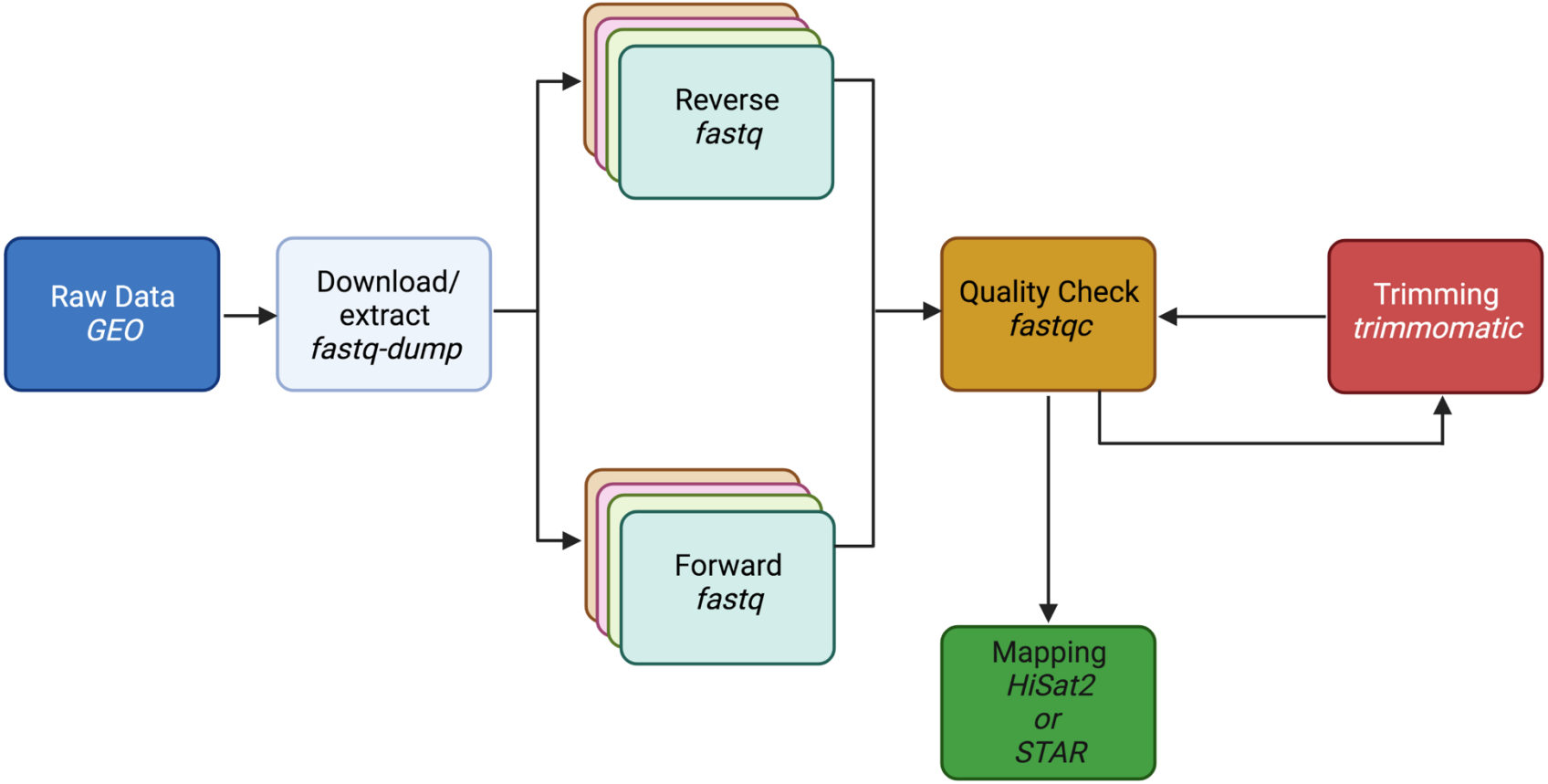


Interpreting Outputs from Alignment Step

February 26, 2025

Outline for Today

- Discuss outputs from alignment (hisat2)
- Class Exercise #1
- Class Exercise #2
- Class Exercise #3

Data Analysis Workflow: File formats

- Quality Control
  - Sample Quality and consistency (FASTQC)
  - Is trimming appropriate - quality/adapters (trimmomatic)
  - **FASTQ file**
- Alignment/Mapping
  - Reference Target (Sequence and annotation files)
  - Alignment programs & parameters (hisat2)
  - **BAM file**
- Quantification (next week)
  - Counting methods and parameters
  - **Count matrices**

Discussion Points

1. What written code was found inside the hisat2_align.sh script?
2. What will need to be modified when running the hisat2_align.sh script on your samples?
3. What outputs should you expect after alignment?

