Whole exome sequencing data analysis by GATK

(germline short variant discovery)

whole exome sequencing

Exome sequencing is a method that enables the selective sequencing of the exonic regions of a genome - that is the transcribed parts of the genome present in mature mRNA, including protein-coding sequences, but also untranslated regions (UTRs).

In humans, there are about 180,000 exons with a combined length of ~ 30 million base pairs (30 Mb). Thus, the exome represents only 1% of the human genome, but has been estimated to harbor up to 85% of all disease-causing variants.

Exome sequencing, thus, offers an affordable alternative to whole-genome sequencing in the diagnosis of genetic disease, while still covering far more potential disease-causing variant sites than genotyping arrays. This is of special relevance in the case of rare genetic diseases, for which the causative variants may occur at too low a frequency in the human population to be included on genotyping arrays.

In this tutorial, we will apply WES analysis by using GATK. In the following, GATK is explained briefly.

GATK

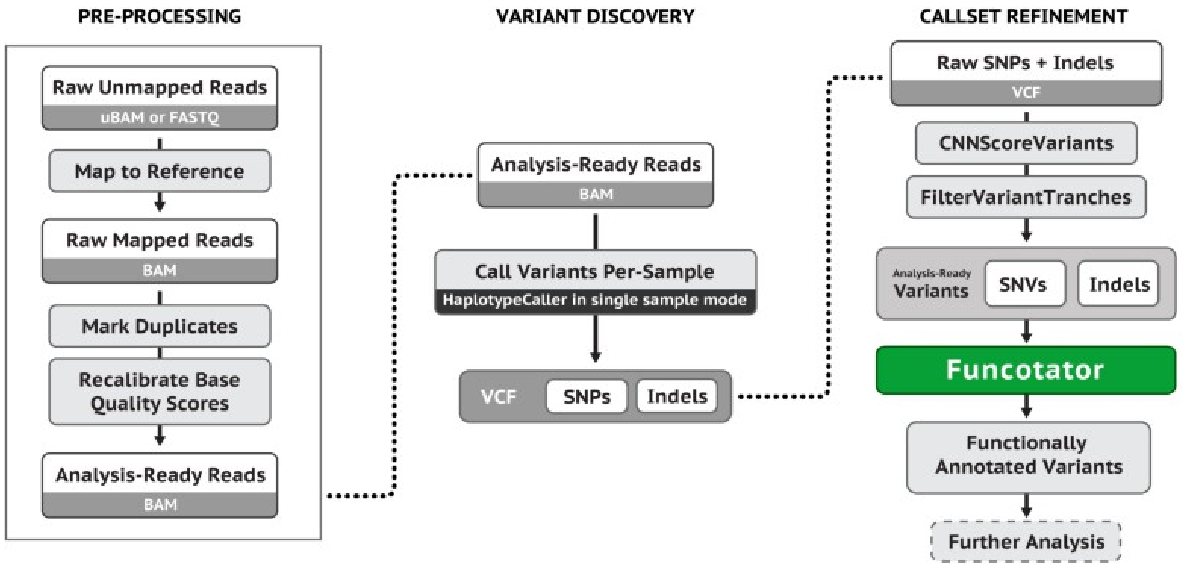
GATK (pronounced*"Gee-ay-tee-kay"*, not*"Gat-kay"*), stands for **G**enome**A**nalysis**T**ool**k**it. It is a collection of command-line tools for analyzing high-throughput sequencing data with a primary focus on variant discovery. The tools can be used individually or chained together into complete workflows.

Here, we will learn how to use GATK step by step and manually.
The next section which includes GATK pipelines will be discussed comprehensively.

Germline short variant discovery by GATK4

Single sample variant discovery uses HaplotypeCaller in its default single-sample mode to call variants in an analysis-ready BAM file. The VCF that HaplotypeCaller emits errs on the side of sensitivity, so some filtering is often desired. To filter variants first run the CNNScoreVariants tool. This tool annotates each variant with a score indicating the model's prediction of the quality of each variant. To apply filters based on those scores run the FIlterVariantTranches tool with SNP and INDEL sensitivity tranches appropriate for your task.

Main steps for Germline Single-Sample Data



Preprocessing of the raw reads

The following steps prepare reads for analysis and must be performed in sequence.

Quality control

Once the adaptors have been trimmed, it is useful to inspect the quality of reads in bulk, and try to trim low quality nucleotides. Also, frequently the quality tends to drop off toward one end of the read. FASTQC and PrinSeq will show that very nicely . These read ends with low average quality can then be trimmed, if desired, using Trimmomatic, FASTX-Toolkit fastq_quality_filter, PrinSeq, or SolexaQA.

