# Physalia-course: Population genomics

*Thibault Leroy (thibault.leroy@inrae.fr)*

## Part 5: Genotype-environment associations
## November 29, 2024

### General context

The primary objective of this second practical session is to introduce the use of Genotype-Environment Association (GEA) analysis with populations sampled across a European climate gradient. Before conducting GEA, we will first integrate various layers of information from prior sessions, including bioinformatics, population structure, and the detection of selection footprints (using XtX, a $F_{ST}$-like method that accounts for population structure levels, as implemented in tools like Bayenv or BayPass).
In brief, we will reanalyze data from a study on sessile oak populations, published in Leroy et al. 2020 (New Phytologist). A preprint version of the paper is available on bioRxiv (https://www.biorxiv.org/content/10.1101/584847v1). To provide context, the dataset includes 18 populations of sessile oak: 10 from a latitudinal gradient across Europe (mainly France, with additional samples from Germany (populations 253 and 256) and Ireland (population 124)) and 8 populations from two adjacent valleys in the French Pyrenees (Ossau Valley [O] and Luz Valley [L]) at varying altitudes (100m, 400m, 1200m, and 1600m, labeled as 1, 4, 12, and 16, respectively). For each population, DNA was pooled prior to sequencing (pool-seq strategy). Although we will use pool-seq data today, the same approach can be applied to individual sequencing data if at least a sufficient large number of individuals per population are available (e.g. 15 diploid), as these methods are population-oriented).
A general objective today will be to identify SNPs that show significant associations with the climate gradient, exceeding what we would expect under a neutral selection hypothesis. Easier said than done! We aim to introduce a strategy to address this challenge.

**IMPORTANT: You can choose to use R/RStudio either on your own computer or via the Physalia instance. You can also use a third posibility, by using the online RStudio IDE on the posit website (https://posit.cloud/, require to register for a free account).** Using your own computer is likely the best option, especially if you already have RStudio installed. However, ensure that you can load or install the following R packages based on your R version: poolfstat, ape, ggplot2, reshape, and corrplot. If you encounter issues with your R/RStudio setup or the required packages, you can use R in the associated Conda environment for this specific course: `conda activate Workshop_TL_YB_landscape`

If you use your own computer, use scp to download the files from the "~/Share/Day5_landscape".
If you use the physalia instance, just use cp as usual:
`cp -r ~/Share/Day5_landscape/ ~`

**Earlier steps**

Due to the limited time available in this course and the large volume of raw sequencing data, the initial analysis steps (such as trimming, mapping and "SNP calling") have already been completed. However, note it was based on a similar strategy than introduced Monday for poolseq data (but using bowtie2 rather than bwa-mem2). If you are interested in learning more about these processes, the code is publicly accessible on a GitHub repository (*https://github.com/ThibaultLeroyFr/Intro2PopGenomics*, section 3.3.3).

We will begin our analyses using the synchronized pileup file generated at the end of the poolseq pipeline, a format introduced by Robert Kofler and collaborators in Popoolation2 (Kofler et al. 2011 Bioinformatics).
In this file, each line represents a genomic position, with the first three columns indicating the scaffold name, the position on the scaffold, and the reference allele. The fourth column provides allele counts for the first population, the fifth for the second population, and so forth. For each population, allele counts are displayed in the order: 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 ...
...
```

For the first position, all read counts (101 for population 1 and 130 for population 2) align with a T, suggesting this site is very likely non-variant, at least based on data from the first two populations. In contrast, the counts at the second position suggest a biallelic SNP with an A/C polymorphism. Given

the size of the oak genome (814 Mb; Plomion et al., 2018, Nature Plants), it is today unmanageable to analyze data across the entire genome. Therefore, for the initial three steps, we will use a dataset of approximately 200,000 randomly selected positions from a synchronized pileup spanning the oak genome. For the final three steps, we will focus specifically on data from chromosome 1.

### Step 1: Identify SNPs and compute a pairwise $F_{ST}$ matrix

Launch R (physalia cluster)
`R` OR use Rstudio (your own computer or online Rstudio IDE)

And then under R, load the packages (or install them):

```
library("poolfstat") # if not already installed: install.packages("poolfstat")
library("reshape2")
library("ggplot2")
library("ape")
library(corrplot)
```

Set your working directory (the folder containing the files to be used during the practical, usage: setwd("FULL PATH TO YOUR FOLDER") ):
e.g.

