Genetic and genomic analyses using RAD-seq and Stacks

Instructors:
Julian Catchen <jcatchen@illinois.edu>
Department of Animal Biology, University of Illinois at Urbana-Champaign
Josephine Paris <j.r.paris@exeter.ac.uk>
Department of Biosciences, University of Exeter

Objectives:

The goal of this exercise is to familiarize students with the use of next generation sequence data produced from Reduced Representation Libraries (RRL) approaches such as Restriction site Associated DNA (RAD-tags). These libraries are often used for genotyping by sequencing, and can provide a dense set of single nucleotide polymorphism (SNP) markers that are spread evenly across a genome. These markers are useful for a variety of genetic and genomic analyses in model and non-model organisms. Students will gain experience with a computational pipeline called *Stacks* that was designed for the analysis of such data. Data will be analyzed *de novo* to perform a population analysis without the aid of a reference genome, and from an organism with a reference genome to identify signatures of selection. *Stacks* can be used for other analyses of RAD-seq data as well, such as constructing genetic maps and phylogeography, although those are beyond the scope of this exercise.

Introduction

The advent of short-read sequencing technologies has revolutionized the study of genomic variation from complete sequences in model organisms such as nematode worms, fruit flies, zebrafish, mice and humans. In addition, the long-standing dream of biologists of having complete genomic information from numerous individuals from different populations in the lab and wild, the field of **population genomics**, is becoming a possibility for a variety of ecological and evolutionary studies.

Until just a few years ago the goal of acquiring complete genomic information from numerous individuals in many populations was out of reach for all but a small number of model organisms. For example, producing a high density genetic map for an organism required an immense investment of resources to first produce and then type the large number of genetic markers needed to adequately cover the genome. Furthermore, identifying genomic regions associated with phenotypic variation, or involved in the adaptation of organisms to novel conditions, was restricted to organisms for which re-sequencing projects produced a dense battery of genetic markers at a significant cost.

The limits to population genomic studies will gradually fade as the costs of second generation sequencing continue to drop. However, many studies using complete genome re-sequencing will not be feasible for a while because costs are still significant, high quality read lengths are still too short, and analysis remains challenging. Luckily, many population genomic studies can now efficiently be performed by using an alternative approach called genotype-by-sequencing that occurs by the sequencing of reduced representation libraries (RRL), and subsequent identification and scoring of SNPs and inference of haplotypes. Although they do not provide complete genomic information, these approaches provide a sufficient picture via data on hundreds of thousands of SNPs and haplotypes spread across a genome at a fraction of the cost of complete re-sequencing.

We developed one such approach called **R**estriction-site **A**ssociated **D**NA sequencing (RAD-seq), which has been used to identify signatures of selection, produce high density genetic maps, help assemble genomes, and be useful for studies of allelic specific transcriptional profiling. Because these data are so new, and the sample sizes of sequences often so massive, a critical related breakthrough has been the development of algorithms and software pipelines for the analysis of such data. We have produced *Stacks* for the analysis of RAD-seq data. You will learn how to analyze RAD data with and without the use of a reference genome with the goal of identifying population structure in one case and identifying signatures of selection in a second case. Through the completion of these tasks you will learn how to process RAD-seq data and use the software programs *Stacks*, *BWA and Structure*.

For more information on RAD genotyping and related methods, in particular conceptual and statistical issues, see the papers listed at the end of this document. These papers will help you better understand both the molecular biology, computational analyses, and conceptual framework for the analysis of RRL data such as RAD.

Datasets and Software

Datasets

- ***Dataset 1 (DS1)*** - This dataset comprises a subset of RAD-seq data generated from 30 individuals from three populations of threespine stickleback, each of which has been individually barcoded. The RAD-seq data were prepared using the restriction enzyme *SbfI*, and **single-end** sequenced using an Illumina HiSeq sequencer. These data are a component of the data originally published in Catchen, et al. 2013.

- ***Dataset 2 (DS2)*** - This is a set of population genomic data from the threespine stickleback. The dataset comprises 5 individuals from each of two differentiated populations, for a total of 10 barcoded individuals. The RAD data were prepared using the restriction enzyme *SbfI*, and **paired-end** sequenced using an Illumina HiSeq sequencer. These data are published in Nelson, et al. 2018.

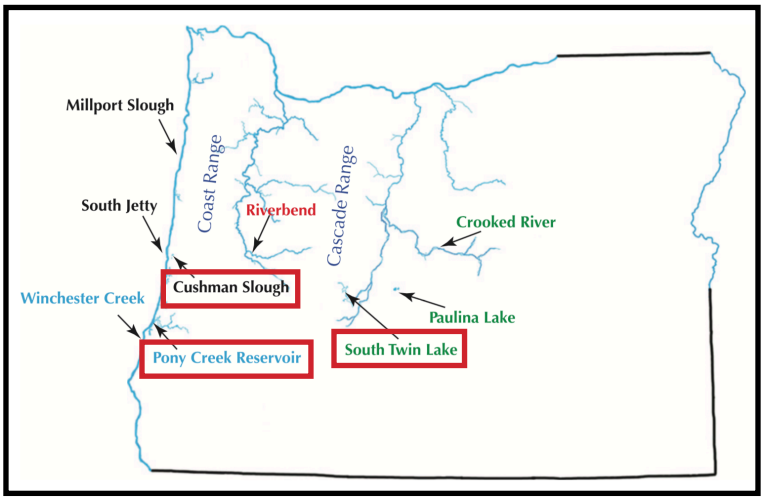
Bonus exercises

- ***Dataset 3 (DS3)*** - This data set comprises just a small proportion of a lane of single-end standard RAD data.
- ***Dataset 4 (DS4)*** - A fragment of a lane of paired-end RAD sequences that have been double-digested with two restriction enzymes.

