QG Final Project

Yuchen Sun

2023-05-05

Note: 1. ***All codes can compile, but some needs longer time to run.*** 2. ***Redundant codes are hidden in the report, but all codes are shown in the RMarkdown file***

Data Components

There are five data files included in this project: The file `phenotypes.csv` contains the phenotype data for 344 samples and 5 genes; The file `genotypes.csv` contains SNPs data for 344 samples and 50000 genotypes with 0 = homozygote, 1 = heterozygote, and 2 = the other homozygote; The file `covars.csv` contains the information about population origin and gender of each sample; The file `gene_info.csv` contains the information about each gene; And the file `SNP_info.csv` contains the information about each SNP.

Load data

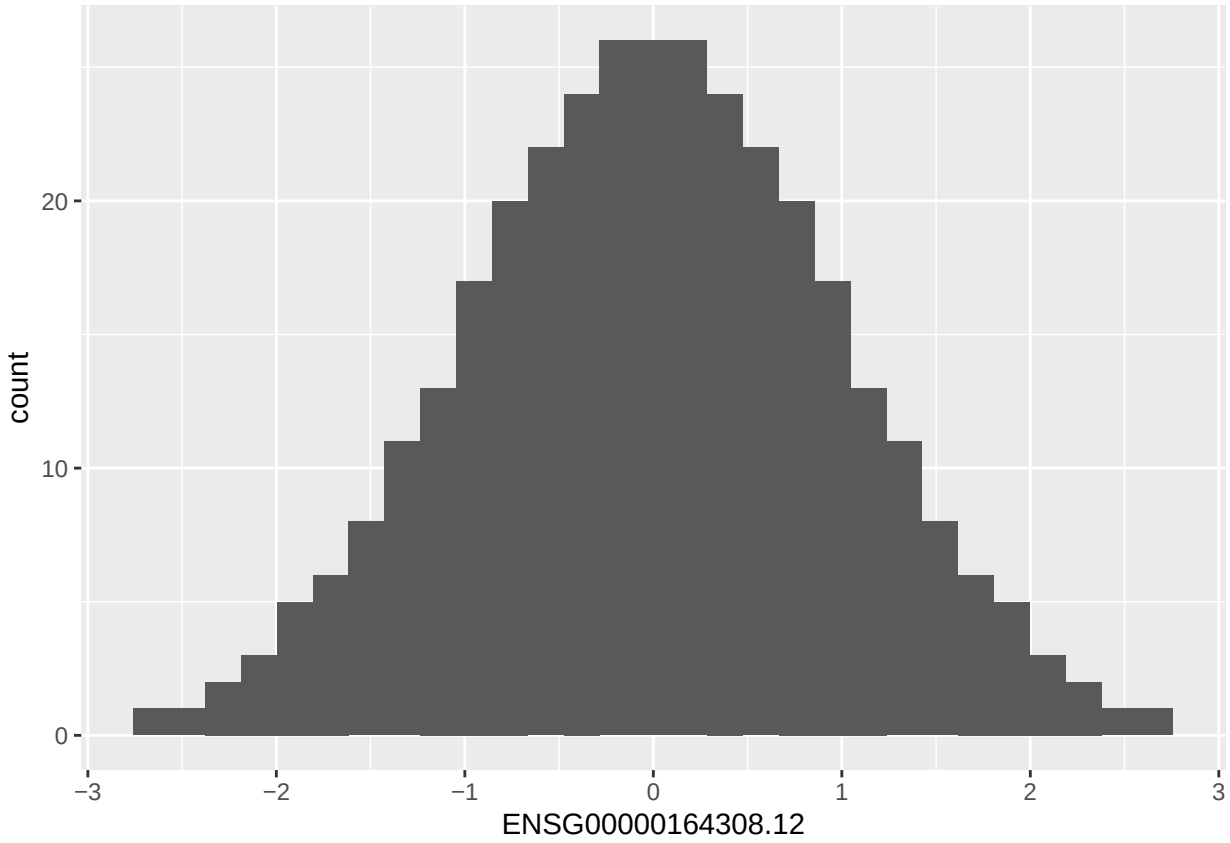
When loading the genotype data, `fread` is used because the dataset is too large for `read.csv`. After loading the covariate data, the covariates for sex is used to generate a numeric sex coding where `MALE = 1` and `FEMALE = 0`. The covariates for population is used to generate four columns for the four individual populations, with a `1` noting that the sample belongs to a certain population and a `0` indicating otherwise. Alpha is set to 0.05 and the number of samples `n` and the number of SNPs `N` are also retrieved after loading the data.

```
# load genotypes with fread
genotypes <- data.frame(fread(paste0(wd, "genotypes.csv"), header = T))
rownames <- genotypes$V1
genotypes <- genotypes[, -1]
rownames(genotypes) <- rownames
# load phenotypes
phenotypes <- read.csv(paste0(wd, "phenotypes.csv"), header = T,
    row.names = 1)
# load covariates
covars <- read.csv(paste0(wd, "covars.csv"), header = T, row.names = 1)
covars$SexCode <- as.numeric(covars$Sex == "MALE")  # MALE=1, FEMALE=0
covars <- setNames(data.frame(covars$Sex, covars$SexCode, covars$Population,
    do.call(rbind, lapply(covars$Population, function(x) as.integer(unique(covars$Population) %in%
        x)))), c("Sex", "SexCode", "Population", unique(covars$Population)))
# load SNPs info
SNPinfo <- read.csv(paste0(wd, "SNP_info.csv"), header = T)
# load gene info
geneinfo <- read.csv(paste0(wd, "gene_info.csv"), header = T)
# set constants
alpha <- 0.05
n <- dim(genotypes)[1]
N <- dim(genotypes)[2]
```

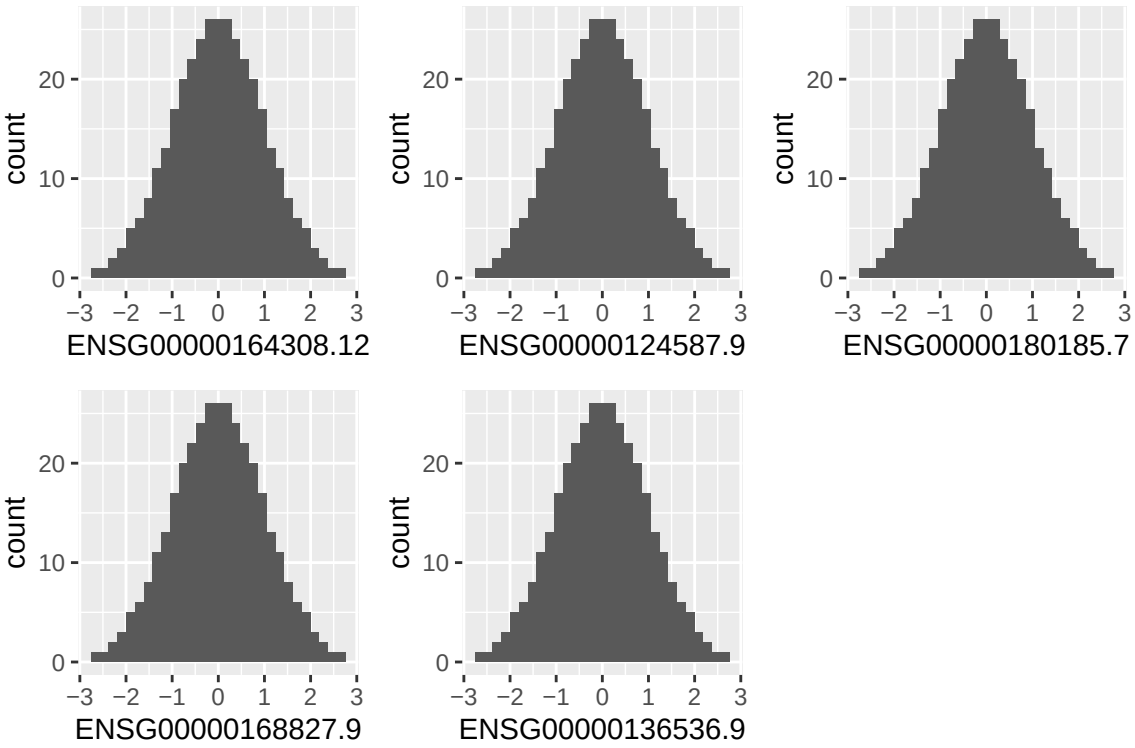
Phenotypes

To examine the quality of the phenotypes data, a histogram is plotted for each of the phenotypes. Since the phenotype data is continuous, a normal distribution is expected. The plot shows normal distribution with almost no outliers, indicating that the quality for the phenotype data is acceptable.

```
# plot for single histogram, substitute the gene id to plot
# for all phenotypes
ggplot(phenotypes, aes(ENSG00000164308.12)) + geom_histogram(bins = 30)
```




```
# histogram for each phenotype
grid.arrange(ggplot(phenotypes, aes(ENSG00000164308.12)) + geom_histogram(bins = 30),
    ggplot(phenotypes, aes(ENSG00000124587.9)) + geom_histogram(bins = 30),
    ggplot(phenotypes, aes(ENSG00000180185.7)) + geom_histogram(bins = 30),
    ggplot(phenotypes, aes(ENSG00000168827.9)) + geom_histogram(bins = 30),
    ggplot(phenotypes, aes(ENSG00000136536.9)) + geom_histogram(bins = 30),
    ncol = 3)
```



