If you want to see equations, please see the PDF version of this document, produced by pandoc.

Introduction

In this lab, we will learn basic concepts of genome-wide association studies through realistic simulations. For simplicity we assume a linear model that describes a phenotype vector as a linear combination of multiple genotype vectors (still widely accepted in most researches). We will explore the impacts of the following simulation parameters:

- Effect size (~ regression coefficients) and number of causal variants
- Heritability (h^2; h2)
- Linkage disequilibrium structure

If you are familiar with `R`, you can start your experiments modifying the script `lab2.R` in this directory, but any other language will work as well.

Simulation of phenotype data

We will use the genotype matrix of 1000 genomes project (Lab #1).

Loading genotype dosage matrix

- We only deal with biallelic variants (two possibilities of haplotype).
- We assume an additive model, which renders encoding of each variant by 0, 1, or 2 for diploid.
- We will use precompiled chromosome 22 data `chr22.Rd`

```
load('chr22.Rd') # will have plink object
```

- Here, we use standardized genotype matrix for simplicity (mean = 0 and variance = 1 for all variants)

```
## Just give names
colnames(plink$BIM) <- c('chr', 'rs', 'missing', 'snp.loc', 'plink.a1', 'plink.a2')

## Standardize genotype matrix and replace NAs with zeros
## check dim(X): individual x SNP matrix
X <- plink$BED %>% scale() %>% rm.na.zero()
n.snps <- ncol(X)
n.ind <- nrow(X)
```

- If you want to break LD structure while preserving the same distribution of minor allele frequency:

```
X.noLD <- apply(X, 2, function(x) x[sample(n.ind)])
```

Simulation of quantitative trait

Generative model

We assume genotype matrix `X` is standardized to simplify the equations. Here we use n for number of individuals and p for total number of SNPs. We can generate phenotype for an individual i as

$$y_i = \sum_{j=1}^{p} X_{ij}\beta_j + \epsilon_i$$

where K of β_j take a non-zero value; ϵ_i represents non-genetic noise on that sample i. Additionally we assume independent noise sampled from Normal distribution with the variance parameter σ^2 shared across all individuals

$$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

and causal variant effect sizes also follow Normal distribution

$$\beta_j \sim \mathcal{N}\left(0, \tau_j^2\right).$$

You can also think that non-causal variants $\tau_j = 0$. We can fully describe our simulation with

- K : number of causal variant
- σ^2 : non-genetic variance parameter
- τ^2 : genetic effect size variance parameter

Now the question is how do we set σ^2 and τ^2 given K.

Heritability

We will introduce the concept of heritability (narrow sense) and use it to set σ^2 and τ^2. Here, we define the heritability parameter as ratio of genetic variance to total variance. We simply decompose total variance into genetic and non-genetic components. More precisely, we define

$$h_g^2 \equiv \frac{\mathbb{V}[X\beta]}{\mathbb{V}[y]} = \frac{\mathbb{V}[X\beta]}{\mathbb{V}[X\beta] + \mathbb{V}[\epsilon]}.$$

Assuming causal variants and individuals are independent, for each causal variant j, we can derive variance of effects of this causal variant β_j on n individuals as:

$$\mathbb{V}[\sum_{i=1}^{n} X_{ij}\beta_j] = \sum_{i=1}^{n} X_{ij}^2 \mathbb{V}[\beta_j] = n\tau_j^2$$

And remaining non-genetic component we have:

$$\mathbb{V}[\sum_{i=1}^{n} \epsilon_i] = n\sigma^2$$

Now, back to the problem of setting τ^2 and σ^2 values. Given a h^2 value, we want to achieve

$$h^2 = \frac{n\sum_j \tau_j^2}{n\sum_j \tau_j^2 + n\sigma^2} = \frac{K\tau^2}{K\tau^2 + \sigma^2}$$

where we simplified that causal variants share the same variance parameter (τ^2). We can easily fill in τ^2, σ^2 values by setting:

$$\tau^2 \leftarrow h^2/K, \sigma^2 \leftarrow 1 - h^2$$

Therefore we can start simulation first sampling causal effect size: $\beta_j \sim \mathcal{N}(0, h^2/K)$ and then sampling isotropic noise: $\epsilon_i \sim \mathcal{N}(0, 1-h^2)$.