- **Software - All are open source software**
  - ***Stacks*** **(http://catchenlab.life.illinois.edu/stacks/)** - A set of interconnected open source programs designed initially for the *de novo* assembly of RAD sequences into loci for genetic maps, and extended to be used more flexibly in studies of organisms with and without a reference genome. The pipeline has a Perl wrapper allowing sets of programs to be run. However, the software is modular, allowing it to be applied to many scenarios. You will use the Perl wrapper in class and the modules on your own.
  - ***Structure*** **(http://pritch.bsd.uchicago.edu/structure.html)** - A software program originally written by Jonathan Pritchard and colleagues that uses Bayesian stochastic models of multi-locus genotype data. The package was written to estimate the distribution and abundance of genetic variation within and among populations, patterns that are now commonly called the *genetic structure* of populations.
  - ***BWA*** **(bio-bwa.sourceforge.net/)** - *BWA* is a very fast and efficient software package used for aligning sequences against a reference genome. We will use *BWA* to align RAD reads against the stickleback reference genome, and then analyze these reads within the *Stacks* pipeline. Although we will use *BWA* for this exercise, many other algorithms and software exist for aligning against a reference genome, and these could be used in conjunction with *Stacks* as well.
  - ***Samtools*** **(www.htslib.org/)** - A program that manipulates SAM and BAM files (the primary file format that read alignments are stored) in very useful ways.
  - ***RAxML*** **(http://sco.h-its.org/exelixis/web/software/raxml/index.html)** - A software program written by the Exelixis Lab for the construction of maximum likelihood phylogenetic trees.

Exercise 1. Stacks parameter optimisation

In this first exercise we will be working on a subset of data from threespine stickleback data from Oregon (DS1), on the west coast of the United States. These stickleback can be found in a number of habitats from costal marine and freshwater habitats, to inland river habitats, to high mountain lakes in the interior of Oregon. We want to understand how these populations relate to one another and in this exercise, you will examine three of these populations: a coastal marine population, a costal freshwater, and an inland river population.



Stickleback populations sampled from Oregon, USA in Catchen, et al., 2013. Populations in red are sampled for this tutorial.

Without a reference genome, we want to assemble the RAD loci and examine population structure. However, before we can do that, we want to explore the *de novo* parameter space in order to be confident that we are assembling our data in an optimal way. As you have seen, stack formation is controlled by three main parameters: `m` (the minimum read depth); `M` (the number of mismatches between alleles) and `n` (the number of mismatches between loci in the catalog). Here, we will optimize `M` for the stickleback data using a subset of the full dataset provided. After this, we can use the optimal value we have found for `M` in the *de novo* exercise below (part 2). We will be using the guidelines of parameter optimization as outlined in Paris *et al.* (2017), and will create a 'hockey stick' plot to assess which value for `M` recovers the highest number of new polymorphic loci (*r80* loci).

1. 30 minute mini-lecture on *Stacks,* primary/secondary reads, and parameter optimisation.
2. In each exercise you will set up a directory structure on the remote server (in this case our Amazon Virtual Machine) that will hold your data and the different steps of your analysis.
3. Firstly, we will make the directory ~/`working` on the cloud to hold these analyses. Be careful that you're reading and writing files to the appropriate directories within your hierarchy. You'll be making many directories, so stay organized!

- Each step of your analysis goes into the hierarchy of the workspace, and each step of the analysis takes its input from one directory and places it into another directory, this is known as a '**waterfall workspace**'. We will name the directories in a way that correspond to each stage and that allow us to remember where they are. A well organized workspace makes analyses easier and prevents data from being overwritten.

4. In your ~/working workspace, create a directory called **denovo** to contain all the data for this exercise. Inside that directory, create two additional directories: **samples**, and **opt**. To save time, we have already cleaned and demultiplexed these data and will start from the cleaned samples stage. We have already prepared the clean samples for this exercise.

Unarchive dataset 1 (DS1):

```
/home/genomics/workshop_materials/stacks/denovo/oregon_stickleback.tar
```

into the **samples** directory. The unarchived dataset contains 30 stickleback samples (DS1), and we will use 9 of them in this exercise (remaining samples will be used in the de novo exercise)

```
cs_1335.01, cs_1335.02, cs_1335.05, pcr_1211.04, pcr_1211.05,
pcr_1211.06, stl_1274.33, stl_1274.35, stl_1274.37
```

5. denovo_map.pl will run ustacks, cstacks, and sstacks on the individuals in our study as well as the populations program.

- Information on denovo_map.pl and its parameters can be found online:
  - http://catchenlab.life.illinois.edu/stacks/comp/denovo_map.php

- We want Stacks to only use nine individuals for our parameter optimization. To specify this, create a file in the working directory called *opt_popmap*, using a text editor (nano or gedit). Have a look at the manual for information on population maps:
  - http://catchenlab.life.illinois.edu/stacks/manual/#popmap
  - The file should be formatted like this

```
<sample file prefix><tab><population ID>
```

- Include all 9 samples in this file and specify that all individuals belong to one population (i.e. put "one" or any word you like in the population ID column). You must supply the population map to denovo_map.pl when you execute it for each parameter run.

