BIOL-GA.1132 Assignment01

Wanru Lin

10/2/2021

```
# load packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(VariantAnnotation)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
```

```
## Loading required package: MatrixGenerics
## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following object is masked from 'package:dplyr':
##
##     count

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: GenomeInfoDb

## Loading required package: S4Vectors

## Loading required package: stats4

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following object is masked from 'package:tidyr':
##
##     expand

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

## The following object is masked from 'package:purrr':
```

```
##
##     reduce
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: SummarizedExperiment
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
##
## Attaching package: 'Biobase'
```

```
## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians
```

```
## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
## Loading required package: Rsamtools
```

```
## Loading required package: Biostrings
```

```
## Loading required package: XVector
```

```
##
## Attaching package: 'XVector'
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

```
##
## Attaching package: 'Biostrings'
```

```
## The following object is masked from 'package:base':
##
##     strsplit
```

```
##
## Attaching package: 'VariantAnnotation'
```

```
## The following object is masked from 'package:stringr':
##
##     fixed
```

```
## The following object is masked from 'package:base':
##
##     tabulate
```

Use the VariantAnnotation package in R to assist with parsing the VCF

```
# read the vcf file
vcf.chr7 <- readVcf("chr7YRI_6000000_8000000.vcf","hg19")
class(vcf.chr7)
```

```
## [1] "CollapsedVCF"
## attr(,"package")
## [1] "VariantAnnotation"
```

```
dim(vcf.chr7) # 73783 refSNP, 107 samples
```

```
## [1] 73783   107
```

```
snv.chr7 <- isSNV(vcf.chr7)
sum(snv.chr7) # 70843 SNVs
```

```
## [1] 70843
```

```
# get a logical vector with biallelic SNPs only
bi.snv.chr7 <- isSNV(vcf.chr7,singleAltOnly=TRUE)
# 'singleAltOnly': single alternate allele
sum(bi.snv.chr7) # 70843 biallelic SNPs
```

```
## [1] 70843
```

```
vcf.snps_only.chr7 <- vcf.chr7[bi.snv.chr7,]
```

```
chr7.geno <- geno(vcf.snps_only.chr7)

chr7.gt.mat <- chr7.geno[['GT']]
head(chr7.gt.mat[,1:10])
```

```
##             NA18486 NA18488 NA18489 NA18498 NA18499 NA18501 NA18502 NA18504
## rs62454735  "0|0"   "0|0"   "1|0"   "0|1"   "0|0"   "0|1"   "1|0"   "0|0"
## rs147838625 "0|0"   "0|0"   "1|0"   "0|1"   "0|0"   "0|1"   "1|0"   "0|0"
## rs573650620 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
## rs201408132 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
## rs553135573 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
## rs577832347 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
##             NA18505 NA18507
## rs62454735  "0|1"   "1|1"
## rs147838625 "0|1"   "1|1"
## rs573650620 "0|0"   "0|0"
## rs201408132 "0|0"   "0|0"
## rs553135573 "0|0"   "0|0"
## rs577832347 "0|0"   "0|0"
```

```
dim(chr7.gt.mat)
```

```
## [1] 70843   107
```

Count alleles and deriving the allele frequencies from genotype frequencies

```
####
geno.tbl_df <- chr7.gt.mat %>% as.data.frame %>%
  as.data.frame(stringsAsFactors = F) %>%
  rownames_to_column(var = "id") %>%
  as_tibble

geno.tbl_df[1:6, 1:6] # view
```

```
## # A tibble: 6 x 6
##   id          NA18486 NA18488 NA18489 NA18498 NA18499
##   <chr>       <chr>    <chr>    <chr>    <chr>    <chr>
## 1 rs62454735  0|0      0|0      1|0      0|1      0|0
## 2 rs147838625 0|0      0|0      1|0      0|1      0|0
## 3 rs573650620 0|0      0|0      0|0      0|0      0|0
## 4 rs201408132 0|0      0|0      0|0      0|0      0|0
## 5 rs553135573 0|0      0|0      0|0      0|0      0|0
## 6 rs577832347 0|0      0|0      0|0      0|0      0|0
```

```
class(geno.tbl_df)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
# Now we can make our data into long format with pivot_longer
geno.long.tbl_df <- geno.tbl_df %>%
  pivot_longer(cols = -id, names_to = "sample", values_to = "GT")

head(geno.long.tbl_df,15)
```

```
## # A tibble: 15 x 3
##    id         sample  GT
##    <chr>      <chr>   <chr>
##  1 rs62454735 NA18486 0|0
##  2 rs62454735 NA18488 0|0
##  3 rs62454735 NA18489 1|0
##  4 rs62454735 NA18498 0|1
##  5 rs62454735 NA18499 0|0
##  6 rs62454735 NA18501 0|1
##  7 rs62454735 NA18502 1|0
##  8 rs62454735 NA18504 0|0
##  9 rs62454735 NA18505 0|1
## 10 rs62454735 NA18507 1|1
## 11 rs62454735 NA18508 0|1
## 12 rs62454735 NA18510 1|1
## 13 rs62454735 NA18511 1|1
## 14 rs62454735 NA18516 0|0
## 15 rs62454735 NA18517 0|0
```

```
# I want to know what kinds of genotypes are there in the 'GT' column
levels(factor(geno.long.tbl_df$GT))
```