Discussion Points

1. What written code was found inside the hisat2_align.sh script?
2. What will need to be modified when running the hisat2_align.sh script on your samples?
3. What outputs should you expect?

Class Exercise #1

1. Navigate to **HISAT2_example**

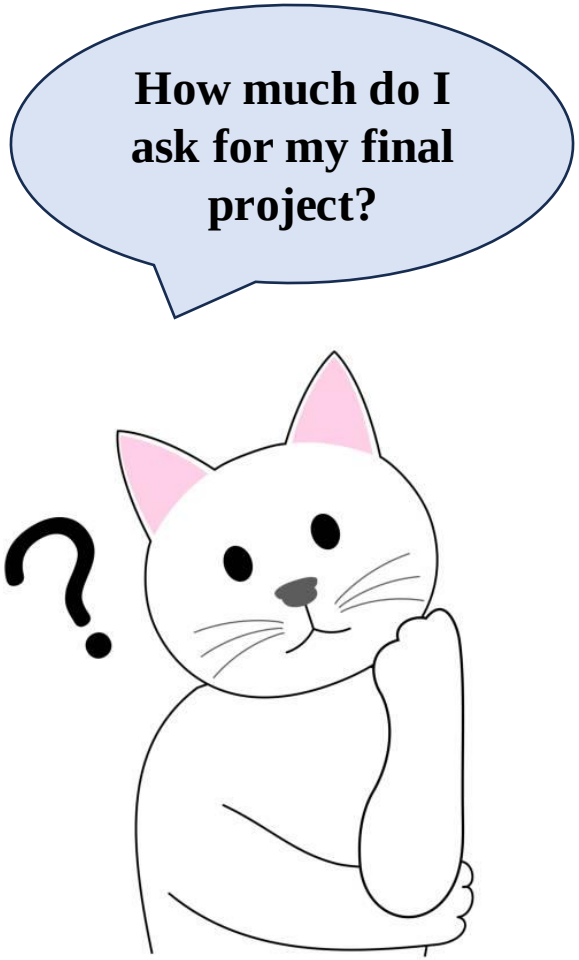
2. Make a copy of **hisat2_align.sh**

3. Call it **hisat2_finalproj.sh**

4. Open **hisat2_finalproj.sh** using Jupyter Notebooks

Part 1: SLURM Directives

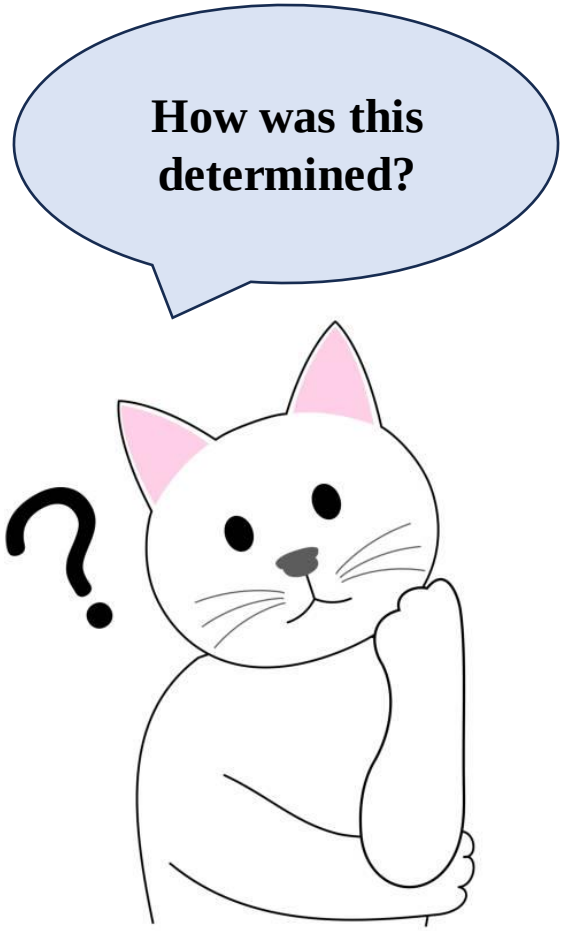
```
#!/bin/bash
#SBATCH --partition=general
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=10G
#SBATCH --time=3:00:00
#SBATCH --job-name=align_CD8
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```



Alignment specific

For 8 samples or less

```
#!/bin/bash
#SBATCH --partition=general
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=48G
#SBATCH --time=12:00:00
#SBATCH --array=1-8
#SBATCH --job-name=align_CD8
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```