```
setwd("~/Day5_landscape/data-oak")
```

Read the sync file and identify SNPs by specifying the number of haplotypes in each pool and setting a threshold for the minor allele frequency. For detailed information on these parameters, read the 'PoolFstat' vignette: https://cran.r-project.org/web/packages/poolfstat/poolfstat.pdf.

```
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 have you identfied with this piece of R code (poolfstat function)? On average, every how many bases a SNP can be identified in the genome?

Compute a pairwise $F_{ST}$ matrix with compute.pairwiseFST under poolfstat v.2 (note that the function was initially called computePairwiseFSTmatrix poolfstat v1, in case, it is not working check the version of your package, by executing

"packageVersion("poolfstat")" in your R script.

```
pairwise_fst_results <- compute.pairwiseFST(pooldata, method =
"Anova",output.snp.values = FALSE)
Fstmatrix <- pairwise_fst_results@PairwiseFSTmatrix
```

**ALTERNATIVE**: in case the function compute.pairwiseFST does not work properly, import directly the matrix provided as a backup option.

```
Fstmatrix=as.matrix(read.table("Fstmatrix.txt"))
colnames(Fstmatrix)=c("9","97","124","204","217","218",
"219","233","253","256","L1","O1","L8","O8","O12","L12","O16","L16")
```

Show the variation of the averaged $F_{ST}$ among all pairs of populations (populations are in the same order than read by popsync2pooldata):

```
pdf("Fst_matrix_res.pdf",width=10,height=10)
image(Fstmatrix, axes=FALSE,col=ifelse(Fstmatrix<0,
rev(cm.colors(200)),rev(heat.colors(200))))
axis(1, at = seq(0, 1, length = nrow(Fstmatrix)), labels = rownames(Fstmatrix))
axis(2, at = seq(0, 1, length = ncol(Fstmatrix)), labels = colnames(Fstmatrix),las=2)
dev.off()
```

***IMPORTANT****: If you are currently using R in a terminal (e.g. on the Physalia cluster), always use:* `pdf(filename="XXX.pdf",width=10,height=10)` *or similar* `png(filename = "XXX.png", width = 800, height = 800)` *to print a plot in a file. Once the file is saved, you can download it to your own computer using scp to open and view the plots locally. If you use Rstudio on your own computer, you can directly use the code without the "pdf()" and "dev.off()" functions and visualize the output under Rstudio.*

To IMPROVE the visualization, consider plotting the results using ggplot2.

```
melted_Fstmatrix<- melt(Fstmatrix)
```

Warning messages are indeed expected ;)

```
pdf("Fst_matrix_res_ggplot2.pdf",width=11,height=10)
ggplot(data = melted_Fstmatrix, aes(x=X1, y=X2, 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"))
dev.off()
```

The pairwise $F_{ST}$ matrix is informative about the population structure. Which population(s) appear to be the most differentiated from the others?

A convenient way to look at these among-population differences, is to represent this structure as a tree with the R package ape:

```
pdf("Tree-like_Fst_matrix_res_ape.pdf",width=7,height=11)
Fstmatrix2 = Fstmatrix
Fstmatrix2[Fstmatrix2 <0] <- 0  #replace negative values by 0
plot(ladderize(as.phylo(hclust(as.dist(Fstmatrix2)))))
dev.off()
```

The distances shown here are proportional to the difference in average $F_{ST}$ among populations. As mentioned in the script just above, I have replaced the negative $F_{ST}$ values by 0 to built this tree. Although slightly negative values can occur due to corrections for small sample sizes, these can be considered as zero (indicating no differentiation).

