PCA and Admixture

Maria Izabel cavassim Alves

2018-02-23

## Population Structure analysis

With the advent of SNP data it is possible to precisely infer the genetic distance across individuals or populations. As written in the book, one way of doing it is by comparing each SNP from each individual against every other individual. This comparison produces the so called: covariance matrix, which in genetic terms means the number of shared polymorphisms across individuals. There are many ways to visualize this data, in this tutorial you will be exposed to Principal Component Analysis and Admixture software.

We will use the R package SNPRelate, which can easily handle vcf files and do the PCA. If you want to explore a bit more on the functionality of the package access [here](https://www.rdocumentation.org/packages/SNPRelate/versions/1.6.4).

# Dependencies  
#install.packages('SNPRelate')  
library(SNPRelate)

## Loading required package: gdsfmt

## SNPRelate -- supported by Streaming SIMD Extensions 2 (SSE2)

library(ggplot2)  
  
# Reading the metadata information   
info = read.csv("/Users/PM/Downloads/Sample\_meta\_data.csv", header = T, sep = ';')  
  
# Setting the directory of the VCF file   
vcf.fn <- "/Users/PM/Downloads/Allvariants\_135\_145\_chr2\_test.vcf.recode.vcf"  
  
# Reading the vcf file  
snpgdsVCF2GDS(vcf.fn, "/Users/PM/Downloads/example2.gds", method="biallelic.only")

## VCF Format ==> SNP GDS Format  
## Method: exacting biallelic SNPs  
## Number of samples: 27  
## Parsing "/Users/PM/Downloads/Allvariants\_135\_145\_chr2\_test.vcf.recode.vcf" ...  
## import 49868 variants.  
## + genotype { Bit2 27x49868, 328.7K } \*  
## Optimize the access efficiency ...  
## Clean up the fragments of GDS file:  
## open the file '/Users/PM/Downloads/example2.gds' (626.8K)  
## # of fragments: 48  
## save to '/Users/PM/Downloads/example2.gds.tmp'  
## rename '/Users/PM/Downloads/example2.gds.tmp' (626.5K, reduced: 336B)  
## # of fragments: 20

genofile <- snpgdsOpen("/Users/PM/Downloads/example2.gds", FALSE, TRUE, TRUE)  
pca <- snpgdsPCA(genofile)

## Principal Component Analysis (PCA) on genotypes:  
## Excluding 0 SNP on non-autosomes  
## Excluding 397 SNPs (monomorphic: TRUE, < MAF: NaN, or > missing rate: NaN)  
## Working space: 27 samples, 49,471 SNPs  
## using 1 (CPU) core  
## PCA: the sum of all selected genotypes (0, 1 and 2) = 2250084  
## Fri Feb 23 17:31:49 2018 (internal increment: 27760)  
##   
[..................................................] 0%, ETC: ---   
[==================================================] 100%, completed   
## Fri Feb 23 17:31:49 2018 Begin (eigenvalues and eigenvectors)  
## Fri Feb 23 17:31:49 2018 Done.

summary(pca)

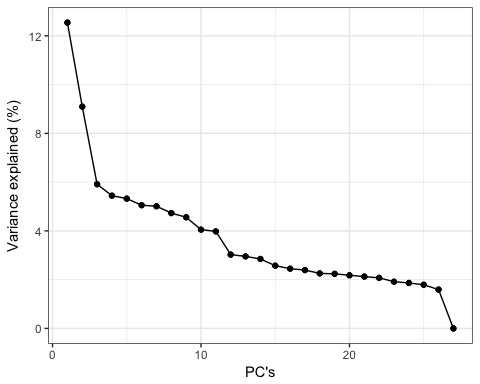
## Length Class Mode   
## sample.id 27 -none- character  
## snp.id 49471 -none- numeric   
## eigenval 27 -none- numeric   
## eigenvect 729 -none- numeric   
## varprop 27 -none- numeric   
## TraceXTX 1 -none- numeric   
## Bayesian 1 -none- logical   
## genmat 0 -none- NULL

