Polygenic Risk Score Analyses Workshop 2022

Day 4: Introduction to Cross-Ancestry PRS: descriptive analyses using the 1000 Genomes dataset

University of Mauritius

Contents

1 Descriptive analyses using 1000 Genomes data

Key Learning Outcomes

After completing this practical, you should:

1. Be more comfortable with handling, exploring and performing basic descriptive analyses using 1000 genomes data.

2. Be aware of the genetic distinctions that distinction the 1000 genomes superpopulations.

3. Be aware of the reasons why these continental differences exist and understand their impact on cross-ancestry PRS prediction.

1.1 Introduction to the multi-ancestry 1000Genomes dataset

Today we will be working with the multi-ancestry 1000Genomes dataset. These data are the product of a whole-genome sequencing initiative completed in 2013, which includes individuals from 26 different source populations around the world. For simplicity these divergent populations have been collapsed into 5 continental *super-populations*. The scripts used to download and process the 1000Genomes data for the purposes of this course can be found in the appendix at the end of today's (Day 4) booklet. The cleaned Plink binary files should be located inside your home directory

1.2 Sample sizes

First of all we would like to know the number of individuals within each superpopulation. Type the following command to query the number of European ancestry individuals in the downloaded dataset

```
grep -F 'EUR' all_phase3.king.psam | wc -l
```

Repeat the same command for East Asian, African, South Asian and Amerindian *superpopulations*, by inserting the relevant ancestry codes (EAS, AFR, SAS, AMR).

Q How many individuals are there overall?

1.3 Number of genetic variants

We do not need to use the full genome-wide data for this tutorial, only a small fraction of the 80 million total available variants. This provides a reliable approximation for the genomic analyses in this tutorial and importantly, reduces the computation time required to complete the tutorial. The following command derives the number of genetic variants on chromosomes 1 to chromsome 22 by counting the number of lines in the relevant (.bim) file, which contains a single variant per line.

```
wc chr1-22.bim -l
```

1.4 How many SNPs are contained in this dataset?

To quantify the number of single nucleotide polymorphisms (SNPs) we can ask plink to write a list of SNPs

```
./plink --bfile chr1-22 --snps-only --write-snplist
```

See the output file plink.snplist, which contains a list of all the SNPs in the dataset

How can the number of SNPs in the file be determined? Hint: we can count the number of lines in the above output file

1.5 Quantification of SNPs with minor allele frequencies greater than zero

The rate at which a genetic variant occurs in a population is also known as its allelic frequency. Allele frequencies are shaped by evolutionary forces over a long period of time and hence can vary. This has implications for PRS research as the allelic frequency distribution of a disease or trait may vary between populations. It is possible to generate allele frequency statistics for each SNP in a given population, using the population information in the file pop_info.pheno.

```
./plink --bfile chr1-22 --snps-only --freq --within pop\_info.pheno
```

Population-stratified allele frequency results can be found in the output file **plink.frq.strat**. For each population, print the numbers of total SNPs to screen, as follows:

```
grep -F 'AFR' plink.frq.strat | wc -l
```

Compare the totals against number of SNPs which have minor allele frequencies greater than 0 (and hence are useful for statistical analysis). Do this for all 5 populations (EAS, EUR, SAS, EUR and AFR), using the code given below:

```
grep -F 'AFR' plink.frq.strat > freq_report.afr
grep -F 'AMR' plink.frq.strat > freq_report.amr
grep -F 'EUR' plink.frq.strat > freq_report.eur
grep -F 'EAS' plink.frq.strat > freq_report.eas
grep -F 'SAS' plink.frq.strat > freq_report.sas
grep -F 'AFR' plink.frq.strat | awk '$6 >0' freq_report.afr | wc -l
grep -F 'EUR' plink.frq.strat | awk '$6 >0' freq_report.eur | wc -l
grep -F 'EAS' plink.frq.strat | awk '$6 >0' freq_report.eas | wc -l
grep -F 'AMR' plink.frq.strat | awk '$6 >0' freq_report.amr | wc -l
grep -F 'SAS' plink.frq.strat | awk '$6 >0' freq_report.sas | wc -l
```

Having compared the number of SNPs that show variation in each population:

(i) Which populations have the largest number (density) of SNP sites that can actually be considered polymorphic?
(ii) What do you think is the significance of the observed population order?

1.6 Investigating missingness

Genotype missingness, caused by genotyping failure can potentially lead to biased allele frequency estimation. Therefore missingness needs to be excluded as a possible source of bias when calculating allele frequency differences.

```
./plink --bfile chr1-22 --missing --within pop_info.pheno
```

The output file **plink.lmiss** provides a variant-based missing data report). Use the following code to query the number of genotyping failures based on the missingness information in the **NMISS** column:

```
awk '$4 > 0' plink.lmiss | wc -l
```

Can genotyping failure explain large population differences in the numbers of SNPs which can be analysed?

