Volcano Plot using ggplot2

Debojyoti Das

2025-03-02

Introduction

This document demonstrates how to generate a volcano plot using ggplot2 by reading a CSV file that contains gene expression data. The dataset must include at least three mandatory columns:

- log2FC (Log2 Fold Change)
- p_value (P-value for statistical significance)
- Gene_symbol (or Gene EntrezID or Gene ENSEMBL ID)

Each step is explained in detail, with code chunks for clarity.

Setting Up Project Directories

```r
project_dir <- "/Users/debda22/Projects/core_facility_projects/ggplot_basics"
data_dir <- paste0(project_dir, "/input_data")
result_dir <- paste0(project_dir, "/results")

# Create directories if they don't exist
if (!dir.exists(result_dir)) dir.create(result_dir, recursive = TRUE)
```

Reading the Input Data

```r
input_file <- paste0(data_dir, "/test_input_file.csv")
data <- read.csv(input_file)

# Display first few rows of the dataset
display_data <- head(data)
display_data
```

```
##   Gene_symbol    log2FC neg_log10pval log2FC_sq      p_value
## 1       Gene7  2.267283     1.7250171  5.140572 0.0188357482
## 2       Gene9  3.027636     3.8045651  9.166577 0.0001568321
## 3      Gene11  1.957304     0.2616621  3.831041 0.5474417710
## 4      Gene12  3.429968     3.7879732 11.764681 0.0001629396
## 5      Gene13 -2.083291     3.8993947  4.340102 0.0001260681
## 6      Gene18 -3.984683     3.9222951 15.877700 0.0001195928
```

Installing and Loading Required Libraries

```r
# Check if required packages are installed, if not install them
if (!requireNamespace("ggplot2", quietly = TRUE)) {
  install.packages("ggplot2")
}
if (!requireNamespace("dplyr", quietly = TRUE)) {
  install.packages("dplyr")
}

# Load libraries
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Transforming Data for Visualization

Before plotting, we transform the data: - Convert the `p_value` column to `-log10(p_value)` to emphasize small p-values. - Define upregulated and downregulated genes based on cutoff values. - Reverse the `log2FC` values for visualization.

We define thresholds for classification: - `p_value` cutoff: 0.05 - `log2FC` cutoff: 1

```r
# Define cutoffs
pval_cutoff <- 0.05
log2fc_cutoff <- 1

# Classify genes into upregulated, downregulated, or neutral
data <- data %>%
  mutate(
    logP = -log10(p_value),
    negLog2FC = -log2FC,
    regulation = case_when(
      p_value < pval_cutoff & log2FC > log2fc_cutoff ~ "Upregulated",
      p_value < pval_cutoff & log2FC < -log2fc_cutoff ~ "Downregulated",
      TRUE ~ "Non-significant"
    )
  )
```

Selecting Top Genes for Labeling

To highlight important genes, we select the top `n` genes from the upregulated and downregulated groups.

```r
top_n <- 10  # Number of genes to label

top_up <- data %>%
  filter(regulation == "Upregulated") %>%
  arrange(p_value) %>%
  head(top_n)

top_down <- data %>%
  filter(regulation == "Downregulated") %>%
  arrange(p_value) %>%
  head(top_n)

