# Felix Boos - 2DTPP

SILAC - three replicates - Version 2

Frank Stein

2022-10-05

This analysis pipeline is partly based on an analysis workflow developed by Bernd Klaus.

#R setup

##Defining a working directory

We defined the working directory for the analysis. This directory should contain the metadata file and the tab delimited text files containing the data.

```
setwd("~/home/path to the right file location")
```

##Loading packages

```
library(vsn)
library(limma)
library(MSnbase)
library(gplots)
library(fdrtool)
library(biobroom)
library(tidyverse)
```

##Defining some functions and parameters

```
customPlot <- list(
  theme_bw(base_size = 12),
  scale_fill_manual(values = c("#e41a1c", "#377eb8", "#4daf4a", "#984ea3", "#ff7f00", "#ffff33", "#a656
  scale_colour_manual(values = c("#e41a1c", "#377eb8", "#4daf4a", "#984ea3", "#ff7f00", "#ffff33", "#a6
)
script.version = "V2"
dir_save = "data_analysis_results_V2"
```

#Loading data and annotating experimental conditions

```
conditions <- read.delim("metadata.csv", sep = ",")
# conditions$ID <- paste0("X", conditions$Temperature, "_", conditions$condition, "_", conditions$rep)
files <- file.path(unique(conditions$file))
files <- files[file.exists(files)]
data <- NULL
for (i in seq_along(files))
{
  print(files[i])
  x <- read_delim(file.path(files[i]), delim = "\t")
  names(x) <- make.names(names(x))
  #keep only proteins with at least two unique peptide matches
  x <- x %>%
```

```
    dplyr::filter(qupm >= 2)
  #remove keratines
  x <- x %>%
    dplyr::filter(!grepl("[K, k][R, r][T, t][0-9]", gene_name))
  #remove known contaminants
  x <- x %>%
    dplyr::filter(!grepl("#+", protein_id))
  x$file <- files[i]
  keepnames <- as.character(subset(conditions, file == files[i])$col.name)
  x <- x[, c("protein_id", "description", "gene_name", "qupm", "top3", "file", keepnames)]
  x <- x %>%
    group_by(protein_id, description, gene_name, qupm, top3, file) %>%
    gather(key = "col.name", value = "value", keepnames)
  data <- bind_rows(data, x)
  rm(x, keepnames)
}
```

```
## [1] "T1_190402_light_only_merged_results_20190403_0919_proteins.txt"
## [1] "T2_190402_light_only_merged_results_20190403_0919_proteins.txt"
## [1] "T3_190502_light_only_merged_results_20190503_0911_proteins.txt"
## [1] "T4_190402_light_only_merged_results_20190402_1343_proteins.txt"
## [1] "T5_190402_light_only_merged_results_20190403_0334_proteins.txt"
## [1] "T6_190402_light_only_merged_results_20190403_0439_proteins.txt"
## [1] "T7_190402_light_only_merged_results_20190403_0730_proteins.txt"
## [1] "T8_190402_light_only_merged_results_20190403_0531_proteins.txt"
## [1] "T9_190405_light_only_merged_results_20190405_2250_proteins.txt"
## [1] "T10_190405_light_only_merged_results_20190405_2326_proteins.txt"
## [1] "T12_190408_light_only_merged_results_20190424_1110_proteins.txt"
## [1] "T1_190403_heavy_only_merged_results_20190403_1440_proteins.txt"
## [1] "T2_190403_heavy_only_merged_results_20190403_1440_proteins.txt"
## [1] "T3_190502_heavy_only_merged_results_20190503_0910_proteins.txt"
## [1] "T4_190403_heavy_only_merged_results_20190403_1623_proteins.txt"
## [1] "T5_190403_heavy_only_merged_results_20190403_1618_proteins.txt"
## [1] "T6_190403_heavy_only_merged_results_20190403_1626_proteins.txt"
## [1] "T7_190403_heavy_only_merged_results_20190403_1628_proteins.txt"
## [1] "T8_190403_heavy_only_merged_results_20190403_1616_proteins.txt"
## [1] "T9_190405_heavy_only_merged_results_20190405_2038_proteins.txt"
## [1] "T10_190405_heavy_only_merged_results_20190405_2122_proteins.txt"
## [1] "T12_190408_heavy_only_merged_results_20190424_1112_proteins.txt"
## [1] "P1111_PH_FB_TPP_n2_T1_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T2_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T3_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T4_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T5_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T6_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T7_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T8_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T9_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T10_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T12_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n2_T1_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T2_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T3_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T4_merged_results_heavy.txt"
```

```
## [1] "P1111_PH_FB_TPP_n2_T5_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T6_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T7_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T8_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T9_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T10_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n2_T12_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T1_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T2_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T3_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T4_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T5_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T6_merged_results_light_2ndSP2.txt"
## [1] "P1111_PH_FB_TPP_n3_T7_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T8_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T9_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T10_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T12_merged_results_light.txt"
## [1] "P1111_PH_FB_TPP_n3_T1_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T2_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T3_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T4_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T5_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T6_merged_results_heavy_2ndSP2.txt"
## [1] "P1111_PH_FB_TPP_n3_T7_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T8_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T9_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T10_merged_results_heavy.txt"
## [1] "P1111_PH_FB_TPP_n3_T12_merged_results_heavy.txt"
```

```
rm(i, files)
data <- left_join(data, conditions)
# write.csv(data, file.path(dir_save, paste0("unfiltered_raw_data_", script.version, ".csv")))
data <- subset(data, value > 0)
```

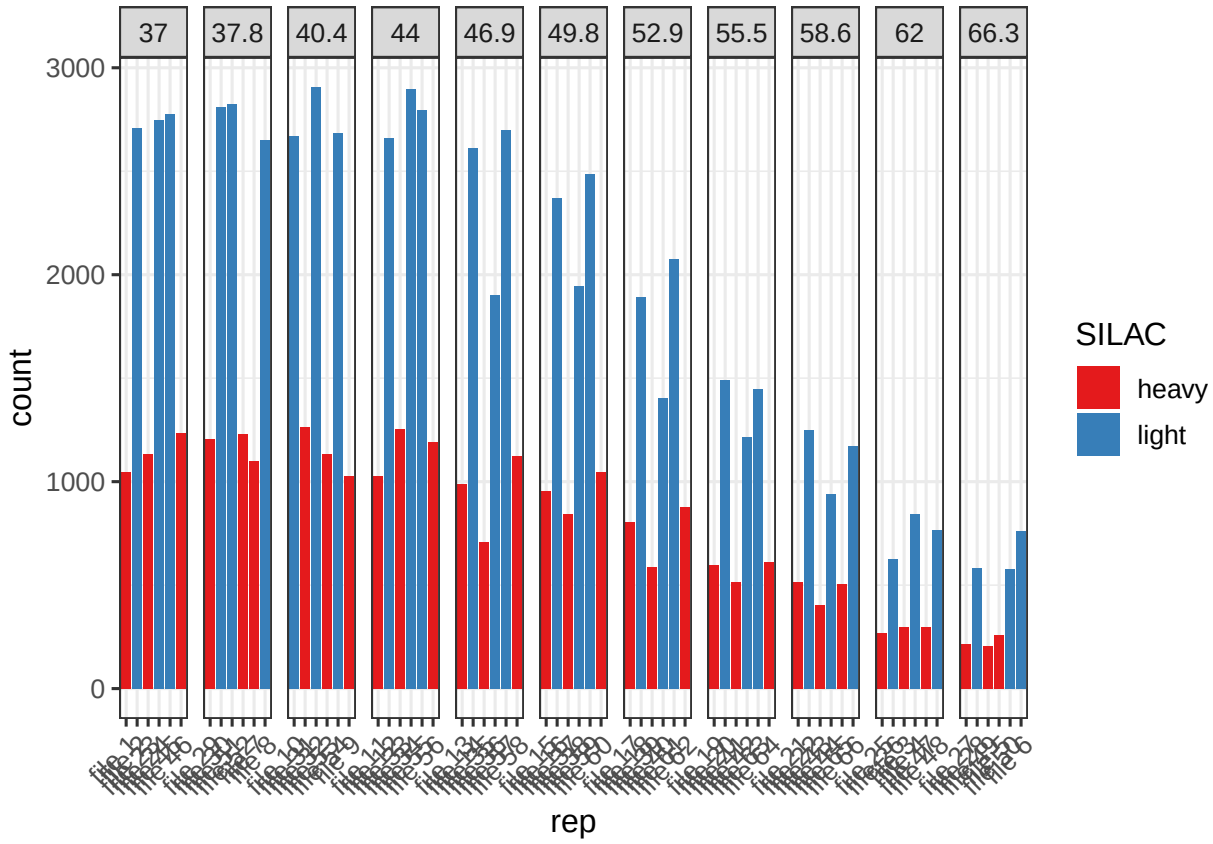
#Protein identification overview

```
sub <- unique(data[, c("gene_name", "file")])
sub <- merge(sub, unique(conditions[, c("file", "rep", "Temperature", "SILAC")]))
sub$found <- 1
sub$rep <- paste("file", as.numeric(factor(sub$file)))
print(unique(with(sub, paste(rep, "-", file))))
```

```
##  [1] "file 1 - P1111_PH_FB_TPP_n2_T1_merged_results_heavy.txt"
##  [2] "file 2 - P1111_PH_FB_TPP_n2_T1_merged_results_light.txt"
##  [3] "file 3 - P1111_PH_FB_TPP_n2_T10_merged_results_heavy.txt"
##  [4] "file 4 - P1111_PH_FB_TPP_n2_T10_merged_results_light.txt"
##  [5] "file 5 - P1111_PH_FB_TPP_n2_T12_merged_results_heavy.txt"
##  [6] "file 6 - P1111_PH_FB_TPP_n2_T12_merged_results_light.txt"
##  [7] "file 7 - P1111_PH_FB_TPP_n2_T2_merged_results_heavy.txt"
##  [8] "file 8 - P1111_PH_FB_TPP_n2_T2_merged_results_light.txt"
##  [9] "file 9 - P1111_PH_FB_TPP_n2_T3_merged_results_heavy.txt"
## [10] "file 10 - P1111_PH_FB_TPP_n2_T3_merged_results_light.txt"
## [11] "file 11 - P1111_PH_FB_TPP_n2_T4_merged_results_heavy.txt"
## [12] "file 12 - P1111_PH_FB_TPP_n2_T4_merged_results_light.txt"
```

```
## [13] "file 13 - P1111_PH_FB_TPP_n2_T5_merged_results_heavy.txt"
## [14] "file 14 - P1111_PH_FB_TPP_n2_T5_merged_results_light.txt"
## [15] "file 15 - P1111_PH_FB_TPP_n2_T6_merged_results_heavy.txt"
## [16] "file 16 - P1111_PH_FB_TPP_n2_T6_merged_results_light.txt"
## [17] "file 17 - P1111_PH_FB_TPP_n2_T7_merged_results_heavy.txt"
## [18] "file 18 - P1111_PH_FB_TPP_n2_T7_merged_results_light.txt"
## [19] "file 19 - P1111_PH_FB_TPP_n2_T8_merged_results_heavy.txt"
## [20] "file 20 - P1111_PH_FB_TPP_n2_T8_merged_results_light.txt"
## [21] "file 21 - P1111_PH_FB_TPP_n2_T9_merged_results_heavy.txt"
## [22] "file 22 - P1111_PH_FB_TPP_n2_T9_merged_results_light.txt"
## [23] "file 23 - P1111_PH_FB_TPP_n3_T1_merged_results_heavy.txt"
## [24] "file 24 - P1111_PH_FB_TPP_n3_T1_merged_results_light.txt"
## [25] "file 25 - P1111_PH_FB_TPP_n3_T10_merged_results_heavy.txt"
## [26] "file 26 - P1111_PH_FB_TPP_n3_T10_merged_results_light.txt"
## [27] "file 27 - P1111_PH_FB_TPP_n3_T12_merged_results_heavy.txt"
## [28] "file 28 - P1111_PH_FB_TPP_n3_T12_merged_results_light.txt"
## [29] "file 29 - P1111_PH_FB_TPP_n3_T2_merged_results_heavy.txt"
## [30] "file 30 - P1111_PH_FB_TPP_n3_T2_merged_results_light.txt"
## [31] "file 31 - P1111_PH_FB_TPP_n3_T3_merged_results_heavy.txt"
## [32] "file 32 - P1111_PH_FB_TPP_n3_T3_merged_results_light.txt"
## [33] "file 33 - P1111_PH_FB_TPP_n3_T4_merged_results_heavy.txt"
## [34] "file 34 - P1111_PH_FB_TPP_n3_T4_merged_results_light.txt"
## [35] "file 35 - P1111_PH_FB_TPP_n3_T5_merged_results_heavy.txt"
## [36] "file 36 - P1111_PH_FB_TPP_n3_T5_merged_results_light.txt"
## [37] "file 37 - P1111_PH_FB_TPP_n3_T6_merged_results_heavy_2ndSP2.txt"
## [38] "file 38 - P1111_PH_FB_TPP_n3_T6_merged_results_light_2ndSP2.txt"
## [39] "file 39 - P1111_PH_FB_TPP_n3_T7_merged_results_heavy.txt"
## [40] "file 40 - P1111_PH_FB_TPP_n3_T7_merged_results_light.txt"
## [41] "file 41 - P1111_PH_FB_TPP_n3_T8_merged_results_heavy.txt"
## [42] "file 42 - P1111_PH_FB_TPP_n3_T8_merged_results_light.txt"
## [43] "file 43 - P1111_PH_FB_TPP_n3_T9_merged_results_heavy.txt"
## [44] "file 44 - P1111_PH_FB_TPP_n3_T9_merged_results_light.txt"
## [45] "file 45 - T1_190402_light_only_merged_results_20190403_0919_proteins.txt"
## [46] "file 46 - T1_190403_heavy_only_merged_results_20190403_1440_proteins.txt"
## [47] "file 47 - T10_190405_heavy_only_merged_results_20190405_2122_proteins.txt"
## [48] "file 48 - T10_190405_light_only_merged_results_20190405_2326_proteins.txt"
## [49] "file 49 - T12_190408_heavy_only_merged_results_20190424_1112_proteins.txt"
## [50] "file 50 - T12_190408_light_only_merged_results_20190424_1110_proteins.txt"
## [51] "file 51 - T2_190402_light_only_merged_results_20190403_0919_proteins.txt"
## [52] "file 52 - T2_190403_heavy_only_merged_results_20190403_1440_proteins.txt"
## [53] "file 53 - T3_190502_heavy_only_merged_results_20190503_0910_proteins.txt"
## [54] "file 54 - T3_190502_light_only_merged_results_20190503_0911_proteins.txt"
## [55] "file 55 - T4_190402_light_only_merged_results_20190402_1343_proteins.txt"
## [56] "file 56 - T4_190403_heavy_only_merged_results_20190403_1623_proteins.txt"
## [57] "file 57 - T5_190402_light_only_merged_results_20190403_0334_proteins.txt"
## [58] "file 58 - T5_190403_heavy_only_merged_results_20190403_1618_proteins.txt"
## [59] "file 59 - T6_190402_light_only_merged_results_20190403_0439_proteins.txt"
## [60] "file 60 - T6_190403_heavy_only_merged_results_20190403_1626_proteins.txt"
## [61] "file 61 - T7_190402_light_only_merged_results_20190403_0730_proteins.txt"
## [62] "file 62 - T7_190403_heavy_only_merged_results_20190403_1628_proteins.txt"
## [63] "file 63 - T8_190402_light_only_merged_results_20190403_0531_proteins.txt"
## [64] "file 64 - T8_190403_heavy_only_merged_results_20190403_1616_proteins.txt"
## [65] "file 65 - T9_190405_heavy_only_merged_results_20190405_2038_proteins.txt"
## [66] "file 66 - T9_190405_light_only_merged_results_20190405_2250_proteins.txt"
```

```
sub <- unique(sub)
ggplot(data = sub, aes(rep, fill = SILAC)) +
  geom_bar() +
  customPlot +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1)) +
  facet_grid(. ~ Temperature, space = "free", scale = "free_x")
```




```
sub$rep <- paste(sub$Temperature, sub$rep)
with(sub, table(rep))
```

