RWT-46 PRC2 IP_RBRID

Robert Warneford-Thomson

6/24/2016

Import RBR-ID data

```
library(gdata)
installXLSXsupport()
```

```
##
## Perl XLSX support libraries successfully installed.
```

```
Prot_table <-read.xls(xls = "Robert test 1 with mouse proteome.xlsx", sheet = 1)
```

Sanitize Sample names + Add description

```
samples <-c("no4su_01","no4su_02","no4su0.1x_01", "no4su0.1x_02","4su_01","4su_02","4su0.1x_01","4su_0.
colnames(Prot_table)[8:15] <- samples
Prot_table[,8:15] <- as.numeric(unlist(Prot_table[,8:15]))
Prot_table <- cbind(Pept_ID = c(1:nrow(Prot_table)), Prot_table)
```

normalize peptide intensity relative to total intensity

```
# Divide data by Sum of intensities
Norm_data <-sweep(Prot_table[,9:16], 2, colSums(Prot_table[9:16], na.rm = T), FUN = "/")

Prot_table <- cbind(Prot_table, Norm_data)
colnames(Prot_table)[18:25] <- paste0(samples,"_Norm")
```

Calculate Average/Log-fold change

```
#Compute averages of technical replicates
Avgs <- as.data.frame(lapply(c(18,20,22,24), function(x) {apply(Prot_table[,c(x,x+1)], 1, function(x) {
colnames(Avgs)<- c("no4sU1x_mean", "no4sU0.1x_mean", "4su1x_mean", "4su0.1x_mean")

Prot_table <- cbind(Prot_table, Avgs)
ncol(Prot_table)
```

```
## [1] 29
```

```
# Calculate Samples negative log2-fold change
Prot_table$Log2fold_1x <- log2(Prot_table[,28]/Prot_table[,26])
Prot_table$Log2fold_0.1x <- log2(Prot_table[,29]/Prot_table[,27])

Prot_table$Log2fold_1x[is.infinite(Prot_table$Log2fold_1x) | is.nan(Prot_table$Log2fold_1x)] <- NA
Prot_table$Log2fold_0.1x[is.infinite(Prot_table$Log2fold_0.1x) | is.nan(Prot_table$Log2fold_0.1x)] <- N
```

Process data and calculate RBR-ID score

```
# Calculate p-value from two-sided t-test
Prot_table$p.value_1x <- apply(Prot_table[, c( "no4su_01_Norm","no4su_02_Norm","4su_01_Norm", "4su_02_N

Prot_table$p.value_0.1x <- apply(Prot_table[, c( "no4su0.1x_01_Norm","no4su0.1x_02_Norm", "4su0.1x_01_N

# RBR-ID score
Prot_table$score_1x <- log(Prot_table$p.value_1x)*Prot_table$Log2fold_1x

Prot_table$score_0.1x <- log(Prot_table$p.value_0.1x)*Prot_table$Log2fold_0.1x
```

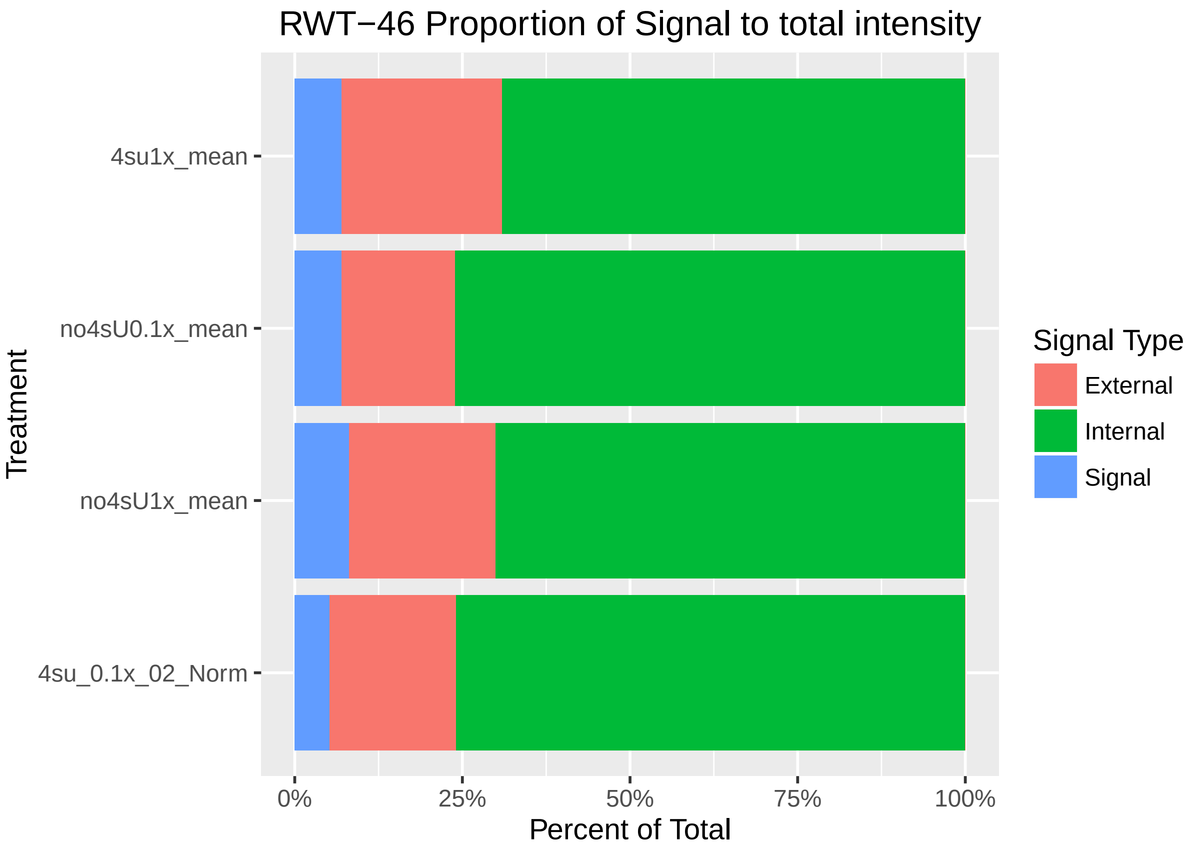
Calculate 'Internal' contaminant-to-signal ratio

