SNV calling practical

CNRS Workshop 2023

Carina Mugal, Anouk Necsulea and Annabelle Haudry

SNV calling and detection of balancing selection in *Ficedula* flycatchers

Table of Contents

Introduction ***2***

Variant calling in Ficedula flycatchers (part I) ***3***

Download data to your working directory **3**

Variant calling in cohort with GATK **4**

Filtering of the VCF file **5**

Visualization of read depth / coverage **6**

Detection of balancing selection in Ficedula flycatchers (part II) ***8***

Access and installation of the work environment **8**

Preparation of data files for downstream analysis **8**

Description of genetic diversity **9**

Running BalLeRMix **10**

References ***12***

Introduction

Understanding how species adapt to their environment is a central question in evolutionary biology. Notably, it is still unclear how much of the observed phenotypic diversity in nature can be explained by neutral versus selective processes. The study of the genetic variation within species can allow us to reconstruct their evolutionary history, and to trace back adaptive events. Among the different adaptive forces, positive and negative selection are characterized by a reduction of genetic diversity within species, while balancing selection is defined as a process that maintains genetic diversity. There are different mechanisms that result in balancing selection: heterozygotes advantage (heterozygote individuals have a higher fitness than either homozygote), negative frequency-dependent selection (the rarer forms have a higher fitness) or fluctuating selection over time or space (Charlesworth, 2006). In addition, host-pathogen co-evolutionary dynamics can result in such maintenance of genetic diversity (Ebert & Fields, 2020). However, it is at present not known which of the different mechanisms is prevailing in nature, and if balancing makes a relevant contribution to species diversity.

The aim of this practical is to first perform SNV calling and second scans of balancing selection in the red-breasted flycatcher (RB), a passerine bird within the genus *Ficedula*, which together with the taiga flycatcher (TAI) forms a sister group to the four black-and-white flycatchers, collared flycatcher (C), pied flycatcher (P), atlas flycatcher (ATL) and semi-collared flycatcher (SEMI), Figure 1.

