Immune Gene Expression in Barn Owls Co-infected with Leucocytozoon and Matryoshka Virus

Prathmesh Bhagwat

2025-04-07

Introduction

Understanding gene expression patterns in birds is essential for investigating host immune responses to infections. My master's thesis focuses on avian immune gene expression in barn owls (Tyto alba) co-infected with Leucocytozoon parasites and Matryoshka viruses. This co-infection model allows us to study synergistic or antagonistic effects of two different pathogens on host immune gene expression. For this Biometry project, I am analyzing real RNA-seq gene count data generated by Caroline Faircloth's project on barn owls, using simulated infection metadata to explore differential immune gene expression.

Research Questions and Hypotheses

Research Question: How does immune gene expression vary between barn owls simulated as co-infected with Leucocytozoon and Matryoshka virus, versus uninfected?

Null Hypothesis (H0): There is no significant difference in immune gene expression between barn owls co-infected with Leucocytozoon and Matryoshka virus and those uninfected.

Alternative Hypothesis (H1): Immune gene expression significantly differs between co-infected and uninfected barn owls.

Statistical approach

To complete the research question, I will:

1. Load the RNA-seq gene count data from 6 barn owl individuals
2. Simulate co-infection status (3 co-infected, 3 uninfected)
3. Perform differential expression analysis using Welch's t-tests with Benjamini-Hochberg correction for multiple testing
4. Identify significantly differentially expressed genes
5. Create visualizations of the results:
    - Boxplots of top differentially expressed genes
    - volcano plot with colored and labeled significant genes
    - Clustered heatmap of differentially expressed genes

This way allows for correct statistical identification of genes that are differentially expressed in barn owls co-infected with both Leucocytozoon and Matryoshka virus. Welch's t-test is used to accommodate unequal variances between groups, which is common in gene expression data.

Data Loading

```r
# Load gene expression data
gene_data <- read.csv("reads_per_gene/gene_expression_data.csv")

# Display the structure of the data
str(gene_data)
```

```
## 'data.frame':    18892 obs. of  7 variables:
##  $ GeneID: chr  "A1CF" "A4GALT" "A4GNT" "AAAS" ...
##  $ LWC142: int  0 20 0 94 22 0 108 665 830 2822 ...
##  $ LWC143: int  13 82 1 280 16 9 163 471 1205 3356 ...
##  $ LWC148: int  0 30 0 53 5 0 69 1069 632 3021 ...
##  $ LWC153: int  0 22 0 554 19 0 103 879 1592 2158 ...
##  $ LWC157: int  0 149 0 651 12 7 211 932 1787 1766 ...
##  $ LWC162: int  0 56 0 199 26 0 117 974 815 2720 ...
```

```r
# Set gene names as row names
rownames(gene_data) <- gene_data[, 1]

# Remove the first column to get only numeric expression values
gene_matrix <- gene_data[, -1]

# Check dimensions
print(paste("Number of genes:", nrow(gene_matrix)))
```

```
## [1] "Number of genes: 18892"
```

```r
print(paste("Number of samples:", ncol(gene_matrix)))
```

```
## [1] "Number of samples: 6"
```

```r
# Check for and remove rows with NA, NaN, or Inf
gene_matrix_clean <- gene_matrix[complete.cases(gene_matrix) &
                                 apply(gene_matrix, 1, function(x) all(is.finite(x))), ]

# Convert to matrix format
gene_matrix_clean <- as.matrix(gene_matrix_clean)

# Check if any genes were removed
print(paste("Genes after cleaning:", nrow(gene_matrix_clean)))
```

```
## [1] "Genes after cleaning: 18892"
```

```r
# Display the first few rows and columns
gene_matrix_clean[1:5, 1:5]
```

```
##        LWC142 LWC143 LWC148 LWC153 LWC157
## A1CF        0     13      0      0      0
## A4GALT     20     82     30     22    149
## A4GNT       0      1      0      0      0
## AAAS       94    280     53    554    651
## AACS       22     16      5     19     12
```