Too much computational power for FASTQC*

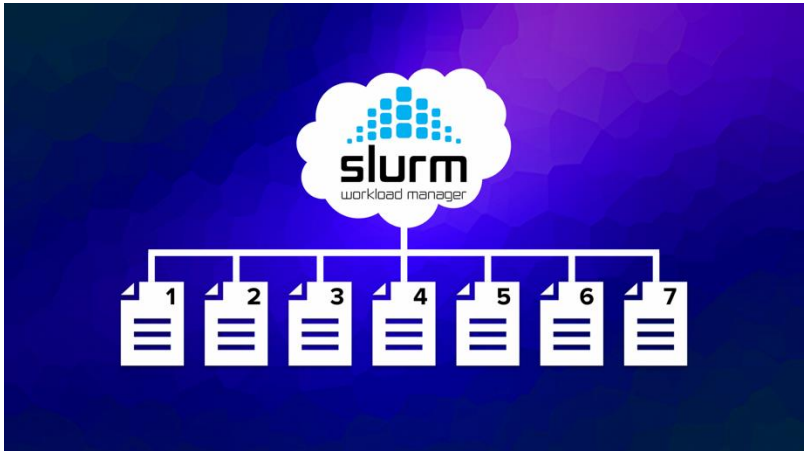
Breaking down the code

```
#SBATCH --nodes=1 #benefit is in multi-threading
#SBATCH --ntasks=8 # 1 task per sample
#SBATCH --mem=48G #6GB per sample
#SBATCH --time=24:00:00
#SBATCH --array=1-8
```

SLURM job array

- ***Each sample runs independently***

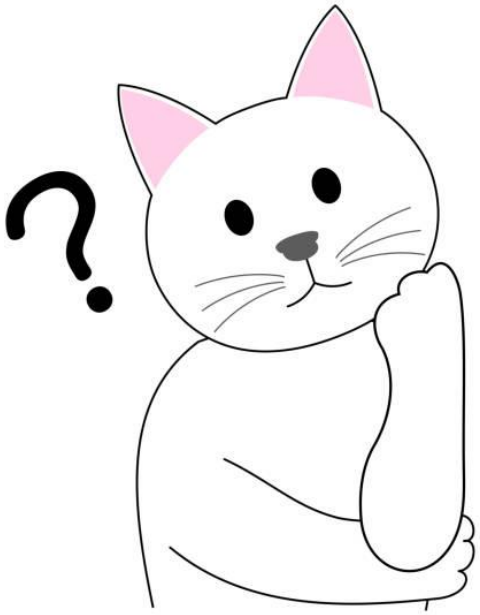
```
[[redacted]@ohead1 Reptile_validation_data]$ sbatch run_foml_trans_continuous.sh 6300
Submitted batch job 260191
[[redacted]@ohead1 Reptile_validation_data]$ squeue -u [redacted]
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
260191_[15-900%10]    normal fotrcont [redacted] PD       0:00      1 (JobArrayTaskLimit)
          260191_1    normal fotrcont [redacted]  R       0:45      1 onode19
          260191_2    normal fotrcont [redacted]  R       0:45      1 onode19
          260191_3    normal fotrcont [redacted]  R       0:45      1 onode19
          260191_4    normal fotrcont [redacted]  R       0:45      1 onode19
          260191_9    normal fotrcont [redacted]  R       0:45      1 onode08
         260191_10    normal fotrcont [redacted]  R       0:45      1 onode08
         260191_11    normal fotrcont [redacted]  R       0:45      1 onode08
         260191_12    normal fotrcont [redacted]  R       0:45      1 onode08
         260191_13    normal fotrcont [redacted]  R       0:45      1 onode09
         260191_14    normal fotrcont [redacted]  R       0:45      1 onode09
[[redacted]@ohead1 Reptile_validation_data]$
```



Alignment specific

For 8-16 samples

```
#!/bin/bash
#SBATCH --partition=general
#SBATCH --nodes=1
#SBATCH --ntasks=16
#SBATCH --mem=80G
#SBATCH --time=24:00:00
#SBATCH --array=1-16
#SBATCH --job-name=align_CD8
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out
```



Too much computational power for FASTQC*

8 samples or less	8-16 samples
#SBATCH --nodes=1	#SBATCH --nodes=1
#SBATCH --ntasks=8	#SBATCH --ntasks=16
#SBATCH --mem=48G	#SBATCH --mem=80G
#SBATCH --time=24:00:00	#SBATCH --time=24:00:00
#SBATCH --array=1-8	#SBATCH --array=1-16

Take a few minutes to change your parameters according to your final project

```
# Iterate through each fastq.gz file in the current directory
for fastq_file in *fastq.gz; do

    # Extract sample name from the file name
    SAMPLE=$(echo ${fastq_file} | sed "s/.fastq.gz//")
    echo ${SAMPLE}.fastq.gz
```

