assignment_eQTL

2023-09-29

Name: Abdullah Faqih Al Mubarok

KU id: vpx267

Part 1: Understanding The Basic

```
design_data <- read.table("design.tab", header=TRUE, sep="\t") |>
  rownames_to_column() |>  as_tibble()
sub_expr_data <- read.table("sub_expr.tab", header=TRUE, sep="\t") |>
  rownames_to_column() |>
  as_tibble() |>
  rename(gene=rowname)
sub_geno_data <- read.table("sub_geno.tab", header=TRUE, sep="\t") |>
  rownames_to_column() |>
  as_tibble() |>
  rename(snp=rowname)
```

Task 1

1. What do the -1,0,1,2 values represent in the sub geno.tab file?

```
head(sub_geno_data)
```

```
## # A tibble: 6 x 463
##   snp    HG00096 HG00097 HG00099 HG00100 HG00101 HG00102 HG00103 HG00104 HG00105
##   <chr>    <int>   <int>   <int>   <int>   <int>   <int>   <int>   <int>   <int>
## 1 snp_2~       0       0       0       0       0       0       0       0       0
## 2 snp_2~       0       1       0       0       1       0       1       0       0
## 3 snp_2~       0       0       0       1       0       0       0       0       0
## 4 snp_2~       0       0       0       0       0       0       0       0       0
## 5 snp_2~       0       0       0       0       0       0       0       0       0
## 6 snp_2~       2       1       0       2       2       2       2       0       0
## # i 453 more variables: HG00106 <int>, HG00108 <int>, HG00109 <int>,
## #   HG00110 <int>, HG00111 <int>, HG00112 <int>, HG00114 <int>, HG00115 <int>,
## #   HG00116 <int>, HG00117 <int>, HG00118 <int>, HG00119 <int>, HG00120 <int>,
## #   HG00121 <int>, HG00122 <int>, HG00123 <int>, HG00124 <int>, HG00125 <int>,
## #   HG00126 <int>, HG00127 <int>, HG00128 <int>, HG00129 <int>, HG00130 <int>,
## #   HG00131 <int>, HG00132 <int>, HG00133 <int>, HG00134 <int>, HG00135 <int>,
## #   HG00136 <int>, HG00137 <int>, HG00138 <int>, HG00139 <int>, ...
```

The 0 from the file should represent the homozygous major (AA), 1 represents the heterozygous (Aa) and 2 come from the homozygous minor (aa). In addition, -1 should represent unknown genotype.

2. What is stored in the sub expr.tab file and what has been done with this data

```
head(sub_expr_data)
```

```
## # A tibble: 6 x 463
##   gene         HG00096 HG00097 HG00099 HG00100 HG00101  HG00102 HG00103  HG00104
##   <chr>          <dbl>   <dbl>   <dbl>   <dbl>   <dbl>    <dbl>   <dbl>    <dbl>
## 1 ENSG0000018~  3.39    3.14    1.76   4.64e+0  3.32    3.48     3.22    3.31e+0
## 2 ENSG0000020~  0.137   0.0649  0.0707 1.35e-1  0.0850  0.0807   0.0790  1.69e-1
## 3 ENSG0000006~ 21.8    27.1    18.1    2.91e+1 28.9    24.3     28.4     2.71e+1
## 4 ENSG0000024~  0.0500  0.243   0.0517 7.62e-3  0.130  -0.00571  0.0241 -6.06e-4
## 5 ENSG0000023~  0.0554  0.102   0.0919 1.22e-1  0.139   0.493    0.0445  6.04e-2
## 6 ENSG0000010~  5.35    4.01    2.91   3.71e+0  6.08    4.09     7.15    5.17e+0
## # i 454 more variables: HG00105 <dbl>, HG00106 <dbl>, HG00108 <dbl>,
## #   HG00109 <dbl>, HG00110 <dbl>, HG00111 <dbl>, HG00112 <dbl>, HG00114 <dbl>,
## #   HG00115 <dbl>, HG00116 <dbl>, HG00117 <dbl>, HG00118 <dbl>, HG00119 <dbl>,
## #   HG00120 <dbl>, HG00121 <dbl>, HG00122 <dbl>, HG00123 <dbl>, HG00124 <dbl>,
## #   HG00125 <dbl>, HG00126 <dbl>, HG00127 <dbl>, HG00128 <dbl>, HG00129 <dbl>,
## #   HG00130 <dbl>, HG00131 <dbl>, HG00132 <dbl>, HG00133 <dbl>, HG00134 <dbl>,
## #   HG00135 <dbl>, HG00136 <dbl>, HG00137 <dbl>, HG00138 <dbl>, ...
```

The file should contain the normalized expression level of genes (rows) from several samples (columns)

3. What information is stored in the design.txt file?

```
design_data
```

```
## # A tibble: 462 x 8
##    rowname Source.Name Comment.ENA_SAMPLE. Characteristics.Organism.
##    <chr>   <chr>       <chr>               <chr>
##  1 HG00096 HG00096     ERS185276           Homo sapiens
##  2 HG00097 HG00097     ERS185206           Homo sapiens
##  3 HG00099 HG00099     ERS185128           Homo sapiens
##  4 HG00100 HG00100     ERS185086           Homo sapiens
##  5 HG00101 HG00101     ERS185085           Homo sapiens
##  6 HG00102 HG00102     ERS185453           Homo sapiens
##  7 HG00103 HG00103     ERS185490           Homo sapiens
##  8 HG00104 HG00104     ERS185300           Homo sapiens
##  9 HG00105 HG00105     ERS185272           Homo sapiens
## 10 HG00106 HG00106     ERS185411           Homo sapiens
## # i 452 more rows
## # i 4 more variables: Characteristics.Strain. <chr>,
## #   Characteristics.population. <chr>, Comment.1000g.Phase1.Genotypes. <int>,
## #   Factor.Value.laboratory. <int>
```

That file contains the metadata of each samples (rows) such as their species name, population group, and strain type

Task 2

1. Calculate the number of missing genotypes for each SNP across all individuals.

```
sub_geno_data <- sub_geno_data |>
  rowwise(snp) |>
  mutate(missing_snps =  sum(c_across(where(is.numeric))==-1)) |>
  ungroup()
sub_geno_data |> select(snp, missing_snps) |>  arrange(desc(missing_snps))
```

```
## # A tibble: 39 x 2
##    snp             missing_snps
##    <chr>                  <int>
```

```
##  1 snp_22_33671358           41
##  2 snp_22_47528038           41
##  3 snp_22_42606894           41
##  4 snp_22_30772686            0
##  5 snp_22_34965577            0
##  6 snp_22_49436707            0
##  7 snp_22_30631851            0
##  8 snp_22_46215888            0
##  9 snp_22_34153853            0
## 10 snp_22_21970216            0
## # i 29 more rows
```

From the calculation, we can see that there are three SNPs that have missing genotypes: snp 22 33671358,snp 22 47528038, snp 22 42606894. Each of them have 41 missing genotypes

2. Calculate the minor allele frequency (MAF) for all SNPs across all individuals. (hint: divide mean of genotypes by 2)

```
sub_geno_data <- sub_geno_data |>
  rowwise(snp) |>
  mutate(MAF = mean(c_across(-c(missing_snps))[c_across(-c(missing_snps)) != -1])/2) |>
  ungroup()

sub_geno_data |>
  select(snp, MAF)
```

```
## # A tibble: 39 x 2
##    snp                 MAF
##    <chr>             <dbl>
##  1 snp_22_30772686 0.0974
##  2 snp_22_34965577 0.135
##  3 snp_22_49436707 0.0833
##  4 snp_22_30631851 0
##  5 snp_22_46215888 0.00108
##  6 snp_22_34153853 0.801
##  7 snp_22_33671358 0
##  8 snp_22_21970216 0.0671
##  9 snp_22_47528038 0.00119
## 10 snp_22_48286671 0.0574
## # i 29 more rows
```

