Greenish Warbler Genomic Analysis

Darren Irwin

2023-09-10

This page contains notes and code describing the data analysis for a manuscript on Greenish Warbler genomics. I've been working with the data for several years, and the R and then Julia code has been in development for a while. This is a Quarto notebook, which can run and display the results of Julia (or other) code blocks, along with text narration, and output in html, pdf, Word, etc.

The Julia code here is loosely based on R code written for Greenish Warbler analysis (Irwin et al. 2016), and then the North American warbler analyses (Irwin et al. 2019), and then my (unpublished) 2019 Greenish Warbler analysis. Most recently, this was adapted from the scripts called GW2022_R_analysis_script.R and IrwinLabGenomicsAnalysisScript.jl but has had a lot of opimizations since then. The SNP data here are a result of GBS reads mapped to our new 2022 Biozeron genome assembly for a greenish warbler from southern China.

Load packages

If running this for the first time, you will need to load packages used in the script, so run what is in this section below. It will take some time to install and precompile the packages:

```
import Pkg; Pkg.add("CSV") # took less than a minute
Pkg.add("DataFrames") # took about a minute
Pkg.add("Plots") # seems to install and working more simply than Makie (but less powerful)
Pkg.add("Haversine") # for great circle (Haversine) distances
Pkg.add("Distributions") # this seemed to fix a problem installing GLMakie
Pkg.add("MultivariateStats")
Pkg.add("StatsBase")
Pkg.add("Impute")
Pkg.add("JLD2")
Pkg.add("CairoMakie")
Pkg.add("PrettyTables") # for printing nice tables to REPL
```

Now actually load those packages into the Julia session:

```
using CSV # for reading in delimited files
using DataFrames # for storing data as type DataFrame
using Haversine # for calculating Great Circle (haversine) distances between sites
using MultivariateStats # for Principal Coordinates Analysis (multidimensional scaling)
using DelimitedFiles # for reading delimited files (the genotypic data)
using Impute # for imputing missing genotypes
using JLD2 # for saving data
using CairoMakie # for plots
using PrettyTables
CairoMakie.activate!()  # this makes CairoMakie the main package for figures (in case another already
```

Load my custom package `SNPlots`:

```
include("SNPlots.jl") # load file containing custom-built functions
using .SNPlots # actually make SNPlots module available with SNPlots.functionName(),
# or if functions are exported from SNPlots then they are available.
```

Test Julia:

```
x = 1; y = 2; z = x+y
println("z = ", z)
```

```
z = 3
```

(If Quarto is calling Julia properly, you will see `z = 3` as the output of the code block above.)

Choose working directory:

```
repoDirectory = pwd() # this gets the starting working directory, for later use
cd("/Users/darrenirwin/Dropbox/Darren's current work/")
```

OK, let's load the genomic data!

```
# choose path and filename for the 012NA files
baseName = "GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs.
filenameTextMiddle = ".max2allele_noindel.vcf.maxmiss"
# indicate percent threshold for missing genotypes for each SNP--
# this was set by earlier filtering, and is just a record-keeper for the filenames:
missingGenotypeThreshold = 60
```

```
filenameTextEnd = ".MQ20.lowHet.tab"
tagName = ".Sept2023."   # choose a tag name for this analysis
# indicate name of metadata file, a text file with these column headings:
# ID    location    group   Fst_group   plot_order
metadataFile = "GW_genomics_2022_with_new_genome/GW_all4plates.Fst_groups.txt"
# load metadata
metadata = DataFrame(CSV.File(metadataFile)) # the CSV.File function interprets the correct delimite
num_metadata_cols = ncol(metadata)
num_individuals = nrow(metadata)
# read in individual names for this dataset
individuals_file_name = string(baseName, filenameTextMiddle, missingGenotypeThreshold, filenameTextEn
ind = DataFrame(CSV.File(individuals_file_name; header=["ind"], types=[String]))
indNum = size(ind, 1) # number of individuals
if num_individuals != indNum
    println("WARNING: number of rows in metadata file different than number of individuals in .indv
end
# read in position data for this dataset
position_file_name = string(baseName, filenameTextMiddle, missingGenotypeThreshold, filenameTextEnd,
pos_whole_genome = DataFrame(CSV.File(position_file_name; header=["chrom", "position"], types=[Strin
# read in genotype data
column_names = ["null"; string.("c.", pos_whole_genome.chrom, ".", pos_whole_genome.position)]
genotype_file_name = string(baseName, filenameTextMiddle, missingGenotypeThreshold, filenameTextEnd,
@time if 1 <= indNum <= 127
    geno = readdlm(genotype_file_name, '\t', Int8, '\n'); # this has been sped up dramatically, by fi
elseif 128 <= indNum <= 32767
    geno = readdlm(genotype_file_name, '\t', Int16, '\n'); # this needed for first column, which is n
else
    print("Error: Number of individuals in .indv appears outside of range from 1 to 32767")
end
loci_count = size(geno, 2) - 1   # because the first column is not a SNP (just a count from zero)
print(string("Read in genotypic data at ", loci_count," loci for ", indNum, " individuals. \n"))
```

```
 66.787300 seconds (3.94 M allocations: 15.994 GiB, 0.34% gc time, 1.15% compilation time)
Read in genotypic data at 2431709 loci for 310 individuals.
```

Check that individuals are same in genotype data and metadata

```
ind_with_metadata = hcat(ind, metadata)
println(ind_with_metadata)
println()  # prints a line break
if isequal(ind_with_metadata.ind, ind_with_metadata.ID)
    println("GOOD NEWS: names of individuals in metadata file and genotype ind file match perfectly."
else
    println("WARNING: names of individuals in metadata file and genotype ind file do not completely m
end
```

```
310×6 DataFrame
 Row │ ind                    ID                     location  group       Fst_group     plot_order
     │ String                 String31               String7   String15    String15      Float64
─────┼──────────────────────────────────────────────────────────────────────────────────────────────
   1 │ GW_Armando_plate1_AB1  GW_Armando_plate1_AB1  AB        vir         vir             20.01
   2 │ GW_Armando_plate1_JF07G02  GW_Armando_plate1_JF07G02  ST  plumb  plumb  87.0
   3 │ GW_Armando_plate1_JF07G03  GW_Armando_plate1_JF07G03  ST  plumb  plumb  87.0
   4 │ GW_Armando_plate1_JF07G04  GW_Armando_plate1_JF07G04  ST  plumb  plumb  87.0
   5 │ GW_Armando_plate1_JF08G02  GW_Armando_plate1_JF08G02  ST  plumb  plumb  87.0
   6 │ GW_Armando_plate1_JF09G01  GW_Armando_plate1_JF09G01  ST  plumb  plumb  87.0
   7 │ GW_Armando_plate1_JF09G02  GW_Armando_plate1_JF09G02  ST  plumb  plumb  87.0
   8 │ GW_Armando_plate1_JF10G03  GW_Armando_plate1_JF10G03  ST  plumb_vir  plumb_vir  9
   9 │ GW_Armando_plate1_JF11G01  GW_Armando_plate1_JF11G01  ST  plumb  plumb  87.0
  10 │ GW_Armando_plate1_JF12G01  GW_Armando_plate1_JF12G01  ST  plumb  plumb  87.0
  11 │ GW_Armando_plate1_JF12G02  GW_Armando_plate1_JF12G02  ST  plumb  plumb  87.0
  12 │ GW_Armando_plate1_JF12G04  GW_Armando_plate1_JF12G04  ST_vi  vir  vir  24.0
  13 │ GW_Armando_plate1_JF13G01  GW_Armando_plate1_JF13G01  ST  plumb  plumb  87.0
  14 │ GW_Armando_plate1_JF15G03  GW_Armando_plate1_JF15G03  KK  plumb  plumb  87.0
  15 │ GW_Armando_plate1_JF16G01  GW_Armando_plate1_JF16G01  KK_vi  plumb_vir  vir  24
  16 │ GW_Armando_plate1_JF20G01  GW_Armando_plate1_JF20G01  KK  plumb  plumb  87.0
  17 │ GW_Armando_plate1_JF22G01  GW_Armando_plate1_JF22G01  KK  plumb  plumb  87.0
  18 │ GW_Armando_plate1_JF23G01  GW_Armando_plate1_JF23G01  KK  plumb  plumb  87.0
  19 │ GW_Armando_plate1_JF23G02  GW_Armando_plate1_JF23G02  KK  plumb  plumb  87.0
  20 │ GW_Armando_plate1_JF24G02  GW_Armando_plate1_JF24G02  KK  plumb  plumb  87.0
  21 │ GW_Armando_plate1_JF26G01  GW_Armando_plate1_JF26G01  ST  plumb  plumb  87.0
  22 │ GW_Armando_plate1_JF27G01  GW_Armando_plate1_JF27G01  ST  plumb  plumb  87.0
  23 │ GW_Armando_plate1_JF29G01  GW_Armando_plate1_JF29G01  ST  plumb  plumb  87.0
```

```
GW_Armando_plate1_JF29G02     GW_Armando_plate1_JF29G02     ST      plumb        plumb          87.0
GW_Armando_plate1_JF29G03     GW_Armando_plate1_JF29G03     ST      plumb        plumb          87.0
GW_Armando_plate1_JG02G02     GW_Armando_plate1_JG02G02     KK      plumb        plumb          87.0
GW_Armando_plate1_JG02G04     GW_Armando_plate1_JG02G04     KK      plumb        plumb          87.0
GW_Armando_plate1_JG08G01     GW_Armando_plate1_JG08G01     ST      plumb        plumb          87.0
GW_Armando_plate1_JG08G02     GW_Armando_plate1_JG08G02     ST      plumb        plumb          87.0
GW_Armando_plate1_JG10G01     GW_Armando_plate1_JG10G01     ST      plumb        plumb          87.0
GW_Armando_plate1_JG12G01     GW_Armando_plate1_JG12G01     ST      plumb        plumb          87.0
GW_Armando_plate1_JG17G01     GW_Armando_plate1_JG17G01     ST      plumb_vir     plumb          77
GW_Armando_plate1_NO_BC_TTGW05 GW_Armando_plate1_NO_BC_TTGW05 blank    blank        blank
GW_Armando_plate1_NO_DNA      GW_Armando_plate1_NO_DNA      blank    blank        blank          -99.
GW_Armando_plate1_RF20G01     GW_Armando_plate1_RF20G01     BJ      obs_plumb     plumb_BJ        7
GW_Armando_plate1_RF29G02     GW_Armando_plate1_RF29G02     BJ      obs_plumb     plumb_BJ        7
GW_Armando_plate1_TL3        GW_Armando_plate1_TL3        TL      vir          vir            11.01
GW_Armando_plate1_TTGW01      GW_Armando_plate1_TTGW01      MN      troch_MN      troch_west       5
GW_Armando_plate1_TTGW05_rep1  GW_Armando_plate1_TTGW05_rep1  MN_rep   troch_MN_rep   troch_west_r
GW_Armando_plate1_TTGW05_rep2  GW_Armando_plate1_TTGW05_rep2  MN      troch_MN      troch_west
GW_Armando_plate1_TTGW06      GW_Armando_plate1_TTGW06      SU      lud_Sukhto    lud_central      4
GW_Armando_plate1_TTGW07      GW_Armando_plate1_TTGW07      SU      lud_Sukhto    lud_central      4
GW_Armando_plate1_TTGW10      GW_Armando_plate1_TTGW10      SU      lud_Sukhto    lud_central      4
GW_Armando_plate1_TTGW11      GW_Armando_plate1_TTGW11      SU      lud_Sukhto    lud_central      4
GW_Armando_plate1_TTGW13      GW_Armando_plate1_TTGW13      TH      lud_Thallighar lud_central
GW_Armando_plate1_TTGW17      GW_Armando_plate1_TTGW17      TH      lud_Thallighar lud_central
GW_Armando_plate1_TTGW19      GW_Armando_plate1_TTGW19      TH      lud_Thallighar lud_central
GW_Armando_plate1_TTGW21      GW_Armando_plate1_TTGW21      SR      lud_Sural     lud_central      4
GW_Armando_plate1_TTGW22      GW_Armando_plate1_TTGW22      SR      lud_Sural     lud_central      4
GW_Armando_plate1_TTGW23      GW_Armando_plate1_TTGW23      SR      lud_Sural     lud_central      4
GW_Armando_plate1_TTGW29      GW_Armando_plate1_TTGW29      SR      lud_Sural     lud_central      4
GW_Armando_plate1_TTGW52      GW_Armando_plate1_TTGW52      NG      lud_Nainaghar  lud_central
GW_Armando_plate1_TTGW53      GW_Armando_plate1_TTGW53      NG      lud_Nainaghar  lud_central
GW_Armando_plate1_TTGW55      GW_Armando_plate1_TTGW55      NG      lud_Nainaghar  lud_central
GW_Armando_plate1_TTGW57      GW_Armando_plate1_TTGW57      NG      lud_Nainaghar  lud_central
GW_Armando_plate1_TTGW58      GW_Armando_plate1_TTGW58      NG      lud_Nainaghar  lud_central
GW_Armando_plate1_TTGW59      GW_Armando_plate1_TTGW59      NG      lud_Nainaghar  lud_central
GW_Armando_plate1_TTGW63      GW_Armando_plate1_TTGW63      SP      lud_Spiti     troch_west       5
GW_Armando_plate1_TTGW64      GW_Armando_plate1_TTGW64      SP      lud_Spiti     troch_west       5
GW_Armando_plate1_TTGW65      GW_Armando_plate1_TTGW65      SP      lud_Spiti     troch_west       5
```

```
GW_Armando_plate1_TTGW66     GW_Armando_plate1_TTGW66     SP     lud_Spiti     troch_west      5
GW_Armando_plate1_TTGW68     GW_Armando_plate1_TTGW68     SP     lud_Spiti     troch_west      5
GW_Armando_plate1_TTGW70     GW_Armando_plate1_TTGW70     SA     lud_Sathrundi  lud_Sath       4
GW_Armando_plate1_TTGW71     GW_Armando_plate1_TTGW71     SA     lud_Sathrundi  lud_Sath       4
GW_Armando_plate1_TTGW72     GW_Armando_plate1_TTGW72     SA     lud_Sathrundi  lud_Sath       4
GW_Armando_plate1_TTGW74     GW_Armando_plate1_TTGW74     SA     lud_Sathrundi  lud_Sath       4
GW_Armando_plate1_TTGW78     GW_Armando_plate1_TTGW78     SA     lud_Sathrundi  lud_Sath       4
GW_Armando_plate1_TTGW_15_05  GW_Armando_plate1_TTGW_15_05  SR     lud_Sural     lud_central
GW_Armando_plate1_TTGW_15_07  GW_Armando_plate1_TTGW_15_07  SR     lud_Sural     lud_central
GW_Armando_plate1_TTGW_15_08  GW_Armando_plate1_TTGW_15_08  SR     lud_Sural     lud_central
GW_Armando_plate1_TTGW_15_09  GW_Armando_plate1_TTGW_15_09  SR     lud_Sural     lud_central
GW_Armando_plate1_UY1       GW_Armando_plate1_UY1       UY     plumb       plumb          88.01
GW_Armando_plate2_IL2       GW_Armando_plate2_IL2       IL_rep  plumb_rep    plumb_rep       83
GW_Armando_plate2_JE31G01    GW_Armando_plate2_JE31G01    KK_vi   vir_misID    vir           24
GW_Armando_plate2_JF03G01    GW_Armando_plate2_JF03G01    ST_vi   vir_misID    vir           24
GW_Armando_plate2_JF03G02    GW_Armando_plate2_JF03G02    KK_vi   vir_misID    vir           24
GW_Armando_plate2_JF07G01    GW_Armando_plate2_JF07G01    ST     plumb       plumb          87.0
GW_Armando_plate2_JF08G04    GW_Armando_plate2_JF08G04    ST     plumb       plumb          87.0
GW_Armando_plate2_JF10G02    GW_Armando_plate2_JF10G02    ST     plumb       plumb          87.0
GW_Armando_plate2_JF11G02    GW_Armando_plate2_JF11G02    ST     plumb       plumb          87.0
GW_Armando_plate2_JF12G03    GW_Armando_plate2_JF12G03    ST     plumb       plumb          87.0
GW_Armando_plate2_JF12G05    GW_Armando_plate2_JF12G05    ST     plumb       plumb          87.0
GW_Armando_plate2_JF13G02    GW_Armando_plate2_JF13G02    ST     plumb       plumb          87.0
GW_Armando_plate2_JF14G01    GW_Armando_plate2_JF14G01    KK     plumb       plumb          87.0
GW_Armando_plate2_JF14G02    GW_Armando_plate2_JF14G02    KK     plumb       plumb          87.0
GW_Armando_plate2_JF15G01    GW_Armando_plate2_JF15G01    KK     plumb       plumb          87.0
GW_Armando_plate2_JF15G02    GW_Armando_plate2_JF15G02    KK     plumb       plumb          87.0
GW_Armando_plate2_JF16G02    GW_Armando_plate2_JF16G02    KK_vi   plumb_vir    vir           24
GW_Armando_plate2_JF19G01    GW_Armando_plate2_JF19G01    KK     plumb       plumb          87.0
GW_Armando_plate2_JF20G02    GW_Armando_plate2_JF20G02    KK     plumb       plumb          87.0
GW_Armando_plate2_JF24G01    GW_Armando_plate2_JF24G01    KK     plumb       plumb          87.0
GW_Armando_plate2_JF24G03    GW_Armando_plate2_JF24G03    ST     plumb       plumb          87.0
GW_Armando_plate2_JF25G01    GW_Armando_plate2_JF25G01    KK     plumb       plumb          87.0
GW_Armando_plate2_JF26G02    GW_Armando_plate2_JF26G02    KK     plumb       plumb          87.0
GW_Armando_plate2_JF27G02    GW_Armando_plate2_JF27G02    KK     plumb       plumb          87.0
GW_Armando_plate2_JF30G01    GW_Armando_plate2_JF30G01    ST_vi   vir_misID    vir           24
GW_Armando_plate2_JG01G01    GW_Armando_plate2_JG01G01    KK     plumb       plumb          87.0
```

98	GW_Armando_plate2_JG02G01	GW_Armando_plate2_JG02G01	KK	plumb	plumb	87.0
99	GW_Armando_plate2_JG02G03	GW_Armando_plate2_JG02G03	KK	plumb	plumb	87.0
100	GW_Armando_plate2_JG10G02	GW_Armando_plate2_JG10G02	ST	plumb	plumb	87.
101	GW_Armando_plate2_JG10G03	GW_Armando_plate2_JG10G03	ST	plumb	plumb	87.
102	GW_Armando_plate2_JG12G02	GW_Armando_plate2_JG12G02	ST	plumb	plumb	87.
103	GW_Armando_plate2_JG12G03	GW_Armando_plate2_JG12G03	ST	plumb	plumb	87.
104	GW_Armando_plate2_LN11	GW_Armando_plate2_LN11	LN_rep	troch_LN_rep	troch_LN_rep	
105	GW_Armando_plate2_LN2	GW_Armando_plate2_LN2	LN	troch_LN	troch_LN	58.0
106	GW_Armando_plate2_NO_BC_TTGW05	GW_Armando_plate2_NO_BC_TTGW05	blank	blank	blank	
107	GW_Armando_plate2_NO_DNA	GW_Armando_plate2_NO_DNA	blank	blank	blank	-99.
108	GW_Armando_plate2_RF29G01	GW_Armando_plate2_RF29G01	BJ	obs_plumb	plumb_BJ	7
109	GW_Armando_plate2_TTGW02	GW_Armando_plate2_TTGW02	MN	troch_MN	troch_west	5
110	GW_Armando_plate2_TTGW03	GW_Armando_plate2_TTGW03	MN	troch_MN	troch_west	5
111	GW_Armando_plate2_TTGW05_rep3	GW_Armando_plate2_TTGW05_rep3	MN_rep	troch_MN_rep	troch_west_r	
112	GW_Armando_plate2_TTGW05_rep4	GW_Armando_plate2_TTGW05_rep4	MN_rep	troch_MN_rep	troch_west_r	
113	GW_Armando_plate2_TTGW08	GW_Armando_plate2_TTGW08	SU	lud_Sukhto	lud_central	
114	GW_Armando_plate2_TTGW09	GW_Armando_plate2_TTGW09	SU	lud_Sukhto	lud_central	
115	GW_Armando_plate2_TTGW12	GW_Armando_plate2_TTGW12	TH	lud_Thallighar	lud_central	
116	GW_Armando_plate2_TTGW14	GW_Armando_plate2_TTGW14	TH	lud_Thallighar	lud_central	
117	GW_Armando_plate2_TTGW15	GW_Armando_plate2_TTGW15	TH	lud_Thallighar	lud_central	
118	GW_Armando_plate2_TTGW16	GW_Armando_plate2_TTGW16	TH	lud_Thallighar	lud_central	
119	GW_Armando_plate2_TTGW18	GW_Armando_plate2_TTGW18	TH	lud_Thallighar	lud_central	
120	GW_Armando_plate2_TTGW20	GW_Armando_plate2_TTGW20	SR	lud_Sural	lud_central	4
121	GW_Armando_plate2_TTGW24	GW_Armando_plate2_TTGW24	SR	lud_Sural	lud_central	4
122	GW_Armando_plate2_TTGW25	GW_Armando_plate2_TTGW25	SR	lud_Sural	lud_central	4
123	GW_Armando_plate2_TTGW27	GW_Armando_plate2_TTGW27	SR	lud_Sural	lud_central	4
124	GW_Armando_plate2_TTGW28	GW_Armando_plate2_TTGW28	SR	lud_Sural	lud_central	4
125	GW_Armando_plate2_TTGW50	GW_Armando_plate2_TTGW50	NG	lud_Nainaghar	lud_central	
126	GW_Armando_plate2_TTGW51	GW_Armando_plate2_TTGW51	NG	lud_Nainaghar	lud_central	
127	GW_Armando_plate2_TTGW54	GW_Armando_plate2_TTGW54	NG	lud_Nainaghar	lud_central	
128	GW_Armando_plate2_TTGW56	GW_Armando_plate2_TTGW56	NG	lud_Nainaghar	lud_central	
129	GW_Armando_plate2_TTGW60	GW_Armando_plate2_TTGW60	SP	lud_Spiti	troch_west	5
130	GW_Armando_plate2_TTGW61	GW_Armando_plate2_TTGW61	SP	lud_Spiti	troch_west	5
131	GW_Armando_plate2_TTGW62	GW_Armando_plate2_TTGW62	SP	lud_Spiti	troch_west	5
132	GW_Armando_plate2_TTGW67	GW_Armando_plate2_TTGW67	SP	lud_Spiti	troch_west	5
133	GW_Armando_plate2_TTGW69	GW_Armando_plate2_TTGW69	SP	lud_Spiti	troch_west	5
134	GW_Armando_plate2_TTGW73	GW_Armando_plate2_TTGW73	SA	lud_Sathrundi	lud_Sath	

```
GW_Armando_plate2_TTGW75     GW_Armando_plate2_TTGW75     SA     lud_Sathrundi  lud_Sath
GW_Armando_plate2_TTGW77     GW_Armando_plate2_TTGW77     SA     lud_Sathrundi  lud_Sath
GW_Armando_plate2_TTGW79     GW_Armando_plate2_TTGW79     SA     lud_Sathrundi  lud_Sath
GW_Armando_plate2_TTGW80     GW_Armando_plate2_TTGW80     SA     lud_Sathrundi  lud_Sath
GW_Armando_plate2_TTGW_15_01   GW_Armando_plate2_TTGW_15_01   SR     lud_Sural     lud_central
GW_Armando_plate2_TTGW_15_02   GW_Armando_plate2_TTGW_15_02   SR     lud_Sural     lud_central
GW_Armando_plate2_TTGW_15_03   GW_Armando_plate2_TTGW_15_03   SR     lud_Sural     lud_central
GW_Armando_plate2_TTGW_15_04   GW_Armando_plate2_TTGW_15_04   SR     lud_Sural     lud_central
GW_Armando_plate2_TTGW_15_06   GW_Armando_plate2_TTGW_15_06   SR     lud_Sural     lud_central
GW_Armando_plate2_TTGW_15_10   GW_Armando_plate2_TTGW_15_10   SR     lud_Sural     lud_central
GW_Lane5_AA1              GW_Lane5_AA1              AA      vir_S        vir_S          25.0
GW_Lane5_AA10             GW_Lane5_AA10             AA      vir_S        vir_S          33.0
GW_Lane5_AA11             GW_Lane5_AA11             AA      vir_S        vir_S          34.0
GW_Lane5_AA3              GW_Lane5_AA3              AA      vir_S        vir_S          26.0
GW_Lane5_AA4              GW_Lane5_AA4              AA      vir_S        vir_S          27.0
GW_Lane5_AA5              GW_Lane5_AA5              AA      vir_S        vir_S          28.0
GW_Lane5_AA6              GW_Lane5_AA6              AA      vir_S        vir_S          29.0
GW_Lane5_AA7              GW_Lane5_AA7              AA      vir_S        vir_S          30.0
GW_Lane5_AA8              GW_Lane5_AA8              AA      vir_S        vir_S          31.0
GW_Lane5_AA9              GW_Lane5_AA9              AA      vir_S        vir_S          32.0
GW_Lane5_AB1              GW_Lane5_AB1              AB_rep   vir_rep      vir_rep         20.0
GW_Lane5_AB2              GW_Lane5_AB2              AB      vir          vir            21.0
GW_Lane5_AN1              GW_Lane5_AN1              AN      plumb        plumb          80.0
GW_Lane5_AN2              GW_Lane5_AN2              AN      plumb        plumb          81.0
GW_Lane5_BK2              GW_Lane5_BK2              BK      plumb        plumb          78.0
GW_Lane5_BK3              GW_Lane5_BK3              BK      plumb        plumb          79.0
GW_Lane5_DA2              GW_Lane5_DA2              XN      obs          obs            73.0
GW_Lane5_DA3              GW_Lane5_DA3              XN      obs          obs            74.0
GW_Lane5_DA4              GW_Lane5_DA4              XN      obs          obs            75.0
GW_Lane5_DA6              GW_Lane5_DA6              XN      obs          low_reads       76.0
GW_Lane5_DA7              GW_Lane5_DA7              XN      obs          obs            77.0
GW_Lane5_EM1              GW_Lane5_EM1              EM      troch_EM     troch_EM        72.0
GW_Lane5_IL1              GW_Lane5_IL1              IL      plumb        plumb          82.0
GW_Lane5_IL2              GW_Lane5_IL2              IL_rep   plumb_rep     plumb_rep        83.0
GW_Lane5_IL4              GW_Lane5_IL4              IL      plumb        plumb          84.0
GW_Lane5_KS1              GW_Lane5_KS1              OV      lud_KS       lud_KS         40.0
GW_Lane5_KS2              GW_Lane5_KS2              OV      lud_KS       lud_KS         40.0
```

172	GW_Lane5_LN1	GW_Lane5_LN1	LN	troch_LN	troch_LN	57.0
173	GW_Lane5_LN10	GW_Lane5_LN10	LN	troch_LN	troch_LN	64.0
174	GW_Lane5_LN11	GW_Lane5_LN11	LN	troch_LN	troch_LN	65.0
175	GW_Lane5_LN12	GW_Lane5_LN12	LN	troch_LN	troch_LN	66.0
176	GW_Lane5_LN14	GW_Lane5_LN14	LN	troch_LN	troch_LN	67.0
177	GW_Lane5_LN16	GW_Lane5_LN16	LN	troch_LN	troch_LN	68.0
178	GW_Lane5_LN18	GW_Lane5_LN18	LN	troch_LN	troch_LN	69.0
179	GW_Lane5_LN19	GW_Lane5_LN19	LN	troch_LN	troch_LN	70.0
180	GW_Lane5_LN2	GW_Lane5_LN2	LN_rep	troch_LN_rep	troch_LN_rep	58.0
181	GW_Lane5_LN20	GW_Lane5_LN20	LN	troch_LN	troch_LN	71.0
182	GW_Lane5_LN3	GW_Lane5_LN3	LN	troch_LN	troch_LN	59.0
183	GW_Lane5_LN4	GW_Lane5_LN4	LN	troch_LN	troch_LN	60.0
184	GW_Lane5_LN6	GW_Lane5_LN6	LN	troch_LN	troch_LN	61.0
185	GW_Lane5_LN7	GW_Lane5_LN7	LN	troch_LN	troch_LN	62.0
186	GW_Lane5_LN8	GW_Lane5_LN8	LN	troch_LN	troch_LN	63.0
187	GW_Lane5_MN1	GW_Lane5_MN1	MN	troch_MN	troch_west	51.0
188	GW_Lane5_MN12	GW_Lane5_MN12	MN	troch_MN	troch_west	56.0
189	GW_Lane5_MN3	GW_Lane5_MN3	MN	troch_MN	troch_west	52.0
190	GW_Lane5_MN5	GW_Lane5_MN5	MN	troch_MN	troch_west	53.0
191	GW_Lane5_MN8	GW_Lane5_MN8	MN	troch_MN	troch_west	54.0
192	GW_Lane5_MN9	GW_Lane5_MN9	MN	troch_MN	troch_west	55.0
193	GW_Lane5_NA1	GW_Lane5_NA1	NR	lud_PK	lud_PK	39.2
194	GW_Lane5_NA3-3ul	GW_Lane5_NA3-3ul	NR	lud_PK	lud_PK	39.2
195	GW_Lane5_PT11	GW_Lane5_PT11	KL	lud_KL	lud_central	42.0
196	GW_Lane5_PT12	GW_Lane5_PT12	KL	lud_KL	lud_central	42.0
197	GW_Lane5_PT2	GW_Lane5_PT2	ML	lud_ML	lud_ML	51.0
198	GW_Lane5_PT3	GW_Lane5_PT3	PA	lud_PA	lud_central	46.0
199	GW_Lane5_PT4	GW_Lane5_PT4	PA	lud_PA	lud_central	46.0
200	GW_Lane5_PT6	GW_Lane5_PT6	KL	lud_KL	lud_central	42.0
201	GW_Lane5_SH1	GW_Lane5_SH1	PK	lud_PK	lud_PK	39.1
202	GW_Lane5_SH2	GW_Lane5_SH2	PK	lud_PK	lud_PK	39.1
203	GW_Lane5_SH4	GW_Lane5_SH4	PK	lud_PK	lud_PK	39.1
204	GW_Lane5_SH5	GW_Lane5_SH5	PK	lud_PK	lud_PK	39.1
205	GW_Lane5_SL1	GW_Lane5_SL1	SL	plumb	plumb	95.0
206	GW_Lane5_SL2	GW_Lane5_SL2	SL	plumb	plumb	96.0
207	GW_Lane5_ST1	GW_Lane5_ST1	ST	plumb	plumb	85.0
208	GW_Lane5_ST12	GW_Lane5_ST12	ST	plumb	plumb	87.0

209	GW_Lane5_ST3	GW_Lane5_ST3	ST	plumb	plumb	86.0
210	GW_Lane5_STvi1	GW_Lane5_STvi1	STvi	vir	vir	22.0
211	GW_Lane5_STvi2	GW_Lane5_STvi2	STvi	vir	vir	23.0
212	GW_Lane5_STvi3	GW_Lane5_STvi3	STvi	vir	vir	24.0
213	GW_Lane5_TA1	GW_Lane5_TA1	TA	plumb	plumb	94.0
214	GW_Lane5_TL1	GW_Lane5_TL1	TL	vir	vir	9.0
215	GW_Lane5_TL10	GW_Lane5_TL10	TL	vir	vir	17.0
216	GW_Lane5_TL11	GW_Lane5_TL11	TL	vir	vir	18.0
217	GW_Lane5_TL12	GW_Lane5_TL12	TL	vir	vir	19.0
218	GW_Lane5_TL2	GW_Lane5_TL2	TL	vir	vir	10.0
219	GW_Lane5_TL3	GW_Lane5_TL3	TL_rep	vir_rep	vir_rep	11.0
220	GW_Lane5_TL4	GW_Lane5_TL4	TL	vir	vir	12.0
221	GW_Lane5_TL5	GW_Lane5_TL5	TL	vir	vir	13.0
222	GW_Lane5_TL7	GW_Lane5_TL7	TL	vir	vir	14.0
223	GW_Lane5_TL8	GW_Lane5_TL8	TL	vir	vir	15.0
224	GW_Lane5_TL9	GW_Lane5_TL9	TL	vir	vir	16.0
225	GW_Lane5_TU1	GW_Lane5_TU1	TU	nit	nit	35.0
226	GW_Lane5_TU2	GW_Lane5_TU2	TU	nit	nit	36.0
227	GW_Lane5_UY1	GW_Lane5_UY1	UY_rep	plumb_rep	plumb_rep	88.0
228	GW_Lane5_UY2	GW_Lane5_UY2	UY	plumb	plumb	89.0
229	GW_Lane5_UY3	GW_Lane5_UY3	UY	plumb	plumb	90.0
230	GW_Lane5_UY4	GW_Lane5_UY4	UY	plumb	plumb	91.0
231	GW_Lane5_UY5	GW_Lane5_UY5	UY	plumb	plumb	92.0
232	GW_Lane5_UY6	GW_Lane5_UY6	UY	plumb	plumb	93.0
233	GW_Lane5_YK1	GW_Lane5_YK1	YK	vir	vir	1.0
234	GW_Lane5_YK11	GW_Lane5_YK11	YK	vir	vir	8.0
235	GW_Lane5_YK3	GW_Lane5_YK3	YK	vir	vir	2.0
236	GW_Lane5_YK4	GW_Lane5_YK4	YK	vir	vir	3.0
237	GW_Lane5_YK5	GW_Lane5_YK5	YK	vir	vir	4.0
238	GW_Lane5_YK6	GW_Lane5_YK6	YK	vir	vir	5.0
239	GW_Lane5_YK7	GW_Lane5_YK7	YK	vir	vir	6.0
240	GW_Lane5_YK9	GW_Lane5_YK9	YK	vir	vir	7.0
241	GW_Liz_GBS_Liz10045	GW_Liz_GBS_Liz10045	ML	lud	lud_ML	51.01
242	GW_Liz_GBS_Liz10094	GW_Liz_GBS_Liz10094	ML	lud	lud_ML	51.02
243	GW_Liz_GBS_Liz5101	GW_Liz_GBS_Liz5101	ML	lud	lud_ML	51.03
244	GW_Liz_GBS_Liz5101_R	GW_Liz_GBS_Liz5101_R	ML_rep	lud_rep	lud_ML_rep	51.
245	GW_Liz_GBS_Liz5118	GW_Liz_GBS_Liz5118	ML	lud	lud_ML	51.05

246	GW_Liz_GBS_Liz5139	GW_Liz_GBS_Liz5139	ML	lud	lud_ML	51.06
247	GW_Liz_GBS_Liz5142	GW_Liz_GBS_Liz5142	ML	lud	lud_ML	51.07
248	GW_Liz_GBS_Liz5144	GW_Liz_GBS_Liz5144	ML	lud	lud_ML	51.08
249	GW_Liz_GBS_Liz5150	GW_Liz_GBS_Liz5150	ML	lud	lud_ML	51.09
250	GW_Liz_GBS_Liz5159	GW_Liz_GBS_Liz5159	ML	lud_chick	lud_ML	51.1
251	GW_Liz_GBS_Liz5162	GW_Liz_GBS_Liz5162	ML	lud_chick	lud_ML	51.11
252	GW_Liz_GBS_Liz5163	GW_Liz_GBS_Liz5163	ML	lud_chick	lud_ML	51.12
253	GW_Liz_GBS_Liz5164	GW_Liz_GBS_Liz5164	ML	lud_chick	lud_ML	51.13
254	GW_Liz_GBS_Liz5165	GW_Liz_GBS_Liz5165	ML	lud	lud_ML	51.14
255	GW_Liz_GBS_Liz5167	GW_Liz_GBS_Liz5167	ML	lud_chick	lud_ML	51.15
256	GW_Liz_GBS_Liz5168	GW_Liz_GBS_Liz5168	ML	lud_chick	lud_ML	51.16
257	GW_Liz_GBS_Liz5169	GW_Liz_GBS_Liz5169	ML	lud_chick	lud_ML	51.17
258	GW_Liz_GBS_Liz5171	GW_Liz_GBS_Liz5171	ML	lud	lud_ML	51.18
259	GW_Liz_GBS_Liz5172	GW_Liz_GBS_Liz5172	ML	lud_chick	lud_ML	51.19
260	GW_Liz_GBS_Liz5173	GW_Liz_GBS_Liz5173	ML	lud_chick	lud_ML	51.2
261	GW_Liz_GBS_Liz5174	GW_Liz_GBS_Liz5174	ML	lud	lud_ML	51.21
262	GW_Liz_GBS_Liz5175	GW_Liz_GBS_Liz5175	ML	lud	lud_ML	51.22
263	GW_Liz_GBS_Liz5176	GW_Liz_GBS_Liz5176	ML	lud	lud_ML	51.23
264	GW_Liz_GBS_Liz5177	GW_Liz_GBS_Liz5177	ML	lud_chick	lud_ML	51.24
265	GW_Liz_GBS_Liz5178	GW_Liz_GBS_Liz5178	ML	lud_chick	lud_ML	51.25
266	GW_Liz_GBS_Liz5179	GW_Liz_GBS_Liz5179	ML	lud_chick	lud_ML	51.26
267	GW_Liz_GBS_Liz5180	GW_Liz_GBS_Liz5180	ML	lud	lud_ML	51.27
268	GW_Liz_GBS_Liz5182	GW_Liz_GBS_Liz5182	ML	lud_chick	lud_ML	51.28
269	GW_Liz_GBS_Liz5184	GW_Liz_GBS_Liz5184	ML	lud_chick	lud_ML	51.29
270	GW_Liz_GBS_Liz5185	GW_Liz_GBS_Liz5185	ML	lud	lud_ML	51.3
271	GW_Liz_GBS_Liz5186	GW_Liz_GBS_Liz5186	ML	lud_chick	lud_ML	51.31
272	GW_Liz_GBS_Liz5187	GW_Liz_GBS_Liz5187	ML	lud_chick	lud_ML	51.32
273	GW_Liz_GBS_Liz5188	GW_Liz_GBS_Liz5188	ML	lud	lud_ML	51.33
274	GW_Liz_GBS_Liz5189	GW_Liz_GBS_Liz5189	ML	lud_chick	lud_ML	51.34
275	GW_Liz_GBS_Liz5190	GW_Liz_GBS_Liz5190	ML	lud_chick	lud_ML	51.35
276	GW_Liz_GBS_Liz5191	GW_Liz_GBS_Liz5191	ML	lud_chick	lud_ML	51.36
277	GW_Liz_GBS_Liz5192	GW_Liz_GBS_Liz5192	ML	lud_chick	lud_ML	51.37
278	GW_Liz_GBS_Liz5193	GW_Liz_GBS_Liz5193	ML	lud_chick	lud_ML	51.38
279	GW_Liz_GBS_Liz5194	GW_Liz_GBS_Liz5194	ML	lud_chick	lud_ML	51.39
280	GW_Liz_GBS_Liz5195	GW_Liz_GBS_Liz5195	ML	lud	lud_ML	51.4
281	GW_Liz_GBS_Liz5197	GW_Liz_GBS_Liz5197	ML	lud	lud_ML	51.41
282	GW_Liz_GBS_Liz5199	GW_Liz_GBS_Liz5199	ML	lud_chick	lud_ML	51.42

283	GW_Liz_GBS_Liz6002	GW_Liz_GBS_Liz6002	ML	lud	lud_ML	51.43
284	GW_Liz_GBS_Liz6006	GW_Liz_GBS_Liz6006	ML	lud	lud_ML	51.44
285	GW_Liz_GBS_Liz6008	GW_Liz_GBS_Liz6008	ML	lud	lud_ML	51.45
286	GW_Liz_GBS_Liz6009	GW_Liz_GBS_Liz6009	ML	lud	lud_ML	51.46
287	GW_Liz_GBS_Liz6010	GW_Liz_GBS_Liz6010	ML	lud	lud_ML	51.47
288	GW_Liz_GBS_Liz6012	GW_Liz_GBS_Liz6012	ML	lud	lud_ML	51.48
289	GW_Liz_GBS_Liz6014	GW_Liz_GBS_Liz6014	ML	lud	lud_ML	51.49
290	GW_Liz_GBS_Liz6055	GW_Liz_GBS_Liz6055	ML	lud	lud_ML	51.5
291	GW_Liz_GBS_Liz6057	GW_Liz_GBS_Liz6057	ML	lud	lud_ML	51.51
292	GW_Liz_GBS_Liz6060	GW_Liz_GBS_Liz6060	ML	lud	lud_ML	51.52
293	GW_Liz_GBS_Liz6062	GW_Liz_GBS_Liz6062	ML	lud	lud_ML	51.53
294	GW_Liz_GBS_Liz6063	GW_Liz_GBS_Liz6063	ML	lud	lud_ML	51.54
295	GW_Liz_GBS_Liz6066	GW_Liz_GBS_Liz6066	ML	lud	lud_ML	51.55
296	GW_Liz_GBS_Liz6072	GW_Liz_GBS_Liz6072	ML	lud	lud_ML	51.56
297	GW_Liz_GBS_Liz6079	GW_Liz_GBS_Liz6079	ML	lud	lud_ML	51.57
298	GW_Liz_GBS_Liz6203	GW_Liz_GBS_Liz6203	ML	lud_chick	lud_ML	51.58
299	GW_Liz_GBS_Liz6204	GW_Liz_GBS_Liz6204	ML	lud_chick	lud_ML	51.59
300	GW_Liz_GBS_Liz6461	GW_Liz_GBS_Liz6461	ML	lud	lud_ML	51.6
301	GW_Liz_GBS_Liz6472	GW_Liz_GBS_Liz6472	ML	lud	lud_ML	51.61
302	GW_Liz_GBS_Liz6478	GW_Liz_GBS_Liz6478	ML	lud	lud_ML	51.62
303	GW_Liz_GBS_Liz6766	GW_Liz_GBS_Liz6766	ML	lud	lud_ML	51.63
304	GW_Liz_GBS_Liz6776	GW_Liz_GBS_Liz6776	ML	lud	lud_ML	51.64
305	GW_Liz_GBS_Liz6794	GW_Liz_GBS_Liz6794	ML	lud	lud_ML	51.65
306	GW_Liz_GBS_P_fusc	GW_Liz_GBS_P_fusc	fusc	fusc	fusc	101.0
307	GW_Liz_GBS_P_h_man	GW_Liz_GBS_P_h_man	hmand	hmand	hmand	102.0
308	GW_Liz_GBS_P_humei	GW_Liz_GBS_P_humei	hume	hume	hume	103.0
309	GW_Liz_GBS_P_inor	GW_Liz_GBS_P_inor	inor	inor	inor	104.0
310	GW_Liz_GBS_S_burk	GW_Liz_GBS_S_burk	burk	burk	burk	105.0

GOOD NEWS: names of individuals in metadata file and genotype ind file match perfectly.

