Genomic approaches: population genomics

Thibault Leroy (thibault.leroy@univie.ac.at)

Part 2: Detecting selection within genomes: April 23, 2020

General context

In this second part, the objective is to introduce some approaches to detect recent footprints of natural (or not) positive selection in genomes. Among all the methods available, I will only focus on methods that are based on the allele frequency differences among populations (the most frequently used). Even considering this subset of methods, I have had to make choices. I decided to introduce the use of Fst, XtX (Fst "corrected" for the levels of population structure, as implemented in Bayenv or BayPass) and Genotype-Environment Association (GEA), which represent quite conventional methods.
Today, I propose to work on a totally different model than yesterday, by rapidly reanalyzing some oak data, published in Leroy et al. 2020 (New Phytologist), this manuscript is also available on bioRxiv (https://www.biorxiv.org/content/10.1101/584847v1). To introduce a little bit more the context, we collected 18 populations, 10 from a latitudinal gradient (pop indicated only with numbers) in Europe, mostly in France, but also in Germany (253 & 256) and Ireland (124). We also collected 8 populations from two close valleys in the French Pyrenees (O=Ossau Valley, L=Luz Valley), at different altitudes (nubers 1, 4, 12, 16 corresponding to 100m, 400m, 1200m & 1600m above the sea level, respectively). Today, we will use pool-seq data, but the same approach is also completely valid for individual data (if you have at least >15 diploid individual per population). We just need to have allele counts per population.
The general idea of these tests is to try to detect SNPs that exhibit levels of differentiation that deviate from the expectations assuming the null hypothesis of selective neutrality (i.e. a neutral marker). But, as you can imagine, this task is still quite complex. The objective of this course is to introduce how to do this step by step.

Earlier steps

With the limited time available for this course and the quantity of data, the first steps of the analysis (mapping, "SNP calling", ...) cannot be done during this course. Ovidiu already introduced these steps, so it is not really useful for today. In addition, the code for these data is publicly available on my github (*https://github.com/ThibaultLeroyFr/Intro2PopGenomics*, section 3.3.3).

We will start our analyses by using the synchronized pileup at the end of this pipeline (a file format introduced by Robert Kofler in Popoolation2).
Each line corresponds to a position in the genome and the 3 first colomns indicate the name of the scaffold, the position on this scaffold, the reference allele. The 4th column indicates the allele counts for the first population, the 5th for the 2nd population, and so on. A:T:C:G:N:*

```
Sc0000000       1   T       0:101:0:0:0:0   0:130:0:0:0:0 ...
Sc0000000       2   A       74:0:7:0:0:0    105:0:12:0:0:0 ...
...
```

The first position seems to be a non-variant, while the second position seems to be a SNP with two alleles A/C. Of course, given the length of the oak genome sequence (814 Mb, Plomion et al. 2018, Nature plants), It is impossible to work on the complete synchronized pileup today. For the three first steps, we will use a file corresponding to ~200,000 positions of this synchronized pileup randomly selected across the oak genome.

Step 1: Identify SNPs and compute a pairwise Fst matrix

load the packages (or install):

```
library("poolfstat") # to install: install.packages("poolfstat")
library(reshape2)
library(ggplot2)
```

Read the sync file and detect SNPs, assuming the number of haplotypes in each pool and a threshold for the minor allele frequency

```
pooldata=popsync2pooldata(sync.file="BNP-BHW_18pops_v2.3.pileup.200k.sync",
poolsizes=c(50,50,50,50,50,44,50,50,50,50,40,40,40,40,40,40,20,36),
poolnames=c("9","97","124","204","217","218","219","233","253","256",
"L1","O1","L8","O8","O12","L12","O16","L16"),
min.rc=10,min.cov.per.pool = 50, max.cov.per.pool = 250,
min.maf=0.02,noindel=TRUE)
```

How many SNPs you identified? On average, every how many bases a SNP can be identified in the genome?