```
## [1] "0|0" "0|1" "1|0" "1|1"
```

```
geno.long_w_counts.tbl_df <- geno.long.tbl_df %>%
  mutate(homref_count = ifelse(GT == "0|0",1,0 )) %>% # homo referenct count
  mutate(het_count = ifelse(GT == "0|1" | GT =="1|0",1,0)) %>%
  mutate(homalt_count = ifelse(GT == "1|1",1,0)) %>%
  mutate(alt_count = homalt_count * 2 + het_count) %>% # alternative allele
  mutate(ref_count = homref_count * 2 + het_count ) # reference allele

head(geno.long_w_counts.tbl_df,15)
```

```
## # A tibble: 15 x 8
##    id        sample GT    homref_count het_count homalt_count alt_count ref_count
##    <chr>     <chr>  <chr>        <dbl>     <dbl>        <dbl>     <dbl>     <dbl>
##  1 rs62454~ NA184~ 0|0              1         0            0         0         2
##  2 rs62454~ NA184~ 0|0              1         0            0         0         2
##  3 rs62454~ NA184~ 1|0              0         1            0         1         1
##  4 rs62454~ NA184~ 0|1             0         1            0          1          1
##  5 rs62454~ NA184~ 0|0             1         0            0          0          2
##  6 rs62454~ NA185~ 0|1             0         1            0          1          1
##  7 rs62454~ NA185~ 1|0             0         1            0          1          1
##  8 rs62454~ NA185~ 0|0             1         0            0          0          2
##  9 rs62454~ NA185~ 0|1             0         1            0          1          1
## 10 rs62454~ NA185~ 1|1             0         0            1          2          0
## 11 rs62454~ NA185~ 0|1             0         1            0          1          1
## 12 rs62454~ NA185~ 1|1             0         0            1          2          0
## 13 rs62454~ NA185~ 1|1             0         0            1          2          0
## 14 rs62454~ NA185~ 0|0             1         0            0          0          2
## 15 rs62454~ NA185~ 0|0             1         0            0          0          2
```

```
geno.summary_per_locus <- geno.long_w_counts.tbl_df %>%
  group_by(id) %>%
  summarise(tot_alt_count = sum(alt_count),
            tot_ref_count = sum(ref_count))
head(geno.summary_per_locus)
```

```
## # A tibble: 6 x 3
##   id         tot_alt_count tot_ref_count
##   <chr>              <dbl>         <dbl>
## 1 rs10000                4           210
## 2 rs10046499            27           187
## 3 rs10046572            50           164
## 4 rs10046580            21           193
## 5 rs1007999             19           195
## 6 rs1008000             26           188
```

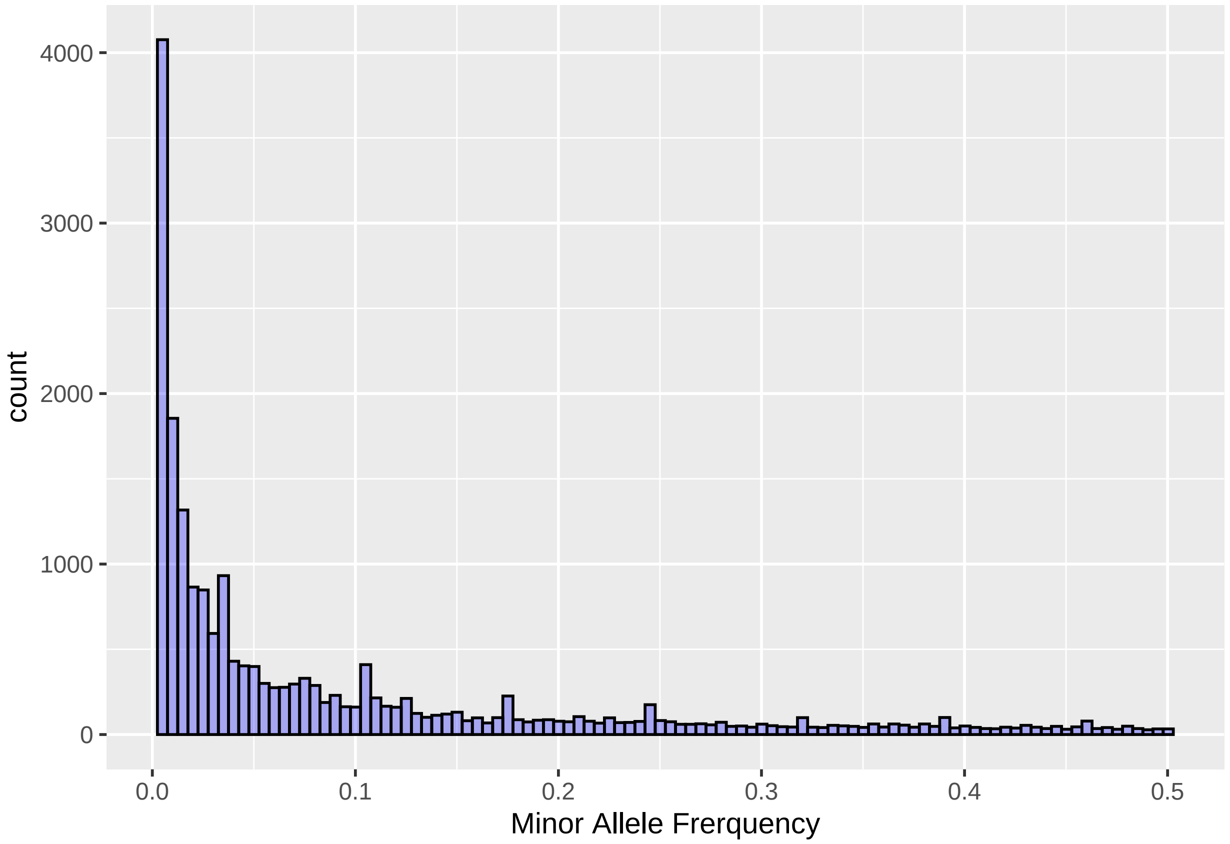
```
# filter refSNPs which only have "1|1" or "0|0" in all samples
# If only have "1|1" or "0|0" in all samples,
# the tot_alt_count or tot_ref_count will both equal to 214
index_only_alt_or_ref = which((geno.summary_per_locus$tot_ref_count==214)|
                                (geno.summary_per_locus$tot_alt_count==214))
geno.without.only = geno.summary_per_locus[-index_only_alt_or_ref,]
```