These include proteins that were purposefully added to the samples during the experiment (e.g. Protein G from beads, rabbit IgG). External contaminants are those that are commonly found in mass spectrometry runs

```
#Label contaminant types
Addin_ID <- c("P19909", "P01870","P01826", "P01840","P01687")
temp <- lapply(Addin_ID, function(x) {grep(pattern = paste0("^.+", x,".+$"), Prot_table$Proteins, value
temp <- unlist(temp)

Prot_table[,"Contaminant_type"] <- "Signal"
Prot_table[temp,"Contaminant_type"] <- "Internal"
Prot_table$Contaminant_type[(Prot_table$Potential.contaminant=="+") & (Prot_table$Contaminant_type != "


library(ggplot2)
library(reshape2)

## Calcuate proportion of Contaminants to total peptide intensities
## Calculate total contaminant-to-signal ratio
Contam_prop <- data.frame(ID = names(Prot_table)[25:28],
                          Internal = colSums(Prot_table[which(Prot_table$Contaminant_type == "Internal"
                          External = colSums(Prot_table[which(Prot_table$Contaminant_type == "External"
                          Signal = colSums(Prot_table[which(Prot_table$Contaminant_type == "Signal"),c(
rownames(Contam_prop)
```

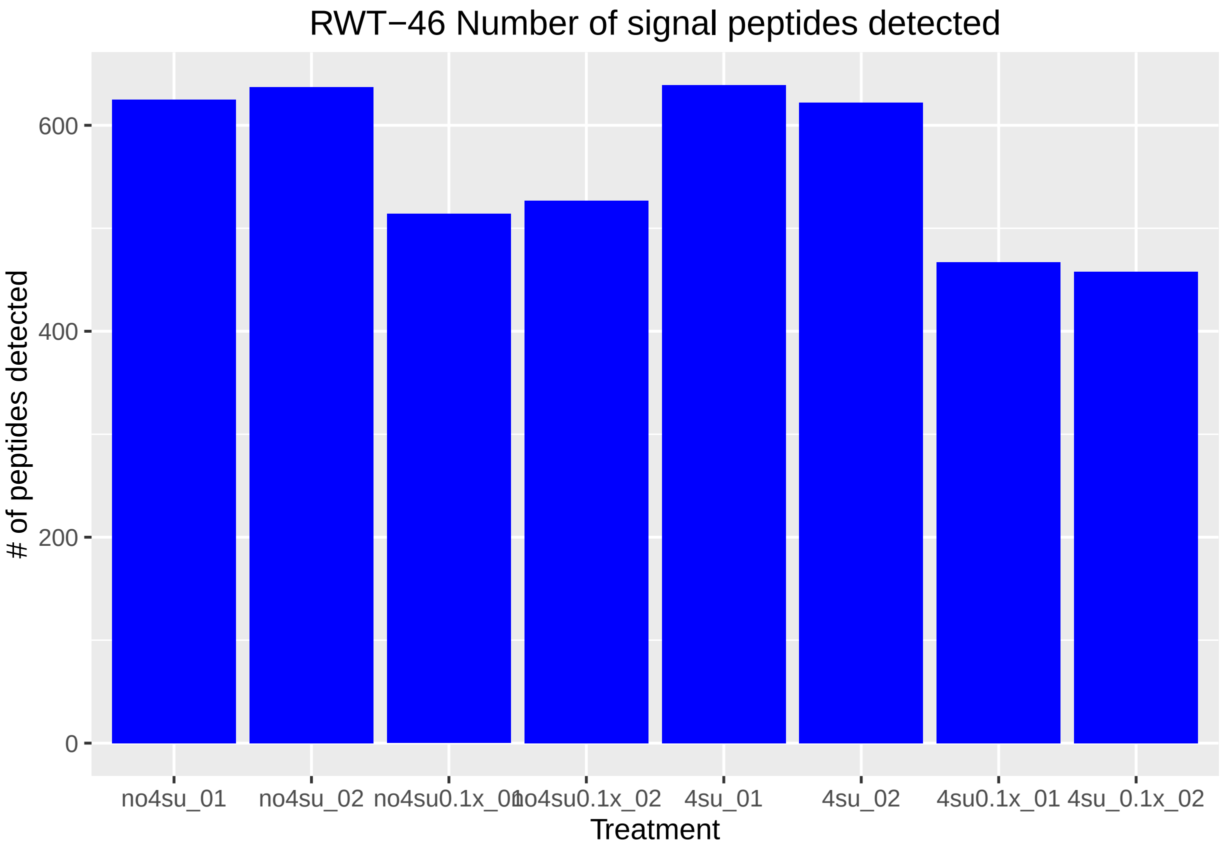
```
## [1] "4su_0.1x_02_Norm" "no4sU1x_mean"     "no4sU0.1x_mean"
## [4] "4su1x_mean"
```

```
    #Add an id variable for the filled regions

    datm <- melt(cbind(Prot_table[,c(25:28)], Contaminant = Prot_table$Contaminant_type), id.vars = c('C

library(scales)
ggplot(datm,aes(x = variable, y = value, fill = Contaminant)) +
    geom_bar(position = "fill",stat = "identity") +
    scale_y_continuous(labels = percent_format()) +
    coord_flip() +
    scale_fill_discrete(guide = guide_legend(title = "Signal Type")) +
    labs(x = "Treatment", y = "Percent of Total", title = "RWT-46 Proportion of Signal to total intensi
```



