

Analysing the first set of SILAC-based LOPIT data

```
#-----  
# Author      : Manasa Ramakrishna, mr325@le.ac.uk  
# Date started : 1st June, 2017  
# Last modified : 15th June, 2017  
# Aim         : To take a look at first SILAC labelled LOPIT data on Trizol  
# Depends     : On 'silacFunctions.R'. Make sure they are in the same directory  
# Notes       : Works on data from Rayner's first experiments  
#-----
```

```
# Invoking libraries
```

```
library(clusterProfiler)  
library(ggplot2)  
library(gplots)  
library(limma)  
library(org.Hs.eg.db)  
library(outliers)  
library(RColorBrewer)  
library(reshape2)  
library(stringr)
```

```
#Setting working directories
```

```
wd = "/Users/manasa/Documents/Work/TTT/02_Proteomics/01_First-SILAC-LOPIT/"  
setwd(wd)  
getwd()
```

```
## [1] "/Users/manasa/Documents/Work/TTT/02_Proteomics/01_First-SILAC-LOPIT"
```

```
indir = paste(wd,"Input",sep="/")  
outdir = paste(wd,paste(Sys.Date(),"Output",sep = "_"),sep = "/")
```

```
if (exists(outdir)){  
  print("Outdir exists")  
}else{  
  dir.create(outdir)  
}
```

```
# Sourcing function file
```

```
source("silacFunctions.R")
```

Now that we have loaded all the packages we need for working with this data, let's move on to the data.

```
# -----  
# Step 0: Read data  
# Read in all the data required for analysis  
# -----
```

```
# File of contaminants - proteins to exclude from analysis as are things like keratin, alcohol dehydrog  
contam = read.delim("Input/Common contaminant_all.csv",sep=",",header=T)
```

```
# Read in the data files that contain peptide level output from Proteome discoverer...
```

```
# Note: I have converted the excel files to text files and removed '(02)' from the filenames to make it
```

```
# Modify the headers to be all lower case as well as remove unwanted spaces, symbols etc...to keep it s
# Columns of interest are "sequence", "modifications", "master.protein.accessions", "abundance.heavy", "ab
```

```
infiles = grep("Trizol",list.files("Input/",full.names = T),value=T)
prot.data = NULL
for (i in infiles){
  in.dat = read.delim(i,sep="\t",comment.char="",as.is=T,header=F)
  in.dat$sample = strsplit(i,"/")[1][2]
  #print(i)
  prot.data = rbind(prot.data,in.dat)
}

colnames(prot.data) = prot.data[1,]
dim(prot.data)
```

```
## [1] 81595    29
```

```
# Remove header lines as they differ in one of the columns (fraction number I think)
remove.head = which(prot.data[,1]=="Checked")
prot.data = prot.data[-(remove.head),]
dim(prot.data)
```

```
## [1] 81586    29
```

```
# Change header names a little to make them neutral and remove space, special characters
colnames(prot.data) = tolower(colnames(prot.data))
colnames(prot.data) = gsub(" ",".",colnames(prot.data))
colnames(prot.data) = gsub("#","no",colnames(prot.data))
colnames(prot.data) = gsub("\\:\\.f2\\:", "n",colnames(prot.data))
colnames(prot.data)[12] = "theoretical.mass"
colnames(prot.data)[13] = "light.sample"
colnames(prot.data)[14] = "heavy.sample"
colnames(prot.data)[15] = "abundance.ratio.heavy.to.light"
colnames(prot.data)[16] = "abundance.light"
colnames(prot.data)[17] = "abundance.heavy"
colnames(prot.data)[29] = "sample"

# Convert abundance values to numeric from character
prot.data$abundance.heavy = as.numeric(prot.data$abundance.heavy)
prot.data$abundance.light = as.numeric(prot.data$abundance.light)
prot.data$abundance.ratio.heavy.to.light = as.numeric(prot.data$abundance.ratio.heavy.to.light)

# Add rep, reagent and UV amount columns
prot.data$uv = sapply(strsplit(prot.data$sample,"_"), "[",2)
prot.data$repl = gsub(".txt","",gsub("rep","",sapply(strsplit(prot.data$sample,"_"), "[",3)))
prot.data$repl = paste(prot.data$uv,prot.data$repl,sep=".")
prot.data$reagent = sapply(strsplit(prot.data$sample,"_"), "[",1)
head(prot.data)
```

##	checked	confidence	sequence	modifications
## 2	FALSE	High	AGAHLQGGAK	
## 3	FALSE	High	IMNTFSVVPSPK	1xLabel:13C(6)15N(2) [K12]
## 4	FALSE	High	IMNTFSVVPSPK	
## 5	FALSE	High	NQVTQLKEQVPGFTR	
## 6	FALSE	High	EQELQQTLLQEQSVLDQLR	

```

## 7 FALSE High TTPSVVAFTADGER
## quality.pep quality.q-value no.protein.groups no.proteins no.psms
## 2 5.61944E-06 0 1 2 16
## 3 3.4751E-06 0 4 11 8
## 4 2.42801E-06 0 4 11 6
## 5 0.000142696 0 1 2 4
## 6 1.97349E-07 0 1 1 6
## 7 9.80731E-06 0 1 3 4
## master.protein.accessions no.missed.cleavages theoretical.mass
## 2 P04406 0 909.4900898
## 3 Q13509; P04350; P07437; P68371 0 1327.716983
## 4 Q13509; P04350; P07437; P68371 0 1319.702784
## 5 F5H2F4 1 1841.986822
## 6 Q15149 0 2313.168093
## 7 P38646 0 1450.717248
## light.sample heavy.sample abundance.ratio.heavy.to.light abundance.light
## 2 High Peak Found 0.01 5200000
## 3 Not Found Not Found NA NA
## 4 Not Found Not Found NA NA
## 5 Not Found Not Found NA NA
## 6 High Not Found 0.01 656400
## 7 High Not Found NA NA
## abundance.heavy quan.info amanda.score.ms.amanda
## 2 43050 Unique 135.209857
## 3 NA No Quan Values 201.8662137
## 4 NA No Quan Values 176.2597659
## 5 NA No Quan Values 127.0712654
## 6 NA Unique 152.1398947
## 7 NA No Quan Values 104.2831076
## confidence.ms.amanda search.space.ms.amanda percolator.q-value.ms.amanda
## 2 High 1636 0
## 3 High 2601 0
## 4 High 2728 0
## 5 High 3048 0
## 6 High 3758 0
## 7 High 2414 0
## percolator.pep.ms.amanda ions.score.mascot confidence.mascot
## 2 0.00001018 65.04 High
## 3 0.000002614 75.28 High
## 4 0.00001715 79.87 High
## 5 0.00005757 40.09 High
## 6 1.928E-07 44.95 High
## 7 0.00002452 48.27 High
## search.space.mascot percolator.q-value.mascot percolator.pep.mascot
## 2 0 2.945E-07
## 3 0 1.695E-07
## 4 0 1.112E-07
## 5 0 0.00003356
## 6 0 1.211E-08
## 7 0 5.664E-07
## sample uv repl reagent
## 2 Trizol_150mJ_rep1.txt 150mJ 150mJ.1 Trizol
## 3 Trizol_150mJ_rep1.txt 150mJ 150mJ.1 Trizol
## 4 Trizol_150mJ_rep1.txt 150mJ 150mJ.1 Trizol

```

```
## 5 Trizol_150mJ_rep1.txt 150mJ 150mJ.1 Trizol
## 6 Trizol_150mJ_rep1.txt 150mJ 150mJ.1 Trizol
## 7 Trizol_150mJ_rep1.txt 150mJ 150mJ.1 Trizol
```

```
dim(prot.data)
```

```
## [1] 81586    32
```

prot.data has 32 columns and 81,586 rows - each row belonging to a peptide. We now go through a series of filtering steps to obtain a dataset we can use for downstream analyses.

```
# -----
# Step 1 : Filter
# We perform 3 layers of filtering - unique proteins, contaminants, missing values
# -----
```

```
# Step 1a : Filter only for those peptides that have a unique master protein. Done using column "quan.info"
dim(prot.data)
```

```
## [1] 81586    32
```

```
peptide.stats = table(prot.data$sample, prot.data$quan.info)
peptide.stats
```

```
##
##               No Quan Values Not Unique Unique
## Trizol_150mJ_rep1.txt           5211        287  4088
## Trizol_150mJ_rep2.txt           3869        248  3048
## Trizol_150mJ_rep3.txt           4571        247  3483
## Trizol_275mJ_rep1.txt           5266        289  3845
## Trizol_275mJ_rep2.txt           4303        212  3091
## Trizol_275mJ_rep3.txt           6578        468  6383
## Trizol_400mJ_rep1.txt           5365        286  4129
## Trizol_400mJ_rep2.txt           5203        265  3422
## Trizol_400mJ_rep3.txt           4320        219  2890
```

```
filt.1a = prot.data[which(prot.data$quan.info == "Unique"),]
length(which(filt.1a$quan.info == "Unique"))
```

```
## [1] 34379
```

```
dim(filt.1a) #34279 are unique proteins, 47207 are non-unique or are missing values
```

```
## [1] 34379    32
```

```
# This table is very odd. Rayner had an explanation - "High" was equivalent to "Peak found"
# but also indicates which label "heavy" or "light" is higher in abundance
# However, there are peptides where it is "High" but the peptide values are NA. Hmmm...
table(light=filt.1a$light.sample, heavy=filt.1a$heavy.sample)
```

```
##           heavy
## light      High Not Found Peak Found
## High         0    15810    11775
## Not Found   1672         0         0
## Peak Found  5122         0         0
```

```
# Step 1b : Filter out those proteins that are contaminants from the contaminants list and annotate mis
filt.1b = filt.1a[-which(filt.1a$master.protein.accessions %in% contam$Protein.Group.Accessions),]
num.contams = length(which(filt.1a$master.protein.accessions %in% contam$Protein.Group.Accessions))
```

```

# Annotate which peptides are missing heavy, light or both, abundance values
filt.1b$missing.val = rowSums(is.na(filt.1b[,c("abundance.heavy", "abundance.light")])) > 0

dim(filt.1a) # 34379 in total

## [1] 34379    32

dim(filt.1b) # 33657 filtered proteins

## [1] 33657    33

print(num.contams) # 722 contaminant proteins

## [1] 722

# Want to do some stats with missing values.
table(filt.1b$sample, filt.1b$missing.val) # More missing values in 150mJ_rep, 275mK_rep2 and 450mJ_rep3

##
##                FALSE TRUE
## Trizol_150mJ_rep1.txt 1914 2097
## Trizol_150mJ_rep2.txt 1460 1523
## Trizol_150mJ_rep3.txt 1667 1723
## Trizol_275mJ_rep1.txt 1836 1915
## Trizol_275mJ_rep2.txt 1358 1670
## Trizol_275mJ_rep3.txt 3055 3175
## Trizol_400mJ_rep1.txt 2024 2031
## Trizol_400mJ_rep2.txt 1572 1797
## Trizol_400mJ_rep3.txt 1428 1412

round(table(filt.1b$sample, filt.1b$missing.val)/rowSums(table(filt.1b$sample, filt.1b$missing.val))*100, 1)

##
##                FALSE TRUE
## Trizol_150mJ_rep1.txt 47.72 52.28
## Trizol_150mJ_rep2.txt 48.94 51.06
## Trizol_150mJ_rep3.txt 49.17 50.83
## Trizol_275mJ_rep1.txt 48.95 51.05
## Trizol_275mJ_rep2.txt 44.85 55.15
## Trizol_275mJ_rep3.txt 49.04 50.96
## Trizol_400mJ_rep1.txt 49.91 50.09
## Trizol_400mJ_rep2.txt 46.66 53.34
## Trizol_400mJ_rep3.txt 50.28 49.72

# How many missing in heavy, how many missing in light
miss.l = table("_Missing light values_"=filt.1b$missing.val, filt.1b$light.sample)
miss.h = table("_Missing heavy values_"=filt.1b$missing.val, filt.1b$heavy.sample)
miss = cbind(miss.l, miss.h)
colnames(miss) = c("Light_High", "Light_NotFound", "Light_Found", "Heavy_High", "Heavy_NotFound", "Heavy_Found")
rownames(miss) = c("notMissing", "missing")
print(miss)

##                Light_High Light_NotFound Light_Found Heavy_High Heavy_NotFound
## notMissing           11258              0         5056         5056              0
## missing              15644            1658           41         1699            15215
##                Heavy_Found
## notMissing           11258

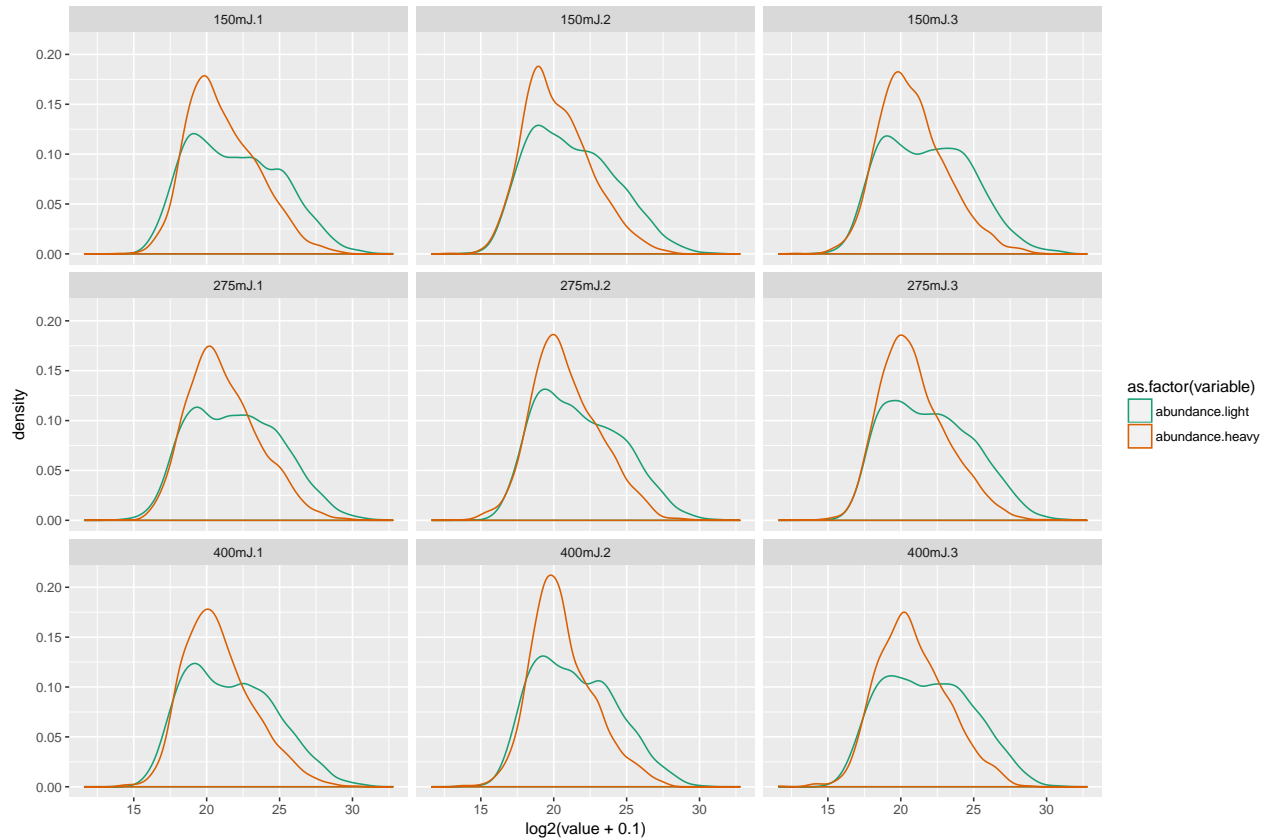
```

```
## missing 429
```

