conda create -n s\_aureus\_rnaseq -c conda-forge -c bioconda \

sra-tools fastp multiqc bowtie2 bwa samtools subread bedtools salmon \

r-base r-essentials bioconductor-deseq2 bioconductor-tximport bioconductor-clusterprofiler r-apeglm r-pheatmap r-enhancedvolcano

conda activate s\_aureus\_rnaseq

project\_root/

├── data/ # raw fastq files

├── ref/ # reference genome and annotation (fasta + gff/gtf)

├── fastp\_reports/

├── alignments/ # bam files

├── counts/ # featureCounts outputs

├── R/ # R scripts

├── results/ # DE results, figures, enrichment

└── reports/ # clinical microbiology report (markdown -> html)

mkdir -p data raw\_sra

cd raw\_sra

Prefetch (downloads SRA files locally)

for SRR in SRR20959676 SRR20959677 SRR20959678 SRR20959679 SRR20959680 SRR20959681 SRR20959682 SRR20959683; do

prefetch $SRR || echo "prefetch failed for $SRR"

done

Convert to fastq (paired-end assumed; if single-end remove --split-3)

mkdir -p ../data

for SRR in SRR20959676 SRR20959677 SRR20959678 SRR20959679 SRR20959680 SRR20959681 SRR20959682 SRR20959683; do

fasterq-dump --split-3 -p -O ../data $SRR || echo "fasterq-dump failed for $SRR"

done

cd ..

Optional: gzip the fastqs

cd data

for fq in .fastq; do gzip -f "$fq

mkdir -p fastp\_reports cleaned\_fastq

cat > samples.tsv <<'TSV'

SRR20959676 chronic

SRR20959677 chronic

SRR20959678 chronic

SRR20959679 chronic

SRR20959680 acute

SRR20959681 acute

SRR20959682 acute

SRR20959683 acute

TSV

while IFS=$'\t' read -r SRR STATE; do

R1="data/${SRR}\_1.fastq.gz"

R2="data/${SRR}\_2.fastq.gz"

outR1="cleaned\_fastq/${SRR}\_1.clean.fastq.gz"

outR2="cleaned\_fastq/${SRR}\_2.clean.fastq.gz"

fastp -i $R1 -I $R2 -o $outR1 -O $outR2 \

--detect\_adapter\_for\_pe --thread 4 \

--html fastp\_reports/${SRR}\_fastp.html --json fastp\_reports/${SRR}\_fastp.json

done < samples.tsv

multiqc fastp\_reports -o fastp\_reports/

mkdir -p ref

Example (user to replace with the exact RefSeq assembly accession):

wget -O ref/genome.fna "ftp://ftp.ncbi.nlm.nih.gov/..../GCF\_XXXXXX\_genomic.fna.gz"

wget -O ref/annotation.gff "ftp://.../GCF\_XXXXXX\_genomic.gff.gz"

gunzip ref/\*.gz

Build aligner index (choose one)

Bowtie2 index:

bowtie2-build ref/genome.fna ref/s\_aureus\_bowtie2\_index

Or bwa index:

bwa index ref/genome.fna

mkdir -p alignments

for SRR in SRR20959676 SRR20959677 SRR20959678 SRR20959679 SRR20959680 SRR20959681 SRR20959682 SRR20959683; do

R1="cleaned\_fastq/${SRR}\_1.clean.fastq.gz"

R2="cleaned\_fastq/${SRR}\_2.clean.fastq.gz"

outbam="alignments/${SRR}.sorted.bam"

bowtie2 -x ref/s\_aureus\_bowtie2\_index -1 $R1 -2 $R2 -p 8 | \

samtools view -b - | samtools sort -o $outbam

samtools index $outbam

done

mkdir -p counts

create a list of bam files

ls alignments/.sorted.bam > bamlist.txt

featureCounts -T 6 -p -t gene -g ID -a ref/annotation.gff -o counts/featureCounts\_raw.txt -s 0 -B -C -F GFF bamlist.txt

The output counts matrix is in counts/featureCounts\_raw.txt

Convert to a simpler counts-only matrix (for DESeq2)

cut -f1,7- counts/featureCounts\_raw.txt > counts/count\_matrix.txt

R/deseq2\_analysis.R

library(DESeq2)

library(tximport)

library(apeglm)

library(pheatmap)

library(EnhancedVolcano)

Read count matrix produced above

counts <- read.table('counts/count\_matrix.txt', header=TRUE, row.names=1, comment.char='\t')

featureCounts output often has columns like: Geneid, length, SRR... Adjust accordingly

If needed, ensure columns are just sample columns

Create sample metadata

samples <- data.frame(

row.names = c('SRR20959676','SRR20959677','SRR20959678','SRR20959679','SRR20959680','SRR20959681','SRR20959682','SRR20959683'),

condition = factor(c('chronic','chronic','chronic','chronic','acute','acute','acute','acute'))

)

Ensure order of samples matches count columns

counts <- counts[, rownames(samples)]

Build DESeq2 dataset

dds <- DESeqDataSetFromMatrix(countData = counts, colData = samples, design = ~ condition)

Prefilter low counts

keep <- rowSums(counts(dds) >= 10) >= 2

dds <- dds[keep,]

dds <- DESeq(dds)

res <- results(dds, contrast=c('condition','acute','chronic'))

res <- lfcShrink(dds, coef='condition\_acute\_vs\_chronic', type='apeglm')

Order and save

resOrdered <- res[order(res$padj),]

write.csv(as.data.frame(resOrdered), file='results/deseq2\_results\_acute\_vs\_chronic.csv')

PCA

vsd <- vst(dds, blind=FALSE)

png('results/PCA\_samples.png', width=800, height=600)

plotPCA(vsd, intgroup='condition')

dev.off()

Volcano (EnhancedVolcano)

png('results/volcano.png', width=1200, height=1000)

EnhancedVolcano(res,

lab = rownames(res),

x = 'log2FoldChange',

y = 'padj',

pCutoff = 0.05,

FCcutoff = 1.0)

dev.off()

Heatmap of top DEGs

topgenes <- head(order(res$padj), 50)

mat <- assay(vsd)[topgenes,]

mat <- mat - rowMeans(mat)

png('results/top50\_heatmap.png', width=1600, height=1200)

pheatmap(mat, annotation\_col = samples)

dev.off()

library(clusterProfiler)

sig\_genes\_kegg <- c('sau:SAUSA300\_0001','sau:SAUSA300\_0002', ... ) # KEGG IDs or Entrez

enrichKEGG example

kegg\_res <- enrichKEGG(gene = sig\_genes\_kegg, organism = 'sau', keyType = 'kegg')

write.csv(as.data.frame(kegg\_res), 'results/enrichKEGG.csv')

If GO terms available as gene -> GO mapping, use enricher/enrichGO with custom annotation

create bedgraph + bigWig for IGV

mkdir -p coverage

for bam in alignments/.sorted.bam; do

sample=$(basename $bam .sorted.bam)

bedtools genomecov -ibam $bam -bg > coverage/${sample}.bg

convert to bigWig (requires chromosome sizes file)

bedGraphToBigWig coverage/${sample}.bg ref/chrom.sizes coverage/${sample}.bw

done