Supplementary File

Across two continents: the genomic basis of environmental adaptation in house mice (Mus musculus domesticus) from the Americas

Yocelyn T. Gutierrez-Guerrero, Megan Phifer-Rixey, and Michael W. Nachman

Index

4- Environmental space distribution

4.1- Environmental variables and PCA

4.2- Mapping localities and mean annual temperature

5- Latent Factor Mixed Models- Genome Scan for local adaptation

5.1- Evaluate K

5.2- Running LFMM

5.3- GIF calibration

5.4- Adjust p-values and evaluate lamba

5.5- Manhattan plot

6- Annotation of candidate genes

6.1- Variant Effect Predictor annotation

6.2- Gene Ontology enrichment

7- Body weight Genome-Wide Association

7.1- Filtering adults

7.2- Run GEMMA

8- Parallel evolution in genes candidates

8.1- Allele frequencies and changes in frequencies in the same direction

1- Filtering and Mapping

1.1- Download data (SRA-NCBI)

Using the SRA ids from the BioProjects: PRJNA397150 and PRJNA718321

```
##Generate a variable with the SRA accession ID, for example:

# SRA= "SRR14097241 SRR7758152 SRR7758153 SRR7758154 SRR7758155 SRR7758156"

for s in ${SRA}
do
       fastq-dump --defline-seq '@$sn[_$rn]/$ri' --split-3 --outdir Rae_Reads
       --gzip ${s}

done
```

1.2- Cleaning raw reads

Cleaning raw reads by quality (PHRED >= 30), removing adapters (using AdapterRemoval) and possible contaminations

1_cleaning.sh

```
#!/bin/bash

cat Reads.txt | while read id
do
    AdapterRemoval --file1 ../${id}_R1_001.fastq --file2 ../${id}_R2_001.fastq
    --threads 10 --basename ${id}_adaptremov --output1 ${id}_R1_clean.fastq
    --output2 ${id}_R2_clean.fastq --minquality 30 --minlength 80

done
```

Removing contaminats using the E.coli genome assembly ASM584v2 and the software HISAT2

```
#!/bin/bash

hisat2-build GCF_000005845.2_ASM584v2_genomic.fna ecoli
cat Reads.txt | while read seq
do

    hisat2 -x ecol -1 ${seq}_R1_clean.fastq -2 ${seq}_R2_clean.fastq
    -S ${seq}_mapped.sam --un-conc ${seq}_clean_filter --threads 15
    rm ${seq}_mapped.sam

done
```

1.3- Mapping genomic reads

Using the M. musculus domesticus genome assembly GRCm38.p6 and the high-quality genomic reads.

The alignment was performed using BWA MEM, Samtools and Picard

2_bwa_mapping.sh

```
#!/bin/bash

bwa index GCF_000001635.26_GRCm38.p6_genomic.fna

mkdir Mapping_Metrics

cat Reads.id | while read exome
do

    bwa mem -M GCF_000001635.26_GRCm38.p6_genomic.fna ${exome}_R1.fastq
    ${exome}_R2.fastq -t 15 > ${exome}.sam

    java -jar picard.jar SortSam I= ${exome}.sam O= ${exome}_sorted.bam
    SORT_ORDER=coordinate

    rm ${exome}.sam

    java -jar picard.jar MarkDuplicates I= ${exome}_sorted.bam O= ${exome}.bam
    M= ${exome}_metrics.txt MAX_FILE_HANDLES_FOR_READ_ENDS_MAP=100

    mv ${exome}_metrics.txt Mapping_Metrics/

    rm ${exome}_sorted.bam

    samtools view -F 0x04 -b ${exome}.bam > ${exome}_align.bam

done
```

