**Packages used:**

**ProteoViz:** [**https://github.com/ByrumLab/ProteoViz.git**](https://github.com/ByrumLab/ProteoViz.git)

**Proteus:** [**https://github.com/bartongroup/Proteus.git**](https://github.com/bartongroup/Proteus.git)

**Legend:**

|  |  |
| --- | --- |
| Script 1 – Make Metadata | |
| Purpose | Files used |
| Creates metadata by extracting necessary data columns. | proteinGroups.txt (MaxQuant)  Phospho (STY)Sites.txt (MaxQuant) |
| Script 2 – Total Proteome Analysis | |
| Purpose | Files used |
| Data filtering of total proteome   * Reverse * Potential contaminant * Only identified by site   Data normalisation of total proteome using CONSTANd (Proteus)  Statistical comparison using limma | proteinGroups.txt (MaxQuant)  Sample\_metadata.tsv (User-defined)  Contrast\_matrix.tsv (User-defined)  Protein\_metadata.tsv (ProteoViz-Script1) |
| Script 3 – Global Phosphoproteome Analysis | |
| Purpose | Files used |
| Data filtering of phosphoproteome   * Reverse * Potential contaminant   Data normalisation of phosphoproteome using CONSTANd (Proteus)  Statistical comparison using limma  Consensus motif column generation | Phospho (STY)Sites.txt (MaxQuant)  Sample\_metadata.tsv (User-defined)  Contrast\_matrix.tsv (User-defined)  Phospho\_metadata.tsv (ProteoViz-Script1) |

**Sample\_metadata.tsv:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sample | Enrichment | Batch | Replicate | Pool | Sample\_name | Model\_group |
| Reporter\_intensity\_corrected\_1 | Lysate | 0 | 1 |  | Ctr 1 | Ctr |
| Reporter\_intensity\_corrected\_2 | Lysate | 0 | 2 |  | Ctr 2 | Ctr |
| Reporter\_intensity\_corrected\_3 | Lysate | 0 | 3 |  | Ctr 3 | Ctr |
| Reporter\_intensity\_corrected\_4 | Lysate | 0 | 4 |  | Ctr 4 | Ctr |
| Reporter\_intensity\_corrected\_5 | Lysate | 0 | 5 |  | Ctr 5 | Ctr |
| Reporter\_intensity\_corrected\_6 | Lysate | 0 | 1 |  | dKO 1 | dKO |
| Reporter\_intensity\_corrected\_7 | Lysate | 0 | 2 |  | dKO 2 | dKO |
| Reporter\_intensity\_corrected\_8 | Lysate | 0 | 3 |  | dKO 3 | dKO |
| Reporter\_intensity\_corrected\_9 | Lysate | 0 | 4 |  | dKO 4 | dKO |
| Reporter\_intensity\_corrected\_10 | Lysate | 0 | 5 |  | dKO 5 | dKO |
| Reporter\_intensity\_corrected\_1 | Phos | 0 | 1 |  | Ctr 1 | Ctr |
| Reporter\_intensity\_corrected\_2 | Phos | 0 | 2 |  | Ctr 2 | Ctr |
| Reporter\_intensity\_corrected\_3 | Phos | 0 | 3 |  | Ctr 3 | Ctr |
| Reporter\_intensity\_corrected\_4 | Phos | 0 | 4 |  | Ctr 4 | Ctr |
| Reporter\_intensity\_corrected\_5 | Phos | 0 | 5 |  | Ctr 5 | Ctr |
| Reporter\_intensity\_corrected\_6 | Phos | 0 | 1 |  | dKO 1 | dKO |
| Reporter\_intensity\_corrected\_7 | Phos | 0 | 2 |  | dKO 2 | dKO |
| Reporter\_intensity\_corrected\_8 | Phos | 0 | 3 |  | dKO 3 | dKO |
| Reporter\_intensity\_corrected\_9 | Phos | 0 | 4 |  | dKO 4 | dKO |
| Reporter\_intensity\_corrected\_10 | Phos | 0 | 5 |  | dKO 5 | dKO |

