Early/Late fortis/scandens project

Fan Han

November 20, 2018

```
library(ggplot2)
```

prepare dataset

Extract the 6 pools and rename the samples

Sample information

Original ID	Sample ID
150731_D00457_0112_BC71DHANXX/Sample_Pool1	sca_Early_p
150731_D00457_0112_BC71DHANXX/Sample_Pool2	sca_Late_b
150813_D00118_0225_BC79PNANXX/Sample_Pool3	sca_Late_p
150819_D00118_0226_AC79LFANXX/Sample_Pool4	for_Early_b
150819_D00118_0226_AC79LFANXX/Sample_Pool5	for_Late_b
150819_D00118_0226_AC79LFANXX/Sample_Pool6	for_Late_p

pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools

```
vcftools --vcf Early_Late.Filtered.vcf --keep sample.list --maf 0.001 --recode --out Early_Late

bcftools reheader -s sample.list --threads 10 -o Early_Late.vcf Early_Late.recode.vcf

gzip Early_Late.vcf

rm Early_Late.recode.vcf
```

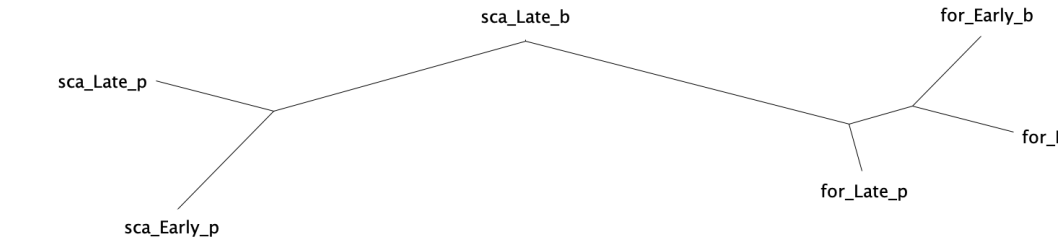
Number of SNPs: 10,814,262

Whole-genome NJ tree

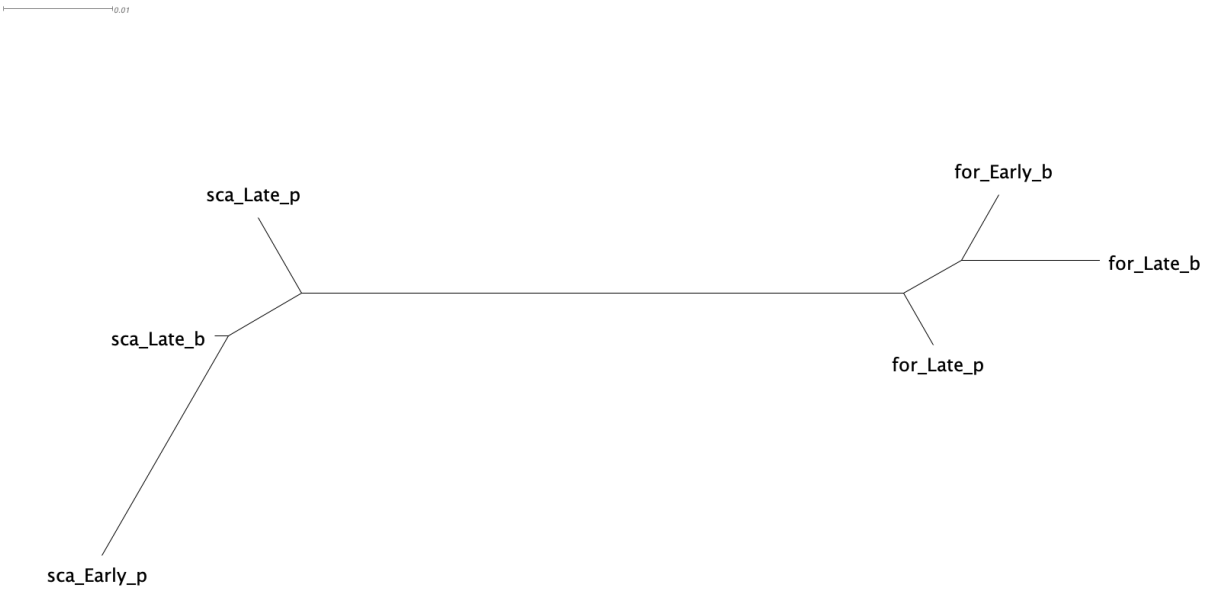
- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/WGTree
- Number of SNPs: 10,795,687

```
ln -s ../TreeScan/Early_Late.freq
head -n 1 Early_Late.freq > Early_Late.Auto.freq
grep -Ff ../../2019_correctedSNPs/Auto_scaffold.list Early_Late.freq >> Early_Late.Auto.freq
perl ../TreeScan/freq2gendist.pl Early_Late.Auto.freq > Early_Late.Auto.GendistIN
gendist
mv outfile Early_Late.Auto.GendistOUT
neighbor
mv outtree Early_Late.Auto.NJ.tree
mv outfile Early_Late.Auto.outfile
```

```
head -n 1 Early_Late.freq > Early_Late.Z.freq
grep -Ff ../../2019_correctedSNPs/Z_scaffold.list Early_Late.freq >> Early_Late.Z.freq


# For publication purpose, we need to bootstrap
# that can be done with PHYLIP seqboot. It could take either molecular sequences or allele frequencies
# Z
echo -e "Early_Late.Z.GendistIN\nD\nD\nD\nR\n1000\nY\n7" > input1
seqboot <input1> screenout && mv outfile Early_Late.Z.seqboot
echo -e "Early_Late.Z.seqboot\nM\n1000\nY" > input2
gendist <input2> screenout && mv outfile Early_Late.Z.gendist
echo -e "Early_Late.Z.gendist\nM\n1000\n7\nY" > input3
neighbor <input3> screenout && mv outfile Early_Late.Z.nj && mv outtree Early_Late.Z.nj.trees
echo -e "Early_Late.Z.nj.trees\nY" > input4
consense <input4> screenout && mv outfile Early_Late.Z.consense && mv outtree Early_Late.Z.consense.tre

# Autosomes
echo -e "Early_Late.Auto.GendistIN\nD\nD\nD\nR\n1000\nY\n7" > input1
seqboot <input1> screenout && mv outfile Early_Late.Auto.seqboot
echo -e "Early_Late.Auto.seqboot\nM\n1000\nY" > input2
gendist <input2> screenout && mv outfile Early_Late.Auto.gendist
echo -e "Early_Late.Auto.gendist\nM\n1000\n7\nY" > input3
neighbor <input3> screenout && mv outfile Early_Late.Auto.nj && mv outtree Early_Late.Auto.nj.trees
echo -e "Early_Late.Auto.nj.trees\nY" > input4
consense <input4> screenout && mv outfile Early_Late.Auto.consense && mv outtree Early_Late.Auto.consens
```



- Plot of whole-genome trees



* It shows that the introgression primarily happened on autosomes

TreeScan with 50kb window

- Because the data is pooled sequencing, I can only build NJ tree for each window
- Calculate the allele frequency for each population
- Remember to midpoint-rooting the trees with retree program from PHYLIP otherwise the visualization in Densitree will be not pretty
- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/TreeScan
- Run time 01:02:51

```
vcftools --gzvcf Early_Late.vcf.gz --out Early_Late --extract-FORMAT-info AD --max-missing 1

./AF_each_pool.pl ../Early_Late.AD.FORMAT > Early_Late.freq

# Runtime: ~2 hrs
./geneFlow_WGScan_pool.pl -freq Early_Late.freq  -window 50000

awk -F "\t" '{print $2}' out.trees >  all.trees
wc -l all.trees number_snp.window
paste number_snp.window all.trees > all.snp.trees

# To get better view in DensiTree, I rooted all the trees
./midpoint_rooting.pl all.snp.trees > all.snp.midpoint.trees

grep -Ff ../../2019_correctedSNPs/Auto_scaffold.list all.snp.midpoint.trees | awk -F "\t" '($2>=20)&&($
grep -Ff ../../2019_correctedSNPs/Z_scaffold.list all.snp.midpoint.trees | awk -F "\t" '($2>=20)&&($2<=
```

```
rm all.trees
```

Number of trees: 20,632

DensiTree to visualize all the midpoint-rooted trees

- It is noticeable that within-species introgression happens more on Z and between-species introgression happens more on autosomes

Summary of the tree topologies across the genome

- The output of TreeScan is a list of newick trees. The tricky thing is to compare them in pairwise and decide which topologies are the same.
  It seems ETE Toolkit could help with this: http://etetoolkit.org/documentation/ete-compare/. It compares the Robinson-Foulds' distance of multiple trees with reference tree and report mismatches. But it needs reference trees as input. So first I need to produce all the possible topologies
- Treedist from PHYLIP package could also report pairwise distance and much faster than ETE. Change the Distance type to symmetric Difference. http://evolution.genetics.washington.edu/phylip/doc/treedist.html
- I cannot really determine whether two populations are clustered in unrooted trees if they are present on the internal branch. So I need to root every tree and compare if they are identical
- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/TreeScan/Rooted
- Runtime: ~4 days

```
# simplify trees by removing the branch length. Only focus on the toplogy
./treeSimple.pl all.snp.midpoint.trees > all.snp.midpoint.simple.trees
# Produce all the unique trees as reference. (62 unique trees)
./uniq_trees_rooted.pl all.snp.midpoint.simple.trees > reference.rooted.trees
# compare each window to all reference trees. It is time-consuming and I culd split the files to run in
./compare_window_tree_rooted.pl reference.rooted.trees all.snp.midpoint.simple.trees > all.snp.midpoint
```

- Simplify tree categories. The trees that have two distinct clusters of fortis and scandens should be considered as one type no matter how the within-species divergence is. But how...

```
./category_trees.pl reference.rooted.trees > reference.rooted.trees.class
./tree_class.pl all.snp.midpoint.assign.trees > all.snp.midpoint.assign.class.trees
grep -Ff ../../../2019_correctedSNPs/Auto_scaffold.list all.snp.midpoint.assign.class.trees > all.snp.mi
grep -Ff ../../../2019_correctedSNPs/Z_scaffold.list all.snp.midpoint.assign.class.trees > all.snp.midpo
```

- Count the number of each rooted toplogy across the genome

```
# Autosome
auto <- read.table("TreeScan/Rooted/all.snp.midpoint.assign.class.Auto.trees", header=F)
colnames(auto) <- c("CHR", "BIN_START", "N_SNP", "Tree", "Ref_Tree", "Tree_Type")
auto <- auto[auto$N_SNP>=20&auto$N_SNP<=2000,]
auto$TYPE <- rep("Auto", nrow(auto))

# Z-linked scaffolds
Z <- read.table("TreeScan/Rooted/all.snp.midpoint.assign.class.Z.trees", header=F)
colnames(Z) <- c("CHR", "BIN_START", "N_SNP", "Tree", "Ref_Tree", "Tree_Type")
Z <- Z[Z$N_SNP>=20&Z$N_SNP<=2000,]
Z$TYPE <- rep("Z", nrow(Z))

# combine them
genome <- rbind(auto,Z)
```

