eQTL practicals TrainMalta

These are the practials used for the eQTL introductory course in the TrainMalta course. The practicals are based on ideas and data in http://jknightlab.github.io/eqtl-intro/exercises/exercises_and_solutions.html and on genotyping data from HapMap 2.

Preparation and basic stats

We will start by loading the expression and the genotype data into R, you can do that by:

```
options(stringsAsFactors=FALSE)
setwd("/data/day5/eqtl_intro")
expr = read.table("simulated/sim_expression1.tab", sep="\t", header=TRUE, row.names = 1)
gt = read.table("simulated/sim_genotypes.tab", sep="\t", header=TRUE, row.names = 1)
```

Basic stats

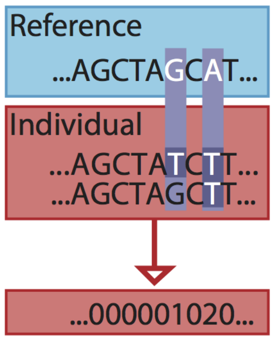
Task

To get a basic idea of how many samples, genes and which genotyped positions are available in the dataset, print out the dimensions of the loaded datasets. For the first 10 genes print out the mean expression levels + standard deviation. Do the same for the first 10 SNPs in the table.

MAF

Minor allele frequency (MAF) is a measure of the presence an allele in a population. Every individual person has about 4.1-5 million bases in which he differs from the reference genome http://www.nature.com/nature/journal/v526/n7571/full/nature15393.html#a-typical-genome. Some of those variants are common in a certain population, others not. In order to measure the rareness of a specific variant (allele) MAF can be calculated. In case of single nucleotide polymorphisms there can be up to four different alleles in one position (A, C, G, T). Those alleles can be homozygous or heterozygous, when the maternal allele was different from the paternal allele at that position.

SNPs in a population are always defined by the genomic position and by two alleles: The allele defined in the reference genome and one allele present in some individuals, but different from the reference sequence. An example:



Spotting the differences:

This individual has two SNPs, he is heterozygous in one of them and homozygous in the other. In eQTL analyses a SNP is always defined as a single allele being different from the reference. If in a population there are multiple different alleles for one position, they would be treated as independent entities of SNPs.

Now we know that in eQTL analyses SNPs can only have two alleles: The reference and the alternative. Calculating MAF is essentially counting the presence of the alleles in a population and representing it as a percentage. Each individual can have 0, 1 or 2 times the alternative allele.

Task

Try to find out what the values of the genotype matrix mean.

Answer

The values are allele counts of a SNP in each individual.

Task

Calculate the MAF of the SNPs among all inidividuals.

Answer

```r
apply(gt,2,mean)/2
```

```
##       snp_1       snp_2       snp_3       snp_4       snp_5       snp_6
## 0.006666667 0.030000000 0.020000000 0.065000000 0.046666667 0.126666667
##       snp_7       snp_8       snp_9      snp_10
## 0.171666667 0.298333333 0.390000000 0.511666667
```

The term MAF implies that the allele for which we return the measure has to be the minor (= less common) allele. This means that the MAF is smaller than 0.5 by definition.

Task

Calculate the MAF for all SNPs among all individuals and correct the returned values so that the value is always given in respect to the minor allele. Then plot a histogram of the MAFs of all SNPs

Answer

```r
maf <- colMeans(gt)/2
maf <- pmin(maf, 1-maf)
maf
```

```
##       snp_1       snp_2       snp_3       snp_4       snp_5       snp_6
## 0.006666667 0.030000000 0.020000000 0.065000000 0.046666667 0.126666667
##       snp_7       snp_8       snp_9      snp_10
## 0.171666667 0.298333333 0.390000000 0.488333333
```

Filtering SNPs by MAF

In an eQTL study often a minimum MAF is required. Since MAF essentially reflects how often an allele has been observed in a population, it also defines how often the gene expression levels have been observed for heterozygous and homozygous alleles.

Task

Calculate the number of heterozygous and homozygous observations expected for SNPs with a MAF of 1%, 5% and 10% in a sample of 300 individuals given Hardy-Weinberg equilibrium. What are useful MAF thresholds for SNPs to include in an eQTL analysis?

Answer

```
p = c(0.01, 0.05, 0.1)
q = 1-p
# Calulate frequency of minor allele being present in homozygous and heterozygous state
f_hom = p^2
f_het = 2*p*q
# Expected number of observations in a sample size of 300
sample_size = 300
round(f_hom * sample_size)
```

```
## [1] 0 1 3
```

```
round(f_het * sample_size)
```

```
## [1]  6 28 54
```

Gene expression profiling

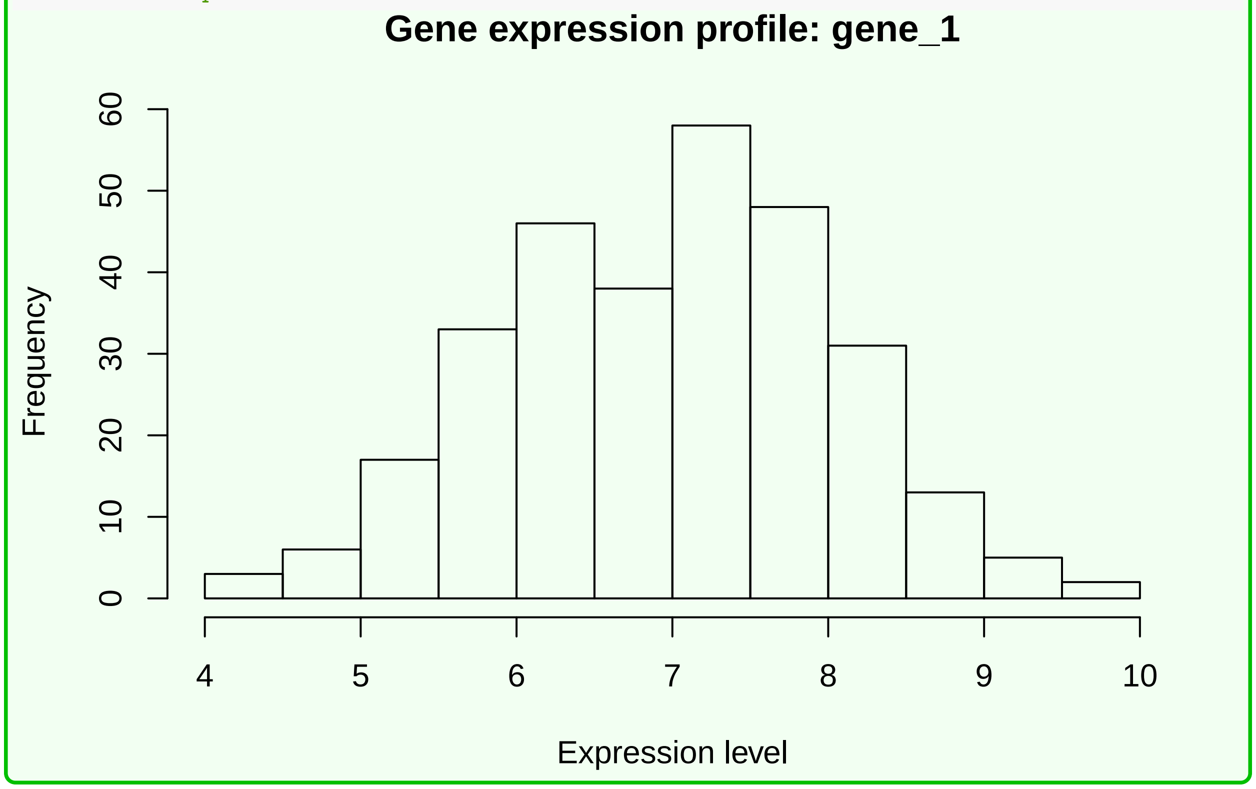
Now that we have an idea of what is stored in the genotype matrix we take a look at the expression data. Important for eQTL analyses is that the gene expression has to be normally distributed among samples, therefore RNA-seq data has to be transformed by, for example quantile normalisation.