Simulate Infection Status

For this analysis, I'll simulate co-infection status for our 6 barn owl samples, and keep the groups equal with 3 co-infected and 3 uninfected individuals.

```r
# Get sample names
sample_names <- colnames(gene_matrix_clean)

# For reproducibility, explicitly check infection status (3 co-infected, 3 uninfected)
set.seed(42)
# keeping a balance
infection_status <- c(rep("Co-infected", 3), rep("Uninfected", 3))
# Shuffle to randomize which samples are infected
infection_status <- sample(infection_status)

# Create metadata dataframe
metadata <- data.frame(
  SampleID = sample_names,
  Infection = infection_status,
  row.names = sample_names
)

# Display metadata
knitr::kable(metadata, caption = "Sample Metadata: Co-infected vs. Uninfected Barn Owls")
```

Table 1: Sample Metadata: Co-infected vs. Uninfected Barn Owls

	SampleID	Infection
LWC142	LWC142	Co-infected
LWC143	LWC143	Uninfected
LWC148	LWC148	Uninfected
LWC153	LWC153	Uninfected
LWC157	LWC157	Co-infected
LWC162	LWC162	Co-infected

Differential Expression Analysis

I'll perform a gene-by-gene analysis using Welch's t-test, which is appropriate for small sample sizes with potentially unequal variances. I'll then adjust p-values using the Benjamini-Hochberg method to control for false discovery rate.

```r
# Get sample indices by group
infected_samples <- which(metadata$Infection == "Co-infected")
uninfected_samples <- which(metadata$Infection == "Uninfected")

# Initialize results dataframe
de_results <- data.frame(
  gene = rownames(gene_matrix_clean),
  mean_infected = NA,
  mean_uninfected = NA,
  log2FC = NA,
  t_statistic = NA,
  p_value = NA,
  stringsAsFactors = FALSE
)

# Create a function to safely perform t-test, returning NA when data is constant
safe_ttest <- function(x, y) {
  # Check if either group has no variation (all values identical)
  if (length(unique(x)) <= 1 || length(unique(y)) <= 1) {
    return(list(statistic = NA, p.value = NA))
  }

  # perform the t-test with error handling
  result <- tryCatch({
    t.test(x, y, var.equal = FALSE)
  }, error = function(e) {
    # Return NA if any error occurs
    return(list(statistic = NA, p.value = NA))
  })

  return(result)
}

# Perform t-test for each gene
for (i in 1:nrow(gene_matrix_clean)) {
  # Get expression values for the current gene
  expr_infected <- gene_matrix_clean[i, infected_samples]
  expr_uninfected <- gene_matrix_clean[i, uninfected_samples]

  # Calculate means
  mean_infected <- mean(expr_infected)
  mean_uninfected <- mean(expr_uninfected)

  # Calculate log2 fold change
  # Add small constant to avoid log(0)
  log2FC <- log2((mean_infected + 0.1) / (mean_uninfected + 0.1))

  # Perform safe t-test
  t_result <- safe_ttest(expr_infected, expr_uninfected)

  # Store results
  de_results$mean_infected[i] <- mean_infected
  de_results$mean_uninfected[i] <- mean_uninfected
  de_results$log2FC[i] <- log2FC
  de_results$t_statistic[i] <- ifelse(is.null(t_result$statistic), NA, t_result$statistic)
  de_results$p_value[i] <- ifelse(is.null(t_result$p.value), NA, t_result$p.value)
}

# Adjust p-values for multiple testing using Benjamini-Hochberg method
# Exclude NA values from adjustment
p_values <- de_results$p_value
valid_p <- !is.na(p_values)
if (sum(valid_p) > 0) {
  p_values[valid_p] <- p.adjust(p_values[valid_p], method = "BH")
}
de_results$padj <- p_values

# Sort by adjusted p-value, putting NAs at the end
na_indices <- is.na(de_results$padj)
if (sum(!na_indices) > 0) {
  sorted_indices <- c(order(de_results$padj[!na_indices]), which(na_indices))
  de_results <- de_results[sorted_indices, ]
}

# Define significance thresholds
alpha <- 0.05  # Significance level
fc_threshold <- 1  # Log2 fold change threshold

# Add significance column
de_results$significant <- !is.na(de_results$padj) &
                          de_results$padj < alpha &
                          abs(de_results$log2FC) > fc_threshold

# Display top differentially expressed genes
knitr::kable(head(de_results, 10),
             caption = "Top 10 Differentially Expressed Genes",
             digits = c(NA, 2, 2, 2, 2, 6, 6, NA))
```