**Q.1** How many individuals and snps does this dataset have? What is an eigenvector and an eigenvalue? Hint: Have a look at page 180 of HEG.

eigenvectors = as.data.frame(pca$eigenvect)  
colnames(eigenvectors) = as.vector(sprintf("PC%s", seq(1:nrow(pca$eigenvect))))  
pca$sample.id = sub("\_chr2\_piece\_dedup", "", pca$sample.id)  
  
# Matching the sample names with their origin and population  
eigenvectors$region = info[match(pca$sample.id, info$ENA.RUN),]$region   
eigenvectors$population = info[match(pca$sample.id, info$ENA.RUN),]$population

Let’s first look at how much of the variance of the data is explained by each eigenvalue (or PC):

# Variance proportion:  
pca\_percent <- pca$varprop\*100  
  
qplot(y = pca\_percent, x = seq(1, length(pca$eigenval))) + geom\_line() + geom\_point() + theme\_bw() + xlab("PC's") + ylab("Variance explained (%)")



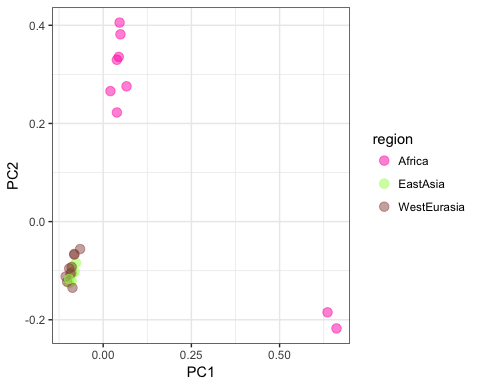
dev.off()

## null device   
## 1

**Q.2** How many PC’s do we need in order to explain 50% of the variance of the data? Can you make an accumulative plot of the variance explained PC?

Now, let’s plot the two first PC’s and color the datapoints by the origin of each individual sample.

ggplot(data = eigenvectors, aes(x = PC1, y = PC2, col = region)) +   
 geom\_point(size=3,alpha=0.5) +  
 scale\_color\_manual(values = c("#FF1BB3","#A7FF5B","#99554D")) +  
 theme\_bw()

 **Q.2** Try to plot PC2 and PC3. Do you see the same patterns?

**Q.3** Try also to color the graph based on population. What do you observe?

Now we will implement LD prunning.

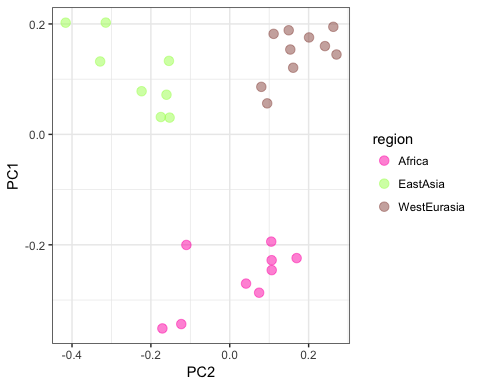
set.seed(1000)  
  
# Try different LD thresholds for sensitivity analysis  
snpset <- snpgdsLDpruning(genofile, ld.threshold=0.3)

## SNP pruning based on LD:  
## Excluding 0 SNP on non-autosomes  
## Excluding 397 SNPs (monomorphic: TRUE, < MAF: NaN, or > missing rate: NaN)  
## Working space: 27 samples, 49,471 SNPs  
## using 1 (CPU) core  
## Sliding window: 500000 basepairs, Inf SNPs  
## |LD| threshold: 0.3  
## Chromosome 2: 1.20%, 598/49868  
## 598 SNPs are selected in total.

