BayPass 2.3: tutoriel de génomique d'association adapté au séquençage en pool

Jérôme OLIVARES[1]

2023-09-19

[1]INRAE, UR-1115 PSH, 228 route de l'aérodrome, 84914 Avignon, France

Contents

Résumé

Ce tutoriel détaille d'une part la manière de générer les fichiers d'entrées du logiciel BayPass à partir de données de séquençage en pool, d'autre part le paramétrage optimal du logiciel BayPass et en fin propose une méthode d'exploreration des résultats d'analyses produits. Un pipeline d'analyse mixant des packages sous Rstudio et des lignes de commandes Linux, est décrit afin de guider pas à pas l'utilisateur tout au long du processus depuis les données brutes jusqu'à la liste finale de loci/variants candidats. Ce tutoriel est à destination des étudiants et des bioinformaticiens débutants.

mots clefs

Logiciel BayPass, séquençage en pool, GWAS, études d'associations pangénomiques, Rstudio.

prérequis

Les commandes décrites dans cet article ont été regroupées dans un fichier au format « R markdown » (Rmd) « Poolseq_pipeline.Rmd » librement téléchargeable à l'adresse : https://github.com/Jolivares-INRAE/Download. Ce tutoriel est conçu pour décrire pas à pas les différentes étapes du fichier Rmd et permettre à l'utilisateur de les exécuter en parallèle. L'utilisateur devra avoir une connaissance basique du logiciel Rstudio et être capable d'écrire et lancer des scripts sur un cluster de calcul. Les commandes ont été rédigées sous Rstudio version 1.4.1106 couplé à R 64 bits version 4.0.5. avec toutes les librairies nécessaires à jour (Capture 1) et dans l'environnement bash/SLURM des clusters de calculs de la plateforme GenoToul de bioinformatique (GenoToul Bioinfo). Dans le cas d'une utilisation dans un autre environnement logiciel, l'utilisateur devra probablement effectuer des adaptations du code. Bien que l'essentiel des calculs de BayPass seront réalisés sur un cluster de calcul, certains de ses utilitaires seront utilisés en local sous Rstudio, la dernière version du logiciel sera donc téléchargée depuis l'adresse https://forgemia.inra.fr/mathieu.gautier/baypass_public et décompressée dans un répertoire local par l'utilisateur. Dans tous les codes qui suivent l'expression «/../» sera à remplacer par les chemins personnels de l'utilisateur. Le terme de chromosome sera utilisé en références aux appellations de contigs, scaffold, ou chromosomes qui correspondent aux séquences nucléotidiques du génome de référence, plus ou moins mature, qui sera utilisé.

Introduction

Dans un contexte agronomique actuel de réduction de l'utilisation des pesticides ou de réchauffement climatique, analyser et comprendre les bases génétiques de l'adaptation des organismes aux méthodes de luttes qui leurs sont opposées ou à l'évolution de leur environnement est un enjeu majeur des recherches de ces dernières années.

Les études d'associations pangénomiques (GWAS en anglais pour genome-wide association study) adossées aux techniques de séquençage haut débit (NGS) permettant le séquençage de génome complet, sont un outil de choix pour ce type d'analyse. L'étude au niveau populationnel demandant le séquençage d'un grand nombre d'individus, les couts d'analyses étaient initialement très élevés et ces études étaient souvent réservées aux organismes dit "modèles", humain en tête. Néanmoins il a été démontré depuis, que le séquençage en pool d'individus (poolseq) c'est-à-dire en mélangeant de manière équimolaire l'ADN d'un grand nombre d'individus (50 à 100) issus d'une même population permettait non seulement de réduire drastiquement les couts puisqu'on ne réalise qu'un seul séquençage mais aussi que la découverte des points de mutations (SNP) et l'estimation de leur fréquence allélique étaient souvent plus efficaces et précises [Futschik and Schlötterer, 2010]. Parmi les logiciels à même d'analyser ces fréquences alléliques on compte Baypass [Gautier, 2015] qui évalue par une approche baysienne, la différenciation des SNP en liaison avec une covariable environnementale en tenant compte de la structure et de la parenté entre les populations en estimant la covariance (Ω) des fréquences alléliques. Baypass 2.3 a la particularité supplémentaire de pouvoir calculer un contraste des fréquences alléliques entre deux groupes de populations caractérisés par un caractère binaire, sensible ou résistant par exemple.

La documentation disponible de BayPass 2.3 décrit par le menu les algorithmes de fonctionnement et les différents paramètres du logiciel mais reste néanmoins, et assez logiquement, succincte sur les étapes en amont et en aval. A ma connaissance un seul tutoriel est disponible sur le web [Nielsen, 2020] et décrit de manière plus détaillée la préparation des données brutes et l'analyse en mode « poolseq » de Baypass, mais il nécessite des bases quelque peu avancées de codage sous R et en environnement bash. Si les bio-informaticiens chevronnés ne rencontreront pas de difficultés particulières, il n'en va pas forcément de même pour bon nombre d'agents que les orientations des recherches associées à la baisse des coûts de séquençage ont aiguillé vers les voies de la génomique. Cet article a pour but de détailler les différentes étapes d'une analyse de type GWAS/poolseq avec le logiciel BayPass 2.3 et d'éclairer les points qui sont habituellement peu explicités car considérés comme évident.

Présentation du logiciel BayPass 2.3

Le logiciel BayPass est un logiciel de génomique des populations qui vise principalement à l'identification de marqueurs génétiques soumis à la sélection et/ou associés à des covariables spécifiques à la population (variables environnementales, phénotypiques, quantitatives, catégorielles…). Par une approche baysienne il évalue une **matrice Ω** de covariance des fréquences alléliques des populations résultant de leur histoire démographique. Deux manières d'estimer ces fréquences alléliques sont disponibles soit en se basant sur les génotypes référence/mutant des individus analysés soit, lorsque l'on active le « pool-seq mode », ces fréquences alléliques sont calculées en regard de la profondeur de séquençage (reads count) et pondérées par le nombre d'individus qui ont contribué à cette profondeur. C'est cette seconde approche que nous considérerons dans cet ouvrage.

BayPass propose 3 modèles statistiques d'analyse :

Le Core Model

C'est le modèle de base, il permet de calculer la matrice de covariance Ω et d'attribuer une statistique de différenciation ***XtX*** à chaque SNP, et ainsi de scanner le génome pour identifier les régions génomiques différenciées entre les populations.Le *XtX* est défini comme la variance des fréquences alléliques standardisées du SNP dans les population, c'est une statistique analogue au Fst mais tient compte de la co-évolution des populations grace à la matrice Ω.

