ESP-2011-29 `GenABEL-package` tutorial

Copyright 2011, Yurii Aulchenko

Independent researcher and consultant
yurii [dot] aulchenko [at] gmail [dot] com

October 18, 2011

Contents

Abstract

More extensive documenation can be found in `GenABEL-suite` tutorial available at (`http://www.genabel.org`).

1 Import the data from PLINK files

```
GenABEL v. 1.7-0 (October 05, 2011) loaded
```

In this section, you will convert PLINK data in `GenABEL-package` format and load the data set in `R`. More extensive introduction to converting data from different formats to `GenABEL-package` format can be found in chapter "Importing data to GenABEL-package" of `GenABEL-suite` tutorial and especially through the help pages.

To convert PLINK data in `GenABEL-package` format, we suggest the following procedure. First, you convert PLINK 'ped' files into PLINK 'tped' files. Then you can use `GenABEL-package` function `convert.snp.tped` to convert genotypic data from the 'tped' format to `GenABEL-package` binary 'raw' format. These data could then be loaded to `GenABEL-package`. While first step is not strictly necessary (`GenABEL-package` has a function to convert the data from 'ped' format, `convert.snp.ped`), this is recommended, especially for large data sets, as it works faster and is less demanding on computer RAM. Afetr loading the data into `GenABEL-package`, you will need to perform few extra operations, such as chromosome and strand renaming, to finalize the data import.

You can convert PLINK 'ped' formatted data to 'tped' format using the shell command

```
user@server> plink --file indata/ge03d2ex --recode --transpose --out ge03d2ex

@----------------------------------------------------------@
|        PLINK!       |     v1.07      |   10/Aug/2009     |
|----------------------------------------------------------|
|  (C) 2009 Shaun Purcell, GNU General Public License, v2  |
|----------------------------------------------------------|
|  For documentation, citation & bug-report instructions:  |
|        http://pngu.mgh.harvard.edu/purcell/plink/        |
@----------------------------------------------------------@

Web-based version check ( --noweb to skip )
Recent cached web-check found...Problem connecting to web

Writing this text to log file [ ge03d2ex.log ]
Analysis started: Tue Oct 18 17:11:44 2011

Options in effect:
        --file indata/ge03d2ex
        --recode
        --transpose
        --out ge03d2ex

4000 (of 4000) markers to be included from [ indata/ge03d2ex.map ]
136 individuals read from [ indata/ge03d2ex.ped ]
0 individuals with nonmissing phenotypes
Assuming a disease phenotype (1=unaff, 2=aff, 0=miss)
Missing phenotype value is also -9
0 cases, 0 controls and 136 missing
```

```
72 males, 64 females, and 0 of unspecified sex
Before frequency and genotyping pruning, there are 4000 SNPs
136 founders and 0 non-founders found
197 heterozygous haploid genotypes; set to missing
Writing list of heterozygous haploid genotypes to [ ge03d2ex.hh ]
2 SNPs with no founder genotypes observed
Warning, MAF set to 0 for these SNPs (see --nonfounders)
Writing list of these SNPs to [ ge03d2ex.nof ]
Total genotyping rate in remaining individuals is 0.981283
0 SNPs failed missingness test ( GENO > 1 )
0 SNPs failed frequency test ( MAF < 0 )
After frequency and genotyping pruning, there are 4000 SNPs
After filtering, 0 cases, 0 controls and 136 missing
After filtering, 72 males, 64 females, and 0 of unspecified sex
Writing transposed ped file to [ ge03d2ex.tped ]
Writing family information to [ ge03d2ex.tfam ]

Analysis finished: Tue Oct 18 17:11:45 2011
```

This should produce two files, 'ge03d2ex.tfam' and 'ge03d2ex.tped'. The 'tfam' file follows the same format as regular 'fam' file. Here are few first lines of this file:

```
1 id199 0 0 1 -9
2 id287 0 0 2 -9
3 id300 0 0 1 -9
4 id403 0 0 2 -9
5 id415 0 0 2 -9
```

The 'tped' contains transposed genetic data, i.e. each SNP is occupying one row, while samples are 'columns':

```
1 rs1646456 0 653 G C G C C C G C C C G C G C C C C C G C G C G C G G C C G C G  ...
1 rs4435802 0 5291 C C C C C C C C A C C C A C C C C C C C A C A C C C C C C C C ...
1 rs946364 0 8533 C T C T T T C T 0 0 C T T T T T T T C T C T T T C C T T C T C  ...
1 rs299251 0 10737 0 0 G A A A A A A A G A A A A A A A A A A A A A A A A A A A A A ...
1 rs2456488 0 11779 G G C G G G C G G G C G C G G G G G G G G G C G G G C C G G  ...
```

Note that information on chromosome and position is included as part of this file.

To convert the 'tped' file to `GenABEL-package` 'raw' format, start `R` and load `GenABEL-package` with

```
> library(GenABEL)
```

then use `convert.snp.tped` to convert the data:

```
> convert.snp.tped(tpedfile="ge03d2ex.tped",tfamfile="ge03d2ex.tfam",out="ge03d2ex.raw")
```

```
Reading individual ids from file 'ge03d2ex.tfam' ...
... done.  Read 136 individual ids from file 'ge03d2ex.tfam'
Reading genotypes from file 'ge03d2ex.tped' ...
...done.  Read 4000 SNPs from file 'ge03d2ex.tped'
Writing to file 'ge03d2ex.raw' ...
... done.
```

Next, you can load the data into RAM with

```
> ge03d2ex <- load.gwaa.data(phenofile="indata/ge03d2ex.phe",
+   genofile="ge03d2ex.raw",id="IID")

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!
```

here, 'id="IID"' indicates that in the phenotypic data file 'indata/ge03d2ex.phe' the column containing sample identification code is named "IID".

However, this is not the end of the story. As you might expect, we still miss information on strand (it is neither part of 'tped' nor 'tfam'!). Thus, strand is unknown (coded with "u") for just loaded data, as you can see from

```
> strand(ge03d2ex)[1:10]

rs1646456 rs4435802  rs946364  rs299251 rs2456488 rs3712159 rs4602970  rs175910
      "u"       "u"       "u"       "u"       "u"       "u"       "u"       "u"
rs1919938 rs8892781
      "u"       "u"

> table(strand(ge03d2ex))

   u
4000
```

The information on strand can be found in the 'extended' map file 'indata/ge03d2ex.map.ext'. Here are first few lines of this file:

```
chromosome markername position strand coding
1 rs1646456 653 + CG
1 rs4435802 5291 + CA
1 rs946364 8533 - TC
1 rs299251 10737 + AG
```

We will read this information in R and use it to fix the strand. Read the extended map with and check first three raws:

```
> map <- read.table("indata/ge03d2ex.map.ext",head=TRUE)
> map[1:3,]

  chromosome markername position strand coding
1          1  rs1646456      653      +     CG
2          1  rs4435802     5291      +     CA
3          1   rs946364     8533      -     TC
```

Before fixing the starnd using `strand<-` method, we need to ceck if strand information in 'map' ('map$strand') is specified for exacly the same SNPs (number and order is the same). This can be done by

```
> table( snpnames(ge03d2ex) == map$markername )

TRUE
4000
```

Because we see that number and order of markers is the same, we can now fix strand with

```
> strand(ge03d2ex) <- map$strand
```

and check what is now strand information summary with

```
> table(strand(ge03d2ex))

   +    -
1991 2009
```

Finally, you may need to fix the chromosome numeration. PLINK was designed to work with human data, and in 'tped' format uses integers to denote chromosomes. Let us check what chromosomes are present in the data:

```
> table(chromosome(ge03d2ex))

   1    2   23    3
1904 1055  288  753
```

You can see that there are 4 chromosomes in the data set, with chromosome '23' corresponding to X-chromosome. `GenABEL-package` assumes that chromosomes denoted as integers are autosomes and other parts of the genome are denotes using strings, such as 'X' for X-chromosome, 'Y' for Y-chromosome and 'mt' for mitochondrial genome. Thus, in order for `GenABEL-package` to interpret the data correctly, you need to recode chromosome 23 to 'X'. This can be done using `GenABEL-package` function 'recodeChromosome':

```
> ge03d2ex <- recodeChromosome(ge03d2ex,rules=list("23"="X"))

Recoded chromosome for 288 SNPs ( 23 -> X )
```

After this, table of chromosomes should produce

```
> table(chromosome(ge03d2ex))

   1    2    3    X
1904 1055  753  288
```

Now `GenABEL-package` will be able to distingush sex chromosomes from autosomes.

This completes conversion from PLINK to `GenABEL-package` format. You can save the data with

```
> save(ge03d2ex,file="ge03d2ex.RData")
```

for later use.

Finally, you can check how many SNPs are in the data set by

```
> nsnps(ge03d2ex)
```

```
[1] 4000
```

and check the number of samples by

```
> nids(ge03d2ex)
```

```
[1] 136
```

1.1 Exercise

Ex. 1 — Convert larger data set 'ge03d2' to `GenABEL-package` format and load the data into `GenABEL-package`.

Ex. 2 — How many SNPs are in the data set?

Ex. 3 — How many samples are in the data set?

2 Exploring the data with GenABEL-package

In this section you will learn basic command for exploring genetic data with GenABEL-package. More extensive introduction to this topic is available as part of GenABEL-suite tutorial, see chapter "Introduction to GenABEL-package".

If you are continuing the exercise from previous section 1 ("Import the data from PLINK files"), skip this paragraph, bacause the library and data required is already loaded in your working R environment. If you have left R environment meanwhile, start R and load GenABEL-package with

```
> library(GenABEL)
```

```
GenABEL v. 1.7-0 (October 05, 2011) loaded
```

and load the data you have saved in previous section with

```
> load("ge03d2ex.RData")
```

Alternatively, the data were saved for you in "outdata" directory and can be loaded with

```
> load("outdata/ge03d2ex.RData")
```

When you ensure that GenABEL-package and data set 'ge03d2ex' are loaded, you can continue with this exercise. In previous section you have already learned some methods defined for the GWA data in GenABEL-package. For example, to check the number of SNPs in the data set 'ge03d2ex' you can request

```
> nsnps(ge03d2ex)
```

```
[1] 4000
```

Very similarly, the number of samples can be learned by running the command

```
> nids(ge03d2ex)
```

```
[1] 136
```

While above two methods return a scalar (single number), many methods return a vector. Among these are such methdods as 'strand', 'chromosome', 'snpnames', 'coding', 'refallele', 'effallele' etc., which return information regarding every SNPs in the data; and such methods as `male` and `idnames`[1], which return information about every sample in the data set.

