paper

Ali Youssef

# UPS1 Case Study 0

The dataset consists of DIA proteomics data from a UPS1-spiked E. coli protein mixture. Eight different concentrations of UPS1 were used (0.1 to 50 fmol), grouped into two categories: low concentrations (0.1–2.5 fmol, labeled as Conc. 2) and high concentrations (5–50 fmol, labeled as Conc. 1). In this dataset 48 HUMAN UPS1 proteins were detected. The data is provided in a SummarizedExperiment format. This example illustrates the application of LimROTS in a simple case where two groups are compared, considering only the concentrations and no other parameters. 48 UPS1-spiked human proteins were detected in this dataset.

## Run LimROTS

# Load necessary packages  
library(LimROTS)  
library(parallel)  
# Load the dataset  
data("UPS1.Case0")  
print(UPS1.Case0)  
# Set random seed for reproducibility  
set.seed(1234, kind = "default")  
# Set metadata and formula for LimROTS analysis  
meta.info <- "Conc."  
B <- 1000 # Number of bootstrap samples  
K <- dim(UPS1.Case0)[1]/4 # Set the value for K based on the data size  
K <- floor(K)   
num\_cores <- 30 # Number of cores for parallel processing  
cluster <- makeCluster(num\_cores) # Create a parallel cluster  
group.name <- "Conc."  
formula.str <- "~ 0 + Conc." # Formula for group comparison  
# Run LimROTS analysis with trend and robust settings enabled  
limrots.result <- LimROTS(data.exp = UPS1.Case0,   
 B = B, K = K, meta.info = meta.info,   
 cluster = cluster, group.name = group.name,   
 formula.str = formula.str, trend = TRUE, robust = TRUE)  
saveRDS(limrots.result, "../LimROTS paper/UPS1.Case0.LimROTS.rds")

## Run ROTS

# Load necessary packages  
library(LimROTS)  
library(parallel)  
# Load the dataset  
data("UPS1.Case0")  
print(UPS1.Case0)  
# Set random seed for reproducibility  
set.seed(1234, kind = "default")  
# Set metadata and formula for LimROTS analysis  
meta.info <- "Conc."  
B <- 1000 # Number of bootstrap samples  
K <- dim(UPS1.Case0)[1]/4 # Set the value for K based on the data size  
K <- floor(K)   
num\_cores <- 30 # Number of cores for parallel processing  
cluster <- makeCluster(num\_cores) # Create a parallel cluster  
group.name <- "Conc."  
formula.str <- "Conc."  
# Run LimROTS analysis with trend and robust settings enabled  
rots.result <- LimROTS(data.exp = UPS1.Case0,   
 B = B, K = K, meta.info = meta.info,   
 cluster = cluster, group.name = group.name,   
 formula.str = formula.str, trend = TRUE, robust = TRUE, n.ROTS = TRUE)  
saveRDS(rots.result, "../LimROTS paper/UPS1.Case0.ROTS.rds")

## Run limma

library(limma)  
library(SummarizedExperiment, quietly = T, warn.conflicts = T)  
data("UPS1.Case0")  
data <- assay(UPS1.Case0)  
formula.str <- "~ 0 + Conc."  
meta.info <- data.frame(colData(UPS1.Case0), check.names = FALSE, row.names = row.names(colData(UPS1.Case0)))  
meta.info$sample.id <- NULL  
design.matrix <- model.matrix(formula(formula.str), data = meta.info)  
colnames(design.matrix) <- make.names(colnames(design.matrix))  
fit <- lmFit(data, design.matrix)  
cont\_matrix <- makeContrasts("Conc.1-Conc.2", levels=design.matrix)  
fit2 <- contrasts.fit(fit, cont\_matrix)  
fit.ebayes <- eBayes(fit2, trend=TRUE, robust=TRUE)  
limma.result <- topTable(fit.ebayes, coef="Conc.1-Conc.2" , number = "Inf")  
saveRDS(limma.result, "../LimROTS paper/UPS1.Case0.limma.rds")

## Comparison

library(caret)  
  