```
geno.chr7 = geno.without.only %>%
  mutate(tot_minor =
           case_when(tot_alt_count <= tot_ref_count ~ tot_alt_count,
                     tot_ref_count < tot_alt_count ~ tot_ref_count,
                     TRUE ~ NA_real_)) %>%
  mutate(maf = tot_minor / (tot_alt_count + tot_ref_count))
head(geno.chr7)
```

```
## # A tibble: 6 x 5
##   id         tot_alt_count tot_ref_count tot_minor    maf
##   <chr>              <dbl>         <dbl>     <dbl>  <dbl>
## 1 rs10000                4           210         4 0.0187
## 2 rs10046499            27           187        27 0.126
## 3 rs10046572            50           164        50 0.234
## 4 rs10046580            21           193        21 0.0981
## 5 rs1007999             19           195        19 0.0888
## 6 rs1008000             26           188        26 0.121
```

Create the minor allele site frequency spectrum

```
library(ggplot2)
ggplot(data=geno.chr7,aes(maf))+
  geom_histogram(bins =100,fill="blue",color="black",alpha=0.3)+
  xlab("Minor Allele Frerquency") -> SFS.Yoruban
SFS.Yoruban
```



Identify counts per frequency class

```
level_of_maf = levels(factor(geno.chr7$maf))

count_of_every_maf=numeric()

for (i in 1:length(level_of_maf)){
  count_of_every_maf[i] = sum(geno.chr7$maf == level_of_maf[i])
}
names(count_of_every_maf) = level_of_maf
count_of_every_maf
```

```
## 0.00467289719626168 0.00934579439252336   0.014018691588785   0.0186915887850467
##                4076                1855                1317                 865
##   0.0233644859813084   0.0280373831775701   0.0327102803738318   0.0373831775700935
##                 848                 593                 507                 425
##   0.0420560747663551   0.0467289719626168   0.0514018691588785   0.0560747663551402
##                 430                 403                 399                 300
##   0.0607476635514019   0.0654205607476635   0.0700934579439252   0.0747663551401869
##                275                277                296                330 
## 0.0794392523364486 0.0841121495327103  0.088785046728972 0.0934579439252336 
##                288                188                230                163 
## 0.0981308411214953  0.102803738317757  0.107476635514019   0.11214953271028 
##                161                219                191                215 
##  0.116822429906542  0.121495327102804  0.126168224299065   0.130841121495327 
##                166                160                212                124 
##  0.135514018691589   0.14018691588785  0.144859813084112  0.149532710280374 
##                101                113                120                131 
##  0.154205607476636  0.158878504672897  0.163551401869159  0.168224299065421 
##                 81                 98                 68                 99 
##  0.172897196261682  0.177570093457944  0.182242990654206  0.186915887850467 
##                136                 90                 87                 74 
##  0.191588785046729  0.196261682242991  0.200934579439252  0.205607476635514 
##                 84                 87                 78                 75 
##  0.210280373831776  0.214953271028037  0.219626168224299  0.224299065420561 
##                105                 78                 67                 98 
##  0.228971962616822  0.233644859813084  0.238317757009346  0.242990654205607 
##                 70                 71                 77                 91 
##  0.247663551401869  0.252336448598131  0.257009345794392  0.261682242990654 
##                 84                 82                 75                 60 
##  0.266355140186916  0.271028037383178  0.275700934579439  0.280373831775701 
##                 60                 63                 57                 72 
##  0.285046728971963  0.289719626168224  0.294392523364486  0.299065420560748 
##                 48                 50                 43                 61 
##  0.303738317757009  0.308411214953271  0.313084112149533  0.317757009345794 
##                 52                 46                 44                 53 
##  0.322429906542056  0.327102803738318  0.331775700934579  0.336448598130841 
##                 46                 43                 41                 54 
##  0.341121495327103  0.345794392523364  0.350467289719626  0.355140186915888 
##                 51                 48                 42                 62 
##   0.35981308411215  0.364485981308411  0.369158878504673  0.373831775700935 
##                 44                 62                 55                 43 
##  0.378504672897196  0.383177570093458   0.38785046728972  0.392523364485981 
##                 62                 48                 59                 41 
##  0.397196261682243  0.401869158878505  0.406542056074766  0.411214953271028 
##                 39                 50                 42                 35 
##   0.41588785046729  0.420560747663551  0.425233644859813  0.429906542056075 
##                 34                 43                 38                 54 
##  0.434579439252336  0.439252336448598   0.44392523364486  0.448598130841121 
##                 43                 35                 48                 31 
##  0.453271028037383  0.457943925233645  0.462616822429907  0.467289719626168 
##                 45                 45                 34                 35 
##   0.47196261682243  0.476635514018692  0.481308411214953  0.485981308411215 
##                 41                 31                 49                 35 
##  0.490654205607477  0.495327102803738                0.5 
##                 28                 33                 33
```

```
level_of_every_minor = levels(factor(geno.chr7$tot_minor))

count_of_every_minor=numeric()

for (i in 1:length(level_of_every_minor)){
  count_of_every_minor[i] = sum(geno.chr7$tot_minor == level_of_every_minor[i])
}
names(count_of_every_minor) = level_of_every_minor
count_of_every_minor
```

