PCA and Admixture

Maria Izabel cavassim Alves

2018-02-19

## Population Structure analysis

With the advent of SNP chip 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.

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

## Loading required package: gdsfmt

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

library(ggplot2)  
  
# Reading the vcf file and doing the eigendecomposition  
info = read.csv("/Users/PM/Downloads/sample\_infos\_accessionnb.csv", header = T, sep = ';')  
  
# Opening the VCF file and calculating eigendecomposition  
vcf.fn <- "/Users/PM/Downloads/Allvariants\_135\_145\_chr2.vcf"  
snpgdsVCF2GDS(vcf.fn, "/Users/PM/Downloads/first2.gds", method="biallelic.only")

## VCF Format ==> SNP GDS Format  
## Method: exacting biallelic SNPs  
## Number of samples: 28  
## Parsing "/Users/PM/Downloads/Allvariants\_135\_145\_chr2.vcf" ...  
## import 49868 variants.  
## + genotype { Bit2 28x49868, 340.9K } \*  
## Optimize the access efficiency ...  
## Clean up the fragments of GDS file:  
## open the file '/Users/PM/Downloads/first2.gds' (639.0K)  
## # of fragments: 48  
## save to '/Users/PM/Downloads/first2.gds.tmp'  
## rename '/Users/PM/Downloads/first2.gds.tmp' (638.7K, reduced: 336B)  
## # of fragments: 20

genofile <- snpgdsOpen("/Users/PM/Downloads/first2.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: 28 samples, 49,471 SNPs  
## using 1 (CPU) core  
## PCA: the sum of all selected genotypes (0, 1 and 2) = 2250084  
## Mon Feb 19 21:38:13 2018 (internal increment: 26768)  
##   
[..................................................] 0%, ETC: ---   
[==================================================] 100%, completed   
## Mon Feb 19 21:38:13 2018 Begin (eigenvalues and eigenvectors)  
## Mon Feb 19 21:38:13 2018 Done.

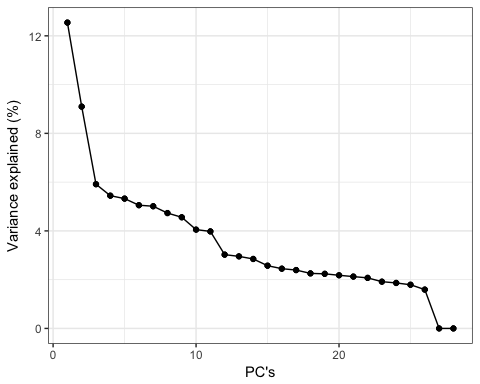
summary(pca)

## Length Class Mode   
## sample.id 28 -none- character  
## snp.id 49471 -none- numeric   
## eigenval 28 -none- numeric   
## eigenvect 784 -none- numeric   
## varprop 28 -none- numeric   
## TraceXTX 1 -none- numeric   
## Bayesian 1 -none- logical   
## genmat 0 -none- NULL

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

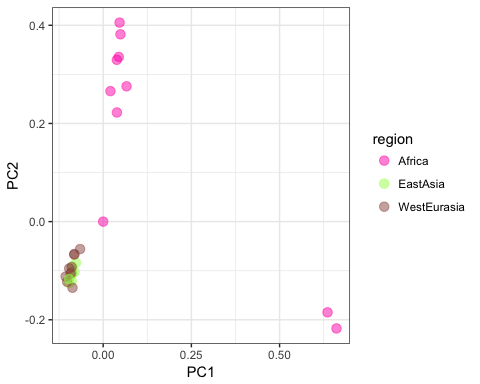
Let’s first produce look at how much of the variance of the data is explained by each eigenvalue (or PC):

# 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 (%)")



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()



Try to plot PC2 and PC3. Do you see the same patterns? Now we will implement LD prunning

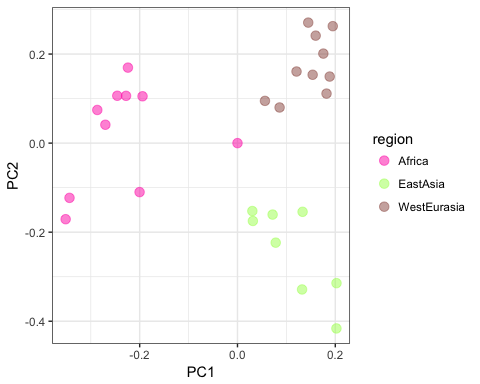
set.seed(1000)  
  
# Try different LD thresholds for sensitivity analysis  
snpset <- snpgdsLDpruning(genofile, ld.threshold=0.3)

## SNP pruning based on LD:  
## Excluding 0 SNP on non-autosomes  
## Excluding 397 SNPs (monomorphic: TRUE, < MAF: NaN, or > missing rate: NaN)  
## Working space: 28 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 id  
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: 28 samples, 598 SNPs  
## using 2 (CPU) cores  
## PCA: the sum of all selected genotypes (0, 1 and 2) = 29329  
## Mon Feb 19 21:38:14 2018 (internal increment: 26768)  
##   
[..................................................] 0%, ETC: ---   
[==================================================] 100%, completed   
## Mon Feb 19 21:38:14 2018 Begin (eigenvalues and eigenvectors)  
## Mon Feb 19 21:38:14 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()



# 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. Therefore we first need to convert our vcf file into .bed:

plink --vcf chr2\_135\_145.vcf --maf 0.05 --indep-pairwise 1000 50 0.8 --mind 0.02 --recode12 --biallelic-only --out test

Note that at the same time we are filtering the data based on: - --maf: minor allele frequency above 0.05

* --indep-pairwise: pruned subset of markers that are in approximate linkage equilibrium
* --mind: excludes individuals with missing data above 0.02

Now with adjusted format and pruned snps, we are able to run the admixture analysis:

admixture\_macosx-1.3.0/admixture test.ped 3

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:

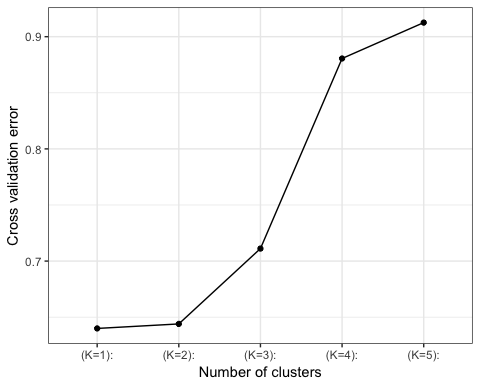
for K in 1 2 3 4 5; \  
 do admixture\_macosx-1.3.0/admixture --cv test.ped $K | tee log${K}.out; done

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

grep -h CV log\*.out

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

 Based on this graph, what is the best K?

Plotting the Q estimates:

tbl=read.table("/Users/PM/Downloads/test.3.P")  
barplot(t(as.matrix(tbl)), col=rainbow(3),  
xlab="Individual #", ylab="Ancestry", border=NA)