6. Get into groups of six (three pairs). Each person take one value for the M parameter (for values M2, M3, M4, M5, M6 and M7). Run the Stacks denovo_map.pl pipeline program, with each person using a different value of M. Choose between yourselves!

- To optimise for *r80* loci you will need to tell denovo_map.pl to use an "Arbitrary command line option" using the -X option so it will pass additional options (-r parameter) to the populations program. Run iterations for M=2, M=3, M=4, M=5, M=6 and M=7. Change the corresponding value for n so that we follow the n=M rule.

All on one line:

```
denovo_map.pl -M value -n value --samples dir --popmap path -o
              dir –X "populations: -r 0.80" -T 2
```

[Once you get `denovo_map.pl` running, it will take approximately 10 minutes.]

7. You should now have the `denovo_map.pl` output files for each iteration of M (M2, M3, M4, M5, M6 and M7). In your groups, work together to determine the number of *r80* loci that were assembled for each value. To obtain how many *r80* loci were assembled for each parameter run you will want to look at the `populations.hapstats.tsv` file.

Have a look in the Stacks manual to inform you on the data contained in each of the columns in this file: http://catchenlab.life.illinois.edu/stacks/manual/#pfiles

- What is the number of the first locus assembled for `M2`?
- What is the number of the last locus assembled for `M7`?
- Use unix to count how many loci were assembled for each iteration of `M`.

```
cat populations.hapstats.tsv | grep -v "^#" | cut -f 1 | sort |
uniq | wc -l
```

- Which iteration of `M` provided the highest number of *r80* loci?

8. We now want to explore how many new *r80* loci were found between each iteration for `M`. Using the number of *r80* loci from the `populations.hapstats.tsv` file, count how many new loci were assembled with each iteration of M and record them in a text file like so:

```
M2/M3<tab>xxx
M3/M4<tab>xxx
M4/M5<tab>xxx
M5/M6<tab>xxx
M6/M7<tab>xxx
```

Save this file as r80_loci.tsv.

9. Copy the Gnuplot script to the opt directory:

```
/home/genomics/workshop_materials/stacks/denovo/
    hockey_stick.gnuplot
```

- Cat the file to see what it does.

10. Put the `r80_loci.tsv` file in the same directory as the `hockey_stick.gnuplot` script.

- Execute Gnuplot:

```
% gnuplot < hockey_stick.gnuplot
```

- Open the PDF to examine the hockey stick plot. Based on the plot and the number of *r80* loci, which value for `M` do you think is most appropriate for the Oregon stickleback data?

Exercise II. *de novo* assembly of RAD tags without a genome

1. In this second exercise we will now be working on the full set of threespine stickleback data (DS1) sampled from throughout Oregon, on the west coast of the United States. These data consist of three populations: a coastal marine population, a costal freshwater, and an inland river population.
2. Now that we have optimized the assembly parameters, we will assemble loci and determine population structure using the Structure program. *For more information on the study this data originated with, see Catchen, et al. 2013.*
3. In your `~/working/denovo` workspace, create a directory called `stacks` to contain the assembled data for this exercise.
4. Run the Stacks' `denovo_map.pl` pipeline program. This program will run `ustacks`, `cstacks`, and `sstacks` on the individuals in our study as well as the `populations` program.
    - A reminder that the information on `denovo_map.pl` and its parameters can be found online:
        - http://catchenlab.life.illinois.edu/stacks/comp/denovo_map.php
    - We want Stacks to understand which individuals in our study belong to which population. To specify this, create a file in the working directory called *popmap*, using an editor. The file should be formatted like this:

        `<sample file prefix><tab><population ID>`

        This time, include all 30 samples in this file and specify which individuals belong to which populations (cs, pcr, stl). You must supply the population map to `denovo_map.pl` when you execute it.
    - There are three important parameters that must be specified to `denovo_map.pl`, the *minimum stack depth* (`m`), the *distance allowed between stacks* (`M`), and the *distance allowed between catalog loci* (`n`). Use the values we determined for these parameters in the previous exercise.
    - Also, you must set the `stacks` directory as the output, and use all the threads available on your instance (four threads).
    - Finally, specify the path to the directory containing your sample files. The `denovo_map.pl` program will read the sample names out of the population map, and look for them in the samples directory you specify.
    - **Execute the Stacks pipeline.**

        [Once you get `denovo_map.pl` running, it will take approximately 30 minutes.]

5. Examine the *Stacks* log and output files when execution is complete.

- After processing all the individual samples, `denovo_map.pl` will print a table containing the depth of coverage of each sample. Find this table in the log, what were the depths of coverage?
- Familiarize yourself with the population genetics statistics produced by the `populations` component of stacks:
    - populations.sumstats.tsv, populations.sumstats_summary.tsv
        - What is the mean value of nucleotide diversity (π) and F_{IS} for each of the three populations? [The `less -S` command may help you view these files easily.]

