Genome-wide Association Study Analysis using GenABEL

Purpose

In this exercise, you will learn commands to carry out association tests using the R [1], and the R library GenABEL [2] in the presence of population stratification. You will be working with a pre-cleaned dataset from your PLINK exercise containing ~250 individuals each genotyped for ~6,500 SNPs and also phenotyped for both a binary and quantitative phenotype.

Methodology

You will use multidimensional scaling (MDS), principal component analysis (PCA) and genomic admixture control for detecting and adjusting for population stratification in your analysis.

Resources

For additional information on this exercise please refer to 3.3, 3.4, 5.1, 5.3, 5.4, 5.5 and 7.1 of the June 19, 2013 version GenABEL tutorial. The analysis on the binary trait partially overlaps with the PLINK exercise, with which you may compare your results and output.

This exercise is based on GenABEL **v. 1.8-0**. Complete result output could be found at the end of this exercise.

GenABEL tutorial: http://www.genabel.org/sites/default/files/pdfs/GenABEL-tutorial.pdf

Data files

You are provided with a dataset consisting of 5 files. The *.ped, *.map and *.phen files are slightly different files from the ones you have worked with in your PLINK GWAS exercise. *Please keep in mind that the data has been 'cleaned', which is a necessary requirement for all analysis below.* There are two additional phenotype files (*.praw) formatted for GenABEL input.

We will be working with GenABEL, which use the *.tped and *.tfam format files rather than the *.ped and the *.map files you used for the PLINK exercise. The following PLINK commands will allow you to re-format your input; you can copy these commands from the code files provided.

```
plink --file GWAS_clean4 --pheno pheno.phen --pheno-name Aff --recode transpose --out
gwa_gabel
```

```
plink --file GWAS_clean4 --pheno pheno.phen --pheno-name systolic --recode transpose
--out gwa_gabel_qtl
```

The two *.praw files have been formatted from the *.tfam files you just generated. Such re-formatting could be done with simple shell scripts. Please take a look at the files to check that you understand how the information is coded. Please read the PLINK documentation for *.tfam file specification if you are not familiar with it.

Q1: Compare gwa_gabel.praw ***with*** gwa_gabel.tfam***. How is information sex and disease coded in both files?***

Answer: _____

Now we are ready to use GenABEL for GWAS. The individual tests are generally the same as were previously introduced in the R exercise, but with GenABEL the whole genome can be analyzed quickly while easily correcting for population structure and other confounders.

1. A Preliminary GWAS Analysis

For this preliminary study we ignore the presence of population structure. Please carry-out these commands in R, and copy-paste the following to load GenABEL and our data files:

R

```
library(GenABEL)# load files to GenABEL

convert.snp.tped(tped = "gwa_gabel_qtl.tped", tfam = "gwa_gabel_qtl.tfam", out =
"gwa_gabel_qtl.raw", strand = "u")

g.dat <- load.gwaa.data(phen = "gwa_gabel_qtl.praw", gen = "gwa_gabel_qtl.raw", force
= T)
```

Now you have created a GenABEL object 'g.dat' that contains all information in your original data set. To view members in that data object:

R

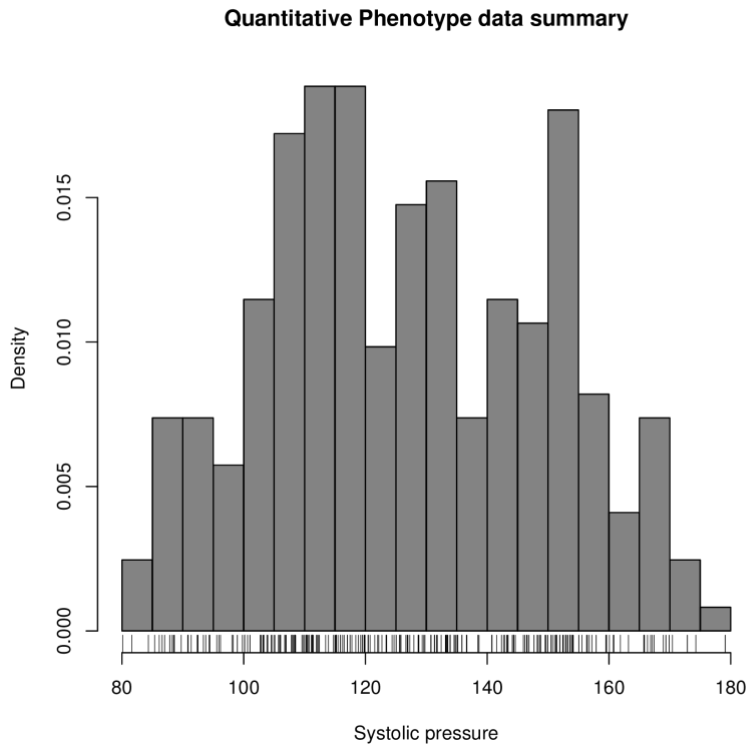
```
slotNames(g.dat)
slotNames(g.dat@gtdata)
colnames(g.dat@phdata)
```

It is always necessary to become familiar with your data before engaging into analysis. The following code provides you a starting point. The histogram generated provides visualization of the distribution and relative frequency of the quantitative values, accompanied by a rug plot showing distinct values on the X-axis.

R

```
# sample size
sample.size <- g.dat@gtdata@nids

# number of SNPs
snps.total <- g.dat@gtdata@nsnps

print(c(sample.size, snps.total))

# Trait
summary(g.dat@phdata$disease)

hist(g.dat@phdata$disease, main="Quantitative Phenotype data summary", xlab =
"Systolic pressure", freq = F,breaks=20, col="gray")
rug(g.dat@phdata$disease)
```

Figure 1: Quantitative Phenotype data summary



Tip

You may also want to look at other information such as mean and variance of trait, percentage of males/females, missing data, etc. For more detailed data information summary, please read documentation for `descriptives.trait()` and `descriptives.marker()` functions.

Q2: What is the sample size, and how many SNPs have been genotyped in this study? What type of trait will we analyze (binary or continuous)?

Answer: _____

Let us begin our analysis to try to detect an association with the quantitative trait using the `scan.glm()` function, which scans the whole genome using the `glm()` function. Note that this test will be very slow because in GLMs parameters are estimated numerically via MLE method. However, we introduce this function because it allows you to add interaction terms (gene-gene, gene-environment or any interactions) the same manner as you used the `glm()` function in R.

R

```
test.snp <- scan.glm('disease ~ CRSNP', family = gaussian(), data = g.dat)
names(test.snp)
alpha <- 5e-8
test.snp$snpnames[test.snp$P1df < alpha]
test.snp$P1df[test.snp$P1df < alpha]
```

From the test, you find that for the entire "genome", only 'rs4571722' is associated with the quantitative trait with a p-value of 2E-17.

We can also use a score test, the `qtscore()` function, which is efficient. For quantitative traits, it is a score test for the linear regression model.

Note

A "warning message" pops up in R, when the estimated genomic inflation factor (GIF) is lower than 1. You can use `quiet = T` argument in `qscore()` to turn it off.