Genotypes

To filter the genotypes data, several aspects are checked to remove the samples without phenotypes or with more than 10% of missing genotypes, and genotypes with larger than 5% of missing samples or with smaller than 5% minor allele frequencies. Fortunately, no violation is detected so no samples or genotypes should necessarily be removed. The X_a and X_d matrices are then converted from the genotypes data.

```
af <- colSums(genotypes)/(2 * dim(genotypes)[1])
mafs <- data.frame(maf1 = af, maf2 = 1 - af)
maf <- as.data.frame(apply(mafs, 1, min))
colnames(maf) <- "maf"
cat("Sample without phenotype:", sum(!rownames(genotypes) %in%
    rownames(phenotypes)), "\n")
```

```
## Sample without phenotype: 0
```

```
cat("Sample with >10% missing data across all genotypes:", sum(rowSums(is.na(genotypes))/dim(genotypes)
    0.1), "\n")
```

```
## Sample with >10% missing data across all genotypes: 0
```

```
cat("Genotype with >5% missing data across all samples:", sum(colSums(is.na(genotypes))/dim(genotypes)[
    0.05), "\n")
```

```
## Genotype with >5% missing data across all samples: 0
```

```
cat("Genotype with <5% MAF:", sum(maf$maf < 0.05), "\n")
```

```
## Genotype with <5% MAF: 0
```

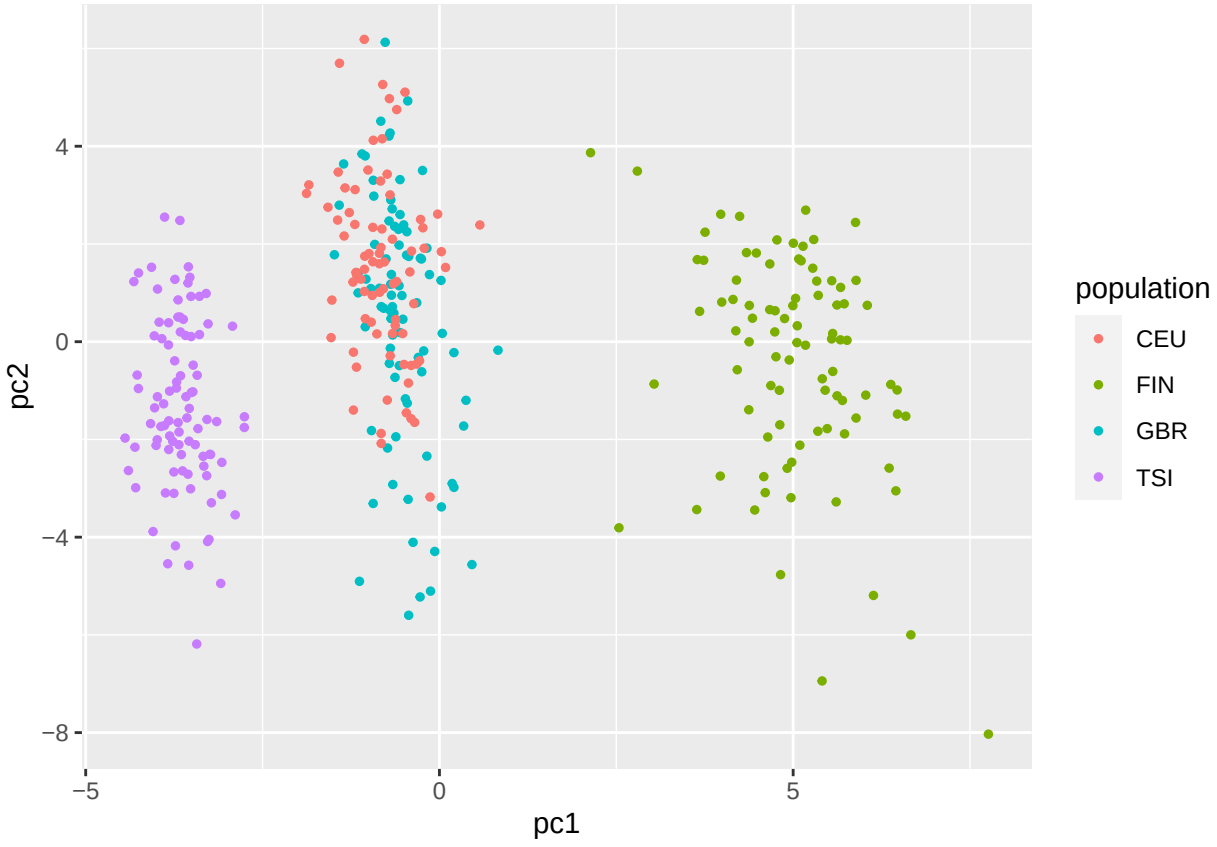
```
xa <- as.matrix(genotypes - 1)
xd <- 1 - 2 * abs(xa)
```

PCA

A PCA is conducted in order to verify that the samples can be separated according to the covariates. It is observed that the first 2 PCs can distinguish the population very well. Therefore, the first 2 PCs are also added to the covariate dataset as they can be potentially used as another type of populational covariate.

```
# pca on the xa matrix with xa*xa.T
pca.result <- prcomp(xa %*% t(xa), center = T, scale = T, tol = 0.4)
pca.df <- data.frame(pc1 = pca.result$x[, 1], pc2 = pca.result$x[,
    2], pc3 = pca.result$x[, 3], pc4 = pca.result$x[, 4], population = covars$Population,
    sex = covars$Sex)
# add first 2 PCs to covars
covars$PC1 <- as.numeric(pca.result$x[, 1])
covars$PC2 <- as.numeric(pca.result$x[, 2])
```

```
# plot for single pc plot, substitute the aes and color to
# plot for all pcs
ggplot(pca.df, aes(pc1, pc2)) + geom_point(size = 1, aes(color = population))
```



```
grid.arrange(ggplot(pca.df, aes(pc1, pc2)) + geom_point(size = 1,
    aes(color = population)), ggplot(pca.df, aes(pc3, pc4)) +
    geom_point(size = 1, aes(color = population)), ncol = 2)
```