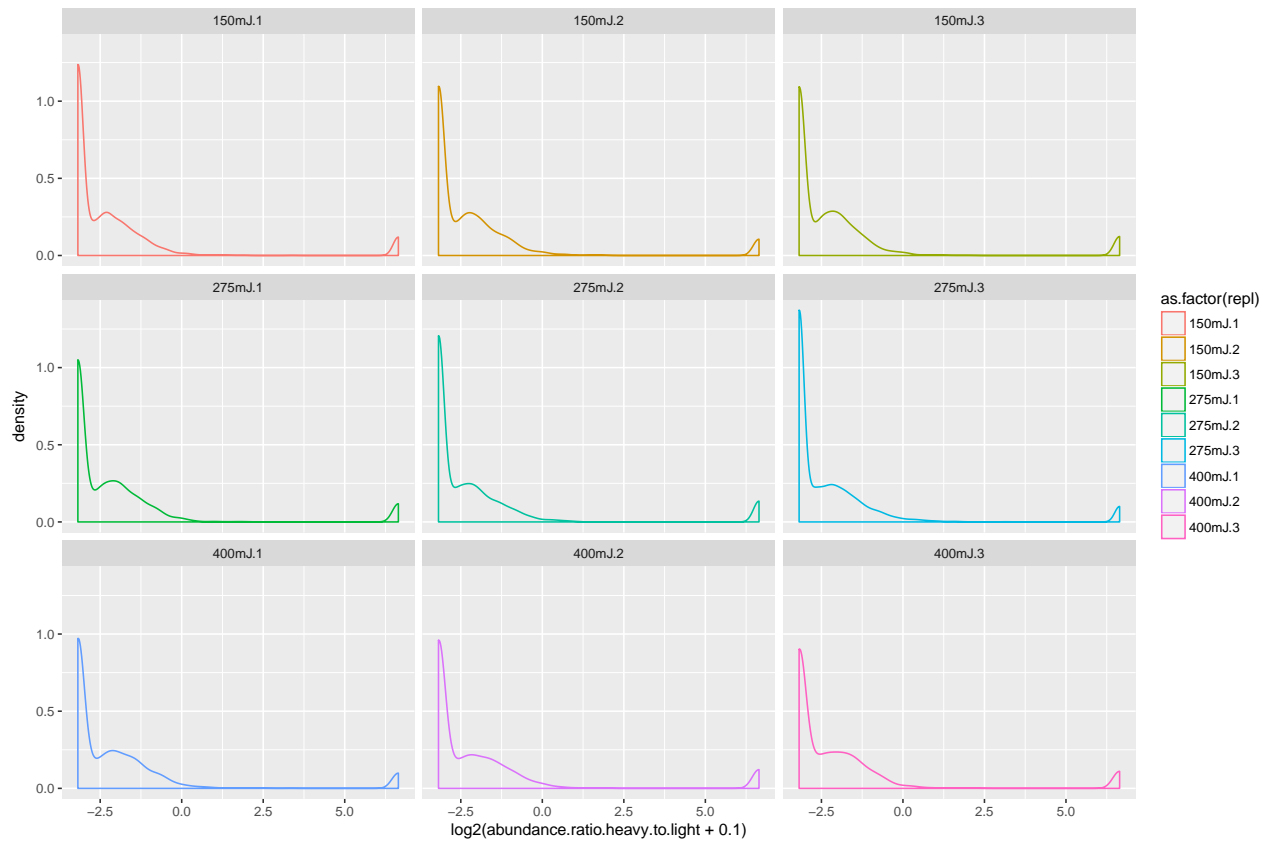
```
# Plot density plots
```

```
melt.1b = melt(filt.1b, id.vars = "repl", measure.vars = c("abundance.light", "abundance.heavy"))
```

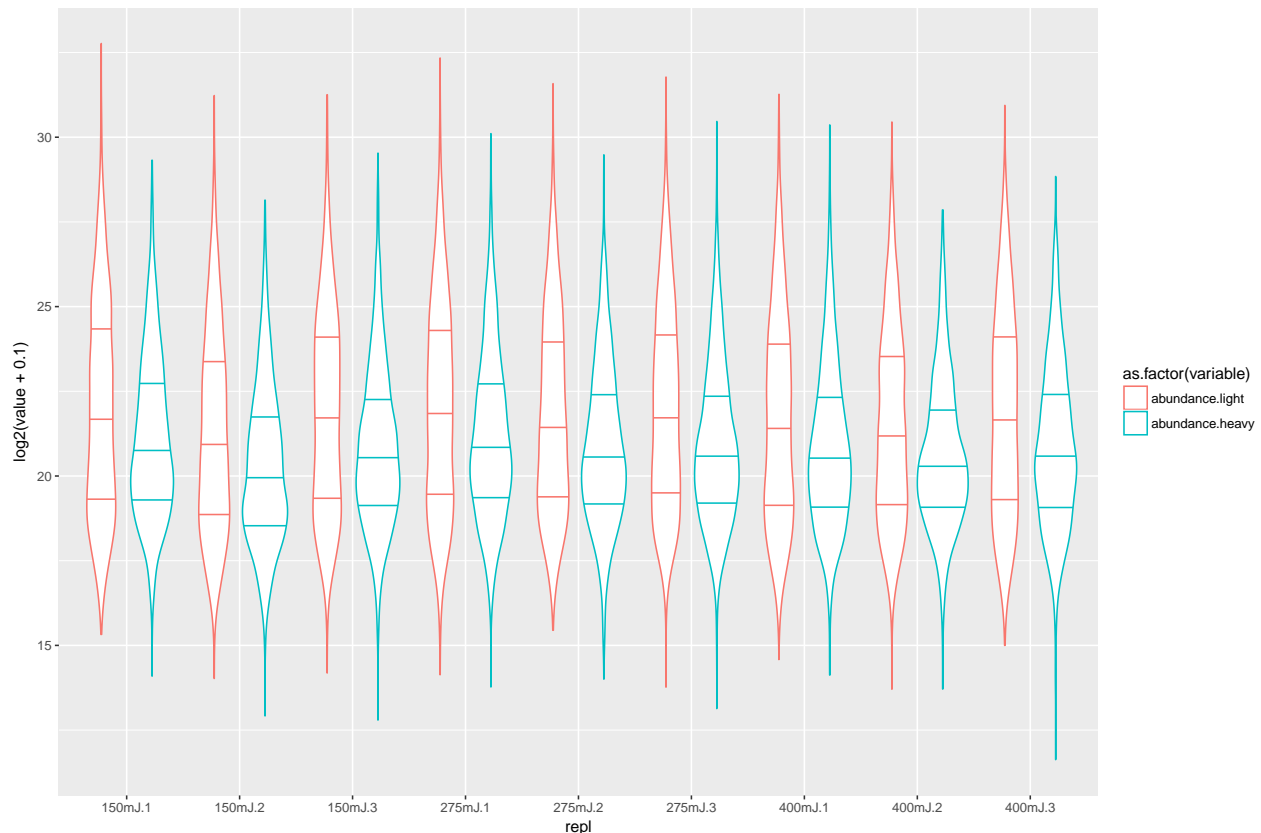
```
ggplot(melt.1b, aes(x = log2(value+0.1))) + geom_density(aes(col = as.factor(variable)))+facet_wrap(~repl)
```



```
ggplot(filt.1b, aes(x = log2(abundance.ratio.heavy.to.light+0.1))) + geom_density(aes(col = as.factor(repl)))
```



```
ggplot(melt.1b,aes(x=repl,y = log2(value+0.1))) + geom_violin(aes(col = as.factor(variable)),draw_quant
```



We have a column called “missing.val” to identify which peptides have either a heavy or light abundance value missing. TRUE means it is missing one or both. FALSE means both values are present. A lot more “missing” values in the “heavy/non-crosslinked samples than”light/cross-linked samples“.

The above plots - density plots and violin plots include both missing and non-missing values. Hence, in the heavy/light density plots, you see a huge overlap in the curve with a tiny portion of the “light” curve going beyond the heavy curve. This is where we expect the interesting RNA binding proteins to lie. The next step however is to filter out the missing values.

```
# Step 1c : Filter out those peptides which are missing either "high" or "low" abundance values. We will
# as we do not know for sure whether these are a result of extremely low signal due to enrichment or ex
# There are 16314 peptides where we have quantification in both light and heavy abundance columns
```

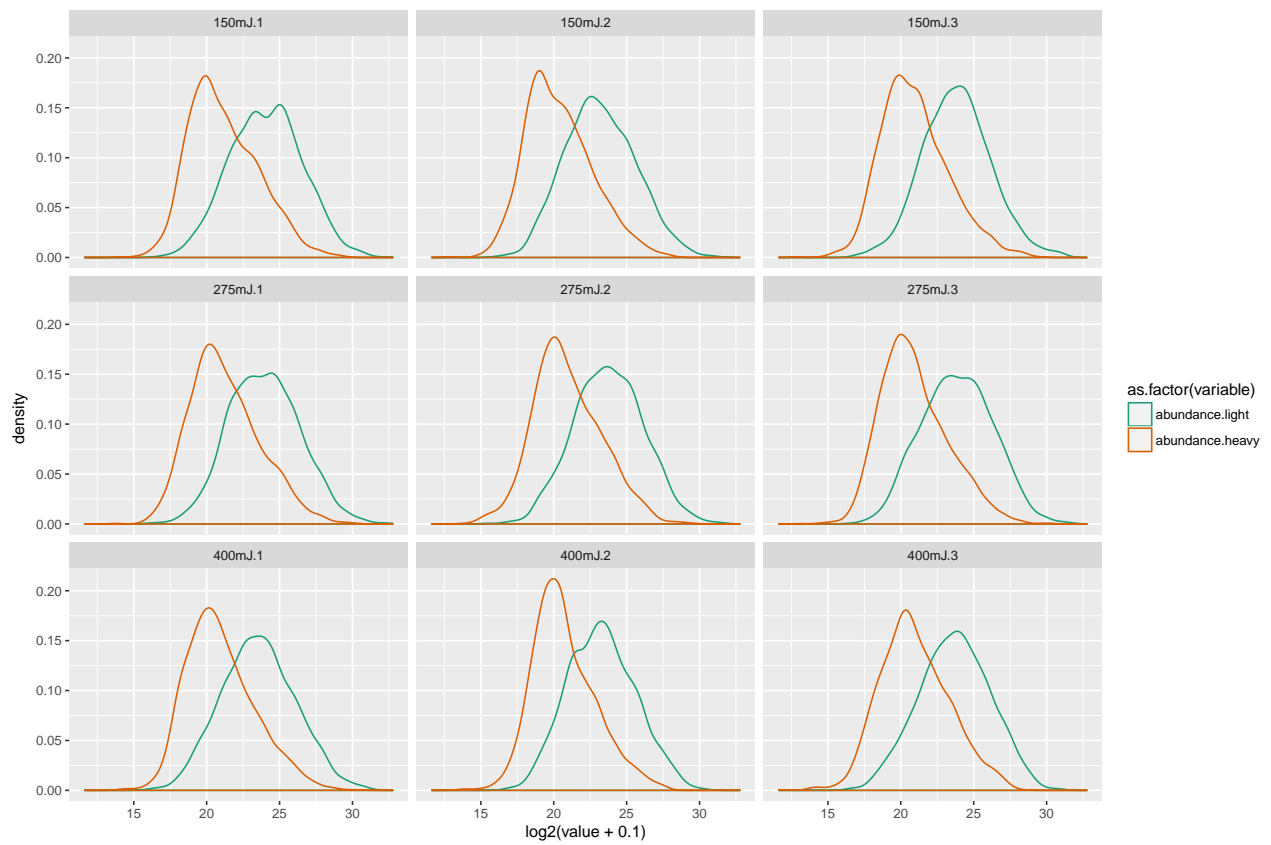
```
filt.1c = filt.1b[which(rowSums(is.na(filt.1b[,c("abundance.heavy", "abundance.light")])) == 0),]
dim(filt.1c)
```

```
## [1] 16314    33
```