Task

Plot the distribution of gene expression levels across samples for gene "gene_1".

Answer

```
gname = "gene_1"
hist(expr[,gname], main=paste("Gene expression profile:",gname),
     xlab="Expression level")
```



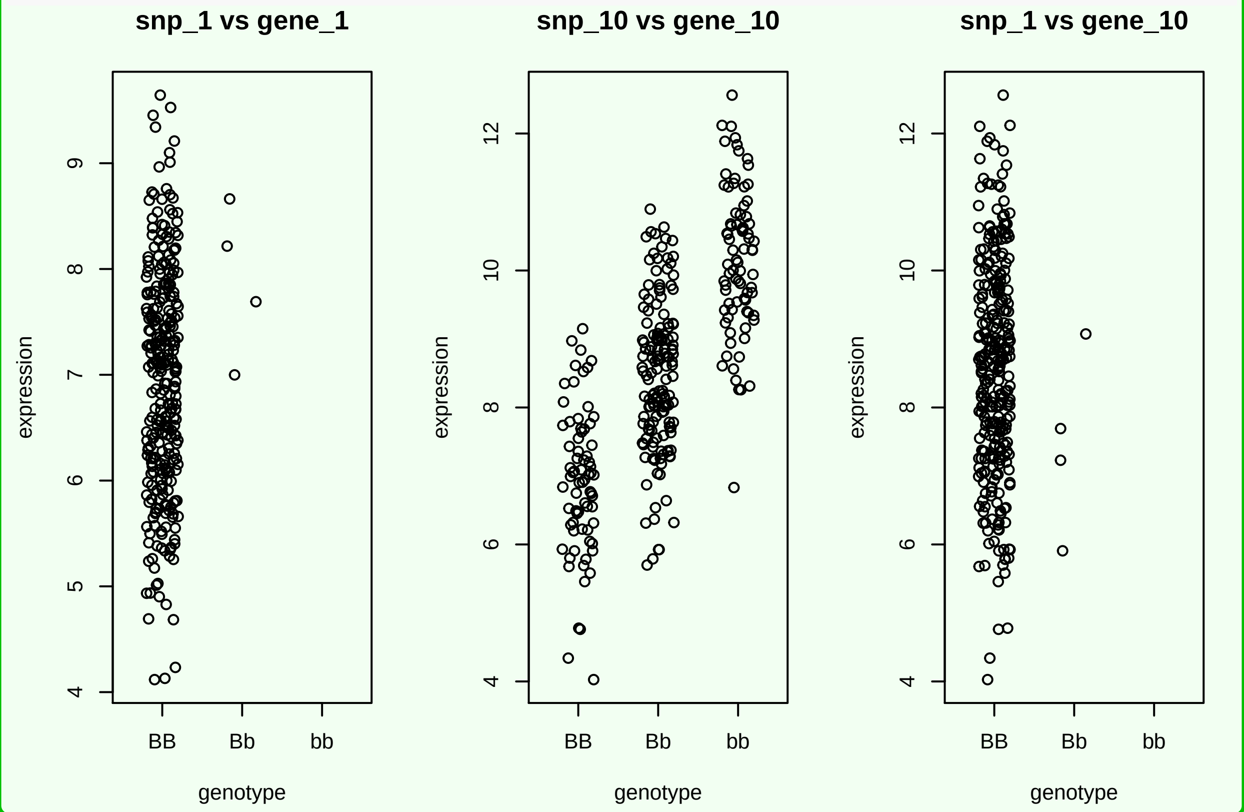
Task

Now plot the expression levels of gene "gene_1" against "snp_1" and "gene_10" against "snp_10" depending on the genotypes of the samples by using dot plots. Consider adding a bit of random noise to the genotype data to make it look nicer.

Answer

```
snps = c("snp_1", "snp_10", "snp_1")
genes = c("gene_1", "gene_10", "gene_10")


par(mfrow=c(1,length(snps)))
for (index in seq(length(snps))){
  genotype = gt[,snps[index]]
  expression = expr[,genes[index]]
  plot(genotype + runif(length(genotype), min=-0.2, max=0.2), expression,
       main=paste(snps[index], "vs", genes[index]), xlim= c(-0.5,2.5),
       xlab = "genotype", xaxt="n")
  axis(1, at=c(0,1,2), labels = c("BB", "Bb", "bb"))
}
```



Understanding the basics

This chapter should explain the basic ideas behind eQTL analyses. What we are doing here is not what one would do to run an eQTL analysis, but it explains how eQTL calling works in general.

Linear regression of genotype on phenotype

The most common way of estimating the effect of a SNP on gene expression is by performing a linear regression of sample genotypes on sample gene expression levels. The p-value indicates the significance of the genetic component in the model. Let's try that for gene 10 with snp 1 and snp 10.

```r
lm_1_10 = lm(expr[,"gene_10"] ~ gt[,"snp_1"])
summary(lm_1_10)
```

```
##
## Call:
## lm(formula = expr[, "gene_10"] ~ gt[, "snp_1"])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5262 -1.1412  0.0577  1.1552  4.0071
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.55324    0.09138  93.600   <2e-16 ***
## gt[, "snp_1"] -1.07768    0.79138  -1.362    0.174
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.572 on 298 degrees of freedom
## Multiple R-squared:  0.006185,   Adjusted R-squared:  0.00285
## F-statistic: 1.854 on 1 and 298 DF,  p-value: 0.1743
```

```r
lm_10_10 = lm(expr[,"gene_10"] ~ gt[,"snp_10"])
summary(lm_10_10)
```

