**Analyzing RAD data with RADTools**

This tutorial was made with RADTools0.13.6

Contact [kebi@berkeley.edu](mailto:kebi@berkeley.edu) for latest version

Tutorial by Alex Krohn

You will need to download and install these programs and their dependencies:

cutadapt, cope, bowtie2, trimmomatic, fastQC (optional), GATK, Picard, cd-hit, RepeatMasker, novosort, novoindex, seqtk, bcf tools

This tutorial was created based on 100bp SE reads used for population genomics analyses. The process will be different for PE data or phylogenetic analyses.

N.B. If you have multiple Illumina indexes (as is likely in ddRAD), complete steps 1-3 separately for each index, then rename the demultiplexed files something that distinguishes the re-used adapters. If you have multiple species, you can process them all together until step 5 (unless you’re doing phylogenetic analyses, where they should be processed together throughout).

1. Pre-clean adapters. This step formats the heading for each read and does very high level trimming/removal of reads.

Use:

perl RADTools0.13.6 preClean \*.fastq.gz

1. Demultiplex. This step demultiplexes your RAD reads based on their individual adapters (called barcodes or indexes in RADTools). It will create files named by the adapter sequence that contain the demultiplexed reads with the adapters removed. You can allow for mismatches in the adapter sequence. Most of the RAD adapters differ in at least 2 places, so it is recommended to allow 1 mismatch.

This step requires a file with the adapter sequences (-b, confusingly referred to as indexes in 13.6). It should have each adapter sequence on a separate line, one for each one used.

Use:

perl RADTools0.13.6 demultiplex -a /pre-clean/\*.fq -b (your barcode file) -d 1 -c HS

1. Filter. This step filters reads based on whether or not they have the proper cut site and then outputs the reads without the cut site. The documentation in the perl script is very straightforward.

Use:

perl RADTools0.13.6 filter …

1. CleanSE. This step removes contaminated reads, trims the ends of reads, removes reads with runs of bases or NN’s. This step requires two files. First, the “barcode” file a file containing your entire P7 (Illumina) index (confusingly called a barcode). The file should look something like this, with the second line being the one that varies depending on your P7 index:

>P7\_index1

CAAGCAGAAGACGGCATACGAGATCGGTCTCGGCATTCCTGCTGAACCGCTCTTCCGATCT

The second file contains two tab-separated columns with the library information. The first column (labeled lib) contains the entire name of each file that will be cleaned. Include only the part before \_R1.fq in this column. The second column (labeled P7) has a 1 for each file being cleaned. Below is an example where, obviously, <tab> should be replaced with a tab.

lib <tab> P7

EBRARK002\_AAACGG <tab> 1

EBRARK002\_AACGTT <tab> 1

…

For the command, use -f, -o, -a, -b, -t, -c, -e, -i, and -l. All others can be left as defaults, unless you want to change them. -t should be the full path to the .jar file. -c should be the entire FASTQ genome of whatever species you think is most likely to contaminate your samples (usually *E. coli*). -e should be the read length minus the adapter sequence and the cut site.

This will output two files per input file, one \*\_1*\_*final.fastq file that continues to the next step and another \*.contam.out file for your reference of which reads did not pass filters.

1. contig. This step will take much longer to run. Here you build contigs of reads for each individual in order to construct loci and align them to a de-novo reference genome. If you have multiple species in your library, here is where you should separate them. This step also removes repetitive sequences from know repetitive libraries (see RepeatMasker for more information).

The documentation in the perl script is sufficient for this step. Use -a, -E, -f, -R, -c, -G and -P (the latter two even if -E = 1).

-a should be the output of cleanSE, -f should be the same as -e above, -R should be the most closely related RepeatMasker library available to your organism (use the exact spelling as in the perl script, including quotes), -c can be lowered to a lower limit of 0.92 and -G and -P should be full paths.

1. popRad. This step finally creates a de-novo reference genome and aligns your individuals’ sequences to the reference genome.

Use -a, -s, -b, -G, -P, -s, -c, -u, -k, -e, -h, -H, -l, -x, and -S.

-a should be the ouput of contig. -b should be the output of cleanSE. -G and -P should be full paths. -c can be left at a default, but can also be increased/decreased depending on your situation. -u and -k are the most important here. Playing with these will greatly influence your results. Their equivalents are also in Stacks. -u should not be less than 3, and, generally, -k should not be a number of individuals less than 70% the total number of individuals. -e, -S and -h should be 0 for RAD data. -H should be 1e-4, -l can be decreased to 5.

This step will output a folder called PopGen\_reference. In this you will find your reference genome (Reference.fasta), with a header line naming each locus followed by a line with that locus’ reference sequence. You will also find the alignments of your individuals to the reference genome in BAM format in the alignment folder. Importantly, there is a file called Sites\_passed\_filters.keep, which indicates which of the sites in your reference file actually pass the -e, -S, -h, and -H filters. When at all possible, you should only include these sites in your analyses by using the -sites flag in ANGSD. There is also a .vcf file for all of your reference loci.

1. From here you can continue with ANGSD.

Good luck!