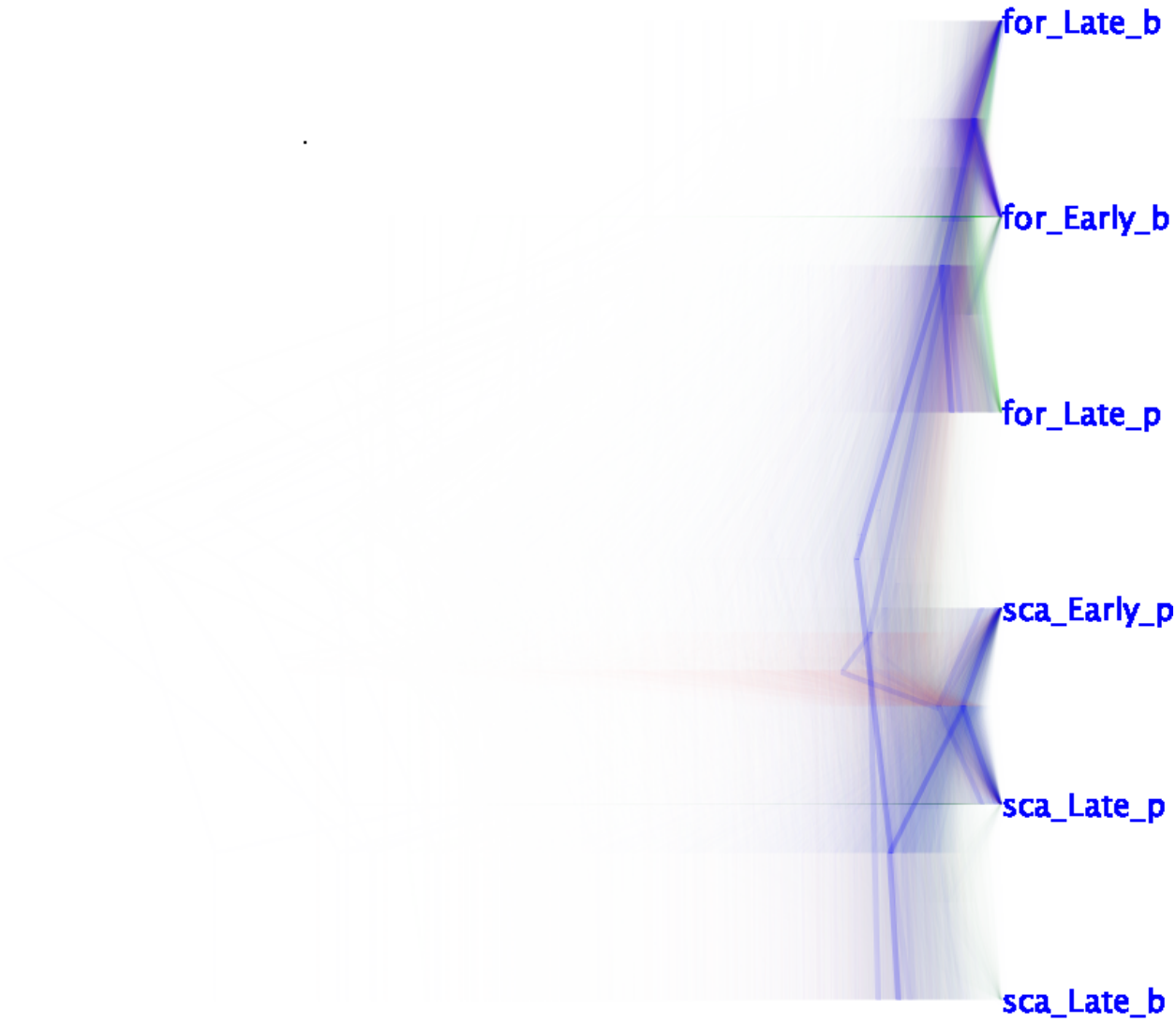


Figure 1: 19,040 Autosomal trees from DensiTree

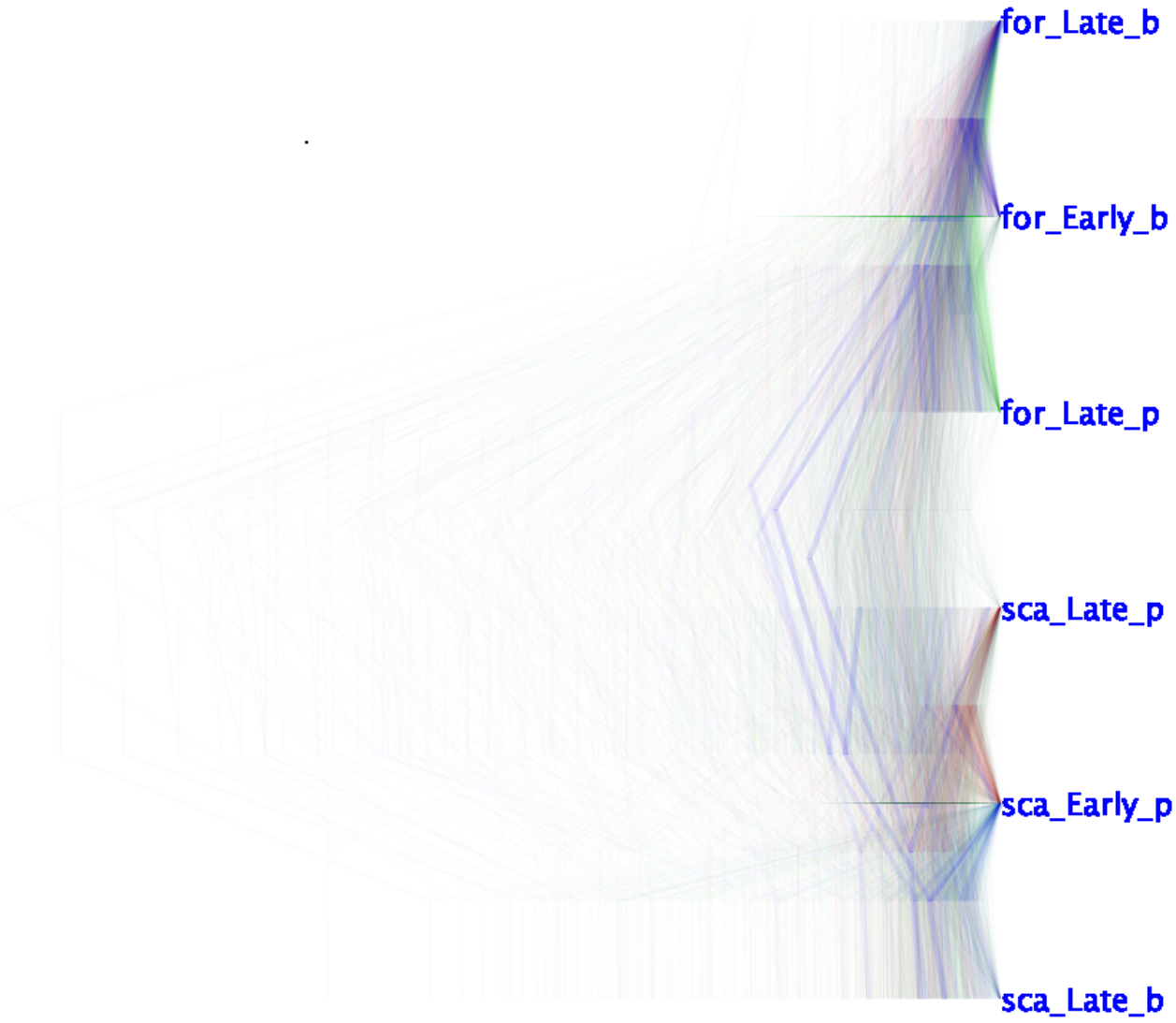
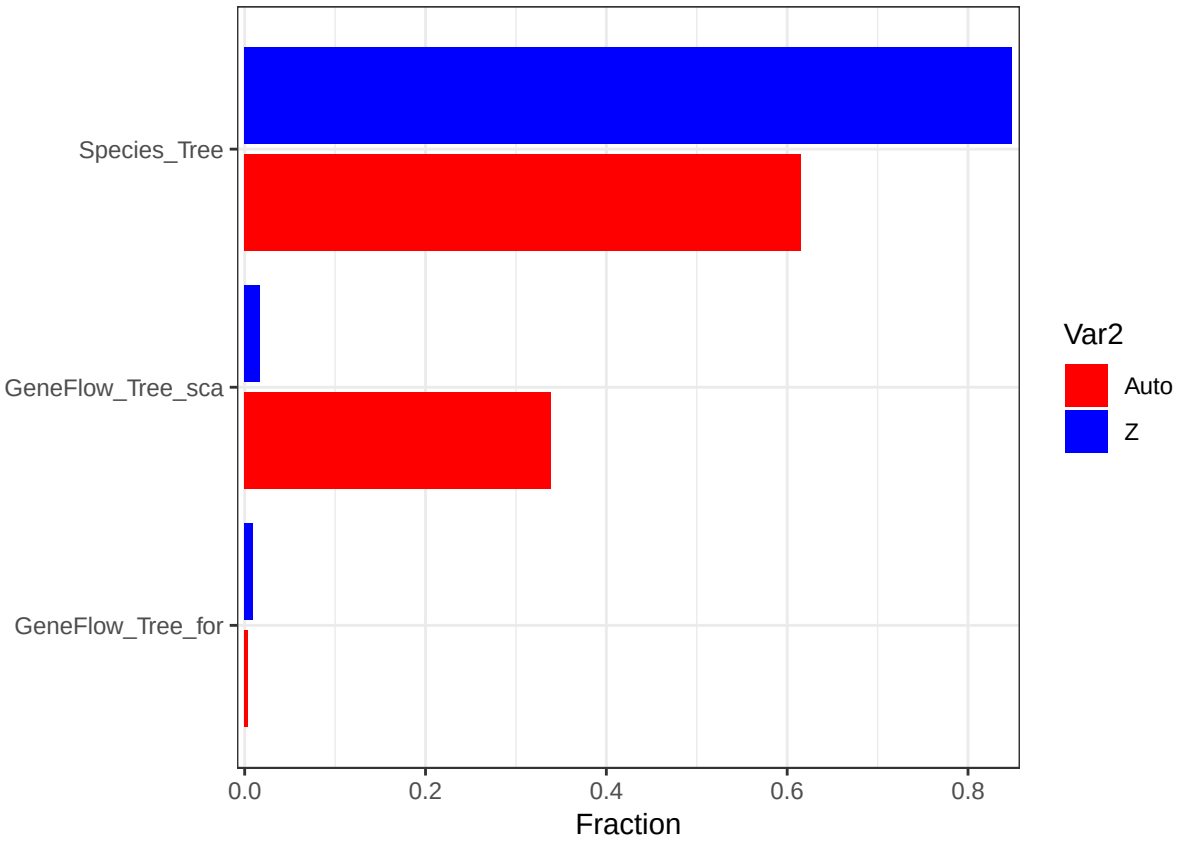


Figure 2: 1,536 Z trees from DensiTree

```
genome.table <- as.data.frame(table(genome$Tree_Type, genome$TYPE))
library(dplyr)
genome.table <- genome.table %>% group_by(Var2) %>% mutate(percent = Freq/sum(Freq))

# Don't plot the Other_tree type
genome.table <- genome.table[genome.table$Var1!="Other_Tree",]
library(ggplot2)
#pdf("TreeScan/Rooted/Rooted_tree_hist.pdf", width = 5.8, height = 4.1)
ggplot(genome.table) + geom_col(aes(x=reorder(Var1,percent), y=percent, fill=Var2), position = "dodge2")
```




```
#print(g)
#dev.off()
genome.table
```

```
## # A tibble: 6 x 4
## # Groups:   Var2 [2]
##   Var1              Var2   Freq percent
##   <fct>             <fct> <int>   <dbl>
## 1 GeneFlow_Tree_for Auto     76 0.00399
## 2 GeneFlow_Tree_sca Auto   6449 0.339
## 3 Species_Tree      Auto  11712 0.615
## 4 GeneFlow_Tree_for Z        14 0.00911
## 5 GeneFlow_Tree_sca Z        26 0.0169
## 6 Species_Tree      Z      1304 0.849
```

- Representives of each type of trees
- Recombination rate in the regions of introgression