Adaptor trimming

Sequencing facilities usually produce read files in fastq format, which contain a base sequence and a quality score for each base in a read. Usually the adaptor sequences have already been removed from the reads, but sometimes bits of adapters are left behind, anywhere from 90% to 20% of the adaptor length. These need to be removed from the reads. This can be done using your own script based on a sliding window algorithm. A number of tools will also perform this operation: Trimmomatic, Fastx-toolkit (fastx_clipper), Bioconductor (ShortRead package), Flexbar, as well as a number of tools listed on BioScholar and Omics tools databases.

Selection of the tool to use depends on the amount of adaptor sequence leftover in the data. This can be assessed manually by grepping for parts of known adaptor sequences on the command line.

Removal of very short reads

Once the adaptor remnants and low quality ends have been trimmed, some reads may end up being very short (i.e. <20 bases). These short reads are likely to align to multiple (wrong) locations on the reference, introducing noise into the variation calls. They can be removed using PrinSeq, Trimmomatic (using the MINLEN option), or a simple in-house script. Minimum acceptable read length should be chosen based on the length of sequencing fragment: longer for longer fragments, shorter for shorter ones – it is a matter of some experimentation with the data.

The three pre-processing steps above can be parallelized by chunking the initial fastq file (hundreds of millions of reads, up to 50-150 G of hard disk space per file depending on sequencing depth) into several files that can be processed simultaneously. The results can then be combined.

Initial variant discovery

Analysis proceeds as a series of the following sequential steps.

Alignment

Reads need to be aligned to the reference genome in order to identify the similar and polymorphic regions in the. As of 2016, the team recommends their b37 bundle as the standard reference for Whole Exome and Whole Genome Sequencing analyses pending the completion of the GRcH38/Hg38 bundle However, the 2018 functional equivalence specifications recommends the GRCh38DH from the 1000 Genomes project. Either way, a number of aligners can perform the alignment task.

Among these, BWA MEM and bowtie2 have become trusted tools for short reads Illumina data, because they are accurate, fast, well supported, and open-source. Combined with variant callers, different aligners can offer different performance advantages with respect to SNPs, InDels and other structural variants, benchmarked in works like specifications recommends BWA-MEM v0.7.15 in particular (with at least the following parameters-K 100000000 -Y, and without-M so that split reads are marked as supplementary reads in congruence with specification).

The output file is usually in a binary format still taking tens or hundreds of Gigabytes of hard disk space. The alignment step tends to be I/O intensive, so it is useful to place the reference onto an SDD, as opposed to HDD, to speed up the process. The alignment can be easily parallelized by chunking the data into subsets of reads and aligning each subset independently, then combining the results.

De-duplication

The presence of duplicate reads in a sequencing project is a notorious problem. The causes are discussed in a blog post by Eric Vallabh Minikel (2012). Duplicately sequenced molecules should not be counted as additional evidence for or against a putative variant – they must be removed prior to the analysis. A number of tools can be used including: samblaster, sambamba, the commercial novosort from the novocraft suit, Picard, and FASTX-Toolkit has fastx_collapser for this purpose. Additionally, MarkDuplicates is shipped as part of GATK4, but is called from Picard tools in older releases. For functional equivalence, it is recommended to use Picard tools v>2.4.1.

De-duplication can also be performed by a simple in-house written Perl script.

Artifact removal: local realignment around indels

Some artifacts may arise due to the alignment stage, especially around indels where reads covering the start or the end of an indel are often incorrectly mapped. This results in mismatches between the reference and reads near the misalignment region, which can easily be mistaken for SNPs. Thus, the realignment stage aims to correct these artifacts by transforming those regions with misalignment due to indels into reads with a consensus indel for correct variant calling.

Realignment can be accomplished using the IndelRealigner. Alternatives include Dindel and SRMA.

The inclusion of the realignment stage in a variant calling pipeline depends on the variant caller used downstream. This stage might be of value when using non-haplotype-aware variant caller like the UnifiedGenotyper. However, if the tool used for variant calling is haplotype-aware like Platypus, FreeBayes or the HaplotypeCaller, then it is not needed nor recommended. The recommendations starting from their 3.6 release onwards, and the guidelines for functional equivalence also vote against this stage.

Ultimately however, characteristics of the dataset at hand would dictate whether realignment and other clean-up stages are needed. Ebbert et al paper for example argues against PCR duplicates removal, while Olson et al recommends all the stages of clean up applied to the dataset at hand. Some experimentation is therefore recommended when handling real datasets.

