ASSIGNMENT-6

Exercise 1: Understanding Base Recalibration

a. What is the base quality score recalibration (BQSR) step in a variant calling workflow, and why is it important?

Base Quality Score Recalibration (BQSR) is a critical preprocessing step in NGS variant calling pipelines that **adjusts the base quality scores assigned by sequencing machines** to correct systematic errors. These base quality scores estimate the likelihood that a base call is correct, usually expressed in **Phred-scale** (e.g., Q20 = 99% confidence).

Why BQSR is important?

- Sequencing machines are prone to **systematic errors** influenced by:
  - Sequencing chemistry and physics.
  - Instrument-specific biases or manufacturing inconsistencies.
  - Sequence context (neighboring bases) or read position (cycle).
- Base quality scores reported by sequencers are often **not perfectly accurate**, which can mislead variant calling.
  - Overestimated scores may cause false confidence in errors.
  - Underestimated scores may cause true variants to be ignored.
- Correcting these errors ensures that **variant callers weigh the evidence properly**, improving the reliability of SNP and indel calls.

How BQSR works?

Step 1: Build the recalibration model (BaseRecalibrator)

Inputs: Aligned BAM file and a database of **known variants** to avoid counting true variants as errors (e.g., dbSNP for humans). For each base, the tool considers:

- **Read group:** which library or lane the read comes from.
- **Reported quality score:** the original score assigned by the sequencer.
- **Cycle:** position of the base in the read.
- **Sequence context:** neighboring bases (usually dinucleotides or a window of 6 bases).

It counts **mismatches** (excluding known variants) and calculates empirical error rates. The output is a **recalibration table** detailing adjustments needed for different quality bins, cycles, and contexts.

Step 2: Apply the recalibration (ApplyBQSR)

Uses the recalibration table to adjust each base's quality score based on:

- Global differences between reported and empirical qualities.
- Cycle-specific and context-specific effects.
- Read-group-specific effects.

Produces a new BAM file with **more accurate base quality scores**.

Optional QC Step:

Generate **pre- and post-recalibration plots** to visualize the improvement in base quality accuracy. These plots can show how empirical quality scores now align with true error rates.

Benefits of BQSR:

- Corrects systematic errors, reducing false positives and false negatives in variant calling.
- Rescues bases that were undervalued by the sequencer, improving variant detection sensitivity.
- Produces statistically robust base quality scores for downstream analyses like SNP/indel calling, genotyping, and downstream annotation.

b. Which databases can be used as known variant resources in the variant calling pipeline for:

- **Human samples**
- ***Saccharomyces cerevisiae***

Known variant databases are crucial in BQSR **to mask true biological variants** (prevent true variants from being mistaken as errors) while recalibrating sequencing errors.

Human samples:

1. **dbSNP** – widely used database of single nucleotide polymorphisms (SNPs).
2. **1000 Genomes Project** – provides high-confidence SNPs and indels.
3. **Mills and 1000G Gold Standard Indels** – high-quality insertions/deletions dataset.
4. **HapMap** – for common SNPs, sometimes used for calibration.
5. ClinVar
6. genomAD (Genome Aggregation Database)

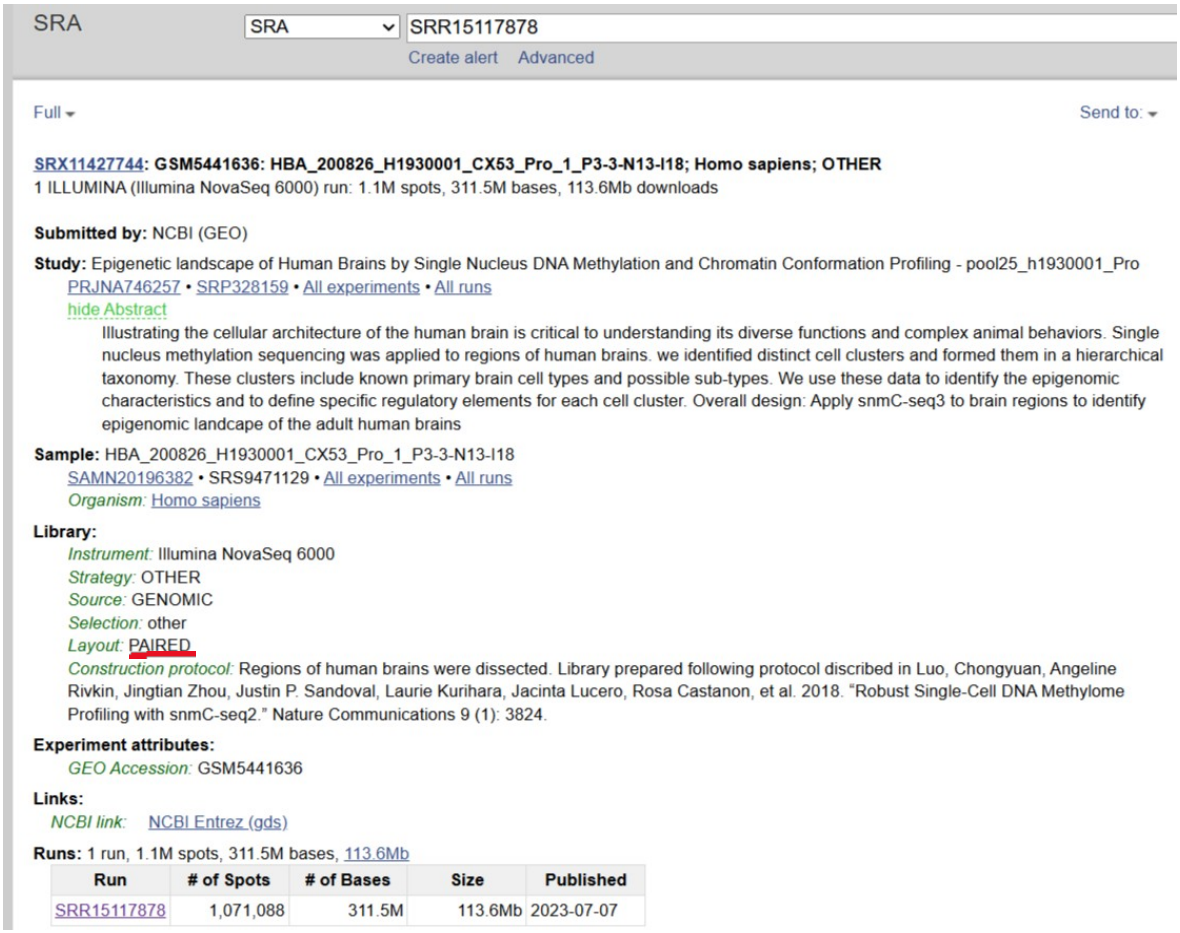
Saccharomyces cerevisiae (yeast):

1. **Saccharomyces Genome Database (SGD)** – provides curated variants for reference strains.
2. **Yeast SNP and Indel datasets** – available from publications or SRA studies, e.g., "S288C strain variants."
3. **Custom variant sets** – in many cases, labs generate a strain-specific variant list for BQSR since public databases may be limited.

Exercise 2: Run a variant calling workflow for chromosome 20 using the UCSC reference genome and process the data up to the MarkDuplicates and AddOrReplaceReadGroups steps.

1. Part A: Data Preparation

a. Download the raw sequencing reads for accession SRR15117878 from the NCBI SRA database.



SRA | SRA | SRR15117878

Create alert Advanced

Full Send to:

SRX11427744: GSM5441636: HBA_200826_H1930001_CX53_Pro_1_P3-3-N13-I18; Homo sapiens; OTHER
1 ILLUMINA (Illumina NovaSeq 6000) run: 1.1M spots, 311.5M bases, 113.6Mb downloads

Submitted by: NCBI (GEO)

Study: Epigenetic landscape of Human Brains by Single Nucleus DNA Methylation and Chromatin Conformation Profiling - pool25_h1930001_Pro
PRJNA746257 • SRP328159 • All experiments • All runs
hide Abstract
Illustrating the cellular architecture of the human brain is critical to understanding its diverse functions and complex animal behaviors. Single nucleus methylation sequencing was applied to regions of human brains. we identified distinct cell clusters and formed them in a hierarchical taxonomy. These clusters include known primary brain cell types and possible sub-types. We use these data to identify the epigenomic characteristics and to define specific regulatory elements for each cell cluster. Overall design: Apply snmC-seq3 to brain regions to identify epigenomic landcape of the adult human brains

Sample: HBA_200826_H1930001_CX53_Pro_1_P3-3-N13-I18
SAMN20196382 • SRS9471129 • All experiments • All runs
Organism: Homo sapiens