6. Our goal now is to export a subset of loci for analysis in *Structure*, which analyzes the distribution of multi-locus genotypes within and among populations in a Bayesian framework to make predictions about the most probable population of origin for each individual. The assignment of each individual to a population is quantified in terms of Bayesian posterior probabilities, and visualized via a plot of posterior probabilities for each individual and population. A key user defined parameter is the hypothesized number of populations of origin which is represented by **k**. Sometimes the value of **k** is clear from from the biology, but more often a range of potential **k**-values must be explored and evaluated using a variety of likelihood based approaches to decide upon the ultimate **k**. In the interest of time we won't be exploring different values of **k** here, but this will be a key step in any data analysis. In addition, at the moment *Structure* can only handle a small number of loci because of the MCMC algorithms involved in the Bayesian computations, usually many fewer than are generated in a typical RAD data set. We therefore want to randomly choose a random subset of loci that are well represented in our three populations. Nonetheless, this random subset contains more than enough information to define population structure.

- The final stage of the `denovo_map.pl` pipeline is to run the `populations` program to calculate population genetic statistics for our data. We want to execute this program by hand again, specifying filters that will give us only the most well represented loci.
    - Run `populations` again, specifying that loci must be present in at least 80% of individuals in all three populations. You will have to tell `populations` where to find the output of the *Stacks* pipeline (this should be in the `stacks` output directory).
    - One final detail: *Structure* assumes that each SNP locus is independent, so we don't want to output multiple SNPs from the same RAD locus, since they are not independent but are in linkage blocks within each RAD tag. We can achieve this behavior by specifying the `--write_single_snp` parameter to `populations`.

7. Now we want to select 1,000 loci randomly from the results and save these loci into a file. We can easily do this using the shell given a list of catalog IDs output in the previous step. The `populations.sumstats.tsv` file gives a list of all polymorphic loci. Use the `cat`, `grep`, `cut`, `sort`, `uniq`, `shuf`, and `head` commands to generate a list of 1000 random loci. Save this list of loci as a *whitelist*, that we can feed back into `populations`.

```
cat populations.sumstats.tsv | grep -v "^#" | cut -f 1 | sort -n
          | uniq | shuf | head -n 1000 > whitelist.tsv
```

8. Now execute `populations` again, this time feeding back in the whitelist you just generated. This will cause `populations` to only process the loci in the whitelist. Specify that a Structure output file be included this time and again insist that only a single SNP is output from each RAD locus. Finally, you will need to again specify the population map that you generated above to `populations` so that this information is passed into the *Structure* output file.

9. Create a new directory called `structure` and copy the *Structure* output file that Stacks generated to this directory.

   - Edit the *Structure* output file to remove the comment (first line in the file, starts with "#").
   - You may need to edit your Structure output file to change the alphanumeric population names to be numbers. This can be done using `sed`.
   - The parameters to run *Structure* (including a value of **k=3**) have already been prepared, you can find them here:

     `/home/genomics/workshop_materials/stacks/denovo/mainparams`

     and

     `/home/genomics/workshop_materials/stacks/denovo/extraparams`

     Copy them into your `structure` directory as well.

10. Execute *Structure*, saving the data into this new directory:

    ```
    structure > populations.structure.console
    ```

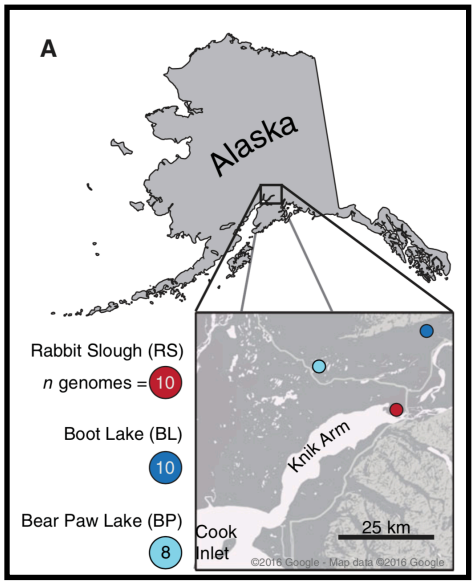
    [A common STRUCTURE error happens when your population output contains fewer than 1000 loci. You may need to adjust the number of loci in the `mainparams` file to match your exact Stacks output.]

11. You will need to load the `populations.structure.console` and `populations.structure.out_f` files into the graphical interface for *Structure* in Guacamole. Select the "File" menu and then "Load structure results…" to load the Structure output. Choose the "Barplot" menu and then "Show".

    - Are the three Oregon threespine stickleback populations related to one another? How can you tell?

Exercise III. Population genomics with a reference genome

Population genetics is a very old field that has a rich mathematical theory and a core set of statistical approaches for inferring parameters from genetic data. These statistics are such things as nucleotide diversity (**π**), differentiation statistics (i.e. $\mathbf{F_{ST}}$), and measures of genetic covariance such as Linkage Disequilibrium (**D** and **D'**). However, because of methodological limitations, the majority of the theoretical, statistical and empirical work in population genetics has focused on a small number of loci. With the advent of second generation sequencing, tens or hundreds of thousands of genetic markers can now be examined in dozens of individuals, allowing the field of population genomics to truly come to fruition. An exciting new activity in population genomics is the identification of signatures of selection in wild populations. Today you will process RAD data from one oceanic and one freshwater population of threespine stickleback from Cook Inlet (DS2), which is located on the coast of Alaska, USA. One set of data comes from an ancestral oceanic population, whereas the other is from a derived freshwater population that is just a few thousand years old. We will align these data to the stickleback reference genome using *BWA*, and then feed the alignments into *Stacks*. After *Stacks* determines the loci and associated alleles present in each population, we will export the data and calculate several population genomic statistics, including F_{ST}. Performing a study like this was nearly impossible before the advent of next generation sequencing. *For more information on population genomics, see the papers listed in that section at the end of this document.*



Stickleback populations sampled from Alaska, USA in Nelson, et al., 2017. Populations in red and dark blue are sampled for this tutorial.

