数据处理部分代码

1. SSH 连接和环境设置

```
ssh .....  # 连接到服务器
cd /lustre1  # 切换到指定目录
conda activate RNA-seq  # 激活 RNA-seq 环境
```

功能：连接到服务器并进入工作目录，激活特定的 Conda 环境。
参数：
ssh .....: SSH 连接命令，省略的部分通常是用户名和主机地址。
cd: 更改目录命令，进入指定的文件夹。
conda activate: 激活指定的 Conda 环境（在此为 RNA-seq）。

2. 安装所需的程序（使用 Anaconda）

```
# conda create -N RNA-seq
# conda install -n RNA-seq -c bioconda fastqc
# conda install -n RNA-seq -c bioconda fastp
# conda install -n RNA-seq -c bioconda multiqc
# conda install -n RNA-seq -c bioconda star
# conda install -n RNA-seq -c bioconda samtools
# conda install -n RNA-seq -c bioconda deeptools
# conda install -n RNA-seq -c bioconda salmon
# conda install -n RNA-seq -c bioconda qualimap
```

功能：创建 Conda 环境并安装 RNA-seq 数据分析所需的软件。
参数：
-N RNA-seq: 指定新环境的名称为 RNA-seq。
-n RNA-seq: 指定操作的 Conda 环境。
-c bioconda: 从 Bioconda 通道安装软件包。

3. 运行 FastQC

```
fastqc <sample>.fastq.gz -d . -o .
```

功能：对给定的 FASTQ 文件运行质量控制。
参数：
<sample>.fastq.gz: 输入的 FASTQ 文件名（使用实际文件名替代）。
-d .: 指定输出目录，这里是当前目录。
-o .: 同样指定输出目录为当前目录。

4. 批量运行 FastQC

find . -name "*.fastq.gz" -exec fastqc {} -d . -o . \;
功能：查找当前目录及子目录中所有 .fastq.gz 文件，并对每个文件运行 FastQC。
参数：
find .：在当前目录查找文件。
-name "*.fastq.gz"：查找所有以 .fastq.gz 结尾的文件。
-exec fastqc {} -d . -o . \;：对找到的每个文件执行 FastQC，{} 是查找到的文件的占位符。

5. 统计总读取数

totalreads=$(unzip -c HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2_fastqc.zip */fastqc_data.txt | grep 'Total Sequences' | cut -f 2)
echo "Total Sequences: $totalreads"
功能：统计并输出总读取数。
参数：
unzip -c：解压缩命令，-c 选项将内容输出到标准输出。
grep 'Total Sequences'：查找包含 "Total Sequences" 的行。
cut -f 2：从查找结果中提取第二列（读取数）。
echo：输出结果。

6. 处理单个样本的 FastP

fastp -i HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.fastq.gz -I HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2.fastq.gz -o HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.trimmed.fastq.gz -O HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2.trimmed.fastq.gz --detect_adapter_for_pe -l 25 -j HBR_Rep1.fastp.json -h HBR_Rep1.fastp.html

功能：使用 FastP 进行数据清理和修剪。
参数：
-i：指定输入的读段 1 文件。
-I：指定输入的读段 2 文件（配对读段）。
-o：指定输出的修剪后的读段 1 文件。
-O：指定输出的修剪后的读段 2 文件。
--detect_adapter_for_pe：自动检测配对读取的接头序列。
-l 25：设置输出读取的最小长度为 25。
-j 和 -h：指定输出 JSON 和 HTML 报告文件名。

7. 批量处理 FastP

```
for file in *.read1.fastq.gz
do
    base=$(basename "$file" .read1.fastq.gz)
    read1_file="${base}.read1.fastq.gz"
    read2_file="${base}.read2.fastq.gz"
```

```
    output_read1="${base}.read1.trimmed.fastq.gz"
    output_read2="${base}.read2.trimmed.fastq.gz"
    json_report="${base}.fastp.json"
    html_report="${base}.fastp.html"
    fastp -i "$read1_file" -I "$read2_file" -o "$output_read1" -O "$output_read2" --detect_adapter_for_pe -l 25 -j "$json_report" -h "$html_report"
done
```

