Assignment-1
Indraprastha Institute of Information Technology Delhi (IIIT-Delhi)
DSG - DATA SCIENCE IN GENOMICS

PARAS DHIMAN
2021482

a. **Create a soft link for the FASTQ file:**

```
ln -s /storage/vibhor/RNA-seq/SRR16292072.fastq.gz SRR16292072.fastq.gz
```

Align the FASTQ file to the hg19 genome using Tophat

```
tophat2 -o tophat_out /storage/vibhor/genomes/hg19/hg19
SRR16292072.fastq.gz
```

Align the FASTQ file to the hg19 genome using STAR:

```
STAR --genomeDir /storage/vibhor/genomes/hg19/starIndex/hg19
--outFileNamePrefix STAR_out --runThreadN 8 --readFilesIn
/storage/DSG/paras21482/SRR16292072.fastq.gz --readFilesCommand zcat
```

b. **Run Cufflinks to get FPKM-based expression of genes:**

```
cufflinks -p 8 -G /storage/vibhor/genomes/hg19/hg19-Gencode.gtf -o
cufflinks_out_tophat tophat_out/accepted_hits.bam

cufflinks -p 8 -G /storage/vibhor/genomes/hg19/hg19-Gencode.gtf -o
cufflinks_out_star STAR_outAligned.sortedByCoord.out.bam
```

c. **There was an error of numpy here then corrected**

```
htseq-count -f bam -r pos -s no -i gene_id tophat_out/accepted_hits.bam
/storage/vibhor/genomes/hg19/hg19-Gencode.gtf > read_counts_tophat.txt

htseq-count -f bam -r pos -s no -i gene_id
STAR_outAligned.sortedByCoord.out.bam
/storage/vibhor/genomes/hg19/hg19-Gencode.gtf > read_counts_star.txt
```

d. **Python code**

```
cut -f 10 cufflinks_out_tophat/genes.fpkm_tracking > fpkm_values_tophat.txt
cut -f 10 cufflinks_out_star/genes.fpkm_tracking > fpkm_values_star.txt
```

```
# For Tophat
fpkm_values = []
with open('fpkm_values_tophat.txt', 'r') as file:
    next(file)  # Skip the header line
    for line in file:
        fpkm_values.append(float(line.strip()))

# Define histogram bins with increased number of bins
num_bins = 10
bin_width = (max(fpkm_values) - min(fpkm_values)) / num_bins
bins = [min(fpkm_values) + i * bin_width for i in range(num_bins + 1)]

# Count occurrences of FPKM values within each bin
hist_counts = [0] * num_bins
for fpkm in fpkm_values:
    bin_index = int((fpkm - min(fpkm_values)) // bin_width)
    if 0 <= bin_index < num_bins:
        hist_counts[bin_index] += 1

# Print the text-based histogram
print("Distribution of FPKM values:")
for i in range(num_bins):
    print("{:.2f} - {:.2f}: {}".format(bins[i], bins[i+1], "#" *
hist_counts[i]))
```

For STAR:

```
# Star
fpkm_values = []
with open('fpkm_values_star.txt', 'r') as file:
    next(file)  # Skip the header line
    for line in file:
        fpkm_values.append(float(line.strip()))

# Define histogram bins with increased number of bins
num_bins = 10
bin_width = (max(fpkm_values) - min(fpkm_values)) / num_bins
```

```
bins = [min(fpkm_values) + i * bin_width for i in range(num_bins + 1)]

# Count occurrences of FPKM values within each bin
hist_counts = [0] * num_bins
for fpkm in fpkm_values:
    bin_index = int((fpkm - min(fpkm_values)) // bin_width)
    if 0 <= bin_index < num_bins:
        hist_counts[bin_index] += 1

# Print the text-based histogram
print("Distribution of FPKM values:")
for i in range(num_bins):
    print("{:.2f} - {:.2f}: {}".format(bins[i], bins[i+1], "#" *
hist_counts[i]))
```

e. **Discover novel transcripts and promoters using Cufflinks:**
For STAR:

```
grep -v -F -f /storage/vibhor/genomes/hg19/hg19-Gencode.gtf
cufflinks_out_tophat/transcripts.gtf > novel_transcripts.gtf
```