Filtering

Filter out duplicate runs (indicated with `_rep` in `Fst_group` column)

```
selection = occursin.("_rep", ind_with_metadata.Fst_group)
println("""Filtering out these runs because they are duplicates of another,
```

```
    according to having "rep" in Fst_group: """)
    display(ind_with_metadata.ind[selection])
    ind_with_metadata_indFiltered = ind_with_metadata[Not(selection), :];
    geno_indFiltered = view(geno, Not(selection), :);  # use of view() avoids copying large memory ob
```

```
Filtering out these runs because they are duplicates of another,
according to having "rep" in Fst_group:
```

```
11-element Vector{String}:
 "GW_Armando_plate1_TTGW05_rep1"
 "GW_Armando_plate2_IL2"
 "GW_Armando_plate2_LN11"
 "GW_Armando_plate2_TTGW05_rep3"
 "GW_Armando_plate2_TTGW05_rep4"
 "GW_Lane5_AB1"
 "GW_Lane5_IL2"
 "GW_Lane5_LN2"
 "GW_Lane5_TL3"
 "GW_Lane5_UY1"
 "GW_Liz_GBS_Liz5101_R"
```

Filter specific individuals

If there are certain individuals that we want to filter out prior to any additional analysis, we can do so here by setting filter to `true` and specifying the individual row numbers in `filter_out_inds`:

```
filter = true
# Specify individuals to filter out:
filter_out_inds = ["GW_Liz_GBS_P_fusc", "GW_Liz_GBS_P_h_man", "GW_Liz_GBS_P_humei", "GW_Liz_GBS_P_in
if filter
    selection = map(in(filter_out_inds), ind_with_metadata_indFiltered.ind)
    filtered_out = ind_with_metadata_indFiltered.ind[selection]
    ind_with_metadata_indFiltered = ind_with_metadata_indFiltered[Not(selection), :]
    geno_indFiltered = view(geno_indFiltered, Not(selection), :)
    println("Specific individuals filtered out as requested: ")
    display(filtered_out)
else
    println("No specific individuals filtered (because filter not true)")
end
```

```
Specific individuals filtered out as requested:


5-element Vector{String}:
 "GW_Liz_GBS_P_fusc"
 "GW_Liz_GBS_P_h_man"
 "GW_Liz_GBS_P_humei"
 "GW_Liz_GBS_P_inor"
 "GW_Liz_GBS_S_burk"
```

Filter individuals based on missing genotypes

Here we determine number of missing SNPs per individual (40% for this round), and filter out those individual datasets with more than a certain percent of missing SNPs:

```
SNPmissing_percent_allowed_per_ind = 40   # this is the percentage threshold
threshold_missing = loci_count * SNPmissing_percent_allowed_per_ind/100
numMissings = sum(geno_indFiltered .== -1, dims=2)
ind_with_metadata_indFiltered.numMissings .= numMissings
selection = vec(numMissings .<= threshold_missing) # the vec command converts to BitVector rather tha
println("Filtering out these individuals based on too many missing genotypes: ")
filtered_inds = ind_with_metadata_indFiltered.ind[selection.==false]
println(DataFrame(filtered_inds = filtered_inds)) # did this to print all lines
ind_with_metadata_indFiltered = ind_with_metadata_indFiltered[selection, :]
geno_indFiltered = view(geno_indFiltered, selection, :);
println()
println("Here are the remaining individuals: ")
println(DataFrame(ind_with_metadata_indFiltered))
```

```
Filtering out these individuals based on too many missing genotypes:
33×1 DataFrame
 Row │ filtered_inds
     │ String
─────┼──────────────────────────────────
   1 │ GW_Armando_plate1_JG08G02
   2 │ GW_Armando_plate1_JG10G01
   3 │ GW_Armando_plate1_NO_BC_TTGW05
   4 │ GW_Armando_plate1_NO_DNA
   5 │ GW_Armando_plate1_TTGW21
   6 │ GW_Armando_plate1_TTGW71
   7 │ GW_Armando_plate2_NO_BC_TTGW05
```

```
  8 │ GW_Armando_plate2_NO_DNA
  9 │ GW_Armando_plate2_TTGW15
 10 │ GW_Lane5_AA10
 11 │ GW_Lane5_DA6
 12 │ GW_Lane5_LN11
 13 │ GW_Liz_GBS_Liz5101
 14 │ GW_Liz_GBS_Liz5118
 15 │ GW_Liz_GBS_Liz5139
 16 │ GW_Liz_GBS_Liz5142
 17 │ GW_Liz_GBS_Liz5150
 18 │ GW_Liz_GBS_Liz5159
 19 │ GW_Liz_GBS_Liz5162
 20 │ GW_Liz_GBS_Liz5169
 21 │ GW_Liz_GBS_Liz5171
 22 │ GW_Liz_GBS_Liz5172
 23 │ GW_Liz_GBS_Liz5174
 24 │ GW_Liz_GBS_Liz5176
 25 │ GW_Liz_GBS_Liz5177
 26 │ GW_Liz_GBS_Liz5180
 27 │ GW_Liz_GBS_Liz5186
 28 │ GW_Liz_GBS_Liz5187
 29 │ GW_Liz_GBS_Liz5192
 30 │ GW_Liz_GBS_Liz5195
 31 │ GW_Liz_GBS_Liz6012
 32 │ GW_Liz_GBS_Liz6203
 33 │ GW_Liz_GBS_Liz6766
```

Here are the remaining individuals:

```
261×7 DataFrame
 Row │ ind                   ID                    location  group       Fst_group   plot_order  numMissing
     │ String                String31              String7   String15    String15    Float64     Int64
─────┼──────────────────────────────────────────────────────────────────────────────────────────────────────
   1 │ GW_Armando_plate1_AB1       GW_Armando_plate1_AB1       AB        vir         vir          20.01
   2 │ GW_Armando_plate1_JF07G02   GW_Armando_plate1_JF07G02   ST        plumb       plumb        87.001
   3 │ GW_Armando_plate1_JF07G03   GW_Armando_plate1_JF07G03   ST        plumb       plumb        87.002
   4 │ GW_Armando_plate1_JF07G04   GW_Armando_plate1_JF07G04   ST        plumb       plumb        87.003
   5 │ GW_Armando_plate1_JF08G02   GW_Armando_plate1_JF08G02   ST        plumb       plumb        87.004
   6 │ GW_Armando_plate1_JF09G01   GW_Armando_plate1_JF09G01   ST        plumb       plumb        87.005
```

7	GW_Armando_plate1_JF09G02	GW_Armando_plate1_JF09G02	ST	plumb	plumb	87.006
8	GW_Armando_plate1_JF10G03	GW_Armando_plate1_JF10G03	ST	plumb_vir	plumb_vir	98.0
9	GW_Armando_plate1_JF11G01	GW_Armando_plate1_JF11G01	ST	plumb	plumb	87.008
10	GW_Armando_plate1_JF12G01	GW_Armando_plate1_JF12G01	ST	plumb	plumb	87.009
11	GW_Armando_plate1_JF12G02	GW_Armando_plate1_JF12G02	ST	plumb	plumb	87.01
12	GW_Armando_plate1_JF12G04	GW_Armando_plate1_JF12G04	ST_vi	vir	vir	24.001
13	GW_Armando_plate1_JF13G01	GW_Armando_plate1_JF13G01	ST	plumb	plumb	87.011
14	GW_Armando_plate1_JF15G03	GW_Armando_plate1_JF15G03	KK	plumb	plumb	87.012
15	GW_Armando_plate1_JF16G01	GW_Armando_plate1_JF16G01	KK_vi	plumb_vir	vir	24.04
16	GW_Armando_plate1_JF20G01	GW_Armando_plate1_JF20G01	KK	plumb	plumb	87.014
17	GW_Armando_plate1_JF22G01	GW_Armando_plate1_JF22G01	KK	plumb	plumb	87.015
18	GW_Armando_plate1_JF23G01	GW_Armando_plate1_JF23G01	KK	plumb	plumb	87.016
19	GW_Armando_plate1_JF23G02	GW_Armando_plate1_JF23G02	KK	plumb	plumb	87.017
20	GW_Armando_plate1_JF24G02	GW_Armando_plate1_JF24G02	KK	plumb	plumb	87.018
21	GW_Armando_plate1_JF26G01	GW_Armando_plate1_JF26G01	ST	plumb	plumb	87.019
22	GW_Armando_plate1_JF27G01	GW_Armando_plate1_JF27G01	ST	plumb	plumb	87.02
23	GW_Armando_plate1_JF29G01	GW_Armando_plate1_JF29G01	ST	plumb	plumb	87.021
24	GW_Armando_plate1_JF29G02	GW_Armando_plate1_JF29G02	ST	plumb	plumb	87.022
25	GW_Armando_plate1_JF29G03	GW_Armando_plate1_JF29G03	ST	plumb	plumb	87.023
26	GW_Armando_plate1_JG02G02	GW_Armando_plate1_JG02G02	KK	plumb	plumb	87.024
27	GW_Armando_plate1_JG02G04	GW_Armando_plate1_JG02G04	KK	plumb	plumb	87.025
28	GW_Armando_plate1_JG08G01	GW_Armando_plate1_JG08G01	ST	plumb	plumb	87.026
29	GW_Armando_plate1_JG12G01	GW_Armando_plate1_JG12G01	ST	plumb	plumb	87.029
30	GW_Armando_plate1_JG17G01	GW_Armando_plate1_JG17G01	ST	plumb_vir	plumb	77.92
31	GW_Armando_plate1_RF20G01	GW_Armando_plate1_RF20G01	BJ	obs_plumb	plumb_BJ	77.5
32	GW_Armando_plate1_RF29G02	GW_Armando_plate1_RF29G02	BJ	obs_plumb	plumb_BJ	77.5
33	GW_Armando_plate1_TL3	GW_Armando_plate1_TL3	TL	vir	vir	11.01
34	GW_Armando_plate1_TTGW01	GW_Armando_plate1_TTGW01	MN	troch_MN	troch_west	53.0
35	GW_Armando_plate1_TTGW05_rep2	GW_Armando_plate1_TTGW05_rep2	MN	troch_MN	troch_west	5
36	GW_Armando_plate1_TTGW06	GW_Armando_plate1_TTGW06	SU	lud_Sukhto	lud_central	47.0
37	GW_Armando_plate1_TTGW07	GW_Armando_plate1_TTGW07	SU	lud_Sukhto	lud_central	47.0
38	GW_Armando_plate1_TTGW10	GW_Armando_plate1_TTGW10	SU	lud_Sukhto	lud_central	47.0
39	GW_Armando_plate1_TTGW11	GW_Armando_plate1_TTGW11	SU	lud_Sukhto	lud_central	47.0
40	GW_Armando_plate1_TTGW13	GW_Armando_plate1_TTGW13	TH	lud_Thallighar	lud_central	43
41	GW_Armando_plate1_TTGW17	GW_Armando_plate1_TTGW17	TH	lud_Thallighar	lud_central	43
42	GW_Armando_plate1_TTGW19	GW_Armando_plate1_TTGW19	TH	lud_Thallighar	lud_central	43
43	GW_Armando_plate1_TTGW22	GW_Armando_plate1_TTGW22	SR	lud_Sural	lud_central	45.0

```
GW_Armando_plate1_TTGW23    GW_Armando_plate1_TTGW23    SR     lud_Sural     lud_central    45.0
GW_Armando_plate1_TTGW29    GW_Armando_plate1_TTGW29    SR     lud_Sural     lud_central    45.0
GW_Armando_plate1_TTGW52    GW_Armando_plate1_TTGW52    NG     lud_Nainaghar  lud_central    49.
GW_Armando_plate1_TTGW53    GW_Armando_plate1_TTGW53    NG     lud_Nainaghar  lud_central    49.
GW_Armando_plate1_TTGW55    GW_Armando_plate1_TTGW55    NG     lud_Nainaghar  lud_central    49.
GW_Armando_plate1_TTGW57    GW_Armando_plate1_TTGW57    NG     lud_Nainaghar  lud_central    49.
GW_Armando_plate1_TTGW58    GW_Armando_plate1_TTGW58    NG     lud_Nainaghar  lud_central    49.
GW_Armando_plate1_TTGW59    GW_Armando_plate1_TTGW59    NG     lud_Nainaghar  lud_central    49.
GW_Armando_plate1_TTGW63    GW_Armando_plate1_TTGW63    SP     lud_Spiti     troch_west     55.0
GW_Armando_plate1_TTGW64    GW_Armando_plate1_TTGW64    SP     lud_Spiti     troch_west     55.0
GW_Armando_plate1_TTGW65    GW_Armando_plate1_TTGW65    SP     lud_Spiti     troch_west     55.0
GW_Armando_plate1_TTGW66    GW_Armando_plate1_TTGW66    SP     lud_Spiti     troch_west     55.0
GW_Armando_plate1_TTGW68    GW_Armando_plate1_TTGW68    SP     lud_Spiti     troch_west     55.0
GW_Armando_plate1_TTGW70    GW_Armando_plate1_TTGW70    SA     lud_Sathrundi  lud_Sath      41.0
GW_Armando_plate1_TTGW72    GW_Armando_plate1_TTGW72    SA     lud_Sathrundi  lud_Sath      41.0
GW_Armando_plate1_TTGW74    GW_Armando_plate1_TTGW74    SA     lud_Sathrundi  lud_Sath      41.0
GW_Armando_plate1_TTGW78    GW_Armando_plate1_TTGW78    SA     lud_Sathrundi  lud_Sath      41.0
GW_Armando_plate1_TTGW_15_05  GW_Armando_plate1_TTGW_15_05  SR     lud_Sural     lud_central    45
GW_Armando_plate1_TTGW_15_07  GW_Armando_plate1_TTGW_15_07  SR     lud_Sural     lud_central    45
GW_Armando_plate1_TTGW_15_08  GW_Armando_plate1_TTGW_15_08  SR     lud_Sural     lud_central    45
GW_Armando_plate1_TTGW_15_09  GW_Armando_plate1_TTGW_15_09  SR     lud_Sural     lud_central    45
GW_Armando_plate1_UY1       GW_Armando_plate1_UY1       UY     plumb       plumb        88.01
GW_Armando_plate2_JE31G01    GW_Armando_plate2_JE31G01    KK_vi   vir_misID     vir         24.00
GW_Armando_plate2_JF03G01    GW_Armando_plate2_JF03G01    ST_vi   vir_misID     vir         24.00
GW_Armando_plate2_JF03G02    GW_Armando_plate2_JF03G02    KK_vi   vir_misID     vir         24.00
GW_Armando_plate2_JF07G01    GW_Armando_plate2_JF07G01    ST     plumb       plumb        87.031
GW_Armando_plate2_JF08G04    GW_Armando_plate2_JF08G04    ST     plumb       plumb        87.032
GW_Armando_plate2_JF10G02    GW_Armando_plate2_JF10G02    ST     plumb       plumb        87.033
GW_Armando_plate2_JF11G02    GW_Armando_plate2_JF11G02    ST     plumb       plumb        87.034
GW_Armando_plate2_JF12G03    GW_Armando_plate2_JF12G03    ST     plumb       plumb        87.035
GW_Armando_plate2_JF12G05    GW_Armando_plate2_JF12G05    ST     plumb       plumb        87.036
GW_Armando_plate2_JF13G02    GW_Armando_plate2_JF13G02    ST     plumb       plumb        87.037
GW_Armando_plate2_JF14G01    GW_Armando_plate2_JF14G01    KK     plumb       plumb        87.038
GW_Armando_plate2_JF14G02    GW_Armando_plate2_JF14G02    KK     plumb       plumb        87.039
GW_Armando_plate2_JF15G01    GW_Armando_plate2_JF15G01    KK     plumb       plumb        87.04
GW_Armando_plate2_JF15G02    GW_Armando_plate2_JF15G02    KK     plumb       plumb        87.041
GW_Armando_plate2_JF16G02    GW_Armando_plate2_JF16G02    KK_vi   plumb_vir     vir         24.04
```

81	GW_Armando_plate2_JF19G01	GW_Armando_plate2_JF19G01	KK	plumb	plumb	87.043
82	GW_Armando_plate2_JF20G02	GW_Armando_plate2_JF20G02	KK	plumb	plumb	87.044
83	GW_Armando_plate2_JF24G01	GW_Armando_plate2_JF24G01	KK	plumb	plumb	87.045
84	GW_Armando_plate2_JF24G03	GW_Armando_plate2_JF24G03	ST	plumb	plumb	87.046
85	GW_Armando_plate2_JF25G01	GW_Armando_plate2_JF25G01	KK	plumb	plumb	87.047
86	GW_Armando_plate2_JF26G02	GW_Armando_plate2_JF26G02	KK	plumb	plumb	87.048
87	GW_Armando_plate2_JF27G02	GW_Armando_plate2_JF27G02	KK	plumb	plumb	87.049
88	GW_Armando_plate2_JF30G01	GW_Armando_plate2_JF30G01	ST_vi	vir_misID	vir	24.00
89	GW_Armando_plate2_JG01G01	GW_Armando_plate2_JG01G01	KK	plumb	plumb	87.05
90	GW_Armando_plate2_JG02G01	GW_Armando_plate2_JG02G01	KK	plumb	plumb	87.051
91	GW_Armando_plate2_JG02G03	GW_Armando_plate2_JG02G03	KK	plumb	plumb	87.052
92	GW_Armando_plate2_JG10G02	GW_Armando_plate2_JG10G02	ST	plumb	plumb	87.053
93	GW_Armando_plate2_JG10G03	GW_Armando_plate2_JG10G03	ST	plumb	plumb	87.054
94	GW_Armando_plate2_JG12G02	GW_Armando_plate2_JG12G02	ST	plumb	plumb	87.055
95	GW_Armando_plate2_JG12G03	GW_Armando_plate2_JG12G03	ST	plumb	plumb	87.056
96	GW_Armando_plate2_LN2	GW_Armando_plate2_LN2	LN	troch_LN	troch_LN	58.01
97	GW_Armando_plate2_RF29G01	GW_Armando_plate2_RF29G01	BJ	obs_plumb	plumb_BJ	77.5
98	GW_Armando_plate2_TTGW02	GW_Armando_plate2_TTGW02	MN	troch_MN	troch_west	53.0
99	GW_Armando_plate2_TTGW03	GW_Armando_plate2_TTGW03	MN	troch_MN	troch_west	53.0
100	GW_Armando_plate2_TTGW08	GW_Armando_plate2_TTGW08	SU	lud_Sukhto	lud_central	47.0
101	GW_Armando_plate2_TTGW09	GW_Armando_plate2_TTGW09	SU	lud_Sukhto	lud_central	47.0
102	GW_Armando_plate2_TTGW12	GW_Armando_plate2_TTGW12	TH	lud_Thallighar	lud_central	43
103	GW_Armando_plate2_TTGW14	GW_Armando_plate2_TTGW14	TH	lud_Thallighar	lud_central	43
104	GW_Armando_plate2_TTGW16	GW_Armando_plate2_TTGW16	TH	lud_Thallighar	lud_central	43
105	GW_Armando_plate2_TTGW18	GW_Armando_plate2_TTGW18	TH	lud_Thallighar	lud_central	43
106	GW_Armando_plate2_TTGW20	GW_Armando_plate2_TTGW20	SR	lud_Sural	lud_central	45.0
107	GW_Armando_plate2_TTGW24	GW_Armando_plate2_TTGW24	SR	lud_Sural	lud_central	45.0
108	GW_Armando_plate2_TTGW25	GW_Armando_plate2_TTGW25	SR	lud_Sural	lud_central	45.0
109	GW_Armando_plate2_TTGW27	GW_Armando_plate2_TTGW27	SR	lud_Sural	lud_central	45.0
110	GW_Armando_plate2_TTGW28	GW_Armando_plate2_TTGW28	SR	lud_Sural	lud_central	45.0
111	GW_Armando_plate2_TTGW50	GW_Armando_plate2_TTGW50	NG	lud_Nainaghar	lud_central	49.
112	GW_Armando_plate2_TTGW51	GW_Armando_plate2_TTGW51	NG	lud_Nainaghar	lud_central	49.
113	GW_Armando_plate2_TTGW54	GW_Armando_plate2_TTGW54	NG	lud_Nainaghar	lud_central	49.
114	GW_Armando_plate2_TTGW56	GW_Armando_plate2_TTGW56	NG	lud_Nainaghar	lud_central	49.
115	GW_Armando_plate2_TTGW60	GW_Armando_plate2_TTGW60	SP	lud_Spiti	troch_west	55.0
116	GW_Armando_plate2_TTGW61	GW_Armando_plate2_TTGW61	SP	lud_Spiti	troch_west	55.0
117	GW_Armando_plate2_TTGW62	GW_Armando_plate2_TTGW62	SP	lud_Spiti	troch_west	55.0

118	GW_Armando_plate2_TTGW67	GW_Armando_plate2_TTGW67	SP	lud_Spiti	troch_west	55.0	
119	GW_Armando_plate2_TTGW69	GW_Armando_plate2_TTGW69	SP	lud_Spiti	troch_west	55.0	
120	GW_Armando_plate2_TTGW73	GW_Armando_plate2_TTGW73	SA	lud_Sathrundi	lud_Sath	41.0	
121	GW_Armando_plate2_TTGW75	GW_Armando_plate2_TTGW75	SA	lud_Sathrundi	lud_Sath	41.0	
122	GW_Armando_plate2_TTGW77	GW_Armando_plate2_TTGW77	SA	lud_Sathrundi	lud_Sath	41.0	
123	GW_Armando_plate2_TTGW79	GW_Armando_plate2_TTGW79	SA	lud_Sathrundi	lud_Sath	41.0	
124	GW_Armando_plate2_TTGW80	GW_Armando_plate2_TTGW80	SA	lud_Sathrundi	lud_Sath	41.0	
125	GW_Armando_plate2_TTGW_15_01	GW_Armando_plate2_TTGW_15_01	SR	lud_Sural	lud_central	4	
126	GW_Armando_plate2_TTGW_15_02	GW_Armando_plate2_TTGW_15_02	SR	lud_Sural	lud_central	4	
127	GW_Armando_plate2_TTGW_15_03	GW_Armando_plate2_TTGW_15_03	SR	lud_Sural	lud_central	4	
128	GW_Armando_plate2_TTGW_15_04	GW_Armando_plate2_TTGW_15_04	SR	lud_Sural	lud_central	4	
129	GW_Armando_plate2_TTGW_15_06	GW_Armando_plate2_TTGW_15_06	SR	lud_Sural	lud_central	4	
130	GW_Armando_plate2_TTGW_15_10	GW_Armando_plate2_TTGW_15_10	SR	lud_Sural	lud_central	4	
131	GW_Lane5_AA1	GW_Lane5_AA1	AA	vir_S	vir_S	25.0	71767
132	GW_Lane5_AA11	GW_Lane5_AA11	AA	vir_S	vir_S	34.0	73608
133	GW_Lane5_AA3	GW_Lane5_AA3	AA	vir_S	vir_S	26.0	66255
134	GW_Lane5_AA4	GW_Lane5_AA4	AA	vir_S	vir_S	27.0	66596
135	GW_Lane5_AA5	GW_Lane5_AA5	AA	vir_S	vir_S	28.0	71861
136	GW_Lane5_AA6	GW_Lane5_AA6	AA	vir_S	vir_S	29.0	77882
137	GW_Lane5_AA7	GW_Lane5_AA7	AA	vir_S	vir_S	30.0	72471
138	GW_Lane5_AA8	GW_Lane5_AA8	AA	vir_S	vir_S	31.0	92231
139	GW_Lane5_AA9	GW_Lane5_AA9	AA	vir_S	vir_S	32.0	79151
140	GW_Lane5_AB2	GW_Lane5_AB2	AB	vir	vir	21.0	915622
141	GW_Lane5_AN1	GW_Lane5_AN1	AN	plumb	plumb	80.0	71904
142	GW_Lane5_AN2	GW_Lane5_AN2	AN	plumb	plumb	81.0	68434
143	GW_Lane5_BK2	GW_Lane5_BK2	BK	plumb	plumb	78.0	66601
144	GW_Lane5_BK3	GW_Lane5_BK3	BK	plumb	plumb	79.0	66509
145	GW_Lane5_DA2	GW_Lane5_DA2	XN	obs	obs	73.0	655980
146	GW_Lane5_DA3	GW_Lane5_DA3	XN	obs	obs	74.0	696722
147	GW_Lane5_DA4	GW_Lane5_DA4	XN	obs	obs	75.0	634940
148	GW_Lane5_DA7	GW_Lane5_DA7	XN	obs	obs	77.0	733664
149	GW_Lane5_EM1	GW_Lane5_EM1	EM	troch_EM	troch_EM	72.0	711
150	GW_Lane5_IL1	GW_Lane5_IL1	IL	plumb	plumb	82.0	69803
151	GW_Lane5_IL4	GW_Lane5_IL4	IL	plumb	plumb	84.0	72605
152	GW_Lane5_KS1	GW_Lane5_KS1	OV	lud_KS	lud_KS	40.0	75164
153	GW_Lane5_KS2	GW_Lane5_KS2	OV	lud_KS	lud_KS	40.0	82024
154	GW_Lane5_LN1	GW_Lane5_LN1	LN	troch_LN	troch_LN	57.0	707

155	GW_Lane5_LN10	GW_Lane5_LN10	LN	troch_LN	troch_LN	64.0	783
156	GW_Lane5_LN12	GW_Lane5_LN12	LN	troch_LN	troch_LN	66.0	884
157	GW_Lane5_LN14	GW_Lane5_LN14	LN	troch_LN	troch_LN	67.0	753
158	GW_Lane5_LN16	GW_Lane5_LN16	LN	troch_LN	troch_LN	68.0	738
159	GW_Lane5_LN18	GW_Lane5_LN18	LN	troch_LN	troch_LN	69.0	711
160	GW_Lane5_LN19	GW_Lane5_LN19	LN	troch_LN	troch_LN	70.0	731
161	GW_Lane5_LN20	GW_Lane5_LN20	LN	troch_LN	troch_LN	71.0	738
162	GW_Lane5_LN3	GW_Lane5_LN3	LN	troch_LN	troch_LN	59.0	702
163	GW_Lane5_LN4	GW_Lane5_LN4	LN	troch_LN	troch_LN	60.0	683
164	GW_Lane5_LN6	GW_Lane5_LN6	LN	troch_LN	troch_LN	61.0	690
165	GW_Lane5_LN7	GW_Lane5_LN7	LN	troch_LN	troch_LN	62.0	758
166	GW_Lane5_LN8	GW_Lane5_LN8	LN	troch_LN	troch_LN	63.0	662
167	GW_Lane5_MN1	GW_Lane5_MN1	MN	troch_MN	troch_west	51.0	943
168	GW_Lane5_MN12	GW_Lane5_MN12	MN	troch_MN	troch_west	56.0	67
169	GW_Lane5_MN3	GW_Lane5_MN3	MN	troch_MN	troch_west	52.0	946
170	GW_Lane5_MN5	GW_Lane5_MN5	MN	troch_MN	troch_west	53.0	752
171	GW_Lane5_MN8	GW_Lane5_MN8	MN	troch_MN	troch_west	54.0	771
172	GW_Lane5_MN9	GW_Lane5_MN9	MN	troch_MN	troch_west	55.0	897
173	GW_Lane5_NA1	GW_Lane5_NA1	NR	lud_PK	lud_PK	39.2	91921
174	GW_Lane5_NA3-3ul	GW_Lane5_NA3-3ul	NR	lud_PK	lud_PK	39.2	79
175	GW_Lane5_PT11	GW_Lane5_PT11	KL	lud_KL	lud_central	42.0	77
176	GW_Lane5_PT12	GW_Lane5_PT12	KL	lud_KL	lud_central	42.0	79
177	GW_Lane5_PT2	GW_Lane5_PT2	ML	lud_ML	lud_ML	51.0	76034
178	GW_Lane5_PT3	GW_Lane5_PT3	PA	lud_PA	lud_central	46.0	722
179	GW_Lane5_PT4	GW_Lane5_PT4	PA	lud_PA	lud_central	46.0	705
180	GW_Lane5_PT6	GW_Lane5_PT6	KL	lud_KL	lud_central	42.0	763
181	GW_Lane5_SH1	GW_Lane5_SH1	PK	lud_PK	lud_PK	39.1	96670
182	GW_Lane5_SH2	GW_Lane5_SH2	PK	lud_PK	lud_PK	39.1	76864
183	GW_Lane5_SH4	GW_Lane5_SH4	PK	lud_PK	lud_PK	39.1	84964
184	GW_Lane5_SH5	GW_Lane5_SH5	PK	lud_PK	lud_PK	39.1	92939
185	GW_Lane5_SL1	GW_Lane5_SL1	SL	plumb	plumb	95.0	64888
186	GW_Lane5_SL2	GW_Lane5_SL2	SL	plumb	plumb	96.0	65473
187	GW_Lane5_ST1	GW_Lane5_ST1	ST	plumb	plumb	85.0	60624
188	GW_Lane5_ST12	GW_Lane5_ST12	ST	plumb	plumb	87.0	69120
189	GW_Lane5_ST3	GW_Lane5_ST3	ST	plumb	plumb	86.0	69699
190	GW_Lane5_STvi1	GW_Lane5_STvi1	STvi	vir	vir	22.0	69009
191	GW_Lane5_STvi2	GW_Lane5_STvi2	STvi	vir	vir	23.0	89733

192	GW_Lane5_STvi3	GW_Lane5_STvi3	STvi	vir	vir	24.0	76816
193	GW_Lane5_TA1	GW_Lane5_TA1	TA	plumb	plumb	94.0	71190
194	GW_Lane5_TL1	GW_Lane5_TL1	TL	vir	vir	9.0	743509
195	GW_Lane5_TL10	GW_Lane5_TL10	TL	vir	vir	17.0	669934
196	GW_Lane5_TL11	GW_Lane5_TL11	TL	vir	vir	18.0	638402
197	GW_Lane5_TL12	GW_Lane5_TL12	TL	vir	vir	19.0	585697
198	GW_Lane5_TL2	GW_Lane5_TL2	TL	vir	vir	10.0	770857
199	GW_Lane5_TL4	GW_Lane5_TL4	TL	vir	vir	12.0	758037
200	GW_Lane5_TL5	GW_Lane5_TL5	TL	vir	vir	13.0	867165
201	GW_Lane5_TL7	GW_Lane5_TL7	TL	vir	vir	14.0	803407
202	GW_Lane5_TL8	GW_Lane5_TL8	TL	vir	vir	15.0	698745
203	GW_Lane5_TL9	GW_Lane5_TL9	TL	vir	vir	16.0	606969
204	GW_Lane5_TU1	GW_Lane5_TU1	TU	nit	nit	35.0	793640
205	GW_Lane5_TU2	GW_Lane5_TU2	TU	nit	nit	36.0	736785
206	GW_Lane5_UY2	GW_Lane5_UY2	UY	plumb	plumb	89.0	72900
207	GW_Lane5_UY3	GW_Lane5_UY3	UY	plumb	plumb	90.0	67752
208	GW_Lane5_UY4	GW_Lane5_UY4	UY	plumb	plumb	91.0	74984
209	GW_Lane5_UY5	GW_Lane5_UY5	UY	plumb	plumb	92.0	71837
210	GW_Lane5_UY6	GW_Lane5_UY6	UY	plumb	plumb	93.0	71367
211	GW_Lane5_YK1	GW_Lane5_YK1	YK	vir	vir	1.0	831245
212	GW_Lane5_YK11	GW_Lane5_YK11	YK	vir	vir	8.0	730798
213	GW_Lane5_YK3	GW_Lane5_YK3	YK	vir	vir	2.0	731944
214	GW_Lane5_YK4	GW_Lane5_YK4	YK	vir	vir	3.0	740051
215	GW_Lane5_YK5	GW_Lane5_YK5	YK	vir	vir	4.0	738740
216	GW_Lane5_YK6	GW_Lane5_YK6	YK	vir	vir	5.0	697420
217	GW_Lane5_YK7	GW_Lane5_YK7	YK	vir	vir	6.0	692052
218	GW_Lane5_YK9	GW_Lane5_YK9	YK	vir	vir	7.0	768722
219	GW_Liz_GBS_Liz10045	GW_Liz_GBS_Liz10045	ML	lud	lud_ML	51.01	8
220	GW_Liz_GBS_Liz10094	GW_Liz_GBS_Liz10094	ML	lud	lud_ML	51.02	9
221	GW_Liz_GBS_Liz5144	GW_Liz_GBS_Liz5144	ML	lud	lud_ML	51.08	9
222	GW_Liz_GBS_Liz5163	GW_Liz_GBS_Liz5163	ML	lud_chick	lud_ML	51.12	
223	GW_Liz_GBS_Liz5164	GW_Liz_GBS_Liz5164	ML	lud_chick	lud_ML	51.13	
224	GW_Liz_GBS_Liz5165	GW_Liz_GBS_Liz5165	ML	lud	lud_ML	51.14	9
225	GW_Liz_GBS_Liz5167	GW_Liz_GBS_Liz5167	ML	lud_chick	lud_ML	51.15	
226	GW_Liz_GBS_Liz5168	GW_Liz_GBS_Liz5168	ML	lud_chick	lud_ML	51.16	
227	GW_Liz_GBS_Liz5173	GW_Liz_GBS_Liz5173	ML	lud_chick	lud_ML	51.2	
228	GW_Liz_GBS_Liz5175	GW_Liz_GBS_Liz5175	ML	lud	lud_ML	51.22	9

229	GW_Liz_GBS_Liz5178	GW_Liz_GBS_Liz5178	ML	lud_chick	lud_ML	51.25	
230	GW_Liz_GBS_Liz5179	GW_Liz_GBS_Liz5179	ML	lud_chick	lud_ML	51.26	
231	GW_Liz_GBS_Liz5182	GW_Liz_GBS_Liz5182	ML	lud_chick	lud_ML	51.28	
232	GW_Liz_GBS_Liz5184	GW_Liz_GBS_Liz5184	ML	lud_chick	lud_ML	51.29	
233	GW_Liz_GBS_Liz5185	GW_Liz_GBS_Liz5185	ML	lud	lud_ML	51.3	94
234	GW_Liz_GBS_Liz5188	GW_Liz_GBS_Liz5188	ML	lud	lud_ML	51.33	8
235	GW_Liz_GBS_Liz5189	GW_Liz_GBS_Liz5189	ML	lud_chick	lud_ML	51.34	
236	GW_Liz_GBS_Liz5190	GW_Liz_GBS_Liz5190	ML	lud_chick	lud_ML	51.35	
237	GW_Liz_GBS_Liz5191	GW_Liz_GBS_Liz5191	ML	lud_chick	lud_ML	51.36	
238	GW_Liz_GBS_Liz5193	GW_Liz_GBS_Liz5193	ML	lud_chick	lud_ML	51.38	
239	GW_Liz_GBS_Liz5194	GW_Liz_GBS_Liz5194	ML	lud_chick	lud_ML	51.39	
240	GW_Liz_GBS_Liz5197	GW_Liz_GBS_Liz5197	ML	lud	lud_ML	51.41	8
241	GW_Liz_GBS_Liz5199	GW_Liz_GBS_Liz5199	ML	lud_chick	lud_ML	51.42	
242	GW_Liz_GBS_Liz6002	GW_Liz_GBS_Liz6002	ML	lud	lud_ML	51.43	9
243	GW_Liz_GBS_Liz6006	GW_Liz_GBS_Liz6006	ML	lud	lud_ML	51.44	8
244	GW_Liz_GBS_Liz6008	GW_Liz_GBS_Liz6008	ML	lud	lud_ML	51.45	9
245	GW_Liz_GBS_Liz6009	GW_Liz_GBS_Liz6009	ML	lud	lud_ML	51.46	9
246	GW_Liz_GBS_Liz6010	GW_Liz_GBS_Liz6010	ML	lud	lud_ML	51.47	8
247	GW_Liz_GBS_Liz6014	GW_Liz_GBS_Liz6014	ML	lud	lud_ML	51.49	8
248	GW_Liz_GBS_Liz6055	GW_Liz_GBS_Liz6055	ML	lud	lud_ML	51.5	87
249	GW_Liz_GBS_Liz6057	GW_Liz_GBS_Liz6057	ML	lud	lud_ML	51.51	8
250	GW_Liz_GBS_Liz6060	GW_Liz_GBS_Liz6060	ML	lud	lud_ML	51.52	8
251	GW_Liz_GBS_Liz6062	GW_Liz_GBS_Liz6062	ML	lud	lud_ML	51.53	8
252	GW_Liz_GBS_Liz6063	GW_Liz_GBS_Liz6063	ML	lud	lud_ML	51.54	8
253	GW_Liz_GBS_Liz6066	GW_Liz_GBS_Liz6066	ML	lud	lud_ML	51.55	8
254	GW_Liz_GBS_Liz6072	GW_Liz_GBS_Liz6072	ML	lud	lud_ML	51.56	8
255	GW_Liz_GBS_Liz6079	GW_Liz_GBS_Liz6079	ML	lud	lud_ML	51.57	8
256	GW_Liz_GBS_Liz6204	GW_Liz_GBS_Liz6204	ML	lud_chick	lud_ML	51.59	
257	GW_Liz_GBS_Liz6461	GW_Liz_GBS_Liz6461	ML	lud	lud_ML	51.6	81
258	GW_Liz_GBS_Liz6472	GW_Liz_GBS_Liz6472	ML	lud	lud_ML	51.61	9
259	GW_Liz_GBS_Liz6478	GW_Liz_GBS_Liz6478	ML	lud	lud_ML	51.62	7
260	GW_Liz_GBS_Liz6776	GW_Liz_GBS_Liz6776	ML	lud	lud_ML	51.64	9
261	GW_Liz_GBS_Liz6794	GW_Liz_GBS_Liz6794	ML	lud	lud_ML	51.65	8

Filter SNPs with too many missing genotypes:

```
# (remember that first column is arbitrary row number in input file)
missing_genotypes_per_SNP = sum(geno_indFiltered .== -1, dims=1)
missing_genotypes_percent_allowed_per_site = 5   # this is the percentage threshold
threshold_genotypes_missing = size(geno_indFiltered)[1] * missing_genotypes_percent_allowed_per_site/
selection = vec(missing_genotypes_per_SNP .<= threshold_genotypes_missing)
geno_ind_SNP_filtered = geno_indFiltered[:, selection]
pos_SNP_filtered = pos_whole_genome[selection[Not(1)],:]  # the Not(1) is needed because first columm
println("Started with ", size(geno_indFiltered, 2)-1, " SNPs.
After filtering SNPs for no more than ", missing_genotypes_percent_allowed_per_site, "% missing genot
```

```
Started with 2431709 SNPs.
After filtering SNPs for no more than 5% missing genotypes, 1017581 SNPs remain.
```

2nd round of filtering individuals

I added this in August 2023, to improve accuracy of imputation-based PCA, because I noticed outliers tended to have more missing data. Now I only allow up to 10% missing SNPs per individual.

```
SNPmissing_percent_allowed_per_ind_round2 = 10   # this is the percentage threshold
threshold_missing = (size(geno_ind_SNP_filtered, 2) - 1) * SNPmissing_percent_allowed_per_ind_round2/
numMissings = sum(geno_ind_SNP_filtered .== -1, dims=2)
selection = vec(numMissings .<= threshold_missing) # the vec command converts to BitVector rather tha
geno_ind_SNP_ind_filtered = geno_ind_SNP_filtered[selection, :]
println("Filtering out these individuals based on too many missing genotypes: ")
filtered_inds = ind_with_metadata_indFiltered.ind[selection.==false]
println(DataFrame(filtered_inds = filtered_inds)) # did this to print all lines
ind_with_metadata_indFiltered = ind_with_metadata_indFiltered[selection, :]
println("This leaves ", size(geno_ind_SNP_ind_filtered, 1), " individuals and ", size(geno_ind_SNP_in
with no individuals missing more than ", SNPmissing_percent_allowed_per_ind_round2, "% of genotypes
and no loci missing in more than ", missing_genotypes_percent_allowed_per_site, "% of individuals.")
```

```
Filtering out these individuals based on too many missing genotypes:
4×1 DataFrame
 Row │ filtered_inds
     │ String
─────┼──────────────────────────
   1 │ GW_Armando_plate1_TTGW74
   2 │ GW_Armando_plate2_TTGW54
   3 │ GW_Lane5_AA8
```

```
   4 │ GW_Lane5_YK1
This leaves 257 individuals and 1017581 loci,
with no individuals missing more than 10% of genotypes
and no loci missing in more than 5% of individuals.
```

Estimate relationships of individuals using PCA

Our goal is to produce plots showing individuals in genotype space, using Principal Components Analysis. First we need to do a couple changes to our data matrix:

Remove the first column of the genotype matrix (which was an initial row number):

```
genosOnly = geno_ind_SNP_ind_filtered[:, Not(1)]
```

```
257×1017581 Matrix{Int16}:
 0  0  0  0  0  1  0   0  0  0  0  0  …  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0  …  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   1  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0  …  0  0  0  0  0  0  0  0  0  0  0  1
 0  0  0  0  0  0  0   0  0  0  0  1     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 ⋮                 ⋮                 ⋮     ⋱     ⋮                 ⋮           ⋮
 0  0  0  0  0  0  0   0  0  0  0  0  …  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  1  0     1  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  -1  0  0  0  0  …  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0  …  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0   0  0  0  0  0     0  0  0  0  0  0  0  0  0  0  0  0
```

For missing genotypes, change our code of `-1` to `missing`:

```
genosOnly_with_missing = Matrix{Union{Missing, Int16}}(genosOnly)
genosOnly_with_missing[genosOnly_with_missing .== -1] .= missing;
```

