GBS Pilot Analysis. MiSeq data. PstI.

Mario Vallejo-Marin

April 13 2017

Contents

Mimulus guttatus, *M. luteus*, *M. peregrinus*

Data Format and Preparation

Genotype data. Created in Tassel 5 by **Alex Twyford**: *../genotype_data/Mimulus_Mario_MiSeq100417.vcf*

Analysis in R

Method 1: Using the *vcfR* package

```
library(vcfR)  ##For reading and manipulating VCF files
library(pegas)
library(adegenet)
library(ape)
library(devtools)  #For extracting information about packages and versions used
```

Read data

You can read the VCF file directly using vcfR

```
#ndat<-read.vcfR(file="/Users/Mario/Documents/Mimulus/Twyford_Friedman_2015_SNP/Working_Data/Twyford_Mi

ndat<-read.vcfR(file="../genotype_data/Mimulus_Mario_MiSeq100417.vcf", verbose = F, skip= 0)

ndat
```

```
## ***** Object of Class vcfR *****
## 6 samples
## 3,301 variants
## Object size: 1.1 Mb
## 0 percent missing data
## *****        *****         *****
```

The resulting file is 1.1Mb.

You can now transform this file into a *genlight* object, which is a format that stores genotypes in binary format, and is more memory efficient.

```
genlight.ndat<-vcfR2genlight(ndat)
```

```
## Warning in vcfR2genlight(ndat): Found 9 loci with more than two alleles.
## Objects of class genlight only support loci with two alleles.
## 9 loci will be omitted from the genlight object.
```

```
## Loading required package: parallel
```

```
genlight.ndat
```

```
##  /// GENLIGHT OBJECT /////////
##
##  // 6 genotypes,  3,292 binary SNPs, size: 290.4 Kb
##  2435 (12.33 %) missing data
##
##  // Basic content
##    @gen: list of 6 SNPbin
##
##  // Optional content
##    @ind.names:  6 individual labels
##    @loc.names:  3292 locus labels
##    @chromosome: factor storing chromosomes of the SNPs
##    @position: integer storing positions of the SNPs
##    @other: a list containing: elements without names
```

```
genlight.ndat@ind.names
```

```
## [1] "P1481C" "p24411" "P2444B" "P07341" "P24951" "P07021"
```

You can now create a list of more descriptive names

```
genlight.ndat@ind.names
```

```
## [1] "P1481C" "p24411" "P2444B" "P07341" "P24951" "P07021"
```

```
temp<-c("gut-DBL", "lut-EY", "gut-4x-QUA", "per-LED", "lut-COL", "per-STR")
genlight.ndat@ind.names<-temp
```

The resulting object is smaller (209.4Kb). The file consisists of **6 individuals** and **3,292 loci**. Missing data: 12.33%

Calculating genetic distances

The first step is to obtain a matrix of DNA sequences that can then be used to calculate genetic distances. The *vcfR* package has an option to transform genlight objects to DNAbin objects, which is the format used by *ape*

```
dna.dat<-vcfR2DNAbin(ndat, consensus=T, extract.haps=F, ref.seq=NULL, verbose=T)
```

```
## After extracting indels, 3298 variants remain.
```

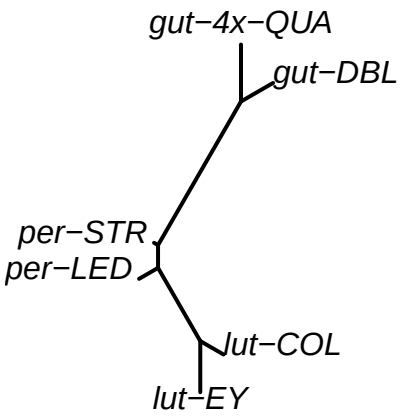
Now you can calculate individual genetic distance. The *pairwise.deletion* = *T* option allows keeping loci with some missing values; it just omits the locus if missing data in a particular taxon pairwise comparison

```
D <- dist.dna(dna.dat, model="raw", pairwise.deletion=T) #proportion of sites that differ between each
```

