

# AftrRAD Version 4.1 User Manual

AftrRAD is a bioinformatic pipeline consisting of perl and R scripts that is designed to aid in analyzing restriction site associated DNA sequence (RADseq) data. If you have questions about AftrRAD that are not addressed in this manual, two additional resources include the FAQ file and the AftrRAD google group (<https://groups.google.com/forum/#!forum/aftrrad>).

## Contents

I.	Before you use AftrRAD.....	p.2
II.	Running AftrRAD.....	p.5
	a. AftrRAD.pl.....	p.5
	b. Genotypes.pl.....	p.7
	c. FilterSNPs.pl.....	p.9
III.	Evaluating your RADseq dataset.....	p.10
IV.	Formatting Data Files.....	p.12

## Before you start using AftrRAD, make sure...

- 1.) You're using a Mac or Linux machine. Note that both work, but the program usually runs faster on Macs. Windows version hopefully coming soon.
- 2.) You're analyzing single-end reads. Currently the only option for paired end data is to analyze each end separately.
- 3.) If you have undemultiplexed data with inline barcodes (see Fig 1), and the barcodes have variable lengths, they must still each be unique when trimmed down to the length of the shortest one.
- 4.) Your reads are all the same length. If you have multiple sequencing runs of varying lengths, I'd suggest trimming them all down to the shortest for AftrRAD analyses.
- 5.) You have plenty of free memory available on your computer (on the order of 10's to 100's of GB to be safe). Some of the temporary files created by AftrRAD can get rather large, depending on the size of the datafile(s) you're analyzing. This is rarely a problem, but we've had a couple of cases where computers have gotten nearly filled up with Illumina data file storage before the AftrRAD run, and there wasn't sufficient free memory to complete the run.
- 6.) You have the following dependencies correctly installed and working...
  - a. ACANA -> Download this program from: <https://www.niehs.nih.gov/research/resources/software/biostatistics/acana/index.cfm> and copy the two executable files 'ACANA' and 'dnaMatrix' into your working AftrRAD directory. In a Terminal window, move to this directory and type ./ACANA to make sure it's working.
  - b. Mafft -> Get it at: <http://mafft.cbrc.jp/alignment/software/>. It should be in your path after you download it. To make sure it's working, simply type mafft in the terminal window from any directory and the program should run (it should ask for an input file).
  - c. R -> Available from [www.rproject.org](http://www.rproject.org)

```
@HWI-ST1052:100:D1F28ACXX:6:1101:1947:2130 1:N:0:ATCACG
CTCAGTTGCAGGCCACCCAAGCGGCTAAGGACCTGGCCAGAGCAGAAACA
+
BCCFFFFFFHGHGDIEGGHHGIGIJJJGIGIIIIJJJGGHIGIIIIJIIIEIG
```

**Figure 1. Example read in fastq format. In default mode, AftrRAD expects each read to begin with an in-line barcode (bold, black), followed by a restriction enzyme recognition site (red, bold). However, data that are already demultiplexed can be analyzed by setting the command line argument *dplexedData* to '1'.**

\*Note that a small example data file is available at [u.osu.edu/sovic.1/downloads](http://u.osu.edu/sovic.1/downloads) if you want to make sure everything is installed correctly before running your (presumably larger) dataset.

**After everything is downloaded and installed, do one of the following:**

**A.) If your data are not already demultiplexed, and have inline barcodes as in Fig 1:**

- 1.) Add all of your data files (in fastq format) to the “Data” subdirectory.
- 2.) Add all of your barcode files to the “Barcodes” subdirectory. These barcode files are text files that contain barcodes and their associated sample names, separated by a tab (Figure 2). **The barcode files must have the exact same name as the data file they are associated with (including the extension)**, and there should be nothing else in the barcode directory (extra files in the Data directory should be OK). Note that hidden characters can cause problems with how AftrRAD reads the barcode files. I suggest a good text editor such as JEdit, TextWrangler, or nano in Terminal to create these files.