Impute and save genotypes for each scaffold

PCA requires imputation of missing genotypes. I did imputation for each scaffold above a certain size threshold. Those scaffolds (many of which correspond to whole chromosomes) are listed here:

```
chromosomes_to_process = vec(["gw2",
                              "gw1",
                              "gw3",
                              "gwZ",
                              "gw1A",
                              "gw4",
                              "gw5",
                              "gw7",
                              "gw6",
                              "gw8",
                              "gw9",
                              "gw11",
                              "gw12",
                              "gw10",
                              "gw13",
                              "gw14",
                              "gw18",
                              "gw20",
                              "gw15",
                              "gw1B",
                              "gws100",
                              "gw17",
                              "gw19",
                              "gws101",
                              "gw4A",
                              "gw21",
                              "gw26",
                              "gws102",
                              "gw23",
                              "gw25",
```

```
                                "gws103",
                                "gw22",
                                "gws104",
                                "gw28",
                                "gw27",
                                "gw24",
                                "gws105",
                                "gws106",
                                "gws107",
                                "gws108",
                                "gws109",
                                "gws110",
                                "gws112"]);
```

Imputation can take several minutes per scaffold, so I ran this imputation step separately from this Quarto notebook (otherwise render would take long) and saved the genotype data for each scaffold for loading in the next step. This is the code I used for imputing:

```
for i in eachindex(chromosomes_to_process)
    chrom = chromosomes_to_process[i]
    regionText = string("chr", chrom)
    loci_selection = (pos_SNP_filtered.chrom .== chrom)
    pos_SNP_filtered_region = pos_SNP_filtered[loci_selection,:]
    genosOnly_region_for_imputing = Matrix{Union{Missing, Float32}}(genosOnly_with_missing[:,loci_se
    @time imputed_genos = Impute.svd(genosOnly_region_for_imputing)
    filename = string(baseName, tagName, regionText, ".imputedMissing.jld2")
    jldsave(filename; imputed_genos, ind_with_metadata_indFiltered, pos_SNP_filtered_region)
    println(string("Chromosome ", chrom, ": Saved real and imputed genotypes for ", size(pos_SNP_fil
end
```

Now we can cycle through a set of chromosomes and plot a PCA for each. We need to first specify some groups to include in the plot, and their colors:

```
groups_to_plot_PCA = ["vir","vir_misID","vir_S","nit", "lud_PK", "lud_KS", "lud_central", "lud_Sath"
group_colors_PCA = ["blue","blue","turquoise1","grey","seagreen4","seagreen3","seagreen2","olivedrab3
```

Now we'll actually do the PCA and make the plot for each scaffold. The code block below is what I initially used, based on the SVD method of imputing. I've now decided that KNN is better (see further below).

```
for i in eachindex(chromosomes_to_process)
    chrom = chromosomes_to_process[i]
    regionText = string("chr", chrom)
    filename = string(baseName, tagName, regionText, ".imputedMissing.jld2")
    imputed_genos = load(filename, "imputed_genos")
    ind_with_metadata_indFiltered = load(filename, "ind_with_metadata_indFiltered")
    pos_SNP_filtered_region = load(filename, "pos_SNP_filtered_region")
    println(string("Loaded ", filename))
    println(string(regionText, ": ", size(imputed_genos, 2), " SNPs from ", size(imputed_genos, 1), "
    flipPC1 = true
    flipPC2 = true
    PCAmodel = plotPCA(imputed_genos, ind_with_metadata_indFiltered,
        groups_to_plot_PCA, group_colors_PCA;
        sampleSet="greenish warblers", regionText=regionText,
        flip1=flipPC1, flip2=flipPC2,
        showPlot=false)
    # add position of reference genome
    refGenomePCAposition = predict(PCAmodel.model, zeros(size(imputed_genos, 2)))
    flipPC1 && (refGenomePCAposition[1] *= -1)  # this flips PC1 if flipPC1 = true
    flipPC2 && (refGenomePCAposition[2] *= -1)  # same for PC2
    CairoMakie.scatter!(refGenomePCAposition[1], refGenomePCAposition[2], marker=:diamond, color="bla
    try
        display(PCAmodel.PCAfig)
    catch
        println("NOTICE: Figure for ", regionText, " could not be shown due to an unknown error.")
    end
end
```

Imputation using KNN

Troyanskaya *et al.* (2001) recommend imputation using K-nearest neighbors approach as being better than SVD, both of which are better than other methods for DNA genotyping. They also recommend using Euclidian distance. So I will try KNN with Euclidian distance, which like SVD is provided by Impute.jl. I am going with the default of setting `dims` to `:rows`, as that seems to run much faster and produces PCAs that make a lot of sense. I've already run this next code cell, which does the imputing and saves the imputed data matrix for each scaffold:

```
for i in eachindex(chromosomes_to_process)
    chrom = chromosomes_to_process[i]
    regionText = string("chr", chrom)
    loci_selection = (pos_SNP_filtered.chrom .== chrom)
```

```
    pos_SNP_filtered_region = pos_SNP_filtered[loci_selection,:]
    genosOnly_region_for_imputing = Matrix{Union{Missing, Float32}}(genosOnly_with_missing[:,loci_se
    @time imputed_genos = Impute.knn(genosOnly_region_for_imputing; k=1, dims=:rows)
    filename = string(baseName, tagName, regionText, ".KNNimputedMissing.jld2")
    jldsave(filename; imputed_genos, ind_with_metadata_indFiltered, pos_SNP_filtered_region)
    println(string("Chromosome ", chrom, ": Saved real and imputed genotypes for ", size(pos_SNP_fil
end
```

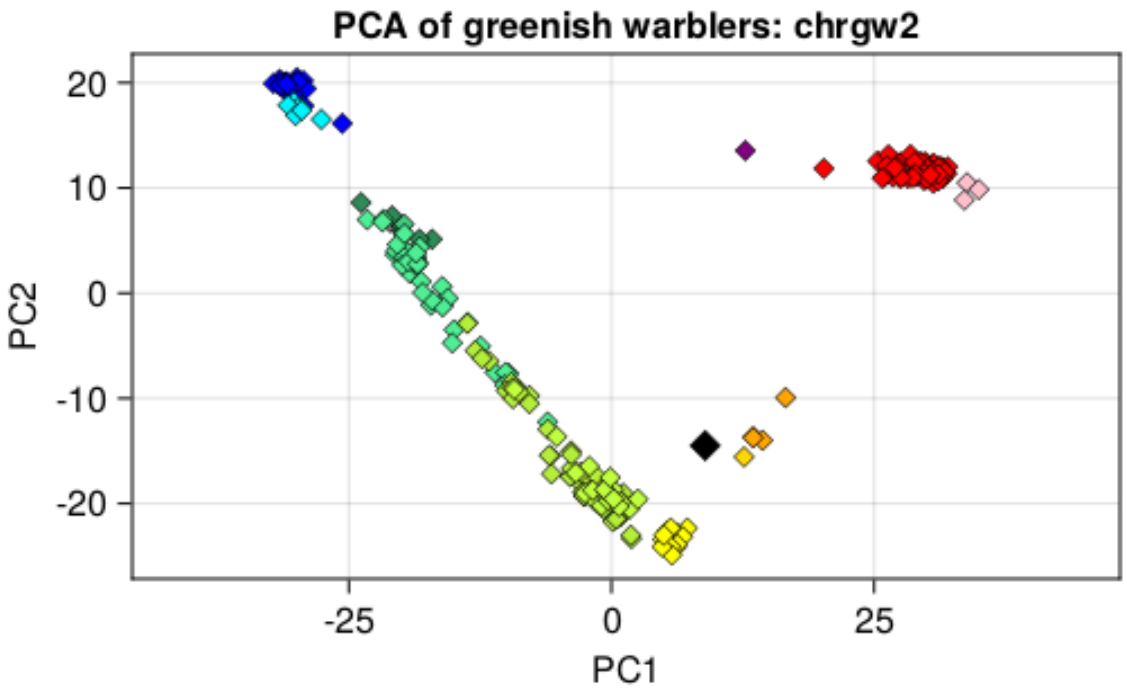
Now do the KNN PCA:

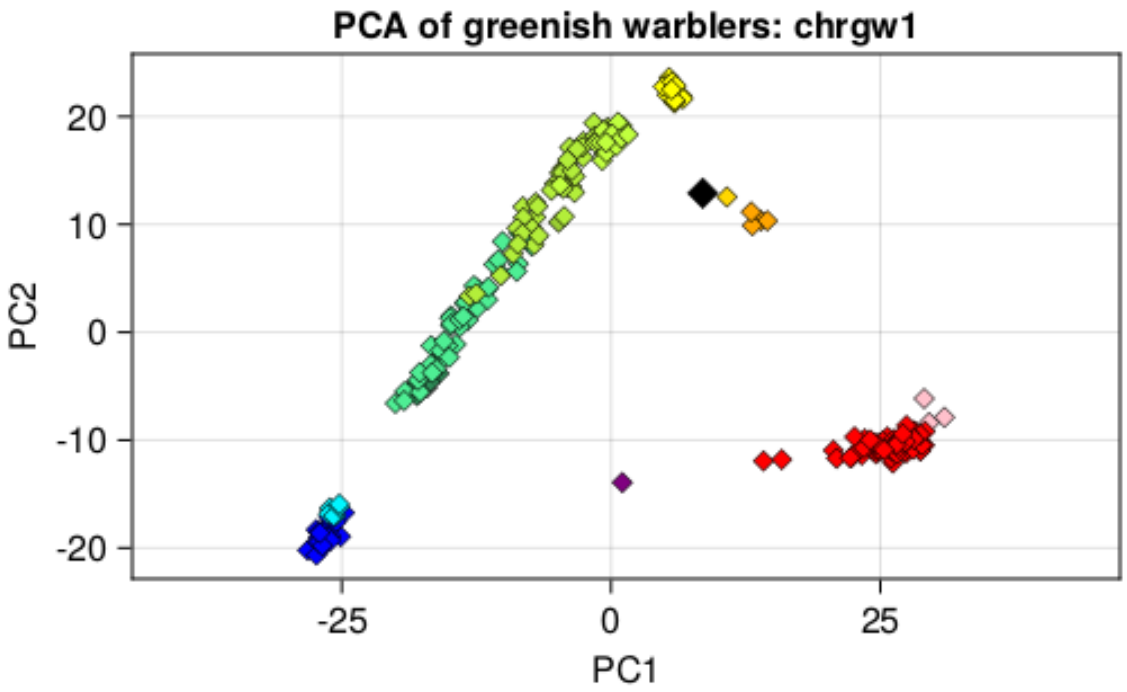
```
for i in eachindex(chromosomes_to_process)
    chrom = chromosomes_to_process[i]
    regionText = string("chr", chrom)
    filename = string(baseName, tagName, regionText, ".KNNimputedMissing.jld2")
    imputed_genos = load(filename, "imputed_genos")
    ind_with_metadata_indFiltered = load(filename, "ind_with_metadata_indFiltered")
    pos_SNP_filtered_region = load(filename, "pos_SNP_filtered_region")
    println(string("Loaded ",filename))
    println(string(regionText, ": ", size(imputed_genos,2), " SNPs from ", size(imputed_genos,1), " i
    flipPC1 = true
    flipPC2 = true
    PCAmodel = plotPCA(imputed_genos, ind_with_metadata_indFiltered,
            groups_to_plot_PCA, group_colors_PCA;
            sampleSet = "greenish warblers", regionText=regionText,
            flip1 = true, flip2 = true,
            showPlot = false)
    # add position of reference genome
    refGenomePCAposition = predict(PCAmodel.model, zeros(size(imputed_genos, 2)))
    flipPC1 && (refGenomePCAposition[1] *= -1)  # this flips PC1 if flipPC1 = true
    flipPC2 && (refGenomePCAposition[2] *= -1)  # same for PC2
    CairoMakie.scatter!(refGenomePCAposition[1], refGenomePCAposition[2], marker = :diamond, color="
    try
        display(PCAmodel.PCAfig)
    catch
        println("NOTICE: Figure for ", regionText, " could not be shown due to an unknown error.")
    end
end
```

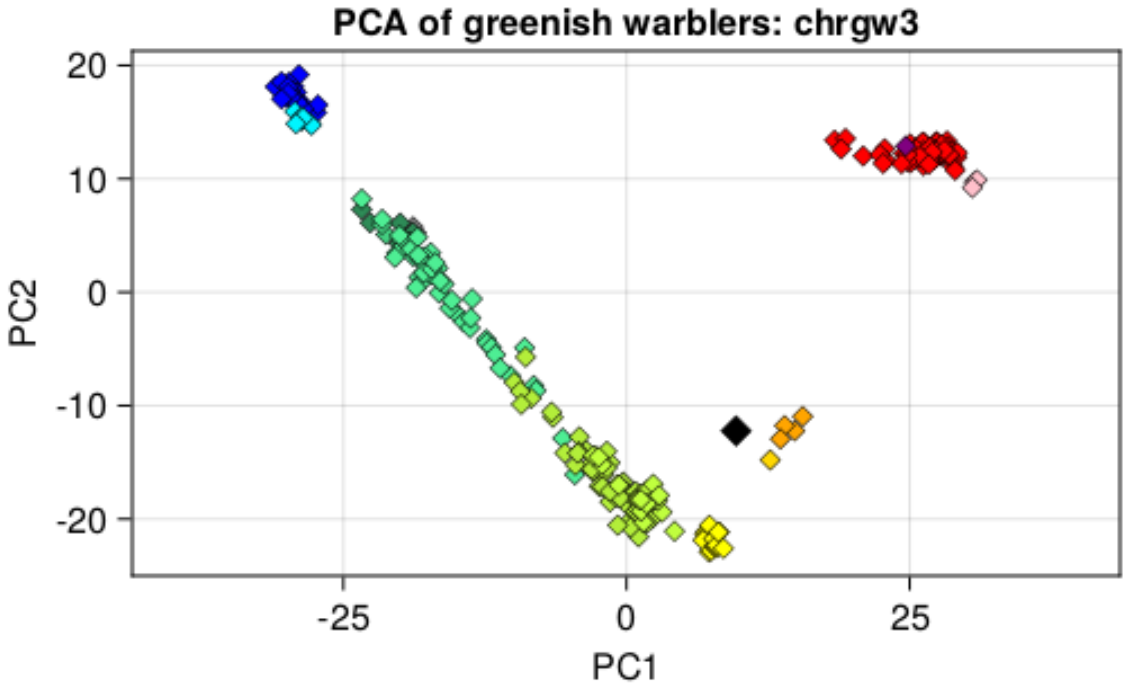
```
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.wh
chrgw2: 93292 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.wh
chrgw1: 80862 SNPs from 257 individuals
```

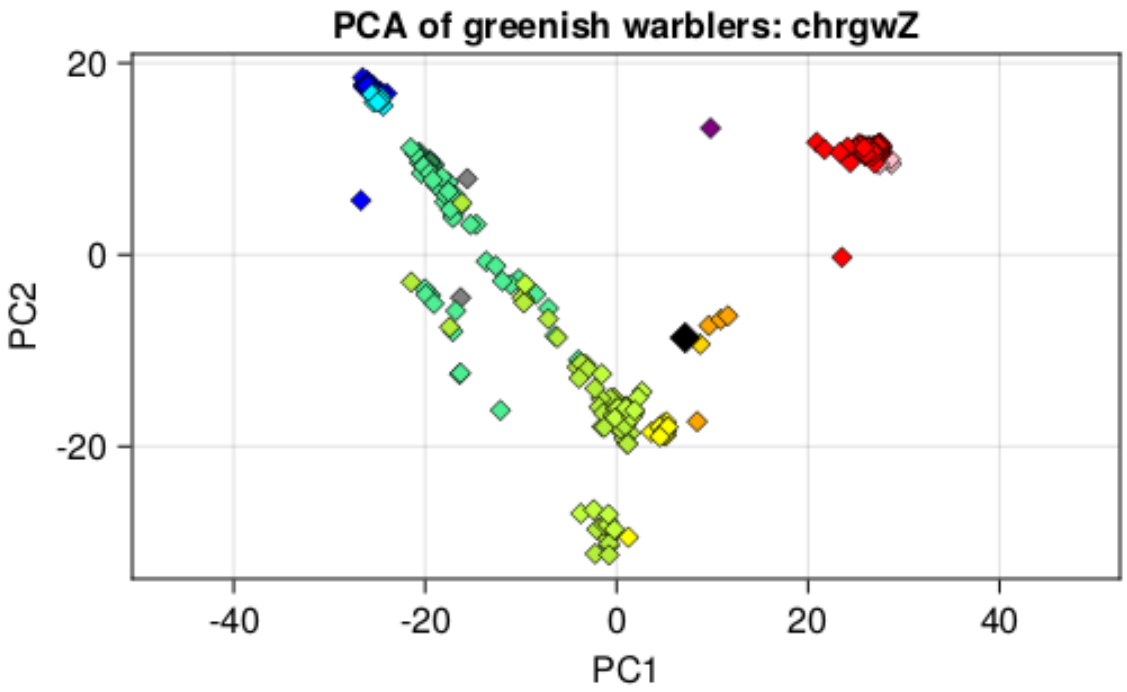
```
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw3: 82372 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgwZ: 53336 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw1A: 50051 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw4: 49980 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw5: 55329 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw7: 36575 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw6: 40175 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw8: 37818 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw9: 38180 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw11: 27683 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw12: 33294 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw10: 26962 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw13: 33543 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw14: 30969 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw18: 19359 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw20: 32739 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw15: 27517 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw1B: 638 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws100: 208 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw17: 26313 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw19: 25414 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
```

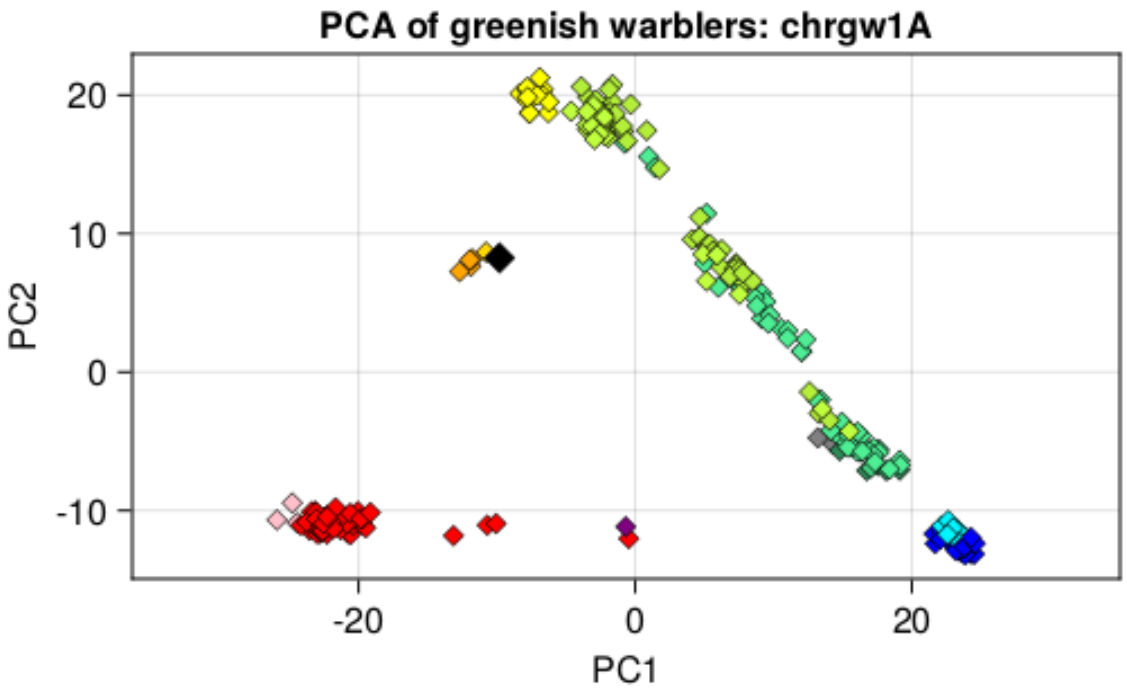
```
chrgws101: 158 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw4A: 18467 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw21: 13321 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw26: 14303 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws102: 302 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw23: 13949 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw25: 3794 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws103: 322 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw22: 5473 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws104: 369 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw28: 11180 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw27: 9684 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgw24: 13821 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws105: 475 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws106: 115 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws107: 260 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws108: 160 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws109: 310 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws110: 175 SNPs from 257 individuals
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgws112: 1884 SNPs from 257 individuals
```

