­­strataG new user guide June 2018

Shreya Banerjee

May 30, 2018

# What is strataG?

strataG is an R package developed by NOAA scientists for population genetic analyses. strataG is a toolkit for haploid sequence and multilocus genetic data summaries, and analyses of population structure. This package uses existing population genetics program, and allows you to use them all in one console, without having to reformat your raw data for each program! The purpose of this document is to guide you through using strataG to conduct basic genetic data QA/QC and Population genetic analyses. strataG has lots of other applications which we do not cover here.

For more information check out <https://github.com/EricArcher/strataG>

Archer, F. I., Adams, P. E. and Schneiders, B. B. (2016), strataG: An R package for manipulating, summarizing and analysing population genetic data. Mol Ecol Resour. <doi:10.1111/1755-0998.12559>

# Set-up

This guide assumes you have a basic familiarity with R. Click [here](https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf) for a basic introduction to R. You can also use the code below to install and in-console interactive R tutorial.

install.packages('swirl')library(swirl)swirl()

Remember, you can look up how to use any ‘function’ with the code:

?function()?plot()

StrataG uses the programs like GenePop and STRUCTURE for some functions. In order to run these functions in R you need to install these programs on your computer, compile them, and move them to a location where R can access them. Below are instructions to install GenePop on a Mac computer. Installing other programs (STRUCTURE) should follow the same general instructions.

**MAC USERS**

1. Compiling GenePop:

* go to: <http://kimura.univ-montp2.fr/%7Erousset/Genepop.htm>
* download ‘sources.zip’
* open a terminal window, change the directory to wherever your download goes to/where you put the sources.zip folder. For example: cd Downloads/sources
* note that command line is sensitive to cases
* once you are in the right directory, enter the following command into terminal and hit enter to compile Genepop: g++ -DNO\_MODULES -o Genepop GenepopS.cpp -O3
* If you do not have Apple developer tools installed already (Xcode), it will prompt you asking if you want it to install them. Say yes, it will install them, then run the command above again to compile GenePop.
* see this website for further/updated information: <http://genepop.curtin.edu.au>

1. Moving files:

* cd to where Genepop is if you aren’t already
* can check that you are in the right place with the program using ls -lh
* type in the command: cp Genepop /usr/local/bin/.
* likely it will tell you permission is denied, because there are safeguards on this folder so anyone can’t come in and put executable files in there that could damage your system.
* so then type command: sudo cp file /usr/local/bin/
* it will prompt for password-this is the password you likely already set up for logging in/making changes to your computer
* if it worked, you won’t see anything (except the prompt line comes back). If not, you’ll get an error message of some sort for why not

**WINDOWS USERS** \* go to: <http://kimura.univ-montp2.fr/%7Erousset/Genepop.htm> \* download ‘exe.zip’ \* unzip folders until you can access the genepop.exe file \* drag and drop the genepop.exe file to your working directory

**Installing the strataG package**

You need to install ‘devtools’ and ‘Rtools’ to install the latest version of strataG (from github). Run the following:

if (!require('devtools')) install.packages('devtools')if (!require('Rtools')) install.packages('Rtools')

Now you can install the most current version of StrataG! If you like, you can run this code each time before you start working to have the most updated version, but it takes a while to run, so it is recommended to do this monthly (or when you get an email from Eric telling you to do it ☺).

devtools::install\_github('ericarcher/strataG', build\_vignettes = TRUE)

Note for new R users: if the package is not installing read the error messages. If it says something about missing package “package.name” you can install this package with the code install.packages("package.name").

To familiarize yourself with strataG you can check out the vignettes or examples of code using preloaded data. You can access these online <https://github.com/EricArcher/strataG> or in console with the code:

browseVignettes("strataG")vignette("gtypes")

## Load the necessary packages and set up your working directory

Follow along with this [tutorial](https://github.com/lkomoro/Pac-Green-Turtle-Pop-Gen-Analysis/blob/master/CmPac_msat_popgen_project.Rmd)

library(strataG)

library(dplyr)

make it so that all values in any data frames are converted to factors

options(stringAsFactors = F)

Check that you are in the correct working directory (and change to a different directory if needed; *N.B. the syntax for this in R is slightly different in Mac and Windows*).

getwd()

## [1] "C:/Users/Genetics Lab/Desktop/strataG"

setwd("home/desireddirectory")

# Getting your Data into R

In this guide, we use a microsatellite data set for an example. You can also use strataG to analyze other types of genetic data (SNPs, mitochondrial DNA, etc.).