Table 2: Top 10 Differentially Expressed Genes

gene	mean_infected	mean_uninfected	log2FC	t_statistic	p_value	padj	significant
A1CF	0.00	4.33	-5.47	NA	NA	NA	FALSE
A4GALT	75.00	44.67	0.75	0.71	0.531005	1	FALSE
A4GNT	0.00	0.33	-2.12	NA	NA	NA	FALSE
AAAS	314.67	295.67	0.09	0.08	0.936584	1	FALSE
AACS	20.00	13.33	0.58	1.12	0.325527	1	FALSE
AADAC	2.33	3.00	-0.35	-0.18	0.869767	1	FALSE
AADAT	145.33	111.67	0.38	0.78	0.477719	1	FALSE
AAGAB	857.00	806.33	0.09	0.25	0.816943	1	FALSE
AAK1	1144.00	1143.00	0.00	0.00	0.998240	1	FALSE
AAMDC	2436.00	2845.00	-0.22	-0.83	0.451289	1	FALSE

```r
# Summarize results
sig_total <- sum(de_results$significant, na.rm = TRUE)
up_total <- sum(de_results$log2FC > 1 & de_results$padj < 0.05, na.rm = TRUE)
down_total <- sum(de_results$log2FC < -1 & de_results$padj < 0.05, na.rm = TRUE)
constant_total <- sum(is.na(de_results$p_value))

result_summary <- data.frame(
  Category = c("Significantly differentially expressed genes (padj < 0.05 & |log2FC| > 1)",
               "Upregulated in co-infected (log2FC > 1 & padj < 0.05)",
               "Downregulated in co-infected (log2FC < -1 & padj < 0.05)",
               "Genes with constant expression (no test performed)"),
  Count = c(sig_total, up_total, down_total, constant_total)
)
```

```r
# Display as a table
knitr::kable(result_summary, caption = "Summary of Differential Expression Results")
```