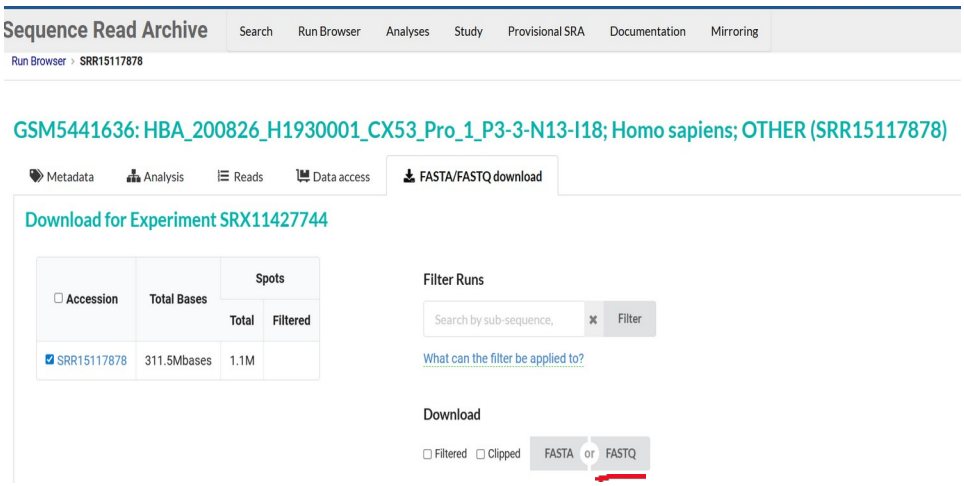
Library:
Instrument: Illumina NovaSeq 6000
Strategy: OTHER
Source: GENOMIC
Selection: other
Layout: PAIRED
Construction protocol: Regions of human brains were dissected. Library prepared following protocol discribed in Luo, Chongyuan, Angeline Rivkin, Jingtian Zhou, Justin P. Sandoval, Laurie Kurihara, Jacinta Lucero, Rosa Castanon, et al. 2018. "Robust Single-Cell DNA Methylome Profiling with snmC-seq2." Nature Communications 9 (1): 3824.

Experiment attributes:
GEO Accession: GSM5441636

Links:
NCBI link: NCBI Entrez (gds)

Runs: 1 run, 1.1M spots, 311.5M bases, 113.6Mb

Run	# of Spots	# of Bases	Size	Published
SRR15117878	1,071,088	311.5M	113.6Mb	2023-07-07



While trying to download the paired-end data manually in fastq.gz format, it showed size as 0 after completion. So, to avoid this issue, SRA Toolkit (can be installed via sudo apt-get install sra-toolkit) was used with the following command:

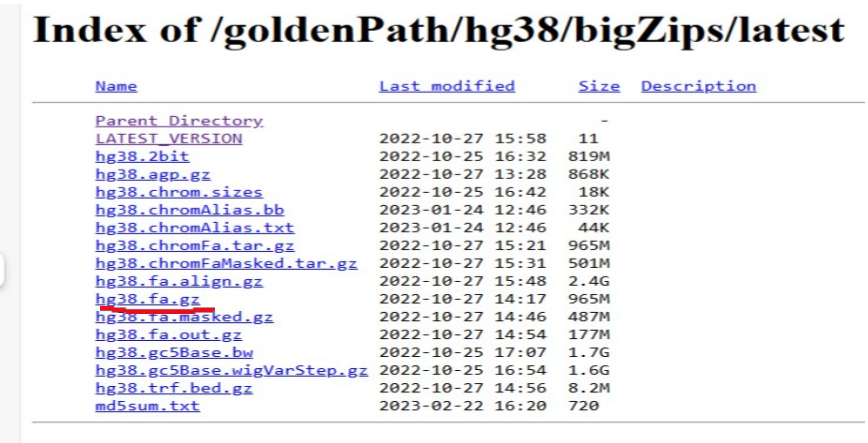
fastq-dump --split-files --gzip SRR15117878

- **fastq-dump:** downloads sequencing reads from NCBI SRA.
- **--split-files:** separates the paired-end reads into two files: _1 (forward) and _2 (reverse).
- **--gzip:** compresses the output into .fastq.gz to save space.

```
ibab@LAPTOP-BVSTVK8Q:~$ cd NGS/
ibab@LAPTOP-BVSTVK8Q:~/NGS$ cd Variant_calling_GATK/
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK$ cd 1_Raw_data/
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/1_Raw_data$ ls -lh
total 151M
-rw-r--r-- 1 ibab ibab 71M Aug 18 02:42 SRR15117878_1.fastq.gz
-rw-r--r-- 1 ibab ibab 80M Aug 18 02:42 SRR15117878_2.fastq.gz
```

b. Download the UCSC human reference genome (hg38) and extract only chromosome 20.

- To download UCSC hg38 latest reference genome, visit – Index of /goldenPath/hg38/bigZips/latest



Index of /goldenPath/hg38/bigZips/latest

Name	Last modified	Size	Description
Parent Directory		-	
LATEST_VERSION	2022-10-27 15:58	11	
hg38.2bit	2022-10-25 16:32	819M	
hg38.agp.gz	2022-10-27 13:28	868K	
hg38.chrom.sizes	2022-10-25 16:42	18K	
hg38.chromAlias.bb	2023-01-24 12:46	332K	
hg38.chromAlias.txt	2023-01-24 12:46	44K	
hg38.chromFa.tar.gz	2022-10-27 15:21	965M	
hg38.chromFaMasked.tar.gz	2022-10-27 15:31	501M	
hg38.fa.align.gz	2022-10-27 15:48	2.4G	
hg38.fa.gz	2022-10-27 14:17	965M	
hg38.fa.masked.gz	2022-10-27 14:46	487M	
hg38.fa.out.gz	2022-10-27 14:54	177M	
hg38.gc5Base.bw	2022-10-25 17:07	1.7G	
hg38.gc5Base.wigVarStep.gz	2022-10-25 16:54	1.6G	
hg38.trf.bed.gz	2022-10-27 14:56	8.2M	
md5sum.txt	2023-02-22 16:20	720	

```
ibab@LAPTOP-BVSTVK8Q:~$ wget https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/latest/hg38.fa.gz
--2025-08-18 05:37:46--  https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/latest/hg38.fa.gz
Resolving hgdownload.soe.ucsc.edu (hgdownload.soe.ucsc.edu)... 128.114.119.163
Connecting to hgdownload.soe.ucsc.edu (hgdownload.soe.ucsc.edu)|128.114.119.163|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1012013082 (965M) [application/x-gzip]
Saving to: 'hg38.fa.gz'

hg38.fa.gz                          100%[=============================================================================>] 965.13M  4.70MB/s    in 7m 21s

2025-08-18 05:45:42 (2.19 MB/s) - 'hg38.fa.gz' saved [1012013082/1012013082]
```

- Or, we can directly download chromosome 20 from - Index of /goldenPath/hg38/chromosomes (using wget)

chr20.fa.gz	2014-01-23 16:39	19M
chr20_GL383577v2_alt.fa.gz	2014-01-23 16:40	42K
chr20_KI270869v1_alt.fa.gz	2014-01-23 16:40	37K
chr20_KI270870v1_alt.fa.gz	2014-01-23 16:40	55K
chr20_KI270871v1_alt.fa.gz	2014-01-23 16:40	18K
chr21.fa.gz	2014-01-23 16:39	12M
chr21_GL383578v2_alt.fa.gz	2014-01-23 16:40	21K
chr21_GL383579v2_alt.fa.gz	2014-01-23 16:40	65K
chr21_GL383580v2_alt.fa.gz	2014-01-23 16:40	25K
chr21_GL383581v2_alt.fa.gz	2014-01-23 16:40	36K
chr21_KI270872v1_alt.fa.gz	2014-01-23 16:40	26K
chr21_KI270873v1_alt.fa.gz	2014-01-23 16:40	47K
chr21_KI270874v1_alt.fa.gz	2014-01-23 16:40	53K
chr22.fa.gz	2014-01-23 16:39	12M
chr22_GL383582v2_alt.fa.gz	2014-01-23 16:40	52K
chr22_GL383583v2_alt.fa.gz	2014-01-23 16:40	31K

- To extract chromosome 2 from hg38, we can use any of the following:

  Option 1:
  samtools faidx hg38.fa.gz chr20 > hg38_chr20.fa

  Option 2:
  zcat hg38.fa.gz | awk '/^>chr20$/,/^>/{if(!/^>/) print $0}' > hg38_chr20.fa

```
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ samtools faidx hg38.fa.gz chr20 > hg38_chr20.fa
[E::fai_build3_core] Cannot index files compressed with gzip, please use bgzip
[faidx] Could not build fai index hg38.fa.gz.fai
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ gunzip  hg38.fa.gz
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ samtools faidx hg38.fa chr20 > hg38_chr20.fa
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ ls -lh
total 3.2G
-rw-r--r-- 1 ibab ibab 3.2G Oct 28  2022 hg38.fa
-rw-r--r-- 1 ibab ibab  31K Aug 18 05:55 hg38.fa.fai
-rw-r--r-- 1 ibab ibab  63M Aug 18 05:55 hg38_chr20.fa
```

- Now, to verify the files –
  1. Check the file sizes.
  2. head -n 20 hg38_chr20.fa (it will show Ns)
  3. wc -c hg38_chr20.fa (should be ~65 million)

```
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ head -n 20 hg38_chr20.fa
>chr20
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ wc -c hg38_chr20.fa
65518244 hg38_chr20.fa
```

