# Whole Exome Sequencing for SNP and Indel Discovery source code

## Tools and Libraries Used

- **FastQC**
  Tool used to perform quality control checks on raw sequence data.

  ```
  sudo apt install fastqc
  ```

- **MultiQC**
  Aggregates results from FastQC into a single HTML report.

  ```
  pip install multiqc
  multiqc .
  ```

- **GATK (Genome Analysis Toolkit)**
  Toolkit for variant discovery and genotyping developed by the Broad Institute.

  ```
  # Download GATK from https://github.com/broadinstitute/gatk/releases
  # Make it executable:
  chmod +x gatk
  ./gatk --help
  ```

- **Samtools**
  Utilities for manipulating alignments in the SAM/BAM format.

  ```
  sudo apt install samtools
  ```

- **BWA (Burrows-Wheeler Aligner)**
  For aligning sequence reads to a large reference genome.

  ```
  sudo apt install bwa
  bwa index hg38.fa
  bwa mem hg38.fa reads_1.fastq.gz reads_2.fastq.gz > aligned.sam
  ```

- **Picard**
  Toolset for manipulating high-throughput sequencing data.

  ```
  # Download from https://broadinstitute.github.io/picard/
  java -jar picard.jar --help
  ```

- **Funcotator (Functional Annotation)**
  GATK tool for annotating variants using known biological databases.

  ```
  # Part of GATK bundle. Requires downloading data sources:
  wget https://...funcotator_dataSources.v1.7.20200521g.tar.gz
  tar -xvzf funcotator_dataSources.v1.7.20200521g.tar.gz
  ```

- **wget / curl**
  Used to download required reference and known site files.

  ```
  sudo apt install wget
  ```

# 1 Project Folder Structure

```
project_root/
 supporting_files/
    hg38/
        hg38.fa
        hg38.fai
        hg38.dict
        Homo_sapiens_assembly38.dbsnp138.vcf
        Homo_sapiens_assembly38.dbsnp138.vcf.idx

 reads/
    SRR062634_1.filt.fastq.gz
    SRR062634_2.filt.fastq.gz

 aligned_reads/
    SRR062634_sorted_dedup_bqsr_reads.bam

 data/
    recal_data.table

 results/
    raw_variants.vcf
    raw_snps.vcf
    raw_indels.vcf

    filtered_snps.vcf
    filtered_indels.vcf

    analysis-ready-snps.vcf
    analysis-ready-indels.vcf

    analysis-ready-snps-filteredGT.vcf
    analysis-ready-indels-filteredGT.vcf

    analysis-ready-snps-filteredGT-functotated.vcf
    analysis-ready-indels-filteredGT-functotated.vcf

 scripts/
    variant_calling.sh
    filter_and_annotation.sh
```

# 2 DataSets

```
#!/bin/bash
# Script: Variant_calling.sh
# Purpose: Download paired-end FASTQ files for sample SRR794247 (HG03012)
# Project: Whole Exome Sequencing (WES) Pipeline

# Define output directory
OUTPUT_DIR="/Users/aravindhsudhakar/Desktop/BioInformatics/reads"

# Download paired-end FASTQ files
wget -P $OUTPUT_DIR
↪ ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR794/SRR794247/SRR794247_1.fastq.gz
wget -P $OUTPUT_DIR
↪ ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR794/SRR794247/SRR794247_2.fastq.gz
```

## 3 Reference Genome Preparation

```
#!/bin/bash
# ------------------------------------------------------------------------
# Purpose: Prepare reference genome and supporting files for variant calling
# ------------------------------------------------------------------------

# Define base directory
REF_DIR=~/Desktop/demo/supporting_files/hg38/

# Download reference genome (hg38) and unzip
wget -P $REF_DIR https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/hg38.fa.gz
gunzip $REF_DIR/hg38.fa.gz

# Index the reference genome (.fai index) using samtools
samtools faidx $REF_DIR/hg38.fa

# Create sequence dictionary (.dict) using GATK
gatk CreateSequenceDictionary \
  -R $REF_DIR/hg38.fa \
  -O $REF_DIR/hg38.dict

# Download known sites VCF files for Base Quality Score Recalibration (BQSR)
wget -P $REF_DIR \
  https://storage.googleapis.com/genomics-public-data/resources/broad/hg38/v0/\
Homo_sapiens_assembly38.dbsnp138.vcf

wget -P $REF_DIR \
  https://storage.googleapis.com/genomics-public-data/resources/broad/hg38/v0/\
Homo_sapiens_assembly38.dbsnp138.vcf.idxs
```

