Data_exploration

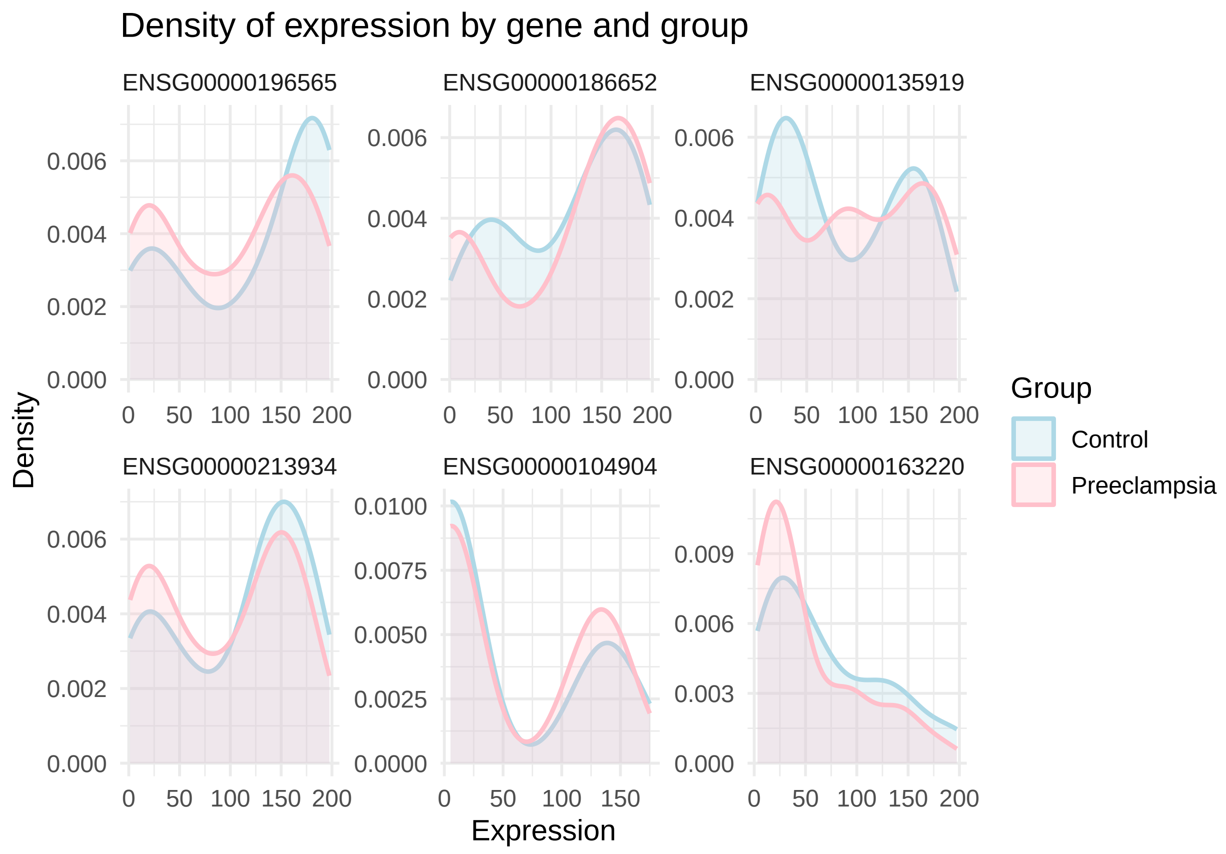
2025-09-22

```r
#installing libraries/packages


options(repos = c(CRAN = "https://cloud.r-project.org"))
if (requireNamespace("BiocManager", quietly = TRUE)) {
  options(repos = BiocManager::repositories())
}


library(readr); library(dplyr); library(ggplot2)
library(Rtsne); library(uwot); library(pheatmap); library(gprofiler2)
library(limma)
```

```r
#loading data and meta data
#reading raw TSV file
tsv_path <- "C:/Users/anae2/Preeclampsia_Vs_Normal/refine_bio_data/GSE60438/GSE60438.tsv"
raw_data <- read_tsv(tsv_path)
gene_ids <- raw_data[[1]]
sample_names <- colnames(raw_data)[-1]

expr_mat <- raw_data %>%
  select(-1) %>%                          # drop the Gene column
  mutate(across(everything(), as.numeric)) %>%
  as.matrix()

rownames(expr_mat) <- gene_ids
colnames(expr_mat) <- sample_names

#reading meta data
meta_path <- "C:/Users/anae2/Preeclampsia_Vs_Normal/refine_bio_data/GSE60438/metadata_GSE60438.tsv"
meta <- readr::read_tsv(meta_path, show_col_types = FALSE)

group_raw <- tolower(paste(
  meta$refinebio_disease_stage,
  meta$`characteristics_ch1_subject status`,
  meta$title,
  meta$description,
  sep = " | "
))