NJ Tree

Once you created a distance matrix, you can build a NJ tree

```
njtree1<-nj(D)
njtree1$tip.label<-c("gut-DBL", "lut-EY", "gut-4x-QUA", "per-LED", "lut-COL", "per-STR")
plot(njtree1, type="u", use.edge.length=T, edge.width=2)
```




```
#plot(root(nj(D),"LYC1"), type="p", use.edge.length=T, edge.width=2)
#LYC is the outgroup, M. moniliformis
```

You can calculate bootstrap support using the *boot.phylo* function from *ape*

```
#plot(tree, type="p", use.edge.length=T, edge.width=2,
#     show.node.label=T)  #With bootstrap values
```

You can then save the tree and export it for plotting with other programs such as *FigTree*

```
saved.tre.nj<-write.tree(njtree1)
write.table(saved.tre.nj, "savedNJtree.txt", sep="\t")
```

PCA

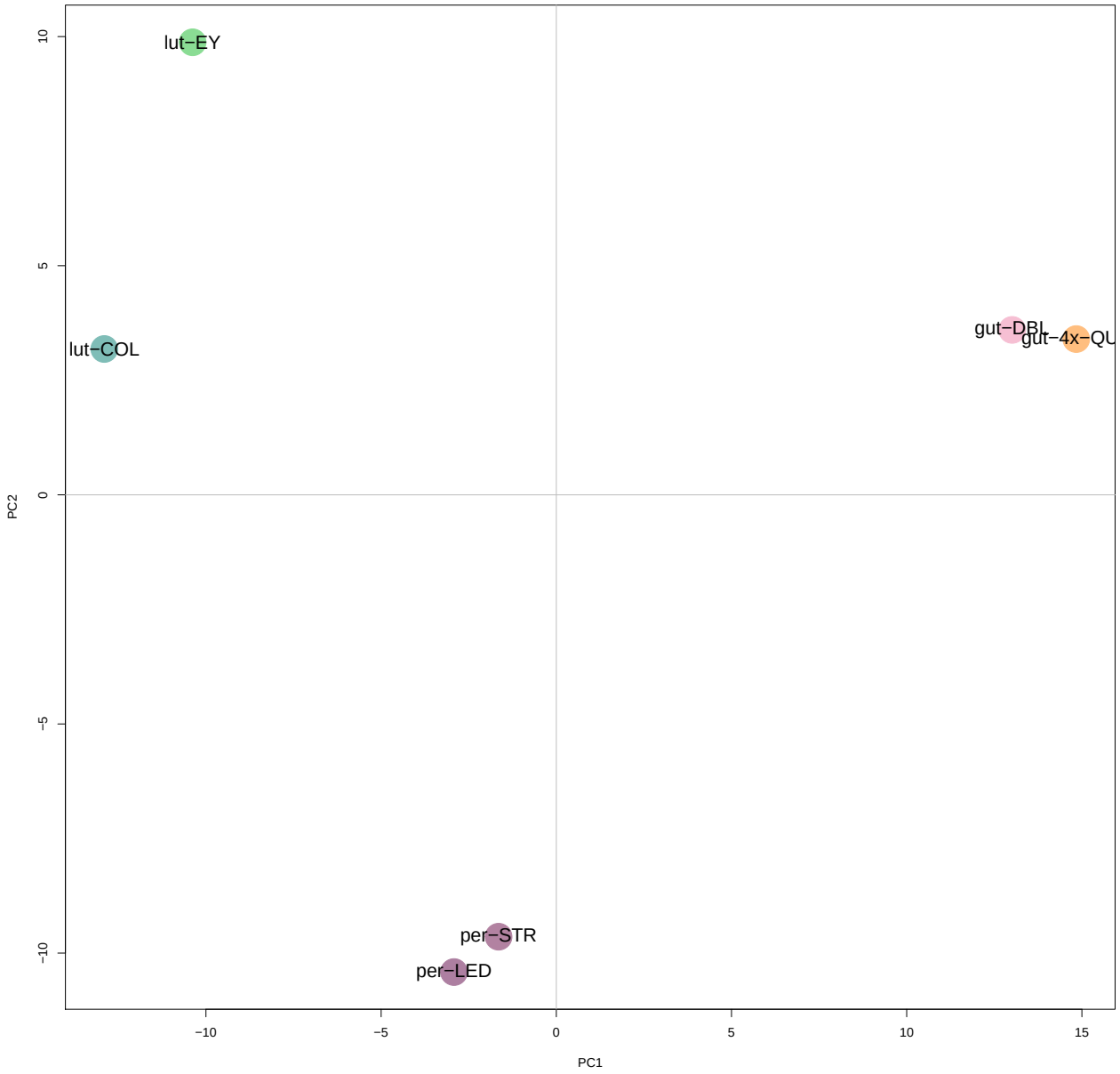
Do a PCA of the genetic data using the *glPca* function. This is a time-consuming step.

```
pca<-glPca(genlight.ndat, parallel=T, nf=4)
#The 'nf' option selects the number of axes for the PCA, which otherwise needs to be done interactively
```

```
temp<-indNames(genlight.ndat)
taxon.labs<-c("gut-DBL", "lut-EY", "gut-4x-QUA", "per-LED", "lut-COL", "per-STR")
taxon.labs
```

```
## [1] "gut-DBL"    "lut-EY"     "gut-4x-QUA" "per-LED"    "lut-COL"
## [6] "per-STR"
```

```
#PCA Plots:
myCol <- colorplot(pca$scores,pca$scores, transp=TRUE, cex=7)
#text(pca$scores[,1], pca$scores[,2], indNames(genlight.ndat), cex=0.7)
text(pca$scores[,1], pca$scores[,2], taxon.labs, cex=1.5)
abline(h=0,v=0, col="grey")
```