To get started, you need a [csv file](https://support.bigcommerce.com/articles/Public/What-is-a-CSV-file-and-how-do-I-save-my-spreadsheet-as-one) That includes your Strata (Grouping variable, likely your different populations), your sample names, and your genotypes (each allele as a column). For example: *(\*N.B. remember that in general avoid spaces or symbols in naming columns in R)*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Strata | SampleID | LocA\_1 | LocA\_2 | LocB\_1 | LocB\_2 |
| Australia | 1 | 123 | 125 | 180 | 180 |
| Australia | 2 | 127 | 131 | 180 | 189 |
| Mexico | 3 | 125 | 129 | 180 | 180 |
| Mexico | 4 | 129 | 129 | 180 | 192 |
| Hawaii | 5 | 119 | 129 | 189 | 192 |
| Hawaii | 6 | 125 | 127 | 183 | 189 |

You can also include a csv file of your stratification schemes, which should include a column for your sample names, and a column for each of your strata (grouping variables). Your stratification schemes determine how you group individuals for genetic structure analysis. You can analyze genetic structure on different spatial scales, with different groupings of individuals. For example:

|  |  |
| --- | --- |
| SampleID | Strata |
| 1 | Australia |
| 2 | Australia |
| 3 | Mexico |
| 4 | Mexico |
| 5 | Hawaii |
| 6 | Hawaii |

It is important that both your genotypes file and stratification scheme file have the same sample IDs.

## Let’s load our data

Here we set up variables for our specific files. If you adjust this section to fit your data you should be able to run most of the code with few or no modifications.

setwd("C:/Users/Genetics Lab/Desktop/strataG")  
Description <- "Msats\_strataG"  
strata.file <-"Strata.csv"   
genotypes.file <- "genodata.csv"

Now we want to read our csv files into R

Msat.geno <- readGenData(genotypes.file)   
Msat.strata <- readGenData(strata.file)

Pick how you want to stratification schemes you want to use to group your individuals for genetic structure analysis.

strata.schemes <- Msat.strata[, c("Broad","Fine")]  
rownames(strata.schemes) <- Msat.strata$Lab\_ID

We need to change any 0’s to NAs because strataG will read 0’s as alleles rather than as missing data (obscuring QA/QC results for missing data, and possibly skew subsequent analyses).

Msat.geno[Msat.geno==0]=NA

you can use following code to check that you have removed all the 0’s

View(Msat.geno)

# Data QA/QC

Before we run any genetic analyses, we must examine several key aspects of our data in order to correctly interpret results. The following steps ensure that our genotype data are high quality and that our conclusions are robust.

Note that this package does not explicitly test genotyping error, but it can be semi-quantitatively assessed via the duplicate checks, especially if you include replicates in your sample processing. For microsatellites, it is good practice to have a second genotyper independently call some portion of your alleles to assess genotyping error, and important to only include allele calls you are confident in; for SNPs it is useful to include replicates of a subset of samples in multiple libraries and sequencing runs (if applicable).

The first few steps of QA/QC can be done with the data frames as imported above. Then we will then create a gtypes object, which is a combination of all our data in a format that is compatible with the strataG package, to use for the next steps of QA/QC and for genetic structure analysis.

## Remove problematic samples and strata

### 1. Remove samples with lots of missing genotypes

We want to remove samples that are missing data for too many loci. We traditionally generally err on the side of being conservative and only include samples with 80% of called genotypes in which we have confidence, but this may depend on the project and dataset. A sample that did not amplify for multiple markers may yield false results for markers for which it did amplify. Make this number of columns in your genotype data that are not alleles (aka sample IDs or strata)

geno.data.start<-3

Set the minimum percent of missing alleles an individual can have to be included in your analysis

min.percent <- .20

Make a list of the number of NAs in each row.

numNAs<-apply(Msat.geno, 1, function(z) sum(is.na(z)))

remove rows where there are more NA alleles than min.percent of all alleles

Msat.geno.a<-Msat.geno[!(numNAs>((ncol(Msat.geno)-geno.data.start))\*min.percent),]

### 2. Remove strata with low sample sizes

It is generally recommended that the strata you are looking for genetic structure between have similar sample sizes to avoid sampling biases affecting your results, for example because often increasing sample sizes increase genetic diversity found in a group of individuals (but, depends on biology your species, etc.). One strata with very few samples may erroneously seem to be strongly divergent from other samples. You can choose to remove strata with small samples sizes, or you can also switch to a broader stratification scheme (less groups with more samples in each group).