Promoter:

```
awk '$3 == "transcript" { if ($7 == "+") print $1, $4-500, $4; else if ($7
== "-") print $1, $5, $5+500 }' cufflinks_out_tophat/transcripts.gtf >
promoters.gtf
```

For Tophat:

```
grep -v -F -f /storage/vibhor/genomes/hg19/hg19-Gencode.gtf
cufflinks_out_star/transcripts.gtf > novel_transcripts_star.gtf
```

Promoter:

```
awk '$3 == "transcript" { if ($7 == "+") print $1, $4-500, $4; else if ($7
== "-") print $1, $5, $5+500 }' cufflinks_out_star/transcripts.gtf >
promoters_star.gtf
```

```
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ samtools view -c tophat_out/accepted_hits.bam
29826753
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ samtools view -c STAR_outAligned.sortedByCoord.out.bam
29608178
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ samtools view -c -F 4 tophat_out/accepted_hits.bam
29826753
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ samtools view -c -F 4 STAR_outAligned.sortedByCoord.out.bam
29608178
```

Alignment Statistics:

Tophat:
Total number of reads: 29,826,753
Number of reads with alignment: 29,826,753
STAR:
Total number of reads: 29,608,178
Number of reads with alignment: 29,608,178
Both Tophat and STAR aligner achieved a 100% alignment rate, with the number of reads with alignment matching the total number of reads for each tool. This indicates that all reads were successfully aligned to the reference genome without any unmapped reads.

Speed:

STAR aligner generally exhibited faster alignment compared to Tophat. However, the specific speed may vary depending on several factors, including:
Genome size and complexity: Larger and more complex genomes may require more computational resources and time for alignment.
Available computational resources: The speed of alignment can be influenced by the computational resources (e.g., CPU, memory) allocated to the process.
Alignment parameters: The choice of alignment parameters may affect the alignment speed.
Alignment Percentage:

Both STAR and Tophat achieved a 100% alignment rate, indicating that all reads were successfully aligned to the reference genome. This high alignment percentage demonstrates the effectiveness of both tools in accurately aligning RNA-seq reads.
In summary, while STAR aligner demonstrated faster alignment speed compared to Tophat, both tools exhibited high alignment accuracy with a complete alignment of all reads. The choice between STAR and Tophat may depend on various factors such as the specific requirements of the analysis, computational resources, and user preferences. Further optimization and parameter tuning may also influence the performance of each tool in different scenarios.

All the files that were generated are as follows
In paras21482:

```
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ ls
SRR16292072.fastq.gz                STAR_outLog.final.out     STAR_outSJ.out.tab  cufflinks_out_star    fpkm_values_tophat.txt  tophat_out
STAR_outAligned.sorted.bam          STAR_outLog.out           STAR_out_STARtmp    cufflinks_out_tophat  read_counts_tophat.txt  tophat_out_copy
STAR_outAligned.sortedByCoord.out.bam  STAR_outLog.progress.out  check.py            fpkm_values_star.txt  star.py
```

In cufflinks folder:

```
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ cd cufflinks_out_star/
(base) paras21482@c39129c41926:/storage/DSG/paras21482/cufflinks_out_star$ ls
genes.fpkm_tracking     novel_transcripts.gtf       promoters.gtf       skipped.gtf
isoforms.fpkm_tracking  novel_transcripts_star.gtf  promoters_star.gtf  transcripts.gtf
(base) paras21482@c39129c41926:/storage/DSG/paras21482/cufflinks_out_star$ cd ..
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ cd cufflinks_out_tophat/
(base) paras21482@c39129c41926:/storage/DSG/paras21482/cufflinks_out_tophat$ ls
genes.fpkm_tracking  isoforms.fpkm_tracking  novel_transcripts.gtf  promoters.gtf  skipped.gtf  transcripts.gtf
(base) paras21482@c39129c41926:/storage/DSG/paras21482/cufflinks_out_tophat$ cd ..
```

In STAR_out_STARtmp:

```
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ cd STAR_out_STARtmp/
(base) paras21482@c39129c41926:/storage/DSG/paras21482/STAR_out_STARtmp$ ls
BAMsort
(base) paras21482@c39129c41926:/storage/DSG/paras21482/STAR_out_STARtmp$
```

