External Dataset Analysis - PXD052728

Day 5: Real-world Proteomics Application

Miguel Casanova & Dany Mukesha
2025-11-28

External Dataset Analysis

PXD052728 - Day 5 (Group work)

Learning Objectives

By the end of Day 5, you will be able to:

- Load and clean MaxQuant proteinGroups files
- Build protein matrices and sample metadata from raw data
- Apply complete QC and EDA pipelines to real datasets
- Perform differential expression analysis with limma
- Conduct functional enrichment analysis
- Work with public proteomics datasets

Dataset Overview

PRIDE ID: PXD052728

Study: Human 2D and 3D in vitro models of muscular dystrophies

Conditions:

- **Disease models**: Duchenne (DD) and Myotonic dystrophy type 1 (MD)
- **Controls**: Healthy control (HC) samples
- **Model types**: 2D monolayer and 3D organoid cultures

Relevance: Study dystrophin-associated protein changes, extracellular matrix remodeling, and muscle fiber degeneration markers

Exercise 1.1: Dataset Comparison

What are the main differences between this dataset and the internal one used on Day 2?

- Organism
- Disease model
- Number of samples
- Experimental design

Discuss with your team member!

Loading and Preprocessing proteinGroups

Reading proteinGroups File

```
file_path ← "proteinGroups.txt"   # Adjust if needed

pg_raw ← readr::read_tsv(
  file_path,
  col_types = cols(.default = "c"),
  na = c("", "NA", "NaN")
) %>%
  readr::type_convert()

dim(pg_raw)
head(colnames(pg_raw))
```

Filtering Proteins

Remove contaminants, reverse hits, and site-only identifications:

```
pg_filtered ← pg_raw %>%
  dplyr::filter(
    is.na(`Only identified by site`),
    is.na(Reverse),
    is.na(`Potential contaminant`)
  )

dim(pg_filtered)
```

Exercise 2.1: Filtering Impact

How many rows are removed by this filtering step?

```
cat("Original proteins:", nrow(pg_raw), "\n")
cat("Filtered proteins:", nrow(pg_filtered), "\n")
cat("Percentage retained:",
    round(nrow(pg_filtered)/nrow(pg_raw)*100, 1), "%\n")
```

What percentage of the original dataset is retained?

Intensity Columns and Sample IDs

```r
# Intensity columns
intensity_cols ← grep("^Intensity ", colnames(pg_filtered), value = TRUE)

# Raw labels like "Intensity 2D,DD-1" → "2D,DD-1"
raw_labels ← gsub("^Intensity\\s+", "", intensity_cols)

# Sample IDs like "2D,DD-1" → "2D_DD_1"
sample_ids ← raw_labels %>%
  gsub(",", "_", x = _) %>%
  gsub("-", "_", x = _)

intensity_cols
sample_ids
```

Filter by Quantification Rate

```
quant_per_protein ← rowSums(!is.na(pg_filtered[intensity_cols])) /
  length(intensity_cols)

summary(quant_per_protein)

# Keep proteins quantified in ≥80% of samples
good_protein_ids ← pg_filtered$`Majority protein IDs`[quant_per_protein ⩾ 0.8]

pg_good ← pg_filtered %>%
  dplyr::filter(`Majority protein IDs` %in% good_protein_ids)

dim(pg_good)
```

Exercise 2.2: Threshold Impact

Change the threshold from 80% to 60%:

```
good_protein_ids_60 ← pg_filtered$`Majority protein IDs`[quant_per_protein ⩾ 0.6]
pg_good_60 ← pg_filtered %>%
  dplyr::filter(`Majority protein IDs` %in% good_protein_ids_60)

cat("80% threshold:", nrow(pg_good), "proteins\n")
cat("60% threshold:", nrow(pg_good_60), "proteins\n")
```

What is the trade-off between more proteins and missing values?