Even though we only used a few thousand SNPs in this analysis, the results are already meaningful. In fact, to assess population structure, a few thousand of unlinked neutral SNP markers are generally sufficient.
Regarding the two sampling zones (comprising 10 populations from Western Europe at low elevation and within two French Pyrenees valleys, spanning from low to high elevation), which populations appear to be the most differentiated?

**Step 2: Compute the empirical distribution of $F_{ST}$ values over all populations**

Another important evaluation is to look at the distribution of $F_{ST}$ values across all SNPs.
Using your subset, you can plot this distribution.

```
SNPvalues=computeFST(pooldata, method="Anova")$snp.FST
SNPvalues=as.data.frame(SNPvalues)
pdf("Distribution_Fstloci_allpops.pdf",width=9,height=8)
plot(density(SNPvalues$SNPvalues),xlim=c(0,1),lwd=1.5,col="navyblue",xlab="Fst",main="")
dev.off()
```

The first piece of information you are probably interested in is the $F_{ST}$ value of the most differentiated SNP.

```
print(paste("The greatest level of FST observed is:",max(SNPvalues$SNPvalues)))
```

Now, you are probably curious about which specific marker this corresponds to: (Here is a useful function for quickly retrieving this information from a large dataset)

```
which(SNPvalues$SNPvalues==max(SNPvalues))
```

Now you know the SNP position within the full list, but this information alone is still limited.
You are likely more interested in the exact genomic location of this SNP. Let's retrieve this information.

```
pooldata@snp.info[which(SNPvalues$SNPvalues==max(SNPvalues)),]
```

This SNP shows a higher level of differentiation than any other SNP in the dataset - quite fascinating, isn't it? That said, you are probably more interested in identifying which populations have the most extreme allele frequencies at this SNP. To explore this, you can compare the reference allele counts to the total allele counts for each population.

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

To obtain allele frequency, just do:

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

Which population shows the highest allele frequency for the reference allele, and

which shows the lowest?
(Note that populations are listed in the same order as initially read by the function popsync2pooldata)

Of course, you might also be interested in identifying the SNPs that fall within the top 1% of $F_{ST}$ values, which could provide preliminary evidence for selection.

First, determine the $F_{ST}$ value that represents this top 1% threshold:

```
quantile(SNPvalues$SNPvalues, probs = 0.99, na.rm = TRUE)
```

Then, identify the SNPs that meet or exceed this threshold

```
which(SNPvalues$SNPvalues>quantile(SNPvalues$SNPvalues, probs =
0.99, na.rm = TRUE))
```

Great! But you are likely even more interested in finding the exact genomic locations of these SNPs, no?

```
pooldata@snp.info[which(SNPvalues$SNPvalues>quantile(SNPvalues$SNPvalues,
probs = 0.99, na.rm = TRUE)),]
```

Note: By doing this, you have conducted your first "genome scan" (congratulations) to identify SNPs showing elevated differentiation across all populations. At this stage, this scan is fairly basic, as we are simply selecting SNPs from the upper tail of the $F_{ST}$ distribution of the overall differentation (i.e. across all populations together). However, selection can sometimes act on a single population, making it distinct from all others. In such cases, it is necessary to conduct all possible pairwise comparisons.

