> **Note**
> You are not reading the most recent version of this documentation. 2019 (/en/2019/GATK_pipeline.html) is the latest version available.

Variant calling pipeline for a mammalian genome

We will run a variant calling pipeline using Genome Analysis Toolkit (GATK) (https://software.broadinstitute.org/gatk/) using a subset sample of dog WGS as a representative to large mammalian genomes.

Getting started

Start up an m1.medium instance running Ubuntu 16.04 on Jetstream. (jetstream/boot.html)

log in, and then make & change into a working directory:

```
mkdir ~/GATK_tutorial && cd ~/GATK_tutorial
```

Download trimmed Fastq files

```
wget https://de.cyverse.org/dl/d/3CE425D7-ECDE-46B8-AB7F-FAF07048AD42/sample
tar xvzf samples.tar.gz
rm samples.tar.gz
```

Quick notes about read trimming for variant calling:

1. Trimming is data loss so be careful.
2. Sequence trimming is complementary to variant filtration
3. Sources of errors: a) The call is suspicious ==> low quality score (variant filtration is better than quality trimming) b) Technical problems (e.g. sequencing chemistry or physics) ==> systematic errors (can be removed by careful kmer based trimming but GATK recalibration is an alternative)
4. Very mild quality trimming: SLIDINGWINDOW:4:2 ==> this means that the Base call accuracy is ~ 40% (https://en.wikipedia.org/wiki/Phred_quality_score)

Mapping

1. Install bwa (http://bio-bwa.sourceforge.net/bwa.shtml):

```
cd
curl -L https://sourceforge.net/projects/bio-bwa/files/bwa-0.7.15.tar.bz2/
tar xjvf bwa-0.7.15.tar.bz2
cd bwa-0.7.15
make

sudo cp bwa /usr/local/bin

echo 'export PATH=$PATH:/usr/local/bin' >> ~/.bashrc
source ~/.bashrc
```

2. change into a working directory:

```
cd ~/GATK_tutorial
```

3. download and prepare the reference for mapping

```
wget https://de.cyverse.org/dl/d/A9330898-FC54-42A5-B205-B1B2DC0E91AE/dog_
gunzip dog_chr5.fa.gz
bwa index -a bwtsw dog_chr5.fa
```

4. Add Read group information (https://angus.readthedocs.io/en/2017/Read_group_info.html) and do mapping

Read group information is typically added during this step, but can also be added or modified after mapping using Picard AddOrReplaceReadGroups.

```
for R1 in *_R1_001.pe.fq.gz;do
  SM=$(echo $R1 | cut -d"_" -f1)                                          ##
  LB=$(echo $R1 | cut -d"_" -f1,2)                                        ##
  PL="Illumina"                                                           ##
  RGID=$(zcat $R1 | head -n1 | sed 's/:/_/g' |cut -d "_" -f1,2,3,4)       ##
  PU=$RGID.$LB                                                            ##
  echo -e "@RG\tID:$RGID\tSM:$SM\tPL:$PL\tLB:$LB\tPU:$PU"

  R2=$(echo $R1 | sed 's/_R1_/_R2_/')
  echo $R1 $R2
  bwa mem -t 4 -M -R "@RG\tID:$RGID\tSM:$SM\tPL:$PL\tLB:$LB\tPU:$PU" dog_chr
done
```

Generate sorted BAM files

1. install samtools (http://www.htslib.org/doc/samtools-0.1.19.html)

```
sudo apt-get -y install samtools
```

2. generate & sort BAM file

```
for samfile in *.sam;do
  sample=${samfile%.sam}
  samtools view -bS -o $sample.bam $samfile
  samtools sort $sample.bam $sample.sorted
done
rm *.sam *_L00[0-9].bam
```

Merge replicates (one library running on two lanes):