1. Acquire and process DS2 (Cook Inlet).

- In your `./working` workspace, create a directory called `scan` to contain all the data for this exercise. Inside that directory, create three directories: `samples`, `aligned`, and `stacks`. To save time, we have already cleaned and demultiplexed this data set and will start from the cleaned samples stage.
- Unarchive DS2 from

  `/home/genomics/workshop_materials/stacks/scan/cookinlet_stickleback.tar`

  into the `samples` directory.

2. Align the stickleback sequences against the genome with *BWA*.

- Run *BWA* on the single and paired-end data of the first freshwater sample: `samples/BT_2827.01`

  [Running *BWA* could take 10-20 minutes total.]

- Running *BWA* with no parameters will give you a list of the major options.
- We will use the `mem` algorithm, running `bwa mem` will give you a further list of options for this particular algorithm.

  Some hints for command line parameters: *BWA* can read gzipped input files by default so no need to unzip them; make sure you use all the threads available on your instance (four threads). Although we can control the stringency of matching in a lot of detail, it will be fine to accept the default matching parameters. Name the output file the same as the input file, with a "`.sam`" suffix instead of "`.[12].fa.gz`".
- The stickleback genome has already been indexed by *BWA* into a searchable database is located within this directory:

  `/home/genomics/workshop_materials/stacks/scan/bwa_db`

  The BWA database is stored in several files within that directory, and **we must specify the common prefix of those files** to BWA.
- When alignment is complete, use `samtools` to convert the SAM output file to a BAM file. Delete the SAM file.
  - Why do we convert the SAM file to a BAM file, what is the advantage?
  - Why was our `cookinlet_stickleback.tar` file not also gzipped?
- Sort the BAM file using `samtools`.
- Run *BWA* again with the first oceanic sample: `samples/RS_1827.05` and convert it and sort it again to a BAM file using `samtools`.
- To save time, the remaining 8 alignments can be found here:

  `/home/genomics/workshop_materials/stacks/scan/cookinlet_aligned_key.tar`

  `Untar` these remaining *BWA* alignments into the `aligned` directory.

3. We next want to run *Stacks* on the freshwater and anadromous population.

- Run the *Stacks* `ref_map.pl` pipeline program. This program will run `gstacks` on the members of the population, accounting for the alignments of each read.
  - Information on `ref_map.pl` and its parameters can be found online:
    - http://catchenlab.life.illinois.edu/stacks/comp/ref_map.php
  - We need to create a population map that tells Stacks which individuals belong to which biological population. Create a file in the working directory called `popmap` that is formatted like this:

    `<sample file prefix><tab><population ID>`

    Include all 10 samples in this file and specify which individuals belong to which populations and specify it to the `ref_map.pl` program. A sensible set of population IDs might be "`fw`" for the freshwater samples, and "`oc`" for the oceanic (anadromous) samples.

- Specify the `stacks` directory as the output location.
- Finally, specify the path to the directory containing your sample files. The `ref_map.pl` program will read the sample names out of the population map, and look for them in the samples directory you specify.
- **Execute the pipeline.**

- Once *Stacks* has completed running, investigate the output files and the log files.

4. The program `populations` calculates population genetic statistics for each SNP in the two populations for one level of population subdivision, as we have here. So, it will calculate expected and observed heterozygosity, π, F_{IS}, and it includes F_{ST} as a measure of genetic differentiation between populations. It uses the same method for calculating F_{ST} as was used in the human HapMap project.

- Now look at the output in the file `populations.sumstats.tsv`. There are a large number of statistics calculated at each SNP, such as the frequency of the major allele (*P*), and the observed and expected heterozygosity, and F_{IS}. Use UNIX commands like `head`, `zcat`, `cut`, `more`, `column`, and `sort` to focus on the minimum and maximum heterozygosity and F_{IS} statistics. Are these statistics in the same genomic locations within the genome between the two populations? Optional: How are these summary statistics related to Hardy-Weinberg equilibrium?

- What are the population mean and standard error for π in the two populations? (Check the `populations.sumstats_summary.tsv` file.)

5. Because RAD produces so many genetic markers, and because we have a reference genome sequence, we can examine population genetic statistics like F_{ST} as continuous distributions along the genome. The populations program does this using a kernel-smoothing sliding window approach.

- Run the `populations` program again, this time turning on kernel smoothing for F_{ST} (`--smooth_fstats`). Also, turn on filters so we only include loci available in both populations that are found in 75% of individuals of each population (`-r` 0.75). Be sure to again specify the population map you created previously and turn on multithreading (four threads).
  - Information on `populations` and its parameters, as well as examples of how to execute it, can be found online:
    - http://catchenlab.life.illinois.edu/stacks/comp/populations.php
- The output file `populations.fst_summary` contains the mean F_{ST} measure between populations. What is the mean F_{ST} between the marine and freshwater populations?
- The output file `populations.fst_fw-oc.tsv` contains F_{ST}, a measure of genetic differentiation between the two populations. What is the maximum value of F_{ST} at any SNP? How many SNPs reach this F_{ST} value?