**Step 3: Compute the empirical distribution of $F_{ST}$ values over pairs of populations**

First, for each SNP, we can compute the Fst value for all possible pairs of species thanks to the compute.pairwiseFST (=computePairwiseFSTmatrix for poolfstat v.1.2), with the "output.snp.values = TRUE")

```
pairwise_fst_locus_results <- compute.pairwiseFST(pooldata,
method = "Anova",output.snp.values = TRUE)
Fstlocus <- pairwise_fst_locus_results@PairwiseSnpFST
Fstlocus[Fstlocus == "NaN"] <- NA
colnames(Fstlocus) <- gsub(";", "vs", colnames(Fstlocus))
melted_Fstlocus<- melt(Fstlocus)
```

```
colnames(melted_Fstlocus)<- c("index","comparison_pops","Fstvalues")
```

As a BACKUP OPTION, to simply upload the matrix:

```
melted_Fstlocus=read.table("Fstlocus.txt",header=TRUE)
```

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

```
pdf("Distribution_Fstloci_pairs.pdf",width=9,height=8)
plot(density(melted_Fstlocus$Fstvalues[melted_Fstlocus$comparison_pops=="97_vs_253"],
na.rm=TRUE),main="",col="darkgreen",xlim=c(-0.1,1))+
lines(density(melted_Fstlocus$Fstvalues[melted_Fstlocus$comparison_pops=="97_vs_124"],
na.rm=TRUE),main="",col="darkred")
lines(density(melted_Fstlocus$Fstvalues[melted_Fstlocus$comparison_pops=="O1_vs_O16"],
na.rm=TRUE),main="",col="navyblue")
dev.off()
```

Based on the distributions and considering that populations 97, 124, and 256 are located along a longitudinal gradient—from Southern France, Ireland, and Northern Germany, respectively - and that populations O1 and O16 were sampled within the same Pyrenean valley but at different elevations (low to high), does this result align with your expectations? What conclusions can be drawn regarding genetic differentiation in Quercus petraea, and the relative roles of isolation by distance versus environmental factors in shaping this differentiation?

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

```
pdf("Distribution_Fstloci_allpairstogether.pdf",width=14,height=10)
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()
dev.off()
```

*Note: a few warnings are also expected ;)*

Based on this plot, you can see that indeed performed a lot of pairwise comparisons!
OPTIONAL: Can you determine the total number of pairwise comparisons performed?

In order to be able to evaluate how variable are the distributions, the best is to remove the legend by adding "theme(legend.position = "none")"

```
pdf("Distribution_Fstloci_allpairstogether_withoutlegend.pdf",width=14,height=10)
```

```
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")
dev.off()
```

OPTIONAL: Using one or two pairwise comparisons of your choice, try to identify the SNPs exhibiting the 1% highest observed values. How many $F_{ST}$ "outliers" do you observe? Is it surprising?

### Step 4: XtX an analogous to $F_{ST}$, but accounting for the population structure

Note: A major limitation of our previous approaches, which rely on the distribution of $F_{ST}$ is that they do not explicitly account for the neutral processes shaping genome-wide variation in $F_{ST}$. Additionally, by selecting the top 1% of SNPs with the highest $F_{ST}$ values, we implicitly assume that 1% of the genome is under selection. Imagine, however, that all focal populations evolve under strict neutrality: using the top 1% threshold would still identify 1% of SNPs as outliers, which would all be false positives. Ideally, a robust method would detect no outliers in such a fully neutral scenario. Conversely, it is possible that in some datasets, 5% or more of the genome is genuinely under selection (or linked to loci under selection), but by using a 1% threshold, we would detect only the top 1% SNPs, missing a substantial portion of true signals. How can we address this issue?
The ideal approach would involve reconstructing the demographic history of each population (see Wednesday's course) to run neutral simulations under these historical conditions, allowing us to determine the proportion of SNPs that deviate from neutral expectations. In practice, this is challenging. A simpler approach is to compute an adjusted version of $F_{ST}$ that accounts for the variance-covariance structure of allele frequencies among populations (i.e. population structure). Neutral simulations based on this structure can then provide an expected distribution of $F_{ST}$ under strict neutrality, allowing for more accurate detection of outliers.