For example, you can check the sex for first 10 samples with

```
> male(ge03d2ex)[c(1:10)]
```

```
id199 id287 id300 id403 id415 id666 id689 id765 id830 id908
    1     0     1     0     0     1     1     0     0     1
```

or make sex table with

```
> table(male(ge03d2ex))
```

[1] See help("snp.data-class"), help("gwaa.data-class") and GenABEL-suite tutorial for more details

```
 0  1
64 72
```

which shows that there are 72 and 64 females in the data.

SNP summary data can be generated using function 'summary' for genotypic part of the data (accessed with method 'gtdata'). Let us generate summary for the data set 'g03d2ex':

```
> snpSummary <- summary(gtdata(ge03d2ex))
```

'snpSummary' is a regular `R` object (data frame), and you can easily analyse and explore this table. For example, let us check the first 5 rows of this table:

```
> snpSummary[c(1:5), ]
```

```
          Chromosome Position Strand A1 A2 NoMeasured  CallRate        Q.2 P.11
rs1646456          1      653      +  C  G        135 0.9926471 0.33333333   57
rs4435802          1     5291      +  C  A        134 0.9852941 0.07462687  114
rs946364           1     8533      -  T  C        134 0.9852941 0.27611940   68
rs299251           1    10737      +  A  G        135 0.9926471 0.04444444  123
rs2456488          1    11779      +  G  C        135 0.9926471 0.34814815   59
          P.12 P.22    Pexact        Fmax      Plrt
rs1646456   66   12 0.3323747 -0.10000000 0.2404314
rs4435802   20    0 1.0000000 -0.08064516 0.2038385
rs946364    58    8 0.3949055 -0.08275286 0.3302839
rs299251    12    0 1.0000000 -0.04651163 0.4549295
rs2456488   58   18 0.5698988  0.05343327 0.5360019
```

here, `[c(1:5), ]` means that we want to see rows from 1 to 5, and all columns (nothing is specifid after the colon). The table first provides SNP annotation information such as chromosome, position, and strand. It also reports the reference ('A1') and effective (aka 'coded') allele ('A2'). It then reports number of samples in which genotyping was successful, proportion of successful genotypes ('CallRate'), frequency of the effective allele 'A2', genotypic counts, and exact p-value for the Hardy-Weinberg equilibium (HWE) test.

From this summary table, you can obtain information for specific SNPs if you know their name or index. For example, to obtain information for SNPs 2 (rs4435802) and 5 (rs2456488), you can request

```
> snpSummary[c(1,5), ]
```

```
          Chromosome Position Strand A1 A2 NoMeasured  CallRate       Q.2 P.11
rs1646456          1      653      +  C  G        135 0.9926471 0.3333333   57
rs2456488          1    11779      +  G  C        135 0.9926471 0.3481481   59
          P.12 P.22    Pexact        Fmax      Plrt
rs1646456   66   12 0.3323747 -0.10000000 0.2404314
rs2456488   58   18 0.5698988  0.05343327 0.5360019
```

or

```
> snpSummary[c("rs4435802","rs2456488"), ]
```

```
          Chromosome Position Strand A1 A2 NoMeasured  CallRate        Q.2 P.11
rs4435802          1     5291      +  C  A        134 0.9852941 0.07462687  114
rs2456488          1    11779      +  G  C        135 0.9926471 0.34814815   59
          P.12 P.22    Pexact        Fmax      Plrt
rs4435802   20    0 1.0000000 -0.08064516 0.2038385
rs2456488   58   18 0.5698988  0.05343327 0.5360019
```

Next, you can perform computations using this table. For example, we can check how many SNPs were called with 'CallRate' below 0.98. To retrieve the 'CallRate', you can use 'snpSummary$CallRate' (which is a long vector containing the 'CallRate' column of the 'snpSummary' table):

```
> cr <- snpSummary$CallRate
```

You can try to see the content of 'cr' by simply typing 'cr' and pressing 'enter. Now, on this vector, you can perform logical operation '<0.98':

```
> cr98 <- (cr<0.98)
```

Try to see the content of 'cr98' by simply typing 'cr98' and pressing 'enter. The element of this vector is TRUE when the condition holds. Finally you can generate the table:

```
> table(cr98)
```

```
cr98
FALSE  TRUE
 2961  1039
```

Thus, 1039 SNPs have call rate below 0.98. You can obtain the same answer in a number of ways in one-go:

```
> table(summary(gtdata(ge03d2ex))$CallRate < 0.98)
```

```
FALSE  TRUE
 2961  1039
```

```
> sum(summary(gtdata(ge03d2ex))$CallRate < 0.98)
```

```
[1] 1039
```

In similar manner you can check how many SNPs have minor allele frequency below 0.05. For this, you need to compute the minor allele frequency (MAF) first ('Q.2' of the summary is the frequency of the effective allele, which does not have to be minor one). By definition, MAF is the smaller of the frequencies of the two SNP allelels and thus can be computed with

```
> maf <- pmin(snpSummary$Q.2,1.-snpSummary$Q.2)
```

from summary of 'maf'

```
> summary(maf)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00000 0.06015 0.14180 0.18240 0.28890 0.50000
```

you can see that it ranhes between 0 and 0.5, as it should.

Now, to figure out the number of SNPs with MAF < 0.05, you can run

```
> sum(maf<0.05)

[1] 830
```

Note that 'minor' allele is very poor notion in many situations, because minor allele is relative to population, e.g. for an 'AG' polymorphism it could be that 'G' is less frequent than 'A' in Europeans (thus 'A' is minor'), but the same 'A' is prevalent in say some other popualtion (in which 'G' will be minor). Also, if frequency is close to 0.5, the 'minor' allele can easily swap depending on the subgroup (cases, controls, males, femailes ...) you consider.

You can compute SNP summaries for subset of individuals, for example for all males by running 'summary' command on males only:

```
> snpSumMale <- summary(gtdata(ge03d2ex[which(male(ge03d2ex)==1), ]))
```

Here, expression 'which(male(ge03d2ex)==1)' return the indexes for individuals for which the condition 'male(ge03d2ex)==1' is true (so, they are '1', males). Similarly, for females:

```
> snpSumFemale <- summary(gtdata(ge03d2ex[which(male(ge03d2ex)==0), ]))
```

Then you can check summaries for e.g. SNP rs7760396 in total population, males and females:

```
> snpSummary["rs7760396",]   # all

          Chromosome Position Strand A1 A2 NoMeasured CallRate   Q.2 P.11 P.12
rs7760396          2  6567001      +  G  A        136        1 0.375   55   60
          P.22   Pexact       Fmax      Plrt
rs7760396   21 0.470375 0.05882353 0.4936544

> snpSumMale["rs7760396",]   # Male population

          Chromosome Position Strand A1 A2 NoMeasured CallRate       Q.2 P.11
rs7760396          2  6567001      +  G  A         72        1 0.3888889   28
          P.12 P.22    Pexact       Fmax      Plrt
rs7760396   32   12 0.6215491 0.06493506 0.5823002

> snpSumFemale["rs7760396",] # Female population

          Chromosome Position Strand A1 A2 NoMeasured CallRate      Q.2 P.11
rs7760396          2  6567001      +  G  A         64        1 0.359375   27
          P.12 P.22   Pexact       Fmax      Plrt
rs7760396   28    9 0.785734 0.04984093 0.6908271
```

Above we have considered summaries for SNPs. Similarly, function `perid.summary` provides summaries for samples:

```
> idSummary <- perid.summary(ge03d2ex)
```

(note that you do not need to apply 'gtdata' to the data in this case). Summary for first five individuals is

```
> idSummary[1:5,]

      NoMeasured NoPoly       Hom    E(Hom)       Var           F  CallPP
id199       3934   3917 0.7638536 0.7396254 0.5008711  0.09305118 0.98350
id287       3941   3925 0.5102766 0.7397034 1.4415981 -0.88140515 0.98525
id300       3939   3922 0.7623762 0.7393671 0.4520686  0.08828176 0.98475
id403       3943   3926 0.7479077 0.7396142 0.4420231  0.03185092 0.98575
id415       3927   3909 0.7593583 0.7399439 0.4483163  0.07465464 0.98175
            Het
id199 0.2361464
id287 0.4897234
id300 0.2376238
id403 0.2520923
id415 0.2406417
```

Here, 'NoMeasured' is total number of SNPs successfully genotyped for particular sample, and 'CallPP' is the sample call rate[2].

Similarly to SNP summary table, you can perform operations on the elements of sample summary table. For example, to check what number of samples had call rate below 0.95, run

```
> sum(idSummary$CallPP<0.95)

[1] 1
```

To inspect this sample in more details you can combine computations and subsetting:

```
> idSummary[which(idSummary$CallPP<0.98),]

       NoMeasured NoPoly       Hom    E(Hom)      Var         F  CallPP
id7715       2105   2094 0.7904988 0.7406885 1.710966 0.1920869 0.52625
             Het
id7715 0.2095012
```

Thus, the sample with call rate < 0.95 actually had very low call rate of 0.526

2.1 Exercises

Explore the larger data set 'ge03d2' (not 'ge03d2ex'!)
Ex. 4 — Load the data set 'ge03d2'

Ex. 5 — How many SNPs are in this data set?

Ex. 6 — How many people are described in the data set?

Ex. 7 — How many of these people are males and how many are females?

Ex. 8 — How many SNPs are called poorely (call $< 98\%$)?

Ex. 9 — How many SNPs have a MAF<=0.05?

[2] see help(perid.summary) for more details

Ex. 10 — How many SNPs are out of HWE if you use a p-value threshold of 0.001?

Ex. 11 — How many samples have call rate below 95%?

Ex. 12 — What is the HWE *p*-value for `rs9807290`?

Ex. 13 — What are the alleles for SNP `rs7760396`; which allele is the minor allele?

Ex. 14 — What is the frequeny of the effective allele of `rs7760396` in the total population?

Ex. 15 — What is the frequeny of the minor allele of `rs7760396` in males?

Ex. 16 — What is the frequeny of the minor allele of `rs7760396` in females?

3 Quality control

```
GenABEL v. 1.7-0 (October 05, 2011) loaded
```

In this section you will perform extensive quality control (QC) of the data. This will be done in three steps. First, you will filter out samples and SNPs with do not meet certain (relaxed) quality criterial. Secondly, you will identify 'genetic outliers' - samples, which seem to belong to a different genetic population compared with the majority of the samples. Thirdly, you will repeat filtering of samples and SNP using strict criteria.