1.5- Recalibration and variant identification

Using GATK v4 and Picard to perform the variant identification (HaplotypeCaller). The variants (vcf files) were used to recalibrate the exome-seq alignments

```
#!/bin/bash
mkdir VCF_files
mkdir Recal_align

samtools faidx GCF_000001635.26_GRCm38.p6_genomic.fna
java -jar picard.jar CreateSequenceDictionary
R=GCF_000001635.26_GRCm38.p6_genomic.fna
O=GCF_000001635.26_GRCm38.p6_genomic.fna.dict

cat Reads.id | while read id
do

#AddOrReplaceReadGroups
    java -jar picard.jar AddOrReplaceReadGroups I= ${id}_align.bam
    O= VCF_files/${id}_2.bam RGID=1 RGLB=lib1 RGPL=illumina RGPU=${id} RGSM=${id}

#BuidBamIndex
    java -jar picard.jar BuildBamIndex I= VCF_files/${id}_2.bam

#HaplotypeCaller (raw variants identification)
    gatk --java-options "-Xmx16g -XX:ParallelGCThreads=10" HaplotypeCaller
```

```
    -R GCF_000001635.26_GRCm38.p6_genomic.fna -I VCF_files/${id}_2.bam
    -O VCF_files/${id}_1.vcf --pcr-indel-model CONSERVATIVE
    --native-pair-hmm-threads 10

#BaseRecalibrator
    gatk BaseRecalibrator -I VCF_files/${id}_2.bam
    -R GCF_000001635.26_GRCm38.p6_genomic.fna --known-sites VCF_files/${id}_1.vcf
    -O VCF_files/recal_${id}.table

    gatk ApplyBQSR -R GCF_000001635.26_GRCm38.p6_genomic.fna
    -I VCF_files/${id}_2.bam --bqsr-recal-file VCF_files/recal_${id}.table
    -O Recal_align/${id}_recalibration.bam

    gatk --java-options "-Xmx16g -XX:ParallelGCThreads=10" HaplotypeCaller
    -R GCF_000001635.26_GRCm38.p6_genomic.fna -I ${id}_recalibration.bam
    -O VCF_files/${id}_recal.vcf --native-pair-hmm-threads 10

done
```

1.6- Removing sex chromosomes from bam files

Removing sex chromosomes sequences from bam files and estimating alignment depth

```
#!/bin/bash

cat list.bam | while read bam
do

    samtools view -h ${bam}_recalibration.bam | awk '{if($3 != "NC_000086.7" &&
    $3 != "NC_000087.7"){print $0}}' | samtools view -Sb - > ${bam}_autosomes.bam
    samtools index -b ${bam}_autosomes.bam

done
```

1.7- Calculate depth mean coverage on BAM files using samtools and

Note: Only if every base is covered at least once

```
##Script name: mean_coverage.pl

($num,$den)=(0,0);
while ($cov=<STDIN>) {
    $num=$num+$cov;
    $den++;
}
$cov=$num/$den;
print "Mean Coverage = $cov\n";
```

```
#!/bin/bash

cat Reads.id | while read cov
do
```

```
    nohup samtools mpileup Recal_align/${cov}_recalibration.bam | awk '{print $4}'
    | perl mean_coverage.pl | sed "s/^/${cov}\t/g" >> Mapping_coverage.txt

done
```

1.8- Sex identity based on chromosome X and Y

```
#!/bin/bash

cat list.bam | while read x
do
##Chrm X
    samtools view -hb -@16 ${x} NC_000086.7 > ChrmX_${x}
    samtools depth -a ChrmX_${x} | awk '{sum+=$3; sumsq+=$3*$3}
    END { print "Average = ",sum/NR; print sqrt(sumsq/NR - (sum/NR)**2)}'
    | perl -pe 's/\n/ Sqrt = / unless eof' | sed "s/^/${x}\t/g" >> ChrmX_depth.txt
    rm ChrmX_${x}

##Chrm Y
    samtools view -hb -@16 ${x} NC_000087.7 > ChrmX_${x}
    samtools depth -a ChrmX_${x} | awk '{sum+=$3; sumsq+=$3*$3}
    END { print "Average = ",sum/NR; print sqrt(sumsq/NR - (sum/NR)**2)}'
    | perl -pe 's/\n/ Sqrt = / unless eof' | sed "s/^/${x}\t/g" >> ChrmY_depth.txt
    rm ChrmX_${x}
done
```

2- SNP calling and population genomic analysis

ANGSD version: 0.929-19-gb2b41b5

2.1- Distribution of quality scores

Adjust the mapping quality, individual depth and missing data.

Quality_ANGSD.sh

```
genome= "GCF_000001635.26_GRCm38.p6_genomic.fna"

angsd -bam list.bam -ref ${genome}
-remove_bads 1 -minMapQ 20 -only_proper_pairs 1 -baq 1 -setMinDepthInd 5
-minInd 86 -nind 86 -C 50 -doQsDist 1 -doCounts 1 -doDepth 1
-maxDepth 4000 -nthreads 12 -out SU_quality
```

Outputs:

Counts of quality scores: qc.qs

Counts of per-sampledepth: qc.depthSample

Counts of global depth: qc.depthGlobal

2.2- Estimate genotype likelihoods

Calculate the genotype likelihoods using a MAF cut-off 0.05

Note Adjust the parameters -minMapQ, -setMinDepthInd, -minInd, and -nind

```
mkdir Filter
genome= "GCF_000001635.26_GRCm38.p6_genomic.fna"

angsd -bam Population.txt -ref ${genome} -only_proper_pairs 1
-baq 1 -trim 0 -C 50 -minMapQ 20 -setMinDepthInd 3 -minInd 5 -nind 5
-uniqueOnly 1 -remove_bads 1 -skipTriallelic 1 -gl 1 -doCounts 1 -dosnpstat 1
-doPost 2 -doGeno 11 -domajorminor 4 -domaf 1 -minmaf 0.05 -snp_pval 0.001
-doHWE 1 -nthreads 15 -out Filter/Population_filter


zcat Population_filter.mafs.gz | awk ' NR>1 {printf ("%s\t%s\t%s\t%s\n",$1,$2,
$3,$4)}' > Filter/Population_filter_sites.txt

angsd sites index Filter/Population_filter_sites.txt
```

Genotype likelihoods and BCF file

```
genome= "GCF_000001635.26_GRCm38.p6_genomic.fna"
angsd -b Population.txt -ref ${genome} -uniqueOnly 1
-remove_bads 1 -only_proper_pairs 1 -trim 0 -C 50 -baq 1 -minMapQ 20
-setMinDepthInd 3 -geno_minDepth 5 -postCutoff 0.5 -skipTriallelic 1 -gl 1
-dopost 1 -domajorminor 1 -domaf 1 -dobcf 1 --ignore-RG 0 -dogeno 2 -docounts 1
-nthreads 15 -pest Filter/Population.sfs
-sites Filter/Population_filter_sites.txt -out ANGSD_vcf/Population_snps
```

2.3- Filter by MAF and HW

Using Plink, bcftools and vcftools

Merge the bcf files

```
cat list | while read bcf
do
    bgzip -c ${bcf} > ${bcf}.gz
    tabix ${bcf}.gz
done

bcftools merge Population_snps.bcf.gz Population2_snps.bcf.gz
Population3_snps.bcf.gz Population4_snps.bcf.gz Population5_snps.bcf.gz
Population6_snps.bcf.gz -o All_pops_merge.vcf.gz -O z --threads 4
```

```
## Filtering
plink --vcf SouthAmerica_pop_merge.vcf --maf 0.05 --hwe 0.001 --recode vcf
--geno 0.2 --out SouthAmerica_pops_filter.vcf.gz --allow-extra-chr
--biallelic-only --keep-allele-order

#or
vcftools --vcf SouthAmerica_merge.vcf --out SouthAmerica_missing02 --not-chr X
--maf 0.05 --hwe 0.001 --max-missing 0.8 --recode
```

2.4- Calculate Allele Frequencies

Using VCFtools

```
vcftools --vcf Population_gatk.vcf --freq --allow-extra-chr --biallelic-only
--keep-allele-order --out Population_AlleleFreq
```

2.5- Generate 012 genotype file

Convert to genotype format 012

```
gzip SouthAmerica_missing02.vcf.gz

vcftools --gzvcf SouthAmerica_missing02.vcf.gz --out S
outhAmerica_missing02_genotype.vcf --not-chr X --012
```

```
awk '{$1=""}1' SouthAmerica_missing02_genotype.vcf.012 |
awk -v OFS="\t" '{$1=$1}1'| sed 's/-1/9/g' > genotype.lfmm
```

3- Population genomic analysis

3.1- Phylogeny reconstruction

VCF to Nexus

```
vcftools --vcf Mmusculus_sp.vcf --out Mmusculus_sp_filter.vcf --not-chr X
--max-missing 0.8 --thin 100 --recode

bcftools view -e 'AC==0 || AC==AN' -m2 -M2 -O z -o Mmusculus_sp_filter2.vcf.gz
Mmusculus_sp_filter.vcf

gatk IndexFeatureFile -I Mmusculus_sp_filter2.vcf.gz

gatk SelectVariants -R GCF_000001635.26_GRCm38.p6_genomic.fna
-V Mmusculus_sp_filter2.vcf.gz --output Mmusculus_sp_final.vcf
--select-type-to-include SNP --restrict-alleles-to BIALLELIC

ruby convert_vcf_to_nexus.rb Mmusculus_sp_final.vcf Mmusculus_sp.nexus
```

SVDQuartets and PAUP

```
paup4a168_ubuntu64
exe Mmusculus_sp.nexus;
OUTGROUP ERR1124353;
SET root=outgroup;
SVDQuartets nquartets=500000 speciesTree taxpartition=none nrep=500
seed=1234568 nthreads=4 boostrap;
```