```
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 4076 1855 1317  865  848  593  507  425  430  403  399  300  275  277  296  330
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
##  288  188  230  163  161  219  191  215  166  160  212  124  101  113  120  131
##   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48
##   81   98   68   99  136   90   87   74   84   87   78   75  105   78   67   98
##   49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64
##   70   71   77   91   84   82   75   60   60   63   57   72   48   50   43   61
##   65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80
##   52   46   44   53   46   43   41   54   51   48   42   62   44   62   55   43
##   81   82   83   84   85   86   87   88   89   90   91   92   93   94   95   96
##   62   48   59   41   39   50   42   35   34   43   38   54   43   35   48   31
##   97   98   99  100  101  102  103  104  105  106  107
##   45   45   34   35   41   31   49   35   28   33   33
```

```
which(count_of_every_minor!=count_of_every_maf)
```

```
## named integer(0)
```

```
count_table=data.frame(total_minor=names(count_of_every_minor),
                       minor_allele_frequency=names(count_of_every_maf),
                       count_number=count_of_every_minor)
count_table
```

```
##     total_minor minor_allele_frequency count_number
## 1             1    0.00467289719626168         4076
## 2             2    0.00934579439252336         1855
## 3             3      0.014018691588785         1317
## 4             4     0.0186915887850467          865
## 5             5     0.0233644859813084          848
## 6             6     0.0280373831775701          593
## 7             7     0.0327102803738318          507
## 8             8     0.0373831775700935          425
## 9             9     0.0420560747663551          430
## 10           10     0.0467289719626168          403
## 11           11     0.0514018691588785          399
## 12           12     0.0560747663551402          300
## 13           13     0.0607476635514019          275
## 14           14     0.0654205607476635          277
## 15           15     0.0700934579439252          296
## 16           16     0.0747663551401869          330
## 17           17     0.0794392523364486          288
## 18           18     0.0841121495327103          188
## 19           19      0.088785046728972          230
## 20           20     0.0934579439252336          163
## 21           21     0.0981308411214953          161
## 22           22      0.102803738317757          219
## 23           23      0.107476635514019          191
## 24           24       0.11214953271028          215
## 25           25      0.116822429906542          166
## 26           26      0.121495327102804          160
## 27              27      0.126168224299065          212
## 28              28      0.130841121495327          124
## 29              29      0.135514018691589          101
## 30              30       0.14018691588785          113
## 31              31      0.144859813084112          120
## 32              32      0.149532710280374          131
## 33              33      0.154205607476636           81
## 34              34      0.158878504672897           98
## 35              35      0.163551401869159           68
## 36              36      0.168224299065421           99
## 37              37      0.172897196261682          136
## 38              38      0.177570093457944           90
## 39              39      0.182242990654206           87
## 40              40      0.186915887850467           74
## 41              41      0.191588785046729           84
## 42              42      0.196261682242991           87
## 43              43      0.200934579439252           78
## 44              44      0.205607476635514           75
## 45              45      0.210280373831776          105
## 46              46      0.214953271028037           78
## 47              47      0.219626168224299           67
## 48              48      0.224299065420561           98
## 49              49      0.228971962616822           70
## 50              50      0.233644859813084           71
## 51              51      0.238317757009346           77
## 52              52      0.242990654205607           91
## 53              53      0.247663551401869           84
## 54              54      0.252336448598131           82
## 55              55      0.257009345794392           75
## 56              56      0.261682242990654           60
## 57              57      0.266355140186916           60
## 58              58      0.271028037383178           63
## 59              59      0.275700934579439           57
## 60              60      0.280373831775701           72
## 61              61      0.285046728971963           48
## 62              62      0.289719626168224           50
## 63              63      0.294392523364486           43
## 64              64      0.299065420560748           61
## 65              65      0.303738317757009           52
## 66              66      0.308411214953271           46
## 67              67      0.313084112149533           44
## 68              68      0.317757009345794           53
## 69              69      0.322429906542056           46
## 70              70      0.327102803738318           43
## 71              71      0.331775700934579           41
## 72              72      0.336448598130841           54
## 73              73      0.341121495327103           51
## 74              74      0.345794392523364           48
## 75              75      0.350467289719626           42
## 76              76      0.355140186915888           62
## 77              77       0.35981308411215           44
## 78              78      0.364485981308411           62
## 79              79      0.369158878504673           55
## 80              80      0.373831775700935           43
## 81           81      0.378504672897196          62
## 82           82      0.383177570093458          48
## 83           83       0.38785046728972          59
## 84           84      0.392523364485981          41
## 85           85      0.397196261682243          39
## 86           86      0.401869158878505          50
## 87           87      0.406542056074766          42
## 88           88      0.411214953271028          35
## 89           89       0.41588785046729          34
## 90           90      0.420560747663551          43
## 91           91      0.425233644859813          38
## 92           92      0.429906542056075          54
## 93           93      0.434579439252336          43
## 94           94      0.439252336448598          35
## 95           95       0.44392523364486          48
## 96           96      0.448598130841121          31
## 97           97      0.453271028037383          45
## 98           98      0.457943925233645          45
## 99           99      0.462616822429907          34
## 100         100      0.467289719626168          35
## 101         101       0.47196261682243          41
## 102         102      0.476635514018692          31
## 103         103      0.481308411214953          49
## 104         104      0.485981308411215          35
## 105         105      0.490654205607477          28
## 106         106      0.495327102803738          33
## 107         107                    0.5          33
```

Extra Credit: choose ACB

```
# read the vcf file
vcf.acb <- readVcf("chr7ACB_6000000_8000000.vcf","hg19")
class(vcf.acb)
```