pheno <- tibble::tibble(
  sample = meta$refinebio_accession_code,
  Group  = dplyr::case_when(
    grepl("preeclamp|pre[- ]eclamp|\\bpe\\b", group_raw) ~ "Preeclampsia",
    grepl("normotensive|control", group_raw)              ~ "Control",
    TRUE ~ NA_character_
  )
) %>%
  dplyr::filter(!is.na(sample), !is.na(Group)) %>%
  dplyr::distinct(sample, .keep_all = TRUE)

# Align expression to pheno
expr_mat <- expr_mat[, pheno$sample, drop = FALSE]
stopifnot(all(colnames(expr_mat) == pheno$sample))

# Colors and factors used to graphing
grp_cols <- c(Control = "lightblue", Preeclampsia = "pink")
pheno$Group <- factor(pheno$Group, levels = c("Control", "Preeclampsia"))


#plots
```

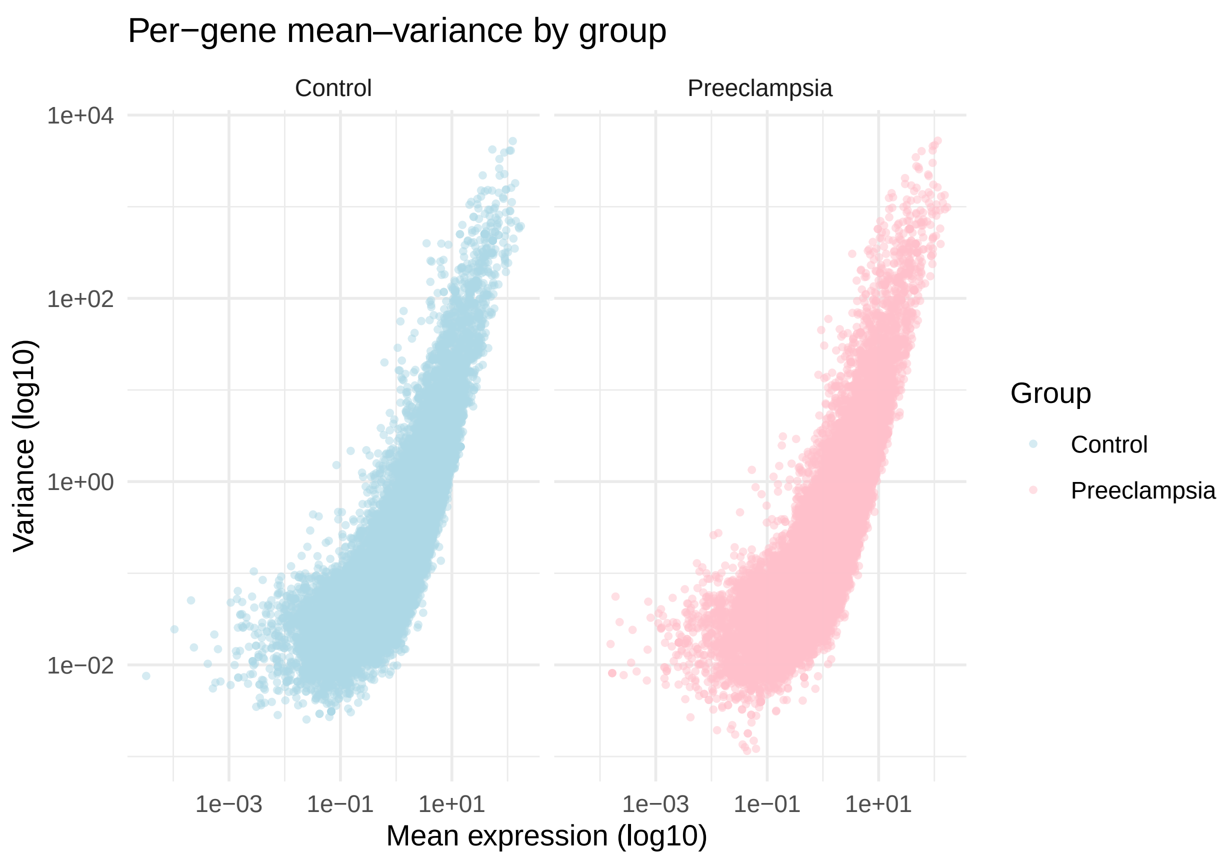
```r
#density plots with color per group

# Map rownames (ENSEMBL IDs) to HGNC symbols and update expr_mat rownames
ens_ids  <- rownames(expr_mat)
ens_core <- sub("\\.\\d+$", "", ens_ids)

mp <- gconvert(ens_core, organism = "hsapiens", target = "SYMBOL", filter_na = FALSE)
map_named   <- setNames(mp$name, mp$input)
symbol_vec  <- map_named[ens_core]
missing     <- is.na(symbol_vec) | symbol_vec == ""
symbol_vec[missing] <- ens_ids[missing]                 # fallback to Ensembl