Compute a pairwise Fst matrix:

```
Fstmatrix=computePairwiseFSTmatrix(pooldata, method = "Anova",
min.cov.per.pool = 50, max.cov.per.pool = 250, min.maf = 0.02,
output.snp.values = FALSE)$PairwiseFSTmatrix
```

Show the variation of the Fst among all pairs of populations (populations are in the same order than read by popsync2pooldata)

```
image(Fstmatrix, axes=FALSE)
```

If you have ggplot2 already installed (or can install it), you can generate an improved plot:

```
melted_Fstmatrix<- melt(Fstmatrix)
ggplot(data = melted_Fstmatrix, aes(x=Var1, y=Var2, fill=value))+
geom_tile()+
scale_fill_gradient2("Pairwise\nFST values", low = "navyblue",
mid = "white",high = "red",  midpoint = 0)+
xlab("")+ylab("")+
scale_x_discrete(labels=c("Pool1" = "9", "Pool2" = "97",
"Pool3" = "124","Pool4" = "204", "Pool5" = "217",
"Pool6" = "218", "Pool7" = "219", "Pool8" = "233",
"Pool9" = "253","Pool10" = "256","Pool11" = "L1",
"Pool12" = "O1","Pool13" = "L8", "Pool14" = "O8",
"Pool15" = "O12","Pool16" = "L12", "Pool17" = "O16",
"Pool18" = "L16"))+
scale_y_discrete(labels=c("Pool1" = "9", "Pool2" = "97",
"Pool3" = "124", "Pool4" = "204", "Pool5" = "217",
"Pool6" = "218","Pool7" = "219", "Pool8" = "233",
"Pool9" = "253","Pool10" = "256", "Pool11" = "L1",
"Pool12" = "O1","Pool13" = "L8", "Pool14" = "O8",
"Pool15" = "O12", "Pool16" = "L12", "Pool17" = "O16",
"Pool18" = "L16"))+
theme_bw() + theme(panel.border = element_blank(),
panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
axis.line = element_line(colour = "black"))+ theme(legend.key =
element_blank())+
theme(legend.title=element_text(size=12, face="bold"))
```

The pairwise Fst matrix captures the population structure. Which population(s) seem the most different from all others?

A convenient way to look at these differences, is to represents them as a tree: (here an exemple using the package ape we used yesterday):

```
library("ape")
Fstmatrix2 = Fstmatrix
Fstmatrix2[Fstmatrix2 <0] <- 0
plot(ladderize(as.phylo(hclust(as.dist(Fstmatrix2)))))
```

The distances shown here are proportional to the difference in average Fst among populations. As you can see in the text, to built this tree, I have replaced the negative Fst values by 0. Indeeed, slightly negative values can be observed because of some corrections for the small sample sizes etc, but such a value can be considered as a zero value (i.e. it means no differentiation).

Even if we only used a few thousands SNPs in this analysis, this result is already meaningful. Indeed, to evaluate population structure
Regarding the two sampling zones (across 10 populations from Western Europe at low elevation or within the 2 French Pyrenees Valleys from low to high elevation), which populations seem the most differentiated?

Step 2: Compute the distribution of Fst values over all populations

Another important evaluation is to look at the distribution of Fst values across all SNPs.
Based on your subset, it is possible to draw this distribution of the Fst value per SNPs over all populations.

```
SNPvalues=computeFST(pooldata, method="Anova")$snp.FST
SNPvalues=as.data.frame(SNPvalues)
plot(density(SNPvalues$SNPvalues),xlim=c(0,1),lwd=1.5,col="navyblue",xlab="Fst",main="")
```

The first question you probably want to answer is : which marker is the most differentiated?

The greatest level of Fst observed is:

```
max(SNPvalues$SNPvalues)
```

Ok but you are probably interested to know which marker is it:

(the function which is really useful to rapidly get such an information in a large dataset)
`which(SNPvalues$SNPvalues==max(SNPvalues))`

Now, you know the number of the SNP in the full list of SNPs, but this information is still poorly informative.
But I guess, you are more interested by the genomic location of this SNP. The idea is to get the information from your file
`pooldata@snp.info[which(SNPvalues$SNPvalues==max(SNPvalues)),]`

Overall, this SNP exhibits greater levels of differentiation than any other SNP of our dataset, but you are probably interested to know which populations are the most different. To do this, you can compare the total number of reference alleles and contrast it to the total number of each population.

```
pooldata@refallele.readcount[which(SNPvalues$SNPvalues==max(SNPvalues)),]
pooldata@readcoverage[which(SNPvalues$SNPvalues==max(SNPvalues)),]
```

Which population exhibits the highest allele frequency (reference allele)? Which one the lowest?
(The population are listed in the same way than initially read by the function popsync2pooldata). < br>

Of course you are perhaps to know which SNPs exhibit the top 1% values (that you would like to interpret as a first evidence for positive selection).