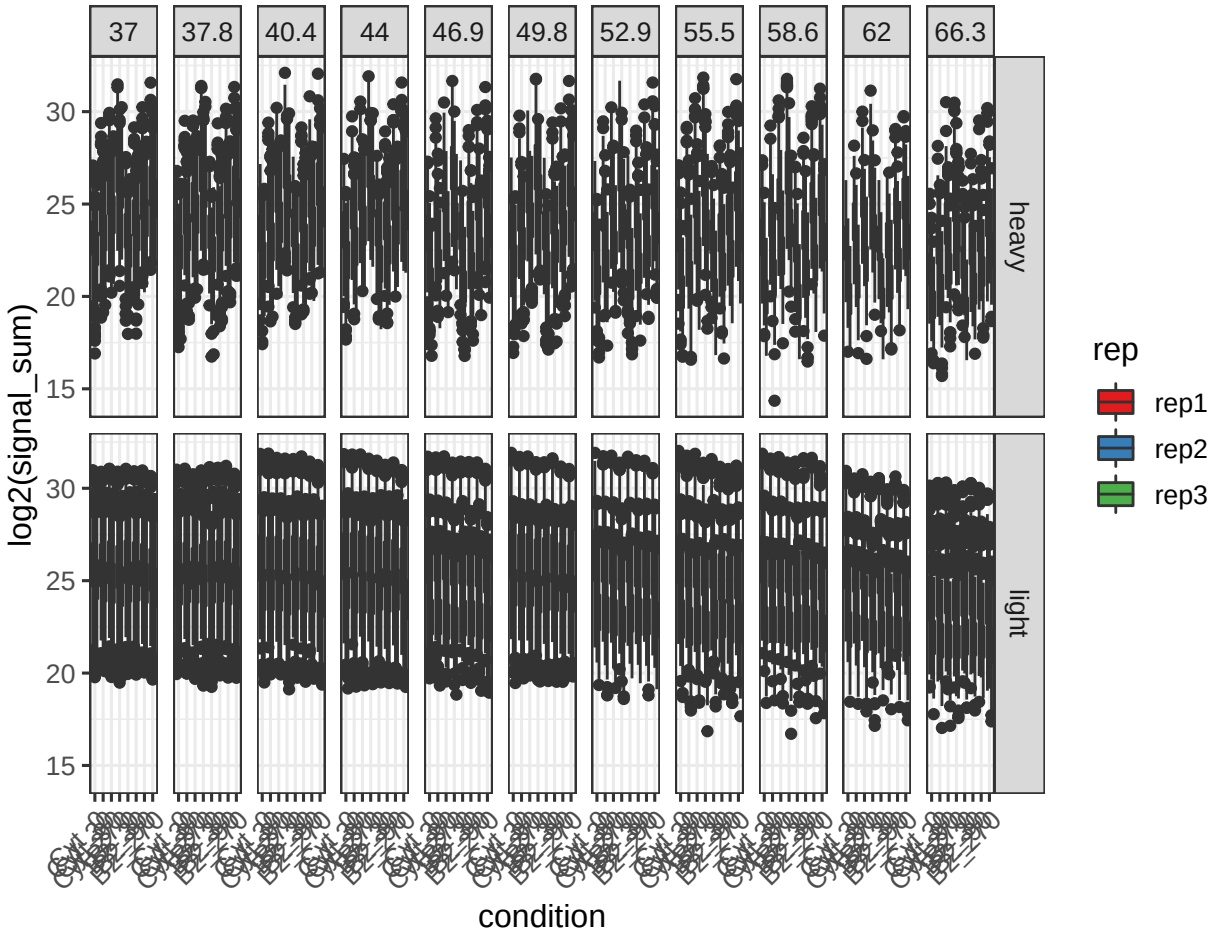
```
## rep
##    37 file 1    37 file 2   37 file 23   37 file 24   37 file 45   37 file 46
##         1045         2708         1132         2746         2777         1237
## 37.8 file 29 37.8 file 30 37.8 file 51 37.8 file 52  37.8 file 7  37.8 file 8
##         1208         2811         2826         1230         1101         2650
## 40.4 file 10 40.4 file 31 40.4 file 32 40.4 file 53 40.4 file 54  40.4 file 9
##         2670         1264         2905         1133         2684         1027
##   44 file 11   44 file 12   44 file 33   44 file 34   44 file 55   44 file 56
##         1027         2658         1253         2897         2797         1190
## 46.9 file 13 46.9 file 14 46.9 file 35 46.9 file 36 46.9 file 57 46.9 file 58
##          990         2612          708         1902         2698         1125
## 49.8 file 15 49.8 file 16 49.8 file 37 49.8 file 38 49.8 file 59 49.8 file 60
##          956         2370          845         1947         2486         1047
## 52.9 file 17 52.9 file 18 52.9 file 39 52.9 file 40 52.9 file 61 52.9 file 62
##          803         1893          588         1406         2074          879
## 55.5 file 19 55.5 file 20 55.5 file 41 55.5 file 42 55.5 file 63 55.5 file 64
```

```
##          598         1489          517         1214         1446          609
## 58.6 file 21 58.6 file 22 58.6 file 43 58.6 file 44 58.6 file 65 58.6 file 66
##          517         1250          403          942          504         1170
##   62 file 25   62 file 26    62 file 3    62 file 4   62 file 47   62 file 48
##          267          627          299          842          295          767
## 66.3 file 27 66.3 file 28 66.3 file 49  66.3 file 5 66.3 file 50  66.3 file 6
##          217          583          206          261          579          763
```

```
rm(sub.sub)
rm(sub)
```

#Transforming long data to wide data

```
data$condition <- factor(data$condition, ordered = TRUE, levels =
                           c("Cyt_0", "Cyt_30", "Cyt_90", "Cyt_270",
                             "B2_0", "B2_30", "B2_90","B2_270"))
ggplot(data = data, aes(condition, log2(value), fill = rep)) +
  geom_boxplot() +
  ylab("log2(signal_sum)") +
  facet_grid(SILAC ~ Temperature, scale = "free_x", space = "free") +
  customPlot +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1))
```




```
cdata <- data %>%
  group_by(gene_name) %>%
```

```
  mutate(key = paste("signal_sum", SILAC, Temperature, condition, rep, sep = "_")) %>%
  dplyr::select(gene_name, key, value) %>%
  group_by(gene_name, key) %>%
  summarise(value = sum(value, na.rm = TRUE)) %>%
  spread(key = key, value = value)
#found.in.ms.runs
fdata <- data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, file) %>%
  unique() %>%
  summarise(found.in.ms.runs = n())
fdata_i <- data %>%
  group_by(gene_name, sample) %>%
  dplyr::select(gene_name, file) %>%
  unique() %>%
  summarise(found.in.ms.runs = n()) %>%
  mutate(sample = paste0("found.in.ms.runs_", sample)) %>%
  spread(key = sample, value = found.in.ms.runs)
fdata <- left_join(fdata, fdata_i)
fdata_i <- data %>%
  group_by(gene_name, SILAC) %>%
  dplyr::select(gene_name, file) %>%
  unique() %>%
  summarise(found.in.ms.runs = n()) %>%
  mutate(SILAC = paste0("found.in.ms.runs_", SILAC)) %>%
  spread(key = SILAC, value = found.in.ms.runs)
fdata <- left_join(fdata, fdata_i)
fdata_i <- data %>%
  group_by(gene_name, Temperature) %>%
  dplyr::select(gene_name, file) %>%
  unique() %>%
  summarise(found.in.ms.runs = n()) %>%
  mutate(Temperature = paste0("found.in.ms.runs_", Temperature)) %>%
  spread(key = Temperature, value = found.in.ms.runs)
fdata <- left_join(fdata, fdata_i)
#found.in.reps
fdata_i <- data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, rep) %>%
  unique() %>%
  summarise(found.in.reps = n())
fdata <- left_join(fdata, fdata_i)
fdata_i <- data %>%
  group_by(gene_name, sample) %>%
  dplyr::select(gene_name, rep) %>%
  unique() %>%
  summarise(found.in.reps = n()) %>%
  mutate(sample = paste0("found.in.reps_", sample)) %>%
  spread(key = sample, value = found.in.reps)
fdata <- left_join(fdata, fdata_i)
#qupm overview
fdata_i <- data %>%
  group_by(gene_name) %>%
```

```
  dplyr::select(gene_name, qupm) %>%
  unique() %>%
  summarise(max.qupm = max(qupm, na.rm = TRUE))
fdata <- left_join(fdata, fdata_i)
fdata_i <- data %>%
  group_by(gene_name, sample) %>%
  dplyr::select(gene_name, sample, qupm) %>%
  unique() %>%
  summarise(max.qupm = max(qupm, na.rm = TRUE)) %>%
  mutate(sample = paste0("max.qupm_", sample)) %>%
  spread(key = sample, value = max.qupm)
fdata <- left_join(fdata, fdata_i)
#abundance overview
fdata_i <- data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, top3) %>%
  unique() %>%
  summarise(average.top3 = mean(top3, na.rm = TRUE))
fdata <- left_join(fdata, fdata_i)
fdata_i <- data %>%
  group_by(gene_name, sample) %>%
  dplyr::select(gene_name, sample, top3) %>%
  unique() %>%
  summarise(average.top3 = mean(top3, na.rm = TRUE)) %>%
  mutate(sample = paste0("average.top3_", sample)) %>%
  spread(key = sample, value = average.top3)
id_data <- data %>%
  ungroup() %>%
  dplyr::select(gene_name, protein_id, description) %>%
  unique()
dups <- id_data %>%
  group_by(gene_name) %>%
  summarise(n = n()) %>%
  dplyr::filter(n > 1) %>%
  dplyr::select(gene_name)
if (nrow(dups) > 0)
{
  dups <- dups$gene_name
  dup.data <- subset(id_data, gene_name %in% dups)
  id_data <- subset(id_data, !gene_name %in% dups)
  combine_ids <- function(ids)
  {
    ids <- as.character(ids)
    return(paste(
      sort(unique(unlist(strsplit(ids, split = "[|]")))), collapse = "|"))
  }
  dup.data <- dup.data %>%
    group_by(gene_name) %>%
    summarise(protein_id = combine_ids(protein_id),
              description = combine_ids(description))
  id_data <- bind_rows(id_data, dup.data)
  rm(dup.data, combine_ids)
}
```

```
rm(dups)
fdata <- left_join(id_data, fdata)
rm(id_data)
fdata <- left_join(fdata, fdata_i)
cdata <- left_join(fdata, cdata)
rm(fdata_i, fdata)
```

