wilding_urbano_outline_results

Josquin Daron

22 septembre 2021

1. Samples

- **AG1000g phase 2: 1142 genomes and 16 populations**

GNcol (4), GQgam (9), GHgam (12), FRgam (24), GNgam (40), KE (48), GHcol (55), GM (65), GAgam (69), Icol (71), BFcol (75), AOcol (78), GW (91), BFgam (92), UGgam (112), CMgam (297)

- **Wilding genomes: 96 genomes and 3 populations**

32 LVBdom (Libreville, Gabon domestic)
32 LPdom (La lope, Gabon domestic)
32 LPfor (La lope, Gabon forest)

- **Urbano genomes: 88 genomes and 3 populations**

10 BZV (Brazzaville, Congo)
36 DLA (Douala, Cameroon)
42 LBV (Libreville, Gabon)

---

2. Dataset creation: reads mapping, SNP calling and filtering

2.1 Reads mapping

2.1.1 Bash script to perform: FASTQC, cutadapt, bwa mem, gatk realigner, bam report

- FASTQC report:

-> Wilding fastqc report: *not available because we've got mapped reads.* -> Urbano fastqc report: fastqc report git hub link.

- cutadapt:

```
cutadapt -a AGATCGGAAGAGCACACGTCTGAA -A AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT  -q 3
```

- bwa mem:

```
header=$(zcat $sampleId.R1.fastq.gz | head -n 1)
id=$(echo $header | head -n 1 | cut -f 1-4 -d":" | sed 's/@//' | sed 's/:/_/g')
sm=$(echo $header | head -n 1 | grep -Eo "[ATGCN]+$")
echo "Read Group @RG\tID:$id\tSM:$id"_"$sm\tLB:$id"_"$sm\tPL:ILLUMINA"

bwa mem -t 1 Anopheles_gambiae.AgamP4.dna.chr.fna $sampleId.R1.fastq.gz $sampleId.R2.fastq.gz -R $(echo
m\tPL:ILLUMINA") | samtools view -F 4 -b - | samtools sort - -o $sampleId.map.sort.bam
```

- gatk realigner:

```
java -jar ~/bioInf/bin/GenomeAnalysisTK-3.8-0-ge9d806836/GenomeAnalysisTK.jar -T RealignerTargetCreator
 -I $sampleId.map.sort.bam -o $sampleId.realignertargetcreator.intervals

java -Xmx8G -Djava.io.tmpdir=/tmp -jar ~/bioInf/bin/GenomeAnalysisTK-3.8-0-ge9d806836/GenomeAnalysisTK.j
ae.AgamP4.dna.chr.fna -targetIntervals $sampleId.realignertargetcreator.intervals -I $sampleId.map.sort
```

- bam file report (qualimap):

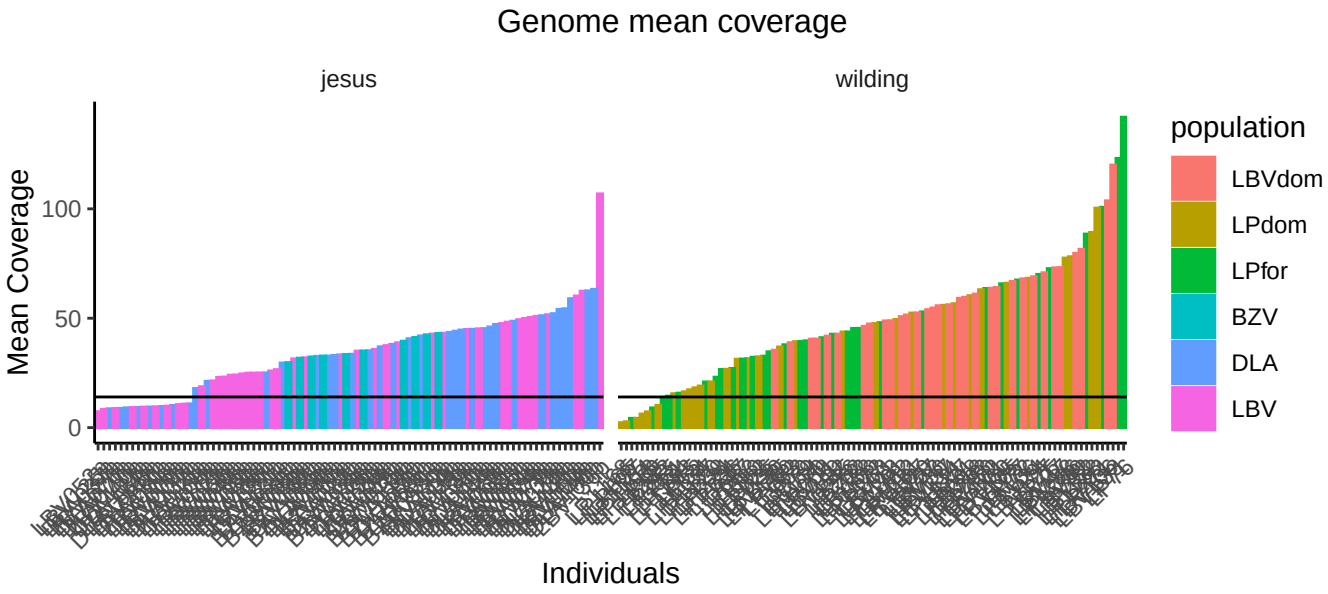
```
qualimap bamqc -bam /scratch/daron_anopheles/bam/$inputFile.indelrealigner.bam -c --java-mem-size=8G -ou
es/fastqBamInfo/bamInfo/jesus_qualimap/$inputFile.outqualimap -nt 2 -outformat HTML
```