1. Install Java

```
sudo mkdir -p /usr/local/java
cd /usr/local/java
sudo wget -c --header "Cookie: oraclelicense=accept-securebackup-cookie" h
sudo tar xvzf jdk-8u131-linux-x64.tar.gz
echo 'export PATH=$PATH:/usr/local/java/jdk1.8.0_131/jre/bin' >> ~/.bashrc
source ~/.bashrc
```

2. Download Picard tools

```
cd ~/GATK_tutorial
wget https://github.com/broadinstitute/picard/releases/download/2.9.4/pica
```

3. merge the replicates

```
java  -Xmx10g -jar picard.jar MergeSamFiles I=BD143_TGACCA_L005.sorted.bar
```

4. check for the changes in the header

```
samtools view -H BD143_TGACCA_L005.sorted.bam
samtools view -H BD143_TGACCA_L006.sorted.bam
samtools view -H BD143_TGACCA_merged.sorted.bam
```

5. remove the individual replicates

```
rm BD143_TGACCA_L00*.sorted.bam
```

Mark duplicates

Duplicates:

- PCR duplicates (originating from a single fragment of DNA) or
- optical duplicates (result from a single amplification cluster, incorrectly detected as multiple clusters by the optical sensor of the sequencing instrument)

Duplicate marking should NOT be applied to amplicon sequencing data or other data types where reads start and stop at the same positions by design.

```
for sample in *.sorted.bam;do
  name=${sample%.sorted.bam}
  java  -Xmx10g -jar picard.jar MarkDuplicates INPUT=$sample OUTPUT=$name.dedu
done
```

Prepare for the Genome Analysis Toolkit (GATK) analysis

1. download Genome Analysis Toolkit (GATK)

```
wget https://de.cyverse.org/dl/d/6177B1E0-718A-4F95-A83B-C3B88E23C093/Genc
tar xjf GenomeAnalysisTK-3.7-0.tar.bz2
```

2. Prepare GATK dictionary and index for the reference genome

```
java -Xmx10g -jar picard.jar CreateSequenceDictionary R=dog_chr5.fa O=dog
samtools faidx dog_chr5.fa
```

Recalibrate Bases

1. Download known polymorphic sites

```
wget 'ftp://ftp.ensembl.org/pub/release-89/variation/vcf/canis_familiaris/
```

2. Select variants on chr5 and correct chr name

```
gunzip canis_familiaris.vcf.gz
grep "^#" canis_familiaris.vcf > canis_fam_chr5.vcf
grep "^5" canis_familiaris.vcf | sed 's/^5/chr5/' >> canis_fam_chr5.vcf
```

This algorithm treats every reference mismatch as an indication of error, so it is critical that a “comprehensive” database of known polymorphic sites is given to the tool in order to be masked and not counted as errors. What we can do with semi-model organisms?

Note the differences between genome annotation databases. Not only chromosome names but more importantly the coordinate system (interesting post) (https://www.biostars.org/p/84686/)

1. download R (only to generate figures to observe the changes, but we will need it later as well)

```
sudo apt-get update && sudo apt-get -y install gdebi-core r-base
```

After that finishes, download and install RStudio:

```
wget https://download2.rstudio.org/rstudio-server-1.0.143-amd64.deb
sudo gdebi -n rstudio-server-1.0.143-amd64.deb
```

Install some packages

```
sudo Rscript -e "install.packages('ggplot2', contriburl=contrib.url('http:
sudo Rscript -e "install.packages('gplots', contriburl=contrib.url('http:/
sudo Rscript -e "install.packages('reshape', contriburl=contrib.url('http:
sudo Rscript -e "install.packages('gsalib', contriburl=contrib.url('http:/
sudo Rscript -e "install.packages('Biobase', contriburl=contrib.url('http:
```

Add a password to your instance


```
sudo passwd tx160085
```

You will be prompted to enter a new password. Make a password you can remember:

```
Enter new UNIX password:
Retype new UNIX password:
```

Get the address of your own RStudio web server

```
echo My RStudio Web server is running at: http://$(hostname):8787/
```