In tophat_out

```
(base) paras21482@c39129c41926:/storage/DSG/paras21482$ cd tophat_out
(base) paras21482@c39129c41926:/storage/DSG/paras21482/tophat_out$ ls
accepted_hits.bam  align_summary.txt  cufflinks_out_tophat  deletions.bed  insertions.bed  junctions.bed  logs  prep_reads.info  unmapped.bam
(base) paras21482@c39129c41926:/storage/DSG/paras21482/tophat_out$
```

Plot show by tophat.py:

```
################################################################################################################################################################
################################################################################################################################################################
################################################################################################################################################################
################################################################################################################################################################
9345.75 - 18691.00: ##############
18691.00 - 28036.25: #########
28036.25 - 37381.50: ##########
37381.50 - 46726.75: #
46726.75 - 56072.00: #
56072.00 - 65417.25:
65417.25 - 74762.50: ##
74762.50 - 84107.75:
84107.75 - 93453.00: #
93453.00 - 102798.25:
102798.25 - 112143.50:
112143.50 - 121488.75:
121488.75 - 130834.00:
130834.00 - 140179.25:
140179.25 - 149524.50:
149524.50 - 158869.75:
158869.75 - 168215.00:
168215.00 - 177560.25:
177560.25 - 186905.50:
186905.50 - 196250.75:
196250.75 - 205596.00:
205596.00 - 214941.25:
214941.25 - 224286.50:
224286.50 - 233631.75:
233631.75 - 242977.00:
242977.00 - 252322.25: #
252322.25 - 261667.50:
261667.50 - 271012.75:
271012.75 - 280358.00: #
```

Star.py

```
################################################################################################################################################################
################################################################################################################################################################
################################################################################################################################################################
################################################################################################################################################################
#########################################################################
21835.44 - 43670.29: ################
43670.29 - 65505.14: ######
65505.14 - 87339.98:
87339.98 - 109174.83: ###
109174.83 - 131009.68: ##
131009.68 - 152844.52: ##
152844.52 - 174679.37: ##
174679.37 - 196514.22:
196514.22 - 218349.06:
218349.06 - 240183.91:
240183.91 - 262018.76:
262018.76 - 283853.60:
283853.60 - 305688.45:
305688.45 - 327523.30:
327523.30 - 349358.14:
349358.14 - 371192.99:
371192.99 - 393027.84:
393027.84 - 414862.69:
414862.69 - 436697.53:
436697.53 - 458532.38:
458532.38 - 480367.23:
480367.23 - 502202.07:
502202.07 - 524036.92:
524036.92 - 545871.77:
545871.77 - 567706.61:
567706.61 - 589541.46:
589541.46 - 611376.31:
611376.31 - 633211.15:
633211.15 - 655046.00: #
```

DATE
PAGE

(2.) (a)

The poisson distribut^n is actually a limiting case of Binomial distribut^n when the no. of trials n gets very large and p, the probability of success, is small. As a rule of thumb, if $n \geq 100$ & $np \leq 10$, the poisson distribut^n (taking $\lambda = np$) can provide very good approx. to binomial distribut^n.

Particularly useful in calculating the combinations inherent in probability formula associated with binomial distribut^n can become difficult when n is large

Consider Binomial probab. of seeing x successes in n trials,

$$P(x) = {}^nC_x \, p^x \, q^{n-x}$$

expected value of binomial distribut^n np by λ.

$$p = \frac{\lambda}{n}$$

$$q = 1 - p$$

$$q = 1 - \frac{\lambda}{n}$$

rewrite $P(x)$ in terms of $\lambda, n,$ & x we obtain

$$P(x) = {}^nC_x \left(\frac{\lambda}{n}\right)^x \left(1 - \frac{\lambda}{n}\right)^{n-x}$$

using standard formula for combinat^n of n things taken x at a time and some simple properties exponent

$$P(x) = \frac{n(n-1)(n-2)\ldots(n-x+1)}{x!} \cdot \frac{\lambda^x}{n^x} \left(1 - \frac{\lambda}{n}\right)^{n-x}$$

there are exactly x factors in the numerator of first fraction. We use swap denominator b/c 1st & 2nd factors, splitting n^x across all of the factors of 1st fract^n's numerator

$$P(x) = \frac{n}{n} \cdot \frac{n-1}{n} \cdots \frac{n-x+1}{n} \cdot \frac{\lambda^x}{x!} \left(1 - \frac{\lambda}{n}\right)^{n-x}$$