First, you want to know the Fst value corresponding to this threshold: `quantile(SNPvalues$SNPvalues, probs = 0.99, na.rm = TRUE)`

Ok but now, you want to identify all the SNPs corresponding to this. `which(SNPvalues$SNPvalues>quantile(SNPvalues$SNPvalues, probs = 0.99, na.rm = TRUE))`

Great, but I guess you are still more interested to know in which genomic location are these SNPs. `pooldata@snp.info[which(SNPvalues$SNPvalues>quantile(SNPvalues$SNPvalues, probs = 0.99, na.rm = TRUE)),]`

Note: In fact by doing this, you have done your first "genome scan" for Fst outliers, because you tried to identify SNPs exhibiting more differentiation as compared to all others (the tail of the distribution).

Step 3: Compute the distribution of Fst values over pairs of populations

We first we compute Fst for all SNPs and all pairs of species

```
Fstlocus=computePairwiseFSTmatrix(pooldata, method = "Anova",
min.cov.per.pool = 50, max.cov.per.pool = 250, min.maf = 0.02,
output.snp.values = TRUE)$PairwiseSnpFST
```

```
head(Fstlocus)
```

We can reshape a little bit the matrix using a function of the package "reshape2".

```
head(melted_Fstlocus <- melt(Fstlocus))
hcolnames(melted_Fstlocus) <- c("index","comparison_pops","Fstvalues")
```

Then generate distribution for the first two pairwise comparisons (or any pairs of populations you want):

```
plot(density(melted_Fstlocus$Fstvalues[melted_Fstlocus$comparison_pops=="9_vs_97"],
na.rm=TRUE),main="",col="darkgreen",xlim=c(-0.1,1))+
lines(density(melted_Fstlocus$Fstvalues[melted_Fstlocus$comparison_pops=="9_vs_124"],
na.rm=TRUE),main="",col="darkred")
```

You can also use ggplot2, to plot all in the same time:

```
ggplot(data=melted_Fstlocus,aes(x=Fstvalues,color=comparison_pops))
+ geom_density(size=0.1) +
xlim(-0.1,1) +xlab("Pairwise Fst values")+ theme_bw()
```

Based on this plot, you can see that indeed performed a lot of pairwise comparisons!!!
The best to do is therfore to remove the legend by adding "theme(legend.position = "none")"

```
ggplot(data=melted_Fstlocus,aes(x=Fstvalues,color=comparison_pops))
+
geom_density(size=0.1) +  xlim(-0.1,1) + xlab("Pairwise Fst
values")+
theme_bw()+theme(legend.position = "none")
```

Using a pairwise comparison of your choice, try to identify the SNPs exhibiting the 1% highest observed values.

Step 4: XtX an analogous to FST, but accounting for the population structure

Note: A main problem of our previous approaches which were based on the distribution of Fst is the following: we do not explitly take into account the neutral processes that have shaped the genome-wide variation in Fst. In addition, using the top 1% of SNPs exhibiting the highest values, we explictly assume that 1% of the genome was under selection. Let us imagine, that all the populations evolved under strict neutrality, using the top 1% strategy, you will still detect 1% of the SNPs as outliers. In this case, the best method would be a method allowing to detect nothing. At the opposite, in your sets of populations, perhaps 5% or more of the genome was targeted (or in linkage disequilibrium with the loci targeted) by natural selection, but using this criteria you will only detect the top 1%. How to solve this?
The best would be to reconstruct the demographic history of all the investigated populations to perform neutral simulations assuming these past history to determine the proportion of SNPs that deviate from these neutral expectations. In practice, such a solution remains very difficult to do. A simplest way to try to solve this problem is to compute an analog of Fst accounting for the variance-covariance matrix of allele frequencies among populations (i.e. the population structure), and then to perform neutral simulations assuming this matrix to generate the expected distribution of Fst under neutrality.