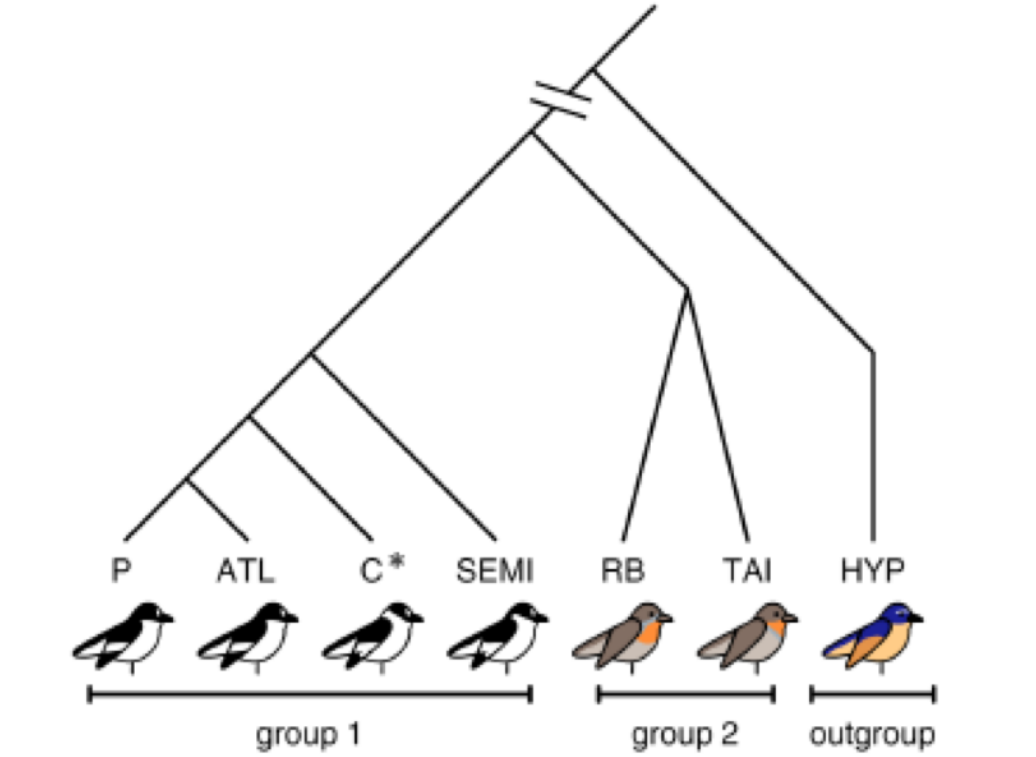

Figure 1: Topology of seven Ficedula flycatcher species, for which population re-sequencing data have been generated. Group 1 consists of the four black-and-white flycatchers, group 2 of the red-breasted and the taiga flycatcher, and the snowy-browed flycatcher (HYP) represents an outgroup species.

The black-and-white flycatchers, in particular the collared flycatcher and pied flycatcher, have been an ecological model system for speciation research for decades (Qvarnström et al. 2010) and, in the last 15 years, a growing resource of genomic data including a reference genome of the collared flycatcher has been acquired. Much less is known about their sister group, consisting of the red-breasted and the taiga flycatcher. Previous genetic analyses of the two species have been limited to a few mitochondrial and nuclear genes, with no genome-wide analyses (Hung & Zink, 2014; Zink et al., 2008). To advance our understanding of these two species, we have recently generated population re-sequencing data of 15 red-breasted and 65 taiga flycatcher individuals (Chase et al. 2021).

This practical will take advantage of this recent data set. Specifically, in the morning session (part I) the course participants will perform SNV calling in cohort for all 15 red-breasted flycatcher individuals. Population re-sequencing data have been mapped to the collared flycatcher reference genome (version 1.5) and are available as re-calibrated BAM files. From these BAM files, the course participants will first produce a raw variant call format (VCF) file for one scaffold, and then filter the VCF file based on criteria commonly applied within non-model organisms. To evaluate their filtering steps, the course participants will visualize relevant statistics of the VCF file in R. During the afternoon session (part II), the course participants will start from the filtered VCF files and perform scans of balancing selection with the BalLeRMix software (DeGiorgio et al, 2014; Cheng & DeGiorgio, 2019). Finally, the course participants will characterize genomic regions that show signatures of balancing selection and interpret their results, e.g. investigate if signatures of balancing selection are true positives or could represent methodological artifacts.

Variant calling in *Ficedula* flycatchers (part I)

Download data to your working directory

1. Access your working directory

```
ssh ubuntu@<host_name>
```

You should now be located in your working directory
/home/ubuntu

Check with the following command

```
pwd
```

2. Download relevant input files

For the execution of the practical, you need to download the input files from a fileserver

```
wget https://filesender.renater.fr/download.php?token=fdbbf359-f179-4f2c-af31-142025cc5bed\&files_ids=29157868 -O Variant_Calling.tar.gz
```

Decompress the downloaded data

```
tar -zxvf Variant_Calling.tar.gz
```

Change into the new directory

```
cd Variant_Calling
```

3. You will find the following files in your directory
- A genome assembly file, and the corresponding dictionary and index file
/home/ubuntu/Variant_Calling/assembly/fAlb15_MTmask_mtfZan.fasta

- A BAM file for one RB individual (Sample 31), and the corresponding index file
/home/ubuntu/Variant_Calling/bam/N00208_Sample_31.merged.recal.bam
/home/ubuntu/Variant_Calling/bam/N00208_Sample_31.merged.recal.bai

- GVCF files for 14 additional RB individuals for scaffold N00208
/home/ubuntu/Variant_Calling/gvcf/

- A map file that lists all 15 samples and their corresponding gvcf files for scaffold N00208
/home/ubuntu/Variant_Calling/map_file_15ind_N00208.txt

- A BED file for the repeat annotation
/home/ubuntu/Variant_Calling/fAlb15.fa.rm3.0.out.bed

- A filtered VCF file for another scaffold (N00135)
/home/ubuntu/Variant_Calling/vcf/N00135_15ind_allsites_finalFiltered.vcf.gz

Variant calling in cohort with GATK

Modify the input and output names appropriately. Inspect all input files and clarify for yourself what the command is doing.