```
## [1] "CollapsedVCF"
## attr(,"package")
## [1] "VariantAnnotation"
```

```
dim(vcf.acb) # 73783 refSNP, 92 samples
```

```
## [1] 73783    92
```

```
snv.acb <- isSNV(vcf.acb)
sum(snv.acb) # 70843 SNVs
```

```
## [1] 70843
```

```
# get a logical vector with biallelic SNPs only
bi.snv.acb <- isSNV(vcf.acb,singleAltOnly=TRUE)
# 'singleAltOnly': single alternate allele
sum(bi.snv.acb) # 70843 biallelic SNPs
```

```
## [1] 70843
```

```
vcf.snps_only.acb <- vcf.acb[bi.snv.acb,]
```

```r
acb.geno <- geno(vcf.snps_only.acb)

acb.gt.mat <- acb.geno[['GT']]
head(acb.gt.mat[,1:10])
```

```
##             HG01879 HG01880 HG01882 HG01883 HG01885 HG01886 HG01889 HG01890
## rs62454735  "0|1"   "0|0"   "1|0"   "0|1"   "1|0"   "0|0"   "0|0"   "1|1"
## rs147838625 "0|1"   "0|0"   "1|0"   "0|1"   "1|0"   "0|0"   "0|0"   "1|1"
## rs573650620 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
## rs201408132 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
## rs553135573 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
## rs577832347 "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"   "0|0"
##             HG01894 HG01896
## rs62454735  "0|0"   "1|1"
## rs147838625 "0|0"   "1|1"
## rs573650620 "0|0"   "0|0"
## rs201408132 "1|0"   "0|0"
## rs553135573 "0|0"   "0|0"
## rs577832347 "0|0"   "0|0"
```

```r
dim(acb.gt.mat)
```

```
## [1] 70843    92
```

```r
acb.maf.tbl_df <- acb.gt.mat %>% as.data.frame %>%
  as.data.frame(stringsAsFactors = F) %>%
  rownames_to_column(var = "id") %>%
  as_tibble %>%
  pivot_longer(cols = -id,
               names_to = "sample",
               values_to = "genotype") %>%
  mutate(alt_count = case_when(
    genotype == "0/0" ~ 0,
    genotype == "0|0" ~ 0,
    genotype == "0/1" ~ 1,
    genotype == "1/0" ~ 1,
    genotype == "0|1" ~ 1,
    genotype == "1|0" ~ 1,
    genotype == "1/1" ~ 2,
    genotype == "1|1" ~ 2,
    TRUE ~ NA_real_
  )) %>%
  mutate(ref_count = 2 - alt_count) %>%
  group_by(id) %>%
  summarise(tot_alt_count = sum(alt_count),
            tot_ref_count = sum(ref_count)) %>%
  mutate(tot_minor = case_when(tot_alt_count <= tot_ref_count ~ tot_alt_count,
                               tot_ref_count < tot_alt_count ~ tot_ref_count,
                               TRUE ~ NA_real_)) %>%
  mutate(maf = tot_minor / (tot_alt_count + tot_ref_count))
```

```r
# filter refSNPs which only have "1/1" or "0/0" in all samples
# If only have "1/1" or "0/0" in all samples,
# the tot_alt_count or tot_ref_count will both equal to 92*2
index_only_alt_or_ref = which((acb.maf.tbl_df$tot_ref_count==92*2)|
                               (acb.maf.tbl_df$tot_alt_count==92*2))
acb.without.only = acb.maf.tbl_df[-index_only_alt_or_ref,]
```

```
level_of_maf = levels(factor(acb.without.only$maf))

count_of_every_maf=numeric()

for (i in 1:length(level_of_maf)){
  count_of_every_maf[i] = sum(acb.without.only$maf == level_of_maf[i])
}
names(count_of_every_maf) = level_of_maf
count_of_every_maf
```

```
## 0.00543478260869565  0.0108695652173913   0.016304347826087  0.0217391304347826
##                4703                2044                1214                 911
##  0.0271739130434783  0.0326086956521739  0.0380434782608696  0.0434782608695652
##                 928                 634                 420                 422
##  0.0489130434782609  0.0543478260869565  0.0597826086956522  0.0652173913043478
##                 430                 364                 345                 366
##  0.0706521739130435  0.0760869565217391  0.0815217391304348  0.0869565217391304
##                 361                 323                 315                 360
##  0.0923913043478261  0.0978260869565217   0.103260869565217   0.108695652173913
##                 278                 226                 287                 218
##   0.114130434782609   0.119565217391304               0.125   0.130434782608696
##                 177                 124                 135                 107
##   0.135869565217391   0.141304347826087   0.146739130434783   0.152173913043478
##                 123                 138                 137                 155
##   0.157608695652174    0.16304347826087   0.168478260869565   0.173913043478261
##                 140                 169                  92                 122
##   0.179347826086957   0.184782608695652   0.190217391304348   0.195652173913043
##                 120                  95                 106                 112
##   0.201086956521739   0.206521739130435    0.21195652173913   0.217391304347826
##                 103                  81                 138                  93
##   0.222826086956522   0.228260869565217   0.233695652173913   0.239130434782609
##                 130                  88                  70                  70
##   0.244565217391304                0.25   0.255434782608696   0.260869565217391
##                  63                  73                  52                  65
##   0.266304347826087   0.271739130434783   0.277173913043478   0.282608695652174
##                  85                  65                  84                  71
##    0.28804347826087   0.293478260869565   0.298913043478261   0.304347826086957
##                  48                  58                  60                  46
##   0.309782608695652   0.315217391304348   0.320652173913043   0.326086956521739
##                  76                  67                  47                  69
##   0.331521739130435    0.33695652173913   0.342391304347826   0.347826086956522
##                  59                  66                  62                  89
##   0.353260869565217   0.358695652173913   0.364130434782609   0.369565217391304
##                  56                  45                  52                  48
##               0.375   0.380434782608696   0.385869565217391   0.391304347826087
##                  33                  31                  47                  48
##   0.396739130434783   0.402173913043478   0.407608695652174    0.41304347826087
##                  39                  60                  54                  54
##   0.418478260869565   0.423913043478261   0.429347826086957   0.434782608695652
##                  62                  46                  59                  51
##   0.440217391304348   0.445652173913043   0.451086956521739   0.456521739130435
##                  42                  47                  67                  49
##    0.46195652173913   0.467391304347826   0.472826086956522   0.478260869565217
##                  44                  46                  46                  30
##   0.483695652173913   0.489130434782609   0.494565217391304                 0.5
##                  49                  40                  65                  14
```