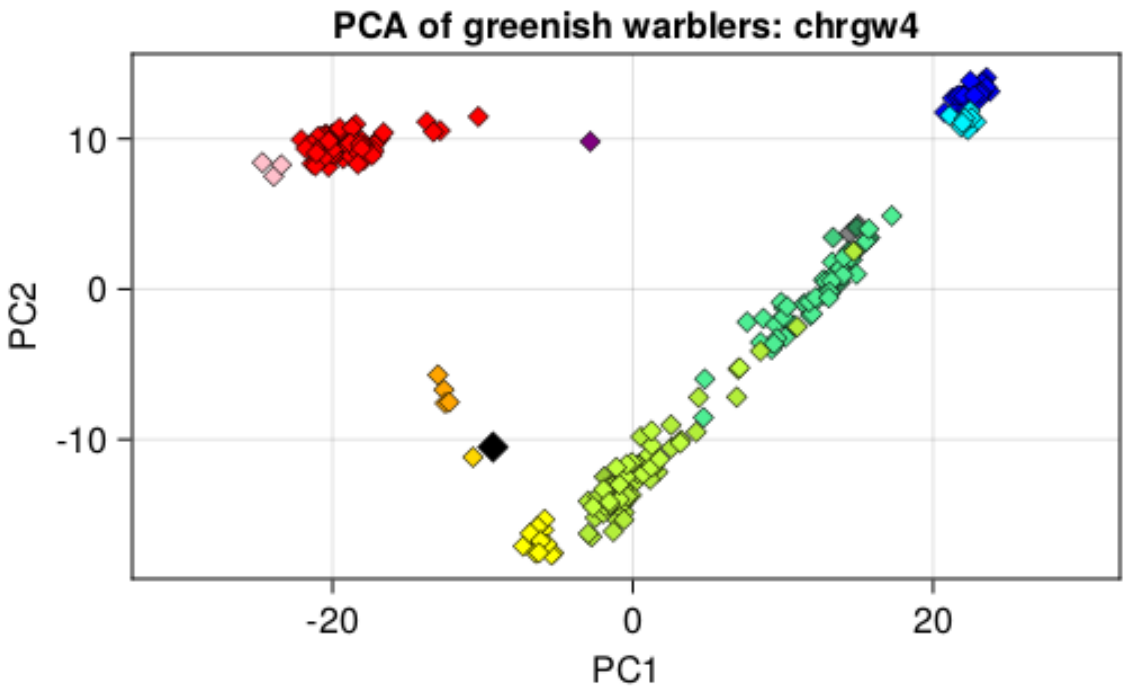

PCA of greenish warblers: chrgw2
20
10
0
-10
-20
PC2
-25
0
25
PC1

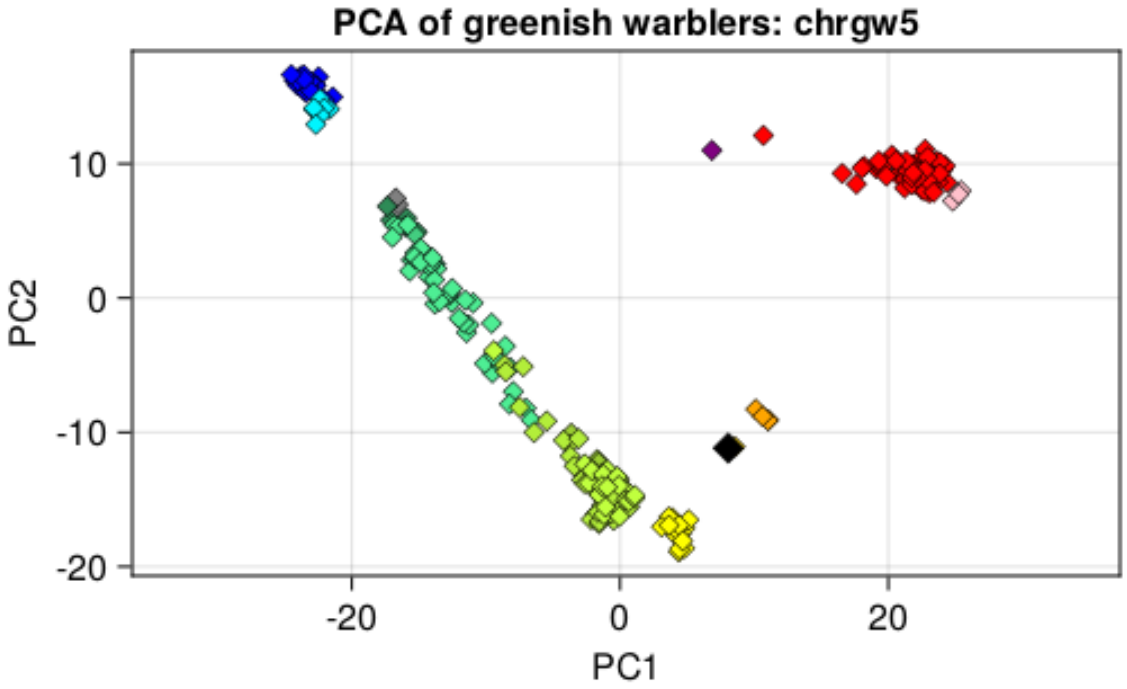

PCA of greenish warblers: chrgw1
PC2
20
10
0
-10
-20
-25
0
25
PC1

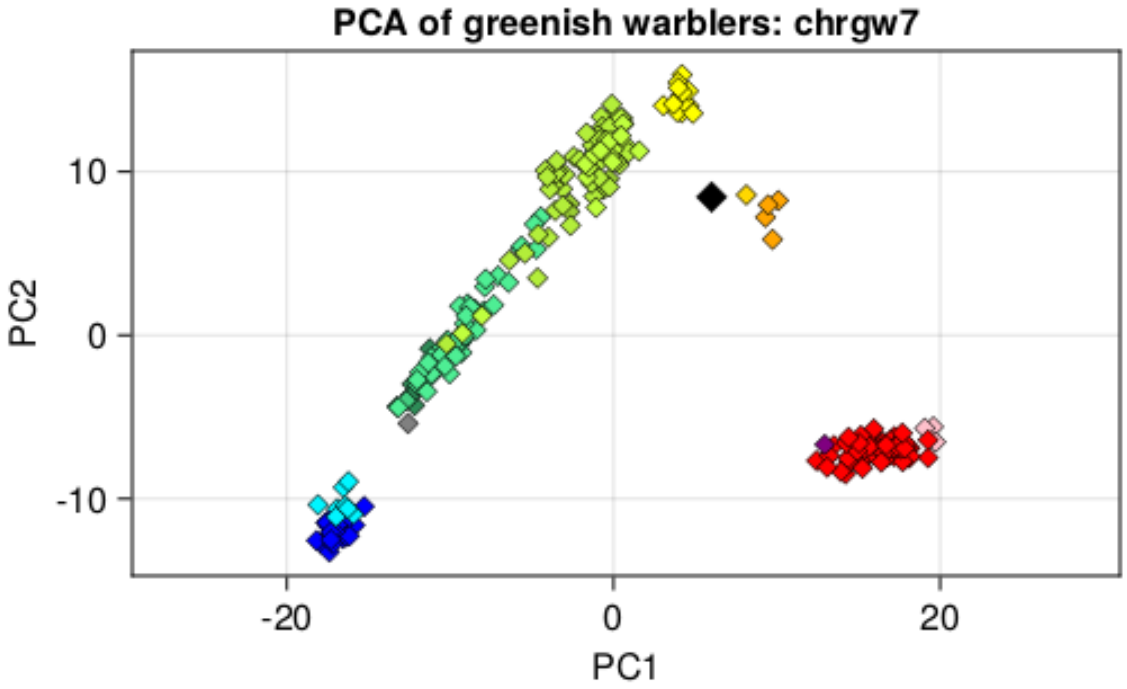

PCA of greenish warblers: chrgw3
20
10
0
-10
-20
PC2
-25
0
25
PC1

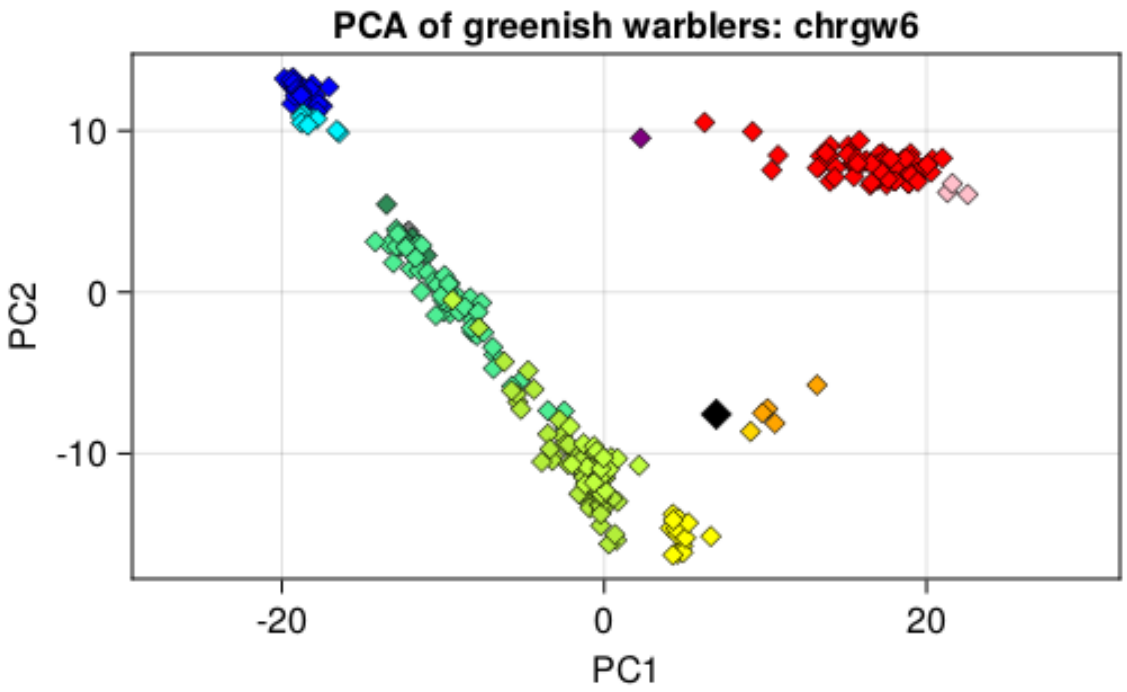

PCA of greenish warblers: chrgwZ
20
0
-20
PC2
-40
-20
0
20
40
PC1

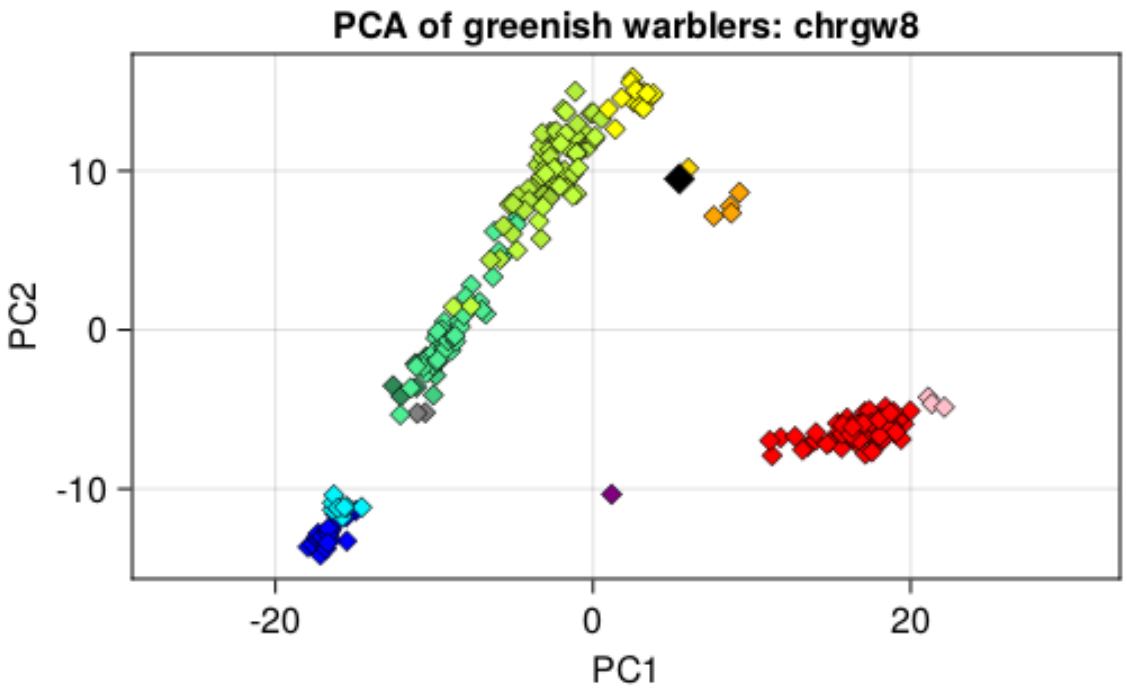

PCA of greenish warblers: chrgw1A
20
10
0
-10
PC2
-20
0
20
PC1

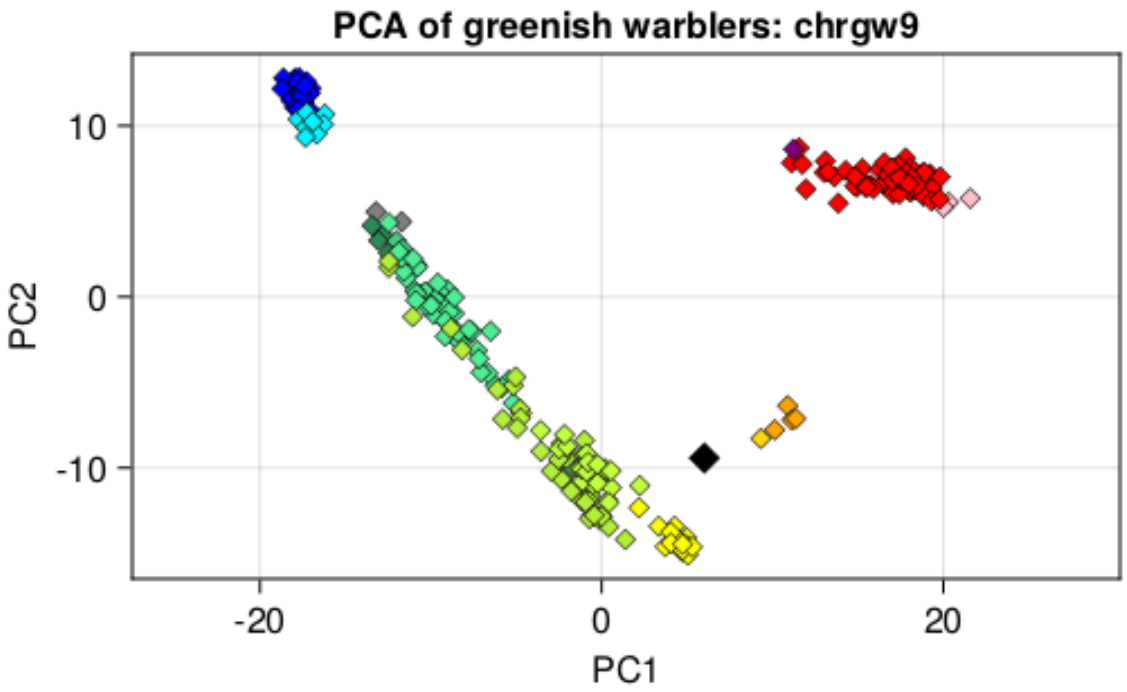

PCA of greenish warblers: chrgw4
10
0
-10
PC2
-20
0
20
PC1

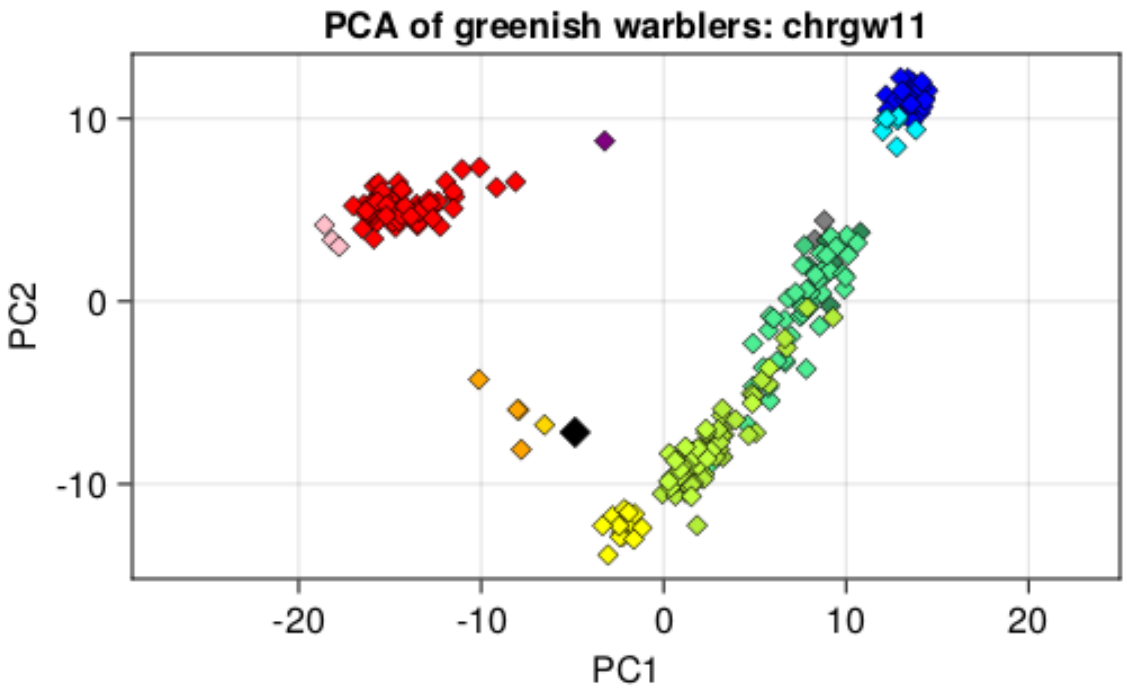

PCA of greenish warblers: chrgw5
PC2
10
0
-10
-20
-20
0
20
PC1

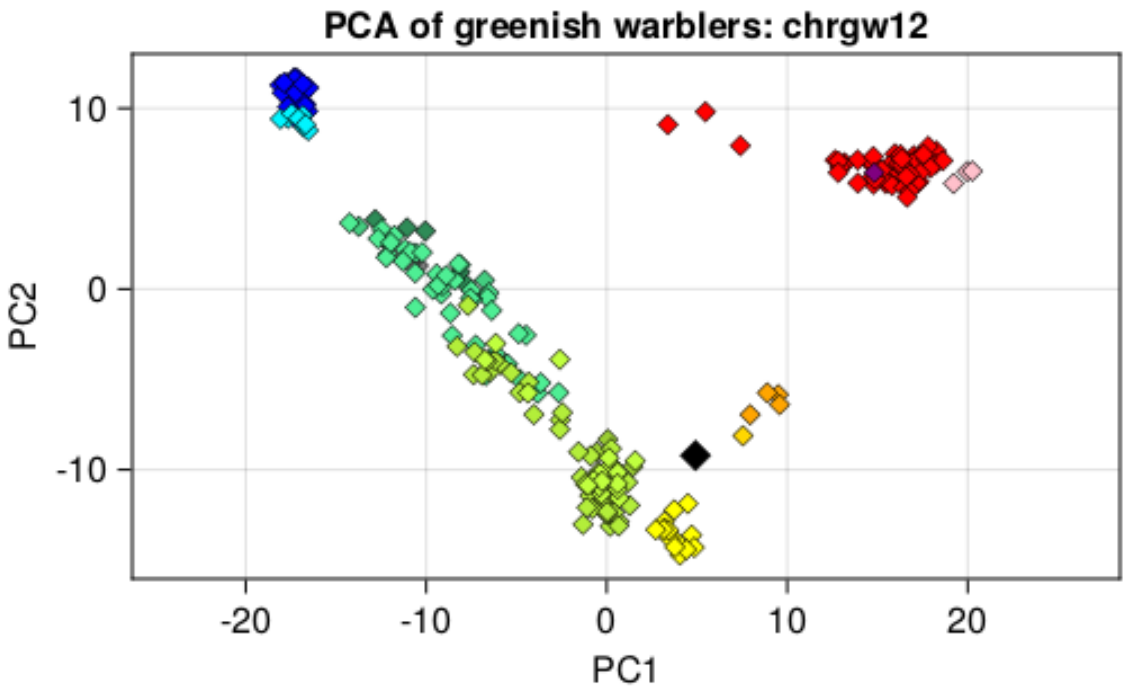

PCA of greenish warblers: chrgw7
10
0
-10
PC2
-20
0
20
PC1

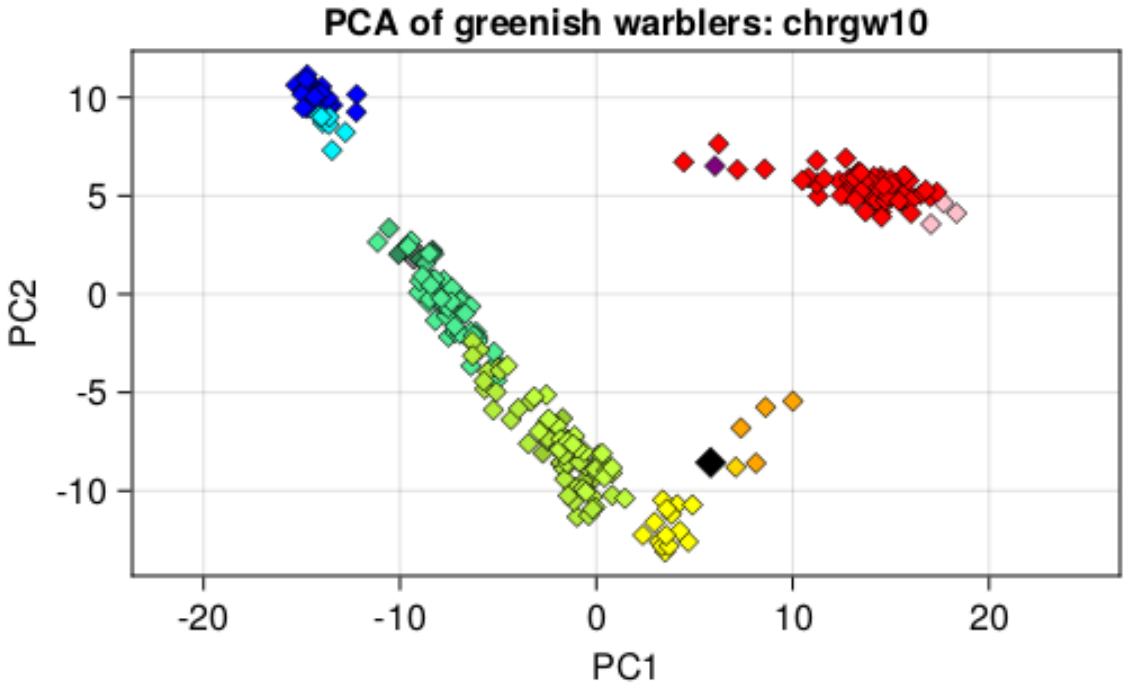

PCA of greenish warblers: chrgw6
10
0
-10
PC2
-20
0
20
PC1

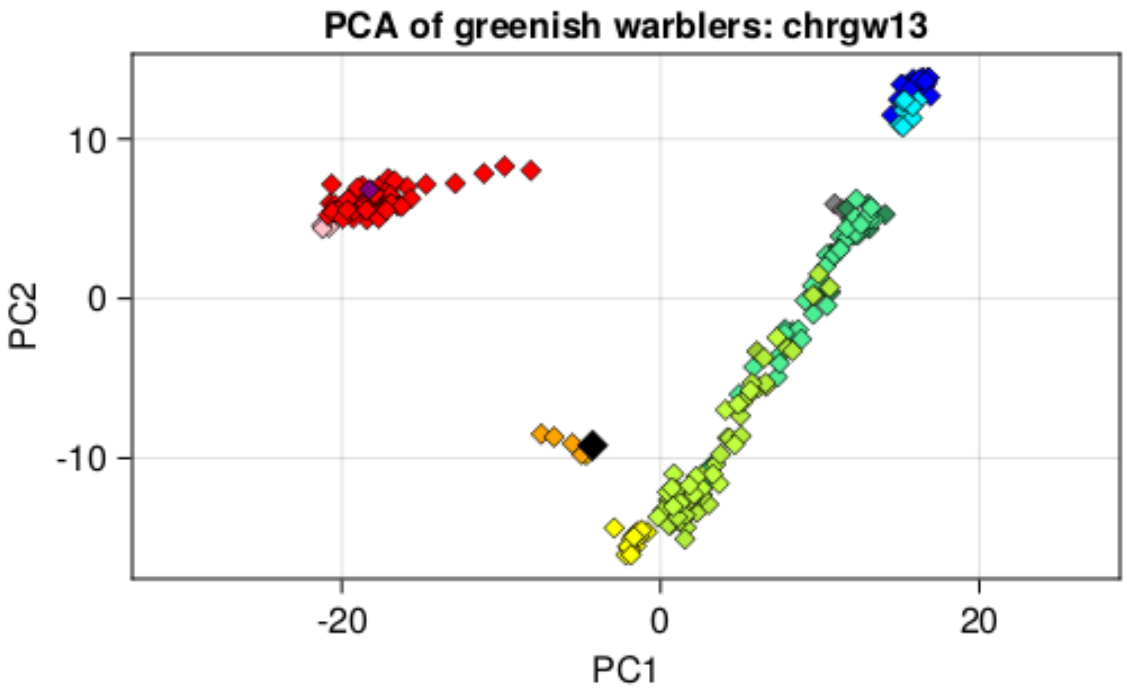

PCA of greenish warblers: chrgw8
PC2
10
0
-10
-20
0
20
PC1

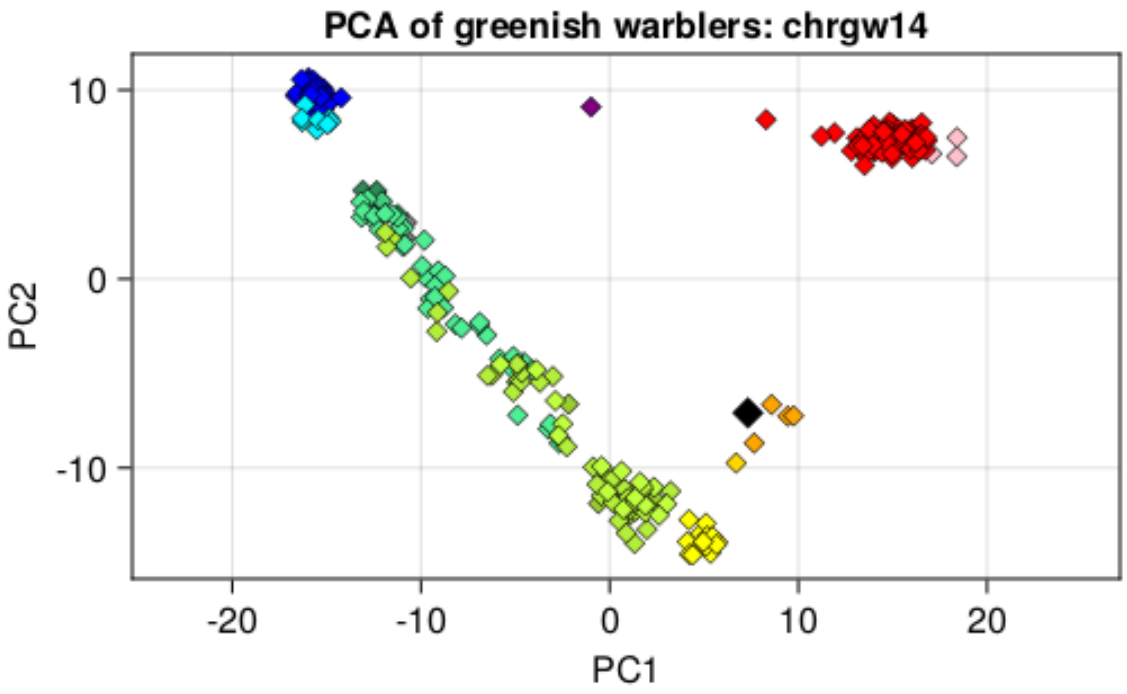

PCA of greenish warblers: chrgw9
10
0
-10
PC2
-20
0
20
PC1

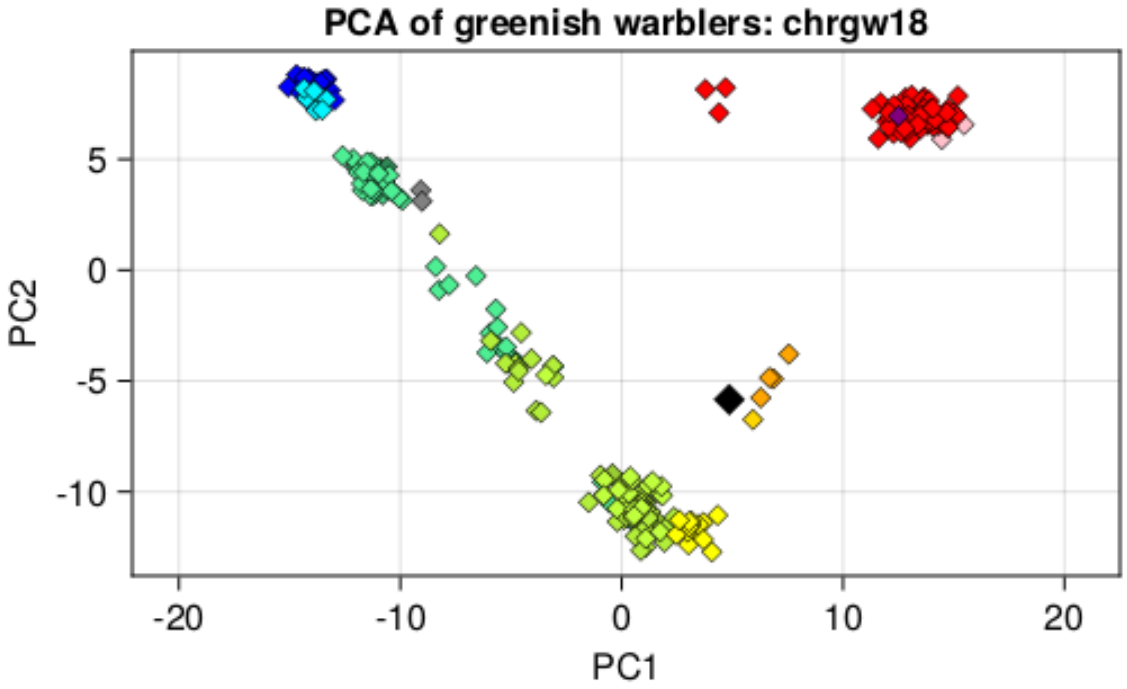

PCA of greenish warblers: chrgw11
10
0
-10
PC2
-20
-10
0
10
20
PC1

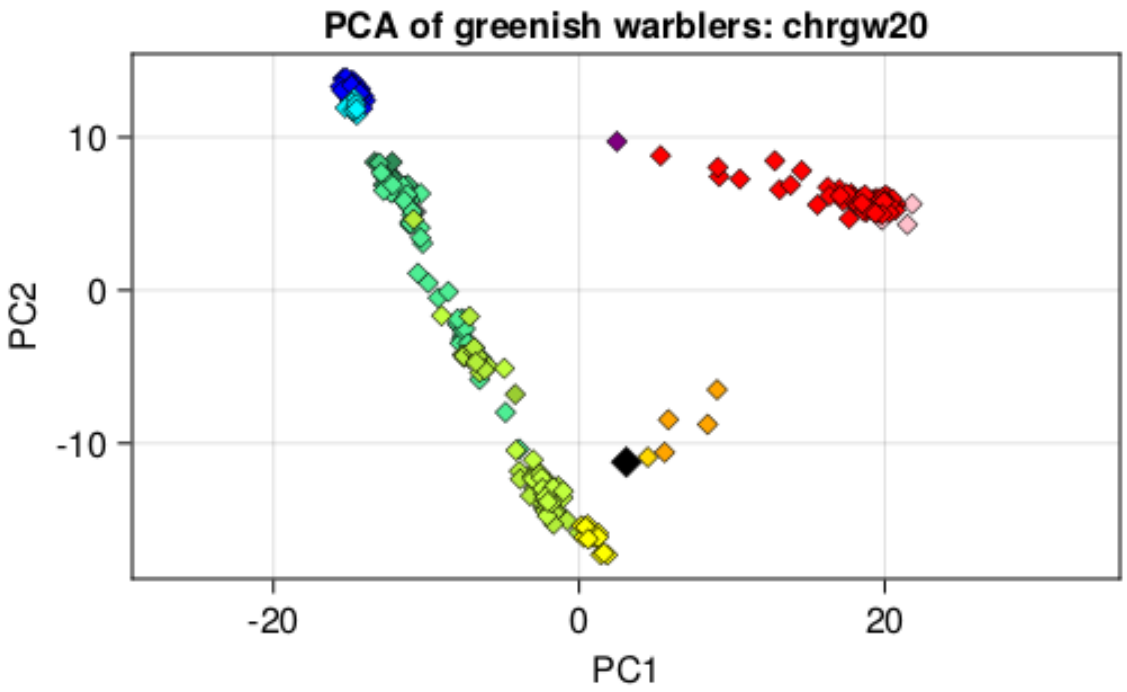

PCA of greenish warblers: chrgw12
10
0
-10
PC2
-20
-10
0
10
20
PC1

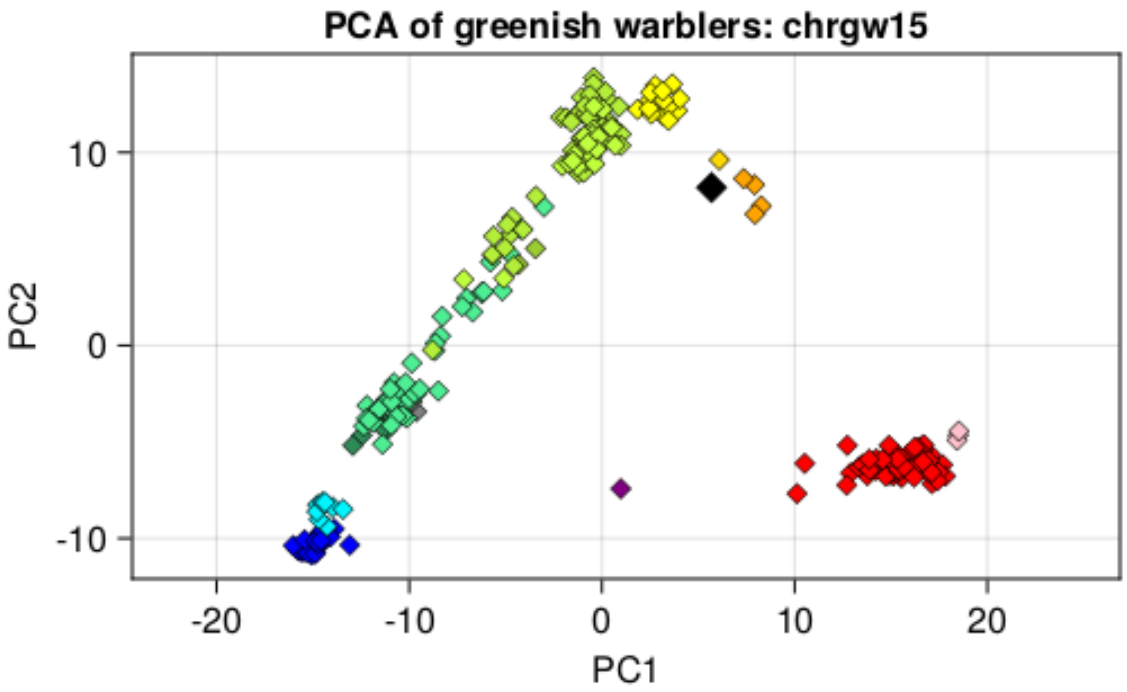

PCA of greenish warblers: chrgw10
PC2
PC1
10
5
0
-5
-10
-20
-10
0
10
20

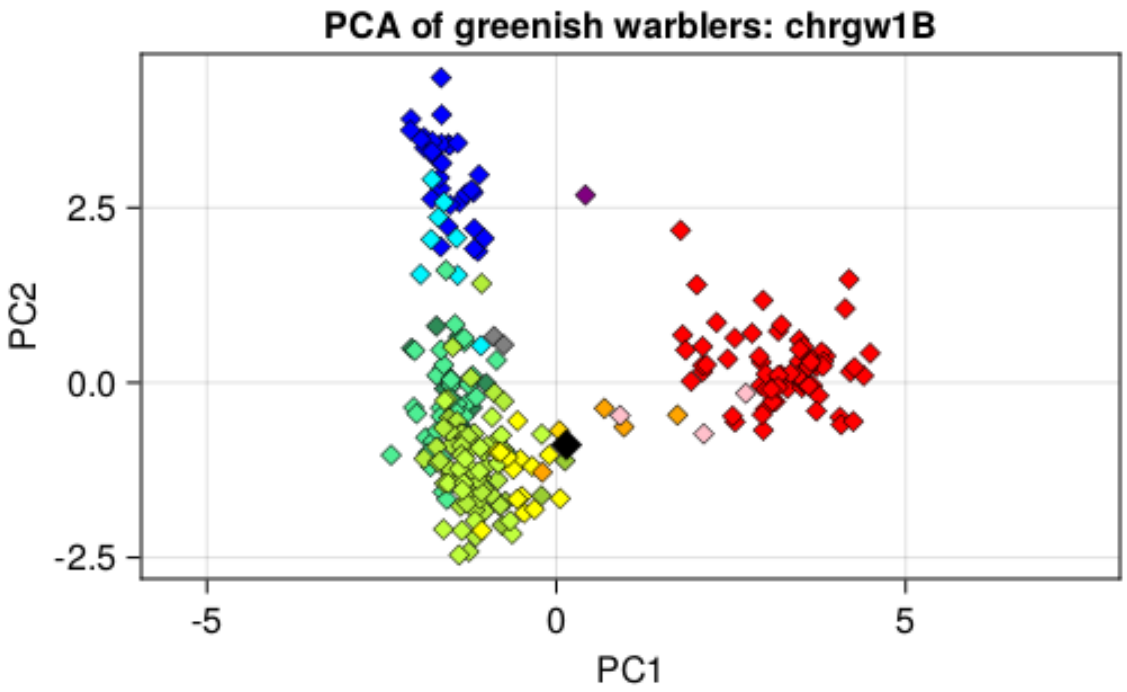

PCA of greenish warblers: chrgw13
10
0
-10
PC2
-20
0
20
PC1

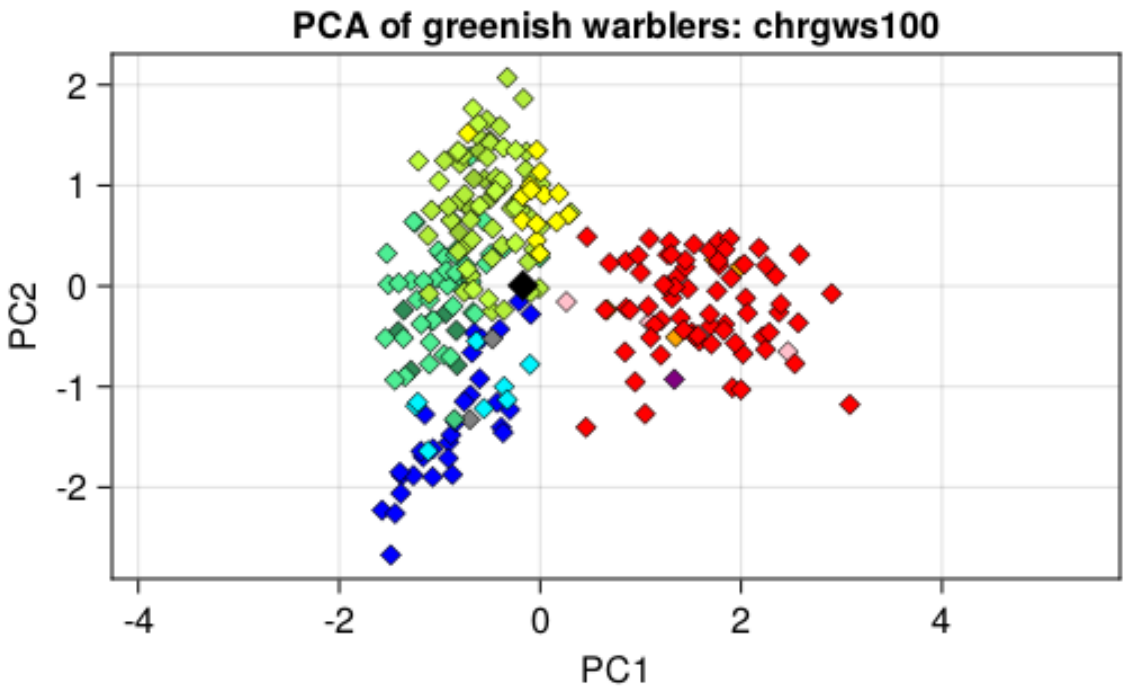

PCA of greenish warblers: chrgw14
10
0
-10
PC2
-20
-10
0
10
20
PC1

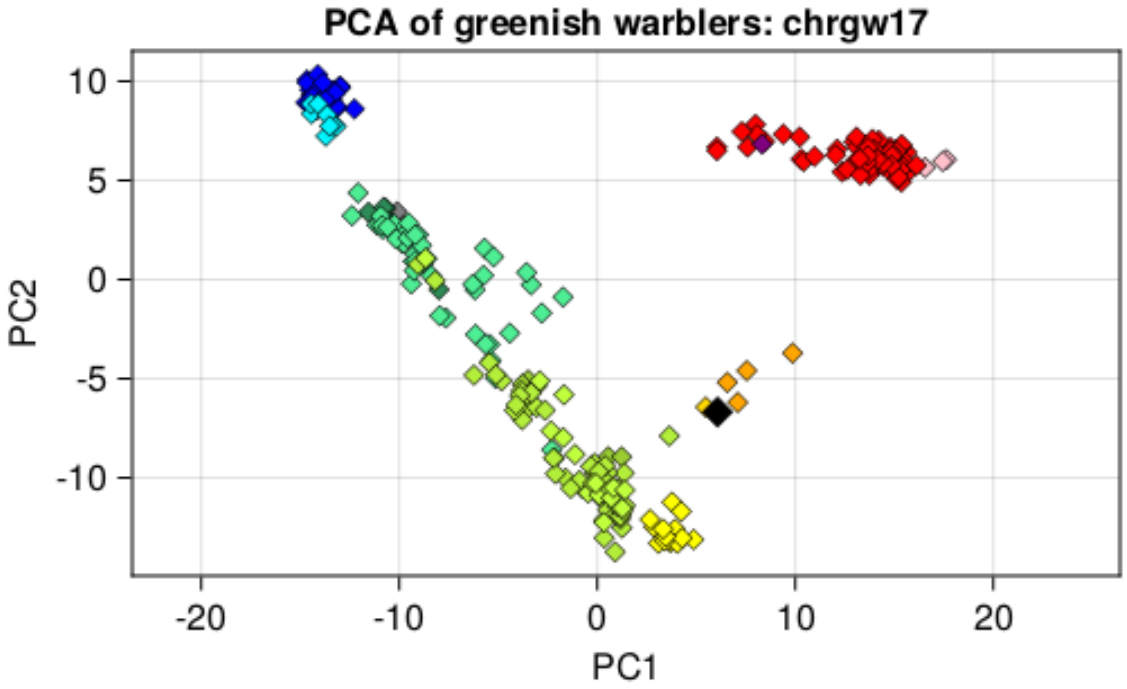

PCA of greenish warblers: chrgw18
PC2
PC1
5
0
-5
-10
-20
-10
0
10
20

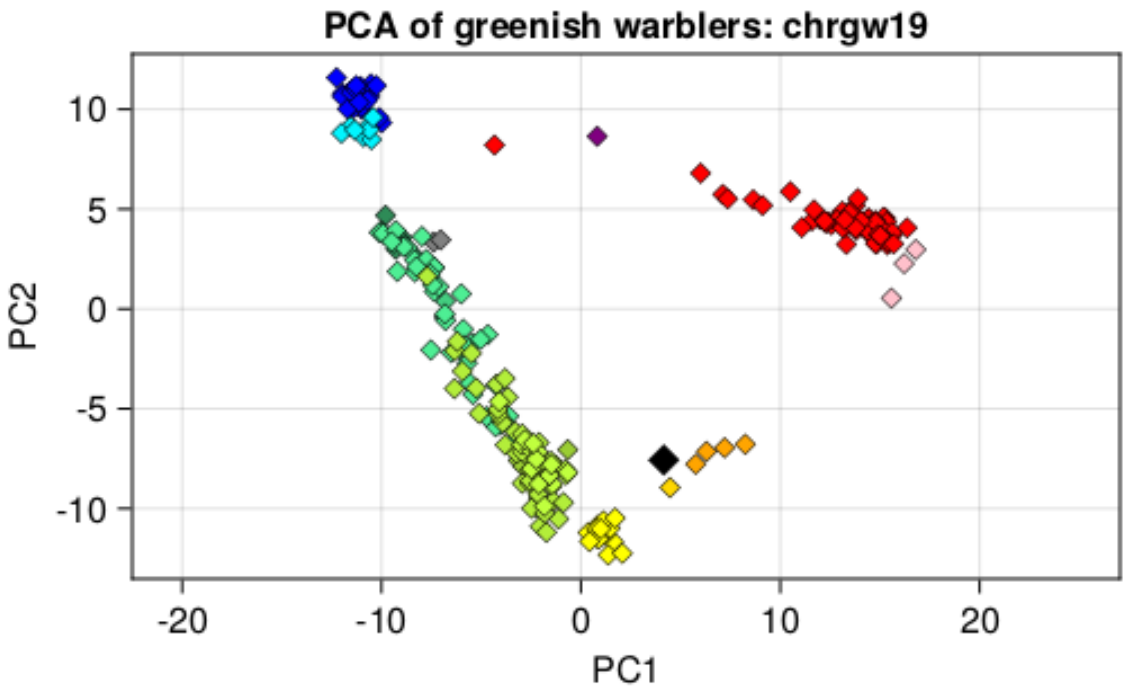

PCA of greenish warblers: chrgw20
10
0
-10
PC2
-20
0
20
PC1

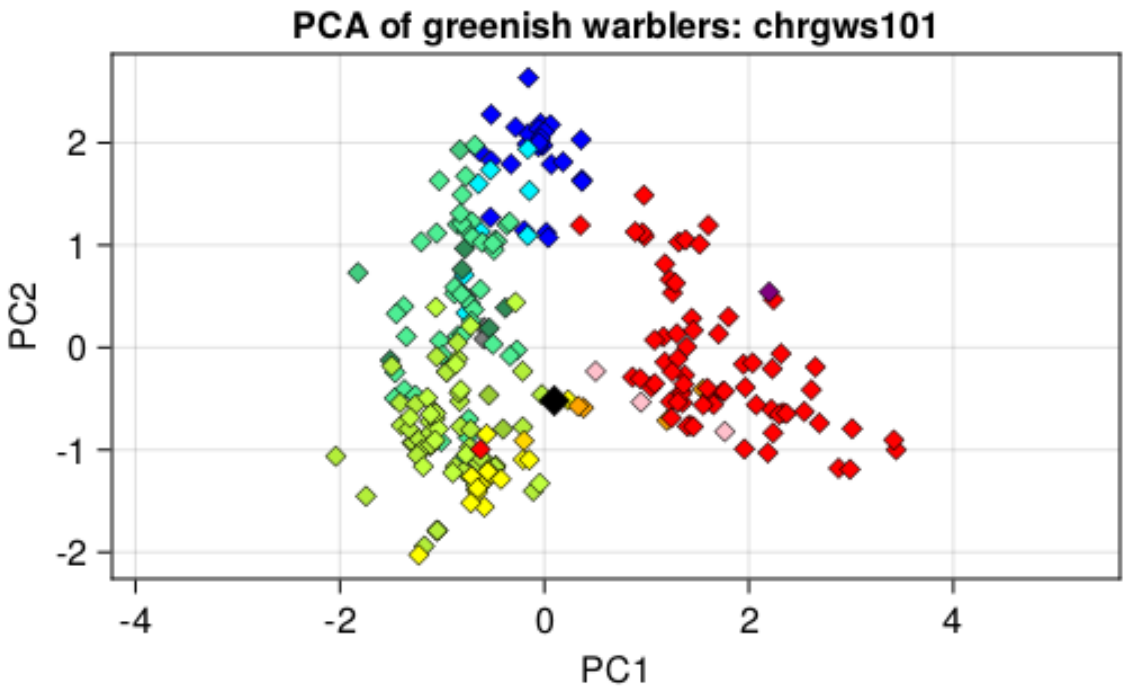

PCA of greenish warblers: chrgw15
PC2
PC1
10
0
-10
-20
-10
0
10
20

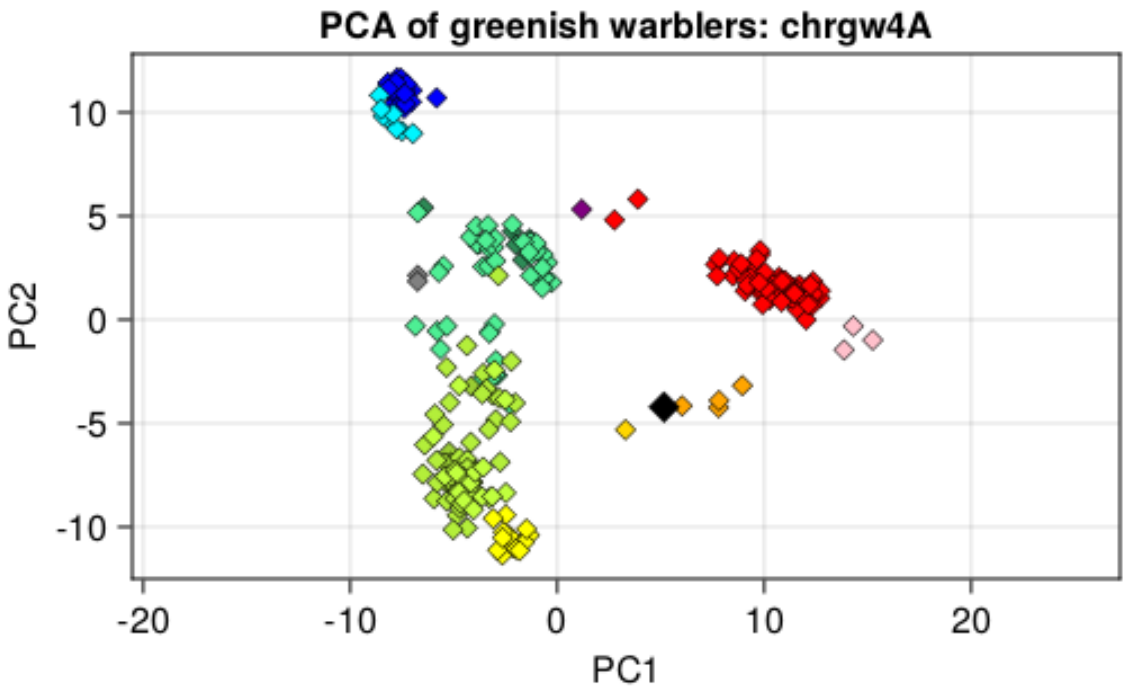

PCA of greenish warblers: chrgw1B
2.5
0.0
-2.5
PC2
-5
0
5
PC1

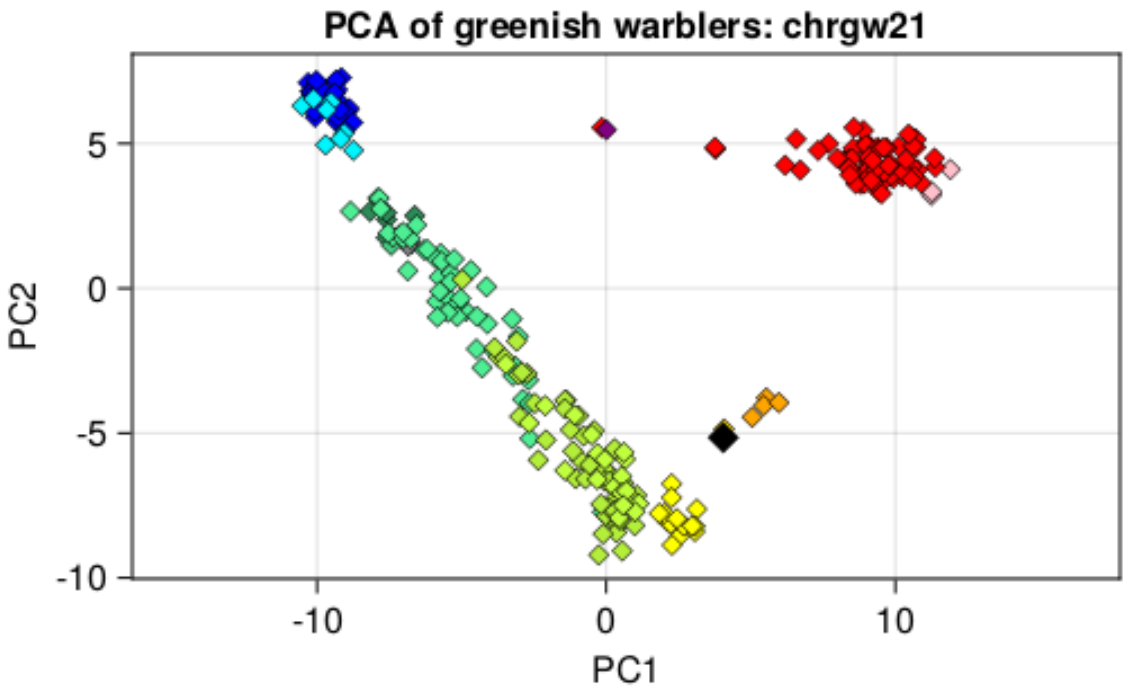

PCA of greenish warblers: chrgws100
PC2
PC1
2
1
0
-1
-2
-4
-2
0
2
4

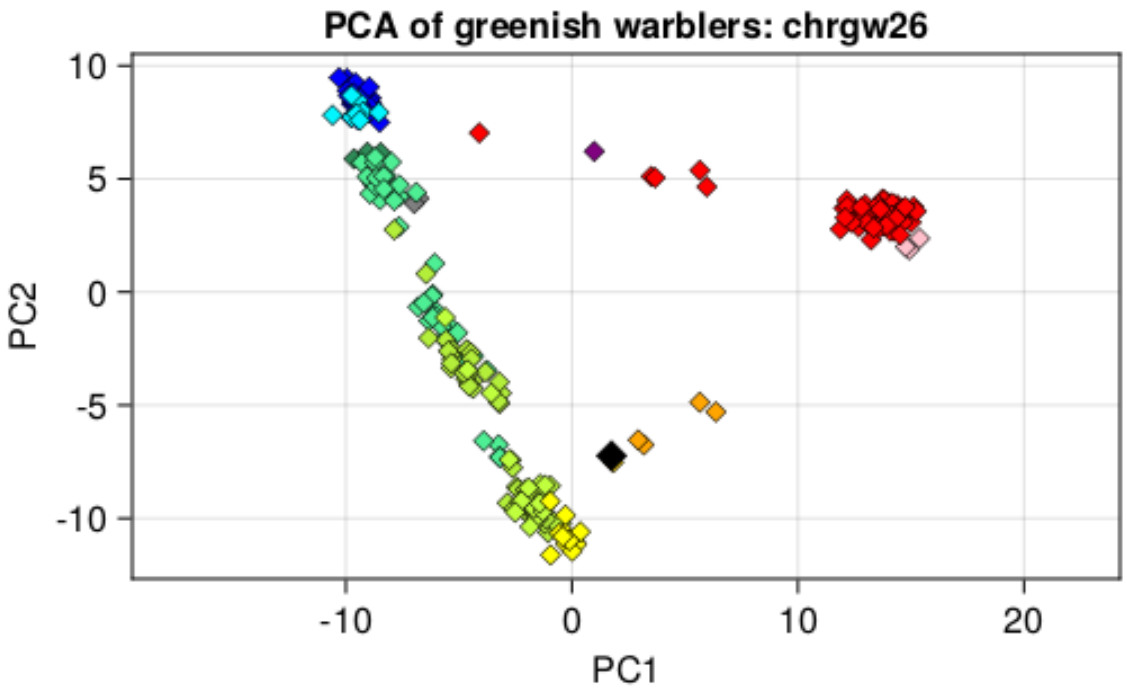

PCA of greenish warblers: chrgw17
PC2
PC1
10
5
0
-5
-10
-20
-10
0
10
20

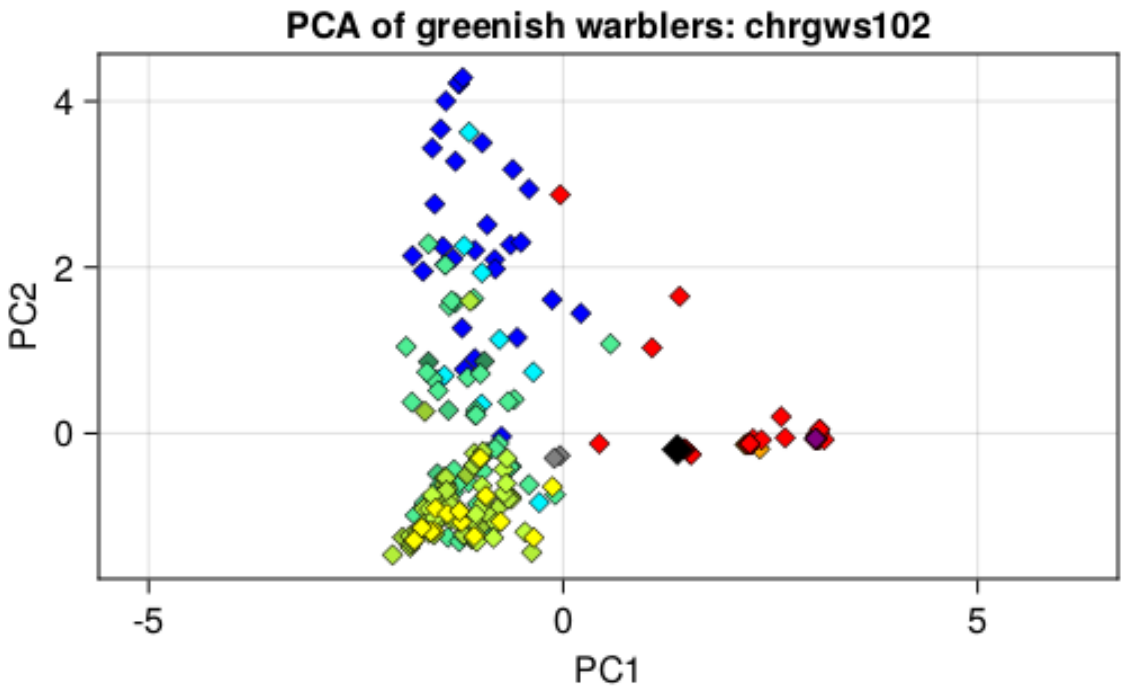

PCA of greenish warblers: chrgw19
PC2
PC1
10
5
0
-5
-10
-20
-10
0
10
20

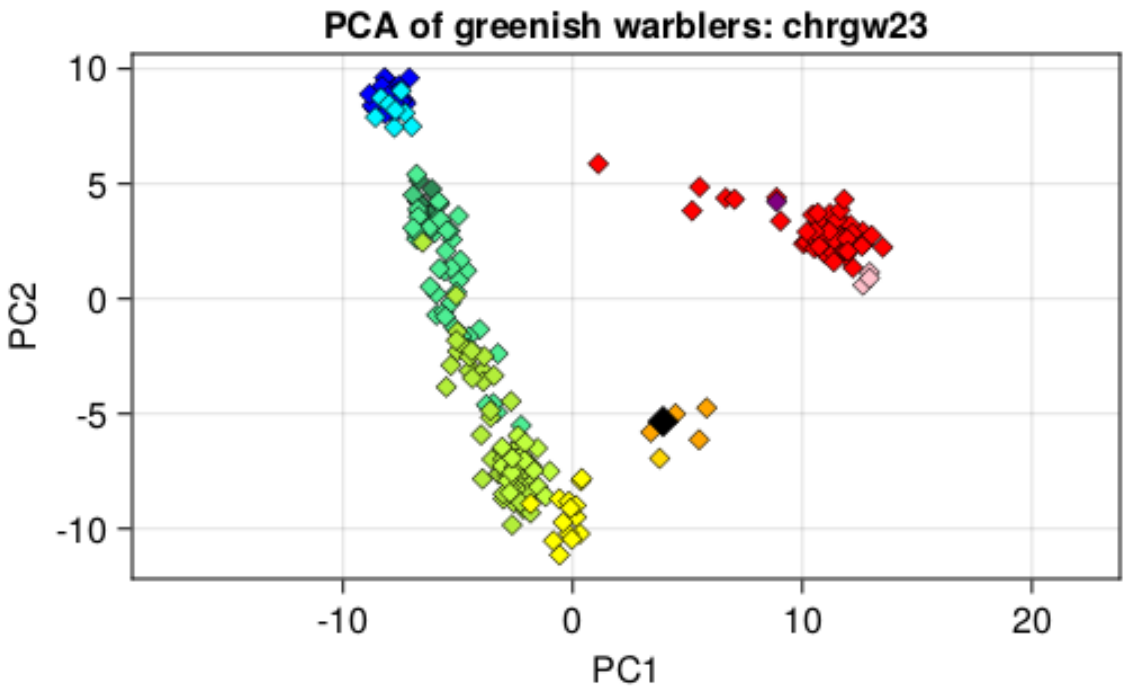

PCA of greenish warblers: chrgws101
PC2
PC1
2
1
0
-1
-2
-4
-2
0
2
4

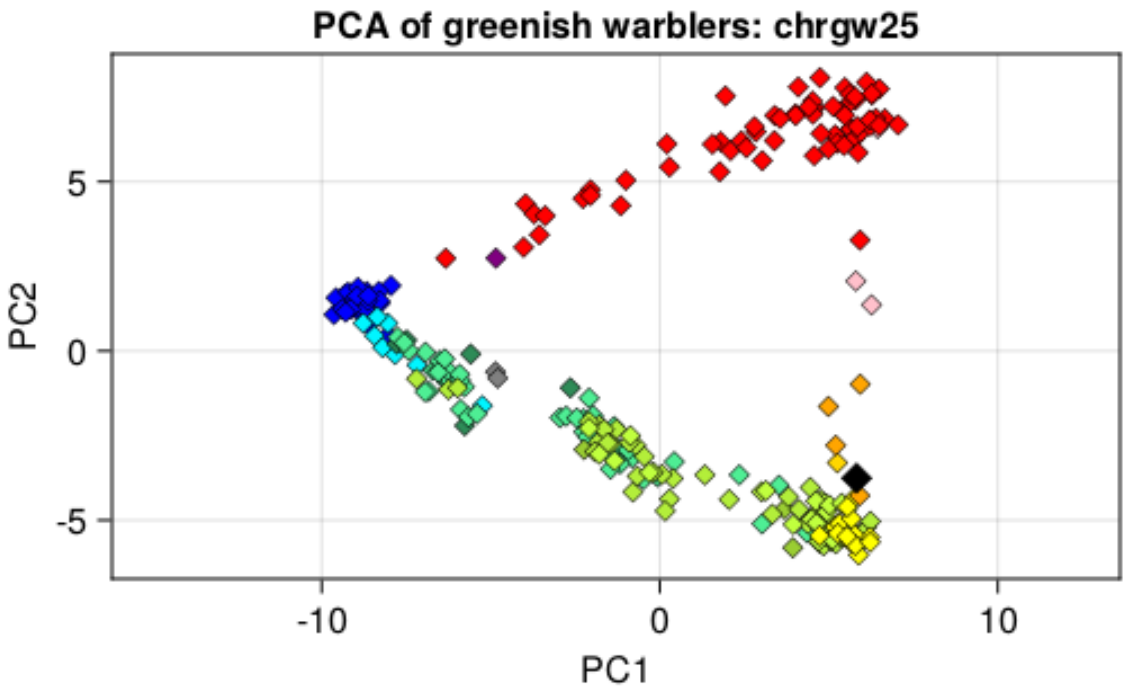

PCA of greenish warblers: chrgw4A
PC2
PC1
10
5
0
-5
-10
-20
-10
0
10
20

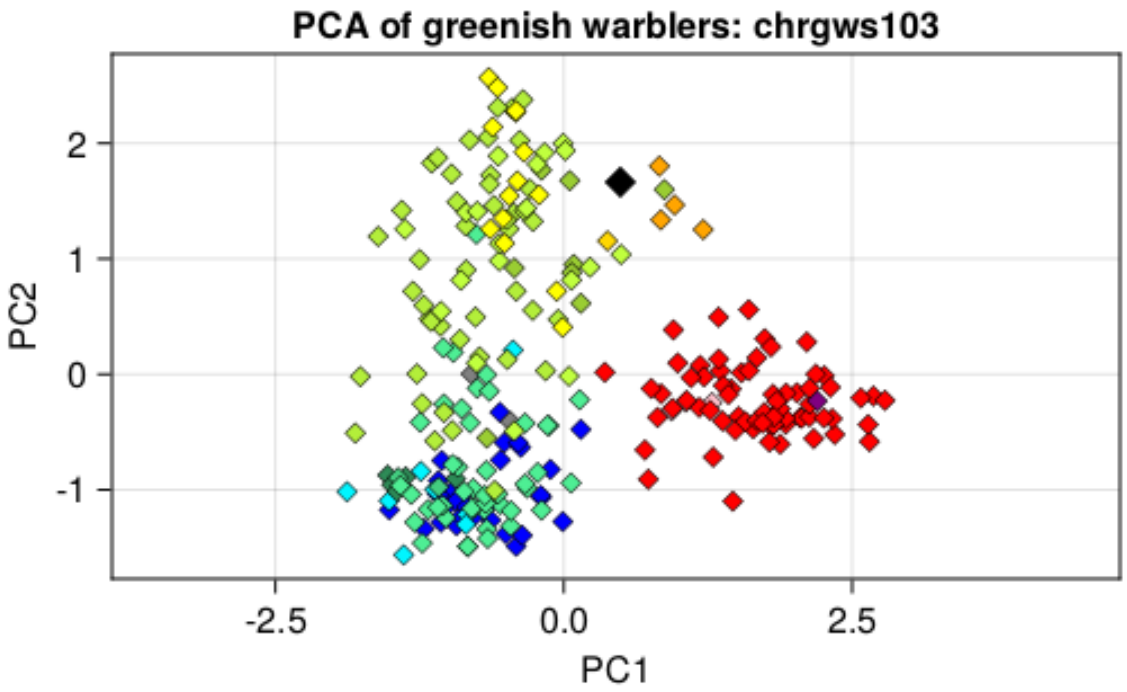

PCA of greenish warblers: chrgw21
PC2
PC1
5
0
-5
-10
-10
0
10

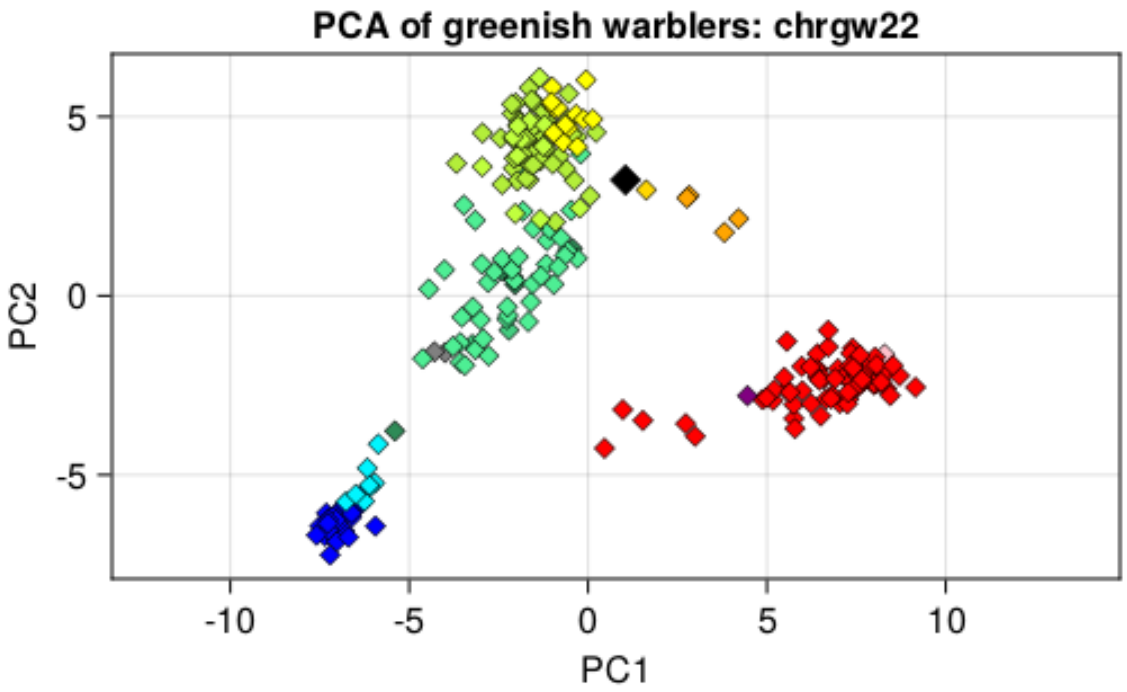

PCA of greenish warblers: chrgw26
10
5
0
-5
-10
PC2
-10
0
10
20
PC1

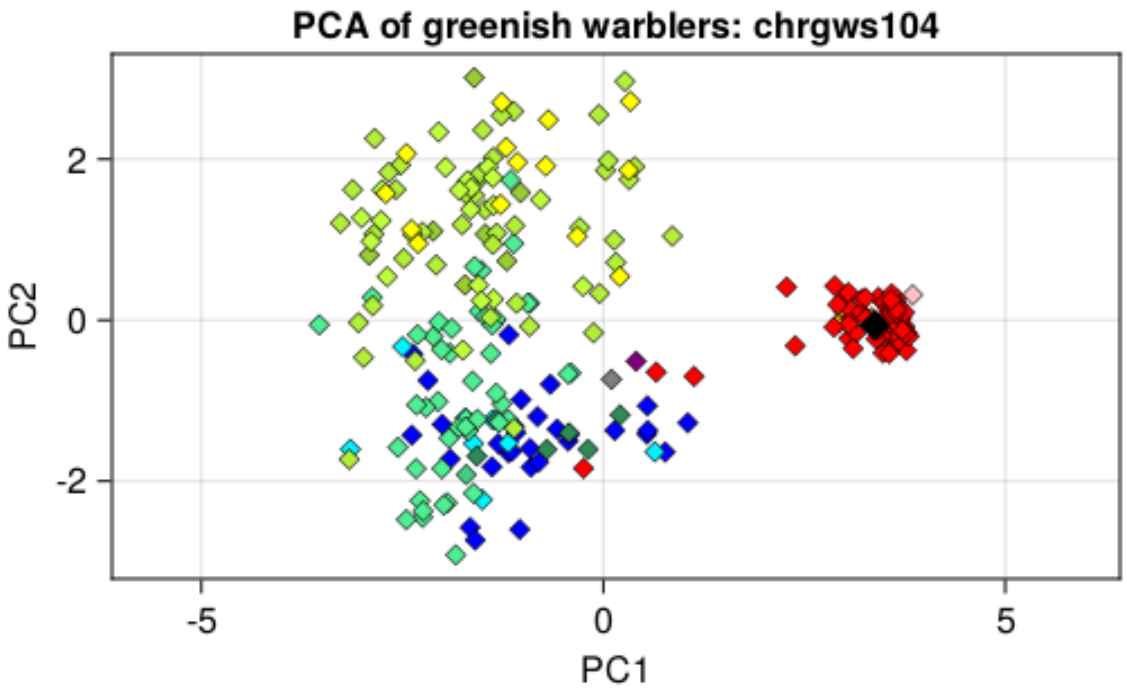

PCA of greenish warblers: chrgws102
PC2
PC1
4
2
0
-5
0
5

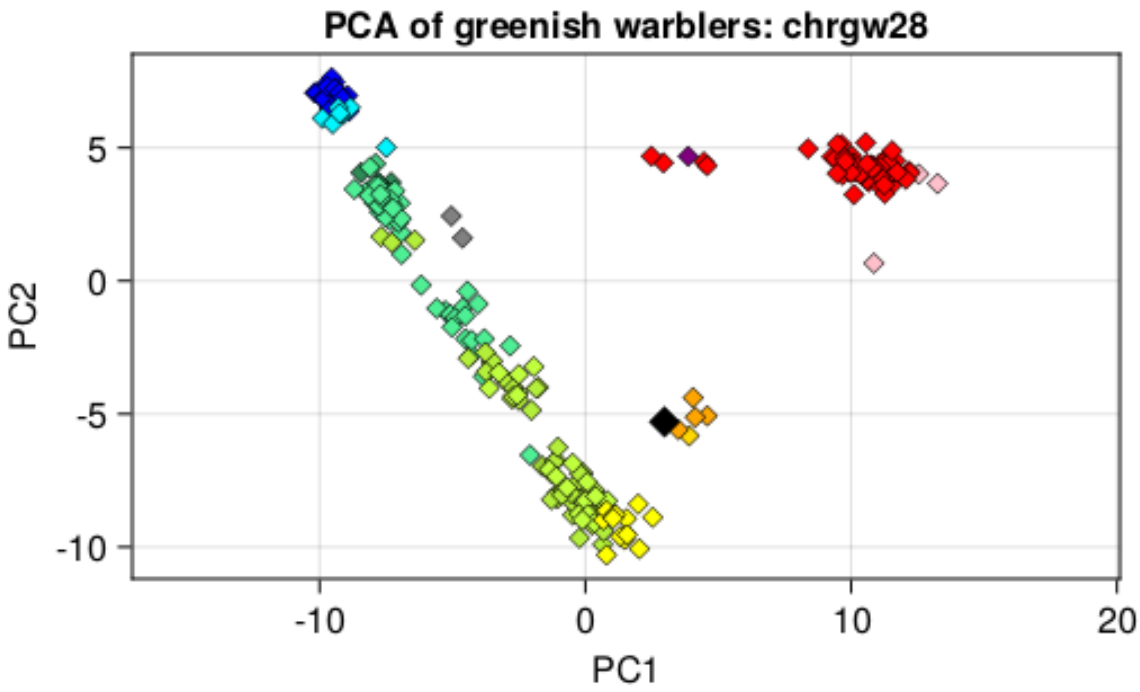

PCA of greenish warblers: chrgw23
10
5
0
-5
-10
PC2
-10
0
10
20
PC1

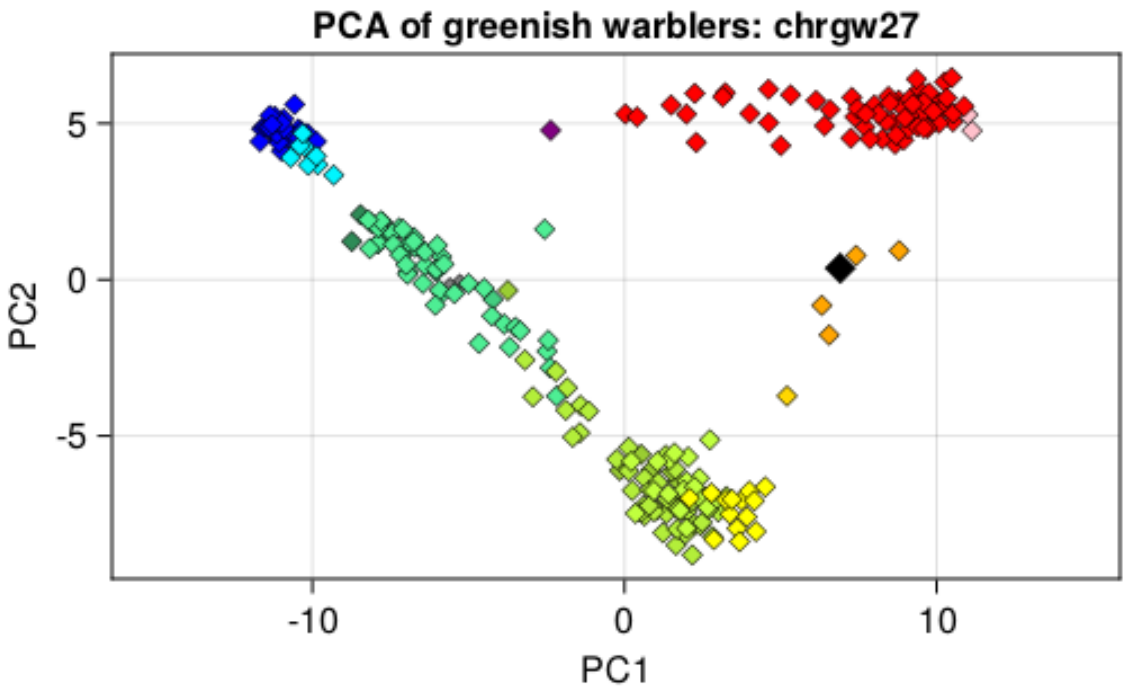

PCA of greenish warblers: chrgw25
PC2
5
0
-5
-10
0
10
PC1

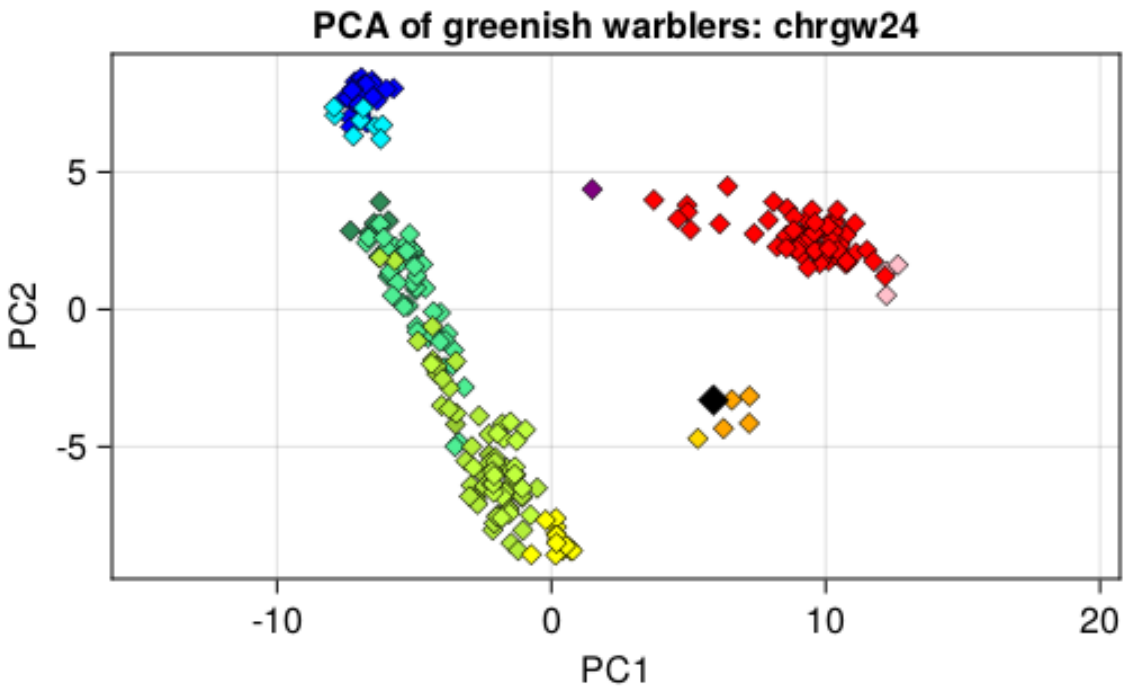

PCA of greenish warblers: chrgws103
PC2
2
1
0
-1
-2.5
0.0
2.5
PC1

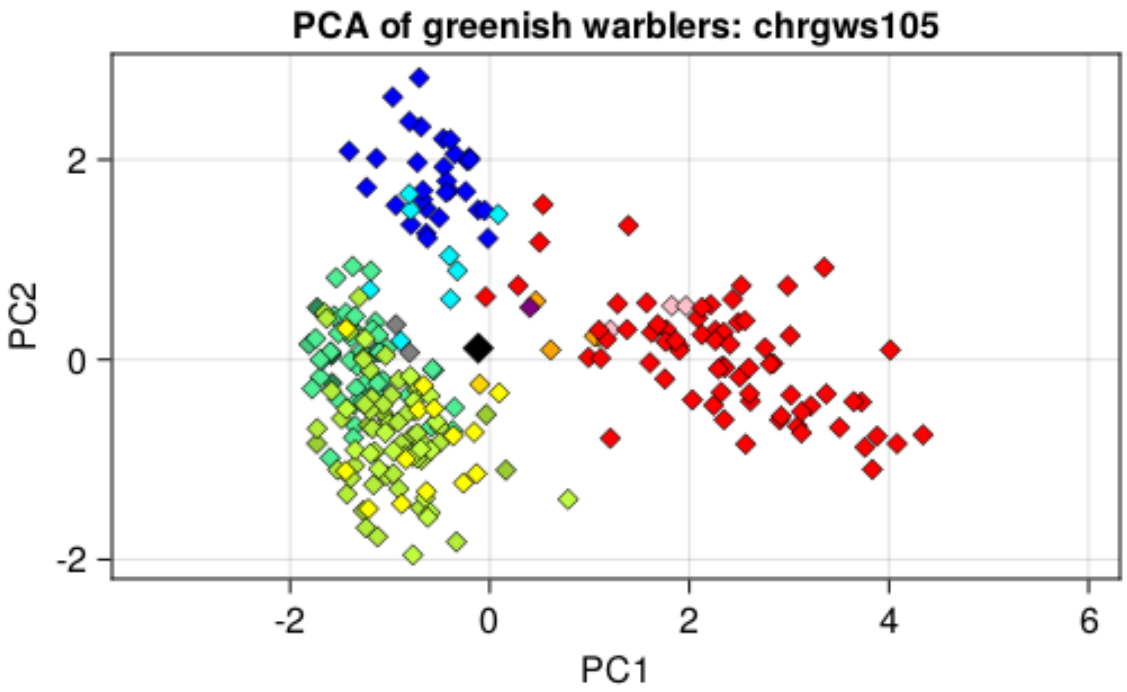

PCA of greenish warblers: chrgw22
PC2
PC1
5
0
-5
-10
-5
0
5
10


PCA of greenish warblers: chrgws104
PC2
2
0
-2
-5
0
5
PC1

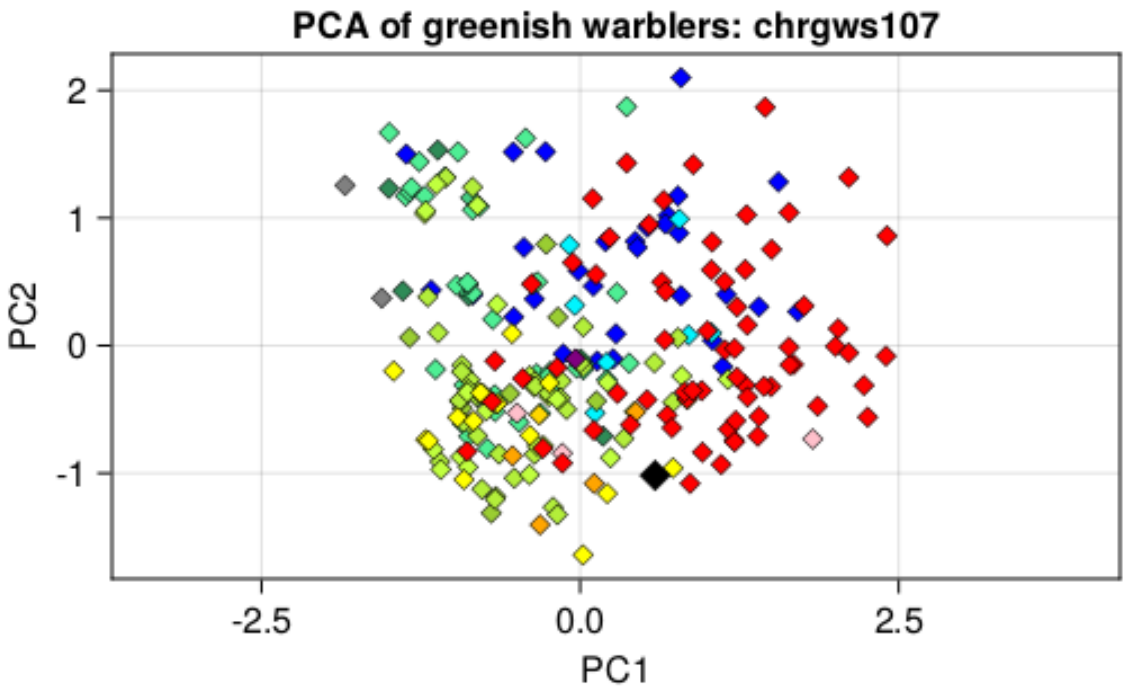

PCA of greenish warblers: chrgw28
PC2
PC1
5
0
-5
-10
-10
0
10
20

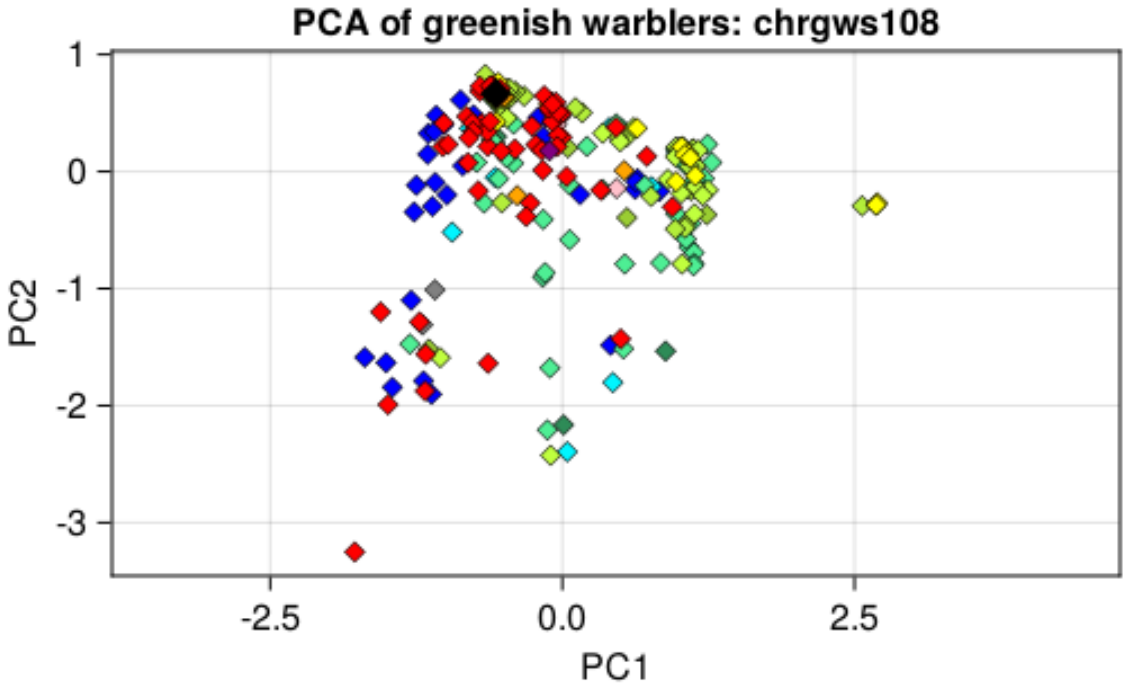

PCA of greenish warblers: chrgw27
5
0
-5
PC2
-10
0
10
PC1

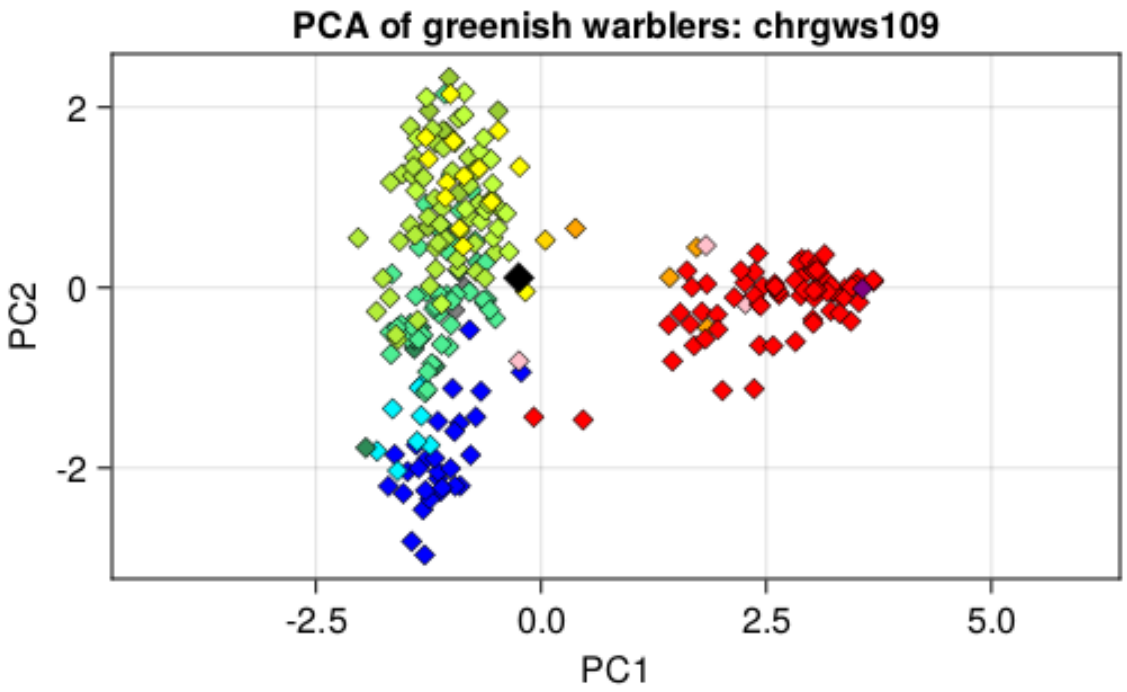

PCA of greenish warblers: chrgw24
PC2
PC1
5
0
-5
-10
0
10
20

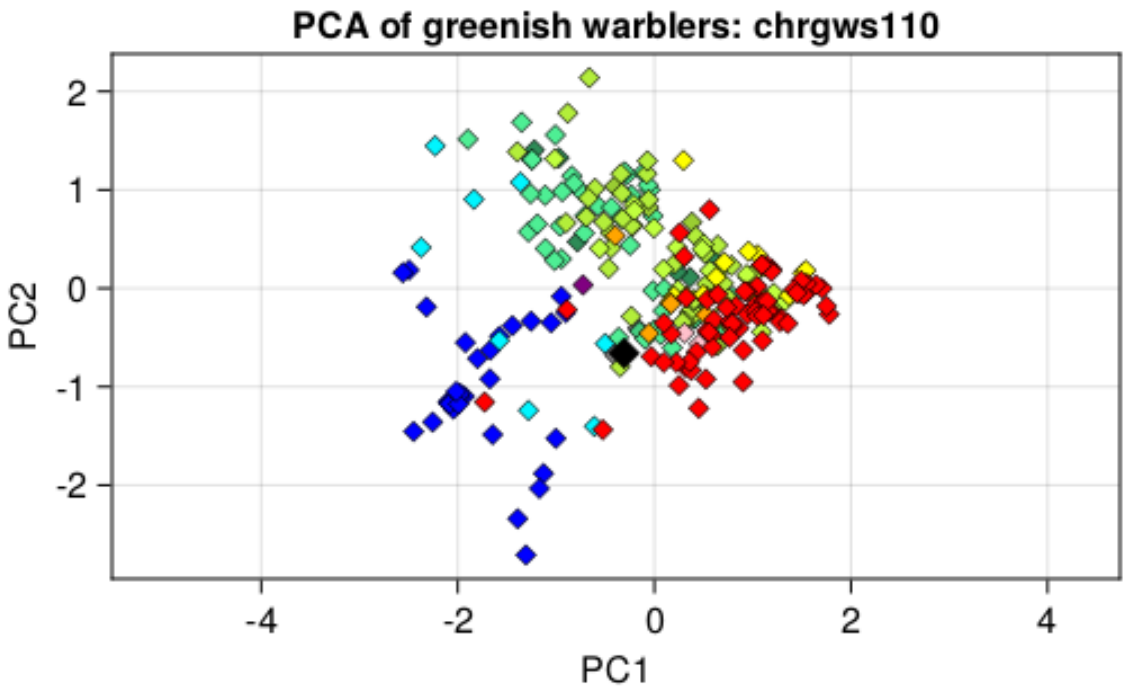

PCA of greenish warblers: chrgws105
PC2
PC1
2
0
-2
-2
0
2
4
6

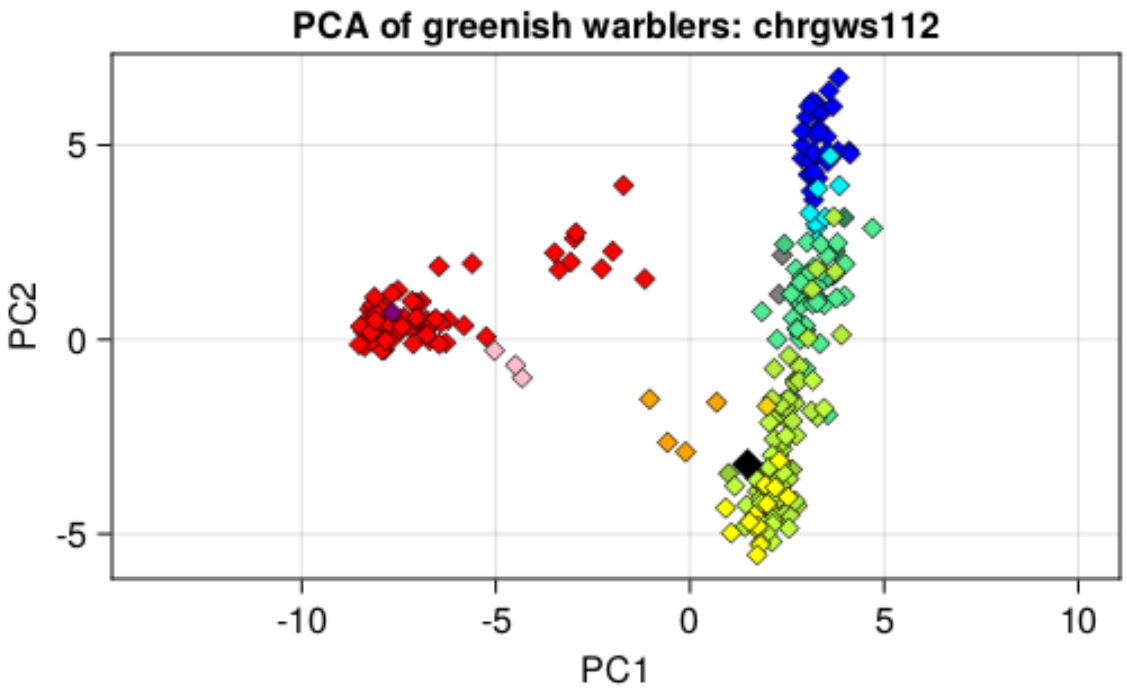

PCA of greenish warblers: chrgws106
PC2
PC1
2
1
0
-1
-2
0
2


PCA of greenish warblers: chrgws107
PC2
PC1
2
1
0
-1
-2.5
0.0
2.5


PCA of greenish warblers: chrgws108
1
0
-1
-2
-3
PC2
-2.5
0.0
2.5
PC1


PCA of greenish warblers: chrgws109
PC2
2
0
-2
-2.5
0.0
2.5
5.0
PC1


PCA of greenish warblers: chrgws110
PC2
PC1
2
1
0
-1
-2
-4
-2
0
2
4



Whole-genome PCA

In addition to making PCA plots for each scaffold, we can do one for the whole genome. The imputing for the whole genome takes some time (almost 2 hours!) for this dataset because the earlier GBS plates had low read depth so more missing genotypes, so I did this in advance and saved a file. This is incorporated into the code below–to actually do the imputing, set `do_imputing = true`. Otherwise this code will load the previously-imputed data.

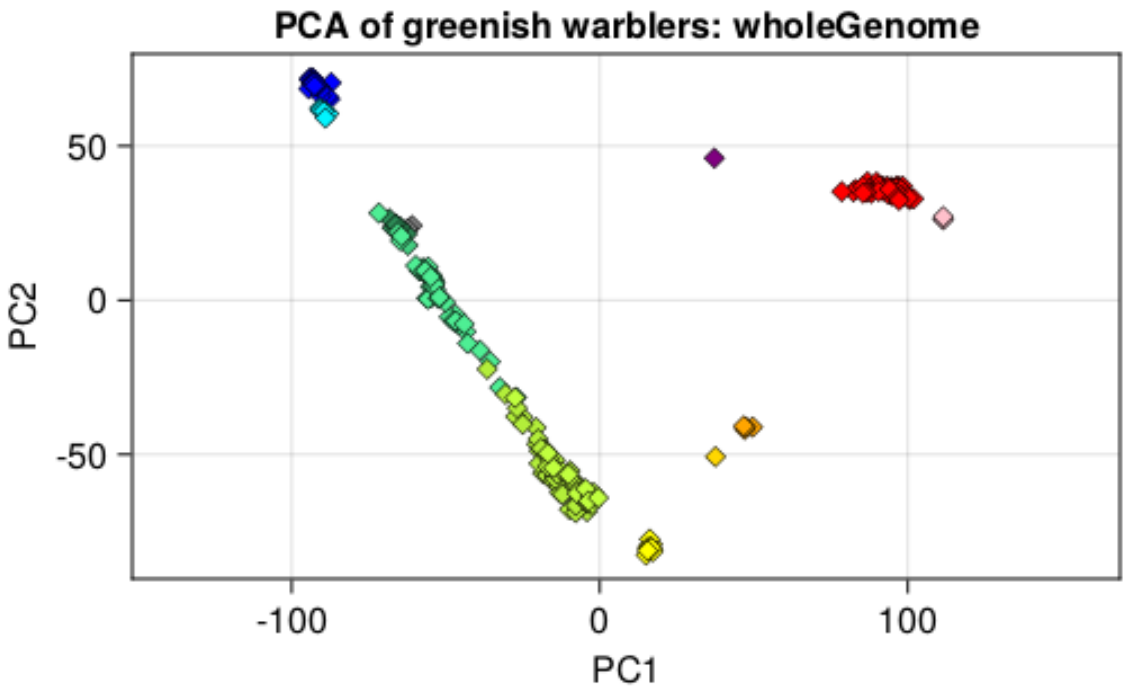
```
genosOnly_for_imputing = Matrix{Union{Missing, Float32}}(genosOnly_with_missing)
regionText = "wholeGenome"
filename = string(baseName, tagName, regionText, ".KNNimputedMissing.jld2")
# to do the imputing, do this by setting to true, but TAKES A LONG TIME (e.g. took 8 hours on laptop
do_imputing = false
if do_imputing
    # @time imputed_genosOnly = Impute.svd(genosOnly_for_imputing) # old method; knn should work bet
    @time imputed_genosOnly = Impute.knn(genosOnly_for_imputing; k=1, dims=:rows)
    jldsave(filename; imputed_genosOnly, ind_with_metadata_indFiltered, pos_SNP_filtered)
    imputed_genosOnly_wholeGenome = imputed_genosOnly