3.2- PCA Run ngsCovar

```
genome="GCF_000001635.26_GRCm38.p6_genomic.fna"

angsd -bam list.bam -ref ${genome}
-only_proper_pairs 1 -baq 1 -trim 0 -C 50 -minMapQ 20 -minQ 20
-setMinDepthInd 5 -setMaxDepth 4000 -minInd 86 -nind 86 -uniqueOnly 1
-remove_bads 1 -skipTriallelic 1 -gl 1 -domajorminor 1 -snp_pval 1e-3
-doHWE 1 -domaf 1 -minmaf 0.05 -doCounts 1 -doGlf 3 -dosnpstat 1 -doPost 1
-doGeno 32 -nthreads 12 -out All_pop/All_covar
```

Plot PCA #PopulationsMap.txt has two columns with headers (ID"^CLUSTERS)

```
Rscript plotPCA.R -i All_covar.matrix -c 1-2 -a PopulationsMap.txt
-o ALL.pca.pdf
```

3.3- Relatedness

Identify close relatives per population using ngsRelate

-L : Number of sites per population

Example:

```
ngsRelate -g Mmus_pop.glf.gz -n 10 -L 342642 -O Pop_Relate.txt -z Mex.txt -p 10

#Pairwise relatedness for each population using the rab statistic
#(Hedrick et al)

awk -F"\t" '{print$3"\t"$4"\t"$15}' Pop_Relate.txt | sed 1d > temp
cat Mex.txt | while read pop ; do grep "${pop}" temp | awk '{print$NF}'
| perl -pe 's/\n/, / unless eof' >> temp2 ; done
```

Pairwise relatedness heatmap

```
library(pheatmap)
rownames =c() #load ids
colnames =c() #load ids
mex <- matrix(c(), nrow=10, byrow = TRUE,
              dimnames = list(rownames, colnames)) #load temp2
color5 <- colorRampPalette(c("#a9b6d0","#F5F186","#FDB813"))
pheatmap(mex,col=color5(10), cex=1.1, main="Population",
         cutree_rows = 2, cutree_cols = 2)
```

3.4- Fst pair-wise calculation

```
#Heatmap
pfst <- read.table(file="FST_weighted.txt", header=TRUE, sep="\t")

an <- with(pfst, sort(unique(c(as.character(Pop1),
                               as.character(Pop2)))))
M <- array(0, c(length(an), length(an)), list(an, an))

i <- match(pfst$Pop1, an)

j <- match(pfst$Pop2, an)

M[cbind(i,j)] <- M[cbind(j,i)] <- pfst$fst

png("SouthAmerica_Fst_pairwise.png", width = 1000, height = 600)
corrplot::corrplot(M, is.corr = FALSE,type="upper",
                   diag=FALSE, method = "color", cl.cex = 1, tl.pos = 'n',
                   col=COL1('Blues',10), tl.cex =3)
```

3.5- Mantel Test

```
library(PopGenReport)
library(permute)
library(lattice)
library(vegan)

##From vcf to genid
##VCF file filter by missing data and HW
sa_vcf <- read.vcfR("Southamerica_Mmusculus.vcf", verbose = FALSE )
mmus_genind <- vcfR2genind(sa_vcf)
```

```
##Check the ids
ids <- mmus_genind$tab[,1]

##Generate Population factor
pop_vcf <- read.table(file="Population_genid.txt", header=FALSE, sep="\t")
colnames(pop_vcf) <- c("Ind","Pops")
strata(mmus_genind) <- as.data.frame(pop_vcf)
setPop(mmus_genind) <- ~Pops
mmus_genind$pop

##Add Coordinates
coords <- read.table(file="Population_coords.txt", header=FALSE, sep="\t")
colnames(coords) <- c("Ind","Lat","Lon")
#### First longitude and second latitude
mmus_genind@other$xy<-coords[,2:3]

mmus_genpop <- adegenet::genind2genpop(mmus_genind)
GD.pop.Nei <- adegenet::dist.genpop(mmus_genpop, method=1)
GD.pop.Edwards <- adegenet::dist.genpop(mmus_genpop, method=2)
GD.pop.Reynolds <- adegenet::dist.genpop(mmus_genpop, method=3)
GD.pop.Rogers <- adegenet::dist.genpop(mmus_genpop, method=4)
GD.pop.Provesti <- adegenet::dist.genpop(mmus_genpop, method=5)

##More methods
GD.pop.Joost <- mmod::pairwise_D(mmus_genind, linearized = FALSE)
GD.pop.Hedrick <- mmod::pairwise_Gst_Hedrick(mmus_genind, linearized = FALSE)
GD.pop.NeiGst <- mmod::pairwise_Gst_Nei(mmus_genind, linearized = FALSE)


##Calculate individual distances
hauss.dist <- dist(x=mmus_genind, method="euclidean", diag=T, upper=T)
hauss.gdist <- dist(x=mmus_genind$other$xy, method="euclidean", diag=TRUE,
upper=TRUE)

##Mantel Test
hauss.mantel <- mantel.randtest(m1=hauss.dist, m2=hauss.gdist, nrepet = 1000)

hauss.mantel.cor <- mantel.correlog(D.eco=hauss.dist,D.geo=hauss.gdist,XY=NULL,
n.class=0, break.pts=NULL, cutoff=FALSE, r.type="pearson",
nperm=1000, mult="holm", progressive=TRUE)

summary(hauss.mantel.cor)

##Distances Tree
plot(hclust(d=hauss.dist,method="complete"))
```