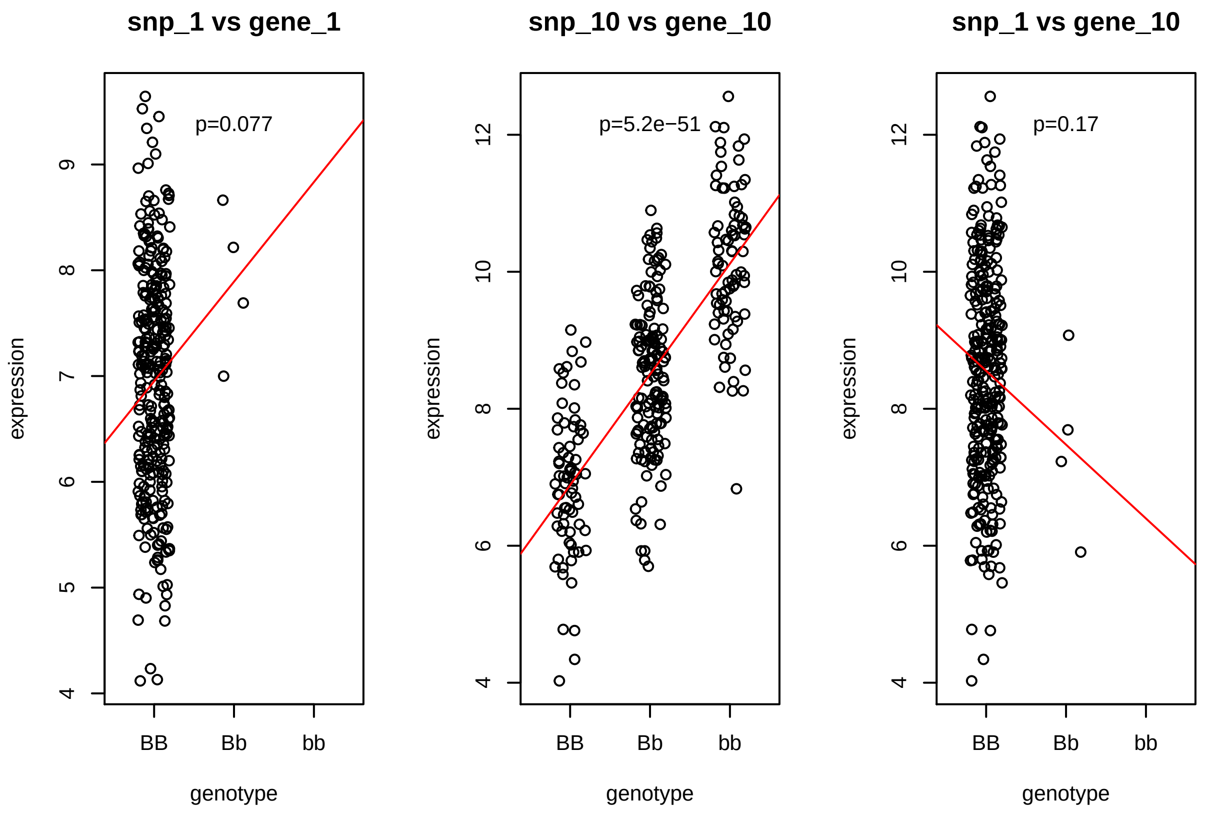
```
##
## Call:
## lm(formula = expr[, "gene_10"] ~ gt[, "snp_10"])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2863 -0.7161  0.0474  0.6658  2.4429
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     6.88488    0.10941   62.93   <2e-16 ***
## gt[, "snp_10"]  1.61627    0.08788   18.39   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.079 on 298 degrees of freedom
## Multiple R-squared:  0.5317, Adjusted R-squared:  0.5301
## F-statistic: 338.3 on 1 and 298 DF,  p-value: < 2.2e-16
```

This is the standard summary output from R for linear regressions. Since we are interested in eQTLs our main interest lies in the second line of "Coefficients". What is stated as "Estimate" is the slope of the linear regression, which in eQTL terms is called "effect size" or "beta". In eQTL studies one normally compares thousands of genes for which each hundreds to thousands of SNPs have been tested. The common way to identify eQTLs is by their p-value. The p-value given here (Pr(>|t|)) will later be referred to as raw p-value. It can be calculated in many different ways, here it is based on the t-value which is derived from the estimation of the coefficient and its standard error. For a nice explanation of the summary(lm) output see: http://stats.stackexchange.com/questions/5135/interpretation-of-rs-lm-output

Now let's display our results using the code from before:

```r
snps = c("snp_1", "snp_10", "snp_1")
genes = c("gene_1", "gene_10", "gene_10")


par(mfrow=c(1,length(snps)))
for (index in seq(length(snps))){
  genotype = gt[,snps[index]]
  expression = expr[,genes[index]]
  lm_result = lm(expression ~ genotype)
  plot(genotype + runif(length(genotype), min=-0.2, max=0.2),
       expression, main=paste(snps[index], "vs", genes[index]),
       xlim= c(-0.5,2.5), xlab = "genotype", xaxt="n")
  abline(lm_result, col="red")
  axis(1, at=c(0,1,2), labels = c("BB", "Bb", "bb"))
  # Add p-values as text
  y_range = range(expression)
  text(x=1, y=y_range[1] + 0.95*diff(y_range), paste0("p=",
        format(summary(lm_result)$coefficients[2,4],
        scientific=TRUE, digits=2)))
}
```



A nicer way to plot it

In ggplot2 these kinds of plots can be produced very nicely by doing the following:

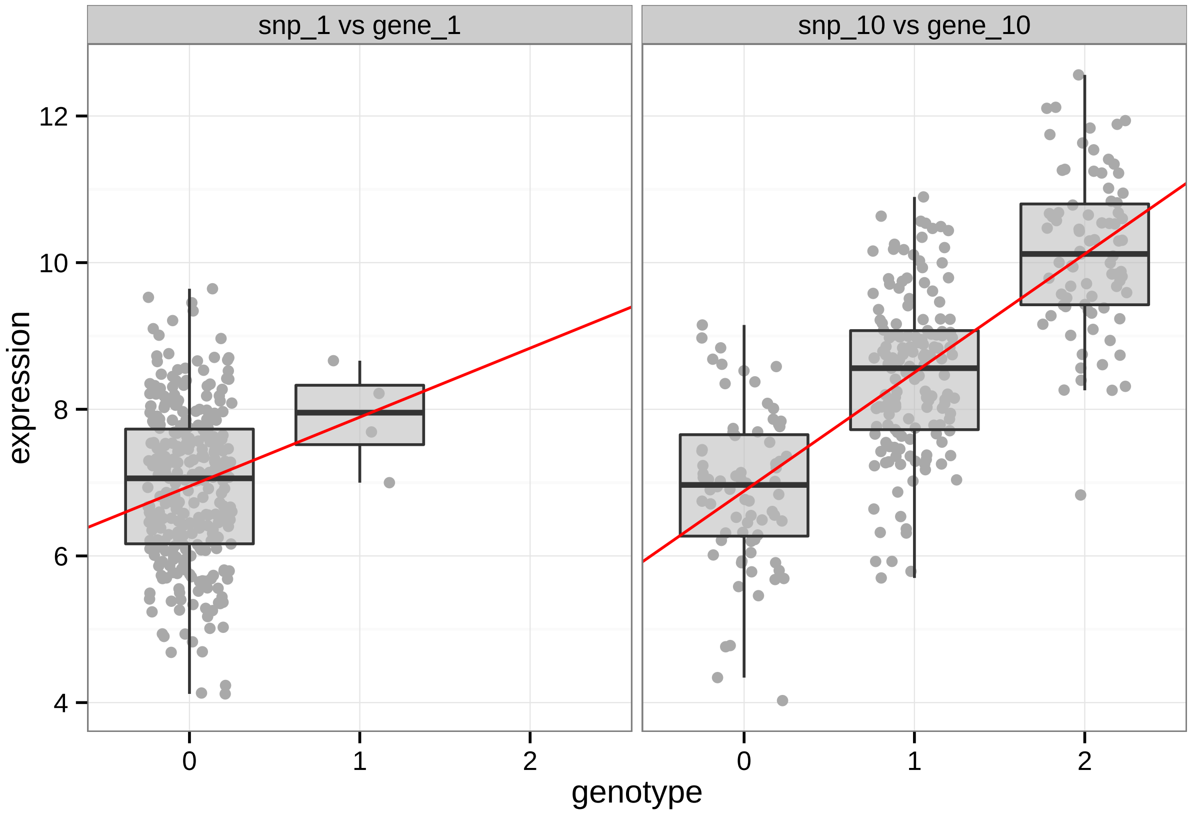
```r
require(tidyr)
```

```
## Loading required package: tidyr
## Warning: package 'tidyr' was built under R version 3.1.3
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.1.3
```

```
genoLong = tidyr::gather(gt, snp, genotype, snp_1, snp_10)
exprLong = tidyr::gather(expr, gene, expression, gene_1, gene_10)
dataLong = cbind(genoLong[,c("snp", "genotype")], exprLong[,c("gene", "expression")])
dataLong$comparison = paste(dataLong$snp, "vs", dataLong$gene)
dataLong$genotype = factor(dataLong$genotype)
p = ggplot(dataLong, aes(genotype, expression)) +
        geom_jitter(colour="darkgrey", position=position_jitter(width=0.25)) +
        geom_boxplot(outlier.size=0, alpha=0.6, fill="grey") +
        facet_wrap(~comparison) + theme_bw()
plot_lm_results = c()
for (i in c(1,10)){
  lm_out = lm(expr[,i] ~ gt[,i])
  new_data = data.frame("intercept"=lm_out$coefficients[1] - lm_out$coefficients[2],
                        "slope"=lm_out$coefficients[2], "comparison" =
                         paste(colnames(gt)[i],"vs", colnames(expr)[i]))
  plot_lm_results = rbind(plot_lm_results, new_data)
}
p = p+ geom_abline(data=plot_lm_results,aes(slope = slope, intercept = intercept),
                   col="red") + facet_grid(~comparison)
p
```



Covariates

Many different factors can affect gene expression, such as age, sex, smoking habits, genetic mutations and environmental factors, such as nutrition, etc. The more factors can be described in the model, the more accurate it will be and the higher are chances to find more subtle genetic effects.

Covariates therefore are features of samples which may describe effects on gene expression. In technical terms one covariate is therefore a vector of the same length as there are samples, e.g.: age.

The examples before worked nicely, because it was simulated data without any covariates. Now we will be using data where one covariate has been modelled additionally, your task is now to calculate the p-value with and with the use of covariates and plot the results.

```
setwd("/data/day5/eqtl_intro")
expr_cov = read.table("simulated/sim_expression2.tab", sep="\t",
                      header=TRUE, row.names = 1)
covariates = read.table("simulated/sim_covariates.tab", sep="\t",
                        header=TRUE, row.names = 1)
```

Task

Calculate the linear regression with and without the covariates for combination of gene 10 with snp 10. Then plot the results from the linear regression with and without the use of covariates. What are the differences in the plot and in the summary?

Answer

```r
lm_10_10 = lm(expr_cov[,"gene_10"] ~ gt[,"snp_10"])
summary(lm_10_10)
```

```
##
## Call:
## lm(formula = expr_cov[, "gene_10"] ~ gt[, "snp_10"])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -21.8886  -4.4990  -0.2714   5.1006  20.2046
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       5.8321     0.7492   7.784 1.16e-13 ***
## gt[, "snp_10"]    2.9265     0.6018   4.863 1.87e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.391 on 298 degrees of freedom
## Multiple R-squared:  0.07353,    Adjusted R-squared:  0.07042
## F-statistic: 23.65 on 1 and 298 DF,  p-value: 1.873e-06
```

```r
lm_10_10_covs = lm(expr_cov[,"gene_10"] ~ gt[,"snp_10"] + as.matrix(covariates))
summary(lm_10_10_covs)
```

```
##
## Call:
## lm(formula = expr_cov[, "gene_10"] ~ gt[, "snp_10"] + as.matrix(covariates))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4494 -0.6804  0.0657  0.6722  2.5683
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   6.82775    0.11393  59.928  < 2e-16 ***
## gt[, "snp_10"]                1.65776    0.09172  18.074  < 2e-16 ***
## as.matrix(covariates)var_1    2.50006    0.06416  38.967  < 2e-16 ***
## as.matrix(covariates)var_2    2.24993    0.07045  31.935  < 2e-16 ***
## as.matrix(covariates)var_3    2.21608    0.06487  34.163  < 2e-16 ***
## as.matrix(covariates)var_4    2.20670    0.06667  33.097  < 2e-16 ***
## as.matrix(covariates)var_5    2.18179    0.06466  33.740  < 2e-16 ***
## as.matrix(covariates)var_6    1.90274    0.06336  30.031  < 2e-16 ***
## as.matrix(covariates)var_7    1.87298    0.06779  27.631  < 2e-16 ***
## as.matrix(covariates)var_8    1.67675    0.06664  25.163  < 2e-16 ***
## as.matrix(covariates)var_9    1.64621    0.06771  24.311  < 2e-16 ***
## as.matrix(covariates)var_10   1.55134    0.06076  25.531  < 2e-16 ***
## as.matrix(covariates)var_11   1.48276    0.06598  22.475  < 2e-16 ***
## as.matrix(covariates)var_12   1.31934    0.06802  19.396  < 2e-16 ***
## as.matrix(covariates)var_13   1.23396    0.06564  18.799  < 2e-16 ***
## as.matrix(covariates)var_14   1.14690    0.06383  17.967  < 2e-16 ***
## as.matrix(covariates)var_15   1.13261    0.06344  17.854  < 2e-16 ***
## as.matrix(covariates)var_16   0.96792    0.06647  14.562  < 2e-16 ***
## as.matrix(covariates)var_17   0.75899    0.06875  11.039  < 2e-16 ***
## as.matrix(covariates)var_18   0.70516    0.06483  10.876  < 2e-16 ***
## as.matrix(covariates)var_19   0.64040    0.06584   9.727  < 2e-16 ***
## as.matrix(covariates)var_20   0.50829    0.06822   7.450 1.18e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

As we can see the slope is different when covariates are incorporated in the model. This addition modifies the estimated slope and it's associated p-value. The lower p-value when using covariates indicates that the regression describes the data more accurately. If we were to apply multiple testing correction on the p-values it could happen that the first case would not reach significance even though biologically there is an effect. The reason for this is if we don't include covariates we try to explain many sources of variation by just the genotype, which clearly cannot account for all the individual sources, therefore the p-value is higher and also the estimated effect size will be wrong.

As a side note: There is another way to deal with covariates, instead of including them in the model directly one can "clean" the expression data from the effects caused by covariates by "regressing them out". This can be done by building a linear model only on the covariates and then removing the covariates from the expression data using the estimated effect sizes. The eQTL calculation can then be performed by directly just using the genotype data.

An eQTL analysis

Now the basics are defined and we can focus on real expression and genotyping data. For this example we will be using the famous R package "MatrixEQTL". It is a very fast way to perform a very basic eQTL analysis.