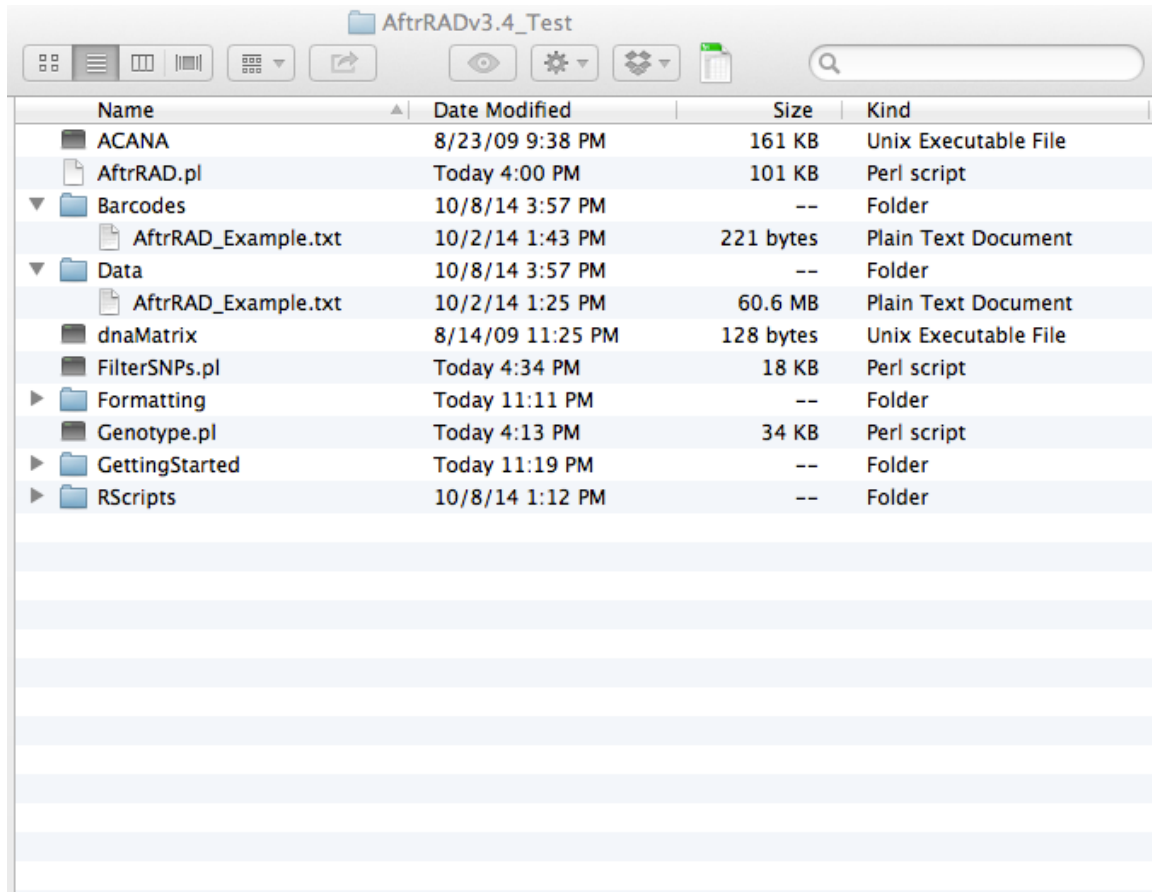
```
CTAGTC CA_KBPP142
CTTCTC IN_RYSN156
CTGTGT IN_WLSN155
ACACCT OH_PRDF098
CAACTC CA_BEAU118
CAAGTG CA_BPNP124
GATCTC CA_KBPP181
GAGACT IN_RYSN158a
GACTGA IN_WLSN157
CAGTCT NY_CCR0132
ACCAAG OH_PRDF100
AGACCA CA_BEAU103
TCGATC CA_BPNP458
GAGTCA MI_DNLP167
```

**Figure 2. Example format of a barcode file – plain text file with barcode <tab> sample name <return>**

**B.) If your data are already demultiplexed, without inline barcodes as in Fig 1:**

- 1.) Create a folder in your working directory named ‘DemultiplexedFiles’ and add all of your demultiplexed data files (in fastq format) to this folder. I suggest naming these files with their respective sample name, as these will be used to create the barcode file.
- 2.) When you run AftrRAD.pl (see below), make sure to include the command line argument ‘dplexedData-1’.