4- Environmental space distribution

4.1- Environmental variables and PCA

Bioclim variables obtained from WorldClim database

Principal Component to define environmental variables

```
require(tidyverse)
require(spData)
require(sf)
require(raster)
library(rasterVis)
library(rgdal)
require(mapview)
require(ggplot2)
require(sf)
require(maps)
require("rnaturalearth")
require("rnaturalearthdata")
library(sp)

##Download Bioclimatic
clim=getData('worldclim', var='bio', res=10)

##Extact bioclim data using the latutide and longitud
paths_capas <- list.files("wc_10/",pattern = "*.bil$",full.names = TRUE)
bios_wc <- stack(paths_capas)

data <- read.csv("Matriz_dat.csv",header=T, row.names = NULL)

e_vars <- extract(bios_wc, data[,c("LON","LAT")])

e_varsptos <- na.omit(e_vars)

write.csv(e_vars,"bios_Matriz_dat.csv",sep=",",row.names = TRUE,col.names = TRUE)
```

4.2- Mapping localities

WorldClim temperature dataset has a gain of 0.1, meaning that it must be multipled by 0.1 to convert back to degrees Celsius. Precipitation is in mm, so a gain of 0.1 would turn that into cm.

```
gain(clim)=0.1

##Coordinates and Bio value
clim <- raster("bio1.bil")
gain(clim)=0.1
plot(clim[[1]])
e <- extent(-95,-35,-60,15)
extent(r) <- e
r <- setExtent(r, e, keepres=TRUE)
climatefocus<-crop(clim,e)
```

```
##Convert Raster object into a Data frame
r.pts <- rasterToPoints(climatefocus, spatial=TRUE)
proj4string(r.pts)
geo.prj <- "+proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0"
r.pts <- spTransform(r.pts, CRS(geo.prj))
proj4string(r.pts)
r.pts_data <- data.frame(r.pts@data, long=coordinates(r.pts)[,1],
lat=coordinates(r.pts)[,2])
r.pts_data <- r.pts_data[,1:3]
```

5-Latent Factor Mixed Models: Genome Scan for local adaptation

Recommend **R** version **>=4.1**

Install the following libraries:

```
install.packages("RSpectra", dependencies=TRUE)
install.packaged("devtools", dependencies=TRUE)
devtools::install_github("bcm-uga/lfmm")

if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("LEA")

library(devtools)
devtools::install_github("bcm-uga/lfmm")
install.packages("foreach", dependencies=TRUE)
if(!("adegenet" %in% installed.packages()))
  {install.packages("adegenet")}
install.packages("pcadapt")
install_github("whitlock/OutFLANK")
library(OutFLANK)
library(pcadapt)
library(vcfR)
library(RSpectra)
library(qqman)
library(ggplot2)
library(LEA)
library(vegan)
library(lfmm)
library(qvalue)
```

5.1- Evaluate K

The number of genetic cluster=K can be explored using different methods, analyzing the admixture with ngsAdmix (see 1.4), perform a PCA or using the program snmf. Note: snmf is similar to STRUCTURE, but snmf is faster

```
#!/usr/bin/env Rscript

obj.snmf = snmf("genotype.lfmm", K = 1:12, project = "new",
```

```
                entropy = T, ploidy = 2, CPU = 15)
pdf("k_snmf.pdf")
plot(obj.snmf)

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

The best K is the minimum value. Note: For LFMM and LFMM2 is not necessary an accurate estimate of K, lfmm is to end with well-calibrated p-values, not to estimate genetic ancestry.

5.2- Running LFMM