- Look at the genomic distribution of F_{ST} in the file `populations.fst_fw-oc.tsv`. Use UNIX commands like `cut` and `sort` to find the genomic regions that show the highest levels of population differentiation.
  - What does the p-value generated by Fisher's exact test tell you about a particular F_{ST} score?
- Now plot F_{ST} over a single linkage group. First use `grep` to produce a new F_{ST} file with only data for Linkage Group IV (labeled **groupIV**), call it `populations.fst_fw-oc_groupIV.tsv`. Now plot this file using gnuplot.
- Copy the Gnuplot script to the `stacks` directory:

  `/home/genomics/workshop_materials/stacks/scan/fst_groupIV.gnuplot`
- Cat the file to see what it does.
- Execute Gnuplot:

```
% gnuplot < fst_groupIV.gnuplot
```

- Open the resulting PDF file and open it. The red crosses represent the raw F_{ST} measures while the green line is the kernel-smoothed average value.

6. While so far we have viewed our RAD data as a set of SNPs, since we have assembled the paired-end reads we can view each RAD locus as a set of haplotypes (composed of one or several SNPs). Stacks has implemented another set of haplotype-based populations genetic statistics. One of these is Φ_{ST}, which is an AMOVA-based F statistic designed for haplotypes. We can also look at these smoothed values on the genome.

- Look at the genomic distribution of Φ_{ST} in the file `populations.phistats_fw-oc.tsv`. Use UNIX commands like `cut` and `sort` to find the genomic regions that show the highest levels of population differentiation.
- Use `grep` to produce a new Φ_{ST} file with only data for Linkage Group IV (labeled **groupIV**), call it `populations.phist_fw-oc_groupIV.tsv`. Now plot this file using gnuplot. Copy the Gnuplot script to the `stacks` directory:

  `/home/genomics/workshop_materials/stacks/scan/phist_groupIV.gnuplot`
- Cat the file to see what it does.
- Execute Gnuplot:

```
% gnuplot < phist_groupIV.gnuplot
```

- Download the resulting PDF file and open it. The red crosses represent the raw F_{ST} measures while the green line is the kernel-smoothed average value.
- How do the smoothed values for F_{ST} and Φ_{ST} differ?

Bonus Exercise 1. Data preparation

1. In `working`, create a directory called `clean` to contain all the data for this exercise. Inside that directory create two additional directories: `lane1` and `samples`. We will refer to the `clean` directory as the *working directory*.
2. Unarchive data set 3 (DS3):

   `/home/genomics/workshop_materials/stacks/clean/lane1.tar`

   to the `lane1` directory.
3. You can copy the file to your working directory and use `tar` to unarchive it, or you can change to your working directory and *untar* it without moving the file (this will save you time and will dump the unarchived files into the directory you are currently in).
4. Your decompressed files has millions of reads in it, too many for you to examine in a spreadsheet or word processor. Examine the contents of the set of files in the terminal (the `head`, `more`, and `tail` commands may be of use).
   - You should see multiple different lines with different encodings.
     - How does the FASTQ file format work?
     - How are quality scores encoded? (See the Wikipedia entry for the FASTQ format.)
     - How could you tell by eye which type of encoding your data are using?
5. You probably noticed that not all of the data is high quality. In general, you will want to remove the lowest quality sequences from your data set before you proceed. However, the stringency of the filtering will depend on the final application. In general, higher stringency is needed for *de novo* assemblies as compared to alignments to a reference genome. However, low quality data will almost always affect downstream analysis, producing false positives, such as errant SNP predictions.
6. We will use the Stacks's program `process_radtags` to clean and demultiplex our samples.
   - Take advantage of the Stacks manual as well as the individual manual page for `process_radtags` on the Stacks website to find information and examples:

     http://catchenlab.life.illinois.edu/stacks/manual/#specbc

     http://catchenlab.life.illinois.edu/stacks/comp/process_radtags.php
   - You will need to specify the set of barcodes used in the construction of the RAD library. Remember, each P1 adaptor in RAD has a particular DNA sequence (an inline barcode) that gets sequenced first, allowing data to be associated with samples such as individuals or populations.
   - Enter the following barcodes into a file called `lane1_barcodes` in your working directory (make sure you enter them in the right format, i.e. one barcode per line and watch out for whitespace!):

- AAACGG AACGTT AACTGA AAGACG
- AAGCTA AATGAG ACAAGA ACAGCG
- ACATAC ACCATG ACCCCC ACTCTT
- ACTGGC AGCCAT AGCGCA

[Use the nano editor to create your file.]

- Assign a sample name for each barcode. Normally, these sample names would correspond to the individuals used in a particular experiment, but for this exercise, we could name the samples in a simple way, say `indv_01`, `indv_02`, etc.
- Copy the remaining barcodes for this lane of samples from the file:

  `/home/genomics/workshop_materials/stacks/clean/lane1_barcodes`

  and append them to your barcodes file in your working directory.
  - You can concatenate this file onto the end of your file using the cat command and the shell's append operator: `cat file1 >> file2`, or you can cut+paste.
  - Based on the barcode file, how many samples were multiplexed together in this RAD library? (The `wc` command can tell you this.)
- You will need to specify the restriction enzyme used to construct the library (*SbfI*), the directory of input files (the `lane1` directory), the list of barcodes, the output directory (`samples`), and specify that `process_radtags` *clean*, *discard*, and *rescue* reads.
- The `process_radtags` program will write a log file into the output directory. Examine the log and answer the following questions:
  - How many raw reads were there?
  - How many were retained?
  - Of those discarded, what were the reasons?
- In the `process_radtags` log file what can the list of "sequences not recorded" tell you about the barcodes analyzed and about the sequencing quality in general?
  - If you found that something is possibly missing from your `process_radtags` input, correct the error and re-run the `process_radtags`.