Implementation

For your convenience, you can use the following function implemented in `lab2-functions.R`

```
## X        = individual x SNP genotype matrix
## n.causal = number of causal SNPs
## h2       = assumed heritability
## n.pheno  = number of phenotypes (traits)
simulate.phenotype <- function(X, n.causal, h2, n.pheno = 1) {

    Y <- matrix(NA, nrow = nrow(X), n.pheno)
    causal <- NULL

    for(j in 1:n.pheno) {
        causal.pos <- sample(n.snps, n.causal) # causal variant
     effect <- rnorm.mat(n.causal, 1) * sqrt(h2 / n.causal) # effect size
        y.hat <- X[, causal.pos, drop = FALSE] %*% effect
```

```
        y.err <- rnorm.mat(n.ind, 1) * sqrt(1 - h2)
        y <- y.hat + y.err

        Y[, j] <- y
      causal <- rbind(causal, data.frame(x.pos = causal.pos, y.col = j))
    }

    ret <- list(y = Y, causal = causal)
}
```

Genetic association

SNP-by-SNP Association test

Without environmental covariates, we assume the following linear model:

$$y \sim \mathbf{x}_j \beta_j + \epsilon$$

We want to test whether β takes zero or non-zero. More formally,

$$H_0 : \beta_j = 0 \, H_1 : \beta_j \neq 0$$

We can use test statistic (z-score) $Z = \mathbb{E}[\beta_j]/\sqrt{\mathbb{V}[\beta_j]}$ to check significant deviation from zero. But we do not know true mean and variance, so we estimate from the data.

Since we have standardized the genotype matrix (column-wise) and there is no intercept term in the linear model,

$$\hat{\beta}_j = \frac{\sum_i X_{ij} y_i}{\sum_i X_{ij}^2} = \frac{\sum_i X_{ij} y_i}{n}$$

with the variance

$$\mathbb{V}[\hat{\beta}_j] = \frac{\sum_i \mathbb{V}[\epsilon_i]}{\sum_i X_{ij}^2} = \frac{\sum_i (y_i - X_{ij}\hat{\beta}_j)^2/(n-1)}{n}$$

If you are not sure, you can quickly review (or learn) here.

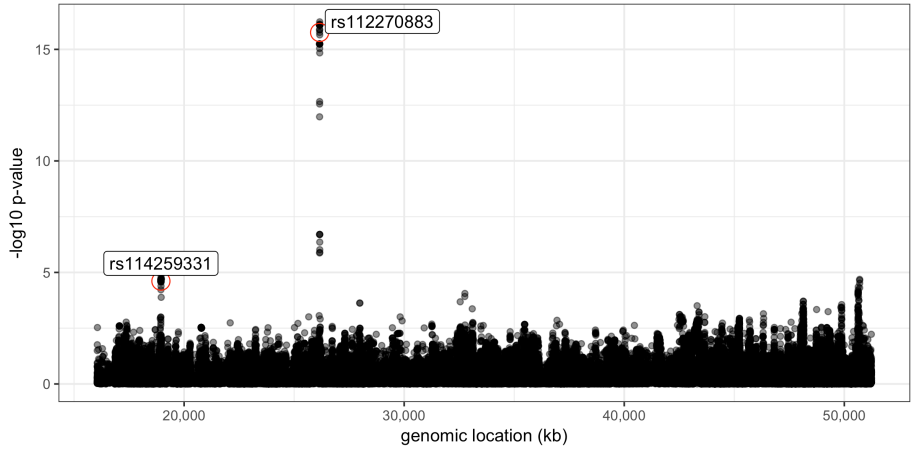
Implementation

```
######################################################################
## Calculation of SNP-by-SNP QTL statistics

## convert z-score to p-values (two-sided test)
zscore.pvalue <- function(z) {
    2 * pnorm(abs(z), lower.tail = FALSE)
}

## calculate univariate effect sizes and p-values
calc.qtl.stat <- function(xx, yy) {

    require(dplyr)
    require(tidyr)

    .xx <- scale(xx)
    .yy <- scale(yy)

    ## cross-product is much faster than covariance function
    n.obs <- crossprod(!is.na(.xx), !is.na(.yy))
   beta.mat <- crossprod(.xx %>% rm.na.zero(), .yy %>% rm.na.zero()) / n.obs

    ## residual standard deviation
    resid.se.mat <- matrix(NA, ncol(.xx), ncol(.yy))

    for(k in 1:ncol(.yy)) {

        beta.k <- beta.mat[, k]
        yy.k <- .yy[, k]
        err.k <- sweep(sweep(.xx, 2, beta.k, `*`), 1, .yy, `-`)
        se.k <- apply(err.k, 2, sd, na.rm = TRUE)

        resid.se.mat[, k] <- se.k
    }

    ## organize as consolidated table
    y.cols <- 1:ncol(yy)
    colnames(beta.mat) <- y.cols
    colnames(n.obs) <- y.cols
    colnames(resid.se.mat) <- y.cols

    beta.tab <- beta.mat %>%
        as.data.frame() %>%
            dplyr::mutate(x.col = 1:n()) %>%
              tidyr::gather(key = 'y.col', value = 'beta', y.cols)
```