At this point, your working directory (with undemultiplexed data) should look something like this...



Name	Date Modified	Size	Kind
ACANA	8/23/09 9:38 PM	161 KB	Unix Executable File
AftrRAD.pl	Today 4:00 PM	101 KB	Perl script
Barcodes	10/8/14 3:57 PM	--	Folder
AftrRAD_Example.txt	10/2/14 1:43 PM	221 bytes	Plain Text Document
Data	10/8/14 3:57 PM	--	Folder
AftrRAD_Example.txt	10/2/14 1:25 PM	60.6 MB	Plain Text Document
dnaMatrix	8/14/09 11:25 PM	128 bytes	Unix Executable File
FilterSNPs.pl	Today 4:34 PM	18 KB	Perl script
Formatting	Today 11:11 PM	--	Folder
Genotype.pl	Today 4:13 PM	34 KB	Perl script
GettingStarted	Today 11:19 PM	--	Folder
RScripts	10/8/14 1:12 PM	--	Folder

**Figure 3. Example of working directory before running AftrRAD with undemultiplexed data. If your data are already demultiplexed, the files should be in a folder named DemultiplexedFiles, and the Barcodes and Data folders in the figure are not needed.**

**\*\*\*If you have questions beyond what is covered in this manual, please check the FAQ document and/or the AftrRAD google group.**

# Running AftrRAD

## Part I: AftrRAD.pl

In the terminal window, cd to your working directory. AftrRAD can be run by typing “**perl**<space>**AftrRAD.pl**”. There are several parameters that need to be set for each run. All have default values, which can be edited by including arguments on the command line when starting an AftrRAD run. The parameters are...

re	The restriction enzyme recognition sequence that occurs in your reads. In Figure 1 above, this is TGCAGG, which corresponds to a digestion with the enzyme SbfI. This is the default value. If there is no restriction enzyme recognition site in your reads, enter 0.
minQual	The minimum quality (Phred) score for retaining reads. Reads that contain bases with scores below this value are removed from the analysis. The default value is 20.
minDepth	The minimum mean nonzero read depth to retain a read. For each unique read in the dataset, the counts of that read will be obtained for each sample. The average of the nonzero counts will be obtained, and checked against the value entered here. Any reads with values less than this threshold are assumed to be error and are eliminated from the dataset. Default value is 5.
minIden	The minimum percent identity (after alignment) to consider two reads to be alternative alleles from the same locus. The default is 90%.
numIndels	The maximum number of indels allowed between any two reads to consider them alternative alleles from the same locus. The default is 3.
P2	The beginning of the P2 adaptor sequence. Reads containing this string are removed from the analysis. If you don't want to search for the P2 adaptor, you can enter “noP2” here, or any other string of characters that will not be found in any of your reads. The default is ATTAGATC.
minParalog	The minimum number of reads that must occur at a third allele in an individual to flag a locus as paralogous. Default is 5.
Phred	The quality score methodology used in the sequencing. Most often Phred33, which is the default. An alternative is Phred64.
dplexedData	If data are demultiplexed prior to the AftrRAD run, set this to ‘1’. In this case, the data files must be stored in a folder named ‘DemultiplexedFiles’.
stringLength	Reads containing strings of homopolymers of this length will be removed. Default is 15.
DataPath	Path to directory containing fastq data files for the run. Default is the Data directory in the AftrRAD working directory.
BarcodePath	Path to directory containing barcode files. Default is the Barcodes directory in the AftrRAD working directory.
MaxH	Maximum proportion of samples allowed to be heterozygous at a locus. Loci for which heterozygosity exceeds this value will be flagged as paralogous. Default is 90%.

