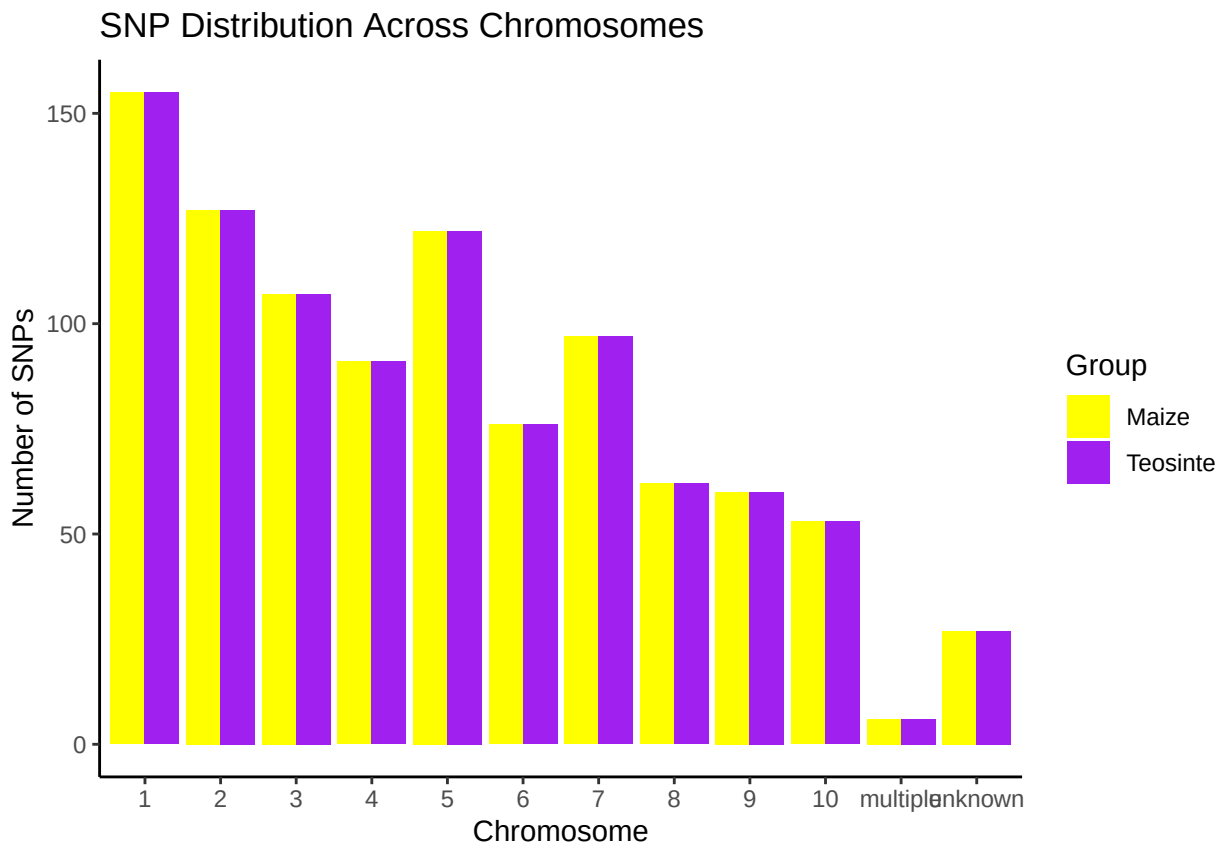
```
# Count SNPs per chromosome for maize
maize_snp <- maize %>%
  group_by(Chromosome) %>%
  summarise(SNP_Count = n()) %>%
  mutate(Group = "Maize")

# Count SNPs per chromosome for teosinte
teosinte_snp <- teosinte %>%
  group_by(Chromosome) %>%
  summarise(SNP_Count = n()) %>%
  mutate(Group = "Teosinte")

# Combine data
snp_counts <- bind_rows(maize_snp, teosinte_snp)
```

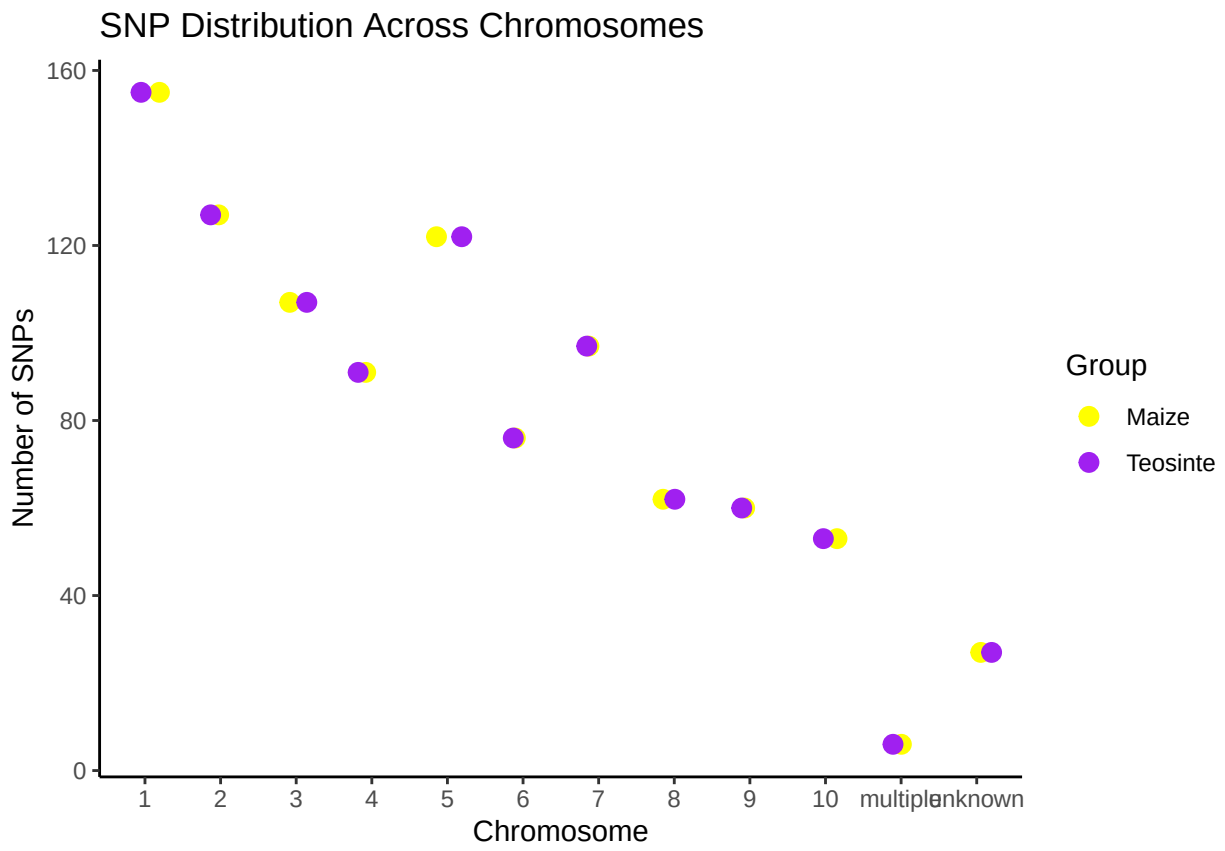
Stacked Bar chart of SNP count per chromosome

```
ggplot(snp_counts, aes(x = Chromosome, y = SNP_Count, fill = Group)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "SNP Distribution Across Chromosomes",