#frozen.pca<-pca

Method 2

Read the data. The command "read.vcf" will create an object of type 'loci' Full data set is 6 individuals, and 3,301 loci

```
dat<-read.vcf(file="../genotype_data/Mimulus_Mario_MiSeq100417.vcf"
              #from = 1, to = 38872   #Use partial data set for exploratory analyses
              #from = 1, to = 1000
```

```
              )
```

```
dat
```

```
## Allelic data frame: 6 individuals
##                     3301 loci
```

```
#summary(dat)
#names(dat) #Loci names
```

Next step is to transform the file into "genind" format.

Very *important*. Make sure to explicitly state what is the missing character in the original file. Otherwise the "." as missing data will cause trouble downstream.

```
genind.dat<-df2genind(dat, sep="/", NA.char = ".")
#summary(genind.dat)
```

Now, add the Population information to the *genind* object:

```
# (1) First, extract individual names:
#temp<-rownames(genind.dat@tab)
#The population is (usually) only the letters in the name. The only exception is 'M2L'
#(2) Remove the numeric characters:
#pop.labs<-gsub("[0-9]", "",temp)
#(3) Add the Population information to the 'genind.dat data' set
pop(genind.dat)<-taxon.labs
pop(genind.dat)
```

```
## [1] gut-DBL    lut-EY     gut-4x-QUA per-LED    lut-COL    per-STR
## Levels: gut-DBL lut-EY gut-4x-QUA per-LED lut-COL per-STR
```