Arguments to change any of the default values are entered after the 'perl AfrRAD.pl' command, with the argument and its value separated by a dash. For example, the following command would start an AfrRAD run, and would change the defaults for the restriction enzyme recognition sequence and the maximum number of indels allowed between two alleles at a locus to ATCACG and 4, respectively...

```
perl AfrRAD.pl re-ATCACG numIndels-4
```

The first step in AfrRAD.pl is quality filtering of the reads (among a few other things). An update ("Filtering sequence X") will print to the screen after completion of every 1000000 reads. The first update will usually appear within the first minute. After the initial quality filtering, the main function of this script is locus identification. The runtime generally ranges from less than 30 minutes for small datasets to up to a few days for large datasets, especially on Linux systems, which are slower with this step. The main factors determining the length of the run are... 1.) the total number of sequencing reads in the dataset, 2.) the number of polymorphic loci in the dataset. To date, we've analyzed datasets up to ~300 individuals and >50,000 total loci.

## Part II: Genotype.pl

After running the AftrRAD.pl script, run Genotype.pl by typing “perl<space>Genotype.pl”. The following parameters can be set for this script in the same way as above.

MinReads	The minimum coverage required at a locus in an individual to apply a binomial test and call a genotype. Loci with fewer reads than this value will be scored as missing data for the sample. Default is 10.
pvalLow	For each locus in each individual, a binomial test is applied to score the sample as heterozygous or homozygous. The assumption is that the two alleles in a heterozygote will be sequenced in equal frequencies. So, for a locus at which an individual has read counts of 35 and 50 for two alleles, respectively, a binomial test with a probability of success of 0.5 gives a p-value of 0.064. If this value is greater than pvalLow, the locus is scored as heterozygous. Alternatively for two read counts of 25 and 1, the p-value is $7.75e-7$ , and this will be scored as homozygous for the allele with 25 reads, as long as the pvalLow is $>7.75e-7$ . Default is 0.0001.
pvalHigh	Same as pvalLow, but allows for a different p-value threshold for loci that have relatively high total counts. For example, a locus with read counts of 350 and 500 gives a p-value of $1.50e-7$ , which is close to the 1 vs 25 test above. pvalHigh is applied when the total number of reads at the locus is greater than or equal to pvalThresh. Default is 0.00001.
pvalThresh	The threshold number of reads at a locus in each binomial test that determines whether pvalLow or pvalHigh is used as the critical p-value. Default is 100. With this default, tests based on $<100$ reads will use the p-value set as ‘pvalLow’, while those with $>100$ read will use ‘pvalHigh’.

The first major function of this script is to score genotypes for all loci in each individual. These genotyped loci are printed to a file named Genotypes.txt in the TempFiles directory.

Next, the script will quantify the amount of missing data at polymorphic loci in each of the samples, and flag any samples that have  $>2$  standard deviations more than the average amount of missing data. You’ll be asked whether to include these samples in the remainder of the analyses. **Use the flagged samples as a guide only.** The proportions of missing data for each individual are recorded in “MissingDataProportions.txt”, and counts of loci with missing data for each individual are plotted in a PDF file named “MissingDataCounts”. Both of these are in the “Output/RunInfo” directory. It’s a good idea to check these files, and base your decisions off of them. There may be additional samples you want to remove (i.e. – just under 2 stdev greater than the mean), or you may decide to keep some of the flagged samples. You’ll have the option to do this. If you choose to remove additional individuals, simply type the names of the samples you want to remove, with each separated by a tab. Note that ‘Individual’ is added to the beginning

of each name in some of the output files – don't include this when typing in samples to remove.