Remember the workflow conventions and choose informative filenames!
Remember the workflow conventions and do not overwrite any files!

For help on using GATK, run
```
docker run broadinstitute/gatk:latest gatk --help
```

To print help for a particular tool (see below), run
```
docker run broadinstitute/gatk:latest gatk ToolName --help
```

1. Run GATK HaplotypeCaller on one sample (Sample 31), for one scaffold (N002008)
```
docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx16g" HaplotypeCaller -R {input.reference} -I {input.bam}
-O {output.g.vcf.gz} -ERC GVCF -L {scaff}
```

```
docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx16g" HaplotypeCaller -R
/home/ubuntu/Variant_Calling/assembly/fAlb15_MTmask_mtfZan.fasta -I
/home/ubuntu/Variant_Calling/bam/N00208_Sample_31.merged.recal.bam -O
/home/ubuntu/Variant_Calling/gvcf/RB_Sample_31.N00208.g.vcf.gz -ERC GVCF -L
N00208 (~28 minutes)
```

If you want to find out more about the HaplotypeCaller tool, run,
```
docker run broadinstitute/gatk:latest gatk HaplotypeCaller --help
```

2. Consolidate the GVCFs and create a database for all 15 samples,

```
mkdir temp_dir

docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx16g -Xms12g" GenomicsDBImport --genomicsdb-workspace-path
{output.db_path} -L {scaff} --tmp-dir temp_dir --sample-name-map
{input.map_file}

mkdir temp_dir

docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx16g -Xms12g" GenomicsDBImport --genomicsdb-workspace-path
/home/ubuntu/Variant_Calling/N00208_15ind.db_path -L N00208 --tmp-dir
/home/ubuntu/Variant_Calling/temp_dir --sample-name-map
/home/ubuntu/Variant_Calling/map_file_15ind_N00208.txt (~2 minutes)
```

If you want to find out more about the GenomicDBImport tool, run,

```
docker run broadinstitute/gatk:latest gatk GenomicsDBImport --help
```

3. Perform joint genotyping on all individuals

```
docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx12g" GenotypeGVCFs -R {input.reference} -V
gendb://{input.db_path} -all-sites -L {scaff} -O {output.vcf}

docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx12g" GenotypeGVCFs -R
/home/ubuntu/Variant_Calling/assembly/fAlb15_MTmask_mtfZan.fasta -V
gendb:///home/ubuntu/Variant_Calling/N00208_15ind.db_path -all-sites -L
N00208 -O /home/ubuntu/Variant_Calling/vcf/N00208_15ind_allsites.vcf (~4
minutes)
```

If you want to find out more about the GenotypeGVCFs tool, run,

```
docker run broadinstitute/gatk:latest gatk GenotypeGVCFs --help
```

Please inspect the header of the VCF file and compare it to the header of the GVCF file. What differences do you observe? Please also inspect the final rows of the VCF file and compare it to the final rows of the GVCF file. What differences do you observe?

Filtering of the VCF file

Modify the input and output names appropriately. Inspect all input files and clarify for yourself what the command is doing.

Remember the workflow conventions and choose informative filenames!
Remember the workflow conventions and do not overwrite any files!