#Filter data

Only proteins that were quantified with two unique peptide matches are kept for the analysis. Proteins were filtered according to these condition already when loading the data. Moreover, only proteins were kept if they were quantified in at least 2/3 of the replicates.

```
dim(cdata)
```

```
## [1] 3775  636
```

```
min.found.in.ms.runs <- 2
min.qupm <- 2
for (s in unique(conditions$sample))
{
  cname <- paste0("keep_", s)
  cdata[, cname] <- ifelse(
    cdata[, paste0("found.in.reps_", s)] >=
      min.found.in.ms.runs & cdata[, paste0("max.qupm_", s)] >= min.qupm, TRUE, FALSE)
  cdata[is.na(cdata[, cname]), cname] <- FALSE
  rm(cname)
}
rm(s)
cdata$keep <- FALSE
for (i in seq_along(rownames(cdata)))
{
  cdata$keep[i] <- any(as.logical(cdata[i, grep("keep_", names(cdata), value = TRUE)]))
}
rm(i)
table(cdata$keep)
```

```
##
## FALSE  TRUE
##   545  3230
```

```
cdata <- subset(cdata, keep)
dim(cdata)
```

```
## [1] 3230  659
```

```
rm(min.found.in.ms.runs, min.qupm)
cdata <- subset(cdata, (found.in.ms.runs_heavy >= 8 | found.in.ms.runs_light >= 8) &
                  (found.in.ms.runs_37 >= 2 | found.in.ms.runs_37.8 >= 2))
dim(cdata)
```

```
## [1] 2804  659
```

#Building an expression set object

```
#Constructing assay data
raw_data <- cdata %>%
  dplyr::select(starts_with("signal_sum")) %>%
  as.data.frame()
```

```
rownames(raw_data) <- cdata$gene_name
names(raw_data) <- gsub("signal_sum_", "", names(raw_data))

#Constructing metadata
conditions_i <- data.frame(ID = names(raw_data))
# conditions_i$ID <- paste0("X", conditions_i$ID)
conditions_i <- left_join(conditions_i, conditions)

#Constructing fdata
fdata <- cdata %>%
  dplyr::select(-starts_with("signal_sum")) %>%
  as.data.frame()

#Defining ID columns
rownames(fdata) <- fdata$gene_name
rownames(conditions_i) <- conditions_i$ID
colnames(raw_data) <- conditions_i$ID

#Log2-transformation of raw_ signal_sums
raw_data_m <- log2(as.matrix(raw_data))
raw_data_m[is.infinite((raw_data_m))] <- NA
raw_data_m[is.na((raw_data_m))] <- NA

#Creating an expression set
raw_dataE <- ExpressionSet(assayData = raw_data_m,
                           phenoData = AnnotatedDataFrame(conditions_i),
                           featureData = AnnotatedDataFrame(fdata))
validObject(raw_dataE)
```

```
## [1] TRUE
```

```
rm(raw_data, raw_data_m, conditions_i, fdata)
```

#Batch effect removal

```
batchcl_raw_dataE <- raw_dataE
for (s in unique(conditions$sample))
{
  print(s)
  batchcl_raw_dataE_i <-
    batchcl_raw_dataE[fData(batchcl_raw_dataE)[, paste0("keep_", s)],
                      batchcl_raw_dataE$sample == s]

  exprs(batchcl_raw_dataE_i) <-
    removeBatchEffect(exprs(batchcl_raw_dataE_i),
                      batch = as.character(pData(batchcl_raw_dataE_i)$rep),
                      design = model.matrix(~0 + as.character(pData(batchcl_raw_dataE_i)$condition)))

  for (cn in colnames(batchcl_raw_dataE_i))
  {
    exprs(batchcl_raw_dataE)[fData(batchcl_raw_dataE)[, paste0("keep_", s)], cn] <-
      exprs(batchcl_raw_dataE_i)[, cn]
  }
  rm(cn)
}
```

```
## [1] "light_37"
## [1] "light_37.8"
## [1] "light_40.4"
## [1] "light_44"
## [1] "light_46.9"
## [1] "light_49.8"
## [1] "light_52.9"
## [1] "light_55.5"
## [1] "light_58.6"
## [1] "light_62"
## [1] "light_66.3"
## [1] "heavy_37"
## [1] "heavy_37.8"
## [1] "heavy_40.4"
## [1] "heavy_44"
## [1] "heavy_46.9"
## [1] "heavy_49.8"
## [1] "heavy_52.9"
## [1] "heavy_55.5"
## [1] "heavy_58.6"
## [1] "heavy_62"
## [1] "heavy_66.3"
```

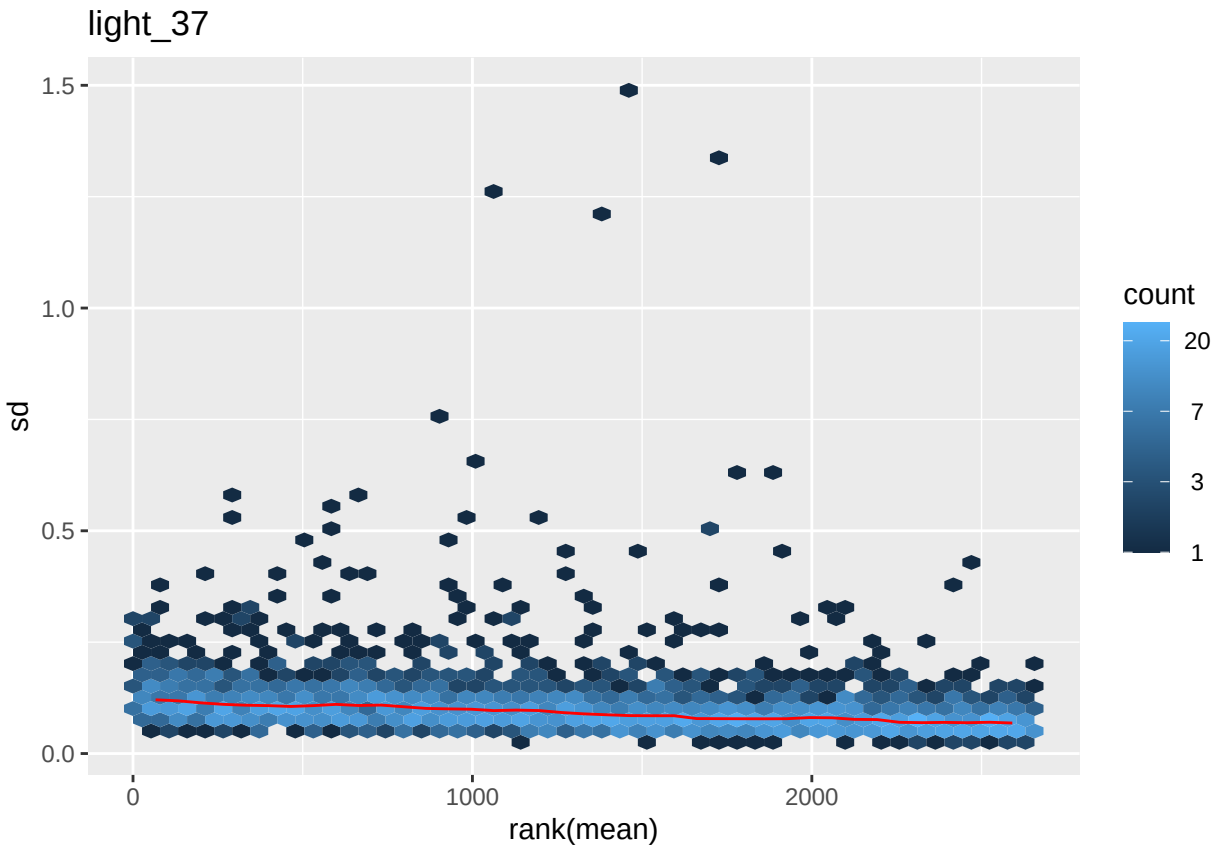
```
rm(s)
```

#Data normalization

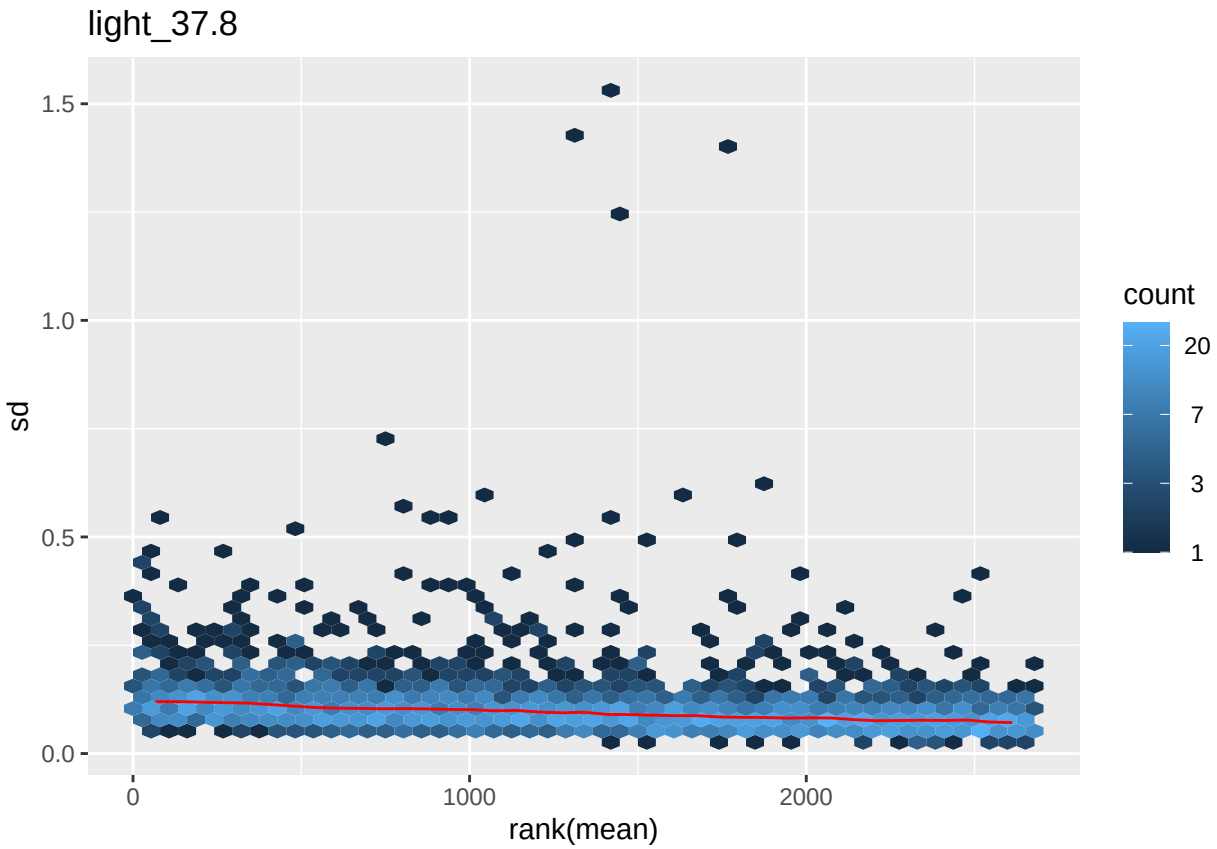
The vsn package from Wolfgang Huber is used to apply a variance stabilization normalization method on the log2 raw data.

```
norm_batchcl_raw_dataE <- batchcl_raw_dataE
for (s in unique(conditions$sample))
{
  print(paste0(s))
  vsn.fit <- vsn2(2^exprs(norm_batchcl_raw_dataE[
    fData(norm_batchcl_raw_dataE)[, paste0("keep_", s)],
    norm_batchcl_raw_dataE$sample == s]))
  norm_batchcl_raw_dataE_i <- predict(vsn.fit, 2^exprs(norm_batchcl_raw_dataE[
    fData(norm_batchcl_raw_dataE)[, paste0("keep_", s)],
    norm_batchcl_raw_dataE$sample == s]))
  for (cn in colnames(norm_batchcl_raw_dataE_i))
  {
    exprs(norm_batchcl_raw_dataE)[fData(norm_batchcl_raw_dataE)[, paste0("keep_", s)], cn] <-
      norm_batchcl_raw_dataE_i[, cn]
  }
  rm(cn, norm_batchcl_raw_dataE_i)
  sdplot.object <- meanSdPlot(vsn.fit, plot = FALSE)
  print(sdplot.object$gg + ggtitle(s))
  rm(vsn.fit, sdplot.object)
}
```