gene_symbol <- make.unique(symbol_vec)                  # ensure uniqueness

# Keep Ensembl as rownames (stable), but store a lookup from Ensembl -> Symbol
names(gene_symbol) <- ens_ids
gene_lookup <- gene_symbol

# fallback to ENSG
rownames(expr_mat) <- make.unique(symbol_vec)
# ensure each is unique

# Choose top 6 by variance to visualize
vars      <- apply(expr_mat, 1, var, na.rm = TRUE)
top_ids   <- names(sort(vars, decreasing = TRUE))[1:min(6, nrow(expr_mat))]
top_names <- unname(gene_lookup[top_ids])

# Long format for density plots
library(dplyr); library(tidyr); library(ggplot2)
df_long <- as.data.frame(t(expr_mat[top_ids, , drop = FALSE]))  # samples x selected genes (Ensembl col
df_long$sample <- rownames(df_long)
df_long <- df_long %>%
  left_join(pheno, by = "sample") %>%
  pivot_longer(cols = all_of(top_ids), names_to = "Ensembl", values_to = "Expr") %>%
  mutate(Gene = factor(gene_lookup[Ensembl], levels = unique(top_names)))
```

```r
# Density plots of expression per gene, colored by Group and faceted
ggplot(df_long, aes(x = Expr, color = Group, fill = Group)) +
  geom_density(alpha = 0.25, linewidth = 0.8) +
  facet_wrap(~ Gene, scales = "free") +