TP <- UPS1.Case0@elementMetadata@listData[["GeneID"]]  
TP <- TP[grepl("HUMAN" , TP)]  
  
predictions\_limrots <- limrots.result$FDR < 0.01  
predictions\_limrots <- factor(predictions\_limrots, levels = c(TRUE, FALSE))  
true\_labels\_limrots <- ifelse(row.names(limrots.result$data) %in% TP , TRUE, FALSE)  
true\_labels\_limrots <- factor(true\_labels\_limrots, levels = c(TRUE, FALSE))  
conf\_matrix\_limrots <- confusionMatrix(predictions\_limrots, true\_labels\_limrots)  
  
predictions\_rots <- rots.result$FDR < 0.01  
predictions\_rots <- factor(predictions\_rots, levels = c(TRUE, FALSE))  
true\_labels\_rots <- ifelse(row.names(rots.result$data) %in% TP , TRUE, FALSE)  
true\_labels\_rots <- factor(true\_labels\_rots, levels = c(TRUE, FALSE))  
conf\_matrix\_rots <- confusionMatrix(predictions\_rots, true\_labels\_rots)  
  
  
predictions\_limma <- limma.result$adj.P.Val < 0.01  
predictions\_limma <- factor(predictions\_limma, levels = c(TRUE, FALSE))  
true\_labels\_limma <- ifelse(row.names(limma.result) %in% TP , TRUE, FALSE)  
true\_labels\_limma <- factor(true\_labels\_limma, levels = c(TRUE, FALSE))  
conf\_matrix\_limma <- confusionMatrix(predictions\_limma, true\_labels\_limma)

library(dplyr, quietly = T, warn.conflicts = T)

Attaching package: 'dplyr'

The following object is masked from 'package:Biobase':  
  
 combine

The following objects are masked from 'package:GenomicRanges':  
  
 intersect, setdiff, union

The following object is masked from 'package:GenomeInfoDb':  
  
 intersect

The following objects are masked from 'package:IRanges':  
  
 collapse, desc, intersect, setdiff, slice, union

The following objects are masked from 'package:S4Vectors':  
  
 first, intersect, rename, setdiff, setequal, union

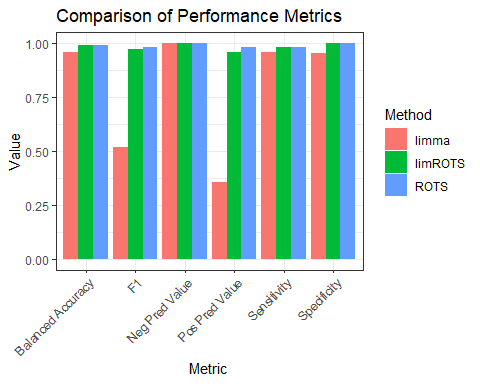
The following objects are masked from 'package:BiocGenerics':  
  
 combine, intersect, setdiff, union

The following object is masked from 'package:matrixStats':  
  
 count

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

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

library(ggplot2)  
  
extract\_metrics <- function(conf\_matrix, method\_name) {  
 metrics <- c(  
 conf\_matrix$byClass["Sensitivity"],  
 conf\_matrix$byClass["Specificity"],  
 conf\_matrix$byClass["Pos Pred Value"],  
 conf\_matrix$byClass["Neg Pred Value"],  
 conf\_matrix$byClass["F1"],  
 conf\_matrix$byClass["Balanced Accuracy"]  
 )  
   
 data.frame(Method = method\_name, Metric = names(metrics), Value = as.numeric(metrics))  
}  
  
metrics\_limrots <- extract\_metrics(conf\_matrix\_limrots, "limROTS")  
metrics\_rots <- extract\_metrics(conf\_matrix\_rots, "ROTS")  
metrics\_limma <- extract\_metrics(conf\_matrix\_limma, "limma")  
  
all\_metrics <- bind\_rows(metrics\_limrots, metrics\_rots, metrics\_limma)  
  
  
ggplot(all\_metrics, aes(x = Metric, y = Value, fill = Method)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 theme\_bw() +  
 labs(title = "Comparison of Performance Metrics",  
 y = "Value", x = "Metric") +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



* **ROTS** and **limROTS** perform almost equally well across all metrics, with very high sensitivity, specificity, PPV, NPV, and balanced accuracy. However, ROTS edges slightly ahead in F1 score and PPV.
* **limma** performs significantly worse than both **limROTS** and **ROTS**, especially in PPV and F1 score, making it less reliable for this particular dataset and analysis task. The lower sensitivity and specificity suggest that limma struggles with both true positive and true negative identification compared to the other two methods.
* Previous studies have demonstrated that ROTS outperforms limma in similar scenarios across various types of omics data. This case study is intended to illustrate that the new LimROTS implementation can achieve comparable performance to ROTS when applied to this type of data, particularly in a straightforward experimental design where only two groups are being compared and no covariates are considered.
* In the upcoming case studies, we will demonstrate how LimROTS can outperform ROTS, limma, and ANCOVA in the analysis of IDA proteomics datasets.