# Get all selected snp's ids  
snpset.id <- unlist(snpset)  
  
pca\_pruned <- snpgdsPCA(genofile, snp.id=snpset.id, num.thread=2)

## Principal Component Analysis (PCA) on genotypes:  
## Excluding 49,270 SNPs (non-autosomes or non-selection)  
## Excluding 0 SNP (monomorphic: TRUE, < MAF: NaN, or > missing rate: NaN)  
## Working space: 27 samples, 598 SNPs  
## using 2 (CPU) cores  
## PCA: the sum of all selected genotypes (0, 1 and 2) = 29329  
## Fri Feb 23 17:31:50 2018 (internal increment: 27760)  
##   
[..................................................] 0%, ETC: ---   
[==================================================] 100%, completed   
## Fri Feb 23 17:31:50 2018 Begin (eigenvalues and eigenvectors)  
## Fri Feb 23 17:31:50 2018 Done.

eigenvectors = as.data.frame(pca\_pruned$eigenvect)  
colnames(eigenvectors) = as.vector(sprintf("PC%s", seq(1:nrow(pca$eigenvect))))  
pca\_pruned$sample.id = sub("\_chr2\_piece\_dedup", "", pca$sample.id)  
  
# Matching the sample names with their origin and population  
eigenvectors$region = info[match(pca\_pruned$sample.id, info$ENA.RUN),]$region   
eigenvectors$population = info[match(pca\_pruned$sample.id, info$ENA.RUN),]$population  
  
ggplot(data = eigenvectors, aes(x = PC1, y = PC2, col = region)) +   
 geom\_point(size=3,alpha=0.5) +  
 scale\_color\_manual(values = c("#FF1BB3","#A7FF5B","#99554D")) +  
 theme\_bw() + coord\_flip()



\*\* Q.4\*\* Implement different LD thresholds (0.1, 0.2, 0.3, 0.4, 0.5). How many SNPs are left after each filtering threshold? Are these SNPs linked?

Now we are going to convert this GDS file into a plink format, to be later used in the admixture exercise:

snpgdsGDS2BED(genofile, "/Users/PM/Downloads/test2", sample.id=NULL, snp.id=snpset.id, snpfirstdim=NULL, verbose=TRUE)

## Converting from GDS to PLINK binary PED:  
## Working space: 27 samples, 598 SNPs  
## Output a BIM file.  
## Output a BED file ...  
## Fri Feb 23 17:31:50 2018 0%  
## Fri Feb 23 17:31:50 2018 100%  
## Done.

# Admixture

Admixture is a program for estimating ancestry in a model based manner from autossomal SNP genotype datasets, where individuals are unrelated. The input format required by the software is in binary PLINK (.bed) file. That is why we converted our vcf file into .bed.

Now with adjusted format and pruned snps, we are ready to run the admixture analysis. We believe that our individuals in the sample data derive their ancestry from three ancestral populations:

admixture\_macosx-1.3.0/admixture test2.bed 3

**Q.5** Have a look at the Fst across populations, that is printed in the terminal. Would you guess which populations are Pop0, Pop1 and Pop2 referring to?

After running admixture, 2 outuputs are generated:

* Q: the ancestry fractions
* P: the allele frequencies of the inferred ancestral populations

Sometimes we may have no priori about K, one good way of choosing the best K is by doing a cross-validation procedure impletemented in admixture as follow:

for K in 1 2 3 4 5; \  
 do /admixture\_macosx-1.3.0/admixture --cv hapmap.bed $K | tee log${K}.out; done

Have a look at the Cross Validation error of each K:

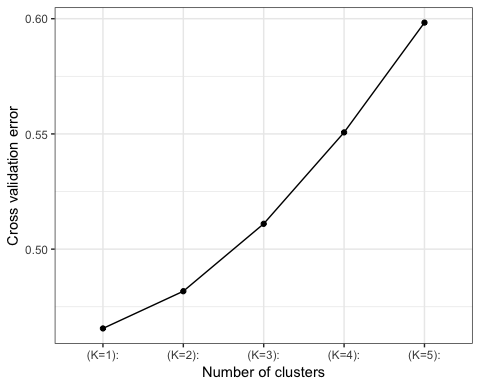
grep -h CV log\*.out

Save it in a text file:

grep -h CV log\*.out > CV\_logs.txt

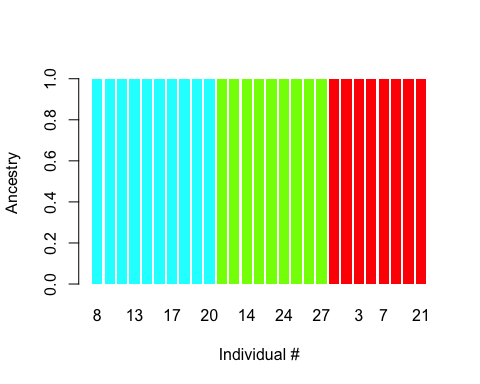
Look at the distribution of CV error:

CV = read.table('/Users/PM/Downloads/CV\_logs.txt')  
p <- ggplot(data = CV, aes(x = V3, y = V4, group = 1)) + geom\_line() + geom\_point() + theme\_bw() + labs(x = 'Number of clusters', y = 'Cross validation error')  
p

 **Q.6** What do you understand of Cross validation error? Based on this graph, what is the best K?

Plotting the Q estimates. Choose the K that makes more sense to you.

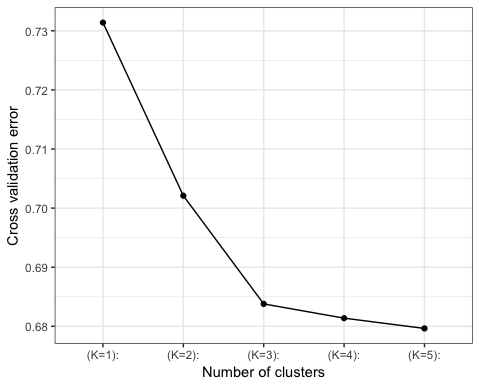
tbl = read.table("/Users/PM/Downloads/test2.3.Q")  
ord = tbl[order(tbl$V1,tbl$V2,tbl$V3),]  
bp = barplot(t(as.matrix(ord)),   
 space = c(0.2),  
 col=rainbow(4),  
 xlab="Individual #",   
 ylab="Ancestry",  
 border=NA)



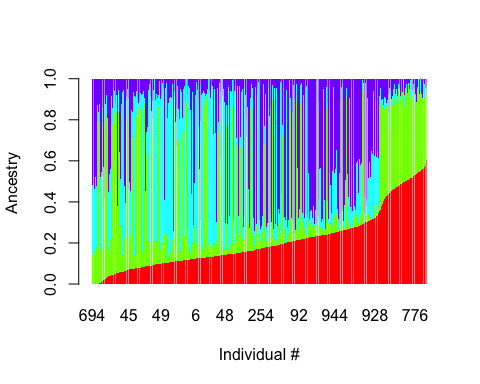
**Q.7** How many cluster do you identify in this plot? Does that agree with what was found using PCA?

In the following part of this exercise you will do both analysis (PCA and Admixture) using a different dataset. The data comes from the HAPMAP Consortium, to learn more about the populations studied in this project access [here](http://www.sanger.ac.uk/resources/downloads/human/hapmap3.html). A information file **relationships\_w\_pops\_121708.txt**, as well as **.bim**, **.bed**, **.fam** files are available for the admixture analysis. Answer the same questions as answered in this tutorial and write a report (5 pages maximum) about the results and the analysis you have done. The deadline of the report will be given during the lecture.