```
level_of_every_minor = levels(factor(acb.without.only$tot_minor))

count_of_every_minor=numeric()

for (i in 1:length(level_of_every_minor)){
  count_of_every_minor[i] = sum(acb.without.only$tot_minor == level_of_every_minor[i])
}
names(count_of_every_minor) = level_of_every_minor
count_of_every_minor
```

```
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 4703 2044 1214  911  928  634  420  422  430  364  345  366  361  323  315  360
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
##  278  226  287  218  177  124  135  107  123  138  137  155  140  169   92  122
##   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48
##  120   95  106  112  103   81  138   93  130   88   70   70   63   73   52   65
##   49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64
##   85   65   84   71   48   58   60   46   76   67   47   69   59   66   62   89
##   65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80
##   56   45   52   48   33   31   47   48   39   60   54   54   62   46   59   51
##   81   82   83   84   85   86   87   88   89   90   91   92
##   42   47   67   49   44   46   46   30   49   40   65   14
```

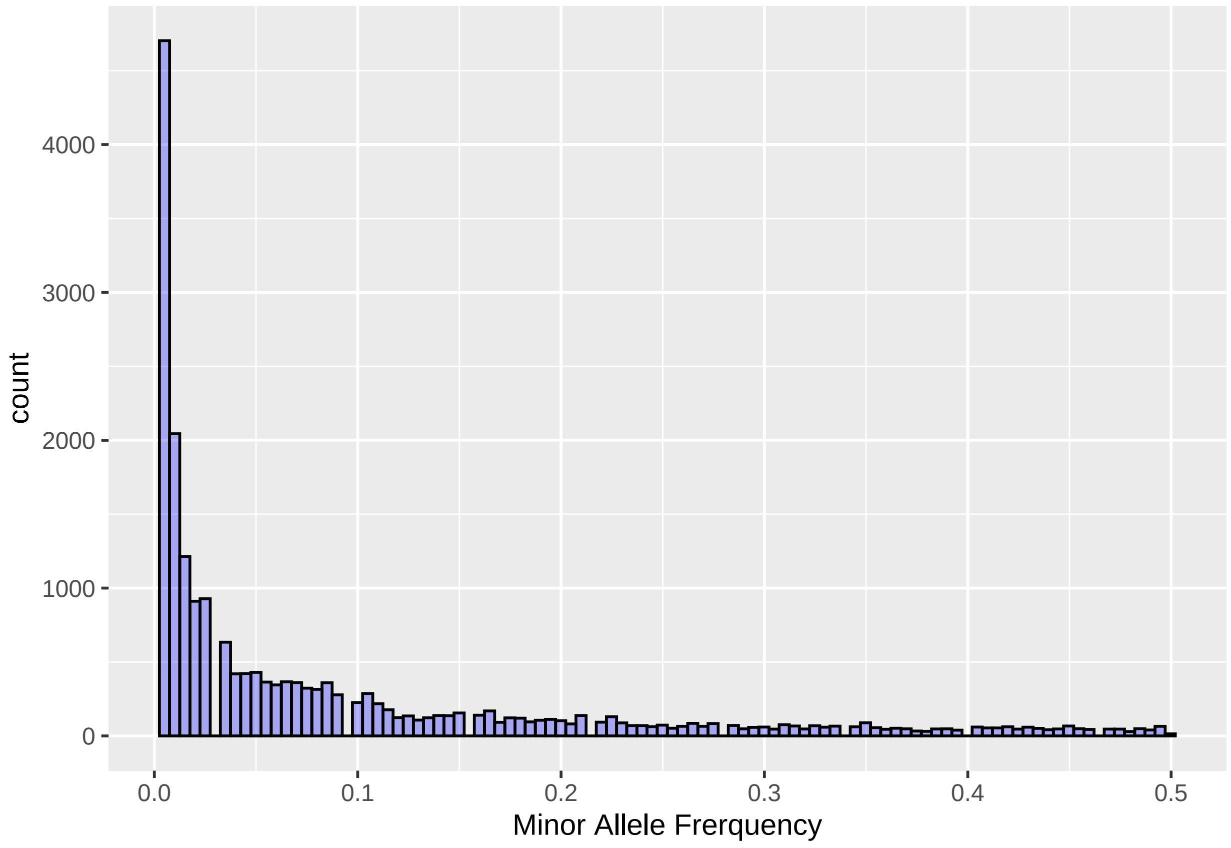
```
which(count_of_every_minor!=count_of_every_maf)
```