Bonus Exercise 11. Data preparation

1. We will now work with DS4. These data contain paired-end reads that have been double-digested and dual barcoded. Each set of paired reads contains an inline barcode on the first read, and an indexed barcode on both reads. These are known as *combinatorial barcodes* as many unique combinations can be made from pairs of barcodes.

- In ./`working/clean`, create a directory called `lane2` to contain the raw data for this exercise and create the directory `ddsamples` to contain the cleaned output.
- Unarchive data set 2 (DS2):

`/home/genomics/workshop_materials/stacks/clean/lane2.tar`

into the `lane2` directory.

2. Examine the contents of the pairs of files in the terminal again.
   - How are the FASTQ headers related between pairs of files?
   - Can you identify the indexed barcode in the FASTQ header?
3. We will again use the Stacks' program `process_radtags` to clean and demultiplex our samples.
   - You will need to specify the set of barcode pairs used in the construction of the RAD library.
   - Enter the following barcodes into a file called `lane2_barcodes` in your working directory (make sure you enter them in the right format and watch out for whitespace!):
     - `AACCA/ATCACG` `CATAT/ATCACG` `GAGAT/ATCACG`
     - `TACCG/ATCACG` `AAGGA/CGATGT` `CAACC/CGATGT`
     - `GACAC/CGATGT` `TACGT/CGATGT`

     [Use the `nano` editor to create your file.]
   - Modify your barcodes file by adding a third column to it, specifying a human-readable name for each sample (instead of having the output files named after the barcodes). As we saw in the previous exercise, these sample names would normally coincide with your particular experimental design. Here, for simplicity, we can just use `indv_01`, `indv_02`, etc.
   - Copy the remaining barcodes for this lane of samples from the file:

     `/home/genomics/workshop_materials/stacks/clean/lane2_barcodes`

     and append them to your barcodes file in your working directory.
     - You can concatenate this file onto the end of your file using the cat command and the shell's append operator: `cat file1 >> file2`, or you can cut+paste.
     - How many samples were multiplexed together in this RAD library? (The `wc` command can tell you this.)
   - You will need to specify the two restriction enzymes used to construct the library (*NlaIII and MluCI*), the directory of input files (the `lane2` directory), the list of barcodes, the output directory (`ddsamples`) and specify that `process_radtags` *clean*, *discard*, and *rescue* reads.
   - The `process_radtags` program will write a log file into the output directory. Examine the log and answer the following questions:
     - What is the purpose of the four different output files for each set of barcodes?
     - How many raw reads were there?
     - How many were retained?

Citations and Readings

Protocols for running a Stacks analysis

Paris, J. R., Stevens, J. R., & Catchen, J. M. (2017). Lost in parameter space: a road map for stacks. Methods in Ecology and Evolution, 188, 799–14.

Rochette, N. C., & Catchen, J. M. (2017). Deriving genotypes from RAD-seq short-read data using Stacks. Nature Protocols, 12(12), 2640–2659.

Sources for data sets in the tutorial

Catchen, J. et al. 2013a. The population structure and recent colonization history of Oregon threespine stickleback determined using restriction-site associated DNA-sequencing. **Molecular Ecology**, 22:2864–2883.

Nelson, T. C. and Cresko, W. A. 2018. Ancient genomic variation underlies repeated ecological adaptation in young stickleback populations. **Evolution Letters**, 2: 9-21.

McCluskey, B. M., & Postlethwait, J. H. (2015). Phylogeny of zebrafish, a "model species," within Danio, a "model genus". **Molecular Biology and Evolution**, 32(3), 635–652.

Core readings in preparation for the lecture and workshop

Amores, A., et al. 2011. Genome evolution and meiotic maps by massively parallel DNA sequencing. **Genetics** 188:799-808.

Andrews, K. R., Good, J. M., Miller, M. R., Luikart, G., & Hohenlohe, P. A. (2016). Harnessing the power of RADseq for ecological and evolutionary genomics. **Nature Reviews Genetics** 1–12.

Arnold, B. et al. 2013. RADseq underestimates diversity and introduces genealogical biases due to nonrandom haplotype sampling. **Molecular Ecology** 22: 3179–3190.

Catchen, J. et al. 2011. *Stacks*: building and genotyping loci de novo from short-read sequences. **G3: Genes, Genomes and Genetics** 1:171-182.

Catchen, J. et al. 2013b. Stacks: an analysis tool set for population genomics. **Molecular Ecology** 22:3124–3140.

Davey, J. W., et al. 2011. Genome-wide genetic marker discovery and genotyping using next-generation sequencing. **Nature Reviews Genetics** 12:499-510.

Davey, J. W. et al. Special features of RAD Sequencing data: implications for genotyping. **Molecular Ecology** 22: 3151–3164.

