Computer Lab 4 – AMOVA, F_{ST}, Haplotype Networks

Conservation Genetics (BIOL 4174 / 5174)

Part I – Haplotype Networks in PopArt

PopArt is a free population genetics program primarily used for inferring and visualizing relationships among haplotypes. In this lab we will explore several methods for creating haplotype networks in PopArt.

In your Applications folder, click on the PopArt shortcut to open the program. From the Lab 4 folder, import the "sbl-atp86_PopArt.nex" file (File → Open…). If the file has been imported successfully, you will see your sequences under the "Alignment" tab, and data describing how these samples are grouped into 'populations' in the "Traits" tab.

We will use PopArt to infer several types of haplotype networks. You can see the methods implemented by PopArt under the "Network" tab. The first one we will implement is the "Minimum Spanning Network", which tries to create connections among haplotypes which minimize the number of mutations separating haplotypes (represented here as tick marks):

1. Infer a minimum spanning network (Network → Minimum Spanning Network)
    a. A pop up box will ask for a value of "epsilon". Set this to 0 for now.
        i. In short, this parameter is used to adjust the weights of "edges", or connection between haplotypes. In this case, we don't need such an adjustment.
    b. To make it easier to view, change the colors using the theme button:
        i. Any color scheme is fine
        ii. Notice that populations (GRN, LCS, etc) are each assigned a color
        iii. If your network is drawn messily, drag and drop haplotypes to make it cleaner
    c. Save your tree as PDF for later reference
        i. (File → Export Graphics) and select ".pdf" as format
2. Repeat these steps for the Median Joining Network
    a. This method attempts to infer presence of ancestral, unsampled haplotypes in your network.
    b. "Median Joining" networks starts with a Minimum Spanning Tree, and then tries to further resolve relationships by adding consensus sequences (also referred ti as median vectors, or Steiner points) in order to create a more parsimonious tree. Biologically, this can be thought of as unobserved haplotypes (e.g. extinct, or unsampled), and are represented as small filled circles.

c. Feel free to also explore the other methods. “TCS” uses a method called statistical parsimony, and is another very commonly applied technique

FYI: If you want to graphically modify a haplotype network (e.g. for publication), it can be saved as a .SVG file and edited with Adobe Illustrator, or Inkscape (a free alternative). This is not a requirement for this lab.

Answer questions 1 through 3 in the homework document.

Part II – Population Summary Statistics in R

Terms:
Nucleotide diversity (π) = average number of nucleotide differences per site between two individuals in a population

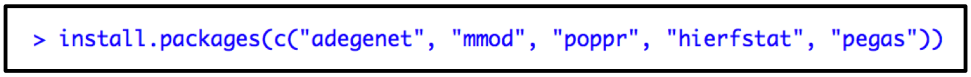
Theta (Θ) = mutation rate/ number of segregating sites. If this number is the same or similar to π, then neutrality is assumed for the molecular marker in question. Tajima’s D provides a test of this hypothesis.

Tajima’s D = A population genetic test statistic used to distinguish between a DNA sequence evolving randomly (e.g. according to neutral processes) and one evolving under a non-random process (e.g. selection).

For this section we will use a popular statistical computing environment “R” (https://www.r-project.org/). R is widely used among many types of data scientists and statisticians, due in part to the ease with which developers from many different fields can expand the environment by adding their own packages with special functions. Some packages with useful functions for population geneticists are: adegenet, ape, diveRsity, geneclust, hierfstat, mmod , pegas, phangorn, and poppr, although many more exist.

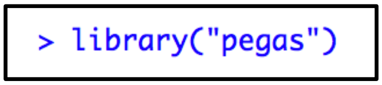
Note: I do not expect you to master R. Just type commands as I provide them and you should be fine.