So first, we need to generate an input file for the BayPass software (please do not do this step, not useful):
(http://www1.montpellier.inra.fr/CBGP/software/baypass/), it is very easy to do with the R package poolfstat

```
pooldata2genobaypass(pooldata,writing.dir=getwd(),prefix="",
subsamplesize=200000,subsamplingmethod="thinning")
```

And then to use this input file (or ideally, a file containing hundreds of thousands SNPs) to perform a BayPass analysis as shown here (given the computating time, we will not do this step today):
(https://github.com/ThibaultLeroyFr/Intro2PopGenomics/blob/master/3.3.6/script_baypass/script_baypass.sh)

For today, you will directly import two output files, the first one corresponding to variance-covariance matrix computed by BayPass using SNPs distributed over the whole genome (omega matrix), the second one corresponding to the XtX values for the SNPs for the first 30 Mb of the chromosome 1 of the oak genome.

Read both files: `tmp <- scan("18pops_omega.out")`

```
omega_matrix=matrix(tmp,ncol = 18, byrow = FALSE)
XtX_BF=read.table("BayPass_Toc_pseudoK.Chr1",header=TRUE)
```

Generate a simple plot of the variance-covariance matrix of allele frequencies among populations:

```
image(omega_matrix,axes=FALSE)
```

Using the packages corrplot, you can generate a beautiful correlation matrix:

```
library(corrplot)
correlation.matrix=cov2cor(omega_matrix)
colnames(correlation.matrix)<- c("9","97","124","204", "217","218",
"219","233","253","256","L1","O1","L8","O8", "O12","L12","O16",
"L16")
rownames(correlation.matrix)<- c("9","97","124","204", "217","218",
"219","233","253","256","L1","O1","L8","O8", "O12","L12","O16",
"L16")
corrplot(correlation.matrix,method="shade")
```

And again use ape to draw a tree showing the differences in the matrix:

```
library(ape)
plot(as.phylo(hclust(as.dist(1-correlation.matrix))))
```

Do you consider that these results are globally consistent with the previously observed results based on the pairwise Fst matrix (step 1)?

Ok this matrix represents the levels of population structure assumed by BayPass to compute the analog of the XtX, but explicitly taking into account the population structure.

You can of course plot the distribution of the XtX values.

```
plot(density(XtX_BF$XtX),xlim=c(0,50),xlab="Observed XtX values",main=""))
```

As you can see, the values are no longer distributed between 0 and 1.

You can also generate a “Manhattan plot”, showing the levels of XtX values along the first 30 Mb of the chromosome 1 (can take one or two minutes to be drawn)

```
plot(XtX_BF$XtX~XtX_BF$position,cex=0.05,pch=20,xlab="Chr1 -
Position",ylab="XtX")
```

Step 5: Calibrating the XtX values

As for the Fst distribution, using the top 1% values of the XtX values is not recommended as well, because we still do not know the proportion of the genome under selection. The alternative is to use the observed variance-covariance matrix (as well as some few files that I will not describe today) to perform neutral simulations and therefore use these neutral simulations to generate thresholds assuming neutrality.

Here I will not describe how to do this step in detail, it can be quite tricky. If you want to do similar analyses on your own data, please ask me. But again, in a nutshell, the rationale is to use some and some parameters generated by the previous simulation, to generate so-called pseudo-observed datasets, so simulated datasets assuming the levels of population structure and strict neutrality for which we can compute the XtX under BayPass. But given that these simulations were performed assuming neutrality, we therefore have a neutral distribution of the XtX and we can use these values as thresholds to discriminate neutral and outliers (SNPs deviating from neutral expectations).

I simulated 100,000 SNPs under strict neutrality (file "simulated_PODS_XtX.txt").

```
xtx.pods=read.table("simulated_PODS_XtX.txt",h=T)$M_XtX
xtx.threshold=quantile(xtx.pods$M_XtX,probs=c(0.999,0.9999,1))
```

What are the quantile values? The quantile "0.999" means that 1 simulated neutral SNP per thousand reached a XtX value of this value or higher than this value, "0.9999" one per ten thousand, 1 corresponds to highest XtX value observed on this set of simulations (it is similar than using max(xtx.pods$M_XtX)).