## 4 Variant Calling Setup: Directory Paths

```
#!/bin/bash
# ------------------------------------------------------------------------
# Script: variant_calling_setup.sh
# Purpose: Define directory paths for variant calling pipeline (common paths)
# ------------------------------------------------------------------------

# Path to reference genome (FASTA)
ref="./supporting_files/hg38/hg38.fa"

# Path to known variant sites for BQSR
known_sites="./supporting_files/hg38/Homo_sapiens_assembly38.dbsnp138.vcf"

# Directory containing aligned BAM files
aligned_reads="./aligned_reads"

# Directory containing input FASTQ read files
reads="./reads"

# Output directory for variant calling results
results="./results"

# Data directory for intermediate or other input files
data="./data"
```

# 5 Variant Calling Setup

```
#!/bin/bash
# ------------------------------------------------------------------------
# Purpose: Define directory paths and input files for variant calling steps
# ------------------------------------------------------------------------

# Define base project directory (modify as needed)
BASE_DIR="./project_root"

# Reference genome FASTA file
ref="${BASE_DIR}/supporting_files/hg38/hg38.fa"

# Known variant sites for Base Quality Score Recalibration (BQSR)
known_sites="${BASE_DIR}/supporting_files/hg38/Homo_sapiens_assembly38.dbsnp138.vcf"

# Directory containing aligned BAM files
aligned_reads="${BASE_DIR}/VC/aligned_reads"

# Directory containing raw paired-end FASTQ files
reads="${BASE_DIR}/VC/reads"

# Directory for storing result files
results="${BASE_DIR}/VC/results"

# Directory for storing intermediate data (e.g., metrics, temp files)
data="${BASE_DIR}/VC/data"

```

## Step 1: Quality Control (FastQC)

```
# ------------------------------------------------------------
# STEP 1: Quality Control using FastQC
# ------------------------------------------------------------

echo "STEP 1: Quality Control - Running FastQC on raw reads"

# Run FastQC on forward and reverse reads
fastqc ${reads}/SRR794247_1.filt.fastq.gz -o ${reads}/
fastqc ${reads}/SRR794247_2.filt.fastq.gz -o ${reads}/
```

## Read Quality and Trimming

Initial quality assessment with FastQC indicated that the sequencing reads were of high quality, with no significant adapter contamination or low-quality bases. Therefore, no trimming was performed prior to alignment.

**Note:** If trimming were necessary, the following command using `Trimmomatic` could be applied to remove adapters and low-quality bases:

```
# trimming command using Trimmomatic for paired-end reads
trimmomatic PE -phred33 \
  ${reads}/SRR794247_1.fastq.gz ${reads}/SRR794247_2.fastq.gz \
  ${reads}/SRR794247_1.trimmed.fastq.gz ${reads}/SRR794247_1.unpaired.fastq.gz \
  ${reads}/SRR794247_2.trimmed.fastq.gz ${reads}/SRR794247_2.unpaired.fastq.gz \
  ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 \
  LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
```

### Step 2: Mapping Reads to Reference Genome using BWA-MEM

```
echo "STEP 2: Map to reference using BWA-MEM"

# BWA index reference
bwa index ${ref}

# BWA alignment
bwa mem -t 4 -R "@RG\tID:SRR794247\tPL:ILLUMINA\tSM:SRR794247" \
  ${ref} ${reads}/SRR794247_1.filt.fastq.gz ${reads}/SRR794247_2.filt.fastq.gz \
  > ${aligned_reads}/SRR794247.paired.sam
```

### Step 3: Mark Duplicates and Sort BAM File using GATK

```
echo "STEP 3: Mark Duplicates and Sort - GATK"

gatk MarkDuplicatesSpark -I ${aligned_reads}/SRR794247.paired.sam \
                         -O ${aligned_reads}/SRR794247_sorted_dedup_reads.bam
```

### Step 4: Base Quality Recalibration

```
# ----------------------------------
# STEP 4: Base quality recalibration
# ----------------------------------

echo "STEP 4: Base quality recalibration"

# 1. Build the recalibration model
gatk BaseRecalibrator -I ${aligned_reads}/SRR794247_sorted_dedup_reads.bam \
                      -R ${ref} \
                      --known-sites ${known_sites} \
                      -O ${data}/recal_data.table

# 2. Apply the recalibration to adjust base quality scores
gatk ApplyBQSR -I ${aligned_reads}/SRR794247_sorted_dedup_reads.bam \
               -R ${ref} \
               --bqsr-recal-file ${data}/recal_data.table \
               -O ${aligned_reads}/SRR794247_sorted_dedup_bqsr_reads.bam
```

### Step 5: Collect Alignment & Insert Size Metrics

```
# ------------------------------------------------
# STEP 5: Collect Alignment & Insert Size Metrics
# ------------------------------------------------

echo "STEP 5: Collect Alignment & Insert Size Metrics"

gatk CollectAlignmentSummaryMetrics \
    R=${ref} \
    I=${aligned_reads}/SRR794247_sorted_dedup_bqsr_reads.bam \
    O=${aligned_reads}/alignment_metrics.txt

gatk CollectInsertSizeMetrics \
    INPUT=${aligned_reads}/SRR794247_sorted_dedup_bqsr_reads.bam \
    OUTPUT=${aligned_reads}/insert_size_metrics.txt \
    HISTOGRAM_FILE=${aligned_reads}/insert_size_histogram.pdf
```