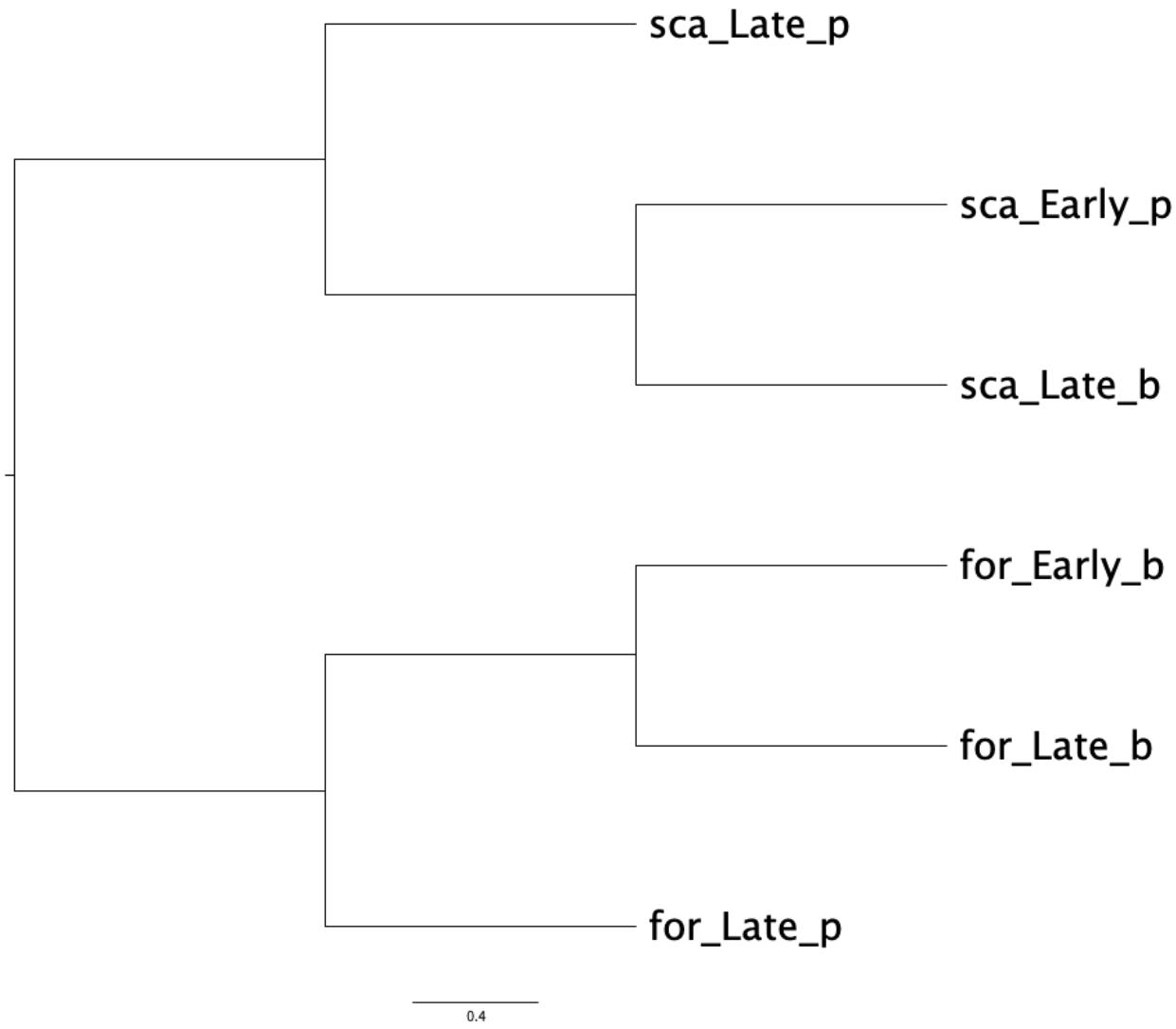


Figure 3: Species Tree

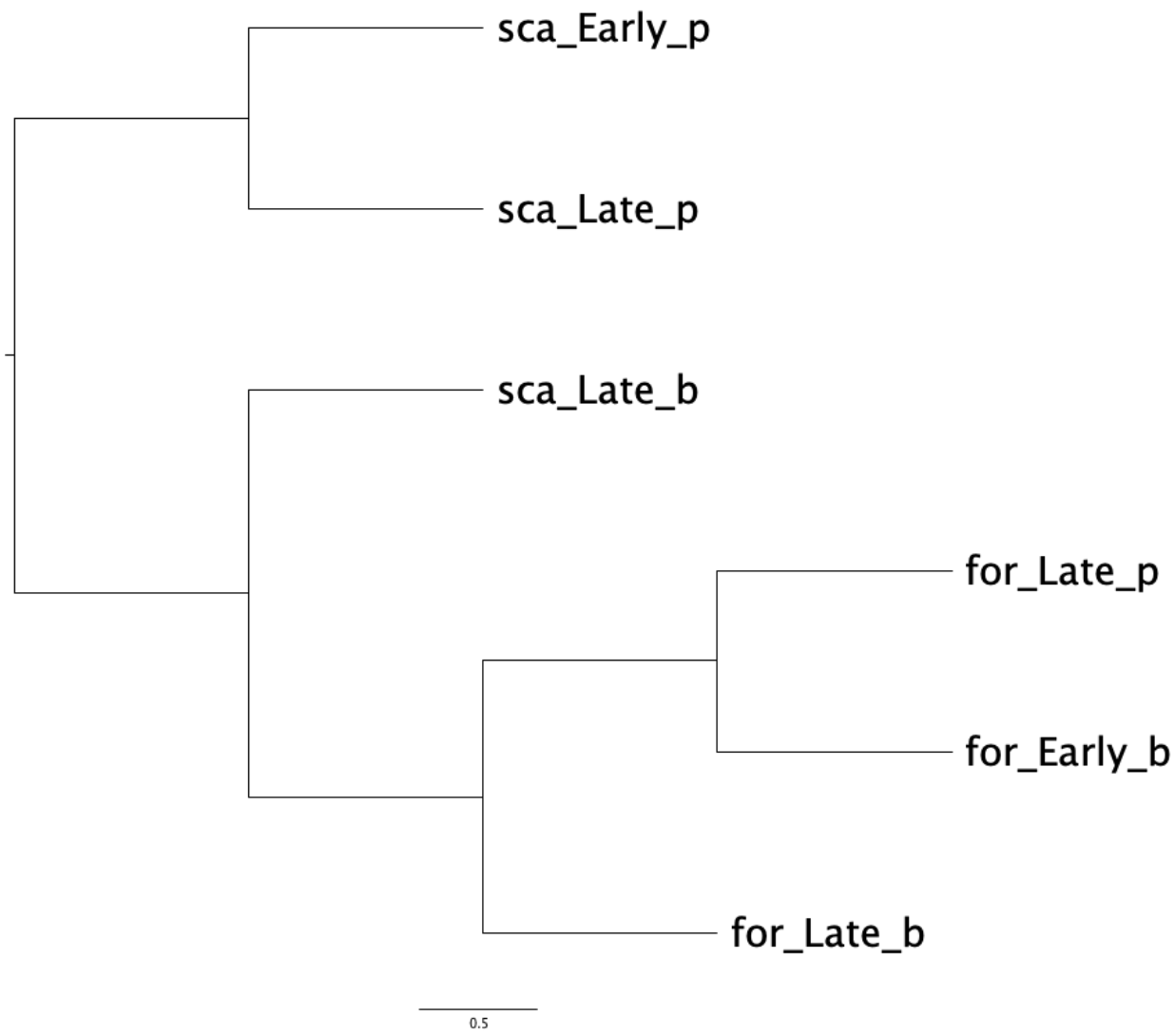


Figure 4: fortis introgressed scandens Late blunt

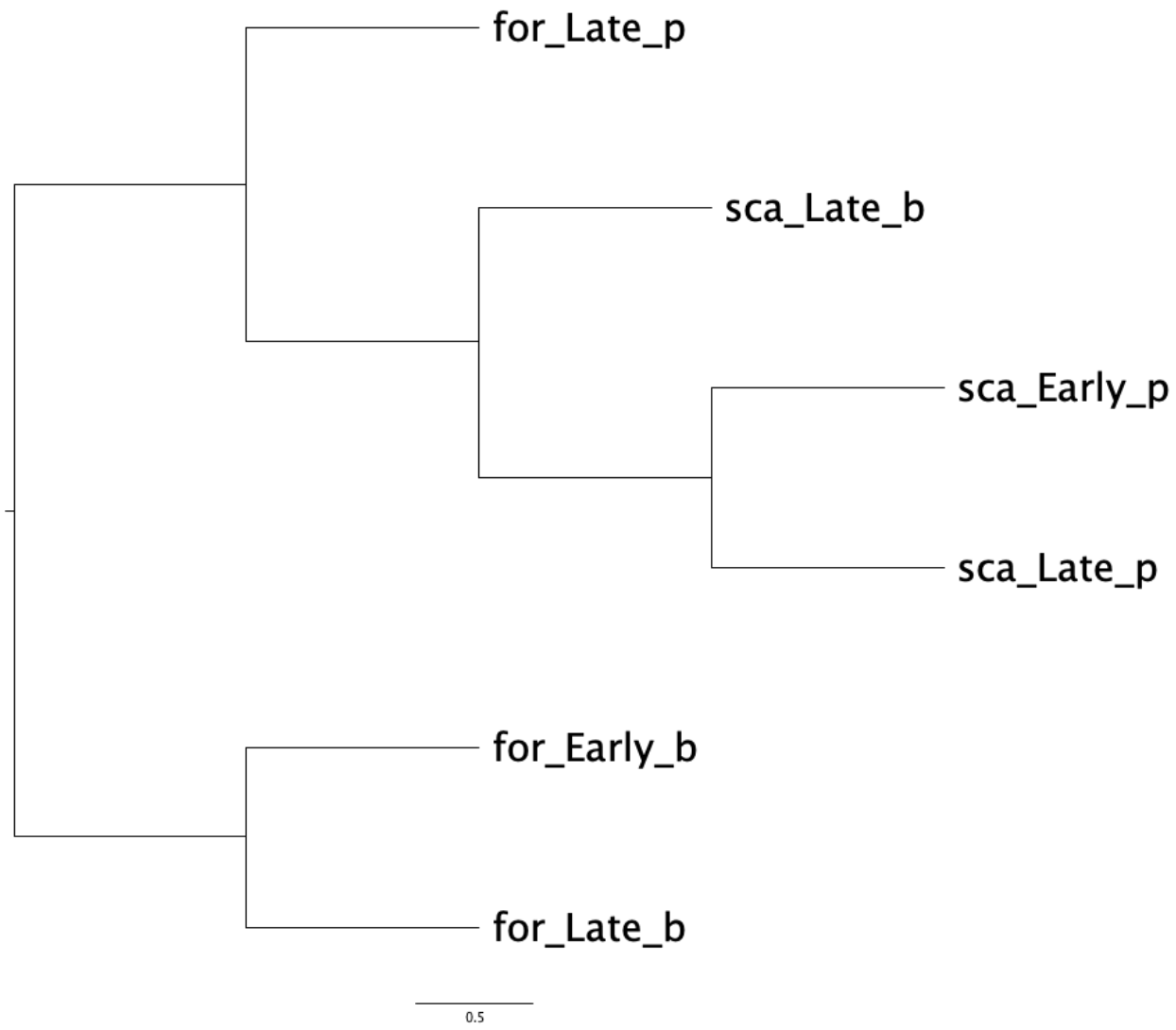


Figure 5: scandens introgressed fortis Late pointed

```
# convert to zebrafinch genome
./convert.sh all.snp.midpoint.assign.class.trees
bedtools intersect -wa -wb -a all.snp.midpoint.assign.class.trees.Auto.zebra.bed -b ../../zebra_RR/zebra
bedtools intersect -wa -wb -a all.snp.midpoint.assign.class.trees.Z.zebra.bed -b ../../zebra_RR/zebrafi
cat all.snp.midpoint.assign.class.trees.Auto.zebra.RR.bed all.snp.midpoint.assign.class.trees.Z.zebra.RR
```

- exam the relation between tree topology and recombination rate

```
mydata <- read.table("TreeScan/Rooted/all.snp.midpoint.assign.class.trees.zebra.RR.bed", header=F)
mydata <- mydata[,c(1,2,4,5,6,8,9,13)]
colnames(mydata) <- c("CHR_zebra","POS_zebra","CHR","BIN_START","N_SNP","REF_TREE","TREE_TYPE","RR_zebra
#mydata$RR_zebra <- (mydata$RR_zebra*1000000/50000)*100/(4*4.8*1e6)
#mydata$RR_zebra <- log10(mydata$RR_zebra)
autosome <- mydata[mydata$CHR_zebra!="chrZ",]
autosome.species <- autosome[autosome$TREE_TYPE=="Species_Tree",]
autosome.flow <- autosome[autosome$TREE_TYPE=="GeneFlow_Tree_sca",]
t.test(autosome.species$RR_zebra, autosome.flow$RR_zebra)
```

```
##
##  Welch Two Sample t-test
##
## data:  autosome.species$RR_zebra and autosome.flow$RR_zebra
## t = 5.4118, df = 7074.6, p-value = 6.445e-08
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  4089.353 8734.531
## sample estimates:
## mean of x mean of y
##  19961.25  13549.31
```

```
wilcox.test(autosome.species$RR_zebra, autosome.flow$RR_zebra)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  autosome.species$RR_zebra and autosome.flow$RR_zebra
## W = 11798000, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

```
# Without chr1
t.test(autosome.species[autosome.species$CHR_zebra!="chr1"&autosome.species$CHR_zebra!="chr1A",]$RR_zebr
```

```
##
##  Welch Two Sample t-test
##
## data:  autosome.species[autosome.species$CHR_zebra != "chr1" & autosome.species$CHR_zebra !=  and au
## t = 0.89956, df = 2838.6, p-value = 0.3684
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2128.050  5735.736
## sample estimates:
## mean of x mean of y
##  24142.68  22338.83
```

```
wilcox.test(autosome.species[autosome.species$CHR_zebra!="chr1"&autosome.species$CHR_zebra!="chr1A",]$RR
```

```
##
```

```
##  Wilcoxon rank sum test with continuity correction
##
## data:  autosome.species[autosome.species$CHR_zebra != "chr1" & autosome.species$CHR_zebra !=  and au
## W = 3655700, p-value = 1.66e-05
## alternative hypothesis: true location shift is not equal to 0
```

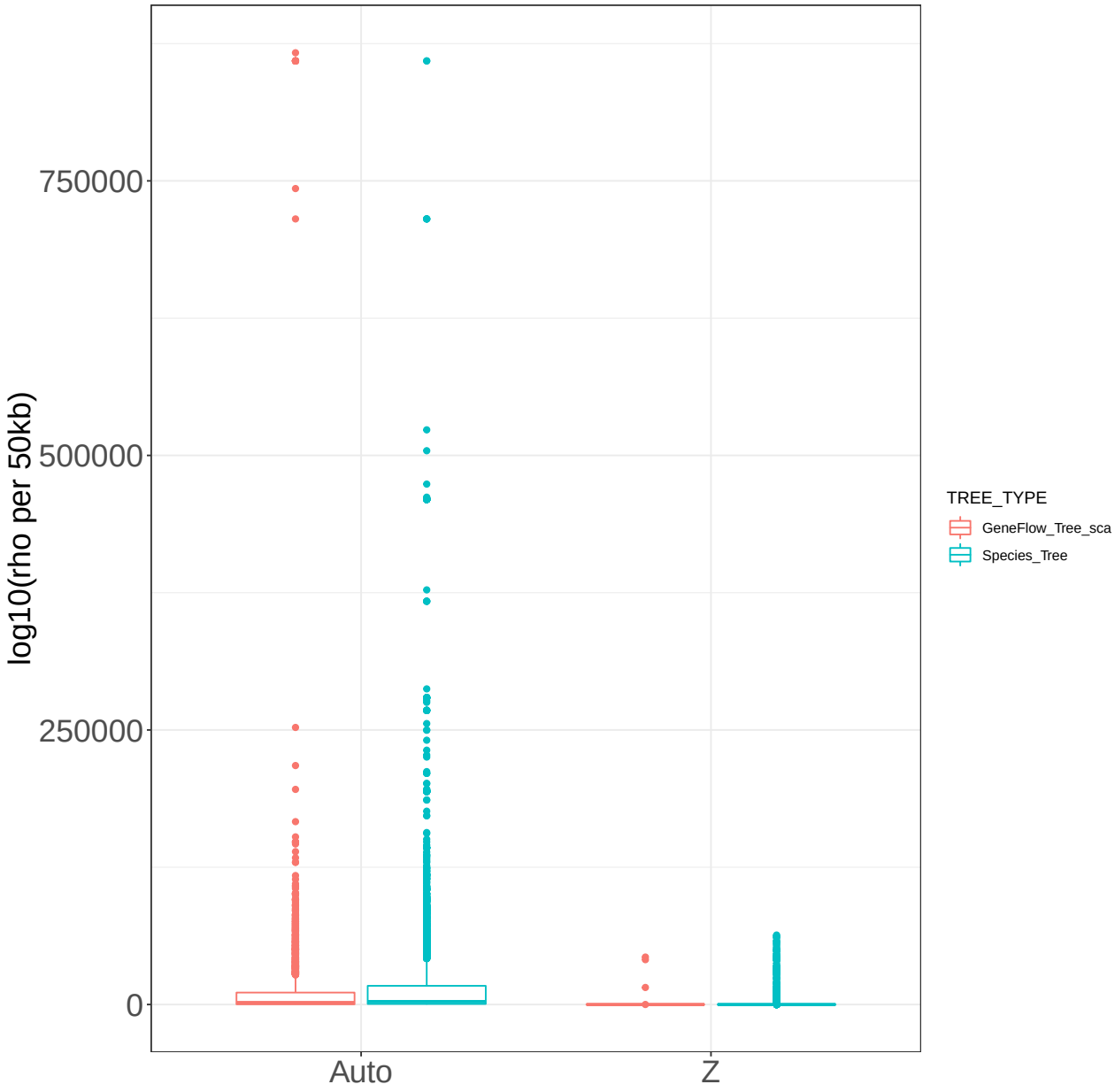
```
Z<-mydata[mydata$CHR_zebra=="chrZ",]
Z.species <- Z[Z$TREE_TYPE=="Species_Tree",]
Z.flow <- Z[Z$TREE_TYPE=="GeneFlow_Tree_sca",]
t.test(Z.species$RR_zebra, Z.flow$RR_zebra)
```

