2bRAD *de novo* scripts: a manual

October 14, 2014 – September 13, 2015

These are the printouts from scripts when run without arguments, in order of their appearance in the walkthrough.

---

ngs_concat.pl :

```
concatenates files by matching pattern in their names

arg1: common pattern for files
arg2: perl-like pattern in the filename to recognize,
       use brackets to specify the unique part, as in
      "FilenameTextImmediatelyBeforeSampleID(.+)FilenameTextImmediatelyAfterSampleID"

Example (to concatenate files names like Sample_Pop1_L1.fastq, Sample_Pop1_L2.fastq):
ngs_concat.pl 'Sample' 'Sample_(.+)_L'
```

2bRAD_trim_launch_dedup.pl :

```
Prints out list of calls to trim2bRAD_2barcodes_dedup.pl, one for
each fastq file, to to trim and deduplicate raw 2bRAD reads.

See help for trim2bRAD_2barcodes_dedup.pl for more details.

prints to STDOUT

Arguments:
arg1, required: glob to fastq files
site=[pattern] perl-style pattern of the restriction site to recognise, in single quotes.
            default is BcgI: \'.{12}CGA.{6}TGC.{12}|.{12}GCA.{6}TCG.{12}\'
            if you need AlfI, here is how to define it: \'.{12}GCA.{6}TGC.{12}\'
adaptor=[dna sequence] adaptor sequence to look for on the far end of the read.
            Default AGATC
sampleID=[integer] the position of name-deriving string in the file name
                  if separated by underscores, such as:
                  for input file Sample_RNA_2DVH_L002_R1.cat.fastq
                  specifying arg2 as \'3\' would create output
                  file with a name \'2DVH.trim'
barcode2=[integer] length of the in-line barcode immediately following
                     the restriction fragment. Default 0.

Example:
2bRAD_trim_launch_dedup.pl fastq sampleID=3 > trims
```

---

trim2bRAD_2barcodes_dedup.pl :

```
This script does three things:

- Filters 2bRAD fastq reads to leave only those with a 100% matching restriction site,
degenerate 5'-leader, secondary 3'-barcode, and adaptor on the far 3'end,
trims away everything except the IIb-fragment itself;

- Deduplicates: removes all but one read sharing the same 64-fold degenerate
leader, the first 34 bases of the insert sequence, and secondary barcode
(this results in 128-fold dynamic range: 64-fold degeneracy x 2 strand orientations);

- Splits reads into separate files according to secondary barcode.

Writes trimmed fastq files named according to the secondary barcodes detected.

Arguments:
input=[fastq filename]
site=[perl-style pattern for restrictase site]
            Default is BcgI: \'.{12}CGA.{6}TGC.{12}|.{12}GCA.{6}TCG.{12}\'
bc2=[perl-style barcode pattern] in-line barcode that immediately follows the RAD
                                  fragment, default \'[ATGC]{4}\'
adaptor=[far-end adaptor sequence] default AGATC
clip=[integer] number of bases to clip off the ends of the reads, default 0
minBCcount=[integer] minimum count per in-line barcode to output a separate file.
                             Default 100000.
sampleID=[integer] the position of name-deriving string in the file name
                              if separated by underscores, such as:
                              for input file Sample_RNA_2DVH_L002_R1.cat.fastq
                              specifying arg2 as \'3\' would create output
                              file with a name \'2DVH.trim'

Example:
trim2bRAD_2barcodes_dedup.pl input=Myproject_L8_G3.fastq sampleID=2
```

---

uniquerOne.pl :

```
Makes uniqued 2bRAD read file for a single fastq file.
This is analogous to making 'stacks' in STACKS.
The script records unique tag sequences, the total
number of their appearances, and how many of those
were in reverse-complement orientation.
(STACKS would consider reverse-complements as separate loci)

prints to STDOUT

arg1: fastq filename

Example:
uniquerOne.pl G4_TCAG.trim >G4_TCAG.trim.uni
```

---

mergeUniq.pl :

```
Merges individual uniqued files produced by uniquerOne.pl into one table.

arg1: common extension of uniqued files
minDP=[integer] minimum sequencing depth to consider a unique tag. Default 5.

prints to STDOUT

Example:
mergeUniq.pl uni >mydataMerged.uniq
```

---

uniq2loci.pl :

```
assembles locus-annotated table from uniquer and cd-hit results
filters by read depth and strand bias (to avoid memory issues in genotype calling)
flips orientation of cluster members to match the most abundant tag in a cluster

saves cluster members seen just once and their references (singl.tab file) to use for
error model development (requires previous running of mergeUniq.p with minDP=1)

arg1: uniquer file
arg2: cd-hit .clstr file
arg3: cd-hit .fasta file

minDP=[integer] minimum read depth (total counts for the tag in the whole dataset).
                Default: 5
minSB=[integer] strand bias cutoff ( min(direct:reverse,reverse:direct)x100 ).
                Default: 5

prints to STDOUT

Example:
uniq2loci.pl mydataMerged.uniq cdh_alltags.fas.clstr cdh_alltags.fas >cdh_alltags.ul
```