    ind_with_metadata_indFiltered_wholeGenome = ind_with_metadata_indFiltered
```

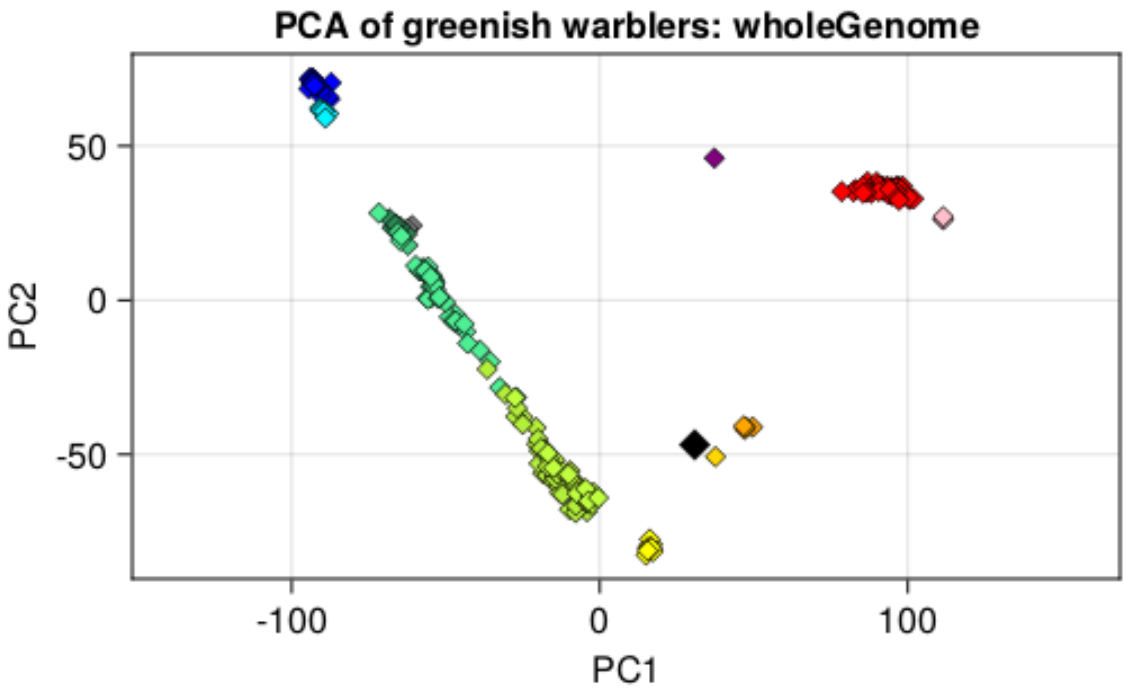
```
    pos_SNP_filtered_wholeGenome = pos_SNP_filtered
    print("Saved matrix of real and imputed genotypes for filtered individuals. \n")
else # load the already saved imputing
    imputed_genosOnly_wholeGenome = load(filename, "imputed_genosOnly")
    ind_with_metadata_indFiltered_wholeGenome = load(filename, "ind_with_metadata_indFiltered")
    pos_SNP_filtered_wholeGenome = load(filename, "pos_SNP_filtered")
    println(string("Loaded ",filename))
    println(string(regionText, ": ", size(imputed_genosOnly_wholeGenome, 2), " SNPs from ", size(impu
end
```

```
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
wholeGenome: 1017581 SNPs from 257 individuals
```

Now make the whole-genome PCA:

```
flipPC1 = true
flipPC2 = true
GW_wholeGenome_PCA = plotPCA(imputed_genosOnly_wholeGenome,
        ind_with_metadata_indFiltered_wholeGenome,
        groups_to_plot_PCA, group_colors_PCA;
        sampleSet = "greenish warblers", regionText=regionText,
        flip1 = flipPC1, flip2 = flipPC2)
ind_with_metadata_indFiltered_wholeGenome.PC1 = GW_wholeGenome_PCA.PC1
ind_with_metadata_indFiltered_wholeGenome.PC2 = GW_wholeGenome_PCA.PC2

# add position of reference genome
refGenomePCAposition = predict(GW_wholeGenome_PCA.model, zeros(size(imputed_genosOnly_wholeGenome, 2
flipPC1 && (refGenomePCAposition[1] *= -1)  # this flips PC1 if flipPC1 = true
flipPC2 && (refGenomePCAposition[2] *= -1)  # same for PC2
CairoMakie.scatter!(refGenomePCAposition[1], refGenomePCAposition[2], marker = :diamond, color="blac
try
    display(GW_wholeGenome_PCA.PCAfig)
catch
    println("NOTICE: Figure for ", regionText, " could not be shown due to an unknown error.")
end
```


PCA of greenish warblers: wholeGenome
50
0
-50
PC2
-100
0
100
PC1




```
CairoMakie.Screen{IMAGE}
```

Genotype-by-individual plots

Now, show individual genotypes for subsets of the dataset. Can choose individuals and genomic regions to plot, along with an Fst cutoff (only show SNPs with greater Fst than the cutoff).

```
set = "vir_plumb"    #"59_inds_around_ring"  #"east_side_of_ring"    #"67_inds_around_ring"  # "west

if set == "59_inds_around_ring"
    groups = ["vir","troch_LN","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["vir","vir_S","lud_PK","lud_KS","lud_central","troch_LN","troch_EM","obs", "plumb_L
    plotGroupColors = ["blue","turquoise1","seagreen4","seagreen3","seagreen2","yellow","gold","orang
    numIndsToPlot = [10, 5, 2, 1, 8, 15, 1, 4, 3, 10] # maximum number of individuals to plot from ea
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "vir_plumb"   #"Fst_among"  #"vir_troch_LN"       #"vir_plumb"      #"troch_LN
    Fst_cutoff = 0.95
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
```

```
elseif set == "37_inds_around_ring_plusAllVirPlumb"
    groups = ["vir","troch_LN","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["vir","lud","troch_LN","troch_EM","obs", "plumb_BJ","plumb"]
    plotGroupColors = ["blue","seagreen4","yellow","gold","orange", "pink","red"]
    numIndsToPlot = [100, 15, 15, 15, 15, 15, 100] # maximum number of individuals to plot from each
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "Fst_among"
    Fst_cutoff = 0.7
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "west_side_of_ring"
    groups = ["vir","troch_LN"] # for purpose of calculating pairwise Fst and Fst_group (to determine
    plotGroups = ["vir","vir_misID","vir_S","nit", "lud_PK", "lud_KS", "lud_central", "lud_Sath", "lu
    plotGroupColors = ["blue","blue","turquoise1","grey","seagreen4","seagreen3","seagreen2","olivedr
    numIndsToPlot = [15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15] # maximum number of individuals to p
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "troch_LN"
    groupsToCompare = "vir_troch_LN" # "Fst_among"
    Fst_cutoff = 0.6
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "all_ludlowi_plus_a_few_other"
    groups = ["vir","troch_LN","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["vir","vir_S","nit", "lud_PK", "lud_KS", "lud_central", "lud_Sath", "lud_ML","troch
    plotGroupColors = ["blue","turquoise1","grey","seagreen4","seagreen3","seagreen2","olivedrab3","c
    numIndsToPlot = [4, 4, 4, 1000, 1000, 1000, 1000, 1000, 1000, 4, 4] # maximum number of individua
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "troch_LN"
    groupsToCompare = "vir_troch_LN" # "Fst_among"
    Fst_cutoff = 0.6
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "east_side_of_ring"
    groups = ["troch_LN","obs","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["troch_LN","troch_EM","obs","plumb_BJ","plumb"]
    plotGroupColors = ["yellow","gold","orange","pink","red"]
    numIndsToPlot = [15, 15, 15, 15, 15] # maximum number of individuals to plot from each group
    group1 = "troch_LN"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "troch_LN_plumb"
    Fst_cutoff = 0.7
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "vir_plumb"
```

```
    groups = ["vir","plumb"]
    plotGroups = ["vir","plumb_vir","plumb"]
    plotGroupColors = ["blue","purple","red"]
    numIndsToPlot = [100,100,100] # maximum number of individuals to plot from each group
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "vir_plumb"
    Fst_cutoff =  0.8
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data amon
end
```

```
0.2
```