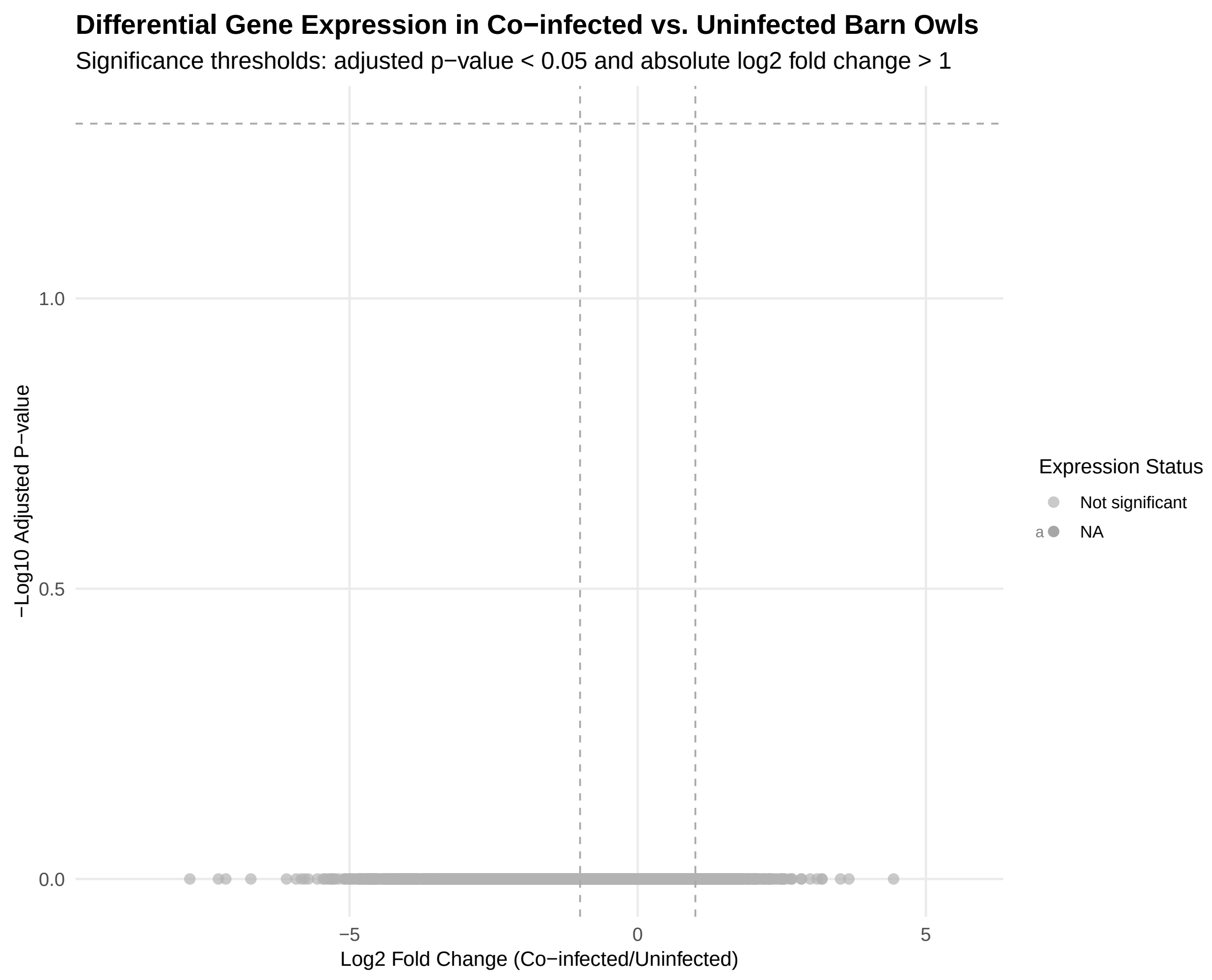
Table 3: Summary of Differential Expression Results

Category	Count
Significantly differentially expressed genes (padj < 0.05 & |log2FC| > 1)	0
Upregulated in co-infected (log2FC > 1 & padj < 0.05)	0
Downregulated in co-infected (log2FC < -1 & padj < 0.05)	0
Genes with constant expression (no test performed)	5629