-> wilding bam report: qualimap report git hub link.

-> urbano bam report: qualimap report git hub link.

2.1.2 Bam files analysis (genome depth, sex determination)

- Genome mean coverage:

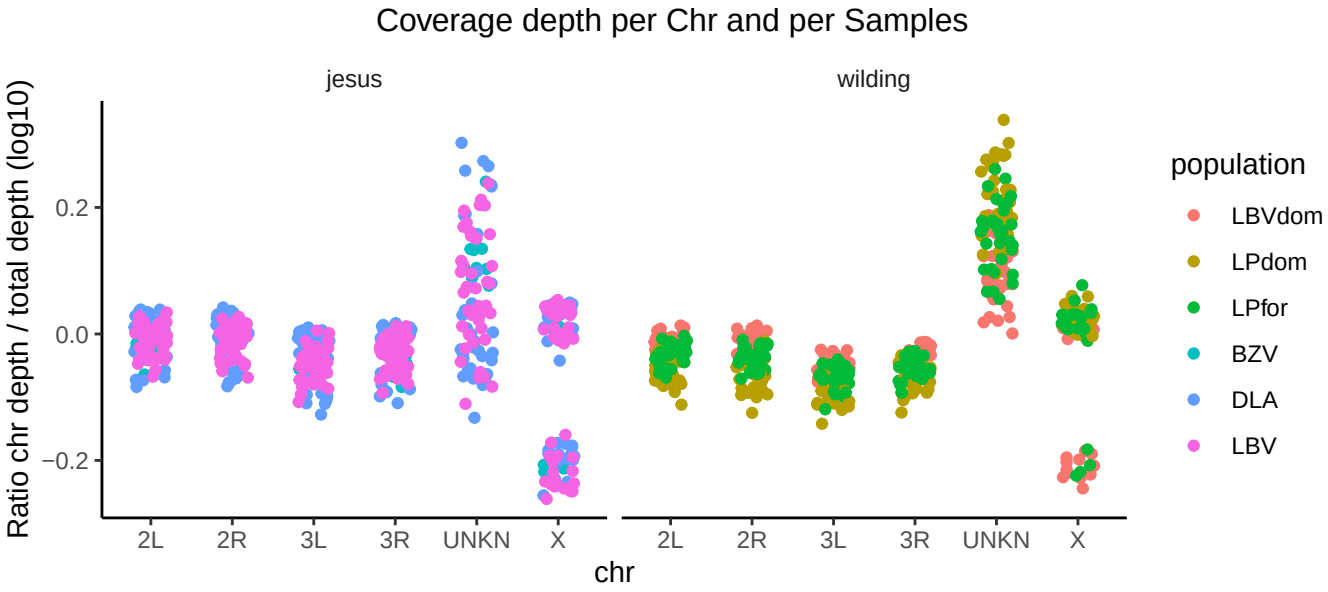


–> Filtering individuals: Remove individuals with mean coverage lower than 14x

9 individuals from Wilding: LP69, LP243, LP697, LP1118, LP1125, LP1164, LP1165, LP1168, LP1285