4. In order to check whether all are Ns or not, so that we can proceed further:
   grep -v "N" hg38_chr20.fa

```
aagatcagatgattgtagatatgtgatgttatttatgagtcctctgttctgttccattgg
tccatatatctgttttggtagcagcaccatgctgttttggttactgtagccttgtagtat
agtttgaagtcaggtggcataatgcttccatctttgttctttttgcttaggattgtctta
gctatgtgggctctttttttggttccatatgaaatttaaagtagtttttttctaattctgtg
aaaacagtcaatggtagcttgacagggataccactgaatctataaattactttaggcagt
atggccattttcatgatattgattcttcctatccctgagcatggaatgtttttccatttg
tttatgtcctcctttatgtccttgtgcagtggtttgtagttctccttgaagagatccttc
acatcccttgtaagttgtgttctgatctttgacaaacctgacaaaaacaagcaatgggga
aaggattccctatttaataaatggtgttggcaaaactggccagccatatgcagaaaactg
aaactggactccttccttacaccttatacaaaaattaactcaagatggattaaagactta
aatgtaagacctaaaaccacaaaaaccctagaagaaaacctaggcaataccattcaggac
ataggcatgggcaaagacttcatgactaaaacaccaaaagcaatggcaacaaaagcaaaa
ggtgacaaatgagatctaattaaactaaagagcttctgcacagcaaaaaaacaaaacaaa
aacaaaaacaaatactatcatcagagtgaacagacaacctacagaatgggagaaaatttt
tgcaatctatccatctgacaaagggctaatatccagaatctacaaagaacttaaacaaat
ttacaagaaaaaaaaaaccatcaaaaagtgggcgagggatatgaactgacacttctcaaa
agaagacatttacgcagccaacaaacatatgaaaaaaagctcatcgttgtgcacatgtac
cctaaaacttaaagtataataataatttaaaaaaTGGAAAACTGAAAAACAAACAAACAA
ACAAAAAAACAGTCATGCAAACTGAAAGTGTGCAAAGCAATCTTCACACTTGAACTGGTC
TTGAAAGATGCCTATTAGATTGCTGGGTGGGAGGCATTTTAAGCCAAGTGATTGCTCCAA
AAGCATGCATGCATaattgcaattgttctgtgacttcttaaattttatcaagacattaa
aaattctcttcctgttggctaaaaaaaaaaaaagctcgtcatcactggtcattagagaaat
gcaaatcaaaaccacaatgagatgccatctcacaccagttagaatgaccatcattaaaaa
gtcaggaaacaacagatactggagaggatgtggagaaatgggaatgctttttcactgttg
gtgggagtgtaaattacttcaaccattgtagaagacaatgtggcaattcctcaaggatct
agaaccagaaatgtcatttgacccagcaatcccattactgggtatatacccaaaggatta
taaatcattctactataaagacacatgcacatgtatgtttattgtgacactattcacaat
agcaaaaacttggaaccaacccaaatgcccatcaatgatagactggataaagaacatgtg
gcacatatataacatggaatactatgcagccatcaaaaaagggtgagtttgtgtcctttg
cagggacatggatgaagctggaaaccatcattctcagcaattaacacaagaacagagaa
ccaaacacagcatattctcactcataagtggaagttgaagaatgagatcacatgaacaca
gggaggggagcatcacacaccggggcctgtcagggtgtgagggactgatggagagatagc
```

- **<u>Note</u>:** If chr20 reference is **all Ns**, here's what will happen if we proceed:
  - BAM files will mostly contain unmapped reads, with very low or zero mapping quality.
  - Picard may either mark almost all reads as duplicates incorrectly, or not mark anything because it has no coordinates. Hence, statistics will be meaningless.

- Variant callers (GATK HaplotypeCaller, etc.) cannot call variants on Ns, because there's no reference sequence. So, output will be empty or invalid VCFs.

2. Part B: Pre-processing Workflow

Before beginning, run FASTQC (fastqc *.gz) on both the forward & reverse end reads, followed by TrimGalore!, followed by FASTQC on the validated files, and compare the before and after trimming outputs/ reports.

```
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/1_Raw_data$ fastqc *.gz
Started analysis of SRR15117878_1.fastq.gz
Approx 5% complete for SRR15117878_1.fastq.gz
Approx 10% complete for SRR15117878_1.fastq.gz
Approx 15% complete for SRR15117878_1.fastq.gz
Approx 20% complete for SRR15117878_1.fastq.gz
Approx 25% complete for SRR15117878_1.fastq.gz
Approx 30% complete for SRR15117878_1.fastq.gz
Approx 35% complete for SRR15117878_1.fastq.gz
Approx 40% complete for SRR15117878_1.fastq.gz
Approx 45% complete for SRR15117878_1.fastq.gz
Approx 50% complete for SRR15117878_1.fastq.gz
Approx 55% complete for SRR15117878_1.fastq.gz
Approx 60% complete for SRR15117878_1.fastq.gz
Approx 65% complete for SRR15117878_1.fastq.gz
Approx 70% complete for SRR15117878_1.fastq.gz
Approx 75% complete for SRR15117878_1.fastq.gz
Approx 80% complete for SRR15117878_1.fastq.gz
Approx 85% complete for SRR15117878_1.fastq.gz
Approx 90% complete for SRR15117878_1.fastq.gz
Approx 95% complete for SRR15117878_1.fastq.gz
Analysis complete for SRR15117878_1.fastq.gz
Started analysis of SRR15117878_2.fastq.gz
Approx 5% complete for SRR15117878_2.fastq.gz
Approx 10% complete for SRR15117878_2.fastq.gz
Approx 15% complete for SRR15117878_2.fastq.gz
Approx 20% complete for SRR15117878_2.fastq.gz
Approx 25% complete for SRR15117878_2.fastq.gz
Approx 30% complete for SRR15117878_2.fastq.gz
Approx 35% complete for SRR15117878_2.fastq.gz
Approx 40% complete for SRR15117878_2.fastq.gz
```

```
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/1_Raw_data$ trim_galore --paired SRR15117878_1.fastq.gz  SRR15117878_2.fastq.gz -q 30 -stringency 5 --fastqc
Multicore support not enabled. Proceeding with single-core trimming.
Path to Cutadapt set as: 'cutadapt' (default)
Cutadapt seems to be working fine (tested command 'cutadapt --version')
Cutadapt version: 3.5
single-core operation.
No quality encoding type selected. Assuming that the data provided uses Sanger encoded Phred scores (default)



AUTO-DETECTING ADAPTER TYPE
===========================
Attempting to auto-detect adapter type from the first 1 million sequences of the first file (>> SRR15117878_1.fastq.gz <<)

Found perfect matches for the following adapter sequences:
Adapter type    Count   Sequence        Sequences analysed      Percentage
Illumina        804     AGATCGGAAGAGC   1000000 0.08
Nextera 7       CTGTCTCTTATA    1000000 0.00
smallRNA        0       TGGAATTCTCGG    1000000 0.00
Using Illumina adapter for trimming (count: 804). Second best hit was Nextera (count: 7)

Writing report to 'SRR15117878_1.fastq.gz_trimming_report.txt'

SUMMARISING RUN PARAMETERS
==========================
Input filename: SRR15117878_1.fastq.gz
Trimming mode: paired-end
Trim Galore version: 0.6.7
Cutadapt version: 3.5
Number of cores used for trimming: 1
Quality Phred score cutoff: 30
Quality encoding type selected: ASCII+33
```

a. Align the chromosome 20 sequencing reads to the UCSC reference genome using BWA-MEM.

