**RNA-Seq Analysis Pipeline Notes**

**# submit job**

sbatch 3map.featurecount.slurm.sh

**# watch it work**

watch -n 30 qstat -u user

**1. RNA-Seq Data Preparation and FASTQ File Handling**

1.1 **Decompression of FASTQ Files Using pigz:**

* RNA-seq data typically arrives in compressed .fastq.gz format. To efficiently decompress files in parallel, we use pigz (parallelized gzip).

**Decompression Script**:

#!/bin/bash

#SBATCH --job-name=unzipFASTQ # Job name

#SBATCH --partition=xeon # Partition (adjust based on your HPC)

#SBATCH --nodes=1 # Number of nodes

#SBATCH --ntasks=1 # Number of tasks (1 task per job)

#SBATCH --cpus-per-task=20 # Number of CPU cores (adjust based on available resources)

#SBATCH --mem=16G # Memory allocation

#SBATCH --time=02:00:00 # Time limit hrs:min:sec

#SBATCH --output=unzip\_%A.out # Standard output

#SBATCH --error=unzip\_%A.err # Standard error log

# Load pigz if necessary

# module load pigz

# Define the directory where the FASTQ files are located

DIR="/home/kdhusia/PROJECTS/RNAnfCore/FASTQC/"

# Change to the directory

cd $DIR

# Decompress all files ending with \*001.fastq.gz using pigz

echo "Starting decompression of all \*001.fastq.gz files in $DIR"

pigz -p 30 -d \*fastq.gz

echo "Decompression complete!"

**2. RNA-Seq Read Quality Control (QC)**

2.1 **Trimming and Filtering Reads Using Trimmomatic**:

* Adapter removal, quality trimming, and filtering are important to improve the accuracy of downstream analysis.

**Trimming Script**:

#!/bin/bash

#SBATCH --job-name=trimFASTQ

#SBATCH --nodes=1

#SBATCH --ntasks=1

#SBATCH --cpus-per-task=30

#SBATCH --mem=16G

#SBATCH --time=03:00:00

#SBATCH --output=trim\_%A.out

#SBATCH --error=trim\_%A.err

# Define directories

FASTQ\_DIR="/path/to/FASTQ/"

OUTPATH="/path/to/TRIMMED/"

# Loop over paired-end files and run Trimmomatic

for f1 in ${FASTQ\_DIR}\*R1.fastq.gz; do

f2=${f1%%R1.fastq.gz}"R2.fastq.gz"

java -jar /path/to/Trimmomatic-0.39/trimmomatic-0.39.jar PE -threads 30 \

-phred33 $f1 $f2 \

${OUTPATH}TRIMMED\_PAIRED\_${f1} ${OUTPATH}TRIMMED\_UNPAIRED\_${f1} \

${OUTPATH}TRIMMED\_PAIRED\_${f2} ${OUTPATH}TRIMMED\_UNPAIRED\_${f2} \

ILLUMINACLIP:/path/to/adapters/TruSeq3-PE.fa:2:30:10 \

LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

done

**3. RNA-Seq Alignment Using STAR**

3.1 **Creating STAR Genome Index**:

* You first need to generate a genome index for alignment using the STAR aligner.

**Genome Indexing Command**:

#!/bin/bash

#SBATCH --job-name=STARindex

#SBATCH --nodes=1

#SBATCH --ntasks=1

#SBATCH --cpus-per-task=16

#SBATCH --mem=50G

#SBATCH --time=10:00:00

#SBATCH --output=star\_index\_%A.out

#SBATCH --error=star\_index\_%A.err

GENOME\_DIR="/path/to/genome/"

GENOME\_FASTA="/path/to/genome/GRCh38.p14.genome.fa"

GTF\_FILE="/path/to/annotation.gtf"

STAR --runThreadN 16 --runMode genomeGenerate --genomeDir $GENOME\_DIR \

--genomeFastaFiles $GENOME\_FASTA --sjdbGTFfile $GTF\_FILE \

--sjdbOverhang 100

3.2 **Mapping Reads with STAR**:

* You map the trimmed reads using STAR. Ensure that both R1 and R2 are mapped.