17 individuals from Urbano: DLA037p, DLA076p, DLA077p, DLA102p, DLA105p, DLA130p, DLA132p, DLA155Bp, LBV001p, LBV007p, LBV009p, LBV052p, LBV125p, LBV127p, LBV137p, LBV140p, LBV142p

- Determining sex of each samples:



Result: **wilding**: 16/96 males ; **urbano**: 36/88 males

2.2 SNPs calling and filtering

2.2.1 SNPs calling script

- gatk unifiedGenotyper:

```
java -jar ~/bin/GenomeAnalysisTK-3.8-0-ge9d806836/GenomeAnalysisTK.jar -T UnifiedGenotyper -R Anopheles_
ist -L $interval --genotyping_mode DISCOVERY --downsampling_type BY_SAMPLE -dcov 250 --output_mode EMIT_
17 --genotype_likelihoods_model BOTH --heterozygosity 0.01 --indel_heterozygosity 0.001 -stand_call_con
rose.$out.unifiedGenotyper.vcf
```

2.2.2 SNP filtering:

- Jupyter-notebook script to generate html report on the newly created VCF file: Jupyter notebook VCF stat report code.

```
launch_ipynb.py -i vcfStats_slurm.ipynb -o wilding.chr3R.vcfStats_slurm.html
```

-> Wilding and Urbano samples SNPs stat report: Wilding and Urbano SNP stat report.

Table 1: Final number of SNPs per chr after filtration

chr	NbSNP
2L	2723196
2R	3485073
3L	2540839
3R	3524803
X	639905

- 4 inds removed because imiss > 10%: LP1124 LP1145 LP47 LP63
  /! for X chr 10 samples are removed: BZV093bu DLA136u DLA137u LBV066u LBV072u LBV131u LP1124w LP1145w LP47w LP63w
- Bash script to perform SNP and ind filtering (cause scikit is only outputing stats)

```
#!/bin/bash

# input file list
IN_VCF=$1         # input VCF file
AG_VCF_ACCESS=$2  # AG1000G VCF for genome accessibility, downloaded at ftp://ngs.sanger.ac.uk/producti
ssibility.X.vcf.gz
REF=$3
IND=$4

IN_PREFIX=`echo $IN_VCF | sed 's,\(.*\).vcf.gz,\1,'`

# Step 1: Select variants using GATK
echo "--> Step 1: Select variant based on GATK metrics `date`"
echo "Filter Expression QD < 5.00 || FS > 60.000 || ReadPosRankSum < -8.000"

java -jar ~/bioInf/bin/GenomeAnalysisTK-3.8-0-ge9d806836/GenomeAnalysisTK.jar -T SelectVariants -R $REF
 -selectType SNP

tabix $IN_PREFIX.snponly.vcf.gz
```

```
java -jar ~/bioInf/bin/GenomeAnalysisTK-3.8-0-ge9d806836/GenomeAnalysisTK.jar -T VariantFiltration -R $
PREFIX.annot.vcf.gz --filterExpression "QD < 5.00 || FS > 60.000 || ReadPosRankSum < -8.000 " --filterNa

zcat $IN_PREFIX.annot.vcf.gz | egrep -v "LOW_QUAL" | bgzip > $IN_PREFIX.passQC.vcf.gz

# Step 2: Select variants from inputed VCF based on genome accessibility
echo "--> Step 2: Choose variant based on genome accessibility `date`"
echo "minGQ 20 min DP 10"

zcat $AG_VCF_ACCESS | awk '{if($7=="PASS"){print $1"\t"$2}}' > $IN_PREFIX.pos

vcftools --gzvcf $IN_PREFIX.passQC.vcf.gz --positions $IN_PREFIX.pos --minGQ 20 --non-ref-ac-any 1 --re
> $IN_PREFIX.snpPassQC.vcf.gz

rm $IN_PREFIX.annot.vcf.gz $IN_PREFIX.snponly.vcf.gz $IN_PREFIX.passQC.vcf.gz

vcftools --gzvcf $IN_PREFIX.snpPassQC.vcf.gz --missing-site --stdout | awk '{if($6<0.05){print $0}}' | c

vcftools --gzvcf $IN_PREFIX.snpPassQC.vcf.gz --positions $IN_PREFIX.lmiss --remove $IND --non-ref-ac-any
| bgzip > $IN_PREFIX.passQC.vcf.gz

rm $IN_PREFIX.snpPassQC.vcf.gz
```