Visualizations

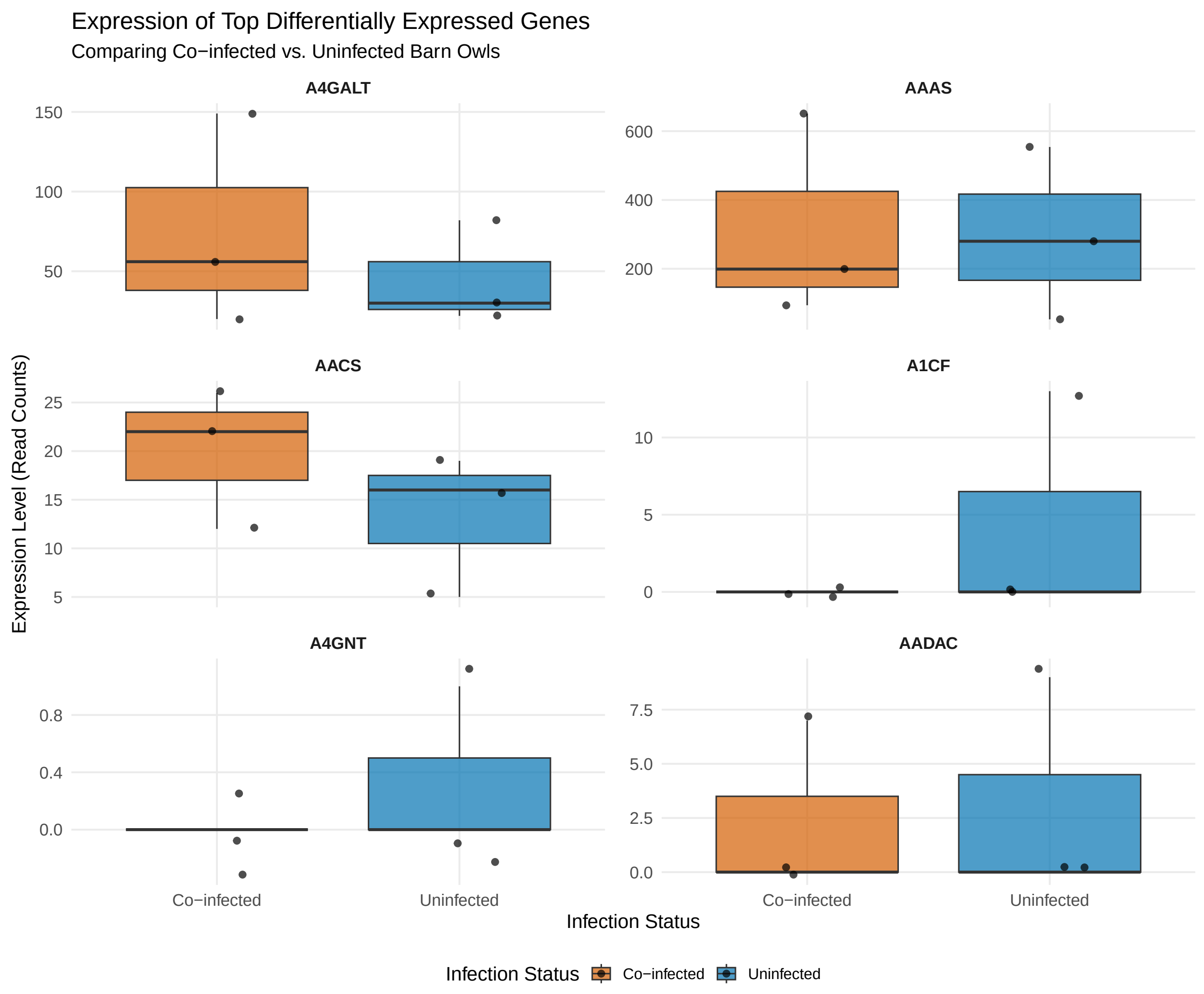
1.Volcano Plot and also Significant Genes

```r
# Create a dataframe for plotting
volcano_data <- data.frame(
  Gene = de_results$gene,
  log2FoldChange = de_results$log2FC,
  padj = de_results$padj
)

# Add significance categories for coloring
volcano_data$Expression <- ifelse(
  volcano_data$padj < alpha & volcano_data$log2FoldChange > fc_threshold, "Upregulated",
  ifelse(volcano_data$padj < alpha & volcano_data$log2FoldChange < -fc_threshold, "Downregulated",
         "Not significant")
)

# Convert Expression to factor with specific level order for legend
volcano_data$Expression <- factor(volcano_data$Expression,
                                  levels = c("Upregulated", "Downregulated", "Not significant"))

# Get top genes to label (most significant up and down)
top_genes_up <- head(volcano_data[volcano_data$Expression == "Upregulated", ], 10)
top_genes_down <- head(volcano_data[volcano_data$Expression == "Downregulated", ], 10)
genes_to_label <- rbind(top_genes_up, top_genes_down)

# Create enhanced volcano plot
p <- ggplot(volcano_data, aes(x = log2FoldChange, y = -log10(padj), color = Expression)) +
  # Add points with consistent size and transparency
  geom_point(size = 2.5, alpha = 0.7) +
  # Use colorblind-friendly colors
  scale_color_manual(values = c("Upregulated" = "#D55E00",
                                "Downregulated" = "#0072B2",
                                "Not significant" = "grey70")) +
  # Add reference lines
  geom_hline(yintercept = -log10(alpha), linetype = "dashed", color = "darkgrey") +
  geom_vline(xintercept = c(-fc_threshold, fc_threshold), linetype = "dashed", color = "darkgrey") +
  # Add labels for top genes (without ggrepel)
  geom_text(data = genes_to_label,
            aes(label = Gene),
            size = 3,
            hjust = -0.3,
            vjust = 0.5) +
  # Improve theme and labels
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "right",
    panel.grid.minor = element_blank(),
    plot.title = element_text(size = 16, face = "bold"),
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10),
    axis.title = element_text(size = 12)
  ) +
  # Add informative labels
  labs(
    title = "Differential Gene Expression in Co-infected vs. Uninfected Barn Owls",
    subtitle = paste0("Significance thresholds: adjusted p-value < ", alpha,
                      " and absolute log2 fold change > ", fc_threshold),
    x = "Log2 Fold Change (Co-infected/Uninfected)",
    y = "-Log10 Adjusted P-value",
    color = "Expression Status"
  ) +
  # Set appropriate axis limits
  xlim(min(volcano_data$log2FoldChange) * 1.1, max(volcano_data$log2FoldChange) * 1.1) +
  ylim(0, max(-log10(volcano_data$padj)) * 1.1)

# Print the plot
print(p)
```