1. Remove indels and keep only mono-allelic and bi-allelic sites

```
vcftools --vcf {input.vcf} --remove-indels --recode --max-alleles 2 --
recode-INFO-all --stdout | bgzip > {output.vcf.gz}

tabix -p vcf {input.vcf.gz}
```

```
vcftools --vcf vcf/N00208_15ind_allsites.vcf --remove-indels --recode --
max-alleles 2 --recode-INFO-all --stdout | bgzip >
vcf/N00208_15ind_allsites_removeIndels_max2.vcf.gz

tabix -p vcf vcf/N00208_15ind_allsites_removeIndels_max2.vcf.gz
```

2. Apply hard-filtering-criteria

```
docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx10G" VariantFiltration -R {input.reference} -V
{input.vcf.gz} -O {output.vcf.gz} --filter-expression "QD < 2.0 || FS >
60.0 || MQ < 40.0 || MQRankSum < -12.5 || StrandOddsRatio > 3 ||
ReadPosRankSum < -8.0" --filter-name "hard_filt" -L {scaff}

vcftools --gzvcf {input.vcf.gz} --remove-filtered-all --recode --recode-
INFO-all --stdout | bgzip > {output.vcf.gz}

docker run -v /home/ubuntu:/home/ubuntu broadinstitute/gatk:latest gatk --
java-options "-Xmx10G" VariantFiltration -R
/home/ubuntu/Variant_Calling/assembly/fAlb15_MTmask_mtfZan.fasta -V
/home/ubuntu/Variant_Calling/vcf/N00208_15ind_allsites_removeIndels_max2.vc
f.gz -O
/home/ubuntu/Variant_Calling/vcf/N00208_15ind_allsites_removeIndels_max2_ha
rdFilter.vcf.gz --filter-expression "QD < 2.0 || FS > 60.0 || MQ < 40.0 ||
MQRankSum < -12.5 || StrandOddsRatio > 3 || ReadPosRankSum < -8.0" --
filter-name "hard_filt" -L N00208

vcftools --gzvcf
vcf/N00208_15ind_allsites_removeIndels_max2_hardFilter.vcf.gz --remove-
filtered-all --recode --recode-INFO-all --stdout | bgzip >
vcf/N00208_15ind_allsites_removeIndels_max2_hardFiltered.vcf.gz
```

3. Remove repetitive DNA sequences included in the repeat annotation

```
vcftools --gzvcf {input.vcf.gz} --exclude-bed {repeat.bed} --recode --
recode-INFO-all --stdout | bgzip > {output.vcf.gz}

vcftools --gzvcf
vcf/N00208_15ind_allsites_removeIndels_max2_hardFiltered.vcf.gz --exclude-
bed fAlb15.fa.rm3.0.out.bed --recode --recode-INFO-all --stdout | bgzip >
vcf/N00208_15ind_allsites_removeIndels_max2_hardFiltered_repeatFiltered.vcf
.gz
```

4. Apply quality score filtering and remove sites with extreme coverage and many missing data

```
vcftools --gzvcf {input.vcf.gz} --minGQ 30 --minDP 5 --maxDP 200 --max-
missing 0.9 --recode --recode-INFO-all --stdout | bgzip > {output.vcf.gz}

vcftools --gzvcf
vcf/N00208_15ind_allsites_removeIndels_max2_hardFiltered_repeatFiltered.vcf
.gz --minGQ 30 --minDP 5 --maxDP 200 --max-missing 0.9 --recode --recode-
INFO-all --stdout | bgzip > vcf/N00208_15ind_allsites_finalFiltered.vcf.gz
```

Visualization of read depth / coverage

Modify the input and output names appropriately. Inspect all input files and clarify for yourself what the command is doing.

Remember the workflow conventions and choose informative filenames!
Remember the workflow conventions and do not overwrite any files!

1. Compute coverage per site, for two scaffolds (N00135 and N00208)

```
vcftools --gzvcf {input.vcf.gz} --site-mean-depth --out {output}
```

```
vcftools --gzvcf
/home/ubuntu/Variant_Calling/vcf/N00135_15ind_allsites_finalFiltered.vcf.gz
--site-mean-depth --out N00135_15ind_allsites_finalFiltered
```

```
vcftools --gzvcf
/home/ubuntu/Variant_Calling/vcf/N00208_15ind_allsites_finalFiltered.vcf.gz
--site-mean-depth --out N00208_15ind_allsites_finalFiltered
```

