# Summary of the SNPpipeline process, report generation, and known issues
# October 10th, 2016

## 1 Purpose

This isn't a how-to document or user guide, rather it's a description of the inner workings of the SNPpipeline, with enough detail so that those of you who understand the requirements, the science and the file formats and third-party tools it depends on can determine whether it's doing what you want it to do, and whether changes or improvements need to be made. I've *underlined* all issues that I would like feedback from Jun-Jun, Isabel or Grace on. If you notice anything else, please let me know.

If you are primarily interested in the generated reports, feel free to skip down to Section 3 on page 5.

## 2 Processing The Raw Data

This section summarizes the process the data goes through to end up with .tab files stored in ./data/output. I've removed as much program logic from it as possible while still preserving the pipeline flow and the commands it makes to third-party tools.

### 2.1 Generate the reference file (formatted_output.fasta):

- Combine and format fasta file to consistent line lengths
- Build new reference, run:
  `bowtie2-build --threads <numCores> ./reference/formatted_output.fasta ./referenceTemp/formatted_output`
- Build .fai for reference, run:
  `samtools faidx "./reference/formatted_output.fasta"`

*Samtools parameters used in the above call:*
- ***faidx***: index/extract FASTA

### 2.2 Process data

#### 2.2.a If Single, for EACH FILE (or R1/R2 pair) in ./data:

**Alignment with bowtie, save to ./dataTemp/single/<filename>.sam:**
This step takes the raw data files found in ./data as input, and outputs a .sam file for each input file (or R1/R2 pair) in ./dataTemp/single/.

- If unpaired (0), run:
  ```
  bowtie2 -p <numCores> -x ./referenceTemp/formatted_output -U
  "./data/"<filename> --local --very-sensitive-local -S
  "./dataTemp/single/<filename>.sam"
  ```
- If paired (1), run:
  ```
  bowtie2 -p <numCores> -x ./referenceTemp/formatted_output -1 "./data/$(ls
  "./data" | grep <filename> | sed -n 1p)" -2 "./data/$(ls "./data" | grep
  <filename> | sed -n 2p)" --local --very-sensitive-local -S
  "./dataTemp/single/<filename>.sam"
  ```

(the above command passes as input the two files (R1 & R2) whose names contain the given <filename> string)

*Bowtie2 parameters used in the above calls:*

- -***p***: number of threads
- -***x***: Index filename prefix (minus trailing .X.bt2).
- -***1***: Files with #1 mates, paired with files in <m2>.
- -***2***: Files with #2 mates, paired with files in <m1>.
- -***U***: Files with unpaired reads.
- -***S***: File for SAM output (default: stdout)
- --***local***: local alignment; ends might be soft clipped (off)
- --***very-sensitive-local***: -D 20 -R 3 -N 0 -L 20 -i S,1,0.50

**Generate .bam:**

Takes as input the .sam files found in ./dataTemp/single/, and outputs .bam files in the same folder, then deletes the .sam files.

- Run:
  ```
  samtools view -Sb ./dataTemp/single/<filename>.sam >
  ./dataTemp/single/<filename>.bam
  ```
- Delete ./dataTemp/single/<filename>.sam

*Samtools parameters used in the above call:*

- ***view***: SAM<->BAM<->CRAM conversion
- -***S***: ignored (input format is auto-detected)
- -***b***: output BAM

**Sort .bam:**

Takes as input the .bam files found in ./dataTemp/single/, and outputs "_sorted.bam" files in the same folder, then deletes the original .bam files.

- Run:
  ```
  samtools sort -o ./dataTemp/single/<filename>_sorted.bam
  ./dataTemp/single/<filename>.bam
  ```
- Delete ./dataTemp/single/<filename>.bam

*Samtools parameters used in the above call:*

- ***sort***: sort alignment file
- -***o*** FILE: Write final output to FILE rather than standard output

**Index .bam:**
Takes as input the "_sorted.bam" files and indexes them.

- Run:
  ```
  samtools index ./dataTemp/single/<filename>_sorted.bam
  ```

*Samtools parameters used in the above call:*

- ***index***: index alignment

**Find SNPs with freebayes, save to ./outputTemp/single/<filename>.vcf:**
Takes as input the ./reference/formatted_output.fasta file and the ./dataTemp/single/<filename>_sorted.bam files, and outputs a .vcf file in ./outputTemp/single/ for each .bam file input.

