Read Mapping and Variant Calling:

1. Introduction:

The process of aligning the read on a reference genome is called read mapping. This is performed for various reasons to understand a set of read alignments to the reference in detail. The reasons may include studying the mismatches, insertions, deletions, location of SNPs and protein binding sites, quality of transcripts, and clipping some fragments on the ends of the reads. We downloaded 12 tomato varietal chloroplasts in fastq (gz) format and a reference chloroplast reference fasta sequence for this lab. We performed read mapping and variant calling for the current lab using various tools. That includes bowtie2 for read mapping, samtools for conversion of SAM files into BAM files and indexing the BAM files, getting vcf files, and a genome browser like IGV for loading the BAM files to check with the reference genome.

The files are named as follows:

- SRR1763770.fastq.gz
- SRR1763771.fastq.gz
- SRR1763772.fastq.gz
- SRR1763773.fastq.gz
- SRR1763774.fastq.gz
- SRR1763775.fastq.gz
- SRR1763776.fastq.gz
- SRR1763777.fastq.gz
- SRR1763778.fastq.gz
- SRR1763779.fastq.gz
- SRR1763780.fastq.gz
- SRR1763781.fastq.gz

2. Methodology:

2.1 Trimming the files:

We use trimmomatic here to trim the files. This is ideally done to remove the Ion Torrent adapter from these reads.

Command:

```
#!/bin/bash
#SBATCH --job-name=trim_readmap
#SBATCH --partition=Centaurus
#SBATCH --time=05:00:00

module load trimmomatic

java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763770.fastq.gz SRR1763770.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763771.fastq.gz SRR1763771.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763772.fastq.gz SRR1763772.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763773.fastq.gz SRR1763773.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763774.fastq.gz SRR1763774.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763775.fastq.gz SRR1763775.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763776.fastq.gz SRR1763776.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763777.fastq.gz SRR1763777.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763778.fastq.gz SRR1763778.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763779.fastq.gz SRR1763779.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763780.fastq.gz SRR1763780.trimmed.fastq
TRAILING:20 MINLEN:50
java -jar /$TRIM/trimmomatic-0.39.jar SE SRR1763781.fastq.gz SRR1763781.trimmed.fastq
TRAILING:20 MINLEN:50
```

2.2 Indexing:

Bowtie is a short-read aligner. It aligns DNA reads. Bowtie2 indexes the genome with a Burrows-Wheeler index. For the assignment, we performed read mapping and indexing using bowtie2.

Code:
For building the bowtie2 index from the reference genome.

- [dmehta12@gal-i1 lab_readmapping]$ bowtie2-build NC_007898.3.fasta NC_007898.3

This produces 6 files as follows:

- NC_007898.3.1.bt2
- NC_007898.3.2.bt2
- NC_007898.3.3.bt2
- NC_007898.3.4.bt2
- NC_007898.3.rev.1.bt2
- NC_007898.3.rev.2.bt2

2.3 Mapping:

Before mapping the files, we rename the files based on the more informative filename (library name). For example, changing the name of 'SRR1763770' to 'BC29'. Similarly, we also rename the trimmed files. This could have been the first step too to avoid changing all the names later on.

Command: (similar for all of the files)

- [dmehta12@gal-i1 lab_readmapping]$mv SRR1763770.fastq.gz BC29.fastq.gz
- [dmehta12@gal-i1 lab_readmapping]$mv SRR1763770.trimmed.fastq. BC29.trimmed.fastq

To run bowtie2, we need to load the module of bowtie2 for producing the output SAM files. We need to load the **bowtie2 module** for this. The version it has is 2.4.1.

Command:
```
#!/bin/bash
#SBATCH --job-name=sam_output
#SBATCH --partition=Centaurus
```

```
#SBATCH --time=01:00:00

module load bowtie2

bowtie2 -x NC_007898.3 -U BC17.trimmed.fastq -S BC17.sam
bowtie2 -x NC_007898.3 -U BC19.trimmed.fastq -S BC19.sam
bowtie2 -x NC_007898.3 -U BC20.trimmed.fastq -S BC20.sam
bowtie2 -x NC_007898.3 -U BC21.trimmed.fastq -S BC21.sam
bowtie2 -x NC_007898.3 -U BC22.trimmed.fastq -S BC22.sam
bowtie2 -x NC_007898.3 -U BC23.trimmed.fastq -S BC23.sam
bowtie2 -x NC_007898.3 -U BC24.trimmed.fastq -S BC24.sam
bowtie2 -x NC_007898.3 -U BC25.trimmed.fastq -S BC25.sam
bowtie2 -x NC_007898.3 -U BC26.trimmed.fastq -S BC26.sam
bowtie2 -x NC_007898.3 -U BC28.trimmed.fastq -S BC28.sam
bowtie2 -x NC_007898.3 -U BC29.trimmed.fastq -S BC29.sam
bowtie2 -x NC_007898.3 -U BC30.trimmed.fastq -S BC30.sam
```

Here, we used the base index as the 'NC_007898.3' which we built in the previous step of indexing. -U here means that these are the unpaired reads given as the input sequences (trimmed files). -S here indicates the output in a SAM format.