Third, this script produces a file named 'SNPLocations.pdf' in the Output/RunInfo folder that plots the locations of each SNP along the read. This provides some insight into whether artifactual SNPs have built up toward the end of the reads, and if so, at what read position this begins to be a problem. This information is used when running the next script, FilterSNPs.pl

Total runtime for Genotype.pl is usually <30 min, but depends on the number of individuals/loci in the dataset.



## Part III: FilterSNPs.pl

After running the Genotypes.pl script, run “FilterSNPs.pl” by typing “**perl**<space>**FilterSNPs.pl**” at the command prompt. The following parameters can be set for this script.

- |           |  |
|-----------|--|
| pctScored | Percent of individuals that must be genotyped in order to retain the locus (1-100). The default is 100, which means all samples must be genotyped in order to retain the locus.  |
| maxSNP    | The maximum location along the reads to score SNPs. This value should be chosen based on the plot in the file Output/RunInfo/SNPLocations.pdf. The default is 0, which prints all SNPs.  |
| MinReads  | As in Genotypes.pl, the minimum coverage required at a locus in an individual to score a genotype. In FilterSNPs.pl, this value is needed when deciding which monomorphic loci to output. Default is to use the value used in the most recent run of Genotypes.pl. |

FilterSNPs.pl filters the data based on two criteria that are set by the pctScored and maxSNP arguments, respectively. The script outputs the major results from the AfrRAD run. Each time FilterSNPs.pl is run, it will output three files in the Output/Genotypes directory. The first is called SNPMatrix\_X.Y.txt, with X being the percent entered, and Y being the max read location for SNPs selected. This file contains inferred genotypes for all of the SNPs identified (note that if a given locus has multiple SNPs, each will appear in a separate column in the matrix, but they will share the same locus number). The second file is called Haplotypes\_X.Y.txt, which is similar to SNPMatrix, but combines the SNPs at each locus into a haplotype. Both of these files are tab delimited, and can be viewed easily in a program such as Excel. The third file is named Monomorphics\_X.txt, and it contains all of the monomorphic loci that were scored in at least X% of the samples, along with information on the read counts for each sample for these loci.

# Evaluating Your RADseq Dataset

AfrRAD outputs a variety of files that can be helpful in evaluating the quality of your RAD run(s). Some of these are described below...

- 1.) Output/RunInfo-> “BarcodeInfoX”: There will be one of these files for each fastq file you started with. This file provides information on the number of reads assigned to each individual (barcode) in the run (demultiplexing). The main thing to look for here is evenness in counts across individuals. Samples with extremely low counts will probably need sequenced deeper, or possibly re-prepped, depending on the reason for the low counts. Any samples with extremely low read counts are likely to have been flagged as potentially bad samples during the Monomorphics.pl run, and may be best omitted from the analyses.
- 2.) Output/RunInfo-> “ReportX”: There will be one of these files for each fastq file you started with. These provide information relating to the initial quality filtering of the data (i.e. numbers of retained reads used for analyses).
- 3.) Output/RunInfo-> “MasterReport.txt”: This file reports the total number of monomorphic, polymorphic, and paralogous loci identified from your dataset, and also the average and median read depth across genotyped loci.
- 4.) TempFiles/RawReadCountFiles-> “RawReadCount\_NonParalogous\_X”: The program creates one of these files for each sample analyzed. The counts for each allele are the only thing that will vary for the different samples. These are the counts used in the binomial tests for genotyping (the two highest if there are more than two). Scrolling through these files can give you a good sense of what kind of read depths you are working with. Note that the current default in Genotype.pl is to only score genotypes when there is a minimum of 10 reads at the locus. So, in this case, an individual with two alleles with counts 6 and 2, respectively, would not be genotyped (scored as missing data). If you have a lot of missing data in your dataset, patterns observed in these files can be extremely valuable in pinpointing the reason for the missing data. As you scroll through a sample, if most of the loci have data, but the read depths are relatively low (i.e less than 10 reads at the locus, as described above), then simply resequencing the library to add more depth will likely address this issue. However, another pattern that sometimes occurs is that many loci will have total depths of zero, while the loci that are scored have relatively high read counts. This suggests that the missing data is due to problems with the library prep, and resequencing the library is probably not going to benefit you very much.
- 5.) Output/Results-> “Haplotypes\_X.Y.txt”: You can have multiple Haplotypes files, with different values of “X” and “Y” - you will get one of these files each time you run FilterSNPs.pl. The “X” and “Y” in the file names refer to the criteria you set for the minimum proportion of individuals genotyped (X), and the maximum SNP position along the read (Y) in each run of FilterSNPs.pl. This file can be opened in Excel, or any text editor. Comparing the number of loci (columns in the Excel file),

