R

#all the washing variables are xxx1 and all the filtering variables are xxx2

setwd("/Users/celestecohen/Downloads")

#This would be Code/QTL mapping on the drive

d<-read.table(file = 'SupplementaryDataset\_S7\_genotype.tsv', sep = '\t', header = TRUE)

library(RFQTL)

#Below is formatting of data to match the formatting in the RFQTL tutorial which is online

genotype2<-d[,-c(1:4)]

genotype2<-data.matrix(genotype2)

genotype2<-t(genotype2)

phenotype2<-read.csv('Biofilm\_bioinformatics/Biofilm\_image\_processing/filtering\_phenotypes.csv')

phenotype2<-phenotype2[,-1]

phenotype2<-phenotype2[which(phenotype2$Strain %in% rownames(genotype2)),]

strainNames2 <- phenotype2$Strain

phenotype2 <- phenotype2$X.flocc

sampleInfo2 <- sapply(strainNames2,FUN=function(x){

which(rownames(genotype2)==x)

})

mappingData2 <- preMap(genotype=genotype2,

phenotype=phenotype2,

sampleInfo=sampleInfo2,

scale=T)

library(randomForest)

#Get accurate realscores by calculating them 20x and averaging them – this can take 1-2h

RS2=c()

for(i in 1:20){

r=rfMapper(mappingData = mappingData2,

permute = F,

nforest = 100,

ntree = 150)

RS2=c(RS2,r)

}

RS2\_df=matrix(RS2, ncol=20, byrow=F)

RS2\_df=data.frame(RS2\_df)

realScores2<-rowMeans(RS2\_df)

#Get accurate p-values by doing 3500 permutations (100x35) – this takes 8-9h

#this wd should be an empty folder where the permutations will be stored

setwd("/Users/celestecohen/Downloads/Biofilm\_bioinformatics/filtering\_permutations")

for(i in 1:35){

permutedScores2 <- rfMapper(mappingData = mappingData2,

permute = T,

nforest = 100,

ntree = 150,

nPermutations=100,

file=paste("filt\_permut",i,".RData",sep=""),

nCl=4,

clType="SOCK")

}

#this wd should be the folder in which the permutations folder is

setwd("/Users/celestecohen/Downloads/Biofilm\_bioinformatics")

#the path should be the permutations folder

#pValues are the p-values of regions of SNPs that are always grouped

pValues2 <- pEst(path="filtering\_permutations/",

scores=realScores2,

markersPerIteration = 350,

printProg = T,

pCorrection = "none")

#pValuesX are the p-values for each SNP

pValuesX2 <- pValues2[mappingData2$genotype2group]

chrVec2 <- d$chromosome

QTL\_list2 <- QTLgrouper(pmat = pValuesX2,

sigThreshold = 0.1,

corThreshold = 0.8,

distThreshold = 9,

genotype = genotype2,

chrVec = chrVec2)

markerPositions2 <- d[,c(1,2)]

markerPositions2$chromosome<-gsub("chromosome\_1",1,markerPositions2$chromosome)

markerPositions2$chromosome<-gsub("chromosome\_2",2,markerPositions2$chromosome)

markerPositions2$chromosome<-gsub("chromosome\_3",3,markerPositions2$chromosome)

markerPositions2[,3]=markerPositions2$position

writeQTL(QTLlist = QTL\_list2,traitNames = "Flocc",markerPositions = markerPositions2,path="myResults2.qtl")

qtl2 <- readQTL(path = "myResults2.qtl")

qtl2

#Presenting results

#manhattan plot

results.tab2=markerPositions2[,-3]

results.tab2[,3]=pValuesX2

colnames(results.tab2)=c("CHR","BP","P")

library(qqman)

results.tab2=results.tab2[-which(is.na(results.tab2)),]

results.tab2$CHR=as.numeric(unlist(results.tab2$CHR))

results.tab2[,4]=rep("snp",length(results.tab2$CHR))

colnames(results.tab2)[4]="SNP"

manhattan(results.tab2,ylim = c(0,4), suggestiveline = F, genomewideline = F,cex = 0.7)

abline(h=-log10(0.01),col="red")

text(x=1,y=-log10(0.01)+0.1,labels=" p=0.01",col="red",cex=0.7)

#from the following, determine the regions to use for reg\_map below

markerPositions2[which(-log10(pValuesX2)>=2),]

#mapping regions on chromosomes

#map chr

chromosomes=c("chr1","chr2","chr3")

start=c(1,1,1)

end=c(5579133,4539804,2452883)

chr\_map=matrix(c(chromosomes, start, end), ncol=3, byrow=F)

chr\_map=data.frame(chr\_map)

#map regions

regions2=c("1","2","3","4") #this just needs to count the number of regions for formatting

chr\_names2=c("chr2","chr2","chr2","chr2")

reg\_start2=c(2201006,2305695,2527314,2588585)

reg\_end2=c(2228418,2503487,2527344,2588600)

reg\_map2=matrix(c(regions2,chr\_names2, reg\_start2, reg\_end2), ncol=4, byrow=F)

reg\_map2=data.frame(reg\_map2)

write.table(chr\_map,"chr\_map.txt",sep="\t",col.names=F,row.names=F)

write.table(reg\_map2,"reg\_map2.txt",sep="\t",col.names=F,row.names=F)

library(chromoMap)

chromoMap("chr\_map.txt","reg\_map2.txt",segment\_annotation=T)