1.7 Cross-population allele frequency comparisons

Here we compare profiles of allele frequency across the five ancestral populations. To do this we will use the previously-generated output on minor allele frequencies per ancestry group (the file "plink.frq.strat").
In R:

```
library(dplyr)
library(ggplot2)
freq <-read.table("plink.frq.strat", header =T)
plotDat <- freq %>%
  mutate(AlleleFrequency = cut(MAF, seq(0, 1, 0.25))) %>%
  group_by(AlleleFrequency, CLST) %>%
  summarise(FractionOfSNPs = n()/nrow(freq) * 100)

```

```
ggplot(na.omit(plotDat),
       aes(AlleleFrequency, FractionOfSNPs, group = CLST, col = CLST)) +
  geom_line() +
  scale_y_continuous(limits = c(0, 12)) +
  ggtitle("Distribution of allele frequency across genome")
```

How are the allele frequencies in AFR distinguishable from the other global reference groups?

1.8 Calculation of Fst

Fst is a formal metric which is used to convey the level of genetic divergence between populations, using information derived from a set of genome-wide and mutually independent SNPs. *Fst* is estimated efficiently in **Plink-2**, which calculates *Fst* for all pairwise ancestry combinations, (the same application in Plink-1.9 has more limited capabilities).
Use the following command to calculate Fst:

```
./plink2 --bfile chr1-22 --fst POP --pheno pop_info.pheno
```

Check the output file **plink2.fst.summary**

(For which populations pairs is Fst greatest?
For which populations is Fst smallest?
What factors contributed to the genetic divergence between human populations that exists today?

2 Comparing LD structure across populations

Linkage disequilibrium (or LD) is used to define regions of the genome at which the correlation in allelic variability, spanning two or more SNPs, exceeds a threshold value. It is an important consideration in GWAS given that it provides an important means of capturing genetic variation linked to disease remotely, (i.e. even when the causal variants in question are not directly typed by the genotyping array used). As well as being one of the factors that determines whether a causal variants can be reliably detected in a given population at GWAS, structural differences in LD between populations also (in part) determine whether the derived PRSs will be portable between ancestrally-diverse populations. Here we take you through some basic approaches used to investigate cross-population LD profiles using 1000Genomes dataset.

2.1 Linkage disequilibrium versus cross-population genomic distance

We will now perform pairwise LD comparisons between genome-wide snps in order to show cross-populations relationships between genomic distance and LD strength. We derive information on pairwise R2 between all SNPs:

```
./plink --bfile chr1-22 \
--keep-cluster-names AFR \
--within pop_info.pheno \
--r2 \
--ld-window-r2 0 \
--ld-window 999999 \
--ld-window-kb 2500 \
--threads 30 \
--out chr1-22.AFR
```

Repeat this step for all five 1000Genomes populations. Output files containing LD info for all pairwise SNPs, have a *'.ld'* suffix Next, create a summary file containing the base-pair distance between each pair and the corresponding r^2 value. The following example shows this for **AFR** and **EUR** populations only, as just these populations will be used in the plot.

```
cat chr1-22.AFR.ld | sed 1,1d | awk -F " " 'function abs(v) {return v < 0 ? -v : \
    v}BEGIN{OFS="\t"}{print abs($5-$2),$7}' | sort -k1,1n > \
    chr1-22.AFR.ld.summary
cat chr1-22.EUR.ld | sed 1,1d | awk -F " " 'function abs(v) {return v < 0 ? -v : \
    v}BEGIN{OFS="\t"}{print abs($5-$2),$7}' | sort -k1,1n > \
    chr1-22.EUR.ld.summary
```

2.2 LD decay versus chromosomal distance

In order to visualise LD behaviour as a function of chromosomal distance we need to add additional functionality to be able to carry out the necessary data transformation (*dplyr*) and manipulation of character strings (*stringr*)

```
R
```

```
install.packages("dplyr")
install.packages("stringr")
install.packages("ggplot2")
library(dplyr)
library(stringr)
library(ggplot2)
```

The next piece of code does the following **4 steps**:

1. Loads the previously-generated information on pairwise LD.
2. Categorises distances into intervals of fixed length (100 Kb).
3. Computes mean and median r^2 within blocks.
4. Obtains mid-points for each distance interval.

```
dfr<-read.delim("chr1-22.AFR.ld.summary",sep="",header=F,check.names=F,\newline \
    stringsAsFactors=F)\newline
colnames(dfr)<-c("dist","rsq")
dfr$distc<-cut(dfr$dist,breaks=seq(from=min(dfr$dist)-1,to=max(dfr$dist)+1,by=100000))
dfr1<-dfr %>% group_by(distc) %>% \
    summarise(mean=mean(rsq),median=median(rsq))
dfr1 <- dfr1 %>% \
    mutate(start=as.integer(str_extract(str_replace_all(distc,"[\\(\\)\\[\\]]",""),"^[0-9-e+.]+")),
                \
    end=as.integer(str_extract(str_replace_all(distc,"[\\(\\)\\[\\]]",""),"[0-9-e+.]+$")),
                mid=start+((end-start)/2))
```