2.4 Conversion of SAM files into BAM format and sorting:

The SAM files are usually very large files. Converting those files to BAM format helps us save space and help programs that operate on the BAM files work easily. Sorting the files is needed as many browsers require the BAM files to be sorted to view them. This has to be done in a way where reads are ordered from left to right across the reference genome. BAM meaning Binary Alignment Map, is a compressed binary file sorting the read sequences whether they have been aligned to the reference sequence and if so, the position on the reference at which they have been aligned. For this, we need to load the samtools module.

Command:
```
#!/bin/bash
#SBATCH --job-name=myjob
#SBATCH --partition=Centaurus
#SBATCH --time=00:10:00
```

```
module load samtools

samtools view -uS BC17.sam | samtools sort - -o BC17-srt.bam
samtools view -uS BC19.sam | samtools sort - -o BC19-srt.bam
samtools view -uS BC20.sam | samtools sort - -o BC20-srt.bam
samtools view -uS BC21.sam | samtools sort - -o BC21-srt.bam
samtools view -uS BC22.sam | samtools sort - -o BC22-srt.bam
samtools view -uS BC23.sam | samtools sort - -o BC23-srt.bam
samtools view -uS BC24.sam | samtools sort - -o BC24-srt.bam
samtools view -uS BC25.sam | samtools sort - -o BC25-srt.bam
samtools view -uS BC26.sam | samtools sort - -o BC26-srt.bam
samtools view -uS BC28.sam | samtools sort - -o BC28-srt.bam
samtools view -uS BC29.sam | samtools sort - -o BC29-srt.bam
samtools view -uS BC30.sam | samtools sort - -o BC30-srt.bam
```

2.5 BAM indexing:

We created a BAM index using samtools.

Command:

```
#!/bin/bash
#SBATCH --job-name=bam_index
#SBATCH --partition=Centaurus
#SBATCH --time=00:10:00

module load samtools

samtools index  BC17-srt.bam
samtools index  BC19-srt.bam
samtools index  BC20-srt.bam
samtools index  BC21-srt.bam
samtools index  BC22-srt.bam
samtools index  BC23-srt.bam
samtools index  BC24-srt.bam
samtools index  BC25-srt.bam
samtools index  BC26-srt.bam
samtools index  BC28-srt.bam
samtools index  BC29-srt.bam
samtools index  BC30-srt.bam
```

This file is usually opened in the genome browser like IGV to compare against the reference genome.

2.6 VCF file production:

To get the pileup and vcf files from the sorted BAM files, the following code was used:

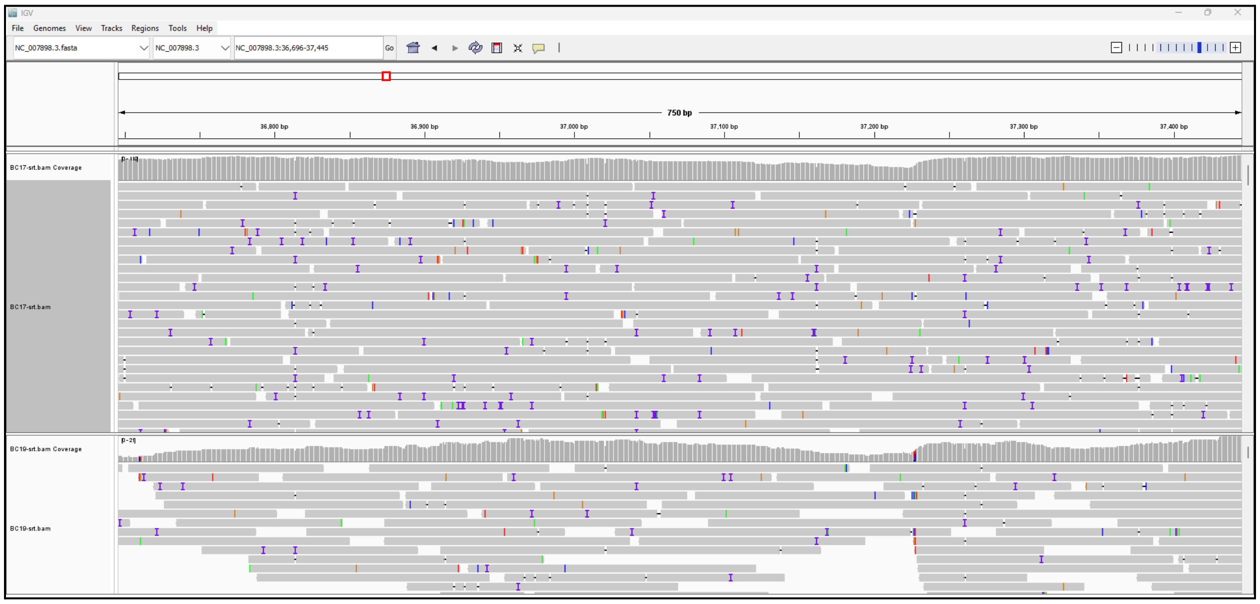
Command:

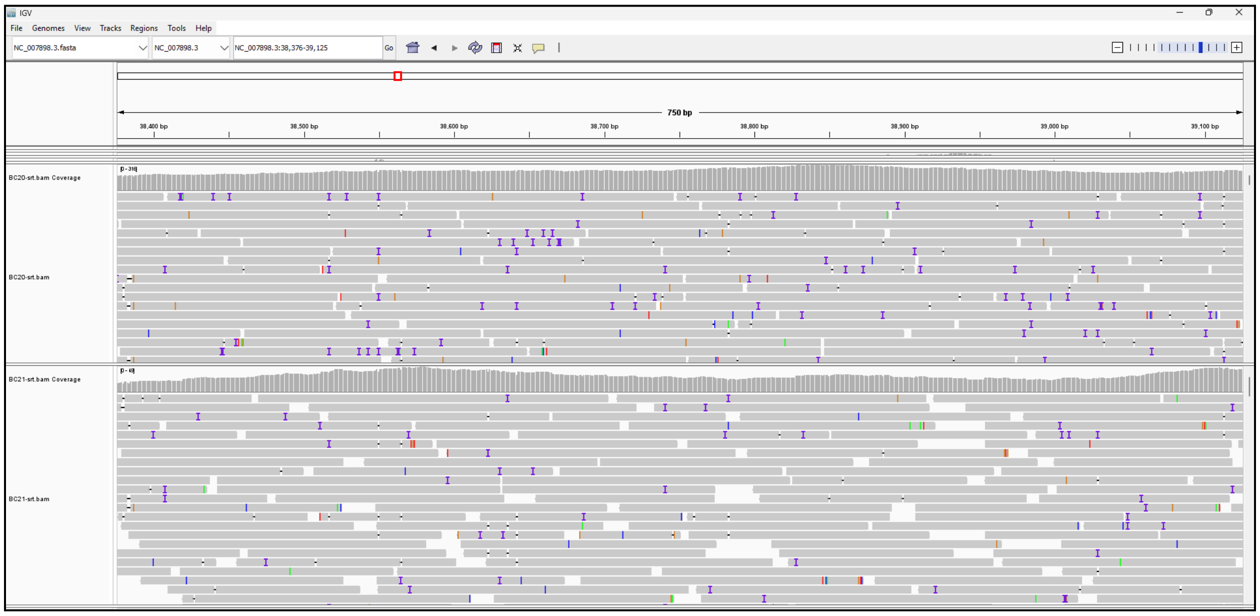
```
#!/bin/bash
#SBATCH --job-name=bam_to_vcf
#SBATCH --partition=Centaurus
#SBATCH --time=05:00:00

bcftools mpileup -f NC_007898.3.fasta BC17-srt.bam | bcftools call -c -v > BC17.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC19-srt.bam | bcftools call -c -v > BC19.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC20-srt.bam | bcftools call -c -v > BC20.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC21-srt.bam | bcftools call -c -v > BC21.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC22-srt.bam | bcftools call -c -v > BC22.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC23-srt.bam | bcftools call -c -v > BC23.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC24-srt.bam | bcftools call -c -v > BC24.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC25-srt.bam | bcftools call -c -v > BC25.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC26-srt.bam | bcftools call -c -v > BC26.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC28-srt.bam | bcftools call -c -v > BC28.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC29-srt.bam | bcftools call -c -v > BC29.srt.vcf
bcftools mpileup -f NC_007898.3.fasta BC30-srt.bam | bcftools call -c -v > BC30.srt.vcf
```