Le Standard Model

Ce modèle, permet, l'orsque l'on fournit une ou plusieurs covariables (environnementales, phénotypiques…), de calculer un facteur de Bayes, ou Bayes factor en anglais, (BF) pour chaque marqueur génétique représentant la force d'association avec une covariable. C'est un modèle tout en un qui intègre le calcul des XtX et de la matrice Ω, il est adapté au faible nombre de population (< 15).

L'Auxiliairy Model

Ce modèle a une approche différente dans le calcul de la statistique BF, sans entrer dans le détail il est plus adapté au grand nombre de population (> 15), En contrepartie il nécessite que l'on fournisse une matrice Ω déjà calculée par une analyse Core Model précédente, il recalcule alors les XtX et la statistique BF.

Ces covariables évoquées doivent être distribuées en gradient entre les populations (différence de température, d'altitude…), en complément, dans le cas où la covariable étudiée serait purement qualitative ou binaire (sensible/résistant, gros/petit…), les modèles Standard et Auxiliaire peuvent calculer une statistique C2 qui évalue le contraste de différence des fréquences alléliques de chaque marqueur entre 2 groupes de populations.

Présentation générale de l'analyse :

La Figure 1 est une vision simplifiée des différentes étapes nécessaire à l'analyses de données poolseq. Les étapes nécessittant une importante puissance de calcul comme le variant calling ou l'analyse

BayPass se déroulent soit dans l'environnement Linux du cluster de calcul, les étapes de filtrage, manipulation de données et de résultats se font sur ordinateur local sous Rstudio. La première étape part des fichiers d'alignement au format « ***bam*** » de chaque population à analyser et consiste à effectuer une recherche de variants (variant calling) pour obtenir un fichier au format « ****vcf*** » regroupant tous les points de mutations ou SNP de toutes les populations qui sont autant de marqueurs génétiques à analyser. Ce fichier ***vcf*** sert d'entrée au package « PoolFstat » [Gautier et al., 2022] qui va permettre de filtrer les SNPs à analyser et générer les fichiers nécessaires au bon fonctionnement de BayPass mais aussi de faire une première analyse des Fst entre populations par exemple. Ces fichiers d'entrées pouvant contenir plusieurs millions de SNP, ils sont découpés en plusieurs dizaines de sous jeux de données (sub sampling) afin de réduire les temps de calculs. Une fois que BayPass a analysé tous les sous jeux de données, l'homogénéité des résultats entre eux est analysée sous Rstudio puis les résultats peuvent être regroupés, filtrés et analysés afin de déterminer les marqueurs génétiques et les régions chromosomiques d'intérêts qui seront visualisées par différents plots.

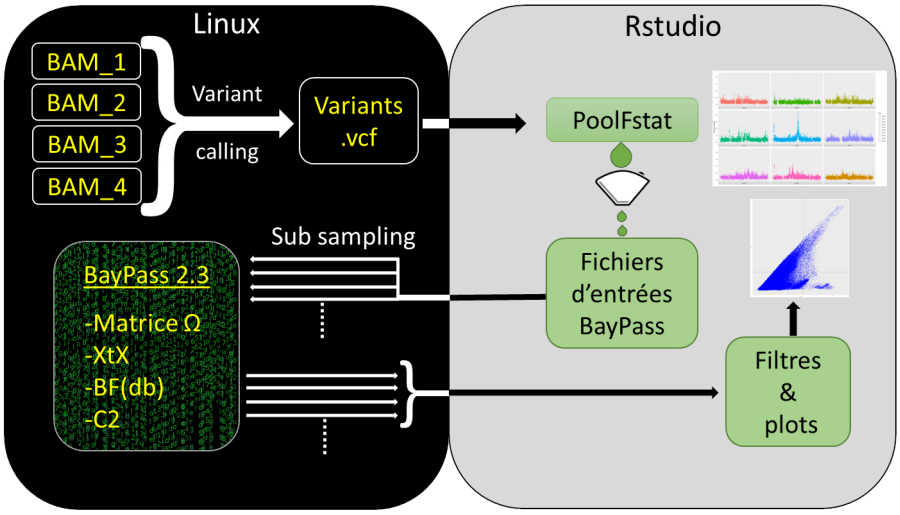


Figure 1: Pipeline d'analyse BayPass

Le pipeline d'analyse est décomposé en plusieurs étapes se déroulant soit en environnement Linux pour celles nécessitant une importante capacité de calcul, soit sous Rstudio pour le filtrage et l'analyse des résultats.

Données brutes

Obtention d'un fichier Poolseq.vcf :

Les étapes de contrôle qualité et d'alignement des données de séquençage sont largement documentés par ailleurs, et ne serons donc pas documenter ici, et nous partirons directement des fichiers d'alignement ***bam***. La première étape consiste à regrouper les fichiers ***bam*** de toutes les populations en un seul fichier puis à effectuer le variant calling avec un logiciel dédié compatible avec le séquençage en pool afin de conserver les informations de profondeur. Nous recommandons l'utilisation des Samtools [Li et al., 2009] et de Varscan 2 [Koboldt et al., 2012] avec une instruction pipe entre les deux pour éviter les fichiers intermédiaires et économiser l'espace de travail (Script 0.1). Les commandes sont effectuées avec les paramètres de base, sauf la p-value qui est montée à 0.5 pour être le moins stringent possible à ce stade. On peut découper le travail en plusieurs chromosomes pour réduire les temps de calculs.

Script 0.1. Exemple d'un script bash pour effectuer un variant calling

Si l'on veut découper le travail en chromosomes, il est indispensable de travailler sur des fichiers bam correctement indexés et d'utiliser l'option -r/–region qui tire profit de cet index.

```
#!/bin/bash
#SBATCH --array=1-29            #création de l'array: un élément/indice par chromosome

module load bioinfo/samtools/1.12
module load bioinfo/VarScan/2.4.2
module load bioinfo/bcftools/1.14

#liste tous les fichiers "bam" et leur chemin
ls /../*.bam > BamList.txt
#extrait les noms des échantillons/populations
ls /../*.bam | sed -r 's/^.+\///' | sed -r 's/.bam//' > NameList.txt

samtools mpileup -C 50 -d 5000 -q 20 \
-r chr${SLURM_ARRAY_TASK_ID} \    #attribut un chr à chaque indice de l'array : chr1, chr2 ...
-f /../ref_genome.fas -b /../BamList.txt | \
java -Xmx2G -jar $VARSCAN mpileup2cns \
--variants --min-coverage 10 \
--min-avg-qual 20 --min-var-freq 0.05 \
--p-value 0.5 --output-vcf 1 \
--vcf-sample-list NameList.txt > /../project_chr${SLURM_ARRAY_TASK_ID}.vcf

bgzip /../project_chr${SLURM_ARRAY_TASK_ID}.vcf
bcftools index /../project_chr${SLURM_ARRAY_TASK_ID}.vcf
```

A l'issue du variant calling les fichiers ***vcf*** des autosomes seront concaténés (Script 0.2), les chromosomes sexuels ayant une ségrégation, une histoire démographique et une ploïdie différente des autosomes il conviendra, lorsque cela est possible, de les analyser à part.