R

```
test.qt <- qtscore(disease, data = g.dat, trait = "gaussian")

slotNames(test.qt)
names(test.qt@results)
head(results(test.qt))
test.qt@lambda
descriptives.scan(test.qt)
row.names(results(test.qt))[results(test.qt)$P1df < alpha]
results(test.qt)$P1df[results(test.qt)$P1df < alpha]
results(test.qt)$Pc1df[results(test.qt)$Pc1df < alpha]
```

For this test SNP 'rs4571722' has p-value 9E-14.

Task 1: Please find `P1df` ***(p-value for the 1 df test),*** `chi2.1df` ***(1 df test),*** `Pc1df` ***(p-value for the 1 df test corrected for inflation factor) and*** `lambda` ***(genomic inflation factor) on your screen. These R objects contain information for the 1df test (additive model). Information for the 2df score test (genotypic model, i.e. effects of AB and BB estimated against reference, AA) are also provided but for the text below we will only be discussing the 1df test. You could extract information from an R object with the '$' symbol, as demonstrated above.***

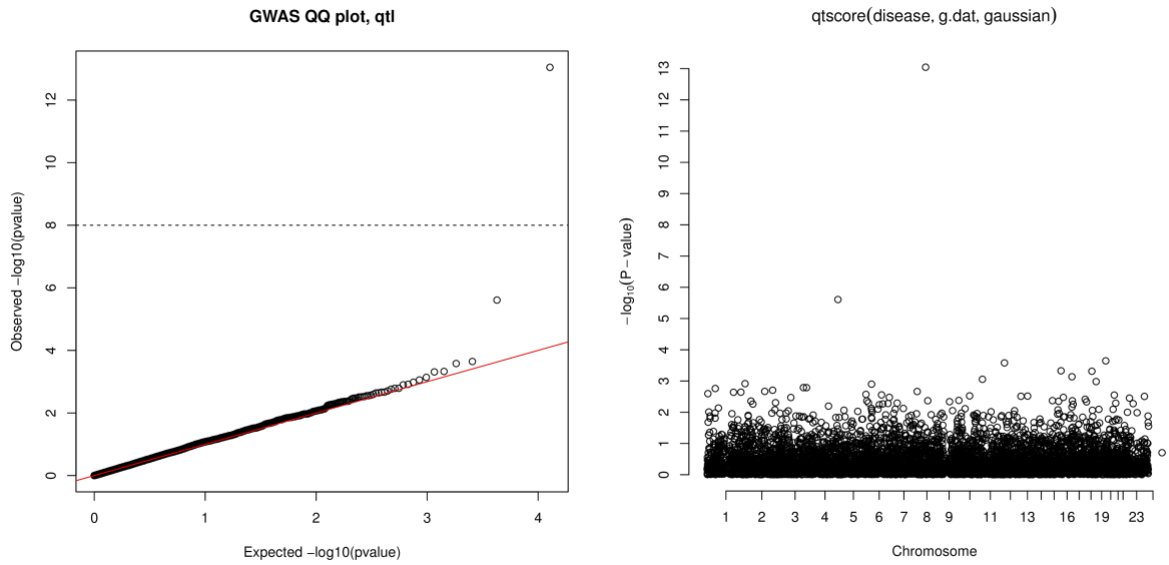
In publications, we often display our result (the 1 df p-value in our case) with a QQ plot and a Manhattan plot. The QQ plot is very useful for diagnostic purposes: it plots the $-\log_{10}$ of the observed p-values vs. the $-\log_{10}$ of the expected p-values (under null). If all points fall on the diagonal line, then none of the p-values are smaller than would be expected when there is no association. If we have many p-values which are smaller than expected ($-\log_{10}$ values on top the line) this is probably not because we have many true associations, but because there is a problem with the data, for example population substructure/admixture. The Manhattan plot is very nice to visualize the findings in chromosomal order, it also helps to see if there are multiple associations within a small genomic region.

We will now make a QQ and Manhattan plot of the p-values which have not been adjusted (`P1df`).

R

```
# QQ plot
obs <- sort(results(test.qt)$P1df)
ept <- ppoints(obs)

plot(-log10(ept), -log10(obs), main = "GWAS QQ plot, qtl", xlab="Expected -
log10(pvalue)", ylab="Observed -log10(pvalue)")

abline(0, 1, col = "red")
abline(h = 8, lty = 2)

# Manhattan plot
plot(test.qt, col = "black")
```

Figure 2: The QQ and Manhattan plots



Notice that the expected p-values under the null follow a uniform distribution and they are plotted vs. the observed p-values, and both are converted into $-\log_{10} P$ in the QQ plot. Please examine the plots to make sure you understand them.

Adding potential confounders to the analysis

You can add predictors to `scan.glm()` or `qtscore()` functions. Covariates could be sex, age, and importantly, principal components that corrects for population structure. For example:

R

```
test.qt.sex <- qtscore(disease ~ sex, data = g.dat, trait = "gaussian")
row.names(results(test.qt.sex))[results(test.qt)$P1df < alpha]
```

`qtscore` in effect runs the analysis on residuals of the linear model specified. Therefore you can evaluate the significance of sex by:

R

```
summary(lm(disease ~ sex, data = g.dat))
```

Working with binary traits

Change the argument *gaussian* into *binomial* in `scan.glm()` and `qtscore()` functions when you analyze case-control data. For binary trait, the 1 df test is an additive test and is equivalent to the Armitage trend test when no covariates are included in the analysis.

Q3: Please use the R codes below (or at the end of Code.R.txt ***document) to work with binary trait. For this part you could check the result with your PLINK exercise. Which SNPs are significantly associated with the trait?***

R

```
# load files to GenABEL
convert.snp.tped(tped = "gwa_gabel.tped", tfam = "gwa_gabel.tfam", out =
"gwa_gabel.raw", strand = "u")
b.dat <- load.gwaa.data(phen = "gwa_gabel.praw", gen = "gwa_gabel.raw", force = T)

slotNames(b.dat)
slotNames(b.dat@gtdata)
colnames(b.dat@phdata)

# sample size
b.dat@gtdata@nids

# number of cases and controls
case.size <- length(which(b.dat@phdata$disease == 1))
control.size <- length(which(b.dat@phdata$disease == 0))
case.size
control.size

# number of SNPs
snpsb.total <- b.dat@gtdata@nsnps

# GLM test
testb.snp <- scan.glm('disease ~ CRSNP', family = binomial(), data = b.dat)
names(testb.snp)
alpha <- 5e-8

testb.snp$snpnames[testb.snp$P1df < alpha]
testb.snp$P1df[testb.snp$P1df < alpha]

# Score test
testb.qt <- qtscore(disease, data = b.dat, trait = "binomial")
slotNames(testb.qt)
descriptives.scan(testb.qt)

row.names(results(testb.qt))[results(testb.qt)$P1df < alpha]
results(testb.qt)$P1df[results(testb.qt)$P1df < alpha]
results(testb.qt)$Pc1df[results(testb.qt)$Pc1df < alpha]
```

2. MDS Analysis for Global Ancestry Inference

Now let us perform Multidimensional Scaling analysis with GenABEL to detect potential population structure. Note that MDS is closely related to PC analysis, and that for PC analysis GenABEL uses the EIGENSTRAT algorithm. Firstly, we compute a genome-wide IBS matrix:

R

```
gkin <- ibs(g.dat, weight = "freq")
gkin[1:10,1:10]
```

Look at the Identity-by-state (IBS) matrix for the first 10 people. This matrix gives average IBS (kinship) values between a pair below the diagonal, and number of genotype successfully typed for both members of the pair above the diagonal. The diagonal is *0.5 + homozygosity*.