  scale_color_manual(values = grp_cols, drop = FALSE) +
  scale_fill_manual(values = grp_cols, drop = FALSE) +
  labs(title = "Density of expression by gene and group",
       x = "Expression", y = "Density", color = "Group", fill = "Group") +
  theme_minimal()
```

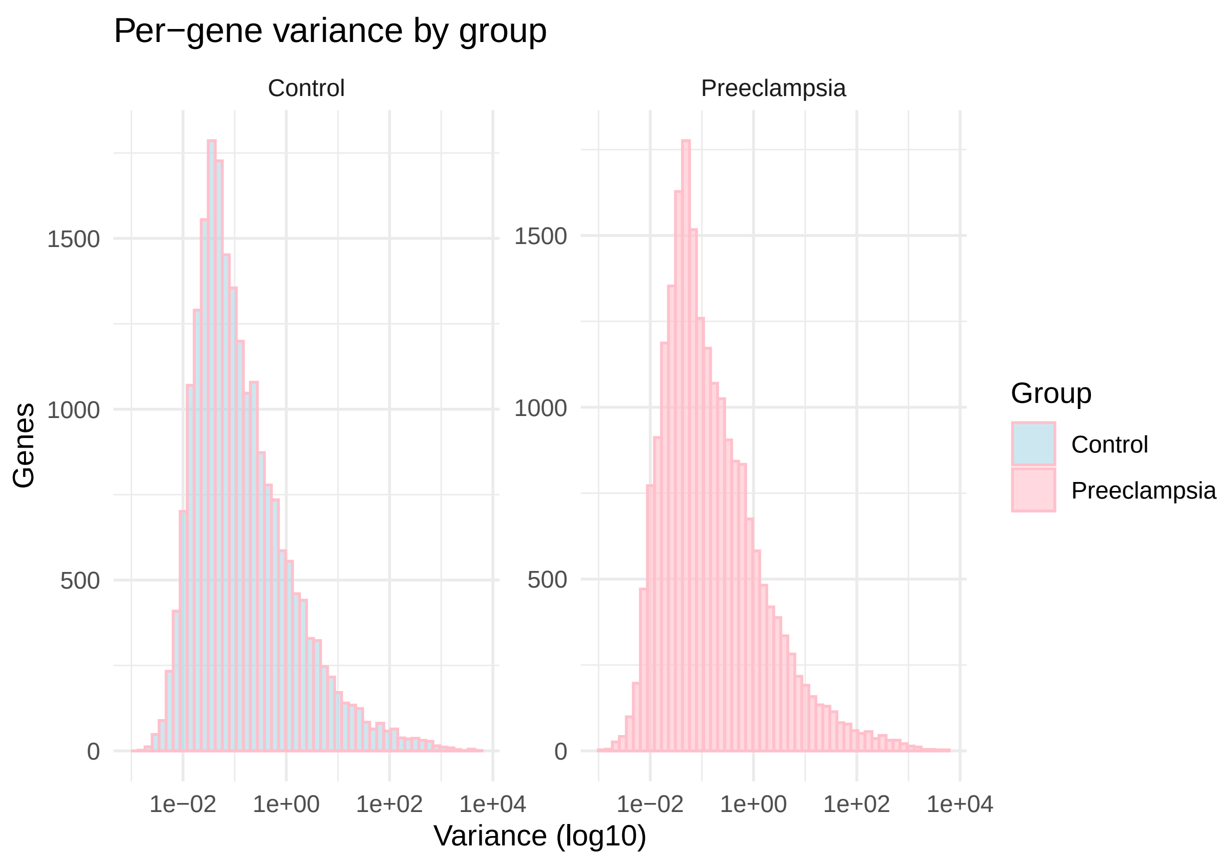


```r
#a helper to plot any single gene by symbol
plot_gene_density <- function(gene_symbol) {
  stopifnot(gene_symbol %in% rownames(expr_mat))
  d <- data.frame(Expr = as.numeric(expr_mat[gene_symbol, ]),
                  sample = colnames(expr_mat)) %>%
       left_join(pheno, by = "sample")
  ggplot(d, aes(x = Expr, color = Group, fill = Group)) +
    geom_density(alpha = 0.25, size = 0.8) +
    scale_color_manual(values = grp_cols, drop = FALSE) +
    scale_fill_manual(values = grp_cols, drop = FALSE) +
    labs(title = paste("Expression density:", gene_symbol),
         x = "Expression", y = "Density", color = "Group", fill = "Group") +
    theme_minimal()
}
```

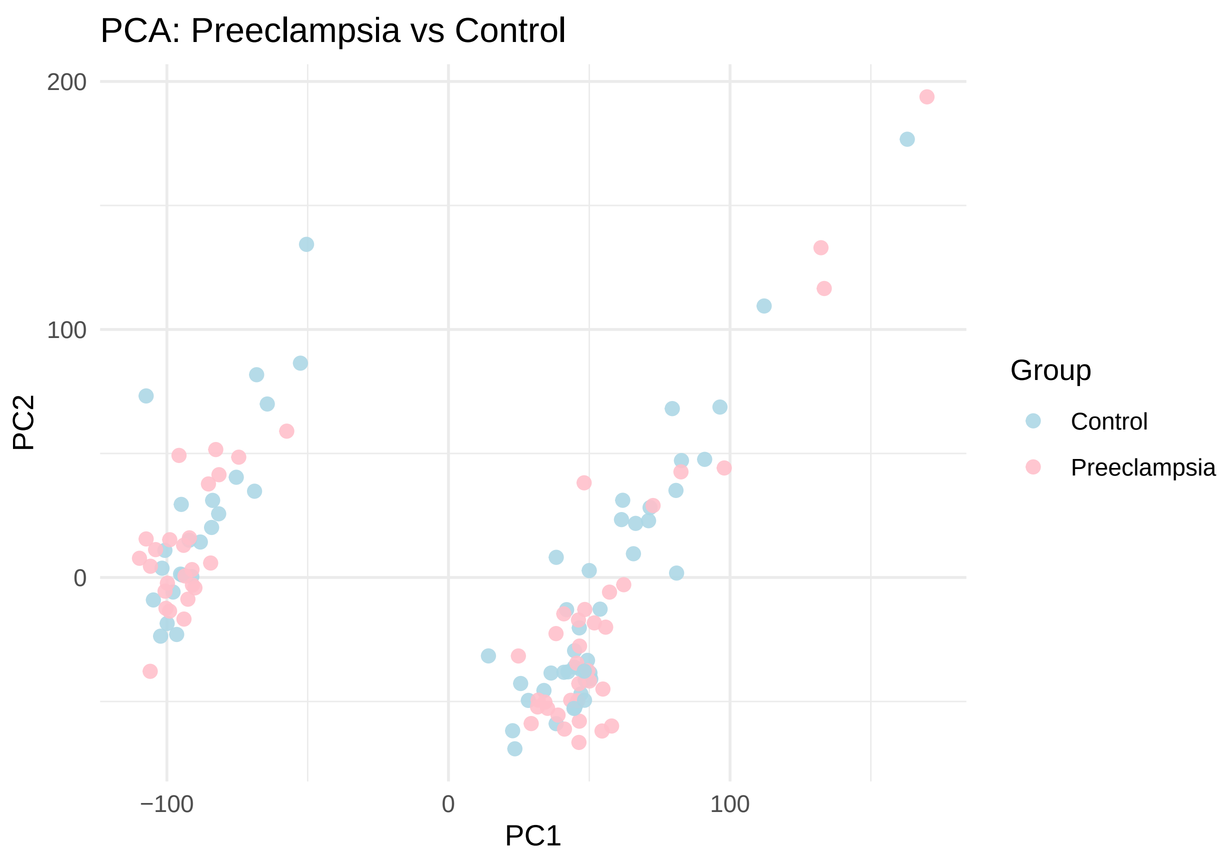
```
#calcuate mean per group
calc_stats <- function(g) {
  idx <- which(pheno$Group == g)
  X   <- expr_mat[, pheno$sample[idx], drop = FALSE]          # genes x samples (that group)
  data.frame(