---

haplocall_denovo.pl :

```
Calls genotypes (as population-wide RAD-tag haplotypes) in de novo 2bRAD data.

Outputs two files in VCF v.4 format:
    "Vhap": whole RAD tags recoded as alternative alleles (for coalescent analysis
            and methods that take advantage of multi-allelic markers);
"Variants": RAD tags broken into individual SNPs, within-tag phase recorded
            (for methods requiring biallelic markers).

The QUAL field in VCF files is allele bias, the average fraction of minor allele
counts in a heterozygote. Ideally this should be close to 50, more if the allele is
found in homozygotes as well.

Recorded variant info fields:
SB : Strand bias (ratio x100)
AB : Minor allele bias in a heterozygote (ratio x100)
TP : SNP position in the tag
NS : Number of samples with data
DP : Combined read depth
AF : Allele frequency

Individual genotype fields:
```

```
GT : Genotype
GQ : Phred-scaled genotype quality
PS : Phase set (for multiple SNPs within a RAD tag)
AD : Allele depth
DP : Read depth

Homozygote quality is calculated as phred-scaled
p(error) = 0.5^(n-1), where n is the number of reads;
Heterozygote quality is the same plus 20 on phred-scale
(we are very sure about our heterozygotes, see allele filters below)

The ends of tags with more than one SNP in the terminal 3 bases
are clipped off since such SNPs are most likely due to indel near the
end of the tag.

Required arguments and defaults:

arg1              input file (produced by uniq2loci.pl)

Optional arguments:

clip=0            number of bases to clip off the ends of reads

----- Allele filters -----

aobs=5           min number of times an allele must be observed across all samples.

strbias=10        strand bias cutoff: minimal ratio between reverse and
                  direct reads, x100. 0 would mean no filtering.

abias=10          allele bias cutoff: minimal fraction of reads per locus corresponding
                  to the candidate allele, averaged across individuals
                  containing the allele, x100.

ind=2             minimum number of individuals in which a candidate new allele
                  has to be found.

----- Locus filters -----

found=0.25        fraction of individuals in which a locus has to be genotyped.
                  Filter for this parameter later with vcftools.

hetero=0.5        maximum allowed fraction of heterozygotes. Guards against lumped
paralogous loci.

mono=toss         whether to keep non-polymorphic loci. Specify mono=keep to keep them
                  (if coalescent analysis is planned downstream).

----- Genotype filters -----

mindp=3           minimum depth to call a homozygote.


Mikhail Matz, July 2013 - April 2015
matz\@utexas.edu

Example:  haplocall_denovo.pl cdh_alltags.ul
```

replicatesMatch.pl : (version 0.6, September 13 2015)

```
Selects polymorphic variants that have identical genotypes among replicates.

Genotypes of two replicates should be identical or missing; the maximal fraction of
genotype pairs with missing genotypes is controlled by missing=  argument.

Arguments:

vcf=[file name]  input vcf file
replicates=[file name] - a two column tab-delimited table listing
                 pairs of samples that are replicates (at least 3 pairs)

matching=[float] required fraction of matching genotypes
                 (missing counts as match if there are other non-missing matching pairs),
                 default 1 (all must match)

missing=[float]  allowed fraction of missing genotypes, default 0.25

altPairs=2       minimal number of matching genotypes involving non-reference alleles.

hetPairs=0       minimal number of matching heterozygotes.

max.het=0.5      maximum fraction of heterozygotes among non-missing genotypes
                 (guards against lumped paralogous loci).

polyonly=[1|0]   extract only polymorphic sites. Default 0.

Example:
replicatesMatch.pl vcf=round2.vcf replicates=clonepairs.tab > vqsr.vcf
```