功能：批量处理所有匹配的 FASTQ 文件。
参数：
for file in *.read1.fastq.gz: 循环遍历所有以 .read1.fastq.gz 结尾的文件。
basename "$file" .read1.fastq.gz: 获取不包含扩展名的文件基名。
其他变量用于构建输入输出文件名。

8. 生成基因组索引（STAR）

```
# STAR --runThreadN 4 --runMode genomeGenerate --genomeDir $GENOMEDIR --genomeFastaFiles $GENOMEDIR/GCA_000001405.15_GRCh38_no_alt_analysis_set.fna --sjdbGTFfile gencode.v36.annotation.gtf --sjdbOverhang readlength -1
```

功能：使用 STAR 生成基因组索引。
参数：
--runThreadN 4: 使用 4 个线程进行并行处理。
--runMode genomeGenerate: 指定运行模式为生成基因组索引。
--genomeDir: 指定索引文件的输出目录。
--genomeFastaFiles: 指定基因组的 FASTA 文件。
--sjdbGTFfile: 指定 GTF 文件，用于提供剪接信息。
--sjdbOverhang: 指定剪接数据库的过 hang 长度，通常为读取长度减去 1。

9. 运行 STAR 对齐

```
STAR --runThreadN 4 --genomeDir $GENOMEDIR --readFilesIn HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read1.trimmed.fastq.gz HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22.read2.trimmed.fastq.gz --outFileNamePrefix HBR_Rep1_ERCC-Mix2_Build37-ErccTranscripts-chr22 --readFilesCommand zcat --outSAMtype BAM Unsorted --outFilterType BySJout --alignSJoverhangMin 8 --outFilterMultimapNmax 20 --alignSJDBoverhangMin 1 --outFilterMismatchNmax 999 --outFilterMismatchNoverReadLmax 0.04 --alignIntronMin 20 --alignIntronMax 1000000 --alignMatesGapMax 1000000 --quantMode TranscriptomeSAM --outSAMattributes NH HI AS NM MD
```

功能：使用 STAR 将修剪后的读取对齐到参考基因组。
参数：
--runThreadN 4: 使用 4 个线程。

--genomeDir: 指定生成的基因组索引目录。
--readFilesIn: 指定输入的配对 FASTQ 文件。
--outFileNamePrefix: 指定输出文件的前缀。
--readFilesCommand zcat: 使用 zcat 解压缩输入的 FASTQ 文件。
--outSAMtype BAM Unsorted: 输出格式为未排序的 BAM 文件。
--outFilterType BySJout: 根据剪接位点过滤输出。
--alignSJoverhangMin: 设置有效剪接的最小悬挂长度。
--outFilterMultimapNmax: 最大允许的多重比对数量。
--alignIntronMin 和 --alignIntronMax: 设置内含子的最小和最大长度。
--alignMatesGapMax: 设置配对读取的最大间距。
--quantMode TranscriptomeSAM: 量化模式，生成适合转录组的 SAM 文件。
--outSAMattributes: 指定输出 SAM 文件中包含的属性。

10. 合并 BAM 文件

samtools merge /lustre1/share/RNA_seq/genome/merged_output.bam /lustre1/share/RNA_seq/genome/*Aligned.out.bam

功能：合并多个 BAM 文件。
参数：
merged_output.bam: 合并后的输出文件名。
*Aligned.out.bam: 所有要合并的输入文件（匹配模式）。

11. 运行 MultiQC

multiqc /lustre1/share/RNA_seq/

功能：运行 MultiQC 以汇总和报告 RNA-seq 数据质量控制结果。

12. 使用 gffread 提取转录本序列

gffread -w GRCh38_no_alt_analysis_set_gencode.v36.transcripts.fa -g GCA_000001405.15_GRCh38_no_alt_analysis_set.fna gencode.v36.annotation.gtf

功能：从 GTF 文件中提取转录本序列并保存为 FASTA 格式。
参数：
-w: 指定输出的 FASTA 文件名。
-g: 指定参考基因组的 FASTA 文件。
gencode.v36.annotation.gtf: 输入的 GTF 文件。