```
library(RSpectra)
library(qqman)
library(ggplot2)
library(LEA)
library(vegan)
library(lfmm)

env <- read.table(file="LFMM_enviromental.txt", header=FALSE, sep="\t")
env <- env[,1]
env <- scale(env)

write.table(env, file="LFMM_environmental.env", quote=FALSE, row.names =FALSE)

project.lfmm=lfmm(input.file="LFMM_input.lfmm",
                  environment.file="LFMM_environmental.env", K=3,
                  iterations=500000, burnin=50000, repetitions=24, CPU=40)
```

5.3- GIF calibration

Changing the value of K influences the GIF and impacts the p-value distribution

An appropriately calibrated set GIF ~ 1

GIF > 1 Too many small p values. It may be overly liberal

GIF closer to 1 is the best

GIF < 1 Too many large p values. It may be too conservative

```
z.table = NULL
for (i in 1:24){
    file.name = paste("Pop_AlleleFreq_r", i, "_s1.2.zscore", sep="")
    z.table = cbind(z.table, read.table(file.name)[,1])
}
z.score = apply(z.table, MARGIN = 1, median)
write.table(z.score, "LFMM_zscoresK3.txt", quote=FALSE, row.names = FALSE)

## In LFMM, lambda is the Genome Inflation Factor (GIF),
#that it must be calibrated
lambda = median(z.score^2) / 0.456

lambda
```

5.4- Adjust p-values and evaluate lamba

```
ap.values = pchisq(z.score^2 / lambda, df = 1, lower = F)

hist(pv2$pvalues, col = "lightgreen", main="LFMM")

qqplot(rexp(length(pv2$pvalues), rate = log(10)), -log10(pv2$pvalues),
       xlab = "Expected quantile", pch = 19, cex = .4)
abline(0,1)

##Evaluate lambda manually
ap.values = pchisq(z.score^2 / 0.6, df = 1, lower = F)
hist(ap.values, col = "red")

ap.values = pchisq(z.score^2 / 0.5, df = 1, lower = F)
hist(ap.values, col = "red")
L = length (ap.values)

q = 0.1
w = which(sort(ap.values) < q * (1:L)/L)
candidates = order(ap.values)[w]

qobj <- qvalue(ap.values, fdr.level=0.10, pi0.method = "bootstrap")
summary(qobj)
```

5.5- Manhattan Plot

```
# Input file is a table separated by "\t" that contains: Chromosome, Position,
#Corrected pvalue, Zscore, Absolute Zscore, SNP

###Check that CHRM is numeric
str(tab_env)
###Change CHRM to numeric (if it is necessary)
tab_env[, 1] <- sapply(tab_env[, 1], as.numeric)

colnames(tab_env) <- c("CHR","BP","P","Pval_cal","Zscore","Score","SNP")

png("AMT_Exclusive2snp.png", width = 1000, height = 600)

manhattan2(amt, chrlabs=1:19, suggestiveline=-log10(0.05),
           genomewideline=-log10(0.001), col=c("gray65","gray29"),
           highlight =, main="Environmental LFMM outliers")
```

6- Annotation of candidate genes

6.1- Variant Effect Predictor Annotation by ENSEMBL

Mus musculus domesticus GRCm38.p6 genome annotation

Extracts candidates positions from a vcf file

```
vcftools --vcf SouthAmerica_all_merge_Chrm.vcf --positions Bp.txt --recode
--recode-INFO-all --out SouthAmerica_filter
```

```
./vep --cache -i SouthAmerica_filter.vcf -o SouthAmerica_filter_vep.txt
--species mus_musculus --refseq
```

6.2 Enrichment Gene Ontology

GO obtained from MGI- Mouse Genome Informatics database to generate the file annotation_allUniverse.txt