**Alignment Script**:

#!/bin/bash

#SBATCH --job-name=mapRNASeq

#SBATCH --nodes=1

#SBATCH --ntasks=1

#SBATCH --cpus-per-task=16

#SBATCH --mem=60G

#SBATCH --time=20:00:00

#SBATCH --output=map\_%A.out

#SBATCH --error=map\_%A.err

GENOME\_DIR="/path/to/genome"

GTF="/path/to/annotation.gtf"

OUT\_DIR="/path/to/alignment\_output/"

for f1 in TRIMMED\_PAIRED\*R1.fastq; do

f2=${f1%%R1.fastq}"R2.fastq"

STAR --runThreadN 16 --genomeDir ${GENOME\_DIR} --readFilesIn ${f1} ${f2} \

--outFileNamePrefix ${OUT\_DIR}/${f1%%R1.fastq}\_Mapped\_ \

--outSAMtype BAM SortedByCoordinate \

--sjdbGTFfile ${GTF}

done

**4. Sorting BAM Files with Samtools**

4.1 **BAM Sorting**:

* You can sort BAM files using samtools to make them compatible for downstream processing like indexing or feature counting.

**BAM Sorting and Indexing**:

for bam in \*\_Mapped\_Aligned.out.bam; do

sorted\_bam=${bam%%.out.bam}\_sorted.bam

samtools sort -@ 16 -o $sorted\_bam $bam

samtools index $sorted\_bam

done

**5. Counting Reads with featureCounts**

5.1 **Read Quantification Using featureCounts**:

* Use featureCounts to quantify reads aligned to exons or other regions defined in the GTF file.

**featureCounts Command**:

for sorted\_bam in \*\_sorted.bam; do

featureCounts -T 16 -p -t exon -a ${GTF} --extraAttributes gene\_name,gene\_biotype \

-o ${COUNT\_PATH}/${sorted\_bam%%\_sorted.bam}\_counts.txt $sorted\_bam

done

**6. Running Multiple Samples in Parallel**

6.1 **Parallel Execution Using xargs**:

* When processing multiple samples, we can use xargs to run multiple jobs in parallel to speed up the pipeline.

**Parallel Execution with xargs**:

export -f run\_sample # Export the function to make it accessible to xargs

# Run STAR pipeline for multiple samples in parallel

**ls TRIMMED\_PAIRED\*R1.fastq | xargs -n 1 -P 2 bash -c 'run\_sample "$@"' \_**

**Explanation**:

* The run\_sample function includes STAR alignment, samtools sorting, and feature counting.
* xargs runs two jobs in parallel (-P 2) and processes each file (-n 1).

**7. Data Management and Downloading Missing Files**

7.1 **Using SCP to Download Missing Files**:

* If some files are missing on your local machine, you can use rsync or scp to download them, showing progress.

**Example rsync Command**:

rsync -avz –progress kdhusia@login.hpc.uams.edu:/home/kdhusia/PROJECTS/ /RNAnfCore/FASTQC/FastQ/4ALIGN/\*.bam /localpath

**8. Error Handling and Troubleshooting**

8.1 **STAR Input File Not Found**:

* Ensure that the input files (R1 and R2) are correctly specified.

**Solution**: Double-check file paths and the pattern used to match files.

**Example**:

PATTERN='TRIMMED\_(PAIRED|UNPAIRED)\_(.\*)\_R[12].fastq'

8.2 **Missing BAM or SAM Files**:

* If the .sam or .bam files are not being created as expected, confirm that the paths and filenames are correct and that STAR is completing successfully.

**Check**:

* Verify that STAR completed successfully by reviewing the log files: \_Mapped\_Log.final.out, \_Mapped\_Log.out.

This RNA-seq pipeline efficiently processes raw sequencing data, starting from decompression to trimming, mapping, sorting, and counting reads. We use parallelization (xargs) to optimize CPU usage on HPC systems. By employing samtools and featureCounts, this pipeline produces sorted BAM files and read counts for each gene. These steps provide an end-to-end guide for analyzing RNA-seq data, with flexible configurations for handling multiple samples in parallel on HPC systems.