13. 使用 Salmon 进行量化

salmon quant -t /lustre1/share/RNA_seq/genome/GRCh38_no_alt_analysis_set_gencode.v36.transcripts.fa --

libType A -a UHR_Rep3_ERCC-Mix1_Build37-ErccTranscripts-chr22_ERCC-Mix2_Build37-ErccTranscripts-chr22Aligned.toTranscriptome.out.bam -o UHR_Rep3_ERCC-Mix1_Build37-ErccTranscripts-chr22_ERCC-Mix2_Build37-ErccTranscripts-chr22.salmon_quant --gcBias --seqBias
功能：使用 Salmon 进行转录组量化。
参数：
-t: 指定转录本 FASTA 文件。
--libType A: 指定文库类型（例如，单端或双端）。
-a: 指定输入 BAM 文件。
-o: 指定输出文件夹的名称。
--gcBias: 纠正 GC 偏倚。
--seqBias: 纠正序列偏倚。

14. 创建批处理脚本用于 Salmon 量化

vi batch_salmon.sh

```
#!/bin/bash
TRANSCRIPTOME="genome/GRCh38_no_alt_analysis_set_gencode.v36.transcripts.fa"

for bam_file in *.Aligned.toTranscriptome.out.bam; do
    sample_name=$(basename "$bam_file" Aligned.toTranscriptome.out.bam)
    salmon quant -t "$TRANSCRIPTOME" --libType A -a "$bam_file" -o "${sample_name}.salmon_quant" --gcBias --seqBias
    echo "Processed: $sample_name"
done
```

R 部分代码

1. 安装和加载必要的 R 包

```
library(readr)        # 用于读取数据
library(edgeR)        # 差异表达分析
library(limma)        # 用于线性模型和差异表达分析
library(ggplot2)      # 数据可视化
library(tximport)     # 用于转录组数据导入
library(ggrepel)      # 用于火山图
library(biomaRt)      # 用于生物信息学数据查询
library(pheatmap)     # 热图绘制
library(clusterProfiler)   # 功能富集分析
```

```
library(org.Hs.eg.db)  # 用于基因注释
```

目的：加载需要的 R 包，以便在 RNA-seq 分析中使用。
参数：每个 library() 函数的参数是要加载的包名。

2. 创建 DGEList 对象

```
dge <- DGEList(counts = txi$counts)
```

目的：创建一个 DGEList 对象，用于存储计数数据，以便进行差异表达分析。
参数：
counts: 传入的计数矩阵（txi$counts），由 tximport 提供。

3. 设置过滤阈值

```
counts_threshold <- 10
samples_threshold <- 2
```

目的：设置计数和样本的过滤阈值，以排除低表达基因。
参数：
counts_threshold: 基因计数的最低阈值。
samples_threshold: 至少有多少个样本需要达到此计数阈值。

4. 过滤低表达基因

```
keep <- rowSums(dge$counts >= counts_threshold) >= samples_threshold
dge_filtered <- dge[keep, ]
```

目的：根据设定的阈值过滤 DGEList 对象中的基因。
参数：
rowSums(dge$counts >= counts_threshold): 计算每个基因在满足计数阈值的样本中出现的次数。
dge[keep,]: 根据 keep 向量保留符合条件的基因。

5. 创建样本信息数据框

```
sample_info <- data.frame(
  sample = colnames(dge_filtered),
  condition = factor(c(rep("HBR", 3), rep("UHR", 3)))  # 根据样本分组修改
)
```

目的：创建一个数据框以存储样本信息，包括样本名称和条件（组别）。
参数：
colnames(dge_filtered): 从过滤后的 DGEList 中提取样本名称。
factor(...): 创建一个因子变量，以表示样本的条件。

6. 设计矩阵

design <- model.matrix(~ condition, data = sample_info)
目的：创建一个设计矩阵，用于差异表达分析。
参数：
~ condition: 矩阵的模型公式，表示使用 condition 作为解释变量。
data: 指定数据框 sample_info。

7. 差异表达分析

dge_filtered <- estimateDisp(dge_filtered, design)
fit <- glmFit(dge_filtered, design)
results <- glmLRT(fit)
目的：使用线性模型分析差异表达基因。
参数：
estimateDisp(...): 估计离散度。
glmFit(...): 拟合广义线性模型。
glmLRT(fit): 执行似然比检验以获得差异表达的结果。