Build Protein Matrix and Annotation

```
protein_matrix ← pg_good %>%
  dplyr::select(all_of(intensity_cols)) %>%
  as.matrix()

rownames(protein_matrix) ← pg_good$`Majority protein IDs`
colnames(protein_matrix) ← sample_ids

# log2 transform
protein_matrix ← log2(protein_matrix + 1)

cat("Matrix dimensions:", dim(protein_matrix), "\n")
```

Protein Annotation

```
protein_annotation ← pg_good %>%
  dplyr::transmute(
    protein_id   = `Majority protein IDs`,
    gene_names   = `Gene names`,
    gene_symbol  = sub(";.*", "", `Gene names`),
    protein_name = `Protein names`
  )

head(protein_annotation)
```

Build Sample Metadata

```
sample_metadata ← data.frame(
  sample_id = sample_ids,
  raw_label = raw_labels,
  stringsAsFactors = FALSE
)

# Extract dimension (2D vs 3D)
sample_metadata$dimension ← ifelse(
  startsWith(sample_metadata$raw_label, "2D"), "2D", "3D"
)

# Extract condition (DD, HC, MD)
sample_metadata$condition ← dplyr::case_when(
  grepl("DD", sample_metadata$raw_label) ~ "DD",
  grepl("HC", sample_metadata$raw_label) ~ "HC",
  grepl("MD", sample_metadata$raw_label) ~ "MD",
  TRUE ~ NA_character_
)

sample_metadata
```

Exercise 2.3: Sample Composition

How many samples are:

- 2D vs 3D?
- DD, HC, MD?
- Are replicates balanced across conditions?

```
table(sample_metadata$dimension)
table(sample_metadata$condition)
table(sample_metadata$dimension, sample_metadata$condition)
```

Day 2 Style: QC and EDA

Summary Statistics

```
summary_stats ← data.frame(
  sample_id  = colnames(protein_matrix),
  mean       = apply(protein_matrix, 2, mean, na.rm = TRUE),
  median     = apply(protein_matrix, 2, median, na.rm = TRUE),
  sd         = apply(protein_matrix, 2, sd, na.rm = TRUE),
  n_missing  = apply(protein_matrix, 2, function(x) sum(is.na(x)))
) %>%
  dplyr::left_join(sample_metadata, by = "sample_id")

summary_stats
```

Exercise 3.1: Data Quality

- Compare medians and standard deviations between samples
- Do you see clear differences in overall intensity between groups?
- Which sample has the most missing values?

Missing Data Pattern

```
missing_df ← reshape2::melt(is.na(protein_matrix))
colnames(missing_df) ← c("protein_id", "sample_id", "missing")

ggplot(missing_df, aes(x = sample_id, y = protein_id, fill = missing)) +
  geom_tile() +
  scale_fill_manual(values = c("TRUE" = "red", "FALSE" = "grey90")) +
  theme_minimal() +
  theme(axis.text.y = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Missing Data Pattern")
```

Exercise 3.2: Missing Data Analysis

- Are missing values evenly distributed?
- Do some samples have more missing data?
- Do you see horizontal bands (proteins missing in many samples)?

Data Transformation to Long Format

```
protein_df ← as.data.frame(protein_matrix)
protein_df$protein_id ← rownames(protein_df)

protein_long ← tidyr::pivot_longer(
  protein_df,
  cols = -protein_id,
  names_to = "sample_id",
  values_to = "abundance"
) %>%
  dplyr::left_join(sample_metadata, by = "sample_id")
```

Boxplots by Sample

```
ggplot(protein_long, aes(x = sample_id, y = abundance, fill = condition)) +
  geom_boxplot(outlier.size = 0.5) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Abundance Distribution by Sample",
       x = "Sample", y = "log2 intensity")
```

Density Plots

```
ggplot(protein_long, aes(x = abundance, colour = sample_id)) +
  geom_density() +
  theme_minimal() +
  labs(title = "Density of Protein Abundances",
       x = "log2 intensity", y = "Density") +
  theme(legend.position = "none")
```

Exercise 3.3: Distribution Analysis

- Do any samples look clearly shifted in median or spread?
- Would normalization help align the distributions?

Normalization (Day 3 Style)

```
protein_matrix_norm ← limma::normalizeBetweenArrays(
  protein_matrix,
  method = "quantile"
)

# Compare before/after
par(mfrow = c(1, 2))
boxplot(protein_matrix, main = "Before", las = 2, cex.axis = 0.6)
boxplot(protein_matrix_norm, main = "After", las = 2, cex.axis = 0.6)
par(mfrow = c(1, 1))
```

Exercise 3.4: Normalization Impact

Compare the boxplots before and after normalization:

- How does quantile normalization change distributions?
- Are medians now aligned?