3. Filter out SNPs that have missing genotypes or a MAF<0.05 and use the filtered snps for the rest of the exercise.

```
sub_geno_data_filtered <- sub_geno_data |>
  filter(missing_snps==0 & MAF>=0.05)

dim(sub_geno_data_filtered)
```

```
## [1]  32 465
```

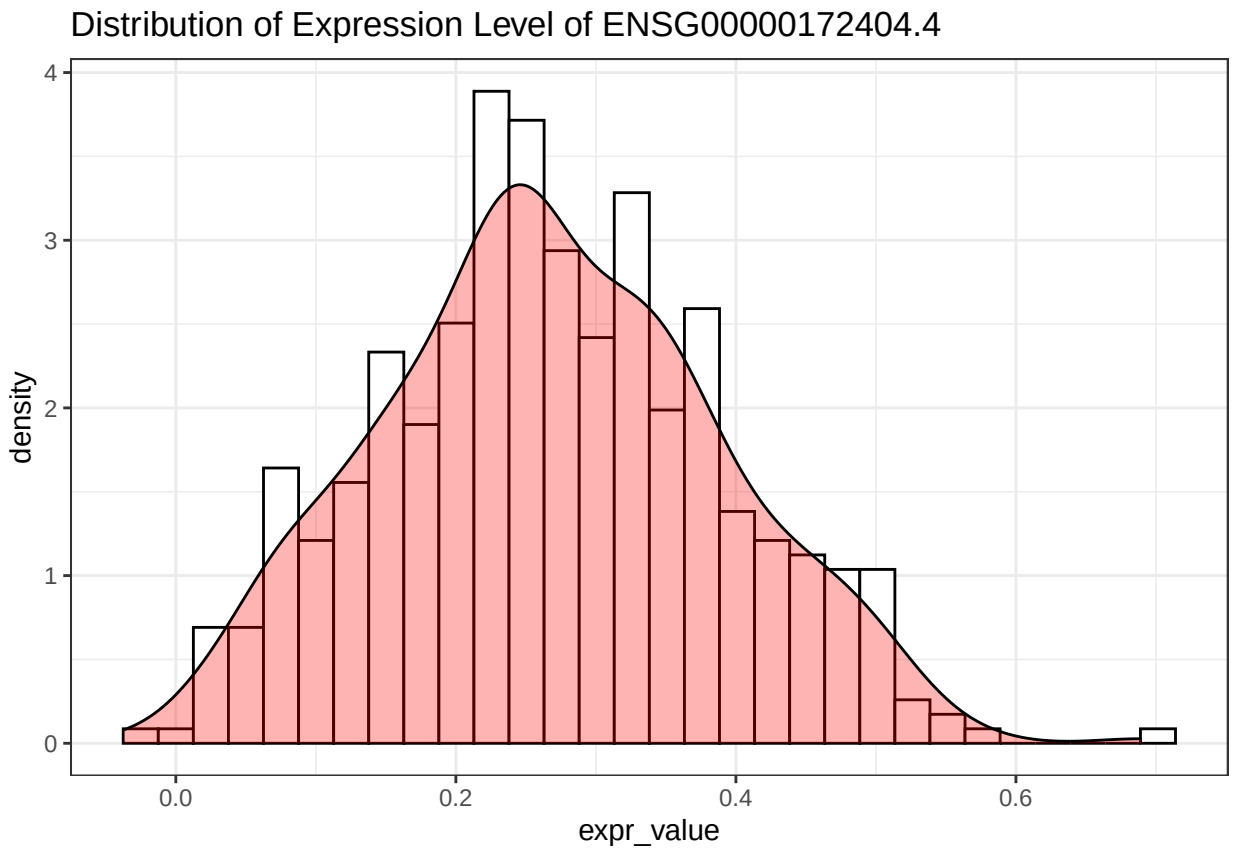
Task 3

1. Plot the distribution of expression levels across all samples for the ENSG00000172404.4 gene

```
selected_gene_expr <- sub_expr_data |>
  filter(gene=="ENSG00000172404.4") |>
  pivot_longer(cols=-c(gene), names_to = "sample_id", values_to = "expr_value")
```

```
ggplot(data = selected_gene_expr) +
  geom_histogram(aes(x = expr_value, y=..density..),      # Histogram with density instead of count on
                 colour="black", fill="white") +
  geom_density(mapping = aes(x = expr_value), fill = "red", alpha = 0.3) +
  labs(title = "Distribution of Expression Level of ENSG00000172404.4") +
  theme_bw()
```

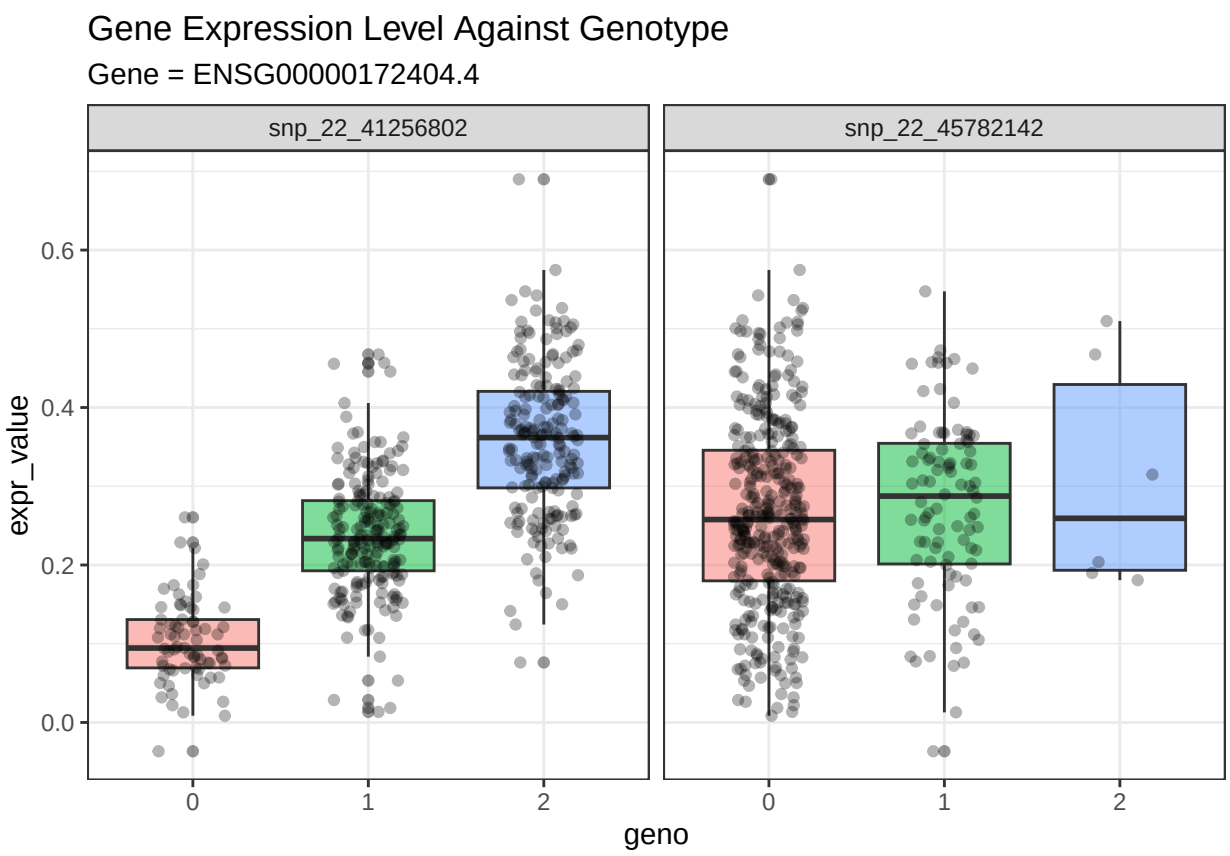
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2. Plot the expression levels of ENSG00000172404.4 against the genotypes of snp_22_41256802 and snp_22_45782142

```
selected_snps_geno <- sub_geno_data_filtered |>
  select(-c("MAF", "missing_snps")) |>
  filter(snp %in% c("snp_22_41256802", "snp_22_45782142")) |>
  pivot_longer(cols = -snp, names_to = "sample_id", values_to = "geno")

selected_expr_snps_geno <- selected_snps_geno |>
  left_join(selected_gene_expr |> select(!gene), by = join_by(sample_id)) |>
  arrange(sample_id, snp)

ggplot(data = selected_expr_snps_geno,
       mapping = aes(x = as.factor(geno), y = expr_value, fill = as.factor(geno))) +
  geom_boxplot(alpha = 0.5,
       show.legend = FALSE) +
  geom_jitter(position = position_jitter(0.2), alpha = 0.3,
```

```
      show.legend = FALSE) +
  labs(title = "Gene Expression Level Against Genotype",
       subtitle = "Gene = ENSG00000172404.4",
       x = "geno") +
  facet_grid(cols = vars(snp), scales = "free", space = "free") +
  theme_bw()
```