2. Visualize the coverage in R
First, open R in the terminal

```
R
```

First, load information on coverage
Then, visualize the data and consider adjusting the limits of the x-axis and the y-axis!

```
sca="{scaff}"
vcf <- /home/ubuntu/Variant_Calling/{input.ldepth.mean}", h=T)
png(paste("coverage_plot_",sca,".png", sep=""))
plot(vcf$POS, vcf$MEAN_DEPTH, cex=0.1, pch= 1, xlab=paste("Physical
position along scaffold", sca), ylab="Coverage", col="black")
dev.off()
```

```
sca="N00135"
vcf <- read.table(paste("/home/ubuntu/Variant_Calling/", sca,
"_15ind_allsites_finalFiltered.ldepth.mean", sep= ""), h=T)
png(paste("coverage_plot_",sca,".png", sep=""))
plot(vcf$POS, vcf$MEAN_DEPTH, cex=0.1, pch= 1, xlab=paste("Physical
position along scaffold", sca), ylab="Coverage", col="black")
dev.off()
```

```
sca="N00208"
vcf <- read.table(paste("/home/ubuntu/Variant_Calling/", sca,
"_15ind_allsites_finalFiltered.ldepth.mean", sep= ""), h=T)
png(paste("coverage_plot_",sca,".png", sep=""))
plot(vcf$POS, vcf$MEAN_DEPTH, cex=0.1, pch= 1, xlab=paste("Physical
position along scaffold", sca), ylab="Coverage", col="black")
dev.off()
```

Ideally coverage should only slightly vary across the genome. How do you interpret the visualization of coverage along each of the two scaffolds? Do the observations meet these expectations, or do you observe some unexpected patterns? To answer the questions, you might have a look at the lecture slides, more specifically the filtering steps for VCF files. Could the observations indicate that some more filtering is needed?

Detection of balancing selection in *Ficedula* flycatchers (part II)

Access and installation of the work environment

1. Change to your home directory
/home/ubuntu

```
cd
```

2. Create a new subdirectory and change directory
```
mkdir Balancing_Selection
cd Balancing_Selection
```

3. Access to relevant input files
You will work with the two filtered VCF files from part I
/home/ubuntu/Variant_Calling/vcf/N00135_15ind_allsites_finalFiltered.vcf.gz
/home/ubuntu/Variant_Calling/vcf/N00208_15ind_allsites_finalFiltered.vcf.gz

In addition, you will re-use the information on coverage
/home/ubuntu/Variant_Calling/N00135_15ind_allsites_finalFiltered.ldepth.mean
/home/ubuntu/Variant_Calling/N00208_15ind_allsites_finalFiltered.ldepth.mean

```
cp
/home/ubuntu/Variant_Calling/vcf/N00*_15ind_allsites_finalFiltered.vcf.gz .
cp
/home/ubuntu/Variant_Calling/N00*_15ind_allsites_finalFiltered.ldepth.mean .
```

Preparation of data files for downstream analysis

Modify the input and output names appropriately. Inspect all input files and clarify for yourself what the command is doing.

Remember the workflow conventions and choose informative filenames!
Remember the workflow conventions and do not overwrite any files!

1. Compute count files
```
vcftools --gzvcf {input.vcf} --counts --out {output}
```

```
vcftools --gzvcf N00135_15ind_allsites_finalFiltered.vcf.gz --counts --out
N00135_15ind_allsites_finalFiltered
```

```
vcftools --gzvcf N00208_15ind_allsites_finalFiltered.vcf.gz --counts --out
N00208_15ind_allsites_finalFiltered
```

2. Remove sites that are not covered by all individuals
```
ind=15
awk -v OFS='\t' -v var=$ind '{{ if(NR>1 && $4==(2*var) && $3>1) print $0
}}' {input.frq.count} > {output.frq.count}
```

```
ind=15
awk -v OFS='\t' -v var=$ind '{{ if(NR>1 && $4==(2*var) && $3>1) print $0
}}' N00135_15ind_allsites_finalFiltered.frq.count >
N00135_15ind_allsites_finalFiltered_allInd.frq.count
```

```
ind=15
awk -v OFS='\t' -v var=$ind '{{ if(NR>1 && $4==(2*var) && $3>1) print $0
}}' N00208_15ind_allsites_finalFiltered.frq.count >
N00208_15ind_allsites_finalFiltered_allInd.frq.count
```