```
##
##  Welch Two Sample t-test
##
## data:  Z.species$RR_zebra and Z.flow$RR_zebra
## t = -0.6865, df = 19.404, p-value = 0.5005
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -8202.345  4146.328
## sample estimates:
## mean of x mean of y
##  2954.793  4982.801
```

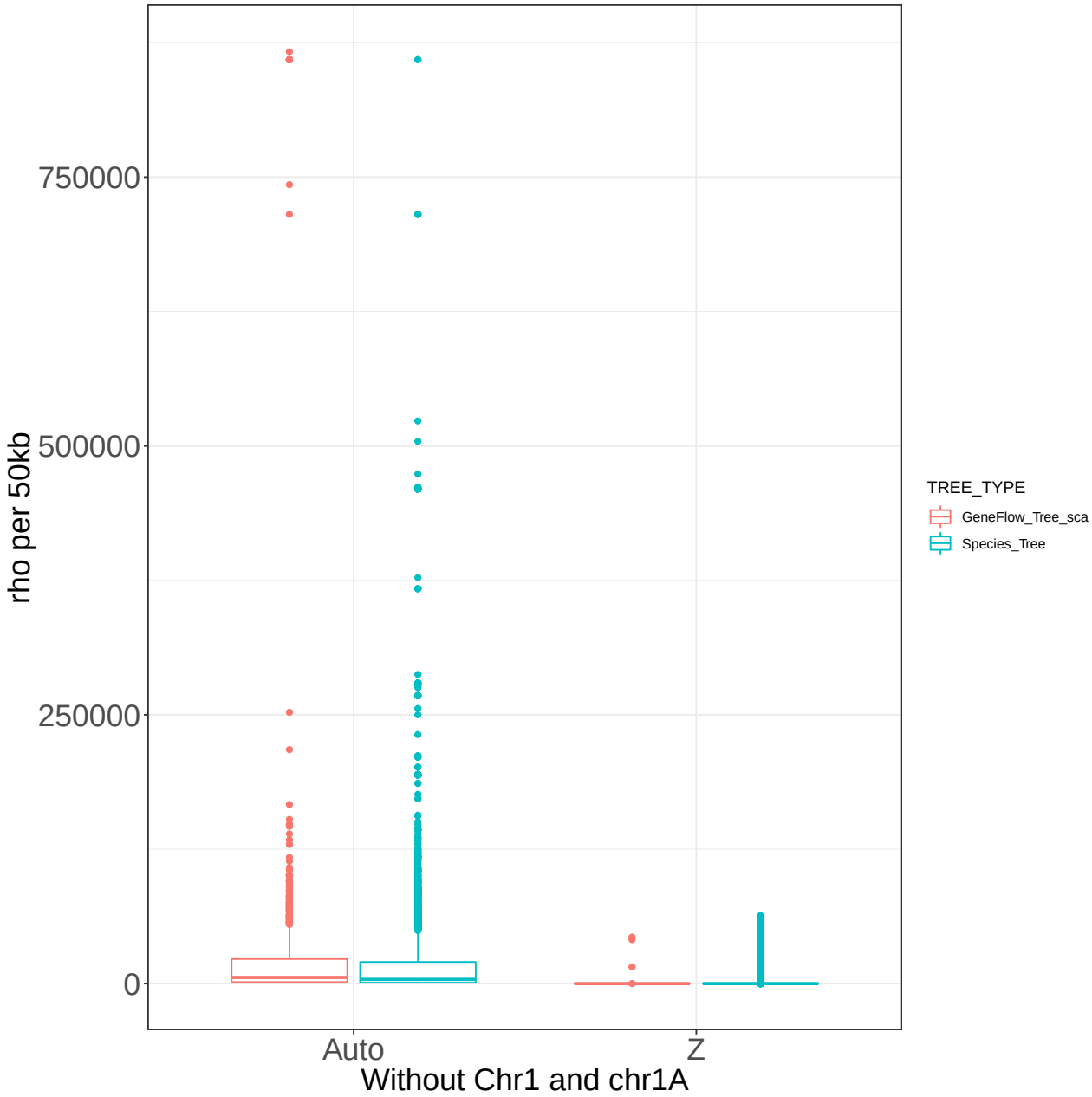
```
wilcox.test(Z.species$RR_zebra, Z.flow$RR_zebra)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Z.species$RR_zebra and Z.flow$RR_zebra
## W = 11497, p-value = 0.839
## alternative hypothesis: true location shift is not equal to 0
```

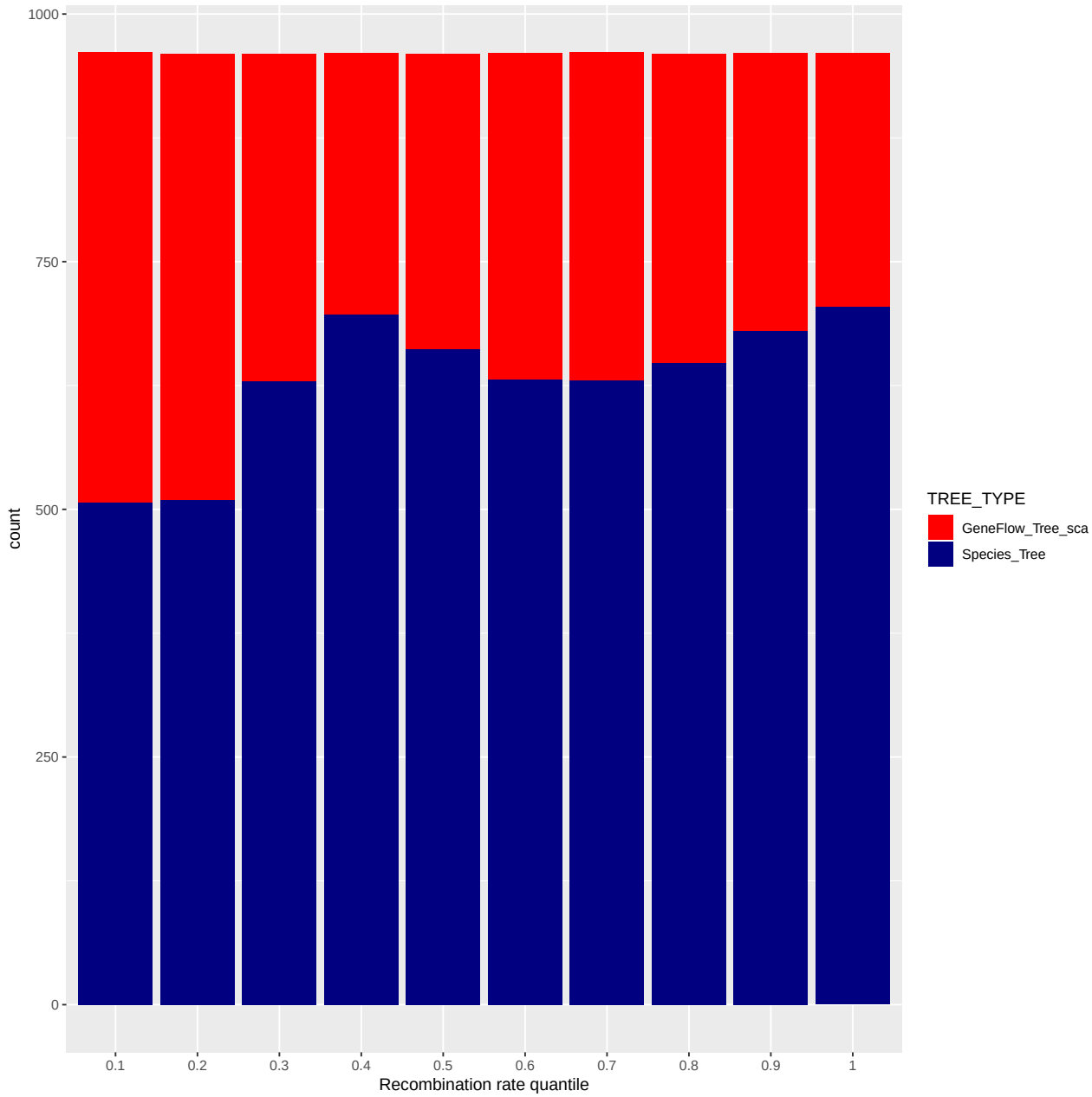
```
mydata <- mydata[mydata$TREE_TYPE=="Species_Tree"|mydata$TREE_TYPE=="GeneFlow_Tree_sca",]
mydata$CHR_TYPE <- ifelse(mydata$CHR_zebra=="chrZ", "Z", "Auto")
# boxplot
library(ggplot2)
ggplot(mydata, aes(CHR_TYPE, RR_zebra, colour=TREE_TYPE)) + geom_boxplot() + ylab("log10(rho per 50kb)"
```



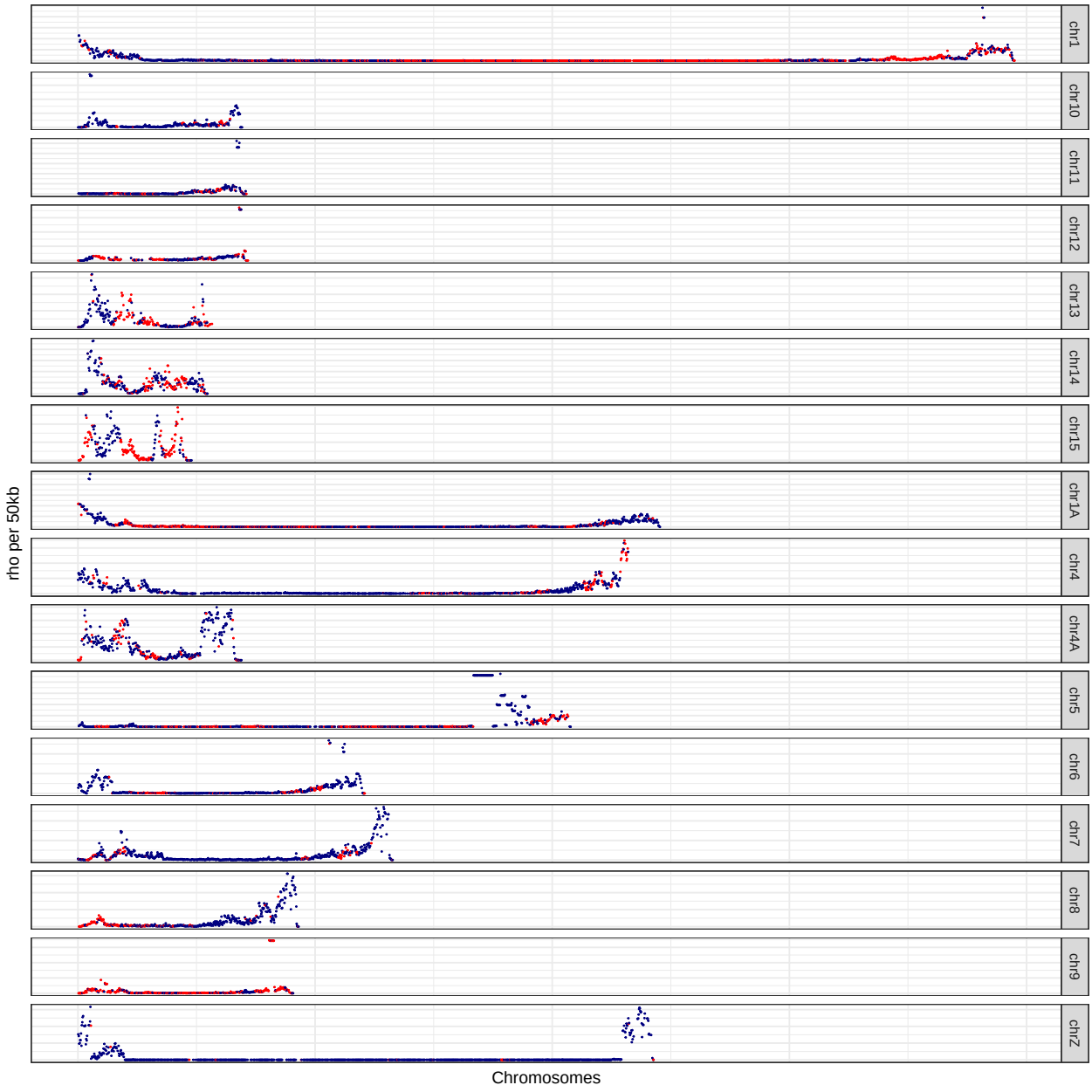

```
# boxplot without chr1 and chr1A
mydata_noChr1 <- mydata[mydata$CHR_zebra!="chr1"&mydata$CHR_zebra!="chr1A",]
ggplot(mydata_noChr1, aes(CHR_TYPE, RR_zebra, colour=TREE_TYPE)) + geom_boxplot() + ylab("rho per 50kb"
```




```
autosome <- droplevels(autosome[autosome$TREE_TYPE=="Species_Tree"|autosome$TREE_TYPE=="GeneFlow_Tree_s
quan_bins <- unname(quantile(autosome$RR_zebra, probs = seq(0,1,0.1), na.rm = T))
autosome$RR_bin <- cut(autosome$RR_zebra, breaks=quan_bins, include.lowest = TRUE, right=TRUE, labels =
RR_df <- as.data.frame(table(autosome$TREE_TYPE,autosome$RR_bin))
# bar plot
ggplot(autosome, aes(RR_bin, fill=TREE_TYPE,group=TREE_TYPE)) + geom_bar()+ scale_fill_manual(values = 
```




```
# For each chromosome
ggplot(mydata) + geom_point(aes(POS_zebra, RR_zebra, colour=TREE_TYPE), size=0.1) + scale_colour_manual
```



introgression in fortis_Late_pointed

- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/geneDensity
- These are 66 windows displaying for_Late_pointed introgression, and most of them sit on chr4

```
grep "GeneFlow_Tree_for" all.snp.midpoint.assign.class.trees.zebra.RR.geneDensity.bed | awk '{OFS="\t";
# 54 genes in the gene-flow regions
awk '{print $12}' for_Late_tree.bed.anno | sort | uniq | awk -F ";" '{print $2}' | sed 's/Name=//'
```

- perform GO analysis for these genes
- using the GO database I constructed before: /proj/uppstore2017190/b2012111/private/UserDirectory/fan/phylogenomics/

```
# convert to chicken ENSEMBL ID
perl geneID_match.pl for_Late_tree.bed.anno.gene.list > for_Late_tree.bed.anno.gene.gal.ens
```

- TopGO

```
#topGO
library(topGO)
library(biomaRt)
#library("org.Gg.eg.db")
rm(list=ls())
background <- read.table("geneDensity/for_Late_pointed_GOterm/finch_gal.background", header=F)
background <-background$V1
names(background)<- background

mydata <- read.table("geneDensity/for_Late_pointed_GOterm/for_Late_tree.bed.anno.gene.gal.ens", header=
mydata <- mydata[complete.cases(mydata),]
target_gene <- unique(mydata)
names(target_gene)<-target_gene
# define interesting genes out of all background genes
gene_list <- factor(as.integer(background %in% target_gene))
names(gene_list) <- background

# get ensembl gene ids with GO term
bm <- useMart("ensembl")
bm <- useDataset("ggallus_gene_ensembl",mart=bm)
gene2GO <- getBM(mart=bm, attributes = c("ensembl_gene_id","go_id"), filters="ensembl_gene_id", values=
gene2GO <- gene2GO[gene2GO$go_id!="",]
gene2GO <-by(gene2GO$go_id,gene2GO$ensembl_gene_id, function(x) as.character(x))

# create GOdata class for topGO
GOdata <- new("topGOdata", ontology="BP", allGenes=gene_list, annot= annFUN.gene2GO, gene2GO=gene2GO, n
# run test
#resultClassic <- runTest(GOdata, algorithm = "classic", statistic = "fisher")
resultWeight <- runTest(GOdata, algorithm = "weight", statistic = "fisher")
resultElim <- runTest(GOdata, algorithm="elim", statistic="fisher")
ug <- usedGO(GOdata)
resultTable <- GenTable(GOdata, Weight_P = resultWeight, Elim_P=resultElim, topNodes = 50, numChar=5000
# if you want to print out all terms
#resultTable <- GenTable(GOdata, Weight_P = resultWeight, Elim_P=resultElim, topNodes = length(ug), num
write.table(resultTable, file="geneDensity/for_Late_pointed_GOterm/for_Late_tree.bed.anno.gene.gal.ens.
```

Correlation between gene flow and gene density

- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/geneDensity

```
# only the protein-coding genes are considered
grep "Gnomon" ../../ref_aln/Geofor1.ncbi.gene.gff3 > Geofor1.ncbi.gene.gff3
bedtools makewindows -g geoFor1.genome -w 50000 -s 50000 > geoFor1.50kb.window
bedtools map -a geoFor1.50kb.window -b Geofor1.ncbi.gene.gff3 -o count -c 9 > geoFor1.50kb.geneDensity
perl density_match.pl all.snp.midpoint.assign.class.trees.zebra.RR.bed > all.snp.midpoint.assign.class.
```

- Plot the correlation

```
mydata<- read.table("geneDensity/all.snp.midpoint.assign.class.trees.zebra.RR.geneDensity.bed", header=
mydata <- mydata[,c(1,2,4,5,6,8,9,13,14)]
colnames(mydata) <- c("CHR_zebra","POS_zebra","CHR","BIN_START","N_SNP","REF_TREE","TREE_TYPE","RR_zebra
#mydata$Gene_Density <- mydata$Gene_Density*1000000/50000
# gene desert
#mydata <- mydata[mydata$Gene_Density>0,]
# autosome
autosome <- mydata[mydata$CHR_zebra!="chrZ",]
autosome.species <- autosome[autosome$TREE_TYPE=="Species_Tree",]
autosome.flow <- autosome[autosome$TREE_TYPE=="GeneFlow_Tree_sca",]
t.test(autosome.species$Gene_Density, autosome.flow$Gene_Density)
```

```
##
##  Welch Two Sample t-test
##
## data:  autosome.species$Gene_Density and autosome.flow$Gene_Density
## t = 4.2617, df = 7164.3, p-value = 2.055e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.05545001 0.14991184
## sample estimates:
## mean of x mean of y
##  1.112063  1.009383
```

```
wilcox.test(autosome.species$Gene_Density, autosome.flow$Gene_Density)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  autosome.species$Gene_Density and autosome.flow$Gene_Density
## W = 10907000, p-value = 4.022e-05
## alternative hypothesis: true location shift is not equal to 0
```

```
Z<-mydata[mydata$CHR_zebra=="chrZ",]
Z.species <- Z[Z$TREE_TYPE=="Species_Tree",]
Z.flow <- Z[Z$TREE_TYPE=="GeneFlow_Tree_sca",]
t.test(Z.species$Gene_Density, Z.flow$Gene_Density)
```

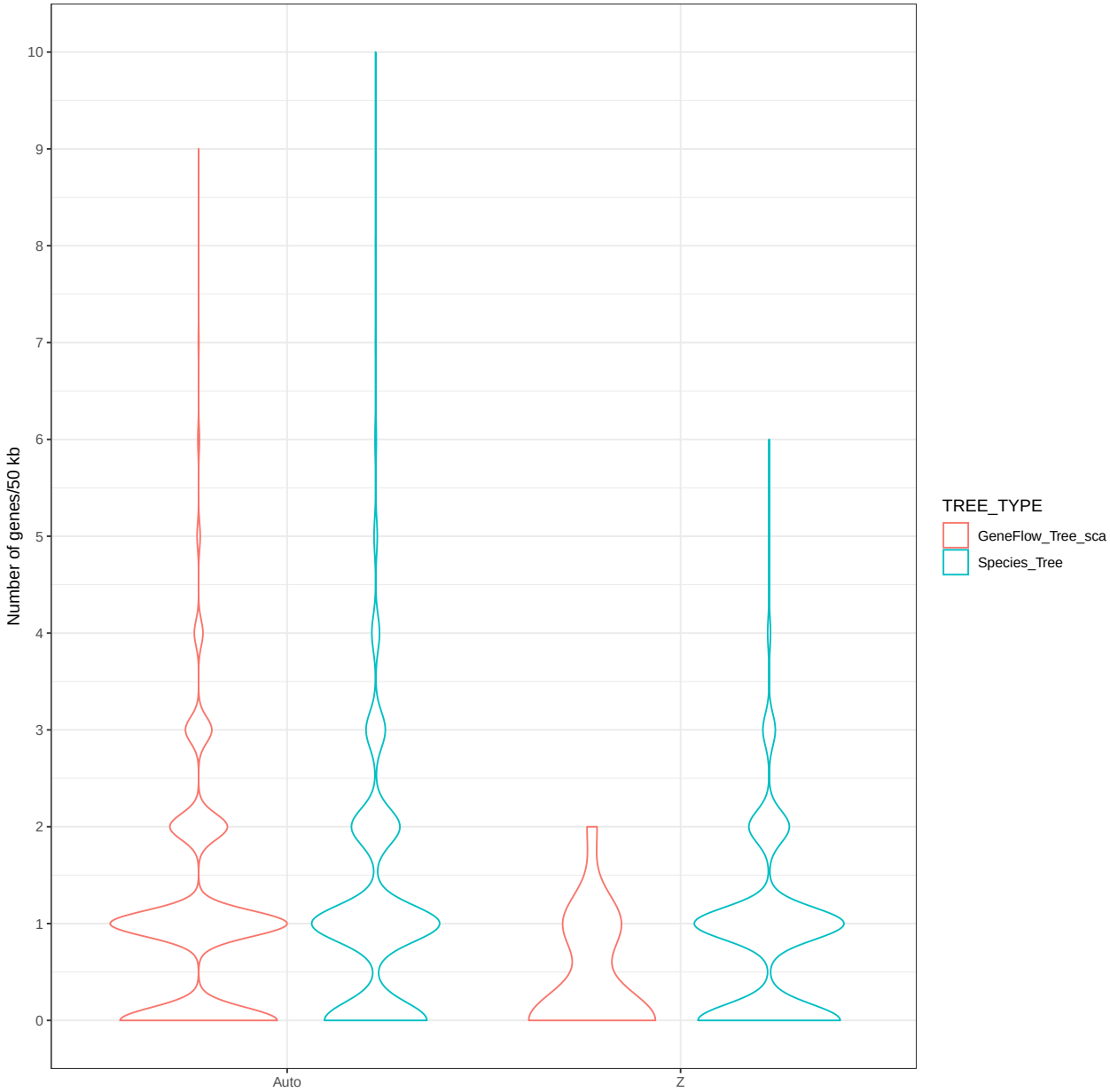
```
##
##  Welch Two Sample t-test
##
## data:  Z.species$Gene_Density and Z.flow$Gene_Density
## t = 3.0467, df = 20.461, p-value = 0.006256
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1313281 0.6990290
## sample estimates:
## mean of x mean of y
## 0.8151786 0.4000000
```