Base quality score recalibration

Base quality scores, which refer to the per-base error estimates assigned by the sequencing machine to each called base, can often be inaccurate or biased. The recalibration stage aims to correct for these errors via an empirical error model built based on the characteristics of the data at hand. The quality score recalibration can be performed using's BQSR protocol, which is also the recommendation for functional equivalence, along with specific reference genome files. For speed up of analysis, and if using < v4, one may skip the PrintReads step and pass the output from BaseRecalibrator to the HaplotypeCaller directly. Bioconductor's ReQON is an alternative tool for this purpose.

Calling the variants

There is no single "best" approach to capture all the genetic variations. For germline variants,suggest using a consensus of results from three tools:

1.CRISP,

2.HaplotypeCaller from the, and

3.mpileup from SAMtools

Recently, MuTect2 was added as a variant discovery tool to the specifically for cancer variants. MuTect2 calls somatic SNPs and indels by combining the original MuTect with the HaplotypeCaller. The HaplotypeCaller relies on diploid assumption, while MuTect2 allows for different allelic fractions for each variant. This makes the caller useful in tumor variant discovery. Joint calling (GVCF generation) is not available in MuTect2.

The variant calls are usually produced in the form of files, occupying much smaller size than the BAMs generating them.

Variant annotation and prioritization

This last preliminary stage is highly dependent on the study design and objectives, so only a brief coverage is provided herein.

Statistical filtering

The files resulting from the previous steps frequently have many sites that are not really genetic variants, but rather machine artifacts that make the site statistically non-reference. In small studies, hard filtering of variants based on annotations of genomic context is typically sufficient.

While, it requires expertise to define appropriate filtering thresholds, Heng Li provides some general guidelines in this paper. For experiments with a sufficiently large number of samples (30 or more), the team designed the Variant Quality Score Recalibrator (VQSR) protocol to separate out the false positive machine artifacts from the true positive genetic variants using a Gaussian Mixture model based on the learned annotations of known datasets.

Annotation and prioritization

This phase serves to select those variants that are of particular interest, depending on the research problem at hand. The methods are specific to the problem, thus we do not elaborate on them, and only provide a list of some commonly used tools below:

•Generating variant and level annotations, and performing many other exploratory and filtration analysis types: Hail

•Exploring and prioritizing genetic variation in the the context of human disease: GEMINI

•Mendelian disease linked variants: VAR-MD, KGGSeq, FamSeq.

•Predicting the deleteriousness of a non-synonymous single nucleotide variant: dbNSFP, HuVariome, Seattle-Seq, ANNOVAR, VAAST, snpEff

•Identifying variants within the regulatory regions: RegulomeDB

To apply all the mentioned steps, we need to collect proper input files and tools. These files are :

High quality reads(fastq files; single or paired end)

Reference genome files(should be downloaded from: https://gatk.broadinstitute.org/hc/en-us/articles/360035890811-Resource-bundle)

GATK jar file (should be downloaded from: https://github.com/broadinstitute/gatk/releases)

Picard jar file (should be downloaded from: https://broadinstitute.github.io/picard/)

After preparing all the required files, in the terminal of your system(Unix systems) the process of analyzing the WES files can be carried out step by step as below:

############### Run manually step by step ##

**** all the mentioned files and their addresses need to be modified based on your own system

1- QC with Fastqc

2- Map to refrence with bwa $bwa mem -M -K 100000000 -p ~/Downloads/cromwell/hg38/Homo_sapiens_assembly38.fasta S1389Nr15.1.fastq.gz S1389Nr15.2.fastq.gz > S1389Nr15.sam

3- Convert sam to bam $samtools view -S -b S1389Nr15.sam > S1389Nr15.bam

4- Add read groups to bam file $ java -jar picard.jar AddOrReplaceReadGroups I=S1389Nr15.bam O=S1389Nr15_RG.bam RGID=4 RGLB=twist RGPL=illumina RGPU=unit1 RGSM=S1389Nr15

5- Sort bam file $java -jar picard.jar SortSam I=S1389Nr15_RG.bam O=S1389Nr15_RG_sorted.bam SORT_ORDER=coordinate

6- Validate sortedbam file $java -jar ~/Downloads/cromwell/WES_analysis/S1389NR15/picard.jar ValidateSamFile I=S1389Nr15_RG_sorted.bam MODE=SUMMARY

7- Index sortedbam file $ java -jar picard.jar BuildBamIndex I=S1389Nr15_RG_sorted.bam