to the total number of polymorphic loci from “MasterReport.txt” above will give you some indication of the level of missing data in your dataset (a more direct place to look for this information is the “NAProportionsPerIndividual” file, and it’s associated plot “MissingDataCounts.pdf”, which are both in the RunInfo folder).

- 6.) Output/RunInfo->”MissingDataProportions.txt”: This file reports the proportion of missing data in each sample based on polymorphic loci. Interpreting the absolute values of these data can sometimes be a little tricky. If the numbers are all relatively low (i.e. <5-10%), this is good (there will always be some level of missing data due to null alleles caused by polymorphism in restriction sites). If most samples have relatively low amounts of missing data, but a subset of samples have higher levels, this likely suggests there’s a problem with those specific samples, and you might consider leaving them out of the analyses. However, sometimes the missing data values can be relatively high across all samples. There are a number of reasons for this. First, it could be insufficient sequencing coverage (see item 4 above). Second, it could result from bad library preps (again, see 4 above). A final possibility is that one, or a few samples in the dataset have a large number of unique loci, elevating the missing data values for the other samples. As a hypothetical example, imagine a scenario where a dataset contains 20 samples, 19 of which were size selected for fragment sizes of 300-450 bp. Size selection for the 20<sup>th</sup> sample inadvertently targeted 300-550. In this case, all of the loci between 450 and 550 have been introduced to the dataset, but will not be scored in the majority of the samples, raising their level of missing data.

If you are comfortable with your dataset based on the above checks, you can use the scripts in the Formatting directory to prepare input files for a variety of downstream analyses.

## Formatting Data Files

A number of scripts are provided that prepare input files for a variety of downstream analyses, such as Structure, Genepop, etc. These are located in the Formatting directory. Move to this formatting directory, and simply call the script you want to run with “perl<space>ScriptName.pl”. Most of these scripts use files in the Output/Genotypes folder as starting points. If you have run FilterSNPs.pl multiple times with different values for missing data and SNP locations, the script will likely prompt you to specify which file you want to format. This will be one of the Haplotypes files (i.e. Haplotypes\_100.35.txt) or one of the SNPMatrix files (i.e. SNPMatrix\_100.35.txt). Most of these should generally run pretty quickly (<10 min each). We’re still working on these formatting scripts on an as-needed basis. Some of them will currently only work with data that are scored in all individuals. Some brief descriptions of the files currently available...

### OutputStructure.pl

Uses a Haplotypes\_X.Y.txt file as input, and allows for missing data, which is coded in the Structure input files as “-9”. Output is file called ‘StructureInput.txt’, which should be ready to load into Structure (each sample is represented with two lines). Note that this script will arrange the samples in alphabetic order, so consider this when naming samples if you want them grouped in a specific order for the Structure run.

### OutputGenepop.pl

Uses a Haplotypes\_X.Y.txt file as input, and allows for missing data. Output is file called ‘GenepopInput.txt’. As with the Structure script, this script will arrange the samples in alphabetical order. This can be especially helpful with this script (if the samples are named with this in mind), as samples must be ordered by population in the Genepop input file. At least a bit of manual editing is required before loading the GenepopInput.txt file into Genepop. First, if your samples are not arranged by population, you’ll have to do this rearrangement manually. Then, separate each population with the POP flag. See Genepop documentation for more information.