GWAS

GWAS is conducted in two manners: without covariates and with covariates.

```
# GWAS function for without covariates
GWAS_no_cov <- function(xa_input, xd_input, pheno_input) {
    n_samples <- length(xa_input)
    X_mx <- cbind(1, xa_input, xd_input)
    MLE_beta <- ginv(t(X_mx) %*% X_mx) %*% t(X_mx) %*% pheno_input
    y_hat <- X_mx %*% MLE_beta
    SSM <- sum((y_hat - mean(pheno_input))^2)
    SSE <- sum((pheno_input - y_hat)^2)
    df_M <- ncol(X_mx) - 1
    df_E <- n_samples - ncol(X_mx)
    MSM <- SSM/df_M
    MSE <- SSE/df_E
    Fstatistic <- MSM/MSE
    pval <- pf(Fstatistic, df_M, df_E, lower.tail = F)
    return(data.table(f_statistic = Fstatistic, p = pval, model = "No Covariate"))
}
# GWAS function for with covariates
GWAS_cov <- function(xa_input, xd_input, xz_input, pheno_input) {
    n_samples <- length(xa_input)
    x_h1 <- cbind(1, xa_input, xd_input, xz_input)
    MLE_h1 <- ginv(t(x_h1) %*% x_h1) %*% t(x_h1) %*% pheno_input
    x_h0 <- cbind(1, xz_input)
    MLE_h0 <- ginv(t(x_h0) %*% x_h0) %*% t(x_h0) %*% pheno_input
    y_hat_0 <- x_h0 %*% MLE_h0
    y_hat_1 <- x_h1 %*% MLE_h1
    SSE_theta_0 <- sum((pheno_input - y_hat_0)^2)
    SSE_theta_1 <- sum((pheno_input - y_hat_1)^2)
    df_M <- ncol(x_h1) - ncol(x_h0)
    df_E <- n_samples - ncol(x_h1)
    numertator <- (SSE_theta_0 - SSE_theta_1)/df_M
    denom <- SSE_theta_1/df_E
```

```
    Fstatistic <- numertator/denom
    pval <- pf(Fstatistic, df_M, df_E, lower.tail = F)
    return(data.table(f_statistic = Fstatistic, p = pval, model = "Covariate"))
}
```

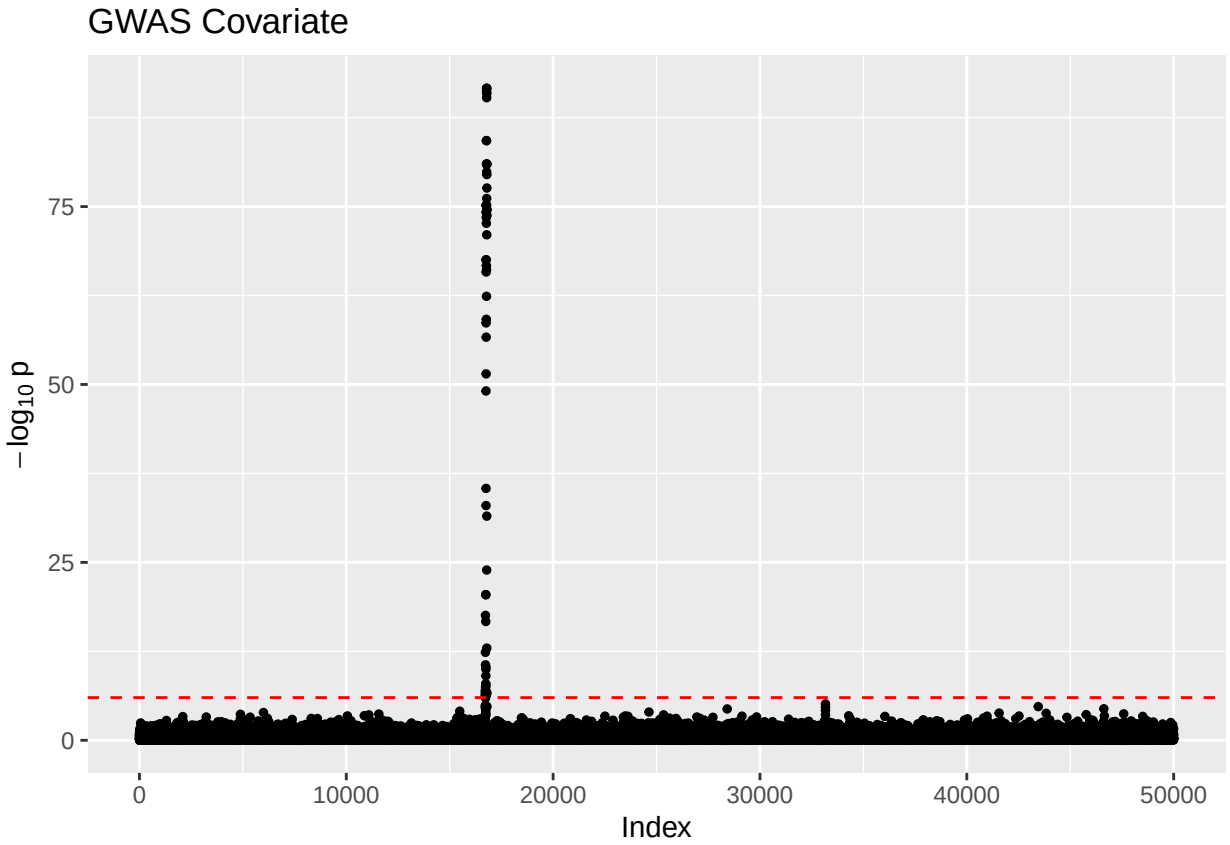
Phenotypes and Covariates Separation The 5 phenotypes are separated into single columns for conducting separate GWAS. The covariates are separated and recombined with different ways: these include single covariate of sex, single covariate of population, and combined covariates of sex and population. In the single covariate of population, individual population coding and PC coding are used. In combined covariates, sex + PC coding is used. There are 4 types of covariates coding in total. It is important to understand whether different covariate codings affect the GWAS result.

```
# separate 5 phenotypes
my.phenotype1 <- phenotypes$ENSG00000164308.12
my.phenotype2 <- phenotypes$ENSG00000124587.9
my.phenotype3 <- phenotypes$ENSG00000180185.7
my.phenotype4 <- phenotypes$ENSG00000168827.9
my.phenotype5 <- phenotypes$ENSG00000136536.9
# try different covariate codings
my.covariate1 <- covars$SexCode
my.covariate2 <- as.matrix(covars[, c("CEU", "FIN", "GBR", "TSI")])
my.covariate3 <- as.matrix(covars[, c("PC1", "PC2")])
my.covariate4 <- as.matrix(covars[, c("SexCode", "PC1", "PC2")])
```

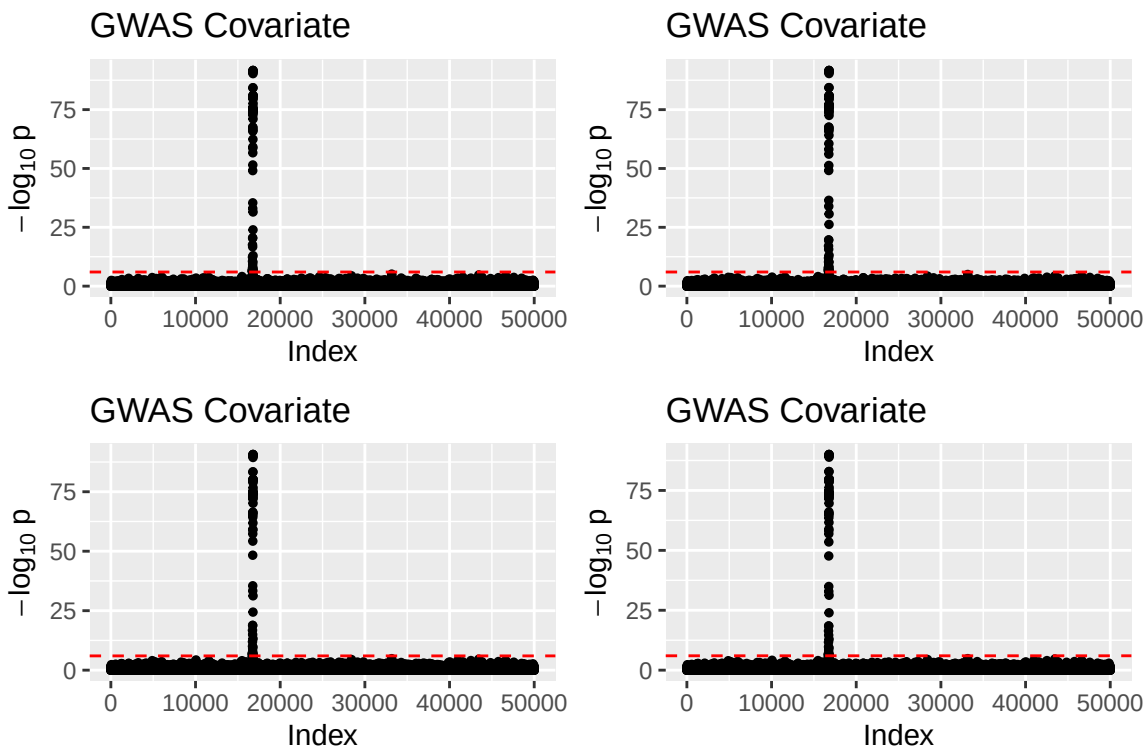
```
# helper function, produce Manhattan plot for with
# covariate
GWAS_plot_cov <- function(my_covariate, my_phenotype) {
    results <- lapply(1:ncol(xa), function(column.counter) {
        GWAS_cov(xa[, column.counter], xd[, column.counter],
            my_covariate, my_phenotype)
    }) %>%
        rbindlist() %>%
        mutate(index = 1:ncol(xa))
    p <- ggplot(results, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
        geom_hline(yintercept = -log10(alpha/N), color = "red",
            lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS Covariate")
    return(p)
}
```