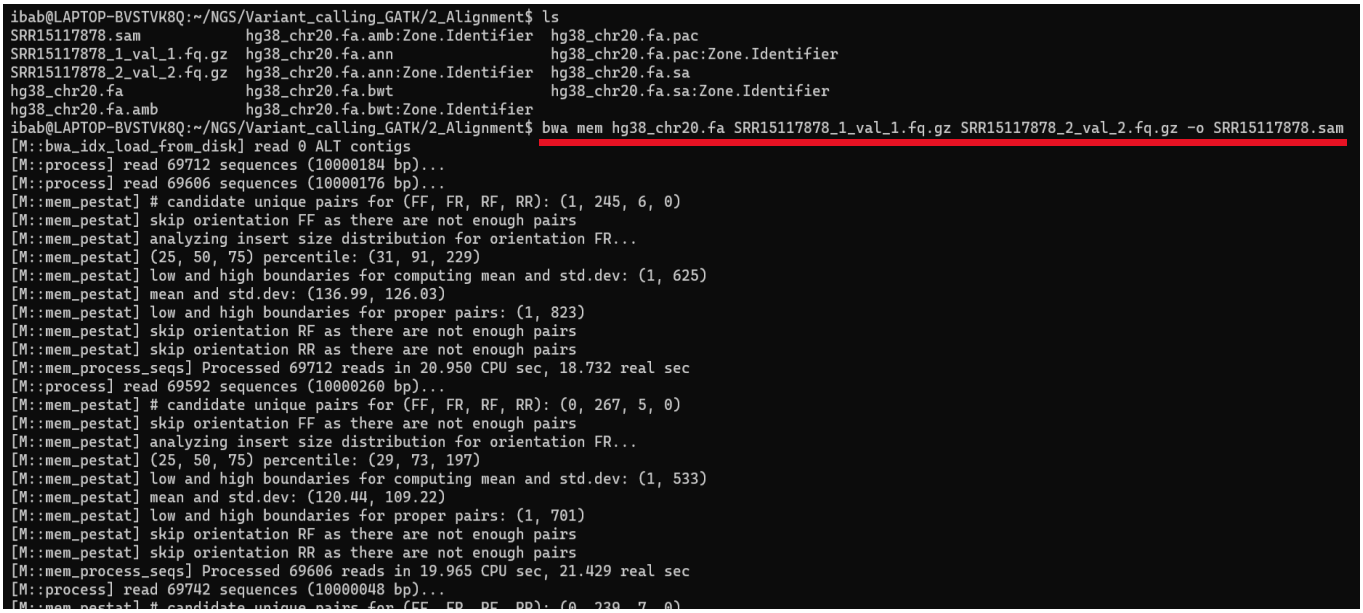
BWA-MEM alignment is a process used to map raw sequencing reads to a reference genome, in this case, chromosome 20 from the UCSC hg38 assembly. Sequencing reads are short fragments of DNA, and they do not contain information about their original location in the genome. BWA-MEM (**Burrows-Wheeler Aligner - Maximal Exact Matches**) efficiently finds the **most likely positions of these reads on the reference genome** by identifying exact matches and extending them to cover the reads. This step is essential because downstream analyses, such as marking duplicates and variant calling, require knowing where each read aligns on the genome. Without alignment, it is impossible to detect mutations, structural variations, or accurately assess coverage, making BWA-MEM a critical first step in any variant calling workflow.

```
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ bwa index hg38_chr20.fa
[bwa_index] Pack FASTA... 0.52 sec
[bwa_index] Construct BWT for the packed sequence...
[BWTIncCreate] textLength=128888334, availableWord=21068624
[BWTIncConstructFromPacked] 10 iterations done. 34753182 characters processed.
[BWTIncConstructFromPacked] 20 iterations done. 64202446 characters processed.
[BWTIncConstructFromPacked] 30 iterations done. 90372990 characters processed.
[BWTIncConstructFromPacked] 40 iterations done. 113629422 characters processed.
[bwt_gen] Finished constructing BWT in 48 iterations.
[bwa_index] 35.95 seconds elapse.
[bwa_index] Update BWT... 0.42 sec
[bwa_index] Pack forward-only FASTA... 0.22 sec
[bwa_index] Construct SA from BWT and Occ... 17.40 sec
[main] Version: 0.7.17-r1188
[main] CMD: bwa index hg38_chr20.fa
[main] Real time: 53.827 sec; CPU: 54.515 sec
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/4_Reference$ ls -lh
total 3.4G
-rw-r--r-- 1 ibab ibab 3.2G Oct 28  2022 hg38.fa
-rw-r--r-- 1 ibab ibab  31K Aug 18 05:55 hg38.fa.fai
-rw-r--r-- 1 ibab ibab  63M Aug 18 05:55 hg38_chr20.fa
-rw-r--r-- 1 ibab ibab 1.4K Aug 18 06:45 hg38_chr20.fa.amb
-rw-r--r-- 1 ibab ibab   43 Aug 18 06:45 hg38_chr20.fa.ann
-rw-r--r-- 1 ibab ibab  62M Aug 18 06:45 hg38_chr20.fa.bwt
-rw-r--r-- 1 ibab ibab  16M Aug 18 06:45 hg38_chr20.fa.pac
-rw-r--r-- 1 ibab ibab  31M Aug 18 06:45 hg38_chr20.fa.sa
```

The raw FASTA file (hg38_chr20.fa) is just a long string of nucleotides (A, C, G, T). If we try to align reads directly, BWA would have to scan the entire chromosome sequence repeatedly for every read, which is extremely slow. Indexing builds a data structure (Burrows-Wheeler Transform + auxiliary tables) that allows BWA-MEM to:

- Locate potential matching regions quickly
- Handle large genomes efficiently
- Perform exact and inexact matching of reads

After indexing, alignment of millions of reads becomes feasible in a reasonable time.



bwa mem hg38_chr20.fa SRR15117878_1_val_1.fq.gz SRR15117878_2_val_2.fq.gz -o SRR15117878.sam

Now, convert the output .sam file into .bam file using the following command:

samtools view -bS SRR15117878.sam -o SRR15117878.bam

Feature	SAM (Sequence Alignment/Map)	BAM (Binary Alignment/Map)
Format	Text (human-readable)	Binary (compressed)
File size	Very large	Much smaller (3–10× smaller)
Readability	Easy to open in text editor	Not human-readable directly
Speed	Slow to parse by tools	Fast to parse and process
Random access	Not possible	Possible (after sorting + indexing)
Pipeline compatibility	Limited	Standard for most downstream tools
Error risk	Easy to accidentally modify	Safer, less prone to corruption

To view the contents of .bam file:

samtools view SRR15117878.bam

```
F:FFFFFFFF:FFFF:FFF:FF:F:FFFFFF:FFF:FFFFFFFFF,FF:F:FFFFFFFFFFFFFF       AS:i:0  XS:i:0
SRR15117878.41171       141     *       0       0       *       *       0       0       GGAGGTTGGAAGTATTATAGAAATTTTTATTAAAGAATTTATTTATGTAATCAAACATCATTTGTTTT
TTAAAATTTTATTGAAATTAAAAAAAAAAAAAAAAATAAGAAGAAGAGGGAGGGGAAGATTTAGGTTATTTCGGGTGAAAGAG     FFFFFFFFFF:FFFFFFFF,FFFFFFFFFFFF:FFFF,FFFF:FFFFF:FFFFFFF:FFFFFFFF:FF
F:FFFFFFFFFFFFFFFFFFFFFFFFFFFFF,FFFF,FF,:,,::FF,FF,FF:,F,:F,F,,:FFFF:F:F,FF,F,FF::F     AS:i:0  XS:i:0
SRR15117878.41172       77      *       0       0       *       *       0       0       TCTTTTACTGACCACACGAACACAACATAAAACAAAAGAAAATTCCTAAATATAACTAAACAAATCCA
ACATAACTAAAAGAATCCAAAATTTACTAAATTTTCTACACCACGTAAAATGAAACCTAAAAATAAAAAAAA        FFFFFFFFFFFFFFFFFF:FFFFFFFFF::FFFFFFFFFFFFFFFFFFFFFFFFFF,FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFF::FFFFFFFF       AS:i:0  XS:i:0
SRR15117878.41172       141     *       0       0       *       *       0       0       GAGACGGGAAATAGAAGAGATTTTGGGAGGTTAGGGATTGTGATTTTAATAGTAGTTATTTTTTAATA
TTTTTTATTTTTTTTTTTTTTATAGGGTTAAGTTTTGGGTGGTTTCAATTATTTTTGATATAATGTTAGTTTTCTTGAGTTTT     FFFF:,FFFF:FFF:FFFFF,,FFFFF:FFF:FFFFF::FFFFFFFFFFFFFF,FFFFFFFFFFFFF:
:::FFFF,:,FFF:F:FF,FF,F:FFFFF:,F:,:,FFF,FFFF:FFFFF:FFF,F::F,FFF,,FFFFFFFF,FFF,,FF,F     AS:i:0  XS:i:0
SRR15117878.41173       117     chr20   47324337        0       *       =       47324337        0       ATTTTTTGATTTTATGATTTATTCGTTTCGGTTTTTTAAAATGTTGGGATTA
TAGGCGTGAGTTATCGTTTTCGGTTTTAAAGTAAGTTTTATTGTTTGTTGGTTAAGAAAGTTTTTAGTTAGGTAAGGATTTAGTATAGTAG     FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FF,FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF     MC:Z:28S36M87S  AS:i:0  XS:i:0
SRR15117878.41173       185     chr20   47324337        0       28S36M87S       =       47324337        0       AAAATCGAAACCATCCTAACGAACACGATAAAACCCAATCTCTA
CTAAAAAATACAAAAAATTAACCGAACGTAATAACGAACTTCTATAATCCCAACTACTCAAAAAACTAAAACAAAAAAATAACGTAAACAAAAAAACGAATTCCCTC     FFFFFFFFF,FF:FFFFFFFFFFFFFFFFFFFFFFFFFF,FFFFFF
FFFFF:FFFFFFFF:F:FFFFFFFF:FFFFF,FFFFFFFFFFFFF:FFF::FFFF,FFFFFF:F:FFFF:FFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFF     NM:i:1  MD:Z:8C27       AS:i:31 XS:i:30
SRR15117878.41174       99      chr20   40732681        58      55S22M66S       =       40732681        22      CCAAACCATAAACTTGAAACACTGGGATACCAAATGAAAAAATC
TAATATCCTTTAAATAAAAATAAACAAAAACCCCTAAACTTAAATAATAATAACAAATATTACAAAAAAAACATCAACAAAATTTAGATCTCTCTTCTC     FFFF:FFFFF::FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF:FFFFFFFFFFFF:FFFFFF:FFFFFFFFFFFFFFFFFFF:FF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF     NM:i:0  MD:Z:22 MC:Z:58S22M71S  AS:i:22 XS:i:0
SRR15117878.41174       147     chr20   40732681        58      58S22M71S       =       40732681        -22     GCTCCAAACCATAAACTTGAAACCCTGGGATACCAAATGAAAAA
ATCTAATATCCTTTAAATAAAAATAAACAAAAACCCCTAAACTTAAATAATAATAACAAATATTACAAAAAAAACATCAACAAAATTTAGATCTCTCTTCTCCTCCC     FFFFFF,FFFFFFF:FFFFFFFF,FFFFFFFFFFFFFFFFFFFF
FFFFFFF:FFFFFFFFF:FFFFFFFFFFFF:,FFFFFFFFFFFFFFF:FFF:FFFFFFFFFFF,FFFFF,FFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFF     NM:i:0  MD:Z:22 MC:Z:55S22M66S  AS:i:22 XS:i
:0
```