**Contrast\_matrix.tsv:**

|  |
| --- |
| Contrast\_name |
| dKO - Ctr |

**Script 1 – Make Metadata**

library(tidyverse)

if(!dir.exists("data")){dir.create("data")}

df <- read\_tsv("txt/proteinGroups.txt", guess\_max = 10000) %>%

{set\_names(., gsub(" ", "\_", names(.)))}

df %>%

select(Majority\_protein\_IDs, Fasta\_headers, Score, id) %>%

mutate(Description = str\_extract(Fasta\_headers, "(?<= )[^\\|]+(?= OS\\=)"),

Gene\_name = str\_extract(Fasta\_headers, "(?<=GN\\=)[^\\|]+(?= PE\\=)"),

Uniprot\_ID = str\_extract(Fasta\_headers, "(?<=\\|)[^\\|]+(?=\\|)")) %>%

write\_tsv("data/Protein\_metadata.tsv")

phos\_df <- read\_tsv("txt/Phospho (STY)Sites.txt", guess\_max = 20000) %>%

{set\_names(., gsub(" ", "\_", names(.)))}

phos\_df %>%

select(Proteins:Score, Amino\_acid, Sequence\_window, `Phospho\_(STY)\_Probabilities`, Charge, id:Evidence\_IDs) %>%

mutate(Description = str\_extract(Fasta\_headers, "(?<= )[^\\|]+(?= OS\\=)"),

Gene\_name = str\_extract(Fasta\_headers, "(?<=GN\\=)[^\\|]+(?= PE\\=)"),

Uniprot\_ID = str\_extract(Fasta\_headers, "(?<=\\|)[^\\|]+(?=\\|)"),

Flanking = gsub("\\;.\*$", "", Sequence\_window) %>%

str\_sub(9,23) %>%

paste0("-p")) %>%

write\_tsv("data/Phospho\_metadata.tsv")

**Script 2 – Total Proteome Analysis**

library(tidyverse)

library(limma)

df <- read\_tsv("txt/proteinGroups.txt", guess\_max = 20000) %>%

{set\_names(., gsub(" ", "\_", names(.)))}

sample\_df <- read\_tsv("Sample\_metadata.tsv") %>%

mutate(Sample\_name = factor(Sample\_name, levels = unique(Sample\_name)),

Model\_group = factor(Model\_group, levels = unique(Model\_group))) %>%

#For protein

filter(grepl("Lysate", Enrichment))

contrast\_df <- read\_tsv("contrast\_matrix.tsv")

df1 <- df %>%

filter(is.na(Reverse),

is.na(Potential\_contaminant),

is.na(Only\_identified\_by\_site))

df1[21:30][df1[21:30] == 0] <- NA

# using RAS (constrained standardisation CONSTANd) to normalise data

# the mean in each column and row equals to 1/n, where n is the number of columns and rows, respectively

# this bit of code was taken from Proteus package

RAS <- function(K, max.iter=50, eps=1e-5) {

n <- ncol(K)

m <- nrow(K)

# ignore rows with only NAs

good.rows <- which(rowSums(!is.na(K)) > 9)

K <- K[good.rows, ]

cnt <- 1

repeat {

row.mult <- 1 / (n \* rowMeans(K, na.rm=TRUE))

K <- K \* row.mult

err1 <- 0.5 \* sum(abs(colMeans(K, na.rm=TRUE) - 1/n))

col.mult <- 1 / (n \* colMeans(K, na.rm=TRUE))

K <- t(t(K) \* col.mult)

err2 <- 0.5 \* sum(abs(rowMeans(K, na.rm=TRUE) - 1/n))

cnt <- cnt + 1

if(cnt > max.iter || (err1 < eps && err2 < eps)) break

}

# reconstruct full table

KF <- matrix(NA, nrow=m, ncol=n)

KF[good.rows, ] <- K

return(KF)

}

normalizeTMT <- function(datf, max.iter=50, eps=1e-5) {{

datf[21:30] <- RAS(datf[21:30])

}

return(datf)

}

df1n <- normalizeTMT(df1)

df2 <- df1n %>%

mutate\_at(vars(matches("corrected.\*")), log2)

df2a <- df2 %>%

select(id, one\_of(sample\_df$Sample)) %>%

gather(Sample, Intensity, one\_of(sample\_df$Sample))

df2b <- df2a %>%

right\_join(sample\_df) %>%

group\_by(id, Batch) %>%

group\_by(Sample) %>%

ungroup()

df2b %>%

select(-Sample) %>%

write\_tsv("data/Normalized\_proteingroup\_intensities.tsv")