Now we convert it into a distance matrix and pass it into our MDS analysis function, the `cmdscale()`:

R

```
cps.full <- cmdscale(as.dist(.5 - gkin), eig = T, k = 10)
names(cps.full)
cps <- cps.full$points
```

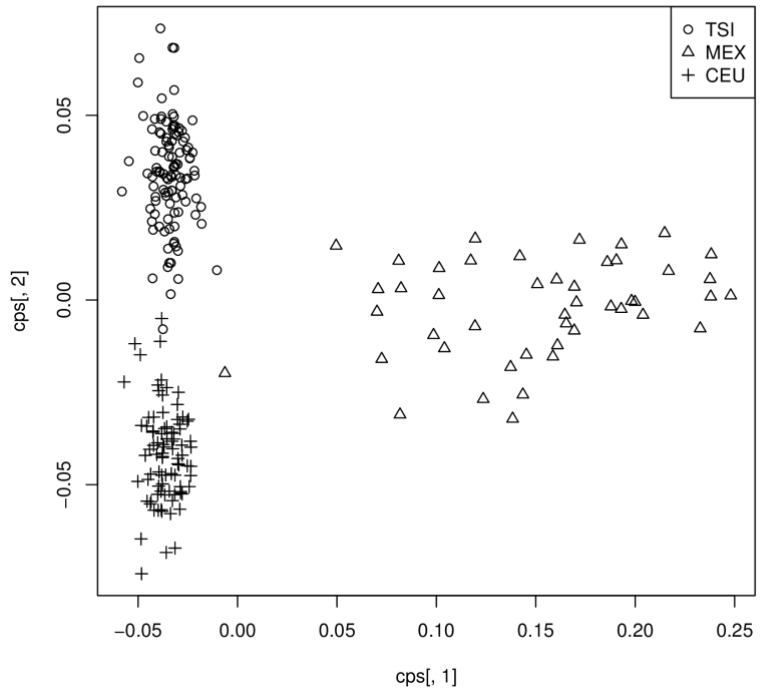
See `?as.dist` if you are interested in the distance.

The distance matrix is a measure of dissimilarities between individuals; *k = 10* means that the output will contain the first 10 components. Let us plot out the first and the second components:

R

```
plot(cps[,1], cps[,2], pch = g.dat@phdata$popn)
legend("topright", c("TSI","MEX", "CEU"), pch = c(1,2,3))
```

Figure 3: The first two components plot



Please read the legend on your plot. If you recall from your PLINK exercise, the dataset is comprised of individuals from three groups from Hapmap CEU (Mormons from Utah, originally from Northern Europe, TSI (Tuscans - Northern Italy) and MEX (Mexican Americans). In practice, you should never knowingly analyze data from different

population ethnic groups together, even if you adjust for population structure. Instead the groups can be analyzed separately and the results combined using meta-analysis.

Q4: Do you see population stratification in our data set? Which component allows you to distinguish between CEU and MEX? What about TSI and CEU?

Answer: _____

3. Association Tests with Correction for Population Structure

We will introduce three methods below, for illustration purpose. They do not uniformly work very well with our dataset but we want you to learn the commands and concepts. You may look at the GenABEL tutorial Chapter 3 where there is a discussion on which method to use in your analysis.

Incorporating PCs as predictors

As we have computed components scores with MDS analysis, we could correct for population structure by adding the components as predictors. For example:

R

```
colnames(cps)<-c('C1','C2','C3','C4','C5','C6','C7','C8','C9','C10')
gpc.dat <- g.dat
gpc.dat@phdata<-cbind(g.dat@phdata, cps)
test.pc.a <- scan.glm('disease ~ CRSNP + C1 + C2 + C3 + C4 + C5', family=gaussian(),
data = gpc.dat)
test.pc.a$snpnames[test.pc.a$P1df < alpha]
test.pc.a$P1df[test.pc.a$P1df < alpha]
test.pc.b <- qtscore(disease ~  C1 + C2 + C3 + C4 + C5, data = gpc.dat, trait =
"gaussian")
test.pc.b@lambda
```

Please check the result on your own. You can produce new QQ and Manhattan plots to compare p-value with previous analysis, and compare `test.qt@lambda` and `test.pc.b@lambda` to see if genomic inflation factor λ has decreased.

Note

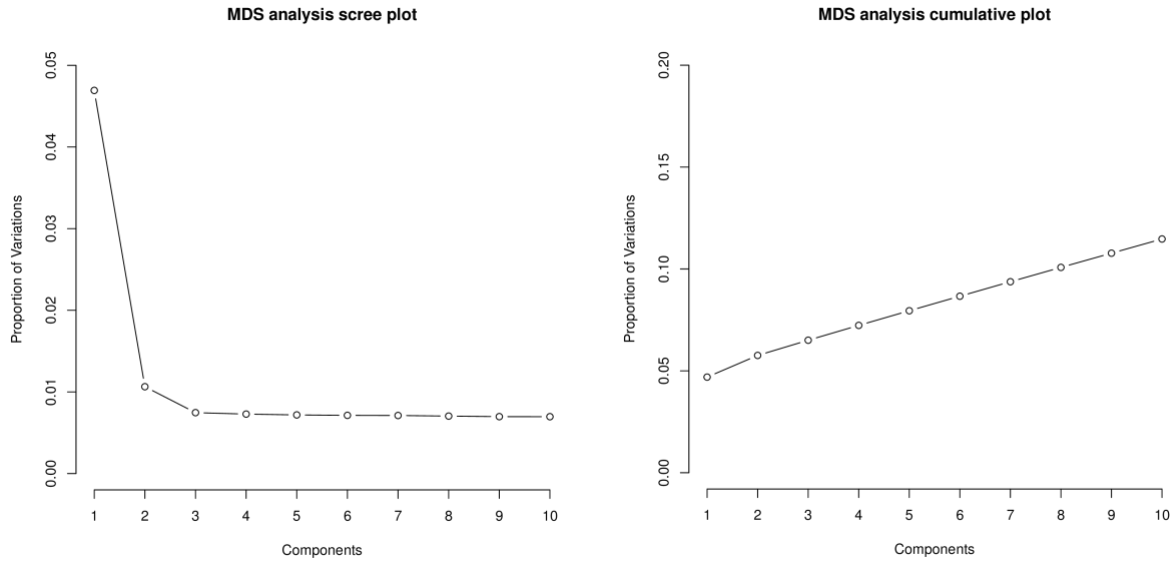
For PCA and MDS analysis, markers should be independent, which is the case for the data for this tutorial. For analysis of real-world GWAS data we need to select a subset of markers that are not in LD with each other.