PCA Analysis

```
var_per_protein ← apply(protein_matrix_norm, 1, var, na.rm = TRUE)
non_zero_var ← var_per_protein > 0

pca_data ← t(protein_matrix_norm[non_zero_var, ])
pca_result ← prcomp(pca_data, scale. = FALSE)

pca_df ← data.frame(
  PC1 = pca_result$x[, 1],
  PC2 = pca_result$x[, 2],
  sample_id = rownames(pca_result$x)
) %>%
  dplyr::left_join(sample_metadata, by = "sample_id")

var_explained ← summary(pca_result)$importance[2, 1:2] * 100
```

PCA Visualization

```
ggplot(pca_df, aes(x = PC1, y = PC2, colour = condition, shape = dimension)) +
  geom_point(size = 4) +
  theme_minimal() +
  labs(title = "PCA of Samples",
       x = paste0("PC1 (", round(var_explained[1], 1), "%)"),
       y = paste0("PC2 (", round(var_explained[2], 1), "%)")) +
  scale_color_brewer(palette = "Set1")
```

Exercise 3.5: PCA Interpretation

- Do samples cluster more by condition (DD, HC, MD) or dimension (2D vs 3D)?
- Any samples that don't group with expected class?
- What do PC1 vs PC2 tell about dominant variation sources?

Sample Correlation Heatmap

```
cor_matrix ← cor(protein_matrix_norm, use = "pairwise.complete.obs")

ann_cols ← sample_metadata %>%
  dplyr::select(sample_id, condition, dimension) %>%
  as.data.frame()
rownames(ann_cols) ← ann_cols$sample_id

pheatmap(cor_matrix,
         annotation_col = ann_cols,
         main = "Sample Correlation Matrix")
```

Exercise 3.6: Correlation Analysis

- Which samples show highest similarity?
- Any samples with systematically lower correlations?
- Do samples cluster by biological or technical factors?

Day 3 Style: Differential Expression

Subset Data for Analysis

Focus on 3D samples, DD vs HC, baseline level:

```
analysis_samples ← sample_metadata %>%
  dplyr::filter(
    dimension == "3D",
    condition %in% c("DD", "HC"),
    level == "None"
  )

expr_mat ← protein_matrix_norm[, analysis_samples$sample_id]
dim(expr_mat)
```

Design Matrix

```
group ← factor(analysis_samples$condition, levels = c("HC", "DD"))
design ← model.matrix(~ group)
colnames(design) ← c("Intercept", "DD_vs_HC")

design
```

Exercise 4.1: Design Considerations

Why set reference level to HC?

- Biological interpretation
- Coefficient meaning
- What changes if DD is reference?

Limma Analysis

```
fit ← lmFit(expr_mat, design)
fit ← eBayes(fit)

de_results ← topTable(fit, coef = "DD_vs_HC", number = Inf)

de_results ← de_results %>%
  dplyr::mutate(protein_id = rownames(de_results)) %>%
  dplyr::left_join(protein_annotation, by = "protein_id")

head(de_results)
```

Exercise 4.2: DE Results

- Proteins with adj.P.Val < 0.05?
- Proteins with adj.P.Val < 0.05 & |logFC| > log2(1.5)?
- Percentage of significant proteins?

Volcano Plot

```
volcano_data ← de_results %>%
  dplyr::mutate(
    significance = case_when(
      adj.P.Val < 0.05 & logFC >  log2(1.5) ~ "Up",
      adj.P.Val < 0.05 & logFC < -log2(1.5) ~ "Down",
      TRUE ~ "NS"
    )
  )

ggplot(volcano_data, aes(x = logFC, y = -log10(adj.P.Val), colour = significance)) +
  geom_point(alpha = 0.6, size = 2) +
  scale_color_manual(values = c("Up" = "red", "Down" = "blue", "NS" = "grey")) +
  geom_hline(yintercept = -log10(0.05), linetype = "dashed") +
  geom_vline(xintercept = c(-log2(1.5), log2(1.5)), linetype = "dashed") +
  labs(title = "Volcano Plot: 3D DD vs HC")
```

Exercise 4.3: Volcano Interpretation

Look at highlighted proteins:

- Recognize muscle structure genes?
- Dystrophy-related genes?
- Biological themes?

