**Opti-GATK4 manual**

**Introduction**

The GATK toolkit was primarily used to call variants in sequencing data in humans. It seems like it was initially optimized for the human genome as default values for the many parameters are for *homo sapiens*. We know that the genome of *Plasmodium falciparum* has a different organization with an AT content > 80% and abundant low-complexity tandem-repeat regions. While set to 2 in germline variant calling in humans, the computational ploidy for malaria parasite samples collected from patients in endemic areas is generally unknown and variable. Furthermore, the mutation rate for malaria parasites is higher, so the default GATK settings can underperform in *falciparum* variant calling and the base recalibration step may not be recommended as it can lead to an underestimation of true variants. Sequencing the entire genome of *P. falciparum* from clinical samples is technically hard to achieve due to several reasons such as low parasite density, contamination by human DNA, polygenomic infections, etc… The quality of the read data remains affected by these factors although existing enrichment methods, such as selective whole genome amplification (sWGA), are performed to increase the yield. It is important to note that sWGA may still struggle to effectively amplify some regions of the genome which can lead to an inconsistent read coverage.

To deal with some of these issues, we developed an optimized variant calling pipeline for *P. falciparum* whole genome sequencing (WGS) based on GATK version 4 (GATK4). We altered several parameters in HaplotypeCaller and GenotypeGVCFs that control read and mapping quality, mutation rate for both SNPs and indels and tandem repeats. The pipeline uses a custom high-quality *in silico* training dataset generated from accurate PACBIO assemblies of 10 lab strains instead of mendelian errors-rich cross data to filter out low-quality variants by machine learning recalibration. We tested it in older low-quality and current high-quality sequencing data in both single and mixed infection samples (<https://pubmed.ncbi.nlm.nih.gov/37420214/>).

The Opti-GATK4 pipeline is for short indel and SNP calling. We recommend the polyploid mode for the detection of drug resistance mutation from WGS data.

**Software requirements and packaging**

* gatk4
* picard
* Trimmomatic
* bwa
* samtools
* bcftools
* datamash

Source commands can be found on the github page. <https://github.com/Karaniare/Optimized_GATK4_pipeline/tree/main>

A sif container of the pipeline using the latest version of gatk is available on seekdeep /nfs/jbailey5/baileyweb/bailey\_share/bin/ .

Container size : 7.9G. We will upload a copy into the Bailey lab website.

**General use**

**The first part** of the pipeline (**PfWGSqc**) performs read trimming and mapping and generates QC reports from improved bam files, including insert sizes, read coverages, estimates of human read contamination and many secondary quality parameters. This step generates data to calculate repool/rebalance volumes if additional reads are needed before you proceed with the variant calling.

* **PfWGSqc:**
  + -i : list of [sample IDs](https://docs.google.com/document/d/1y_mXfAxnYOXf3EGvKxWVw6U4LlQ6nTbwGyWHKli8lGw/edit) in a text file.
  + -f: full path to the fastq files.
  + -u: directory to keep unpaired fastqs after trimming.
  + -s: directory to keep QC stats.
  + -b: bam file directory.
  + -k: name of the library prep kit (Ex.: TruSeq3-PE.fa or /opt/data/Nextera-PE.fa).
  + -r: run name (Ex: run1), useful if you run the pipeline on separate samplesets in parallel.