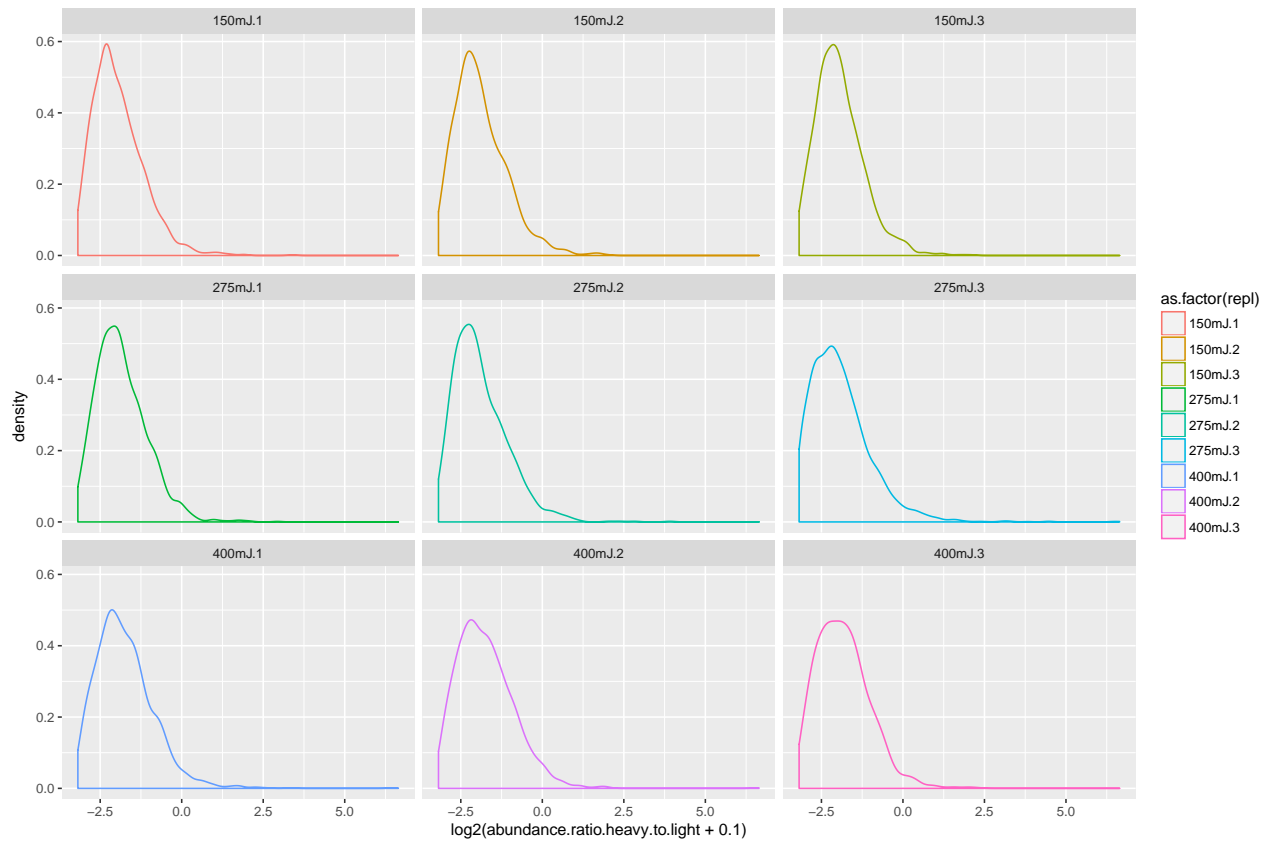
```
# Plot density plots
```

```
melt.1c = melt(filt.1c, id.vars = "repl", measure.vars = c("abundance.light", "abundance.heavy"))
```

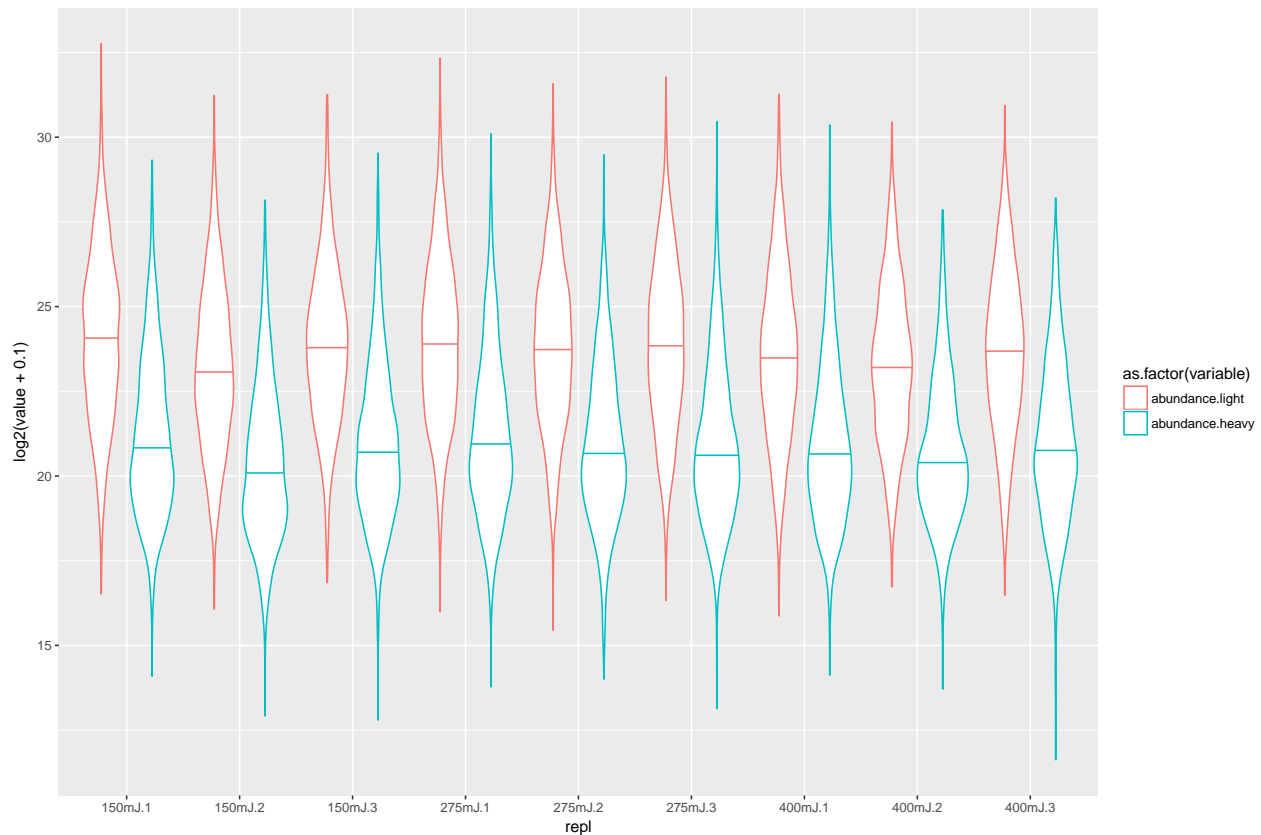
```
ggplot(melt.1c, aes(x = log2(value+0.1))) + geom_density(aes(col = as.factor(variable)))+facet_wrap(~repl)
```

```
ggplot(filt.1c,aes(x = log2(abundance.ratio.heavy.to.light+0.1))) + geom_density(aes(col = as.factor(rep)))
```



```
ggplot(melt.1c,aes(x=repl,y = log2(value+0.1))) + geom_violin(aes(col = as.factor(variable)),draw_quant
```



Once we remove peptides where the “heavy” or “light” value is missing, then there is a clear shift in the curve of intensity values for the “light” labelled sample which is our cross-linked sample and we hope it contains true RNA binding proteins. The median abundance for light samples is visibly higher than in heavy samples.

Note : It is important to remember that in a true experimental setting, we will not have SILAC labelling so we won't have “heavy” and “light” values per peptide - all we will have is one abundance value. Rayner forced the mass spec to run as if it didn't know about the SILAC labelling to partially emulate later experiments. However, we won't be using the singleton data (heavy only or light only) for the purposes of this initial analysis.

```
# -----
# Step 2 : Log-transform
# heavy = non-crosslinked
# light = crosslinked
# -----

# Log convert abundance values
filt.1c$heavy.log = log(filt.1c$abundance.heavy,2)
filt.1c$light.log = log(filt.1c$abundance.light,2)

# Generate an abundance ratio which for log transformed data is a subtraction
filt.1c$norm.abundance.ratio = filt.1c$light.log - filt.1c$heavy.log

# Data is checked
norm.data = filt.1c
dim(norm.data)

## [1] 16314    36
```

```
# Checking the counts of peaks with heavy and light values
table(light=norm.data$light.sample,heavy=norm.data$heavy.sample)
```

```
##
##      heavy
## light      High Peak Found
##   High         0      11258
##   Peak Found 5056         0
```

Once we have filtered the data to remove non-unique peptides and contaminants, we log transform (“normalise”) the data for heavy and light abundances. In addition, we subtract the logged abundances light-heavy to yield logged abundance ratios.

When we re-draw the table of heavy and light sample counts, we don’t have any “Not Found” values anymore. This was part of the exercise.

We have 16314 peptides that are present in both fractions. Analysing the difference in ratios between these two fractions are most likely to inform on whether or not they are enriched for RNA binding proteins.

```
# -----
# Step 3 : Aggregating multiple peptides into a peptide group
# heavy = non-crosslinked
# light = crosslinked
# -----
```

```
# Subset the data to include columns with useful metadata and abundance ratios
# Transform the dataframe using 'melt' so the values for heavy and light are in one column. Can use this
```

```
subset.cols = norm.data[,c("master.protein.accessions","sequence","modifications","repl","uv","missing.
dim(subset.cols)
```

```
## [1] 16314      9
```

```
# Tried various methods of aggregation
# Using sequence and repeat columns to aggregate
# Using all columns but the abundance ratio columns to aggregate
# Using mean, median or max to aggregate
# Using ddply as an alternative to aggregate
# Note : Finally, settled on taking the mean of the logged values and using 'aggregate' function
```

```
# We have 15356 unique peptide groups across all samples
agg.mean = aggregate(cbind(light.log,heavy.log,norm.abundance.ratio)~sequence+repl,data=subset.cols,FUN=
agg.pep.table = table(agg.mean$sequence,agg.mean$repl)
table(agg.mean$repl) # 275mJ, replicate 3 has a lot more peptide groups than other samples
```

```
##
## 150mJ.1 150mJ.2 150mJ.3 275mJ.1 275mJ.2 275mJ.3 400mJ.1 400mJ.2 400mJ.3
##    1802    1366    1590    1728    1275    2882    1894    1466    1353
```

```
# Now that peptides have been aggregated into peptide groups, re-calculate the missing value table...
write.table(agg.pep.table, paste(outdir,"Aggregated-peptides-no-missing-values.txt",sep="/"),sep="\t",qu
```

```
# Will go with agg.mean for further analysis
agg = agg.mean
```

```
# I want to add protein annotations back to the aggregated data. Just want to make sure that one peptid
for(i in 1:nrow(agg)){
  agg$num.prot[i] = length(unique(subset.cols$master.protein.accessions[which(subset.cols$sequence == a
```

```
agg$accessions[i] = paste(unique(subset.cols$master.protein.accessions[which(subset.cols$sequence == a
})
```

```
head(agg)
```

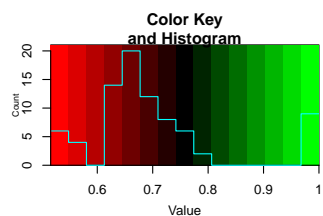
```
##           sequence    repl light.log heavy.log
## 1      AAAAAAALQAK 150mJ.1  27.91815  24.91056
## 2      AAAETQSLR  150mJ.1  26.22839  22.15931
## 3      AAAMANNLQK  150mJ.1  24.43873  20.25003
## 4 AAEAAPPTQEAQGETEPTQAPDALEQAADTSR 150mJ.1  19.35638  19.07300
## 5      AAGPISER  150mJ.1  22.39100  22.81245
## 6      AAGPSLSHTSGGTQSK 150mJ.1  20.91634  17.96804
## norm.abundance.ratio num.prot accessions
## 1          3.0075883      1      P36578
## 2          4.0690834      1      Q13523
## 3          4.1886989      1      Q14498
## 4          0.2833818      1      Q8N163
## 5         -0.4214498      1      Q15427
## 6          2.9483034      1      P27694
```

```
table(agg.mean$repl)
```

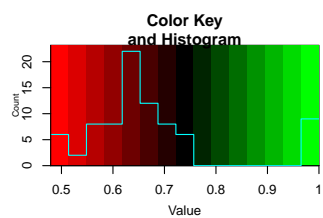
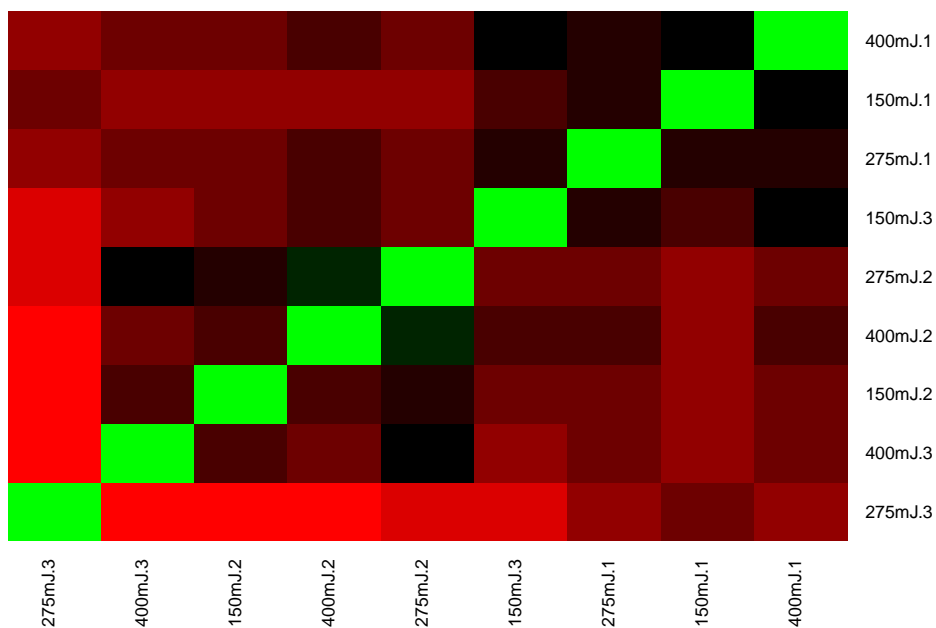
```
##
## 150mJ.1 150mJ.2 150mJ.3 275mJ.1 275mJ.2 275mJ.3 400mJ.1 400mJ.2 400mJ.3
##      1802      1366      1590      1728      1275      2882      1894      1466      1353
```

```
# Temporarily recast data into a matrix to calculate correlations
```