If you include more components than is necessary, you can introduce noise to your analysis. Therefore, you should examine the eigenvalues. Knowing that the sum of the first *k* eigenvalues divided by the sum of all the eigenvalues represents the *proportion of total variation* explained by the first k components, we can visually evaluate the contribution of each component via the following plots:

R

```
# scree plot
plot(cps.full$eig[1:10]/sum(cps.full$eig), axes = F, type = "b", xlab = "Components",
ylim = c(0,0.05), ylab = "Proportion of Variations", main = "MDS analysis scree
plot")
axis(1, 1:10)
axis(2)

# cumulative plot
plot(cumsum(cps.full$eig[1:10])/sum(cps.full$eig), axes = F, type = "b", ylim =
c(0,0.2), xlab = "Components", ylab = "Proportion of Variations", main = "MDS
analysis cumulative plot")

axis(1, 1:10)
axis(2)
```

Figure 4: Scree plot and cumulative plot for the first 10 eigenvalues of dissimilarity matrix



From the plots, we observe that more than 10% of the information can be explained by the first 10 components. The first two components could explain more than 5% of the total variation. We could start building the association model projecting the total variation onto two dimensions by including the first two components, and evaluate the change of λ as has been demonstrated in the PLINK exercise. Although the GenABEL tutorial states that $\lambda > 1.01$ is considered as small, $\lambda < 1.11$ is acceptable and $\lambda > 1.11$ is problematic, in most cases a λ of 1.1 is considered too large particularly when you are trying to detect small effect sizes. There is, however, no general guideline on λ since it describes how much a specific test is inflated – it depends on the test statistic.

Tip

Alternatively, you may look at EIGENSTRAT package, which applies other methods for deciding how many components to include in analysis.

Task 2: You may change the number of components you include in your analysis and evaluate how λ changes.

Answer: _____

Genomic Control

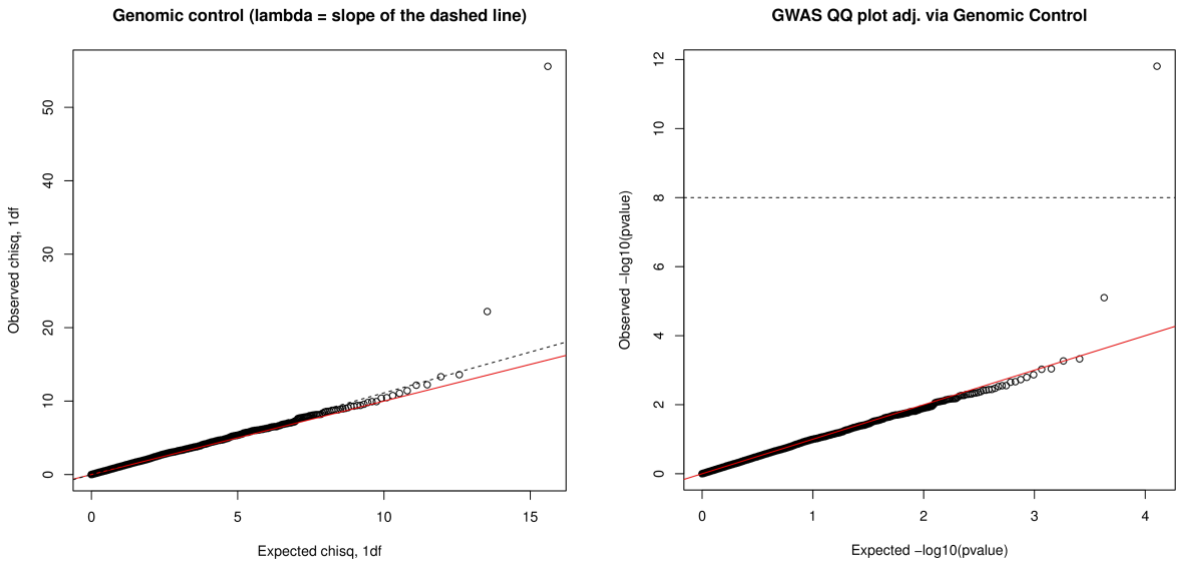
An alternative method for correcting population structure is to use Genomic Control. If you make QQ plot of the chi2.1df (cf. **Task 1**) vs. χ^2_1 under the null you will see many points off the diagonal, indicating the test statistic is inflated, as shown in the plot below (slope of the dashed line is λ). For genomic control method, the observed test statistic will be divided by λ to correct for inflation of test statistic with corrected p-value Pc1df. Note that although in most literature λ is defined as the median of observed χ^2 test statistic divided by 0.456 (see Devlin and Roeder 1999[3]; Devlin et al. 2001[4]), GenABEL uses a different definition, the regression coefficient (slope) of observed χ^2 w.r.t. expected χ^2, which is usually a bit more conservative. GenABEL also uses only the lower 90% of the distribution to calculate λ.

Genomic control method many not adequately correct for population substructure, and it is not recommended to use for admixed populations. Although this method is rarely used to correct for population substructure, information on λ is used to determine if type I error is controlled and also to aid in determining the number of PC or MDC components to include in the analysis.

R

```
# corrected p-value
row.names(results(test.qt))[results(test.qt)$Pc1df < alpha]
results(test.qt)$Pc1df[results(test.qt)$Pc1df < alpha]


# Uncorrected GIF calculated by GenABEL
test.qt@lambda


# Check for inflation of statistic
obs <- sort(results(test.qt)$chi2.1df)
ept <- sort(qchisq(ppoints(obs), df = 1))


plot(ept, obs, main = "Genomic control (lambda = slope of the dashed line)", xlab="Expected chisq, 1df",
ylab="Observed chisq, 1df")
abline(0, 1, col = "red")
abline(0, test.qt@lambda[1], lty = 2)


# Definition of GIF
# Conventional definition
median(results(test.qt)$chi2.1df)/0.456



# QQ plot, corrected p-value
obs <- sort(results(test.qt)$Pc1df)
ept <- ppoints(obs)

plot(-log10(ept), -log10(obs), main = "GWAS QQ plot adj. via Genomic Control", xlab="Expected -
log10(pvalue)", ylab="Observed -log10(pvalue)")
abline(0, 1, col = "red")
abline(h = 8, lty = 2)
```

Figure 5: The QQ plots. Left: Genomic inflation Right: p-value adjusted via genomic control



Q5: Compare the corrected QQ plot for p-value with the uncorrected plot. What is the value for corrected lambda? Does it suggest that genomic control works better than PCA?