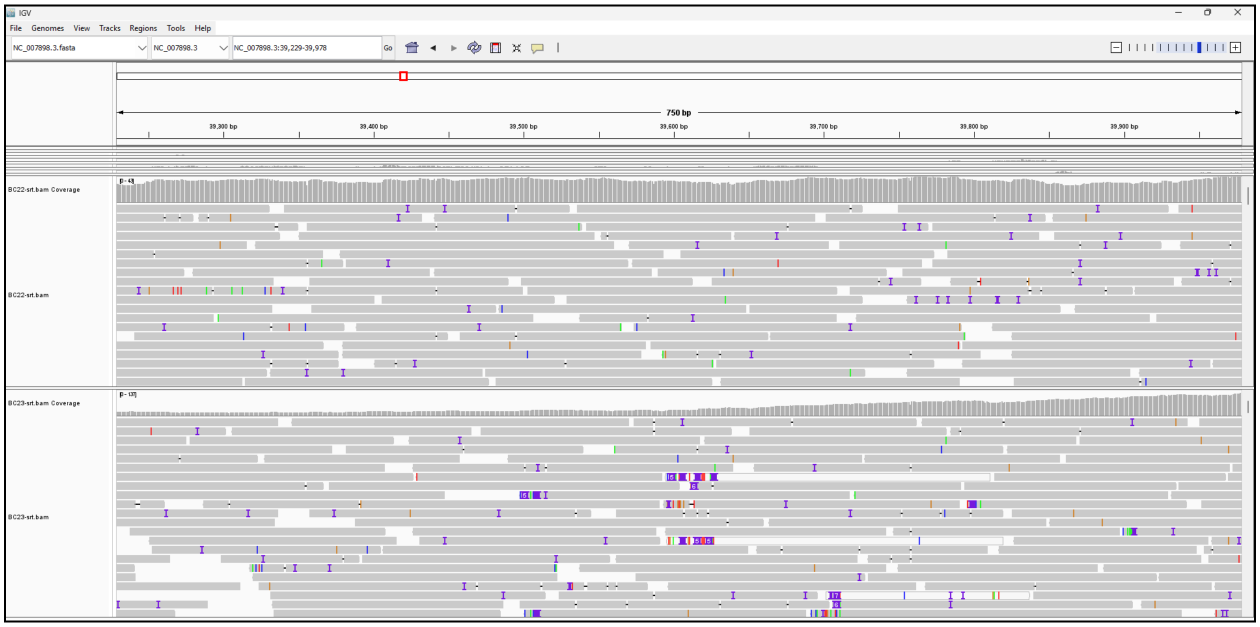
These files have to be loaded in the IGV genome browser. We perform this step for variant calling to get information about SNPs and indels compared to the reference.

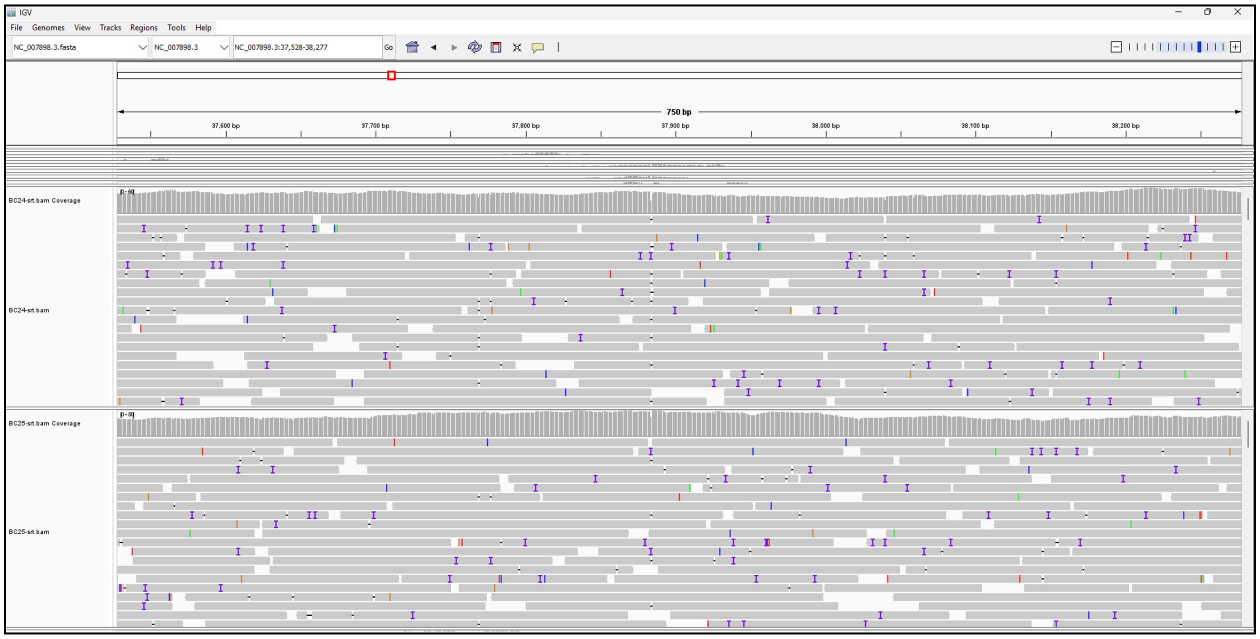
3. Results:

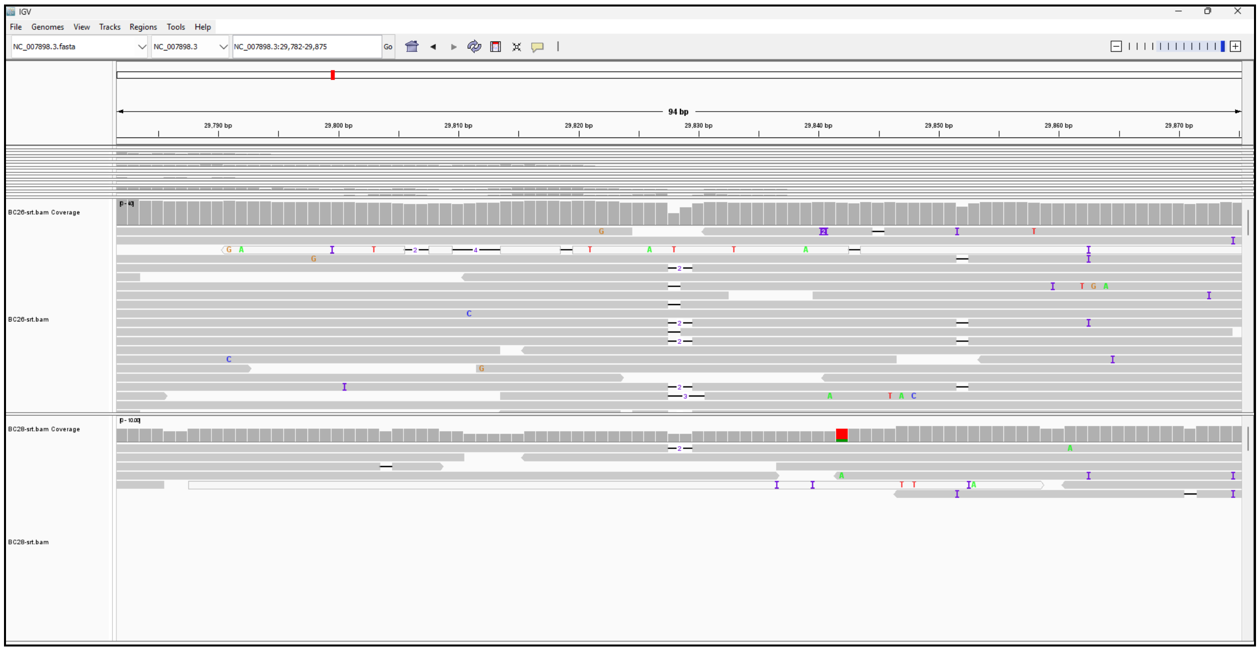
3.1 IGV browser: BAM files against the reference genome of the chloroplast.