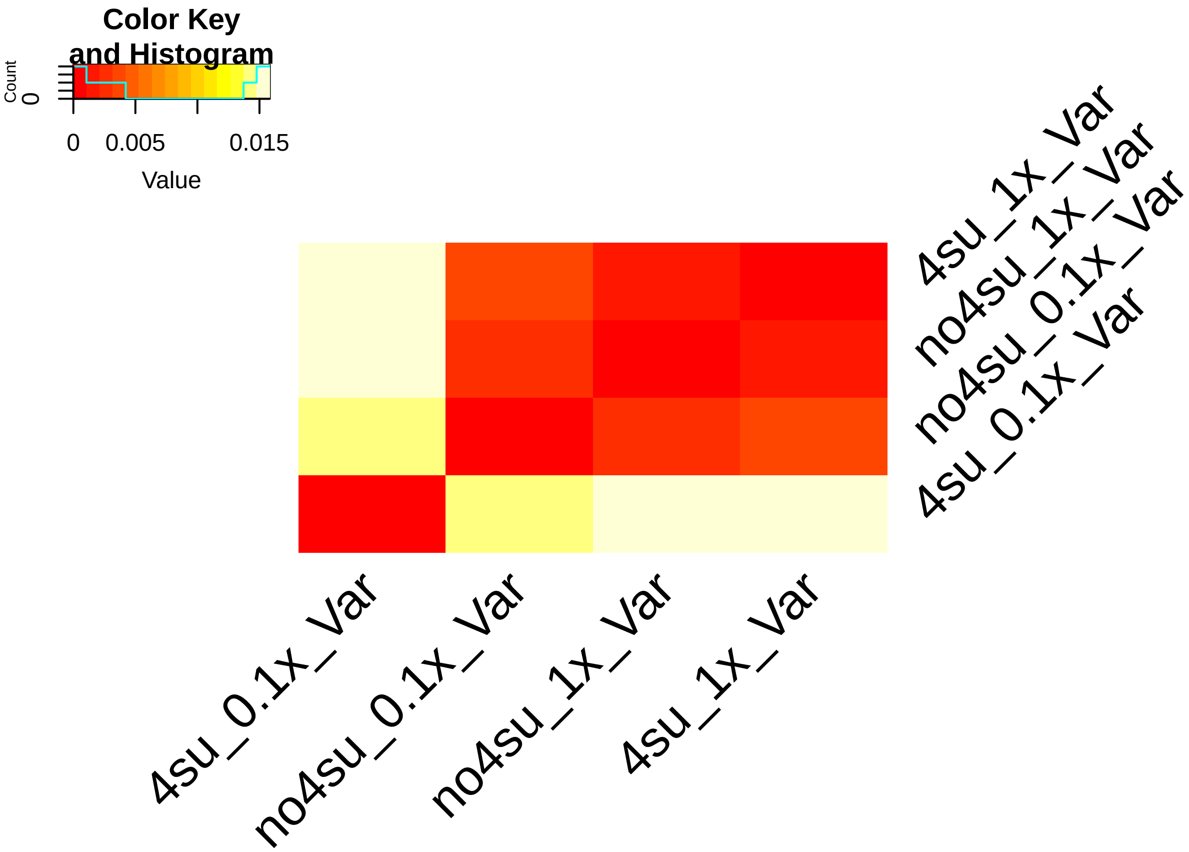
Determine reproducibility as f(x) of MS/MS input material

```
# Obtain number of non-zero signal peptides per sample
Pepdata <- list(Numpep = data.frame(ID = factor(samples, levels=unique(samples)), Numpep =  sapply(sampl
                                    {nrow(Prot_table[(Prot_table[[x]] != 0) & (Prot_table$Contaminant_ty


ggplot(Pepdata$Numpep, aes(x = ID, y = Numpep)) +
      geom_bar(stat = "identity", fill = "blue") +
      labs(x = "Treatment", y = "# of peptides detected", title = "RWT-46 Number of signal peptides dete
```



```
# Identify peptides found in each sample

Pepdata$Peplist <- lapply(samples, function(x) {Prot_table[(is.na(Prot_table[[x]]) == F) & (Prot_table$
names(Pepdata$Peplist) <- samples


# All peptides detected in 1x samples
Pepdata$Peplist$All_1x <- unique(c(Pepdata$Peplist$no4su_01, Pepdata$Peplist$no4su_02,Pepdata$Peplist$`


# All peptides detected in 0.1x samples
Pepdata$Peplist$All_0.1x <- unique(c(Pepdata$Peplist$no4su0.1x_01, Pepdata$Peplist$no4su0.1x_02, Pepdat


# Peptides unique to 1x samples
Pepdata$Peplist$only_1x <- setdiff(Pepdata$Peplist$All_1x, Pepdata$Peplist$All_0.1x)