If you are continuing the exercise from previous section 1 ("Import the data from PLINK files"), skip this paragraph, bacause the library and data required is already loaded in your working `R` environment. If you have left `R` environment meanwhile, start `R` and load `GenABEL-package` with

```
> library(GenABEL)
```

and load the data you have saved in previous section with

```
> load("ge03d2ex.RData")
```

Alternatively, the data were saved for you in "outdata" directory and can be loaded with

```
> load("outdata/ge03d2ex.RData")
```

In the first step, let us filter the SNPs, wich have call rate below 98% and samples with call rate below 95%. These thresholds are SNP-chip specific, consult your manufacturer for the threshold to be used. We filter out all monomorphic SNPs by using $MAF = 1^{-10}$ threshold. We will also disable the checks for autosomal heterozygosity and HWE by setting 'het.fdr' and 'p.level' to zero. These 'relaxed' thresholds are chosen because in a case the study inclues representatives of different populations, such statistics as HWE p-value and heterozygosity may be affected. To run QC with these thresholds, use

```
> qc <- check.marker(ge03d2ex,call=0.98,perid.call=0.95,p.level=0,maf=1e-10,het.fdr=0)

Excluding people/markers with extremely low call rate...
4000 markers and 136 people in total
0 people excluded because of call rate < 0.1
6 markers excluded because of call rate < 0.1
Passed: 3994 markers and 136 people

Running sex chromosome checks...
197 heterozygous X-linked male genotypes found
1 X-linked markers are likely to be autosomal (odds > 1000 )
2 male are likely to be female (odds > 1000 )
0 female are likely to be male (odds > 1000 )
0 people have intermediate X-chromosome inbreeding (0.5 > F > 0.5)
If these people/markers are removed, 0 heterozygous male genotypes are left
Passed: 3993 markers and 134 people

no X/Y/mtDNA-errors to fix
```

```
RUN 1
3993 markers and 134 people in total
25 (0.6260957%) markers excluded as having low (<1e-08%) minor allele frequency
1010 (25.29426%) markers excluded because of low (<98%) call rate
0 (0%) markers excluded because they are out of HWE (P <0)
1 (0.7462687%) people excluded because of low (<95%) call rate
Mean autosomal HET is 0.2541314 (s.e. 0.03744539)
0 people excluded because too high autosomal heterozygosity (FDR <0%)
Mean IBS is 0.7883373 (s.e. 0.02070973), as based on 2000 autosomal markers
2 (1.492537%) people excluded because of too high IBS (>=0.95)
In total, 2964 (74.2299%) markers passed all criteria
In total, 131 (97.76119%) people passed all criteria

RUN 2
2964 markers and 131 people in total
2 (0.06747638%) markers excluded as having low (<1e-08%) minor allele frequency
0 (0%) markers excluded because of low (<98%) call rate
0 (0%) markers excluded because they are out of HWE (P <0)
0 (0%) people excluded because of low (<95%) call rate
Mean autosomal HET is 0.2541993 (s.e. 0.03779273)
0 people excluded because too high autosomal heterozygosity (FDR <0%)
Mean IBS is 0.789515 (s.e. 0.01968781), as based on 2000 autosomal markers
0 (0%) people excluded because of too high IBS (>=0.95)
In total, 2962 (99.93252%) markers passed all criteria
In total, 131 (100%) people passed all criteria

RUN 3
2962 markers and 131 people in total
0 (0%) markers excluded as having low (<1e-08%) minor allele frequency
0 (0%) markers excluded because of low (<98%) call rate
0 (0%) markers excluded because they are out of HWE (P <0)
0 (0%) people excluded because of low (<95%) call rate
Mean autosomal HET is 0.2541993 (s.e. 0.03779273)
0 people excluded because too high autosomal heterozygosity (FDR <0%)
Mean IBS is 0.7887742 (s.e. 0.0196495), as based on 2000 autosomal markers
0 (0%) people excluded because of too high IBS (>=0.95)
In total, 2962 (100%) markers passed all criteria
In total, 131 (100%) people passed all criteria
```

You can see that QC is done iteratively till all thresholds are met. QC needs to be done in such manner bacause as soon as you exclude an SNP or a sample, other statistics change and need to be recomputed and re-checked. The summary of SNPs and samples, whcih do not pass specified thresholds can be obtained with

```
> summary(qc)

$`Per-SNP fails statistics`
          NoCall NoMAF NoHWE Redundant Xsnpfail
NoCall       1010     6     0         0        0
NoMAF          NA    21     0         0        0
NoHWE          NA    NA     0         0        0
Redundant      NA    NA    NA         0        0
Xsnpfail       NA    NA    NA        NA        1

$`Per-person fails statistics`
            IDnoCall HetFail IBSFail isfemale ismale isXXY otherSexErr
IDnoCall           1       0       0        0      0     0           0
HetFail           NA       0       0        0      0     0           0
IBSFail           NA      NA       2        0      0     0           0
isfemale          NA      NA      NA        2      0     0           0
ismale            NA      NA      NA       NA      0     0           0
isXXY             NA      NA      NA       NA     NA     0           0
otherSexErr       NA      NA      NA       NA     NA    NA           0
```

From this output you can see that 6 markers failed both MAF and call criteria; 2 samples were male according to the records while being recognised as females using genomic data, etc.

The data set, which includes SNPs and samples, which pass QC thresholds can be generated with

```
> data1ex <- ge03d2ex[qc$idok,qc$snpok]
```

Finally, you need to run 'Xfix' command to fix heterozygous X-chromosome SNPs in males:

```
> data1ex <- Xfix(data1ex)

no X/Y/mtDNA-errors to fix
```

After first step of QC, there are 131 samples and 2962 SNPs left.

```
> nids(data1ex)

[1] 131

> nsnps(data1ex)

[1] 2962
```

Now, let us detect genetic outliers using analysis of the genomic kinship matrix. For this, we will compute genomic kinship matrix with

```
> data1ex.gkin <- ibs(data1ex[,autosomal(data1ex)],w="freq")
```

and run classical multi-dimensional scaling on distance matrix defined as $(0.5 - data1ex.gkin)$:

```
> data1ex.mds <- cmdscale(as.dist(0.5-data1ex.gkin))
> data1ex.mds[1:5,]
```

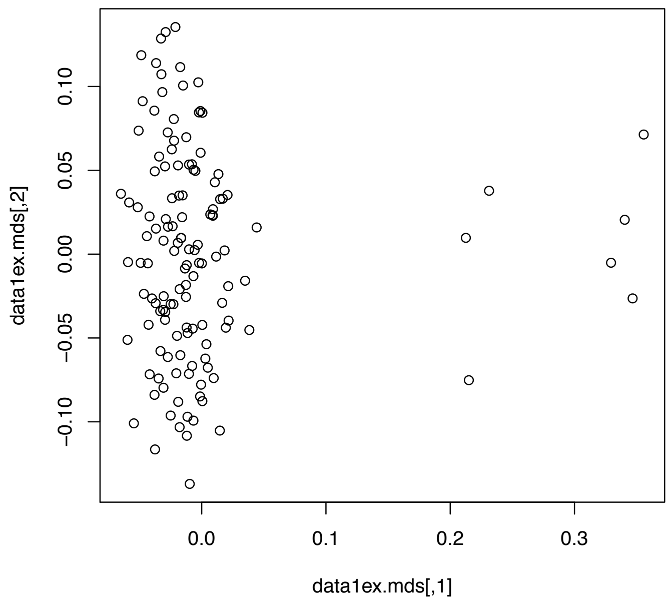

Figure 1: PC plot for genomic kinship of 'data1ex'

```
             [,1]         [,2]
id199 -0.03810276  0.085619047
id287  0.21501021 -0.075162286
id300 -0.05839430  0.030846261
id403 -0.03776546  0.049398829
id415 -0.01205285 -0.006548708
```

'data1ex.mds' consist of two first Principal Components (PCs) of variation of the genomic kinship matrix. You can plot this matrix with

```
> plot(data1ex.mds)
```

In figure 1, each point corresponds to a sample. It is visible from the graph that most people cluster around zero, with few outliers. To check how many people are genetic outliers, we can check how many points have first PC > 0.1:

```
> sum(data1ex.mds[,1]>0.1)
```

```
[1] 7
```

At this point, let us save the GWA data, genomic kinship and PC-matrix as 'data1ex.RData':

```
> save(data1ex,data1ex.mds,data1ex.gkin,file="data1ex.RData")
```

for future use.

Now, let us exclude genetic outliers from the data set and perform the second round of QC using more strict thresholds. The major (non-outliers) group can be identified with

```
> noOutliers <- which(data1ex.mds[,1]<0.1)
```

and the new data set (excluding outliers) can be generated with

```
> data1ex.noOutliers <- data1ex[noOutliers,]
> nids(data1ex.noOutliers)
```

```
[1] 124
```

Let us run QC using SNP call rate of 0.98, sample call rate 0.95, HWE p-value of 0.00001 MAF of 0.05. Let us also keep 'het.fdr' (we can now use strict autosomal heterozygosity and HWE thresholds, as our data should consist of samples coming from the same population, as reflected by PC-plot).

```
> qc2 <- check.marker(data1ex.noOutliers,call=0.98,perid.call=0.95,p.level=1e-5,maf=0.05)
```

```
Excluding people/markers with extremely low call rate...
2962 markers and 124 people in total
0 people excluded because of call rate < 0.1
0 markers excluded because of call rate < 0.1
Passed: 2962 markers and 124 people

Running sex chromosome checks...
0 heterozygous X-linked male genotypes found
0 X-linked markers are likely to be autosomal (odds > 1000 )
0 male are likely to be female (odds > 1000 )
0 female are likely to be male (odds > 1000 )
0 people have intermediate X-chromosome inbreeding (0.5 > F > 0.5)
If these people/markers are removed, 0 heterozygous male genotypes are left
Passed: 2962 markers and 124 people

no X/Y/mtDNA-errors to fix


RUN 1
2962 markers and 124 people in total
708 (23.90277%) markers excluded as having low (<5%) minor allele frequency
0 (0%) markers excluded because of low (<98%) call rate
0 (0%) markers excluded because they are out of HWE (P <1e-05)
0 (0%) people excluded because of low (<95%) call rate
Mean autosomal HET is 0.3140896 (s.e. 0.01992209)
0 people excluded because too high autosomal heterozygosity (FDR <1%)
Mean IBS is 0.7419457 (s.e. 0.01423362), as based on 2000 autosomal markers
0 (0%) people excluded because of too high IBS (>=0.95)
In total, 2254 (76.09723%) markers passed all criteria
In total, 124 (100%) people passed all criteria
```

```
RUN 2
2254 markers and 124 people in total
0 (0%) markers excluded as having low (<5%) minor allele frequency
0 (0%) markers excluded because of low (<98%) call rate
0 (0%) markers excluded because they are out of HWE (P <1e-05)
0 (0%) people excluded because of low (<95%) call rate
Mean autosomal HET is 0.3140896 (s.e. 0.01992209)
0 people excluded because too high autosomal heterozygosity (FDR <1%)
Mean IBS is 0.7422442 (s.e. 0.01409061), as based on 2000 autosomal markers
0 (0%) people excluded because of too high IBS (>=0.95)
In total, 2254 (100%) markers passed all criteria
In total, 124 (100%) people passed all criteria
```

```
> summary(qc2)
```

```
$`Per-SNP fails statistics`
          NoCall NoMAF NoHWE Redundant Xsnpfail
NoCall         0     0     0         0        0
NoMAF         NA   708     0         0        0
NoHWE         NA    NA     0         0        0
Redundant     NA    NA    NA         0        0
Xsnpfail      NA    NA    NA        NA        0

$`Per-person fails statistics`
            IDnoCall HetFail IBSFail isfemale ismale isXXY otherSexErr
IDnoCall           0       0       0        0      0     0           0
HetFail           NA       0       0        0      0     0           0
IBSFail           NA      NA       0        0      0     0           0
isfemale          NA      NA      NA        0      0     0           0
ismale            NA      NA      NA       NA      0     0           0
isXXY             NA      NA      NA       NA     NA     0           0
otherSexErr       NA      NA      NA       NA     NA    NA           0
```