Calculate allele freqs and sample sizes (use column Fst_group)

```
freqs, sampleSizes = getFreqsAndSampleSizes(genosOnly_with_missing, ind_with_metadata_indFiltered.Fs
println("Calculated population allele frequencies and sample sizes")
```

```
Calculated population allele frequencies and sample sizes
```

calculate Fst

```
Fst, FstNumerator, FstDenominator, pairwiseNamesFst = getFst(freqs, sampleSizes, groups; among=true)
println("Calculated Fst values")
```

```
Calculated Fst values
```

limit the individuals to include in plot

```
# For this figure only, filter out individuals with lots of missing genotypes
numMissings_threshold = 800_000
selection = ind_with_metadata_indFiltered.numMissings .< numMissings_threshold
genosOnly_with_missing_selected = view(genosOnly_with_missing, selection, :)
ind_with_metadata_indFiltered_selected = view(ind_with_metadata_indFiltered, selection, :)
# now limit each group to specified numbers
genosOnly_included, ind_with_metadata_included = limitIndsToPlot(plotGroups, numIndsToPlot, genosOnl
```

choose the scaffold and region to show

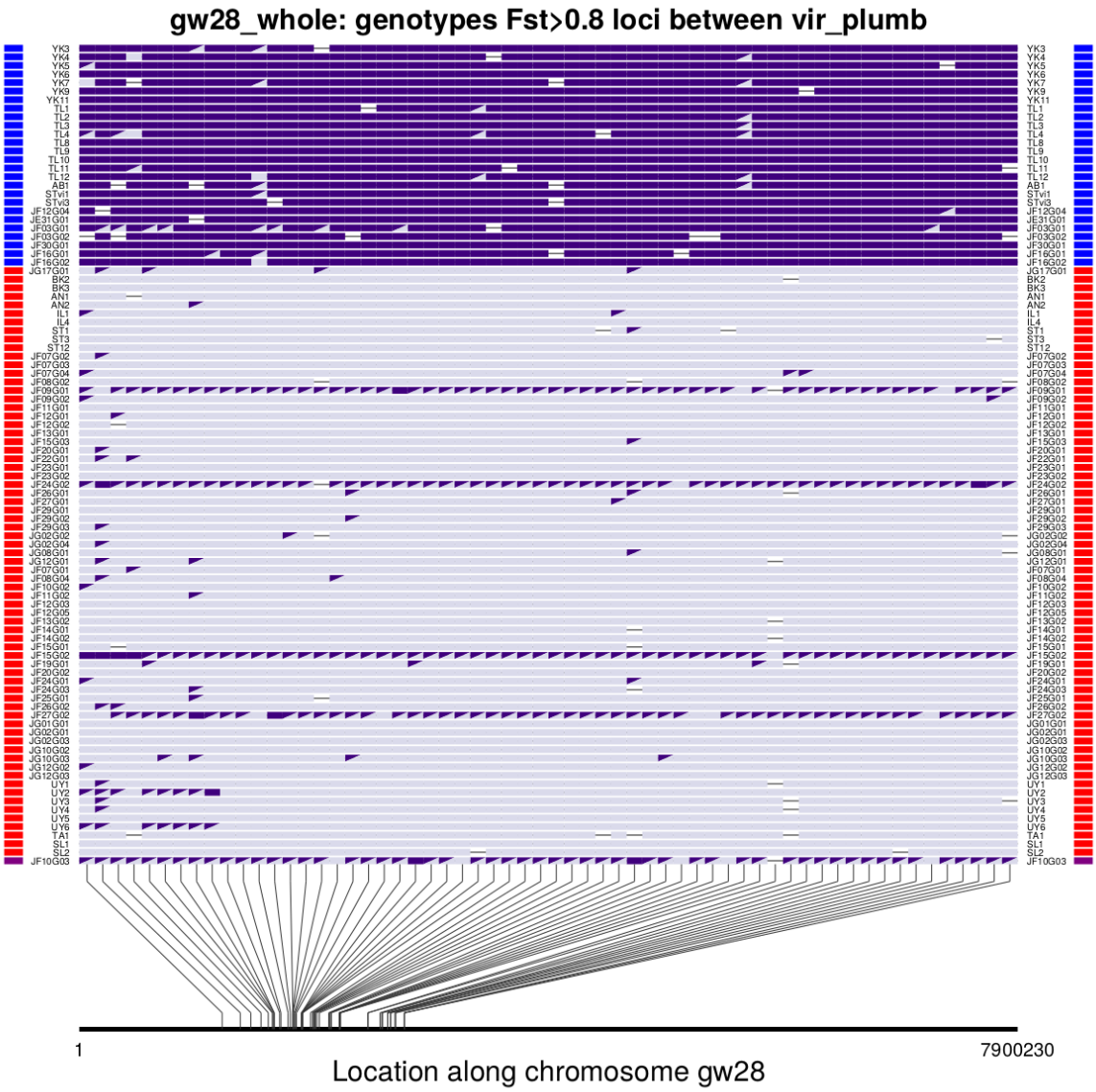
```
chr = "gw28"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN) # this gets the r
```

```
("gw28", 1, 7900230, "gw28_whole")
```

NOTE FOR LATER: SHOULD REALLY GET CHROMOSOME LENGTH FOR positionMax

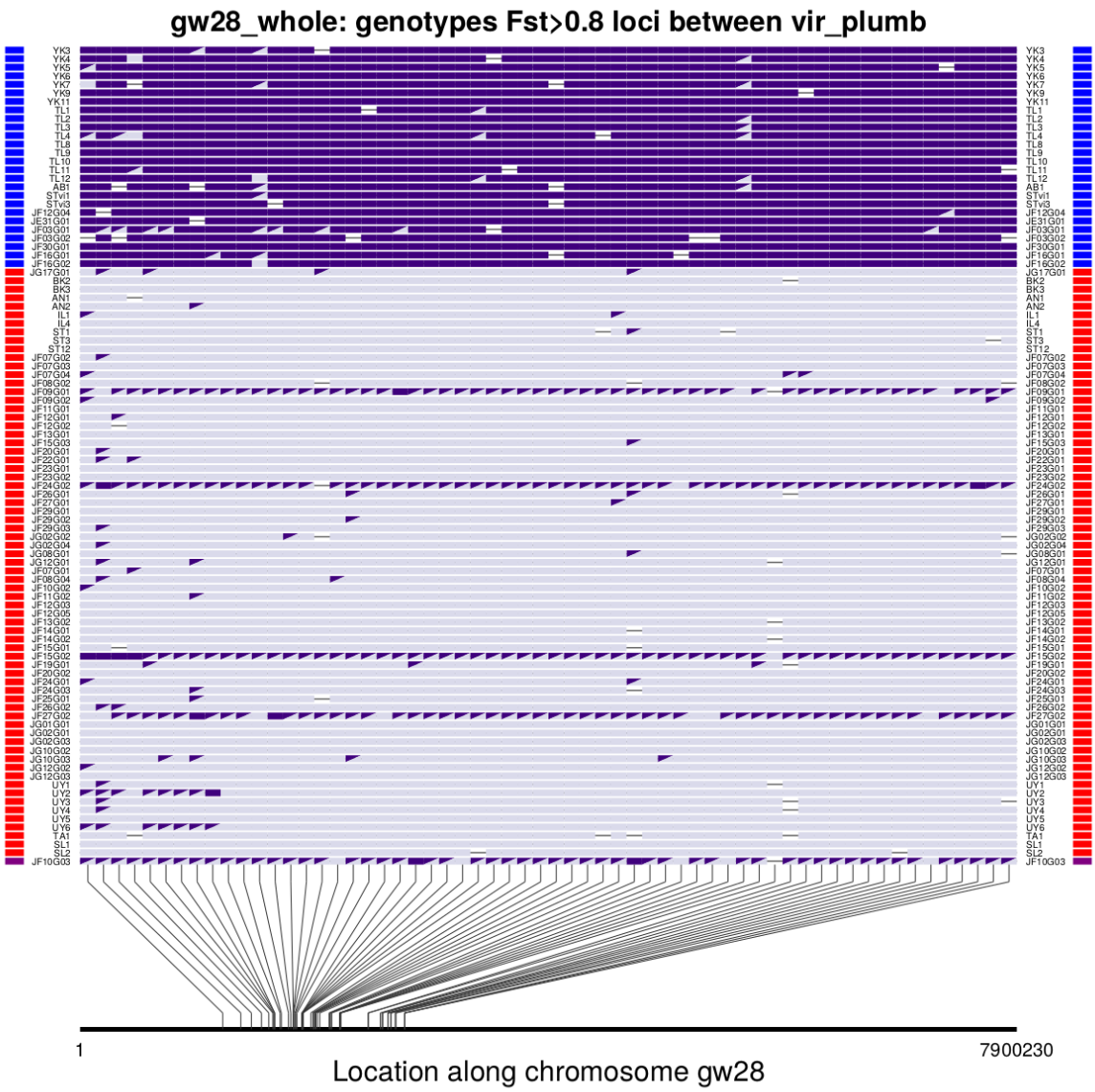
Now actually make the plot

```
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff, missingFractionAllowed,
    regionInfo, pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_included, ind_with_metadata_included, freqs, plotGroups, plotGroupColors);
# plotInfo contains a tuple with: (f, plottedGenotype, locations, plottedMetadata)
```



choose another chromosome, and plot similarly to above

```
chr = "gw28"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN) # this gets the r
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff, missingFractionAllowed,
    regionInfo, pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_included, ind_with_metadata_included, freqs, plotGroups, plotGroupColors);
```

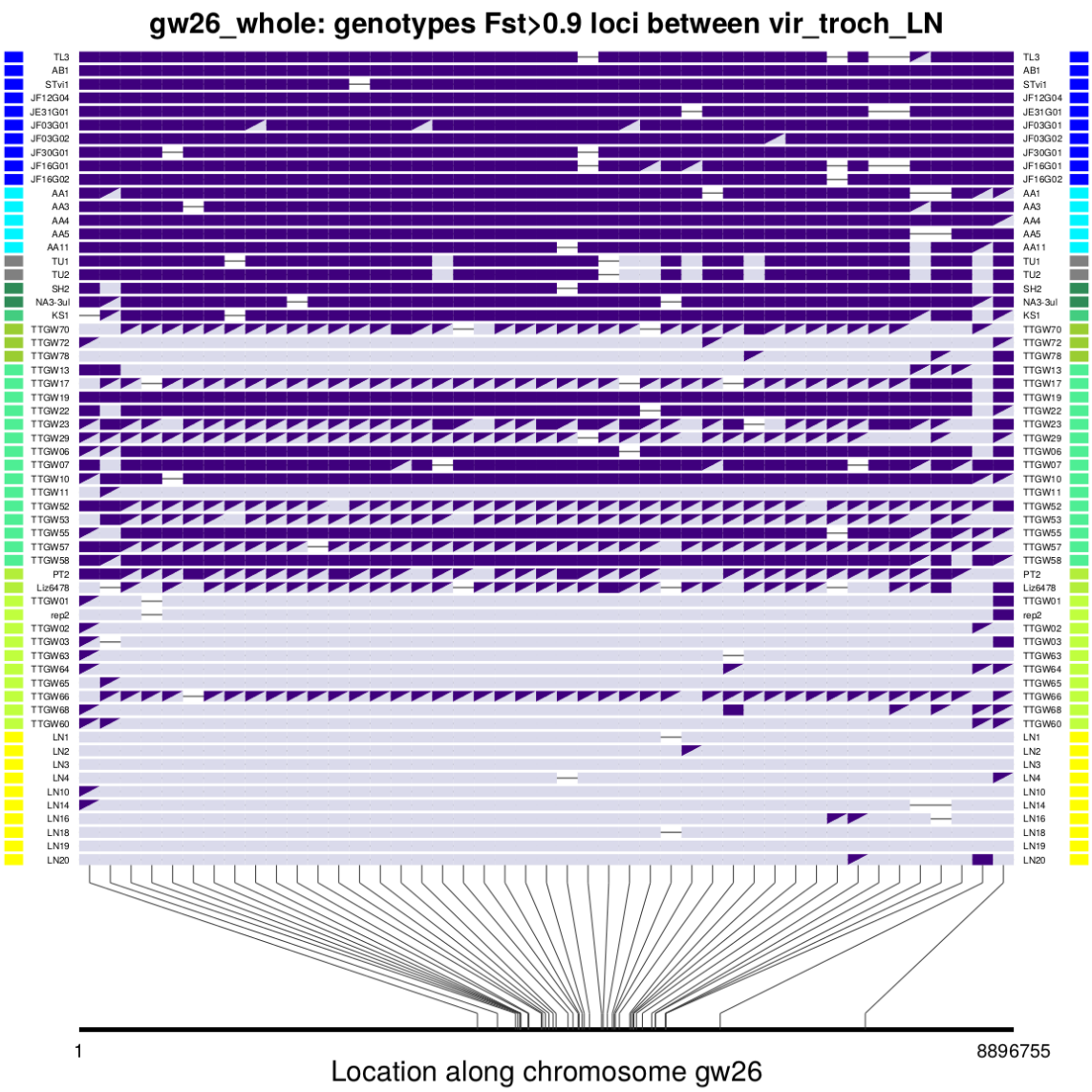


Make a GBI plot to illustrate variation along west side of ring

```
groups = ["vir","troch_LN"] # for purpose of calculating pairwise Fst and Fst_group (to determine SNI
plotGroups = ["vir","vir_S","nit", "lud_PK", "lud_KS", "lud_central", "lud_Sath", "lud_ML","troch_wes
plotGroupColors = ["blue","turquoise1","grey","seagreen4","seagreen3","seagreen2","olivedrab3","olive
numIndsToPlot = [10, 5, 2, 3, 5, 15, 3, 5, 10, 10] # maximum number of individuals to plot from each
group1 = "vir"   # these groups will determine the color used in the graph
```

```
group2 = "troch_LN"
groupsToCompare = "vir_troch_LN" # "Fst_among"
Fst_cutoff = 0.9
missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among ind

# Calculate allele freqs and sample sizes (use column Fst_group)
freqs, sampleSizes = getFreqsAndSampleSizes(genosOnly_with_missing, ind_with_metadata_indFiltered.Fst
println("Calculated population allele frequencies and sample sizes")

# calculate Fst
Fst, FstNumerator, FstDenominator, pairwiseNamesFst = getFst(freqs, sampleSizes, groups; among=true)
println("Calculated Fst values")

# limit the individuals to include in plot
# For this figure only, filter out individuals with lots of missing genotypes
numMissings_threshold = 800_000
selection = ind_with_metadata_indFiltered.numMissings .< numMissings_threshold
genosOnly_with_missing_selected = view(genosOnly_with_missing, selection, :)
ind_with_metadata_indFiltered_selected = view(ind_with_metadata_indFiltered, selection, :)
# now limit each group to specified numbers
genosOnly_included, ind_with_metadata_included = limitIndsToPlot(plotGroups, numIndsToPlot, genosOnly

# choose the scaffold and region to show
chr = "gw26"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN) # this gets the r

#### Now actually make the plot
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff, missingFractionAllowed,
    regionInfo, pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_included, ind_with_metadata_included, freqs, plotGroups, plotGroupColors);
# plotInfo contains a tuple with: (f, plottedGenotype, locations, plottedMetadata)
```

```
Calculated population allele frequencies and sample sizes
Calculated Fst values
```

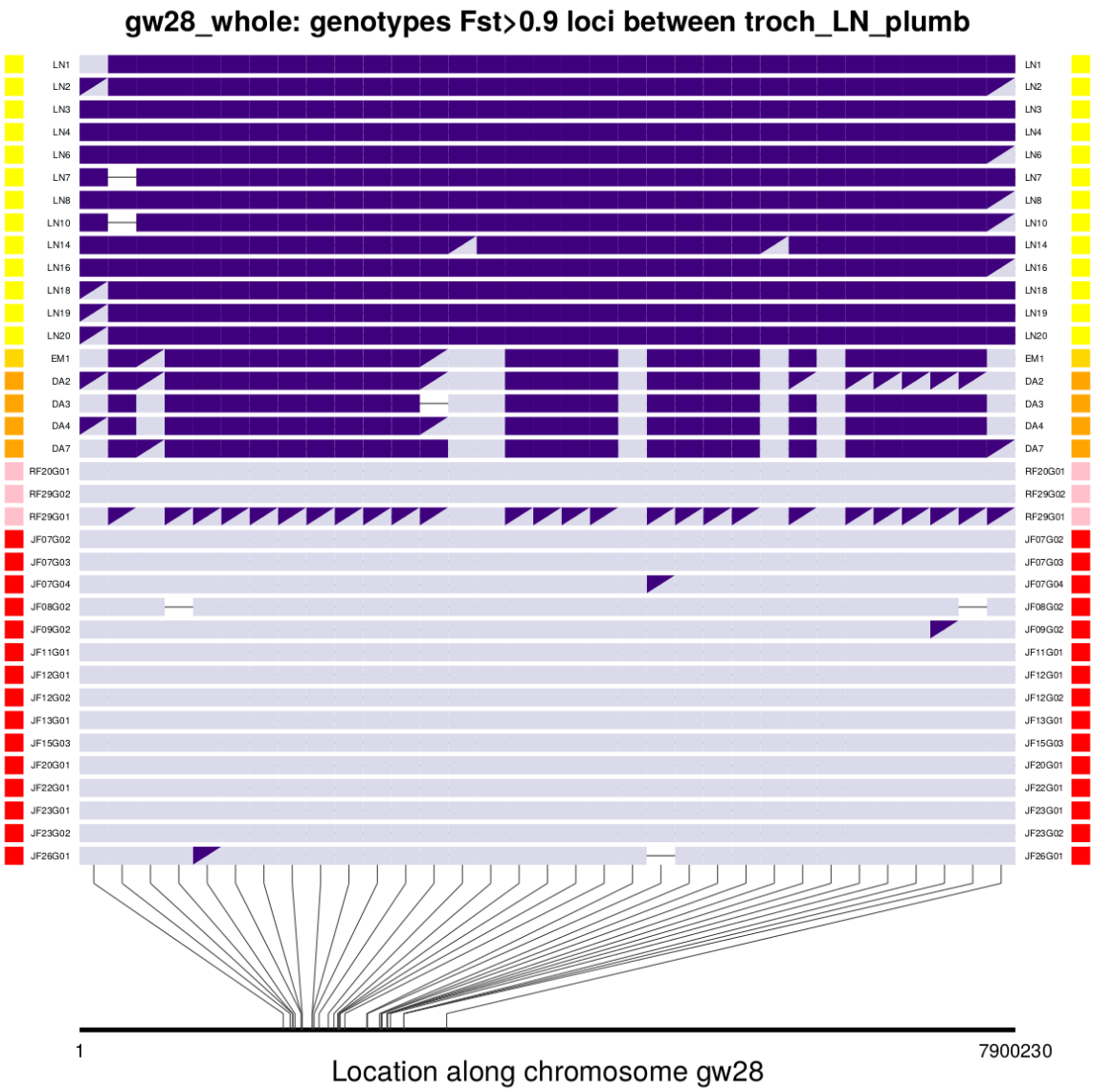


Make a GBI plot to illustrate variation along east side of ring

```
groups = ["troch_LN","obs","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to dete
plotGroups = ["troch_LN","troch_EM","obs","plumb_BJ","plumb"]
plotGroupColors = ["yellow","gold","orange","pink","red"]
numIndsToPlot = [15, 15, 15, 15, 17] # maximum number of individuals to plot from each group
group1 = "troch_LN"   # these groups will determine the color used in the graph
```

```
group2 = "plumb"
groupsToCompare = "troch_LN_plumb"
Fst_cutoff = 0.9
missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among ind

# Calculate allele freqs and sample sizes (use column Fst_group)
freqs, sampleSizes = getFreqsAndSampleSizes(genosOnly_with_missing, ind_with_metadata_indFiltered.Fst
println("Calculated population allele frequencies and sample sizes")

# calculate Fst
Fst, FstNumerator, FstDenominator, pairwiseNamesFst = getFst(freqs, sampleSizes, groups; among=true)
println("Calculated Fst values")

# limit the individuals to include in plot
# For this figure only, filter out individuals with lots of missing genotypes
numMissings_threshold = 800_000
selection = ind_with_metadata_indFiltered.numMissings .< numMissings_threshold
genosOnly_with_missing_selected = view(genosOnly_with_missing, selection, :)
ind_with_metadata_indFiltered_selected = view(ind_with_metadata_indFiltered, selection, :)

# now limit each group to specified numbers
genosOnly_included, ind_with_metadata_included = limitIndsToPlot(plotGroups, numIndsToPlot, genosOnly

selection = Not(ind_with_metadata_included.ind .== "GW_Armando_plate1_JF09G01")

# choose the scaffold and region to show
chr = "gw28"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN) # this gets the r

# need to remove two plumbeitarsus individuals that have introgression from viridanus, as that would
selection = Not(ind_with_metadata_included.ind .∈ Ref(["GW_Armando_plate1_JF09G01", "GW_Armando_plate
ind_with_metadata_included = ind_with_metadata_included[selection,:]
genosOnly_included = genosOnly_included[selection,:]

#### Now actually make the plot
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff, missingFractionAllowed,
    regionInfo, pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_included, ind_with_metadata_included, freqs, plotGroups, plotGroupColors);
# plotInfo contains a tuple with: (f, plottedGenotype, locations, plottedMetadata)
```

```
Calculated population allele frequencies and sample sizes
Calculated Fst values
```

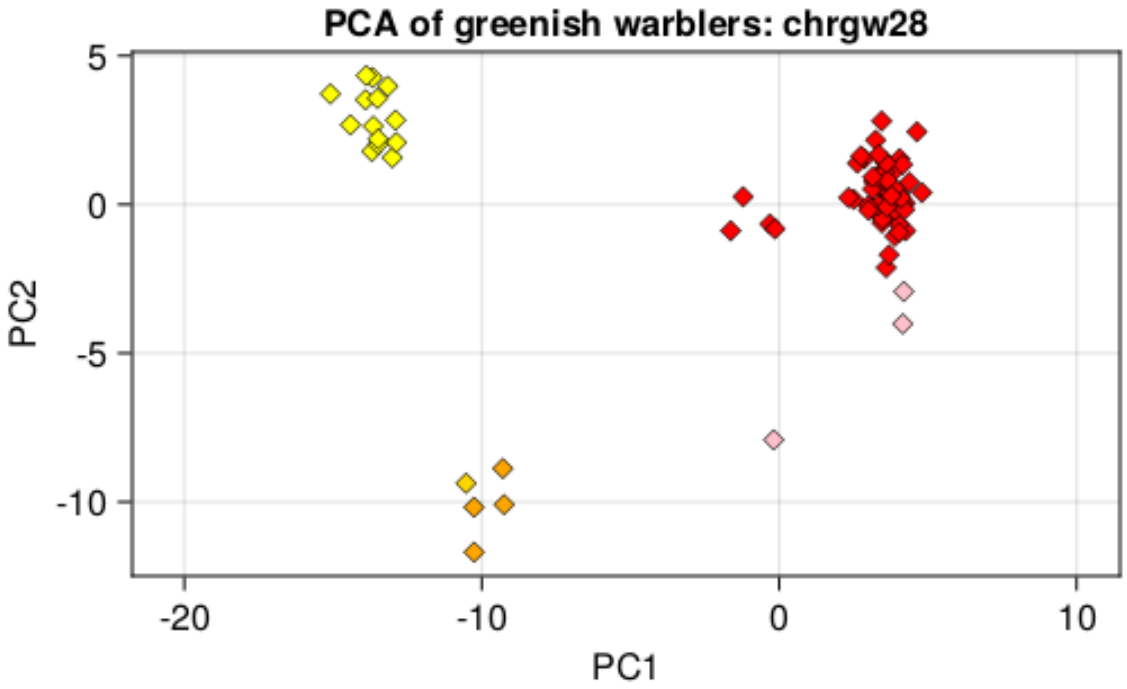


PCA plot of the east side of the ring, for one chromosome

```
eastern_groups_to_plot_PCA = ["troch_LN","troch_EM","obs","plumb_BJ","plumb"]
eastern_group_colors_PCA = ["yellow","gold","orange","pink","red"];
chrom = "gw28"
regionText = string("chr", chrom)
filename = string(baseName, tagName, regionText, ".KNNimputedMissing.jld2")
```

```
imputed_genos = load(filename, "imputed_genos")
ind_with_metadata_indFiltered = load(filename, "ind_with_metadata_indFiltered")
pos_SNP_filtered_region = load(filename, "pos_SNP_filtered_region")
println(string("Loaded ",filename))
println(string(regionText, ": ", size(imputed_genos,2), " SNPs from ", size(imputed_genos,1), " indi
plotPCA(imputed_genos, ind_with_metadata_indFiltered,
        eastern_groups_to_plot_PCA, eastern_group_colors_PCA;
        sampleSet = "greenish warblers", regionText=regionText,
        flip1 = false, flip2 = true)
```

```
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.wh
chrgw28: 11180 SNPs from 257 individuals
```




```
(model = PCA(indim = 11180, outdim = 3, principalratio = 0.16508746), values = Float32[3.4456518 2.861559 … 3
  0 Plots
  1 Child Scene:
    └ Scene (528px, 336px))
```

Calculate distances around ring

The locations around the ring (assuming barrier in North) can be graphed against genomic PC1 (or other variables).

Load lat/long data

```
cd(repoDirectory)
latlong_filepath = "metadata/GW_locations_LatLong_2023.txt"
latlongs = DataFrame(CSV.File(latlong_filepath))
print(latlongs)
```

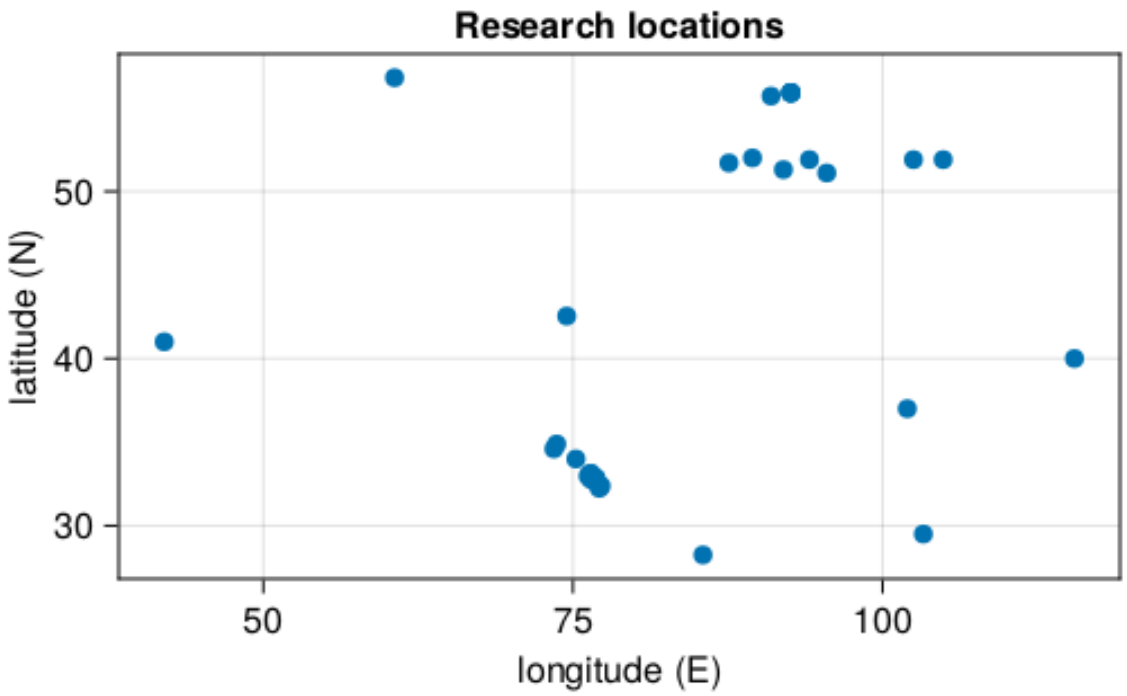
32×5 DataFrame

Row	Location_name	location_short	lat_N	long_E	subspecies
	String31	String7	Float64	Float64	String15
1	Yekaterinburg	YK	56.8	60.6	viridanus
2	Abakan	AB	52.0	89.5	viridanus
3	Teletsk	TL	51.7	87.6	viridanus
4	Stolbi	ST_vi	55.9	92.6	viridanus
5	Krasnoyarski_Krai_vi	KK_vi	55.9	92.6	viridanus
6	Turkey	TU	41.0	42.0	nitidus
7	Ala_Archa	AA	42.54	74.5	viridanus
8	Naran_Pakistan	NR	34.884	73.691	ludlowi
9	Shogran_Pakistan	SH	34.594	73.466	ludlowi
10	Overa_Kashmir	OV	33.991	75.243	ludlowi
11	Satharundhi_ChambaDistrict	SA	32.974	76.222	ludlowi
12	KL_Killar_HP	KL	33.106	76.409	ludlowi
13	Thalighar	TH	32.828	76.45	ludlowi
14	Sural	SR	33.134	76.455	ludlowi
15	PA_Tindi_HP	PA	32.771	76.472	ludlowi
16	Sukhto	SU	32.868	76.855	ludlowi
17	Nainaghar	NG	32.728	76.8594	ludlowi
18	Mooling_and_Keylong	ML	32.508	76.981	ludlowi
19	Manali	MN	32.237	77.13	ludlowi
20	Spiti	SP	32.377	77.281	ludlowi
21	Langtang	LN	28.25	85.5	trochiloides
22	Emeishan	EM	29.5	103.3	obscuratus

23	Xining	XN	37.0	102.0	obscuratus
24	Beijing	BJ	40.0	115.5	plumbeitarsus
25	Baikal	BK	51.9	104.9	plumbeitarsus
26	Arshan	AN	51.9	102.5	plumbeitarsus
27	Ilinka	IL	51.1	95.5	plumbeitarsus
28	Tuva	TA	51.3	92.0	plumbeitarsus
29	Stolbi	ST	55.9	92.6	plumbeitarsus
30	Krasnoyarski_Krai	KK	55.9	92.6	plumbeitarsus
31	Uyukski	UY	51.9	94.1	plumbeitarsus
32	Solgonski	SL	55.7	91.0	plumbeitarsus

Make a quick plot to inspect latlong data:

```
f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "Research locations",
    xlabel = "longitude (E)",
    ylabel = "latitude (N)"
)
scatter!(latlongs.long_E, latlongs.lat_N)
f
```



remove "Green warbler" *nitidus*

Phylloscopus [t.] nitidus is outside of the main ring, so remove these samples from this analysis:

```
latlongs2 = latlongs[Not(latlongs.subspecies .== "nitidus"), :];
print(latlongs2)
```

```
31×5 DataFrame
 Row │ Location_name              location_short  lat_N    long_E    subspecies
     │ String31                   String7         Float64  Float64   String15
─────┼──────────────────────────────────────────────────────────────────────────
   1 │ Yekaterinburg              YK               56.8     60.6     viridanus
   2 │ Abakan                     AB               52.0     89.5     viridanus
   3 │ Teletsk                    TL               51.7     87.6     viridanus
   4 │ Stolbi                     ST_vi            55.9     92.6     viridanus
   5 │ Krasnoyarski_Krai_vi       KK_vi            55.9     92.6     viridanus
   6 │ Ala_Archa                  AA               42.54    74.5     viridanus
```

7	Naran_Pakistan	NR	34.884	73.691	ludlowi
8	Shogran_Pakistan	SH	34.594	73.466	ludlowi
9	Overa_Kashmir	OV	33.991	75.243	ludlowi
10	Satharundhi_ChambaDistrict	SA	32.974	76.222	ludlowi
11	KL_Killar_HP	KL	33.106	76.409	ludlowi
12	Thalighar	TH	32.828	76.45	ludlowi
13	Sural	SR	33.134	76.455	ludlowi
14	PA_Tindi_HP	PA	32.771	76.472	ludlowi
15	Sukhto	SU	32.868	76.855	ludlowi
16	Nainaghar	NG	32.728	76.8594	ludlowi
17	Mooling_and_Keylong	ML	32.508	76.981	ludlowi
18	Manali	MN	32.237	77.13	ludlowi
19	Spiti	SP	32.377	77.281	ludlowi
20	Langtang	LN	28.25	85.5	trochiloides
21	Emeishan	EM	29.5	103.3	obscuratus
22	Xining	XN	37.0	102.0	obscuratus
23	Beijing	BJ	40.0	115.5	plumbeitarsus
24	Baikal	BK	51.9	104.9	plumbeitarsus
25	Arshan	AN	51.9	102.5	plumbeitarsus
26	Ilinka	IL	51.1	95.5	plumbeitarsus
27	Tuva	TA	51.3	92.0	plumbeitarsus
28	Stolbi	ST	55.9	92.6	plumbeitarsus
29	Krasnoyarski_Krai	KK	55.9	92.6	plumbeitarsus
30	Uyukski	UY	51.9	94.1	plumbeitarsus
31	Solgonski	SL	55.7	91.0	plumbeitarsus

Make a matrix of great circle distances

These are Haversine distances, assuming spherical Earth which is really close:

```
geoPoints = GeoLocation.(latlongs2.long_E, latlongs2.lat_N)
# this next line is so neat--uses list comprehension to make a matrix of pairwise calculations
distances = [(HaversineDistance(geoPoints[i], geoPoints[j])/1000) for i in eachindex(geoPoints), j in
```

```
31×31 Matrix{Float64}:
    0.0   1929.03   1829.5    1956.21   …  1956.21   2214.04   1866.47
 1929.03     0.0     134.698   478.643      478.643   315.404   422.996
 1829.5    134.698     0.0     570.583      570.583   447.369   497.776
```

```
1956.21   478.643   570.583     0.0          0.0     455.478   102.442
1956.21   478.643   570.583     0.0          0.0     455.478   102.442
1866.88  1539.57   1417.12   1971.61   …  1971.61   1800.23   1883.0
2629.03  2280.66   2174.21   2744.17      2744.17   2492.37   2664.81
2653.46  2318.81   2212.24   2782.15      2782.15   2530.6    2702.67
2768.57  2304.58   2204.89   2775.3       2775.3    2498.42   2700.28
2905.38  2370.69   2276.33   2845.24      2845.24   2550.34   2773.58
2897.44  2350.4    2256.5    2825.21   …  2825.21   2529.04   2753.85
2927.82  2377.48   2284.21   2852.64      2852.64   2554.28   2781.67
2895.96  2345.88   2252.09   2820.75      2820.75   2524.24   2749.47
   ⋮                                  ⋱                        ⋮
3728.75  2661.7    2613.39   3126.13      3126.13   2725.19   3083.74
4484.94  2747.98   2788.34   3054.87   …  3054.87   2603.84   3071.78
3751.29  1934.14   1987.84   2216.8       2216.8    1768.98   2237.85
4317.27  2390.71   2499.26   2434.21      2434.21   2104.82   2502.82
2869.29  1053.52   1187.02    918.603      918.603   740.322  1003.57
2724.34   889.835  1023.03    785.302      785.302   576.015   863.881
2341.58   426.63    551.809   567.027   …  567.027   131.542   591.859
2117.8    189.217   307.752   513.021      513.021   159.644   493.694
1956.21   478.643   570.583     0.0          0.0     455.478   102.442
1956.21   478.643   570.583     0.0          0.0     455.478   102.442
2214.04   315.404   447.369   455.478      455.478     0.0     468.914
1866.47   422.996   497.776   102.442   …  102.442   468.914     0.0
```

Now adjust distances to assume no gene flow through centre of ring.

```
# get some key distances
function getIndex(name, nameVector = latlongs2.Location_name)
    findfirst(isequal(name), nameVector)
end

index_AA = getIndex("Ala_Archa")
index_PK = getIndex("Naran_Pakistan")
index_LN = getIndex("Langtang")
index_EM = getIndex("Emeishan")
index_XN = getIndex("Xining")
index_BJ = getIndex("Beijing")
index_last = nrow(latlongs2)

dist_PK_to_LN = distances[index_PK, index_LN]
dist_LN_to_EM = distances[index_LN, index_EM]
```

```
dist_EM_to_BJ = distances[index_EM, index_BJ]

# This next part will assume locations in the input file are arranged in order around ring:
distsAroundRing = Matrix{Float32}(undef, size(distances)[1], size(distances)[2])
# accept all distances within viridanus:

# function for accepting straight-line great circle dists as distances between sets of sites
acceptDists = function(straightGreatCircleDists, start, finish, distsAroundRing)
    distsAroundRing[start:finish, start:finish] = straightGreatCircleDists[start:finish, start:finis
    return(distsAroundRing)
end

# accept all distances within viridanus:
distsAroundRing = acceptDists(distances, 1, index_AA, distsAroundRing)

# accept dist from AA to PK:
distsAroundRing = acceptDists(distances, index_AA, index_PK, distsAroundRing)

# accept all distances from PK to LN:
distsAroundRing = acceptDists(distances, index_PK, index_LN, distsAroundRing)

# accept dist from LN to EM:
distsAroundRing = acceptDists(distances, index_LN, index_EM, distsAroundRing)

# accept dists between EM, XN, BJ:
distsAroundRing = acceptDists(distances, index_EM, index_BJ, distsAroundRing)

# accept all distances within plumbeitarsus:
distsAroundRing = acceptDists(distances, index_BJ, index_last, distsAroundRing)

# function for adding up distances measured through certain sites:
addDists = function(set1start, set1end, set2start, set2end, distsAroundRing)
    firstDists = repeat(distsAroundRing[set1start:(set1end-1), set1end], 1, set2end-set2start+1)
    secondDists = repeat(transpose(distsAroundRing[set1end, set2start:set2end]), set1end-set1start, 1
    totalDists = firstDists + secondDists
    distsAroundRing[set1start:(set1end-1), set2start:set2end] = totalDists
    distsAroundRing[set2start:set2end, set1start:(set1end-1)] = transpose(totalDists)
    return(distsAroundRing)
end

# dists from viridanus to PK are sum of dists to AA plus AA to PK:
```

```
distsAroundRing = addDists(1, index_AA, index_PK, index_PK, distsAroundRing)

# dists from "northwest of PK" to Himalayas are sum of ringdists to PK plus PK to locations up to LN
distsAroundRing = addDists(1, index_PK, index_PK+1, index_LN, distsAroundRing)

# dists from "west / northwest of LN" to EM are sum of dists to LN plus LN to EM:
distsAroundRing = addDists(1, index_LN, index_EM, index_EM, distsAroundRing)

# dists from "west / northwest of EM" to China are sum of dists to EM plus EM to (XN, BJ):
distsAroundRing = addDists(1, index_EM, index_XN, index_BJ, distsAroundRing)

# dists from "west of BJ" to east Siberia are sum of dists to BJ plus BJ to other plumbeitarsus:
distsAroundRing = addDists(1, index_BJ, index_BJ+1, index_last, distsAroundRing);
```

Do Principal Coordinates Analysis on the distances around the ring

This produces a single location axis around ring, going from west Siberia south, then east, then north to east Siberia.

```
PCO_around_ring = fit(MDS, distsAroundRing; distances=true, maxoutdim=1)
# add this as a column to the data frame:
latlongs2.LocationAroundRing = vec(-predict(PCO_around_ring))
# another way:
# latlongs2[:, :LocationAroundRing] = vec(-predict(PCO_around_ring))
latlongs2[:, [:location_short, :LocationAroundRing]]
println(latlongs2[:, [:location_short, :LocationAroundRing]])
```

```
31×2 DataFrame
 Row │ location_short  LocationAroundRing
     │ String7         Float32
─────┼──────────────────────────────────
   1 │ YK                        -4484.22
   2 │ AB                        -4223.99
   3 │ TL                        -4104.65
   4 │ ST_vi                     -4655.8
   5 │ KK_vi                     -4655.8
   6 │ AA                        -2705.3
   7 │ NR                        -1849.44
   8 │ SH                        -1844.6
```

9	OV	-1675.97
10	SA	-1539.62
11	KL	-1532.33
12	TH	-1513.07
13	SR	-1530.28
14	PA	-1508.06
15	SU	-1483.0
16	NG	-1474.58
17	ML	-1452.26
18	MN	-1424.96
19	SP	-1420.66
20	LN	-507.711
21	EM	1233.65
22	XN	1914.05
23	BJ	2846.22
24	BK	4390.92
25	AN	4498.45
26	IL	4817.46
27	TA	5039.28
28	ST	5276.86
29	KK	5276.86
30	UY	4946.53
31	SL	5346.01