We can split last factor into two pieces, noting (for those familiar with calculus) that one has a limit of $e^{-\lambda}$

$$P(x) = \frac{n}{n} \cdot \frac{n-1}{n} \cdots \frac{n-x+1}{n} \cdot \frac{\lambda^x}{x!} \left(1 - \frac{\lambda}{n}\right)^{n} \left(1 - \frac{\lambda}{n}\right)^{-x}$$

It should now be relatively easy to see that if we let $n \to \infty$, keeping x & λ fixed, the 1st fract^n's in expression would tend towards 1, as would the last factor in expression. The second to last factor, as was mentioned before, tends towards $e^{-\lambda}$, the remaining factor stays unchanged as it does not depend on n! As such.

$$\text{lt}_{n \to \infty} P(x) = \frac{e^{-\lambda} \lambda^x}{x!}$$

DATE
PAGE

(b) Negative binomial Distribution

describes the prob. of observing x success before observing k failures in a sequence of independent Bernoulli trials, each with prob. of success p.

PMF of negative Binomial distribution is:

$$P(X=x) = {}^{x+k-1}C_x \times p^x \times (1-p)^k$$

x → no of success
k → no. of failure

Poisson Distribution

describes prob. of observing x events in a fixed interval of time or space, given an average rate λ of events per unit interval

$$P(X=x) = \frac{e^{-\lambda} \times \lambda^x}{x!}$$

x → no. of events
λ → avg. rate of events

Derivation

To derive relationship we consider some assumption,

① As the no. of trials $(x+k)$ in the -ve binomial distribution becomes very large $(x+k \to \infty)$

② Probability of success in each trial (p) becomes very small

③ The expected no. of success $(x \times p)$ remains constant, denoted by λ.

So, we have to show that -ve binomial distribution approaches the poisson distribution

starting - ve binomial Pmf using fact that $x \times p = \lambda$

$$P(X=x) = {}^{x+k-1}C_x \times p^x \times (1-p)^k$$

Substituting $p = \frac{\lambda}{r}$ into Pmf, we get

$$P(x = q) = \binom{q+k-1}{q} \times \left(\frac{\lambda}{r}\right)^q \times \left(1 - \frac{\lambda}{r}\right)^k$$

$r \to \infty$,

$$\lim_{r \to \infty} \left(1 - \frac{\lambda}{r}\right)^k = e^{-\lambda}$$

Pmf becomes

$$P(x = q) = \lim_{r \to \infty} \left(\frac{q+k-1}{C_q}\right) \times \left(\frac{\lambda}{r}\right)^q \times e^{-\lambda}$$

using fact that $\binom{q+k-1}{q} \to 1$ when $q >> k$, we get

$$P(x = q) = e^{-\lambda} \times \frac{\lambda^q}{q!}$$

Pmf of poisson distribut^n with λ as parameter no. of trials $\to \infty$ - a binomial distribut^n $\to$ & with p being such that $rxp = \lambda$, - a binomial distribut^n converges to poisson distribut^n.

DATE
PAGE

(c) As to get variance we need to have the mean

Mean $E(x) = \sum_{x=0}^{\infty} x p(x,\lambda)$

$$= \sum_{x=0}^{\infty} x \frac{e^{-\lambda} \lambda^x}{x!}$$

$$= \lambda e^{-\lambda} \left\{ \sum \left(\frac{\lambda^{x-1}}{(x-1)!} \right) \right\}$$

$$= \lambda e^{-\lambda} \lambda \left(\frac{1 + \lambda + \lambda^2}{2!} + \dots \right)$$