- If params==paper(0), run:
  ```
  freebayes -f ./reference/formatted_output.fasta -p <singlep> --pvar 0.75
  --min-alternate-count 1 --theta 0.01  --use-best-n-alleles 4 --min-alternate-
  fraction 0.8 ./dataTemp/single/<filename>_sorted.bam 1>
  ./outputTemp/single/<filename>.vcf
  ```
- If params=default(1) run:
  ```
  freebayes -f ./reference/formatted_output.fasta -p <singlep>
  ./dataTemp/single/<filename>_sorted.bam 1> ./outputTemp/single/<filename>.vcf
  ```

*Freebayes input parameters, based on the paper (with the defaults also shown in parentheses):*

- fasta reference file
- -***p*** default ploidy = $singlep or $combinedp ($singlep or $combinedp, although freebayes default is 2)
- --***pvar*** (report sites if probability of polymorphism is greater than): 0.75 (default 0.0) (Note that post-filtering is generally recommended over the use of this parameter).
- --***min-alternate-count*** (min # of observations supporting an alternate allele within a single individual in order to evaluate this position): 1 (default 0.2)
- --***theta*** (expected mutation rate or pairwise nucleotide diversity): 0.01 (default 0.001)
- --***use-best-n-alleles***: 4 (Set to 0 to use all; default: all)
- --***min-alternate-fraction***: 0.8 (default 0.2)
- .bam input file

**Remove indels:**
Takes as input the .vcf files in ./outputTemp/single/, and outputs a ".recode.vcf" file for each input in the same folder.

- Run:
  ```
  vcftools --vcf ./outputTemp/single/<filename>.vcf --remove-indels --recode
  --recode-INFO-all --out ./outputTemp/single/<filename>
  ```

*Vcftools parameters used in the above call:*

- --***vcf*** <input_filename>: the VCF file to be processed.
- --***remove-indels***: exclude sites that contain an indel. "Indel" means any variant that alters the length of the REF allele.
- --***recode***: generate a new file in VCF from the input VCF or BCF file after applying the filtering options specified by the user. The output file has the suffix ".recode.vcf". By default, the INFO

fields are removed from the output file, as the INFO values may be invalidated by the recoding (e.g. the total depth may need to be recalculated if individuals are removed). This behavior may be overriden.
- --***recode-INFO-all***: can be used with the above recode option to define an INFO key name to keep all INFO values in the original file.
- --***out*** <output_prefix>: the output filename prefix for all files generated by vcftools.

**Filter using cutoffs:**
Takes as input the .recode.vcf files in ./outputTemp/single/, and outputs a "_cutoff" file for each input in the same folder.

- Run:
```
cat ./outputTemp/single/<filename>.recode.vcf |
./tools/vcftools/src/perl/vcf-annotate --filter Qual=5 >
./outputTemp/single/<filename>_cutoff
```

*(The vcf-annotate tool adds or removes filters and custom annotations to VCF files).*

**Tab conversion:**
Takes as input the "_cutoff" files in ./outputTemp/single/, and outputs the final .tab file in ./output/single/.

- Run:
```
vcf-to-tab < ./outputTemp/single/<filename>_cutoff >
./output/single/<filename>.tab
```

*(The vcf-to-tab tool converts the VCF file into a tab-delimited text file listing the actual variants instead of ALT indexes).*

### 2.2.b If Pooled:

**Alignment with bowtie, save to ./dataTemp/pooled/<filename>.sam:**

- If unpaired (0), run:
```
bowtie2 -p <numCores> -x ./referenceTemp/formatted_output -U <list of files
in ./data> --local --very-sensitive-local -S
"./dataTemp/pooled/<filename>.sam"
```
- If paired (1), run:
```
bowtie2 -p <numCores> -x ./referenceTemp/formatted_output -1 <list of R1
files in ./data> -2 <list of R2 files in ./data> --local --very-sensitive-
local -S "./dataTemp/pooled/<filename>.sam"
```

*Bowtie2 parameters used in the above calls:*
(see the parameters listed under section 2.2.a)

The following operations are the same as for single data (above), except instead of being run once per file (or R1/R2 pair) in ./dataTemp/single, these are run only once for the pooled data output file in ./dataTemp/pooled, and the folder "single" is changed to "pooled" in any folder paths in the commands listed for single.

**Generate .bam**
**Sort .bam**
**Index .bam**
**Find SNPs with freebayes, save to ./outputTemp/pooled/<filename>.vcf**
**Remove indels**
**Filter using cutoffs**
**Tab conversion**

# 3 Report Generation

This is an overview of how the various reports are generated from the processed data.