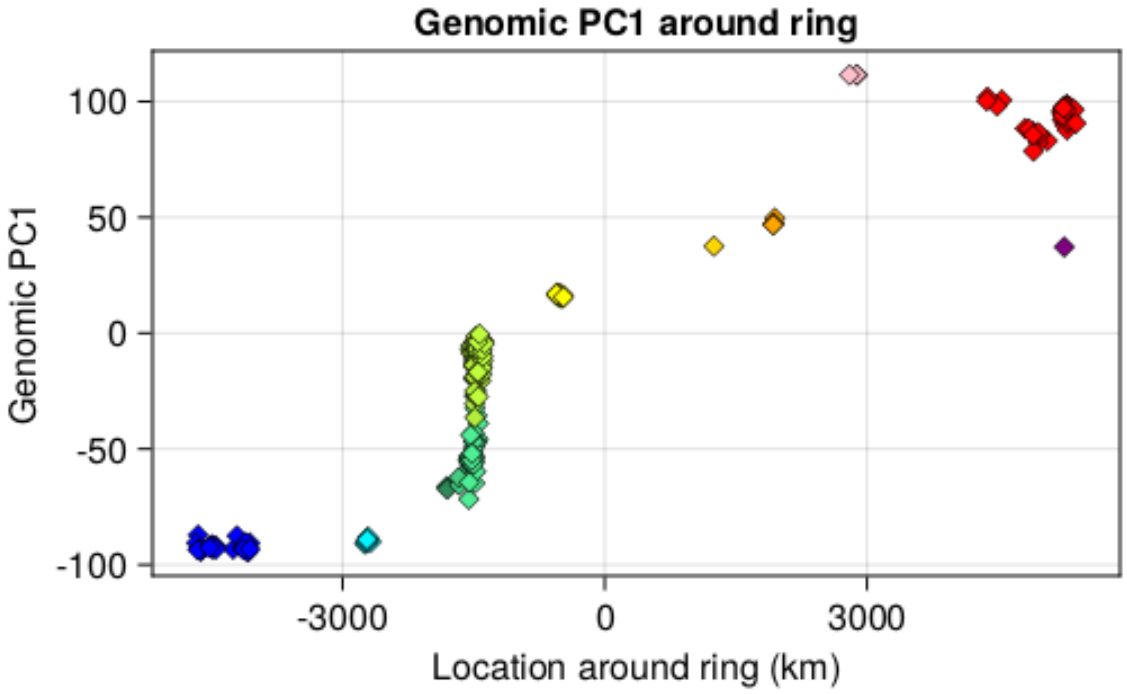
Add these ring locations to the metadata table:

```
ind_with_metadata_indFiltered_wholeGenome.ring_km .= NaN  # pre-allocate the column
for i in axes(latlongs2, 1)
    match_indices = findall(ind_with_metadata_indFiltered_wholeGenome.location .== latlongs2.locatio
    ind_with_metadata_indFiltered_wholeGenome.ring_km[match_indices] .= latlongs2.LocationAroundRing
end
```

Plot location around ring vs. PC1:

```
# plot(ind_with_metadata_indFiltered_wholeGenome.ring_km, ind_with_metadata_indFiltered_wholeGenome.
f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "Genomic PC1 around ring",
```

```
    xlabel = "Location around ring (km)",
    ylabel = "Genomic PC1"
)
jitterSize = 100   # in km
x_plot_values = ind_with_metadata_indFiltered_wholeGenome.ring_km .+ jitterSize .* (rand(length(ind_w
y_plot_values = ind_with_metadata_indFiltered_wholeGenome.PC1
for i in eachindex(groups_to_plot_PCA)
    selection = ind_with_metadata_indFiltered_wholeGenome.Fst_group .== groups_to_plot_PCA[i]
    CairoMakie.scatter!(ax, x_plot_values[selection], y_plot_values[selection], marker = :diamond, co
end
display(f);
```



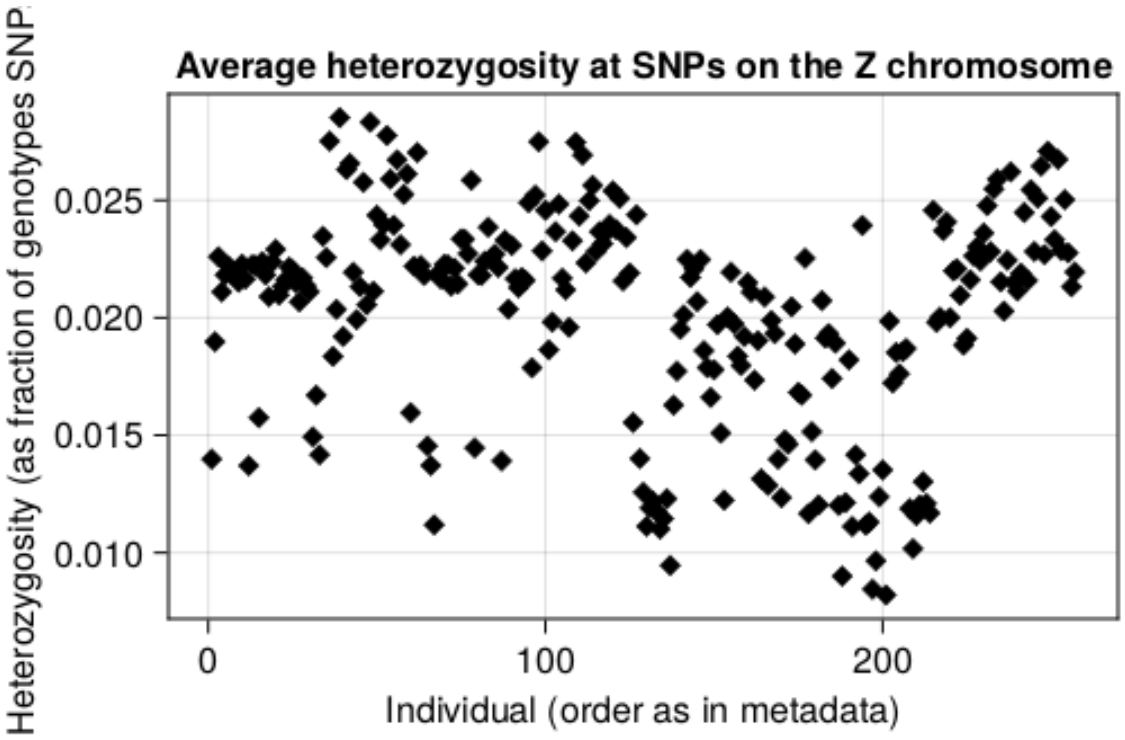
Z chromosome differentiation

The PCA of Z chromosome variation (see far above) showed some unexpected structure, which I think it likely due to females being shifted (especially on PC2) compared to males. I think this is likely due to females having W chromosomes, reads from which are sometimes incorrectly mapped to the Z. Hence a small fraction of SNPs differ between females and males (with

females called as heterozygotes, males as homozygotes, at those SNPs). However, at loci that are only on the Z, males can be heterozygous (2 alleles) whereas females can only be hemizygous (one allele). I am not sure of the net effect.

To remove this problem, we could take several approaches (e.g., remove problematic SNPs, or look at one sex at a time). Here we'll plot only males as that will remove the W chromosome. To infer sex of the birds, we'll try plotting heterozygosity of Z chromosome SNPs per bird.

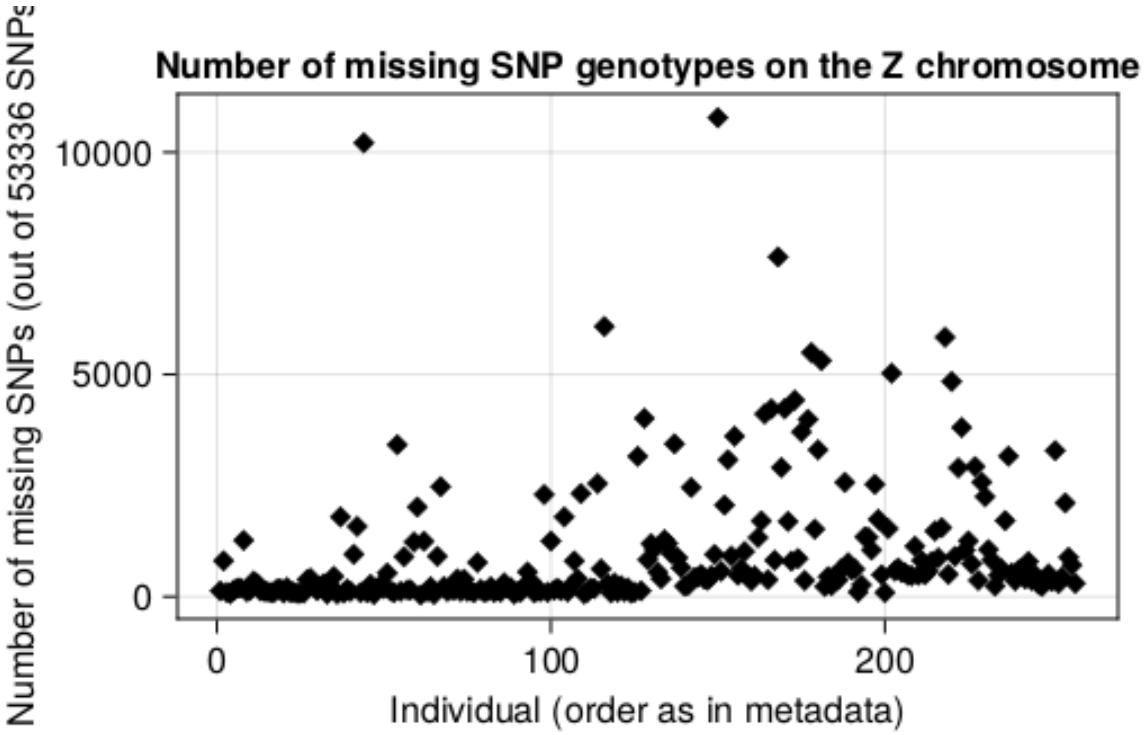
```
chrom = "gwZ"
regionText = string("chr", chrom)
loci_selection = (pos_SNP_filtered.chrom .== chrom)
pos_SNP_filtered_gwZ = pos_SNP_filtered[loci_selection,:]
genotypes_gwZ = genosOnly[:,loci_selection]
numHetSNPs = sum(genotypes_gwZ .== 1, dims=2)
numGenotypedSNPs = sum(map(in([0, 1, 2]), genotypes_gwZ), dims=2)
hetFraction = numHetSNPs ./ numGenotypedSNPs
ind_with_metadata_indFiltered[!, :hetFractionZ] .= hetFraction

f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "Average heterozygosity at SNPs on the Z chromosome",
    xlabel = "Individual (order as in metadata)",
    ylabel = "Heterozygosity (as fraction of genotypes SNPs)"
)
x_plot_values = eachindex(ind_with_metadata_indFiltered.hetFractionZ)
y_plot_values = ind_with_metadata_indFiltered.hetFractionZ
CairoMakie.scatter!(ax, x_plot_values, y_plot_values, marker = :diamond, color="black", markersize=1
display(f);
```



That doesn't provide super clear separation.

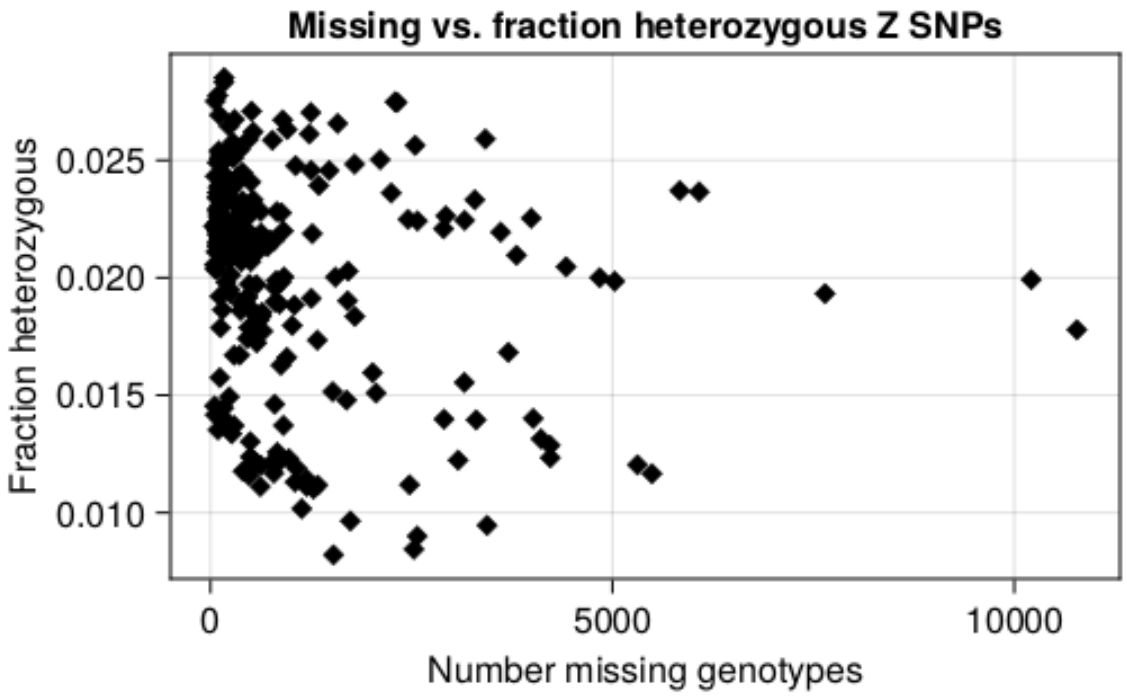
Let's try looking at number of missing genotypes:

```
missingGenotypeCount = sum(genotypes_gwZ .== -1, dims=2)
ind_with_metadata_indFiltered[!, :missingZgenotypeCount] .= missingGenotypeCount
f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "Number of missing SNP genotypes on the Z chromosome",
    xlabel = "Individual (order as in metadata)",
    ylabel = "Number of missing SNPs (out of 53336 SNPs)"
)
x_plot_values = eachindex(ind_with_metadata_indFiltered.missingZgenotypeCount)
y_plot_values = ind_with_metadata_indFiltered.missingZgenotypeCount
CairoMakie.scatter!(ax, x_plot_values, y_plot_values, marker = :diamond, color="black", markersize=1
display(f);
```



Now let's plot those together:

```
f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "Missing vs. fraction heterozygous Z SNPs",
    xlabel = "Number missing genotypes",
    ylabel = "Fraction heterozygous"
)
x_plot_values = ind_with_metadata_indFiltered.missingZgenotypeCount
y_plot_values = ind_with_metadata_indFiltered.hetFractionZ
CairoMakie.scatter!(ax, x_plot_values, y_plot_values, marker = :diamond, color="black", markersize=1
display(f);
```



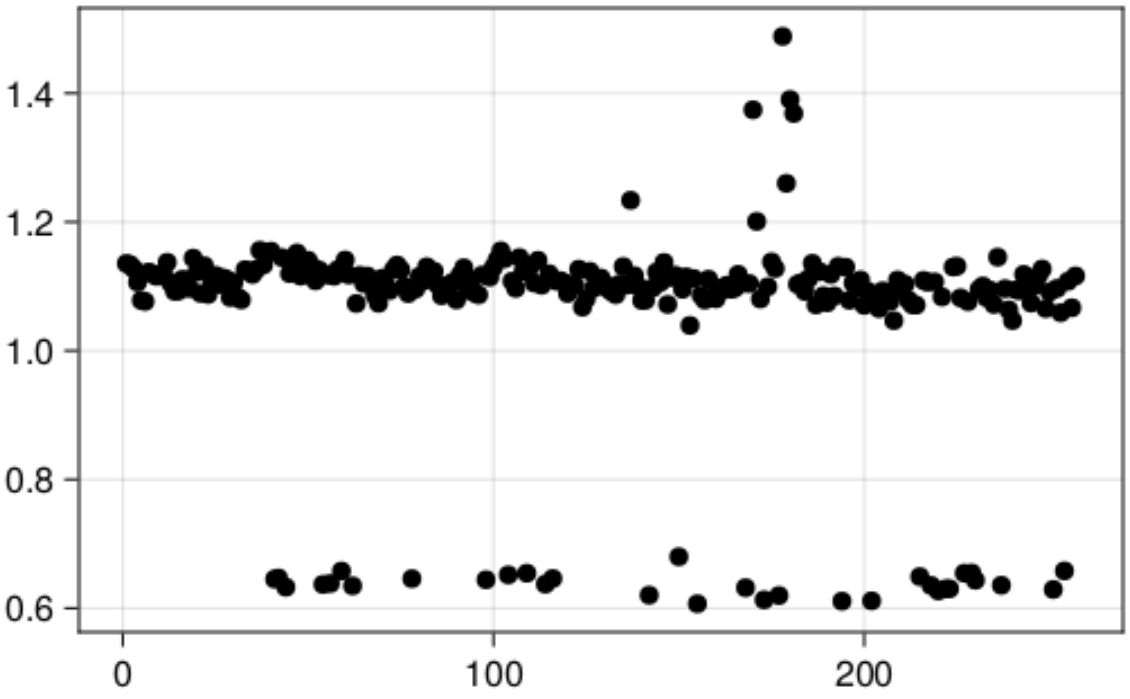
I don't think number missing is very helpful–probably similar in females and males.

I realized that Z-chromosome read depth is probably the best way I presently have to infer sex of individuals. So I used vcftools to compare average SNP read depth on chr gwZ and the largest of the autosomes, gw2. These commands were run in the Terminal (I've simplified the file paths here):

```
vcftools --vcf GW2022_all4plates.genotypes.SNPs_only.whole_genome.max2allele_noindel.vcf.maxmiss60.MQ20.

# After filtering, kept 310 out of 310 Individuals
# Outputting Mean Depth by Individual
# After filtering, kept 134926 out of a possible 2431709 Sites
# Run Time = 155.00 seconds

vcftools --vcf GW2022_all4plates.genotypes.SNPs_only.whole_genome.max2allele_noindel.vcf.maxmiss60.MQ20.

# After filtering, kept 310 out of 310 Individuals
# Outputting Mean Depth by Individual
# After filtering, kept 229227 out of a possible 2431709 Sites
# Run Time = 155.00 seconds
```

The plan is to load these read depth files into Julia and determine sex of individuals from the ratio of read depth on chromosome Z to chromosome 2:

```
filename_gwZ = "metadata/GW2022_all4plates.genotypes.SNPs_only.chrgwZ.max2allele_noindel.vcf.maxmiss
readDepthZ = DataFrame(CSV.File(filename_gwZ))

filename_gw2 = "metadata/GW2022_all4plates.genotypes.SNPs_only.chrgw2.max2allele_noindel.vcf.maxmiss
readDepth2 = DataFrame(CSV.File(filename_gw2))

readDepthRatioZto2 = innerjoin(readDepthZ, readDepth2, on = :INDV, renamecols = "_gwZ" ⇒ "_gw2")

readDepthRatioZto2[!, :depthRatio] = readDepthRatioZto2.MEAN_DEPTH_gwZ ./ readDepthRatioZto2.MEAN_DE

ind_with_metadata_indFiltered_sex = leftjoin(ind_with_metadata_indFiltered, readDepthRatioZto2, on =

plot(ind_with_metadata_indFiltered_sex.depthRatio)
```

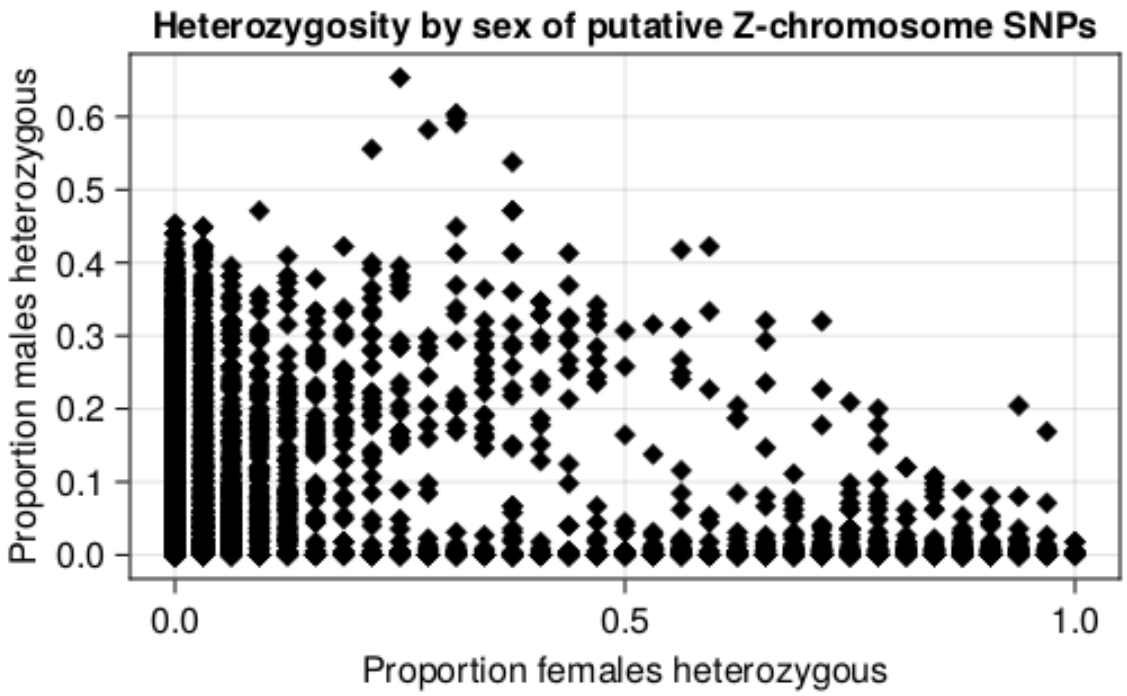


That is quite a clear difference. The cluster with the lower Z / autosome read depth ratio should correspond to females (one copy of Z), whereas the higher Z / autosome ratio should be males. Interestingly, the ratios are a little higher than the expected 0.5 and 1, likely due to

some repetitive elements (on the Z) and some W fragments being mapped to Z. Additional, the pseudoautosomal region, which is homologous and recombines between Z and W, will drive up Z read depth in females.

It is these SNPs with a W copy that I will now try to detect and remove. They should show high heterozygosity in females.

```
ind_with_metadata_indFiltered_sex.sex .= "na"
females = ind_with_metadata_indFiltered_sex.depthRatio .< 0.8
ind_with_metadata_indFiltered_sex.sex[females] .= "F"
num_females = sum(females)
females_genotypes_gwZ = view(genotypes_gwZ, females, :)
numHetsPerSNP_females = vec(sum(females_genotypes_gwZ .== 1, dims=1))
female_heterozygosity = numHetsPerSNP_females ./ num_females
CairoMakie.hist(female_heterozygosity)
CairoMakie.plot(female_heterozygosity)

# compare with males:
males = ind_with_metadata_indFiltered_sex.depthRatio .> 0.9
ind_with_metadata_indFiltered_sex.sex[males] .= "M"
num_males = sum(males)
males_genotypes_gwZ = view(genotypes_gwZ, males, :)
numHetsPerSNP_males = vec(sum(males_genotypes_gwZ .== 1, dims=1))
male_heterozygosity = numHetsPerSNP_males ./ num_males
CairoMakie.hist(male_heterozygosity)
CairoMakie.plot(male_heterozygosity)

CairoMakie.scatter(vec(numHetsPerSNP_females ./ num_females), vec(numHetsPerSNP_males ./ num_males),
    xlabel = "Proportion females heterozygous",
    ylabel = "Proportion males heterozygous")

f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "Heterozygosity by sex of putative Z-chromosome SNPs",
    xlabel = "Proportion females heterozygous",
    ylabel = "Proportion males heterozygous"
)
x_plot_values = female_heterozygosity
y_plot_values = male_heterozygosity
CairoMakie.scatter!(ax, x_plot_values, y_plot_values, marker = :diamond, color="black", markersize=10
display(f);
```



This really clarifies things. The SNPs in the middle of the graph, with similar heterozygosity in females and males, are consistent with being pseudoautosomal SNPs. But the SNPs that have much higher female heterozygosity than male heterozygosity could be non-pseudoautosomal SNPs with reads from the W mapping to the Z, and giving appearance of Z heterozygosity in females (and not males). It is these SNPs we should remove. Also, there are a few with > 0.5 heterozygosity in males, which should be removed because they are likely due to paralogs.

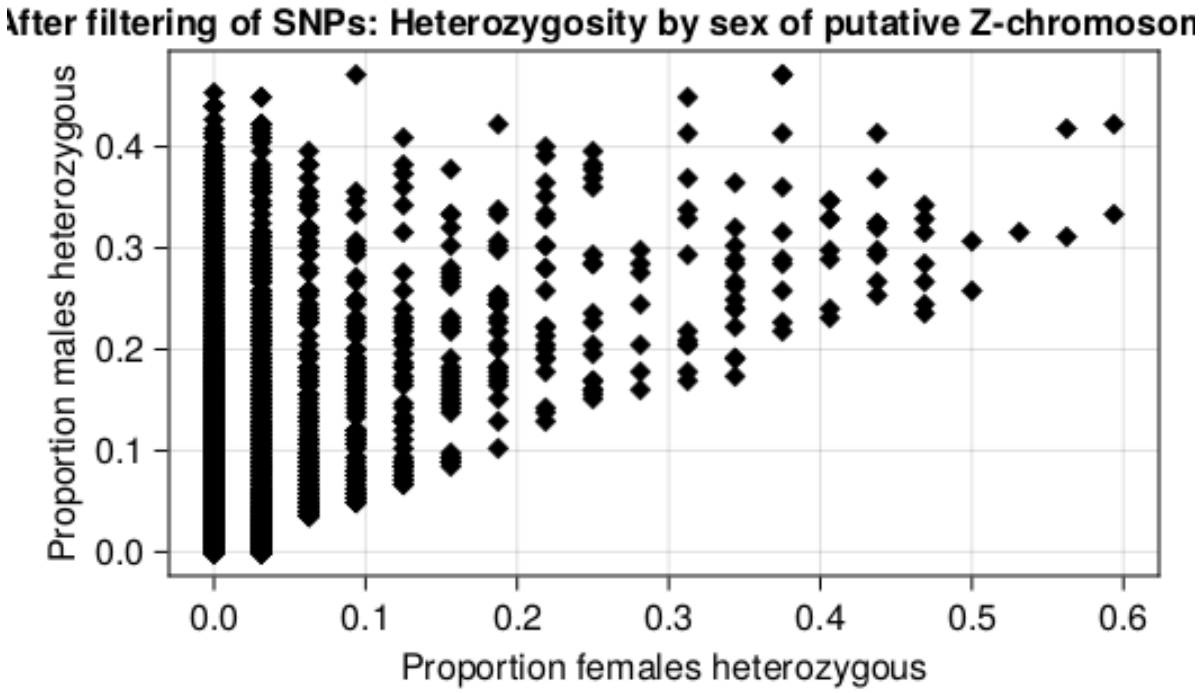
So, going to remove SNPs according to these rules: If (female heterozygosity is above 0.05) AND (ratio of male to female heterozygosity is less than 1/2), REMOVE the SNP from consideration. If (male heterozygosity is above 0.5), REMOVE the SNP.

```
removed_SNPs = (male_heterozygosity .> 0.5) .||
               ((female_heterozygosity .> 0.05) .&&
               ((male_heterozygosity ./ female_heterozygosity) .< 0.5 ))
genotypes_gwZ_SNPfiltered = genotypes_gwZ[:, Not(removed_SNPs)]
pos_SNP_filtered_gwZ_SNPfiltered = pos_SNP_filtered_gwZ[Not(removed_SNPs),:]
println("This filtering removed ", sum(removed_SNPs), " SNPs.")
```

```
This filtering removed 1831 SNPs.
```

Check that the filtering did remove the problematic SNPs:

```
females_genotypes_gwZ = view(genotypes_gwZ_SNPfiltered, females, :)
numHetsPerSNP_females = vec(sum(females_genotypes_gwZ .== 1, dims=1))
female_heterozygosity = numHetsPerSNP_females ./ num_females
CairoMakie.hist(female_heterozygosity)
CairoMakie.plot(female_heterozygosity)

# compare with males:
males_genotypes_gwZ = view(genotypes_gwZ_SNPfiltered, males, :)
numHetsPerSNP_males = vec(sum(males_genotypes_gwZ .== 1, dims=1))
male_heterozygosity = numHetsPerSNP_males ./ num_males
CairoMakie.hist(male_heterozygosity)
CairoMakie.plot(male_heterozygosity)

CairoMakie.scatter(vec(numHetsPerSNP_females ./ num_females), vec(numHetsPerSNP_males ./ num_males),
    xlabel = "Proportion females heterozygous",
    ylabel = "Proportion males heterozygous")

f = CairoMakie.Figure()
ax = Axis(f[1, 1],
    title = "After filtering of SNPs: Heterozygosity by sex of putative Z-chromosome SNPs",
    xlabel = "Proportion females heterozygous",
    ylabel = "Proportion males heterozygous"
)
x_plot_values = female_heterozygosity
y_plot_values = male_heterozygosity
CairoMakie.scatter!(ax, x_plot_values, y_plot_values, marker = :diamond, color="black", markersize=1
display(f);
```

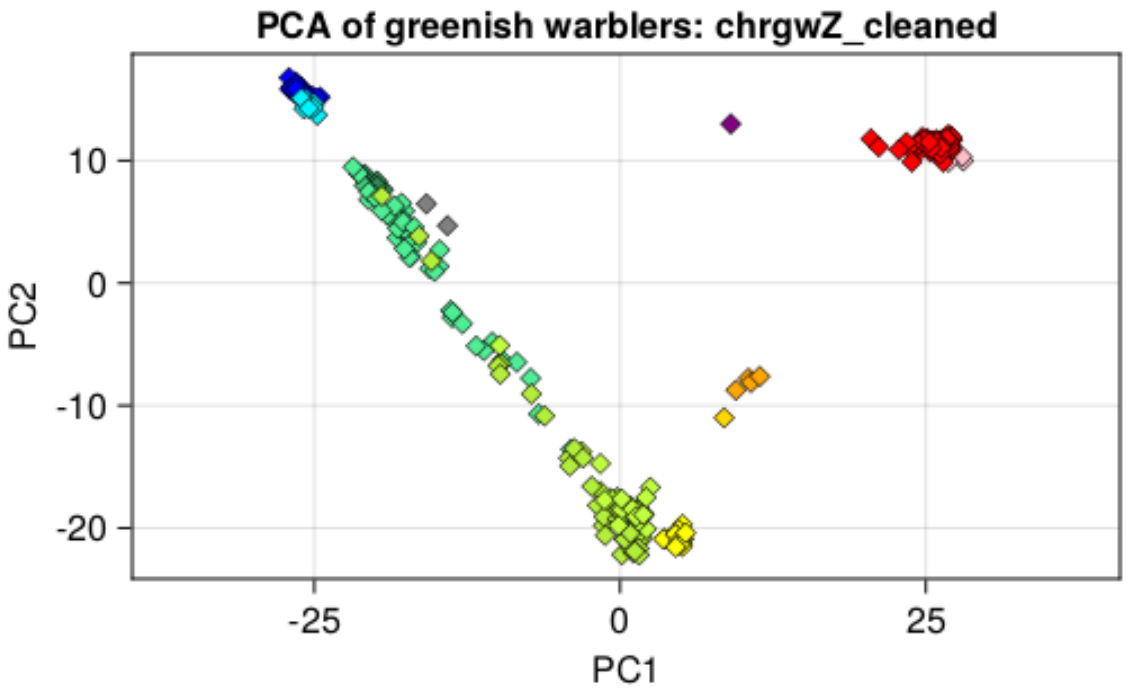


Looks good. Now impute and do the PCA. I did the imputing previously (to do it again, set `do_imputing = true` below).

```
genotypes_gwZ_SNPfiltered_with_missing = Matrix{Union{Missing, Float32}}(genotypes_gwZ_SNPfiltered)
# change "-1" to "missing":
genotypes_gwZ_SNPfiltered_with_missing[genotypes_gwZ_SNPfiltered_with_missing .== -1] .= missing;
regionText = "chrgwZ_cleaned"
filename = string(baseName, tagName, regionText, ".KNNimputedMissing.jld2")
# to do the imputing, do this by setting to true:
cd("/Users/darrenirwin/Dropbox/Darren's current work/")
do_imputing = false
if do_imputing
    @time imputed_genos = Impute.knn(genotypes_gwZ_SNPfiltered_with_missing; k=1, dims=:rows)
    # took 46 sec
    jldsave(filename; imputed_genos, ind_with_metadata_indFiltered = ind_with_metadata_indFiltered_s
    imputed_genos_chrZcleaned = imputed_genos
    ind_with_metadata_indFiltered_sex_chrZcleaned = ind_with_metadata_indFiltered_sex
    pos_SNP_filtered_chZcleaned = pos_SNP_filtered_gwZ_SNPfiltered
    print("Saved matrix of real and imputed Z chromosome genotypes. \n")
```

```
else # load the already saved imputing
    imputed_genos_chrZcleaned = load(filename, "imputed_genos")
    ind_with_metadata_indFiltered_sex_chrZcleaned = load(filename, "ind_with_metadata_indFiltered")
    pos_SNP_filtered_chZcleaned = load(filename, "pos_SNP_filtered_region")
    println(string("Loaded ",filename))
    println(string(regionText, ": ", size(imputed_genos_chrZcleaned, 2), " SNPs from ", size(imputed
end
plotPCA(imputed_genos_chrZcleaned, ind_with_metadata_indFiltered_sex_chrZcleaned,
            groups_to_plot_PCA, group_colors_PCA;
            sampleSet = "greenish warblers", regionText=regionText,
            flip1 = true, flip2 = true)
```

```
Loaded GW_genomics_2022_with_new_genome/GW2022_GBS_012NA_files/GW2022_all4plates.genotypes.SNPs_only.who
chrgwZ_cleaned: 51505 SNPs from 257 individuals
```




```
(model = PCA(indim = 51505, outdim = 3, principalratio = 0.29669666), values = Float32[26.49374 -27.183502 …
  0 Plots
  1 Child Scene:
    └ Scene (528px, 336px))
```

Looks Great!! Now have a Z chromosome PCA with both males and females, based on only Z-chromosome markers.

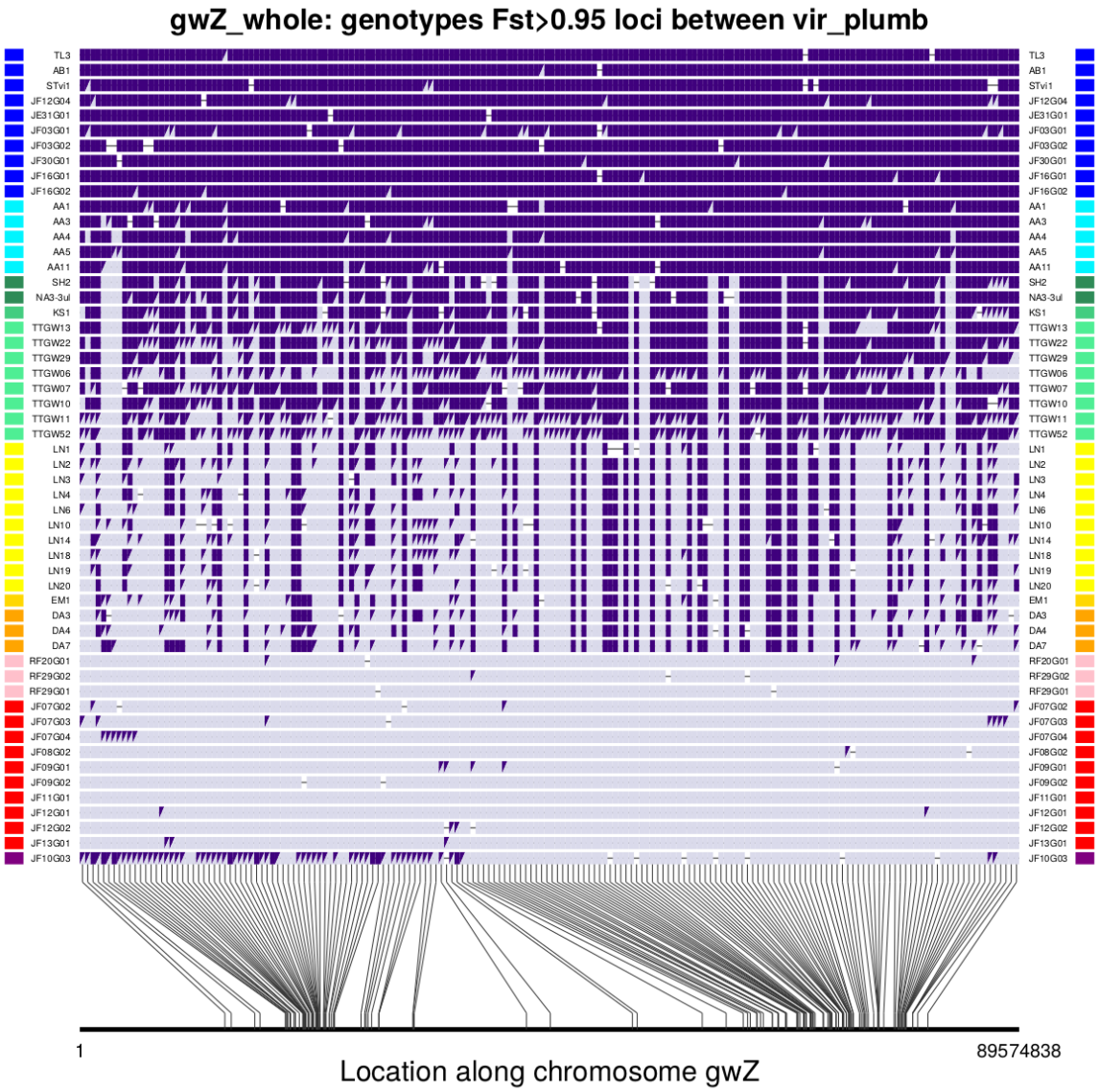
Genotype-by-individual plots for Z chromosome

```
set = "55_inds_around_ring"  #"east_side_of_ring"    #"67_inds_around_ring"  # "west_side_of_ring"

if set == "55_inds_around_ring"
    groups = ["vir","troch_LN","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["vir","vir_S","lud_PK","lud_KS","lud_central","troch_LN","troch_EM","obs","plumb_BJ
    plotGroupColors = ["blue","turquoise1","seagreen4","seagreen3","seagreen2","yellow","gold","orang
    numIndsToPlot = [10, 5, 2, 1, 8, 10, 1, 4, 3, 10, 1] # maximum number of individuals to plot fro
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "vir_plumb"   #"Fst_among"  #"vir_troch_LN"       #"vir_plumb"      #"troch_LN
    Fst_cutoff = 0.95
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "37_inds_around_ring_plusAllVirPlumb"
    groups = ["vir","troch_LN","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["vir","lud","troch_LN","troch_EM","obs", "plumb_BJ","plumb"]
    plotGroupColors = ["blue","seagreen4","yellow","gold","orange", "pink","red"]
    numIndsToPlot = [100, 15, 15, 15, 15, 15, 100] # maximum number of individuals to plot from each
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "Fst_among"
    Fst_cutoff = 0.7
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "west_side_of_ring"
    groups = ["vir","troch_LN"] # for purpose of calculating pairwise Fst and Fst_group (to determine
    plotGroups = ["vir","vir_misID","vir_S","nit", "lud_PK", "lud_KS", "lud_central", "lud_Sath", "lu
    plotGroupColors = ["blue","blue","turquoise1","grey","seagreen4","seagreen3","seagreen2","olived
    numIndsToPlot = [15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15] # maximum number of individuals to p
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "troch_LN"
    groupsToCompare = "vir_troch_LN" # "Fst_among"
    Fst_cutoff = 0.6
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "all_ludlowi_plus_a_few_other"
    groups = ["vir","troch_LN","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["vir","vir_S","nit", "lud_PK", "lud_KS", "lud_central", "lud_Sath", "lud_ML","troch
    plotGroupColors = ["blue","turquoise1","grey","seagreen4","seagreen3","seagreen2","olivedrab3","c
```

```
    numIndsToPlot = [4, 4, 4, 1000, 1000, 1000, 1000, 1000, 1000, 4, 4] # maximum number of individua
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "troch_LN"
    groupsToCompare = "vir_troch_LN" # "Fst_among"
    Fst_cutoff = 0.6
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "east_side_of_ring"
    groups = ["troch_LN","obs","plumb"] # for purpose of calculating pairwise Fst and Fst_group (to c
    plotGroups = ["troch_LN","troch_EM","obs","plumb_BJ","plumb"]
    plotGroupColors = ["yellow","gold","orange","pink","red"]
    numIndsToPlot = [15, 15, 15, 15, 15] # maximum number of individuals to plot from each group
    group1 = "troch_LN"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "troch_LN_plumb"
    Fst_cutoff = 0.7
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
elseif set == "vir_plumb"
    groups = ["vir","plumb"]
    plotGroups = ["vir","plumb_vir","plumb"]
    plotGroupColors = ["blue","purple","red"]
    numIndsToPlot = [100,100,100] # maximum number of individuals to plot from each group
    group1 = "vir"   # these groups will determine the color used in the graph
    group2 = "plumb"
    groupsToCompare = "vir_plumb"
    Fst_cutoff =  0.7
    missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among
end

# Calculate allele freqs and sample sizes (use column Fst_group)
freqs, sampleSizes = getFreqsAndSampleSizes(genotypes_gwZ_SNPfiltered_with_missing, ind_with_metadata
println("Calculated population allele frequencies and sample sizes")

Fst, FstNumerator, FstDenominator, pairwiseNamesFst = getFst(freqs, sampleSizes, groups; among=true)
println("Calculated Fst values")

# For this Z-chromosome genotype-by-individual plot, include only males (so 2 Z chromosomes) and filt
numMissings_threshold = 800_000
selection = (ind_with_metadata_indFiltered_sex_chrZcleaned.numMissings .< numMissings_threshold) .&&
genotypes_gwZ_SNPfiltered_with_missing_selected = view(genotypes_gwZ_SNPfiltered_with_missing, selec
ind_with_metadata_indFiltered_sex_chrZcleaned_selected = view(ind_with_metadata_indFiltered_sex_chrZc
```

```
# now limit each group to specified numbers
genosOnly_included, ind_with_metadata_included = limitIndsToPlot(plotGroups, numIndsToPlot, genotypes

chr = "gwZ"
regionInfo = chooseChrRegion(pos_SNP_filtered_chZcleaned, chr; positionMin=1, positionMax=NaN) # this

plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff, missingFractionAllowed,
    regionInfo, pos_SNP_filtered_chZcleaned, Fst, pairwiseNamesFst,
    genosOnly_included, ind_with_metadata_included, freqs, plotGroups, plotGroupColors);
# plotInfo contains a tuple with: (f, plottedGenotype, locations, plottedMetadata)
```