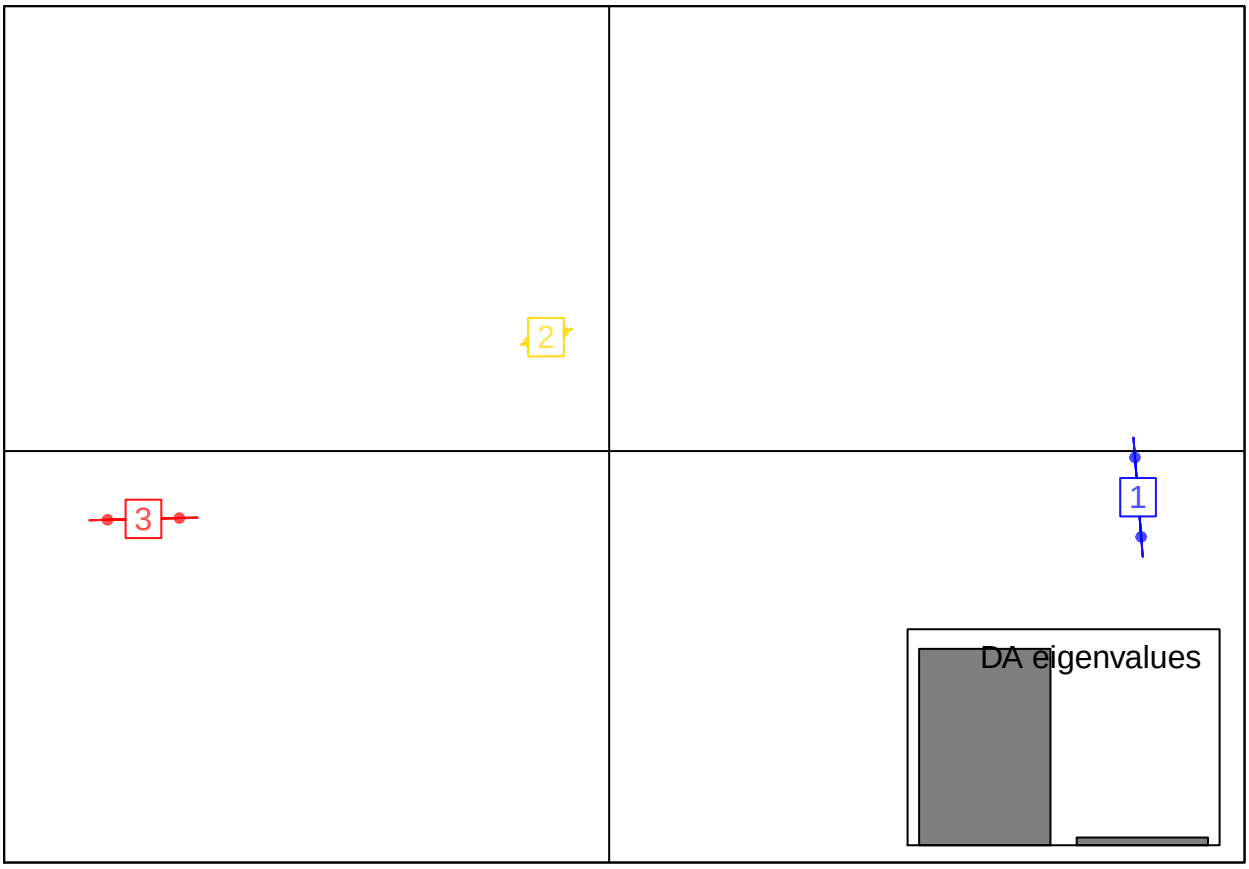
Analysis using DAPC

Now you can analyse the data using DAPC (Find number of PCA and clusters interactively first)