recalibrateSNPs.pl :

```
Non-parametric variant quality recalibration based on INFO fields
in the set of variants that are reproducibly genotyped among replicates
(output of replicatesMatch.pl)

This script is for de novo pipeline. Use recalibrateSNPs_gatk.pl for
reference-based pipeline.

Three parameters (INFO items) are assessed for each SNP:
- total coverage depth (DP)
- strand bias (SB)
- allele bias (AB)
- SNP position withn tag (TP)

For AB and SB, low values are deemed bad and high ones are good;
for DP and TP, both  low and high values are considered bad, the best values
being around the median.

The script computes the product of quantiles for different INFO fields, determines
the quantiles of the result within the 'true' set (JOINT quantiles),
and then computes the new quality scores in the main vcf file.

These scores are supposed to correspond to the probability (x100) that the SNPs comes
from the same distribution as the 'true' SNPs.

Output:
```

```
- Recalibrated VCF (printed to STDOUT) with QUAL field replaced by the new score
  minus 0.1 (for easy filtering)

- a table (printed to STDERR) showing the \"gain\" at each recalibrated quality
  score setting, which is excess variants removed when filtering at this quality score.
  For example, if 40% of all variants are removed at the quality score 15, the gain
  is 40 - 15 = 35% (i.e., in addition to removing 15% of the 'true' variants,
  additional 35% of the dataset, likely corresponding to wrong variants, is removed).
  The optimal filtering score is the one giving maximum gain. Try different combinations
  of the four possible filters to find the one maximizing the gain.

Arguments:

vcf=[file name]  : vcf file to be recalibrated

true=[file name] : vcf file that is subset of the above, with SNPs that are considered
                   true because they show matching and polymorphic genotypes in
replicates
                   (replicatesMatch.pl polyonly=1).

           -nodp : do not use DP
           -noab : do not use AB
           -nosb : do not use SB
           -notp : do not use TP

Example:
recalibrateSNPs.pl vcf=cdh_alltags.ul_Vhap_count10_ab10_sb10_clip0.vcf \
true=vqsr.vhap.vcf -nosb -notp >denovo.vhap.recal.vcf
```

---

filterStats.pl :

```
Calculates quantiles for three INFO fields in a VCF file:
- total coverage depth (DP)
- strand bias (SB)
- allele bias (AB)

For AB and SB, low values are bad and high values are good;
for DP, both very low and very high values are bad, the best values
should be around the median.

Arguments:

vcf=[file name]  vcf file to analyze

Output:
Tables of quantiles (printed to STDOUT).

Example:
filter_stats.pl vcf=alltags.ul_Variants_count10_ab10_sb10_clip0.vcf

NOTE: only use this to select your filtering criteria if you don't have
genotyping replicates.
```

---

repMatchStats.pl :

```
Summarizes genotypic match between replicates in a vcf file.

Arguments:

       vcf=[file name] : input vcf file
replicates=[file name] : a two column tab-delimited table listing
                         pairs of samples that are replicates

Output:
A table (printed to STDOUT) of the following form:

pair      gtyped  match      [ 00  01  11 ] HetMatch HomoHetMism HetNoCall HetsDiscRate
K210:K212  7328  7169(97.8%) [ 78% 17% 5% ]  1200         139        5         0.94
K212:K213  7369  7117(96.6%) [ 78% 17% 5% ]  1202         179        4         0.93

The first four columns are self-evident;
the last four columns show how good is the match between heterozygote calls
(HetNoCall being a heterozygote in one and missing data in another replicate).
The last column is the most important: it is the heterozygote discovery rate,
the fraction of all heterozygotes discovered in each replicate.

Example: repMatchStats.pl vcf=denovo.filt.recode.vcf replicates=clonepairs.tab
```

---

thinner.pl :

```
Leaves only one SNP per given interval, the one with the highest minor allele frequency.

Arguments:

vcf=[file name]    : vcf file name
interval=[integer] : interval length, default 40 (for 2bRAD tags)

Output:  thinned VCF, printed to STDOUT

Example: thinner.pl vcf=denovo.recal.vcf > thinDenov.vcf

NOTE: do not thin your variants if you want to calculate Tajima's D!
Also, for demographic inference using dadi, use --thin option in vcftools instead.
```

---

vcf2genepop.pl :

```
converts VCF to multiallelic GENEPOP, preserves chromosome and position info

Arguments:
vcf=[filename] : vcf file to convert
   pops=[list] : comma-separated perl patterns to determine population
                 affiliation of samples in the VCF file based on sample names

Output:
GENEPOP formatted genotypes, printed to STDOUT

Example:
vcf2genepop.pl vcf=filtered.vcf pops=O,K,M,S > filtered.gen
```

---

vhap2migrate.pl :

```
converts \"Vhap\" vcf file (produced by haplocall_denovo) to MIGRATE-n sequence input
(make sure you keep monomorphic loci when running haplocall_denovo: mono=keep)

Arguments:

           arg1 : input vhap vcf file
arg2 (optional) : comma-delimited list of files listing samples from specific
                  populations

Output:
MIGRATE-n formatted genotypes (printed to STDOUT)

Example:
vhap2migrate.pl denovo.vhap.filt.recode.vcf keppel.pop,orpheus.pop,maggi.pop
```

---

vhap2fasta.pl :

```
extracts reference 2bRAD tags from a vcf "Vhap" file
(produced by haplocall_denovo.pl) in fasta format

Arguments:
arg1: vhap.vcf file name
altalleles=[0|1] whether to output alternative alleles. Default 0.

prints to STDOUT
```

---

vcf2dadi.pl :

```
Converts VCF format to dadi data format
(requires bioperl)

Usage: perl vcf2dadi.pl <genome file> <vcf file> <list file>

Genome file is in fasta format;
List file gives population designations, like this:

BEGIN
sample1    population1
sample2    population1
sample3    population2
END
```