So first, we need to generate an input file for the Baypass software:
(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=1000,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:

```
pdf("Baypass_omega_matrix.pdf",width=10,height=10)
image(omega_matrix,axes=FALSE,col=heat.colors(200))
dev.off()
```

OPTIONAL: try to generate a single plot containing the results of both the omega matrix and the $F_{ST}$ matrix to check whether the results are consistent

```
pdf("Comparison_Fstmatrix_Baypass_omega_matrix.pdf",width=18,height=10)
par(mfrow=c(2,1))
image(Fstmatrix, axes=FALSE,main="Fst matrix",col=rev(heat.colors(200)))
axis(1, at = seq(0, 1, length = nrow(Fstmatrix)), labels = rownames(Fstmatrix))
axis(2, at = seq(0, 1, length = ncol(Fstmatrix)), labels = colnames(Fstmatrix),las=2)


image(omega_matrix,axes=FALSE,main="omega matrix",col=heat.colors(200))
axis(1, at = seq(0, 1, length = nrow(Fstmatrix)), labels = rownames(Fstmatrix))
axis(2, at = seq(0, 1, length = ncol(Fstmatrix)), labels = colnames(Fstmatrix),las=2)
dev.off()
```

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

```
par(mfrow=c(1,1))
correlation.matrix=cov2cor(omega_matrix)

pdf("Baypass_correlation_matrix_corrplot.pdf",width=10,height=10)
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")
dev.off()
```

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

```
pdf("Baypass_tree-like_omega_matrix_res_ape.pdf",width=7,height=11)
plot(as.phylo(hclust(as.dist(1-correlation.matrix))))
dev.off()
```

OPTIONAL: Try creating a combined plot with both the $F_{ST}$-based tree and the omega matrix-based tree.
Do you consider that these results are globally consistent with the previously observed results based on the pairwise $F_{ST}$ matrix (step 1)?

As you have now understood, this omega matrix is informative about the levels of population structure.
Baypass also generates XtX values, an analog of the $F_{ST}$,which explicitly accounts for the population structure (i.e. the previously inferred omega matrix).

Plot the distribution of the XtX values:

```
pdf("Baypass_distribution_XtX_values.pdf",width=11,height=10)
plot(density(XtX_BF$XtX),xlim=c(0,50),xlab="Observed XtX values",main="")
dev.off()
```

Notice that the XtX values now cover a broader range and are no longer restricted to the interval between 0 and 1!

You may also create a "Manhattan plot" to visualize XtX levels across the first 30 Mb of chromosome 1. Please be patient, there is a lot of data, so drawing the plot may take 1 or 2 minutes!

```
pdf("Baypass_Manhattan_distribution_XtX_values.pdf",width=14,height=10)
plot(XtX_BF$XtX~XtX_BF$position,cex=0.05,pch=20,xlab="Chr1 -
Position",ylab="XtX")
dev.off()
```

Again, please be patient! Does the differentiation appear consistent across the 30 Mb region on chromosome 1?"

**Step 5: Calibrating the XtX values**

Similar to the approach of using the top 1% of the $F_{ST}$ distribution, relying solely on the top 1% of XtX values is not recommended, as it lacks information about the actual proportion of the genome under selection. Instead, a better approach is to use the observed variance-covariance matrix (along with additional files that I will not detail today) to perform neutral simulations. These simulations provide a baseline for neutral expectations, allowing us to set a threshold under strict neutrality.

Here, we will not go into detail on how to perform this step, as it can be quite complex in practice, so we have completed this analysis for you. Briefly, the approach involves using the observed levels of population structure (i.e., the omega matrix) and parameters from prior analyses to generate 'pseudo-observed' datasets—simulated data that reflect the same population structure while assuming strict neutrality. These simulations allow us to establish the expected XtX values under neutrality, creating a neutral distribution that we can use as a threshold to distinguish between markers that fit neutral expectations and those with excess differentiation (i.e., outliers). If you would like to perform similar analyses on your own data, feel free to contact us for more details.