Impact of Covariates All covariate codings are tested against the first phenotype, and it is discovered that the different codings of covariates has no significant impact on the result. Therefore, sex + PCs coding will be used for all further GWAS with covariates.

```
# plot for single manhattan plot, substitute the covariate
# and phenotypes to plot for others
GWAS_plot_cov(my.covariate1, my.phenotype1)
```




```
# effect of different covars coding on results
grid.arrange(GWAS_plot_cov(my.covariate1, my.phenotype1), GWAS_plot_cov(my.covariate2,
    my.phenotype1), GWAS_plot_cov(my.covariate3, my.phenotype1),
    GWAS_plot_cov(my.covariate4, my.phenotype1), ncol = 2)
```



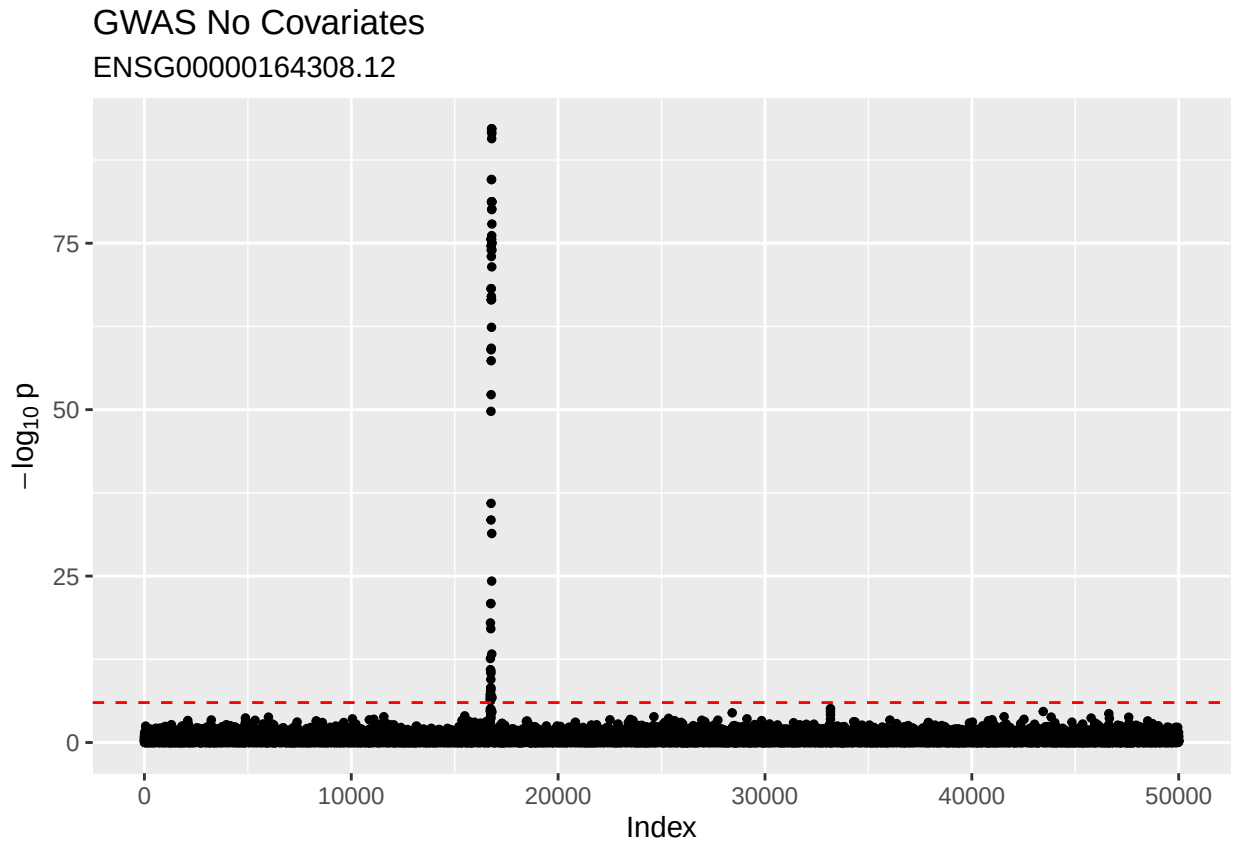
GWAS Without Covariates

```
# get result for single GWAS, substitute the phenotypes to
# perform GWAS for others
results.no_cov.1 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_no_cov(xa[, column.counter], xd[, column.counter], my.phenotype1)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
```

```
# 5 GWAS without covariates results for individual
# phenotypes
results.no_cov.1 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_no_cov(xa[, column.counter], xd[, column.counter], my.phenotype1)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.no_cov.2 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_no_cov(xa[, column.counter], xd[, column.counter], my.phenotype2)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.no_cov.3 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_no_cov(xa[, column.counter], xd[, column.counter], my.phenotype3)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
```

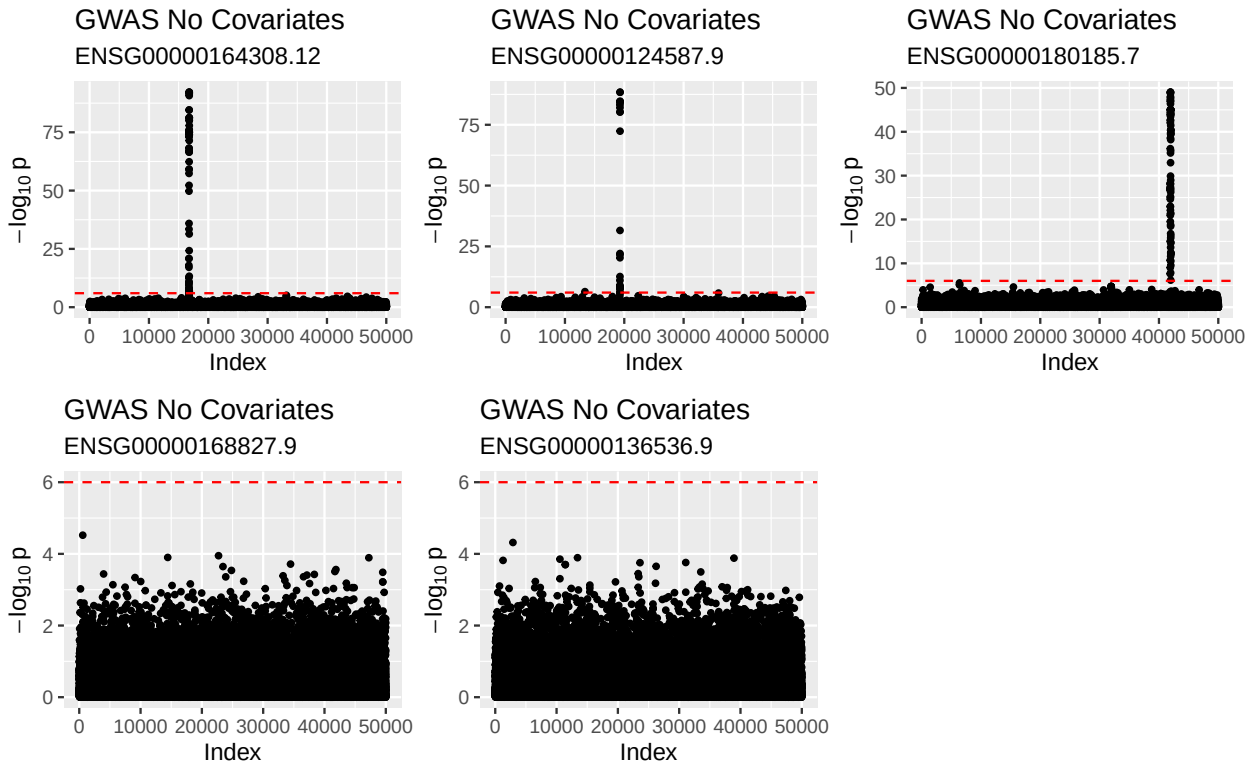
```
results.no_cov.4 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_no_cov(xa[, column.counter], xd[, column.counter], my.phenotype4)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.no_cov.5 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_no_cov(xa[, column.counter], xd[, column.counter], my.phenotype5)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
```

```
# plot for single manhattan plot, substitute the results
# and phenotypes to plot for others
ggplot(results.no_cov.1, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
    geom_hline(yintercept = -log10(alpha/N), color = "red", lty = 2) +
    labs(x = "Index", y = expression(-log[10] ~ p), title = "GWAS No Covariates",
        subtitle = "ENSG00000164308.12")
```




```
# summary of results with 5 Manhattan plot
grid.arrange(ggplot(results.no_cov.1, aes(x = index, y = -log10(p))) +
    geom_point(size = 1) + geom_hline(yintercept = -log10(alpha/N),
    color = "red", lty = 2) + labs(x = "Index", y = expression(-log[10] ~
    p), title = "GWAS No Covariates", subtitle = "ENSG00000164308.12"),
    ggplot(results.no_cov.2, aes(x = index, y = -log10(p))) +
        geom_point(size = 1) + geom_hline(yintercept = -log10(alpha/N),
```

```
        color = "red", lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS No Covariates", subtitle = "ENSG00000124587.9"),
    ggplot(results.no_cov.3, aes(x = index, y = -log10(p))) +
        geom_point(size = 1) + geom_hline(yintercept = -log10(alpha/N),
        color = "red", lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS No Covariates", subtitle = "ENSG00000180185.7"),
    ggplot(results.no_cov.4, aes(x = index, y = -log10(p))) +
        geom_point(size = 1) + geom_hline(yintercept = -log10(alpha/N),
        color = "red", lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS No Covariates", subtitle = "ENSG00000168827.9"),
    ggplot(results.no_cov.5, aes(x = index, y = -log10(p))) +
        geom_point(size = 1) + geom_hline(yintercept = -log10(alpha/N),
        color = "red", lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS No Covariates", subtitle = "ENSG00000136536.9"),
    ncol = 3)
```