## 3.1 Generate report.csv

- Read the .tab file in ./output/**pooled**/ (if it exists), and store it's data in a report column called "COMBINED"
- Read all the .tab files in ./output/**single**/ and add them to the report, adding one column per .tab file (matching rows by "CHROM", "POS", "REF")
  - If the report has a "COMBINED" column, reject any rows in the single .tab files that don't match up with an already existing row in the report.
  - If the report has no "COMBINED" column, accept all rows from the single .tab files, even if they don't match with an already existing row in the report.
- Write this report to ./reports/report.csv

## 3.2 Generate filled_report.csv

- Split the contents of report into separate files of 1000 rows each, then process each file row by row:
  - excluding the "COMBINED" column from calculations (if it exists), for each column in the row, if its data is NA:
    - run:
      ```
      samtools tview ./dataTemp/single/<filename>_sorted.bam
      ./reference/formatted_output.fasta -d T -p \"<the row's CHROM
      value>:<the row's POS value>"
      ```
    - If the first character in line 1 of the samtools ouput is the same as the current row's "REF" value, and the first character in line 2 of the samtools output is either "." or ",", replace the NA in the current column with the "REF" value + "/"
      *(TODO: Jun-Jun requested replacing "REF"/ with "REF"/"REF". I haven't yet made this change because it impacts the code to generate probability.csv (see notes in section 3.8). It will be easy to change, but I would like Jun-Jun's feedback on section 3.8 first).*
- After processing, recombine all the files into one large report, and write it to ./reports/filled_report.csv

### 3.3 Generate edited_report.csv (editing for cutoffs)

- Using the data in filled_report as input, if there are less than 20 data columns, skip this entire step and don't write edited_report.csv
- If there are at least 20 data columns (21 if the "COMBINED" column exists):
  - Excluding the "COMBINED" column from calculations (if it exists):
    - *(BUG: There's an off-by-one error here which is easy to fix but would have affected any previously generated reports: it always excludes the first single data column (after the COMINED colum if it exists) from the following calculations, although the column still remains in the report. So, for example, if there are 54 data columns, it will perform the following checks using columns 2-54, but not column 1):*
    - Remove all rows that have NA for more than half their values
    - Remove all rows that have only two types of data, and NA is one of them
    - Remove all rows that have only one type of data, and it isn't NA
  - Write the results to ./reports/edited_report.csv

### 3.4 Generate percentage_snps.csv (finding snp percentage per site)

- Excluding the "COMBINED" column from calculations (if it exists), for each row in the report (either filled_report if less than 20 data columns, or edited_report if more than 20), calculate and store in a new report the totals of A, C, T, G, NA, the max (excluding NA), second_max (excluding NA), sum (excluding NA), and MAF (second_max / (second_max + max).
- Write this to percentage_snps.csv

### 3.5 Generate MAF_cutoff_report.csv

- Remove all rows from report (either filled_report if less than 20 data columns, or edited_report if more than 20) whose MAF (see percentage_snps.csv) is less than the MAF_CUTOFF, and sort the report by snpp's MAF field, highest to lowest.
- Write this to MAF_cutoff_report.csv

### 3.6 Generate mutation_percentage.csv

- Use the MAF_cutoff_report's "CHROM" column and the reference fasta file to calculate and write mutation_percentage.csv, sorted by percentage SNP. Columns in mutation_percentage.csv are defined as:
  - "" = sequence name
  - Role = sequence annotation (name + trailing info)
  - Snp = number of rows in MAF_cutoff_report containing the current fasta name in their "CHROM" cell
  - Length = number of character's in the sequence
  - Percentage SNP = snp / length * 100
- Write this to mutation_percentage.csv

## 3.7 Generate MAF_cutoff_report_chi.csv (replacing alleles with characters for chi square test)

*NOTE: This procedure was non-operational when I tested it. I've fixed some bugs and made sure it runs they way I think it should, but please make sure it's doing it properly.*

- Using MAF_cutoff_report.csv as a starting point, excluding the "COMBINED" column from operations (if it exists), for each row:
  - If there's only one value in the row, replace all occurences of it with "H"
  - If the two most frequent values make up more than 90% of the row, replace all occurences of the most frequent with H and all occurrences of the 2nd most frequent with A, and replace everything else with NA.
    *(NOTE: What if they both have the same number of occurences? Do we just leave it to the sorting algorithm to determine which gets H and which gets A?)*
  - Else if the two most frequent values make up 90% or less, remove the row from report
- Write report to MAF_cutoff_report_chi.csv

*(NOTE: the way this is written, the two most frequent values must make up MORE THAN 90% of the row, and therefore wouldn't qualify if they make up 90%. Should I change this to be inclusive of 90%?)*
*(NOTE: What if either of the two most frequent values required for a majority of > 90% is NA? Do we delete the row? I have assumed so and programmed it accordingly).*