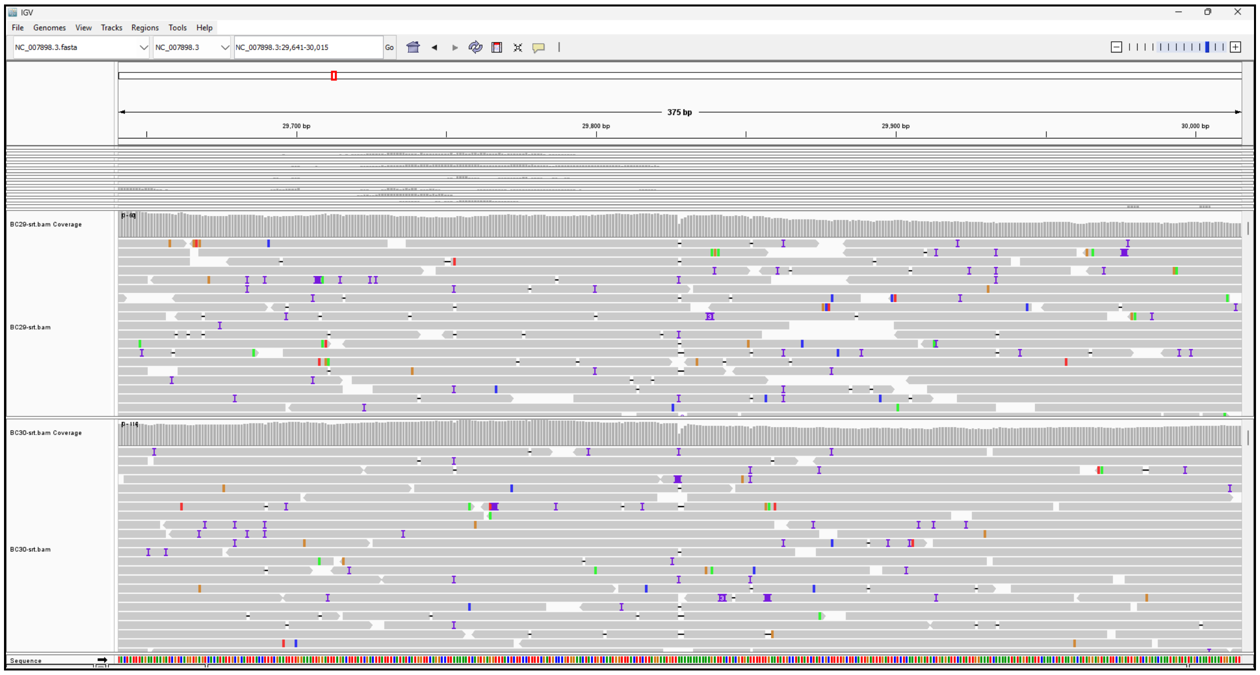




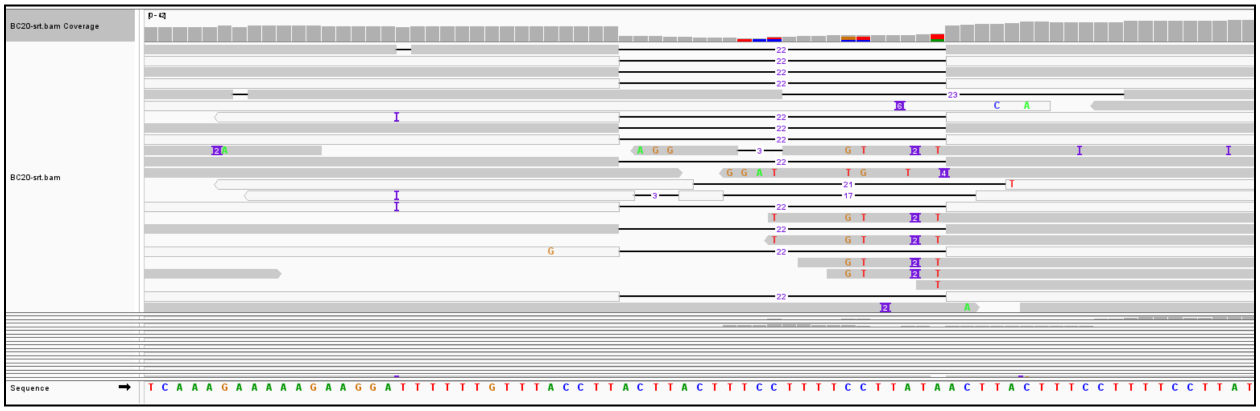

IGV
File Genomes View Tracks Regions Tools Help
NC_007898.3.fasta
NC_007898.3
NC_007898.3:39,229-39,978
Go
750 bp
39,300 bp
39,400 bp
39,500 bp
39,600 bp
39,700 bp
39,800 bp
39,900 bp
BC22-srt.bam Coverage
BC22-srt.bam
BC23-srt.bam Coverage
BC23-srt.bam


IGV
File Genomes View Tracks Regions Tools Help
NC_007898.3.fasta
NC_007898.3
NC_007898.3:37,528-38,277
Go
750 bp
37,600 bp
37,700 bp
37,800 bp
37,900 bp
38,000 bp
38,100 bp
38,200 bp
BC24-srt.bam Coverage
BC24-srt.bam
BC25-srt.bam Coverage
BC25-srt.bam