Script 0.2. Exemple d'un script bash pour concaténer des fichiers ***vcf***

On peut utiliser les bcftools pour concaténer les fichier ***vcf***, la liste des fichiers à traiter est contenue dans une variable alimentée par une boucle “for”.

```
#!/bin/bash
module load bioinfo/bcftools/1.14
my_path='/../'

# une boucle "for" itére sur les numéros des autosomes de 2 à 28
# et ajoute chaque fichier vcf et son chemin dans une variable
for ((i=2; i<=28; i++))
do
    files+=" $my_path/project_chr${i}.vcf.gz"
done

# bcftools concatène la liste des vcf contenu dans la variable $file
bcftools concat $files -O z -o $my_path/project_chr2-28.vcf.gz
```

Filtrage des données brutes

Le fichier ***vcf***, éventuellement compressé, est téléchargé sur un ordinateur local pour être analysé sous Rstudio. Dans un premier temps il faut charger les packages nécessaires et définir les chemins des différents dossiers qui seront utilisés (Chunk 0.1)

Chunk 0.1. Chargement des packages et chemins

```
library(poolfstat)      #filtrage du vcf et création des fichiers d'entrées BayPass
library(RColorBrewer)   #gestion couleur des heatmaps
library(mixOmics)       #"cim" pour heatmap
library(corrplot)       #matrice de corrélation pour heat map
library(VennDiagram)    #création de diagramme de Venn
library(nVennR)         #extraction de données d'un diagramme de Venn
library(tidyverse)      #manipulation de données (dplyr) et plots (ggplot2)

#définition des chemins de travail
path_vcf <- "/../vcf/"
path_input <- "/../Input/"
path_out <- "/../Output/"
path_res <- "/../Resultats/"
source("/../baypass_utils.R")   #fonctions utilitaires de BayPass
```

Le filtrage du ***vcf*** et la sélection des SNPs à analyser ce fait à l’aide du package PoolFstat (Chunk 0.2). Dans un premier temps il faut renseigner les noms des populations dans un objet ***pnames*** et les tailles haploïdes de chaque population dans un objet ***psizes***. Pour les organismes diploïdes, le nombre total de copies des autosomes sera deux fois le nombre d’individus dans le pool, pour les gonosomes Y ou W la ploïdie sera égale au nombre d’individus XY/ZW et pour les gonosomes X ou Z le calcul sera (nombre de XX/ZZ * 2) + (nombre de XY/ZW * 1). La fonction ***vcf2poodata*** va balayer le fichier ***vcf***, sélectionner les SNP bi-alléliques selon des critères définis par l’utilisateur et créer un objet ***pooldata***.

Chunk 0.2. Filtrage et sélection des SNPs

Les options à renseigner sont: min.cov.per.pool = Si au moins un pool n’est pas couvert par au moins au moins min.cov.perpool reads, le SNP est rejeté. max.cov.per.pool = Si au moins un pool est couvert par plus de que max.cov.perpool reads, le SNP est rejeté. min.maf = fréquence allélique minimale autorisée pour l’allèle minoritaire pour qu’un SNP soit retenu.

```
#Infos sur les pops
pnames <- c("pop1S", "pop2S", "pop3S", "pop1R", "pop2R", "pop3R")
psizes_A <- c('114', '180', '160', '180', '200', '120')   #ploydie autosome
psizes_X <- c('86', '135', '120', '135', '150', '90')     #ploydie X/Z
psizes_Y <- c('28', '45', '40', '45', '50', '30')         #ploydie Y/W
#conversion du .vcf
pooldata <- vcf2pooldata(vcf.file = paste(path_vcf, "test_data.vcf.gz", sep=""),
                         poolsizes = psizes_A,
                         poolnames = pnames,
                         min.cov.per.pool = 4,
                         max.cov.per.pool = 1e+06,
                         min.maf = 0.05,
                         remove.indels = FALSE,
                         nlines.per.readblock = 1e+06)
```

```
## Reading Header lines
## VarScan like format detected for allele count data:
##  the AD field contains allele depth
## for the alternate allele and RD field for the reference allele
## (N.B., positions with more than one alternate allele will be ignored)
## Parsing allele counts
## 1e+06  lines processed in 0 h  0 m  20 s : 577119 SNPs found
## 1810324  lines processed in 0 h  0 m  34 s : 1066980 SNPs found
## Data consists of 1066980 SNPs for 6 Pools
```

```
#élimine le 1% supérieur considéré comme trop fortement couvert
#(région très dupliquée, biais de séquençage...)
pooldata<-pooldata.subset(pooldata, cov.qthres.per.pool = c(0,0.99))
```

```
## Data consists of 1041408 SNPs for 6 Pools
```

