HIGH THROUGHPUT SEQUENCING

PIPELINE ASSIGNMENT

Co-ORDINATOR: Mr. Lujumba Ibra

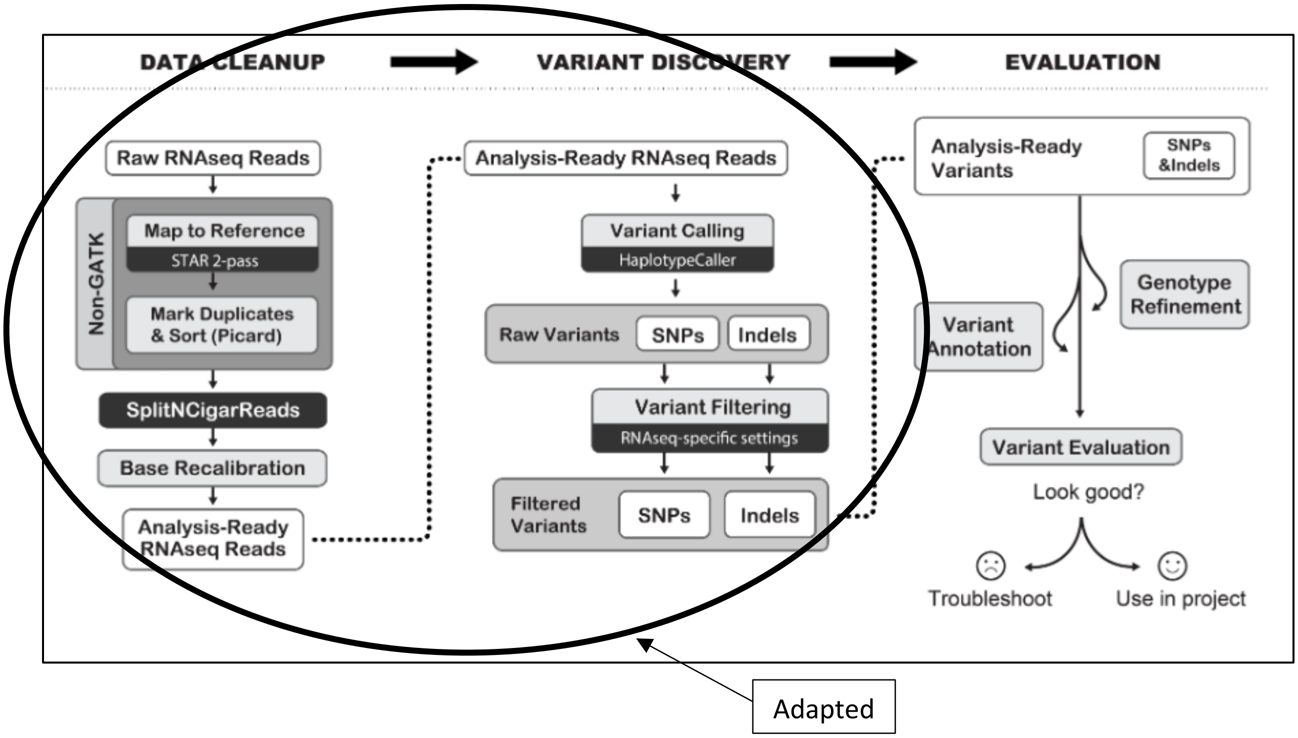
GROUP MEMBERS:

YIGA FAHIM	2022/HD07/2054U
KIMERA REAGAN	2022/HD07/2045U
KIRABO GLORIA	2022/HD07/2043U
LUTAAYA FESTUS DAVID	2022/HD07/2046U
KATONGOLE MIKE	2022/HD07/5010U
KAKANDE PAUL	2022/HD07/2044U

Question: Write a pipeline calling variants following GATK best practices.

Answer:

RNAseq short variant discovery (SNPs + Indels); GATK best practices



SUMMARY

The pipeline performs quality control, alignment, duplicate marking, and variant calling on multiple fastq files using a combination of tools and the GATK toolkit. The pipeline also includes steps to prepare the reference genome and download a vcf file from Ensembl, which contains variant information for the Bos taurus genome. The pipeline applies a series of filters to the variant calls and output separate files for SNPs and INDELs.