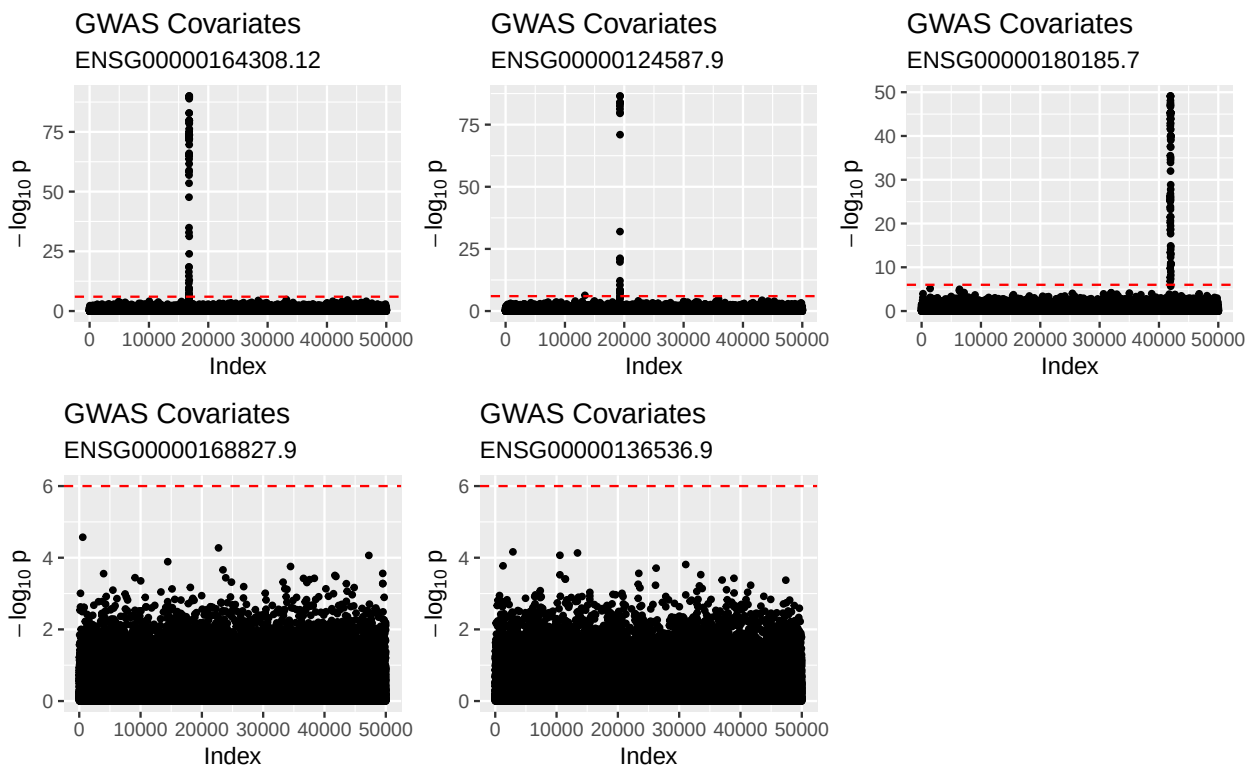
GWAS With Covariates

```
# get result for single GWAS, substitute the covariates and
# phenotypes to perform GWAS for others
results.cov.1 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_cov(xa[, column.counter], xd[, column.counter], my.covariate4,
        my.phenotype1)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
```

```
# 5 GWAS with covariates results for individual phenotypes
results.cov.1 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_cov(xa[, column.counter], xd[, column.counter], my.covariate4,
        my.phenotype1)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.cov.2 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_cov(xa[, column.counter], xd[, column.counter], my.covariate4,
        my.phenotype2)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.cov.3 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_cov(xa[, column.counter], xd[, column.counter], my.covariate4,
        my.phenotype3)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.cov.4 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_cov(xa[, column.counter], xd[, column.counter], my.covariate4,
        my.phenotype4)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
results.cov.5 <- lapply(1:ncol(xa), function(column.counter) {
    GWAS_cov(xa[, column.counter], xd[, column.counter], my.covariate4,
        my.phenotype5)
}) %>%
    rbindlist() %>%
    mutate(index = 1:ncol(xa))
```

```
# summary of results with 5 Manhattan plot
grid.arrange(ggplot(results.cov.1, aes(x = index, y = -log10(p))) +
    geom_point(size = 1) + geom_hline(yintercept = -log10(alpha/N),
    color = "red", lty = 2) + labs(x = "Index", y = expression(-log[10] ~
    p), title = "GWAS Covariates", subtitle = "ENSG00000164308.12"),
    ggplot(results.cov.2, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
        geom_hline(yintercept = -log10(alpha/N), color = "red",
            lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS Covariates", subtitle = "ENSG00000124587.9"),
    ggplot(results.cov.3, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
        geom_hline(yintercept = -log10(alpha/N), color = "red",
            lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS Covariates", subtitle = "ENSG00000180185.7"),
    ggplot(results.cov.4, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
        geom_hline(yintercept = -log10(alpha/N), color = "red",
            lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS Covariates", subtitle = "ENSG00000168827.9"),
    ggplot(results.cov.5, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
        geom_hline(yintercept = -log10(alpha/N), color = "red",
            lty = 2) + labs(x = "Index", y = expression(-log[10] ~
        p), title = "GWAS Covariates", subtitle = "ENSG00000136536.9"),
```