Thi 'final' clean homogenous data set can be generated with

```
> ge03d2ex.clean <- data1ex.noOutliers[qc2$idok,qc2$snpok]
> ge03d2ex.clean <- Xfix(ge03d2ex.clean)
```

```
no X/Y/mtDNA-errors to fix
```

In the final data set, there are 124 samples and 2254 SNPs:

```
> nids(ge03d2ex.clean)
```

```
[1] 124
```

```
> nsnps(ge03d2ex.clean)
```

```
[1] 2254
```

At this point, let us save these data for future use:

```
> save(ge03d2ex.clean,file="ge03d2ex.clean.RData")
```

3.1 Exercises

Ex. 17 — If you did not do this yet, start `R`, load `GenABEL-package` and load the data set 'ge03d2' (not 'ge03d2ex'!).

Ex. 18 — Perform first round of quality control using the follwoing filters: SNP call rate of 0.98, sample call rate 0.95, HWE *p*-value of '0' (extreme) MAF of '1e-10' (exclude only monomorphic SNPs), het.fdr of '0' and default for other values

Ex. 19 — How many people are included into QC'ed data set after the first round of QC (before excluding genetic outliers)?

Ex. 20 — How many SNP's are included in the QC'ed data set after the first round of QC (before excluding genetic outliers)?

Ex. 21 — How many people were excluded because of sex errors?

Ex. 22 — How many people were excluded because of sample duplications (IBS>0.95)?

Ex. 23 — Detect genetic outliers using analysis of the genomic kinship matrix. Save data, genomic kinship and PC-matrix as 'data1.RData'. How many people are to be excluded because they are genetic outliers?

Ex. 24 — Exclude genetic outliers, perform second round of QC using thresholds of SNP call rate of 0.98, sample call rate 0.95, HWE *p*-value of 0.00001 MAF of 0.05. Save the data using the name 'ge03d2.clean'

Ex. 25 — How many samples are in the final data set?

Ex. 26 — How many SNPs are in the final data set?

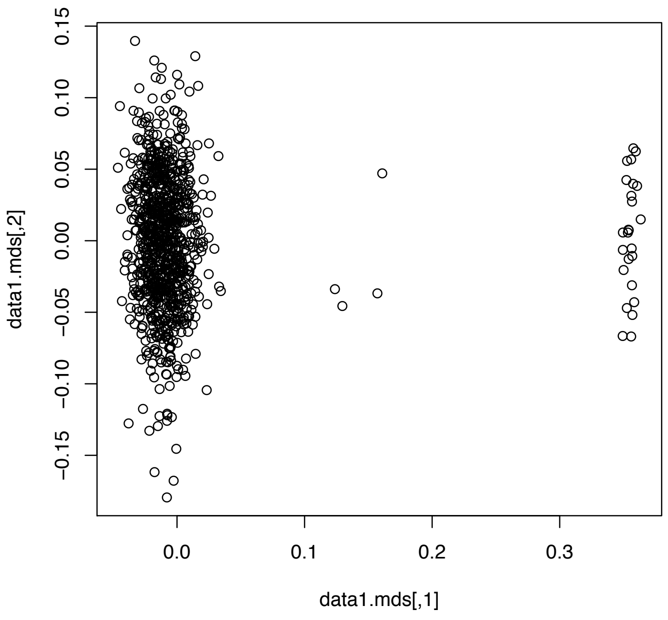

Figure 2: PC plot for genomic kinship of 'data1'

4 Genome-wide association analysis

In this section we will cover basic GenABEL-package GWA analysis of binary and quantitative traits in population-based samples.

If you did not do so yet, start R, and load cleaned data set 'ge03d2ex.clean.RData':

```
> library(GenABEL)

GenABEL v. 1.7-0 (October 05, 2011) loaded

Installed GenABEL version (1.7-0) is not the same as stable
version available from CRAN (1.6-7). Unless used intentionally,
consider updating to the latest CRAN version. For that, use
'install.packages("GenABEL")', or ask your system administrator
to update the package.

> load("ge03d2ex.clean.RData")
```

4.1 Analysis of binary traits

Analysis of binary traits within GenABEL-package can be performed by using a large number of functions, such as `ccfast`, `mlreg`, `qtscore`, `egstrat`, etc. For compatibility with PLINK exercises, we will use `mlreg` function, with allows for logistic regression analysis of binary traits. Other, alternative function of GenABEL-package may be very useful in other specific situations, such as dealing with population stratification or providing very fast analysis.

For the data set 'ge03d2ex.clean', let us perform logistic regression GWA analysis of the binary trait 'dm2'. First, let us check how many cases and control are included in the data set:

```
> table(phdata(ge03d2ex.clean)$dm2)

 0  1
47 77
```

and then run analysis with

```
> logReg <- mlreg(dm2~1,data=ge03d2ex.clean,trait="binomial")
```

here, the argument `trait="binomial"` is very important, because it tells which is the type of the trait.

The genomic control λ can be checked by use of `lambda` method:

```
> lambda(logReg)

$estimate
[1] 1.026291

$se
[1] 0.001006654
```

You can check 'top' association results by using several GenABEL-package functions, e.g.

```
> summary(logReg)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB  chi2.1df
rs1719133          1  4495479      +  T  A 124 -1.2311798 0.3509437 12.307449
rs4534929          1  4474374      +  C  G 123 -0.8412949 0.2707498  9.655179
rs1013473          1  4487262      +  T  A 124 -0.9002116 0.2902631  9.618461
rs3925525          2  6008501      +  C  G 124  1.0558097 0.3428824  9.481582
rs3224311          2  6009769      +  G  C 124  1.0558097 0.3428824  9.481582
rs2975760          3 10518480      +  A  T 123  1.0779360 0.3554682  9.195693
rs2521089          3 10487652      -  T  C 123  0.9037940 0.2993759  9.113920
rs1048031          1  4485591      +  G  T 122 -0.8788970 0.2927094  9.015764
rs7522488          3 11689797      -  A  G 123 -0.8303827 0.2813696  8.709680
rs8258863          1  4735725      +  T  A 123 -1.6257743 0.5715190  8.092061
                  P1df       Pc1df effAB effBB chi2.2df P2df
rs1719133 0.0004511543 0.000534248    NA    NA       NA   NA
rs4534929 0.0018881854 0.002160601    NA    NA       NA   NA
rs1013473 0.0019263105 0.002203179    NA    NA       NA   NA
rs3925525 0.0020754491 0.002369548    NA    NA       NA   NA
rs3224311 0.0020754491 0.002369548    NA    NA       NA   NA
rs2975760 0.0024258527 0.002759366    NA    NA       NA   NA
rs2521089 0.0025367167 0.002882416    NA    NA       NA   NA
rs1048031 0.0026766092 0.003037505    NA    NA       NA   NA
rs7522488 0.0031652482 0.003577799    NA    NA       NA   NA
rs8258863 0.0044459573 0.004985304    NA    NA       NA   NA
```

or

```
> descriptives.scan(logReg)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB  chi2.1df
rs1719133          1  4495479      +  T  A 124 -1.2311798 0.3509437 12.307449
rs4534929          1  4474374      +  C  G 123 -0.8412949 0.2707498  9.655179
rs1013473          1  4487262      +  T  A 124 -0.9002116 0.2902631  9.618461
rs3925525          2  6008501      +  C  G 124  1.0558097 0.3428824  9.481582
rs3224311          2  6009769      +  G  C 124  1.0558097 0.3428824  9.481582
rs2975760          3 10518480      +  A  T 123  1.0779360 0.3554682  9.195693
rs2521089          3 10487652      -  T  C 123  0.9037940 0.2993759  9.113920
rs1048031          1  4485591      +  G  T 122 -0.8788970 0.2927094  9.015764
rs7522488          3 11689797      -  A  G 123 -0.8303827 0.2813696  8.709680
rs8258863          1  4735725      +  T  A 123 -1.6257743 0.5715190  8.092061
                  P1df       Pc1df effAB effBB chi2.2df P2df
rs1719133 0.0004511543 0.000534248    NA    NA       NA   NA
rs4534929 0.0018881854 0.002160601    NA    NA       NA   NA
rs1013473 0.0019263105 0.002203179    NA    NA       NA   NA
rs3925525 0.0020754491 0.002369548    NA    NA       NA   NA
rs3224311 0.0020754491 0.002369548    NA    NA       NA   NA
rs2975760 0.0024258527 0.002759366    NA    NA       NA   NA
rs2521089 0.0025367167 0.002882416    NA    NA       NA   NA
rs1048031 0.0026766092 0.003037505    NA    NA       NA   NA
```

```
rs7522488 0.0031652482 0.003577799    NA    NA       NA   NA
rs8258863 0.0044459573 0.004985304    NA    NA       NA   NA
```

In this summary, SNP annotation is formation is provided. The estimate of the effect (regression coefficient) for the coded allele is given in column "effB", and its standard error is reported in column "se_effB". Note the estimate and the s.e. are reported on logOR scale! "P1df" provides p-value for association before Genomic Control adjustment, and "Pc1df" is GC-adjucted p-value.