Answer: _____

Correction with PCA

We recommend doing correction for population structure using PCA method, as described in Price et al. 2006[5]. GenABEL incorporates EIGENSTRAT method to test for association while correcting for population structure. This is implemented in `egscore()` function. The following code prepares input for the function and then carries out association testing correcting for the first 2 components:

R

```
adj.gkin = gkin
diag(adj.gkin) = hom(g.dat)$Var

test.eg <- egscore(disease, data = g.dat, kin = adj.gkin, naxes = 2)
descriptives.scan(test.eg)

snp.eg <- row.names(results(test.eg))[results(test.eg)$P1df < alpha]
pvalue.eg <- results(test.eg)$P1df[results(test.eg)$P1df < alpha]
lambda.eg <- test.eg@lambda
snp.eg
pvalue.eg
lambda.eg
```

Q6: Is the test still significant? What is the p-value for 'rs4571722'? What about GIF?

Answer: _____

Note

You may notice that the `egscore()` function gives a different λ value than using `qtscore` adjusted for the same number of components, with a smaller λ value for the `qtscore` where the components from MDS have been used in the analysis. The two types of analysis are different: the egscore test (1 df) adjusts both genotype and phenotype onto PCs and computes their correlation after adjustment, while the qtscore test adjusted only for the genotype.

If you want to determine how many PCs to include based on GIF, you may use the following command, which calculates λ for the first 1 to 10 PCs:

R

```
for (k in 1:10){
test.tmp <- egscore(disease, data = g.dat, kin = adj.gkin, naxes = k)
print(test.tmp@lambda$estimate)
}
```

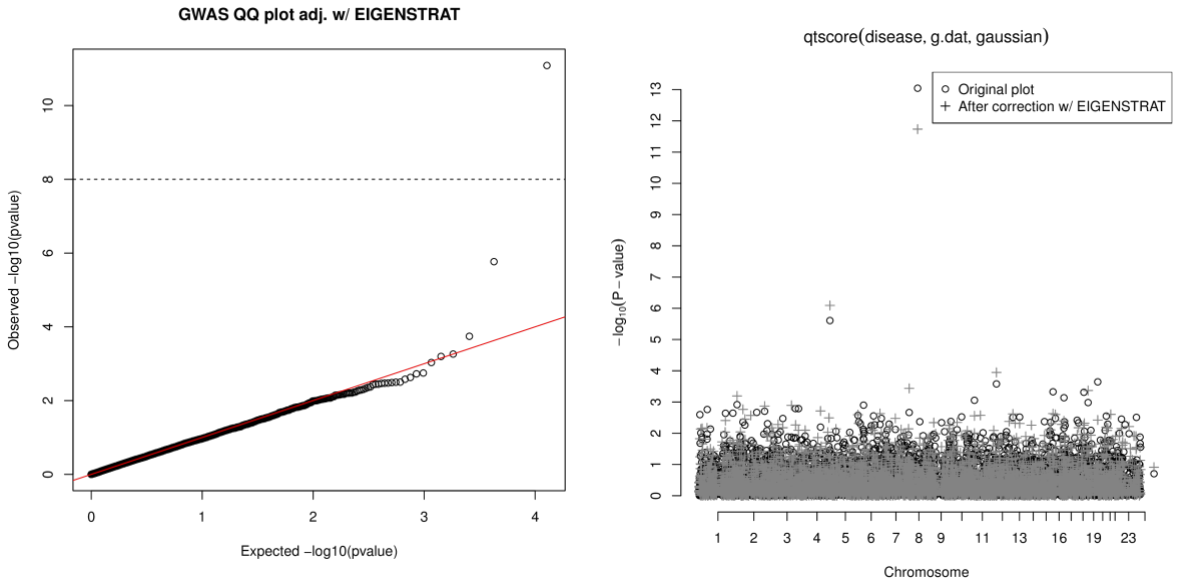
You may update your analysis based on number of PCs that you want to include.

Finally, you can make a new QQ plot after adjusting for population substructure. You can also compare your Manhattan plot adjusting for population substructure with the one that did not adjust for population structure by plotting them onto one graph and using different symbols for the two types of analyses:

R

```
# QQ plot
obs <- sort(results(test.eg)$Pc1df)
ept <- ppoints(obs)

plot(-log10(ept), -log10(obs), main = "GWAS QQ plot adj. w/ EIGENSTRAT",
xlab="Expected -log10(pvalue)", ylab="Observed -log10(pvalue)")

abline(0, 1, col = "red")
abline(h = 8, lty = 2)

# Manhattan plot comparison
plot(test.qt, col = "black")
add.plot(test.eg, col = "gray", pch = 3)
legend("topright", c("Original plot","After correction w/ EIGENSTRAT"), pch =
c(1,3))
```

Figure 6: The QQ plot and Manhattan plot (adjusted for population admixture using PCA)



Please note the change of p-value on your new Manhattan plot: it is now less significant for 'rs4571722'.

Final task: ***You can carry out analysis on binary data as in 1.3 by yourself, if time permits. As you have done Q3 at the end of 1.3, data for binary trait is now loaded as the 'b.dat' object. You need to change the argument 'gaussian' into 'binomial' in 'scan.glm' and 'qtscore' function, and replace 'g.dat' with 'b.dat' for all commands in 1.3 and run the new commands to carry out analysis.***

Answers to Problems

A1: In `gwa_gabel.tfam` file sex is coded as '1' and '2', so is disease status. But `GenABEL` requires both to be coded as '0' and '1', which you may find in `gwa gabel.praw` file. You need to keep in mind these difference when you use a script language to convert *.`tfam` file to `GenABEL` phenotype input file.

A2: We have 244 samples and 6,388 SNPs. We will be working with quantitative trait (See Figure 1).

A3: rs4571722 and rs10008252.

A4: Component 1 distinguishes between CEU and MEX (as well as TSI and MEX). Component 2 distinguishes between TSI and CEU.

A5: Genomic control adjusts for inflation via dividing the observed χ^2 statistic by λ and compute corrected p-values associated with the adjusted χ^2; hence the corrected $\lambda = 1$. If we only consider λ when we evaluate performance of techniques for adjusting population structure, genomic control, by definition will naturally outperform all other methods.

A6: rs4571722 is found significant (p-value 1.84E-12). The inflation for the test is $\lambda = 1.06$.