QNAME	Query Name (read ID)	This is the identifier of the read, usually coming from the FASTQ file. For paired-end reads, both mates share the same root ID (e.g., SRR15117878.41173/1 and SRR15117878.41173/2 or simply SRR15117878.41173). It lets you track which sequencing read each line represents.
FLAG	Bitwise flag	An integer code where each bit has a meaning. It encodes multiple properties at once (paired, mapped/unmapped, read1/read2, reverse strand, etc.). For example, 117 = 64 (first read in pair) + 32 (read reverse strand) + 16 (mate reverse strand) + 4 (read mapped) + 1 (paired). Tools like samtools flagstat or samtools flags help decode these numbers.
RNAME	Reference sequence name	The reference (chromosome, scaffold, contig) where the read is aligned. This corresponds to the sequence names in your reference FASTA (hg38_chr20.fa → chr20). If the read is unmapped, this is *.
POS	Leftmost position	The 1-based coordinate of the first aligned base of the read on the reference sequence. For example, if POS = 47324337, the read starts at nucleotide 47,324,337 of chromosome 20. If unmapped, this is 0.
MAPQ	Mapping quality	A Phred-scaled probability that the read is incorrectly mapped. MAPQ = -10 * log10(p_wrong). Example: MAPQ 30 ≈ 1 in 1000 chance of wrong mapping. A MAPQ of 0 means either low confidence or multiple equally good alignments.
CIGAR	Alignment description	Compact string describing how the read aligns to the reference. Each number+letter describes operations: M = match/mismatch, I = insertion, D = deletion, S = soft-clipped bases (kept in SEQ), H = hard-clipped (discarded), N = skipped region (splicing/gaps), = = exact match, X = mismatch. Example: 28S36M87S means 28 bases clipped, 36 aligned, 87 clipped again.
RNEXT	Mate reference name	For paired-end reads, this shows where the mate read is mapped. = means the mate maps to the same chromosome as RNAME. * if the mate is

QNAME	Query Name (read ID)	This is the identifier of the read, usually coming from the FASTQ file. For paired-end reads, both mates share the same root ID (e.g., SRR15117878.41173/1 and SRR15117878.41173/2 or simply SRR15117878.41173). It lets you track which sequencing read each line represents.
		unmapped or not available.
PNEXT	Mate position	The leftmost position of the mate read's alignment on RNEXT. This helps define insert size. If mate is unmapped, this is 0.
TLEN	Template length (insert size)	The signed distance between the outermost mapped bases of read and its mate. Positive if mate is downstream, negative if upstream. For example, TLEN = 500 means the two reads span 500 bases. If only one read is mapped, TLEN is 0.
SEQ	Read sequence	The nucleotide sequence of the read (as in FASTQ). * means no sequence is stored (sometimes done to save space if SEQ not needed).
QUAL	Base qualities	Encoded in ASCII, one character per base in SEQ, representing the Phred-scaled probability of an incorrect base call. For example, F often = Q37. The higher the symbol, the more reliable the base. A string of all F means high-quality across the read.

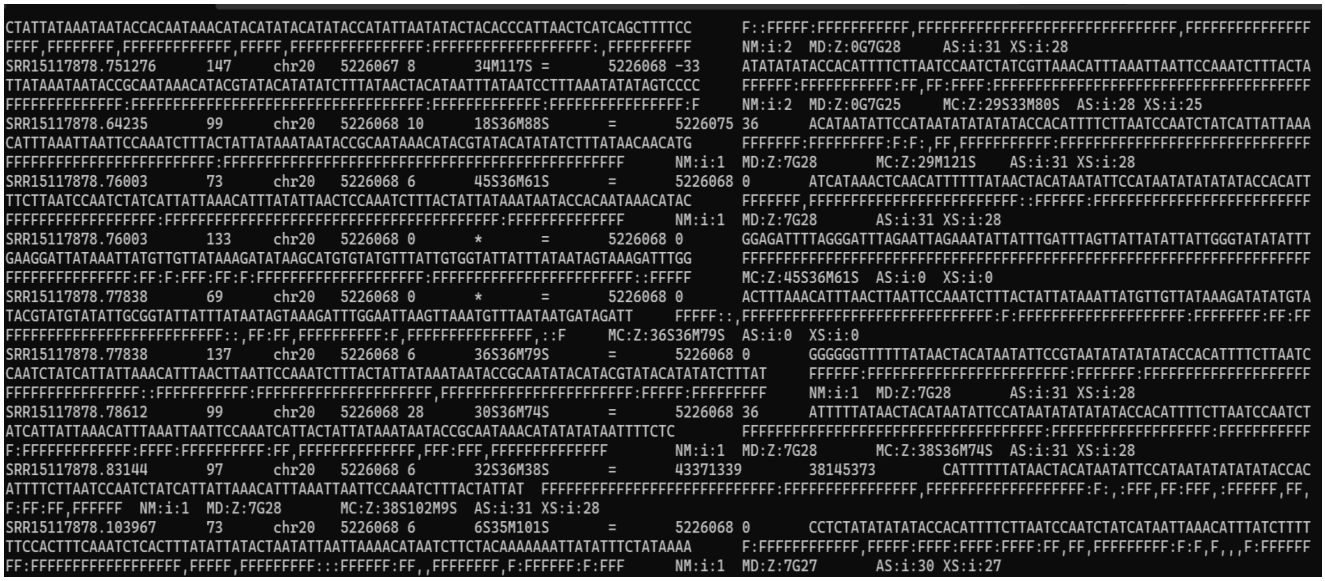
Optional Fields (TAG:TYPE:VALUE)

AS:i	Alignment Score — how well the read aligned (higher = better).
BC:Z	Barcode sequence (for demultiplexing, if present).
BQ:Z	Quality values of the barcode sequence.
CC:Z	Reference name of the next hit (when chimeric).
CM:i	Number of "color mismatches" (color-space only).
CP:i	Leftmost coordinate of next hit (for chimeric reads).
CQ:Z	Base qualities of color-space sequence.
CS:Z	Color-space read sequence.
CT:Z	Type of next hit (for chimeric alignment).
E2:Z	Sequence of the next-best alignment.
FI:i	Fragment index in a chimeric alignment.
FS:i	Fragment span (length of template).

AS:i	Alignment Score — how well the read aligned (higher = better).
H0:i	Number of perfect hits (zero mismatches).
H1:i	Number of 1-difference hits.
H2:i	Number of 2-difference hits.
HI:i	Hit index for reads with multiple mappings.
IH:i	Total number of reported alignments for the read.
MC:Z	Mate's CIGAR string.
MD:Z	Mismatch string: encodes positions of mismatches vs reference.
MQ:i	Mapping quality of the mate.
NH:i	Number of reported alignments for the query (multi-mapping count).
NM:i	Edit distance: # of differences (mismatches + indels).
OQ:Z	Original base quality scores (before recalibration).
PG:Z	Program record identifier (which software generated this alignment).
PQ:i	Phred probability of the template being correct.
PU:Z	Platform unit (flowcell-barcode.lane).
Q2:Z	Base qualities of the mate/second read in the pair.
R2:Z	Sequence of the mate/second read.
RG:Z	Read group tag (links read to metadata in header).
SA:Z	Supplementary alignment info (for split alignments).
SM:i	Mapping quality of the best hit.
TC:i	Number of fragments in the template.
UQ:i	Phred likelihood of the read being mapped incorrectly.
XS:i	Suboptimal alignment score (used in spliced aligners like TopHat/STAR).
YT:Z	Read type (UU, CP, etc. — used in old pipelines).

To sort the .bam file:

```
samtools sort -o SRR15117878_sorted.bam SRR15117878.bam
```



Why BAM files must be sorted?