$$= \lambda e^{-\lambda} \lambda e^{\lambda}$$

As sum begins at $x=0$ since x results zero term we start sum as $x=1$

$$= \sum_{x=1}^{\infty} x \frac{e^{-\lambda} \lambda^x}{x!}$$

$$= e^{-\lambda} \sum_{x=1}^{\infty} \frac{\lambda^x}{(x-1)!}$$

$$= e^{-\lambda} \lambda \sum_{x=1}^{\infty} \frac{\lambda^{x-1}}{(x-1)!}$$

substituting $m = x-1$

$$= e^{-\lambda} \lambda \sum_{m=0}^{\infty} \frac{\lambda^m}{m!}$$

sum now taylor series expansion

$$E(x) = e^{-\lambda} \lambda e^{\lambda}$$

$$\boxed{E(x) = \lambda}$$

$$\text{Variance}(x) = E(x^2) - E(x)^2 \quad \text{---} \textcircled{1}$$

$$E(x^2) = \sum_{x=0}^{\infty} x^2 \, p(x, \lambda)$$

$$= \sum_{x=0}^{\infty} x^2 \, p(x, \lambda)$$

$$= \sum_{x=0}^{\infty} \{x(x-1) + x\} \, p(x, \lambda)$$

$$= \sum_{x=0}^{\infty} \{x(x-1) + x\} \frac{e^{-\lambda} \lambda^x}{x!}$$

$$= e^{-\lambda} \sum x(x-1) \frac{\lambda^x}{x!} + \sum x e^{-\lambda} \frac{\lambda^x}{x!}$$

$$= \lambda^2 e^{-\lambda} \sum_{x=2}^{\infty} \frac{\lambda^{x-2}}{(x-2)!} + \lambda$$

$$= \lambda^2 e^{-\lambda} e^{\lambda} + \lambda$$

$$= \lambda^2 + \lambda$$

Putting value in eq^n ① , we get

$$\text{Variance}(x) = E(x^2) - E(x)^2$$

$$= \lambda^2 + \lambda - (\lambda)^2$$

$$= \lambda$$

DATE
PAGE

(d) $E(x) = r p^r \sum_{k=0}^{\infty} \binom{k+r}{k} (1-p)^k$ — known/given

applying to $E[x^2]$

$$E(x^2) = r p^r \sum_{k=0}^{\infty} (k+r) \binom{k+r}{k} (1-p)^k$$

need to change $(k+r)$ to $k+r-1$ so,

$$n C_k = {}^{n-1}C_{k-1} + {}^{n-1}C_k$$

$$(k+r)\binom{k+r}{k} = (k+r)\binom{k+r-1}{k-1} + (k+r)\binom{k+r-1}{k}$$

$$= (r+1)\binom{k+r}{k-1} + r\binom{k+r}{k}$$

$$E(x^2) = r p^r \sum_{k=0}^{\infty} (k+r)\binom{k+r}{k}(1-p)^k$$

$$= r p^r \sum_{k=0}^{\infty} \left[(r+1)\binom{k+r}{k-1} + r\binom{k+r}{k}\right](1-p)^k$$

$$= r p^r \sum_{k=0}^{\infty} \left[(r+1)\binom{k+r}{k-1}\right](1-p)^k + r p^r \sum_{k=0}^{\infty} \left[r\binom{k+r}{k}\right](1-p)^k$$

$$E(x^2) = \frac{r(1-r-p)}{p^2}$$

$$V(x) = E(x^2) - [E(x)]^2 = \frac{r(1-r-p)}{p^2} + \frac{r^2}{p^2}$$

$$= \frac{q(1-p)}{p^2}$$

(3.) Calculating the poisson parameter (λ):

$$\lambda = \frac{\sum_{i=1}^{n} x_i}{n}$$

calculating mean

$$\lambda = \frac{2+3+4+4+2+1+1+2+3+3+4+5+6+7+2+2+4+1}{19}$$

$$= \frac{55}{19} = 2.89$$

Computing the likelihood:

The pmf is

$$P(x=k) = \frac{e^{-\lambda}\lambda^k}{k!} \quad (k \text{ is observed value})$$

for $k = 6$:

$$P(x=6) = \frac{e^{-2.89} \times (2.89)^6}{6!} = \frac{0.055 \times 582.62}{720}$$

$$= 0.044$$

for $k = 7$

$$P(x=7) = \frac{e^{-2.89} \times (2.89)^7}{7!} = \frac{0.055 \times 1683.77}{5040}$$

$$= 0.018$$

for $k=8$

$$p(x=8) = \frac{e^{-2.89} \times (2.89)^8}{8!} = \frac{0.055 \times 4866.11}{40320}$$

$$= 0.0066$$

Calculate the p-value

Lets calculate p-value

The formula is $P(X \leq k) = \sum_{i=0}^{k} \frac{e^{-\lambda} \lambda^i}{i!}$

$$p(x \leq 6) = \sum_{i=0}^{6} \frac{e^{-2.89} \times (2.89)^i}{i!}$$

$$= 0.158 * 0.229 * 0.221 * 0.159 * 0.092 * 0.044$$

$$= 0.903$$

$$p(x \leq 7) = \sum_{i=0}^{7} \frac{e^{-2.89} \times (2.89)^i}{i!} = 0.903 + 0.018 = 0.921$$

$$p(x \leq 8) = \sum_{i=0}^{8} \frac{e^{-2.89} \times (2.89)^i}{i!} = 0.921 + 0.0066 = 0.9276$$

4.

Central Limit Theorem (CLT):

The Central Limit Theorem (CLT) is a fundamental concept in statistics that states that the sampling distribution of the mean of any independent and identically distributed (iid) random variables approaches a normal distribution as the sample size increases, regardless of the shape of the original population distribution. In simpler terms, it implies that when you take repeated samples from a population and calculate the mean of each sample, those means will be normally distributed, even if the original population is not normally distributed.

Examples in Genomic Data Science:

- **Gene Expression Analysis**: In genomic data science, researchers often analyze gene expression data obtained from technologies like RNA-seq. The expression levels of genes across different conditions or samples can be considered as random variables. By applying the CLT, researchers can make inferences about the population mean expression levels and construct confidence intervals or conduct hypothesis tests even when the underlying distribution of gene expression levels may not be normal.

- **Variant Calling:** When analyzing genomic variations such as single nucleotide polymorphisms (SNPs) or insertions/deletions (indels), researchers often work with data from sequencing experiments. The CLT helps in estimating parameters related to the frequency of variants in a population, providing insights into genetic diversity or identifying potential disease-associated variants, even when the distribution of variants across the genome may not follow a specific pattern.

The CLT serves as a powerful tool in genomic data science by enabling robust statistical inference and hypothesis testing, even in complex and non-normally distributed data scenarios.

5.

3-Prime Bias in RNA-seq Data:

In RNA-seq experiments, 3-prime bias refers to the systematic bias towards the 3-prime end of mRNA transcripts during the sequencing process. This bias can lead to inaccuracies in quantification and analysis of gene expression levels.

Creation of 3-Prime Bias:

Several factors contribute to the creation of 3-prime bias in RNA-seq data:

- **RNA Fragmentation:** During library preparation for RNA-seq, RNA molecules are typically fragmented before sequencing. However, the fragmentation process might not be uniform, leading to a higher probability of fragmentation towards the 3-prime end of

transcripts.

- **Reverse Transcription and Amplification:** Reverse transcription of RNA into cDNA and subsequent amplification steps can introduce biases, with potential preferential amplification of the 3-prime ends of transcripts. This bias can be exacerbated during PCR amplification steps in library preparation.

- **Sequencing Chemistry**: Some sequencing chemistries or protocols may exhibit a higher likelihood of sequencing errors or reduced efficiency towards the 3-prime end of RNA molecules, contributing to bias in the generated sequencing data.

Impact on Analysis:

The presence of 3-prime bias can distort the estimation of gene expression levels, leading to inaccurate quantification of transcript abundance. This bias may affect downstream analyses such as differential expression analysis, pathway analysis, and isoform quantification, potentially resulting in misleading biological conclusions.

Researchers employ various normalization techniques and statistical methods to mitigate the effects of 3-prime bias in RNA-seq data, including trimming low-quality bases, employing bias-correction algorithms, and incorporating appropriate statistical models into the analysis pipeline.

6.