Appendix: Analysis Commands with Output

Analysis Log

```
> #----Load files----#
> library(GenABEL)
Loading required package: MASS
Loading required package: GenABEL.data

> convert.snp.tped(tped = "gwa_gabel_qtl.tped", tfam = "gwa_gabel_qtl.tfam", out = "gwa_gabel_qtl.raw", strand =
"u") Reading individual ids from file *gwa_gabel_qtl.tfam* ...
... done. Read 244 individual ids from file *gwa_gabel_qtl.tfam*
Reading genotypes from file *gwa_gabel_qtl.tped* ...
...done. Read 6388 SNPs from file *gwa_gabel_qtl.tped*
Writing to file *gwa_gabel_qtl.raw* ...
... done.

> g.dat <- load.gwaa.data(phen = "gwa_gabel_qtl.praw", gen = "gwa_gabel_qtl.raw", force = T)
ids loaded...
marker names loaded...
chromosome data
loaded... map data
loaded...
allele coding data
loaded... strand data
loaded... genotype data
loaded... snp.data object
created...
assignment of gwaa.data object FORCED; X-errors were not checked!

> #---Data Summary----#
> slotNames(g.dat)
[1] "phdata" "gtdata"
> slotNames(g.dat@gtdata)
 [1] "nbytes"     "nids"       "nsnps"      "idnames"    "snpnames"
 [6] "chromosome" "map"        "coding"     "strand"     "male"
[11] "gtps"
> colnames(g.dat@phdata)
[1] "id"      "sex"     "disease" "popn"
> # sample size
> sample.size <- g.dat@gtdata@nids
> # number of SNPs
> snps.total <- g.dat@gtdata@nsnps
> print(c(sample.size, snps.total))
[1]  244 6388
> # Trait
> summary(g.dat@phdata$disease)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  80.12  109.70  125.00  126.60  146.30  179.10
> hist(g.dat@phdata$disease, main="Quantitative Phenotype data summary", xlab = "Systolic pressure measure", freq = F,breaks=20, col
="gray")
> rug(g.dat@phdata$disease)

> #----Tests for association----#
> # GLM test
> test.snp <- scan.glm(*disease ~ CRSNP*, family = gaussian(), data = g.dat)
99.41%
> names(test.snp)
 [1] "P1df"       "P2df"       "medChi1df"  "medChi2df"  "snpnames"
[6] "idnames"    "formula"    "family"     "map"        "chromosome"
[11] "effB"       "effAB"      "effBB"      "Pc1df"
> alpha <- 5e-8
> test.snp$snpnames[test.snp$P1df < alpha]
[1] "rs4571722"
> test.snp$P1df[test.snp$P1df < alpha]
[1] 2.712335e-17

> # Score test
> test.qt <- qtscore(disease, data = g.dat, trait = "gaussian")
> slotNames(test.qt)
[1] "results"    "dimresults" "annotation" "lambda"     "idnames"
[6] "call"       "family"
> names(test.qt@results)
[1] "N"        "effB"     "se_effB"  "chi2.1df" "P1df"     "effAB"    "effBB"
[8] "chi2.2df" "P2df"

> test.qt@lambda
$estimate
[1] 1.112075
$se
[1] 0.003559959

> descriptives.scan(test.qt)
Summary for top 10 results, sorted by P1df
```

```
           Chromosome   Position Strand A1 A2   N        effB   se_effB chi2.1df
rs4571722           8   60326734      u  C  T 242 -14.943693  2.004587 55.57329
rs10008252          4  179853616      u  A  G 244  -9.823880  2.085360 22.19239
rs7254875          19   61010979      u  C  T 244  12.171225  3.300709 13.59734
rs2881297          12   27298922      u  C  T 244  -7.757537  2.126015 13.31419
rs11865295         16    7471881      u  G  A 243   6.972167  1.994012 12.22587
rs10520770         18   44856962      u  T  C 244   7.132231  2.045209 12.16117
rs10459871         16   79650568      u  C  G 244  10.597100  3.136921 11.41213
rs10500821         11   15985670      u  C  G 244 -15.127917  4.548706 11.06070
rs9950554          18   72793301      u  T  C 243  22.001829  6.710260 10.75075
rs10495538          2    6707489      u  C  T 244  -9.197564  2.843220 10.46465
…

> rownames(results(test.qt))[results(test.qt)$P1df < alpha] [1]
"rs4571722"
> results(test.qt)$P1df[results(test.qt)$P1df < alpha] [1]
9.003854e-14
> results(test.qt)$Pc1df[results(test.qt)$Pc1df < alpha] [1]
1.559056e-12

> #---- Plots----#
> # QQ plot
> obs <- sort(results(test.qt)$P1df)
> ept <- c(1:length(obs)) / (length(obs) + 1)
> plot(-log10(ept), -log10(obs), main = "GWAS QQ plot, qtl", xlab="Expected -log10(pvalue)", ylab="Observed -log10(pvalue)")
> abline(0, 1, col = "red")
> abline(h = 8, lty = 2)
> # Manhattan plot
> plot(test.qt, col = "black")

> #----Adding confounders----#
> test.qt.sex <- qtscore(disease ~ sex, data = g.dat, trait = "gaussian")
> rownames(results(test.qt.sex))[results(test.qt)$P1df < alpha] [1]
"rs4571722"
> summary(lm(disease ~ sex, data = g.dat)) Call:
lm(formula = disease ~ sex, data = g.dat)
Residuals:
    Min      1Q  Median      3Q     Max
-46.355 -17.045  -1.707  18.993  51.125

Coefficients:
            Estimate Std. Error t value Pr(>|t|) (Intercept)
127.975        2.048   62.478    <2e-16 *** sex           -
2.802        2.909   -0.963     0.336
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 22.72 on 242 degrees of freedom
Multiple R-squared: 0.00382,        Adjusted R-squared: -0.000296
F-statistic: 0.9281 on 1 and 242 DF,  p-value: 0.3363


> #----MDS analysis----#
> gkin <- ibs(g.dat, weight = "freq")
> gkin[1:10,1:10]
              NA06989       NA11891       NA11843       NA06984       NA12275
NA06989   0.5045911748  6.281000e+03  6.359000e+03  6.328000e+03  6.342000e+03
NA11891  -0.0034037618  5.035373e-01  6.294000e+03  6.263000e+03  6.280000e+03
NA11843   0.0004385764  4.235451e-03  5.225284e-01  6.341000e+03  6.355000e+03
NA06984  -0.0066330304  7.436545e-03  6.370382e-03  5.200090e-01  6.323000e+03
NA12275   0.0033352693  1.052177e-02  3.398087e-03 -1.217706e-02  5.005654e-01
NA06986   0.0036653267  1.307613e-02  5.158255e-05  3.553836e-04  7.609489e-03
NA12272  -0.0003230312 -6.838754e-03  3.013671e-03 -7.818281e-04  9.429297e-03
NA07051   0.0064402150  1.662204e-02 -5.109244e-03  7.993169e-03  3.715568e-03
NA12400   0.0086247762  2.812382e-03  9.097334e-03  1.016512e-02  9.235554e-04
NA12777  -0.0062509088  7.882974e-03  6.707316e-03 -1.855177e-04  2.917005e-03