1. **Organizes reads by genomic position**
    - Raw BAM files are in the order that BWA wrote them (basically input read order).
    - Many downstream tools (variant callers, visualization software, etc.) require reads to be arranged **by coordinate along the reference genome**.
2. **Prepares for indexing**
    - An **index (.bai)** can only be built from a **coordinate-sorted BAM**.
    - The index allows **random access** to a specific chromosome region without scanning the entire file (crucial for large genomes).
3. **Enables efficient data retrieval**
    - With a sorted BAM, tools like **samtools view**, **IGV (Integrative Genomics Viewer)**, or **bcftools** can quickly jump to, for example, chr20:47,324,337-47,325,000 and fetch only those reads.
4. **Required by downstream analyses**
    - Variant callers (GATK, FreeBayes), assemblers, and other pipelines **expect sorted BAMs**.
    - Without sorting, they will throw errors or give incorrect results.

To check the alignment stats, we can use any of the following commands:

samtools flagstat SRR15117878_sorted.bam , or

bamtools stats -in SRR15117878_sorted.bam -insert

```
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/2_Alignment$ samtools flagstat SRR15117878_sorted.bam
2132519 + 0 in total (QC-passed reads + QC-failed reads)
2124392 + 0 primary
0 + 0 secondary
8127 + 0 supplementary
0 + 0 duplicates
0 + 0 primary duplicates
610784 + 0 mapped (28.64% : N/A)
602657 + 0 primary mapped (28.37% : N/A)
2124392 + 0 paired in sequencing
1062196 + 0 read1
1062196 + 0 read2
523534 + 0 properly paired (24.64% : N/A)
549580 + 0 with itself and mate mapped
53077 + 0 singletons (2.50% : N/A)
0 + 0 with mate mapped to a different chr
0 + 0 with mate mapped to a different chr (mapQ>=5)
ibab@LAPTOP-BVSTVK8Q:~/NGS/Variant_calling_GATK/2_Alignment$ bamtools stats -in SRR15117878_sorted.bam -insert

*****************************************
Stats for BAM file(s):
*****************************************

Total reads:       2132519
Mapped reads:      610784       (28.6414%)
Forward strand:    1800252      (84.419%)
Reverse strand:    332267       (15.581%)
Failed QC:         0    (0%)
Duplicates:        0    (0%)
Paired-end reads:  2132519      (100%)
'Proper-pairs':    524894       (24.6138%)
Both pairs mapped: 554607       (26.0071%)
Read 1:            1066476
Read 2:            1066043
Singletons:        56177        (2.6343%)
Average insert size (absolute value): 948814
Median insert size (absolute value): 23
```

Parameter	Values	Meaning	Interpretation
Total reads	21,325,19	Total number of reads (including paired reads).	This is your sequencing depth; all reads generated are counted here.
Mapped reads	610,784 (28.64%)	Reads that aligned to the reference genome.	Only ~29% mapped — indicates many reads did not align, likely because only chromosome 20 (hg38 chr20) was used as reference, not the whole genome.
Forward strand	1,800,252 (84.4%)	Reads aligning to the forward (+) strand.	A large proportion map to forward strand — normal depending on sequencing library prep.
Reverse strand	332,267 (15.6%)	Reads aligning to the reverse (−) strand.	Lower than forward strand — imbalance could reflect library orientation bias.
Failed QC	0	Reads filtered due to sequencing quality issues.	None failed QC → sequencing quality was high.
Duplicates	0	PCR or optical duplicates flagged during alignment.	No duplicates marked → dataset either deduplicated or duplication rate is very low.

Parameter	Values	Meaning	Interpretation
Paired-end reads	21,325,19 (100%)	Reads sequenced in pairs.	Confirms data is paired-end sequencing.
'Proper pairs'	524,894 (24.6%)	Both mates aligned with correct orientation & insert size.	Only ~25% are properly paired, which is low, suggesting incomplete mapping due to limited reference.
Both pairs mapped	554,607 (26.0%)	Read pairs where both mates aligned.	~26% mapped in pairs; again consistent with partial reference alignment.
Read 1	10,661,476	First read of each pair.	Count is ~half of total reads, as expected for paired-end sequencing.
Read 2	10,660,403	Second read of each pair.	Same as Read 1, confirming balanced paired data.
Singletons	56,177 (2.63%)	Pairs where only one mate aligned.	Small fraction, normal in sequencing, though here partly due to reference restriction.
Average insert size	948,814 bp	Average distance between paired reads mapped.	Extremely high → skewed by discordant mappings when only chr20 was used.
Median insert size	23 bp	Median distance between paired reads.	More realistic than average; most read pairs are actually close, but a few huge outliers distort the average.

b. Mark duplicates using Picard MarkDuplicates.

MarkDuplicates is a preprocessing step used to **identify and flag duplicate reads in sequencing data**. During sequencing, the same DNA fragment may be sequenced multiple times, creating duplicates that can bias downstream analyses like variant calling. Picard's MarkDuplicates tool examines the aligned reads (BAM file) and marks those that are likely duplicates **based on their mapping positions and orientation**. These marked duplicates are ignored or treated differently by variant callers **to prevent overestimating coverage or calling false-positive variants**. This ensures that the analysis reflects the true biological signal rather than technical artifacts.

```
picard MarkDuplicates I=SRR15117878_sorted.bam O=SRR15117878_markdup.bam M=marked_dup_metrics.txt
```

*M = Creates a file listing the number of duplicates for both single-end and paired-end reads.

```
(base) ibab@IBAB-MScBDB2-Comp007:~/NGS_Lab$ picard MarkDuplicates I=SRR15117878_sorted.bam O=SRR15117878_markdup.bam M=marked_dup_metrics.txt
INFO    2025-08-18 14:10:16     MarkDuplicates

********** NOTE: Picard's command line syntax is changing.
**********
********** For more information, please see:
********** https://github.com/broadinstitute/picard/wiki/Command-Line-Syntax-Transition-For-Users-(Pre-Transition)
**********
********** The command line looks like this in the new syntax:
**********
**********    MarkDuplicates -I SRR15117878_sorted.bam -O SRR15117878_markdup.bam -M marked_dup_metrics.txt
**********


14:10:16.916 INFO  NativeLibraryLoader - Loading libgkl_compression.so from jar:file:/home/ibab/miniconda3/share/picard-2.20.4-0/picard.jar!/com/intel
/gkl/native/libgkl_compression.so
[Mon Aug 18 14:10:16 IST 2025] MarkDuplicates INPUT=[SRR15117878_sorted.bam] OUTPUT=SRR15117878_markdup.bam METRICS_FILE=marked_dup_metrics.txt    MAX
_SEQUENCES_FOR_DISK_READ_ENDS_MAP=50000 MAX_FILE_HANDLES_FOR_READ_ENDS_MAP=8000 SORTING_COLLECTION_SIZE_RATIO=0.25 TAG_DUPLICATE_SET_MEMBERS=false REM
OVE_SEQUENCING_DUPLICATES=false TAGGING_POLICY=DontTag CLEAR_DT=true DUPLEX_UMI=false ADD_PG_TAG_TO_READS=true REMOVE_DUPLICATES=false ASSUME_SORTED=f
alse DUPLICATE_SCORING_STRATEGY=SUM_OF_BASE_QUALITIES PROGRAM_RECORD_ID=MarkDuplicates PROGRAM_GROUP_NAME=MarkDuplicates READ_NAME_REGEX=<optimized ca
pture of last three ':' separated fields as numeric values> OPTICAL_DUPLICATE_PIXEL_DISTANCE=100 MAX_OPTICAL_DUPLICATE_SET_SIZE=300000 VERBOSITY=INFO
QUIET=false VALIDATION_STRINGENCY=STRICT COMPRESSION_LEVEL=5 MAX_RECORDS_IN_RAM=500000 CREATE_INDEX=false CREATE_MD5_FILE=false GA4GH_CLIENT_SECRETS=c
lient_secrets.json USE_JDK_DEFLATER=false USE_JDK_INFLATER=false
[Mon Aug 18 14:10:16 IST 2025] Executing as ibab@IBAB-MScBDB2-Comp007.ibab.ac.in on Linux 6.14.0-27-generic amd64; OpenJDK 64-Bit Server VM 11.0.1+13-
LTS; Deflater: Intel; Inflater: Intel; Provider GCS is not available; Picard version: 2.20.4-SNAPSHOT
INFO    2025-08-18 14:10:16     MarkDuplicates  Start of doWork freeMemory: 530777104; totalMemory: 536870912; maxMemory: 1073741824
INFO    2025-08-18 14:10:16     MarkDuplicates  Reading input file and constructing read end information.
INFO    2025-08-18 14:10:16     MarkDuplicates  Will retain up to 3890368 data points before spilling to disk.
WARNING 2025-08-18 14:10:17     AbstractOpticalDuplicateFinderCommandLineProgram        A field field parsed out of a read name was expected to contai
n an integer and did not. Read name: SRR15117878.644743. Cause: String 'SRR15117878.644743' did not start with a parsable number.
```

```
(base) ibab@IBAB-MScBDB2-Comp007:~/NGS_Lab$ cat marked_dup_metrics.txt
## htsjdk.samtools.metrics.StringHeader
# MarkDuplicates INPUT=[SRR15117878_sorted.bam] OUTPUT=SRR15117878_markdup.bam METRICS_FILE=marked_dup_metrics.txt    MAX_SEQUENCES_FOR_DISK_READ_ENDS
_MAP=50000 MAX_FILE_HANDLES_FOR_READ_ENDS_MAP=8000 SORTING_COLLECTION_SIZE_RATIO=0.25 TAG_DUPLICATE_SET_MEMBERS=false REMOVE_SEQUENCING_DUPLICATES=fal
se TAGGING_POLICY=DontTag CLEAR_DT=true DUPLEX_UMI=false ADD_PG_TAG_TO_READS=true REMOVE_DUPLICATES=false ASSUME_SORTED=false DUPLICATE_SCORING_STRATE
GY=SUM_OF_BASE_QUALITIES PROGRAM_RECORD_ID=MarkDuplicates PROGRAM_GROUP_NAME=MarkDuplicates READ_NAME_REGEX=<optimized capture of last three ':' separ
ated fields as numeric values> OPTICAL_DUPLICATE_PIXEL_DISTANCE=100 MAX_OPTICAL_DUPLICATE_SET_SIZE=300000 VERBOSITY=INFO QUIET=false VALIDATION_STRING
ENCY=STRICT COMPRESSION_LEVEL=5 MAX_RECORDS_IN_RAM=500000 CREATE_INDEX=false CREATE_MD5_FILE=false GA4GH_CLIENT_SECRETS=client_secrets.json USE_JDK_DE
FLATER=false USE_JDK_INFLATER=false
## htsjdk.samtools.metrics.StringHeader
# Started on: Mon Aug 18 14:10:16 IST 2025

## METRICS CLASS        picard.sam.DuplicationMetrics
LIBRARY UNPAIRED_READS_EXAMINED READ_PAIRS_EXAMINED     SECONDARY_OR_SUPPLEMENTARY_RDS  UNMAPPED_READS  UNPAIRED_READ_DUPLICATES        READ_PAIR_DUPL
ICATES  READ_PAIR_OPTICAL_DUPLICATES    PERCENT_DUPLICATION     ESTIMATED_LIBRARY_SIZE
Unknown Library 53077   274790  8127    1521735 17525   33743   0       0.14106 1025286

## HISTOGRAM    java.lang.Double
BIN     CoverageMult    all_sets        non_optical_sets
1.0     1       212039  212039
2.0     1.764898        24914   24914
3.0     2.349966        3558    3558
4.0     2.797484        453     453
5.0     3.139789        65      65
6.0     3.401618        14      14
7.0     3.60189 4       4
8.0     3.755078        0       0
9.0     3.872251        0       0
10.0    3.961876        0       0
11.0    4.03043 0       0
12.0    4.082867        0       0
13.0    4.122976        0       0
14.0    4.153655        0       0
15.0    4.177122        0       0
```