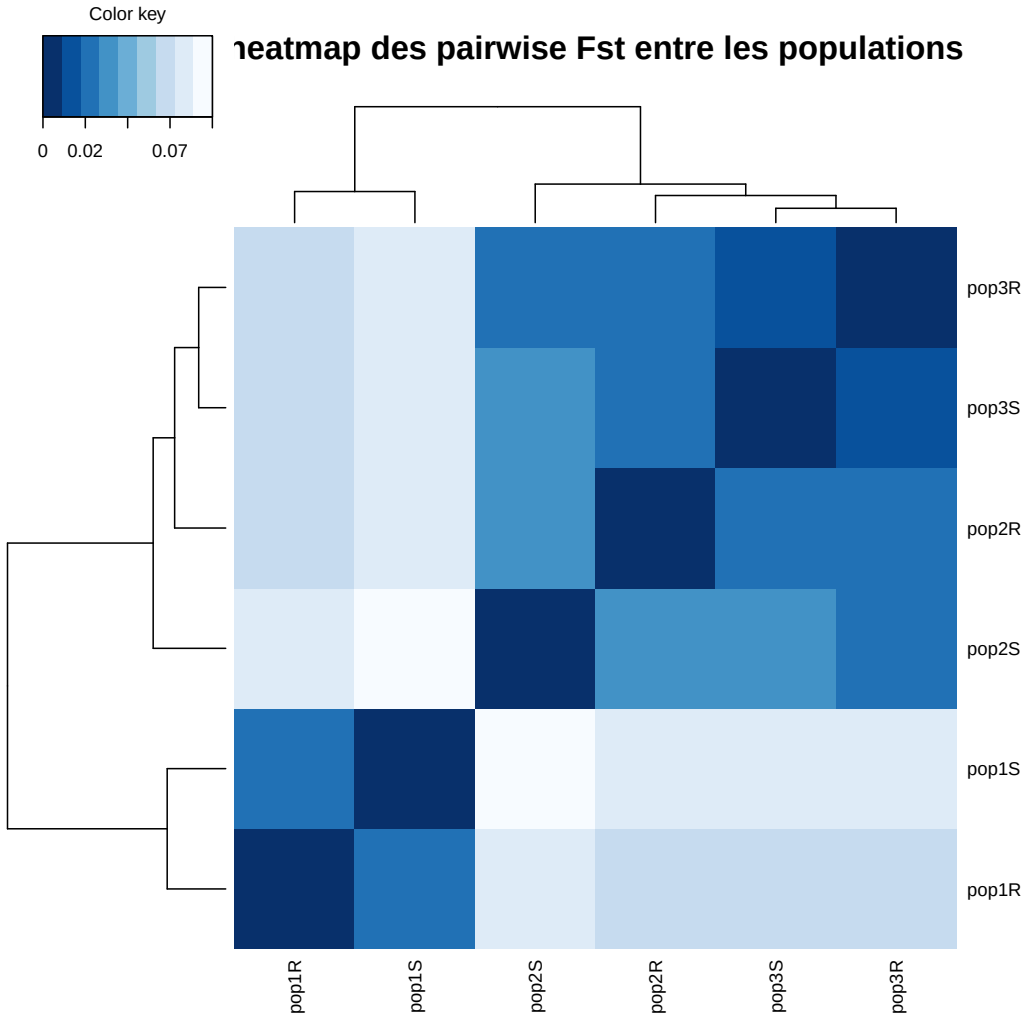
A ce stade cet objet ***pooldata*** peut être utilisé pour calculer diverses statistiques utilisées dans les études de génomique des populations, ces outils sont décrits et exemplifiés dans la vignette de ***poolfstat*** : (https://cran.r-project.org/web/packages/poolfstat/vignettes/vignette.pdf), parmis ceux-ci on trouve l'analyse des Fst entre les populations deux à deux (pairwise) afin de visualiser la proximité génétique entre populations(Chunk 0.3).

Chunk 0.3. Pairwise Fst

```
#Calcul des pairwise Fst
PairWise.fst <- compute.pairwiseFST(pooldata,
                                    method = "Anova",
                                    min.cov.per.pool = 4,
                                    max.cov.per.pool = 1e+06,
                                    min.maf = 0.05,
                                    output.snp.values = FALSE,
                                    verbose = FALSE)
```

```
##
## Overall Analysis Time: 0 h 0 m 6 s
```

```
#conversion en matrice de distance
df <- as.matrix(dist(t(PairWise.fst@PairwiseFSTmatrix)))
#heatmap
cim_color <- colorRampPalette(rev(brewer.pal(9, "Blues")))(9)
cim(df, color = cim_color, symkey = FALSE, margins = c(10, 10),  title = "heatmap des pairwise Fst en
```



un exemple d'estimation de la structure génétique des populations déduite des Fst est donné en Annexe 1

Conversion du pooldata en fichiers d'entrées pour BayPass

La fonction ***pooldata2genobaypass*** (Chunk 0.4) convertit l'ogbjet ***pooldata*** en fichier d'entrée pour BayPass:

Chunk 0.4. Création des fichiers input de BayPass

```
pooldata2genobaypass(pooldata, writing.dir = path_input, subsamplesize = -1, subsamplingmethod = "thi
```

On récupère dans le répertoire ***path_input*** trois fichiers: un fichier "***genobaypass***" qui contient les données filtrées de génotypage, un fichier "***snpdet***" qui contient la liste des positions correspondantes et un fichier "***poolsize***", qui est une copie de l'objet ***psizes***.

Design de l'analyse BayPass

Deux fichiers de paramétrages au format texte/tabulation peuvent être créés en fonction de l'analyse envisagé: Le premier fichier "***ecotype***" (Table 0.1) identifie les covariables quantitatives écologiques de chaque population, avec une covariable par ligne et autant de lignes que de covariables à analyser. Seules sont acceptées les valeurs numériques graduées (taille, poids, mortalité …).

Table 0.1. Example de constitution d'un fichier ***ecotype***

Chaque ligne correspond à une covariable quantitative, chaque colonne correspond à une population.

25

48.5

96.2

17

15.3

…

4.75

0.53

0.27

-0.86

-0.61

…

43.90

44.37

44.29

47.36

47.86

…

Le deuxième fichier ***contraste*** (Table 0.2)contient les covariables qualitatives (petit, chaud, resistant…), il permet d'identifier l'appartenance de chaque population à un groupe qualitatif soit référence (-1), un groupe candidat (1) ou aucun des deux groupes (0). Une ligne par combinaison de contraste.

Table 0.2. Example de constitution d'un fichier ***contraste***

Chaque ligne correspond à une analyse binaire de comparaison des fréquences alléliques de 2 groupes de population. Sur la ligne 1 le groupe composé des population 1 et 2 est comparé au groupe des populations 3,4 et 5. Sur la ligne 2 le groupe composé des population 1 et 2 est comparé au groupe des populations 4 et 5, la population 3 est exclue de l'analyse. Etc…

1

1

-1

-1

-1

…

1

1

0

-1

-1

…

0

1

-1

-1

0

…

Ces fichiers seront aussi transférés sur le cluster de calcul.

IMPORTANT : Les analyses BayPass sont relativement longues (plusieurs heures) mais l'analyses conjointes de plusieurs covariables a un impact relativement faible sur le temps de calcul final, dès lors il est très rentable de multiplier les analyses de covariables en ajoutant autant de lignes que nécessaires dans ces fichiers de paramétrages plutôt que de relancer une analyse complète pour chacune d'entre elles.

Subsampling

Copier les fichiers ***genobaypass, snpdet, poolsize, contraste, ecotype*** sur le cluster de calcul. La découpe en sous jeux de données des fichier genobaypass et snpdet se fait sous Bash avec une commande sed par exemple (Script 0.3).

Script 0.3. Exemple d'une commande de subsampling.

La commande ***sed*** prélève une ligne toutes les 10 et les copies dans un fichier ***.sub*** et s'exécute 10 fois pour balayer tous le fichier initial et créer ainsi 10 fichiers de sous-jeux de données.

```
for i in {1..10}; do sed -n "$i~10p" genobaypass > genobaypass.sub$i; sed -n "$i~10p" snpdet > snpdet
```

Le nombre de fichiers ***.sub*** à créer dépend du nombre total de SNP à analyser, une taille finale de 80 000 à 120 000 SNPs par fichier de sous-jeux de données est un bon compromis entre qualité d'analyse et temps de calcul.