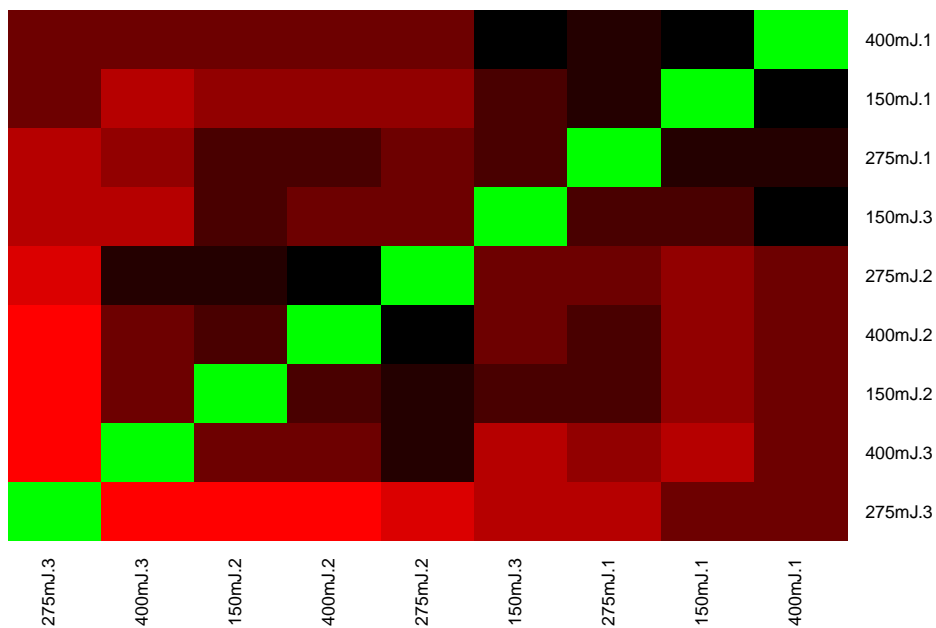
```
for (t in c("light.log", "heavy.log", "norm.abundance.ratio")){
  m = melt(agg, id.vars = c("sequence", "repl"), measure.vars = t)
  m.cast = dcast(m, sequence~repl+variable, fun.aggregate = mean)
  cor.m = cor(m.cast[,2:ncol(m.cast)], use="pairwise.complete.obs")
  colnames(cor.m) = gsub(paste("_", t, sep=""), "", colnames(cor.m))
  rownames(cor.m) = gsub(paste("_", t, sep=""), "", rownames(cor.m))
  heatmap.2(cor.m, trace = "none", dendrogram="none", col="redgreen", main=t)
}
```

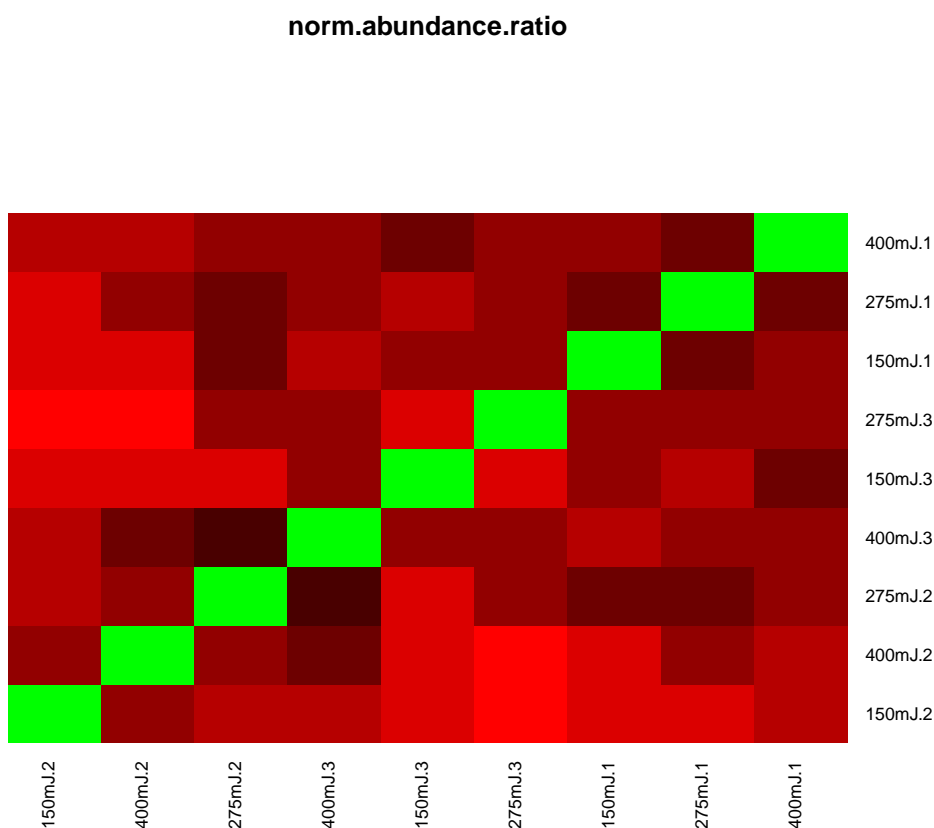
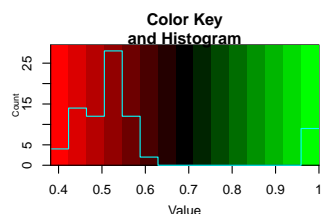


light.log



heavy.log





Looking at the correlations for ‘heavy’ and ‘light’ abundance values across all replicates, it looks like the correlation is more within experimental replicates i.e high for rep1 of 150mJ, 275mJ, 400mJ than between 150mJ.rep1 and 150mJ.rep2 and so on.

```
# -----
# Step 4 : Aggregating multiple peptides into one protein
# heavy = non-crosslinked
# light = crosslinked
# -----

# We have 1262 unique proteins across all samples
agg.prot = aggregate(cbind(light.log,heavy.log,norm.abundance.ratio)~accessions+repl,data=agg,FUN="mean",
dim(agg.prot))

## [1] 5749      5

# Table of proteins vs samples - contingency to say which protein is present in which sample.
# Will help make overlaps
agg.prot.table = table(agg.prot$accessions,agg.prot$repl)
write.table(agg.prot.table, paste(outdir,"Aggregated-proteins-no-missing-values.txt",sep="/"),sep="\t",
table(agg.prot$repl))

##
## 150mJ.1 150mJ.2 150mJ.3 275mJ.1 275mJ.2 275mJ.3 400mJ.1 400mJ.2 400mJ.3
##      676      544      600      648      511      943      696      594      537
```

There seem to be on average, ~640 proteins per sample in this experiment. 275mJ, rep3 has an unusually high number at 943. The samples at 150mJ of UV exposure have ~605 proteins, 275mJ have on average 716 proteins and 400mJ have on average 610 proteins.

```

# -----
# Step 5 : Looking for most commonly enriched proteins
# heavy = non-crosslinked
# light = crosslinked
# -----

# First let us look at the intersects within and between replicates
prot.matrix = as.data.frame.matrix(agg.prot.table)
print(dim(prot.matrix))

## [1] 1262    9

# Contains counts of overlap across 9 samples in various combinations
# Most intersections not very useful except that it tell us how many proteins overlap across all 9 samples
prot.venn = venn(prot.matrix, show.plot=F)
isect = attr(prot.venn, "intersections")

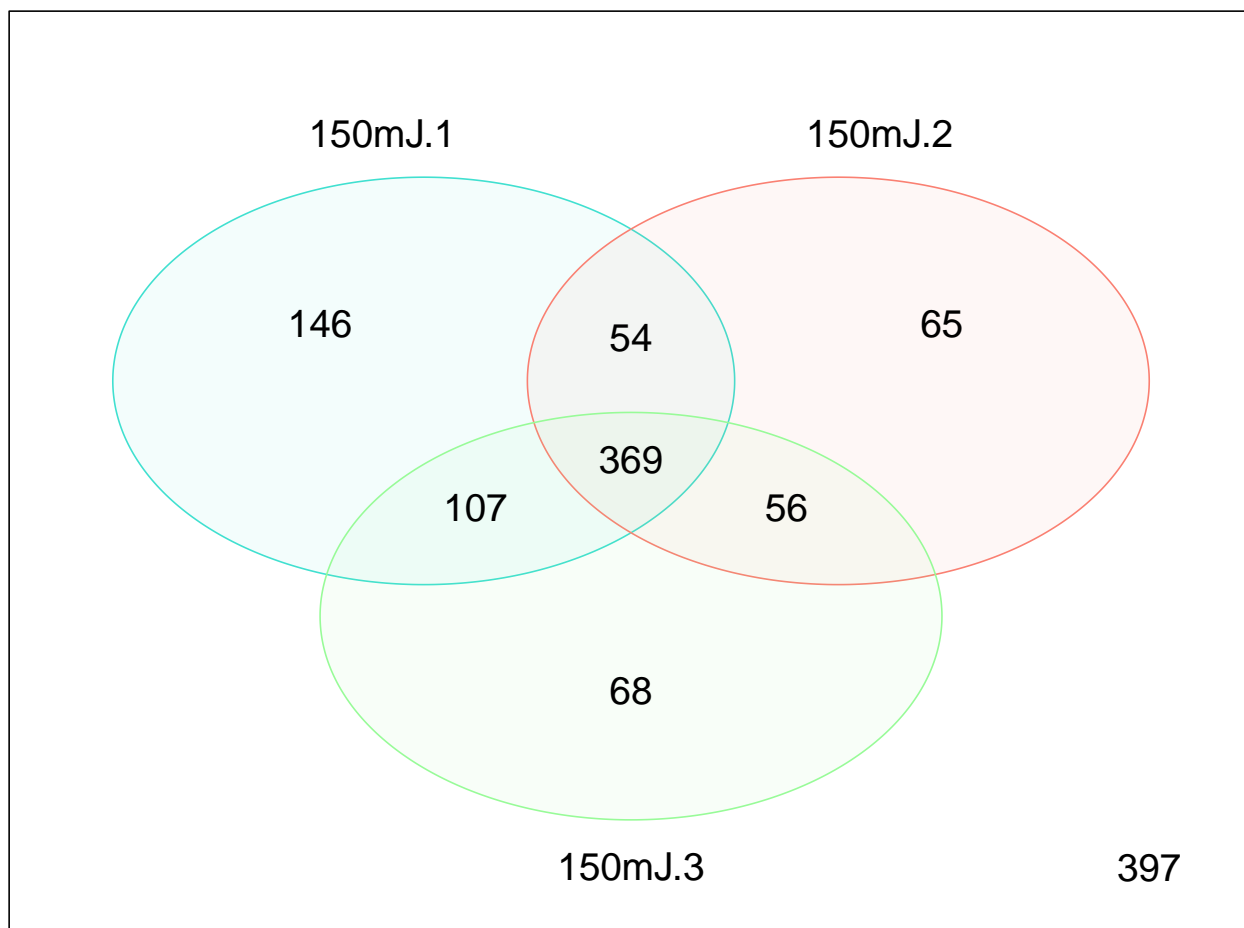
# table of intersections
isect.count = t(as.data.frame(lapply(isect, length)))
colnames(isect.count) = "Count"
write.table(isect.count, paste(outdir, "Count-of-protein-overlaps-across-various-samples.txt", sep="/"), sep=",", as.is=T)

# Looking at overlaps within each uv dose - the more useful intersection exercise
add.int = NULL

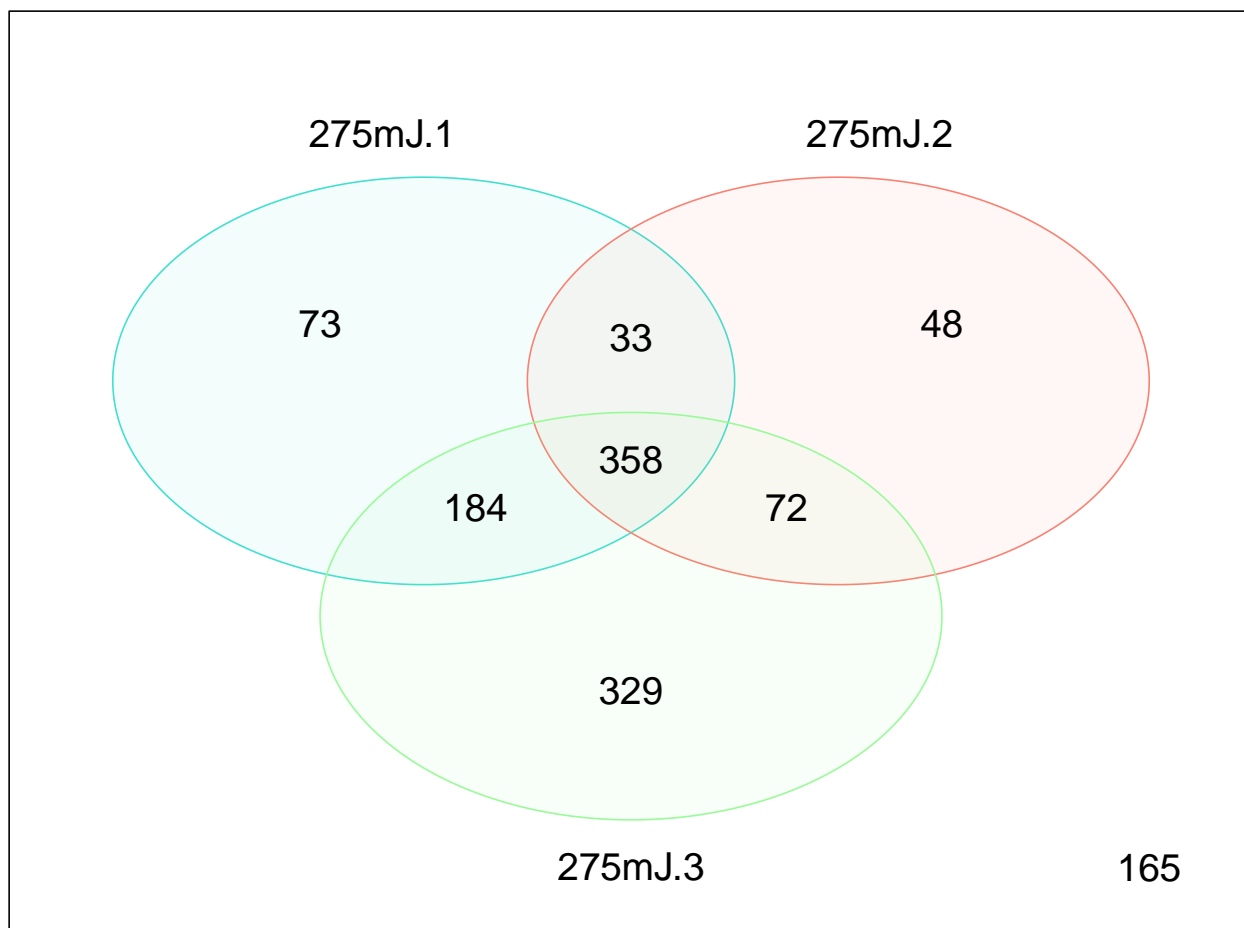
# Looping through each uv dosage triplicate - 1:3, 4:6, 7:9
# add.int contains all intersections for each triplicate
for(k in c(1,4,7)){
  print(k)
  prot.venn.tmp = venn(prot.matrix[,k:(k+2)], show.plot=F)
  vennDiagram(prot.matrix[,k:(k+2)], circle.col=c("turquoise", "salmon", "palegreen"))
  add.int = c(add.int, attr(prot.venn.tmp, "intersections"))
}

## [1] 1

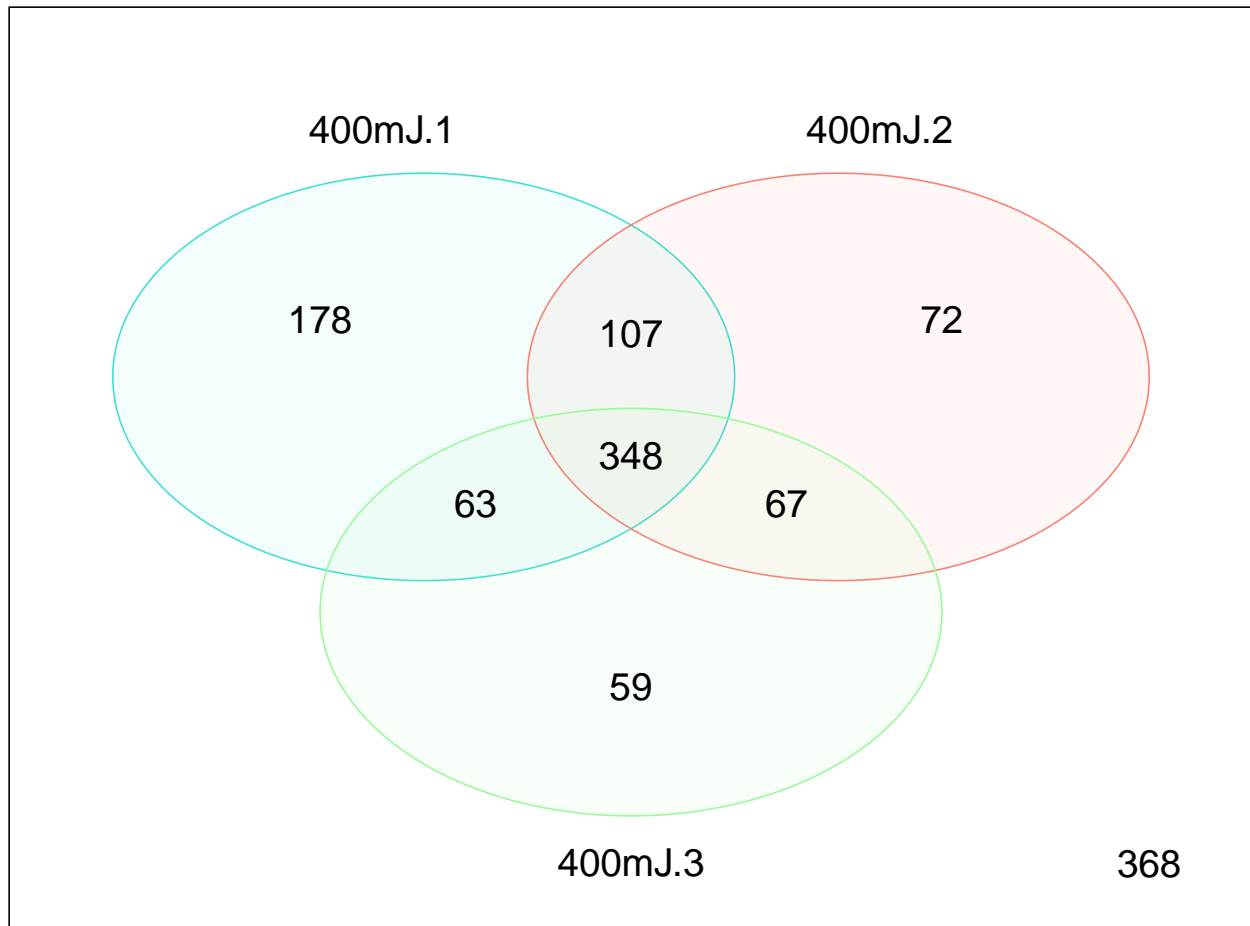
```