Steps 1-4 should be repeated for the file `chr1-22._EUR.ld.summary`.
The output object *dfr1* on lines 4 and 5 should be renamed *dfr2* to prevent the object *df1* being over-written. Finally, we plot LD decay for **AFR** and **EUR** reference populations in a single graph:

```
 ggplot()+
 geom_point(data=dfr1,aes(x=start,y=mean),size=0.4,colour="grey20")+
 geom_line(data=dfr1,aes(x=start,y=mean),size=0.3,alpha=0.5,colour="grey40")+
 labs(x="Distance (Megabases)",y=expression(LD~(r^{2})))+
 \
    scale_x_continuous(breaks=c(0,2*10^6,4*10^6,6*10^6,8*10^6),labels=c("0","2","4","6","8"))+
 theme_bw()
```

(i) What differences do you observe in terms of LD decay between AFR and EUR genomes?
(ii) How is this likely to impact the transferability of PRS performance between the two populations?

2.3 Distribution of LD-block length

The next set of scripts will allow us to visualise the distribution of LD block length across the different 1000Genomes populations.

```
./plink --bfile chr1-22 --keep-cluster-names AFR --blocks no-pheno-req \
      no-small-max-span --blocks-max-kb 250 --within pop_info.pheno  --threads 30 \
      --out AFR
```

The "–block" flag estimates haplotype blocks using the same block definition implemented by the software Haploview. The default setting for the flag *--blocks-max-kb* only considers pairs of variants that are within **200 kilobases** of each other. The output files from the above command is a ***.blocks*** file. Use the same code to generate output for *EUR, EAS, SAS* and *AMR* populations (as it is not possible to generate population-specific information using the --within flag).
Then, in R:

```
R
dfr.afr <- \
      read.delim("AFR.blocks.det",sep="",header=T,check.names=F,stringsAsFactors=F)
colnames(dfr.afr) <- tolower(colnames(dfr.afr))
```

Load each of the 5 datasets and set column names to lower case. Now plot the data

```
plot (density(dfr.afr$kb), main="LD block length distribution", \
      ylab="Density",xlab="LD block length (Kb)" )
lines (density(dfr.eur$kb), col="blue")
lines (density(dfr.eas$kb), col="red")
lines (density(dfr.amr$kb), col="purple")
lines (density(dfr.sas$kb), col="green")
legend("topright",c("AFR","EAS","EUR","SAS","AMR"), \
      fill=c("black","red","blue","green","purple"))
```

(i) What are the main features of this plot
(ii) How would you interpret them?

Principle Component Analysis
In the case where you have genetic data for N individuals whose genetic ancestry status is unknown possible strategies that can narrow the bounds of this uncertainty include Principle Components Analysis **(PCA)**; a visual inquiry that can be used to determine the spatial distance between genetic samples in question and reference populations, of which 1000 genomes is one. Each sample is projected onto principle components that reflect different linear combinations of conditionally independent SNPs). The steps involved are as follows.

```
./plink --bfile chr1-22 --indep-pairwise 250 25 0.1 --maf 0.1 --threads 30 --out \
      chr1-22.ldpruned.all.1kgv2
```

```
./plink --bfile chr1-22 --extract chr1-22.ldpruned.all.1kgv2.prune.in  --pca --threads 30
```

In R:

```
require('RColorBrewer')
options(scipen=100, digits=3)
```

Read in the eigenvectors, produced in PLINK
In R:

```
eigenvec <- read.table('plink.eigenvec', header = F, skip=0, sep = ' ')
rownames(eigenvec) <- eigenvec[,2]
eigenvec <- eigenvec[,3:ncol(eigenvec)]
colnames(eigenvec) <- paste('Principal Component ', c(1:20), sep = '')
```

First we read in the PED data

```
PED <- read.table("all_phase3.king.psam", header = TRUE, skip = 0, sep = '\t')
PED <- PED[which(PED$IID %in% rownames(eigenvec)), ]
PED <- PED[match(rownames(eigenvec), PED$IID),]
```