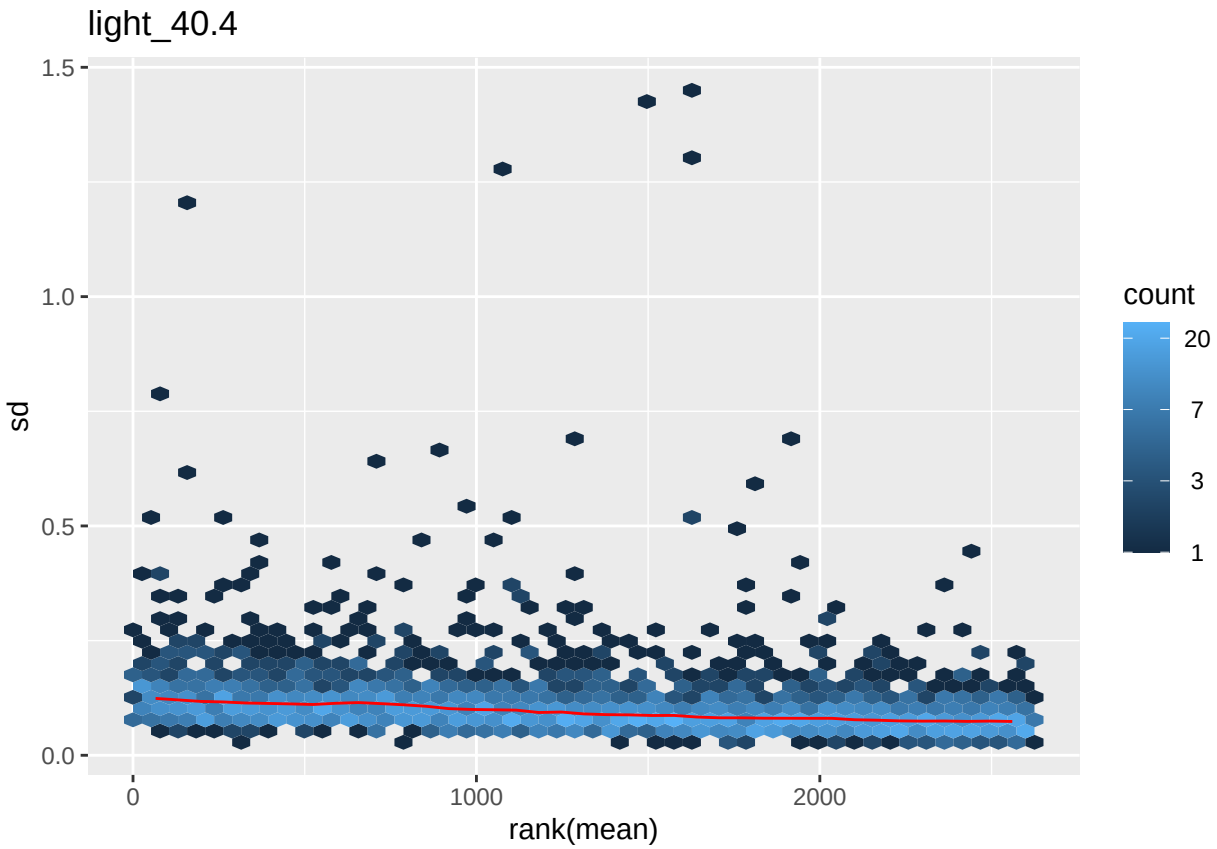
```
## [1] "light_37"
```



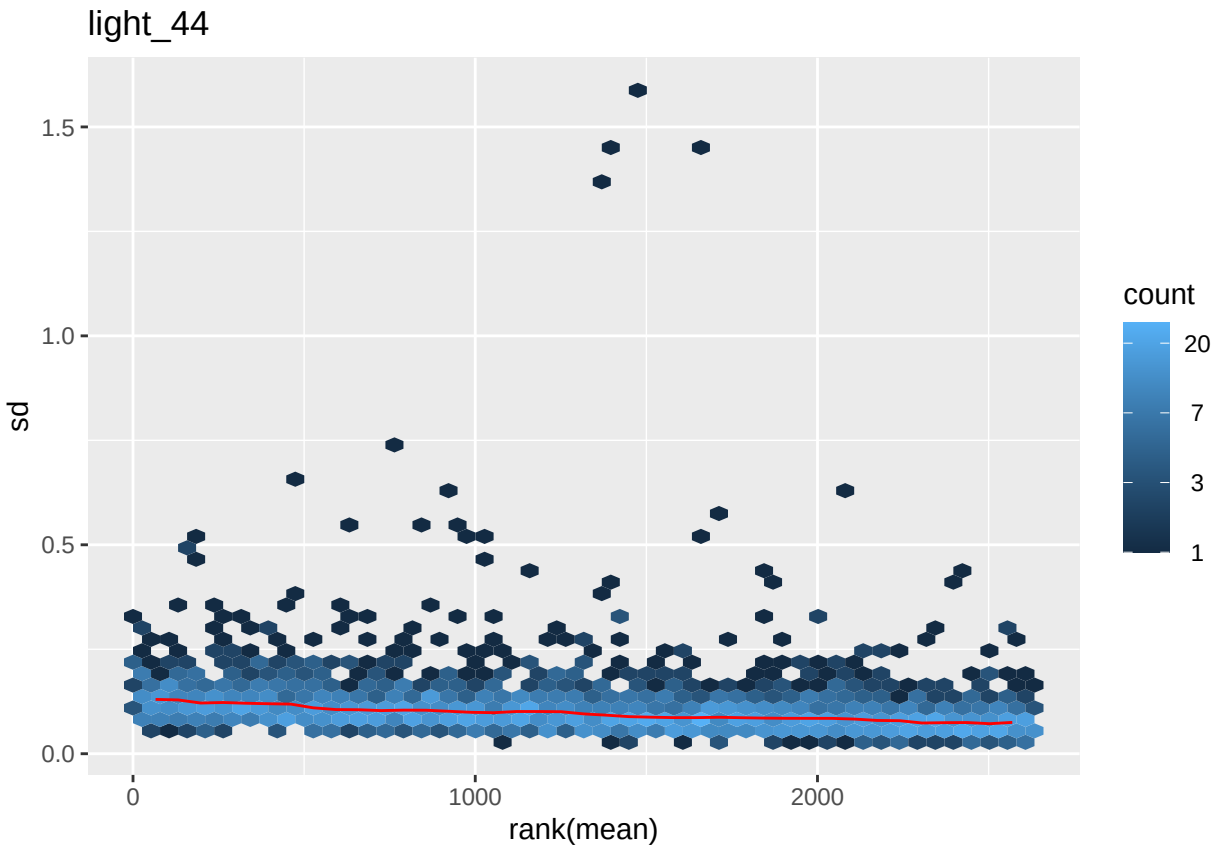

```
## [1] "light_37.8"
```



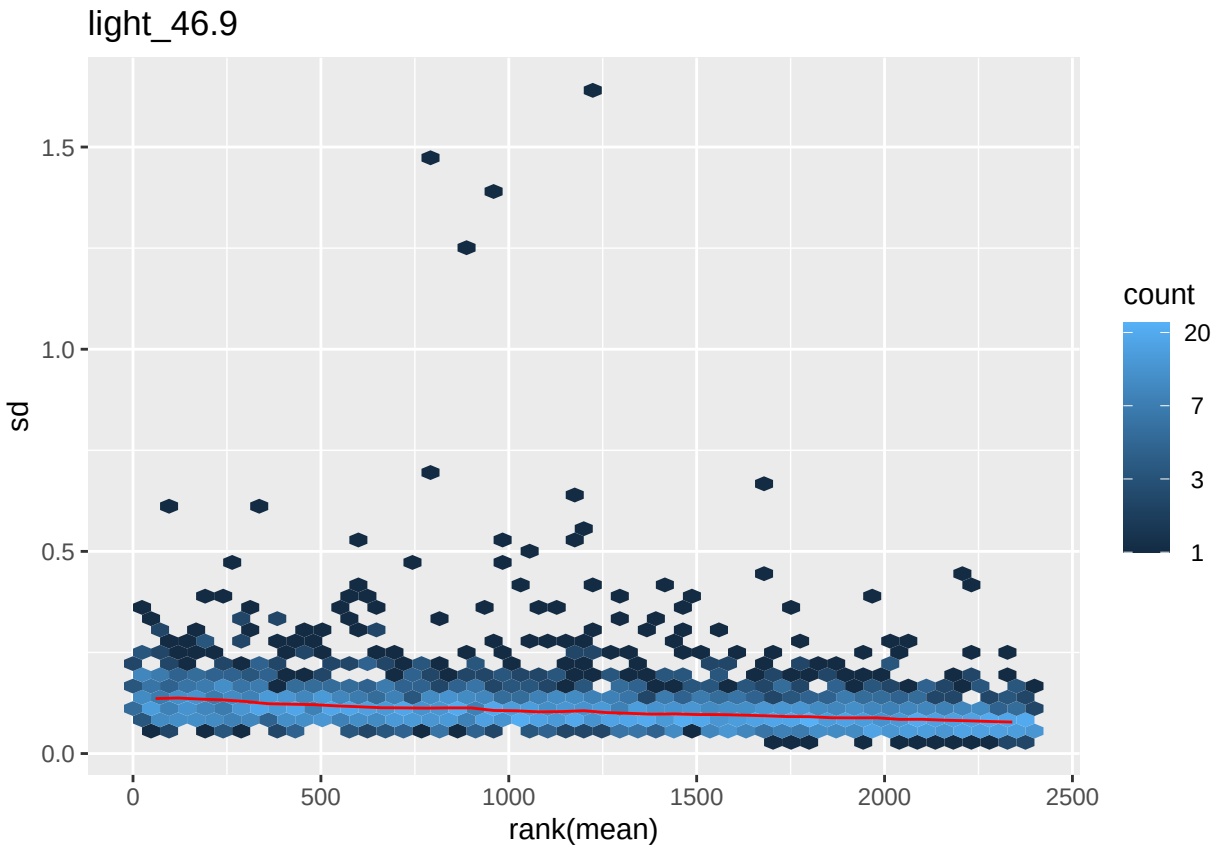

```
## [1] "light_40.4"
```



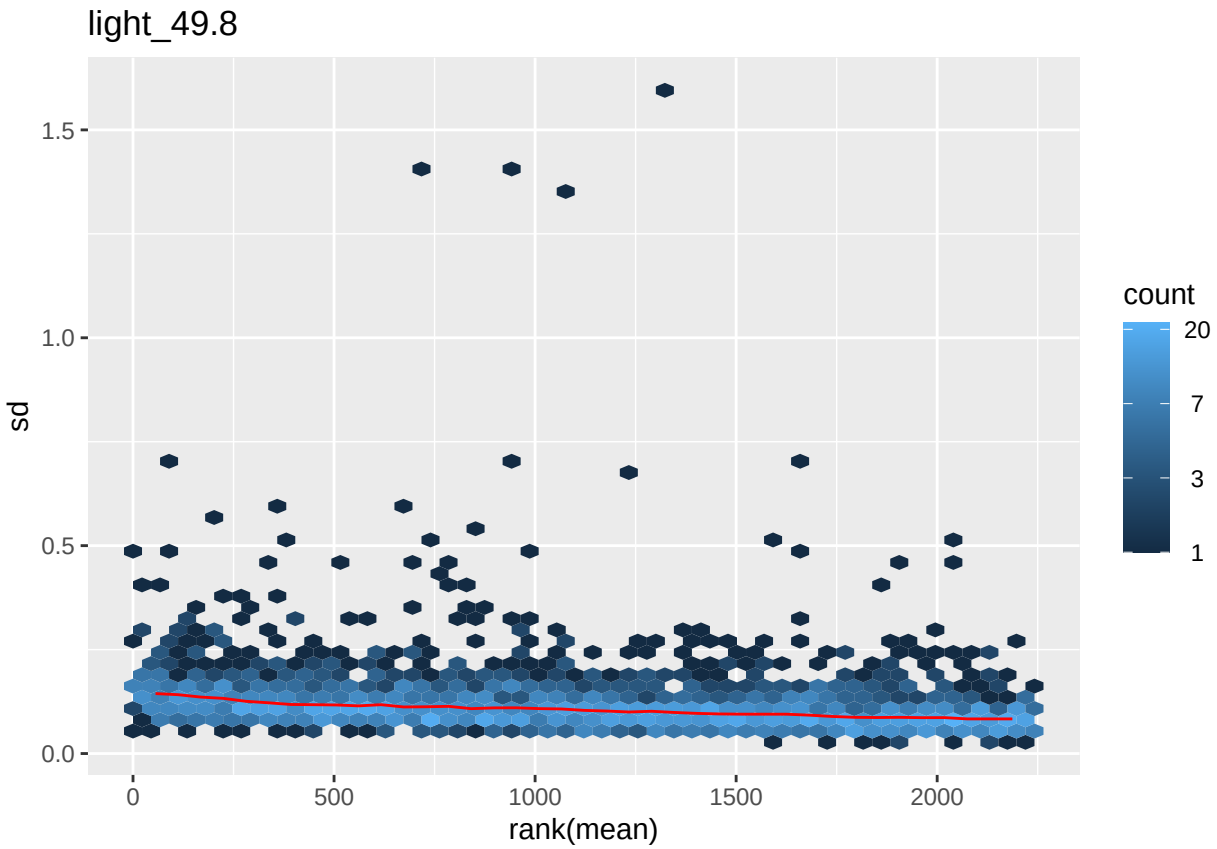

```
## [1] "light_44"
```



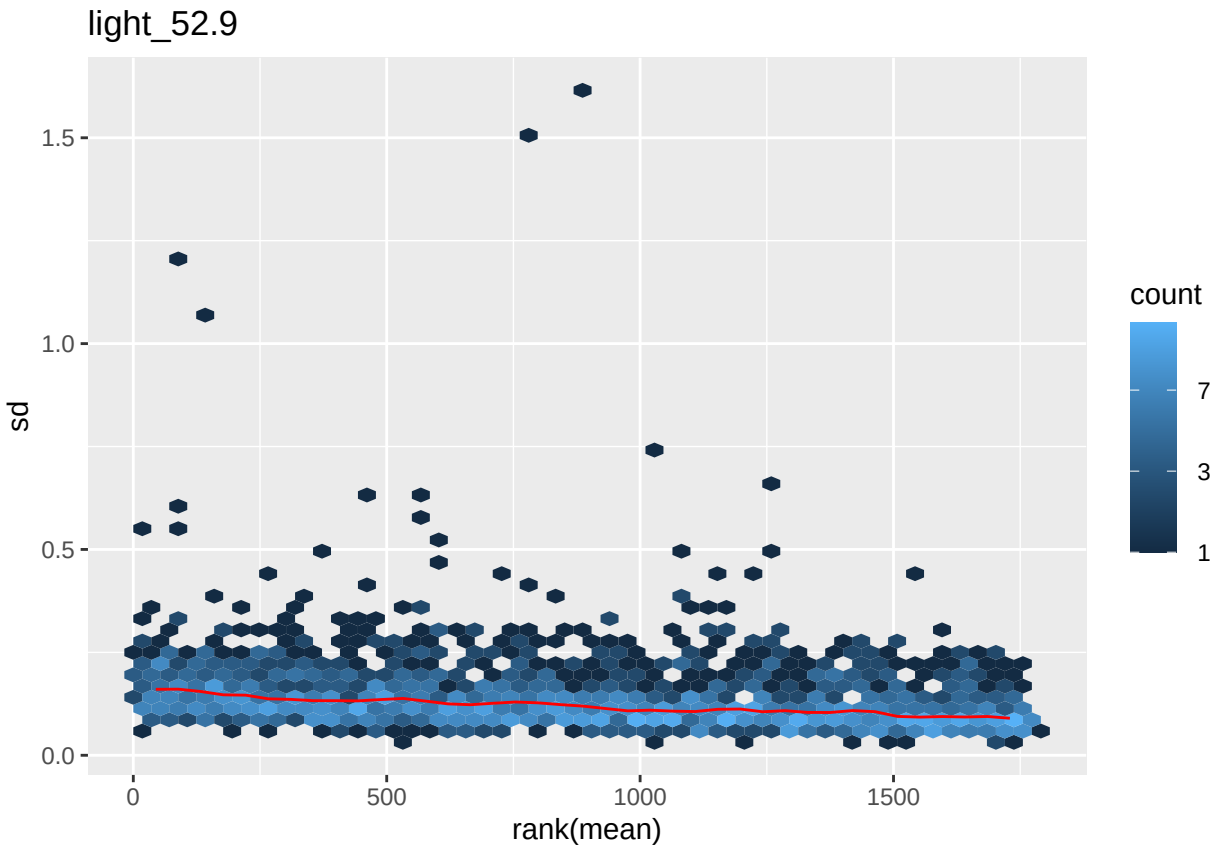

```
## [1] "light_46.9"
```


light_46.9
sd
1.5
1.0
0.5
0.0
0
500
1000
1500
2000
2500
rank(mean)
count
20
7
3
1