Part 2: Initiating the for loop (line 11 - 16)

Looping through FASTQ files (line 12)

```bash
for fastq_file in *fastq.gz; do
```

- This line starts a loop that goes through each file in the current directory that ends with *fastq.gz
  (does your samples say *fq.gz)

- The variable fastq_file will temporarily store the name of each file during each loop iteration

Our directory contains these files:

SRR13423162.fastq.gz

SRR13423165.fastq.gz

The loop will process them **one by one**, setting fastq_file to:

1. **SRR13423162.fastq.gz**
2. **SRR13423165.fastq.gz**

Extracting the Sample Name (line 14)

```bash
SAMPLE=$(echo ${fastq_file} | sed "s/.fastq.gz//")
```

- This line removes `*fastq.gz` from the filename to extract just the sample name
- It uses `sed`

sed (short for stream editor)

- Commonly used for substituting, deleting, inserting, or modifying text in a file

s/pattern/replacement/options

*s=substitution

```
sed "s/.fastq.gz//”
```

```
sed “s/.fastq.gz/nothing/”
```

Extracting the Sample Name (line 14)

```bash
SAMPLE=$(echo ${fastq_file} | sed "s/.fastq.gz//")
```

- The result is stored in the variable called SAMPLE

original	SAMPLE
SRR13423162.fastq.gz	SRR13423162
SRR13423165.fastq.gz	SRR13423165

Part 3: Set database directory, genome, and processor count (line 18)

DBDIR="/gpfs1/cl/mmg3320/course_materials/genome_index/hisat2_index_mm10”

GENOME="GRCm39” **#basename of index files**

p=2

*This is specific for mouse!

Part 4: Load required modules (line 23)

```
# Load required modules
  module load gcc/13.3.0-xp3epyt
  module load hisat2/2.2.1-x7h4grf
  module load samtools/1.19.2-pfmpoam
```

Part 5: align with hisat2

SE only (line 28)

```
hisat2 \
-p ${p} \
-x ${DBDIR}/${GENOME} \
-U ${SAMPLE}.fastq.gz \
-S ${SAMPLE}.sam &> ${SAMPLE}.log
```

*files containing unpaired reads to be aligned

Part 5: align with hisat2

PE only

```
hisat2 \
-p ${p} \
-x ${DBDIR}/${GENOME} \
-1 ${SAMPLE}_R1.fastq.gz \
-2 ${SAMPLE}_R2.fastq.gz \
-S ${SAMPLE}.sam &> ${SAMPLE}.log
```

*files containing R1 and R2

Part 5: Other changes for *PE only (lines 11-16)*

```
# Iterate through each fastq.gz file in the current directory
for fastq_file in *_R1.fastq.gz; do


# Extract sample name from the file name
SAMPLE=$(echo ${fastq_file} | sed "s/_R1.fastq.gz//")
echo ${SAMPLE}_R1.fastq.gz
```

Why will this script not work “as is” for ***PE only ?***

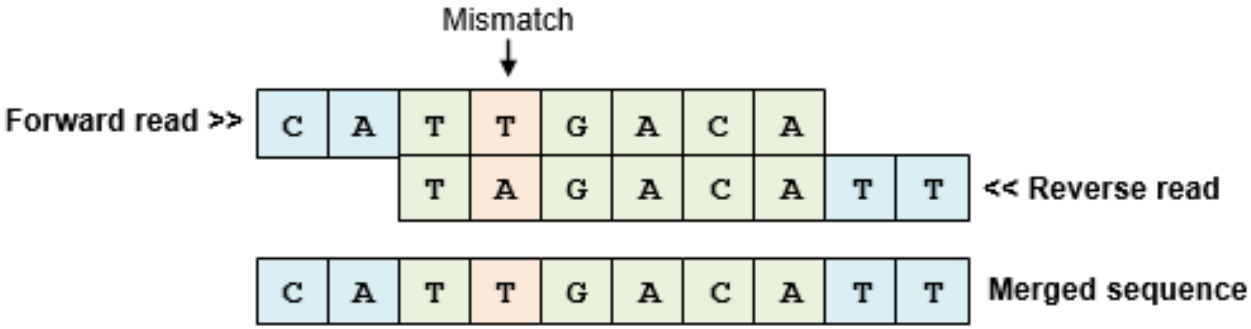
- You are asking it to loop over all *.fastq files
  - The for loop iterates over all files matching *.fastq, including both _R1.fastq and _R2.fastq files
  - ***This means it will process _R2.fastq files separately***

- `sample_1.fastq.gz` → Contains **forward (R1) reads**
- `sample_2.fastq.gz` → Contains **reverse (R2) reads**

What happens if you process _1.fastq.gz and _2.fastq.gz individually? ***PE only***