3. Print position, the counts for each allele and total counts

```
awk -v OFS='\t' '{{ split($5, all1, ":"); split($6, all2, ":"); print $2,
all1[2], all2[2], $4 }}' {input.frq.count} > {output.frq.count}

awk -v OFS='\t' '{{ split($5, all1, ":"); split($6, all2, ":"); print $2,
all1[2], all2[2], $4 }}'
N00135_15ind_allsites_finalFiltered_allInd.frq.count >
N00135_15ind_allsites_finalFiltered_allInd_sel.frq.count

awk -v OFS='\t' '{{ split($5, all1, ":"); split($6, all2, ":"); print $2,
all1[2], all2[2], $4 }}'
N00208_15ind_allsites_finalFiltered_allInd.frq.count >
N00208_15ind_allsites_finalFiltered_allInd_sel.frq.count
```

3. Remove monomorphic sites

```
awk -v OFS='\t' '{{ if($3!=0 && $2!=0) print $0 }}' {input.frq.count} >
{output.frq.count}

awk -v OFS='\t' '{{ if($3!=0 && $2!=0) print $0 }}'
N00135_15ind_allsites_finalFiltered_allInd_sel.frq.count >
N00135_15ind_variants_sites.frq.count

awk -v OFS='\t' '{{ if($3!=0 && $2!=0) print $0 }}'
N00208_15ind_allsites_finalFiltered_allInd_sel.frq.count >
N00208_15ind_variants_sites.frq.count
```

4. Select the minor allele and add a column containing “NA” for lack of recombination rate

```
awk -v OFS='\t' '{{ if($3<$2) print $1, "NA", $3, $4; else print $1, "NA",
$2, $4 }}' {input.frq.count} > {output.frq.count}

awk -v OFS='\t' '{{ if($3<$2) print $1, "NA", $3, $4; else print $1, "NA",
$2, $4 }}' N00135_15ind_variants_sites.frq.count >
N00135_15ind_variants_sites_MAsorted.frq.count

awk -v OFS='\t' '{{ if($3<$2) print $1, "NA", $3, $4; else print $1, "NA",
$2, $4 }}' N00208_15ind_variants_sites.frq.count >
N00208_15ind_variants_sites_MAsorted.frq.count
```

Description of genetic diversity

1. Compute and visualize the site frequency spectrum (SFS)

First, open R in the terminal

```
R
```

We need to load information on the SFS

```
sca="{scaff}"
count <- read.table("/home/ubuntu/Balancing_Selection/{input. frq.count}",
h=F)
png(paste("SFS_plot_",sca,".png", sep=""))
```

```
hist(count$V3, xlab="minor allele count", main=paste("SFS_", sca, sep=""),
breaks=15)
dev.off()

sca="N00135"
count <- read.table("N00135_15ind_variants_sites_MAsorted.frq.count", h=F)
png(paste("SFS_plot_",sca,".png", sep=""))
hist(count$V3, xlab="minor allele count", main=paste("SFS_", sca, sep=""),
breaks=15)
dev.off()

sca="N00208"
count <- read.table("N00208_15ind_variants_sites_MAsorted.frq.count", h=F)
png(paste("SFS_plot_",sca,".png", sep=""))
hist(count$V3, xlab=" minor allele count", main=paste("SFS_", sca, sep=""),
breaks=15)
dev.off()
```