- Summary of filtering step:

1. Remove individual with mean coverage lower than 14x
2. Discard SNPs present in none accessible area (defined in ag1000g), QD $<$ 5.00, FS $>$ 60.000 and ReadPosRankSum $<$ -8.000
3. Replace by NA genotypes with low call confidence (GQ$<$20)
4. Remove SNPs with $>$5% lmiss
5. Remove Inds with $>$10% imiss

---

3. Structure of genetic variation

Global genetic structure

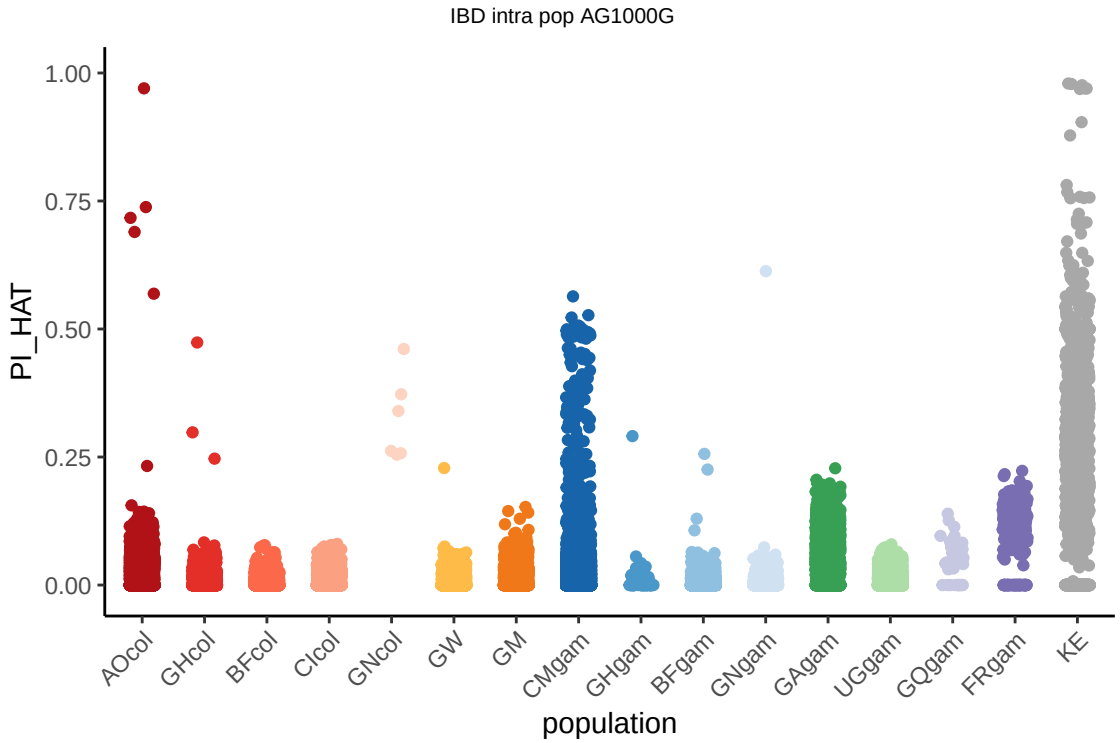
IBD anlaysis to identify closely related samples

Context: Downstream analysis required the absence of high related samples (i.e. more than we would expect by chance in a random sample). Goal: Identify putative closely related pair of sample using plink –genome