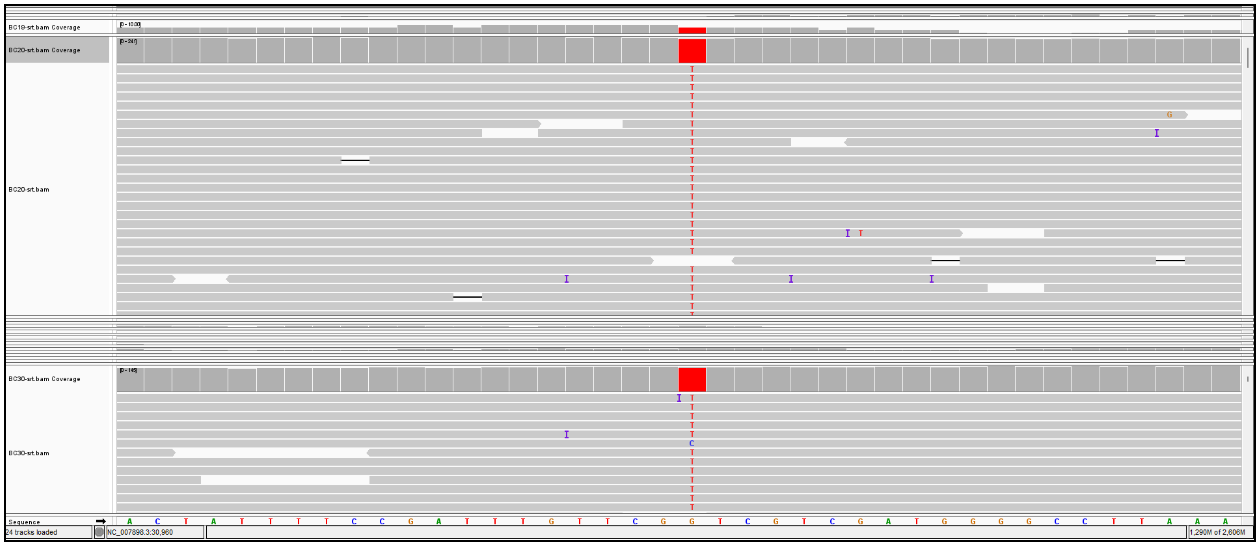


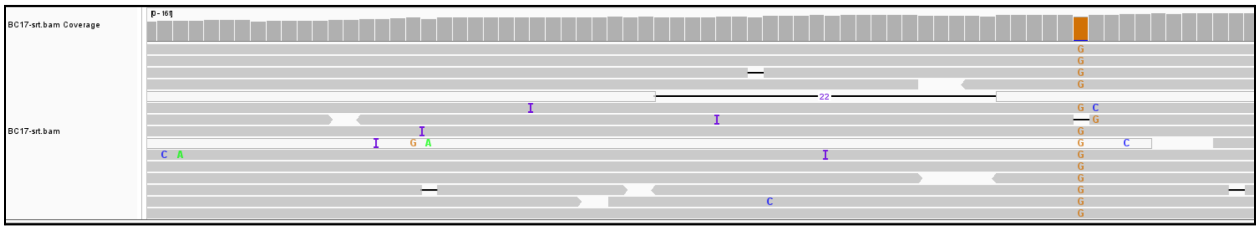


3.2 Heterogeneity evidence:

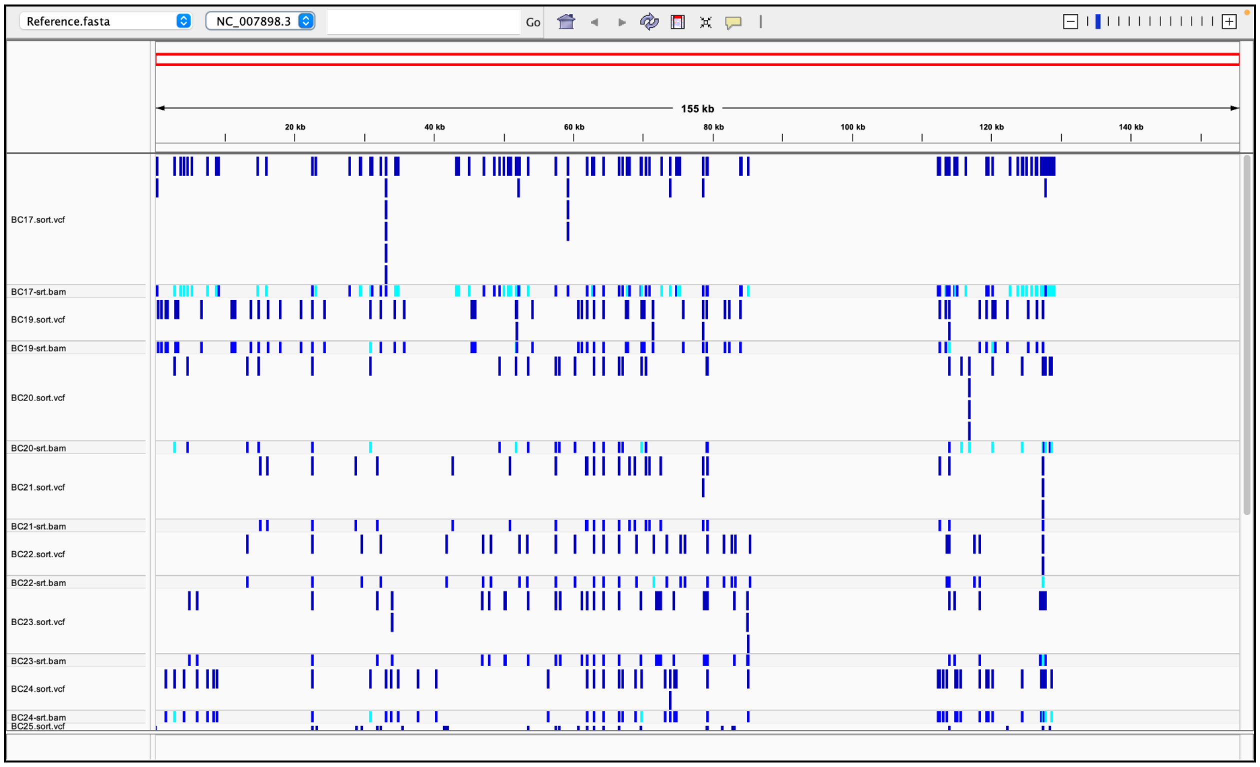

BC20-srt.bam Coverage
BC20-srt.bam
Sequence
T C A A A G A A A A A G A A G G A T T T T T T G T T T A C C T T A C T T A C T T T C C T T T T C C T T A T A A C T T A C T T T C C T T T T C C T T A T

3.3 Mismatches:





3.4 VCF files in the IGV: (1 example)



3.5 Number of variants:

grep -v -c ‘^#’ BC17.srt.vcf

grep -v -c ‘^#’ BC19.srt.vcf

grep -v -c ‘^#’ BC20.srt.vcf

grep -v -c ‘^#’ BC21.srt.vcf

grep -v -c ‘^#’ BC22.srt.vcf

grep -v -c ‘^#’ BC23.srt.vcf

grep -v -c ‘^#’ BC24.srt.vcf

grep -v -c ‘^#’ BC25.srt.vcf

grep -v -c ‘^#’ BC26.srt.vcf

grep -v -c ‘^#’ BC28.srt.vcf

grep -v -c ‘^#’ BC29.srt.vcf

grep -v -c ‘^#’ BC30.srt.vcf

3.6 Number of SNPs:

Bcftools view -v snps BC17.srt.vcf | grep -v -c ‘^#’

Bcftools view -v snps BC19.srt.vcf | grep -v -c ‘^#’

Bcftools view -v snps BC20.srt.vcf | grep -v -c ‘^#’

Bcftools view -v snps BC21.srt.vcf | grep -v -c ‘^#’

Bcftools view -v snps BC22.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC23.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC24.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC25.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC26.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC28.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC29.srt.vcf | grep -v -c ‘^#’
Bcftools view -v snps BC30.srt.vcf | grep -v -c ‘^#’

3.7 Number of indels:

Bcftools view -v indels BC17.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC19.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC20.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC21.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC22.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC23.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC24.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC25.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC26.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC28.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC29.srt.vcf | grep -v -c ‘^#’