2. Compute SNP density in 1-kb non-overlapping windows

```
mv ../Variant_Calling/callable_sites.awk .

vcftools --gzvcf {input.vcf.gz} --window-pi 1000 --out {output}
vcftools --gzvcf {input.vcf.gz} --site-pi --out {output}
awk -f callable_sites.awk {input.sites.pi} > {input.callable1000}

vcftools --gzvcf N00135_15ind_allsites_finalFiltered.vcf.gz --window-pi
1000 --out N00135_15ind_allsites_finalFiltered

vcftools --gzvcf N00135_15ind_allsites_finalFiltered.vcf.gz --site-pi --out
N00135_15ind_allsites_finalFiltered

awk -f callable_sites.awk N00135_15ind_allsites_finalFiltered.sites.pi >
N00135_15ind_allsites_finalFiltered.callable1000

vcftools --gzvcf N00208_15ind_allsites_finalFiltered.vcf.gz --window-pi
1000 --out N00208_15ind_allsites_finalFiltered

vcftools --gzvcf N00208_15ind_allsites_finalFiltered.vcf.gz --site-pi --out
N00208_15ind_allsites_finalFiltered

awk -f callable_sites.awk N00208_15ind_allsites_finalFiltered.sites.pi >
N00208_15ind_allsites_finalFiltered.callable1000
```

Running BalLeRMix

1. Run BalLeRMix

```
mv ../Variant_Calling/BalLeRMix_v2.3.py .

python3 BalLeRMix_v2.3.py -i {input.frq.count} --spect {output.SFS} --MAF -
-getSpect

python3 BalLeRMix_v2.3.py -i {input.frq.count} --spect {input.SFS} -o
{output.BM.out} --MAF --nosub --physPos

python3 BalLeRMix_v2.3.py -i N00135_15ind_variants_sites_MAsorted.frq.count
--spect N00135_15ind_variants_sites_MAsorted.SFS --MAF --getSpect
```

```
python3 BalLeRMix_v2.3.py -i N00135_15ind_variants_sites_MAsorted.frq.count
--spect N00135_15ind_variants_sites_MAsorted.SFS -o
N00135_15ind_variants_sites_MAsorted.BM.out --MAF --nosub -physPos (~20
minutes)

python3 BalLeRMix_v2.3.py -i N00208_15ind_variants_sites_MAsorted.frq.count
--spect N00208_15ind_variants_sites_MAsorted.SFS --MAF --getSpect

python3 BalLeRMix_v2.3.py -i N00208_15ind_variants_sites_MAsorted.frq.count
--spect N00208_15ind_variants_sites_MAsorted.SFS -o
N00208_15ind_variants_sites_MAsorted.BM.out --MAF --nosub -physPos (~20
minutes)
```

2. Visualize the coverage, SNP density and BalLeRMix scores in R
First, open R in the terminal

```
R
```

We need to load information on coverage, SNP density and BalLeRMix scores

```
sca="{scaff}"

vcf <- read.table(paste("/home/ubuntu/Balancing_Selection/", sca,
"_15ind_allsites_finalFiltered.ldepth.mean", sep= ""), h=T)
pi <- read.table(paste("/home/ubuntu/Balancing_Selection/", sca,
"_15ind_allsites_finalFiltered.windowed.pi", sep=""), h=T)
BalSel <- read.table(paste("/home/ubuntu/Balancing_Selection/", sca,
"_15ind_variants_sites_MAsorted.BM.out", sep= ""), h=T)
callable_sites <- read.table(paste("/home/ubuntu/Balancing_Selection/",
sca, "_15ind_allsites_finalFiltered.callable1000", sep= ""), h=F)

sca= "N00135"
sca= "N00208"

vcf <- read.table(paste(sca, "_15ind_allsites_finalFiltered.ldepth.mean",
sep= ""), h=T)
pi <- read.table(paste(sca, "_15ind_allsites_finalFiltered.windowed.pi",
sep=""), h=T)
BalSel <- read.table(paste(sca, "_15ind_variants_sites_MAsorted.BM.out",
sep= ""), h=T)
callable_sites <- read.table(paste(sca,
"_15ind_allsites_finalFiltered.callable1000", sep= ""), h=F)
```