Task 4

1. Linear regression for snp_22_41256802 on ENSG00000172404.4

```
pred_22_41256802 <- lm(data = selected_expr_snps_geno |> filter(snp=="snp_22_41256802"),
   formula = expr_value ~ geno)
print(summary(pred_22_41256802))
```

```
##
## Call:
## lm(formula = expr_value ~ geno, data = filter(selected_expr_snps_geno,
##     snp == "snp_22_41256802"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28348 -0.04934 -0.00143  0.04950  0.33024
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.109393   0.007824   13.98   <2e-16 ***
```

```
## geno        0.125135   0.005391   23.21   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08216 on 460 degrees of freedom
## Multiple R-squared:  0.5394, Adjusted R-squared:  0.5384
## F-statistic: 538.7 on 1 and 460 DF,  p-value: < 2.2e-16
```

2. Linear regression for snp_22_45782142 on ENSG00000172404.4

```
pred_22_45782142 <- lm(data = selected_expr_snps_geno |> filter(snp=="snp_22_45782142"),
   formula = expr_value ~ geno)
print(summary(pred_22_45782142))
```

```
##
## Call:
## lm(formula = expr_value ~ geno, data = filter(selected_expr_snps_geno,
##     snp == "snp_22_45782142"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.31360 -0.08515 -0.00588  0.07998  0.42486
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.265040   0.006332  41.856   <2e-16 ***
## geno        0.011987   0.012425   0.965    0.335
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1209 on 460 degrees of freedom
## Multiple R-squared:  0.002019,   Adjusted R-squared:  -0.0001502
## F-statistic: 0.9308 on 1 and 460 DF,  p-value: 0.3352
```

Question 4

1. Explain the results. What are the important values to look at and what do they tell you?

The important values that need to look for are the estimated effect size (geno) and its p-value from the two-sided t-test with the following hypothesis:

$$H_0 : \beta_{geno} = 0$$
$$H_A : \beta_{geno} \neq 0$$

From the results above, the p-value from the estimated effect size of snp_22_41256802 is very small (<2e-16). If we set our threshold to be 0.05, we can confidently reject the null hypothesis. Thus, there is enough evident that the effect size of additional minor allele on snp_22_41256802 on is not zero for the expression of ENSG00000172404.4 gene. We can see that the estimated effect size is 0.125135.

On the other hand, the p-value from the estimated effect size of additional minor allele on snp_22_45782142 for the expression of ENSG00000172404.4 gene is not significant using the same threshold (0.05). Therefore, we fail to reject the null that stated the effect size is zero.

Task 5

Do a linear regression for snp_22_43336231 on ENSG00000100266.11

1. All individuals together

```
selected_gene_expr2_all <- sub_expr_data |>
  filter(gene=="ENSG00000100266.11") |>
  pivot_longer(cols=-c(gene), names_to = "sample_id", values_to = "expr_value")

selected_snps_geno2 <- sub_geno_data_filtered |>
  select(-c("MAF", "missing_snps")) |>
  filter(snp %in% c("snp_22_43336231")) |>
  pivot_longer(cols = -snp, names_to = "sample_id", values_to = "geno")

selected_expr_snps_geno2 <- selected_snps_geno2 |>
  left_join(selected_gene_expr2_all |> select(!gene), by = join_by(sample_id)) |>
  left_join(design_data |> select(Source.Name,Characteristics.population.),
            by = join_by(sample_id==Source.Name)) |>
  arrange(sample_id, snp)

pred_22_43336231_all <- lm(data = selected_expr_snps_geno2,
   formula = expr_value ~ geno)
print(summary(pred_22_43336231_all))
```

```
##
## Call:
## lm(formula = expr_value ~ geno, data = selected_expr_snps_geno2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.367  -5.791  -0.774   4.563  41.890
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  23.8641     0.5297   45.05  < 2e-16 ***
## geno          3.3238     0.6121    5.43 9.13e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.746 on 460 degrees of freedom
## Multiple R-squared:  0.06024,    Adjusted R-squared:  0.0582
## F-statistic: 29.49 on 1 and 460 DF,  p-value: 9.131e-08
```

2. Separately for african and non-african individuals

```
#Africa individuals YRI Yoruba in Ibadan, Nigera
pred_22_43336231_YRI <- lm(data = selected_expr_snps_geno2
                           |> filter(Characteristics.population.=="YRI"),
   formula = expr_value ~ geno)
print(summary(pred_22_43336231_YRI))
```

```
##
## Call:
## lm(formula = expr_value ~ geno, data = filter(selected_expr_snps_geno2,
##     Characteristics.population. == "YRI"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.0137  -4.1504  -0.3292   5.0336  19.5839
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  26.3095     0.7353  35.781   <2e-16 ***
## geno         -0.7181     2.8319  -0.254      0.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.699 on 87 degrees of freedom
## Multiple R-squared:  0.0007385,  Adjusted R-squared:  -0.01075
## F-statistic: 0.0643 on 1 and 87 DF,  p-value: 0.8004
```

```
# Non Africa individuals
pred_22_43336231_non_YRI <- lm(data = selected_expr_snps_geno2
                          |> filter(Characteristics.population.!="YRI"),
   formula = expr_value ~ geno)
print(summary(pred_22_43336231_non_YRI))
```

```
##
## Call:
## lm(formula = expr_value ~ geno, data = filter(selected_expr_snps_geno2,
##     Characteristics.population. != "YRI"))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.922  -5.727  -0.700   4.583  42.142
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.8046     0.6598  34.562  < 2e-16 ***
## geno          4.1310     0.6911   5.978 5.32e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.075 on 371 degrees of freedom
## Multiple R-squared:  0.08785,    Adjusted R-squared:  0.08539
## F-statistic: 35.73 on 1 and 371 DF,  p-value: 5.321e-09
```