## 3.8 Generate probability.csv

*NOTE: This procedure was non-operational when I tested it. I've fixed some bugs and made sure it runs the way I think it should, but please make sure that it's doing it properly.*

- Using MAF_cutoff_report.csv as a starting point, for each column of the report:
  - If it's the "COMBINED" column, set <filename> to the existing "cutoff" file in outputTemp/pooled/
  - Otherwise set <filename> to ./outputTemp/single/<current col name (minus the last 4 characters)>_cutoff"
  - Import the "_cutoff" file (it's in vcf format), and get the row ranges recorded in the file. We will use their names and indices to look up genotype likelihoods.
  - For each cell in the current report column, if the current cell is not NA:
    - *(POSSIBLE BUG: The code summarized below has special cases for when the data value is just 2 characters (eg. "A/"). This would break once I do the change Jun-Jun requested for filled_report.csv (see section 3.2). I'd like to talk with Jun-Jun about how to resolve this without adversely affecting other data samples.*
    - If the number of characters in the current cell is 2 and the first character is NOT equal to the current row's "REF" value:
      - Set index to the list of the indices of those row range names which contain both this row's "CHROM" value and ":<this row's POS value>_".
        *(We only care about the first one in the list if there are more than one).*
      - If index is not NA, set the current cell to:
        `10^(the 2nd of 3 genotype likelihoods in the vcf "_cutoff" file's $GENO$GL[index] record)` *(NOTE: Is this correct?)*
        *($GENO$GL is a list of genotype likelihoods, indexed by row ranges. Each row range has three likilihood numbers associated with it. For example, if the REF*

*allele associated with the row range is A and the ALT allele associated with the row range is T, the three stored likelihood values for the row range would be for A/A, A/T, T/T, respectively. See footnote* for official documentation on GL and GT data in VCF files.*

- Else if index is NA, set the current cell to NA

- If the number of characters in the current cell is 2 and the first character IS equal to the current row's "REF" value, set the current cell to 1
- *(NOTE: If we are going to replace all two character instances with three characters (eg. "A/" with "A/A"), we will need to implement some custom code here).*
- Otherwise, if the number of characters in the cell is <= 3:
  - Set index to the list of the indices of those row range names which contain both this row's "CHROM" value and ":<this row's POS value>_".
  - If index is not NA
    - If the first character in the cell is not equal to the third character (eg. "A/T")
      - Set the current cell to:
        `10^(the 2nd of 3 genotype likelihoods in the vcf "_cutoff" file's $GENO$GL[index] record)` *(NOTE: Is this correct?)*
        *(see note above for explanation)*
    - Else if the first and third character do equal each other
      - Set the current cell to:
        `10^(the 3rd of 3 genotype likelihoods in the vcf "_cutoff" file's $GENO$GL[index] record)` *(NOTE: Is this correct?)*
        *(see not above for explanation)*
    - *NOTE: Should we have a case for when the first character matches the REF but the third doesn't? Or when the third character matches the REF but the first doesn't? And a case for when neither match the REF? Would comparing to the REF here make any difference?*
  - If index is NA, set the current cell to NA
- Otherwise, set the current cell to NA

- Write report to probability.csv

---

* From: http://samtools.github.io/hts-specs/VCFv4.1.pdf, section 1.4.2, page 5:
GL : genotype likelihoods comprised of comma separated floating point log 10 -scaled likelihoods for all possible genotypes given the set of alleles defined in the REF and ALT fields. In presence of the GT field the same ploidy is expected and the canonical order is used; without GT field, diploidy is assumed. If A is the allele in REF and B,C,... are the alleles as ordered in ALT, the ordering of genotypes for the likelihoods is given by: F(j/k) = (k*(k+1)/2)+j. In other words, for biallelic sites the ordering is: AA,AB,BB; for triallelic sites the ordering is: AA,AB,BB,AC,BC,CC, etc. For example: GT:GL 0/1:-323.03,-99.29,-802.53 (Floats)

GT : genotype, encoded as allele values separated by either of / or |. The allele values are 0 for the reference allele (what is in the REF field), 1 for the first allele listed in ALT, 2 for the second allele list in ALT and so on. For diploid calls examples could be 0/1, 1 | 0, or 1/2, etc. For haploid calls, e.g. on Y, male non-pseudoautosomal X, or mitochondrion, only one allele value should be given; a triploid call might look like 0/0/1. If a call cannot be made for a sample at a given locus, '.' should be specified for each missing allele in the GT field (for example './.' for a diploid genotype and '.' for haploid genotype). The meanings of the separators are as follows (see the PS field below for more details on incorporating phasing information into the genotypes): / : genotype unphased , | : genotype phased