    gene = rownames(expr_mat),
    Group = g,
    mean = rowMeans(X, na.rm = TRUE),
    var  = if (length(idx) > 1) apply(X, 1, var, na.rm = TRUE) else NA_real_
  )
}

groups <- levels(pheno$Group)
gene_stats_by_group <- do.call(rbind, lapply(groups, calc_stats))
gene_stats_by_group$sd <- sqrt(gene_stats_by_group$var)
gene_stats_by_group$cv <- gene_stats_by_group$sd / (abs(gene_stats_by_group$mean) + 1e-8)

# Mean-variance scatter plot
 plot_mv <- ggplot(gene_stats_by_group, aes(x = mean, y = var, color = Group)) +
  geom_point(alpha = 0.5, size = 0.8) +
  scale_x_log10() + scale_y_log10() +
  scale_color_manual(values = grp_cols, drop = FALSE) +
  labs(title = "Per-gene mean-variance by group",
       x = "Mean expression (log10)", y = "Variance (log10)", color = "Group") +
  theme_minimal() +
  facet_wrap(~ Group, nrow = 1)
print(plot_mv)
```



```r
#variance by group
ggplot(gene_stats_by_group, aes(x = var, fill = Group)) +
  geom_histogram(bins = 50, alpha = 0.6, color = "pink") +
  scale_x_log10() +
  scale_fill_manual(values = grp_cols, drop = FALSE) +
  labs(title = "Per-gene variance by group", x = "Variance (log10)", y = "Genes") +
  theme_minimal() +