```
wilcox.test(Z.species$Gene_Density, Z.flow$Gene_Density)
```

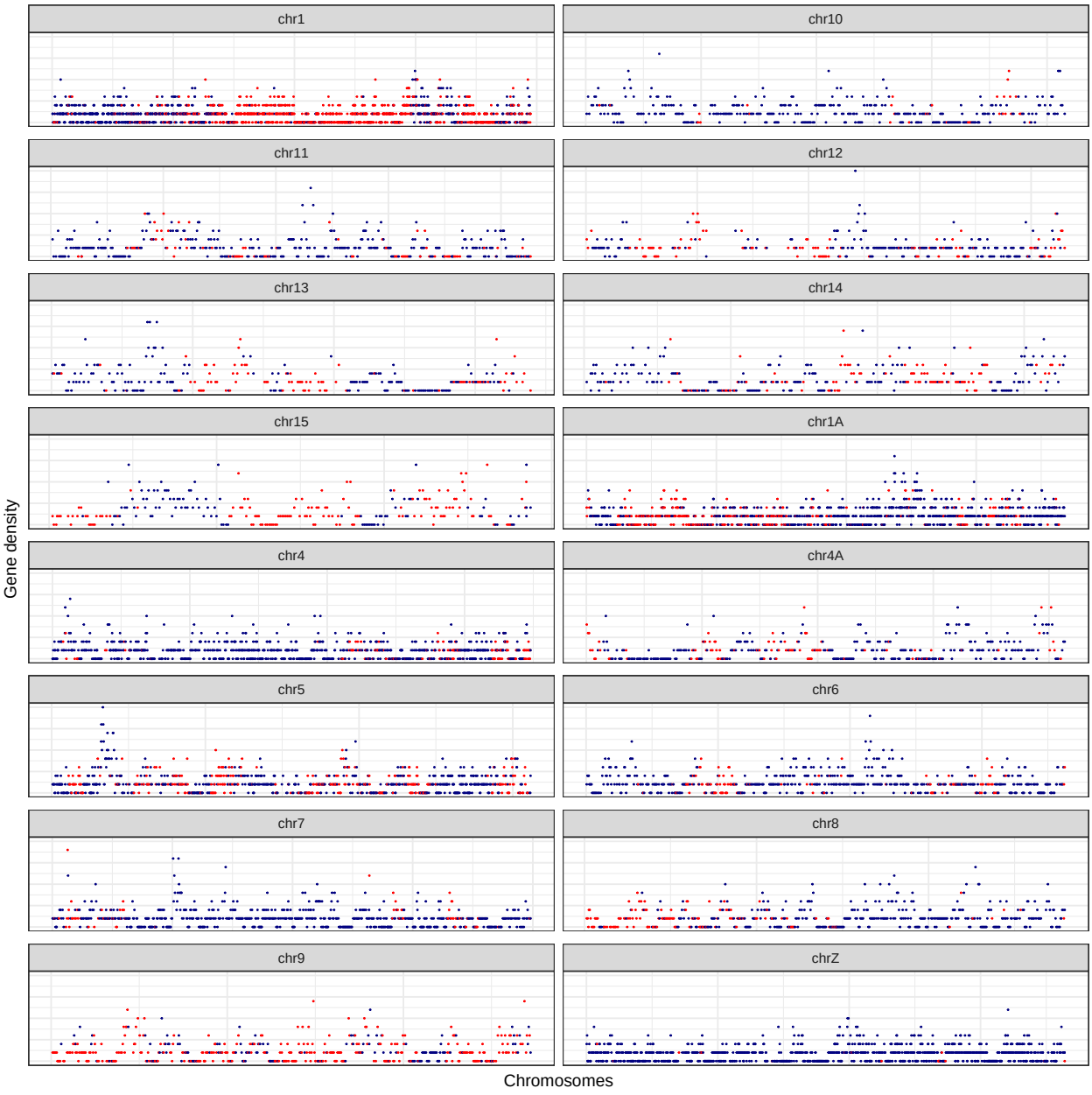
```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Z.species$Gene_Density and Z.flow$Gene_Density
```

```
## W = 14242, p-value = 0.02385
## alternative hypothesis: true location shift is not equal to 0
```

```
# boxplot
mydata <- mydata[mydata$TREE_TYPE=="Species_Tree"|mydata$TREE_TYPE=="GeneFlow_Tree_sca",]
mydata$CHR_TYPE <- ifelse(mydata$CHR_zebra=="chrZ", "Z", "Auto")
library(ggplot2)
ggplot(mydata, aes(CHR_TYPE, Gene_Density, colour=TREE_TYPE)) + geom_violin()  + xlab("") + theme_bw() +
```




```
# For each chromosome
ggplot(mydata) + geom_point(aes(POS_zebra, Gene_Density, colour=TREE_TYPE), size=0.1) + scale_colour_ma
```

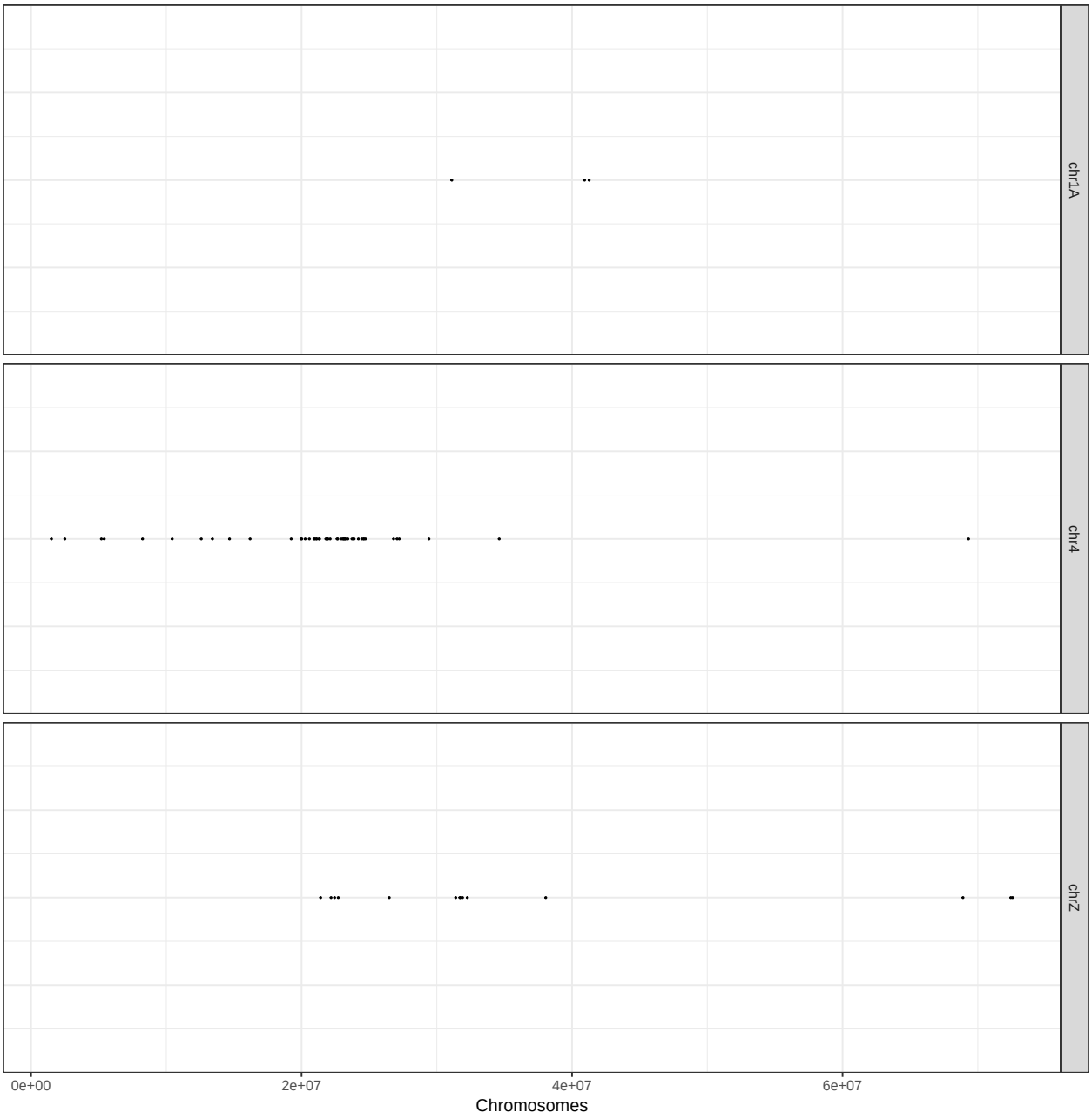


- Introgressed regions in fortis hybrids

```
mydata <- read.table("geneDensity/all.snp.midpoint.assign.class.trees.zebra.RR.geneDensity.bed", header=
mydata <- mydata[,c(1,2,4,5,6,8,9,13,14)]
colnames(mydata) <- c("CHR_zebra","POS_zebra","CHR","BIN_START","N_SNP","REF_TREE","TREE_TYPE","RR_zebra
mydata <- mydata[mydata$TREE_TYPE=="GeneFlow_Tree_for",]
nrow(mydata)
```

```
## [1] 66
```

```
library(ggplot2)
ggplot(mydata, aes(POS_zebra,1)) + geom_point(size=0.2) + facet_grid(CHR_zebra~., scales = "free_x")  +
```



FST ratio

- Matt suggested to calculate the FST between fortis_Early/Late_blunt and fortis_Late_pointed and FST between fortis_Late_pointed and scandens_Early/Late_pointed, and then caluclate the rotio of the FSTs.

- Measure pooled FST with Popoolation2
- "We furthermore recommend that the pool size (number of individuals in the pool) should be larger than the coverage. This minimizes re-sampling the same allele from a single individual several times."

- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/FST_ratio

```
module load bioinfo-tools samtools
module load popoolation2/1201

# Runtime: ~2 days
samtools mpileup -B -q 20 /proj/uppstore2017190/b2012111_nobackup/private/Sangeet_files/New_Sequences/P
java -ea -Xmx40g -jar /sw/apps/bioinfo/popoolation2/1201/rackham/mpileup2sync.jar --input Early_Late.mp
rm Early_Late.mpileup
perl /sw/apps/bioinfo/popoolation2/1201/rackham/fst-sliding.pl --input Early_Late.sync --output Early_La

grep -Ff ../../2019_correctedSNPs/Auto_scaffold.list Early_Late.10k.fst > Early_Late.10k.Auto.fst
grep -Ff ../../2019_correctedSNPs/Z_scaffold.list Early_Late.10k.fst > Early_Late.10k.Z.fst

# Give a shot to 50kb window
perl /sw/apps/bioinfo/popoolation2/1201/rackham/fst-sliding.pl --input Early_Late.sync --output Early_La

# Mapped them onto zebra finch genome
./convert.sh Early_Late.10k.fst
```

- Plot for all pairwise ZFST

```
FST <- read.table("FST_ratio/Early_Late.50k.Auto.fst", header=F, sep="\t")
FST <- data.frame(lapply(FST, function(x) {gsub("\\d:\\d=", "", x)}))
pop <- c("sca_Early_p","sca_Late_b","sca_Late_p","for_Early_b","for_Late_b","for_Late_p")
pairwise_pop <- c()
for(i in 1:(length(pop)-1)){
  for(j in (i+1):length(pop)){
    tmp <- paste(pop[i], pop[j], sep=".")
    pairwise_pop <- c(pairwise_pop, tmp)
    }
  }

colnames(FST) <- c("CHR", "BIN_MID","N_SNP", "FRAC_COV","MIN_COV",pairwise_pop)
FST.df <- as.data.frame(FST)

FST.df[,2:20] <- sapply(FST.df[,2:20], as.character)
FST.df[,2:20] <- sapply(FST.df[,2:20], as.numeric)
# histogram of N_SNP
hist(FST.df$N_SNP, breaks=1000)
hist(FST.df$FRAC_COV, breaks=1000)
hist(FST.df$MIN_COV, breaks=1000)
#FST.df <- FST.df[FST.df$N_SNP >= 20 & FST.df$N_SNP<=1000 & FST.df$FRAC_COV >= 0.6 & FST.df$MIN_COV >= 
FST.df <- FST.df[complete.cases(FST.df),]
FST.df$index <- 1:nrow(FST.df)
N_chr <- length(unique(FST.df$CHR))

library(ggplot2)
for(n in 6:20){
  tmp <- FST.df[,c(1,2,n,21)]
  # Normalize FST
  tmp$ZFST <- scale(tmp[,3])
  g<- ggplot(tmp) + geom_point(aes(x=index, y=ZFST, colour=CHR), size=0.1) + theme_bw() + scale_color_m
bitmap(paste("FST_ratio/ZFST_50kb/",colnames(tmp)[3],".png", sep=""), res=300)
print(g)
dev.off()
```

```
  }
```

- Plot for all pairwise ZFST on zebrafinch genome

```
FST <- read.table("FST_ratio/Early_Late.50k.fst.zebra.bed", header=F, sep="\t")
FST <- data.frame(lapply(FST, function(x) {gsub("\\d:\\d=", "", x)}))
pop <- c("sca_Early_p","sca_Late_b","sca_Late_p","for_Early_b","for_Late_b","for_Late_p")
pairwise_pop <- c()
for(i in 1:(length(pop)-1)){
  for(j in (i+1):length(pop)){
    tmp <- paste(pop[i], pop[j], sep=".")
    pairwise_pop <- c(pairwise_pop, tmp)
    }
  }

colnames(FST) <- c("zebra_CHR","zebra_POS1","zebra_POS2","CHR", "BIN_MID","N_SNP", "FRAC_COV","MIN_COV"
FST.df <- as.data.frame(FST)