We will interact with R through a helpful interface called R Studio. Open it by clicking in your Applications folder. First, install the packages we will be needing using the following code:



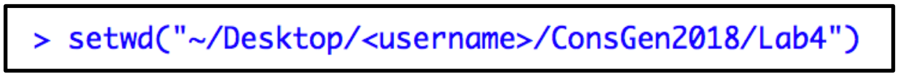
```
> install.packages(c("adegenet", "mmod", "poppr", "hierfstat", "pegas"))
```

We will first use a package called `pegas`. Load is using the `library` command:



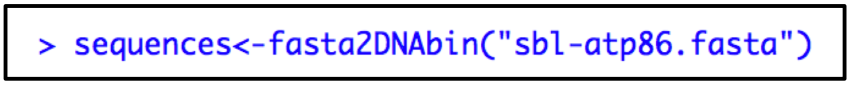
```
> library("pegas")
```

Before loading our dataset into R, first set the working directory to the location of your Lab 4 files. For example, if these are located in ~/Desktop/<username>/ConsGen2018/Lab4:



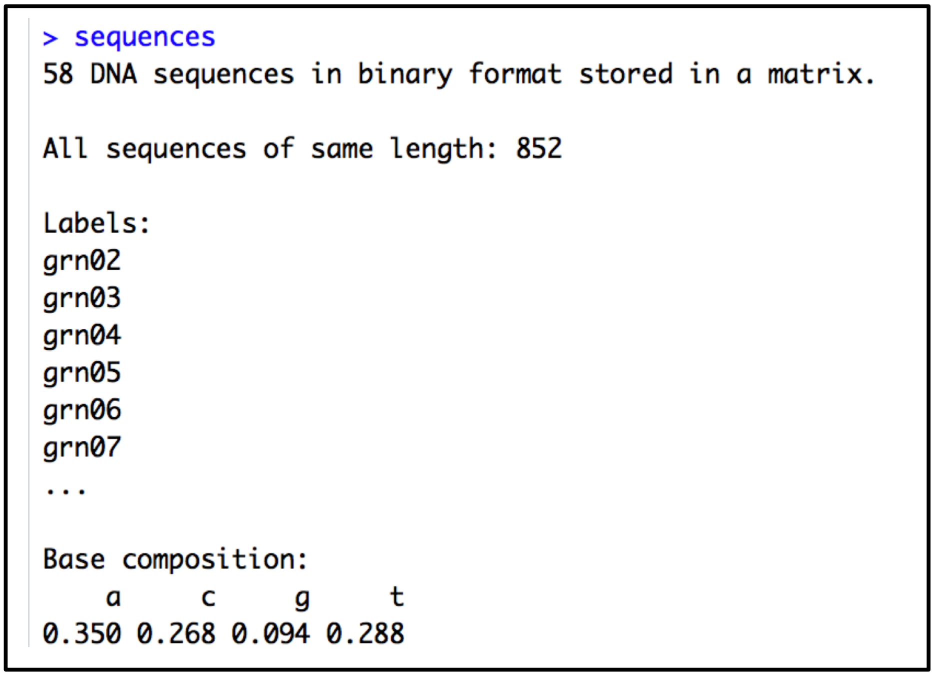
```
> setwd("~/Desktop/<username>/ConsGen2018/Lab4")
```

Next, we will load our sequence data and save is as a **variable** called "sequences":



```
> sequences<-fasta2DNAbin("sbl-atp86.fasta")
```

Whenever we need to use these data, we will access it using this "sequences" variable. Typing "sequences" should show you a summary of your data:



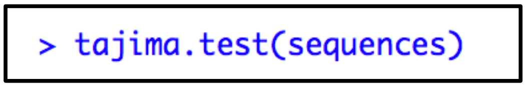
```
> sequences
58 DNA sequences in binary format stored in a matrix.

All sequences of same length: 852

Labels:
grn02
grn03
grn04
grn05
grn06
grn07
...

Base composition:
    a     c     g     t
0.350 0.268 0.094 0.288
```

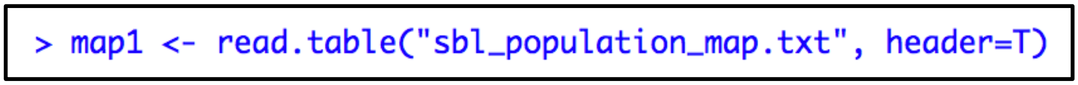
Next, we will calculate Tajima's test of neutrality using the `tajima.test` function in pegas:



```
> tajima.test(sequences)
```