Baypass : l'analyse poolseq

La commande est la même pour les différents modèles employés par BayPass (Script 0.4), ce sont les fichiers optionnels qui définissent le modèle : l'option poolsize active le mode Pool-Seq, pour le core model, seuls les fichiers genobaypass et snpdet sont nécessaires, on peut néanmoins faire une analyse de la statistique de contraste C2 avec le fichier de covariable qualitative ***contraste***. Le modèle standard est activé par l'option ***-efile***, le fichier de covariable quantitative ***ecotype***.

Il est recommandé dans le cadre d'une analyse Pool-Seq d'utiliser et fixer le paramètre -d0yij à 1/5e de la valeur la plus faible du poolsize. (voir page 22 et 39 du manuel de BayPass). Tous les fichiers de résultats cibleront le dossier dans lequel se trouve le script.

Script 0.4. Exemple d'un script bash pour lancer une analyse BayPass

Les sous-jeux de données sont analysés en parallèle

```
#!/bin/bash
#SBATCH --array=0-99        #création de l'array: un élément/indice par job prévu (indice base 0)
#SBATCH --cpus-per-task=2   #nbr de core par job
#SBATCH --mem-per-cpu=2G    #mémoire partagée par tous les cores

module purge
module load statistics/R/4.2.2
module load bioinfo/BayPass/2.4
module load compilers/intel/2023.0.0    #pour i_baypass


#définition du chemin ciblant les fichiers genobaypass et snpdet
my_path='/../Input/'
FILES_R1=($(ls $my_path/genobaypass.sub* | sed -r 's/^.+\///'))
INPUT_F1=${FILES_R1[$SLURM_ARRAY_TASK_ID]}
OUTPUT=${FILES_R1[$SLURM_ARRAY_TASK_ID]/genobaypass/project.output}

i_baypass \
-gfile $my_path/$INPUT_F1 \                  #fichiers genobaypass.sub*
-poolsizefile $my_path/poolsize \            #active le mode poolseq
-d0yij 40 \                                 #1/5e de la valeur la plus faible du poolsize
-seed 5001 \
-nthreads 2 \
-outprefix $OUTPUT \
```

Pour le modèle standard, il faut rajouter au script précédent le fichier contenant les covariables quantitatives:

```
-efile $my_path/ecotype \                    #calcul des Bayes Factors: modèle standard & auxiliaire
-contrastfile $my_path/contrast \            #calcul du contraste C2: modèle core, standard & auxilia
```

Pour le modèle auxiliaire, il faut rajouter au script précédent en plus des fichiers covariables:

```
-auxmodel \                              #active le modèle auxiliaire
-omegafile $my_path/omega.mat \          #matrice Ω modèle standard (optionnel) & auxiliaire (obliga
-auxPbetaprior 0.02 1.98                 #ajuste le Pi beta prior
```

Baypass : les résultats

Validation des données analysées

Chaque sous jeu de données analysé va produire 8 fichiers de résultats avec des extensions différentes, un premier contrôle visuel utile est de vérifier que tous les fichiers partageant la même extension soient de taille identique en kilo ou méga-octets. Des différences manifestes sont signes de problèmes lors de l'analyse (crash, disque plein…) conduisant à des fichiers incomplets. Ceux qui vont focaliser notre attention en premier lieu sont les fichiers ***mat_omega.out*** qui contiennent la matrice Omega (Ω) de covariance des fréquences alléliques des populations qui est calculée à chaque analyse. Avant de regrouper les résultats, il faut s'assurer que chaque fichier du sous jeu de données a été analysé de la même manière que tous les autres, en comparant les matrices Ω entre elles. Cette comparaison se fait en évaluant un indice de distance FMD (Förstner & Moonen, 2003) entre matrices ; plus la distance sera faible (idéalement inférieure à 1) plus les matrices, donc les analyses, seront comparables.(Chunk 0.5)

Chunk 0.5. Comparaison des matrices Ω

Les fichiers de matrice sont copiés dans un dossier local puis comparés deux à deux par la fonction ***fmd.dist*** des utilitaires de BayPass pour évaluer leur distances. Une heatmap et une décomposition en valeurs singulières de la dernière matrice sont alors affichées.

```
source("/../baypass_utils.R")
prefix <- "project.sub"
#liste et compte les matrices Ω du répertoire "path_Omega"
path_Omega <- "/../Omega_files/"
listMatrix <- list.files(path_Omega, pattern="mat_omega.out")
nMatrix<-length(listMatrix)
cat("Nbr matrix files =", nMatrix, "\n")

#boucle sur toutes les matrices, calcule les distances FMD en pairwise et stocke le résultat
ListFMD<-c()
for (i in 1:nMatrix) for (j in 1:nMatrix) if(i!=j) {
omegaA=as.matrix(read.table(paste(path_Omega, prefix, i,"_mat_omega.out", sep="")))
omegaB=as.matrix(read.table(paste(path_Omega, prefix, j,"_mat_omega.out", sep="")))
FMD <- fmd.dist(omegaA, omegaB)
ListFMD <- c(ListFMD,FMD)
}

#calcule la moyenne et la sd de toutes les distances FMD
cat("FMD mean =", mean(ListFMD), "\n")
cat("FMD sd =" , sd(ListFMD), "\n")

#heatmap de la dernière matrice Ω.
colnames(omegaB) <-c(pnames)
rownames(omegaB) <-c(pnames)
cor.mat=cov2cor(omegaB)
#heatmap des liens génétiques entre population
```

```
cim_color <- colorRampPalette(rev(brewer.pal(9, "Blues")))(16)
cim(cor.mat, color = cim_color, symkey = FALSE, margins = c(10, 10), title = "Correlation map based c
#SVD de la dernière matrice Ω.
SVD_omega<-plot.omega(omega=omegaB, pop.names=pnames, main = expression("Singular Value Decomposition
SVD_omega
```

Une fois la moyenne des FMD calculées il est possible de visualiser les liens génétiques entre populations représentés par la matrice Ω par un plot en heatmap ainsi que par la décomposition en valeurs singulières (SVD) d'une matrice prise au hasard.Un exemple de script pour convertir une SVD en plot ggplot2 beaucoup plus maniable et paramétrable pour une publication par exemple est donné en Annexe 2 Cette matrice SVD représente les distances génétiques des populations analysées.

Concaténation des résultats

Une fois la reproductibilité des analyses vérifiées, il faut concaténer tous les fichiers de résultats, l'idée étant de faire le lien entre chaque fichier de résultat et son fichier de position ***snpdet.sub*** correspondant puis de les concaténer les uns aux autres. Les fichiers avec l'extension **___summary_pi_xtx.out_** contiennent les resultats liés à la statistique de differenciation ***XtX***, le script (Script 0.5) est un exemple de script bash adapté à ce type de fichier.