              NA06986       NA12272       NA07051       NA12400       NA12777
NA06989   6.357000e+03  6.363000e+03  6.359000e+03  6.355000e+03  6357.0000000
NA11891   6.291000e+03  6.297000e+03  6.294000e+03  6.290000e+03  6292.0000000
NA11843   6.369000e+03  6.375000e+03  6.373000e+03  6.367000e+03  6370.0000000
NA06984   6.337000e+03  6.343000e+03  6.340000e+03  6.335000e+03  6338.0000000
NA12275   6.352000e+03  6.358000e+03  6.355000e+03  6.350000e+03  6353.0000000
NA06986   5.425816e-01  6.372000e+03  6.369000e+03  6.364000e+03  6368.0000000 …
>
> cps.full <- cmdscale(as.dist(.5 - gkin), eig = T, k = 10)
> names(cps.full)
[1] "points" "eig"    "x"      "ac"     "GOF"
> cps <- cps.full$points
> plot(cps[,1], cps[,2], pch = g.dat@phdata$popn)
> legend(-0.16, 0.06, c("TSI","MEX", "CEU"), pch = c(1,2,3))
>
> #----GLM and score test adjusted by including components----#
> # Incorporating PCs as predictors
> colnames(cps)<-c(*C1*,*C2*,*C3*,*C4*,*C5*,*C6*,*C7*,*C8*,*C9*,*C10*)
```

```
> gpc.dat <- g.dat
> gpc.dat@phdata<-cbind(g.dat@phdata, cps)
> test.pc.a <- scan.glm(*disease ~ CRSNP + C1 + C2 + C3 + C4 + C5*, family=gaussian(), data = gpc.dat)
99.41%
> test.pc.a$snpnames[test.pc.a$P1df < alpha]
[1] "rs4571722"
> test.pc.a$P1df[test.pc.a$P1df < alpha]
[1] 5.544685e-14
>
> test.pc.b <- qtscore(disease ~ C1 + C2 + C3 + C4 + C5, data = gpc.dat, trait = "gaussian")
> test.pc.b@lambda
$estimate
[1] 1.008033
$se
[1] 0.002601868

> # scree plot of eigenvalue
> plot(cps.full$eig[1:10]/sum(cps.full$eig), axes = F, type = "b", xlab = "Components", ylim = c(0,0.05), ylab = "Proportion of Var
iations", main = "MDS analysis scree plot")
> axis(1, 1:10)
> axis(2)
>
> # cumulative plot
> plot(cumsum(cps.full$eig[1:10])/sum(cps.full$eig), axes = F, type = "b", ylim = c(0,0.2), xlab = "Components", ylab = "Proportion
of Variations", main = "MDS analysis cumulative plot")
> axis(1, 1:10)
> axis(2)
>
> #----Genomic control adjustment----#
> # Uncorrected GIF
> test.qt@lambda
$estimate
[1] 1.112075
$se
[1] 0.003559959

> # Corrected p-value
> row.names(results(test.qt))[results(test.qt)$Pc1df < alpha]
[1] "rs4571722"
> results(test.qt)$Pc1df[results(test.qt)$Pc1df < alpha]
[1] 1.559056e-12

> #----Check for inflation of statistic----#
> obs <- sort(results(test.qt)$chi2.1df)
> ept <- sort(qchisq(1:length(obs) / (length(obs) + 1), df = 1))
> plot(ept, obs, main = "Genomic control (slope is the inflation factor)", xlab="Expected chisq, 1df", ylab="Observed chisq, 1df")
> abline(0, 1, col = "red")
> abline(0, test.qt@lambda[1], lty = 2)
> # Definition of GIF
> # Conventional definition
> median(results(test.qt)$chi2.1df)/0.456
[1] 1.082631
> # GenABEL definition
> lm(obs~ept)$coef[2]
     ept
1.121627

> #----Corrected chisq pvalue QQPLOT----#
> obs <- sort(results(test.qt)$Pc1df)
> ept <- c(1:length(obs)) / (length(obs) + 1)
> plot(-log10(ept), -log10(obs), main = "GWAS QQ plot adj. via Genomic Control", xlab="Expected -log10(pvalue)", ylab="Observed -log
10(pvalue)")
> abline(0, 1, col = "red")
```

```
> abline(h = 8, lty = 2)

> #----EIGENSTRAT adjustment, PCA----#
> adj.gkin = gkin
> diag(adj.gkin) = hom(g.dat)$Var
>
> # naxes = 3 is default value
> test.eg <- egscore(disease, data = g.dat, kin = adj.gkin, naxes = 2)
> descriptives.scan(test.eg)
Summary for top 10 results, sorted by P1df
           Chromosome   Position Strand A1 A2   N        effB    se_effB   chi2.1df
rs4571722           8   60326734      u  C  T 242 -2.7656036 0.3924961  49.648852
rs10008252          4  179853616      u  A  G 244 -1.7383304 0.3523691  24.337114
rs2881297          12   27298922      u  C  T 244 -1.6875438 0.4371139  14.904616
rs17071376          8    4643554      u  G  A 242 -1.5747884 0.4419575  12.696481
rs9950554          18   72793301      u  T  C 243  7.2383677 2.0552900  12.403237
rs10495538          2    6707489      u  C  T 244 -2.4190746 0.7085907  11.654875
rs9846935           3  124405352      u  C  T 244  1.8859754 0.5856865  10.369113
rs2028374           2  188691748      u  A  G 244  2.1137666 0.6597101  10.266153
rs534288            2   45739513      u  T  G 244 -1.2623034 0.4026488   9.828216
rs7663989           4  117632226      u  T  C 244 -0.8660629 0.2793947   9.608651
                 P1df         Pc1df       effAB          effBB chi2.2df P2df
rs4571722   1.838777e-12  8.201068e-12 -2.7656036 -5.531207e+00         NA    NA
rs10008252  8.086590e-07  1.706511e-06 -1.7383304 -3.476661e+00         NA    NA
rs2881297   1.130859e-04  1.804442e-04 -1.6875438 -3.375088e+00         NA    NA
rs17071376  3.663443e-04  5.474266e-04 -1.5747884 -3.149577e+00         NA    NA
rs9950554   4.285902e-04  6.348778e-04 21.5943214  6.927740e-02         NA    NA
rs10495538  6.403451e-04  9.276605e-04 -7.9658415  9.955606e-16         NA    NA
rs9846935   1.281411e-03  1.786589e-03  6.7401583  9.955606e-16         NA    NA
rs2028374   1.354930e-03  1.883300e-03  6.8661277  9.955606e-16         NA    NA
rs534288    1.718550e-03  2.357708e-03 -1.2623034 -2.524607e+00         NA    NA
rs7663989   1.936629e-03  2.639536e-03 -0.8660629 -1.732126e+00         NA    NA
>
> snp.eg <- row.names(results(test.eg))[results(test.eg)$P1df < alpha]
> pvalue.eg <- results(test.eg)$P1df[results(test.eg)$P1df < alpha]
> lambda.eg <- test.eg@lambda
> snp.eg
[1] "rs4571722"
> pvalue.eg
[1] 1.838777e-12
> lambda.eg
$estimate
[1] 1.062756
$se
[1] 0.003169877

> #----Inflation factor change----#
> for (k in 1:10){
+ test.tmp <- egscore(disease, data = g.dat, kin = adj.gkin, naxes = k)
+ print(test.tmp@lambda$estimate)
+ }
[1] 1.082203
[1] 1.062756
[1] 1.069796
[1] 1.07633
[1] 1.087112
[1] 1.090747
[1] 1.101529
[1] 1.110265
[1] 1.1159
[1] 1.127466
>
> #----Updated QQ plot with EIGENSTRAT----#