Check how many samples there are per “Broad” strata.

t1<-table(Msat.geno.a$Broad)  
t1  
## Australia Mexico Hawaii GUAM   
## 127 139 95 12

You can run this code to remove a specific stratum. Remember that Broad is the column name for the stratification scheme you are using.

Msat.geno.short<-subset(Msat.geno.a, Broad!= "GUAM")

All of our strata seem to have reasonable sample sizes so we will move forward with all of our strata.

## Make a gtype object for further QA/QC

strataG works with objects called gtypes. A gtype object can include several different components:

* A data frame containing the ids, strata, and allelic data as one column per locus.
* Haploid DNA sequences
* A ploidy integer representing the ploidy of the data.
* A data frame with stratification schemes in each column. Sample names are in the row names and must match the first part of the sample names (row names) of the ‘loci’ slot. Each column is a factor.
* A Description or label for the object.
* Optional related data

Msat.g <- df2gtypes(Msat.geno.a, ploidy = 2, id.col=3, loc.col=4,schemes = strata.schemes)

If you have multiple stratification schemes, you need to specify which column of your strata schemes data frame you want to stratify by use this code:

Msat.g <- stratify(Msat.g, "Broad")

## Remove duplicates

Now that we have a gtype object we want to remove duplicate genotypes. Genotypes are often duplicated to assess genotyping error rates but it is important to make sure to remove them before analyzing data to avoid type II error, etc.

Identify samples with your specified percent of shared single-locus genotypes.

dupGenotypes(Msat.g, num.shared = 0.66)

For example:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID 1 | ID 2 | Strata 1 | Strata 2 | # loci genotyped | # loci shared | Proportion loci shared |
| Sample\_1 | Sample\_13 | Location\_1 | Location\_1 | 10 | 10 | 1.00 |
| Sample\_4 | Sample\_22 | Location\_2 | Location\_2 | 10 | 10 | 1.00 |

You can review these results, decide which samples to remove, and then repeat analyses above to confirm issues have been adequately resolved.

## Loci QA/QC tests

*(N.B. this is written generally for our microsatellite example, and while the general QC concepts apply for other types,e.g., SNPs, adjustments are likely needed).*

We want to ensure that our genetic markers both worked well in our samples and that they have enough diversity to be informative. To assess how well markers worked in our samples we want to look at:

1. The success rate of genotyping (# genotypes for a locus/ number of samples) for each locus.

2. The diversity of alleles in a locus

Generally, it’s not advisable to include markers that had low genotyping success rates in your analysis. Using markers with high proportions of missing data may make populations look more structured than they really are or cause other spurious results.

To assess how useful markers are, we can look at the number of alleles the population (or specific strata) has at a locus. We can also look at a related value, allelic richness. Loci with very few alleles and low allelic richness will not be useful to assess structure between strata because most individuals in the population will have one of few possible allele combinations. We can also look at proportion of unique alleles (alleles present in only one individual in a strata) as a measure of genetic diversity at a locus. Unique alleles may be useful for assigning individuals to populations.

We should also look at both expected and observed heterozygosity. Heterozygosity is another measure of genetic diversity in a locus. Markers with low heterozygosity may not have amplified well, may have null alleles, or may just have low genetic diversity and therefore will not be very useful in analyzing population structure. In a later step we will compare observed and expected heterozygosity.