Question 5

1. Is there a difference between african and non-africans? If so explain why

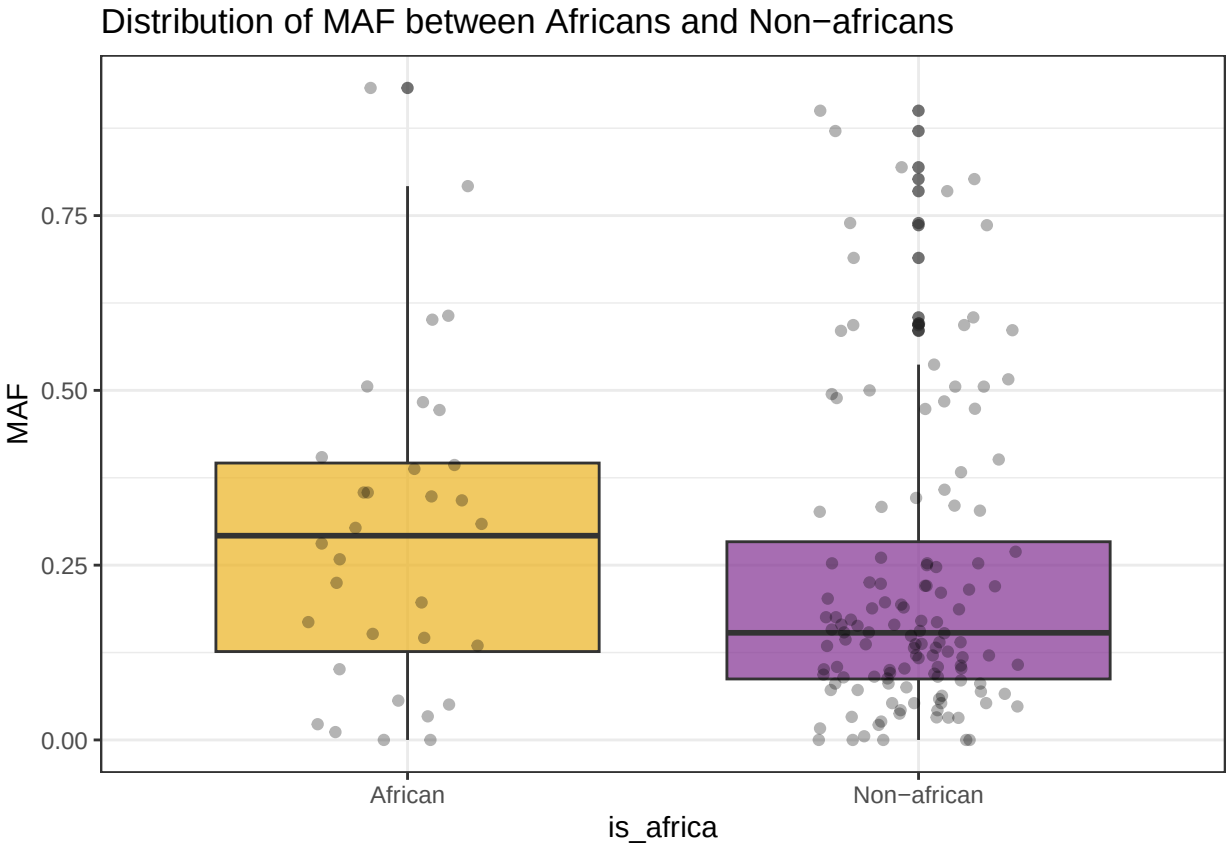
From the result above, we can see that if we regress only for the African individuals, the effect size is not statistically significant anymore at level 5%. Howver, the non-africans effect size is still statistically significant at 5% level but with higher estimated effect size compared with all individuals (4.13 vs 3.32). The difference might be due to population stratification where each population has difference allele distribution which can be described by the following graph

```
temp_geno <- sub_geno_data_filtered  |>
  select(-c("MAF", "missing_snps")) |>
  pivot_longer(cols = -snp, names_to = "sample_id", values_to = "geno") |>
  left_join(design_data |> select(Source.Name,Characteristics.population.),
            by = join_by(sample_id==Source.Name)) |>
  group_by(Characteristics.population., snp) |>
  summarise(MAF = mean(geno)/2) |>
  ungroup() |>
```

```
  mutate(is_africa = if_else(Characteristics.population.=="YRI", "African", "Non-african"))
```

```
## `summarise()` has grouped output by 'Characteristics.population.'. You can
## override using the `.groups` argument.
```

```
ggplot(data = temp_geno, mapping = aes(x = is_africa, y = MAF,
                                       fill = is_africa)) +
  geom_boxplot(show.legend = FALSE, alpha = 0.7) +
  geom_jitter(position = position_jitter(0.2), alpha = 0.3,
       show.legend = FALSE) +
  scale_fill_manual(values = c("#ecb21e", "#812e91")) +
  labs(title = "Distribution of MAF between Africans and Non-africans") +
  theme_bw()
```



From the above figure, we can see that the distribution of allele frequencies between African and Non-african based on the given data is different.

2. If the population structure is unknown, how can you still include it as a covariate?

If the information regarding the population can not be obtained, we could still use the k principal components (PCs) of the genotype data into our linear model. This is due to PCs could help us to maximize the variation of the genotype difference among the population. Therefore, our linear model could be:

$$gene\_expr = \beta_0 + \beta_1 genotype + \beta_2 PC_1 + \beta_3 PC_2 + ... + \beta_{k+1} PC_k \quad (1)$$

Task 6: Do a linear regression on 1st snp on 1st gene, 2nd snp on 2nd gene etc.

- Create a matrix containing the gene_id, snp_id, effect size, t.value and p.value.

```r
# transposing the data for looping
sub_expr_data_t <- sub_expr_data |>
  pivot_longer(cols= -1)  |>
  pivot_wider(names_from = gene,values_from = value) |>
  rename(sample_id = name)


sub_geno_data_filtered_t <- sub_geno_data_filtered |>
  select(-c("missing_snps", "MAF")) |>
  pivot_longer(cols= -1)  |>
  pivot_wider(names_from = snp,values_from = value) |>
  rename(sample_id = name)
```

```r
max_iter <- 32
result <- matrix(nrow=max_iter,ncol=5)
colnames(result) <- c("gene", "snp", "effect_size", "t_val", "p_val")

for(i in c(1:max_iter)){
  # Get gene_id
  result[i,1] <- names(sub_expr_data_t)[i+1]
  # Get snp_id
  result[i,2] <- names(sub_geno_data_filtered_t)[i+1]

  # Predict the linaer model
  pred <- lm(formula = sub_expr_data_t |> pull(i+1) ~
               sub_geno_data_filtered_t |> pull(i+1))
  # Summary coefficient
  summ_pred_coeff <- summary(pred)$coefficient

  result[i,3] <- summ_pred_coeff[2,1]
  #t_val
  result[i,4]<- summ_pred_coeff[2,3]
  #p_val
  result[i,5] <- summ_pred_coeff[2,4]
}
```

- Do a multiple testing correction on the resulting p.values using fdr.