> obs <- sort(results(test.eg)$Pc1df)
> ept <- c(1:length(obs)) / (length(obs) + 1)
> qqplot(-log10(ept), -log10(obs), main = "GWAS QQ plot adj. w/ EIGENSTRAT", xlab="Expected -log10(pvalue)", ylab="Observed -log10(p
value)")
> abline(0, 1, col = "red")
> abline(h = 8, lty = 2)
> # Manhattan plot comparison
> plot(test.qt, col = "black")
> add.plot(test.eg, col = "gray", pch = 3)
> legend("topright", c("Original plot","After correction w/ EIGENSTRAT"), pch = c(1,3))

> #----Question 3, binary trait----#
> # load files to GenABEL
> convert.snp.tped(tped = "gwa_gabel.tped", tfam = "gwa_gabel.tfam", out = "gwa_gabel.raw", strand = "u")
```

```
Reading individual ids from file *gwa_gabel.tfam* ...
... done.  Read 244 individual ids from file *gwa_gabel.tfam*
Reading genotypes from file *gwa_gabel.tped* ...
...done.  Read 6388 SNPs from file *gwa_gabel.tped*
Writing to file *gwa_gabel.raw* ...
... done.
> b.dat <- load.gwaa.data(phen  = "gwa_gabel.praw",  gen = "gwa_gabel.raw",  force = T)
ids loaded...
marker names loaded...
chromosome data loaded...
map data loaded...
allele coding data loaded...
strand data loaded...
genotype data loaded...
snp.data object created...
assignment of gwaa.data object FORCED; X-errors were not checked!
>
> slotNames(b.dat)
[1] "phdata" "gtdata"
> slotNames(b.dat@gtdata)
 [1] "nbytes"     "nids"       "nsnps"      "idnames"    "snpnames"
 [6] "chromosome" "map"        "coding"     "strand"     "male"
[11] "gtps"
> colnames(b.dat@phdata)
[1] "id"      "sex"     "disease" "popn"
> # sample size
> b.dat@gtdata@nids
[1] 244
> # number of cases and controls
> case.size <- length(which(b.dat@phdata$disease == 1))
> control.size <- length(which(b.dat@phdata$disease == 0))
> case.size
[1] 87
> control.size
[1] 157
> # number of SNPs
> snpsb.total <- b.dat@gtdata@nsnps
> # GLM test
> testb.snp <- scan.glm(*disease ~ CRSNP*, family = binomial(), data = b.dat)
99.41%
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
> names(testb.snp)
 [1] "P1df"       "P2df"       "medChi1df"  "medChi2df"  "snpnames"
[6] "idnames"    "formula"    "family"     "map"        "chromosome"
[11] "effB"       "effAB"      "effBB"      "Pc1df"
> alpha <- 5e-8
> testb.snp$snpnames[testb.snp$P1df < alpha]
[1] "rs10008252" "rs4571722"
> testb.snp$P1df[testb.snp$P1df < alpha]
[1] 1.293455e-14 3.233446e-27
>
> # Score test
> testb.qt <- qtscore(disease,  data = b.dat, trait = "binomial")
>
> slotNames(testb.qt)
[1] "results"    "dimresults" "annotation" "lambda"     "idnames"
[6] "call"       "family"
> descriptives.scan(testb.qt)
Summary for top 10 results, sorted by P1df
           Chromosome   Position Strand A1 A2   N         effB     se_effB
rs4571722           8   60326734      u  C  T 242 -0.05946073  0.00639500
rs10008252          4 179853616      u  A  G 244  0.07408615  0.01029479
rs2881297          12   27298922      u  C  T 244  0.33087468  0.07372806
rs2994553           1   56558477      u  T  G 244  2.23225171  0.53814476
rs798658            3 150318957      u  A  G 244  2.26263438  0.54620704
rs10459871         16   79650568      u  C  G 244  3.08341655  0.74526127
rs7254875          19   61010979      u  C  T 244  3.25634032  0.79173503
rs3825391          12 109458229      u  A  C 244  2.05015042  0.57469941
rs10520770         18   44856962      u  T  C 244  1.83909581  0.53639996
rs7557417           2   43270587      u  G  A 244  1.86972960  0.58011923
           chi2.1df          P1df      effAB       effBB   chi2.2df          P2df
rs4571722  86.45287  1.431075e-20  0.0289590  0.01303155  104.99098  1.590370e-23
rs10008252 51.78916  6.179211e-13  0.1797235  0.01850095   54.05192  1.831366e-12
rs2881297  20.14009  7.197216e-06  0.4703037  0.04413619   20.43624  3.650294e-05
rs2994553  17.20632  3.353183e-05  2.3254658  6.02898551   17.35892  1.700432e-04
rs798658   17.15988  3.436179e-05  3.0270270  4.05263158   18.42844  9.961303e-05
```

```
rs10459871  17.11777  3.513198e-05  5.0177580  0.86163522    25.25330  3.283336e-06
rs7254875    16.91608  3.906912e-05  3.7076271  7.11864407    17.28620  1.763394e-04
rs3825391    12.72594  3.606181e-04  1.9294355  5.61290323    13.29198  1.299222e-03
rs10520770  11.75523  6.067242e-04  1.4111842  3.81578947    13.64109  1.091129e-03
rs7557417    10.38779  1.268515e-03  2.2916667  3.02500000    10.99810  4.090650e-03
                  Pc1df
rs4571722    6.380568e-18
rs10008252   2.455945e-11
rs2881297    3.138355e-05
rs2994553    1.190678e-04
rs798658     1.216186e-04
rs10459871   1.239783e-04
rs7254875    1.359378e-04
rs3825391    9.351915e-04
rs10520770   1.469637e-03
rs7557417    2.790680e-03
> row.names(results(testb.qt))[results(testb.qt)$P1df < alpha]
[1] "rs10008252" "rs4571722"
> results(testb.qt)$P1df[results(testb.qt)$P1df < alpha]
[1] 6.179211e-13 1.431075e-20
> results(testb.qt)$Pc1df[results(testb.qt)$Pc1df < alpha]
[1] 2.455945e-11 6.380568e-18
> #----End of Question 3, binary trait----#
```

References

1. [R]
   www.r-project.org

2. [GenABEL]
   Y. S. Aulchenko, S. Ripke, A. Isaacs and C. M. van Duijn (2007). **GenABEL: an R library for genome-wide association analysis**. *Bioinformatics*

3. [Devlin and Roeder 1999]
   B. Devlin and Kathryn Roeder (1999). **Genomic Control for Association Studies**. *Biometrics*

4. [Devlin et al. 2001]
   B Devlin, Kathryn Roeder and Larry Wasserman (2001). **Genomic Control, a New Approach to Genetic-Based Association Studies**. *Theoretical Population Biology*

5. [Price et al. 2006]
   Alkes L Price, Nick J Patterson, Robert M Plenge, Michael E Weinblatt, Nancy A Shadick and David Reich (2006). **Principal components analysis corrects for stratification in genome-wide association studies**. *Nature Genetics*