8- Mark duplicates $java -jar picard.jar MarkDuplicates I=S1389Nr15_RG_sorted.bam O=S1389Nr15_marked_duplicates.bam M=S1389Nr15marked_dup_metrics.tx

9- Index marked_duplicates.bam file

```
$java -jar picard.jar BuildBamIndex I=S1389Nr15_marked_duplicates.bam
```

10- Run haplotypecaller

```
$java -jar ~/Downloads/cromwell/WES_analysis/S1389NR15/gatk-package-4.1.2.0-local.jar HaplotypeCaller -R ~/Downloads/cromwell/hg38/Homo_sapiens_assembly38.fasta -I S1389Nr15_marked_duplicates.bam -O S1389Nr15.vcf
```

(bam file must be indexed)

11- Variant filtering by VariantRecalibrator for snp mode

```
$java -jar ~/Downloads/cromwell/WES_analysis/S1389NR15/gatk-package-4.1.2.0-local.jar VariantRecalibrator -R ~/Downloads/cromwell/hg38/Homo_sapiens_assembly38.fasta -V S1389Nr15.vcf --resource:hapmap,known=false,training=true,truth=true,prior=15.0 ~/Downloads/cromwell/hg38/hapmap_3.3.hg38.vcf.gz --resource:omni,known=false,training=true,truth=false,prior=12.0 ~/Downloads/cromwell/hg38/1000G_omni2.5.hg38.vcf.gz --resource:1000G,known=false,training=true,truth=false,prior=10.0 ~/Downloads/cromwell/hg38/1000G_phase1.snps.high_confidence.hg38.vcf.gz --resource:dbsnp,known=true,training=false,truth=false,prior=2.0 ~/Downloads/cromwell/hg38/beta/Homo_sapiens_assembly38.dbsnp138.vcf -an QD -an MQ -an MQRankSum -an ReadPosRankSum -an FS -an SOR -mode SNP -O S1389Nr15_output.recal --tranches-file S1389Nr15_output.tranches --rscript-file S1389Nr15_output.plots.R
```

12- Variant filtering by VariantRecalibrator for indel mode

```
$java -jar ~/Downloads/cromwell/WES_analysis/S1389NR15/gatk-package-4.1.2.0-local.jar VariantRecalibrator -R ~/Downloads/cromwell/hg38/Homo_sapiens_assembly38.fasta -V S1389Nr15.vcf --resource:dbsnp,known=true,training=false,truth=false,prior=2.0 ~/Downloads/cromwell/hg38/dbsnp_138.hg38.vcf.gz --resource:mills,known=false,training=true,truth=true,prior=12.0 ~/Downloads/cromwell/hg38/Mills_and_1000G_gold_standard.indels.hg38.vcf.gz -an QD -an MQRankSum -an ReadPosRankSum -an FS -an InbreedingCoeff -an SOR -mode INDEL -O S1389Nr15_indel_output.recal --tranches-file S1389Nr15_indel_output.tranches --rscript-file S1389Nr15_indel_output.plots.R
```

13- Apply recalibration/filtering by ApplyVQSR on snp file

```
$java -jar ~/Downloads/cromwell/WES_analysis/S1389NR15/gatk-package-4.1.2.0-local.jar ApplyVQSR -R ~/Downloads/cromwell/hg38/Homo_sapiens_assembly38.fasta -V S1389Nr15.vcf -O S1389Nr15_SNP.vcf --truth-sensitivity-filter-level 99.0 --tranches-file S1389Nr15_snp_output.tranches --recal-file S1389Nr15_snp_output.recal -mode SNP
```

14- Apply recalibration/filtering by ApplyVQSR on indel file $java -jar ~/Downloads/cromwell/WES_analysis/S1389NR15/gatk-package-4.1.2.0-local.jar ApplyVQSR -R ~/Downloads/cromwell/hg38/Homo_sapiens_assembly38.fasta -V S1389Nr15.vcf -O S1389Nr15_INDEL.vcf --tranches-file S1389Nr15_indel_output.tranches --recal-file S1389Nr15_indel_output.recal -mode INDEL

15- Annotate each file (snp, inedel) seperatly with Wannovar

References:

https://gatk.broadinstitute.org/hc/en-us/articles/360036194592-Getting-started-with-GATK4

https://gatk.broadinstitute.org/hc/en-us/articles/360035535932-Germline-short-variant-discovery-SNPs-Indels-

https://h3abionet.github.io/H3ABionet-SOPs/Variant-Calling