**QBS 177: Methods for Statistical Learning for Big Data**

Lab summary for Parallel computing lecture on week4.2

The goal of this lab is to run a genome-wide multi-population correlation (GeMPoC) method by utilizing average epidemiology and 1000 Genomes Project information in Pearson correlation analysis of quantitative smoking behaviors. We already obtained allele frequency for all sub-populations from Chromosome 22 from last lab. This lab will calculate the minor allele frequency (MAF) and obtain the correlation between MAFs and smoking data. We will also calculate the p-value of each correlation and draw a Manhattan plot.

1. Download “lab1.RData”, “data-clean.txt”, “smoking\_outcome.txt” from canvas.
2. The RData contain a matrix ‘fulldat’ with minor allele frequencies in each subpopulation of all loci’s in Chomorsome 21. You can use “data-clean.txt” and all frequency data in the chr22 folder to recreate “fulldat”
3. Transform the subpopulation data to 21 countries:

chn <- (fulldat[,5] + fulldat[,6])/2

ind <- (fulldat[,11] + fulldat[,14])/2

nga <- (fulldat[,25] + fulldat[,8])/2

usa <- (0.777\*fulldat[,4] + 0.132\*fulldat[,2] + 0.053\*(chn + ind + fulldat[,16] + fulldat[,15])/4)/(0.777+0.132+0.053)

fulldat <-fulldat[,c(3,1,5,7,9,12,11,24,15,17,19,25,21,20,22,18,13,23,10,4,16)]

fulldat[,3] <- chn

fulldat[,7] <- ind

fulldat[,12] <- nga

fulldat[,20] <- usa

1. Read in the smoking data.

yy <- read.delim(“smoking\_outcome.txt")

1. Try to write a R code to calculate the correlation between yy and the first 10000 loci. Use proc.time() before and after your code to report the computation time.

y <- yy[,2]

plong <- corlong <- NULL

proc.time()

for (i in 1:10000){

if(var(fulldat[i,])){ # remove loci with all minor allele frequncy 0

fit <- lm(y~fulldat[i,]) # run a linear regression

corlong[i] <- cor(fulldat[i,], y)

plong[i] <- summary(fit)$coef[2,4] #output slope from linear regression

}

}

proc.time()

1. Estimate the total computation time if you would apply the code to all loci’s in Chromosome 22.

Assume that proc.time shows that the program ran for 5 sec. then the total estimated time is: 5\*dim(fulldat)[1]/10000

[1] 549.582

1. Try to use vector operation to rewrite the code and speed it up. You can use the following code or write your own.

n <- dim(yy)[1] # n is the sample size

pvalue <- matrix(0, length(chr), 1) #initial a variable to store p values.

corvalue <- matrix(0, length(chr), 1) #initial a variable to store correlation.

colnames(pvalue) <- c("prevave")

colnames(corvalue) <- c("prevave")

proc.time()

temp <- scale(t(fulldat))

y1 <- y/sqrt(var(y))

asd <- y1%\*%temp/(n-1) # obtain correlation using apply

#asd <- suppressWarnings(apply( t(fulldat) , 2 , cor , y)) # alternative way

asd1 <- 1-asd^2 # mid step to calculate p value

asd2 <- sqrt(n-2)\*asd/sqrt(asd1) # this is the T statistics

pvalue[,1] <- 2\*(1-pt(abs(asd2),(n-2))) # transform T statistics to p-value

corvalue[,1] <- asd # pass correlation to corvalue.

proc.time()

1. Check if you got the same p value and correlation using the two methods.

> abs(sum(plong-pvalue[1:10000], na.rm=T))

[1] 5.057633e-14

> abs(sum(corlong-corvalue[1:10000], na.rm=T))

[1] 0

1. Draw a mahatten Plot using “manhattan” function in “qqman”. You will need specify the SNP name, chromosome number and location for the plot.