FST.df[,c(2,3,5:23)] <- sapply(FST.df[,c(2,3,5:23)], as.character)
FST.df[,c(2,3,5:23)] <- sapply(FST.df[,c(2,3,5:23)], as.numeric)
#FST.df <- FST.df[FST.df$N_SNP >= 20 & FST.df$N_SNP<=250 & FST.df$FRAC_COV >= 0.8 & FST.df$MIN_COV >= 2

snp.right <- c("chr1","chr1A","chr2","chr3","chr4","chr4A","chr5","chr6","chr7","chr8","chr9","chr10","
FST.df <- FST.df[FST.df$zebra_CHR %in% snp.right, ]
FST.df <- FST.df[complete.cases(FST.df),]

FST.df$zebra_CHR <- factor(FST.df$zebra_CHR, levels=snp.right)
FST.df <- FST.df[order(FST.df$zebra_CHR,FST.df$zebra_POS1),]
FST.df$index <- 1:nrow(FST.df)
N_chr <- length(unique(FST.df$zebra_CHR))

library(ggplot2)
for(n in 9:23){
  tmp <- FST.df[,c(1,2,n,24)]
  # Normalize FST
  tmp$ZFST <- scale(tmp[,3])
  g<- ggplot(tmp) + geom_point(aes(x=index, y=ZFST, colour=zebra_CHR), size=0.05) + theme_bw() + scale_
bitmap(paste("FST_ratio/zebra_ZFST_50kb/",colnames(tmp)[3],".zebra.png", sep=""), res=300)
print(g)
dev.off()
  }
```

- Ratio of FST. If there is no introgression between fortis_Late_p and sca_Late_p, the ratio of FST(fortis_Early_b.fortis_Late_p):FST(fortis_Late_p:sca_Early_p) should be uniform along the genome. Otherwise, the ratio should be elevated because fortis_Late_p and sca_Late_p are very similar to each other.

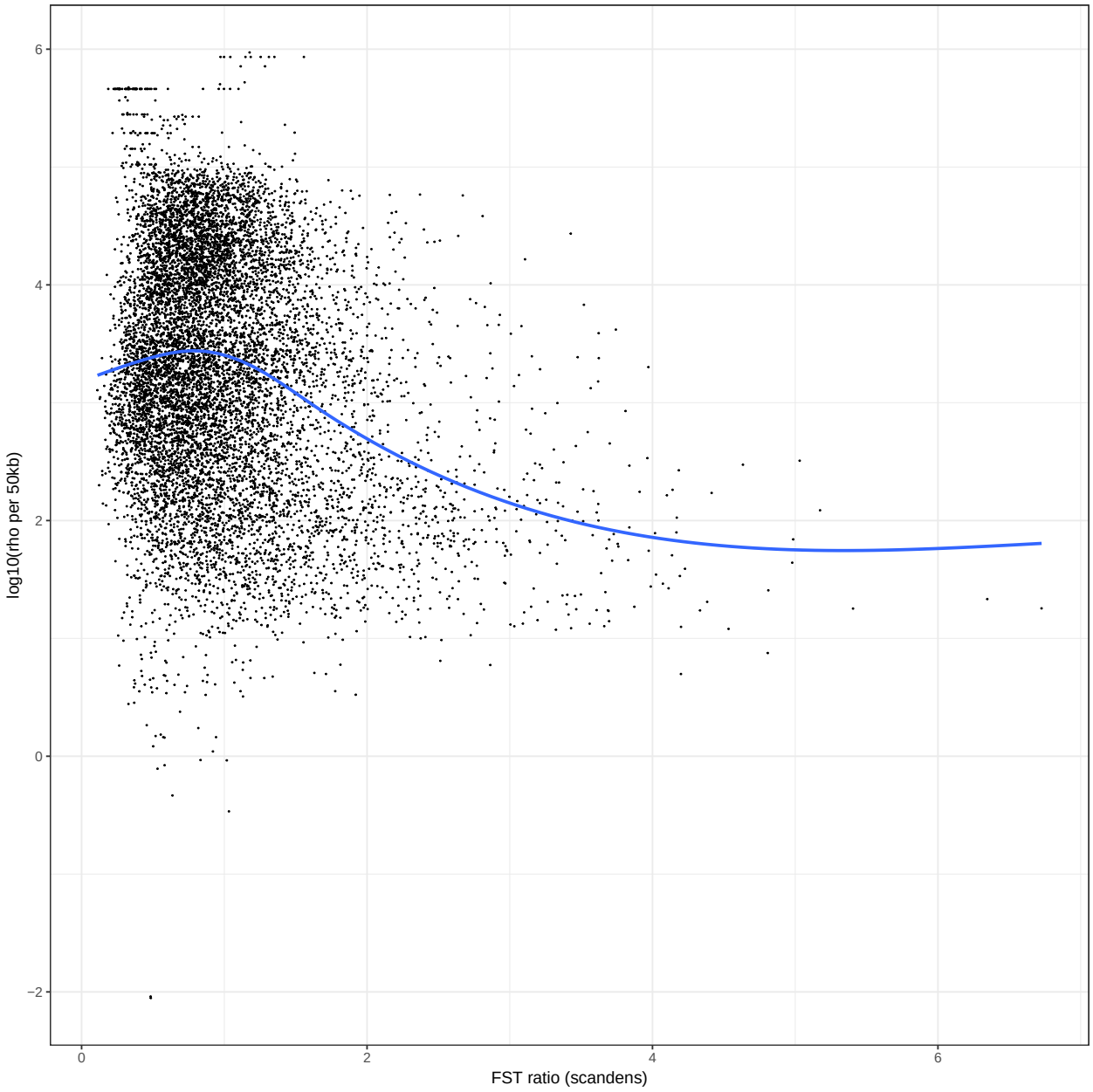
```
FST <- read.table("FST_ratio/Early_Late.50k.fst.zebra.RR.bed", header=F, sep="\t")
FST <- data.frame(lapply(FST, function(x) {gsub("\\d:\\d=", "", x)}))
pop <- c("sca_Early_p","sca_Late_b","sca_Late_p","for_Early_b","for_Late_b","for_Late_p")
pairwise_pop <- c()
for(i in 1:(length(pop)-1)){
  for(j in (i+1):length(pop)){
    tmp <- paste(pop[i], pop[j], sep=".")
    pairwise_pop <- c(pairwise_pop, tmp)
```

```
    }
  }

colnames(FST) <- c("zebra_CHR","zebra_POS1","zebra_POS2","CHR", "BIN_MID","N_SNP", "FRAC_COV","MIN_COV"

FST.df <- as.data.frame(FST)
FST.df[,c(2,3,5:23,25:27)] <- sapply(FST.df[,c(2,3,5:23,25:27)], as.character)
FST.df[,c(2,3,5:23,25:27)] <- sapply(FST.df[,c(2,3,5:23,25:27)], as.numeric)
# 10kb window
#FST.df <- FST.df[FST.df$N_SNP >= 20 & FST.df$N_SNP<=250 & FST.df$FRAC_COV >= 0.8 & FST.df$MIN_COV >= 2
# 50kb window
FST.df <- FST.df[FST.df$N_SNP >= 20 & FST.df$N_SNP<=1000 & FST.df$FRAC_COV >= 0.8 & FST.df$MIN_COV >= 2
# Calculate the ratio
FST.df$Ratio.for_Early_b <- FST.df$for_Early_b.for_Late_p/FST.df$sca_Early_p.for_Late_p
FST.df$Ratio.for_Late_b <- FST.df$for_Late_b.for_Late_p/FST.df$sca_Late_p.for_Late_p
FST.df$Ratio.sca_Early_p <- FST.df$sca_Early_p.sca_Late_b/FST.df$sca_Late_b.for_Early_b
FST.df$Ratio.sca_Late_p <- FST.df$sca_Late_b.sca_Late_p/FST.df$sca_Late_b.for_Late_b

FST.df <- FST.df[,c(1,2,4,5,27,28:31)]
# Plot
snp.right <- c("chr1","chr1A","chr2","chr3","chr4","chr4A","chr5","chr6","chr7","chr8","chr9","chr10","
FST.df <- FST.df[FST.df$zebra_CHR %in% snp.right, ]
FST.df <- FST.df[complete.cases(FST.df),]

FST.df$zebra_CHR <- factor(FST.df$zebra_CHR, levels=snp.right)
FST.df <- FST.df[order(FST.df$zebra_CHR,FST.df$zebra_POS1),]
FST.df$index <- 1:nrow(FST.df)
N_chr <- length(unique(FST.df$zebra_CHR))
if(FALSE){
g <- ggplot(FST.df) + geom_point(aes(x=index, y=Ratio.sca_Late_p, colour=zebra_CHR), size=0.05) + theme_
bitmap("FST_ratio/FST.Ratio.sca_Late_p.sca_Late_b.for_Late_b.png", res=300)
print(g)
dev.off()
}

# shared regions under gene flow (Ratio >1 )
# correlate it with RR
# fortis hybrid
FST.df$geneFlow_for <- ifelse(FST.df$Ratio.for_Early_b>1,"Y", "N")
# scandens hybrid
FST.df$geneFlow_sca <- ifelse(FST.df$Ratio.sca_Early_p>1,"Y", "N")

# coorelation test under different models
# linear regression assumption: data are normally distributed
FST.df <- FST.df[FST.df$zebra_CHR!="chrZ",]
linear <- lm( log10(RR) ~ Ratio.sca_Early_p, data=FST.df)
summary(linear)
```

```
##
## Call:
## lm(formula = log10(RR) ~ Ratio.sca_Early_p, data = FST.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -5.5461 -0.7313  0.0047  0.8147  2.8998
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        3.70190    0.01961  188.79   <2e-16 ***
## Ratio.sca_Early_p -0.42856    0.01697  -25.26   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9963 on 10042 degrees of freedom
## Multiple R-squared:  0.05974,    Adjusted R-squared:  0.05964
## F-statistic:   638 on 1 and 10042 DF,  p-value: < 2.2e-16
```

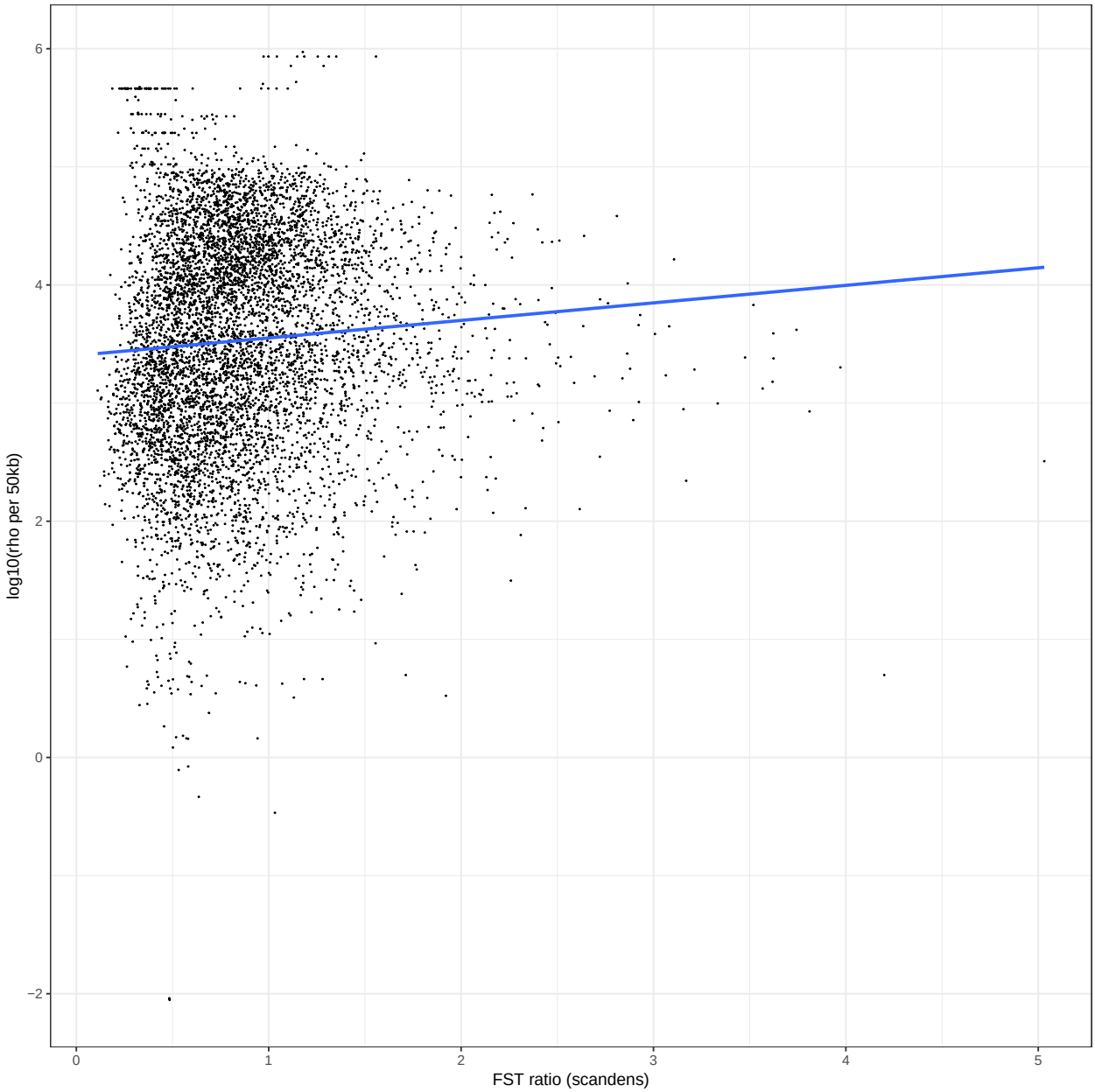
```
ggplot(FST.df, aes(Ratio.sca_Early_p, log10(RR))) + geom_point(size=0.1) + geom_smooth(method="gam", fo
```



```
print("Without chr1")
```

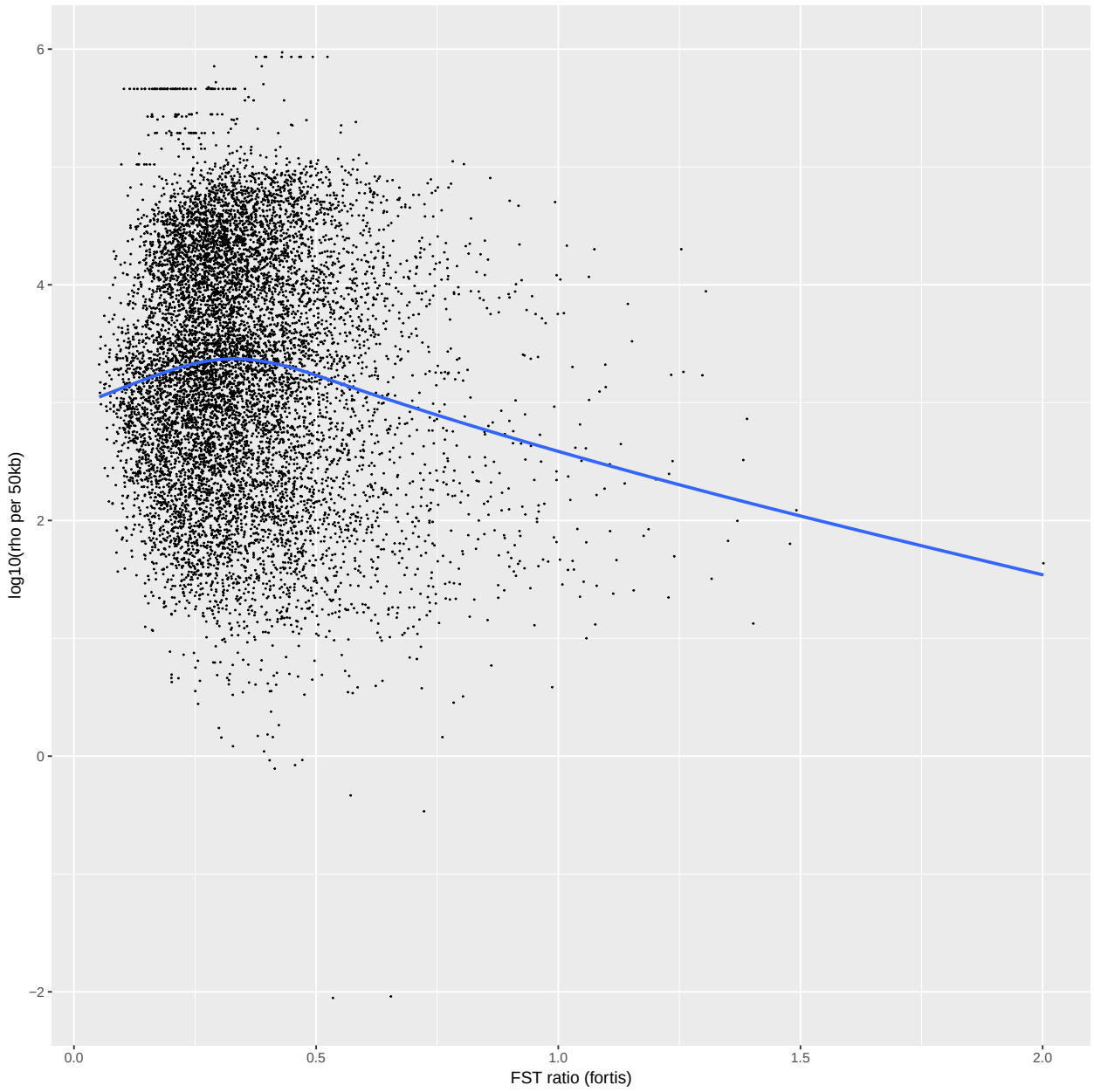
```
## [1] "Without chr1"
```