This code gives you a quick summary of many of the statistics we will calculate below. You can use it before or after you run the above steps as a quick check first look or double check.

summary(Msat.g)

Generate the number of genotypes (sample size), proportion genotyped (success rate of genotyping), number of alleles, allelic richness, proportion of unique alleles, and expected and observed heterozygosity at each marker.

qaqc(Msat.g, label = "QC\_results", num.shared = 0.9)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| locus | # genotyped | Proportion  genotyped | # of alleles | Allelic richness | Proportion unique alleles | Expected heterozygosity | Observed  heterozygosity |
| 1 | 821 | 0.856994 | 15 | 0.01827 | 0.2 | 0.866574 | 0.816078 |
| 2 | 921 | 0.961378 | 8 | 0.008686 | 0.125 | 0.719101 | 0.661238 |
| 3 | 904 | 0.943633 | 8 | 0.00885 | 0.125 | 0.643924 | 0.498894 |
| 4 | 897 | 0.936326 | 23 | 0.025641 | 0.086957 | 0.785247 | 0.638796 |
| 5 | 942 | 0.983299 | 16 | 0.016985 | 0.0625 | 0.87611 | 0.769639 |
| 6 | 950 | 0.991649 | 3 | 0.003158 | 0 | 0.40561 | 0.391579 |
| 7 | 872 | 0.91023 | 39 | 0.044725 | 0.051282 | 0.945069 | 0.900229 |
| 8 | 879 | 0.917537 | 20 | 0.022753 | 0.05 | 0.913994 | 0.591581 |
| 9 | 907 | 0.946764 | 22 | 0.024256 | 0.045455 | 0.928108 | 0.902977 |
| 10 | 902 | 0.941545 | 16 | 0.017738 | 0.0625 | 0.898616 | 0.81929 |

These initial results should help you decide which markers to keep in downstream analysis. Generally, we want markers with a high proportion genotyped, high allelic richness, and high heterozygosity (but this may vary depending on desired analyses, etc.).

## Calculating Allele Frequencies

In addition to the values calculated with the *qaqc* command, you can calculate allele frequencies if you are interested. Allele frequencies can tell you about the evenness of genetic variation in each locus. The distribution of allele frequencies at your loci can inform you about demographic factors such as population size and is used to calculate many summary statistics. Allele frequencies are used to calculate heterozygosity, calculate the power of your markers to identify individuals, determine paternity, assign individuals to a population, etc.

1. Generate allele frequencies for each locus. Each element is a matrix or array with frequencies by count (freq) and proportion (prop) of each allele.

AF<-alleleFreqs(Msat.g, by.strata = FALSE)

|  |  |  |
| --- | --- | --- |
| **Locus6** | | |
| Allele | frequency | proportion |
| 113 | 93 | 0.0566382 |
| 117 | 3 | 0.0018270 |
| 119 | 145 | 0.0883069 |
| 121 | 163 | 0.0992692 |
| 123 | 177 | 0.1077954 |
| 125 | 349 | 0.2125457 |
| 127 | 280 | 0.1705238 |
| To save a file for each locus: | | |

for(loc in names(AF)) {fname <- paste(Description, loc, "overall\_allele.freqs.csv", sep = "\_")write.csv(AF[[loc]], file = fname)}

1. Generate allele frequencies for each locus separated by strata. Each element is a matrix or array with frequencies by count (freq) and proportion (prop) of each allele.

af <- alleleFreqs(Msat.g, by.strata=T)  
print(af[1])

Locus6  
**Location\_1**   
Allele frequency proportion  
113 0 0.00000000  
117 0 0.00000000  
119 2 0.04761905  
121 5 0.11904762  
123 7 0.16666667  
125 11 0.26190476  
127 15 0.35714286  
129 2 0.04761905  
131 0 0.00000000  
  
**Location\_2**   
Allele frequency proportion  
113 3 0.012711864  
117 2 0.008474576  
119 38 0.161016949  
121 26 0.110169492  
123 38 0.161016949  
125 48 0.203389831  
127 11 0.046610169  
129 9 0.038135593  
131 34 0.144067797

To save a file for each locus:

for(loc in names(af)) {fname <- paste(Description, loc, "bystrata\_allele.freqs.csv", sep = "\_")write.csv(af[[loc]][, "freq", ], file = fname)}

## Private alleles

Private alleles are alleles only found in one population/strata. Markers with many private alleles may be useful in some analyses (such as population assignment), but may be problematic for others (e.g., markers with many private alleles, or where different strata share very few alleles, may not be useful for calculating genetic distances). Thus, it may be important to identify the number of private alleles and then interpret and make decisions based on the context and goals of your study.