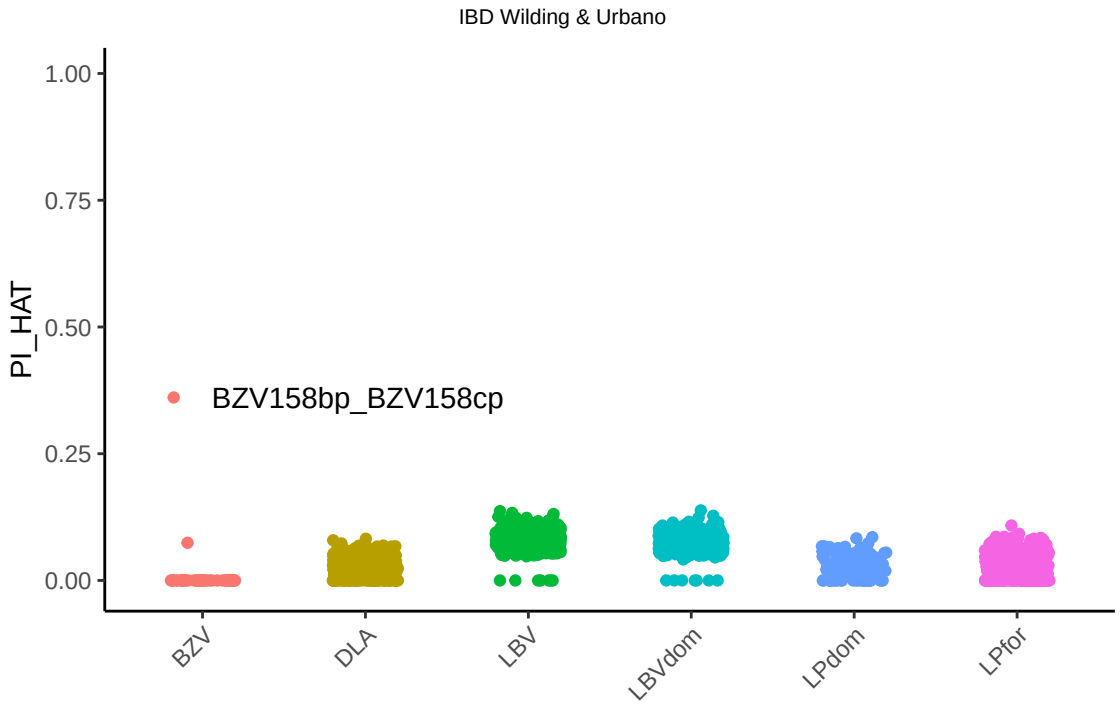
Bellow is the script for the IBD analysis:

```
# 1. Prune VCF using customized script. The program output the list of the coordinate of unlinked SNPs
prune_SNPs.py --snp wilding_urbano.3L.unifiedGenotyper.cov14x.passQC.vcf.gz --pop all --meta wilding_url

prune_SNPs.py --snp wilding_urbano.3R.unifiedGenotyper.cov14x.passQC.vcf.gz --pop all --meta wilding_url

# 2. Fetch pruned SNPs using VCFtools
vcftools --gzvcf wilding_urbano.3L.unifiedGenotyper.cov14x.passQC.vcf.gz --positions wilding_urbano.3L.p
 --recode --stdout | bgzip > wilding_urbano.3L.pruned.vcf.gz

vcftools --gzvcf wilding_urbano.3R.unifiedGenotyper.cov14x.passQC.vcf.gz --positions wilding_urbano.3R.p
 --recode --stdout | bgzip > wilding_urbano.3R.pruned.vcf.gz

vcf-concat wilding_urbano.3L.unifiedGenotyper.cov14x.passQC.vcf.gz wilding_urbano.3R.unifiedGenotyper.co

# 3. IBD with plink --genome
plink --vcf wilding_urbano.3.pruned.vcf.gz --allow-extra-chr --genome --out wilding_urbano.chr3.ibd
```