Copy the link to a new tab of your browser and hit `enter`.

```
Username: `tx160085`
Password: `The one you just created`
```

Keep this tab open and will come back to it in a min. ** Now go to your web shell **

2. run recalibration

```
for sample in *.dedup.bam;do
  name=${sample%.dedup.bam}
  samtools index $sample
  java -Xmx10g -jar GenomeAnalysisTK.jar -T BaseRecalibrator -R dog_chr5.f
  java -Xmx10g -jar GenomeAnalysisTK.jar -T BaseRecalibrator -R dog_chr5.f
  java -Xmx10g -jar GenomeAnalysisTK.jar -T PrintReads -R dog_chr5.fa -I $
  java -Xmx10g -jar GenomeAnalysisTK.jar -T AnalyzeCovariates -R dog_chr5.
done
```

More details (https://software.broadinstitute.org/gatk/documentation/article?id=44) about the tool and interpretation of the output figures

Variant calling

1. per-sample calling

```
for sample in *.recal.bam;do
  name=${sample%.recal.bam}
  java -Xmx10g -jar GenomeAnalysisTK.jar -T HaplotypeCaller -R dog_chr5.fa
done
```

2. Joint Genotyping

```
 java -Xmx10g -jar GenomeAnalysisTK.jar -T GenotypeGVCFs -R dog_chr5.fa --
 --variant BD143_TGACCA_merged.g.vcf \
 --variant BD174_CAGATC_L005.g.vcf \
 --variant BD225_TAGCTT_L007.g.vcf \
 -o raw_variants.vcf
```

Filter Variants

The best way to filter the raw variant callset is to use variant quality score recalibration (VQSR). However this requires high-quality sets of known variants for training, which for many organisms are not yet available. It also requires a lot of data, so it can be difficult or even impossible to use on small datasets that involve only one or a few samples, on targeted sequencing data, or on RNAseq.

Hard filtering flat thresholds for specific annotations: GATK uses VariantFiltration for hard filtering. The documentation page (https://software.broadinstitute.org/gatk/documentation/tooldocs/current/org_broadinstitute_gatk_t provides links to all possible annotation modules. You can get some recommendations here (https://software.broadinstitute.org/gatk/documentation/article.php?id=3225).

1. Split variants into SNPs and INDELs

```
java -Xmx10g -jar GenomeAnalysisTK.jar -T SelectVariants -R dog_chr5.fa -V
java -Xmx10g -jar GenomeAnalysisTK.jar -T SelectVariants -R dog_chr5.fa -V
```

2. Explore the distribution of different annotations

```
wget https://raw.githubusercontent.com/drtamermansour/angus/2017/densityCu
for var in "SNP" "INDEL";do
 for ann in "QD" "MQRankSum" "FS" "SOR" "ReadPosRankSum";do
  annFile=$var.$ann; echo $annFile;
  awk -v k="$ann=" '!/#/{n=split($8,a,";"); for(i=1;i<=n;i++) if(a[i]~"^"k
  grep -v "^\." $annFile > known.$annFile
  grep "^\." $annFile > novel.$annFile
  Rscript densityCurves.R "$annFile"
  rm $annFile known.$annFile novel.$annFile
done; done
```

3. Apply the filters

```
java -Xmx10g -jar GenomeAnalysisTK.jar -T VariantFiltration -R dog_chr5.fa
--filterExpression "QD < 2.0 || FS > 60.0 || MQ < 40.0" \
--filterName "snp_filter" \
-o filtered_SNP.vcf

java -Xmx10g -jar GenomeAnalysisTK.jar -T VariantFiltration -R dog_chr5.fa
--filterExpression "QD < 2.0 || FS > 200.0" \
--filterName "indel_filter" \
-o filtered_INDEL.vcf
```

Exploratory RNAseq data analysis using RMarkdown (rmarkdown_rnaseq.html)

Genome Wide Association analysis (GWAS) (GWAS.html)