Let us check the results for a specific SNP, say 'rs1013473'. To see analysis summary for this SNP, you can subset the 'logReg' analysis object on this SNP name:

```
> logReg["rs1013473",]
```

```
          Chromosome Position Strand A1 A2   N       effB   se_effB chi2.1df
rs1013473          1  4487262      +  T  A 124 -0.9002116 0.2902631 9.618461
                P1df       Pc1df effAB effBB chi2.2df P2df
rs1013473 0.00192631 0.002203179    NA    NA       NA   NA
```

From this output, let us compute the OR and its confidence interval. Let us first retrieve logOR and its standard error:

```
> logOR <- logReg["rs1013473","effB"]
> logOR

[1] -0.9002116

> selogOR <- logReg["rs1013473","se_effB"]
> selogOR

[1] 0.2902631
```

then, OR can be computed with

```
> exp(logOR)

[1] 0.4064836
```

and its confidence interval can be computed with

```
> int <- 1.96*selogOR
> # 95% lower
> exp(logOR-int)

[1] 0.2301262

> # 95% upper
> exp(logOR+int)

[1] 0.7179926
```

In the analysis, you can easily adjust for covariates, e.g.

```
> logRegAdj <- mlreg(dm2~sex+age,data=ge03d2ex.clean,trait="binomial")
> summary(logRegAdj)
```

```
Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB  chi2.1df
rs1719133          1  4495479      +  T  A 124 -1.3021408 0.3803625 11.719810
rs7522488          3 11689797      -  A  G 123 -0.9480035 0.2973642 10.163482
rs1037237          3 11690145      +  G  C 124 -0.8987985 0.2921128  9.467245
rs3925525          2  6008501      +  C  G 124  1.0963291 0.3583527  9.359671
rs3224311          2  6009769      +  G  C 124  1.0963291 0.3583527  9.359671
rs1013473          1  4487262      +  T  A 124 -0.8906400 0.2967849  9.005767
rs1048031          1  4485591      +  G  T 122 -0.8871609 0.3010797  8.682441
rs4534929          1  4474374      +  C  G 123 -0.8125882 0.2762030  8.655342
rs5009559          1  4490338      -  C  A 124 -0.8638342 0.3079870  7.866763
rs2975760          3 10518480      +  A  T 123  1.0272177 0.3670356  7.832650
                  P1df        Pc1df effAB effBB chi2.2df P2df
rs1719133 0.0006183828 0.0007940776    NA    NA       NA   NA
rs7522488 0.0014324983 0.0017830936    NA    NA       NA   NA
rs1037237 0.0020917331 0.0025675912    NA    NA       NA   NA
rs3925525 0.0022181201 0.0027168624    NA    NA       NA   NA
rs3224311 0.0022181201 0.0027168624    NA    NA       NA   NA
rs1013473 0.0026912896 0.0032730983    NA    NA       NA   NA
rs1048031 0.0032129038 0.0038821911    NA    NA       NA   NA
rs4534929 0.0032610390 0.0039382094    NA    NA       NA   NA
rs5009559 0.0050351700 0.0059851007    NA    NA       NA   NA
rs2975760 0.0051310824 0.0060949218    NA    NA       NA   NA
```

You can run GWA-analysis analysis under additive, dominant, recessive or over-dominant models by modifying 'gtmode' argument of `mlreg`; for example for dominant model:

```
> logRegDom <- mlreg(dm2~1,data=ge03d2ex.clean,trait="binomial",gtmode="dominant")
> summary(logRegDom)
```

```
Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N      effB   se_effB  chi2.1df
rs3925525          2  6008501      +  C  G 124  1.317301 0.3924782 11.265211
rs3224311          2  6009769      +  G  C 124  1.317301 0.3924782 11.265211
rs768924           2  6023393      +  C  T 124  1.269162 0.3880372 10.697626
rs2521089          3 10487652      -  T  C 123  1.221672 0.3939072  9.618816
rs7418878          1  2808520      +  A  T 124  1.252763 0.4053217  9.552961
rs1719133          1  4495479      +  T  A 124 -1.180004 0.3858503  9.352527
rs2975760          3 10518480      +  A  T 123  1.228665 0.4060375  9.156616
rs2939190          1    71843      -  A  G 124 -1.158564 0.3937231  8.658805
rs3479695          3 10481902      -  C  T 124  1.129865 0.3869327  8.526720
rs5767131          2  6008810      +  T  A 122 -1.160456 0.3985044  8.479904
                  P1df       Pc1df effAB effBB chi2.2df P2df
rs3925525 0.0007897318 0.001050955    NA    NA       NA   NA
rs3224311 0.0007897318 0.001050955    NA    NA       NA   NA
rs768924  0.0010727312 0.001408471    NA    NA       NA   NA
rs2521089 0.0019259389 0.002464745    NA    NA       NA   NA
rs7418878 0.0019962711 0.002550757    NA    NA       NA   NA
rs1719133 0.0022267819 0.002831767    NA    NA       NA   NA
rs2975760 0.0024782037 0.003136845    NA    NA       NA   NA
```

```
rs2939190 0.0032548460 0.004071370    NA    NA       NA   NA
rs3479695 0.0034996981 0.004363896    NA    NA       NA   NA
rs5767131 0.0035909104 0.004472641    NA    NA       NA   NA
```

For specific SNP, it may be easier to recode the SNP to dominant, recessive, etc. codings

```
> gtSNP <- as.character(ge03d2ex.clean[,"rs1013473"])
> addSNP <- as.numeric(ge03d2ex.clean[,"rs1013473"])
> table(gtSNP,addSNP)

     addSNP
gtSNP  0  1  2
  A/A  0  0 29
  T/A  0 63  0
  T/T 32  0  0

> domSNP <- 1*(addSNP>=1)
> table(gtSNP,domSNP)

     domSNP
gtSNP  0  1
  A/A  0 29
  T/A  0 63
  T/T 32  0

> recSNP <- 1*(addSNP>=2)
> table(gtSNP,recSNP)

     recSNP
gtSNP  0  1
  A/A  0 29
  T/A 63  0
  T/T 32  0
```

and to run analysis using standard `R` tools:

```
> # additive model
> summary(glm(dm2~addSNP,data=phdata(ge03d2ex.clean),family="binomial"))$coef

              Estimate Std. Error   z value     Pr(>|z|)
(Intercept)  1.4180283  0.3667238  3.866747 0.0001102967
addSNP      -0.9002116  0.2902630 -3.101366 0.0019263035

> # genotypic model
> summary(glm(dm2~gtSNP,data=phdata(ge03d2ex.clean),family="binomial"))$coef

              Estimate Std. Error   z value    Pr(>|z|)
(Intercept) -0.4924765  0.3827070 -1.286824 0.198155689
gtSNPT/A     1.1150061  0.4650908  2.397395 0.016512129
gtSNPT/T     1.7654422  0.5738655  3.076404 0.002095136

> # dominant model
> summary(glm(dm2~domSNP,data=phdata(ge03d2ex.clean),family="binomial"))$coef
```

```
             Estimate Std. Error   z value    Pr(>|z|)
(Intercept)  1.272966  0.4276177  2.976878 0.002912003
domSNP      -1.010601  0.4765372 -2.120719 0.033945472
```

```
> # recessive model
> summary(glm(dm2~recSNP,data=phdata(ge03d2ex.clean),family="binomial"))$coef
```

```
              Estimate Std. Error   z value     Pr(>|z|)
(Intercept)  0.8223589  0.2227874  3.691227 0.0002231749
recSNP      -1.3148354  0.4428305 -2.969162 0.0029861348
```

Note that Genomic Control is only defined for additive model, and you can not compute 'GC-adjusted' *p*-value for other models (even when it is technically there, as this happens for 'mlreg').

4.2 Analysis of quantitative traits

Analysis of quantitative trait is 'speciality' of `GenABEL-package`. One of the functions is `mlreg`, which can perform linear regression, and reports results of Wald test. As an alternative, you can use `qtscore` fuction, which performed the score test. The advantage of `qtscore` function is that it runs very fast.

For example, to run GWA analysis for quantitative phenotype 'bmi' you can use

```
> qtReg <- mlreg(bmi~1,data=ge03d2ex.clean)
> lambda(qtReg)

$estimate
[1] 1.064328

$se
[1] 0.0006330663

> summary(qtReg)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N      effB  se_effB  chi2.1df
rs2074814          1  2285234      -  A  T 121  6.188133 1.637544 14.280160
rs32422            1   912973      -  T  A 122  3.669797 1.030753 12.675777
rs2488552          1  3883660      -  T  A 122  5.468278 1.686834 10.508880
rs4093435          1  3255198      -  G  A 122 -3.586090 1.121514 10.224290
rs2498253          1  1190149      -  C  A 123  7.013532 2.204445 10.122205
rs3853863          1  3390077      +  A  C 121  5.188194 1.652234  9.860287
rs4237279          1   230545      -  A  C 123 -5.183293 1.651030  9.856024
rs6460293          1   231622      -  G  C 122 -5.184971 1.659992  9.756200
rs567752           1   894274      -  A  T 123  3.197896 1.034934  9.547806
rs798930           2  6773063      -  A  C 123 -5.432264 1.872909  8.412563
                  P1df        Pc1df effAB effBB chi2.2df P2df
rs2074814 0.0001575164 0.0002493460    NA    NA       NA   NA
rs32422   0.0003704232 0.0005584393    NA    NA       NA   NA
rs2488552 0.0011880221 0.0016765621    NA    NA       NA   NA
rs4093435 0.0013860308 0.0019390777    NA    NA       NA   NA
```

```
rs2498253 0.0014649365 0.0020430895    NA    NA       NA   NA
rs3853863 0.0016888472 0.0023366021    NA    NA       NA   NA
rs4237279 0.0016927657 0.0023417184    NA    NA       NA   NA
rs6460293 0.0017871898 0.0024648131    NA    NA       NA   NA
rs567752  0.0020018857 0.0027433854    NA    NA       NA   NA
rs798930  0.0037263698 0.0049322755    NA    NA       NA   NA
```