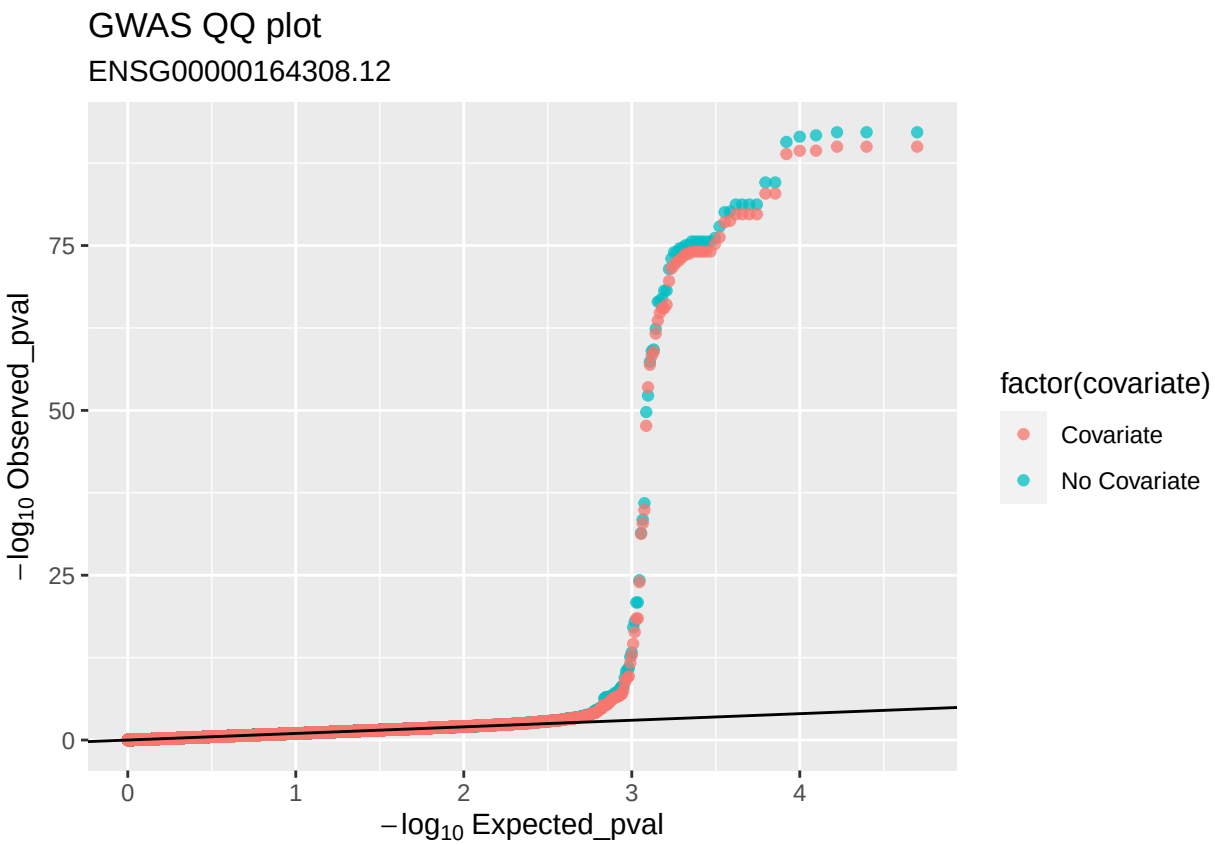
```
    ncol = 3)
```



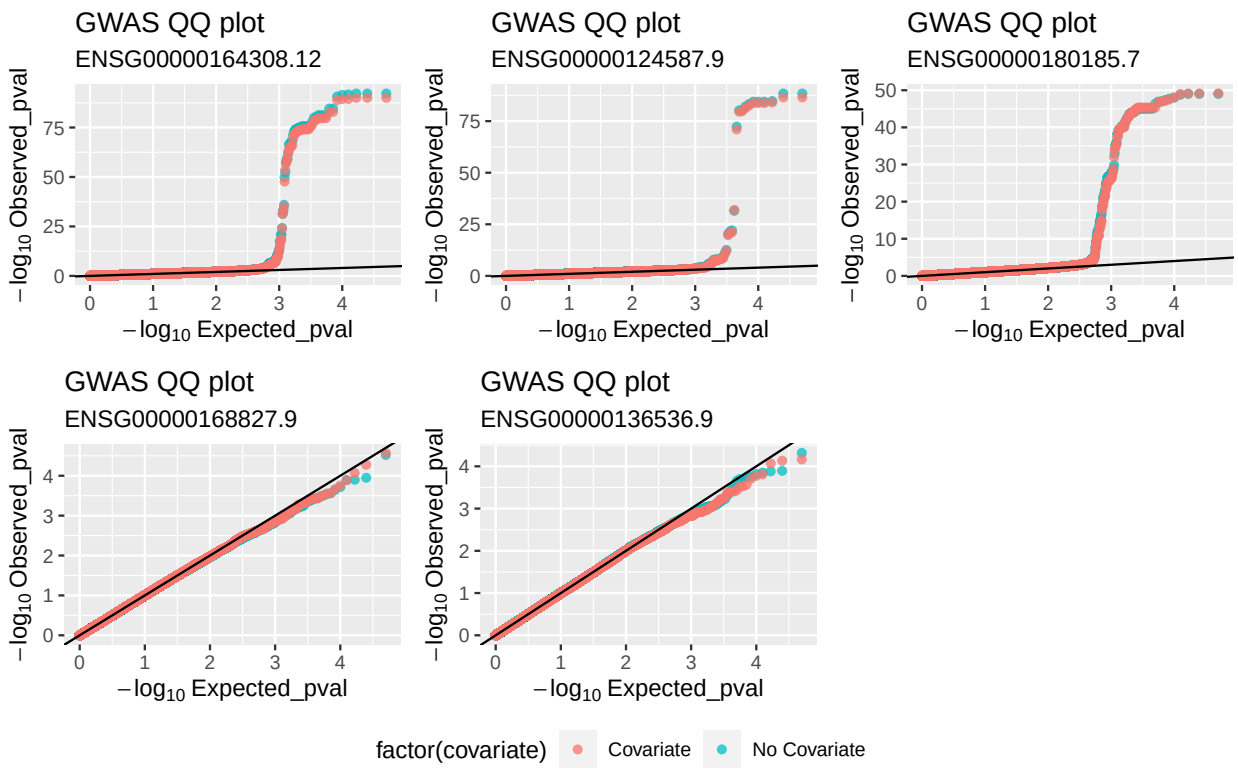
QQ Plots

```
# helper function, produce QQ plot for GWAS analysis
GWAS_qqplot <- function(res_no_cov, res_cov) {
    nocov_obs_pvals <- sort(res_no_cov$p)
    cov_obs_pvals <- sort(res_cov$p)
    expected_pvals <- qunif(seq(0, 1, length.out = length(cov_obs_pvals) +
        2), min = 0, max = 1)
    expected_pvals <- expected_pvals[expected_pvals != 0 & expected_pvals !=
        1]
    p_df <- rbind(data.frame(observed = -log10(nocov_obs_pvals),
        expected = -log10(expected_pvals), covariate = "No Covariate"),
        data.frame(observed = -log10(cov_obs_pvals), expected = -log10(expected_pvals),
            covariate = "Covariate"))
    p <- ggplot(p_df, aes(x = expected, y = observed)) + geom_point(alpha = 0.75,
        aes(colour = factor(covariate))) + geom_abline(intercept = 0,
        slope = 1) + labs(x = expression(-log[10] ~ Expected_pval),
        y = expression(-log[10] ~ Observed_pval), title = "GWAS QQ plot")
    return(p)
}
```

```
# plot for single QQ plot, substitute the results and
# phenotypes to plot for others
GWAS_qqplot(results.no_cov.1, results.cov.1) + labs(subtitle = "ENSG00000164308.12")
```