```
## named integer(0)
```

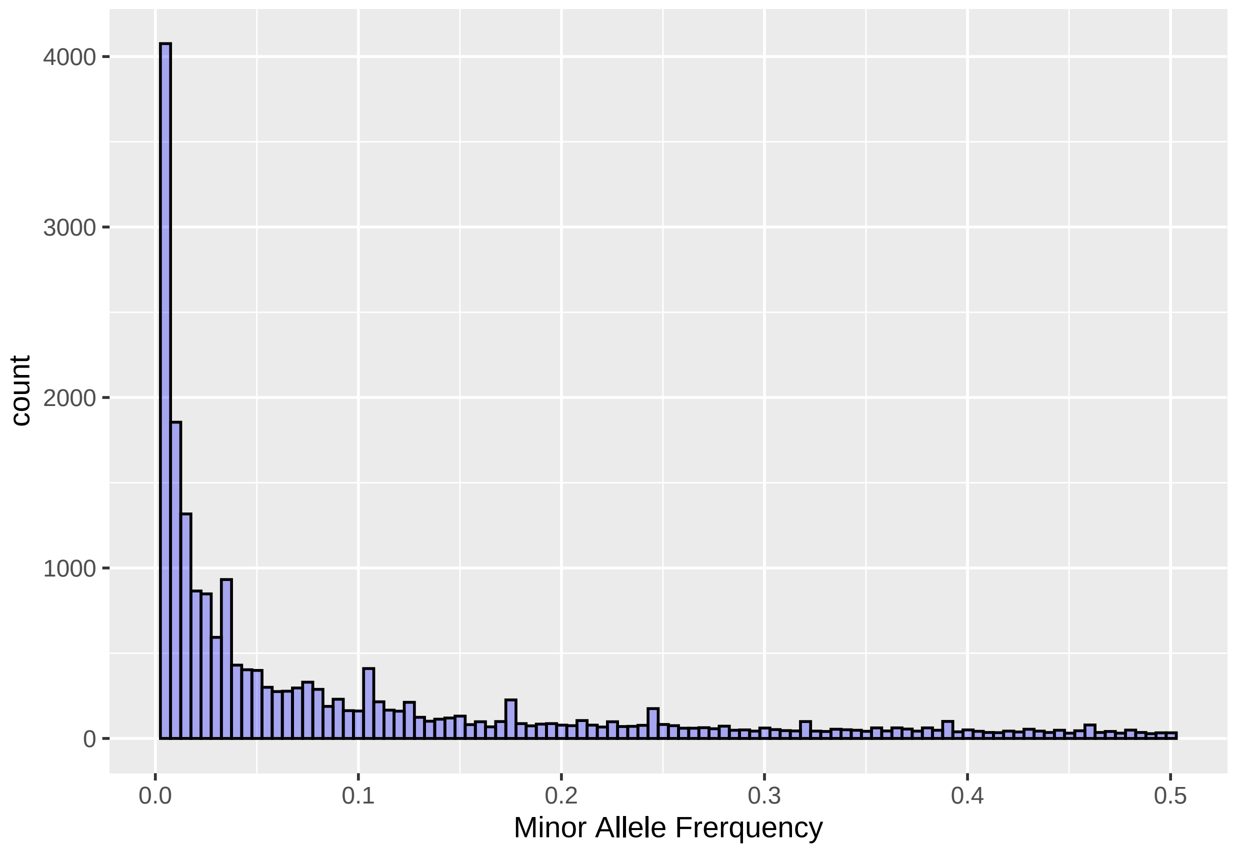
```
count_tb_ACB=data.frame(total_minor=names(count_of_every_minor),
                        minor_allele_frequency=names(count_of_every_maf),
                        count_number=count_of_every_minor)
count_tb_ACB
```

```
##    total_minor minor_allele_frequency count_number
## 1            1    0.00543478260869565         4703
## 2            2     0.0108695652173913         2044
## 3            3      0.016304347826087         1214
## 4            4     0.0217391304347826          911
## 5            5     0.0271739130434783          928
## 6            6     0.0326086956521739          634
## 7            7     0.0380434782608696          420
## 8            8     0.0434782608695652          422
## 9            9     0.0489130434782609          430
## 10          10     0.0543478260869565          364
## 11          11     0.0597826086956522          345
## 12          12     0.0652173913043478          366
## 13          13     0.0706521739130435          361
## 14          14     0.0760869565217391          323
## 15          15     0.0815217391304348          315
## 16          16     0.0869565217391304          360
## 17          17     0.0923913043478261          278
## 18           18     0.0978260869565217          226
## 19           19      0.103260869565217          287
## 20           20      0.108695652173913          218
## 21           21      0.114130434782609          177
## 22           22      0.119565217391304          124
## 23           23                  0.125          135
## 24           24      0.130434782608696          107
## 25           25      0.135869565217391          123
## 26           26      0.141304347826087          138
## 27           27      0.146739130434783          137
## 28           28      0.152173913043478          155
## 29           29      0.157608695652174          140
## 30           30       0.16304347826087          169
## 31           31      0.168478260869565           92
## 32           32      0.173913043478261          122
## 33           33      0.179347826086957          120
## 34           34      0.184782608695652           95
## 35           35      0.190217391304348          106
## 36           36      0.195652173913043          112
## 37           37      0.201086956521739          103
## 38           38      0.206521739130435           81
## 39           39       0.21195652173913          138
## 40           40      0.217391304347826           93
## 41           41      0.222826086956522          130
## 42           42      0.228260869565217           88
## 43           43      0.233695652173913           70
## 44           44      0.239130434782609           70
## 45           45      0.244565217391304           63
## 46           46                   0.25           73
## 47           47      0.255434782608696           52
## 48           48      0.260869565217391           65
## 49           49      0.266304347826087           85
## 50           50      0.271739130434783           65
## 51           51      0.277173913043478           84
## 52           52      0.282608695652174           71
## 53           53       0.28804347826087           48
## 54           54      0.293478260869565           58
## 55           55      0.298913043478261           60
## 56           56      0.304347826086957           46
## 57           57      0.309782608695652           76
## 58           58      0.315217391304348           67
## 59           59      0.320652173913043           47
## 60           60      0.326086956521739           69
## 61           61      0.331521739130435           59
## 62           62       0.33695652173913           66
## 63           63      0.342391304347826           62
## 64           64      0.347826086956522           89
## 65           65      0.353260869565217           56
## 66           66      0.358695652173913           45
## 67           67      0.364130434782609           52
## 68           68      0.369565217391304           48
## 69           69                  0.375           33
## 70           70      0.380434782608696           31
## 71           71      0.385869565217391           47
## 72          72      0.391304347826087          48
## 73          73      0.396739130434783          39
## 74          74      0.402173913043478          60
## 75          75      0.407608695652174          54
## 76          76       0.41304347826087          54
## 77          77      0.418478260869565          62
## 78          78      0.423913043478261          46
## 79          79      0.429347826086957          59
## 80          80      0.434782608695652          51
## 81          81      0.440217391304348          42
## 82          82      0.445652173913043          47
## 83          83      0.451086956521739          67
## 84          84      0.456521739130435          49
## 85          85       0.46195652173913          44
## 86          86      0.467391304347826          46
## 87          87      0.472826086956522          46
## 88          88      0.478260869565217          30
## 89          89      0.483695652173913          49
## 90          90      0.489130434782609          40
## 91          91      0.494565217391304          65
## 92          92                    0.5          14
```