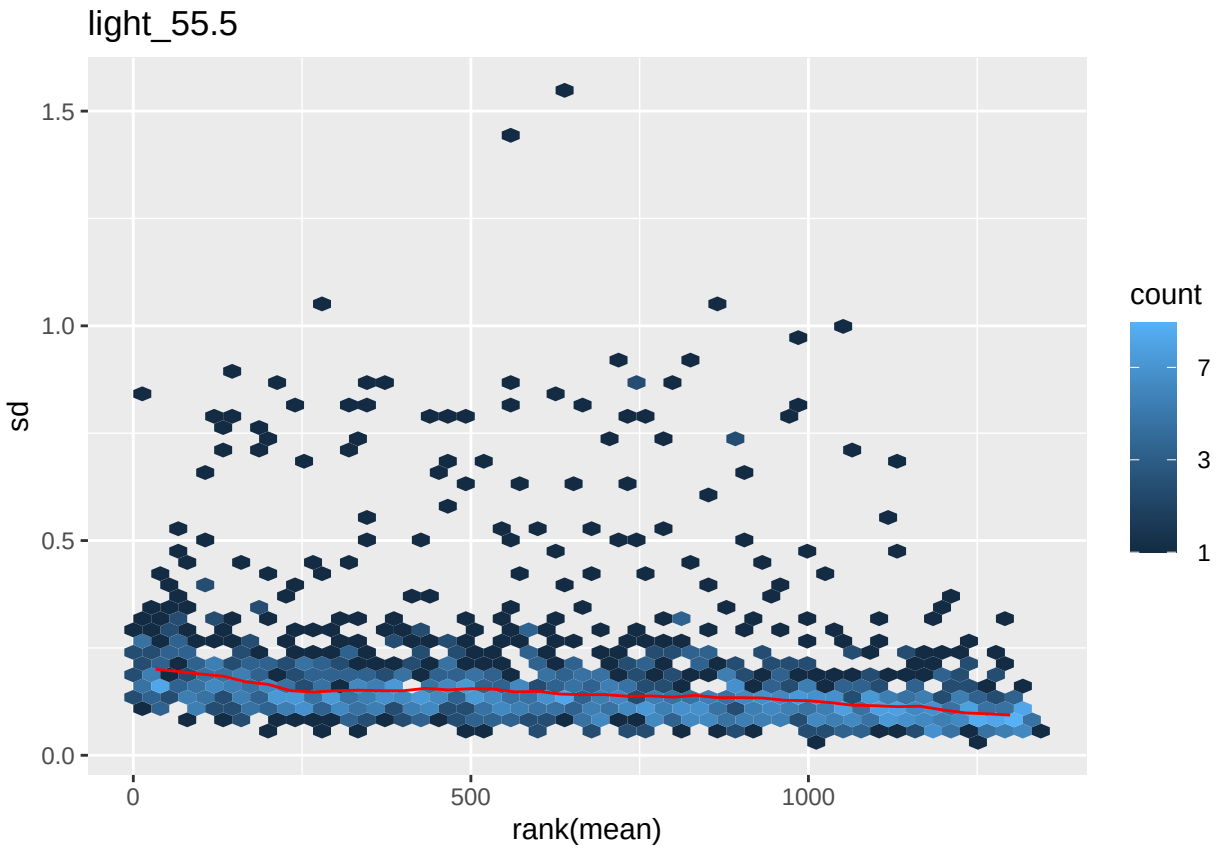
```
## [1] "light_49.8"
```


light_49.8
sd
1.5
1.0
0.5
0.0
0
500
1000
1500
2000
rank(mean)
count
20
7
3
1

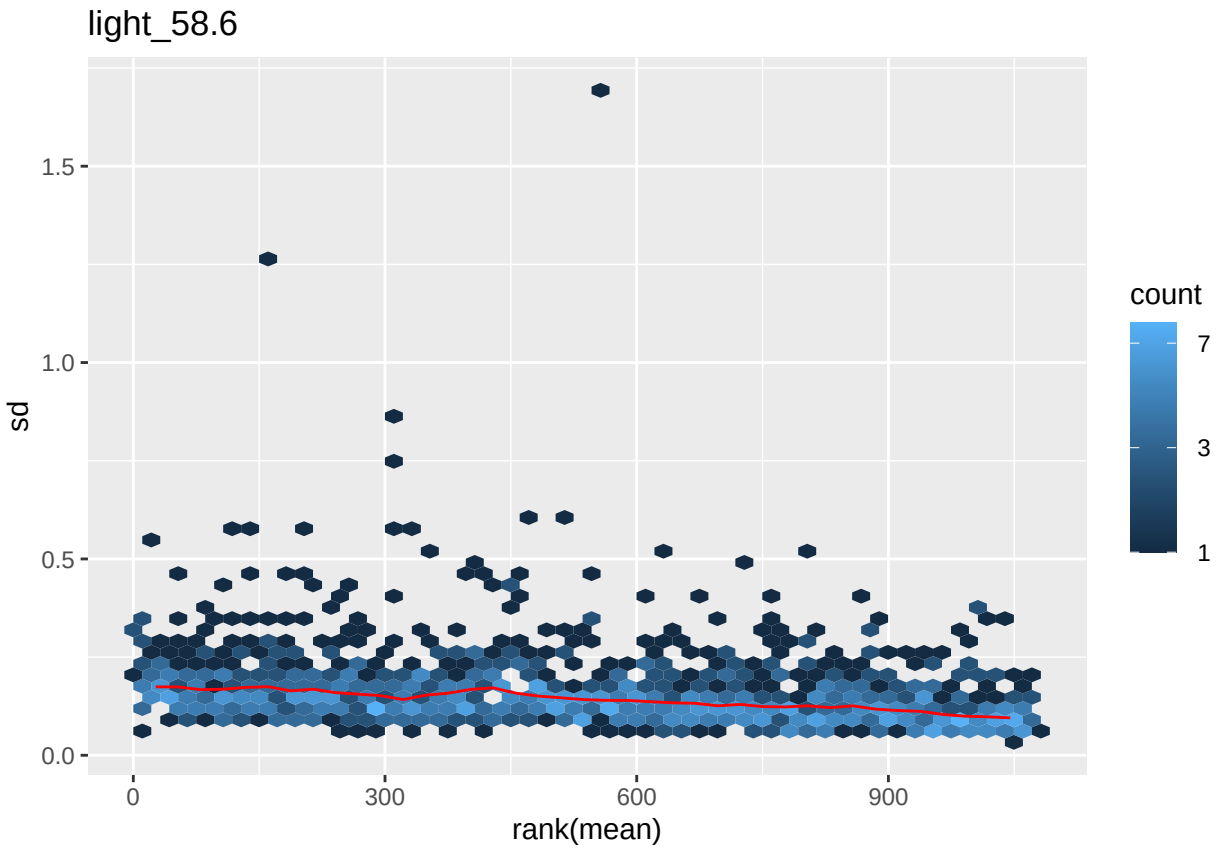
```
## [1] "light_52.9"
```



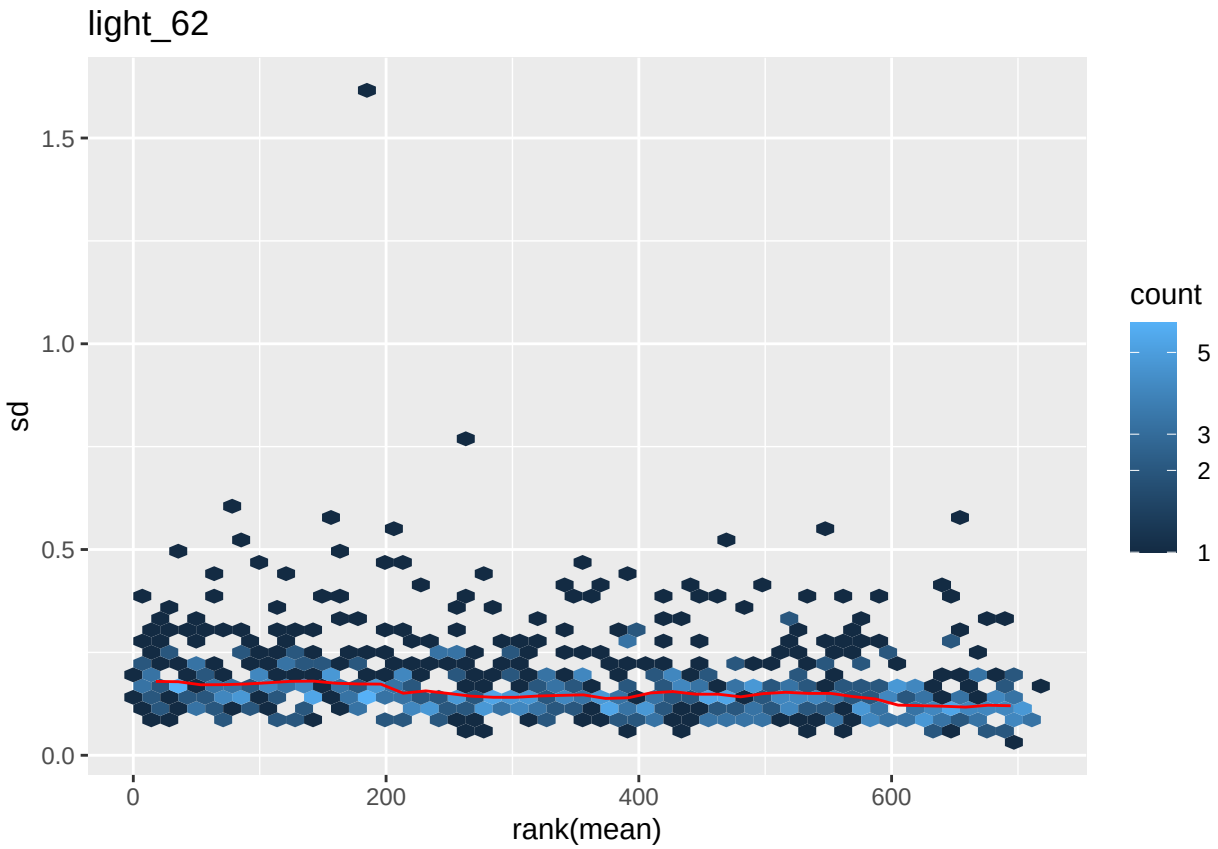

```
## [1] "light_55.5"
```



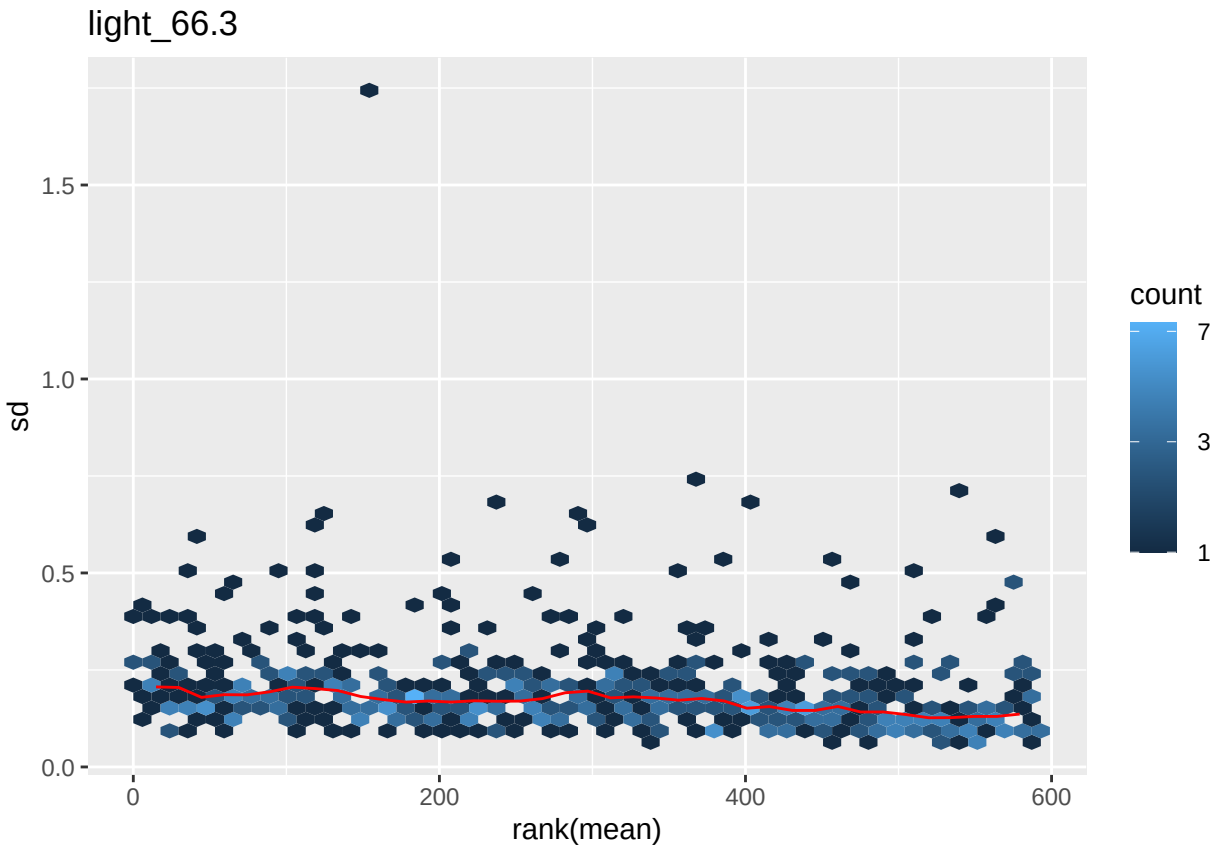

```
## [1] "light_58.6"
```