CV = read.table('/Users/PM/Downloads/hapmap/CV\_logs.txt')  
p <- ggplot(data = CV, aes(x = V3, y = V4, group = 1)) + geom\_line() + geom\_point() + theme\_bw() + labs(x = 'Number of clusters', y = 'Cross validation error')  
p



tbl = read.table("/Users/PM/Downloads/hapmap/hapmap.4.Q")  
ord = tbl[order(tbl$V1,tbl$V2,tbl$V3),]  
bp = barplot(t(as.matrix(ord)),   
 space = c(0.2),  
 col=rainbow(4),  
 xlab="Individual #",   
 ylab="Ancestry",  
 border=NA)



# Setting the directory of the VCF file   
vcf.fn <- "/Users/PM/Downloads/hapmap/hapmap.vcf"  
  
# Reading the vcf file  
snpgdsVCF2GDS(vcf.fn, "/Users/PM/Downloads/hapmap/example2.gds", method="biallelic.only")

## VCF Format ==> SNP GDS Format  
## Method: exacting biallelic SNPs  
## Number of samples: 1184  
## Parsing "/Users/PM/Downloads/hapmap/hapmap.vcf" ...  
## import 1803 variants.  
## + genotype { Bit2 1184x1803, 521.2K } \*  
## Optimize the access efficiency ...  
## Clean up the fragments of GDS file:  
## open the file '/Users/PM/Downloads/hapmap/example2.gds' (546.3K)  
## # of fragments: 40  
## save to '/Users/PM/Downloads/hapmap/example2.gds.tmp'  
## rename '/Users/PM/Downloads/hapmap/example2.gds.tmp' (546.1K, reduced: 240B)  
## # of fragments: 20

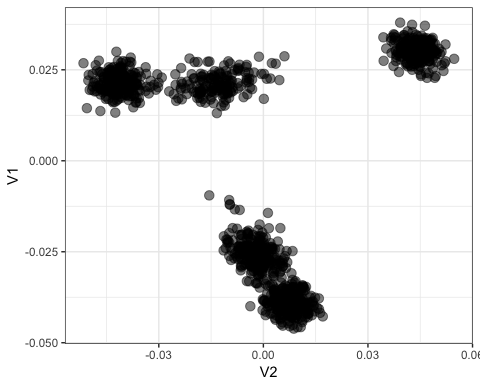
genofile2 <- snpgdsOpen("/Users/PM/Downloads/hapmap/example2.gds", FALSE, TRUE, TRUE)  
pca <- snpgdsPCA(genofile2)

## Principal Component Analysis (PCA) on genotypes:  
## Excluding 0 SNP on non-autosomes  
## Excluding 0 SNP (monomorphic: TRUE, < MAF: NaN, or > missing rate: NaN)  
## Working space: 1,184 samples, 1,803 SNPs  
## using 1 (CPU) core  
## PCA: the sum of all selected genotypes (0, 1 and 2) = 2148105  
## Fri Feb 23 17:31:52 2018 (internal increment: 632)  
##   
[..................................................] 0%, ETC: ---   
[==================================================] 100%, completed   
## Fri Feb 23 17:31:53 2018 Begin (eigenvalues and eigenvectors)  
## Fri Feb 23 17:31:53 2018 Done.

summary(pca)

## Length Class Mode   
## sample.id 1184 -none- character  
## snp.id 1803 -none- numeric   
## eigenval 1184 -none- numeric   
## eigenvect 37888 -none- numeric   
## varprop 1184 -none- numeric   
## TraceXTX 1 -none- numeric   
## Bayesian 1 -none- logical   
## genmat 0 -none- NULL

eigenvectors = as.data.frame(pca$eigenvect)  
  
ggplot(data = eigenvectors, aes(x = V1, y = V2)) +   
 geom\_point(size=3,alpha=0.5) +  
 scale\_color\_manual(values = c("#FF1BB3","#A7FF5B","#99554D")) +  
 theme\_bw() + coord\_flip()



vcftools --vcf Allvariants\_135\_145\_chr2.vcf --recode --out Allvariants\_135\_145\_chr2\_test.vcf --remove-indv ERR1025639\_chr2\_piece\_dedup