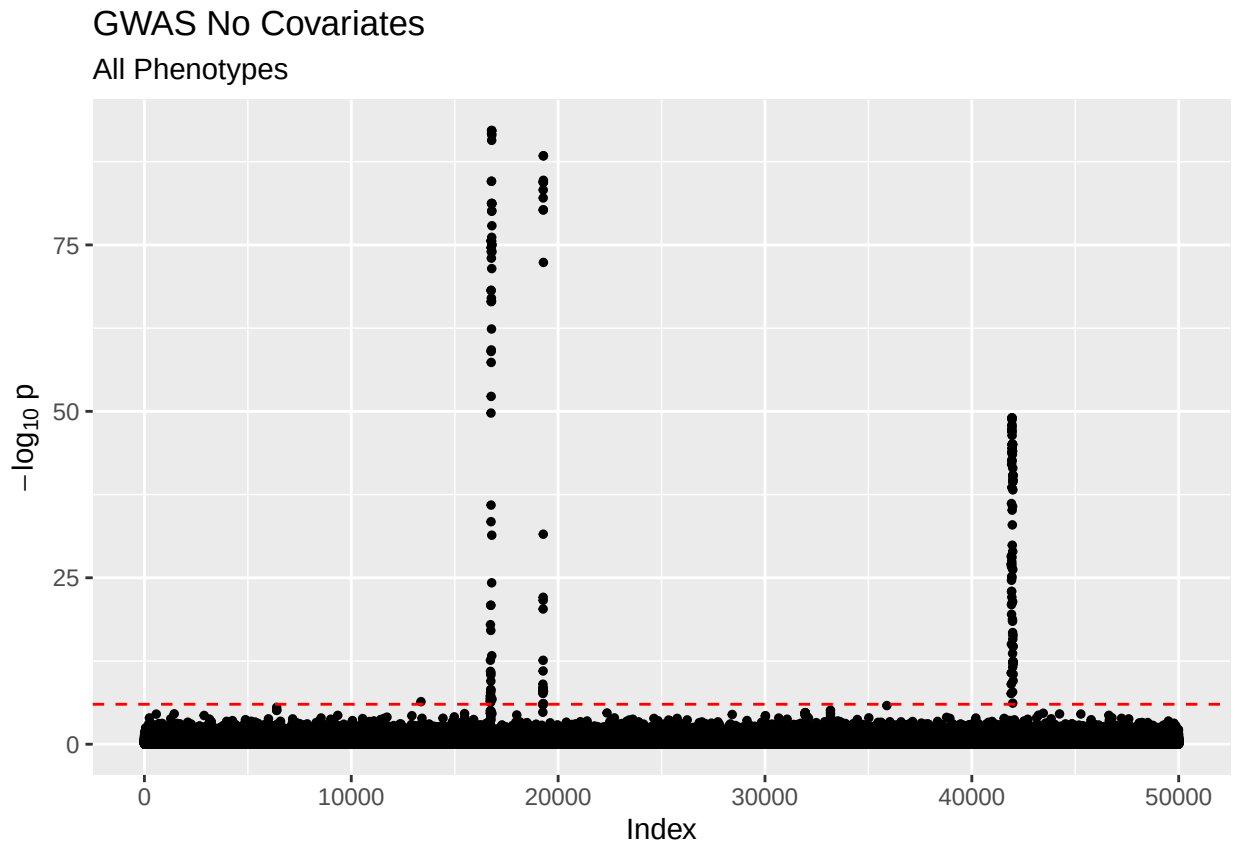



```
# summary of results with 5 QQ plot
ggarrange(GWAS_qqplot(results.no_cov.1, results.cov.1) + labs(subtitle = "ENSG00000164308.12"),
    GWAS_qqplot(results.no_cov.2, results.cov.2) + labs(subtitle = "ENSG00000124587.9"),
    GWAS_qqplot(results.no_cov.3, results.cov.3) + labs(subtitle = "ENSG00000180185.7"),
    GWAS_qqplot(results.no_cov.4, results.cov.4) + labs(subtitle = "ENSG00000168827.9"),
    GWAS_qqplot(results.no_cov.5, results.cov.5) + labs(subtitle = "ENSG00000136536.9"),
    ncol = 3, nrow = 2, common.legend = T, legend = "bottom")
```

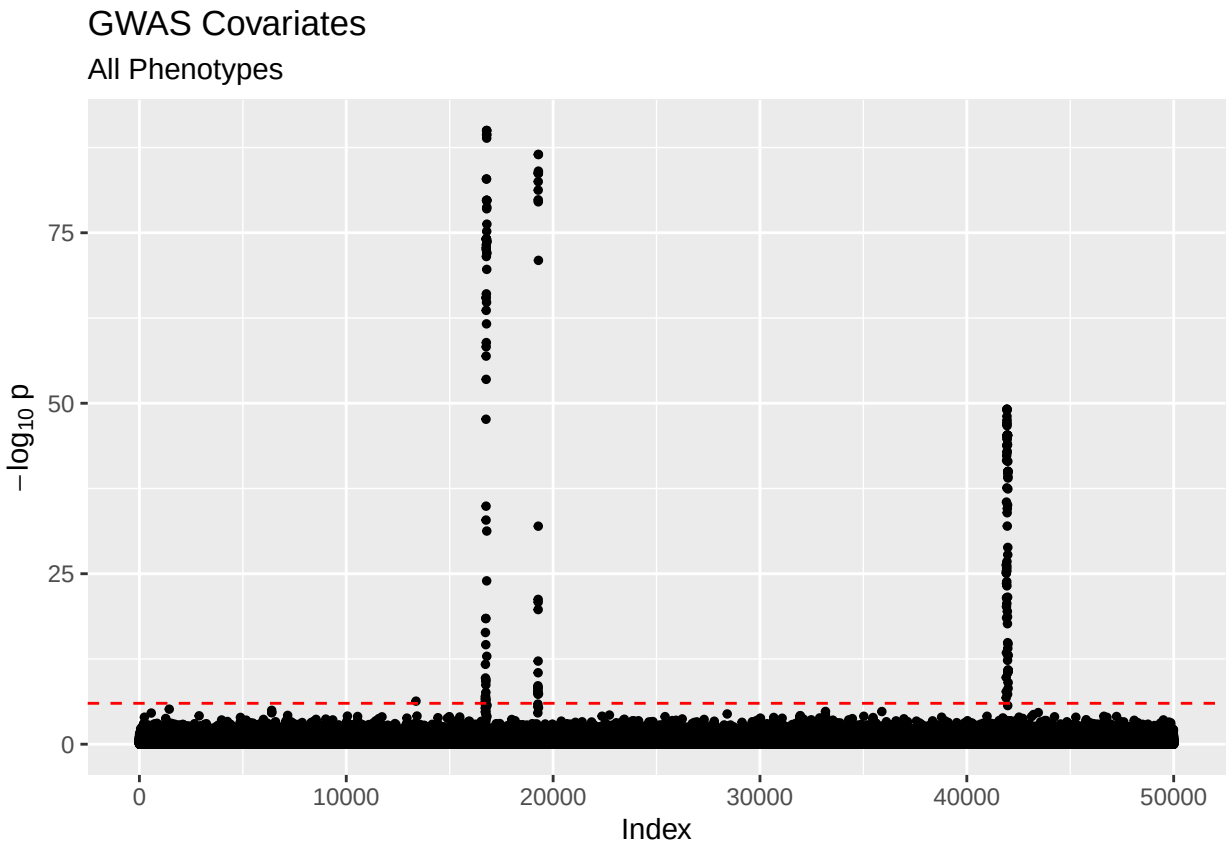



```
# combine all GWAS without covariates results
results.no_cov <- rbind(results.no_cov.1, results.no_cov.2, results.no_cov.3,
    results.no_cov.4, results.no_cov.5)
# combine all GWAS with covariates results
results.cov <- rbind(results.cov.1, results.cov.2, results.cov.3,
    results.cov.4, results.cov.5)
```