It is also very easy to adjust for covariates in the analysis, for example

```
> qtReg <- mlreg(bmi~sex+age,data=ge03d2ex.clean)
> lambda(qtReg)

$estimate
[1] 1.069655

$se
[1] 0.0006668126

> summary(qtReg)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N      effB  se_effB  chi2.1df
rs2074814          1  2285234      -  A  T 121  6.629482 1.667568 15.804909
rs32422            1   912973      -  T  A 122  3.653859 1.049780 12.114541
rs2498253          1  1190149      -  C  A 123  7.382429 2.231020 10.949436
rs2488552          1  3883660      -  T  A 122  5.425779 1.708009 10.091222
rs3853863          1  3390077      +  A  C 121  5.465176 1.733709  9.937017
rs4237279          1   230545      -  A  C 123 -5.322775 1.698238  9.823795
rs4093435          1  3255198      -  G  A 122 -3.532262 1.133483  9.711266
rs6460293          1   231622      -  G  C 122 -5.318015 1.707347  9.701873
rs567752           1   894274      -  A  T 123  3.155069 1.050310  9.023652
rs798930           2  6773063      -  A  C 123 -5.612975 1.893109  8.790929
                  P1df        Pc1df effAB effBB chi2.2df P2df
rs2074814 7.022008e-05 0.0001210850    NA    NA       NA   NA
rs32422   5.003014e-04 0.0007644346    NA    NA       NA   NA
rs2498253 9.363213e-04 0.0013769443    NA    NA       NA   NA
rs2488552 1.489773e-03 0.0021298826    NA    NA       NA   NA
rs3853863 1.619879e-03 0.0023041690    NA    NA       NA   NA
rs4237279 1.722685e-03 0.0024412902    NA    NA       NA   NA
rs4093435 1.831418e-03 0.0025857834    NA    NA       NA   NA
rs6460293 1.840803e-03 0.0025982304    NA    NA       NA   NA
rs567752  2.665083e-03 0.0036785578    NA    NA       NA   NA
rs798930  3.027320e-03 0.0041466042    NA    NA       NA   NA
```

4.3 Exercises

Ex. 27 — If you did not do so yet, start R, and load cleaned data set 'ge03d2ex.clean.RData'

Ex. 28 — Perform GWA logistic regression analysis for binary phenotype 'dm2'. Check results for the 'top' 10 SNPs,

Ex. 29 — What is the Genomic Control (GC) inflation factor λ for this analysis?

Ex. 30 — What is the unadjusted p-value (additive model) for SNP rs1013473?

Ex. 31 — What is OR and confidence interval for SNP rs1013473?

Ex. 32 — What is the GC-adjusted p-value for SNP rs1013473?

Ex. 33 — Does the association results for SNP rs1013473 improve if you assume a genotypic model, dominant or recessive model?

Ex. 34 — Perform GWA analysis for quantitative phenotype 'bmi'. What is GC inflaction factor? Which SNP demonstarted most significant association?

Ex. 35 — Perform GWA analysis for quantitative phenotype 'bmi', now adjusted by sex and age. Did association improve? Which SNP demonstarted most significant association?

5 Analysis of data with complex structure

In this section, we will perform analysis for quit complex stratification scenario – basically, the study data comprise of two genetically different populations; moreover, complex structure is present within each strata as well.

Note that there is no "exercise" at the end of this section – the idea is that you just follow the analysis stems of this manual.

We will use progressively more and more complex models to adjust for stratification; for each analysis, we will use Genomic Control λ to see whether we managed to account for the population stratification well.

We will start with characterization of the data, then will perform stratified analysis, analysis using EIGENSTRAT-based method and finally we will use Mixed Models approach.

If you did not do so yet, start R and load `GenABEL-package`.

```
> library(GenABEL)
```

```
GenABEL v. 1.6-8 (August 30, 2011) loaded

Installed GenABEL version (1.6-8) is not the same as stable
version available from CRAN (1.6-7). Unless used intentionally,
consider updating to the latest CRAN version. For that, use
'install.packages("GenABEL")', or ask your system administrator
to update the package.
```

Load the 'strdat' data set

```
> load("indata/strdat.RData")
> ls()
```

```
[1] "strdat"
```

Let us start analysis by exploring genetic structure of the population via principal axes of genetic variation. Compute genomic kinship with

```
> strdat.gkin <- ibs(strdat[,autosomal(strdat)],w="freq")
```

and derive first 25 PC's with

```
> strdat.mds <- cmdscale(as.dist(0.5-strdat.gkin),25)
```

Now you can plot the results with

```
> plot(strdat.mds[,1:2])
```

(see figure 3). You should see that the samples cluster in two separate groups.

We can compute 'population ID' of the samples with

```
> pop <- 1*(strdat.mds[,1] > 0.1)
> table(pop)
```

```
pop
  0   1
901  50
```

```
> plot(strdat.mds[,1:2])
```

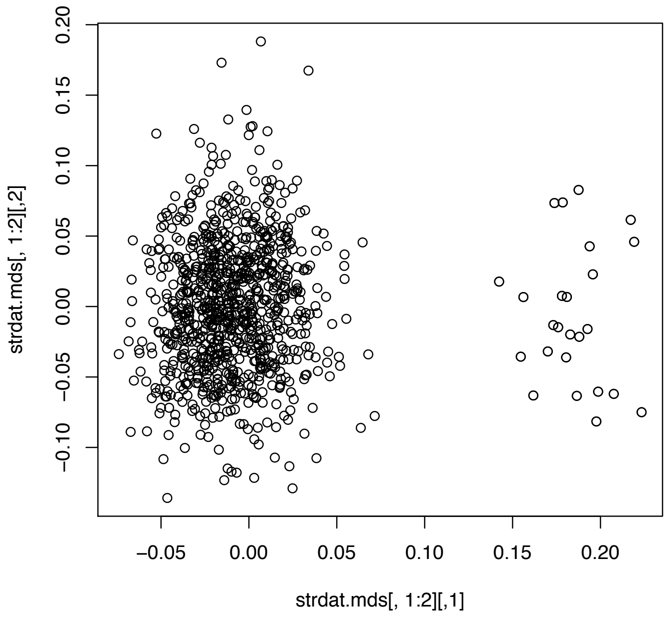

Figure 3: MDS plot for data set 'strdat'

Our phenotype of interest is called 'phe'. Let us check its distribution in two (sub)populations:

```
> attach(phdata(strdat))
> summary(phe)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  43.52   50.70   52.88   53.04   55.11   72.17 

> mean(phe[pop==0])

[1] 52.80205

> var(phe[pop==0])

[1] 10.75975

> mean(phe[pop==1])

[1] 57.32425

> var(phe[pop==1])
```

```
[1] 39.4564
```

You can see that both means and variances in the two groups under the study are larger different. We may expect this to have very big effect on GWA analysis. Let us try running "naive" analysis without accounting for stratification:

```
> rawAn <- qtscore(phe~sex,strdat)
> lambda(rawAn)$est

[1] 2.82229
```

As expected, GC parameter λ is unacceptably high ($\hat{\lambda} = 2.82$), reflecting the fact that this simple model does not reflect the nature of the data.

You can see that association signals, which looked very strong before correction ("P1df") are very far from being GW-significant after GC correction ("Pc1df"):

```
> summary(rawAn)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N      effB    se_effB chi2.1df
rs8639811          1  2724853      +  G  T 943 -1.638075 0.2032602 64.94757
rs323119           3 10347439      -  C  T 945  1.318832 0.1914853 47.43602
rs2364193          1  3538684      +  G  A 946 -1.302570 0.1904261 46.78961
rs2249310          3 11521365      -  A  C 943 -1.307156 0.1933467 45.70684
rs5923408          1  2297114      +  T  C 946 -1.255393 0.1900342 43.64110
rs944126           1   940023      -  G  C 942 -1.302676 0.2031478 41.11960
rs4173187          2  8558477      +  C  G 946  1.368888 0.2149212 40.56741
rs7447005          3 10651800      -  C  A 943  1.276797 0.2007980 40.43194
rs9846056          3 10641552      -  C  T 939 -1.263355 0.2004984 39.70355
rs2888230          1  4014795      +  C  A 943  1.370434 0.2181292 39.47193
                  P1df      effAB     effBB chi2.2df         P2df        Pc1df
rs8639811 7.691754e-16 -2.7726635 -4.040035 69.50677 8.068582e-16 1.609624e-06
rs323119  5.682870e-12  1.2948418  2.673110 47.45219 4.964629e-11 4.136647e-05
rs2364193 7.903149e-12 -2.8292861 -3.499626 57.70330 2.950453e-13 4.667478e-05
rs2249310 1.373423e-11 -2.4800664 -3.346847 51.73651 5.828545e-12 5.714625e-05
rs5923408 3.944661e-11 -2.2968406 -3.178867 48.71100 2.645702e-11 8.413561e-05
rs944126  1.431935e-10 -1.9509272 -3.011900 42.67956 5.398216e-10 1.350771e-04
rs4173187 1.899489e-10  0.9066500  3.591998 45.90025 1.078669e-10 1.498625e-04
rs7447005 2.035857e-10  0.3319721  4.083839 64.07812 1.217904e-14 1.537322e-04
rs9846056 2.955873e-10 -3.8448155 -4.128231 65.54511 5.848726e-15 1.763321e-04
rs2888230 3.328083e-10  0.7399297  3.999063 49.79393 1.539522e-11 1.841980e-04
```

5.1 Structured association and EIGENSTRAT analysis

Let us try structured association approach. You can do this by incorporating the information on (sub)populations in the analysis. You can do that by passing 'strata' argument to the `qtscore` analysis function:

```
> strAn <- qtscore(phe~sex,strdat,strata=pop)
> summary(strAn)
```

```
Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB chi2.1df
rs8163036          1  2084115      +  G  A 947  1.0182532 0.1562439 42.47228
rs1979355          1  3217158      +  C  G 941  1.3106454 0.2299676 32.48157
rs9468061          1  3217614      -  G  T 937 -1.3150132 0.2313452 32.31017
rs6017826          1  3212700      +  A  G 939  1.2698761 0.2251604 31.80817
rs8794226          1  3214917      +  A  G 942 -1.2654387 0.2260327 31.34295
rs6208881          1  2087593      -  T  A 941 -0.9524455 0.1898005 25.18173
rs3969268          1  3234199      +  G  A 946 -0.7676873 0.1593786 23.20111
rs5305978          1  2087292      -  A  G 939 -0.9187667 0.1909492 23.15127
rs3747315          1  2845715      +  C  A 942  0.7626447 0.1599135 22.74438
rs7575802          1  3488580      +  C  T 936  0.7807330 0.1666512 21.94765
                  P1df      effAB     effBB chi2.2df         P2df        Pc1df
rs8163036 7.169121e-11  0.1681250  0.726749 43.68567 3.264203e-10 3.972576e-06
rs1979355 1.203283e-08  1.5907955  2.044152 34.48518 3.248164e-08 5.484489e-05
rs9468061 1.314240e-08 -0.4616762 -2.055356 34.27506 3.607987e-08 5.738796e-05
rs6017826 1.701750e-08  1.5298068  1.926526 33.93608 4.274386e-08 6.553915e-05
rs8794226 2.162430e-08 -0.3981305 -1.919563 33.43977 5.478303e-08 7.413043e-05
rs6208881 5.217418e-07 -0.7202472 -2.842930 29.73257 3.496662e-07 3.825194e-04
rs3969268 1.459129e-06 -0.5518248 -1.710938 25.43263 3.001756e-06 6.512575e-04
rs5305978 1.497443e-06 -0.7245578 -2.665507 26.39405 1.856119e-06 6.600603e-04
rs3747315 1.850450e-06  0.4911269  1.664671 25.50910 2.889150e-06 7.365697e-04
rs7575802 2.801897e-06  1.0178122  1.617192 21.95004 1.712414e-05 9.133460e-04
```

```
> lambda(strAn)$est
```

```
[1] 1.996062
```