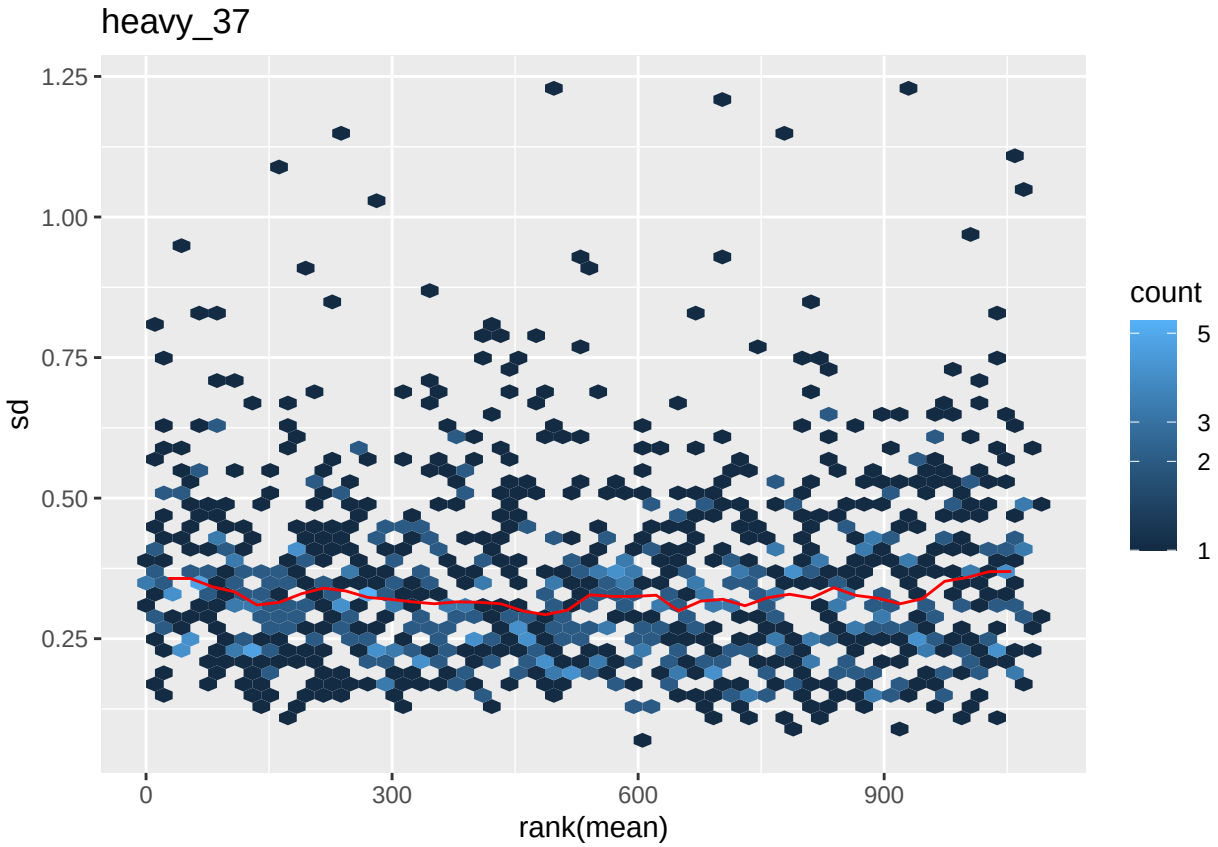

```
## [1] "light_62"
```



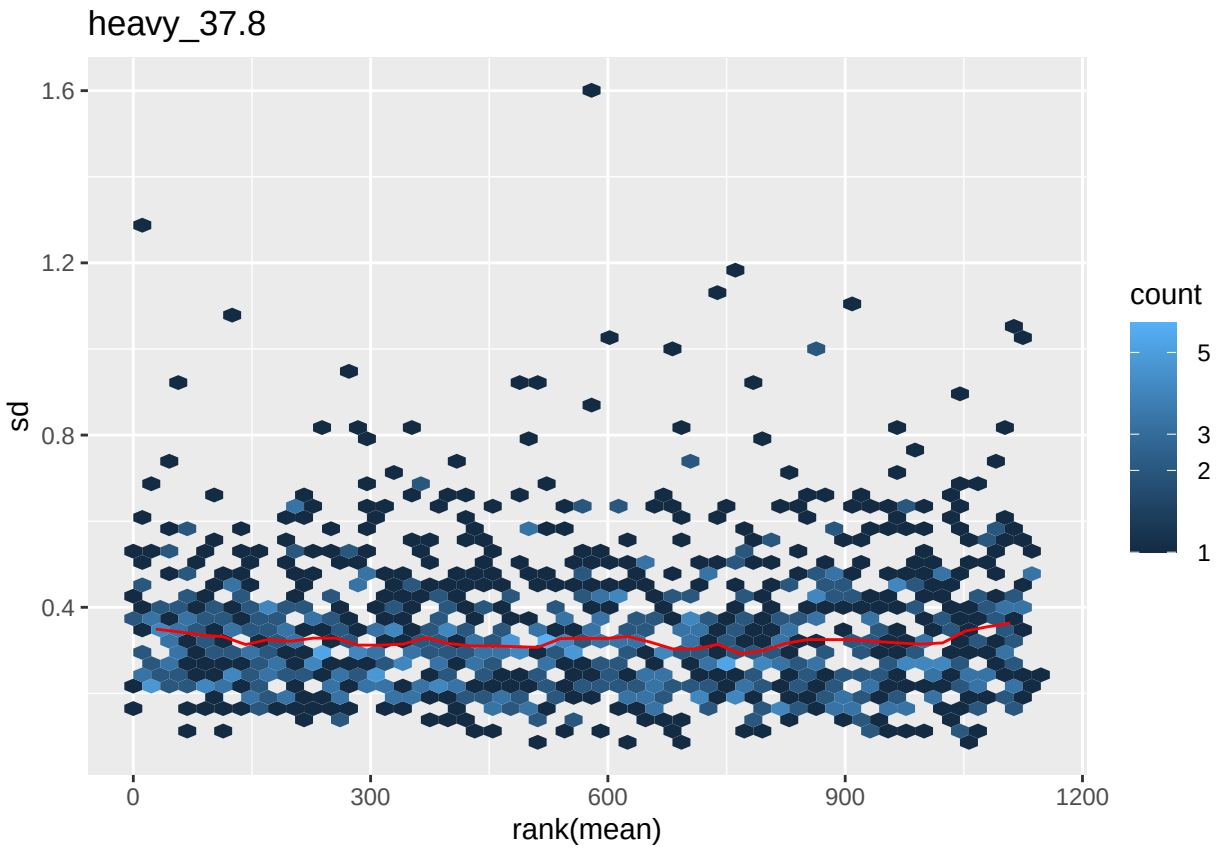

```
## [1] "light_66.3"
```



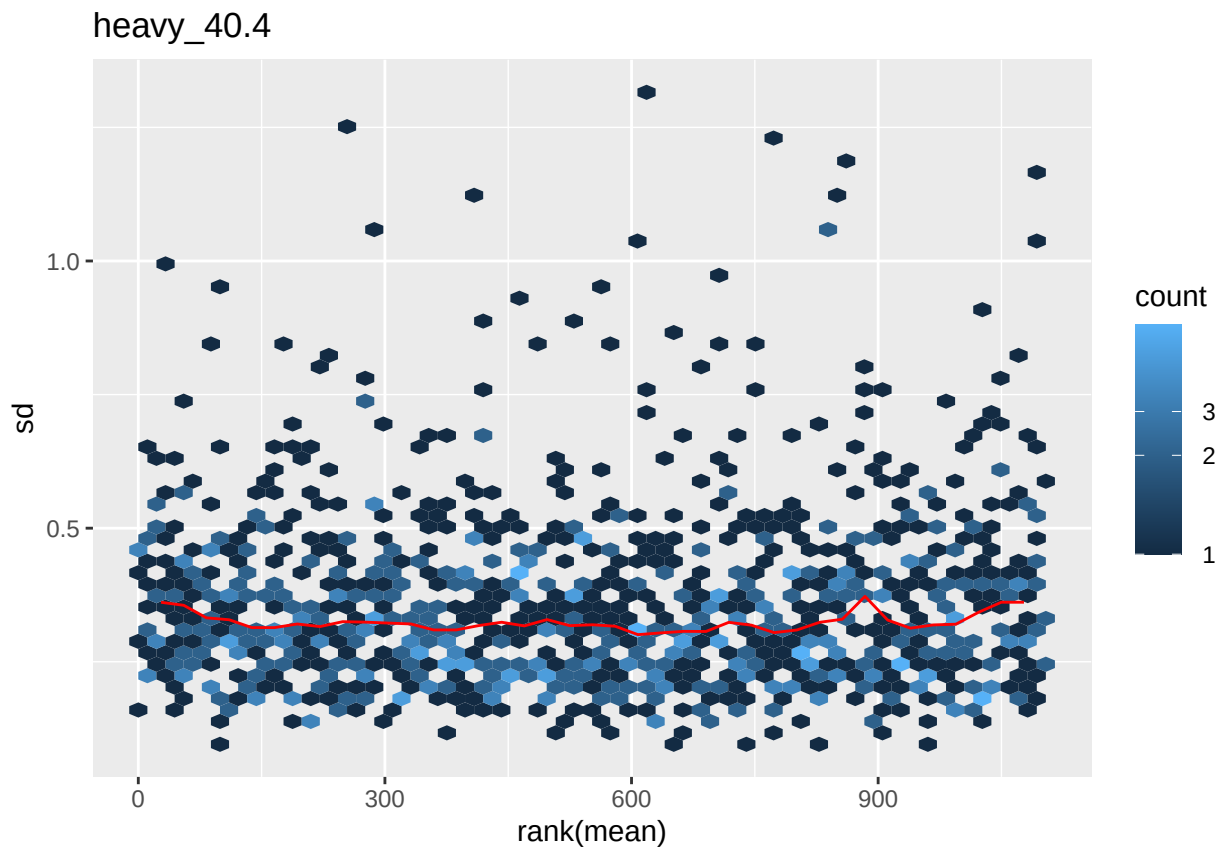

```
## [1] "heavy_37"
```


heavy_37
1.25
1.00
0.75
0.50
0.25
sd
0
300
600
900
rank(mean)
count
5
3
2
1

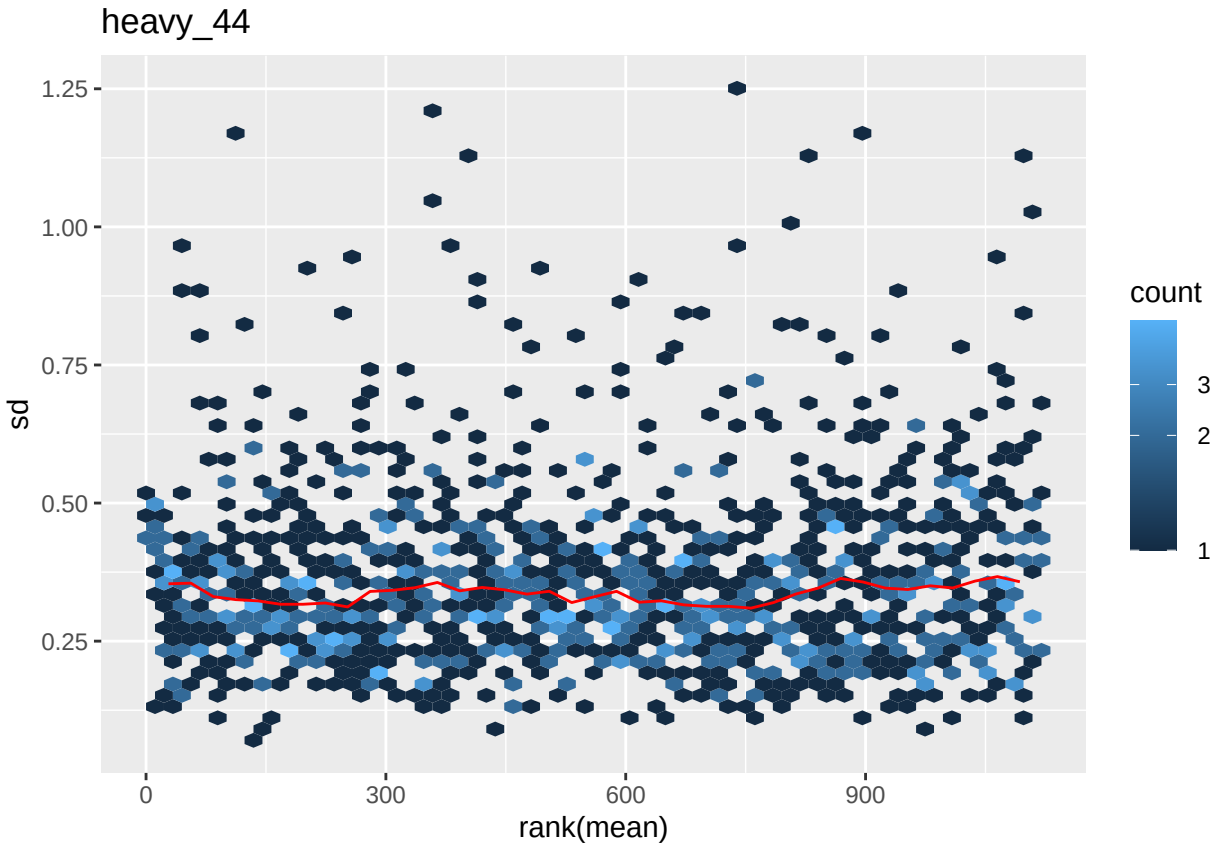
```
## [1] "heavy_37.8"
```