This will output the value for Tajima's D ($D) and a p-value ($Pval.normal). **Use these to answer questions 4 and 5 in your homework.**

Next, we will calculate pairwise Fst between each sampling locality. To do this, we must first tell R which samples correspond to a single locality. Open the file "sbl_population_map.txt" in TextWrangler. You should see a tab-separated table, with each sample name (e.g. "grn02") mapped to a corresponding sampling locality (e.g. "1"). Load these into R as follows:



```
> map1 <- read.table("sbl_population_map.txt", header=T)
```

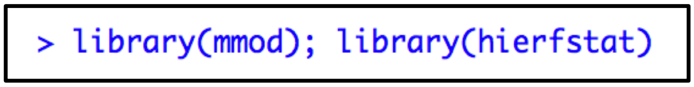
This creates a table called “map1” containing the contents of our file. Type “map1” into your R console to view it. You will see a list of sample names with two columns: “population”, which defines the population each sample came from, and “group”, which places these populations into 3 larger groups:

A. GRN, SEP, TMC

B. RAL, RAR, SHN

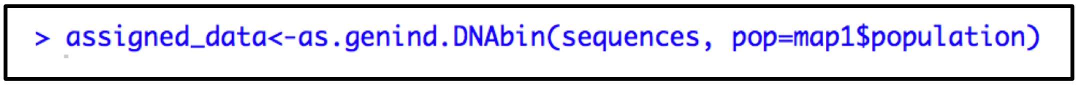
C. LCS, LCN, LCP, PTH

We now will load two more packages: `hierfstat` for calculating pairwise F_{ST} between our sites, and `mmod` to help with some useful data conversions.



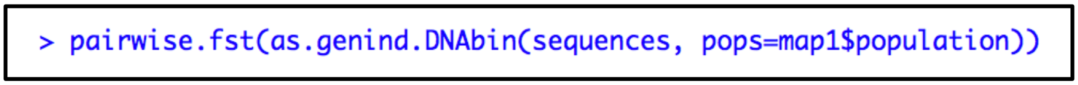
```
> library(mmod); library(hierfstat)
```

Next, we need to convert our data into a different format using this new map1 data:



```
> assigned_data<-as.genind.DNAbin(sequences, pop=map1$population)
```

Now, calculate pairwise F_{ST} using the following command. Note that we reference our “map1” table of population assignments:



```
> pairwise.fst(as.genind.DNAbin(sequences, pops=map1$population))
```

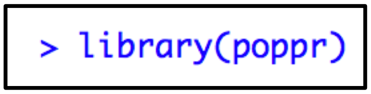
Now, answer questions 6-8 in your homework. Do not close R Studio, as we will use it for Part III.

FYI: The packages mentioned here have many more capabilities than we have explored. You can view additional options to functions or packages using the ‘?’ operator (e.g. `?pairwise.fst`). You can see an index of functions included in a package using the same operator by typing “`?<packagename>`” and clicking “index” in the resulting documentation page.

Part III – Analysis of Molecular Variance in R

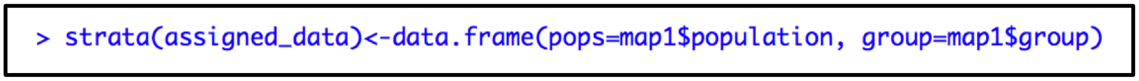
For this section we will perform a common analysis for detecting population differentiation and grouping called an AMOVA (Analysis of Molecular Variance), which is similar to the ANOVA usually taught in introductory statistics classes. The fundamental idea of this analysis is that population subdivision (e.g. reduced gene flow) causes increased genetic variation among subpopulations, and decreased variation within subpopulations relative to the total population.