Heatmap of Top Proteins

```
top_proteins ← de_results %>%
  dplyr::arrange(adj.P.Val) %>%
  dplyr::slice(1:50) %>%
  dplyr::pull(protein_id)

heatmap_mat ← expr_mat[top_proteins, ]

pheatmap(heatmap_mat,
         annotation_col = ann_cols,
         show_rownames = TRUE,
         main = "Top 50 Proteins - 3D DD vs HC")
```

Exercise 4.4: Heatmap Analysis

- Do DD and HC samples form separate clusters?
- Any samples that don't follow general pattern?
- Consistent with PCA results?

Day 4 Style: Functional Enrichment

Prepare Gene Sets

```
# Significant proteins for ORA
de_significant ← de_results %>%
  dplyr::filter(
    !is.na(gene_symbol), gene_symbol ≠ "",
    adj.P.Val < 0.05, abs(logFC) > log2(1.5)
  )

sig_genes ← unique(de_significant$gene_symbol)

# Background genes
bg_genes ← de_results %>%
  dplyr::filter(!is.na(gene_symbol), gene_symbol ≠ "") %>%
  dplyr::pull(gene_symbol) %>%
  unique()

cat("Significant:", length(sig_genes), "\nBackground:", length(bg_genes), "\n")
```

Exercise 5.1: Threshold Impact

Change significance thresholds:

- How does `length(sig_genes)` change?
- What impact on enrichment results?

g:Profiler Enrichment

```
library(gprofiler2)

if (length(sig_genes) ⩾ 10) {
  gost_res ← gprofiler2::gost(
    query = sig_genes,
    custom_bg = bg_genes,
    organism = "hsapiens"
  )

  enrich_tbl ← gost_res$result %>%
    dplyr::arrange(p_value)

  head(enrich_tbl[, c("term_name", "source", "p_value")])
}
```

Exercise 5.2: Enrichment Interpretation

- Which annotation sources dominate top terms?
- Relevant processes for muscle/dystrophy?
- Biological themes?

Enrichment Visualization

```
if (!is.null(gost_res)) {
  gprofiler2::gostplot(gost_res, capped = TRUE, interactive = TRUE)
}
```

Barplot of Top Terms

```
if (!is.null(enrich_tbl)) {
  top_terms ← enrich_tbl[1:min(30, nrow(enrich_tbl)), ]

  ggplot(top_terms, aes(x = reorder(term_name, -log10(p_value)),
                        y = -log10(p_value), fill = source)) +
    geom_col() +
    coord_flip() +
    labs(title = "Top Enriched Terms", x = "Term", y = "-log10 p value")
}
```

Exercise 5.4: Visualization Comparison

Compare visualization methods:

- Which is easier to interpret?
- What insights from each?

Export for Online Tools

```
dir.create("results", showWarnings = FALSE)

# Significant genes
write.table(sig_genes, "results/significant_genes.txt",
            row.names = FALSE, col.names = FALSE, quote = FALSE)

# Ranked list
ranked_out ← data.frame(
  gene = names(gene_list),
  t_stat = as.numeric(gene_list)
)
write_tsv(ranked_out, "results/ranked_genes.tsv")
```

Exercise 5.5: Online Tool Exploration

Upload to:

- STRING (protein interactions)
- Enrichr (comprehensive enrichment)
- g:Profiler web

Compare:

- Biological themes consistency
- Visualization clarity
- Interpretation ease

Extension Ideas

- Repeat for 2D samples only
- Compare H vs L within conditions
- Change fold change thresholds
- Export to DAVID, compare results

Key Takeaways

- ✓ Complete workflow from raw MaxQuant data to biological interpretation
- ✓ Applied Days 2-4 concepts to real dataset
- ✓ Practical experience with public proteomics data
- ✓ Integration of multiple analysis techniques
- ✓ Preparation for independent research projects

Congratulations!

You've completed the 5-day proteomics course with:

- Day 1: R fundamentals and basic analysis
- Day 2: Quality control and exploratory analysis
- Day 3: Preprocessing and differential expression
- Day 4: Functional analysis and data integration
- Day 5: Real-world dataset application

You're now ready to analyze your own and public proteomics data!

Questions?

Thank you for your participation!