# UPS1 Case Study 1

## Run LimROTS

# Load necessary packages  
library(LimROTS)  
library(parallel)  
# Load the dataset  
data("UPS1.Case1")  
print(UPS1.Case1)  
# Set random seed for reproducibility  
set.seed(1234, kind = "default")  
# Set metadata and formula for LimROTS analysis  
meta.info <- c("Conc." , "inj")  
B <- 1000 # Number of bootstrap samples  
K <- dim(UPS1.Case1)[1]/4 # Set the value for K based on the data size  
K <- floor(K)   
num\_cores <- 10 # Number of cores for parallel processing  
cluster <- makeCluster(num\_cores) # Create a parallel cluster  
group.name <- "Conc."  
formula.str <- "~ 0 + Conc.+ inj + Conc.\*inj" # Formula for group comparison + injections  
# Run LimROTS analysis with trend and robust settings enabled  
limrots.result <- LimROTS(data.exp = UPS1.Case1,   
 B = B, K = K, meta.info = meta.info,   
 cluster = cluster, group.name = group.name,   
 formula.str = formula.str, trend = TRUE, robust = TRUE)  
saveRDS(limrots.result, "../LimROTS paper/UPS1.Case1.LimROTS.rds")

## Run ROTS

# Load necessary packages  
library(LimROTS)  
library(parallel)  
# Load the dataset  
data("UPS1.Case1")  
print(UPS1.Case1)  
# Set random seed for reproducibility  
set.seed(1234, kind = "default")  
# Set metadata and formula for LimROTS analysis  
meta.info <- c("Conc." , "inj")  
B <- 1000 # Number of bootstrap samples  
K <- dim(UPS1.Case1)[1]/4 # Set the value for K based on the data size  
K <- floor(K)   
num\_cores <- 30 # Number of cores for parallel processing  
cluster <- makeCluster(num\_cores) # Create a parallel cluster  
group.name <- "Conc."  
formula.str <- "~ 0 + Conc.+ inj + Conc.\*inj" # Formula for group comparison + injections  
# Run LimROTS analysis with trend and robust settings enabled  
rots.result <- LimROTS(data.exp = UPS1.Case1,   
 B = B, K = K, meta.info = meta.info,   
 cluster = cluster, group.name = group.name,   
 formula.str = formula.str, trend = TRUE, robust = TRUE, n.ROTS = TRUE)  
saveRDS(rots.result, "../LimROTS paper/UPS1.Case1.ROTS.rds")

## Run limma

library(limma)  
library(SummarizedExperiment, quietly = T, warn.conflicts = T)  
data("UPS1.Case1")  
data <- assay(UPS1.Case1)  
formula.str <- "~ 0 + Conc.+ inj + Conc.\*inj"  
meta.info <- data.frame(colData(UPS1.Case1), check.names = FALSE, row.names = row.names(colData(UPS1.Case1)))  
meta.info$sample.id <- NULL  
design.matrix <- model.matrix(formula(formula.str), data = meta.info)  
colnames(design.matrix) <- make.names(colnames(design.matrix))  
fit <- lmFit(data, design.matrix)  
cont\_matrix <- makeContrasts("Conc.1-Conc.2", levels=design.matrix)  
fit2 <- contrasts.fit(fit, cont\_matrix)  
fit.ebayes <- eBayes(fit2, trend=TRUE, robust=TRUE)  
limma.result <- topTable(fit.ebayes, coef="Conc.1-Conc.2" , number = "Inf")  
saveRDS(limma.result, "../LimROTS paper/UPS1.Case1.limma.rds")

## Comparison

library(tidyverse, quietly = T, warn.conflicts = T)