```r
library(MatrixEQTL)
```

Importing genotype and expression data

Genotype and expression data come in all sorts of flavours and great attentation has to be drawn on the preprocessing of this data. In this course it will not be possible to cover this in detail, but common input formats for genotypes are VCF, PLINK files, or even other custom files which give the genotype of each sample on all queried genomic positions. Expression data can also be made available in various formats depending on the underlying technology (RNAseq or expression micro array). Most available eQTL datasets are available only from microarray experiments. R-packages like lumi are very helpful for handling expression microarray data.

In this exercise we will be using a simulated dataset, based on genotypes of a part of the HapMap 2 data. This simulation will be different from real data mainly in:

- Here we only have 81 individuals, normally eQTL studies have to have more than 200 samples (detection power, noise)
- We analyse here only 500 genes, normally around 20,000 genes are analysed.

The more samples and genes, the longer the calculation will take, so we are using a subset here. The data preprocessing has already been done for you and you can load the expression and genotyping data directly into MatrixEQTL.

Loading data

Data is stored in the files: "eqtl/genotypes.txt" and "eqtl/expression.txt" where the columns are samples and the rows are SNPs or genes, respectively. The input files for MatrixEQTL should have samples as columns for genotype and phenotype matrices and their order has to be identical.

Task

Using the MatrixEQTL functions import gene expression and genotype data as "SlicedData" objects. Verify that the samples are ordered correctly for genotype and phenotype

Answer

```r
library(MatrixEQTL)
setwd("/data/day5/eqtl_intro")
snp_values = read.table("eqtl/genotypes.txt", row.names=1, header=TRUE)
gene_values = read.table("eqtl/expression.txt", row.names=1, header=TRUE)
snps <- SlicedData$new()
snps$CreateFromMatrix(as.matrix(snp_values))
genes <- SlicedData$new()
genes$CreateFromMatrix(as.matrix(gene_values))
```

Now we make sure that the sample order is identical:

```r
all(colnames(snps) == colnames(genes))
```

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

In order to perform an eQTL analysis we also must know where the individual SNPs and genes are located in the genome so that we can distinguish between cis- and trans-eQTL analyses. Therefore we need to load the SNP and gene annotation files.

```r
setwd("/data/day5/eqtl_intro")
probePos <- read.table("eqtl/probe_loc_hg19.txt", sep="\t", header=TRUE)
snpPos <- read.table("eqtl/gt_anno.txt", sep="\t", header=TRUE)
```

Task

Take a look at the individual files to familiarise yourself with the layout and content.

cis-eQTL analysis

To perform an eQTL analysis we don't only need to know the genotype and gene expression values for every sample, but also the genomic positions of genes and SNPs. This is necessary to define which SNPs should be tested against which genes. For cis-eQTL analyses SNPs in proximity to the gene are chosen and for trans-eQTL analyses SNPs further away are being chosen. The window around the gene body is conventionally chosen as 500kb-1Mb.

Task

Perform a cis-eQTL analysis using the "Matrix_eQTL_main" function. Check the manual (by typing ?Matrix_eQTL_main) how to set the p-value output thresholds in order to get cis (local) and trans (distant) eQTLs. Print a list of found eQTLs and try to understand the meaning of the rows in the table and the individual columns.

Answer

```
eQTL <- Matrix_eQTL_main(snps, genes,
        output_file_name=NULL,
        output_file_name.cis=NULL,
        pvOutputThreshold.cis=1e-3, snpspos=as.data.frame(snpPos[,1:3]),
        genepos=as.data.frame(probePos[1:4]))
```

```
## Matching data files and location files
## 500 of 500  genes matched
## 41462 of 41462  SNPs matched
## Task finished in  0.056  seconds
## Reordering genes
##
## Task finished in  0.157  seconds
## Processing covariates
## Task finished in  0.001  seconds
## Processing gene expression data (imputation, residualization, etc.)
## Task finished in  0.003  seconds
## Creating output file(s)
## Task finished in  0.01  seconds
## Performing eQTL analysis
## 100.00% done, 4,149 cis-eQTLs, 639 trans-eQTLs
## Task finished in  2.402  seconds
##
```

```
head(eQTL$cis$eqtl)
```

```
##         snps               gene statistic       pvalue          FDR
## 1 rs12549085 ENSG00000221542.1 -14.16024 2.214255e-23 6.060705e-18
## 2  rs7015818 ENSG00000252067.1  13.59721 2.173704e-22 2.974855e-17
## 3  rs7001819 ENSG00000255144.1  13.08122 1.825001e-21 1.665088e-16
## 4 rs10503829 ENSG00000104290.6 -12.83505 5.093843e-21 2.788502e-16
## 5 rs10087762 ENSG00000104290.6  12.83505 5.093843e-21 2.788502e-16
## 6  rs2686187 ENSG00000255144.1 -12.74819 7.329855e-21 3.343794e-16
##        beta
## 1 -1.286131
## 2  1.760449
## 3  1.607049
## 4 -1.324083
## 5  1.324083
## 6 -1.593782
```

The first two columns are derived from the rownames given in the snps and genes matrices. The "snps" column displays the ids of SNPs that have been identified. In the gene column gene ids are displayed. The "statistic" column shows the statistic on top of which the values in the "p-value" column are calculated. We will discuss the content of "FDR" soon and the "beta" column gives the effect size which is the slope of the linear regression.

Multiple testing correction

Whenever multiple statistical tests are performed, a multiple testing correction has to be performed. This is necessary because many hypotheses are tested. Therefore each calculated association p-value has to be corrected for multiple testing. MatrixEQTL does this for you automatically and returns the corrected p-value as a false discovery rate (FDR). Common thresholds on FDR are 5% or 10%.

Interpreting eQTL results

LD

Linkage disequilibrium (LD) is a very important effect that plays a big role in genetic association studies. It describes the effect that genetic variants are not always inherited independently due to recombination patterns during reproduction. SNPs in LD are inherited in similar patterns and therefore can explain gene expression in similar ways. This means that LD makes it harder for association studies to identify one single SNP being associated with altered gene expression. Also it is possible that the combination of SNPs (as a haplotype) causes differences in gene expression and not only one single SNP. Watch this video which explains the basics of LD: https://elearning.cpp.edu/learning-objects/linkage-disequilibrium/.

Selecting eQTLs

Commonly one selects at most one associated SNP per gene. If there are many SNPs associated with a gene it is most likely that those SNPs are highly linked to each other ("in high LD") and therefore they describe the same effect. There are still cases in which genes are regulated by different SNPs independently, this cannot be seen in table produced by MatrixEQTL. In this course we will not try to identify the independent lead eQTL signals.

Task

From the cis-eQTL results identify which SNPs are (significantly) associated with which genes at a maximum FDR of 10%. Print a table in which only the lead SNP per gene is given. Also add the MAF for every SNP in the table.

Answer

There many ways to achieve that, I will be using the plyr package here.