### Step 6: Call Variants - GATK HaplotypeCaller

```
# ------------------------------------------------
# STEP 6: Call Variants - gatk haplotype caller
# ------------------------------------------------

echo "STEP 6: Call Variants - gatk haplotype caller"

gatk HaplotypeCaller \
    -R ${ref} \
    -I ${aligned_reads}/SRR794247_sorted_dedup_bqsr_reads.bam \
    -O ${results}/raw_variants.vcf
```

### Step 7: Extract SNPs and Indels

```
# Extract SNPs
gatk SelectVariants \
    -R ${ref} \
    -V ${results}/raw_variants.vcf \
    --select-type SNP \
    -O ${results}/raw_snps.vcf

# Extract Indels
gatk SelectVariants \
    -R ${ref} \
    -V ${results}/raw_variants.vcf \
    --select-type INDEL \
    -O ${results}/raw_indels.vcf
```

## Annotation and Filtering

```
# Set base working directory
BASE_DIR="/workspace/project"

# Define reference genome and known sites
ref="${BASE_DIR}/supporting_files/hg38/hg38.fa"

# Input and output directories
results="${BASE_DIR}/results"
data="${BASE_DIR}/data"
```

### Variant Filtering

```
# --------------------
# Filter Variants - GATK
# --------------------

# Filter SNPs
gatk VariantFiltration \
  -R ${ref} \
  -V ${results}/raw_snps.vcf \
  -O ${results}/filtered_snps.vcf \
  -filter-name "QD_filter" -filter "QD < 2.0" \
  -filter-name "FS_filter" -filter "FS > 60.0" \
  -filter-name "MQ_filter" -filter "MQ < 40.0" \
  -filter-name "SOR_filter" -filter "SOR > 4.0" \
  -filter-name "MQRankSum_filter" -filter "MQRankSum < -12.5" \
  -filter-name "ReadPosRankSum_filter" -filter "ReadPosRankSum < -8.0" \
```

```
  -genotype-filter-expression "DP < 10" \
  -genotype-filter-name "DP_filter" \
  -genotype-filter-expression "GQ < 10" \
  -genotype-filter-name "GQ_filter"

# Filter INDELS
gatk VariantFiltration \
  -R ${ref} \
  -V ${results}/raw_indels.vcf \
  -O ${results}/filtered_indels.vcf \
  -filter-name "QD_filter" -filter "QD < 2.0" \
  -filter-name "FS_filter" -filter "FS > 200.0" \
  -filter-name "SOR_filter" -filter "SOR > 10.0" \
  -genotype-filter-expression "DP < 10" \
  -genotype-filter-name "DP_filter" \
  -genotype-filter-expression "GQ < 10" \
  -genotype-filter-name "GQ_filter"
```

## Selecting Variants That Pass Filters

```
# Select Variants that PASS filters

gatk SelectVariants \
  --exclude-filtered \
  -V ${results}/filtered_snps.vcf \
  -O ${results}/analysis-ready-snps.vcf

gatk SelectVariants \
  --exclude-filtered \
  -V ${results}/filtered_indels.vcf \
  -O ${results}/analysis-ready-indels.vcf

# Remove variants that failed genotype filters
cat analysis-ready-snps.vcf | grep -v -E "DP_filter|GQ_filter" >
↪  analysis-ready-snps-filteredGT.vcf
cat analysis-ready-indels.vcf | grep -v -E "DP_filter|GQ_filter" >
↪  analysis-ready-indels-filteredGT.vcf
```

## Annotating Variants - GATK Funcotator

```
# Annotate SNPs using Funcotator
gatk Funcotator \
  --variant ${results}/analysis-ready-snps-filteredGT.vcf \
  --reference ${ref} \
  --ref-version hg38 \
  --data-sources-path /path/to/funcotator_dataSources.v1.7.20200521g \
  --output ${results}/analysis-ready-snps-filteredGT-functotated.vcf \
  --output-file-format VCF

# Annotate INDELs using Funcotator
gatk Funcotator \
  --variant ${results}/analysis-ready-indels-filteredGT.vcf \
  --reference ${ref} \
  --ref-version hg38 \
  --data-sources-path /path/to/funcotator_dataSources.v1.7.20200521g \
  --output ${results}/analysis-ready-indels-filteredGT-functotated.vcf \
  --output-file-format VCF
```

```
echo "Annotation completed successfully."

# Optional: summarize the number of annotated variants
echo "Summary of annotated SNP variants:"
grep -v "^#" ${results}/analysis-ready-snps-filteredGT-functotated.vcf | wc -l

echo "Summary of annotated INDEL variants:"
grep -v "^#" ${results}/analysis-ready-indels-filteredGT-functotated.vcf | wc -l
```