### OutputSNAPP.pl

Uses a SNPMatrix\_X.Y.txt file as input, and allows for missing data. Produces a nexus file named “Beauti\_Infile.nex” that can be loaded into Beauti to create an xml file to run in SNAPP. This script includes command line arguments that allow you limit the SNAPP file to a subset of your samples. If you choose to include a subset of your samples, set the command line argument ‘subset’ to 1 and add a text file named “IncludedSamples.txt” to the Formatting directory that contains the names of the samples to include, with each on its own line.

### OutputFastSimCoal\_SingleSFS.pl

Uses a SNPMatrix\_X.Y.txt file as input, and does not allow any missing data. Also, note that you will likely have to do some manual editing to the SNPMatrix file prior to running this script. First, the dataset must include an outgroup sample (this creates an

unfolded SFS), and this outgroup individual must be the first sample in the SNPMatrix file (directly below the line with the locus names). This script assumes there is only one population in the dataset. If you have more than one population, use `OutputFastSimCoal_JointSFS.pl`. Output will be a single SFS named `_DAFpop0.obs`. Note that you will need to add a prefix to this name that corresponds to the name of the model you are running – see FSC documentation for more info. Command line arguments are available to resample the dataset to allow for nonparametric approaches to generating confidence intervals on parameter estimates.

### **OutputFastSimCoal\_JointSFS.pl.**

Similar to `OutputFastSimCoal_SingleSFS.pl` in that an outgroup sample must be first in the SNPMatrix file. This script also requires that your ingroup samples be arranged by population in the SNPMatrix file. Keep track of the number of samples (diploid individuals – not alleles) in each population, as you’ll be prompted to enter these numbers during the run. Output will be one or a series of files with the name `jointDAFpopX_Y.obs`, with X and Y referring to the population numbers, respectively. One file is created for each pairwise comparison of populations.

### **ScoreDuplicates.pl**

Not really a formatting script, but it’s in with these anyway. It’s a good idea to run a subset of the samples in your dataset in duplicate to assess levels of repeatability in your genotyping. In this case, the duplicate samples need to have different names in the barcodes file of the AftRAD run. For example, say you run one library that contains ‘Sample1’ with the barcode ATACAT. Then you run a second library that has the same individual, but this time, it has the barcode CCAGAG. If you want to treat these as separate samples for the purpose of evaluating repeatability, then the barcode files should have them labeled with different names, such as ‘Sample1’ and ‘Sample1b’. After the AftRAD run, create a file called “Replicates.txt” that identifies pairs of replicate samples. Each pair of names is on a separate line, and the two names are separated with a tab.

Sample1	Sample1b
Sample2	Sample2b

Put this file in the Formatting directory with the script “ScoreDuplicates.pl” and then run this script. The output file will be called “Duplicate\_Report.txt”. It will have the following columns...

Comparison: The two sample names being compared.

Sites Compared: The number of sites compared – sites with missing data in one or both of the samples are not included so this value will usually vary slightly from one comparison to another.

Number Matches: The number of matching genotype calls.

Proportion Matches: The proportion of scored sites called consistently in the two samples.

Mismatches\_Homozygous\_In\_Sample1: In most cases, mismatches will be homozygous in one of the two samples (they can be genotyped as heterozygous for different alleles, but this is rare). The expectation is that the mismatches observed will be roughly evenly distributed between the two replicates in terms of which sample is homozygous (based on our experience, the homozygote is generally the incorrect call). Regardless, comparing this value to the comparable one for Sample 2 can be helpful, as biases toward one or the other may indicate problems with a sample, and possibly a library.

Mismatches\_Homozygous\_In\_Sample2: Similar to above.

Loci\_With\_Both\_Reps\_Homozygous: Loci that were compared in which both replicate samples were homozygous.

Loci\_With\_At\_Least\_One\_Het: Loci that were compared in which at least one of the two replicates was heterozygous. This value could be used (conservatively) as the denominator when calculating the proportion of mismatches.