  facet_wrap(~ Group, nrow = 1, scales = "free_y")
```



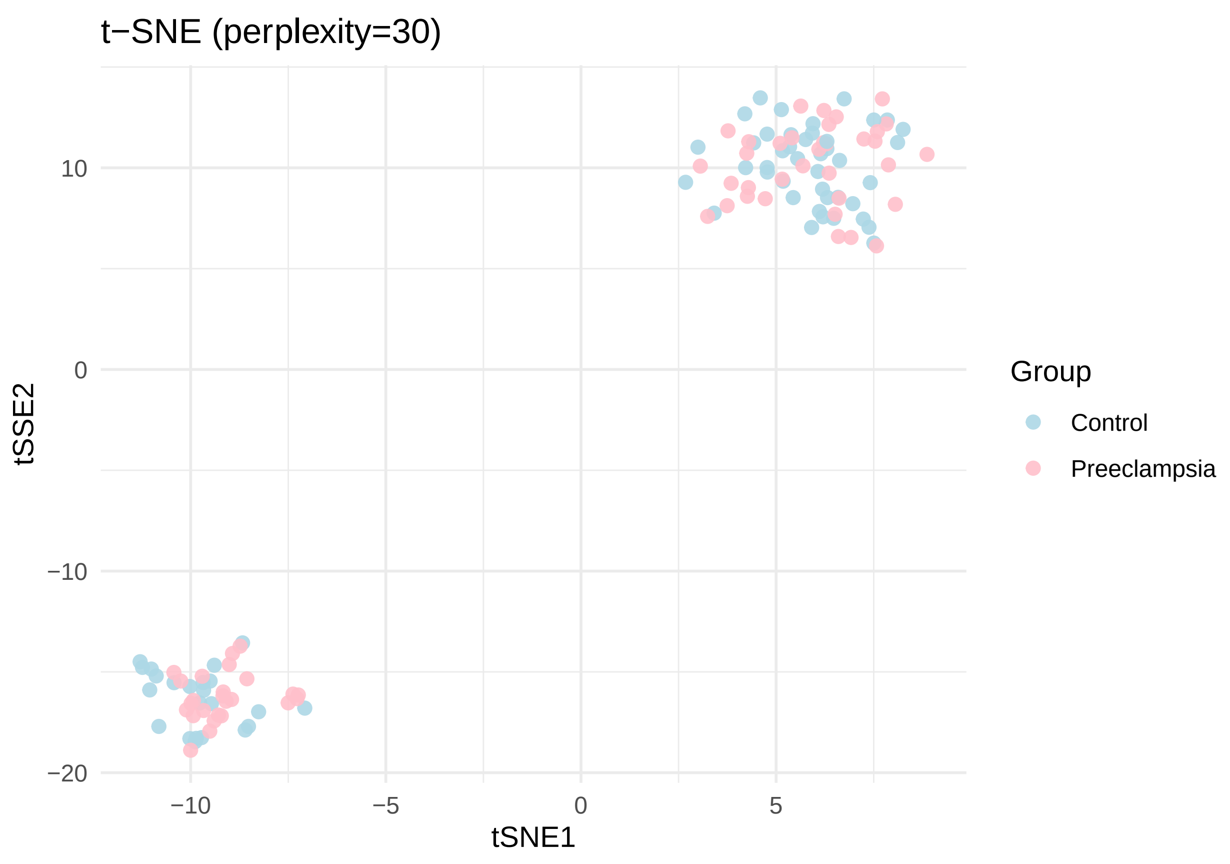
```r
# Create samples x genes matrix with gene-wise z-scoring
Xs <- scale(t(expr_mat), center = TRUE, scale = TRUE)  # rows = samples, cols = genes

# Drop non-informative or non-finite genes
keep <- apply(Xs, 2, function(v) sd(v, na.rm = TRUE) > 0 && all(is.finite(v)))
Xs <- Xs[, keep, drop = FALSE]


#PCA plot
pca <- prcomp(Xs, center = FALSE, scale. = FALSE)
pca_df <- data.frame(sample = pheno$sample, Group = pheno$Group,
                     PC1 = pca$x[,1], PC2 = pca$x[,2])
ggplot(pca_df, aes(PC1, PC2, color = Group)) +
  geom_point(size = 2, alpha = 0.9) +
  scale_color_manual(values = grp_cols) +
  labs(title = "PCA: Preeclampsia vs Control", color = "Group") +
  theme_minimal()
```



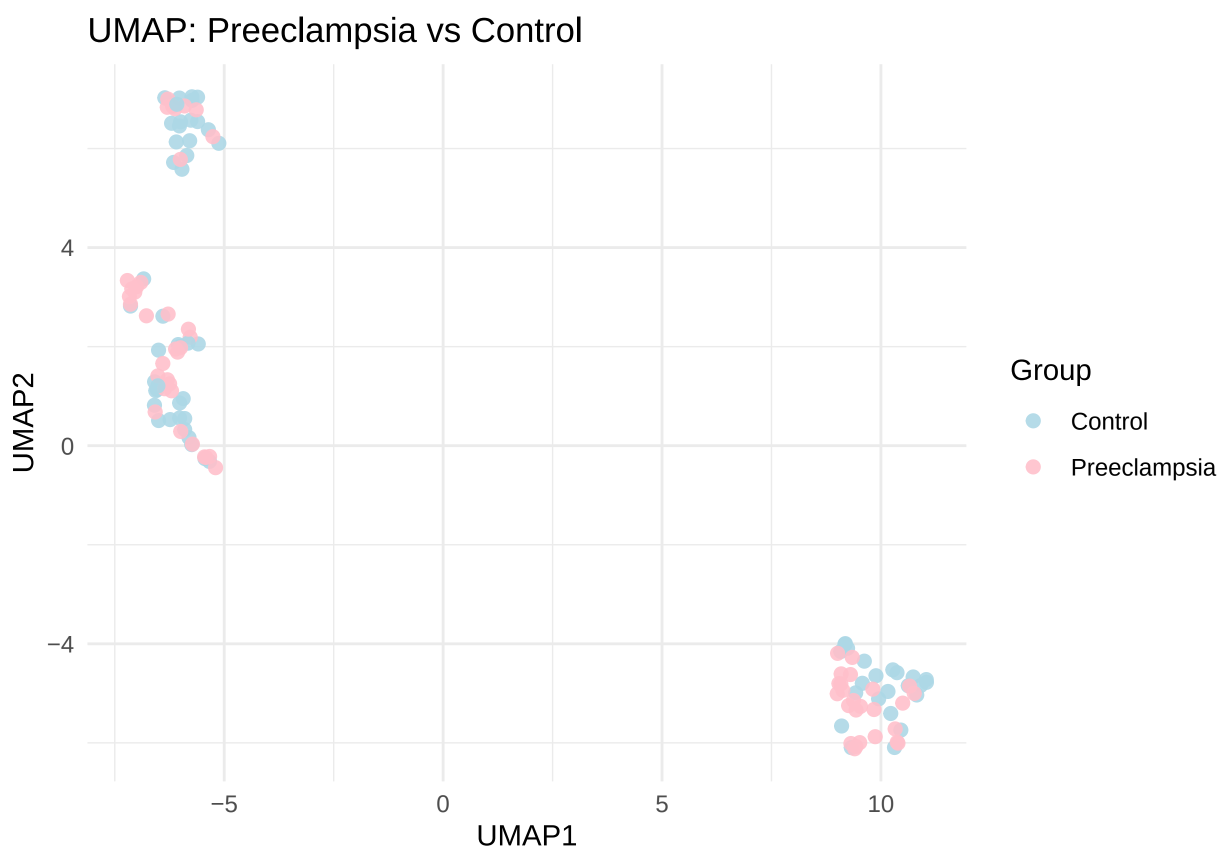
```
#t-SNE plot
set.seed(42)
perp <- max(5, min(30, floor(nrow(Xs)/3)))
tsne <- Rtsne(Xs, perplexity = perp, check_duplicates = FALSE, verbose = TRUE)
```

```
## Performing PCA
## Read the 125 x 50 data matrix successfully!
## OpenMP is working. 1 threads.
## Using no_dims = 2, perplexity = 30.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.02 seconds (sparsity = 0.859648)!