1. Loss of Paired-End Mapping Advantages
2. Increase False-Positive Alignments
3. Loss of Structural Information
4. Potential Read Duplications

Take home message:
Forward and Reverse reads work together to help align DNA fragments to the reference genome more precisely



Why will this script not work “as is” for ***PE only ?***

SE	PE
for i in *.fastq; do	for i in *_R1.fastq; do
sed "s/fastq.gz//")	sed "s/_R1.fastq.gz//")
echo ${SAMPLE}fastq.gz	echo ${SAMPLE}_R1.fastq.gz

- The script assumes that if SAMPLE_R1.fastq exists, SAMPLE_R2.fastq exists too.

samtools view

```
samtools view -b input.sam > input.bam
```

- Input is usually a SAM file, *but can also use a BAM*
- Common uses: extracting a subset of data into a new file, **converting between SAM/BAM files**

samtools sort

```
samtools sort sample.bam -o sample.sorted.bam
```

- Reads need to be ordered in “genomic order” – not the order in which they were sequenced

samtools index

```
samtools index sorted.bam
```

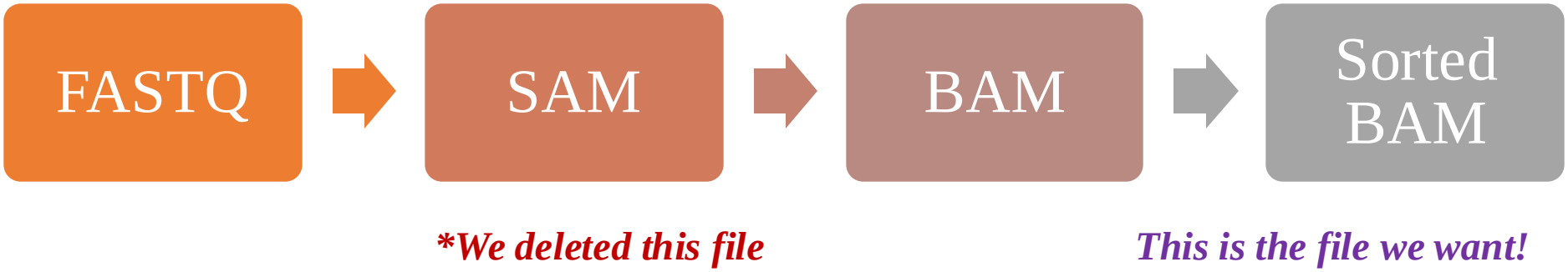
- Creates index file that allows for fast look-up
- Generates *.bam.bai file

Discussion Points

1. What written code was found inside the `hisat2_align.sh` script?
2. What will need to be modified when running the `hisat2_align.sh` script on your samples?
3. What outputs should you expect after aligning?