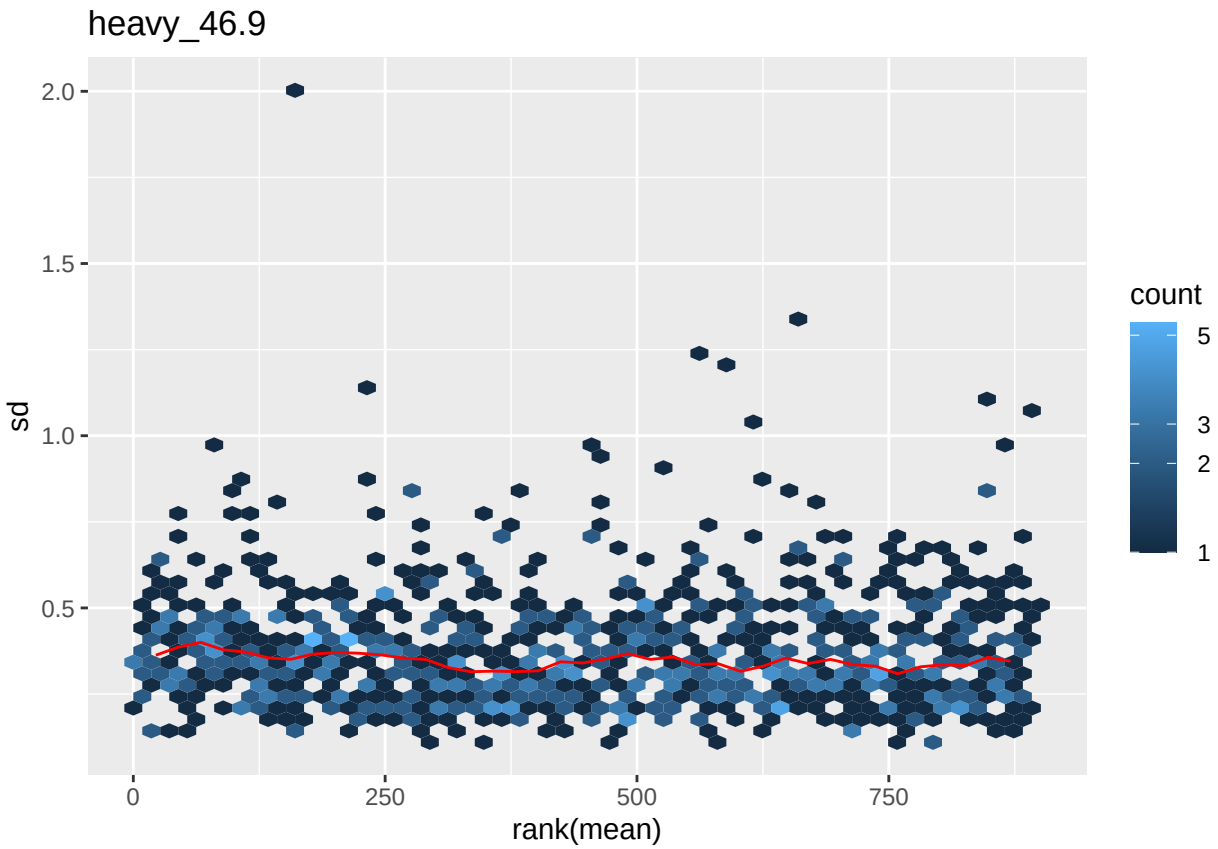

```
## [1] "heavy_40.4"
```



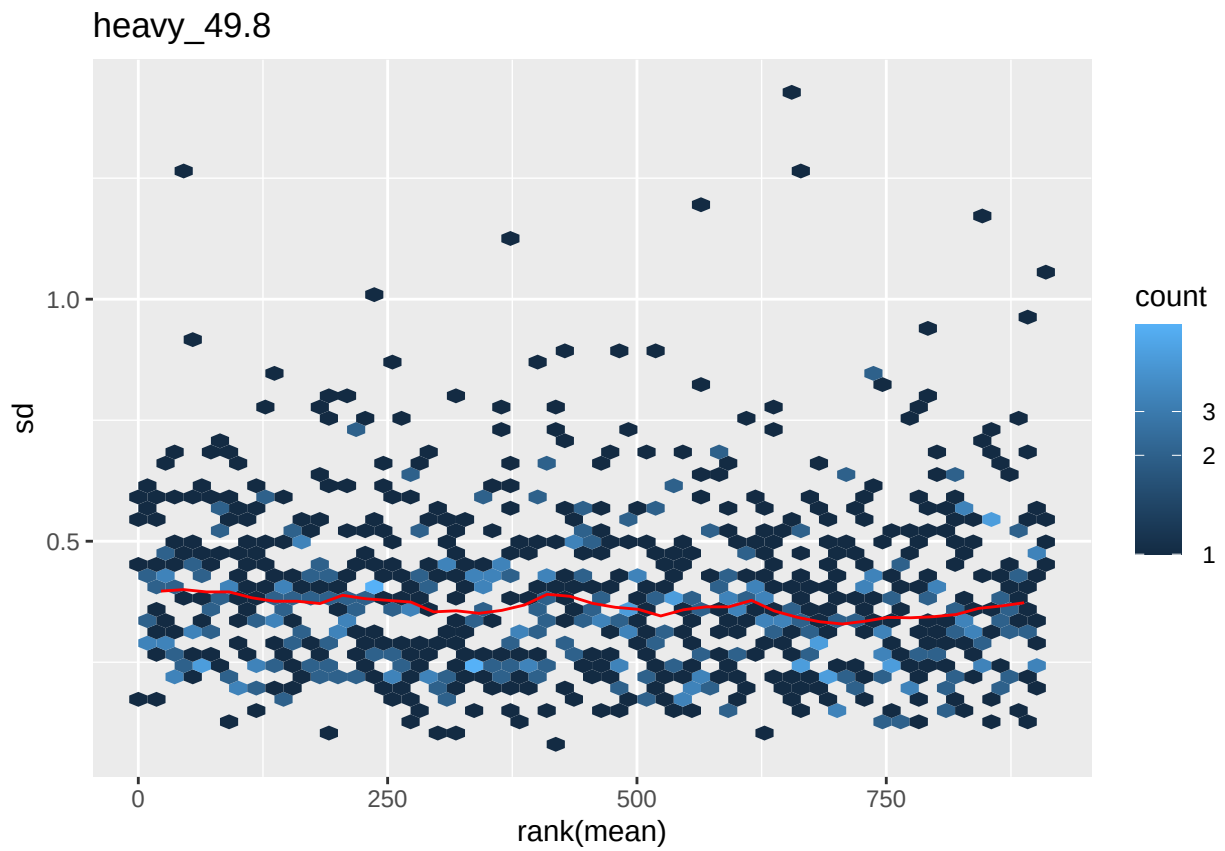

```
## [1] "heavy_44"
```


heavy_44
1.25
1.00
0.75
0.50
0.25
sd
0
300
600
900
rank(mean)
count
3
2
1

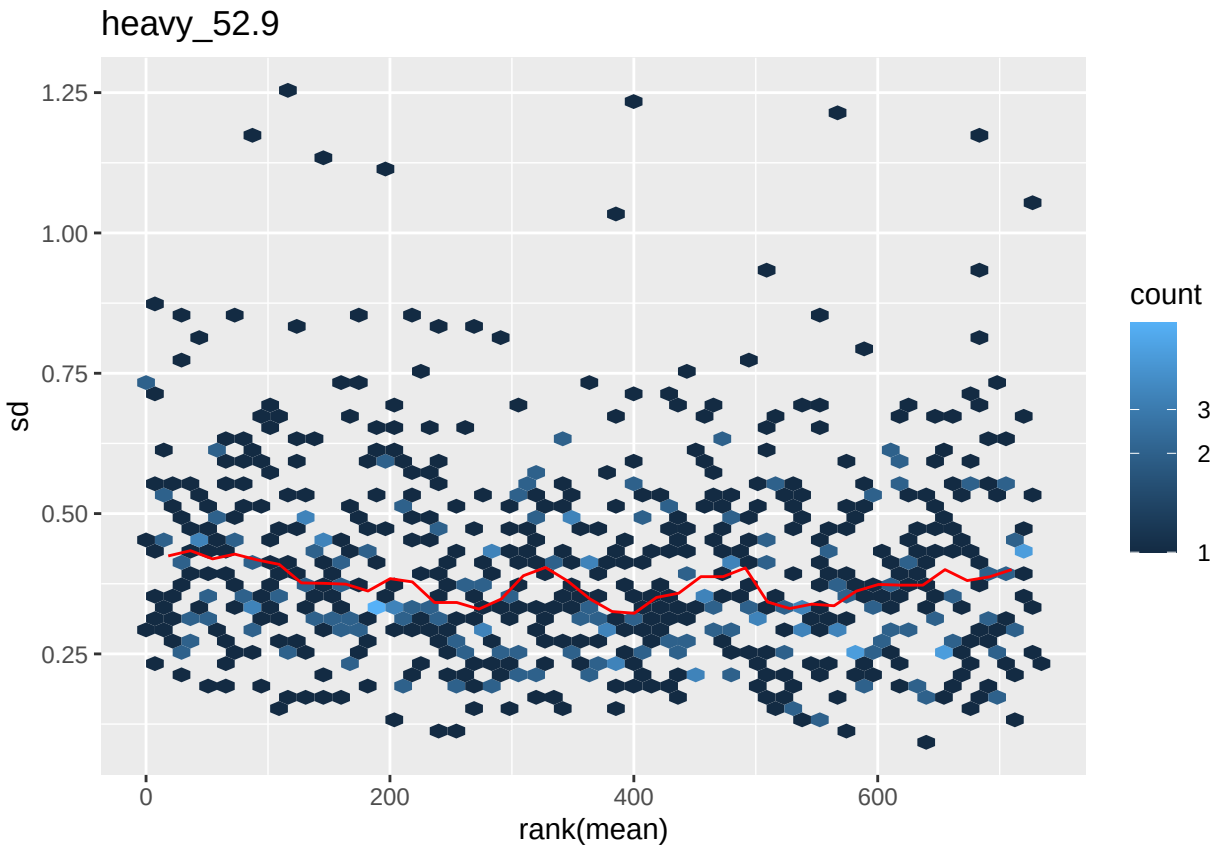
```
## [1] "heavy_46.9"
```



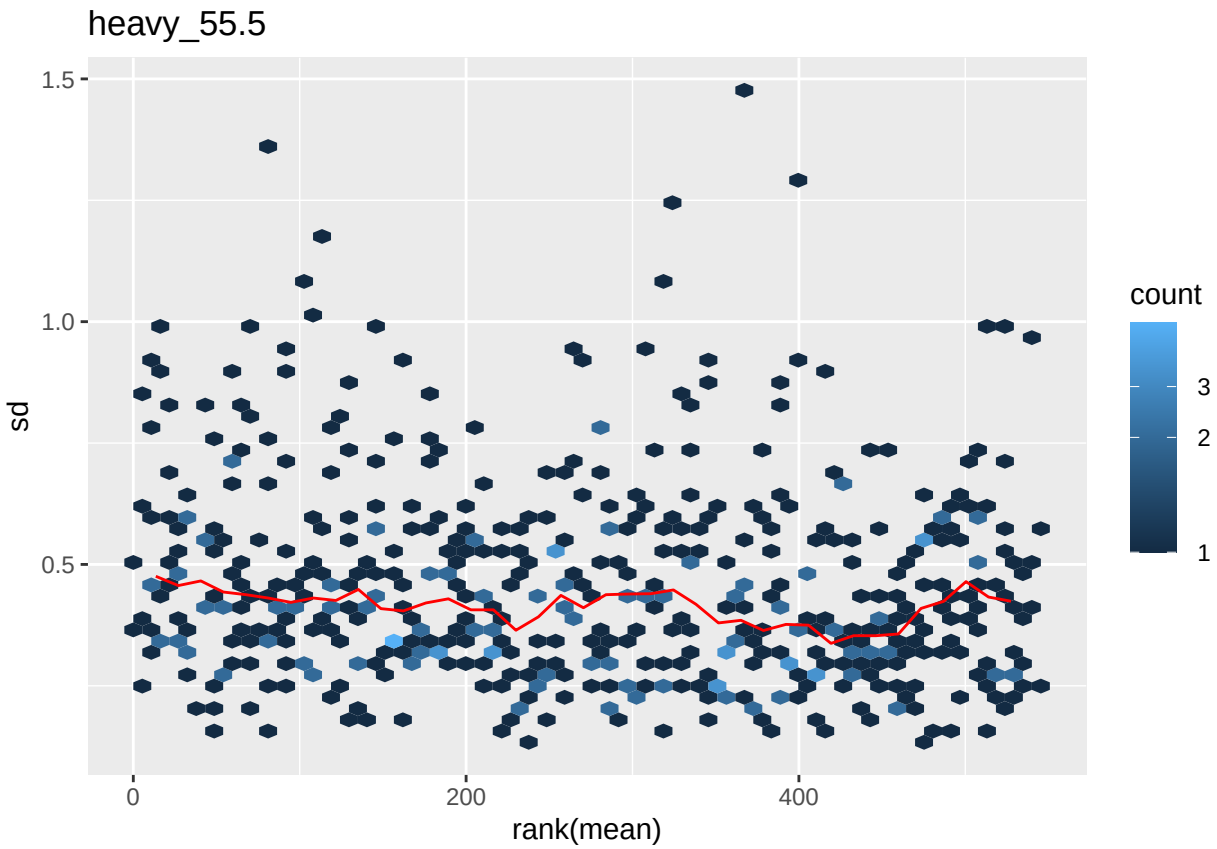

```
## [1] "heavy_49.8"
```