```
cis_eqtl_res = eQTL$cis$eqtls
cis_eqtl_res = cis_eqtl_res[cis_eqtl_res$FDR < 0.1,]

top_eqtls = cis_eqtl_res[order(cis_eqtl_res$pvalue),]
top_eqtls = top_eqtls[!duplicated(top_eqtls$gene),]

mafs = apply(as.matrix(snp_values),1,mean)/2
mafs = data.frame(snps=names(mafs), maf = mafs)
top_eqtls = merge(top_eqtls, mafs, by="snps")
top_eqtls = top_eqtls[order(top_eqtls$FDR),]

head(top_eqtls)
```

```
##            snps               gene statistic       pvalue          FDR
## 38  rs12549085 ENSG00000221542.1 -14.16024 2.214255e-23 6.060705e-18
## 212  rs7015818 ENSG00000252067.1  13.59721 2.173704e-22 2.974855e-17
## 203  rs7001819 ENSG00000255144.1  13.08122 1.825001e-21 1.665088e-16
## 12  rs10503829 ENSG00000104290.6 -12.83505 5.093843e-21 2.788502e-16
## 90   rs2137790 ENSG00000239065.1 -12.25818 5.804971e-20 1.986120e-15
## 179  rs4925810 ENSG00000147804.5 -12.28367 5.208902e-20 1.986120e-15
##          beta       maf
## 38  -1.286131 0.3703704
## 212  1.760449 0.3086420
## 203  1.607049 0.3395062
## 12  -1.324083 0.6666667
## 90  -1.497434 0.2037037
## 179 -1.367674 0.3086420
```

Presenting eQTL results

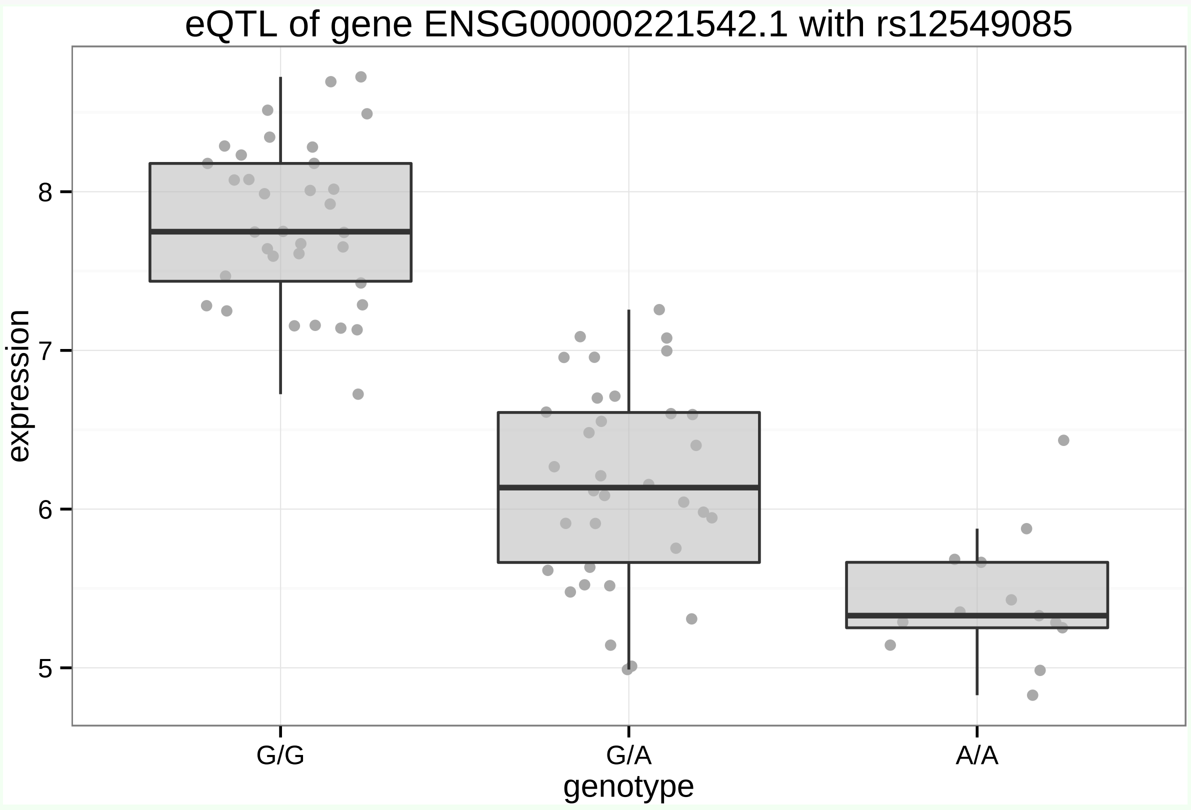
There are a few standard plots which are common in eQTL analyses. We already produced one of them earlier where we plotted gene expression versus genotype. This gives a visual insight in how clear the data was and what the linear regression actually detected.

Task

Plot gene expression vs. genotype for the eqtl with the lowest association p-value. Add a representative title and label the genotype axis using the sample genotypes.

Answer