1. AG1000g



2. Wilding & Urbano



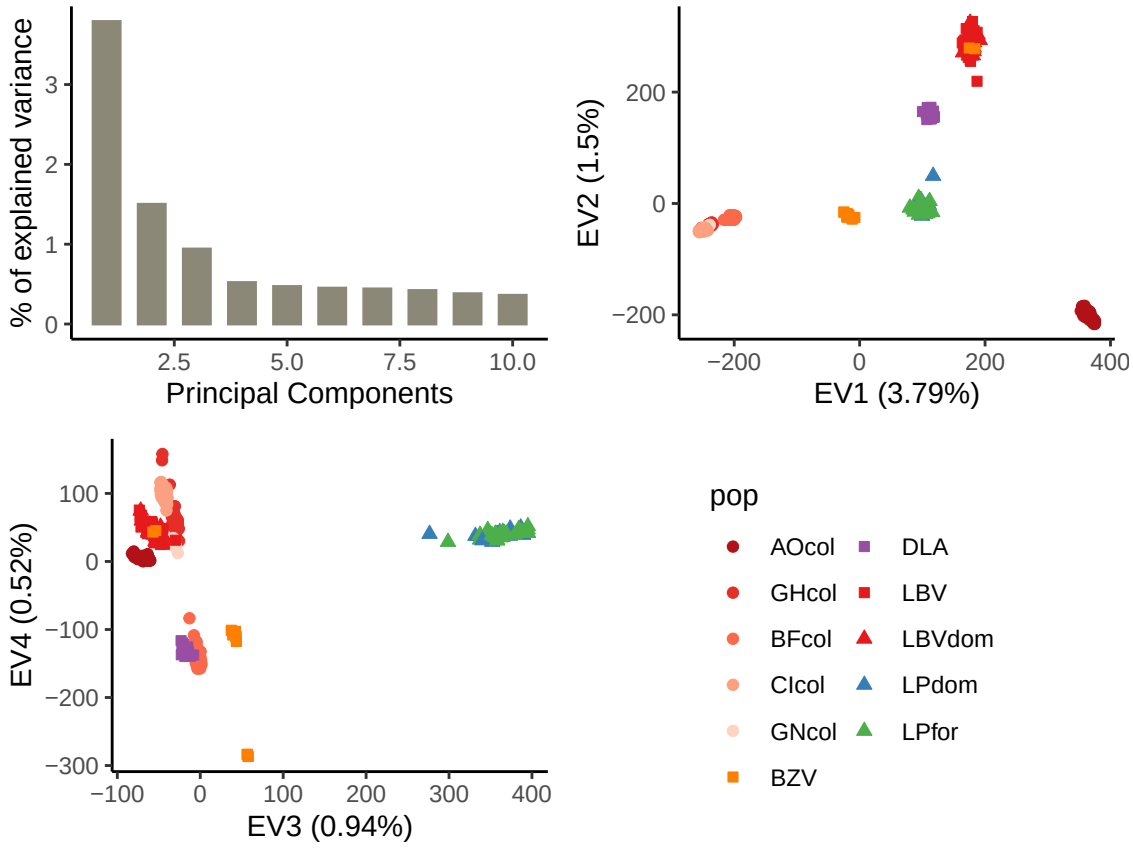
- Conclusion: Most sample pairs have low IBD value, meaning they are not related.

PCA

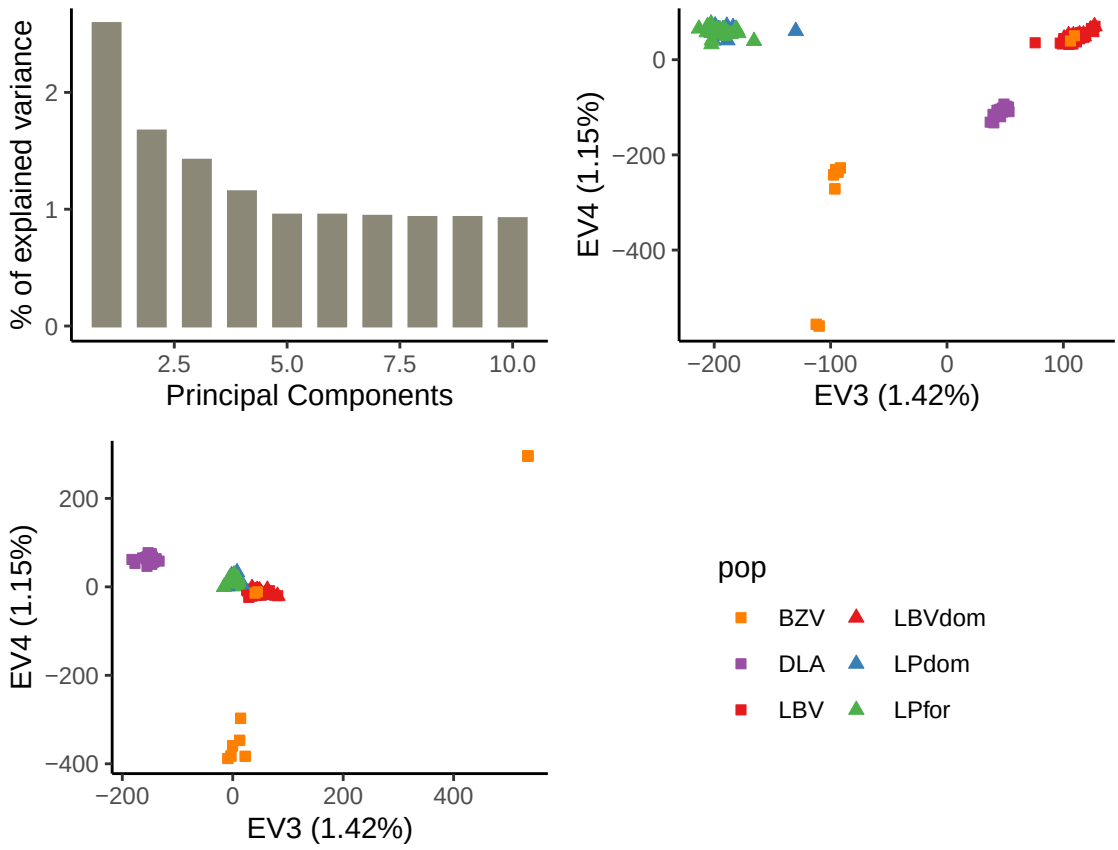
```
# PCA analysis was made using the jupyter-notebook script pca.ipynb
```