[1] 4



[1] 7



```
write.table(t(as.data.frame(lapply(add.int,length))), paste(outdir,"Count-of-protein-overlaps-within-rep",
```

The venn diagrams show the overlap of proteins within each uv dosage across replicates. There are between 350 and 370 overlapping proteins within each UV dosage. Across all 9 replicates, there are 211 proteins that we can extract as shown below. The next step is to map these proteins to some functional annotations. We will map each interaction group separately to see what it yields. Will use 'clusterProfiler' to do this.

```
isect[280]
```

```
## $`150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3`
## [1] "A0A024R4E5" "A0A087WUT6" "A0A087WVQ6" "A0A087X0X3" "A0A0A0MRV0"
## [6] "A0A0C4DG17" "A0A0C4DG49" "A0A0D9SF53" "A0A0G2JNW7" "A0A0U1RRM4"
## [11] "A8MXP9" "E7EVA0" "F5H2F4" "F8W930" "G8JLB6"
## [16] "H3BLZ8" "J3KPP4" "J3KTA4" "MQYS1" "O00203"
## [21] "O00567" "O15479" "O43175" "O43390" "O43493"
## [26] "O60506" "O75369" "O75400" "O75533" "O75534"
## [31] "O76021" "O95218" "P02545" "P02786" "P04406"
## [36] "P04792" "P05023" "P05556" "P06748" "P06756"
## [41] "P07195" "P07237" "P07355" "P07737" "P07814"
## [46] "P07900" "P08238" "P08621" "P08670" "P09429"
## [51] "P09619" "P09651" "P0C7U0" "P10809" "P10909"
## [56] "P11047" "P11142" "P11387" "P11940" "P13639"
## [61] "P13667" "P14618" "P14625" "P14866" "P15880"
## [66] "P16070" "P16989" "P18124" "P18583" "P18827"
## [71] "P19338" "P20700" "P21399" "P22314" "P22626"
```

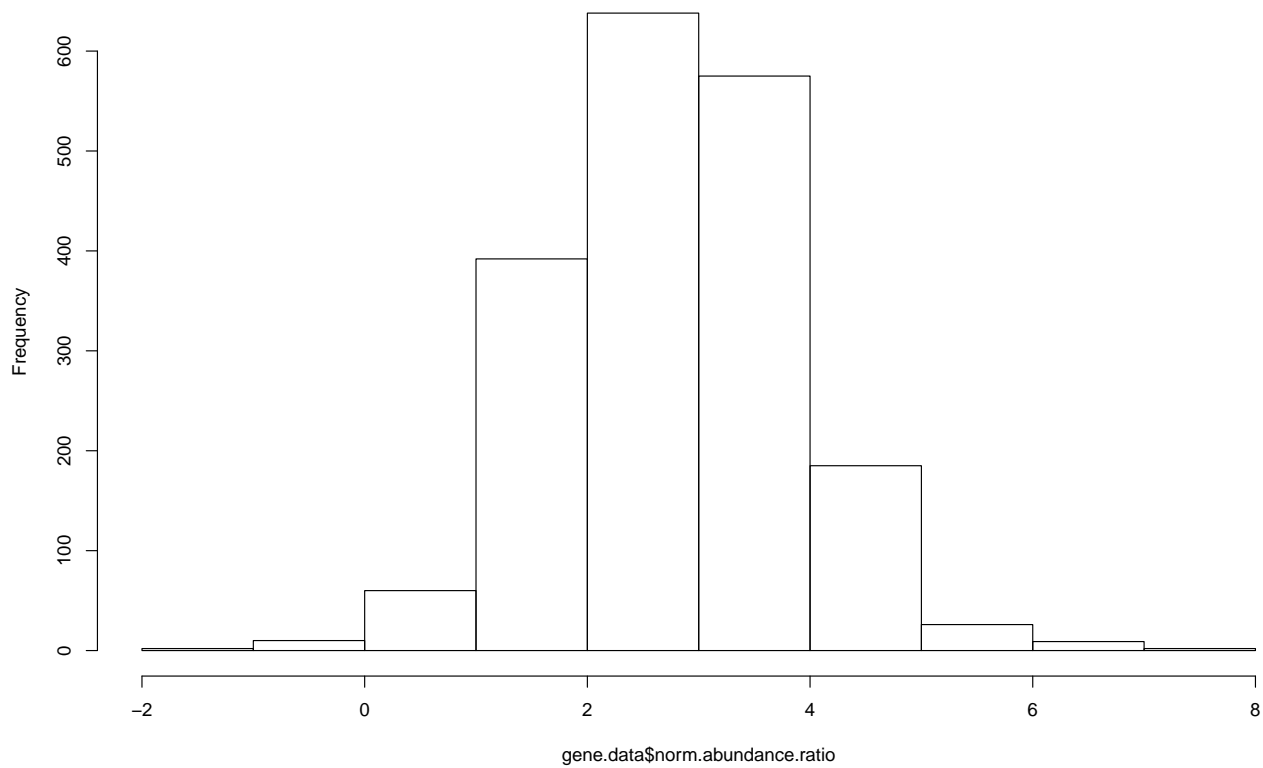
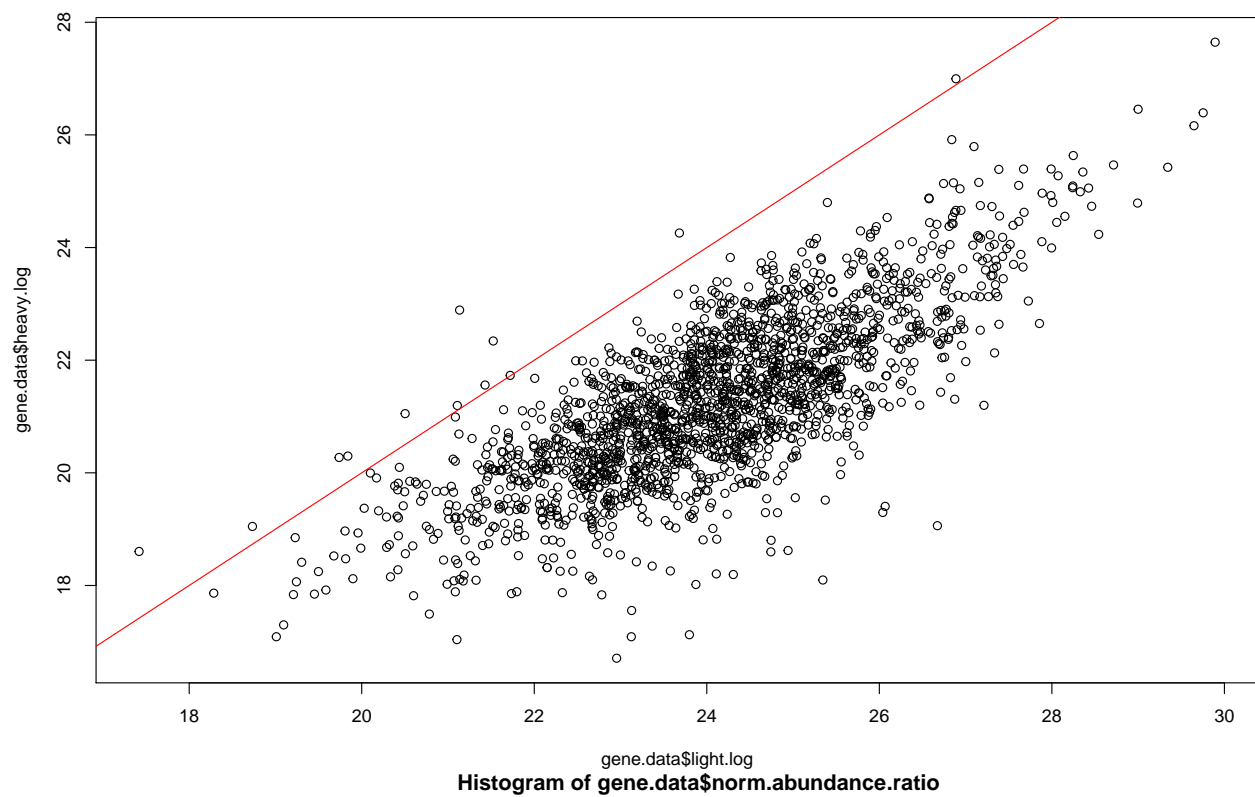
```
## [76] "P23246"      "P23396"      "P23526"      "P26006"      "P26373"
## [81] "P26641"      "P30101"      "P31942"      "P31948"      "P32004"
## [86] "P35052"      "P35579"      "P35613"      "P35659"      "P36578"
## [91] "P37802"      "P38646"      "P39023"      "P42704"      "P42892"
## [96] "P43121"      "P46777"      "P46781"      "P46976"      "P47914"
## [101] "P48634"      "P49327"      "P49411"      "P49588"      "P49756"
## [106] "P50454"      "P50895"      "P50914"      "P50990"      "P51991"
## [111] "P52597"      "P53396"      "P55011"      "P55072"      "P55290"
## [116] "P55795"      "P60709"      "P61247"      "P61978"      "P62241"
## [121] "P62249"      "P62263"      "P62269"      "P62424"      "P62753"
## [126] "P62913"      "P62995"      "P67809"      "P78527"      "Q00839"
## [131] "Q01105"      "Q01130"      "Q01844"      "Q02878"      "Q05519"
## [136] "Q07065"      "Q07666"      "Q07954"      "Q08170"      "Q08211"
## [141] "Q08945"      "Q09666"      "Q12849"      "Q12906"      "Q13148"
## [146] "Q13151"      "Q13243"      "Q13263"      "Q13283"      "Q13641"
## [151] "Q13740"      "Q14103"      "Q14108"      "Q14152"      "Q14315"
## [156] "Q14444"      "Q14498"      "Q14690"      "Q15061"      "Q15084"
## [161] "Q15149"      "Q15233"      "Q15287"      "Q15717"      "Q15758"
## [166] "Q15904"      "Q16629"      "Q16658"      "Q5BKZ1"      "Q5T6F2"
## [171] "Q6PD62"      "Q6UVK1"      "Q7KZF4"      "Q7L2E3"      "Q7L4I2"
## [176] "Q7Z3B1"      "Q86SJ2"      "Q8N7H5"      "Q8NC51"      "Q8NE71"
## [181] "Q8WVC0"      "Q92541"      "Q92879"      "Q92945"      "Q96AE4"
## [186] "Q96I24"      "Q96KR1"      "Q96PK6"      "Q96T37"      "Q99700"
## [191] "Q99714"      "Q9BRL6"      "Q9BUQ8"      "Q9H307"      "Q9NR30"
## [196] "Q9NUM4"      "Q9NWH9"      "Q9NZB2"      "Q9P121"      "Q9UKM9"
## [201] "Q9UQ35"      "Q9UQ80"      "Q9Y2W1"      "Q9Y2X3"      "Q9Y383"
## [206] "Q9Y3Y2"      "Q9Y490"      "Q9Y4C8"      "Q9Y4L1"      "Q9Y520"
## [211] "X5DQS5"
```

Displaying the `length(isect[280])` proteins that are enriched across all 9 replicate samples across 3 different UV dosages. We hope that this is the core set of RBPs we could use as a positive control later on in the project. Need to see what these proteins are and work out the rate of false positives.