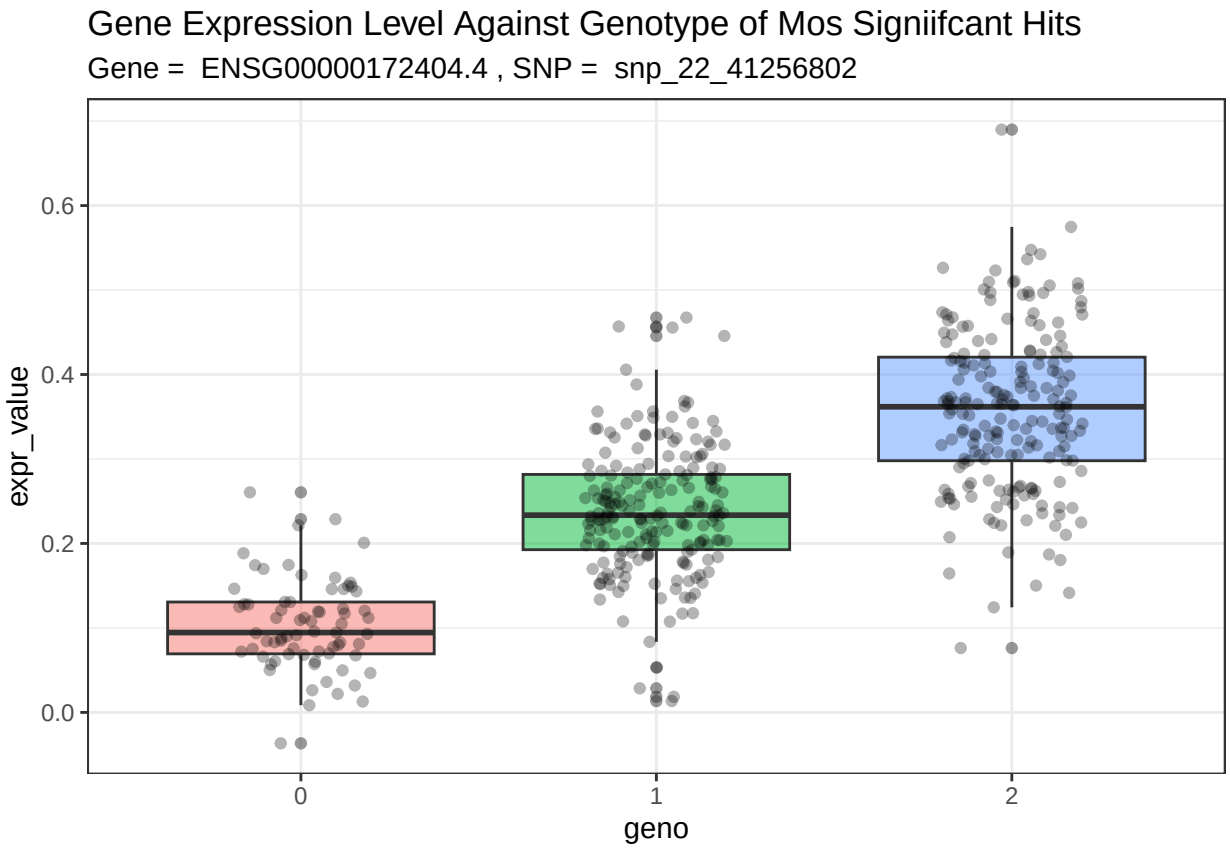
```r
adj_p_val <- p.adjust(as.double(result[,"p_val"]),
                          method = "BH")

result <- cbind(result, adj_p_val)

# reorder result
result <- result[order(adj_p_val),]

head(result)
```

```
##      gene                 snp               effect_size
## [1,] "ENSG00000172404.4"  "snp_22_41256802" "0.125134902343255"
## [2,] "ENSG00000075234.12" "snp_22_46686404" "3.02798810552512"
## [3,] "ENSG00000100266.11" "snp_22_43336231" "3.32381025093088"
## [4,] "ENSG00000205853.5"  "snp_22_32778467" "-0.0967599530934482"
## [5,] "ENSG00000128408.7"  "snp_22_45782142" "-0.276864054676497"
## [6,] "ENSG00000186716.14" "snp_22_23454881" "-0.733481310784517"
```

```
##      t_val               p_val                   adj_p_val
## [1,] "23.2098936670772"  "1.85307796052881e-79" "5.92984947369219e-78"
## [2,] "14.7511872042017"  "1.33599200955179e-40" "2.13758721528286e-39"
## [3,] "5.43023952516774"  "9.13082594165064e-08" "9.73954767109402e-07"
## [4,] "-4.3620694848842"  "1.59165591966163e-05" "0.00012733247357293"
## [5,] "-3.97431356706344" "8.19304808255882e-05" "0.000524355077283765"
## [6,] "-2.63586046871681" "0.00867601927019177"  "0.0462721027743561"
```

- Plot the most significant hit.

```
# Most significant hit
sel_gene <- result[1,"gene"]
sel_snp <- result[1,"snp"]

sel_data <- tibble(expr_value=sub_expr_data_t |> pull(sel_gene),
                   geno=sub_geno_data_filtered_t |> pull(sel_snp))

ggplot(data = sel_data,
       mapping = aes(x = as.factor(geno), y = expr_value, fill = as.factor(geno))) +
  geom_boxplot(alpha = 0.5,
       show.legend = FALSE) +
  geom_jitter(position = position_jitter(0.2), alpha = 0.3,
       show.legend = FALSE) +
  labs(title = "Gene Expression Level Against Genotype of Mos Signiifcant Hits",
       subtitle = paste("Gene = ", sel_gene, ", SNP = ", sel_snp),
       x = "geno") +
  theme_bw()
```



Questions 6:

- How many tests did you perform?

There are 32 tests

- What are you correcting for with the fdr? Why is this important for eQTL analysis?

The FDR corrected the p-value of each test. This is important since in th real eQTL analysis we might do lots of testings (i.e. every SNPs are tested to every genes) and if the p-value is not corrected, we can get many false positive.

Part 2

Task 1

```
# Expression data for chromosome 20
expr_ceu_20 <- read.table("expr_ceu_chr20.tab", header=TRUE, sep="\t") |>
  as_tibble()

# Gene positions for genes on chromosome 20
pos_expr_ceu_20 <- read.table("expr_chr20.pos", header=TRUE, sep="\t") |>
  as_tibble()

# Genotype data for chromosome 20
geno_ceu_20 <- read.table("geno_ceu_chr20_strict.tab", header=TRUE, sep="\t") |>
  as_tibble()

# Position of genotype data for chromosome 20
pos_geno_ceu_20 <- read.table("geno_ceu_chr20_strict.pos", header=TRUE, sep="\t") |>
  as_tibble()

# Genotype data for chromosome 22
geno_ceu_22 <- read.table("geno_ceu_chr22_strict.tab", header=TRUE, sep="\t") |>
  as_tibble()

# Position of genotype data for chromosome 22
pos_geno_ceu_22 <- read.table("geno_ceu_chr22_strict.pos", header=TRUE, sep="\t") |>
  as_tibble()
```