Question: How does the genetics variation of wilding/urbano samples is structured alone, or within the AG1000G samples?

1. Wilding and Urbano samples within ag1000g colluzzi



2. Wilding and Urbano



- Conclusion:

1. Wilding genome are located between the AOcol and GHcol population, as expected.
2. Based on the wilding pca, we can distinguish clearly two pop: (i) LBVdom and (ii) LPdom-LPfor. LPdom and LPfor even they have been samples at 15km away from each other they represent one single population

Admixture

Question: Same question than above adding the admixture component.

- lauch admixture on the cluster, 1 job per k, 10 time a random set of 100kSNPs and 10 seed per samples set (N=100 per k):

```
path_to_tmp="/scratch/daron_$SLURM_JOB_ID"
IN_PREFIX="ag1000g.phase2.anopheles-rose.merged.biallelic.3.pruned"
K=${1}

# download vcf file
scp nas3:/data3/projects/plasmodium/anopheles/vcf_store/$IN_PREFIX.vcf.gz $path_to_tmp

# run admixture 10 resamples of 100kSNPs with 10 different seed; N=100
N=0
for i in {1..10}; do
    zcat $IN_PREFIX.vcf.gz | egrep -v "#" | cut -f 1,2 | shuf -n 100000 | sort -k 1n,1n -k 2n,2n > pos
    vcftools --gzvcf $IN_PREFIX.vcf.gz --positions pos --stdout --recode | sed 's,^3L,1,' | sed 's,^3R,2
```

```
    tabix samp.vcf.gz
    plink --vcf samp.vcf.gz --make-bed  --allow-extra-chr --out samp

    for j in {1..10}; do
        N=$(( $N + 1 ))
        admixture --cv --seed=$RANDOM -j9 samp.bed $K | tee $IN_PREFIX.n$N.log${K}.out
        mv samp.$K.P $IN_PREFIX.n$N.$K.P
        mv samp.$K.Q $IN_PREFIX.n$N.$K.Q
        scp $IN_PREFIX.n$N.$K.P $path_to_dir/
        scp $IN_PREFIX.n$N.$K.Q $path_to_dir/
        scp $IN_PREFIX.n$N.log${K}.out $path_to_dir/
    done
done
```

- reorder ind based on a customized order:

```
R --vanilla --args $PWD/'admixture output file' $PWD/'ind ordered as in the admix' $PWD/'ind new order'
```

- run clumpack

```
CLUMPAK.pl -id 10 -dir ./ -file out_ordered_ag1000gCol_wilding.zip -inputtype Admixture -indtopop ag100

option help:
-id Job id (defaut random number generated)
-inputtype Input file format [Admixture|Structure]
-indtopop population label (only pop name in the same order than input file)
```