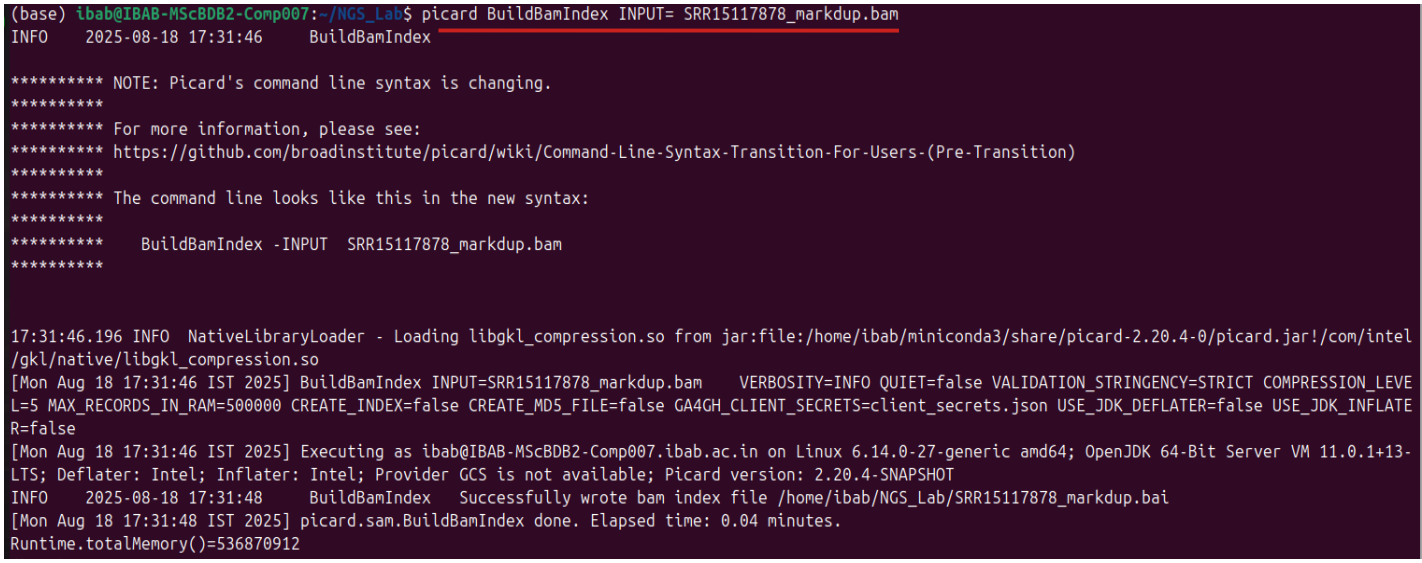
```
A:Z:chr20,-18039462,37S21M42S,0;chr20,-56870809,37S21M42S,0;    MC:Z:90S22M30S  MD:Z:22 PG:Z:MarkDuplicates     NM:i:0  AS:i:22 XS:i:21
SRR15117878.86587       99      chr20   6919647 9       104S20M18S      =       6919647 20      AATACAAAATTCTACCTAAAAAAAATAAAAATATATTCAAATATAAATTTTTAT
AAAATAACTATTAACTAAAAAAAAAAATAACCTTTCTTTCCCCAAACATCAATAAATACAAAATACAAAATAATATTTACAACATACA        FF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFF,FFFFFF:FFF        XA:Z:chr20,+32641157,107S19M16S,0;      MC:Z:6S20M125S
MD:Z:20 PG:Z:MarkDuplicates     NM:i:0  AS:i:20 XS:i:19
SRR15117878.86587       147     chr20   6919647 9       6S20M125S       =       6919647 -20     AACATCAATAAATACAAAATACAAAATAATATTTACAACATACAACATCATATT
TAATTCTAATTAATAAACTATTTCAATAAAATATTATTTTAATAAATATATTCATACTTCATTTATCATGTATTAAAAACAACAATACACGCCTCTC       FFFFFF::FFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFF:FFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFF       XA:Z:chr20,-32641157,9S19M123S,0;    M
C:Z:104S20M18S  MD:Z:20 PG:Z:MarkDuplicates     NM:i:0  AS:i:20 XS:i:19
SRR15117878.39968       147     chr20   6919817 36      19M97S  =       6919631 -205    AAAAAAAAAAAAAAAAAAATTAATATCTCTCCACATACATACACTAAAAAAAAACCATATAA
ACACACAACTAAAAAACAACCATCTACCAACCAAAAAAAAAAACCCTCAATCCT  FFF:FFF:FFFFF,F:F:F,:F,F:F,FF:FF:FF,FFFFFFFF:::FF::F:F,:FF,F:F,FFF,:,FFFF:FF:FFF,FF,F::,FFFFFF
FFFFF,FFFFFFFFFFFFFFF  XA:Z:chr20,-45326389,25M91S,0;chr20,-524422,25M91S,0;chr20,+32346924,91S25M,0;chr20,+34287381,91S25M,0;chr20,-54355258,2S24M90
S,0;    MC:Z:98S26M18S  MD:Z:19 PG:Z:MarkDuplicates     NM:i:0  AS:i:19 XS:i:25
SRR15117878.227007      99      chr20   6920090 60      22S20M1I61M10S  =       6920183 213     CACTAAACAAAAATGTTCTTCACATTAATCTACCAACAACCTATCCTCCTTCCT
```

Interpretation:

The duplication analysis of the BAM file shows a **percent duplication of ~14%**, which is within an **acceptable range** (commonly <20% is considered good for most NGS datasets). This indicates that the majority of the reads are **unique** and not redundant duplicates. Importantly, the **optical duplicate count is zero**, suggesting that there were no significant sequencing artifacts caused by the imaging system, which reflects good sequencing quality. The **estimated library size (~1.02 million unique molecules)** is reasonable and suggests that the library has good complexity, meaning sufficient diversity of fragments was captured and sequenced.

picard BuildBamIndex INPUT= SRR15117878_markdup.bam

Index the marked-duplicate BAM file using Picard Tools to enable rapid access to alignments during downstream analyses.



c. Add or replace read groups using Picard AddOrReplaceReadGroups.