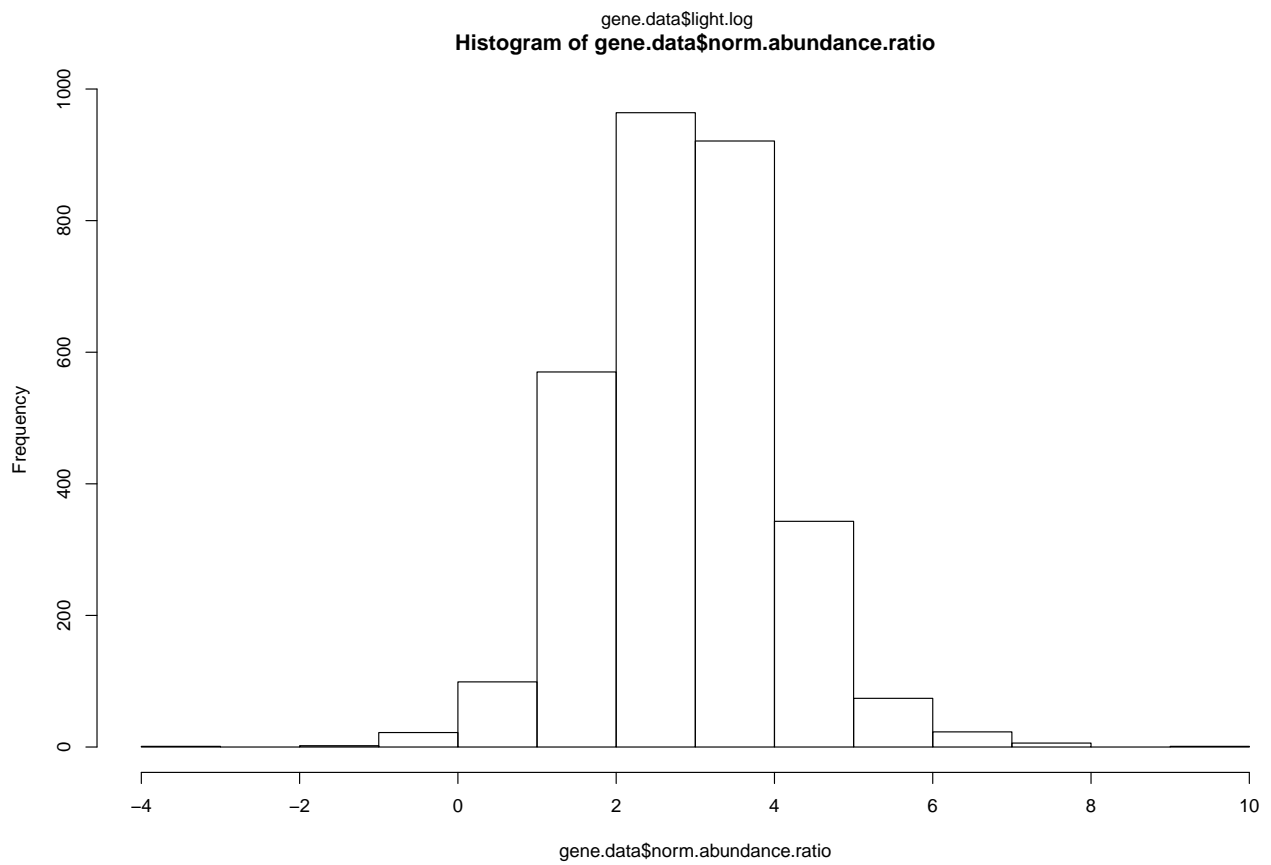
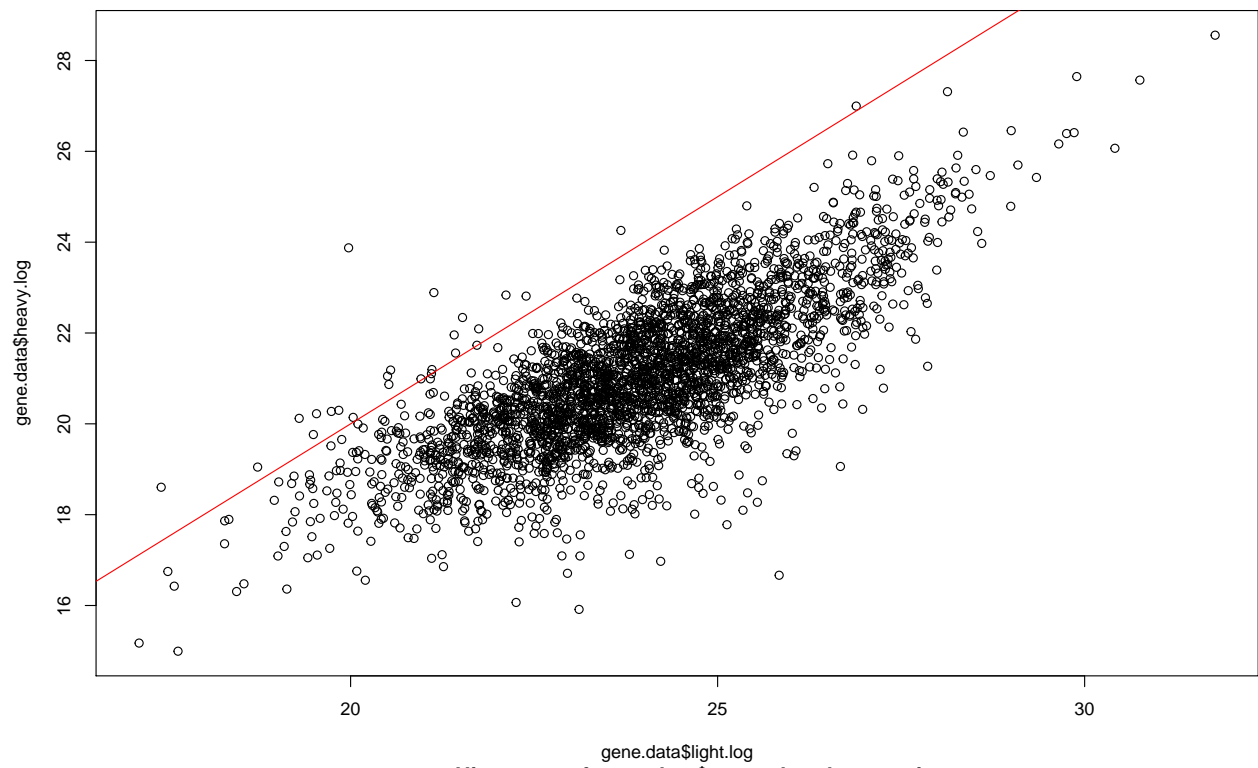
```
# -----
# Step 6: Functional Enrichment
# Using KEGG pathways enrichment for the intersections of proteins within replicates
# Designate proteins as up(enriched) or down(not-enriched) in light:heavy relative to crosslink:non-crosslink
# Main interactions of interest are (1) across all 9 samples (n = 211) (2) Overlap within each triplicate
# -----

# A protein universe to use as background ??
prot.univ = bitr(unique(filt.1a$master.protein.accessions), fromType="UNIPROT", toType=c("ENTREZID"), O

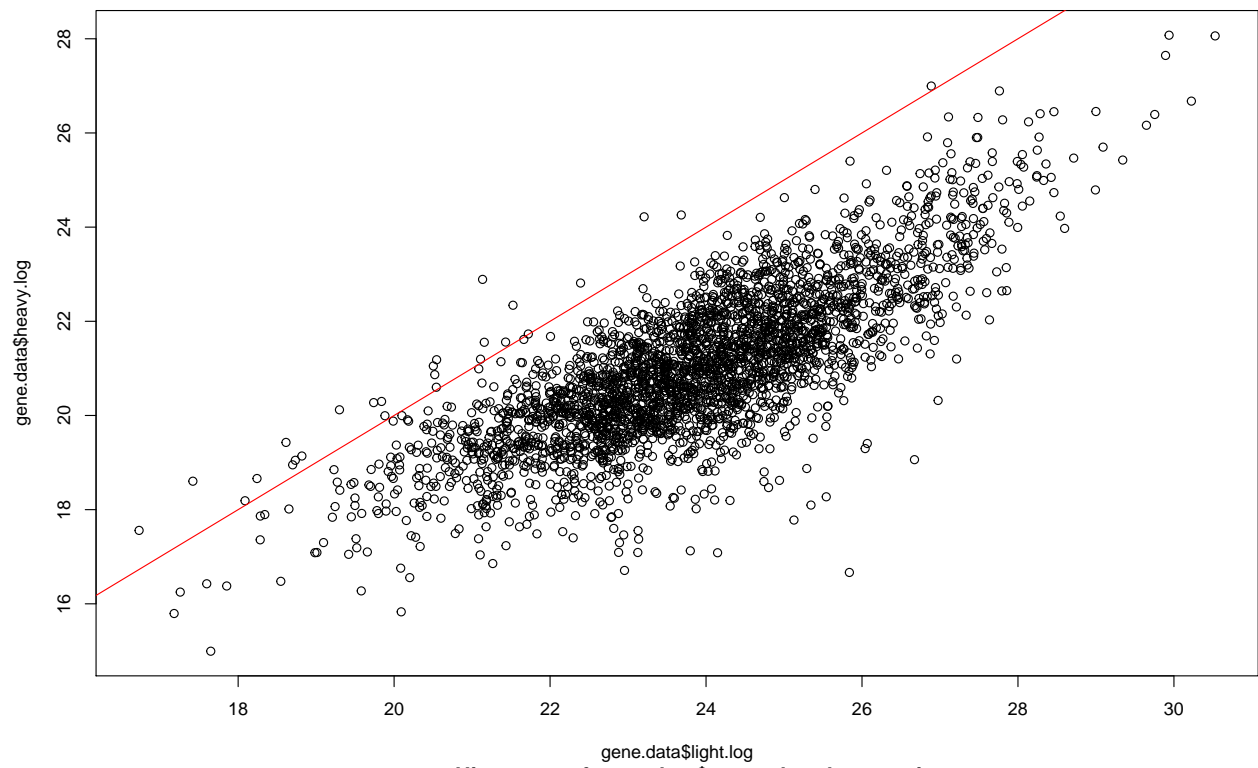
# Calling the function 'enrichKEGG' on intersections of interest
across.9.kegg = enrichK(isect,280,agg.prot,0.05,outdir)
```



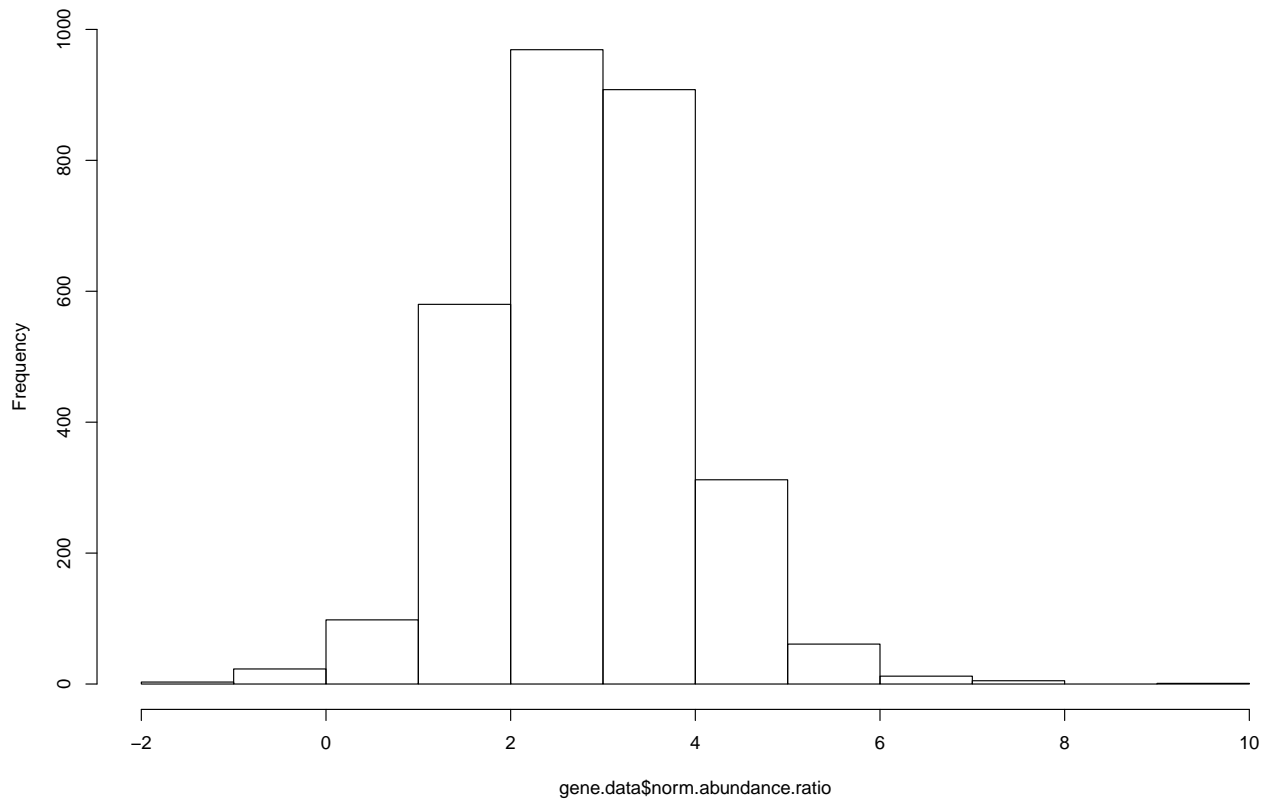
```
across.150mJ = enrichK(add.int, "150mJ.1:150mJ.2:150mJ.3", agg.prot, 0.05, outdir)
```