# Limma -------------------------------------------------------------------

# Make model matrix and contrasts

model\_df <- sample\_df %>%

filter(!is.na(Sample\_name))

design <- model.matrix(~ 0 + model\_df$Model\_group)

colnames(design) <- unique(model\_df$Model\_group)

cont\_table <- makeContrasts(contrasts = as.list(contrast\_df$Contrast\_name), levels = model\_df$Model\_group)

Comparisons <- dimnames(cont\_table)$Contrasts

# Format for Limma

df3 <- df2b %>%

filter(!is.na(Sample\_name)) %>%

select(id, Sample\_name, Intensity) %>%

spread(Sample\_name, Intensity) %>%

as.data.frame() %>%

write\_tsv("data/Protein\_limma\_input.tsv") %>%

column\_to\_rownames("id")

fit <- lmFit(df3, design)

cont\_fit <- contrasts.fit(fit, cont\_table)

fit2 <- eBayes(cont\_fit)

#Extract topTable for each contrast

f1 <- function(x1){

a1\_name <- colnames(fit2$coefficients)[[x1]]

a1 <- topTable(fit2, x1, number = Inf) %>%

rownames\_to\_column("id") %>%

mutate(Comparison = a1\_name) %>%

mutate(id = as.integer(id)) %>%

as\_tibble() %>%

left\_join(df %>% select(id),

by = "id")

a1

}

df4 <- map\_df(seq\_along(Comparisons), f1) %>%

filter(!is.na(adj.P.Val))

df4 %>%

write\_tsv("data/Protein\_limma\_output.tsv")

spread\_protein\_limma <- df4 %>%

select(id, logFC, adj.P.Val, P.Value, Comparison) %>%

gather(Type, Value, logFC, adj.P.Val, P.Value) %>%

unite(Type, Comparison, Type, sep = " ") %>%

spread(Type, Value) %>%

mutate(id = as.integer(id))

protein\_meta <- read\_tsv("data/Protein\_metadata.tsv")

protein\_quantitative <- df3 %>%

rownames\_to\_column("id") %>%

mutate(id = as.integer(id))

#For filtering, only include proteins identified in at least 1 sample

samples <- as.character(na.omit(sample\_df$Sample\_name))

summarized\_protein <- protein\_meta %>%

left\_join(protein\_quantitative) %>%

left\_join(spread\_protein\_limma) %>%

filter\_at(vars(one\_of(samples)), any\_vars(!is.na(.))) %>%

write\_tsv("data/Protein\_summarized\_data.tsv")

**Script 3 – Global Phosphoproteome Analysis**

library(tidyverse)

library(limma)

#Run Protein\_Limma.R first.

df <- read\_tsv("txt/Phospho (STY)Sites.txt", guess\_max = 20000) %>%

{set\_names(., gsub(" ", "\_", names(.)))} %>%

filter(

is.na(Reverse),

is.na(Potential\_contaminant)

)

df0 <- read\_tsv("txt/Phospho (STY)Sites.txt", guess\_max = 20000)

sample\_df <- read\_tsv("Sample\_metadata.tsv") %>%

mutate(Sample\_name = factor(Sample\_name, levels = unique(Sample\_name)),

Model\_group = factor(Model\_group, levels = unique(Model\_group))) %>%

#For protein

filter(grepl("Phos", Enrichment))

contrast\_df <- read\_tsv("contrast\_matrix.tsv")

#Filters for class I sites, selects relevant columns, separates by phosphosite number

df1 <- df %>%

select(id, Protein\_group\_IDs, matches("corrected.\*\\\_{3}[[:digit:]]$")) %>%

gather(Sample, Intensity, everything(), -id, -Protein\_group\_IDs) %>%

separate(Sample, into = c("Sample", "Phos\_number"), sep = "\_\_\_") %>%

right\_join(sample\_df)

#Normalizes to pool, then centers distribution around median = 0, then normalizes to protein fold change

df2 <- df1 %>%

filter(Intensity > 0)

# Limma -------------------------------------------------------------------

model\_df <- sample\_df %>%

filter(!is.na(Sample\_name))

design <- model.matrix(~ 0 + model\_df$Model\_group)

colnames(design) <- unique(model\_df$Model\_group)

cont\_table <- makeContrasts(contrasts = as.list(contrast\_df$Contrast\_name), levels = model\_df$Model\_group)