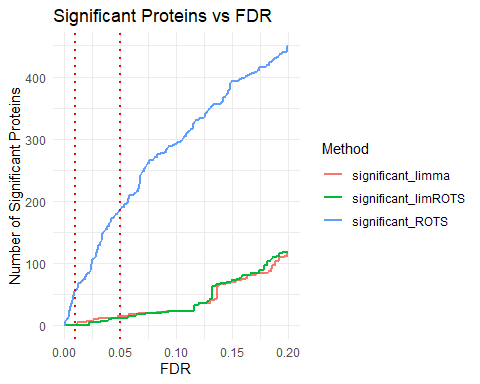
Warning: package 'stringr' was built under R version 4.3.2

Warning: package 'lubridate' was built under R version 4.3.2

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ forcats 1.0.0 ✔ stringr 1.5.1  
✔ lubridate 1.9.3 ✔ tibble 3.2.1  
✔ purrr 1.0.2 ✔ tidyr 1.3.0  
✔ readr 2.1.4   
── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
✖ lubridate::%within%() masks IRanges::%within%()  
✖ dplyr::collapse() masks IRanges::collapse()  
✖ dplyr::combine() masks Biobase::combine(), BiocGenerics::combine()  
✖ dplyr::count() masks matrixStats::count()  
✖ dplyr::desc() masks IRanges::desc()  
✖ tidyr::expand() masks S4Vectors::expand()  
✖ dplyr::filter() masks stats::filter()  
✖ dplyr::first() masks S4Vectors::first()  
✖ dplyr::lag() masks stats::lag()  
✖ purrr::lift() masks caret::lift()  
✖ ggplot2::Position() masks BiocGenerics::Position(), base::Position()  
✖ purrr::reduce() masks GenomicRanges::reduce(), IRanges::reduce()  
✖ dplyr::rename() masks S4Vectors::rename()  
✖ lubridate::second() masks S4Vectors::second()  
✖ lubridate::second<-() masks S4Vectors::second<-()  
✖ dplyr::slice() masks IRanges::slice()  
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

alpha\_thresholds <- seq(0, 0.2, by = 0.001)  
significant\_limROTS <- sapply(alpha\_thresholds, function(alpha) sum(limrots.result$FDR < alpha))  
significant\_ROTS <- sapply(alpha\_thresholds, function(alpha) sum(rots.result$FDR < alpha))  
significant\_limma <- sapply(alpha\_thresholds, function(alpha) sum(limma.result$adj.P.Val < alpha))  
  
  
plot\_data <- data.frame(Alpha = alpha\_thresholds, significant\_limROTS = significant\_limROTS,  
 significant\_ROTS = significant\_ROTS , significant\_limma = significant\_limma)  
  
plot\_data.melt <- plot\_data %>%  
 pivot\_longer(cols = starts\_with("significant"), names\_to = "Method", values\_to = "FDR")  
  
# Plot using ggplot2  
ggplot(plot\_data.melt, aes(x = Alpha, y = FDR, color = Method)) +  
 geom\_line( size = 1) +  
 theme\_minimal() +  
 labs(title = "Significant Proteins vs FDR",  
 x = "FDR",  
 y = "Number of Significant Proteins") +  
 geom\_vline(xintercept = 0.01, linetype="dotted",   
 color = "red", size=1) +   
 geom\_vline(xintercept = 0.05, linetype="dotted",   
 color = "red", size=1)

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
ℹ Please use `linewidth` instead.



Case Study 1 presents a more complex situation where UPS1 human proteins are removed, so no significant differences are expected among the E. coli proteins. The model has been adjusted to account for analytical replicates from different injections. The figure demonstrates that LimROTS and limma handle the data similarly in terms of the number of significant proteins. It is important to note that comparing ROTS in this case is not entirely fair, as ROTS was not designed to account for covariates. However, it is included in this case study to highlight the importance of considering covariates in differential protein analysis, were LimROTS will be more suitable to be used.

# UPS1 Case Study 2

In the final case study for IDA proteomics data, we will explore a more intricate experimental design, the data will be sourced from two software programs, Spectronaut and ScaffoldDIA. Then a synthetic, unbalanced fake batches assigned to the samples. The assignment follows the ratio: c(rep("M", 9), rep("F", 3), rep("M", 3), rep("F", 9), rep("M", 9), rep("F", 3), rep("M", 3), rep("F", 9)). Additionally, 100 E. coli proteins were randomly selected, and an effect size of 10 was added to each. The model was then run using these fake batches. The expected outcome is that only the UPS1 human proteins will be identified as truly significant, while none of the remaining proteins should show significant differences between the concentration groups.  
  