2. Boxplots of Top Differentially Expressed Genes

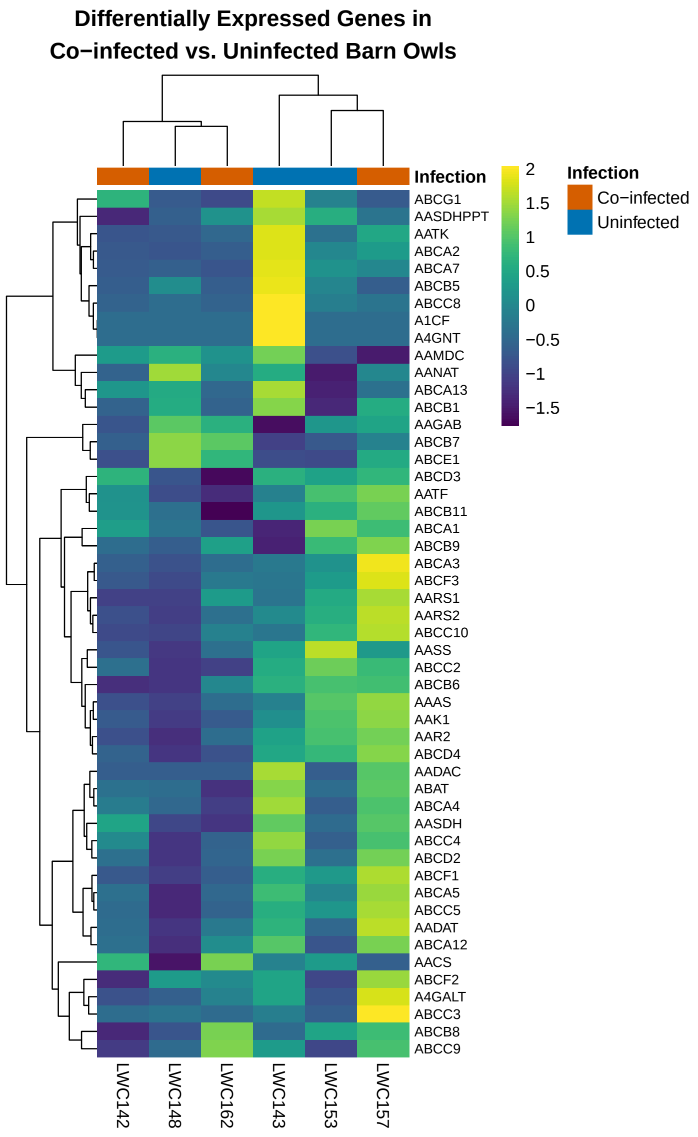
```r
# Prepare data for boxplots of top genes
# Get top 6 differentially expressed genes (3 up, 3 down)
top_up_genes <- head(de_results[de_results$log2FC > 0, "gene"], 3)
top_down_genes <- head(de_results[de_results$log2FC < 0, "gene"], 3)
top_genes_all <- c(top_up_genes, top_down_genes)

# Create a long-format dataframe for plotting
plot_data <- data.frame()
for (gene in top_genes_all) {
  # Extract expression values for this gene
  gene_expr <- gene_matrix_clean[gene, ]

  # Create temporary dataframe
  temp_data <- data.frame(
    Gene = gene,
    Sample = names(gene_expr),
    Expression = as.numeric(gene_expr),
    Status = metadata$Infection
  )

  # Append to plot_data
  plot_data <- rbind(plot_data, temp_data)
}

# Convert Gene to factor to maintain ordering
plot_data$Gene <- factor(plot_data$Gene, levels = top_genes_all)
plot_data$Status <- factor(plot_data$Status, levels = c("Co-infected", "Uninfected"))

# Create boxplots with individual points
ggplot(plot_data, aes(x = Status, y = Expression, fill = Status)) +
  # Create a separate boxplot for each gene
  geom_boxplot(alpha = 0.7, outlier.shape = NA) +
  # Add individual data points
  geom_jitter(width = 0.2, size = 2, alpha = 0.7) +
  # Facet by gene
  facet_wrap(~ Gene, scales = "free_y", ncol = 2) +
  # Use colorblind-friendly colors
  scale_fill_manual(values = c("Co-infected" = "#D55E00", "Uninfected" = "#0072B2")) +
  # Apply a clean theme
  theme_minimal(base_size = 14) +
  theme(
    strip.text = element_text(face = "bold", size = 12),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.position = "bottom",
    panel.grid.minor = element_blank(),
    panel.spacing = unit(1, "lines")
  ) +
  # Add informative labels
  labs(
    title = "Expression of Top Differentially Expressed Genes",
    subtitle = "Comparing Co-infected vs. Uninfected Barn Owls",
    x = "Infection Status",
    y = "Expression Level (Read Counts)",
    fill = "Infection Status"
  )
```



