RNA-seq Analysis Workflow

1. Installation and Package Loading

CRAN Packages:

- **ggplot2**: Advanced data visualization
- **pheatmap**: Creation of heatmaps
- **RColorBrewer**: Color palettes for visualizations
- **readr**: Efficient reading and writing of CSV files
- **dplyr**: Data manipulation and transformation
- **matrixStats**: Matrix operations and statistics
- **vegan**: Community ecology analyses (e.g., PERMANOVA)

Bioconductor Packages:

- **DESeq2**: Differential expression analysis
- **ashr**: Adaptive shrinkage of log-fold changes

Installation and Loading:

```r
# Install CRAN packages
cran_packages <- c("ggplot2", "pheatmap", "RColorBrewer", "readr", "dplyr",
"matrixStats", "vegan")
for (pkg in cran_packages) {
  if (!requireNamespace(pkg, quietly = TRUE)) install.packages(pkg)
}

# Install Bioconductor packages
if (!requireNamespace("BiocManager", quietly = TRUE))
install.packages("BiocManager")
bioc_packages <- c("DESeq2", "ashr")
for (pkg in bioc_packages) {
  if (!requireNamespace(pkg, quietly = TRUE)) BiocManager::install(pkg)
}

# Load libraries
library(DESeq2)
library(ggplot2)
library(pheatmap)
library(RColorBrewer)
library(ashr)
library(readr)
library(dplyr)
library(matrixStats)
library(vegan)
```

2. Set Working Directory & Output Folder

Set your working directory to your data location and create a directory for outputs:

```
setwd("C:/Users/newfaculty/Desktop/Bioinformatics_Club")
plot_dir <- "DESeq2_Plots"
dir.create(plot_dir, showWarnings = FALSE)
```

3. Load Data

```
counts <- read.csv("raw_counts.csv", row.names = 1)
meta <- read.csv("MetaData.csv", row.names = 1)
```

4. Data Cleaning

Ensure the metadata and counts data match:

```
colnames(meta) <- trimws(colnames(meta))
meta$biopsy_site <- meta$Factor.Value.biopsy.site.
counts <- counts[, rownames(meta)]
stopifnot(all(colnames(counts) == rownames(meta)))
```

5. Filter Genes

Remove low-expression genes to reduce noise:

```
keep <- rowSums(counts >= 10) >= 2
counts <- counts[keep, ]
message("Genes retained: ", nrow(counts))
```

6. Grouping Factors

Define groups for comparisons clearly:

```
meta$biopsy_site <- factor(meta$biopsy_site,
                           levels = c("normal", "primary tumor", "colorectal
cancer metastatic in the liver"))
```

7. DESeq2 Analysis

Perform differential expression analysis:

```
dds <- DESeqDataSetFromMatrix(countData = round(counts), colData = meta, design
= ~ biopsy_site)
dds <- DESeq(dds)
```

8. Variance Stabilizing Transformation (VST)

Normalize counts for visualization:

```
vsd <- vst(dds, blind = FALSE)
norm_counts <- assay(vsd)
write.csv(norm_counts, "Normalized_Counts.csv")
```

9. PCA Visualization

Perform PCA to visualize sample clustering:

```
pcaData <- plotPCA(vsd, intgroup = "biopsy_site", returnData = TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"))

png(file.path(plot_dir, "PCA_plot.png"), width = 1200, height = 900)
ggplot(pcaData, aes(PC1, PC2, color = biopsy_site)) +
  geom_point(size = 4, alpha = 0.8) +
  labs(x = paste0("PC1: ", percentVar[1], "% variance"),
       y = paste0("PC2: ", percentVar[2], "% variance"),
       title = "PCA by Biopsy Site") +
  theme_minimal(base_size = 16) +
  scale_color_brewer(palette = "Set1") +
  theme(plot.title = element_text(hjust = 0.5))
dev.off()
```

10. PERMANOVA

Test statistical separation of groups:

```r
dist_matrix <- dist(t(assay(vsd)))
adonis_result <- adonis2(dist_matrix ~ biopsy_site, data = meta, permutations =
4999)
sink("PERMANOVA_results.txt")
print(adonis_result)
sink()
```

11. Differential Expression Results

Analyze differential expression for selected comparisons:

```r
res_primary <- results(dds, contrast = c("biopsy_site", "primary tumor",
"normal"))
res_primary <- lfcShrink(dds, coef = "biopsy_site_primary.tumor_vs_normal",
type = "ashr", res = res_primary)
write.csv(as.data.frame(res_primary[order(res_primary$pvalue), ]),
"Primary_vs_Normal_DEGs.csv")
```

12. Visualization: MA & Volcano Plots

Generate MA and Volcano plots:

```r
# MA Plot
png(file.path(plot_dir, "MA_plot_Primary_vs_Normal.png"), width = 1200, height
= 900)
plotMA(res_primary, ylim = c(-5, 5), main = "MA Plot: Primary vs Normal")
dev.off()

# Volcano Plot
volcano_plot <- function(res, title, filename) {
  df <- na.omit(as.data.frame(res))
  df$threshold <- ifelse(df$padj < 0.05 & abs(df$log2FoldChange) > 1,
                         ifelse(df$log2FoldChange > 1, "Up", "Down"), "NS")
  png(filename, width = 1200, height = 900)
  ggplot(df, aes(log2FoldChange, -log10(padj), color = threshold)) +
    geom_point(alpha = 0.7, size = 2) +
    scale_color_manual(values = c("Up" = "green", "Down" = "red", "NS" =
"gray")) +
    geom_vline(xintercept = c(-1, 1), lty = 2, col = "blue") +
    geom_hline(yintercept = -log10(0.05), lty = 2, col = "blue") +
    labs(title = title, x = "Log2 Fold Change", y = "-Log10 Adjusted P-value") +
    theme_minimal(base_size = 16)
  dev.off()
}
volcano_plot(res_primary, "Volcano: Primary vs Normal", file.path(plot_dir,
"Volcano_Primary_vs_Normal.png"))
```