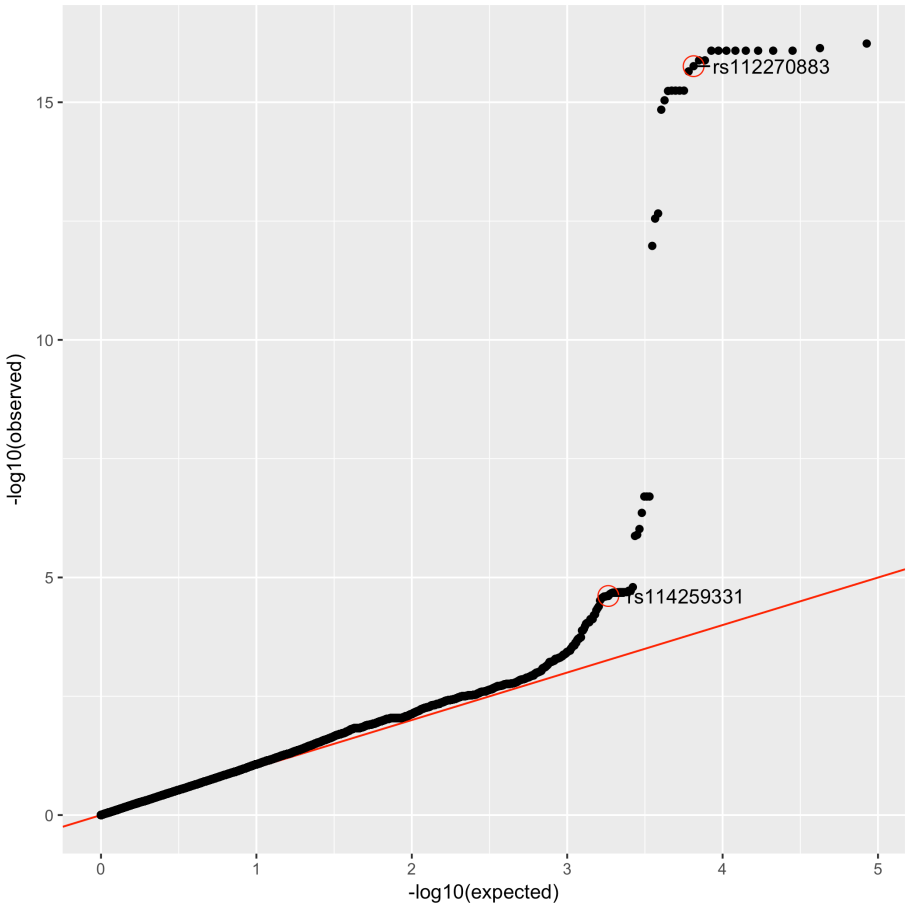
```
    resid.se.tab <- resid.se.mat %>%
        as.data.frame() %>%
            dplyr::mutate(x.col = 1:n()) %>%
           tidyr::gather(key = 'y.col', value = 'resid.se', y.cols)

    nobs.tab <- n.obs %>%
        as.data.frame() %>%
            dplyr::mutate(x.col = 1:n()) %>%
                tidyr::gather(key = 'y.col', value = 'n', y.cols)

    out.tab <- beta.tab %>%
        left_join(nobs.tab) %>%
            left_join(resid.se.tab) %>%
                dplyr::mutate(se = resid.se/sqrt(n)) %>%
                    dplyr::mutate(p.val = zscore.pvalue(beta/se))

    return(out.tab)
}
```

Visualization

Often, GWAS results are presented as so called "Manhattan plot":



Also, it is useful to visualize the results as a quantile-quantile plot, checking that most of SNPs are described by the null hypothesis (diagonal). If not GWAS, results can be either polygenic (a large of number of causal SNPs) or statistics are inflated due to unknown hidden confounding variables (which is bad).



Assignment

1. Simulate and generate GWAS statistics using 1000 genomes genotype matrix (chr22). Make plots or tables and interpret the results.
   - Generate data with different configuration of simulation parameters
   - Run GWAS within `R` or your favorite environment
   - (optional) Run GWAS using PLINK
2. Repeat the same type of experiments breaking the correlation structure between the variants (LD).
3. Simulate case/control GWAS data. Make plots or tables and interpret the results.

- Generate binary phenotype vectors with different threshold value θ. E.g., setting $y_i = 1$ if $\sum_j X_{ij}\beta_j$ is greater than θ; $y_i = 0$ otherwise.
- Run GWAS as if this were quantitative traits (a simple linear model).
- (optional) Run GWAS with PLINK treating it as case/control. What will be an appropriate hypothesis test?

Potential class project ideas

- Checking and estimating inflated statsitics is interesting research topic. See this genomic inflation paper.
- Calibration of p-values in case/control GWAS with skewed distribution is another interesting research topic. See this method paper.

If you want to work on this direction as final project, contact your TA, Yongjin Park (`ypp@csail.mit.edu`)