- How many samples are included in this dataset?

```
head(expr_ceu_20)
```

```
## # A tibble: 6 x 92
##   id     NA06984 NA06985 NA06986 NA06989 NA06994 NA07037 NA07048 NA07051 NA07056
##   <chr>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 ENSG0~   0.157 4.10e-2   0.362 6.00e-2 2.40e-1   0.461 1.65e-1 2.01e-1   0.367
## 2 ENSG0~   5.47  3.82e+0   6.12  5.66e+0 4.50e+0   4.90  8.18e+0 8.90e+0   5.12
## 3 ENSG0~ 149.    1.18e+2 136.    1.35e+2 1.24e+2 126.    1.18e+2 1.30e+2 128.
## 4 ENSG0~   0.181 1.31e-1   0.504 2.43e-2 5.76e-2   0.773 4.49e-2 7.91e-4   0.230
## 5 ENSG0~   8.10  5.80e+0   7.47  6.34e+0 6.71e+0   7.11  7.34e+0 5.30e+0   6.59
## 6 ENSG0~  13.0   1.23e+1  12.0   1.33e+1 9.49e+0  11.9   1.00e+1 1.19e+1  14.5
## # i 82 more variables: NA07346 <dbl>, NA07347 <dbl>, NA07357 <dbl>,
## #   NA10847 <dbl>, NA10851 <dbl>, NA11829 <dbl>, NA11830 <dbl>, NA11831 <dbl>,
## #   NA11832 <dbl>, NA11840 <dbl>, NA11843 <dbl>, NA11881 <dbl>, NA11892 <dbl>,
## #   NA11893 <dbl>, NA11894 <dbl>, NA11918 <dbl>, NA11920 <dbl>, NA11930 <dbl>,
```

```
## #   NA11931 <dbl>, NA11992 <dbl>, NA11993 <dbl>, NA11994 <dbl>, NA11995 <dbl>,
## #   NA12004 <dbl>, NA12005 <dbl>, NA12006 <dbl>, NA12043 <dbl>, NA12044 <dbl>,
## #   NA12045 <dbl>, NA12058 <dbl>, NA12144 <dbl>, NA12154 <dbl>, ...
```

```
head(geno_ceu_20)
```

```
## # A tibble: 6 x 92
##   id     NA06984 NA06985 NA06986 NA06989 NA06994 NA07037 NA07048 NA07051 NA07056
##   <chr>    <int>   <int>   <int>   <int>   <int>   <int>   <int>   <int>   <int>
## 1 snp_2~       1       2       1       1       2       1       1       2       2
## 2 snp_2~       0       0       1       1       0       1       0       0       1
## 3 snp_2~       0       1       1       0       1       1       0       0       0
## 4 snp_2~       0       0       0       0       0       0       2       1       0
## 5 snp_2~       0       0       2       1       0       0       1       0       1
## 6 snp_2~       0       1       0       0       0       0       0       1       1
## # i 82 more variables: NA07346 <int>, NA07347 <int>, NA07357 <int>,
## #   NA10847 <int>, NA10851 <int>, NA11829 <int>, NA11830 <int>, NA11831 <int>,
## #   NA11832 <int>, NA11840 <int>, NA11843 <int>, NA11881 <int>, NA11892 <int>,
## #   NA11893 <int>, NA11894 <int>, NA11918 <int>, NA11920 <int>, NA11930 <int>,
## #   NA11931 <int>, NA11992 <int>, NA11993 <int>, NA11994 <int>, NA11995 <int>,
## #   NA12004 <int>, NA12005 <int>, NA12006 <int>, NA12043 <int>, NA12044 <int>,
## #   NA12045 <int>, NA12058 <int>, NA12144 <int>, NA12154 <int>, ...
```

```
head(geno_ceu_22)
```

```
## # A tibble: 6 x 92
##   id     NA06984 NA06985 NA06986 NA06989 NA06994 NA07037 NA07048 NA07051 NA07056
##   <chr>    <int>   <int>   <int>   <int>   <int>   <int>   <int>   <int>   <int>
## 1 snp_2~       0       1       0       0       0       0       0       1       1
## 2 snp_2~       2       2       1       1       2       2       1       1       0
## 3 snp_2~       0       2       0       1       1       1       1       1       2
## 4 snp_2~       0       0       1       1       1       1       1       0       0
## 5 snp_2~       1       2       1       2       2       2       1       2       2
## 6 snp_2~       1       2       2       2       1       2       2       2       2
## # i 82 more variables: NA07346 <int>, NA07347 <int>, NA07357 <int>,
## #   NA10847 <int>, NA10851 <int>, NA11829 <int>, NA11830 <int>, NA11831 <int>,
## #   NA11832 <int>, NA11840 <int>, NA11843 <int>, NA11881 <int>, NA11892 <int>,
## #   NA11893 <int>, NA11894 <int>, NA11918 <int>, NA11920 <int>, NA11930 <int>,
## #   NA11931 <int>, NA11992 <int>, NA11993 <int>, NA11994 <int>, NA11995 <int>,
## #   NA12004 <int>, NA12005 <int>, NA12006 <int>, NA12043 <int>, NA12044 <int>,
## #   NA12045 <int>, NA12058 <int>, NA12144 <int>, NA12154 <int>, ...
```

```
dim(expr_ceu_20)
```

```
## [1] 561  92
```

```
dim(geno_ceu_20)
```

```
## [1] 30000    92
```

```
dim(geno_ceu_22)
```

```
## [1] 1001   92
```

```
# Check if the individual on different chromosomes data are the same
sum(names(expr_ceu_20 |> select(-c(id))) != names(geno_ceu_20 |> select(-c(id))))
```

```
## [1] 0
```

```r
sum(names(expr_ceu_20 |> select(-c(id))) != names(geno_ceu_22 |> select(-c(id))))
```

```
## [1] 0
```

```r
sum(names(geno_ceu_20 |> select(-c(id))) != names(geno_ceu_22 |> select(-c(id))))
```

```
## [1] 0
```

From the data shown above, we can conclude that each of the data contains samples from 91 individuals.

- How many variants are present on chromosome 20?

Based on the length of the geno_ceu_20, there are 3000 variants within the chromosome 20

- How many homozyguous and heterozygous genotypes are observed for the first individual in the dataset (NA06984)?

To check the number of homozygous and heterozygous genotypes, we check the chromosome 20 and 22 for NA06984

```r
# Check on chromosome 20
chr_20_sum_geno <- table(geno_ceu_20$NA06984)
# Check on chromosome 22
chr_22_sum_geno <- table(geno_ceu_22$NA06984)

tot_hom_geno <- chr_20_sum_geno[1] + chr_20_sum_geno[3] +
  chr_22_sum_geno[1] + chr_22_sum_geno[3]
tot_het_geno <- chr_20_sum_geno[2] + chr_22_sum_geno[2]

cat("Total umber of homozygous genotypes: ", tot_hom_geno,
    "\nTotal number of heterozygous genotypes: ", tot_het_geno )
```

```
## Total umber of homozygous genotypes:  17631
## Total number of heterozygous genotypes:  13370
```

- How many genes are included?

```r
cat("Total number of gene included (expression on chr 20): ",
    length(pos_expr_ceu_20$geneid))
```

```
## Total number of gene included (expression on chr 20):  561
```

- What gene shows the highest mean expression?

```r
expr_ceu_20 |>
  rowwise(id) |>
  summarise(mean_expr = mean(c_across(where(is.numeric))), .groups = "drop") |>
  arrange(desc(mean_expr)) |>
  head()
```

```
## # A tibble: 6 x 2
##   id                 mean_expr
##   <chr>                  <dbl>
## 1 ENSG00000227063.4      3932.
## 2 ENSG00000185834.9      1026.
## 3 ENSG00000124243.11      664.
## 4 ENSG00000235508.2       642.
## 5 ENSG00000214535.3       601.
## 6 ENSG00000236992.1       573.
```

From the data shown above, the "ENSG00000227063.4" gene has the highest mean gene expression

Task 2 - cis-eQTL

```
#Matrix eQTL
library(MatrixEQTL)
# Genotype file names
SNP_file_name = "geno_ceu_chr20_strict.tab" ; #Genotype file path
snps_location_file_name = "geno_ceu_chr20_strict.pos" ; #snp position file path

# Gene expression file names
expression_file_name = "expr_ceu_chr20.tab" ;#Expression file path
gene_location_file_name = "expr_chr20.pos" ;#gene position file path

# Only associations significant at this level will be saved
pvOutputThreshold_cis = 1; #p.value threshold for cis eqtls
pvOutputThreshold_tra = 0; #p.value threshold for trans eqtls


#Covariates file names
covariates_file_name = character();# Set to character() for no covariates

# Distance for local gene-SNP pairs
cisDist = 1e6; #Define cis distance


## Load genotype data
snps = SlicedData$new();
snps$fileDelimiter = "\t"; # the TAB character
snps$fileOmitCharacters = "NA"; # denote missing values;
snps$fileSkipRows = 1; # one row of column labels
snps$fileSkipColumns = 1; # one column of row labels
snps$fileSliceSize = 20000; # read file in slices of 2,000 rows
snps$LoadFile(SNP_file_name);
```

```
## Rows read: 20,000
```

```
## Rows read: 30000 done.
```

```
## Load gene expression data
gene = SlicedData$new();
gene$fileDelimiter = "\t"; # the TAB character
gene$fileOmitCharacters = "NA"; # denote missing values
gene$fileSkipRows = 1;
gene$fileSkipColumns = 1;
gene$fileSliceSize = 20000;
gene$LoadFile(expression_file_name);
```

```
## Rows read: 561 done.
```

```
#Load position files
snpspos = read.table(snps_location_file_name, header = TRUE, stringsAsFactors = FALSE);
genepos = read.table(gene_location_file_name, header = TRUE, stringsAsFactors = FALSE);

## Run the analysis
me = Matrix_eQTL_main(
  snps = snps,
  gene = gene,
```

```
  output_file_name=NULL,
  pvOutputThreshold = pvOutputThreshold_tra,
  useModel = modelLINEAR,
  errorCovariance =numeric(),
  verbose = FALSE,
  output_file_name.cis = NULL, #Do not write out cis results
  pvOutputThreshold.cis = pvOutputThreshold_cis,
  snpspos = snpspos,
  genepos = genepos,
  cisDist = cisDist,
  min.pv.by.genesnp = FALSE,
  noFDRsaveMemory = FALSE,
  pvalue.hist = FALSE)
```

```
## 561 of 561 genes matched
```

```
## 30000 of 30000 SNPs matched
```

```
## 50.00% done, 331,024 cis-eQTLs
```

```
## 100.00% done, 527,117 cis-eQTLs
```

```
cis_eqtls = me$cis$eqtls[,-c(5)]
cis_eqtls["beta_se"] = cis_eqtls["beta"]/cis_eqtls["statistic"]
rm(me)
```

- How many tests were conducted?

```
dim(cis_eqtls)
```

```
## [1] 527117      6
```

There are 527,117 tests conducted.