Generate a table with the number of private alleles per locus in each stratum:

pA<-privateAlleles(Msat.g)   
View(pA)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Location1 | Location2 | Location3 |
| Locus1 | 2 | 0 | 0 |
| Locus2 | 1 | 0 | 0 |
| Locus3 | 1 | 0 | 0 |
| Locus4 | 2 | 0 | 0 |
| Locus5 | 0 | 1 | 0 |
| Locus6 | 0 | 0 | 0 |

## Linkage Disequilibrium

Linkage disequilibrium (LD) occurs when loci are not randomly assorted. This can be due to physical proximity on the chromosome or an artifact of selection. Markers that are linked should generally not be considered as independent when examining genetic structure because they may artificially inflate structure signals.

Importantly it is best to examine linkage disequilibrium within each strata, as genetic structure can inflate LD when considered across populations. If markers are linked in many of your strata you should pick only one of each pair of linked markers to use for population structure analyses or combine markers (e.g., PHASE for SNPs), so as not to overestimate genetic structure. If your populations have significant genetic structure, testing for LD in all populations combined may reveal artificially indicate LD as a product of selection, drift, nonrandom mating, etc. It’s good to check for LD in each of the individual populations. Markers that are linked in many populations should be excluded.

Generate initial LD estimates for entire dataset and write results to file:

LD<-LDgenepop(Msat.g)  
write.csv(LD, file = paste(Description, "\_all\_LD.csv", sep = ""))

Generate initial LD estimates within each strata and write results to file:

x <- strataSplit(Msat.g)  
  
for(g in strataSplit(Msat.g)) {  
 LD <- LDgenepop(g)  
 fname <- paste(Description,"\_LD\_", strataNames(g)[1], ".csv", sep = "")  
 write.csv(LD, fname)  
 }

## Hardy-Weinberg Equilibrium

The Hardy-Weinberg theorem states that allele frequencies should be constant over time in the absence of selection, drift, migration or other processes that shift allele frequencies. This principle allows us to calculate expected levels of heterozygosity from measured allele frequencies. Populations that are not experiencing selection, drift, migration, nonrandom mating, or mutation should have observed heterozygosities similar to expected values and are said to be in Hardy-Weinberg equilibrium (HWE). Loci may be out of HWE for multiple reasons, but generally should not be used to measure gene flow/genetic structure and as they do not meet the assumptions required for informative neutral markers (N.B., especially for SNPs, this does not apply to study questions may involving adaptive variation vs. genetic draft/gene flow).

If your populations have significant genetic structure an HWE test for all populations combined may reveal some deviation from HWE. Therefore, it is good to check for deviation from HWE in each of the individual populations. Markers that are out of HWE in many populations should be excluded, depending on the questions and data set. The following code will use a Monte Carlo procedure (random re-sampling from your data) to test for deviations from HWE for each locus. B indicates number of replicates for monte carlo simulation. If you don’t have Genepop installed you can set use.genepop = FALSE. (but note then the estimation method is different). The output will be a p-value for each marker.

To test for HWE in all strata combined:

HWE<-hweTest(Msat.g, use.genepop = TRUE, B= 1000, label = "HWE.genepop")#note   
write.csv(HWE, file = "msats\_all\_HWE.csv")

|  |  |
| --- | --- |
|  | HWE |
| Locus1 | 0.0004 |
| Locus2 | 0.0103 |
| Locus3 | 0.0000 |
| Locus4 | 0.0000 |
| Locus5 | 0.0000 |

P values < 0.05 indicate that a marker is out of HWE. We can see that when we test for HWE in the combined population many of our markers are out of HWE. Before removing them all we should test for HWE individually in each strata.

To test for HWE in each strata individually

x <- strataSplit(Msat.g)  
for(g in strataSplit(Msat.g)) {  
 hwe <- hweTest(g, use.genepop = T)  
 na <- numAlleles(g)  
 result <- cbind(num.alleles = na, hwe)  
 write.csv(result, file =paste(strataNames(g)[1], ".csv"))  
 }

For example:

|  |  |  |
| --- | --- | --- |
| Location5 | | |
| Locus | No. alleles | HWE p value |
| 1 | 10 | 0.364 |
| 2 | 6 | 0.581 |
| 3 | 4 | 0.009 |
| 4 | 15 | 0.752 |
| 5 | 14 | 0.022 |
| 6 | 3 | 0.105 |
| 7 | 24 | 0 |
| 8 | 17 | 0 |
| 9 | 20 | 0.194 |
| 10 | 12 | 0.87 |

When we test our strata separately most of our markers are in HWE (p > 0.05).