```
library(ggplot2)
ggplot(data=acb.without.only,aes(maf))+
  geom_histogram(bins =100,fill="blue",color="black",alpha=0.3)+
  xlab("Minor Allele Frerquency") -> SFS.ACB
SFS.ACB
```



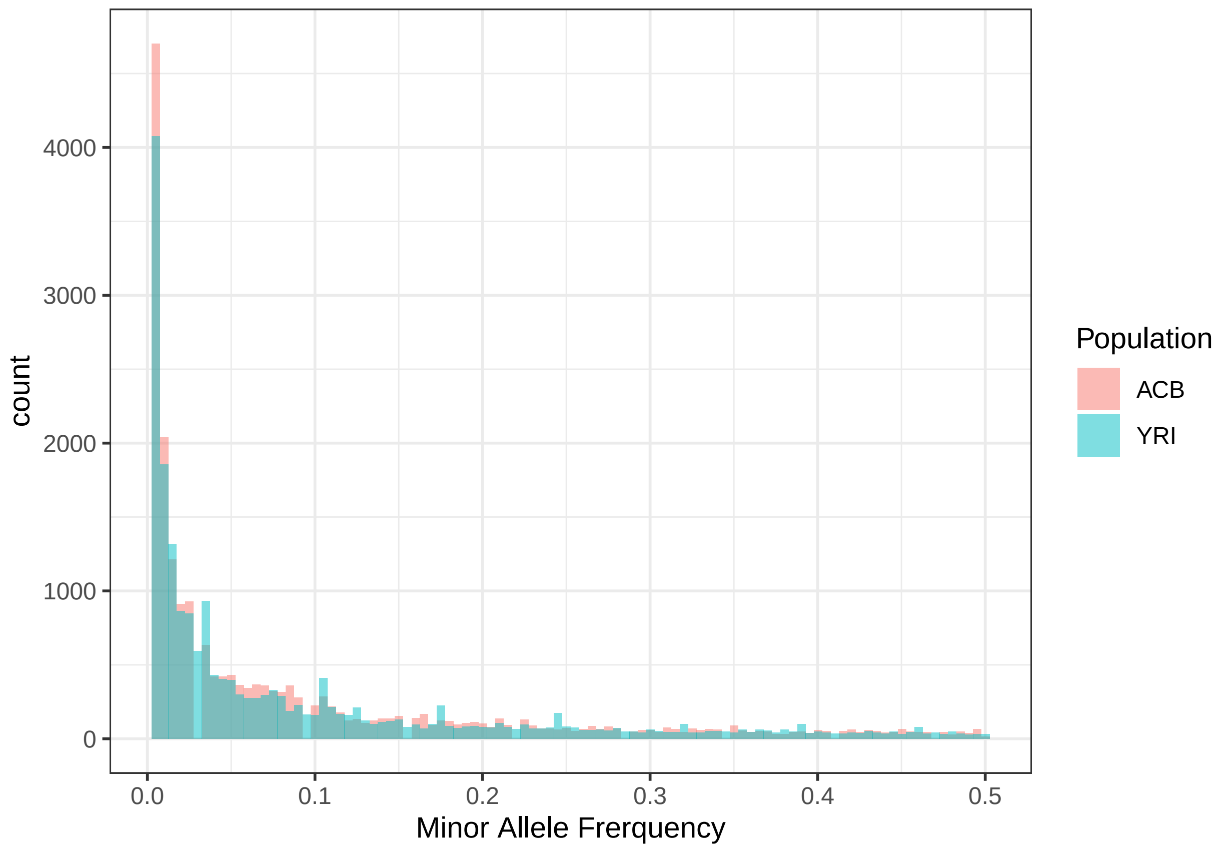
```
SFS.Yoruban
```



Generate an SFS for comparison to the Yoruban data

```
YRI.df <- data.frame(maf=geno.chr7$maf,
                     Population=rep("YRI",length(geno.chr7$maf)))
ACB.df <- data.frame(maf=acb.without.only$maf,
                     Population=rep("ACB",length(acb.without.only$maf)))
hist.df <- rbind(YRI.df,ACB.df)
```

```
ggplot(hist.df,aes(x=maf,fill=Population)) +
 geom_histogram(bins =100,alpha=0.5, position="identity")+
  xlab("Minor Allele Frerquency")+ theme_bw()
```


count
4000
3000
2000
1000
0
0.0
0.1
0.2
0.3
0.4
0.5
Minor Allele Frerquency
Population
ACB
YRI