Now we have a neutral expectation for the distribution of the XtX assuming neutrality, so the idea is to use this quantiles as thresholds for the observed XtX values in our real data. How many SNPs in our real dataset exhibit XtX values higher than the maximum XtX value observed using the neutral simulations?

```
length(XtX_BF$XtX[XtX_BF$XtX>xtx.threshold[3]])
```

Which proportion of our set of SNPs exhibit XtX values which are never observed based on the neutral simulations?

```
plot(XtX_BF$XtX~XtX_BF$position,cex=0.05,pch=20,xlab="Chr1 -
Position",ylab="XtX",col=ifelse(XtX_BF$XtX>=xtx.threshold[3],"red","black"))
```

Try to modify this code, to show a different color for each threshold (i.e. quantiles).
Independently, if you have time left, you can also try to create a small piece of code (with a similar loop than for the course yesterday) to estimate the proportion of outliers using sliding windows along the genome (e.g. 100kb). Where is located the peak with the highest proportion of outliers?

Step 6: Genotype-Environment Associations

Under BayPass, it is also possible to use different population-specific covariates. The general idea is to identify SNPs that covaries with environmental variables. Just to mention briefly, that it also works using phenotypic values at the population level as covariates (i.e. mean trait value for each population, that's why Mathieu Gautier introduced the fact that this method can also be used to perform population GWAS "pGWAS"). If you are interested by this, in our study, we also used the date of leaf unfolding in common gardens as a covariate. Today, we will use on results based on the local climate of each population (and more precisely, the yearly average temperature during the period 1950-2000). The idea is therefore to compare the likelihood of the two models, one model accounting for this potential "fixed" environmental effect, and a model without this effect (i.e. models with vs. without association). The output of this comparison of the two likelihoods is a Bayes Factor (BF) for each SNP, with the most positive values supporting the model with association (i.e. allele frequencies correlate with the temperature gradient). Using Bayes factors, we generally consider that a BF greater 15 provides a strong support and a BF greater than 20 provides a "decisive"" support. We will use this widely accepted criteria today. It must however be noted that a calibration of the BF using neutral simulations (i.e. a similar strategy than already used for the XtX) can also be done.

In this study, we are particularly interested by SNPs exhibiting strong differences in allele frequencies among populations (so a high XtX) and exhibiting an association with the mean local temperatures. To plot this, we can generate a scatterplot showing the values of XtX and the BF and use the thresholds for both metrics.

```
plot(XtX_BF$BF_Toc~XtX_BF$XtX,pch=20,xlab="XtX (differentiation)",
ylab="Bayes factor (association with climate)",
cex=ifelse(XtX_BF$XtX>=xtx.threshold[3] & XtX_BF$BF_Toc >= 20,0.3,
ifelse(XtX_BF$XtX>=xtx.threshold[2] & XtX_BF$BF_Toc >= 15,0.2,0.1)),
col=ifelse(XtX_BF$XtX>=xtx.threshold[3] & XtX_BF$BF_Toc >= 20,"darkred",
ifelse(XtX_BF$XtX>=xtx.threshold[2] & XtX_BF$BF_Toc >= 15,"darkorange","darkgrey")))+
abline(h=15,lwd=1,lty=3,col="grey20")+
abline(h=20,lwd=1.5,lty=3,col="grey20")+
abline(v=xtx.threshold[2],lwd=1,lty=3,col="grey20")+
abline(v=xtx.threshold[3],lwd=1.5,lty=3,col="grey20")
```

Now, you are perhaps interested to know where are located the SNPs showing a strong association with climate along the genome. We can do this quite easily by generating a manhattan plot of the XtX values, and using a dedicated color for the associated SNPs.

```
plot(XtX_BF$XtX~XtX_BF$position,cex=0.05,pch=20,xlab="Chr1 -
Position",
ylab="XtX",col=ifelse(XtX_BF$XtX>=xtx.threshold[3] & XtX_BF$BF_Toc
>= 20,"darkred",
ifelse(XtX_BF$XtX>=xtx.threshold[2] & XtX_BF$BF_Toc >= 15,"darkorange","darkgrey")))
```

In which region(s) of the genome can you observe most associated SNPs?
In Leroy et al. (2020, New Phytologist), we then identified candidate genes in these regions and hypothesized that the gene responsible for these associated SNPs might be due to either SCD1 or ASPG1 (or both). SCD1 is a gene which is know to be associated with the stomatal responses to water stress and can therefore be expected to be associated with temperature. ASPG1 is a gene which is known to play also an important role in the production of the gibberelins, therefore controlling important roles in various plant developmental processes, including seed dormancy. Seed dormancy is of course crucial for determining the time of the year suitable for the germination of the seed (and therefore the plant to survive).