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

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

What are the quantile values? The quantile "0.999" indicates that one simulated neutral SNP per thousand reached an XtX value equal to or greater than this threshold. Similarly, "0.9999" corresponds to one SNP per ten thousand simulations, and "1" represents the highest XtX value observed across all simulations (i.e. this is equivalent to using max(xtx.pods$M_XtX)

Now that we have established a neutral expectation for the distribution of XtX under neutrality, the next step is to use these quantiles as thresholds for the observed XtX values in our real dataset. How many SNPs in our real data exhibit XtX values higher than the maximum XtX value observed in the neutral simulations?

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

What proportion of our SNP set exhibits XtX values that were never observed in the neutral simulations?

```
pdf("Baypass_Manhattan_distribution_XtX_values_withneutralqualibration.pdf",width=14,height=
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],"darkred","darkgrey"))
dev.off()
```
(again here this plot will perhaps require to wait a minute ;) )

OPTIONAL (level 1): Try to modify this code, to show a different color for each threshold (i.e. quantiles).
OPTIONAL (level 2): Try to create a small piece of code (requiring a for or while loop) 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 incorporate population-specific covariates. The main goal is to identify SNPs that covary with environmental variables (e.g. climate data). Briefly, it also works with phenotypes as covariates (i.e. the mean trait value for each population, which corresponds to a population-level association or 'population GWAS' [pGWAS]). In our previous study (Leroy et al. 2020), for example, we used the date of leaf unfolding in common gardens as a covariate.
For today's analysis, we will focus on the local climate of each population, specifically the yearly average temperature between 1950 and 2000. The idea is to compare the likelihood of two models: one that includes the potential "fixed" environmental effect and one without it (i.e., models with vs. without association). The comparison of these two likelihoods results in a Bayes Factor (BF) for each SNP, with higher BF values supporting the model that includes the environmental association (i.e., allele frequencies correlate with the temperature gradient). A BF greater than 15 is generally considered strong support, while a BF greater than 20 is considered decisive. We will apply this widely accepted criterion today. However, it should be noted that BF calibration using neutral simulations (similar to the approach we used for XtX) is also possible (see the OPTIONAL exercise below).

In this study, we are particularly interested in SNPs that exhibit significant allele frequency differences among populations (i.e. high XtX) and show an association with local mean temperatures (i.e. adaptation to climate). To visualize this, we will generate a scatterplot showing both XtX and BF values, applying thresholds for both metrics. The most interesting markers will be those with both high XtX and high BF values.

```
pdf("Baypass_scatterplot_XtX_BF_values_withneutralqualibration.pdf",width=10,height=10)
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")
dev.off()
```

Now, you might be interested in seeing where the highly-differentiated SNPs showing a strong association with climate (temperature) are located along the chromosome 1. This can be easily done by generating a Manhattan plot of the XtX values, highlighting the associated SNPs with a distinct color.

```
pdf("Baypass_Manhattan_XtX_BF_values_withneutralqualibration.pdf",width=14,height=10)
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")))
dev.off()
```
(again here, another minute lost...!)

In which region(s) of chromosome 1 do you observe the highest concentration of associated SNPs?
In our previous study (Leroy et al. 2020), we identified candidate genes in these regions and hypothesized that the genes responsible for these associations might be SCD1 or ASPG1 (or both). SCD1 is known to be involved in stomatal responses to water stress, making it a plausible candidate for temperature-related associations. ASPG1, on the other hand, plays a key role in the production of gibberellins, which are crucial for various plant developmental processes, including seed dormancy. Seed dormancy is critical for determining the optimal timing of seed germination and, consequently, plant survival.

OPTIONAL (level 1): Now, repeat the analysis to identify associations, this time incorporating threshold values for the Bayes Factor (BF) based on the PODS. Use the same methodology as previously applied for XtX, considering both XtX and BF thresholds derived from simulations under the assumption of strict neutrality.
OPTIONAL (level 2): If you have still time, try to write a function to detect

local density in XtX outliers using sliding windows, in order to find more evidence for local enrichment.