- Quality assessment (fastqc, multiqc)
- Building reference indexes for hisat2 (hisat2)
- Alignment of query sequences against reference (hisat2)
- Sorting Sam files by genome coordinates-simultaneously converting to bam (picard tools)
- Indexing the bam files (samtools) for loop
- Marking duplicate reads in the bam files (picard tools)
- Assigning reads to read groups required by GATK (picard tools)
- Indexing the final bam files (picard tools)
- Creating reference sequence dictionary for GATK analysis (gatk4)
- Indexing the reference genome (samtools)
- Downloading the known sites/ known variants file from Ensemble (wget)
- Unzipping the downloaded file (gzip)
- Re-zipping the unzipped file to a tabix readable format (bgzip)
- Indexing the output vcf (tabix)
- Handling spicing events by splitting N cigars (gatk4)
- Building a base quality recalibration model using all reads in the bam files and vcf (gatk4)
- Applying the built model to recalibrate base quality scores in the bam files (gatk4)
- Variant calling but keeping all genomic-sites information (gatk4-haplotype caller)
- *Included a comment for combination of gvcfs incase multiple ones have been created*
- Genotyping the gvcfs to create raw SNP and indel VCFs (gatk4) for loop
- Unzipping the vcfs (gzip)
- Selecting variants according to SNPs (gatk4)
- Selecting variants according to Indels (gatk4)
- Filtering variants according to SNPs considering all mandatory columns (gatk4)
- Filtering variants according to indels considering all mandatory columns (gatk4)

Quality Control:

The first section of the script uses fastqc and multiqc to perform quality control on the fastq files. The command "fastqc *.fastq" runs fastqc on all fastq files in the current directory, generating quality control reports for each file. The command "multiqc ." runs multiqc on the current directory, aggregating the results from the fastqc runs into a single report.

Alignment and Pre-processing:

The next section of the script uses hisat2 to align the fastq files to a reference genome (Bos_taurus.fa), sorts and indexes the resulting alignments, and marks and removes duplicates.

The command "hisat2-build Bos_taurus.fa Bos_taurus.idx" builds a hisat2 index for the reference genome.

The for loop "for i in ls *.fastq | sed 's/_[12].fastq//g' |sort -u" reads in all the fastq files in the current directory, removes the "_1.fastq" or "_2.fastq" suffix, and stores the resulting sample name in the variable "i".

The command "hisat2 -x Bos_taurus.idx -1 ${i}_1.fastq -2 ${i}_2.fastq -S ${i}.sam" runs hisat2 to align the fastq files to the reference genome using the index created earlier. The option -x specifies the index prefix, the option -1 and -2 specify the input fastq files, and the option -S specifies the output file in SAM format.

The command "picard SortSam -INPUT ${i}.sam -OUTPUT ${i}_sorted.bam -SORT_ORDER coordinate" sorts the alignments in the SAM file by coordinate using the picard SortSam tool. The option -INPUT specifies the input file, -OUTPUT specifies the output file, and -SORT_ORDER specifies that the alignments should be sorted by coordinate.

The command "samtools index ${i}_sorted.bam" indexes the sorted BAM file using samtools.

The command "picard MarkDuplicates -INPUT ${i}_sorted.bam -OUTPUT ${i}_sorted_dedup.bam -METRICS_FILE dedup_metrics.txt" marks and removes duplicates from the sorted BAM file using picard MarkDuplicates. The option -INPUT specifies the input file, -OUTPUT specifies the output file, and -METRICS_FILE specifies the file to write the duplication metrics to.

The command "picard AddOrReplaceReadGroups -I ${i}_sorted_dedup.bam -O ${i}_sorted_dedup_RG.bam -RGID 1 -RGLB lib2 -RGPL illumina -RGPU unit1 -RGSM 3" adds read groups to the BAM file using picard AddOrReplaceReadGroups. The option -I specifies the input file, -O specifies the output file, and the other options specify the read group information.

The command "picard BuildBamIndex -INPUT ${i}_sorted_dedup_RG.bam" creates an index for the BAM file using picard BuildBamIndex. The option -INPUT specifies the input file.

Reference genome preparation:

This section of the script uses GATK to create a sequence dictionary, index the reference genome, download and unpack a vcf file from ensembl, and then use the GATK to filter and call variants using the reference genome, alignments, and vcf file.

The command "gatk CreateSequenceDictionary -R Bos_taurus.fa" creates a sequence dictionary for the reference genome using GATK. The option -R specifies the reference genome file.

The command "samtools faidx Bos_taurus.fa" indexes the reference genome using samtools faidx.

The command "wget https://ftp.ensembl.org/pub/release-108/variation/vcf/bos_taurus/bos_taurus.vcf.gz" downloads a vcf file from Ensembl, which contains variant information for the Bos taurus genome.

The command "gzip -d bos_taurus.vcf.gz" unpacks the vcf file.