## Learning embedding...
## Iteration 50: error is 54.844717 (50 iterations in 0.01 seconds)
## Iteration 100: error is 53.560234 (50 iterations in 0.01 seconds)
## Iteration 150: error is 54.149935 (50 iterations in 0.01 seconds)
## Iteration 200: error is 58.029820 (50 iterations in 0.01 seconds)
## Iteration 250: error is 52.827567 (50 iterations in 0.01 seconds)
## Iteration 300: error is 1.467671 (50 iterations in 0.01 seconds)
## Iteration 350: error is 0.828529 (50 iterations in 0.01 seconds)
## Iteration 400: error is 0.374132 (50 iterations in 0.01 seconds)
## Iteration 450: error is 0.223383 (50 iterations in 0.01 seconds)
## Iteration 500: error is 0.220528 (50 iterations in 0.01 seconds)
## Iteration 550: error is 0.220750 (50 iterations in 0.01 seconds)
## Iteration 600: error is 0.222047 (50 iterations in 0.01 seconds)
## Iteration 650: error is 0.220861 (50 iterations in 0.01 seconds)
## Iteration 700: error is 0.220387 (50 iterations in 0.00 seconds)
## Iteration 750: error is 0.220369 (50 iterations in 0.01 seconds)
## Iteration 800: error is 0.220844 (50 iterations in 0.00 seconds)
## Iteration 850: error is 0.220216 (50 iterations in 0.01 seconds)
## Iteration 900: error is 0.220986 (50 iterations in 0.01 seconds)
## Iteration 950: error is 0.220906 (50 iterations in 0.01 seconds)
## Iteration 1000: error is 0.222004 (50 iterations in 0.01 seconds)
## Fitting performed in 0.18 seconds.
```

```
tsne_df <- data.frame(sample = pheno$sample, Group = pheno$Group,
                      tSNE1 = tsne$Y[,1], tSSE2 = tsne$Y[,2])
ggplot(tsne_df, aes(tSNE1, tSSE2, color = Group)) +
  geom_point(size = 2, alpha = 0.9) +
  scale_color_manual(values = grp_cols) +
  labs(title = paste0("t-SNE (perplexity=", perp, ")"), color = "Group") +
  theme_minimal()
```

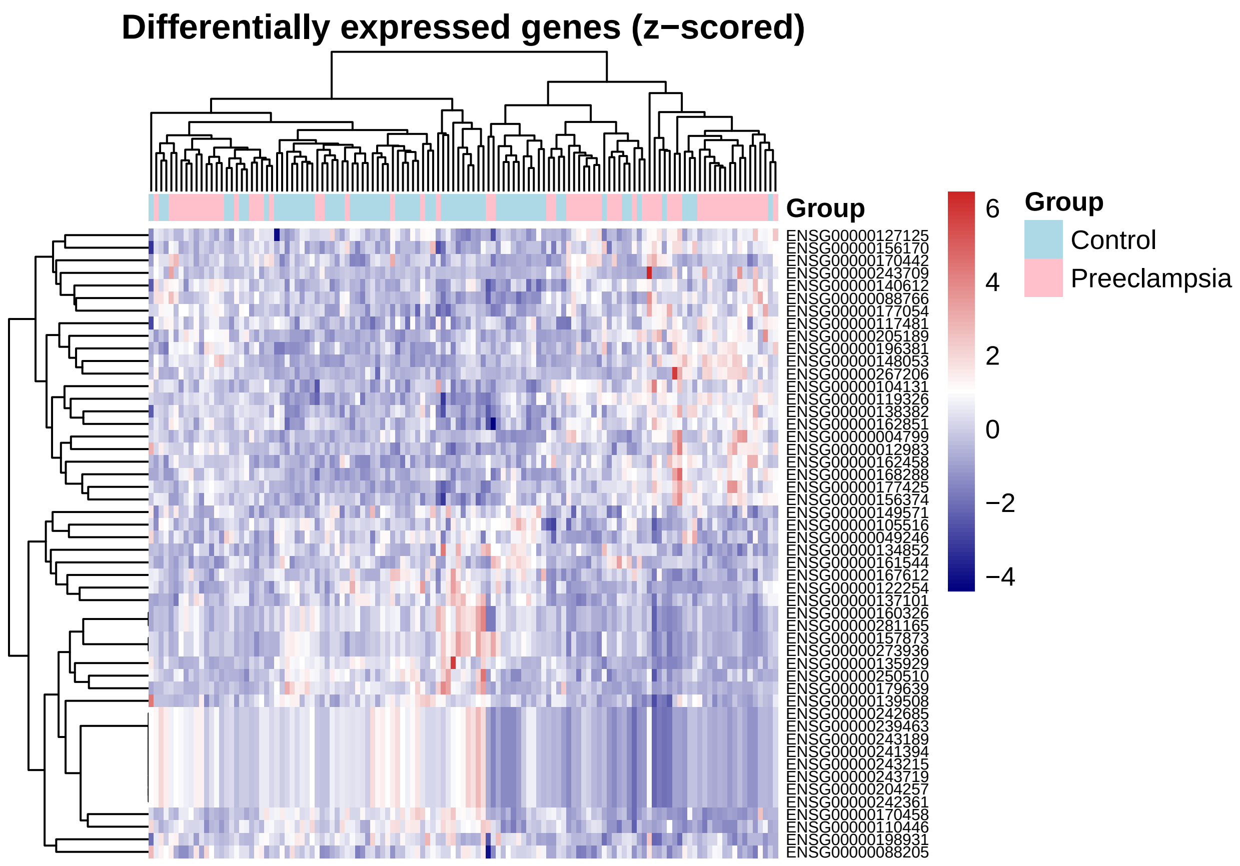


```
#UMAP plot
set.seed(42)
um <- umap(Xs, n_neighbors = min(15, max(2, nrow(Xs) - 1)), metric = "cosine")
umap_df <- data.frame(sample = pheno$sample, Group = pheno$Group,
                      UMAP1 = um[,1], UMAP2 = um[,2])
ggplot(umap_df, aes(UMAP1, UMAP2, color = Group)) +
  geom_point(size = 2, alpha = 0.9) +
  scale_color_manual(values = grp_cols) +
  labs(title = "UMAP: Preeclampsia vs Control", color = "Group") +
  theme_minimal()
```



```
#heat map
expr_mat2 <- limma::avereps(expr_mat)
keep <- apply(expr_mat2, 1, sd, na.rm = TRUE) > 0
expr_mat2 <- expr_mat2[keep, , drop = FALSE]

# Design and contrast (PE vs Control)
pheno$Group <- factor(pheno$Group, levels = c("Control", "Preeclampsia"))