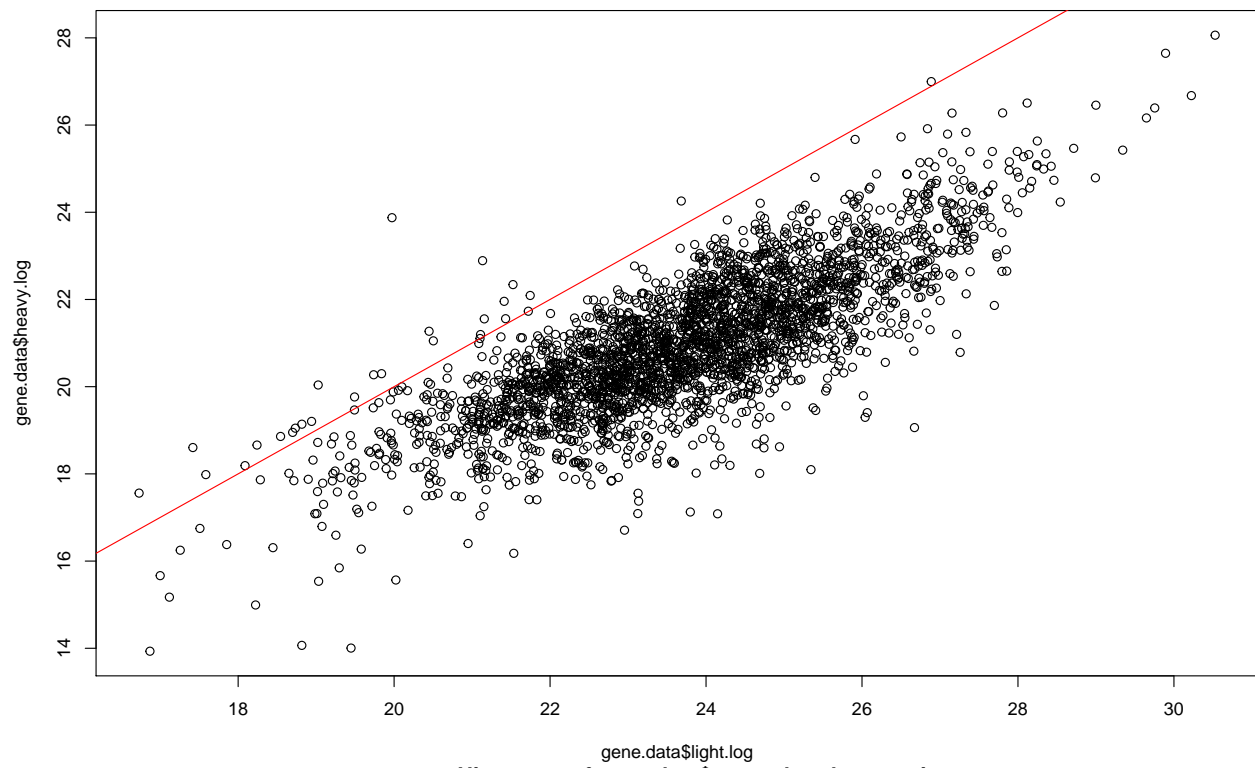
```
across.275mJ = enrichK(add.int,"275mJ.1:275mJ.2:275mJ.3",agg.prot,0.05,outdir)
```



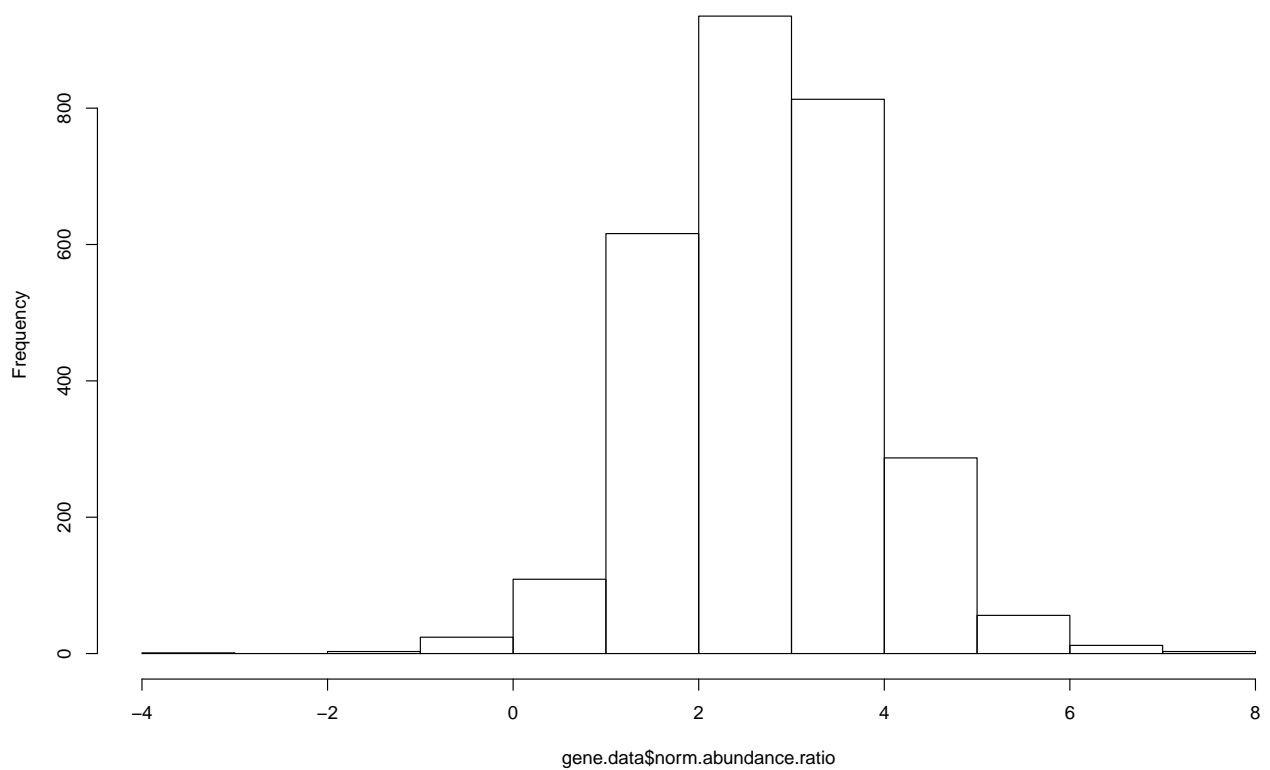
Histogram of `gene.data$norm.abundance.ratio`



```
across.400mJ = enrichK(add.int,"400mJ.1:400mJ.2:400mJ.3",agg.prot,0.05,outdir)
```



Histogram of `gene.data$norm.abundance.ratio`



```
# Binding enriched KEGG pathway outcomes for all comparisons into one data frame for output
all.kegg = rbind(across.9.kegg, across.150mJ, across.275mJ, across.400mJ)
write.table(all.kegg, paste(outdir, "KEGG-enrichment-for-enriched-proteins.txt", sep="/"), sep="\t", quote=)
```


Not sure what to define the protein “universe” as. Used all of the proteins in the aggregated list but this is not sufficient to run the KEGG analysis (throws a “not sufficient members in group” error. Need to read a bit more about the inner workings of enrichKEGG to see if this can be changed.

Meanwhile, the overlapping proteins across all samples are enriched for the terms “Ribosome”, “Spliceosome”, “Protein processing in ER”, “Cell adhesion molecules” etc... Rayner concerned about presence of proteoglycans as these could be unwanted members entering the interface. Experiments are underway to check this.

I have also done an enrichment for proteins that were common within triplicate and each UV dosage. Get very similar terms as before (which is expected) and a few extra. The 150mJ dosage has the most number of significant KEGG mappings of the three dosages. There are a few pathways that aren’t enriched in the crosslinked sample (Down) but majority are. There are instances where the term “spliceosome” appears in both enriched and un-enriched categories but the genes that contribute to this KEGG term are different in the enriched and unenriched cases. Might be worth pursuing these genes that in the unenriched category - they are heterogeneous nuclear ribonucleo protein and small nuclear ribonucleoprotein, RNA helicase and splicing factor subunit.