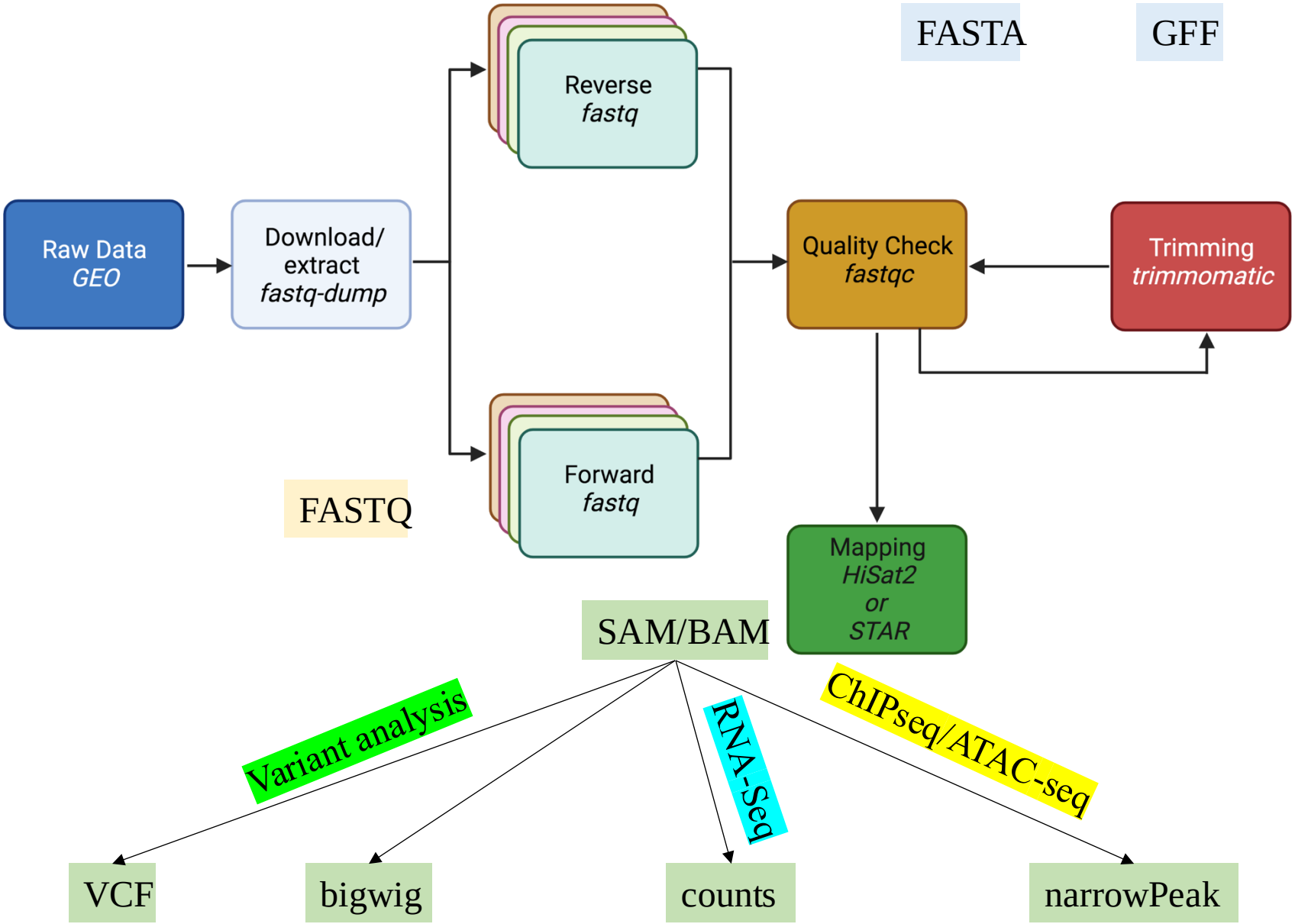
For <u>each</u> fastq file, this script will output:

1. BAM file (.bam)
2. Sorted BAM file (sorted.bam)
3. BAM index file (.bam.bai)
4. Statistics about alignment (.txt/.log; 2 files)



What is a SAM/BAM file?

- A BAM file is a binary version of Sequence Alignment Map (SAM) file.
- Both stores alignment sequencing reads against a reference genome.
- These files are much smaller in size and more efficient for storage and processing
- BAM files can be visualized with Genomic Viewers (IGV)



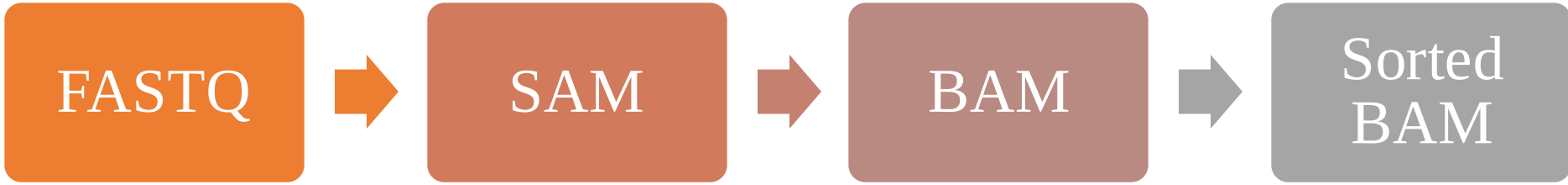
“GATEWAY FILE”

Class Exercise 2: Compare the outputs from HISAT2_exercise vs HISAT2_modify (~5mins)

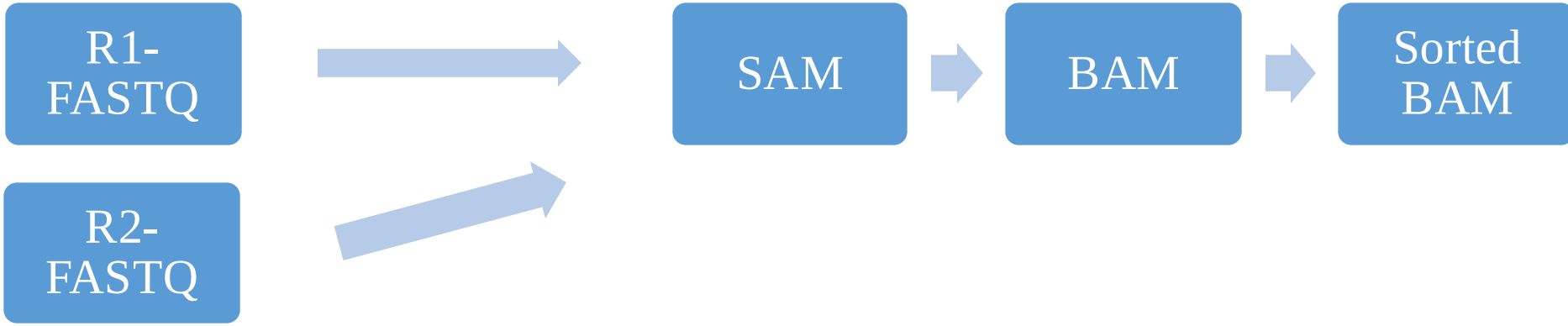
	HISAT2_example **Class Exercise 1**	**HISAT2_modify** **Class Exercise 3**
SE or PE	SE	PE
FASTQ input	SRR13423162.fastq.gz	JC1A_R1.fastq.gz JC1A_R2.fastq.gz
Outputs after alignment	**SRR13423162.bam** **SRR13423162.log** **SRR13423162_sorted.bam** **SRR13423162_sorted.bam.bai** **SRR13423162.txt**	