Script 0.5. Exemple d'un script bash pour concaténer les résultats *XtX*.

Un boucle permet de coller chaque fichier résultat à son fichier snpdet correspondant(même numéro de subset "i"), le fichier obtenu est trié et l'entête reconstruite.

```
Snpdet_path='/../BayPass/Input/'
My_prefix='project_STD.sub'

#on boucle sur tous les sous fichiers
for i in {1..5}
do
file1=$(echo "$Snpdet_path/snpdet.sub""$i""")
file2=$(echo "$My_prefix""$i""_summary_pi_xtx.out")
sed -e "s/[[:space:]]\+/ /g" $file1 > SubSNP.x                          #on remplace tous les séparat
sed -e "s/[[:space:]]\+/ /g" $file2 | grep -v "MRK" - > SubData.x                  #on prend tout sa
paste SubSNP.x SubData.x - >> tmp.xtx.merged                          #on colle/incrémente dans un nou
done
#on tri par chromosome et position
awk '{for (i=1;i<=NF;i++) if ($i+0 == $i && $i ~ /e/) $2 = sprintf("%.0f", $i)} 1' tmp.xtx.merged | s

#on supprime les fichiers tmp
rm tmp.xtx.merged
rm SubSNP.x
rm SubData.x

#on insère l'entête du fichier
sed -i '1i chr pos All1 All2 MRK M_P SD_P M_XtX SD_XtX XtXst log10(1/pval)' 8pops_M-chr1-Core.xtx
```

Les fichiers avec l'extension **___summary_betai_reg.out_** contiennent les résultats liés aux Bayes Factor, ceux avec l'extension **___summary_contrast.out_** contiennent les résultats liés à la statistique de contraste C2. Le script pour concaténer ces résultats est très similaire, une boucle est rajouté pour répéter l'opération autant de fois qu'il y a de covariables ou de combinaisons de contraste analysés, un exemple de script est disponible en Annexe 3.

Exploitation des résultats

Une fois fois regroupés les résultats sont téléchargés sur un ordinateur local, la visualisation et les analyses complémentaires sont effectuées sous Rstudio.

Les résultats de différentiation *XtX*:

Les colonnes ***M_P*** et ***SD_P*** correspondent, pour simplifier et pour chaque SNP, à la moyenne de la fréquence de l'allèle référence dans toutes les populations analysées et sa standard déviation. A noter que les fichiers avec les extensions ***yij_pij.out*** contiennent le détail de ces fréquences alléliques corrigées de l'allèle référence pour chacune des populations analysées. La colonne ***M_xtx*** correspond aux valeurs de la statistique *XtX*, la colonne ***XtXst*** est une version recalibrée du *XtX* et ***log10(1/pval)*** sa p.value. Il est à noté qu'une valeur faible de *XtX* associée à un *log10(1/pval)* élevé est signe de sélection balancée, une valeur élevé de *XtX* et de *log10(1/pval)* est un signe de sélection positive.

Une dernière vérification consiste à vérifier la distribution des p.values associées aux *XtXst* avec le script (Chunk 0.6)

Chunk 0.6. Distribution des p.values

L'histogramme de la colonne *log10(1/pval)* du fichier de résultat permet d'avoir une idée immédiate de la distribution normale ou non des p.values de l'analyse.

```
XtX.res=read.table(paste0(path_out,"project.XtX"),h=T)
hist(10**(-1*XtX.res$log10.1.pval.),freq=F,breaks=50)
abline(h=1)
```

Une explication de comment interpréter cet histogramme de distribution est disponible à l'adresse: http://varianceexplained.org/statistics/interpreting-pvalue-histogram/ Si cette distribution n'est pas normale il est souhaitable de calibrer les résultats avec un jeu de données simulées POD (voir Annexe 4) afin de définir des seuils de significativités fiables.

Un Manhattan plot simple des valeurs de *XtX* permet de visualiser les régions génomiques différentiées des populations analysées (Chunk 0.7). Les valeurs de XtX les plus élevés indiquent une différentiation significative entre les populations, parmi ces SNP ceux qui ont une valeur *log10(1/pval)* élevée indique une signature de sélection positive. A l'inverse, des valeurs de *log10(1/pval)* élevées associées à des XtX faibles sont plutôt signe d'une sélection balancée.

Chunk 0.7. Exploitation des XtX

Les valeurs de *XtX* sont contenues dans la colonne ***M_XtX*** L'affichage des chromosomes peut se faire en grille (facet_wrap) ou en ligne (facet_grid).

```
Manplot.XtX = ggplot(data=XtX.res, aes(x=pos, y=M_XtX)) +
  geom_point(aes(color=chr), alpha=0.8, size=0.25) +
  ggtitle("Plot XtX versus position")
Manplot.XtX + scale_x_continuous() +
  scale_y_continuous() +
```

```
  theme(axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank())+
  facet_wrap(~chr, scales = 'free_x', strip.position = c("bottom"))
  #facet_grid(~chr,scales = 'free_x', space = 'free_x', switch = 'x')
```

Les résultats de Bayes Factor *BF*:

Les valeurs de Bayes Factors représentent la corrélation des fréquences alléliques des SNP au travers des populations avec la covariable écologique associée. Ces valeurs sont logarithmiques et exprimés en decibel (dB), les valeurs comprises entre 15 et 20 db constituant une "preuve très forte" et les valeurs >20 db constituant une "preuve décisive" en faveur de l'association avec la covariable selon la règle de Jeffrey [Jeffreys, 1998]. Un Manhattan plot simple des valeurs de *BF* permet de visualiser les SNPs et les régions génomiques fortement associés à la covariable analysée (Chunk 0.8).

Chunk 0.8. Exploitation des Bayes Factors

Les valeurs de Bayes Factors sont contenues dans la colonne ***BF.dB.*** L'affichage des chromosomes peut se faire en grille (facet_wrap) ou en ligne (facet_grid).

```
#import results
BF.res=read.table(paste0(path_out,"project.Cov1"),h=T)
#plot
Manplot.BF = ggplot(data=BF.res, aes(x=pos, y=BF.dB.)) +
  geom_point(aes(color=chr), alpha=0.8, size=0.25) +
  ggtitle("Plot Bayes Factors versus position")
Manplot.BF + scale_x_continuous() +
  scale_y_continuous() +
  theme(axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank())+
  facet_wrap(~chr, scales = 'free_x', strip.position = c("bottom"))
  #facet_grid(~chr,scales = 'free_x', space = 'free_x', switch = 'x')
```

Les résultats de Contraste *C2*:

Les valeurs de contrastes représentent la différence des fréquences alléliques des SNP entre 2 groupes de populations caractérisées par une variable binaire, sensible versus résistant par exemple. Un Manhattan plot simple des valeurs de la colonne *M_C2* permet de visualiser les SNPs et les régions génomiques fortement contrastées (Chunk 0.9).