The 275mJ dosage has a high number of histones which map to pathways such as Systemic lupus erythematosus, Viral carcinogenesis, ECM-receptor interaction and Alcoholism which are a bit odd. If you remember, 275mJ has on average more proteins per sample than the other two time points. Perhaps this isn’t the ideal UV dosage for the study.

```
#-----
# 07 : Plotting cluster membership
# Diagrammatic representation of functional overlap/replicability
#-----
head(all.kegg)
```

```
##          ID          Description GeneRatio
## hsa03010 hsa03010          Ribosome    20/113
## hsa03040 hsa03040          Spliceosome   17/113
## hsa04141 hsa04141 Protein processing in endoplasmic reticulum 11/113
## hsa05205 hsa05205          Proteoglycans in cancer   10/113
## hsa04512 hsa04512          ECM-receptor interaction    6/113
## hsa04514 hsa04514          Cell adhesion molecules (CAMs) 8/113
##          BgRatio      pvalue      p.adjust      qvalue
## hsa03010 154/7251 1.482033e-13 1.956284e-11 1.747239e-11
## hsa03040 134/7251 1.733643e-11 1.144204e-09 1.021937e-09
## hsa04141 166/7251 5.021528e-05 2.209473e-03 1.973373e-03
## hsa05205 203/7251 1.196691e-03 3.949079e-02 3.527088e-02
## hsa04512  82/7251 1.678011e-03 4.046078e-02 3.613722e-02
## hsa04514 145/7251 1.839126e-03 4.046078e-02 3.613722e-02
##
## hsa03010 3921/6187/6129/6188/6137/6124/6122/6125/6203/6159/9045/6189/6202/6217/6208/6222/6130/6194/6195/6196/6197/6198/6199/6200/6201/6202/6203/6204/6205/6206/6207/6208/6209/6210/6211/6212/6213/6214/6215/6216/6217/6218/6219/6220/6221/6222/6223/6224/6225/6226/6227/6228/6229/6230/6231/6232/6233/6234/6235/6236/6237/6238/6239/6240/6241/6242/6243/6244/6245/6246/6247/6248/6249/6250/6251/6252/6253/6254/6255/6256/6257/6258/6259/6260/6261/6262/6263/6264/6265/6266/6267/6268/6269/6270/6271/6272/6273/6274/6275/6276/6277/6278/6279/6280/6281/6282/6283/6284/6285/6286/6287/6288/6289/6290/6291/6292/6293/6294/6295/6296/6297/6298/6299/6300/6301/6302/6303/6304/6305/6306/6307/6308/6309/6310/6311/6312/6313/6314/6315/6316/6317/6318/6319/6320/6321/6322/6323/6324/6325/6326/6327/6328/6329/6330/6331/6332/6333/6334/6335/6336/6337/6338/6339/6340/6341/6342/6343/6344/6345/6346/6347/6348/6349/6350/6351/6352/6353/6354/6355/6356/6357/6358/6359/6360/6361/6362/6363/6364/6365/6366/6367/6368/6369/6370/6371/6372/6373/6374/6375/6376/6377/6378/6379/6380/6381/6382/6383/6384/6385/6386/6387/6388/6389/6390/6391/6392/6393/6394/6395/6396/6397/6398/6399/6400/6401/6402/6403/6404/6405/6406/6407/6408/6409/6410/6411/6412/6413/6414/6415/6416/6417/6418/6419/6420/6421/6422/6423/6424/6425/6426/6427/6428/6429/6430/6431/6432/6433/6434/6435/6436/6437/6438/6439/6440/6441/6442/6443/6444/6445/6446/6447/6448/6449/6450/6451/6452/6453/6454/6455/6456/6457/6458/6459/6460/6461/6462/6463/6464/6465/6466/6467/6468/6469/6470/6471/6472/6473/6474/6475/6476/6477/6478/6479/6480/6481/6482/6483/6484/6485/6486/6487/6488/6489/6490/6491/6492/6493/6494/6495/6496/6497/6498/6499/6500/6501/6502/6503/6504/6505/6506/6507/6508/6509/6510/6511/6512/6513/6514/6515/6516/6517/6518/6519/6520/6521/6522/6523/6524/6525/6526/6527/6528/6529/6530/6531/6532/6533/6534/6535/6536/6537/6538/6539/6540/6541/6542/6543/6544/6545/6546/6547/6548/6549/6550/6551/6552/6553/6554/6555/6556/6557/6558/6559/6560/6561/6562/6563/6564/6565/6566/6567/6568/6569/6570/6571/6572/6573/6574/6575/6576/6577/6578/6579/6580/6581/6582/6583/6584/6585/6586/6587/6588/6589/6590/6591/6592/6593/6594/6595/6596/6597/6598/6599/6600/6601/6602/6603/6604/6605/6606/6607/6608/6609/6610/6611/6612/6613/6614/6615/6616/6617/6618/6619/6620/6621/6622/6623/6624/6625/6626/6627/6628/6629/6630/6631/6632/6633/6634/6635/6636/6637/6638/6639/6640/6641/6642/6643/6644/6645/6646/6647/6648/6649/6650/6651/6652/6653/6654/6655/6656/6657/6658/6659/6660/6661/6662/6663/6664/6665/6666/6667/6668/6669/6670/6671/6672/6673/6674/6675/6676/6677/6678/6679/6680/6681/6682/6683/6684/6685/6686/6687/6688/6689/6690/6691/6692/6693/6694/6695/6696/6697/6698/6699/6700/6701/6702/6703/6704/6705/6706/6707/6708/6709/6710/6711/6712/6713/6714/6715/6716/6717/6718/6719/6720/6721/6722/6723/6724/6725/6726/6727/6728/6729/6730/6731/6732/6733/6734/6735/6736/6737/6738/6739/6740/6741/6742/6743/6744/6745/6746/6747/6748/6749/6750/6751/6752/6753/6754/6755/6756/6757/6758/6759/6760/6761/6762/6763/6764/6765/6766/6767/6768/6769/6770/6771/6772/6773/6774/6775/6776/6777/6778/6779/6780/6781/6782/6783/6784/6785/6786/6787/6788/6789/6790/6791/6792/6793/6794/6795/6796/6797/6798/6799/6800/6801/6802/6803/6804/6805/6806/6807/6808/6809/6810/6811/6812/6813/6814/6815/6816/6817/6818/6819/6820/6821/6822/6823/6824/6825/6826/6827/6828/6829/6830/6831/6832/6833/6834/6835/6836/6837/6838/6839/6840/6841/6842/6843/6844/6845/6846/6847/6848/6849/6850/6851/6852/6853/6854/6855/6856/6857/6858/6859/6860/6861/6862/6863/6864/6865/6866/6867/6868/6869/6870/6871/6872/6873/6874/6875/6876/6877/6878/6879/6880/6881/6882/6883/6884/6885/6886/6887/6888/6889/6890/6891/6892/6893/6894/6895/6896/6897/6898/6899/6900/6901/6902/6903/6904/6905/6906/6907/6908/6909/6910/6911/6912/6913/6914/6915/6916/6917/6918/6919/6920/6921/6922/6923/6924/6925/6926/6927/6928/6929/6930/6931/6932/6933/6934/6935/6936/6937/6938/6939/6940/6941/6942/6943/6944/6945/6946/6947/6948/6949/6950/6951/6952/6953/6954/6955/6956/6957/6958/6959/6960/6961/6962/6963/6964/6965/6966/6967/6968/6969/6970/6971/6972/6973/6974/6975/6976/6977/6978/6979/6980/6981/6982/6983/6984/6985/6986/6987/6988/6989/6990/6991/6992/6993/6994/6995/6996/6997/6998/6999/7000/7001/7002/7003/7004/7005/7006/7007/7008/7009/7010/7011/7012/7013/7014/7015/7016/7017/7018/7019/7020/7021/7022/7023/7024/7025/7026/7027/7028/7029/7030/7031/7032/7033/7034/7035/7036/7037/7038/7039/7040/7041/7042/7043/7044/7045/7046/7047/7048/7049/7050/7051/7052/7053/7054/7055/7056/7057/7058/7059/7060/7061/7062/7063/7064/7065/7066/7067/7068/7069/7070/7071/7072/7073/7074/7075/7076/7077/7078/7079/7080/7081/7082/7083/7084/7085/7086/7087/7088/7089/7090/7091/7092/7093/7094/7095/7096/7097/7098/7099/7100/7101/7102/7103/7104/7105/7106/7107/7108/7109/7110/7111/7112/7113/7114/7115/7116/7117/7118/7119/7120/7121/7122/7123/7124/7125/7126/7127/7128/7129/7130/7131/7132/7133/7134/7135/7136/7137/7138/7139/7140/7141/7142/7143/7144/7145/7146/7147/7148/7149/7150/7151/7152/7153/7154/7155/7156/7157/7158/7159/7160/7161/7162/7163/7164/7165/7166/7167/7168/7169/7170/7171/7172/7173/7174/7175/7176/7177/7178/7179/7180/7181/7182/7183/7184/7185/7186/7187/7188/7189/7190/7191/7192/7193/7194/7195/7196/7197/7198/7199/7200/7201/7202/7203/7204/7205/7206/7207/7208/7209/7210/7211/7212/7213/7214/7215/7216/7217/7218/7219/7220/7221/7222/7223/7224/7225/7226/7227/7228/7229/7230/7231/7232/7233/7234/7235/7236/7237/7238/7239/7240/7241/7242/7243/7244/7245/7246/7247/7248/7249/7250/7251/7252/7253/7254/7255/7256/7257/7258/7259/7260/7261/7262/7263/7264/7265/7266/7267/7268/7269/7270/7271/7272/7273/7274/7275/7276/7277/7278/7279/7280/7281/7282/7283/7284/7285/7286/7287/7288/7289/7290/7291/7292/7293/7294/7295/7296/7297/7298/7299/7300/7301/7302/7303/7304/7305/7306/7307/7308/7309/7310/7311/7312/7313/7314/7315/7316/7317/7318/7319/7320/7321/7322/7323/7324/7325/7326/7327/7328/7329/7330/7331/7332/7333/7334/7335/7336/7337/7338/7339/7340/7341/7342/7343/7344/7345/7346/7347/7348/7349/7350/7351/7352/7353/7354/7355/7356/7357/7358/7359/7360/7361/7362/7363/7364/7365/7366/7367/7368/7369/7370/7371/7372/7373/7374/7375/7376/7377/7378/7379/7380/7381/7382/7383/7384/7385/7386/7387/7388/7389/7390/7391/7392/7393/7394/7395/7396/7397/7398/7399/7400/7401/7402/7403/7404/7405/7406/7407/7408/7409/7410/7411/7412/7413/7414/7415/7416/7417/7418/7419/7420/7421/7422/7423/7424/7425/7426/7427/7428/7429/7430/7431/7432/7433/7434/7435/7436/7437/7438/7439/7440/7441/7442/7443/7444/7445/7446/7447/7448/7449/7450/7451/7452/7453/7454/7455/7456/7457/7458/7459/7460/7461/7462/7463/7464/7465/7466/7467/7468/7469/7470/7471/7472/7473/7474/7475/7476/7477/7478/7479/7480/7481/7482/7483/7484/7485/7486/7487/7488/7489/7490/7491/7492/7493/7494/7495/7496/7497/7498/7499/7500/7501/7502/7503/7504/7505/7506/7507/7508/7509/7510/7511/7512/7513/7514/7515/7516/7517/7518/7519/7520/7521/7522/7523/7524/7525/7526/7527/7528/7529/7530/7531/7532/7533/7534/7535/7536/7537/7538/7539/7540/7541/7542/7543/7544/7545/7546/7547/7548/7549/7550/7551/7552/7553/7554/7555/7556/7557/7558/7559/7560/7561/7562/7563/7564/7565/7566/7567/7568/7569/7570/7571/7572/7573/7574/7575/7576/7577/7578/7579/7580/7581/7582/7583/7584/7585/7586/7587/7588/7589/7590/7591/7592/7593/7594/7595/7596/7597/7598/7599/7600/7601/7602/7603/7604/7605/7606/7607/7608/7609/7610/7611/7612/7613/7614/7615/7616/7617/7618/7619/7620/7621/7622/7623/7624/7625/7626/7627/7628/7629/7630/7631/7632/7633/7634/7635/7636/7637/7638/7639/7640/7641/7642/7643/7644/7645/7646/7647/7648/7649/7650/7651/7652/7653/7654/7655/7656/7657/7658/7659/7660/7661/7662/7663/7664/7665/7666/7667/7668/7669/7670/7671/7672/7673/7674/7675/7676/7677/7678/7679/7680/7681/7682/7683/7684/7685/7686/7687/7688/7689/7690/7691/7692/7693/7694/7695/7696/7697/7698/7699/7700/7701/7702/7703/7704/7705/7706/7707/7708/7709/7710/7711/7712/7713/7714/7715/7716/7717/7718/7719/7720/7721/7722/7723/7724/7725/7726/7727/7728/7729/7730/7731/7732/7733/7734/7735/7736/7737/7738/7739/7740/7741/7742/7743/7744/7745/7746/7747/7748/7749/7750/7751/7752/7753/7754/7755/7756/7757/7758/7759/7760/7761/7762/7763/7764/7765/7766/7767/7768/7769/7770/7771/7772/7773/7774/7775/7776/7777/7778/7779/7780/7781/7782/7783/7784/7785/7786/7787/7788/7789/7790/7791/7792/7793/7794/7795/7796/7797/7798/7799/7800/7801/7802/7803/7804/7805/7806/7807/7808/7809/7810/7811/7812/7813/7814/7815/7816/7817/7818/7819/7820/7821/7822/7823/7824/7825/7826/7827/7828/7829/7830/7831/7832/7833/7834/7835/7836/7837/7838/7839/7840/7841/7842/7843/7844/7845/7846/7847/7848/7849/7850/7851/7852/7853/7854/7855/7856/7857/7858/7859/7860/7861/7862/7863/7864/7865/7866/7867/7868/7869/7870/7871/7872/7873/7874/7875/7876/7877/7878/7879/7880/7881/7882/7883/7884/7885/7886/7887/7888/7889/7890/7891/7892/7893/7894/7895/7896/7897/7898/7899/7900/7901/7902/7903/7904/7905/7906/7907/7908/7909/7910/7911/7912/7913/7914/7915/7916/7917/7918/7919/7920/7921/7922/7923/7924/7925/7926/7927/7928/7929/7930/7931/7932/7933/7934/7935/7936/7937/7938/7939/7940/7941/7942/7943/7944/7945/7946/7947/7948/7949/7950/7951/7952/7953/7954/7955/7956/7957/7958/7959/7960/7961/7962/7963/7964/7965/7966/7967/7968/7969/7970/7971/7972/7973/7974/7975/7976/7977/7978/7979/7980/7981/7982/7983/7984/7985/7986/7987/7988/7989/7990/7991/7992/7993/7994/7995/7996/7997/7998/7999/8000/8001/8002/8003/8004/8005/8006/8007/8008/8009/8010/8011/8012/8013/8014/8015/8016/8017/8018/8019/8020/8021/8022/8023/8024/8025/8026/8027/8028/8029/8030/8031/8032/8033/8034/8035/8036/8037/8038/8039/8040/8041/8042/8043/8044/8045/8046/8047/8048/8049/8050/8051/8052/8053/8054/8055/8056/8057/8058/8059/8060/8061/8062/8063/8064/8065/8066/8067/8068/8069/8070/8071/8072/8073/8074/8075/8076/8077/8078/8079/8080/8081/8082/8083/8084/8085/8086/8087/8088/8089/8090/8091/8092/8093/8094/8095/8096/8097/8098/8099/8100/8101/8102/8103/8104/8105/8106/8107/8108/8109/8110/8111/8112/8113/8114/8115/8116/8117/8118/8119/8120/8121/8122/8123/8124/8125/8126/8127/8128/8129/8130/8131/8132/8133/8134/8135/8136/8137/8138/8139/8140/8141/8142/8143/8144/8145/8146/8147/8148/8149/8150/8151/8152/8153/8154/8155/8156/8157/8158/8159/8160/8161/8162/8163/8164/8165/8166/8167/8168/8169/8170/8171/8172/8173/8174/8175/8176/8177/8178/8179/8180/8181/8182/8183/8184/8185/8186/8187/8188/8189/8190/8191/8192/8193/8194/8195/8196/8197/8198/8199/8200/8201/8202/8203/8204/8205/8206/8207/8208/8209/8210/8211/8212/8213/8214/8215/8216/8217/8218/8219/8220/8221/8222/8223/8224/8225/8226/8227/8228/8229/8230/8231/8232/8233/8234/8235/8236/8237/8238/8239/8240/8241/8242/8243/8244/8245/8246/8247/8248/8249/8250/8251/8252/8253/8254/8255/8256/8257/8258/8259/8260/8261/8262/8263/8264/8265/8266/8267/8268/8269/8270/8271/8272/8273/8274/8275/8276/8277/8278/8279/8280/8281/8282/8283/8284/8285/8286/8287/8288/8289/8290/8291/8292/8293/8294/8295/8296/8297/8298/8299/8300/8301/8302/8303/8304/8305/8306/8307/8308/8309/8310/8311/8312/8313/8314/8315/8316/8317/8318/8319/8320/8321/8322/8323/8324/8325/8326/8327/8328/8329/8330/8331/8332/8333/8334/8335/8336/8337/8338/8339/8340/8341/8342/8343/8344/8345/8346/8347/8348/8349/8350/8351/8352/8353/8354/8355/8356/8357/8358/8359/8360/8361/8362/8363/8364/8365/8366/8367/8368/8369/8370/8371/8372/8373/8374/8375/8376/8377/8378/8379/8380/8381/8382/8383/8384/8385/8386/8387/8388/8389/8390/8391/8392/8393/8394/8395/8396/8397/8398/8399/8400/8401/8402/8403/8404/8405/8406/8407/8408/8409/8410/8411/8412/8413/8414/8415/8416/8417/841
```

```

## comparison
## hsa03010 150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3
## hsa03040 150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3
## hsa04141 150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3
## hsa05205 150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3
## hsa04512 150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3
## hsa04514 150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3
##
## hsa03010 AOA0C4DG17;P08865;AOA024R2P0;P15880;P18124;A8MUD9;P23396;A8K4C8;P26373;P36578;P
## hsa03040 P17844;J3KTA4;O75400;Q05C41;B4DGZ4;O75533;P08621;AOA024QZD5;Q9UFS1;AOA024RB53;P09651;AOA024
## hsa04141 AOA024R8S5;P07237;P0790
## hsa05205
## hsa04512
## hsa04514
##
## hsa03010 RPSA;RPS2;RPL7;RPS3;RPL13;RPL4;RPL3;RPL5;RPS9;RPL29;RPL14;RPS3A;RPS8;RPS16;RPS14;RPS18;RPL
## hsa03040 DDX5;PRPF40A;SF3B1;SNRNP70;HNRNPA1;HSPA8;RBM25;HNRNPA3;HNRNPK;TRA2B;HNRNPU;SRSF2;SRSF4;SRSF
## hsa04141 P4HB;HSP90AA1;HSP90AB1;HSPA8;PDIA4;HSP90B1;PDIA3;VCL
## hsa05205 DDX5;FLNB;ITGB1;ITGAV;CD44;SDC1;
## hsa04512 ITGB1;ITGAV;LA
## hsa04514 PVR;ITGB1;ITGAV;SDC1;L1CA

# Need to create a 'compareClusterResult' object with the slots described below to be able to plot
# Normally, we'd feed in Entrez gene lists but at this stage we only have UniProt IDs.
# Too much of a pain to re-convert IDs, hence this hack.

# Results
# geneClusters
# fun (function)

# @Cluster
cluster = all.kegg$comparison
cluster = gsub("150mJ.1:150mJ.2:150mJ.3","uv.150mJ",cluster)
cluster = gsub("275mJ.1:275mJ.2:275mJ.3","uv.275mJ",cluster)
cluster = gsub("400mJ.1:400mJ.2:400mJ.3","uv.400mJ",cluster)
cluster = gsub("uv.150mJ:uv.275mJ:uv.400mJ","All.9",cluster)
cluster

## [1] "All.9" "All.9" "All.9" "All.9" "All.9" "All.9"
## [7] "All.9" "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ"
## [13] "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ"
## [19] "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ" "uv.150mJ"
## [25] "uv.150mJ" "uv.275mJ" "uv.275mJ" "uv.275mJ" "uv.275mJ" "uv.275mJ"
## [31] "uv.275mJ" "uv.275mJ" "uv.275mJ" "uv.275mJ" "uv.275mJ" "uv.275mJ"
## [37] "uv.275mJ" "uv.400mJ" "uv.400mJ" "uv.400mJ" "uv.400mJ" "uv.400mJ"
## [43] "uv.400mJ" "uv.400mJ" "uv.400mJ" "uv.400mJ" "uv.400mJ"

all.kegg.compare = cbind(cluster,all.kegg[,1:10])
colnames(all.kegg.compare)[1] = "Cluster"

new.clusts = list(All.9=isect$`150mJ.1:150mJ.2:150mJ.3:275mJ.1:275mJ.2:275mJ.3:400mJ.1:400mJ.2:400mJ.3`

# Need to convert this to a cluster result object
clust.comp = new("compareClusterResult",compareClusterResult = all.kegg.compare,fun="enrichKEGG",geneCl
plot(clust.comp,type="dot", showCategory = 30,font.size=8)

```