	HISAT2_example **Class Exercise 1**	**HISAT2_modify** **Class Exercise 3**
SE or PE	SE	PE
FASTQ input	SRR13423162.fastq.gz	JC1A_R1.fastq.gz JC1A_R2.fastq.gz
Outputs after alignment	**SRR13423162.bam** **SRR13423162.log** **SRR13423162_sorted.bam** **SRR13423162_sorted.bam.bai** **SRR13423162.txt**	**JC1A.bam** **JC1A.log** **JC1A_sorted.bam** **JC1A_sorted.bam.bai** **JC1A.txt**

SINGLE END



PAIRED END



Class Exercise 3

Run multiqc inside of HISAT2_example

```
module load gcc/13.3.0-xp3epyt
module load py-multiqc/1.15-fmpaaj7
```

Command will be:

```
multiqc .
```

View the output: multiqc_report.html

Interpreting multiqc

- Samtools flagstat provides counts for each of the 13 categories

	HISAT2_example **Class Exercise 1**
SE or PE	**SE**
FASTQ input	SRR13423162.fastq.gz
Outputs after alignment	SRR13423162.bam SRR13423162.log SRR13423162_sorted.bam SRR13423162_sorted.bam.bai **SRR13423162.txt**

Samtools flagstat interpretations

1. Total Number of Reads

34818870 + 0 in total (QC-passed reads + QC-failed reads)

- 34,818,870 reads were processed in total
- The +0 means no additional QC-failed reads were included

2. Primary vs Secondary Alignments

25593457 + 0 primary

The total number of reads assigned as a primary alignment; main set of reads used for analysis

9225413 + 0 secondary

The total number of reads assigned as a secondary alignment; align to multiple locations in the genome, often found in repetitive sequences

Samtools flagstat interpretations

3. Duplicate Reads

0 + 0 duplicates

No duplicate reads, meaning PCR duplicates were found

Samtools flagstat interpretations

4. Mapped Reads

33573586 + 0 mapped (96.42% : N/A)

96.42% successfully mapped to the reference genome; comprised of primary and secondary reads

24348173 + 0 primary mapped (95.13% : N/A)

Of the total primary reads identified, 95.13% of those reads were mapped

25593457 - 24348173 = 1,245,284 reads <u>did not align</u>

Samtools flagstat interpretations

5. Paired-End Information

0 + 0 paired in sequencing

0 + 0 read1

0 + 0 read2

0 + 0 properly paired (N/A : N/A)

0 + 0 with itself and mate mapped

0 + 0 singletons (N/A : N/A)

Final Interpretation

✅ **34.8 million reads were processed.**

✅ **25.5 million reads were primary alignments.**

✅ **33.5 million reads (96.42%) mapped to the genome – a good alignment rate.**

✅ **This dataset is single-end sequencing, not paired-end.**

Homework #7

hisat2_align.sh

```
#!/bin/bash
#SBATCH --partition=bluemoon
#SBATCH --nodes=1
#SBATCH --ntasks=2
#SBATCH --mem=10G
#SBATCH --time=3:00:00
#SBATCH --job-name=align_CD8
# %x=job-name %j=jobid
#SBATCH --output=%x_%j.out

for i in *fastq.gz; do
SAMPLE=$(echo ${i} | sed "s/.fastq.gz//")
echo ${SAMPLE}.fastq.gz

DBDIR=/gpfs1/cl/mmg232/course_materials/hisat2_index
GENOME="GRCm39"
p=2

module load hisat2-2.1.0-gcc-7.3.0-knvgwpc
module load samtools-1.10-gcc-7.3.0-pdbkohx

#align to GRCm39
hisat2 \
  -p ${p} \
  -x ${DBDIR}/${GENOME} \
  -U ${SAMPLE}.fastq.gz \
  -S ${SAMPLE}.sam &> ${SAMPLE}.log

#create bam file
samtools view ${SAMPLE}.sam \
  --threads 2 \
  -b \
  -o ${SAMPLE}.bam \
```