AddOrReplaceReadGroups is a step that **assigns metadata to sequencing reads, called read groups**, which describe the sample, library, platform, and sequencing run. Tools like GATK require this information to distinguish reads coming from different experiments or samples and to perform accurate variant calling. Picard's AddOrReplaceReadGroups adds or updates these read groups in the BAM file, making it compatible with downstream pipelines. Including read groups ensures proper handling of multi-sample datasets, accurate duplicate marking, and correct interpretation of sequencing data in variant analysis.

picard AddOrReplaceReadGroups I=SRR15117878_markdup.bam O=SRR15117878_grpadded.bam RGID=4 RGLB=lib1 RGPL=illumina RGPU=unit1 RGSM=sample_name RGCN=bi

This step helps to track the origin of the reads (specific sample, library or sequencing run to differentiate samples for example RGID, RGLB etc).
**RGID identifier of read group, RGLB library used, RGPL sequencing platform, RGPU platform unit like flowcell or lane, RGSM sample name.

```
(base) ibab@IBAB-MScBDB2-Comp007:~/NGS_Lab$ picard AddOrReplaceReadGroups I=SRR15117878_markdup.bam O=SRR15117878_grpadded.bam RGID=4 RGLB=lib1 RGPL=i
llumina RGPU=unit1 RGSM=sample_name RGCN=bi
INFO    2025-08-18 17:34:44     AddOrReplaceReadGroups

********** NOTE: Picard's command line syntax is changing.
**********
********** For more information, please see:
********** https://github.com/broadinstitute/picard/wiki/Command-Line-Syntax-Transition-For-Users-(Pre-Transition)
**********
********** The command line looks like this in the new syntax:
**********
**********    AddOrReplaceReadGroups -I SRR15117878_markdup.bam -O SRR15117878_grpadded.bam -RGID 4 -RGLB lib1 -RGPL illumina -RGPU unit1 -RGSM sample
_name -RGCN bi
**********


17:34:44.408 INFO  NativeLibraryLoader - Loading libgkl_compression.so from jar:file:/home/ibab/miniconda3/share/picard-2.20.4-0/picard.jar!/com/intel
/gkl/native/libgkl_compression.so
[Mon Aug 18 17:34:44 IST 2025] AddOrReplaceReadGroups INPUT=SRR15117878_markdup.bam OUTPUT=SRR15117878_grpadded.bam RGID=4 RGLB=lib1 RGPL=illumina RGP
U=unit1 RGSM=sample_name RGCN=bi    VERBOSITY=INFO QUIET=false VALIDATION_STRINGENCY=STRICT COMPRESSION_LEVEL=5 MAX_RECORDS_IN_RAM=500000 CREATE_INDEX
=false CREATE_MD5_FILE=false GA4GH_CLIENT_SECRETS=client_secrets.json USE_JDK_DEFLATER=false USE_JDK_INFLATER=false
[Mon Aug 18 17:34:44 IST 2025] Executing as ibab@IBAB-MScBDB2-Comp007.ibab.ac.in on Linux 6.14.0-27-generic amd64; OpenJDK 64-Bit Server VM 11.0.1+13-
LTS; Deflater: Intel; Inflater: Intel; Provider GCS is not available; Picard version: 2.20.4-SNAPSHOT
INFO    2025-08-18 17:34:44     AddOrReplaceReadGroups  Created read-group ID=4 PL=illumina LB=lib1 SM=sample_name

INFO    2025-08-18 17:34:54     AddOrReplaceReadGroups  Processed     1,000,000 records.  Elapsed time: 00:00:09s.  Time for last 1,000,000:    9s.  L
ast read position: */*
INFO    2025-08-18 17:35:02     AddOrReplaceReadGroups  Processed     2,000,000 records.  Elapsed time: 00:00:18s.  Time for last 1,000,000:    8s.  L
ast read position: */*
[Mon Aug 18 17:35:03 IST 2025] picard.sam.AddOrReplaceReadGroups done. Elapsed time: 0.32 minutes.
Runtime.totalMemory()=536870912
```

```
1       PG:Z:MarkDuplicates     RG:Z:4  NM:i:0  AS:i:31 XS:i:29
SRR15117878.93315       121     chr20   5814655 0       109S33M =       5814655 0       TAGGGTTTTTTTTTGTGGTAGGTTTTAGTGCAGTGGTACGTTTTTTGATTATTTTAATTTTG
ATTTTTTGGGTTTTTTTTGGTTTGTTTAAAGGATGATTTTATTTTTGTTTTTTTTTTTTTTTTTTTTTTTTGAGATAGGGT        F,FF,::FF:FF,F,,FFF,FFFF,F,,F,,FF,,:F,,,,FF:F,:,FF,F:,FF,FFF,,
,:FFFFF:FF:FFFFFF,FFFF,,FF:,F:F,,:FFFFFF,F:,FFF,,FFFFFFFFFFFFFFFFFFF:FFFFFFFF:        MD:Z:33 PG:Z:MarkDuplicates     RG:Z:4  NM:i:0  AS:i:33 XS:i:3
3
SRR15117878.93315       181     chr20   5814655 0       *       =       5814655 0       AAAAGAGATAAATCACTAATATAATAAATAAAATTTAAAATACTAAAAAAATTAATTTACTT
ATACCAAAAAATTAAAAAAGTACCTATAAAACATTATTAAAATATAATTTCTACAAAAATATATAAAAATTATCCAAATATTTATCTCC       FF:,:,,FFF,,F,F:,,FFFFFFF,,:,,FF,FFFFFFF,F:,,F,FFFF,FF
F,FFFFFFF,,F,,FFFFFFF,FFFF,,,F,,FF,F,FF,FF,FFFFFF,F,,FFF,FFFFFFFF,F,FFF,FFFFFFFF,:,FFFFFF:F::,F:,       MC:Z:109S33M    PG:Z:MarkDuplicates     RG:Z:4
AS:i:0  XS:i:0
SRR15117878.204461      81      chr20   5814655 0       42S33M49S       =       15470040        9655354 TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTCGCAAATTTTTTATTTT
TTTTTTTTTTTTTTTTTTTGAGATAGGGTTTCATTTTGTTATTCAGGTTGGAGTGTGGTGGTACGATTATGTTTATTG  F:FFFFF,FFFFFFF:FFF:FF:FFFF:,,F,FFF,,,FF:,:FF:FFFFFFFFFFFFFF:FFFFFFFF
:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFF  SA:Z:chr20,62251868,-,30M94S,0,0;       MC:Z:79S30M42S  MD:Z:33 PG:Z:MarkDuplicates     RG:Z:4
NM:i:0  AS:i:33 XS:i:33
```

picard BuildBamIndex INPUT= SRR15117878_grpadded.bam
Index the marked-duplicate BAM file using Picard Tools to enable rapid access to alignments during downstream analyses.

```
(base) ibab@IBAB-MScBDB2-Comp007:~/NGS_Lab$ picard BuildBamIndex INPUT= SRR15117878_grpadded.bam
INFO    2025-08-18 17:39:33     BuildBamIndex

********** NOTE: Picard's command line syntax is changing.
**********
********** For more information, please see:
********** https://github.com/broadinstitute/picard/wiki/Command-Line-Syntax-Transition-For-Users-(Pre-Transition)
**********
********** The command line looks like this in the new syntax:
**********
**********    BuildBamIndex -INPUT  SRR15117878_grpadded.bam
**********


17:39:33.816 INFO  NativeLibraryLoader - Loading libgkl_compression.so from jar:file:/home/ibab/miniconda3/share/picard-2.20.4-0/picard.jar!/com/intel
/gkl/native/libgkl_compression.so
[Mon Aug 18 17:39:33 IST 2025] BuildBamIndex INPUT=SRR15117878_grpadded.bam    VERBOSITY=INFO QUIET=false VALIDATION_STRINGENCY=STRICT COMPRESSION_LEV
EL=5 MAX_RECORDS_IN_RAM=500000 CREATE_INDEX=false CREATE_MD5_FILE=false GA4GH_CLIENT_SECRETS=client_secrets.json USE_JDK_DEFLATER=false USE_JDK_INFLAT
ER=false
[Mon Aug 18 17:39:33 IST 2025] Executing as ibab@IBAB-MScBDB2-Comp007.ibab.ac.in on Linux 6.14.0-27-generic amd64; OpenJDK 64-Bit Server VM 11.0.1+13-
LTS; Deflater: Intel; Inflater: Intel; Provider GCS is not available; Picard version: 2.20.4-SNAPSHOT
INFO    2025-08-18 17:39:36     BuildBamIndex   Successfully wrote bam index file /home/ibab/NGS_Lab/SRR15117878_grpadded.bai
[Mon Aug 18 17:39:36 IST 2025] picard.sam.BuildBamIndex done. Elapsed time: 0.05 minutes.
Runtime.totalMemory()=536870912
```