Indeed, incorporation of genetic strata information does help a bit – you can see that $\hat{\lambda}$ is more close to 1 now. However, it is still unacceptably high, most likely indicating the fact that (at least some of the) studied sub-populations may have complex structure as well. We can check this by running strata-specific analyses:

```
> strAn1 <- qtscore(phe~sex,strdat[pop==0,])
> strAn2 <- qtscore(phe~sex,strdat[pop==1,])
```

Indeed, for both populations there is much residual inflation of the test statistics:

```
> lambda(strAn1)$est
```

```
[1] 2.151395
```

```
> lambda(strAn2)$est
```

```
[1] 1.05206
```

Thus, structured association approach does not solve the problem of this data set analysis well. Let us try EIGENSTRAT type of analysis. EIGENSTRAT analysis is done by `egscore` function; however, let us try to simply include Principal Components of genetic variation into 'mlreg' analysis. Let us start by including only the first PC:

```
> strdat.mlPCA <- mlreg(phe~sex+strdat.mds[,1],data=strdat)
> lambda(strdat.mlPCA)$est

[1] 2.023588
```

You can see that including first PC helps, but the residual inflation is still high – 2.02. This is very similar to what happened in stratified analysis, and rightly so as the first PC separates the two sub-populations rather well. Let us try to include more components and see what happens:

```
> strdat.mlPCA2 <- mlreg(phe~sex+strdat.mds[,1:2],data=strdat)
> lambda(strdat.mlPCA2)$est

[1] 2.042886

> strdat.mlPCA3 <- mlreg(phe~sex+strdat.mds[,1:3],data=strdat)
> lambda(strdat.mlPCA3)$est

[1] 1.987247

> strdat.mlPCA4 <- mlreg(phe~sex+strdat.mds[,1:4],data=strdat)
> lambda(strdat.mlPCA4)$est

[1] 2.014081

> strdat.mlPCA5 <- mlreg(phe~sex+strdat.mds[,1:5],data=strdat)
> lambda(strdat.mlPCA5)$est

[1] 2.023585
```

You can see that λ is going down (but some times up); let us check what happens when we include 25 PCs:

```
> strdat.mlPCA25 <- mlreg(phe~sex+strdat.mds[,1:25],data=strdat)
> lambda(strdat.mlPCA25)$est

[1] 1.838252

> summary(strdat.mlPCA25)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB chi2.1df
rs8639811          1  2724853      +  G  T 943 -1.1059081 0.2338299 22.36855
rs6017826          1  3212700      +  A  G 939  1.2226395 0.2655063 21.20543
rs8163036          1  2084115      +  G  A 947  0.8923128 0.1939494 21.16690
rs2845             1  4948934      +  G  A 940  1.3120897 0.2909869 20.33200
rs2937697          1  4037688      +  C  G 938  1.0753556 0.2397252 20.12227
rs1979355          1  3217158      +  C  G 941  1.2250209 0.2737476 20.02563
rs2257761          1  4047695      +  A  C 941  1.0733166 0.2409296 19.84612
rs8794226          1  3214917      +  A  G 942 -1.1854487 0.2668814 19.73008
rs2360642          1  4950033      +  A  T 948  1.3850504 0.3126355 19.62703
rs9468061          1  3217614      -  G  T 937 -1.2100912 0.2758290 19.24671
                  P1df        Pc1df effAB effBB chi2.2df P2df
```

```
rs8639811 2.250292e-06 0.0004860659    NA    NA       NA   NA
rs6017826 4.125933e-06 0.0006827428    NA    NA       NA   NA
rs8163036 4.209731e-06 0.0006904853    NA    NA       NA   NA
rs2845    6.510249e-06 0.0008818602    NA    NA       NA   NA
rs2937697 7.264591e-06 0.0009378504    NA    NA       NA   NA
rs1979355 7.641103e-06 0.0009648447    NA    NA       NA   NA
rs2257761 8.393305e-06 0.0010170937    NA    NA       NA   NA
rs8794226 8.918641e-06 0.0010523804    NA    NA       NA   NA
rs2360642 9.412797e-06 0.0010847548    NA    NA       NA   NA
rs9468061 1.148682e-05 0.0012132136    NA    NA       NA   NA

$estimate
[1] 1.946993

$se
[1] 0.001198608

$iz0
[1] 1.744408

$iz2
[1] 1.398155
```

Thus, including many PC of genetic variation does help in dealing with stratification. However, for this data set, the correction achieved by EIGENSTRAT is yet very far from being good. Let us try modeling stratification by use of Mixed Model.

5.2 Mixed Models

To run mixed model for GWAS, you first need to estimate "heritability" of the trait in your data. This can be done with function `polygenic`:

```
> strdat.h2 <- polygenic(phe~sex,data=strdat,kin=strdat.gkin,quiet=TRUE)
```

and resulting heritability estimate can be viewed with

```
> strdat.h2$esth2

[1] 0.3903
```

Now, you can run mixed model GWAS using `mmscore` function of `GenABEL-package`:

```
> strdat.mms <- mmscore(strdat.h2,data=strdat)
> lambda(strdat.mms)$est

[1] 1.071904

> summary(strdat.mms)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB chi2.1df
rs8639811          1  2724853      +  G  T 943 -1.2615190 0.2515128 25.15749
```

```
rs8163036          1  2084115      +  G  A 947  1.0216832 0.2254907 20.52936
rs6208881          1  2087593      -  T  A 941 -1.0002695 0.2439192 16.81675
rs1776815          1   802671      -  G  A 940  0.9384957 0.2395535 15.34827
rs5305978          1  2087292      -  A  G 939 -0.9549579 0.2441401 15.29996
rs6017826          1  3212700      +  A  G 939  1.1002099 0.2988692 13.55154
rs323119           3 10347439      -  C  T 945  0.9237922 0.2549621 13.12796
rs1979355          1  3217158      +  C  G 941  1.0964234 0.3038423 13.02147
rs5923408          1  2297114      +  T  C 946 -0.8353134 0.2329509 12.85790
rs8794226          1  3214917      +  A  G 942 -1.0724361 0.2993597 12.83383
                  P1df        Pc1df effAB effBB chi2.2df P2df
rs8639811 5.283409e-07 1.268833e-06    NA    NA        0   NA
rs8163036 5.872367e-06 1.206964e-05    NA    NA        0   NA
rs6208881 4.116829e-05 7.467016e-05    NA    NA        0   NA
rs1776815 8.940266e-05 1.543245e-04    NA    NA        0   NA
rs5305978 9.171865e-05 1.580645e-04    NA    NA        0   NA
rs6017826 2.321006e-04 3.770768e-04    NA    NA        0   NA
rs323119  2.909214e-04 4.659257e-04    NA    NA        0   NA
rs1979355 3.079398e-04 4.914108e-04    NA    NA        0   NA
rs5923408 3.360577e-04 5.333261e-04    NA    NA        0   NA
rs8794226 3.404087e-04 5.397920e-04    NA    NA        0   NA
```

Note the results improved a lot! However, the $\hat{\lambda}$ of 1.07 is still high (over the threshold of 1.05). We can do better.

5.3 Mixed Models + Structured Association

In a case of presence of distinct sub-populations it is very important to incorporate both "strata" and "relationship" information in the model. We can tell `mmscore` about presence of strong strata by using 'strata' argument:

```
> strdat.mmsS <- mmscore(strdat.h2,data=strdat,strata=pop)
> summary(strdat.mmsS)

Summary for top 10 results, sorted by P1df
          Chromosome Position Strand A1 A2   N       effB   se_effB  chi2.1df
rs8163036          1  2084115      +  G  A 947  1.3509690 0.2329565 33.631095
rs6208881          1  2087593      -  T  A 941 -0.9926233 0.2439192 16.560639
rs5305978          1  2087292      -  A  G 939 -0.9488243 0.2441361 15.104540
rs8639811          1  2724853      +  G  T 943 -1.0142714 0.2660732 14.531351
rs6017826          1  3212700      +  A  G 939  1.0735002 0.2989172 12.897406
rs8794226          1  3214917      +  A  G 942 -1.0423131 0.2994435 12.116207
rs1979355          1  3217158      +  C  G 941  1.0550271 0.3039958 12.044592
rs9468061          1  3217614      -  G  T 937 -1.0395314 0.3036896 11.716972
rs2179199          3 10585518      +  T  G 943 -0.7847959 0.2426268 10.462508
rs3747315          1  2845715      +  C  A 942  0.6495932 0.2062080  9.923664
                  P1df        Pc1df effAB effBB chi2.2df P2df
rs8163036 6.661921e-09 6.661921e-09    NA    NA        0   NA
rs6208881 4.711889e-05 4.711889e-05    NA    NA        0   NA
rs5305978 1.017184e-04 1.017184e-04    NA    NA        0   NA
rs8639811 1.378463e-04 1.378463e-04    NA    NA        0   NA
rs6017826 3.290378e-04 3.290378e-04    NA    NA        0   NA
```

```
rs8794226 4.998546e-04 4.998546e-04    NA    NA        0   NA
rs1979355 5.194286e-04 5.194286e-04    NA    NA        0   NA
rs9468061 6.193264e-04 6.193264e-04    NA    NA        0   NA
rs2179199 1.218218e-03 1.218218e-03    NA    NA        0   NA
rs3747315 1.631673e-03 1.631673e-03    NA    NA        0   NA
```

Let us check what is residual test inflation with:

```
> lambda(strdat.mmsS)$est
```

```
[1] 1
```

The result of 'exact' 1 means that actual $\hat{\lambda}$ was ≤ 1. When this happens, the $\hat{\lambda}$ is "forced" to be 1, because vales < 1 do not have sense (unless you run 'grammar' type of analysis or your study is severely under-powered). You still can get 'observed' $\hat{\lambda}$ with

```
> estlambda(strdat.mmsS[,"P1df"],plot=F)
```

```
$estimate
[1] 0.99998

$se
[1] 0.001555914
```

Note that now we have achieved almost perfect correction – $\hat{\lambda} = 1$! Also of note, we have obtained genome-wide significant association with nominal p-value of 6.7e-09 ($\leq 5 \cdot 10^{-8}$).