# Combine top genes
top_genes <- bind_rows(top_up, top_down)
```

Creating the Volcano Plot

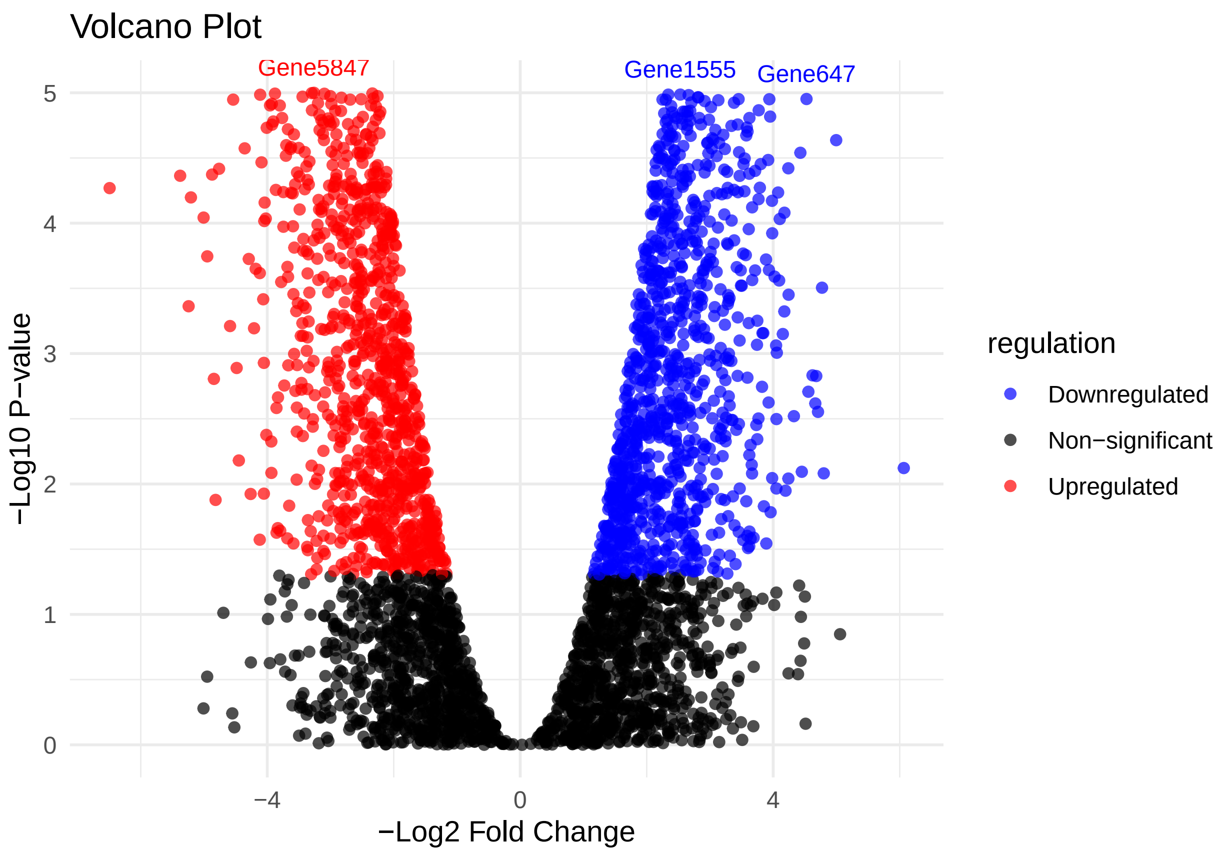
We use `ggplot2` to create a volcano plot with color distinctions:

```r
volcano <- ggplot(data, aes(x = negLog2FC, y = logP, color = regulation)) +
  geom_point(alpha = 0.7) +
  scale_color_manual(values = c("Non-significant" = "black", "Upregulated" = "red", "Downregulated" = "
  labs(title = "Volcano Plot", x = "-Log2 Fold Change", y = "-Log10 P-value") +
  theme_minimal() +
  geom_text(
    data = top_genes,
    aes(label = Gene_symbol),
    vjust = -1,
    size = 3,
    show_guide = FALSE,
    check_overlap = TRUE)
```

```
## Warning: The ‘show_guide‘ argument of ‘layer()‘ is deprecated as of ggplot2 2.0.0.
## i Please use the ‘show.legend‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

Display plot

```r
print(volcano)
```



Saving the Plot

```r
output_file <- paste0(result_dir, "/volcano_plot.png")
ggsave(output_file, plot = volcano, width = 8, height = 6)
```

Conclusion

This document demonstrated how to load, process, and visualize gene expression data using a volcano plot. We added classification for upregulated and downregulated genes, highlighted top genes, and saved the final plot.