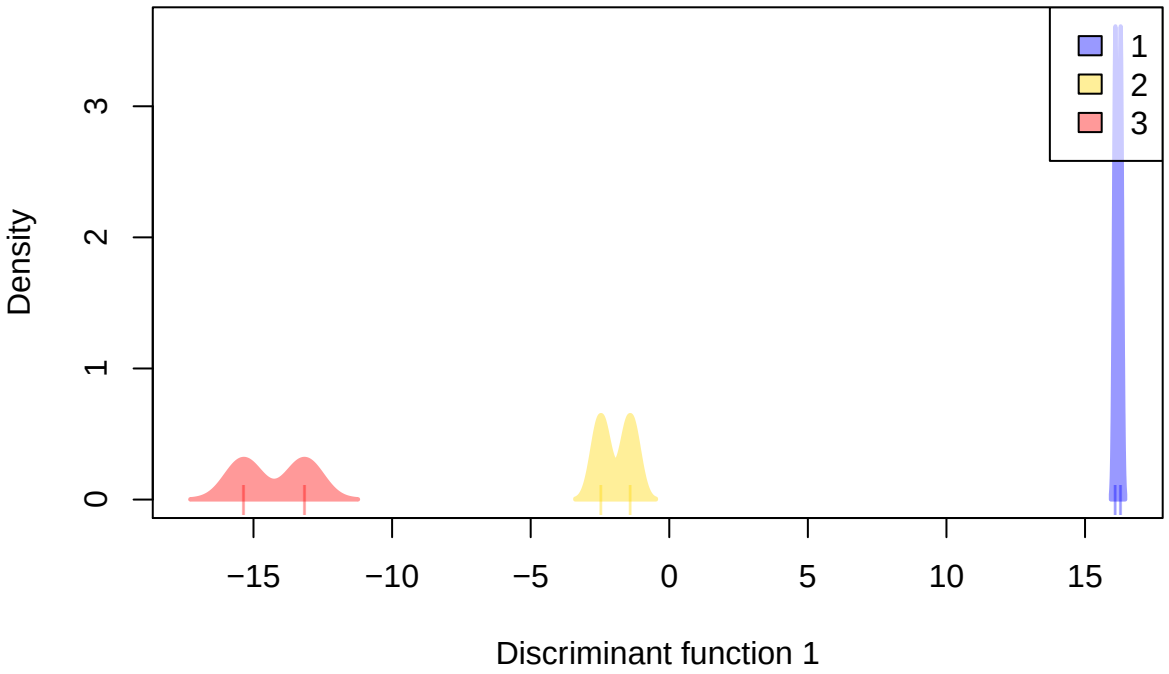
```
grp <- find.clusters(genind.dat, max.n.clust=40, n.pca=50, n.clust=3)

dapc1 <- dapc(genind.dat, grp$grp, n.pca=2, n.da=3)
```

Plot the Results

```
scatter(dapc1)
```




```
scatter(dapc1,1,1, bg="white", scree.da=FALSE, legend=TRUE, solid=.4)
```