design <- model.matrix(~ 0 + pheno$Group)
colnames(design) <- levels(pheno$Group)
contrast <- makeContrasts(PE_vs_Control = Preeclampsia - Control, levels = design)

fit <- lmFit(expr_mat2, design) |> contrasts.fit(contrast) |> eBayes()

# Differentially expressed genes (adjust thresholds as needed)
res <- topTable(fit, coef = "PE_vs_Control", number = Inf, sort.by = "P")
de_ids <- rownames(res %>% filter(adj.P.Val < 0.05, abs(logFC) >= 1))

# Fallback if too few DE genes pass thresholds
if (length(de_ids) < 20) {
  de_ids <- rownames(res)[1:min(50, nrow(res))]
}
```

```r
# Expression matrix for heatmap (DE genes only), gene-wise z-scores
X <- expr_mat2[de_ids, , drop = FALSE]
Xz <- t(scale(t(X), center = TRUE, scale = TRUE))
Xz <- Xz[apply(Xz, 1, function(v) all(is.finite(v))), , drop = FALSE]  # remove any all-NA rows

# Row labels: use gene symbols if available
if (exists("gene_lookup")) {
  rowlabs <- gene_lookup[rownames(Xz)]
rowlabs[is.na(rowlabs) | rowlabs == ""] <- rownames(Xz)
  rownames(Xz) <- make.unique(rowlabs)
}

# Column annotation (sidebar)
ann_col <- data.frame(Group = pheno$Group)
rownames(ann_col) <- pheno$sample
grp_cols <- c(Control = "lightblue", Preeclampsia = "pink")

# Heatmap color palette
pal <- colorRampPalette(c("navy", "white", "firebrick3"))(101)

# Draw heatmap (clusters rows/cols, shows legends and sidebar)
pheatmap(Xz,
         color = pal,
         show_rownames = TRUE,
         show_colnames = FALSE,
         cluster_rows = TRUE,
         cluster_cols = TRUE,
         annotation_col = ann_col,
         annotation_colors = list(Group = grp_cols),
         fontsize_row = 6,
         main = "Differentially expressed genes (z-scored)")
```


Differentially expressed genes (z−scored)
Group
ENSG00000127125
ENSG00000156170
ENSG00000170442
ENSG00000243709
ENSG00000140612
ENSG00000088766
ENSG00000177054
ENSG00000117481
ENSG00000205189
ENSG00000196381
ENSG00000148053
ENSG00000267206
ENSG00000104131
ENSG00000119326
ENSG00000138382
ENSG00000162851
ENSG00000004799
ENSG00000012983
ENSG00000162458
ENSG00000168288
ENSG00000177425
ENSG00000156374
ENSG00000149571
ENSG00000105516
ENSG00000049246
ENSG00000134852
ENSG00000161544
ENSG00000167612
ENSG00000122254
ENSG00000137101
ENSG00000160326
ENSG00000281165
ENSG00000157873
ENSG00000273936
ENSG00000135929
ENSG00000250510
ENSG00000179639
ENSG00000139508
ENSG00000242685
ENSG00000239463
ENSG00000243189
ENSG00000241394
ENSG00000243215
ENSG00000243719
ENSG00000204257
ENSG00000242361
ENSG00000170458
ENSG00000110446
ENSG00000198931
ENSG00000088205
6
4
2
0
−2
−4
Group
Control
Preeclampsia