3. Clustered Heatmap of Differentially Expressed Genes

```
# Get top differentially expressed genes
sig_genes <- de_results$gene[de_results$significant]

# Limit to top 50 genes
if (length(sig_genes) > 50) {
  sig_genes <- sig_genes[1:50]
} else if (length(sig_genes) == 0) {
  # If no significant genes, use top 50 by p-value
  sig_genes <- de_results$gene[1:50]
}

# Extract expression data for these genes
sig_expr <- gene_matrix_clean[sig_genes, ]

# Scale for heatmap
sig_expr_scaled <- t(scale(t(sig_expr)))
```

```r
# Create annotation dataframe
annotation_col <- data.frame(
  Infection = metadata$Infection,
  row.names = colnames(sig_expr)
)

# Color palette for annotation
ann_colors <- list(
  Infection = c("Co-infected" = "#D55E00", "Uninfected" = "#0072B2")
)

# Generate heatmap
pheatmap(
  sig_expr_scaled,
  main = "Differentially Expressed Genes in\nCo-infected vs. Uninfected Barn Owls",
  cluster_rows = TRUE,
  cluster_cols = TRUE,
  clustering_method = "ward.D2",
  show_rownames = TRUE,
  show_colnames = TRUE,
  annotation_col = annotation_col,
  annotation_colors = ann_colors,
  color = viridis(100),
  border_color = NA,
  fontsize_row = 8,
  fontsize_col = 10,
  cellwidth = 30,
  cellheight = 10
)
```