1

2

3

4

This will not be "ready-to-go"

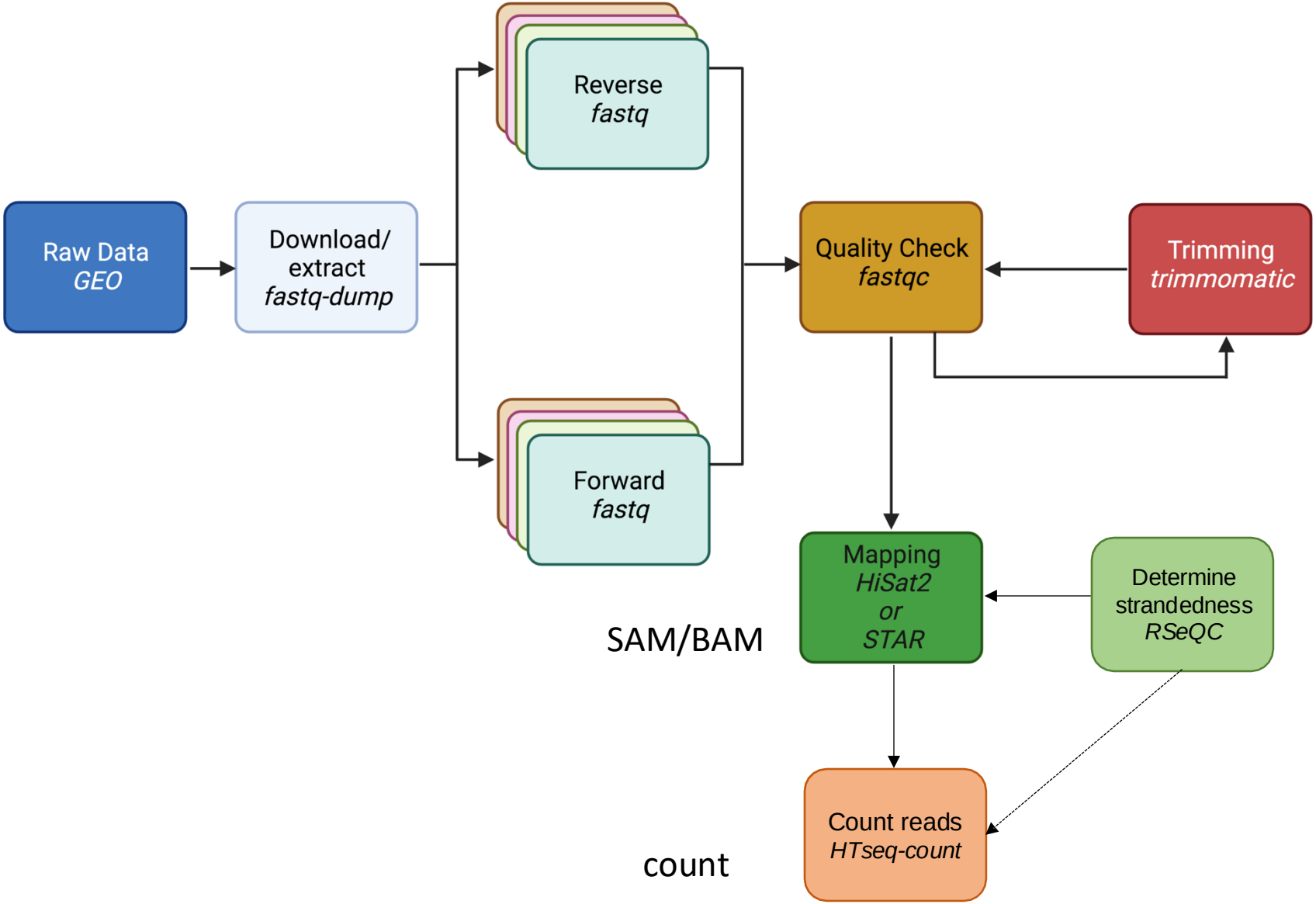
Basic Template

Read the methods

Did the authors add special arguments during alignment?

TRY TO understand why this was done.

Email me and we can chat!


Raw Data
GEO
Download/
extract
fastq-dump
Reverse
fastq
Forward
fastq
Quality Check
fastqc
Trimming
trimmomatic
Mapping
HiSat2
or
STAR
SAM/BAM
Determine
strandedness
RSeQC
Count reads
HTseq-count
count