The command "bgzip bos_taurus.vcf" creates a tabix-indexed version of the vcf file using bgzip. The command "tabix -f -p vcf bos_taurus.vcf.gz" creates a tabix index for the vcf file using tabix.

Variants calling:

This section of the script enters a loop that processes each fastq file separately. In this loop, the pipeline uses GATK SplitNCigarReads to split reads that have been aligned using soft-clipping. The pipeline then uses GATK BaseRecalibrator to perform base quality score recalibration, which is a process of adjusting the quality scores of base calls in raw sequence data to account for variations in sequencing quality.

The command "gatk SplitNCigarReads -R Bos_taurus.fa -I ${f}_sorted_dedup_RG.bam -O ${f}_sorted_dedup_RG_splitreads.bam" splits reads that have been aligned using soft-clipping using GATK SplitNCigarReads. The option -R specifies the reference genome file, -I specifies the input file, and -O specifies the output file.

The command "gatk BaseRecalibrator -I ${f}_sorted_dedup_RG_splitreads.bam -R Bos_taurus.fa --known-sites bos_taurus.vcf.gz -O recal_data.table" performs base quality score recalibration using GATK BaseRecalibrator. The option -I specifies the input file, -R specifies the reference genome file, --known-sites specifies the vcf file containing known variants, and -O specifies the output file.

The command "gatk ApplyBQSR -R Bos_taurus.fa -I ${f}_sorted_dedup_RG_splitreads.bam --bqsr-recal-file recal_data.table -O ${f}_recal_reads.bam" applies the recalibration to the original BAM file using GATK ApplyBQSR. The option -R specifies the reference genome file, -I specifies the input file, --bqsr-recal-file specifies the recalibration data file generated by BaseRecalibrator, and -O specifies the output file.

The command "gatk HaplotypeCaller -R Bos_taurus.fa -I ${f}_recal_reads.bam -O ${f}.g.vcf.gz -ERC GVCF" calls variants using GATK HaplotypeCaller, creating a gVCF file for each fastq file. The option -R specifies the reference genome file, -I specifies the input file, -O specifies the output file, and -ERC GVCF specifies that the output should be in gVCF format.

The command "gatk GenotypeGVCFs -R Bos_taurus.fa -V ${f}.g.vcf.gz -O ${f}_raw_variants.vcf.gz" combines the gVCF files and output a raw variant call file in vcf format for each fastq file using GATK GenotypeGVCFs. The option -R specifies the reference genome file, -V specifies the input gVCF files, and -O specifies the output file.

The command "gzip -d ${f}_raw_variants.vcf.gz" unzips the raw variant call file.

The command "gatk SelectVariants -R Bos_taurus.fa -V ${f}_raw_variants.vcf --select-type-to-include SNP -O ${f}_raw_snps.vcf" selects SNPs from the raw variant call file using GATK

SelectVariants. The option -R specifies the reference genome file, -V specifies the input file, --select-type-to-include SNP specifies that only SNPs should be selected, and -O specifies the output file.

The command "gatk SelectVariants -R Bos_taurus.fa -V ${f}_raw_variants.vcf --select-type-to-include INDEL -O ${f}_raw_indels.vcf" selects INDELs from the raw variant call file using GATK SelectVariants. The option -R specifies the reference genome file, -V specifies the input file, --select-type-to-include INDEL specifies that only INDELs should be selected, and -O specifies the output file.

The command "gatk VariantFiltration -V ${f}_raw_snps.vcf -filter "QD < 2.0" --filter-name "QD2" -filter "QUAL < 30.0" --filter-name "QUAL30" -filter "SOR > 3.0" --filter-name "SOR3" -filter"FS > 60.0" --filter-name "FS60" -filter "MQ < 40.0" --filter-name "MQ40" -filter "MQRankSum <-12.5" --filter-name "MQRankSum-12.5" -filter "ReadPosRankSum <-8.0" --filter-name "ReadPosRankSum-8" -O ${f}_filtered_snps.vcf" applies a series of filtering criteria to the raw SNP calls using GATK VariantFiltration. The option -V specifies the input file, and multiple -filter options specify various filtering criteria and the corresponding filter names. The option -O specifies the output file.

The command "gatk VariantFiltration -V ${f}_raw_indels.vcf -filter "QD < 2.0" --filter-name "QD2" -filter "QUAL < 30.0" --filter-name "QUAL30" -filter "FS > 200.0" --filter-name "FS200" -filter "ReadPosRankSum <-20.0" --filter-name "ReadPosRankSum-20" -O ${f}_filtered_indels.vcf" applies similar filtering criteria to the raw INDEL calls using GATK VariantFiltration.