       x = "Chromosome", y = "Number of SNPs") +
  scale_fill_manual(values = c("Maize" = "yellow", "Teosinte" = "purple")) +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown")) +
  theme_classic()
```



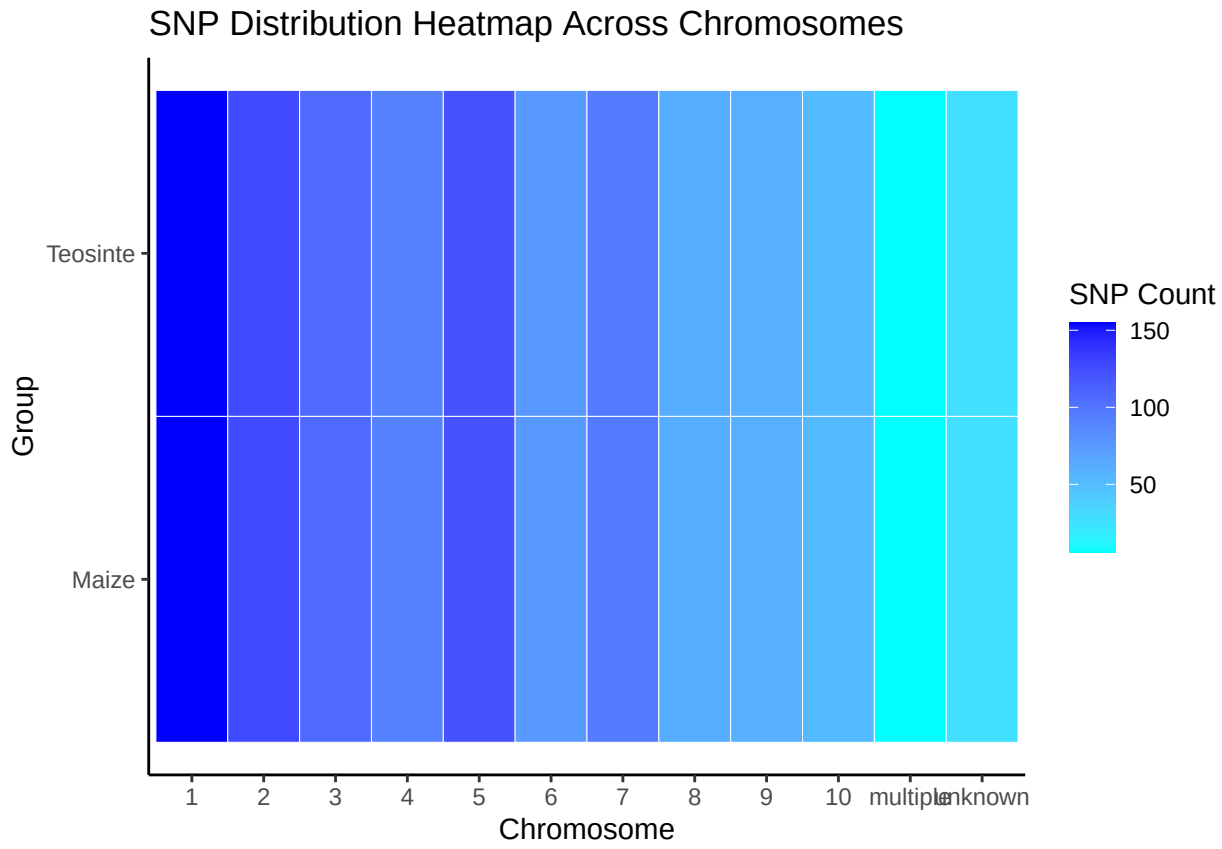
Scatter plot of merged data

```
ggplot(snp_counts, aes(x = Chromosome, y = SNP_Count, color = Group)) +
  geom_point(size = 3, position = position_jitter(width = 0.2, height = 0)) + # Use 'geom_point' to cre
  labs(title = "SNP Distribution Across Chromosomes",
       x = "Chromosome", y = "Number of SNPs") +
  scale_color_manual(values = c("Maize" = "yellow", "Teosinte" = "purple")) +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown"))
    theme_classic()
```



Heat map of merged data

```
ggplot(snp_counts, aes(x = Chromosome, y = Group, fill = SNP_Count)) +
  geom_tile(color = "white ") +
  labs(title = "SNP Distribution Heatmap Across Chromosomes",
       x = "Chromosome", y = "Group", fill = "SNP Count") +
  scale_fill_gradient(low = "cyan", high = "blue") +
  scale_x_discrete(limits = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "multiple", "unknown")) +
  theme_classic()
```


SNP Distribution Heatmap Across Chromosomes
Group
Teosinte
Maize
1
2
3
4
5
6
7
8
9
10
multiple
unknown
Chromosome
SNP Count
150
100
50