Bcftools view -v indels BC30.srt.vcf | grep -v -c ‘^#’

The numerical information:

Cultivar	**# of Variants**	**# of SNPs**	**# of Indels**
BC17	116	69	47
BC19	56	20	36
BC20	32	14	18
BC21	44	7	37
BC22	40	4	36
BC23	33	1	32
BC24	51	20	31
BC25	34	2	32
BC26	29	1	28
BC28	57	17	40
BC29	26	4	22
BC30	113	66	4

4. Discussion:

This lab mentions how important it is to know read mapping in genomics as it is one of the most common practices which we perform for a lot of findings. The information we got at the end of this lab is about SNPs and indels, the heterogeneity in the chloroplast, genomic divergence, the quality of the base and the transcripts, etc.

The individual sequences (BAM files) compared to the reference genome in the IGV browser are shown in section 3.1 of the result. These sequences are shown at different locations in the genome to get an overview of the variations.

There are so many SNPs and indels, primarily SNPs. Checking the VCF files in section 3.4 of the Result, we can see the SNPs compared to the reference sequence.
In section 3.2 of Results, we can see that there are multiple variants at a location giving us evidence of heterogeneity.

And section 3.3 of the Results mentions the mismatches. BC20 shows the maximum number of indels and SNPs. The heterogeneous variants are more prevalent in again BC20. We see so many sites where all the sequences have either indels or SNPs, showing us a similar variation at the same site.

We can see the mapping quality of the reads based on the shade of the read, if it is grey in color, it usually has good quality but if it is hollow, it has a lower quality score. Similarly, base quality can be known from the quality of the bases. The coverage shows an overview of that particular site. If usually there are more variants at a particular site (>28%) then the color of the coverage changes too. The height of the coverage shows the number of reads, the higher the coverage, the more reads.