```
ggplot(FST.df[FST.df$zebra_CHR!="chr1"&FST.df$zebra_CHR!="chr1A",], aes(Ratio.sca_Early_p, log10(RR))) +
```




```
ggplot(FST.df, aes(Ratio.for_Early_b, log10(RR))) + geom_point(size=0.1) + geom_smooth(se=FALSE) + ylab
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

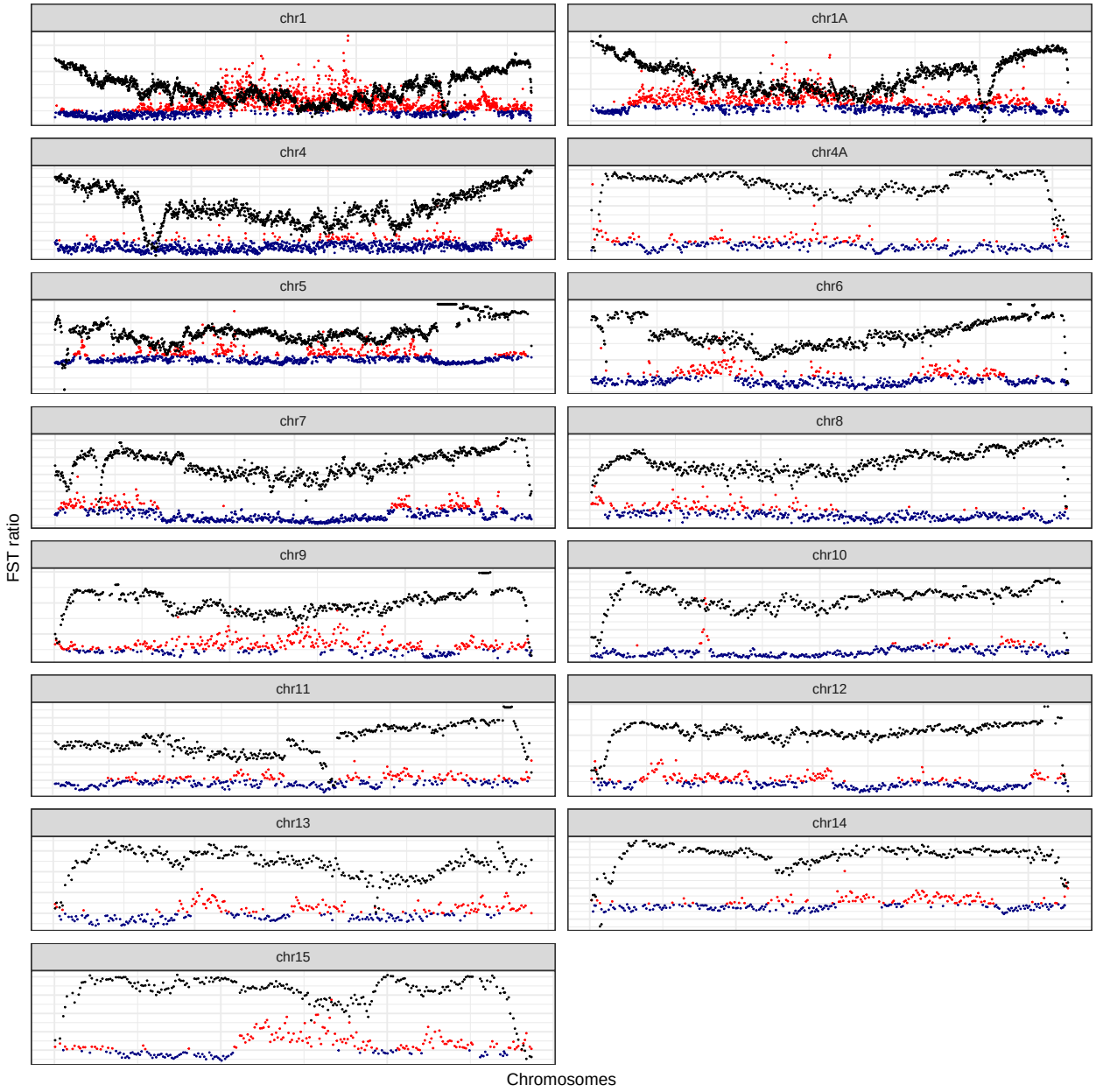



```
test1 <- FST.df[FST.df$geneFlow_sca == "Y"&FST.df$zebra_CHR!="chr1"&FST.df$zebra_CHR!="chr1A",]
test2 <- FST.df[FST.df$geneFlow_sca == "N"&FST.df$zebra_CHR!="chr1"&FST.df$zebra_CHR!="chr1A",]
linear1 <- lm(Ratio.sca_Early_p ~ log10(RR), data=test1)
summary(linear1)
```

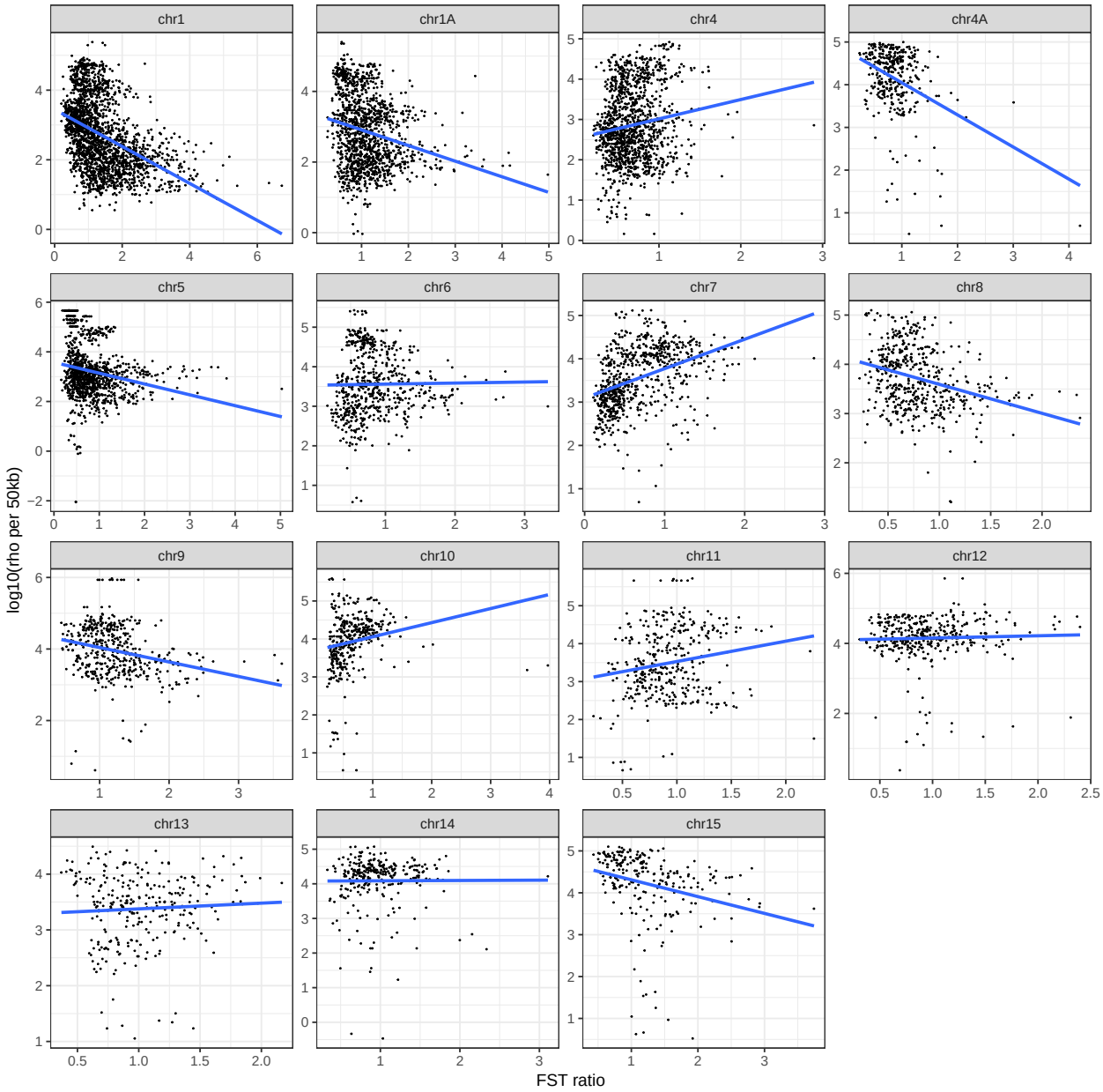
```
##
## Call:
## lm(formula = Ratio.sca_Early_p ~ log10(RR), data = test1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5479 -0.2561 -0.1163  0.1215  3.6039
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.55628    0.03914  39.758  < 2e-16 ***
## log10(RR)   -0.05129    0.01049  -4.888  1.1e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3985 on 1937 degrees of freedom
## Multiple R-squared:  0.01218,    Adjusted R-squared:  0.01168
## F-statistic: 23.89 on 1 and 1937 DF,  p-value: 1.102e-06
```

```
# distribution on each chromosome
ggplot(FST.df) + geom_point(aes(zebra_POS1, Ratio.sca_Early_p, colour=geneFlow_sca), size=0.1) + scale_
```




```
# correlation on each chromosome
ggplot(FST.df,aes(Ratio.sca_Early_p, log10(RR))) + geom_point( size=0.1)+ geom_smooth(method="lm",se=FA
```



- Correlate regions under gene flow with recombination rate in zebra finch
- pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/zebra_RR
- Data is from: http://science.sciencemag.org/content/350/6263/928

```
awk '{OFS="\t";print FILENAME,$0}' recombination_map/*.txt | grep -v -P "#|version" | sed 's/recombinat
awk '{OFS="\t";print $1,$2,$3,$5*($3-$2)}' zebrafinch_bpen5.RR | sort -k 1,1 -k 2,2n > zebrafinch_bpen5_
bedtools map -c 4 -o sum -a zebra_50k_window.bed -b zebrafinch_bpen5_RR.bed > zebrafinch_50k.RR
bedtools map -c 4 -o sum -a zebra_10k_window.bed -b zebrafinch_bpen5_RR.bed > zebrafinch_10k.RR
```

TreeMix analysis

Use TreeMix to infer the pattern of how fortis and scandens populations hybridized
No missing genotype is allowed otherwise TrrMix will throw out errors

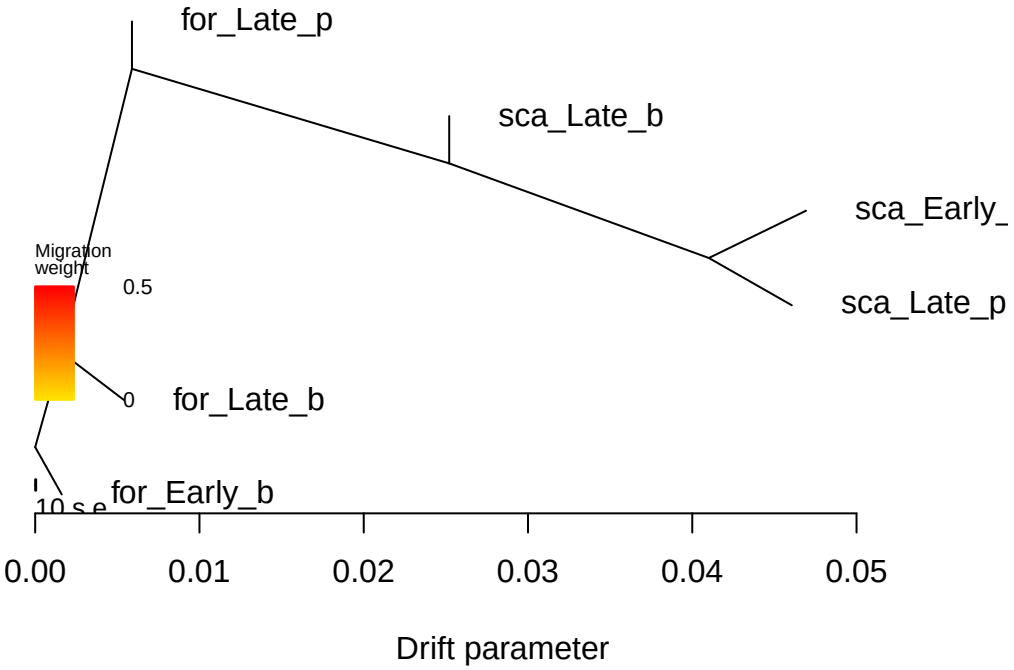
pwd: /proj/uppstore2017190/b2012111_nobackup/private/fan/fortis_scandens_pools/TreeMix