```r
# Get gene name of gene with lowest association p-value
gene_id = top_eqtls$gene[1]
# Get corresponding SNP
snp_id = top_eqtls[top_eqtls$gene == gene_id,"snps"][1]
data = data.frame(t(snp_values[snp_id,]), t(gene_values[gene_id,]))
# Get reference and alternative allele of the SNP
ref_alt = unlist(snpPos[snpPos$rsid== snp_id, c("ref", "alt")])
# Prepare the genotype labels
gt_states= c(paste(ref_alt[1], ref_alt[1], sep="/"), paste(ref_alt[1],
    ref_alt[2], sep="/"), paste(ref_alt[2], ref_alt[2], sep="/"))
gt_states = factor(gt_states, levels=gt_states)
# Assign the labels
data$gt = gt_states[data[,snp_id]+1]
# Subset to only genotype labels and expression
data = data[,c("gt", gene_id)]
colnames(data) = c("genotype", "expression")
# Plot
p = ggplot(data, aes(genotype, expression)) +
        ggtitle(paste("eQTL of gene",gene_id, "with",snp_id))+
        geom_jitter(colour="darkgrey", position=position_jitter(width=0.25)) +
        geom_boxplot(outlier.size=0, alpha=0.6, fill="grey") + theme_bw()
print(p)
```



Manhanttan plots

Manhattan plots are a way to depict association p-values of multiple SNPs at once. They are also very common in GWAS. Manhattan plots are an important measure of interpretation of results, such as interpreting eQTL signals in terms of LD.

Task

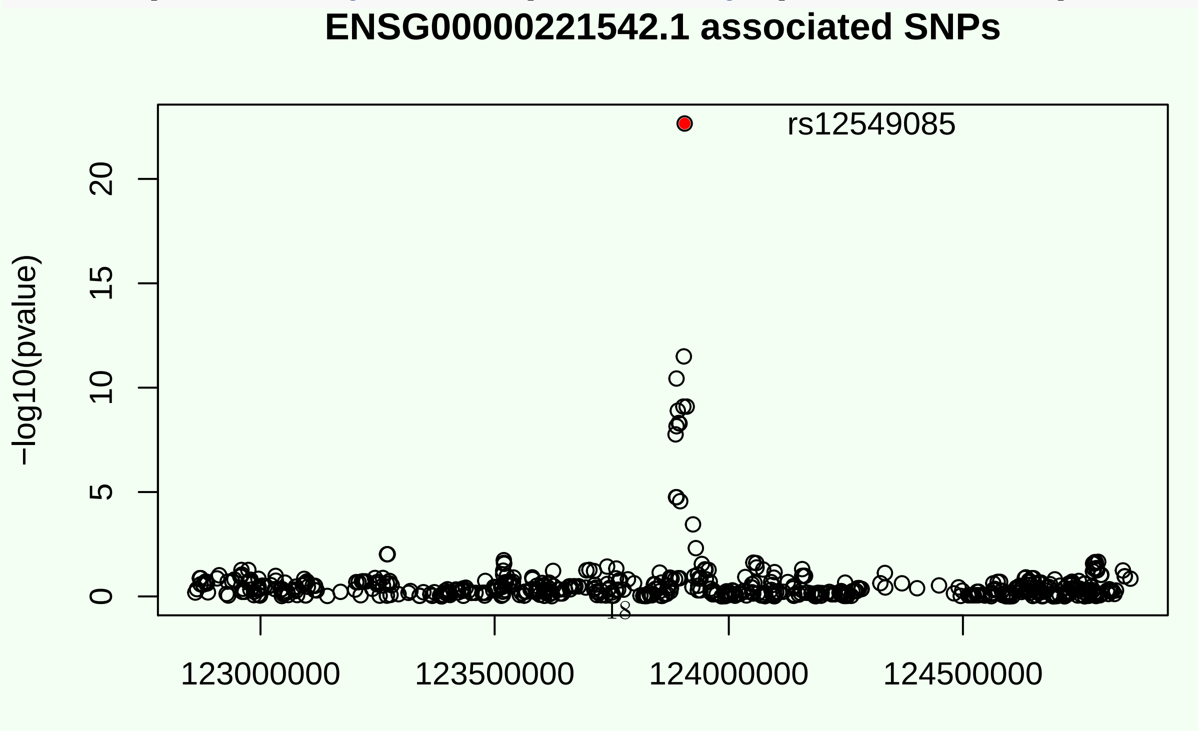
Generate a manhattan plot for gene ENSG00000221542.1, plotting the base-pair position on the x-axis and the -log__10__(p-value) of the SNP in the y axis. Manhattan plots usually depict all tested SNPs, not only the ones passing a certain p-value threshold. Therefore try to obtain all the association p-values for all tested SNPs for gene ENSG00000221542.1.

Answer

```
gene_id = "ENSG00000221542.1"
single_gene_exp = SlicedData$new()
single_gene_exp$CreateFromMatrix(as.matrix(gene_values[gene_id,, drop=FALSE]))
single_cis_eqtl_res = Matrix_eQTL_main(snps, single_gene_exp,
        output_file_name.cis = NULL,
        output_file_name = NULL,
        pvOutputThreshold.cis=1, snpspos=as.data.frame(snpPos[,1:3]),
        genepos=as.data.frame(probePos[1:4]))
```

```
## Matching data files and location files
## 1 of 1  genes matched
## 41462 of 41462  SNPs matched
## Task finished in  0.043  seconds
## Processing covariates
## Task finished in  0.001  seconds
## Processing gene expression data (imputation, residualization, etc.)
## Task finished in  0.002  seconds
## Creating output file(s)
## Task finished in  0.015  seconds
## Performing eQTL analysis
## 100.00% done, 630 cis-eQTLs, 0 trans-eQTLs
## No significant associations were found.
## Task finished in  0.255  seconds
##
```

```
manh_data = merge(single_cis_eqtl_res$cis$eqtl, snpPos, by.x="snps", by.y = "rsid")
manh_data =manh_data[,c("pos", "chrom", "pvalue", "snps")]
par(mfrow=c(1,1))
# Plot the Manhattanplot
with(manh_data ,plot(pos, -log10(pvalue), xlab = "genomic position (bp)",
                     main=paste(gene_id, "associated SNPs")))
# Highlight the lead SNP
with(manh_data[which.min(manh_data$pvalue),,drop=FALSE] ,
     points(pos, -log10(pvalue), pch=20, col="red"))
# Add a label to the lead SNP
with(manh_data[which.min(manh_data$pvalue),,drop=FALSE],
     text(pos + diff(range(manh_data$pos))*0.2, -log10(pvalue), labels = snps))
```



Task

Now produce manhattan plots for the top 10 cis-eQTL results and for the bottom 10 cis-eQTL in the top_eqtls matrix we have created earlier. Try to interpret the structures.

Answer

```r
for (gene_id in top_eqtls$gene[c(1:10,(nrow(top_eqtls)-10):nrow(top_eqtls))]){
  print(gene_id)
  single_gene_exp = SlicedData$new()
  single_gene_exp$CreateFromMatrix(as.matrix(gene_values[gene_id,, drop=FALSE]))
  single_cis_eqtl_res = Matrix_eQTL_main(snps, single_gene_exp,
          output_file_name.cis = NULL,
          output_file_name = NULL,
          pvOutputThreshold.cis=1, snpspos=as.data.frame(snpPos[,1:3]),
          genepos=as.data.frame(probePos[1:4]))
  manh_data = merge(single_cis_eqtl_res$cis$eqtl, snpPos, by.x="snps", by.y = "rsid")
  manh_data =manh_data[,c("pos", "chrom", "pvalue", "snps")]
  par(mfrow=c(1,1))
  # Plot the Manhattanplot
  with(manh_data ,plot(pos, -log10(pvalue), xlab = "genomic position (bp)",
                     main=paste(gene_id, "associated SNPs")))