First, load the poppr package, which we will use to implement the AMOVA:



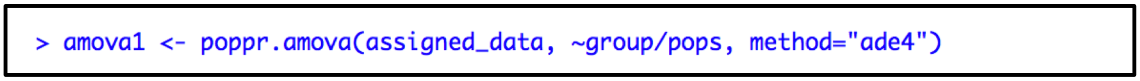
```
> library(poppr)
```

Next, we need to tell poppr that our populations we assigned will create the groups for the AMOVA:



```
> strata(assigned_data)<-data.frame(pops=map1$population, group=map1$group)
```

Finally, perform the AMOVA to calculate variance components **among groups, among groups within populations, and within populations**:

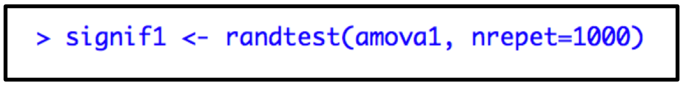


```
> amova1 <- poppr.amova(assigned_data, ~group/pops, method="ade4")
```

This saves the output as a variable “amova1”. Type “amova1” in your console to view it. In the output, you will see a “$componentsofcovariance” table which explains how much genetic variance is detected at each stratification. We expect variations within samples to give the greatest amount of variation if samples are not significantly differentiated. Sigma represents the variance, σ, for each hierarchical level and to the right is the percent of the total.

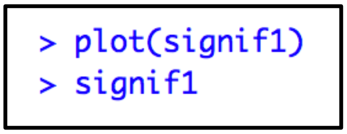
> Variations Between group: This is the percentage of genetic variation explained by the “group” column (e.g. A, B, C) in the “sbl_population_map.txt” file
>
> Variations Between samples Within group: This is the percentage of genetic variation explained by the “population” assignments within “group” (also defined in the “sbl_population_map.txt” file)
>
> Variations within samples: This is the percentage of genetic variation explained by comparing individuals within populations

Next, we will test if populations are significantly different by performing a randomization test using the function `randtest()`, which will randomly permute the sample matrices to generate a null distribution. Under the null hypothesis (=no population subdivision), samples are considered to be randomly drawn from a global population, with any variation only due to random sampling.



```
> signif1 <- randtest(amova1, nrepet=1000)
```

Use “plot(signif1)” and type “signif1” to view the results of the randomization test:



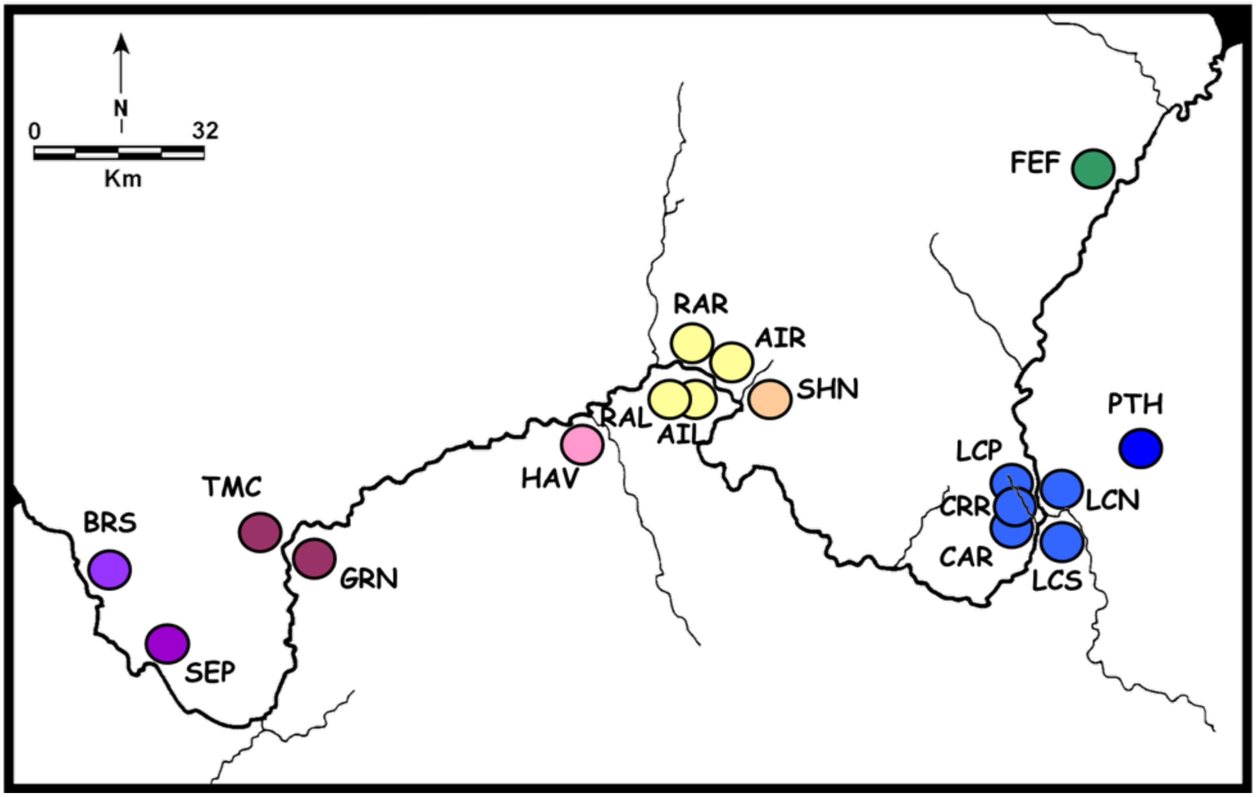
```
> plot(signif1)
> signif1
```

Answer question 9 in the homework document. Do not close R Studio.

Part IV – More AMOVA in R

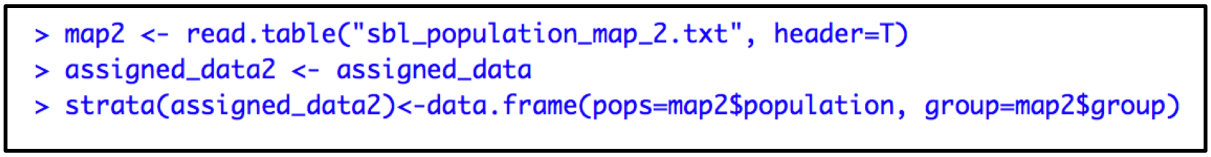
AMOVA can be applied for repeated testing of genetic structure by comparing different *a priori* assumptions regarding population structure (i.e., groups of populations). This means you can regroup the populations and run the AMOVA again until you have maximized the among group variation. In contrast, other programs such as STRUCTURE exist which do not make *a priori* assumptions about population structure. Instead, these programs use Bayesian clustering algorithms to group individuals into populations. We will explore these methods further when we start working with microsatellite data later in the semester.

For Part IV, you must make an alternative hypothesis of how populations might be grouped. Refer to the map below, your haplotype networks, and your estimates of pairwise F_{ST} to group populations in a different way.



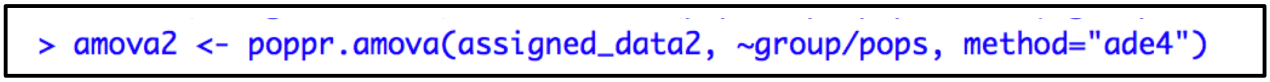
To accomplish this, open the "sbl_population_map.txt" and change the values in the "group" column (e.g. A, B, C…) to reflect your new grouping. For example, if your hypothesis was that populations were divided according to north and south of the large river in the map, you would change all samples from populations GRN, LCS, LCN, RAL, and PTH to "S" and SEP, TMC, RAR, SHN, and LCP to "N". **Save this file as "sbl_population_map_2.txt" in your Lab 4 folder.**

To load these new group assignments into R, we will need to re-do several steps from earlier. Go back to R Studio and load “sbl_population_map_2.txt” as a new variable “map2”, and re-define our groupings in the “assigned_data”:



```
> map2 <- read.table("sbl_population_map_2.txt", header=T)
> assigned_data2 <- assigned_data
> strata(assigned_data2)<-data.frame(pops=map2$population, group=map2$group)
```

Now re-do the AMOVA on our newly re-assigned dataset:



```
> amova2 <- poppr.amova(assigned_data2, ~group/pops, method="ade4")
```

Answer question 10 in the homework.