Chunk 0.9. Exploitation des contrastes

Les valeurs de contraste sont contenues dans la colonne ***M_C2*** L'affichage des chromosomes peut se faire en grille (facet_wrap) ou en ligne (facet_grid).

```
#import results
C2.res=read.table(paste0(path_out,"project.Contrast1"),h=T)
#plot
Manplot.BF = ggplot(data=C2.res, aes(x=pos, y=M_C2)) +
  geom_point(aes(color=chr), alpha=0.8, size=0.25) +
  ggtitle("Plot contrast values versus position")
Manplot.BF + scale_x_continuous() +
  scale_y_continuous() +
  theme(axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank())+
  facet_wrap(~chr, scales = 'free_x', strip.position = c("bottom"))
  #facet_grid(~chr,scales = 'free_x', space = 'free_x', switch = 'x')
```

#Comparaison par diagramme de Venn:

```
#Filtrage selon conditions
thresh.BF2<-c(10)
Datmp1 = subset(Joined.res1, M_XtX> thresh.XtX )
Datmp2 = subset(Joined.res1, M_C2>thresh.C2)
Datmp3 = subset(Joined.res1, BF.dB.>thresh.BF2)

#Conversion en liste au format "chr1_pos1, chr1_pos2, chr1_pos3, ...",
List.SNP1<-paste(Datmp1[,1], "_", Datmp1[,2], sep="")
List.SNP2<-paste(Datmp2[,1], "_", Datmp2[,2], sep="")
List.SNP3<-paste(Datmp3[,1], "_", Datmp3[,2], sep="")
```

```
#Diagramme de Venn
#cat.col = c("darkgreen", "black", "darkblue")
vd0 <- venn.diagram(x=list("XtX top1%" = List.SNP1, "C2 top 1%" = List.SNP2, "BF >50" = List.SNP3), f
grid.newpage()
grid.draw(vd0)
```

#Extraire les listes de chaque intersecte du diagramme:

```
myV <- plotVenn(list("XtX_BF1" = List.SNP3, "XtX_BF2" = List.SNP4))
myV <- plyr::ldply(listVennRegions(myV), cbind)
write.table(myV, file=paste(path_res, "Seed2001_Overlap_XtX_BF.txt", sep=""), quote = FALSE, sep = "\
#récupère la liste des intersectes
myV %>% distinct(myV[,1])
```

#Convertir les liste de SNP dans un intersecte en fichier bed

```
#subset en fonction de l'intersecte que l'on veut
tmp<-subset(myV,myV[,1] == "1, 1, 1 (Contraste01, Contraste02, Contraste03)")
V.tmp<-separate(plyr::ldply(paste(tmp[,2]), cbind), col = "1",  sep = "_", into = c("chr", "pos"))
V.bed <- plyr::ldply(paste(V.tmp[,1], as.numeric(V.tmp[,2])-1, V.tmp[,2]), cbind)
V.bed<-mutate(separate(V.bed, col = "1",  sep = " ", into = c("chr", "start", "end")))
write.table(V.bed, file=paste(path_res, "overlap_3_contrastes.bed", sep=""), quote = FALSE, sep = "\t

<!--chapter:end:06-Resultats.Rmd-->

# Approche par fenêtres glissantes {-}


<!--chapter:end:07-Fenêtre.Rmd-->

# Annexes {-}

## Annexe 1 {-#An1}
Il est possible de calculer des Fst multi-locus en balayant le génome avec une fenêtre glissante de S

### Fst en fenêtre glissante {-}
```

```
#calcul des Fst avec une fenêtre glissante de 100 SNP consécutifs .
Multi.Loc.fst <- computeFST(pooldata,
                            method = "Anova",
                            sliding.window.size = 100)
#conversion en objet data frame
df.fst<-as.data.frame(Multi.Loc.fst$sliding.windows.fst, h=T)
#plot.
Fst.plot = ggplot(data=df.fst, aes(x=CumulatedPosition/1e6, y=MultiLocusFst)) +
  geom_point(aes(color=Chr), alpha=0.8, size=1.5) +
```

```
  ggtitle("Fst en fenêtres glissantes") +
  geom_hline(yintercept=Multi.Loc.fst$FST,lty=2) #le seuil indique la Fst globale estimée à l'échelle
Fst.plot + scale_x_continuous() +
  scale_y_continuous() +
  theme(axis.title.x=element_blank(), axis.text.x=element_blank(), axis.ticks.x=element_blank()) +
  facet_wrap(~Chr, scales = 'free_x', strip.position =c("bottom")) #affichage des chromosomes en gril
```

Annexe 2

Conversion de la matrice SVD en graphique ggplot

Il peut être intéressant de convertir la SVD en graphique ggplot, afin d'avoir la main sur tous les aspects cosmétiques en vue d'une publication par exemple. Le principe consiste à créer une table dans laquelle on intègre les informations à ploter (noms, phenotype, couleurs) ainsi que les les 2 premiers vecteurs propres (eigen vector) et leur variance, de la matrice SVD.

```
# convertit en data.frame
tab_SVD <- data.frame(sample.id = pnames,
    #défini le phenotype de chaque population
    phenotype = factor(c("Sensible",   "Sensible",  "Sensible", "Resistant", "Resistant", "Resistant"
    #attribut une couleur à chaque zone géographique
    col_geo = factor(c("orange",   "darkgreen",  "blue", "orange", "darkgreen","blue")),
    PC = SVD_omega$PC,    # the first eigenvector
    eig = SVD_omega$eig,    # the second eigenvector
    VAR = SVD_omega$pcent.var,    # variance de chaque eigenvector
    stringsAsFactors = FALSE)

ggplot_svd = ggplot(data=tab_SVD, aes(x=PC.1, y=PC.2, shape=phenotype, color= col_geo)) +
  geom_point(alpha=1, size=5)+
  scale_shape_manual(values = c('Sensible'=79, 'Resistant'=16))+
  scale_colour_manual(name = "region",
                      labels = c("sud", "nord", "est"),
                      values = col_geo)+
  ggtitle(paste0("Singular Value Decomposition of the covariance (",expression("\U03A9"),") matrix"))

ggplot_svd +
  scale_x_continuous()+
  scale_y_continuous()+
  xlab(paste("PC1 (",round(tab_SVD$VAR[1], 2), "%)" , sep=""))+
  ylab(paste("PC2 (", round(tab_SVD$VAR[2], 2), "%)" , sep=""))+
  geom_text(aes(label = sample.id), nudge_y = 0.05,fontface = 'bold')
```