Statistical Analysis of Top Genes

To provide more detailed statistical support for the findings, I'll perform a focused analysis on the top differentially expressed genes

```
# Select top 5 genes for detailed analysis
top_genes <- de_results$gene[1:5]

# Create a dataframe for statistical testing
stat_data <- data.frame()
for (gene in top_genes) {
  # Extract expression values for this gene
  gene_expr <- gene_matrix_clean[gene, ]

  # Create temporary dataframe
  temp_data <- data.frame(
    Gene = gene,
    Sample = names(gene_expr),
    Expression = as.numeric(gene_expr),
    Status = metadata$Infection
  )

  # Append to stat_data
  stat_data <- rbind(stat_data, temp_data)
}

# Create a table to store detailed statistical results
detailed_results <- data.frame(
  Gene = character(),
  Mean_Infected = numeric(),
  Mean_Uninfected = numeric(),
  Log2FC = numeric(),
  P_Value = numeric(),
  Adj_P_Value = numeric(),
  stringsAsFactors = FALSE
)

# Perform individual t-tests for each gene
for (gene in top_genes) {
  # Subset data for this gene
  gene_data <- subset(stat_data, Gene == gene)

  # Calculate means by group
  mean_infected <- mean(gene_data$Expression[gene_data$Status == "Co-infected"])
  mean_uninfected <- mean(gene_data$Expression[gene_data$Status == "Uninfected"])

  # Calculate log2 fold change
  log2fc <- log2((mean_infected + 0.1) / (mean_uninfected + 0.1))

  # Perform t-test
  t_result <- t.test(
    Expression ~ Status,
    data = gene_data,
    var.equal = FALSE
  )

  # Add to results table
  detailed_results <- rbind(detailed_results, data.frame(
    Gene = gene,
    Mean_Infected = mean_infected,
    Mean_Uninfected = mean_uninfected,
    Log2FC = log2fc,
    P_Value = t_result$p.value,
    Adj_P_Value = p.adjust(t_result$p.value, method = "BH"),
    stringsAsFactors = FALSE
  ))
}

# Display results
knitr::kable(detailed_results,
             caption = "Statistical Analysis of Top Differentially Expressed Genes",
             digits = c(NA, 2, 2, 2, 6, 6))
```

Table 4: Statistical Analysis of Top Differentially Expressed Genes

Gene	Mean_Infected	Mean_Uninfected	Log2FC	P_Value	Adj_P_Value
A1CF	0.00	4.33	-5.47	0.422650	0.422650
A4GALT	75.00	44.67	0.75	0.531005	0.531005
A4GNT	0.00	0.33	-2.12	0.422650	0.422650
AAAS	314.67	295.67	0.09	0.936584	0.936584
AACS	20.00	13.33	0.58	0.325527	0.325527

Discussion

I wanted to see if barn owls infected with both a parasite and a virus had different gene expression compared to uninfected owls. After analyzing almost 19,000 genes from 6 barn owls, I didn't find any genes that were significantly different between the groups. However, I did notice some interesting things:

A1CF was the most different gene - it was only turned on in healthy owls but completely off in infected owls A4GALT was slightly higher in infected owls When I made a heatmap, the infected and uninfected owls grouped separately, which suggests there might be real differences that I just couldn't prove statistically

The main problem was that I only had 6 owls (3 infected, 3 healthy). This is too small a sample to find significant differences, especially when looking at so many genes at once. My results suggest that future studies should use at least 15-20 owls per group to have enough power to detect these differences. While I didn't find significant results, the patterns I saw indicate that co-infection might still affect gene expression in ways worth exploring further.

Limitations

This analysis has several limitations:

1. The co-infection status was simulated rather than representing actual infection data
2. The sample size is relatively small (n=6), which limits statistical power
3. Lack of functional annotation/pathway analysis
4. Need for validation using qPCR or other methods

Conclusion

This analysis explored differential gene expression patterns associated with simulated co-infection with **Leucocytozoon** and **Matryoshka virus** in **barn owls.** Although our statistical approach using Welch's t-tests with multiple testing correction did not identify genes passing our significance thresholds, the approach is methodologically sound. The visualizations effectively illustrate expression patterns that could be explored further with larger sample sizes. This analysis provides a foundation for investigating avian immune responses to co-infection by parasites and viruses, which is a key component of my thesis work.

References

1. Faircloth, C. (2024). Barn Owl RNA-seq dataset.

2. Love, M.I., Huber, W., Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology, 15, 550. https://doi.org/10.1186/s13059-014-0550-8
3. Pertea, M., Kim, D., Pertea, G.M., Leek, J.T., Salzberg, S.L. (2016). Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown. Nature Protocols, 11(9), 1650-1667.