#Format for Limma

df3 <- df2 %>%

filter(!is.na(Sample\_name)) %>%

mutate(Intensity = ifelse(is.nan(Intensity), NA\_real\_, Intensity)) %>%

unite(id\_phos, id, Phos\_number) %>%

select(id\_phos, Sample\_name, Intensity) %>%

spread(Sample\_name, Intensity, fill = NA) %>%

write\_tsv("data/Phospho\_limma\_input.tsv") %>%

as.data.frame() %>%

column\_to\_rownames("id\_phos")

RAS <- function(K, max.iter=50, eps=1e-5) {

n <- ncol(K)

m <- nrow(K)

# ignore rows with only NAs

good.rows <- which(rowSums(!is.na(K)) > 9)

K <- K[good.rows, ]

cnt <- 1

repeat {

row.mult <- 1 / (n \* rowMeans(K, na.rm=TRUE))

K <- K \* row.mult

err1 <- 0.5 \* sum(abs(colMeans(K, na.rm=TRUE) - 1/n))

col.mult <- 1 / (n \* colMeans(K, na.rm=TRUE))

K <- t(t(K) \* col.mult)

err2 <- 0.5 \* sum(abs(rowMeans(K, na.rm=TRUE) - 1/n))

cnt <- cnt + 1

if(cnt > max.iter || (err1 < eps && err2 < eps)) break

}

# reconstruct full table

KF <- matrix(NA, nrow=m, ncol=n)

KF[good.rows, ] <- K

return(KF)

}

normalizeTMT <- function(datf1, max.iter=50, eps=1e-5) {{

datf1[1:10] <- RAS(datf1[1:10])

}

return(datf1)

}

df4 <- normalizeTMT(df3)

df4[, 1:10] <- log(df4[1:10], 2)

fit <- lmFit(df4, design)

Comparisons <- dimnames(cont\_table)$Contrasts

cont\_fit <- contrasts.fit(fit, cont\_table)

fit2 <- eBayes(cont\_fit)

ncol(fit2$contrasts)

#Extract topTable for each contrast

f1 <- function(x1){

a1\_name <- colnames(fit2$coefficients)[[x1]]

a1 <- topTable(fit2, x1, number = Inf) %>%

rownames\_to\_column("id") %>%

separate(id, into = c("id", "Phos\_number")) %>%

mutate(Comparison = a1\_name) %>%

mutate(id = as.integer(id)) %>%

as\_tibble() %>%

left\_join(df %>% select(id),

by = "id")

a1

}

df5 <- map\_df(seq\_along(Comparisons), f1) %>%

filter(!is.na(adj.P.Val))

df5 %>%

write\_tsv("data/Phospho\_limma\_output.tsv")

spread\_phospho\_limma <- df5 %>%

unite(id, id, Phos\_number) %>%

select(id, logFC, adj.P.Val, Comparison, P.Value) %>%

gather(Type, Value, logFC, adj.P.Val, P.Value) %>%

unite(Type, Comparison, Type, sep = " ") %>%

spread(Type, Value) %>%

separate(id, into = c("id", "Phos\_number"), sep = "\_") %>%

mutate(id = as.integer(id))

phospho\_meta <- read\_tsv("data/Phospho\_metadata.tsv")

phos\_quantitative <- df4 %>%

rownames\_to\_column("id") %>%

separate(id, into = c("id", "Phos\_number"), sep = "\_") %>%

mutate(id = as.integer(id))

phospho\_meta %>%

left\_join(phos\_quantitative) %>%

left\_join(spread\_phospho\_limma) %>%

filter(!is.na(Phos\_number)) %>%

write\_tsv("data/Phospho\_summarized\_data.tsv")

phos <- read\_tsv("data/Phospho\_summarized\_data.tsv")

NDR <- str\_detect(phos$Flanking, "([A-Z])([A-Z])([H])([A-Z])([R])([A-Z])([A-Z])([TS])([A-Z])([A-Z])([A-Z])([A-Z])([A-Z])([A-Z])([A-Z])")

phos$motif <- NDR

phos$motif <- gsub("TRUE", "NDR", phos$motif)

phos %>% write\_tsv("data/Phospho\_summarized\_data\_motif.tsv")