However, if sub-populations are very distinct, the characteristics of the trait may be quite different between them – as you have seen, the mean and the variances are different; also heritability may be different. In such situation, preferred type of analysis would be Mixed Model analysis in each population separately, and then combining the results through meta-analysis. Let us do this.

First, split the data set into two populations

```
> dat0 <- strdat[pop==0,]
> dat1 <- strdat[pop==1,]
```

filter the smaller data set on MAF

```
> dat1.smr <- summary(gtdata(dat1))
> maf <- pmin(dat1.smr$Q.2,1-dat1.smr$Q.2)
> dat1 <- dat1[,which(maf>0.05)]
```

compute genomic kinship

```
> dat0.gkin <- ibs(dat0[,autosomal(dat0)],w="freq")
> dat1.gkin <- ibs(dat1[,autosomal(dat1)],w="freq")
```

run polygenic model estimation

```
> dat0.h2 <- polygenic(phe~sex,data=dat0,kin=dat0.gkin,quiet=TRUE)
> dat0.h2$esth2
```

```
[1] 0.5303567
```

```
> dat1.h2 <- polygenic(phe~sex,data=dat1,kin=dat1.gkin,quiet=TRUE)
> dat1.h2$esth2
```

```
[1] 0.1729434
```

and perform MM-analysis

```
> dat0.mms <- mmscore(dat0.h2,dat0)
> estlambda(dat0.mms[,"P1df"],plot=F)$est
```

```
[1] 1.011224
```

```
> dat1.mms <- mmscore(dat1.h2,dat1)
> estlambda(dat1.mms[,"P1df"],plot=F)$est
```

```
[1] 0.9300606
```

You can see that $\hat{\lambda}$ is < 1 for the smaller data set, which reflects the fact that it is quite under-powered. Moreover, if you estimate the λ for the smaller population with median-method, you will see different result:

```
> median(dat1.mms[,"chi2.1df"])/qchisq(.5,1)
```

```
[1] 1.053439
```

and also the Q-Q plot (figure 4) looks strange. Basically, this is connected to very small size of this group.

Now, let us perform meta-analysis of obtained results. First, let us introduce a convenience function, which re-formats the `mmscore` results into format accepted by `MetABEL-package`:

```
> reformat4MetA <- function(mmsout,summa) {
+  table <- results(mmsout)[,c("A1","A2","N","effB","se_effB")]
+  table$effallele <- table$A2
+  table$name <- snpnames(mmsout)
+  table$pexhwe <- summa[,"Pexact"]
+  table$call <- summa[,"CallRate"]
+  table$effallelefreq <- summa[,"Q.2"]
+  names(table)[1:5] <- c("allele1", "allele2","n","beta", "sebeta")
+  for (i in 1:dim(table)[2]) {if (is(table[,i],"factor"))
+    { table[,i] <- as.character(table[,i]) }}
+  return(table)
+ }
```

Obtain data summaries, and tables usable for `MetABEL-package` with

```
> dat0.sum <- summary(gtdata(dat0))
> table0 <- reformat4MetA(dat0.mms,dat0.sum)
> table0[1:5,]
```

```
$estimate
[1] 0.9300606

$se
[1] 0.001036026
```

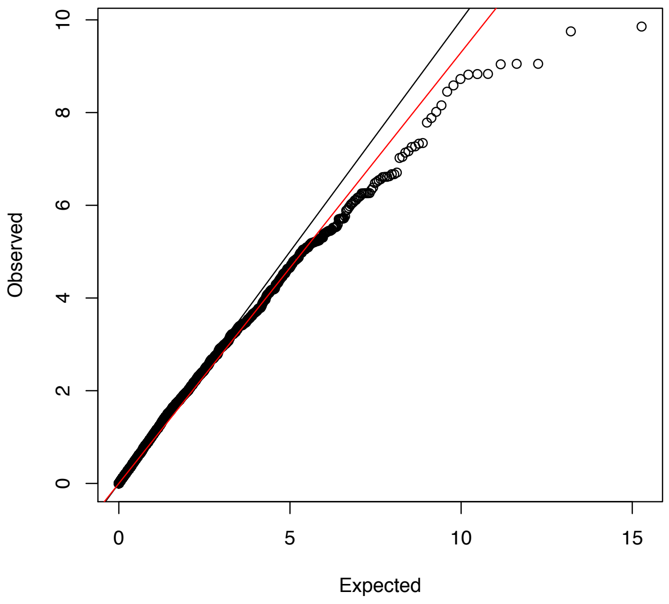

Figure 4: Q-Q plot for analysis of smaller sub-group

```
          allele1 allele2   n        beta    sebeta effallele      name
rs1646456       C       G 889 -0.22270119 0.1858921         G rs1646456
rs7950586       T       A 889  0.60728720 0.4302691         A rs7950586
rs4785242       T       C 889 -0.19141569 0.1753226         C rs4785242
rs4435802       C       A 894 -0.15557777 0.2978518         A rs4435802
rs2847446       T       A 890  0.01931545 0.1823080         A rs2847446
              pexhwe      call effallelefreq
rs1646456 0.87703309 0.9866815    0.31777278
rs7950586 0.08586309 0.9866815    0.03487064
rs4785242 0.11513776 0.9866815    0.62542182
rs4435802 0.51240154 0.9922309    0.08445190
rs2847446 0.08785896 0.9877913    0.31460674
```

```
> dat1.sum <- summary(gtdata(dat1))
> table1 <- reformat4MetA(dat1.mms,dat1.sum)
```

At this point you can run meta-analysis with

```
> library(MetABEL)
> mmsSum <- metagwa.tables(table0,table1,name.x = "dat0", name.y = "dat1")

analysing ...
Lambda dat0 = 0.9974639
Lambda dat1 = 1.053439
Corrected Lambda dat0 = 0.9974639
Corrected Lambda dat1 = 1
Lambda POOLED data = 0.9909414
... DONE

> mmsSum[1:5,]

       name allele1 allele2 effallele   n npops        beta    sebeta
1 rs1000909       A       G         G 943     2  0.16196616 0.2318791
2 rs1006092       T       G         G 941     2 -0.07014899 0.1060194
3  rs100616       G       C         C 935     2 -0.06101721 0.3127620
4 rs1006497       T       G         G 946     2  0.15798942 0.2749471
5 rs1010481       A       C         C 939     2 -0.03546104 0.1984897
  effallelefreq      call    pexhwe   obetadat0   obetadat1   osedat0  osedat1
1     0.8080594 0.9915918 0.4321928  0.12653327  1.64373962 0.2346351 1.517332
2     0.5201913 0.9894910 1.0206416 -0.06287613 -0.95080794 0.1064563 1.171447
3     0.1251337 0.9831916 0.0000000 -0.10121262  2.84623264 0.3149167 2.678235
4     0.1960888 0.9947439 0.4583876  0.16126674 -0.02842770 0.2773534 2.091785
5     0.3035144 0.9873907 1.6155995 -0.03575831 -0.02061136 0.2004666 1.416847
        chi2         p
1 0.48789402 0.4848682
2 0.43779614 0.5081881
3 0.03806068 0.8453213
4 0.33018530 0.5655500
5 0.03191736 0.8582091
```

And check 'top' results with

```
> mmsSum[order(mmsSum$p)[1:10],]

          name allele1 allele2 effallele   n npops       beta    sebeta
5604 rs8163036       A       G         G 947     2 -1.2368900 0.2285473
4077 rs6208881       A       T         T 941     2  1.0456263 0.2332909
3384 rs5305978       A       G         G 939     2 -0.9872184 0.2330870
5967 rs8639811       T       G         G 943     2  0.9451653 0.2574303
3930 rs6017826       A       G         G 939     2  1.0451889 0.2896198
729  rs1979355       G       C         C 941     2 -1.0453994 0.2938206
6081 rs8794226       A       G         G 942     2 -1.0280332 0.2899739
2110 rs3747315       A       C         C 942     2 -0.6983540 0.1979743
6632 rs9468061       T       G         G 937     2  1.0273107 0.2934586
3045 rs4897240       A       T         T 891     1  1.1700415 0.3560837
     effallelefreq      call      pexhwe  obetadat0   obetadat1   osedat0
5604    0.30570222 0.9957949  4.02877476 -1.2089038 -3.37610243 0.2300374
4077    0.74920298 0.9894910  9.51751830  1.0663728 -0.35127464 0.2350170
3384    0.24920128 0.9873907 10.01814544 -1.0070621  0.35127464 0.2348085
```

```
5967     0.18716861 0.9915918  2.25405022  0.9122647  2.80835731 0.2596932
3930     0.15388711 0.9874238  0.01714552  1.0763592 -0.05926228 0.2936782
729      0.85706695 0.9894910  0.01950032 -1.0856208  0.30947015 0.2981499
6081     0.84766454 0.9905413  0.01724178 -1.0666764  0.30947015 0.2941331
2110     0.59076433 0.9905413  0.77228313 -0.7357617  1.30234678 0.1998165
6632     0.14354322 0.9852909  0.01941322  1.0668945 -0.30947015 0.2977717
3045     0.06453423 0.9889012  0.07741391  1.1700415          NA 0.3560837
      osedat1     chi2            p
5604 2.011191 29.28936 6.233701e-08
4077 1.928457 20.08895 7.392251e-06
3384 1.928457 17.93867 2.281388e-05
5967 1.954283 13.48020 2.410941e-04
3930 1.748134 13.02367 3.075792e-04
729  1.730433 12.65902 3.737579e-04
6081 1.730433 12.56887 3.922248e-04
2110 1.461309 12.44325 4.195050e-04
6632 1.730433 12.25491 4.640362e-04
3045       NA 10.79688 1.016711e-03
```

You can see that with this analysis we are 'close' to GW-significance; however, the results are 'worse' compared to previous analysis. Normally, this should not be the case – the problem here is that the smaller data set ('dat1') is very small, and does not provide enough information for polygenic model estimation and GWA analysis when analyzed separately.