Run time 00:04:51

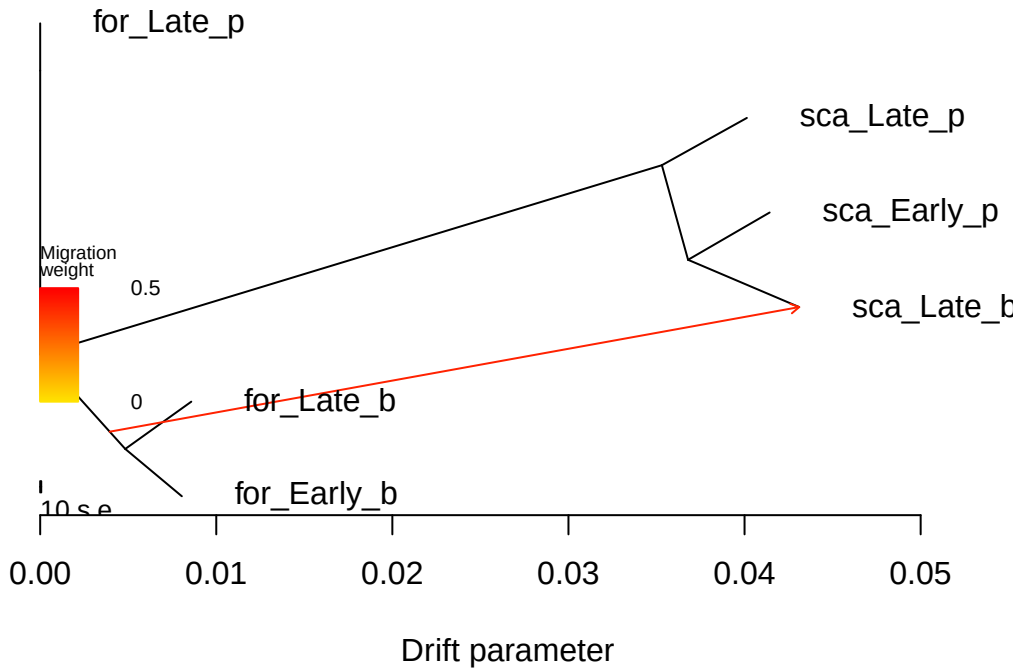
```
vcftools --gzvcf ../Early_Late.vcf.gz --out Early_Late --extract-FORMAT-info AD --max-missing 1

head -n 1 ../Early_Late.AD.FORMAT | cut -f 3-8 > Early_Late.Autosome.AD
grep -Ff ../../2019_correctedSNPs/Auto_scaffold.list ../Early_Late.AD.FORMAT | cut -f 3-8 >> Early_Late

gzip Early_Late.Autosome.AD

module load TreeMix/1.12
treemix -i Early_Late.AD.gz -o Early_Late_m0

treemix -i Early_Late.AD.gz -o Early_Late_m1 -m 1

treemix -i Early_Late.AD.gz -o Early_Late_m2 -m 2

treemix -i Early_Late.AD.gz -o Early_Late_m3 -m 3
```

Plot the trees in different -m parameters (migration)

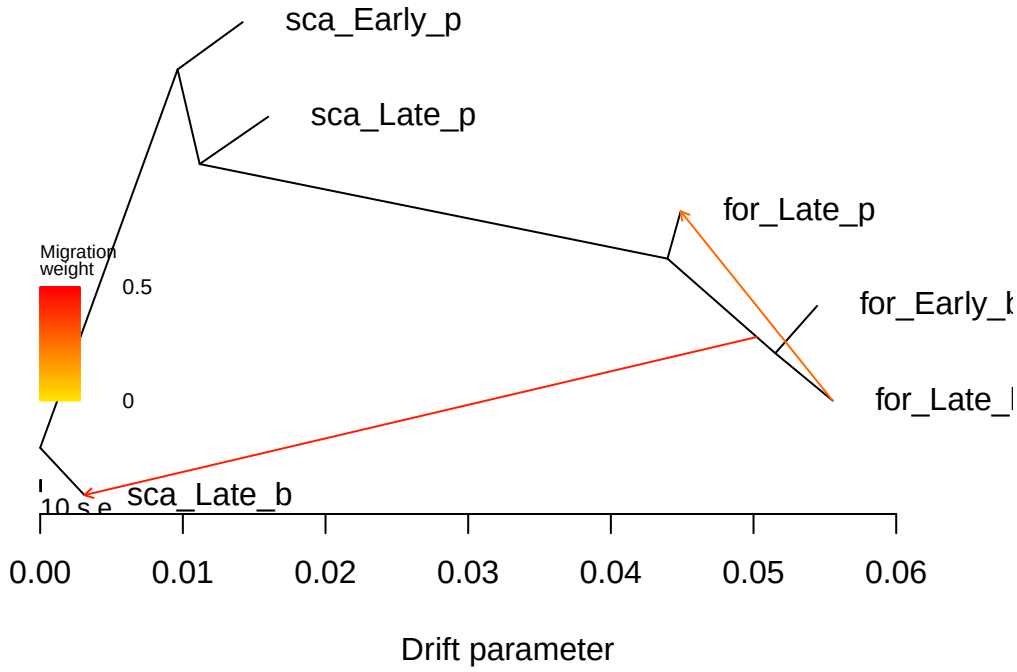
```
source("/Project/b2012111_DarwinsFinch/EarlyLate_FortisScandens/treemix-1.13/src/plotting_funcs.R")
# Autosome
plot_tree("TreeMix/Early_Late.Auto.m0")
```



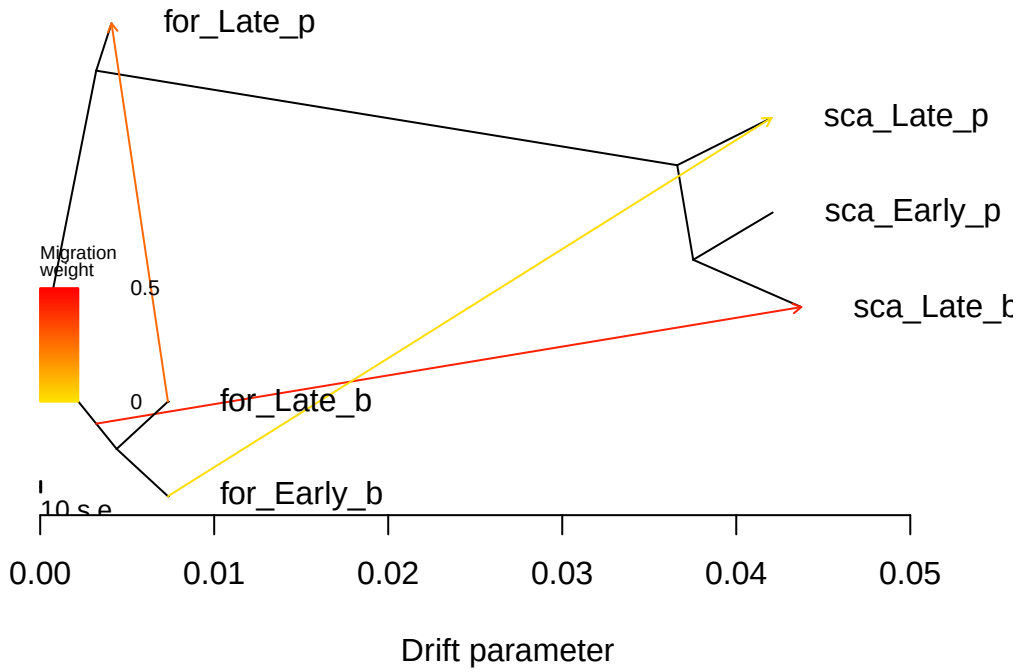

```
plot_tree("TreeMix/Early_Late.Auto.m1")
```



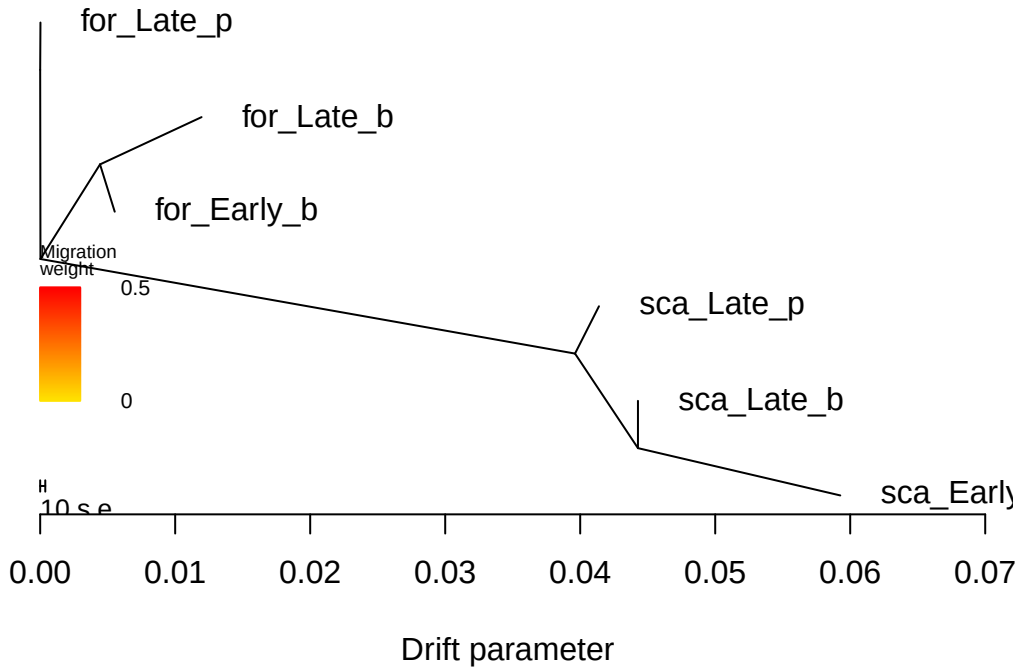

```
plot_tree("TreeMix/Early_Late.Auto.m2")
```



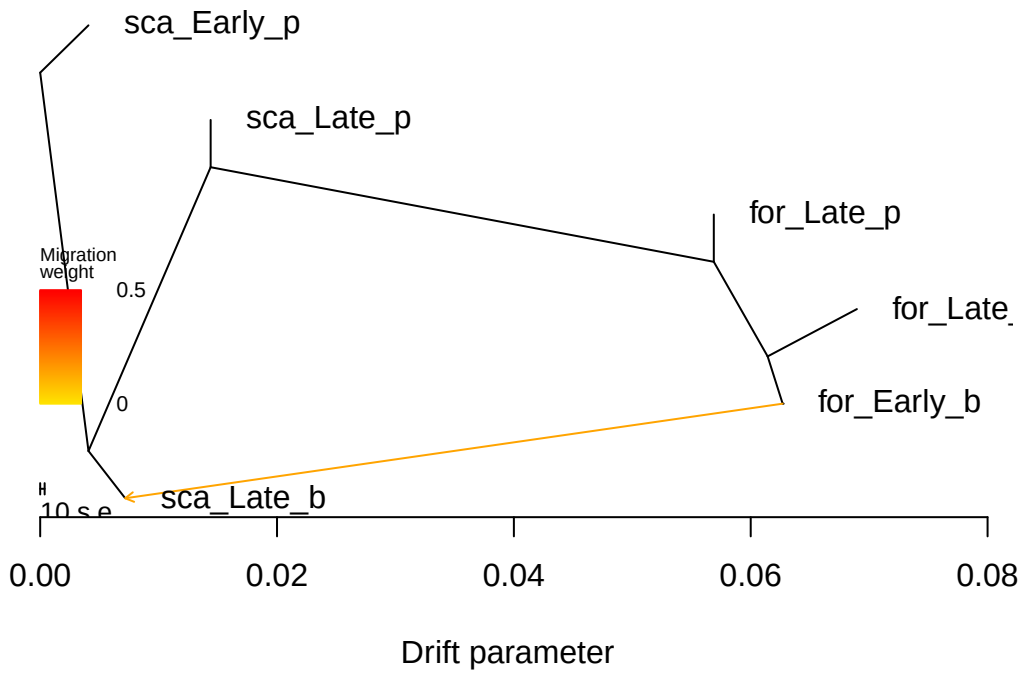

```
plot_tree("TreeMix/Early_Late.Auto.m3")
```



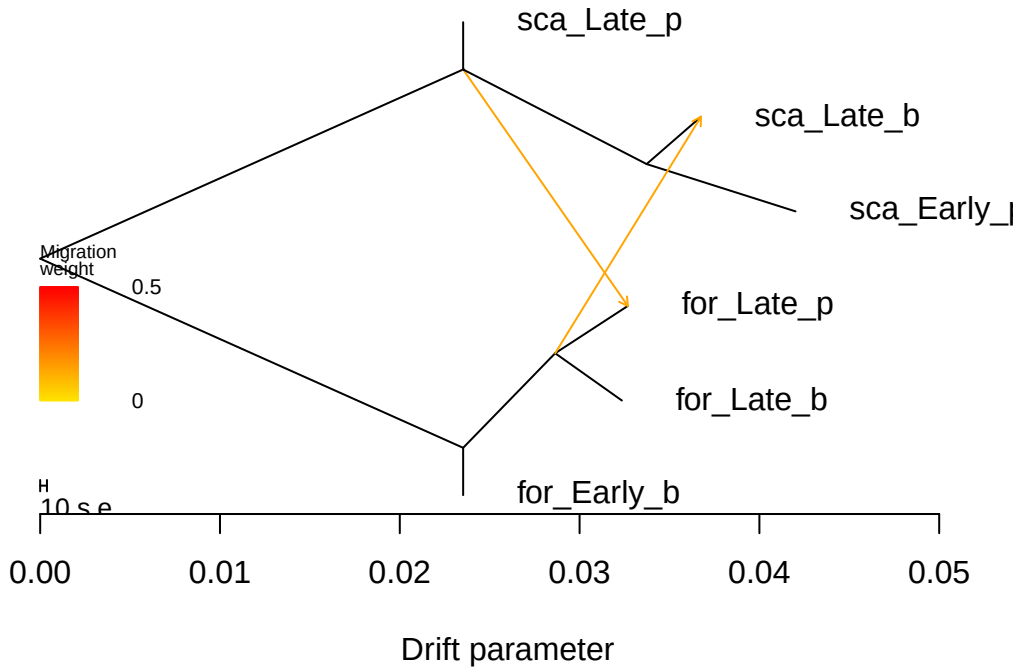

```
# Z
plot_tree("TreeMix/Early_Late.Z.m0")
```



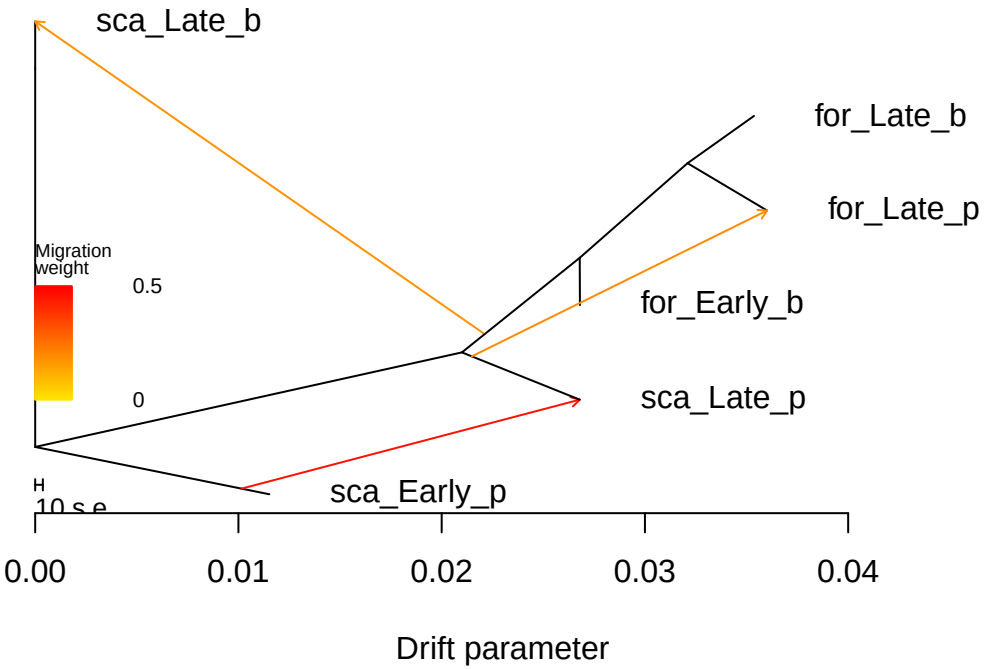

```
plot_tree("TreeMix/Early_Late.Z.m1")
```




```
plot_tree("TreeMix/Early_Late.Z.m2")
```




```
plot_tree("TreeMix/Early_Late.Z.m3")
```


sca_Late_b
for_Late_b
for_Late_p
Migration
weight
0.5
for_Early_b
0
sca_Late_p
H
10 s e
sca_Early_p
0.00
0.01
0.02
0.03
0.04
Drift parameter