**The second part** does the variant calling itself and includes the following packages:

* **MakingGVCFs** : generates gVCFs from improved bam files from **PfWGSqc**.
  + -i: input file ([sample ID list](https://docs.google.com/document/d/1y_mXfAxnYOXf3EGvKxWVw6U4LlQ6nTbwGyWHKli8lGw/edit)) in tab separated value format.
  + -c: chromosome number (ex.: 1).
  + -b: bam file directory.
  + -g: directory to save gVCF files.
* **MakingVCFs** : performs per chromosome joint genotyping on gVCFs and generates several small VCFs by chromosome chunks.
* -c: chromosome number (ex.: 1).
* -v: directory to save VCF files.
* -t: file containing a list of genomic regions (chromosome chunks) to call variants at, one region per line (see [example](https://docs.google.com/document/d/1y7HbVvv2cSY8c12eRjLkZLm_VtNK0rLsXbf2sV-U9Cs/edit) for chromosome 1 ).
* **NB:** files specifying the gVCF IDs (first column) and the full paths to them (second column) should be created in the VCF directory by the user (see [example](https://docs.google.com/document/d/1AGRRhcDDm-vhgrmg0mqA1pcSmt1HW1hykiIfpbRVHy4/edit) for chromosome 1 and and the file should be named as gvcf\_chr1\_list.tsv for chromosome 1, gvcf\_chr2\_list.tsv for chromosome2 and so on)

**Important notice:** For the first attempt, this step generates one bash file by chromosome chunks that need to be run outside the container. They can be submitted as separate slurm jobs. Once they are completed, please rerun this container step again.

* **GatherFilterVCFs** : assembles the small VCFs into one big VCF by chromosome and performs variant recalibration on SNPs and Indels separately.
  + -ms: Gaussian model for snp (ex.: 4).
  + -v : vcf file directory (a.k.a variant calling output).
  + -mi: Gaussian model for indel (ex.: 4).
* **AnnotatingVCFs**: performs functional annotation of the final VCFs.
  + -i: list of VCFs [file](https://docs.google.com/document/d/1fXajfIyT08bvUd2lkVJrTVfILRw0oeG-kbsSeXr63e8/edit) name to annotate (only names without .vcf.gz) in tab separated value format and should be in the VCF directory.
  + -v : VCF file directory (directory of variant call outputs).

**Tutorial** (QC) using the singularity (sif) container on oscar

* Create directories for the analysis
* mkdir /nfs/jbailey5/baileyweb/kniare/Fastq\_dir ## directory for fastq files.
* mkdir /nfs/jbailey5/baileyweb/kniare/Bam\_dir ## directory for bam files.
* mkdir /nfs/jbailey5/baileyweb/kniare/Stat\_dir ## directory for QC output files.
* mkdir /nfs/jbailey5/baileyweb/kniare/Unpaired\_dir ## directory for unpaired fastq files.
* mkdir /nfs/jbailey5/baileyweb/kniare/gVCF\_dir ## directory for gVCFs.
* mkdir /nfs/jbailey5/baileyweb/kniare/VCF\_dir ## directory for VCFs.
* Create [sample lis](https://docs.google.com/document/d/1y_mXfAxnYOXf3EGvKxWVw6U4LlQ6nTbwGyWHKli8lGw/edit)t file and place it in the analysis directory /nfs/jbailey5/baileyweb/kniare.
* Make a bash file to run PfWGSqc within the container
* vi /nfs/jbailey5/baileyweb/kniare/run\_opti-gatk4.sh.
* Paste the following details in the bash file (run\_opti-gatk4.sh) and save

#!/bin/bash

#SBATCH -J Gatk4\_controls\_plate3\_n4

#SBATCH -t 24:00:00

#SBATCH -c 8

#SBATCH --mem-per-cpu=10g

analysis=”/nfs/jbailey5/baileyweb/kniare”

container=”/nfs/jbailey5/baileyweb/bailey\_share/bin/opti\_gatk4\_230825.sif”

singularity exec -B $analysis:/opt/input $container \

PfWGSqc -i /opt/input/sample\_list.tsv \

-b /opt/input/Bam\_dir \

-f /opt/input/Fastq\_dir \

-u /opt/input/Unpaired\_dir \

-s /opt/input/Stat\_dir \

-r run1 -k /opt/data/TruSeq3-PE.fa

* chmod +x /nfs/jbailey5/baileyweb/kniare/run\_opti-gatk4.sh
* sbatch /nfs/jbailey5/baileyweb/kniare/run\_opti-gatk4.sh
* NB: replace /nfs/jbailey5/baileyweb/kniare by the actual directory where you will be running the analysis.