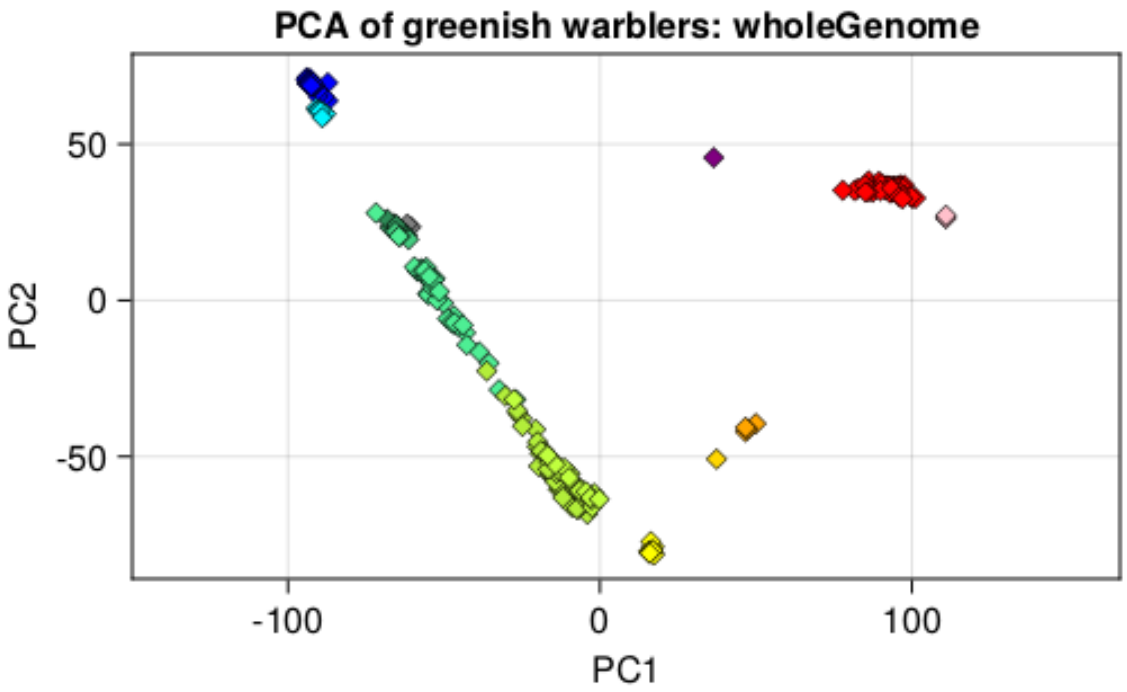
```
Calculated population allele frequencies and sample sizes
Calculated Fst values
```



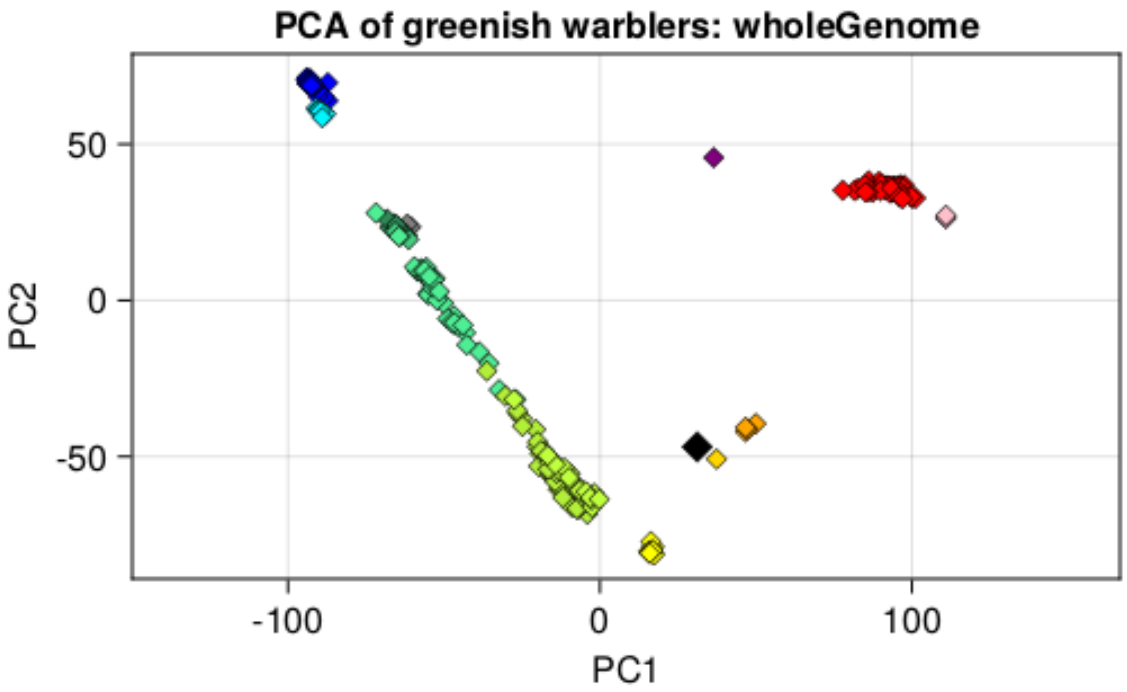
Make final whole-genome PCA

Now that the Z-chromosome problem has been solved (by removing SNPs that had a divergent W sequence mapped onto that Z location), we can construct a new whole-genome PCA. I will combine the saved imputed genotypes for each chromosome into a large data matrix, and conduct PCA on that:

```
chromosomes_to_process[chromosomes_to_process .== "gwZ"] .= "gwZ_cleaned"
# initialize data structures for gneotypes and positions
genos_imputed_loaded = Matrix{Union{Missing, Float32}}(undef, nrow(ind_with_metadata_indFiltered), 0
pos_SNP_loaded = DataFrame(chrom = String[], position = Int64[])
for i in eachindex(chromosomes_to_process)
    chrom = chromosomes_to_process[i]
    regionText = string("chr", chrom)
    filename = string(baseName, tagName, regionText, ".KNNimputedMissing.jld2")
    imputed_genos_one_chr = load(filename, "imputed_genos")
    genos_imputed_loaded = hcat(genos_imputed_loaded, imputed_genos_one_chr)
    if ind_with_metadata_indFiltered.ind != load(filename, "ind_with_metadata_indFiltered")[:, :ind]
        println("""Warning: "ind" columns in loaded data and memory data don't match.""")
    end
    pos_SNP_filtered_region = load(filename, "pos_SNP_filtered_region")
    pos_SNP_loaded = vcat(pos_SNP_loaded, pos_SNP_filtered_region)
    # println(string("Loaded ",filename))
    # println(string(regionText, ": ", size(imputed_genos_one_chr,2), " SNPs from ", size(imputed_ge
end
flipPC1 = true
flipPC2 = true
PCA_wholeGenome = plotPCA(genos_imputed_loaded, ind_with_metadata_indFiltered,
        groups_to_plot_PCA, group_colors_PCA;
        sampleSet = "greenish warblers", regionText = "wholeGenome",
        flip1 = flipPC1, flip2 = flipPC2)
totalObservationVariance = var(PCA_wholeGenome.model)
PC1_variance, PC2_variance = principalvars(PCA_wholeGenome.model)[1:2]
PC1_prop_variance = PC1_variance / totalObservationVariance
PC2_prop_variance = PC2_variance / totalObservationVariance
println("PC1 explains ", 100*PC1_prop_variance, "% of the total variance.
PC2 explains ", 100*PC2_prop_variance, "%.")

# add position of reference genome
refGenomePCAposition = predict(PCA_wholeGenome.model, zeros(size(genos_imputed_loaded, 2)))
flipPC1 && (refGenomePCAposition[1] *= -1)  # this flips PC1 if flipPC1 = true
flipPC2 && (refGenomePCAposition[2] *= -1)  # same for PC2
CairoMakie.scatter!(refGenomePCAposition[1], refGenomePCAposition[2], marker = :diamond, color="blac
try
    display(PCA_wholeGenome.PCAfig)
catch
    println("NOTICE: Figure for ", regionText, " could not be shown due to an unknown error.")
end
```



PC1 explains 12.106356% of the total variance.
PC2 explains 6.406127%.

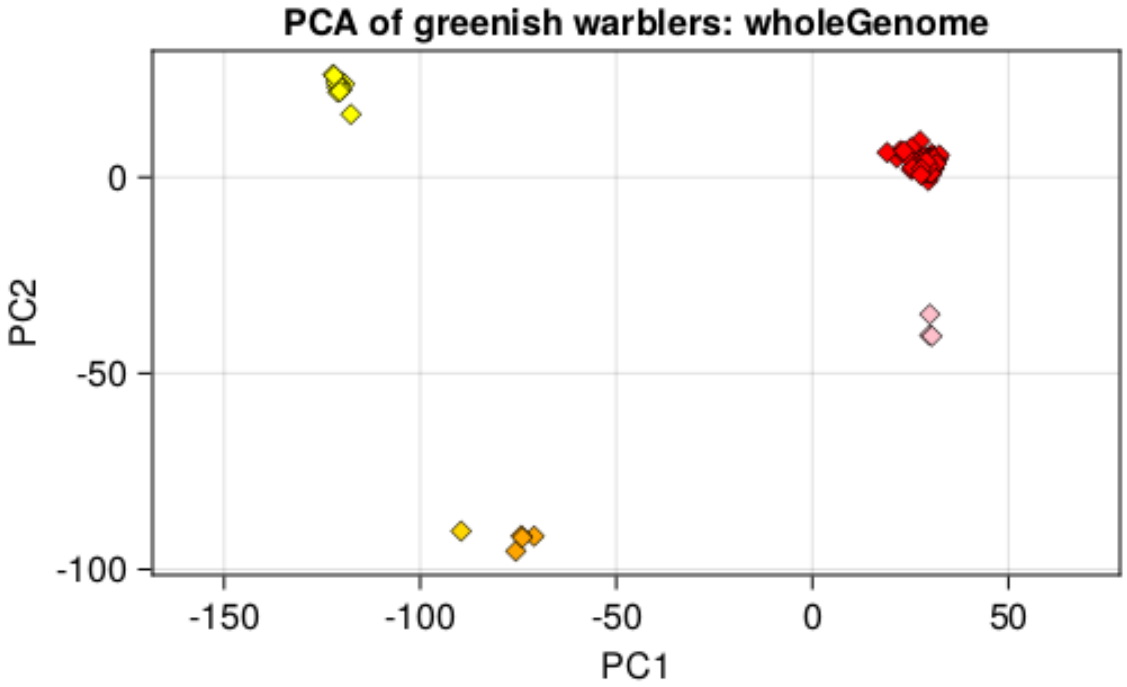



```
CairoMakie.Screen{IMAGE}
```

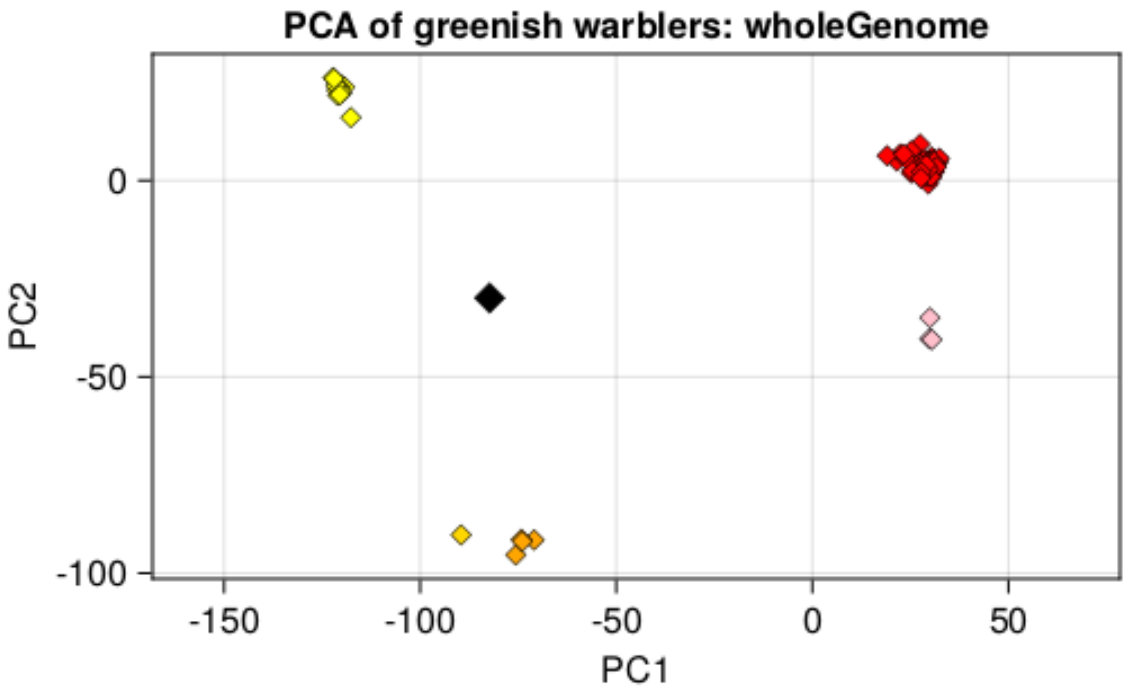
Make a whole-genome PCA just for the eastern side of the ring

```
eastern_groups_to_plot_PCA = ["troch_LN","troch_EM","obs","plumb_BJ","plumb"]
eastern_group_colors_PCA = ["yellow","gold","orange","pink","red"]
flipPC1 = true
flipPC2 = false
PCA_wholeGenome = plotPCA(genos_imputed_loaded, ind_with_metadata_indFiltered,
        eastern_groups_to_plot_PCA, eastern_group_colors_PCA;
        sampleSet = "greenish warblers", regionText = "wholeGenome",
        flip1 = flipPC1, flip2 = flipPC2)
totalObservationVariance = var(PCA_wholeGenome.model)
PC1_variance, PC2_variance = principalvars(PCA_wholeGenome.model)[1:2]
PC1_prop_variance = PC1_variance / totalObservationVariance
PC2_prop_variance = PC2_variance / totalObservationVariance
println("PC1 explains ", 100*PC1_prop_variance, "% of the total variance.
PC2 explains ", 100*PC2_prop_variance, "%.")
```

```
# add position of reference genome
refGenomePCAposition = predict(PCA_wholeGenome.model, zeros(size(genos_imputed_loaded, 2)))
flipPC1 && (refGenomePCAposition[1] *= -1)  # this flips PC1 if flipPC1 = true
flipPC2 && (refGenomePCAposition[2] *= -1)  # same for PC2
CairoMakie.scatter!(refGenomePCAposition[1], refGenomePCAposition[2], marker = :diamond, color="blac
try
    display(PCA_wholeGenome.PCAfig)
catch
    println("NOTICE: Figure for ", regionText, " could not be shown due to an unknown error.")
end
```



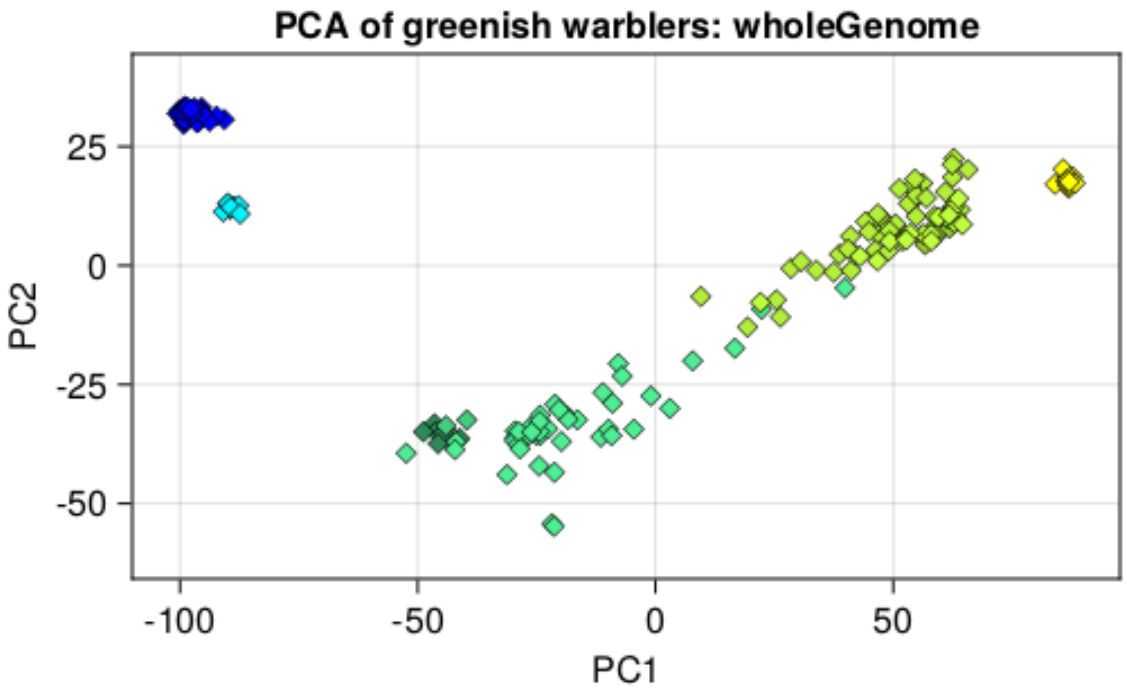

```
PC1 explains 9.776807% of the total variance.
PC2 explains 1.8531415%.
```




```
CairoMakie.Screen{IMAGE}
```

Make a whole-genome PCA just for the western side of the ring

```
western_groups_to_plot_PCA = ["vir", "vir_misID", "vir_S", "nit", "lud_PK", "lud_KS", "lud_central",
western_group_colors_PCA = ["blue", "blue", "turquoise1", "grey", "seagreen4", "seagreen3", "seagreen
PCA_wholeGenome = plotPCA(genos_imputed_loaded, ind_with_metadata_indFiltered,
    western_groups_to_plot_PCA, western_group_colors_PCA;
    sampleSet="greenish warblers", regionText="wholeGenome",
    flip1=true, flip2=true)
totalObservationVariance = var(PCA_wholeGenome.model)
PC1_variance, PC2_variance = principalvars(PCA_wholeGenome.model)[1:2]
PC1_prop_variance = PC1_variance / totalObservationVariance
PC2_prop_variance = PC2_variance / totalObservationVariance
println("PC1 explains ", 100 * PC1_prop_variance, "% of the total variance.
  PC2 explains ", 100 * PC2_prop_variance, "%.")
```




```
PC1 explains 11.392442% of the total variance.
  PC2 explains 1.7764572%.
```

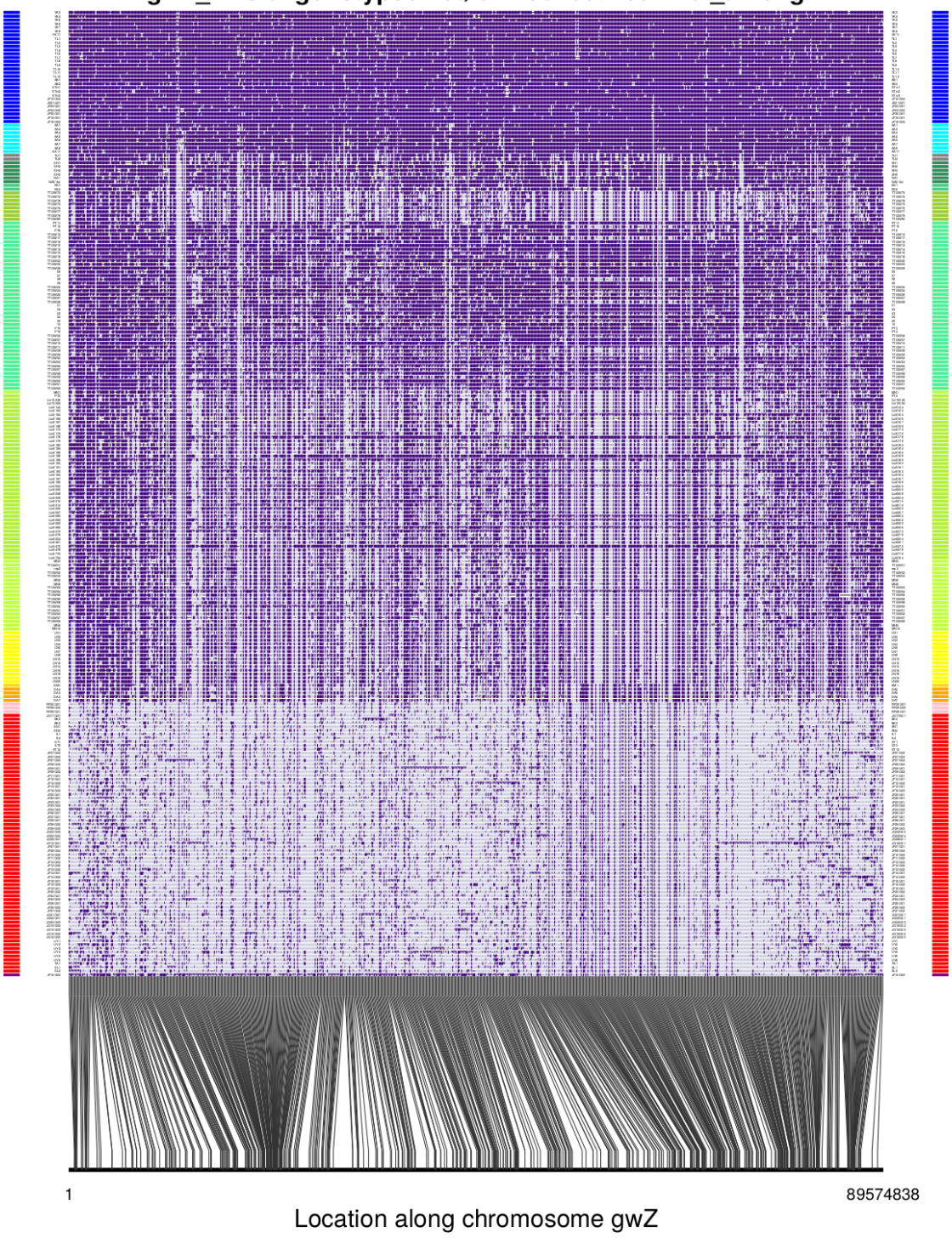
Make GBI plots showing all individuals

For Z chromosome:

```
groups = ["vir","lud_PK","troch_LN","obs","plumb"] # for purpose of calculating pairwise Fst and Fst
plotGroups = groups_to_plot_PCA # all GW individuals
plotGroupColors = group_colors_PCA
group1 = "vir"   # these groups will determine the color used in the graph
group2 = "plumb"
groupsToCompare = "Fst_among" #"vir_plumb"
Fst_cutoff = 0.7
missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among in

# Calculate allele freqs and sample sizes (use column Fst_group)
freqs, sampleSizes = getFreqsAndSampleSizes(genotypes_gwZ_SNPfiltered, ind_with_metadata_indFiltered
```

```
println("Calculated population allele frequencies and sample sizes")

Fst, FstNumerator, FstDenominator, pairwiseNamesFst = getFst(freqs, sampleSizes, groups; among=true)
println("Calculated Fst values")

chr = "gwZ"
regionInfo = chooseChrRegion(pos_SNP_filtered_gwZ_SNPfiltered, chr; positionMin=1, positionMax=NaN) #

plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff,
    missingFractionAllowed, regionInfo,
    pos_SNP_filtered_gwZ_SNPfiltered, Fst, pairwiseNamesFst,
    genotypes_gwZ_SNPfiltered, ind_with_metadata_indFiltered_sex, freqs, plotGroups, plotGroupColors
    indFontSize=4, figureSize=(1200,1600));
# plotInfo contains a tuple with: (f, plottedGenotype, locations, plottedMetadata)
```

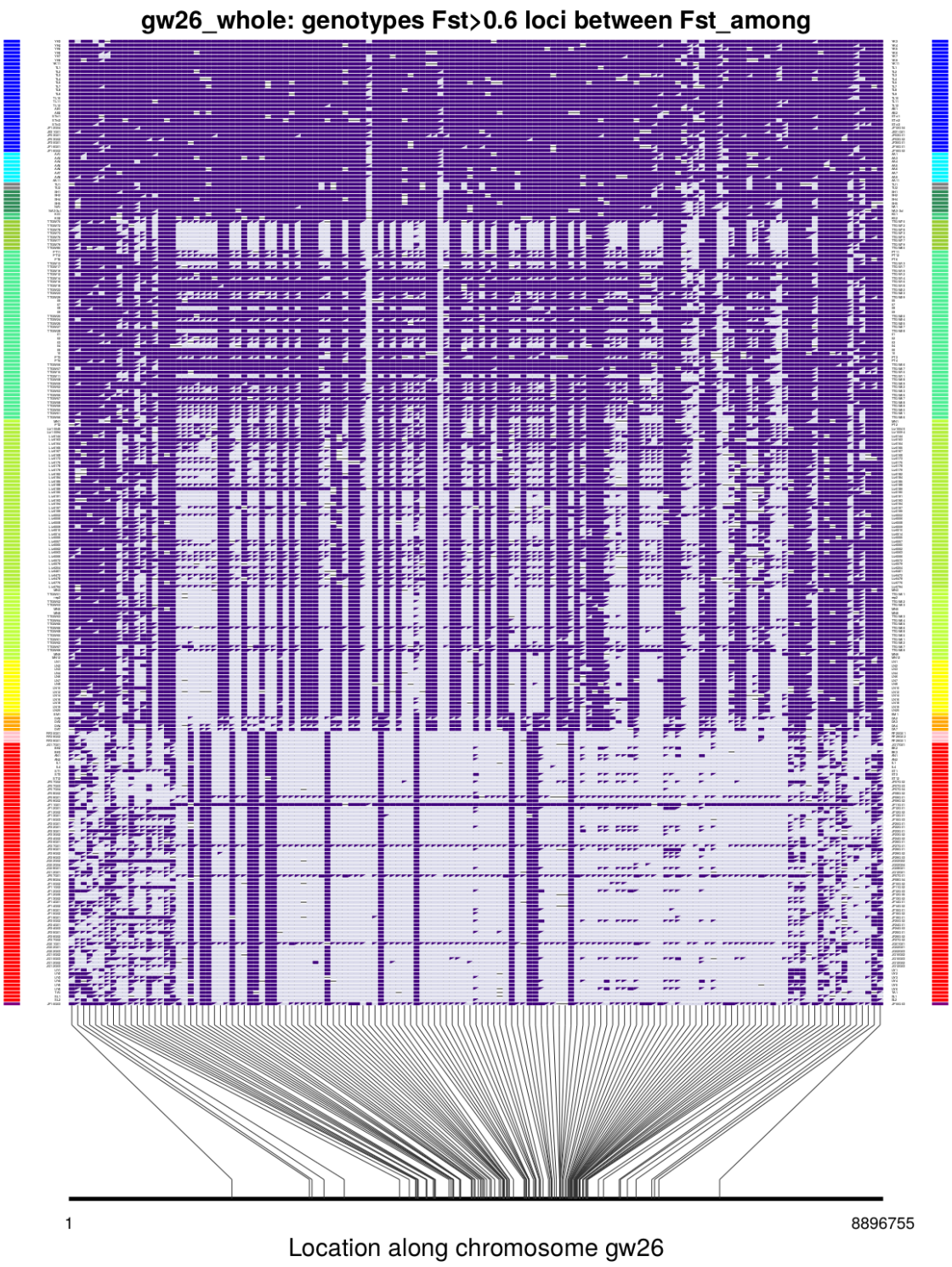
```
Calculated population allele frequencies and sample sizes
Calculated Fst values
```


gwZ_whole: genotypes Fst>0.7 loci between Fst_among
1
89574838
Location along chromosome gwZ

For chr 28:

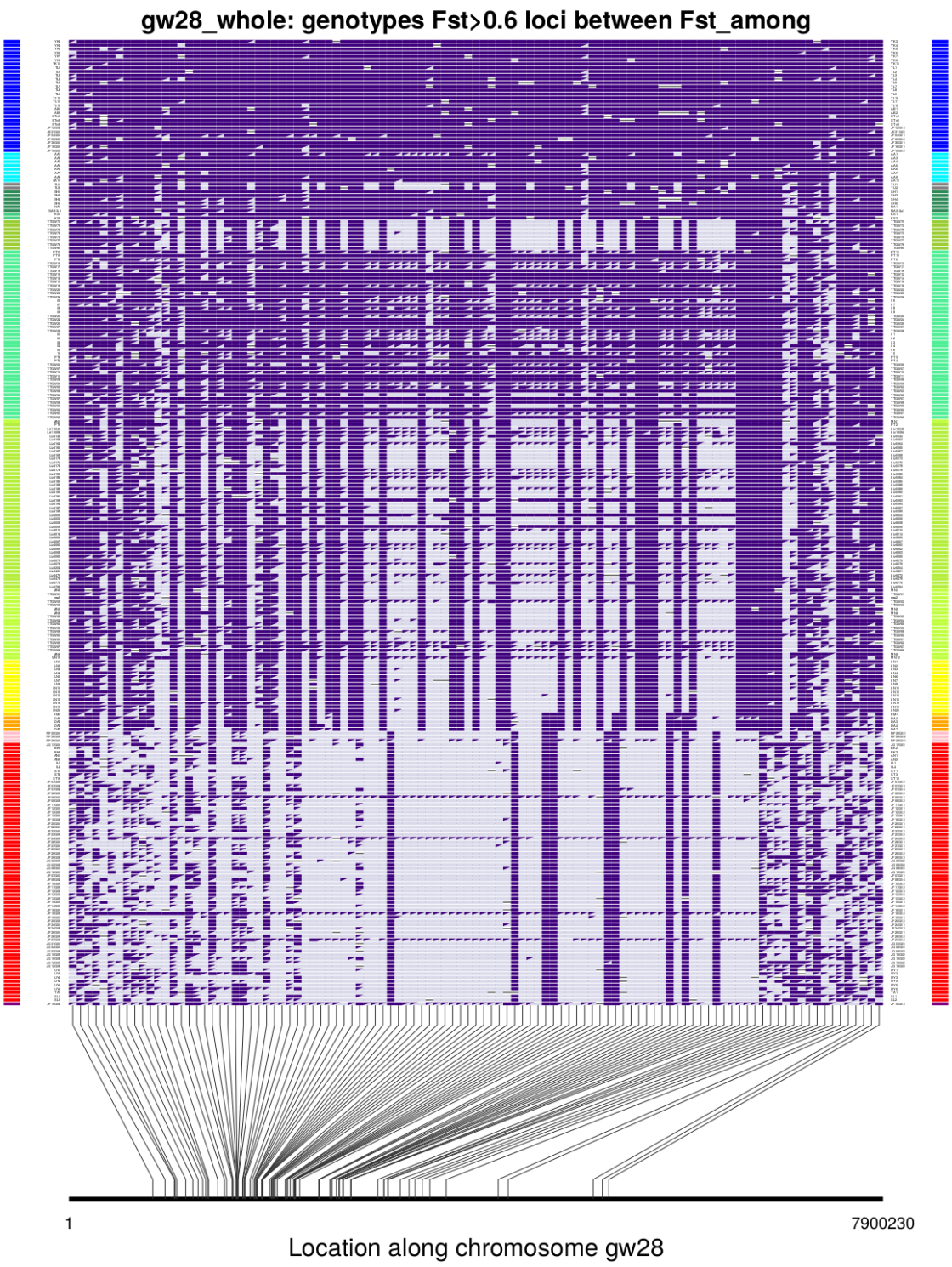
```
groups = ["vir","lud_PK","troch_LN","obs","plumb"] # for purpose of calculating pairwise Fst and Fst
plotGroups = groups_to_plot_PCA # all GW individuals
plotGroupColors = group_colors_PCA
group1 = "vir"   # these groups will determine the color used in the graph
group2 = "plumb"
groupsToCompare = "Fst_among" #"vir_plumb"
Fst_cutoff = 0.6
missingFractionAllowed = 0.2  # only show SNPs with less than this fraction of missing data among ind

# Calculate allele freqs and sample sizes (use column Fst_group)
freqs, sampleSizes = getFreqsAndSampleSizes(genosOnly_with_missing, ind_with_metadata_indFiltered.Fst
println("Calculated population allele frequencies and sample sizes")

Fst, FstNumerator, FstDenominator, pairwiseNamesFst = getFst(freqs, sampleSizes, groups; among=true)
println("Calculated Fst values")

chr = "gw26"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN) # this gets the r

plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff,
    missingFractionAllowed, regionInfo,
    pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_with_missing, ind_with_metadata_indFiltered, freqs,
    plotGroups, plotGroupColors;
    indFontSize=4, figureSize=(1200,1600));
# plotInfo contains a tuple with: (f, plottedGenotype, locations, plottedMetadata)
```

```
Calculated population allele frequencies and sample sizes
Calculated Fst values
```


gw26_whole: genotypes Fst>0.6 loci between Fst_among
1
8896755
Location along chromosome gw26

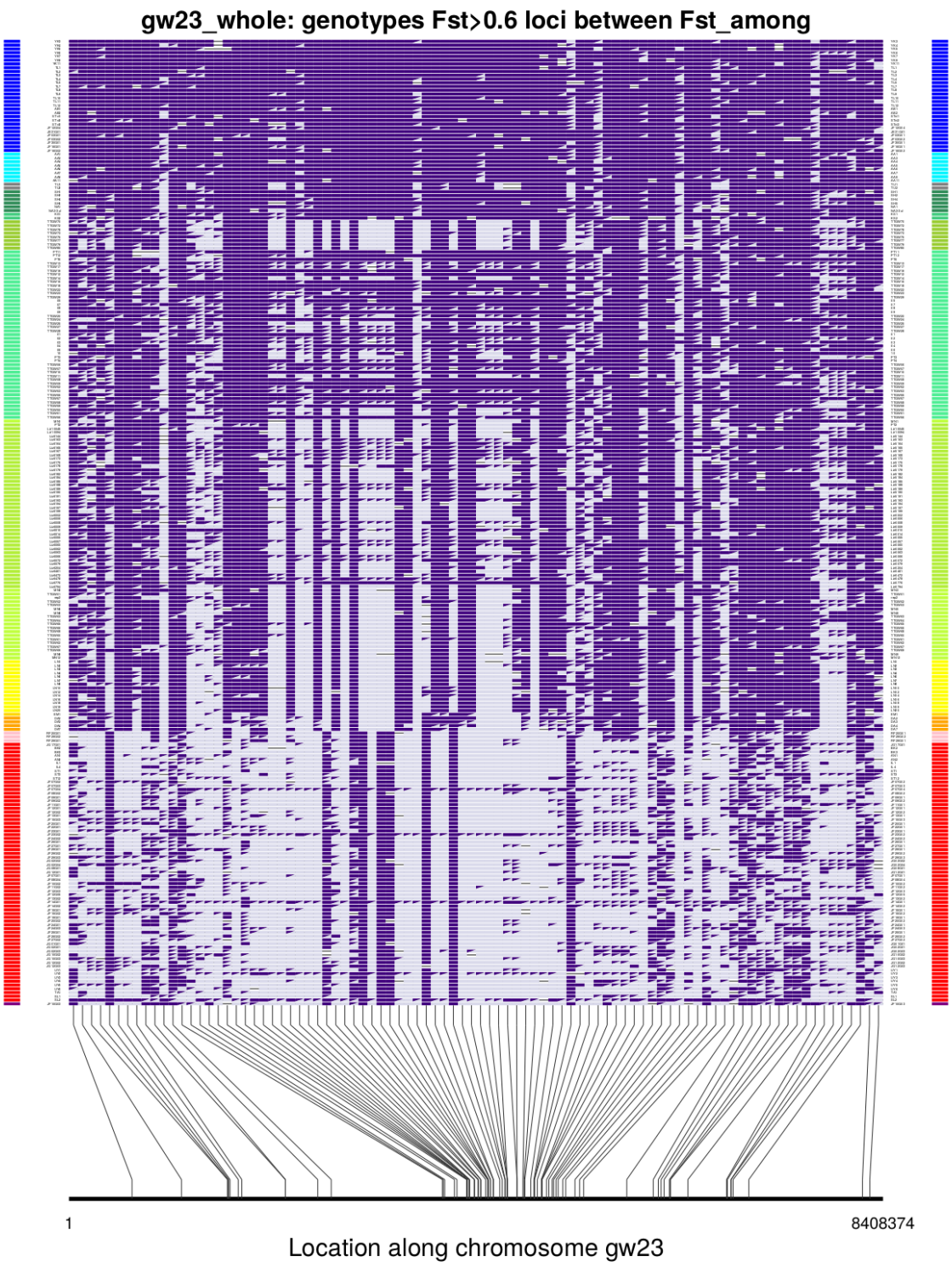
For chr 28:

```
chr = "gw28"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN)
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff,
    missingFractionAllowed, regionInfo,
    pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_with_missing, ind_with_metadata_indFiltered, freqs,
    plotGroups, plotGroupColors;
    indFontSize=4, figureSize=(1200,1600));
```


gw28_whole: genotypes Fst>0.6 loci between Fst_among
1
7900230
Location along chromosome gw28

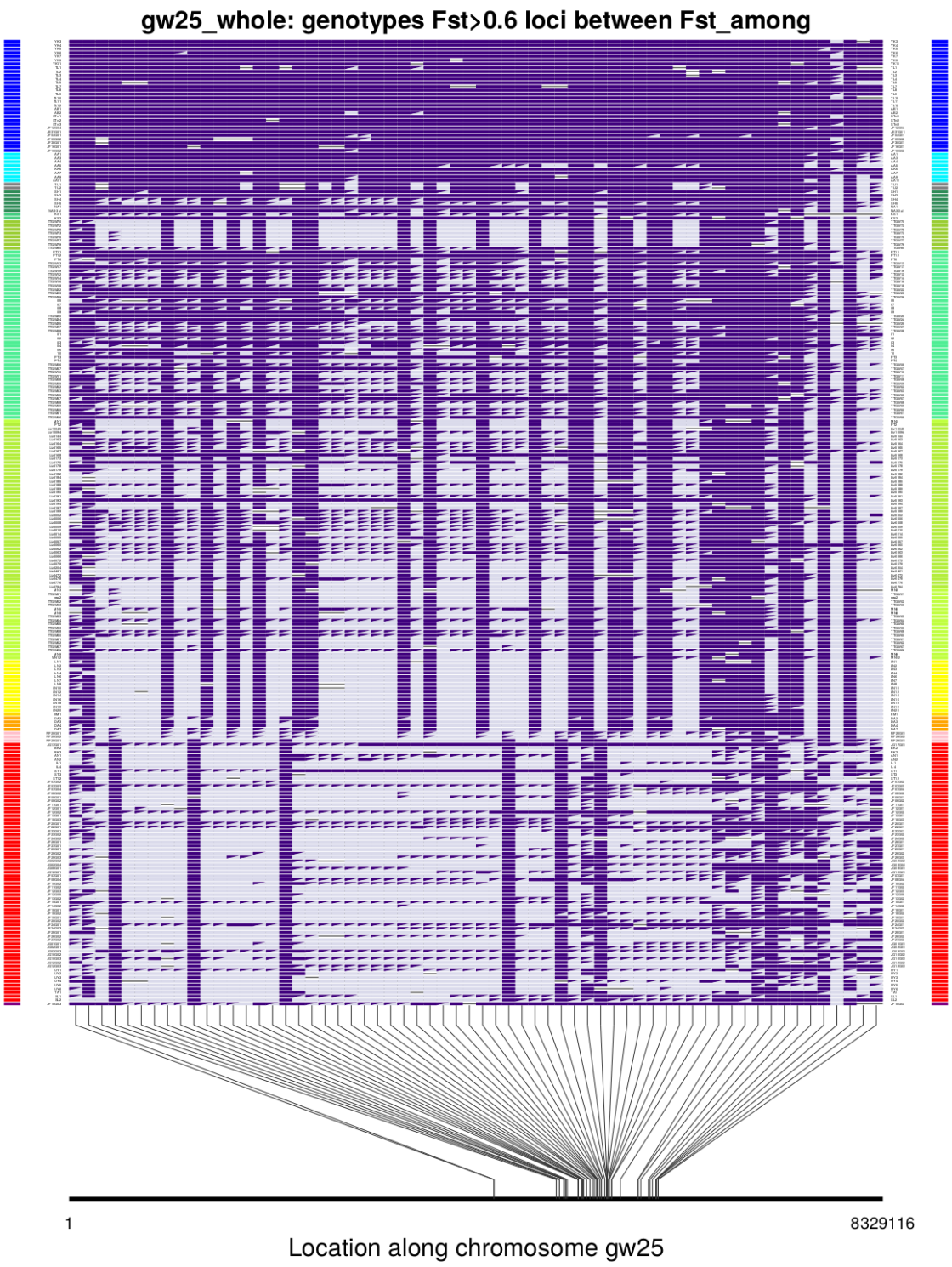
For chr 23:

```
chr = "gw23"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN)
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff,
    missingFractionAllowed, regionInfo,
    pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_with_missing, ind_with_metadata_indFiltered, freqs,
    plotGroups, plotGroupColors;
    indFontSize=4, figureSize=(1200,1600));
```


gw23_whole: genotypes Fst>0.6 loci between Fst_among
1
8408374
Location along chromosome gw23

For chr 25:

```
chr = "gw25"
regionInfo = chooseChrRegion(pos_SNP_filtered, chr; positionMin=1, positionMax=NaN)
plotInfo = plotGenotypeByIndividual(groupsToCompare, Fst_cutoff,
    missingFractionAllowed, regionInfo,
    pos_SNP_filtered, Fst, pairwiseNamesFst,
    genosOnly_with_missing, ind_with_metadata_indFiltered, freqs,
    plotGroups, plotGroupColors;
    indFontSize=4, figureSize=(1200,1600));
```


gw25_whole: genotypes Fst>0.6 loci between Fst_among
1
8329116
Location along chromosome gw25