Limitations of Bridge PCR Methods Compared to Emulsion PCR:
Bridge PCR and emulsion PCR are both techniques used in DNA sequencing, particularly in next-generation sequencing (NGS) technologies like Illumina and 454 sequencing. While both methods facilitate the amplification of DNA fragments, they have distinct advantages and limitations.

Limitations of Bridge PCR Methods:

- **Template Bias**: Bridge PCR methods can suffer from template bias, where certain DNA sequences are preferentially amplified over others. This bias can result in uneven coverage across the genome, leading to inaccuracies in variant calling and quantitative analysis.

- **PCR Errors:** During bridge PCR, amplification errors such as polymerase slippage or misincorporation can occur, particularly in regions with repetitive sequences or high GC content. These errors can introduce artifacts into the sequencing data, complicating downstream analysis and interpretation.

- **Sample-to-Sample Contamination:** Bridge PCR involves the clustering of DNA templates on a solid surface, increasing the risk of sample-to-sample contamination. Cross-contamination between samples can lead to misassignment of sequencing reads, compromising the accuracy of the genomic analysis.

Advantages of Emulsion PCR over Bridge PCR:

- **Reduced Template Bias:** Emulsion PCR, as used in technologies like 454 sequencing, can mitigate template bias by isolating individual DNA fragments within water-in-oil droplets. This isolation reduces competition among templates during amplification, resulting in more uniform coverage across the genome.

- **Error Correction:** Emulsion PCR platforms often incorporate error-correction mechanisms, such as pyrosequencing, which can help identify and correct PCR errors during sequencing. This improves the accuracy of base calling and reduces the impact of amplification artifacts on downstream analysis.

- **Lower Contamination Risk:** Emulsion PCR reduces the risk of sample contamination compared to bridge PCR methods, as individual DNA fragments are compartmentalized within discrete reaction droplets. This isolation minimizes the potential for cross-contamination between samples, ensuring the integrity of the sequencing data.

While bridge PCR methods remain widely used in NGS workflows due to their scalability and cost-effectiveness, researchers must be aware of their limitations and consider alternative PCR strategies, such as emulsion PCR, for applications requiring high accuracy and reduced bias in sequencing data.

7.
Difference Between MAPQ and CIGAR Fields in SAM/BAM Files:
SAM (Sequence Alignment/Map) and its binary counterpart BAM are common file formats used to store nucleotide sequence alignment data generated from high-throughput sequencing experiments. Within these files, the MAPQ and CIGAR fields provide important information about the alignment quality and structure, respectively.

<u>MAPQ (Mapping Quality):</u>

- **Definition:** MAPQ represents the mapping quality of a read, which quantifies the confidence in the accuracy of its alignment to the reference genome.
- **Range:** MAPQ values range from 0 to 255, with higher values indicating higher confidence in the alignment.
- **Interpretation:** A MAPQ of 255 typically denotes a unique, high-confidence alignment, while lower MAPQ scores may indicate potential alignment ambiguities or mapping errors.
- **Usage:** MAPQ values are crucial for filtering and downstream analysis, such as variant

calling and differential expression analysis, as they help distinguish reliable alignments from potential artifacts or sequencing errors.

CIGAR (Compact Idiosyncratic Gapped Alignment Report):

- **Definition:** The CIGAR string describes the alignment structure of the read relative to the reference genome, summarizing information about matches, insertions, deletions, and gaps.
- **Format:** The CIGAR string is a series of alphanumeric characters, where each character represents a specific type of alignment operation (e.g., 'M' for match/mismatch, 'I' for insertion, 'D' for deletion).
- **Example:** For instance, a CIGAR string of "50M2D30M" indicates a read that aligns with 50 matching bases, followed by a 2-base deletion, and then 30 additional matching bases.
- **Usage:** The CIGAR string is essential for reconstructing the alignment and identifying structural variations (e.g., insertions, deletions) between the read and the reference genome. It is often used in downstream analyses, including variant calling, indel detection, and transcriptome assembly.

In summary, while the MAPQ field quantifies the confidence of a read's alignment, the CIGAR field provides detailed information about the alignment structure, facilitating downstream analysis and interpretation of sequencing data. Both fields play critical roles in quality control, variant detection, and other bioinformatics applications.