Etter, P. D., et al. 2011. SNP Discovery and Genotyping for Evolutionary Genetics using RAD sequencing. *in* Molecular Methods in Evolutionary Genetics, Rockman, M., and Orgonogozo, V., eds.

Ekblom, R., and J. Galindo. 2010. Applications of next generation sequencing in molecular ecology of non-model organisms. **Heredity** 107:1-15.

Hohenlohe, P. A. et al. 2010. Population genomics of parallel adaptation in threespine stickleback using sequenced RAD tags. **PLoS Genetics** 6: 1-23.

Lescak, E. A., Bassham, S. L., Catchen, J., gelmond, O., Sherbick, M. L., Hippel, von, F. A., & Cresko, W. A. (2015). Evolution of stickleback in 50 years on earthquake-uplifted islands. **Proceedings of the National Academy of Sciences** 112(52), E7204–12.

Population genomics background, concepts and statistical considerations

Cariou, M. et al. 2013. Is RAD-seq suitable for phylogenetic inference? An *in silico* assessment and optimization. **Ecol Evol** 3, 846–852.

Gompert, Z., and C. A. Buerkle. 2011a. A hierarchical Bayesian model for next-generation population genomics. **Genetics** 187:903-917.

Hohenlohe, P. A., et al. 2010. Using population genomics to detect selection in natural populations: Key concepts and methodological considerations. **International Journal of Plant Sciences** 171:1059-1071.

Luikart, G.,et al. 2003. The power and promise of population genomics: from genotyping to genome typing. **Nature Reviews Genetics** 4:981-994.

Lynch, M. 2009. Estimation of allele frequencies from high-coverage genome-sequencing projects. **Genetics** 182:295-301.

Nielsen, R., et al. 2005. Genomic scans for selective sweeps using SNP data. **Genome Research** 15:1566-1575.

Nielsen, R., et al. 2011. Genotype and SNP calling from next-generation sequencing data. **Nature Reviews Genetics** 12:443-451.

Rubin, B. E. R. et al. 2012. Inferring Phylogenies from RAD Sequence Data. **PLoS ONE** 7, e33394–e33394.

Stapley, J., et al. 2010. Adaptation genomics: the next generation. **Trends in Ecology and Evolution** 25:705-712.

Empirical studies using RRL and RAD sequencing

Altshuler, D., et al. 2000. An SNP map of the human genome generated by reduced representation shotgun sequencing. **Nature** 407:513-516.

Baxter, S. W., et al. 2011. Linkage mapping and comparative genomics using next-generation RAD sequencing of a non-model organism. **PLoS ONE** 6:e19315.

Chutimanitsakun, Y., et al. 2011. Construction and application for QTL analysis of a Restriction Site Associated DNA (RAD) linkage map in barley. **BMC Genomics** 12: 1-13.

Emerson, K. J., et al. 2010. Resolving postglacial phylogeography using high-throughput sequencing. **Proceedings of the National Academy of Sciences** 107:16196-16200.

Gore, M. A., et al. 2009. A first-generation haplotype map of maize. **Science** 326:1115-1117.

Richards, P. M. et al. 2013. RAD-Seq derived markers flank the shell colour and banding loci of the *Cepaea nemoralis* supergene. **Molecular Ecology** 22, 3077–3089.

RAD-seq genotyping methodology

Baird, N. A., et al. 2008. Rapid SNP discovery and genetic mapping using sequenced RAD markers. **PLoS ONE** 3:e3376.

Etter, P. D., et al. 2011. Local De Novo Assembly of RAD Paired-End Contigs Using Short Sequencing Reads. **PLoS ONE** 6:e18561

Hohenlohe, P. A., et al. 2011. Next-generation RAD sequencing identifies thousands of SNPs for assessing hybridization between rainbow and westslope cutthroat trout. **Molecular Ecology Resources** 11 Suppl 1:117-122.

Miller, M. R., et al. 2007. Rapid and cost-effective polymorphism identification and genotyping using restriction site associated DNA (RAD) markers. **Genome Research** 17:240-248.

Willing, E. M., et al. 2011. Paired-end RAD-seq for de novo assembly and marker design without available reference. **Bioinformatics** 27:2187-2193.

Related reduced representation library (RRL) methodologies

Andolfatto, P., et al. 2011. Multiplexed shotgun genotyping for rapid and efficient genetic mapping. **Genome Research** 21:610-617.

Elshire, R. J., et al. 2011. A Robust, Simple Genotyping-by-Sequencing (GBS) Approach for High Diversity Species. **PLoS ONE** 6:e19379.

Peterson, B. K. et al. 2012. Double digest RADseq: an inexpensive method for de novo SNP discovery and genotyping in model and non-model species. **PLoS ONE** 7, e37135.

Rigola, D., et al. 2009. High-Throughput Detection of Induced Mutations and Natural Variation Using KeyPoint™ Technology. **PLoS ONE** 4:e4761.

van Orsouw, N. J., et al. 2007. Complexity reduction of polymorphic sequences (CRoPS): a novel approach for large-scale polymorphism discovery in complex genomes. **PLoS ONE** 2:e1172.

van Tassell, C. P., et al. 2008. SNP discovery and allele frequency estimation by deep sequencing of reduced representation libraries. **Nature Methods** 5:247-252

Wang, S. et al. 2012. 2b-RAD: a simple and flexible method for genome-wide genotyping. **Nature Methods** 9, 808–810.