  # Highlight the lead SNP
  with(manh_data[which.min(manh_data$pvalue),,drop=FALSE] ,
       points(pos, -log10(pvalue), pch=20, col="red"))

  # Add a label to the lead SNP
  with(manh_data[which.min(manh_data$pvalue),,drop=FALSE],
       text(pos + diff(range(manh_data$pos))*0.2, -log10(pvalue), labels = snps))
  scan(stdin())
}
```

Mostly there is a very clear eQTL signal visible, a clean peak. Variants which are similar in “height” as the lead cis-eQTL SNP, but lower are most likely SNPs in LD with the lead SNP. In some cases horizontal lines become visible which means that those variants are in very high LD among each other - they are usually inherited together. Here you can see again that when variants are in very high LD (horizontal lines) their importance for gene expression cannot be distinguished. Other methods such as fine mapping try to use information like genome segmentation to break the LD blocks into smaller fractions of being more or less likely causal.

eQTL SNP distance from the TSS

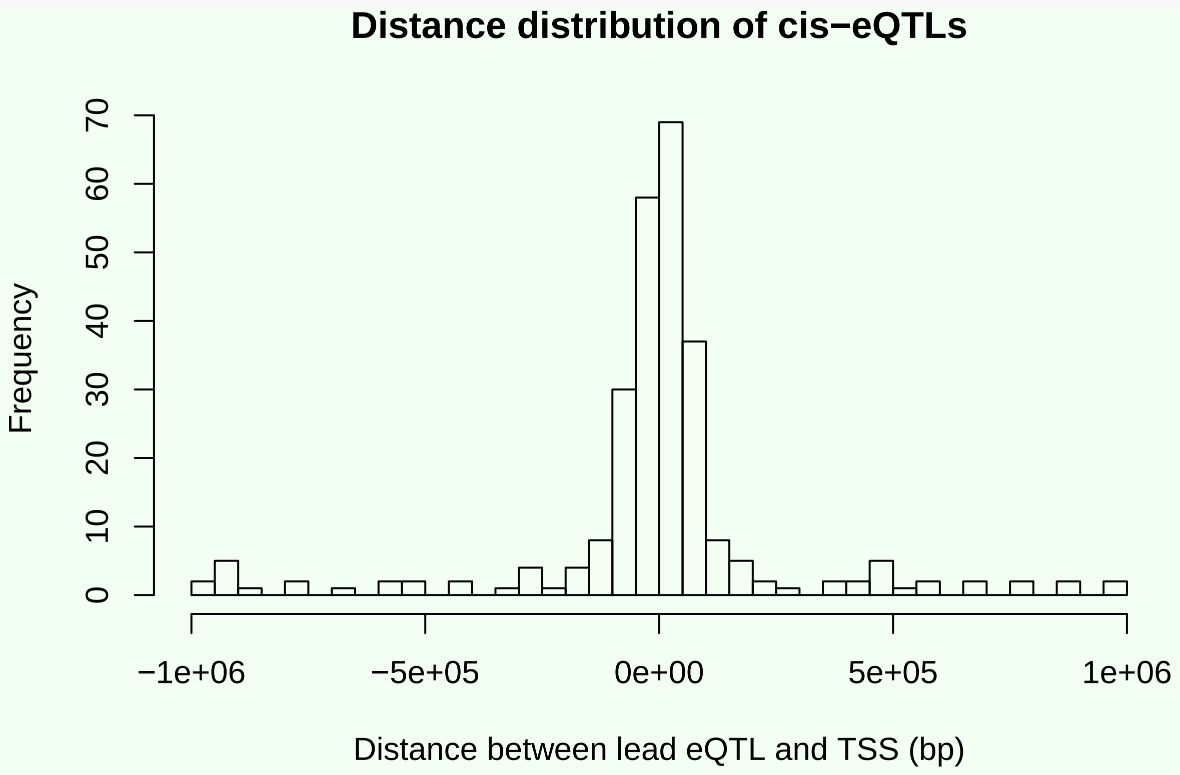
Usually cis-eQTL SNPs are located around the transcription starting site (TSS) of the associated gene. Depending on the dataset there may be a slight bias to more associations upstream the TSS. When looking at the SNP positions relative to the TSS one has to take the strand of the gene into account, as up- and downstream are always relative to the strand the gene lies on.

Task

Display the distribution of the distance between lead cis-eQTL SNPs and the TSS of the associated gene. Take the gene strand into account.

Answer

```r
top_eqtls = merge(top_eqtls,probePos, by.x = "gene", by.y="gene_ids")
top_eqtls = merge(top_eqtls,snpPos, by.x = "snps", by.y="rsid")
dtss = top_eqtls$gene_tsss - top_eqtls$pos
dtss[top_eqtls$strand == "-"] = -dtss[top_eqtls$strand == "-"]
hist(dtss, breaks=50, main="Distance distribution of cis-eQTLs",
     xlab="Distance between lead eQTL and TSS (bp)")
```



Trans-eQTL analysis

So far we have only worked with cis-eQTL analyses. Trans- as opposed to cis-analyses test genes which are linearly far away from the gene body. This increases the multiple testing burden produced by performing many more tests. Cis-eQTLs are believed to describe direct effects between a SNP and gene expression of the respective gene. Trans effects are believed to be describe indirect effects. These could be alterations of gene expression of the tested gene which is caused by e.g. other genes in a pathway of which some are directly controlled by the associated SNP. Here the question of causality comes into play, which will not be tackled in this course.

Other tools and QTL analyses

Locus zoom

Locus zoom is an online tool which produces a more informative version of Manhattan plots. For each tested SNP it adds information of LD between the individual SNPs, it displays the position of genes and one can add multiple additional layers of information such as results from ChIPseq experiments and many other annotations.

Apart from locus zoom you might also find the package gviz very interesting, which offers a wide range of functionality for plotting genomics data in R.

Other QTLs

Apart from expression QTLs also other molecular features which have been linked to genetic variation:

- expression QTL
- protein QTL
- splicing QTL
- histone mark QTL
- intron retention QTL (PSI QTL)
- …

Basically one can think of relating any kind of molecular measurement with genetic variation, but not everything will be biologically important and many things will be highly correlated. Always have a hypothesis of what should be shown before starting an analysis.

Other (e)QTL calling algorithms

There are many other algorithms for eQTL calling. Most of which are based on a linear regression. Other tools involve:

- Linear (mixed) models
  - matrixEqtl (R package)
  - FastQTL
  - LIMIX
  - PLINK
  - …
- Random Forests
  - LIMIX
  - …
- Bayesian approach
  - GOAL (R package)