# Calling Variants using Bowtie2 + Samtools

## Alignment

In order to call variants, the first step is to align the short reads to the reference genome. In this case, we will be using Bowtie2 as our aligner. For a description of Bowtie2, click [here](http://bowtie-bio.sourceforge.net/bowtie2/index.shtml).

I am using Bowtie2 on the Stanford scg cluster, so I first login to a large memory node and load bowtie using:

$ ssh <user\_id>@greenie

$ module load bowtie/2.2.1

The basic command of bowtie2 is:

$ bowtie2 -x <bowtie2\_index> -U <input\_fastq\_files> -S <output\_sam\_files>

The –x, -U and –S are called flags. There meanings are:

* -x indicates this command line input is the bowtie2 inde[x].
* -U Comma-separated list of files containing [U]npaired reads to be aligned, e.g. lane1.fq,lane2.fq,lane3.fq,lane4.fq. Reads may be a mix of different lengths. If - is specified, bowtie2 gets the reads from the "standard in" or "stdin" filehandle.
* -S output [S]am files

To make our lives easier, let’s create a few atlases first:

fastq\_dir=<directory\_to\_fastq>

sam\_dir=<directory\_to\_sam>

index\_dir=<directory\_to\_bowtie\_index>

log\_dir=<directory\_to\_logs>

Now let us run bowtie2:

bowtie2 -x $index\_dir/<index\_basename> -U $fastq\_dir/<fastq\_file> -S $sam\_dir/<sam\_file> 2><log\_file>

The last part of the command “2> <log\_file>” redirects stderr to a log file for later reference.

A moment later, bowtie2 spits out:

137294 reads; of these:

137294 (100.00%) were unpaired; of these:

2189 (1.59%) aligned 0 times

128799 (93.81%) aligned exactly 1 time

6306 (4.59%) aligned >1 times

98.41% overall alignment rate

That’s pretty much all we need to do about alignment. Simple, right? A few additional resources.

* Bowtie2 manual ([link](http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml))
* Run bowtie2 on all files in a directory: **run\_bowtie2.sh**

## Compress and Sort

After alignment has finished, the resultant sam files need to sorted and compressed using:

samtools view -bS <sam\_file> | samtools sort - <bam\_file\_prefix>

The dash (-) tells samtools to take stdin as input.

For instance:

samtools view -bS 020311-1\_S40\_L001\_R1\_001.sam | samtools sort - 020311-1\_S40\_L001\_R1\_001.sorted

Ignore the following warning, it is a known bug.

[bam\_header\_read] EOF marker is absent. The input is probably truncated.

To compress and sort all files in a directory, use **compress\_and\_sort.sh.**

## Call Genotype Likelihoods using Samtools

In this pipeline we will use the Mpileup command in Samtools to call genotype likelihoods or to call variants.

Type the following to call genotype likelihood:

samtools mpileup -g -f <genome> <sorted\_bam\_file> > <bcf\_file>

* -g Compute genotype likelihoods and output them in the binary call format (BCF).
* -f indicate <genome> is indexed by samtools faidx (a module in samtools).

For example:

hg19=/srv/gs1/projects/montgomery/shared/genome/hg19/hg19.fa

samtools mpileup -g -f $hg19 020311-1\_S40\_L001\_R1\_001.sorted.bam > 020311-1\_S40\_L001\_R1\_001.bcf

To inspect bcf files, use bcftools, which comes with samtools.

bcftools view 020311-1\_S40\_L001\_R1\_001.bcf

Additional information:

* bcf file format specification: [link](http://www.1000genomes.org/wiki/analysis/variant-call-format/bcf-binary-vcf-version-2)
* sam file format specification: [pdf](http://samtools.github.io/hts-specs/SAMv1.pdf)
* call genotype over all files in a directory: **call\_genotype\_likelihood.sh**

## Call Variants With Samtools

### One diploid individual:

samtools mpileup -uf ref.fa aln1.bam

bcftools view -bvcg <bcf\_file> > <variants\_bcf\_file>

bcftools view <variants\_bcf\_file> | vcfutils.pl varFilter –D 100 > <filtered\_ variants\_bcf\_file>

samtools mpileup flags

* -u compute genotype likelihood and output in [u]ncompressed format (vcf).
* -f samtools [f]aidx indexed file

bcftools flags

* -b output in bcf format
* -v output variant sites only
* -c call variants using Bayesian inference
* -g call per-sample genotype at variant site

vcfutils flags

* -D 100 controls the maximum read depth, which should be adjusted to about twice the average read depth

### Multiple diploid individuals

TBD

### More information:

* Powerpoint from EBI explaining this pipeline: [pdf](https://www.ebi.ac.uk/training/sites/ebi.ac.uk.training/files/materials/2014/140217_AgriOmics/dan_bolser_snp_calling.pdf)

# Call Variants using GATK

## Preprocessing

### Alignment

Align with BWA using:

bwa mem -M -R ’<read group info>’ reference.fa raw\_reads.fq > aligned\_reads.sam

The ‘<read group info>’ can be:

@RG\tID:group1\tSM:sample1\tPL:illumina\tLB:lib1\tPU:unit1

Batch implementation: **run\_bwa\_single\_end.sh**

### Dedupping

Use Picard. This step is only necessary for whole-genome sequencing; you don’t need to dedup for mmPCR-seq data.

### Indel Realignment

TBD

### Base Recalibration

TBD

## Variant Discovery

TBD

# Troubleshoot

* SAM file doesn't have any read groups defined in the header. The GATK no longer supports SAM files without read groups

Many GATK algorithms needs information that certain reads are sequenced together as these algorithms try to compensate the variation from one sequencing to the next. A read group, or RG, provides this information. Reads in the same read group are assumed to be sequenced on one lane. ([ref](https://www.biostars.org/p/43897/))

Implementation: **add\_read\_groups.sh** to add read group.

Click [here](https://www.broadinstitute.org/gatk/guide/article?id=2909) to know more about how to fix a badly formatted BAM.

TODO: add batch functionality to **add\_read\_groups.sh**

* Input files reads and reference have incompatible contigs: Relative ordering of overlapping contigs differs, which is unsafe.

Solution: reorder the sam/bam file with picard/ReorderSam. ([ref](https://www.broadinstitute.org/gatk/guide/article?id=58), [help](http://broadinstitute.github.io/picard/command-line-overview.html#ReorderSam))

Implementation: **reorder\_sam.sh**

TODO: add batch functionality to reoder\_sam.sh