8. 火山图绘制

```
# 创建火山图数据
volcano_data <- data.frame(
  logFC = results$table$logFC,
  PValue = results$table$PValue,
  row.names = rownames(results$table)
)

# 添加显著性标记
volcano_data$significant <- ifelse(volcano_data$PValue < 0.05 & abs(volcano_data$logFC) >
1, "yes", "no")

# 绘制火山图
ggplot(volcano_data, aes(x = logFC, y = -log10(PValue), color = significant)) +
  geom_point(alpha = 0.6, size = 2) +
  scale_color_manual(values = c("no" = "grey", "yes" = "red")) +
  theme_minimal() +
  labs(title = "Volcano Plot", x = "Log Fold Change", y = "-Log10 P-value") +
  theme(plot.title = element_text(hjust = 0.5))
```

代码解释：
volcano_data <- data.frame(

```
  logFC = results$table$logFC,
  PValue = results$table$PValue,
  row.names = rownames(results$table)
)
```

目的：构建用于火山图的数据框，包括每个基因的对数变化（Log Fold Change）和 P 值。
参数：
logFC: 从差异分析结果中提取的对数变化值。
PValue: 从差异分析结果中提取的 P 值。
row.names: 将基因名作为行名。

添加显著性标记

```
volcano_data$significant <- ifelse(volcano_data$PValue < 0.05 & abs(volcano_data$logFC) > 1, "yes", "no")
```

目的：为显著基因添加标记，判断哪些基因显著（P 值小于 0.05 且绝对对数变化大于 1）。
参数：
ifelse(...): 条件语句，基于 P 值和对数变化值进行判断。

绘制火山图

```
    ggplot(volcano_data, aes(x = logFC, y = -log10(PValue), color = significant)) +
      geom_point(alpha = 0.6, size = 2) +
      scale_color_manual(values = c("no" = "grey", "yes" = "red")) +
      theme_minimal() +
      labs(title = "Volcano Plot", x = "Log Fold Change", y = "-Log10 P-value") +
      theme(plot.title = element_text(hjust = 0.5))
```

目的：使用 ggplot2 绘制火山图，展示基因表达的显著性。
参数：
aes(...): 指定 x 轴、y 轴和颜色美学。
geom_point(...): 绘制散点图，alpha 控制点的透明度，size 控制点的大小。
scale_color_manual(...): 自定义颜色映射，灰色表示不显著，红色表示显著。
theme_minimal(): 使用简单的主题样式。
labs(...): 添加标题和轴标签。
theme(...): 调整标题位置。

9. 热图绘制

```
# 选择前 50 个差异表达基因
top_genes <- head(order(results$table$PValue), 50)
heatmap_data <- dge_filtered$counts[top_genes, ]

# 创建热图
pheatmap(heatmap_data,
         cluster_rows = TRUE,
```

```
        cluster_cols = TRUE,
        scale = "row",
        show_rownames = TRUE,
        show_colnames = TRUE,
        main = "Heatmap of Top 50 Differentially Expressed Genes")
```

代码解释：
选择前 50 个差异表达基因
```
top_genes <- head(order(results$table$PValue), 50)
heatmap_data <- dge_filtered$counts[top_genes, ]
```

目的：获取 P 值最小的前 50 个基因，用于热图展示。
参数：
order(results$table$PValue): 对 P 值进行排序。
head(..., 50): 获取排序后的前 50 个基因索引。
dge_filtered$counts[top_genes,]: 提取对应基因的计数数据。

创建热图
```
pheatmap(heatmap_data,
        cluster_rows = TRUE,
        cluster_cols = TRUE,
        scale = "row",
        show_rownames = TRUE,
        show_colnames = TRUE,
        main = "Heatmap of Top 50 Differentially Expressed Genes")
```

目的：绘制热图以展示前 50 个差异表达基因的表达模式。
参数：
cluster_rows = TRUE: 行进行聚类。
cluster_cols = TRUE: 列进行聚类。
scale = "row": 对行数据进行标准化（均值为 0，方差为 1）。
show_rownames = TRUE: 显示行名称（基因名）。
show_colnames = TRUE: 显示列名称（样本名）。
main: 添加热图标题。