**Title: Variant Calling Analysis Using GATK**

**Introduction**

In this I give a brief introduction on each step along the way for performing variant calling analysis using GATK4. I am following the GATK4 best practices workflow - <https://gatk.broadinstitute.org/hc/en-us/articles/360035535932-Germline-short-variant-discovery-SNPs-Indels->.

* **Variant calling analysis and its importance**: Variants/mutation in the genetic structure of an organism can provide a plethora of information. Ranging from helping us understand the disease mechanism to create personalized treatment plans based on their genetic profile ultimately leading to more precise and effective medicine.
* **Overview of GATK(Genome Analysis Tool Kit)**: GATK is basically an Industry standard toolkit for human whole genome and whole exome sequencing data. It can be used for NGS data processing, genotyping and variant discovery, variant filtering and evaluation. It is a package of command line tools written in JAVA.
* **Purpose:** I aim to perform a series of steps on sequencing reads following GATK Best practice germline short variant discovery(SNPs + Indels) Workflow to create a VCF file with set of genetic variants.

**Prerequisites**

* + **Data Requirements**:
  + Reference genome (e.g., GRCh38)
  + Input sequencing data (e.g., human WGS paired end reads from Phase 3 individual-HG00096)
  + 2x100 bp reads(Design: Illumina sequencing, paired end random Library)
  + Known variant datasets for base recalibration (e.g.,dbsnp138.vcf)
  + **Tools Used**: GATK(v4.6.0.1), BWA-MEM, SAMtools(v…….), FastQC, MultiQC, Picard, and R for visualization.
  + **System Setup**: Instructions for installing GATK and dependencies.

A screenshot of a computer

Description automatically generated

**Workflow Steps**

1. **Data Preparation**
2. **Quality Control**

* Use tools like FastQC and Trimmomatic to assess and clean raw reads.
* we start with raw unmapped reads in fastq files and first perform quality control to make sure there are no adapter sequence. If there are any adapter sequence present we trim them out.

fastqc ${reads}/SRR062634\_1.filt.fastq.gz -o ${reads}/ #contains forward pairs

fastqc ${reads}/SRR062634\_2.filt.fastq.gz -o ${reads}/ #reverse reads

* Looking at the quality report no such poor-quality sequence was found hence there is no need to perform trimming.

1. **Read Alignment**

* Map reads to the reference genome(GRCh38) using **BWA-MEM**.
* After being mapped to a reference genome, the aligned reads are stored in BAM/SAM file.
* BWA can be used to index reference as it allows for more efficient search of the genome while performing alignment.

bwa index ${ref}

* We perform alignment using bwa mem, using 4 threads and providing read group information(-R). storing the final sam file in aligned reads folder.

bwa mem -t 4 -R "@RG\tID:SRR062634\tPL:ILLUMINA\tSM:SRR062634" ${ref} ${reads}/SRR062634\_1.filt.fastq.gz ${reads}/SRR062634\_2.filt.fastq.gz > ${aligned\_reads}/SRR062634.paired.sam

$samtools view SRR062634.paired.sam | less

$samtools flagstat SRR062634.paired.sam

1. **Mark Duplicates and sort** 
   * It is important to filter out duplicate reads as these are not informative and may also hinder by overrepresenting in certain areas leading to unwanted and unexpected results.
   * We create a new column called bitwise flag and the flag value indicates that the read is flagged as a duplicate
   * we use the MarkDuplicatesSpark tool from GATK, it marks the duplicate reads as well as sorts the alignment files.
2. **Base Quality Score Recalibration (BQSR)**
   * The last step in preprocessing is Recalibrate base quality scores. This step is very much recommended in case of Variant calling analysis.
   * we provide GATK Base Recalibrator with a set of known variants. GATK base recalibrator analyses all reads looking for mismatches between the read and reference, and skips positions that are included in the set of known variants. It computes statistics on the mismatches based on the reported quality score, the position in the read and the sequencing context. And based on the statistics, an empirical quality score is assigned to each mismatch overwriting the original reported quality score.
   * In cases when we do not have a known set of variants, these can be generated by calling variants first without RBQS and filtering the variants to obtain high quality variants and then using that to Recalibrate. This step is called the Bootstrapping method.
   * **Show GATK commands for recalibration.**
3. **Variant Discovery**

* The next step in the process is variant discovery, using algorithm in the analysis ready reads that we generated in the previous step through preprocessing. The variant calling algorithm choice is made based on some criteria (Germline/somatic variants, diploid organism, number of samples, pooled/unpooled samples).
* For our Germline diploid one sample read we chose **HaplotypeCaller.**
* HaplotypeCaller is a computational algorithm used to identify genetic variations (variants) within a genome by reconstructing the possible haplotypes (combinations of alleles on a chromosome) present in a sample.
* Some downsides of HaplotypeCaller is it is not recommended when we have more than 100 samples as it can create a high number of false positive
* **Include a snippet of the GATK command and mention common pitfalls.**

1. **Joint Genotyping**

• Explain how **GenotypeGVCFs** combines multiple samples for variant calling.

• Example outputs showing SNPs and indels.

**6. Variant Filtering**

• Use **VariantFiltration** to apply quality filters.

• Include an example of thresholds and filtering logic.

**Results**

• Present a table or plot summarizing the variants (e.g., SNP/indel count, transitions/transversions).

• Visualize variants using a genome browser (e.g., IGV) or plotting tools in R.

**Discussion**

• Reflect on the quality and significance of the results.

• Discuss possible applications, such as identifying functional variants or creating a dataset for downstream analyses.

**Challenges and Tips**

• Mention potential issues (e.g., low coverage, false positives).

• Share troubleshooting advice and recommendations for optimization.

**Conclusion**

• Summarize the workflow and results.

• Emphasize the importance of reproducibility and scalability.

• Suggest next steps (e.g., annotating variants with tools like ANNOVAR).

**Additional Resources**

• Link to the official GATK documentation and tutorials.

• Share GitHub repo or script files for reproducibility.

**Call to Action**

• Invite readers to try the workflow and share feedback.

• Encourage them to ask questions or suggest future blog topics.

Would you like help elaborating any section or refining the commands and visuals?