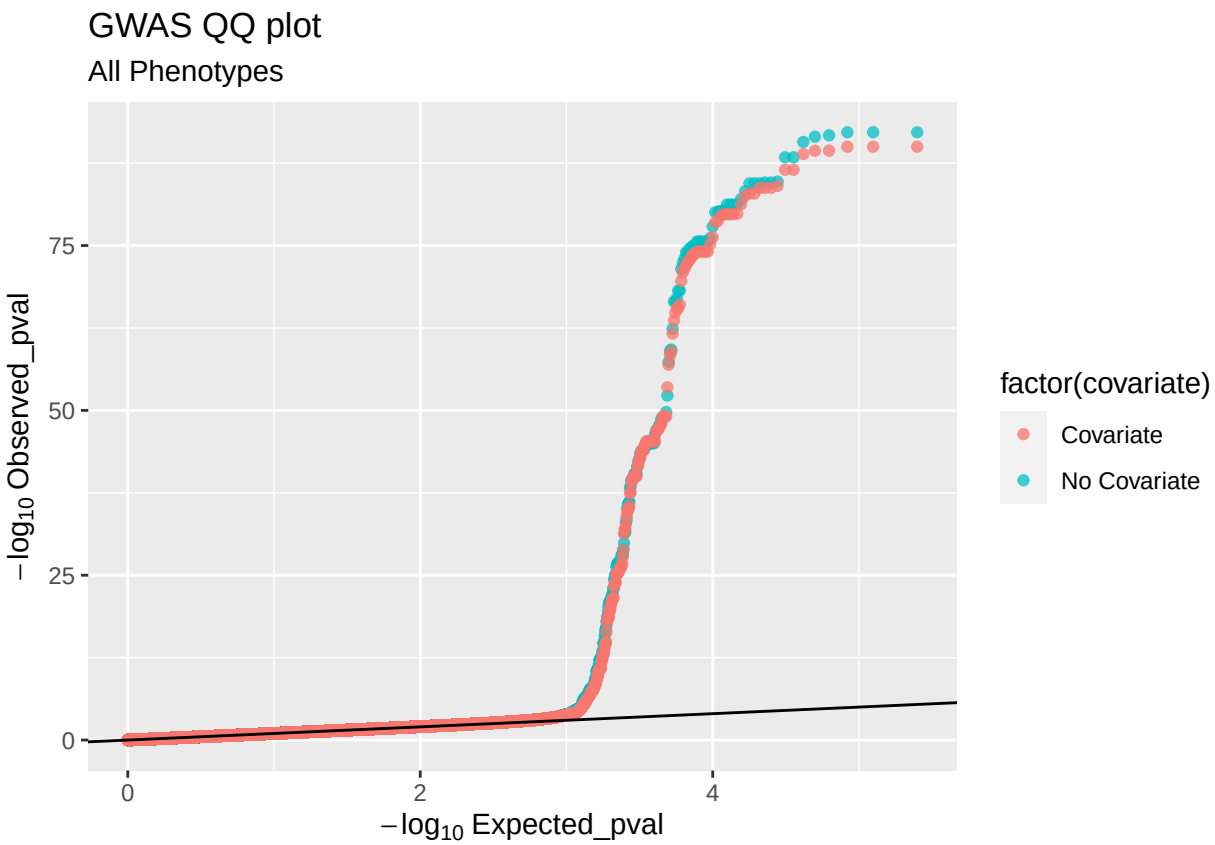
```
# summary of results with single Manhattan plot
ggplot(results.no_cov, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
    geom_hline(yintercept = -log10(alpha/N), color = "red", lty = 2) +
    labs(x = "Index", y = expression(-log[10] ~ p), title = "GWAS No Covariates",
        subtitle = "All Phenotypes")
```




```
# summary of results with single Manhattan plot
ggplot(results.cov, aes(x = index, y = -log10(p))) + geom_point(size = 1) +
    geom_hline(yintercept = -log10(alpha/N), color = "red", lty = 2) +
    labs(x = "Index", y = expression(-log[10] ~ p), title = "GWAS Covariates",
        subtitle = "All Phenotypes")
```




```
# summary of results with single QQ plot
GWAS_qqplot(results.no_cov, results.cov) + labs(subtitle = "All Phenotypes")
```



Analysis

All QQ plot shows that most of the samples follow the null trend, with a tail towards the end corresponding to the significant hits of GWAS results. This means that the GWAS analysis is valid. However, from the Manhattan plot and the QQ plot, it is observed that the addition of covariate information does not have a significant impact on the GWAS results. The same three towers are discovered in the GWAS with or without covariates (there is also a single SNP hit just above the threshold but is ignored): the first phenotype (`ENSG00000164308.12`) produced a tower at chromosome 5 at around 96,774,230-97,035,174bp, the second phenotype (`ENSG00000124587.9`) produced a tower at chromosome 6 at around 42,889,467-43,108,015bp, the third phenotype (`ENSG00000180185.7`) produced a tower at chromosome 16 at around 1,524,250-1,929,366bp, whereas the rest of the two phenotypes (`ENSG00000168827.9` and `ENSG00000136536.9`) did not produce any valid hits. All p-value thresholds are adjusted by Bonferroni correction.

```
# all valid hits
hits <- SNPinfo[unique(results.cov[results.cov$p < alpha/N, ]$index),
    ]
cat(dim(hits)[1], "valid hits from", length(unique(hits$chromosome)),
    "chromosomes")
```

```
## 182 valid hits from 4 chromosomes
```

```
head(hits, 5)
```

```
##       chromosome position          id
```

```
## 16725          5 96774230     rs27433
## 16726          5 96775264 rs146925065
## 16727          5 96781595     rs27043
## 16728          5 96784605 rs200528525
## 16729          5 96785819    rs469783
```

```
geneinfo %>%
    slice(match(colnames(phenotypes), probe))
```

```
##                probe chromosome     start       end symbol
## 1 ENSG00000164308.12          5  96875939  96919702  ERAP2
## 2  ENSG00000124587.9          6  42963872  42979242   PEX6
## 3  ENSG00000180185.7         16   1827223   1840206  FAHD1
## 4  ENSG00000168827.9          3 158644496 158692571   GFM1
## 5  ENSG00000136536.9          2 159712456 159768582 MARCH7
```

These results are verified by checking the UCSC Genome browser with the RefSeq for human genome hg38. The hit range discovered from the GWAS of the first phenotype contains a gene named ERAP2, as indicated in the gene info dataset. The function of ERAP2 is related to N-terminal trimming of epitopes for MHC-I antigen presentation. Similarly, the hit range discovered from the GWAS of the second phenotype contains a gene named PEX6, also as indicated in the gene info dataset. The function of PEX6 is related to peroxisomal protein import. The hit range discovered from the GWAS of the third phenotype contains a gene named FAHD1, also as indicated in the gene info dataset. The function of FAHD1 is related to acetylpyruvate hydrolase activity. However, the GFM1 and MARCH7 genes, located at chromosome 3 and 2 respectively, are not discovered for the fourth and fifth phenotype.

Besides the verified hits, there are other genes discovered in the GWAS, such as PPP2R5D (negative control of cell growth and division) and CUL7 (interaction with TP53, etc.) for the second phenotype and CRAMP1 (enabling chromatin binding activity) and MAPK8IP3 (JNK signaling pathway) for the third phenotype. It might be that these genes are also related with the phenotype expression associated with the target gene, which is worth discovering in further studies.

References

1. GeneCards - The Human Gene Database
2. IGV - Integrative Genomics Viewer
3. UCSC Genome Browser - Genome Browser (Dec.2013 GRCh38/hg38)


```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22621)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggpubr_0.6.0      reshape2_1.4.4    gridExtra_2.3     ggplot2_3.4.1
## [5] MASS_7.3-58.3     dplyr_1.1.0       magrittr_2.0.3    data.table_1.14.8
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.10      pillar_1.9.0     compiler_4.2.2   formatR_1.14
##  [5] plyr_1.8.8       tools_4.2.2      digest_0.6.31    evaluate_0.20
##  [9] lifecycle_1.0.3  tibble_3.2.0     gtable_0.3.3     pkgconfig_2.0.3
## [13] rlang_1.1.0      cli_3.6.0        rstudioapi_0.14  yaml_2.3.6
## [17] xfun_0.36        fastmap_1.1.0    withr_2.5.0      stringr_1.5.0
## [21] knitr_1.42       generics_0.1.3   vctrs_0.5.2      cowplot_1.1.1
## [25] grid_4.2.2       tidyselect_1.2.0 glue_1.6.2       R6_2.5.1
## [29] rstatix_0.7.2    fansi_1.0.4      rmarkdown_2.21   carData_3.0-5
## [33] farver_2.1.1     car_3.1-2        tidyr_1.3.0      purrr_1.0.1
## [37] backports_1.4.1  scales_1.2.1     htmltools_0.5.4  abind_1.4-5
## [41] colorspace_2.1-0 ggsignif_0.6.4   labeling_0.4.2   utf8_1.2.3
## [45] stringi_1.7.12   munsell_0.5.0    broom_1.0.4
```