- Using a bonferroni correction ($\alpha = 0.05$), how many genes are significant?

```
cis_eqtls["adj_pvalue"] <- p.adjust(cis_eqtls$pvalue, method = "bonferroni")

# Filtering cis_eqlts with significant value
signf_cis_eqtls <- cis_eqtls[cis_eqtls["adj_pvalue"] < 0.05, ]
length(unique(signf_cis_eqtls$gene))
```

```
## [1] 6
```

From the test, There are only 6 genes that are significant

- Report the gene-snp pair show the lowest pvalue? What is the effect size of this snp-gene pair?

```
cis_eqtls |>
  arrange(pvalue) |>
  head(1)
```

```
##              snps              gene statistic       pvalue      beta   beta_se
## 1 snp_20_37055875 ENSG00000196756.5 -9.420136 5.066679e-15 -8.146604 0.8648075
##     adj_pvalue
## 1 2.670733e-09
```

From the result above, ENSG00000196756.5-snp_20_37055875 is the gene-snp pair with the lowest pvalue and its estimated effect size is ~-8.147.

- What is the biotype of this gene?

This gene has biotype of lncRNA

Task 3 - trans-eQTL

```
#Matrix eQTL
# Genotype file names
SNP_file_name = "geno_ceu_chr22_strict.tab" ; #Genotype file path
snps_location_file_name = "geno_ceu_chr22_strict.pos" ; #snp position file path


# Gene expression file names
expression_file_name = "expr_ceu_chr20.tab" ;#Expression file path
gene_location_file_name = "expr_chr20.pos" ;#gene position file path

# Only associations significant at this level will be saved
pvOutputThreshold_cis = 0; #p.value threshold for cis eqtls
pvOutputThreshold_tra = 1; #p.value threshold for trans eqtls

#Covariates file names
covariates_file_name = character();# Set to character() for no covariates

# Distance for local gene-SNP pairs
cisDist = 1e6; #Define cis distance

## Load genotype data
snps = SlicedData$new();
snps$fileDelimiter = "\t"; # the TAB character
snps$fileOmitCharacters = "NA"; # denote missing values;
snps$fileSkipRows = 1; # one row of column labels
snps$fileSkipColumns = 1; # one column of row labels
snps$fileSliceSize = 20000; # read file in slices of 2,000 rows
snps$LoadFile(SNP_file_name);
```

```
## Rows read: 1001 done.
```

```
## Load gene expression data
gene = SlicedData$new();
gene$fileDelimiter = "\t"; # the TAB character
gene$fileOmitCharacters = "NA"; # denote missing values;
gene$fileSkipRows = 1;
gene$fileSkipColumns = 1;
gene$fileSliceSize = 20000;
gene$LoadFile(expression_file_name);
```

```
## Rows read: 561 done.
```

```
snpspos = read.table(snps_location_file_name, header = TRUE, stringsAsFactors = FALSE);
genepos = read.table(gene_location_file_name, header = TRUE, stringsAsFactors = FALSE);

## Run the analysis
me = Matrix_eQTL_main(
  snps = snps,
  gene = gene,
  output_file_name=NULL,
  pvOutputThreshold = pvOutputThreshold_tra,
```

```
  useModel = modelLINEAR,
  errorCovariance =numeric(),
  verbose = FALSE,
  output_file_name.cis = NULL, #Do not write out cis results
  pvOutputThreshold.cis = pvOutputThreshold_cis,
  snpspos = snpspos,
  genepos = genepos,
  cisDist = cisDist,
  min.pv.by.genesnp = FALSE,
  noFDRsaveMemory = FALSE,
  pvalue.hist = FALSE)
```

```
## 100.00% done, 561,561 eQTLs
```

```
trans_eqtls = me$all$eqtls[,-c(5)]
trans_eqtls["beta_se"] = trans_eqtls["beta"]/trans_eqtls["statistic"]
rm(me)
```

```
dim(trans_eqtls)
```

```
## [1] 561561      6
```

- How many tests were conducted?

From the size of trans_eqtls data, there are 561,561 tests conducted

- Using a bonferroni correction ($\alpha = 0.05$), how many genes are significant?

```
trans_eqtls["adj_pvalue"] <- p.adjust(trans_eqtls$pvalue, method = "bonferroni")

# Filtering trans_eqlts with significant value
signf_trans_eqtls <- trans_eqtls[trans_eqtls["adj_pvalue"] < 0.05, ]
length(unique(signf_trans_eqtls$gene))
```

```
## [1] 0
```

From the result above, with the bonferroni correction, there is not any significant genes.

Task 4 - QQ-plot In this section we will explore QQ-plots (Quantile-Quantile plots) for both the cis-eQTLS and the trans-eQTLS.

- Briefly explain what a QQ-plot can be used for (2-3 sentences)

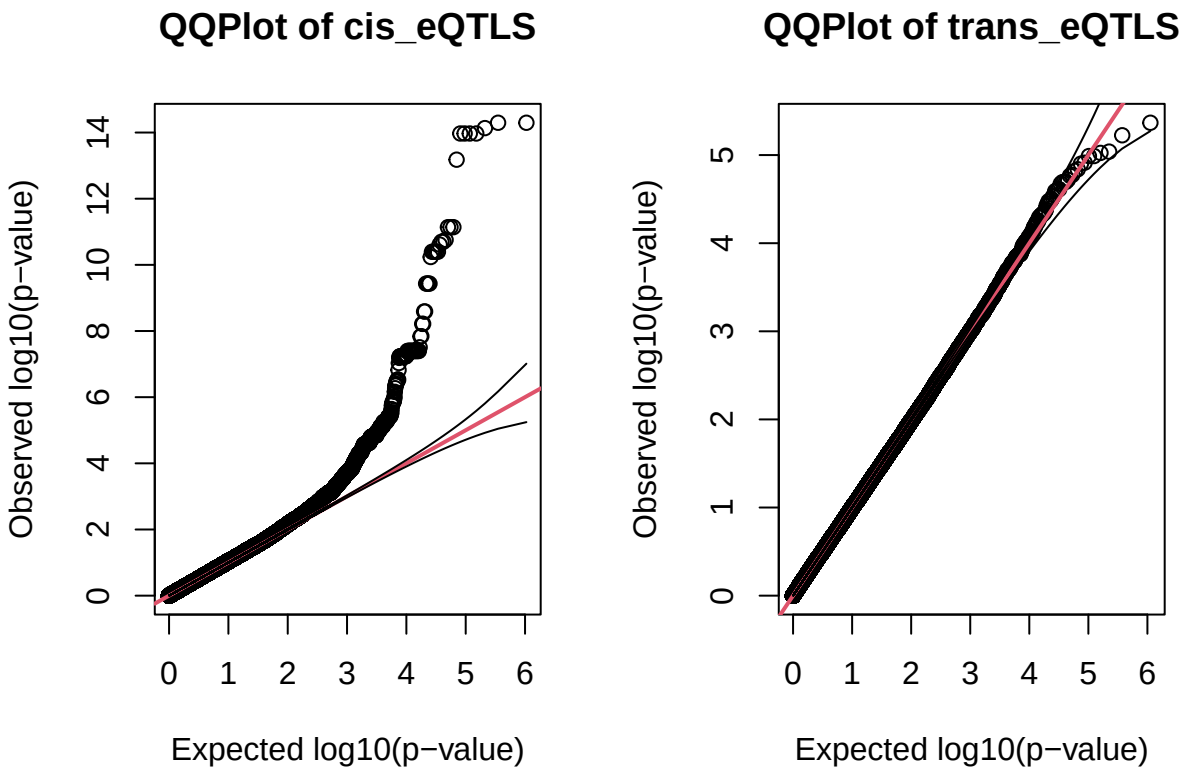
QQplot is a plot that can be used to compare two distributions with plotting their quantiles. If the two distributions are the same, the points would lie on the linear line.

```
qqp<-function(x, title, maxLogP=30,...){
  x<-x[!is.na(x)]
  if(!missing(maxLogP)){
    x[x<10^(-maxLogP)]<-10^(-maxLogP)
  }
  N<-length(x)
  chi1<-qchisq(1-x,1)
  x<-sort(x)
  e<- -log((1:N-0.5)/N,10)
  plot(e,-log(x,10),main=title,ylab="Observed log10(p-value)",xlab="Expected log10(p-value)",...)
  abline(0,1,col=2,lwd=2)
  c95<-qbeta(0.95,1:N,N-(1:N)+1)
  c05<-qbeta(0.05,1:N,N-(1:N)+1)
```

```
  lines(e,-log(c95,10))
  lines(e,-log(c05,10))
}
```