Annexe 3

Concaténages des résultats Betai et contrastes

Valable pour le modèle standard pour Betai et pour les 3 modèles pour contraste.

```
Snpdet_path='/../BayPass/Input/'
My_prefix='project_STD.sub'

#on boucle sur toutes les covariables ou combinaisons de contraste
for k in {1..3}
do
```

```
#on boucle sur tous les fichiers
for i in {1..5}
do
file1=$(echo "$Snpdet_path/snpdet.sub""$i""")
file2=$(echo "$My_prefix""$i""_summary_betai_reg.out")                    #adapter le préfixe
sed -e "s/[[:space:]]\+/ /g" $file1 > SubSNP.b                            #on remplace tous les
sed -e "s/[[:space:]]\+/ /g" $file2 | awk -vK="$k" '{if($1 == K) {print}}' - > SubData.b
paste SubSNP.b SubData.b >> tmp-Cov$k.merged                          #on colle/incrémente dans un
done
#on tri par pos et on reconstruit l'entête
awk '{for (i=1;i<=NF;i++) if ($i+0 == $i && $i ~ /e/) $2 = sprintf("%.0f", $i)} 1' tmp-Cov$k.merged |

#on vire les fichiers tmp
rm tmp-Cov$k.merged
rm SubSNP.b
rm SubData.b

#entête pour les résultats Betai:
sed -i '1i chr pos All1 All2 COVARIABLE MRK M_Pearson SD_Pearson M_Spearman SD_Spearman BF(dB) Beta_i

#entête pour les résultats de contraste:
sed -i '1i chr pos All1 All2 CONTRAST MRK M_C2 SD_C2 C2_std C2_log10(1/pval)' project_contrast-C$k
```

Annexe 4

Création et analyse d'un jeu de données pseudo-observées (POD)

La fonction geno2YN extrait les données de comptages brutes en « Pseudo-Observed Data » (POD) et la fonction simulate.baypass génère un jeu de données simulées à partir de la matrice Ω déjà calculée (omegaB) ainsi qu'un constante Pi.beta que l'on récupère dans un des fichiers de sorties.

```
source("/../baypass_utils.R")
#extrait les données de comptage
POD.data=geno2YN(paste(path_input, "genobaypass", sep=""))
#Extrait le beta pi moyen
pi.betaK=read.table(paste0(path_out, "project.sub3_summary_beta_params.out"),h=T)$Mean
#créé un jeu de données de 10 000 SNPs
POD_BayPass<-simulate.baypass(omega.mat=omegaB,
                              nsnp = 10000,
                              beta.coef = NA,
                              beta.pi = pi.betaK,
                              sample.size=POD.data$NN,
                              pi.maf=0, suffix="project.POD" )
```

Les 4 fichiers .POD générés (dans le dossier actif du pipeline) sont à copier sur le cluster de calcul. Le fichier G.project.POD sera analysé de la même manière que le jeu de données initial (contraste, ecotype…) sans qu'il soit nécéssaire de le subdiviser.

###Analyse des résultats POD: Les fichiers résultats _*mat_omega_, _*summary_pi_xtx.out_, sont copiés tel quel en local, le fichier _*summary_contrast.out_, si il contient les résultats de plusieurs combinaison de contrastes, doit être subdivisé par combinaison au préalable et aussi copiées en local. La matrice Ω POD est comparée à la matrice Ω initialement calculée (omegaB) afin de valider la similarité des analyses. La fonction quantile calcule le seuil en fonction de la valeur probs qu'on lui donne : probs = 0,99 pour un seuil à 1%, probs=0,999 pour un seuil à 0,1% etc. Ces seuils seront utilisé comme seuil de significativité pour le jeu de données réel.

```
POD.omega=as.matrix(read.table(paste(path_POD, "project.POD_mat_omega.out", sep="")))
plot(POD.omega,omegaB) ; abline(a=0,b=1)
FMD.POD <- fmd.dist(POD.omega,omegaB)
cat("Distance FMD =", FMD.POD, "\n")

#Extrait la colonne des XtX et calcule un seuil correspondant au quantile que l'on souhaite
POD.XtX=read.table(paste(path_POD, "project.POD_summary_pi_xtx.out", sep=""),h=T)$M_XtX
thresh.XtX=quantile(POD.XtX,probs=0.99)
cat("Seuil XtX =", thresh.XtX, "(Max=", max(POD.XtX) ,")", "\n")

#Extrait la colonne des C2 et calcule un seuil correspondant au quantile que l'on souhaite
POD.C2=read.table(paste(path_POD, "project.POD_summary_contrast.out", sep=""),h=T)$M_C2
thresh.C2=quantile(POD.C2, probs=0.99)
cat("Seuil de contraste C2 = ", thresh.C2, "(Max=", max(POD.C2) ,")", "\n")
```

Références

Bibliography

Andreas Futschik and Christian Schlötterer. The Next Generation of Molecular Markers From Massively Parallel Sequencing of Pooled DNA Samples. *Genetics*, 186(1):207–218, September 2010. ISSN 1943-2631. doi: 10.1534/genetics.110.114397. URL https://academic.oup.com/genetics/article/186/1/207/6063740.

Mathieu Gautier. Genome-Wide Scan for Adaptive Divergence and Association with Population-Specific Covariates. *Genetics*, 201(4):1555–1579, December 2015. ISSN 1943-2631. doi: 10.1534/genetics.115.181453.

Mathieu Gautier, Renaud Vitalis, Laurence Flori, and Arnaud Estoup. f-statistics estimation and admixture graph construction with Pool-Seq or allele count data using the R package poolfstat. 2022.

Harold Jeffreys. *The theory of probability*. OuP Oxford, 1998.

Daniel C. Koboldt, Qunyuan Zhang, David E. Larson, Dong Shen, Michael D. McLellan, Ling Lin, Christopher A. Miller, Elaine R. Mardis, Li Ding, and Richard K. Wilson. VarScan 2: Somatic mutation and copy number alteration discovery in cancer by exome sequencing. *Genome Research*, 22(3):568–576, March 2012. ISSN 1088-9051. doi: 10.1101/gr.129684.111. URL http://genome.cshlp.org/lookup/doi/10.1101/gr.129684.111.

Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, and 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079, August 2009. ISSN 1367-4811, 1367-4803. doi: 10.1093/bioinformatics/btp352. URL https://academic.oup.com/bioinformatics/article/25/16/2078/204688.

Erica S. Nielsen. Pool-Seq Analyses: PoolFstat & BayPass, 2020. URL https://esnielsen.github.io/post/pool-seq-analyses-poolfstat-baypass/.