## Jackknife

Jackknife analysis calculates Hardy-Weinberg equilibriums for your population excluding each sample. This analysis identifies individuals who influence HWE.

This is a computationally intense step because it calculates HWE n-1 times. It can therefore take a long time to run, especially if you have a large dataset.

Perform HWE jackknife analysis:

Msat.JackHWE<-jackHWE(Msat.g, exclude.num = 1, min.hwe.samples = 3, show.progress = TRUE, use.genepop = F)

This code uses jackknife analysis to identify influential samples. Samples are influential if HWE p values are less than alpha when you exclude them from you HWE calculation, but HWE p values are greater than alpha when you do include these samples in your HWE calculations.

Msat.JackHWE.influential<-jackInfluential(Msat.JackHWE, alpha = 0.05)Msat.JackHWE.influential

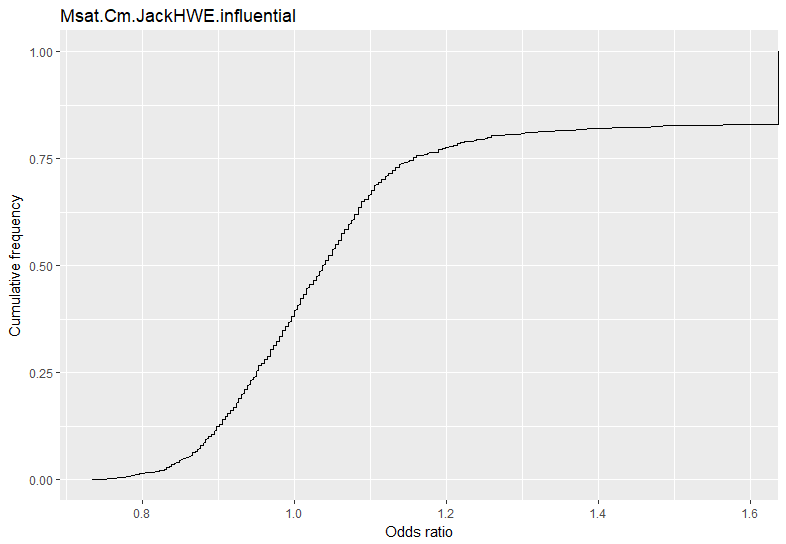
Write results of jackknife analysis to file:

write.csv(Msat.JackHWE.influential$influential, file = paste(Description, "\_JackHWE.influential.csv", sep = ""))write.csv(Msat.JackHWE.influential$allele.freqs, file = paste(Description, "\_JackHWE.allele.freqs.csv", sep = ""))write.csv(Msat.JackHWE.influential$odds.ratio, file = paste(Description, "\_JackHWE.odds.ratio.csv", sep = ""))

Create a cumulative frequency plot of all odds-ratios from jack.influential where a vertical dashed line marks the smallest influential exclusion:

pdf(paste(Description, "\_JackHWE.influential", ".pdf", sep = ""), width = 15, height = 8)plot(Msat.JackHWE.influential, main = "Msat.JackHWE.influential")dev.off()

Cumulative frequency plot of all odds-ratios from jack.influential :



The above plot shows how many of the calculations of HWE (each calculation missing a different sample) yielded which odds ratio (#of permutations in HWE/ # permutations out of HWE). From this plot it seems that most of our samples do not affect whether our loci are in HWE. If your plot shows that the majority of your permutations resulted in low odds ratios you should go back to HWE analysis of your markers to identify a marker that may be out of equilibrium.

### You should continue running QA/QC until you decide your data is satisfactory. Once you are confident in your samples and markers, then move on to desired analyses.

# Data Analysis: Population Structure

There are many analyses you can perform with genotype data. Fall into two broad categories.

1) You can define groups a priori and look at some measure of genetic distance between them (e.g., Fst)

2) Clustering methods where genetic clusters are not predefined units, but rather inferred from estimating genetic ancestry from unobserved source populations (e.g., STRUCTURE)

Here we explore a couple basic metrics used in the first category above to measure levels of genetic structure or genetic variation between strata (*N.B., the second type of analysis can also be run in strataG; we plan to add that to this guide soon*). Measures of genetic distance can be used to identify distinct population segments, assess connectivity/gene flow between groups, determine how long populations have been reproductively isolated, etc.