- Compute the QQ-plot for both the cis and trans eQTL separately

```
par(mfrow=c(1,2))
qqp(cis_eqtls$pvalue, "QQPlot of cis_eQTLS")
qqp(trans_eqtls$pvalue, "QQPlot of trans_eQTLS")
```



- What is the main difference between these two QQ-plots and what drives this difference? From the figure above, we can see that the cis-eQTL tends to deviate from the null distribution (p-value uniformly distributed [0,1]) shown by the red line. The deviation of cis eQTL QQplot makes sense since there should be local regulation near the genes (cis-SNPs). However, the trans-eQTL QQ plot seems to be expected as the null distribution. This could be due to to the given genes are not associated with the given distal regulators (trans-SNPs).

Task 5 - PVE In the last exercise, we will calculate how much of the variance in gene expression can be explained by a SNP. This is called proportion of variance explained (PVE).

- Calculate the PVE for all cis SNP-gene pairs and make a histogram of them

```
# Use matrixStats for faster calculation
library(matrixStats)
```

```
##
## Attaching package: 'matrixStats'
```

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

```r
# Data Table for faster join
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

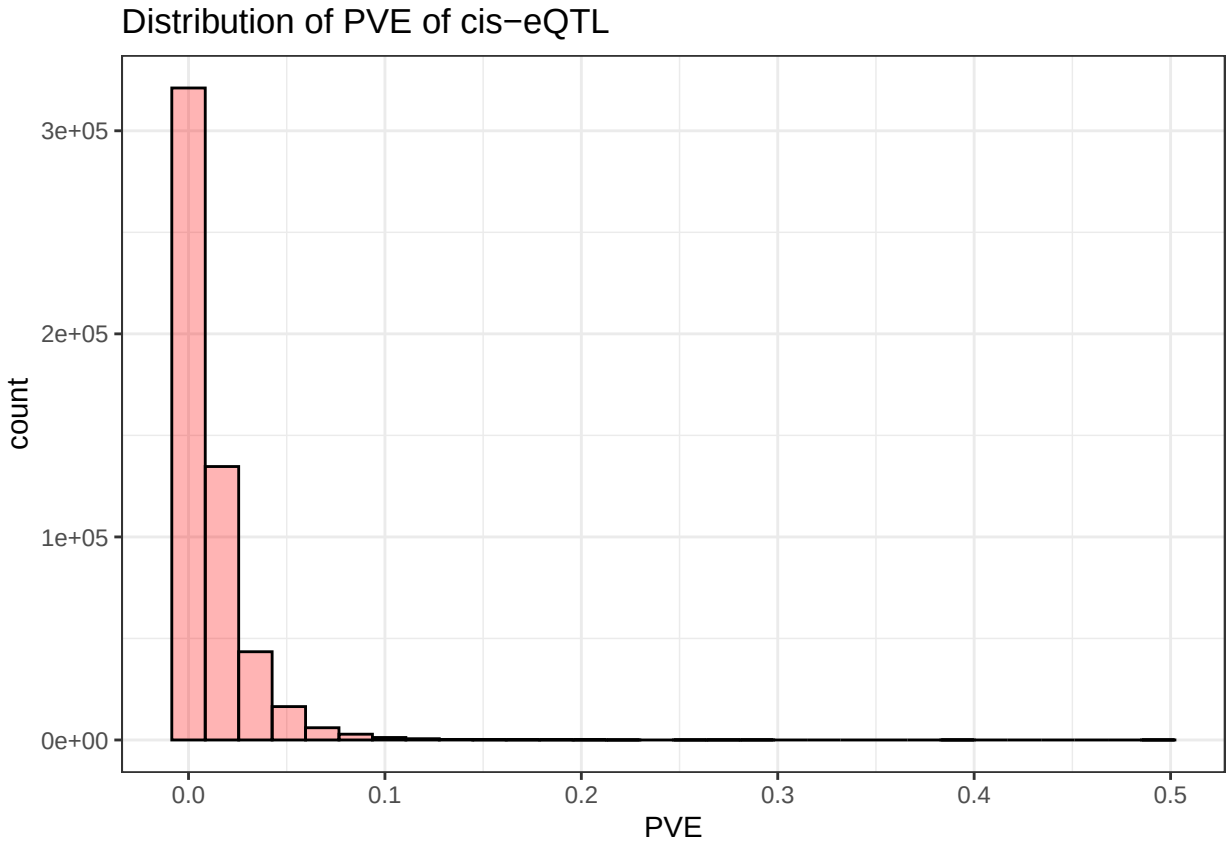
```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

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

```r
geno_ceu_20_mat <- as.matrix(geno_ceu_20 |> select(-c(id)))
geno_ceu_20_MAF <- data.table(snps = geno_ceu_20$id,
                          MAF = matrixStats::rowSums2(geno_ceu_20_mat)/dim(geno_ceu_20_mat)[2]/2)
setindex(geno_ceu_20_MAF, snps)

cis_eqtls <- as.data.table(cis_eqtls)
setindex(cis_eqtls, snps)

cis_eqtls <- cis_eqtls[geno_ceu_20_MAF, on= .(snps), nomatch = NULL ]

N <- dim(geno_ceu_20_mat)[2]
cis_eqtls[, PVE := (2*(beta^2)*MAF*(1-MAF))/( 2*(beta^2)*MAF*(1-MAF) +  (beta_se^2)*2*N*MAF*(1-MAF)), n
```

```r
ggplot(data = cis_eqtls, mapping = aes(x = PVE)) +
  geom_histogram(colour="black", fill="red", alpha = 0.3) +
  labs(title = "Distribution of PVE of cis-eQTL") +
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



- Report the mean PVE across all snp-gene pairs.

```
cat("Mean PVE across all snp-gene pairs: ", mean(cis_eqtls$PVE))
```

```
## Mean PVE across all snp-gene pairs:  0.01155724
```

- Report the snp-gene pair and PVE that explains the largest amount of the variance

```
cis_eqtls |>
  arrange(desc(PVE)) |>
  head(5) |>
  select(-c(statistic))
```

```
##                 snps             gene       pvalue      beta   beta_se
## 1:   snp_20_37055875 ENSG00000196756.5 5.066679e-15 -8.146604 0.8648075
## 2: snp_20_37055875.1 ENSG00000196756.5 5.066679e-15 -8.146604 0.8648075
## 3:   snp_20_37033582 ENSG00000196756.5 7.347426e-15 -7.359478 0.7877680
## 4:   snp_20_37025918 ENSG00000196756.5 1.070654e-14 -7.954230 0.8586880
## 5:   snp_20_37026379 ENSG00000196756.5 1.070654e-14 -7.954230 0.8586880
##      adj_pvalue       MAF       PVE
## 1: 2.670733e-09 0.8406593 0.4937102
## 2: 2.670733e-09 0.8406593 0.4937102
## 3: 3.872953e-09 0.8296703 0.4895570
## 4: 5.643600e-09 0.8296703 0.4853161
## 5: 5.643600e-09 0.8296703 0.4853161
```

From the above data, we can see that snp-gene pair with the highest PVE is snp_20_37055875-ENSG00000196756.5