This scenario resembles a real-world case where the experiment involves unbalanced batch assignments or, for instance, an uneven gender ratio among the samples. LimROTS, limma and ANCOVA will be compared in that case study.

## Run LimROTS

# Load necessary packages  
library(LimROTS)  
library(parallel)  
  
# Load the dataset  
data("UPS1.Case4")  
print(UPS1.Case4)  
  
# Set random seed for reproducibility  
set.seed(1234, kind = "default")  
  
# Set metadata and formula for LimROTS analysis  
meta.info <- c("Conc." , "tool" , "fake.batch")  
B <- 1000 # Number of bootstrap samples  
K <- dim(UPS1.Case4)[1]/4 # Set the value for K based on the data size  
K <- floor(K)   
num\_cores <- 30 # Number of cores for parallel processing  
cluster <- makeCluster(num\_cores) # Create a parallel cluster  
group.name <- "Conc."  
formula.str <- "~ 0 + Conc. + tool + fake.batch" # Formula for group comparison  
  
# Run LimROTS analysis with trend and robust settings enabled  
limrots.result <- LimROTS(data.exp = UPS1.Case4,   
 B = B, K = K, meta.info = meta.info,   
 cluster = cluster, group.name = group.name,   
 formula.str = formula.str, trend = TRUE, robust = TRUE)  
saveRDS(limrots.result, "../LimROTS paper/UPS1.Case2.LimROTS.rds")

## Run limma

library(limma)  
library(SummarizedExperiment, quietly = T, warn.conflicts = T)  
data("UPS1.Case4")  
data <- assay(UPS1.Case4)  
formula.str <- "~ 0 + Conc. + tool + fake.batch"  
meta.info <- data.frame(colData(UPS1.Case4), check.names = FALSE, row.names = row.names(colData(UPS1.Case4)))  
meta.info$sample.id <- NULL  
design.matrix <- model.matrix(formula(formula.str), data = meta.info)  
colnames(design.matrix) <- make.names(colnames(design.matrix))  
fit <- lmFit(data, design.matrix)  
cont\_matrix <- makeContrasts("Conc.1-Conc.2", levels=design.matrix)  
fit2 <- contrasts.fit(fit, cont\_matrix)  
fit.ebayes <- eBayes(fit2, trend=TRUE, robust=TRUE)  
limma.result <- topTable(fit.ebayes, coef="Conc.1-Conc.2" , number = "Inf")  
saveRDS(limma.result, "../LimROTS paper/UPS1.Case2.limma.rds")

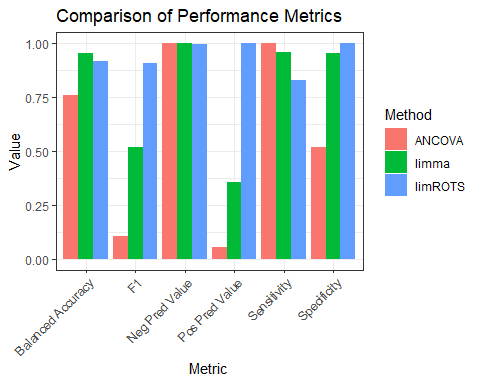
## Run ANCOVA

formula.str <- "~ Conc. + tool + fake.batch"  
data("UPS1.Case4")  
data <- assay(UPS1.Case4)  
meta.info <- data.frame(colData(UPS1.Case4), check.names = FALSE, row.names = row.names(colData(UPS1.Case4)))  
meta.info$sample.id <- NULL  
ancova.result <- data.frame()  
for(i in 1:nrow(data)){  
 df.temp <- data[i,]  
 data.temp <- data.frame(response = as.numeric(df.temp) , Conc. = meta.info$Conc.,  
 tool = meta.info$tool, fake.batch = meta.info$fake.batch )  
 ancova\_model <- lm(formula(paste("response", formula.str)), data = data.temp)  
 coef\_table <- anova(ancova\_model)  
 coef\_table <- data.frame(coef\_table)  
 significant\_groups <- coef\_table[rownames(coef\_table) == "Conc." , ]  
 ancova.result <- rbind(ancova.result, data.frame(protein = row.names(data)[i], p.value =significant\_groups[, 5] ))  
}  
ancova.result$FDR <- p.adjust(ancova.result$p.value, method = "fdr")