There are many of metrics of genetic distances, each of which has strengths and weaknesses. It is important to think about which measure is best for answering your question, because as you will see, they can sometimes be different. Briefly, F statistics (Fst, Fis) measure the partitioning of genetic distance within and among groups. Gst is similar to Fst but adapted to multialellic loci. G’st accounts for bias from hypervariable loci, and G’’st accounts for bias from sampling small populations. D measures the fraction of allelic variation among populations as an absolute rather than relatively. For additional background information, see the recent discussion of contrasting these different metrics, see [Jost et al. 2018, Evol Apps.](https://onlinelibrary.wiley.com/doi/full/10.1111/eva.12590)

The following method outputs the overall level of genetic structure between your specified strata by the measures of genetic structure you specify with stats = "...". This function calculates overall genetic structure by calculating many permutations of pairwise comparisons between strata. You can specify the number of permutations with nrep = xxx. Increasing the number of permutations you use will increase your statistical power, but will also make this more computationally intensive and slow. It is recommended to test code with 100 permutations, but use 10,000 permutations for actual analyses.

overall <- overallTest(Msat.g, stats = "all", nrep = 100, write.output=FALSE)

write.csv(overall$result, file = paste(Description, "\_overall\_test.csv", sep = ""))

|  |  |  |
| --- | --- | --- |
| Genetic distance metric | estimate | P value |
| Chi2 | 8311.402 | 0.009901 |
| D | 0.12395 | 0.009901 |
| Fst | 0.051331 | 0.009901 |
| F'st | 0.227732 | 0.009901 |
| Fis | 0.227732 | 0.009901 |
| Gst | 0.049236 | 0.009901 |
| G'st | 0.053894 | 0.009901 |
| G''st | 0.21274 | 0.009901 |
| PHIst | NA | NA |

The following code allows you to look at genetic distances between each possible pair of strata. As above, you can specify which measures of structure you want to use and how many permutations you want to use to calculate this. Increasing the number of permutations you use will increase your statistical power, but will also make this more computationally intensive and slow. It is generally recommended to test code with 100 permutations, and then use 10,000 permutations for actual analyses.

pairwise.1 <- pairwiseTest(Msat.g, stats = c("fst"), nrep = 50, write.output=FALSE)

write.csv(pairwise.1$result, file = paste(Description, "\_pairwise\_results.csv", sep = ""))

An example of the output:

|  |  |  |  |
| --- | --- | --- | --- |
| Pair label | Fst estimate | Fst p-value | |
| Pop1 (21) v. Pop2 (127) | 0.0264299 | 0.0196078 |
| Pop1 (21) v. Pop3 (30) | 0.0523936 | 0.0196078 |
| Pop1 (21) v. Pop4 (292) | 0.0523754 | 0.0196078 |
| Pop1 (21) v. Pop5 (11) | 0.0217448 | 0.0588235 |
| Pop1 (21) v. Pop6 (12) | 0.0537278 | 0.0196078 |
| Pop2 (127) v. Pop3 (30) | 0.0516723 | 0.0196078 |
| Pop2 (127) v. Pop4 (292) | 0.0387865 | 0.0196078 |
| Pop2 (127) v. Pop5 (11) | 0.0294583 | 0.0196078 |
| Pop2 (127) v. Pop6 (12) | 0.0624071 | 0.0196078 |
| *…(and so on…)* |  |  |
| You can save the Warning messages about dropped loci to check later if applicable: | | |

warnings()

There is a lot more you can do with strataG. For more information refer to the resources presented at the beginning of the guide.

*(\*\*Coming soon: STRUCTURE analyses, basic genetics graphing in R, etc.-there is currently code for this on github that you are welcome to use, we plan to create analogous tutorials soon!) j*

# StrataGUI

Additionally, all of QAQC and initial population structure tests described above can be completed in a GUI format via the StrataG Shiny App. The GUI lets you avoid writing code directly, provides some useful data QC visualizations, and an output summary report. However, for those that are comfortable coding directly may wish to use R (and RStudio if desired), because the app can be very slow with large data sets, does not allow as much flexibility or clear documentation for reproducibility, and thus can be more difficult to troubleshoot, etc. See StrtatGUI guide for details.

You can call it in R after loading the strataG package as described above, and walk through with one of the vignette datasets:

strataGUI()