```
geneID2GO <- readMappings(file = "annotation_allUniverse.txt")
geneUniverse <- names(geneID2GO)
genesOfInterest <- read.table("Lat_uniq.genes", header=FALSE)
genesOfInterest <- as.character(genesOfInterest$V1)
geneList <- factor(as.integer(geneUniverse %in% genesOfInterest))
names(geneList) <- geneUniverse
myGOdataMF <- new("topGOdata", description="My project", ontology="MF",
                  allGenes=geneList,  annot = annFUN.gene2GO,
                  gene2GO = geneID2GO)
sg <- sigGenes(myGOdataMF)
str(sg)
resultFisher <- runTest(myGOdataMF, algorithm="weight01", statistic="fisher")
allRes <- GenTable(myGOdataMF, classicFisher = resultFisher,
                   orderBy = "resultFisher", ranksOf = "classicFisher",
                   topNodes = 100)
allRes$FDR <- p.adjust(allRes$classicFisher, method = "fdr")
write.table(allRes, "Lat_uniq_MF.txt", sep="\t", quote = FALSE, row.names=FALSE)

myGOdataCC <- new("topGOdata", description="My project", ontology="CC",
                  allGenes=geneList,  annot = annFUN.gene2GO, gene2GO = geneID2GO)
sg <- sigGenes(myGOdataCC)
str(sg)
resultFisher <- runTest(myGOdataCC, algorithm="weight01", statistic="fisher")
allRes <- GenTable(myGOdataCC, classicFisher = resultFisher,
                   orderBy = "resultFisher", ranksOf = "classicFisher",
                   topNodes = 100)
allRes$FDR <- p.adjust(allRes$classicFisher, method = "fdr")
write.table(allRes, "Lat_uniq_CC.txt", sep="\t", quote = FALSE, row.names=FALSE)

myGOdataBP <- new("topGOdata", description="My project", ontology="BP",
                  allGenes=geneList,annot = annFUN.gene2GO, gene2GO = geneID2GO)
sg <- sigGenes(myGOdataBP)
str(sg)
resultFisher <- runTest(myGOdataBP, algorithm="weight01", statistic="fisher")
allRes <- GenTable(myGOdataBP, classicFisher = resultFisher,
                   orderBy = "resultFisher", ranksOf = "classicFisher",
                   topNodes = 100)
allRes$FDR <- p.adjust(allRes$classicFisher, method = "fdr")
write.table(allRes, "Lat_uniq_BP.txt", sep="\t", quote = FALSE, row.names=FALSE)
```

7- Body weight Genome-Wide Association

7.1- Filtering adults

Prepare the inputs No missing data allowed or it must be imputed

```
./plink2 --vcf SouthAmerica_merge.vcf.gz --maf 0.05 --recode vcf --geno 0
--out SouthAmerica_nomissing.vcf --allow-extra-chr --max-alleles 2
--keep-allele-order

./plink2 --vcf SouthAmerica_nomissing.vcf --make-bed
--out SouthAmerica_nomissing --allow-extra-chr
```

```
vcftools --vcf SouthAmerica_MmusAdults.vcf --out SouthAmerica_MmusAdults02
--not-chr X --maf 0.05 --hwe 0.001 --max-missing 0.8 --recode
--min-alleles 2 --max-alleles 2


vcftools --vcf SouthAmerica_MmusAdults02_recode.vcf
--out SouthAmerica_MmusAdults02.txt --not-chr X --012

##Remove first column
awk '{$1=""}1' SouthAmerica_MmusAdults02.txt.012 | awk -vOFS="\t" '{$1=$1}1' |
awk '{print$1}' > SouthAmerica_MmusAdults012.tx
```

7.2- Run GEMMA

```
####Relatedness analysis
gemma -g SouthAmerica_MmAdults012_geno.txt
-p SouthAmerica_MmAdultsBodyLength_pheno.txt -gk 1 -miss 0.05 -o MmuSA_kinship

gemma -g SouthAmerica_MmAdults012_geno.txt
-p SouthAmerica_MmAdultsBodyLength_pheno.txt -k output/MmuSA_kinship.cXX.txt
-lmm 4 -miss 0.05 -c Sex_covar.txt -o SouthAmerica_Bodylenght_GEMMA_out
```

```
###QQ-plot to check inflation, q-q plot is commonly used to assess inflation
gwscan1 <- read.table(file="SouthAmerica_BodyLength_GEMMA_1out.assoc.txt",
                      header=TRUE, sep="\t")
gwscan2 <- read.table(file="SouthAmerica_Bodylength_GEMMA_2out.assoc.txt",
                      header=TRUE, sep="\t")
p1 <- plot.inflation(gwscan1$p_lrt)
print(p1)

p2 <- plot.inflation(gwscan2$p_lrt)
print(p2)
```

```
##Visualization Manhattan GWAS plot
library(ggplot2)
library(cowplot)
library(qvalue)
theme_set(theme_cowplot())
```

```
bw_map <-read.table(file="BodyWeight_gwas_positions.txt",
                    header=TRUE, sep="\t")