Then we set up the colour scheme we are going to use

```
require('RColorBrewer')
PED$Population <- factor(PED$Population, levels=c(
  "ACB","ASW","ESN","GWD","LWK","MSL","YRI",
  "CLM","MXL","PEL","PUR",
  "CDX","CHB","CHS","JPT","KHV",
  "CEU","FIN","GBR","IBS","TSI",
  "BEB","GIH","ITU","PJL","STU"))

col <- colorRampPalette(c(
  "yellow","yellow","yellow","yellow","yellow","yellow","yellow",
  "forestgreen","forestgreen","forestgreen","forestgreen",
  "grey","grey","grey","grey","grey",
  "royalblue","royalblue","royalblue","royalblue","royalblue",
  \
     "black","black","black","black","black"))(length(unique(PED$Population)))[factor(PED$Population)]
```

Finally we generate our PCA plots

```
project.pca <- eigenvec
par(mar = c(5,5,5,5), cex = 2.0,
    cex.main = 7, cex.axis = 2.75, cex.lab = 2.75, mfrow = c(1,2))

plot(project.pca[,1], project.pca[,2],
     type = 'n',
     main = 'A',
     adj = 0.5,
     xlab = 'First component',
     ylab = 'Second component',
     font = 2,
     font.lab = 2)
points(project.pca[,1], project.pca[,2], col = col, pch = 20, cex = 2.25)
legend('bottomright',
       bty = 'n',
        cex = 3.0,
        title = '',
        c('AFR', 'AMR', 'EAS',
          'EUR', 'SAS'),
        fill = c('yellow', 'forestgreen', 'grey', 'royalblue', 'black'))

plot(project.pca[,1], project.pca[,3],
      type="n",
      main="B",
      adj=0.5,
      xlab="First component",
      ylab="Third component",
      font=2,
      font.lab=2)
points(project.pca[,1], project.pca[,3], col=col, pch=20, cex=2.25)
```

Q

(i) What is distinct about the PC projections of the AMR group relative to other populations? (ii) Why does this occur and (iii) what does it tell us about the ancestral origins of this population?

3 Appendices

3.1 Downloading and preparation of 1000Genomes data

The instructions in this first appendix can be implemented as a script by typing vi download.sh and copying and pasting the code that follows. When finished type esc, full-colon, then w, q and return, (in sequence) to save. Once this is done enter chmod 777 download.sh to make the file executable. Then type ./download.sh to run the commands automatically.
Download and install Plink v1.9 and Plink v2.0 to your home directory

```
wget https://s3.amazonaws.com/plink1-assets/plink_linux_x86_64_20220402.zip
wget https://s3.amazonaws.com/plink2-assets/alpha3/plink2_linux_avx2_20220519.zip
unzip plink_linux_x86_64_20220402.zip
unzip plink2_linux_avx2_20220519.zip
```

Download and decompress 1000 Genomes phase 3 data

```
refdir='{Path/to/home/directory}' #Enter path to your home directory path here

cd $refdir
pgen=https://www.dropbox.com/s/e5n8yr4n7y91fyp/all_hg38.pgen.zst?dl=1
pvar=https://www.dropbox.com/s/cy46f1c8yutd1h4/all_hg38.pvar.zst?dl=1
sample=https://www.dropbox.com/s/3j9zg103fi8cjfs/hg38_corrected.psam?dl=1

wget $pgen
mv 'all_hg38.pgen.zst?dl=1' all_phase3.pgen.zst
./plink2 --zst-decompress all_phase3.pgen.zst > all_phase3.pgen

wget $pvar
mv 'all_hg38.pvar.zst?dl=1' all_phase3.pvar.zst

wget $sample
mv 'hg38_corrected.psam?dl=1' all_phase3.psam
```

Download list of 1st and 2nd degree relatiives from above plink page (based on KING output)

```
wget https://www.dropbox.com/s/129gx0gl2v7ndg6/deg2_hg38.king.cutoff.out.id?dl=1
mv deg2_hg38.king.cutoff.out.id?dl=1 deg2_hg38.king.cutoff.out

./plink2 \
        --pfile $refdir/all_phase3 vzs \
        --allow-extra-chr \
        --remove deg2_hg38.king.cutoff.out \
        --make-pgen \
        --out $refdir/all_phase3.king
```

Convert 1000 Genomes phase 3 data to plink 1 binary format

```
./plink2 \
  --pfile $refdir/all_phase3.king \
  --allow-extra-chr \
  --max-alleles 2 \
  --make-bed \
  --out $refdir/all_phase3
```

```
mv $refdir/all_phase3.log $refdir/log
```

The following selects the correct versions of chromosomes 1 – 22.

```
for i in {1..22}; do
        plink --bfile all_phase3 --chr ${i} --make-bed --allow-extra-chr --threads 50 --out \
      chr${i}
done

for i in {1..22}; do
        echo chr${i} >> mergelist.txt;
done

plink --merge-list mergelist.txt --make-bed --threads 50 --out chr1-22   #This step takes \
      a while
```

Cleanup

```
for i in {1..22}; do
        rm chr${i}.bed chr${i}.fam chr${i}.bim chr${i}.log
done
```