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Session info

```
options(width = 100)
devtools::session_info()
```

```
## Session info -------------------------------------------------------------------------------------
```

```
##  setting  value
##  version  R version 3.3.3 (2017-03-06)
##  system   x86_64, darwin13.4.0
##  ui       X11
##  language (EN)
##  collate  en_GB.UTF-8
##  tz       Europe/London
##  date     2017-04-13
```

```
## Packages -----------------------------------------------------------------------------------------
```

```
##  package     * version date       source
##  ade4        * 1.7-5   2016-12-13 CRAN (R 3.3.2)
##  adegenet    * 2.0.1   2016-02-15 CRAN (R 3.3.0)
##  ape         * 4.0     2016-12-01 CRAN (R 3.3.2)
##  assertthat    0.1     2013-12-06 CRAN (R 3.3.0)
##  backports     1.0.4   2016-10-24 CRAN (R 3.3.0)
##  boot          1.3-18  2016-02-23 CRAN (R 3.3.3)
##  cluster       2.0.5   2016-10-08 CRAN (R 3.3.3)
##  coda          0.19-1  2016-12-08 CRAN (R 3.3.2)
##  colorspace    1.3-2   2016-12-14 CRAN (R 3.3.2)
##  DBI           0.5-1   2016-09-10 CRAN (R 3.3.0)
##  deldir        0.1-12  2016-03-06 CRAN (R 3.3.0)
##  devtools    * 1.12.0  2016-06-24 CRAN (R 3.3.0)
##  digest        0.6.11  2017-01-03 CRAN (R 3.3.2)
##  dplyr         0.5.0   2016-06-24 CRAN (R 3.3.0)
##  evaluate      0.10    2016-10-11 CRAN (R 3.3.0)
##  gdata         2.17.0  2015-07-04 CRAN (R 3.3.0)
##  ggplot2       2.2.1   2016-12-30 CRAN (R 3.3.2)
##  gmodels       2.16.2  2015-07-22 CRAN (R 3.3.0)
##  gtable        0.2.0   2016-02-26 CRAN (R 3.3.0)
##  gtools        3.5.0   2015-05-29 CRAN (R 3.3.0)
##  htmltools     0.3.5   2016-03-21 CRAN (R 3.3.0)
##  httpuv        1.3.3   2015-08-04 CRAN (R 3.3.0)
##  igraph        1.0.1   2015-06-26 CRAN (R 3.3.0)
##  knitr         1.15.1  2016-11-22 CRAN (R 3.3.2)
##  lattice       0.20-34 2016-09-06 CRAN (R 3.3.3)
##  lazyeval      0.2.0   2016-06-12 CRAN (R 3.3.0)
##  LearnBayes    2.15    2014-05-29 CRAN (R 3.3.0)
##  magrittr      1.5     2014-11-22 CRAN (R 3.3.0)
##  MASS          7.3-45  2016-04-21 CRAN (R 3.3.3)
##  Matrix        1.2-8   2017-01-20 CRAN (R 3.3.3)
##  memoise       1.0.0   2016-01-29 CRAN (R 3.3.0)
##  memuse        3.0-1   2016-09-20 CRAN (R 3.3.0)
##  mgcv          1.8-17  2017-02-08 CRAN (R 3.3.3)
##  mime          0.5     2016-07-07 CRAN (R 3.3.0)
##  munsell       0.4.3   2016-02-13 CRAN (R 3.3.0)
##  nlme          3.1-131 2017-02-06 CRAN (R 3.3.3)
```

```
##  pegas       * 0.9     2016-04-16 CRAN (R 3.3.0)
##  permute       0.9-4   2016-09-09 CRAN (R 3.3.0)
##  pinfsc50      1.1.0   2016-12-02 CRAN (R 3.3.2)
##  plyr          1.8.4   2016-06-08 CRAN (R 3.3.0)
##  R6            2.2.0   2016-10-05 CRAN (R 3.3.0)
##  Rcpp          0.12.9  2017-01-14 CRAN (R 3.3.2)
##  reshape2      1.4.2   2016-10-22 CRAN (R 3.3.0)
##  rmarkdown     1.3     2016-12-21 CRAN (R 3.3.2)
##  rprojroot     1.2     2017-01-16 CRAN (R 3.3.2)
##  scales        0.4.1   2016-11-09 CRAN (R 3.3.2)
##  seqinr        3.3-3   2016-10-13 CRAN (R 3.3.0)
##  shiny         1.0.0   2017-01-12 CRAN (R 3.3.2)
##  sp            1.2-4   2016-12-22 CRAN (R 3.3.2)
##  spdep         0.6-9   2017-01-09 CRAN (R 3.3.2)
##  stringi       1.1.2   2016-10-01 CRAN (R 3.3.0)
##  stringr       1.1.0   2016-08-19 CRAN (R 3.3.0)
##  tibble        1.2     2016-08-26 CRAN (R 3.3.0)
##  vcfR        * 1.4.0   2017-01-07 CRAN (R 3.3.2)
##  vegan         2.4-1   2016-09-07 CRAN (R 3.3.0)
##  viridisLite   0.1.3   2016-03-12 CRAN (R 3.3.0)
##  withr         1.0.2   2016-06-20 CRAN (R 3.3.0)
##  xtable        1.8-2   2016-02-05 CRAN (R 3.3.0)
##  yaml          2.1.14  2016-11-12 CRAN (R 3.3.2)
```