####Analyzed GEMMA

bw <- read.table(file="SouthAmerica_Bodylength_GEMMA_2out.assoc.txt",
                 header=TRUE, sep="\t")
bw$FDR <- p.adjust(bw$p_lrt, method = "fdr")
qbw <- qvalue(bw$p_lrt)
bw$qval <- qbw$qvalues

gwscan2 <- bw[,c(4:15)]
gwscan <- cbind(bw_map,gwscan2)

n <- nrow(gwscan)
gwscan <- cbind(gwscan,marker = 1:n)

gwscan[, 1] <- sapply(gwscan[, 1], as.numeric)
gwscan[, 2] <- sapply(gwscan[, 2], as.numeric)

gwscan <- transform(gwscan,p_lrt = -log10(p_lrt))
gwscan <- transform(gwscan,odd.chr = (CHR %% 2) == 1)
x.chr  <- tapply(gwscan$marker,gwscan$CHR,mean)

bw_candidates <- head(bw[order(bw$lrt),],10)
write.table(bw_candidates, file="BW_GEMMA_candidates.txt",
            sep="\t", quote=FALSE, row.names = FALSE)

####Plotting
ggplot(gwscan,aes(x = marker,y = p_lrt,color = odd.chr)) +
    geom_point(size = 4,shape = 20) +
    scale_x_continuous(breaks = x.chr,labels = 1:19) +
    scale_color_manual(values = c("gray74","gray25"),guide = "none") +
    labs(x = "",y = "-log10 p-value") +
    theme_cowplot(font_size = 20) +
    theme(axis.line = element_blank(),
          axis.ticks.x = element_blank(), axis.text.x = element_text(size = 14),
          axis.text.y = element_text(size = 16),
          axis.title = element_text(size = 20))
          + geom_hline(yintercept=5, col="blue", linetype="dashed" )
```

8- Parallel evolution in genes candidates

Allele frequencies and changes in frequencies in the same direction

```
#Calculate allele frequencies for each population

cat list_genes.txt | while read freq
do
```

```
    bcftools view -S Pop.txt ${freq}.vcf > temp.vcf

    vcftools --vcf temp.vcf --freq --out ${freq}

    awk '{print$1":"$2}' ${freq}.frq | sed '1d' | perl -pe 's/\n/\t/ unless eof'
    > temp

    awk -F "\t" '{print$5";"$6}' ${freq}.frq | sed '1d' |  awk -F";" '{print$1}'
    | awk -F":" '{print$NF}' | perl -pe 's/\n/\t/ unless eof' > temp2

    cat temp temp2 > ${freq}_frequency.txt

    rm temp temp2 temp.vcf ${freq}.freq ${freq}.log ${freq}.vcf *.frq

done
```

```
file_list <- list.files(pattern = "*_frequency.txt", full.names = TRUE)

for (file in file_list) {
  Pop <- read.table(file, header=TRUE, sep="\t")

  # Replace 0 with NA
  Pop[Pop == 0] <- NA

  # Remove the first column (POP column)
  Pop <- Pop[, -1]

  # Perform the comparison
  parall <- Pop[c(1, 3, 5), ] > Pop[c(2, 4, 6), ]

  # Select columns where all values are TRUE
  true_columns <- colnames(parall)[apply(parall, 2, all)]

  # Select columns where all values are FALSE
  false_columns <- colnames(parall)[apply(parall, 2, function(x) all(!x))]

  # Check if selected columns exist in the dataframe
  true_columns <- intersect(true_columns, colnames(Pop))
  false_columns <- intersect(false_columns, colnames(Pop))

  # Create empty data frames to store results
  true_df <- data.frame()
  false_df <- data.frame()

  if (length(true_columns) > 0) {
    true_df <- Pop[, true_columns, drop = FALSE]
  }
  if (length(false_columns) > 0) {
    false_df <- Pop[, false_columns, drop = FALSE]
  }

  # Check if true_df and false_df are empty
  if (nrow(true_df) > 0 && nrow(false_df) > 0) {
```

```
    # Combine the data frames if both are non-empty
    results <- cbind(true_df, false_df)
  } else if (nrow(true_df) > 0) {
    # Use true_df if false_df is empty
    results <- true_df
  } else if (nrow(false_df) > 0) {
    # Use false_df if true_df is empty
    results <- false_df
  } else {
    # If both data frames are empty, create an empty data frame
    results <- data.frame()
  }

  new_file <- sub("_frequency.txt", "_Freq_new.txt", basename(file))

  # Save the transposed data to a new file
  write.table(results, new_file, sep="\t", row.names = FALSE, col.names = TRUE, quote=FALSE)
}
```