Merge the two R data.frames containing information on 1) variant sites and 2) callable sites

```
ii<-as.numeric(rownames(callable_sites))
window <- (ii-1)*1000+1
data.CS <-data.frame(window=window, callable_sites=callable_sites$V1)

my.pi <- merge(data.CS, pi, by.x = 1, by.y = 2)
my.pi$corDens <- my.pi$N_VARIANTS/my.pi$callable_sites
my.pi$corPi <- my.pi$PI/my.pi$callable_sites*1000
```

Now visualize the data and consider adjusting the limits of the x-axis and the y-axis!
Visualize coverage and BalLeRMiX scores

```
png(paste("BalLeRMix_Coverage_",sca,".png", sep=""))
par(mar = c(5, 4, 4, 4) + 0.3, mfrow=c(1,1))
plot(vcf$POS, vcf$MEAN_DEPTH, cex=0.1, pch= 1, xlab="", ylab="",
col="black", ylim=c(0,200), axes=FALSE, bty="n")
axis(side=4, at = pretty(c(0,200)))
```

```
mtext("Coverage", side=4, line=3)
par(new=TRUE)
plot(BalSel$physPos, BalSel$LR, cex=0.4, xlab= paste("Physical position
along scaffold", sca), ylab="Ballermix LR score (B0)", col="turquoise",
pch=16, ylim=c(0,500))
legend("topright", legend=c("BalLeRMix score", "Coverage"),
col=c("turquoise","black"), pch=c(16,16))
dev.off()
```

Visualize SNP density and BalLeRMix scores

```
png(paste("BalLeRMix_SNPdensity_",sca,".png", sep=""))
par(mar = c(5, 4, 4, 4) + 0.3, mfrow=c(1,1))
plot((my.pi$window+my.pi$BIN_END)/2, my.pi$corDens, cex=0.1, pch= 1,
xlab="", ylab="", col="black", ylim=c(0,1), axes=FALSE, bty="n")
axis(side=4, at = pretty(c(0,1)))
mtext("SNP density", side=4, line=3)
par(new=TRUE)
plot(BalSel$physPos, BalSel$LR, cex=0.4, xlab= paste("Physical position
along scaffold", sca), ylab="Ballermix LR score (B0)", col="turquoise",
pch=16, ylim=c(0,500))
legend("topright", legend=c("BalLeRMix score", "SNP density"),
col=c("turquoise","black"), pch=c(16,16))
dev.off()
```

How do you interpret the visualization of BalLeRMix scores, coverage and SNP density along each of the two scaffolds? Do the three parameters covary with each other? If yes, what could this covariation indicate? Do you observe any interesting differences between the two scaffolds? Do the observations meet your expectations, or do you observe some unexpected aspects?

References

Charlesworth D. 2006. Balancing selection and its effects on sequences in nearby genome regions. *PLoS Genet.* 2:e64

Chase MA, Ellegren H, Mugal CF. 2021. Positive selection plays a major role in shaping signatures of differentiation across the genomic landscape of two independent Ficedula flycatcher species pairs. *Evolution.* 75:2179-2196

Cheng X, DeGiorgio M. 2019. Detection of Shared Balancing Selection in the Absence of Trans-Species Polymorphism. *Mol Biol Evol.* 36:177-199

DeGiorgio M, Lohmueller KE, Nielsen R. 2014. A model-based approach for identifying signatures of ancient balancing selection in genetic data. *PLoS Genet.* 10:e1004561

Ebert D, Fields PD. 2020. Host-parasite co-evolution and its genomic signature. *Nat Rev Genet.* 21:754-768

Hung, C.-M., Zink, R.M., 2014. Distinguishing the effects of selection from demographic history in the genetic variation of two sister passerines based on mitochondrial–nuclear comparison. *Heredity* 113:42–51

Qvarnström A, Rice AM, and Ellegren H. 2010. Speciation in Ficedula flycatchers. *Philos Trans R Soc Lond B Biol Sci* 365:1841-1852

Zink, R.M., Pavlova, A., Drovetski, S., Rohwer, S., 2008. Mitochondrial phylogeographies of five widespread Eurasian bird species. *J. Ornithology* 149:399–413