# Peptides unique to 0.1x samples
Pepdata$Peplist$only_0.1x <- setdiff(Pepdata$Peplist$All_0.1x, Pepdata$Peplist$All_1x)

#Measure variance of samples
Prot_table$no4su_1x_Var <- apply(Prot_table[,c("no4su_01_Norm","no4su_02_Norm")],1,var)
Prot_table$no4su_0.1x_Var <- apply(Prot_table[,c("no4su0.1x_01_Norm","no4su0.1x_02_Norm")],1,var)
Prot_table$`4su_1x_Var` <- apply(Prot_table[,c("4su_01_Norm","4su_02_Norm")],1,var)
Prot_table$`4su_0.1x_Var` <- apply(Prot_table[,c("4su0.1x_01_Norm", "4su_0.1x_02_Norm")],1,var)

#Assign NA to variance=0 instances
```

```r
Prot_table[,c("no4su_1x_Var", "no4su_0.1x_Var", "4su_1x_Var", "4su_0.1x_Var")][ Prot_table[,c("no4su_1x

#Compute Distance Matrix

foo <- data.matrix(dist(rbind(Prot_table$no4su_1x_Var, Prot_table$no4su_0.1x_Var, Prot_table$`4su_1x_Va
rownames(foo) <- c("no4su_1x_Var", "no4su_0.1x_Var", "4su_1x_Var", "4su_0.1x_Var")
colnames(foo) <- c("no4su_1x_Var", "no4su_0.1x_Var", "4su_1x_Var", "4su_0.1x_Var")

# Plot distance function as heatmap
library(gplots)
heatmap.2(foo,dendrogram='none', Rowv=TRUE, Colv=TRUE,trace='none',margins = c(10, 10), srtRow = 45, sr
```


Color Key
and Histogram
Count
Value
4su_1x_Var
no4su_1x_Var
no4su_0.1x_Var
4su_0.1x_Var
4su_0.1x_Var
no4su_0.1x_Var
no4su_1x_Var
4su_1x_Var