## Comparison

library(caret)  
  
TP <- UPS1.Case4@elementMetadata@listData[["GeneID"]]  
TP <- TP[grepl("HUMAN" , TP)]  
  
predictions\_limrots <- limrots.result$FDR < 0.01  
predictions\_limrots <- factor(predictions\_limrots, levels = c(TRUE, FALSE))  
true\_labels\_limrots <- ifelse(row.names(limrots.result$data) %in% TP , TRUE, FALSE)  
true\_labels\_limrots <- factor(true\_labels\_limrots, levels = c(TRUE, FALSE))  
conf\_matrix\_limrots <- confusionMatrix(predictions\_limrots, true\_labels\_limrots)  
  
predictions\_limma <- limma.result$adj.P.Val < 0.01  
predictions\_limma <- factor(predictions\_limma, levels = c(TRUE, FALSE))  
true\_labels\_limma <- ifelse(row.names(limma.result) %in% TP , TRUE, FALSE)  
true\_labels\_limma <- factor(true\_labels\_limma, levels = c(TRUE, FALSE))  
true\_labels\_limma <- confusionMatrix(predictions\_limma, true\_labels\_limma)  
  
  
predictions\_ancova <- ancova.result$FDR < 0.01  
predictions\_ancova <- factor(predictions\_ancova, levels = c(TRUE, FALSE))  
true\_labels\_ancova <- ifelse(ancova.result$protein %in% TP , TRUE, FALSE)  
true\_labels\_ancova <- factor(true\_labels\_ancova, levels = c(TRUE, FALSE))  
conf\_matrix\_ancova <- confusionMatrix(predictions\_ancova, true\_labels\_ancova)

library(dplyr, quietly = T, warn.conflicts = T)   
library(ggplot2)  
  
extract\_metrics <- function(conf\_matrix, method\_name) {  
 metrics <- c(  
 conf\_matrix$byClass["Sensitivity"],  
 conf\_matrix$byClass["Specificity"],  
 conf\_matrix$byClass["Pos Pred Value"],  
 conf\_matrix$byClass["Neg Pred Value"],  
 conf\_matrix$byClass["F1"],  
 conf\_matrix$byClass["Balanced Accuracy"]  
 )  
   
 data.frame(Method = method\_name, Metric = names(metrics), Value = as.numeric(metrics))  
}  
  
metrics\_limrots <- extract\_metrics(conf\_matrix\_limrots, "limROTS")  
metrics\_limma <- extract\_metrics(conf\_matrix\_limma, "limma")  
metrics\_ancova <- extract\_metrics(conf\_matrix\_ancova, "ANCOVA")  
  
all\_metrics <- bind\_rows(metrics\_limrots, metrics\_limma, metrics\_ancova)  
  
  
ggplot(all\_metrics, aes(x = Metric, y = Value, fill = Method)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 theme\_bw() +  
 labs(title = "Comparison of Performance Metrics",  
 y = "Value", x = "Metric") +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



* **limROTS** is clearly the best performer, with strong results across sensitivity, specificity, PPV, and F1 score, indicating a well-balanced method with accurate classification of significant and non-significant proteins.
* **limma** and **ANCOVA** both achieve perfect sensitivity and NPV, but their extremely poor specificity and PPV suggest they misclassify many proteins as significant, leading to a high number of false positives. This results in very low F1 scores and reduced balanced accuracy, making them unreliable for this particular case study.

In biomarker and drug discovery, particularly during the discovery phase using omics approaches, accurate identification of true significant features is crucial to streamline subsequent steps, making them faster, more precise, and reliable. The selection of an appropriate statistical tool is a critical decision, given the multitude of available methods. In the context of DIA proteomics, which is an ideal experimental setup for early biomarker and drug discovery phases, many previous studies have relied on tools such as ANCOVA and limma, which were developed for broader omics applications. While these tools may be acceptable, more specialized methods should be employed for greater accuracy. LimROTS offers a robust statistical approach for identifying true significant proteins in DIA experiments, minimizing false positive discoveries. LimROTS excels by providing high balanced accuracy, strong sensitivity, and excellent specificity, making it a reliable tool for these critical phases of discovery.  
In theory, LimROTS is applicable to other types of omics, such as transcriptomics and metabolomics. However, further case studies are needed to validate its effectiveness in these areas.