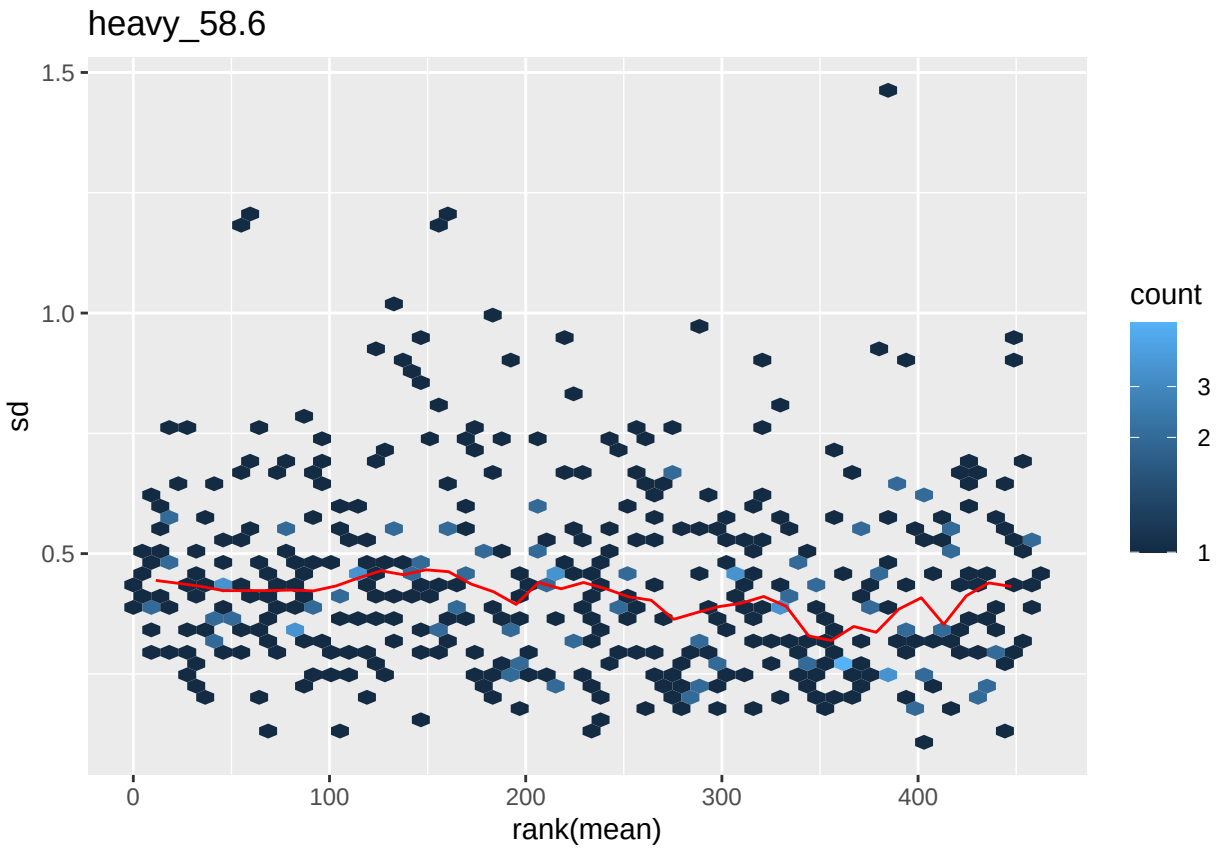

```
## [1] "heavy_52.9"
```


heavy_52.9
1.25
1.00
0.75
0.50
0.25
sd
0
200
400
600
rank(mean)
count
3
2
1

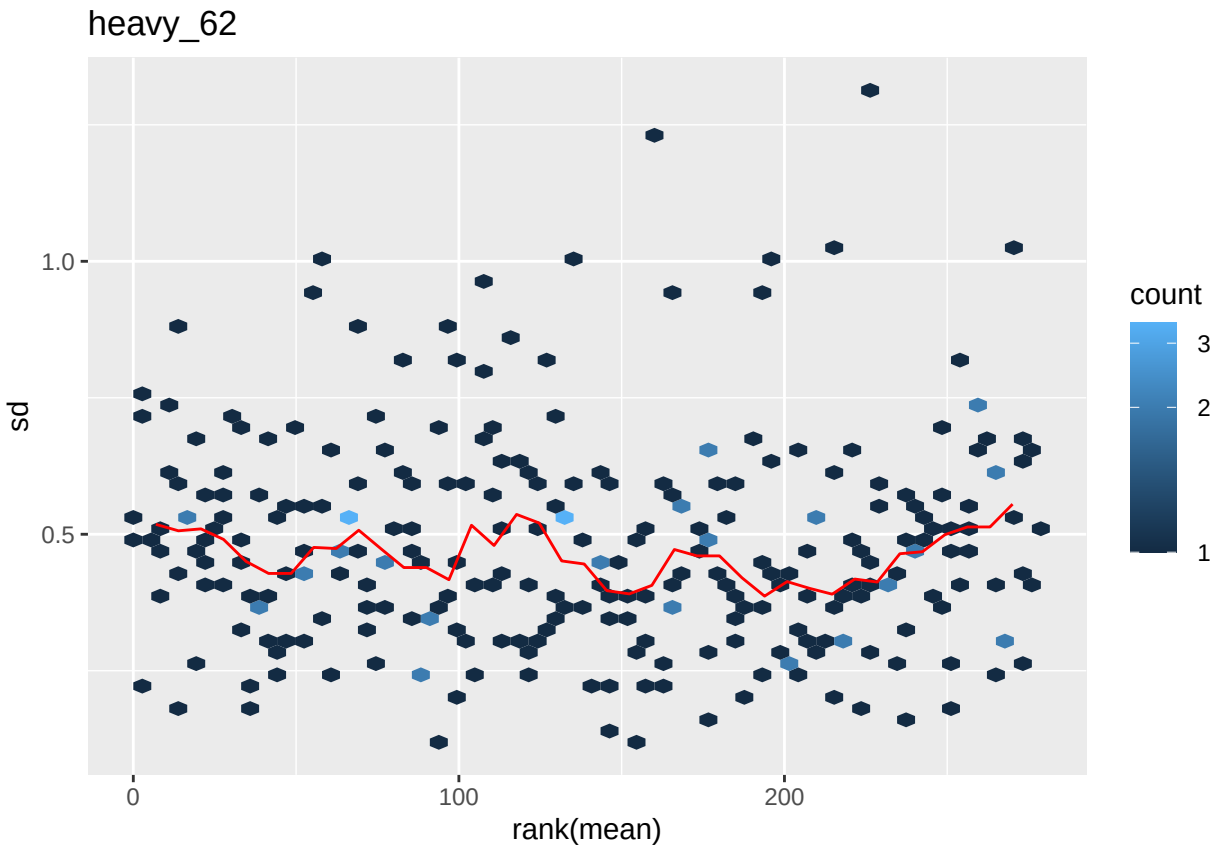
```
## [1] "heavy_55.5"
```



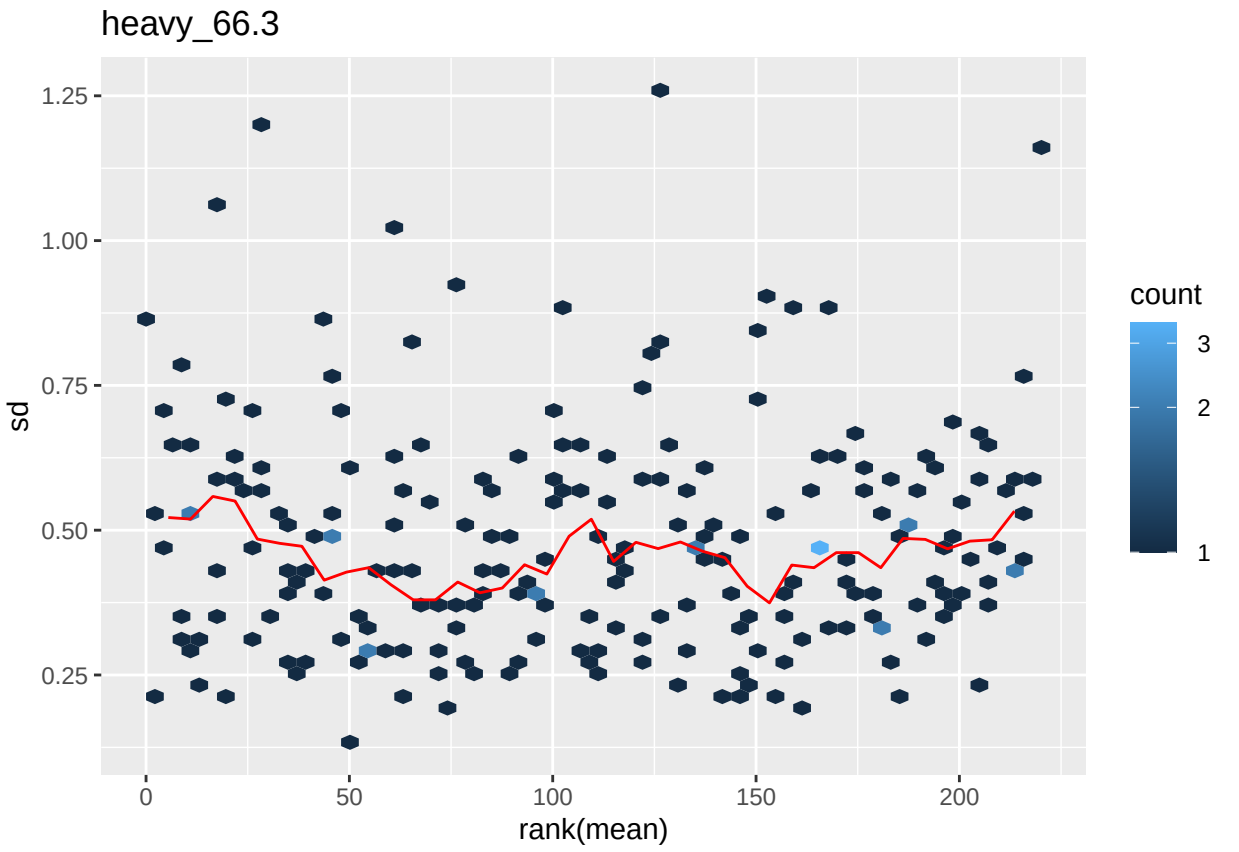

```
## [1] "heavy_58.6"
```




```
## [1] "heavy_62"
```




```
## [1] "heavy_66.3"
```

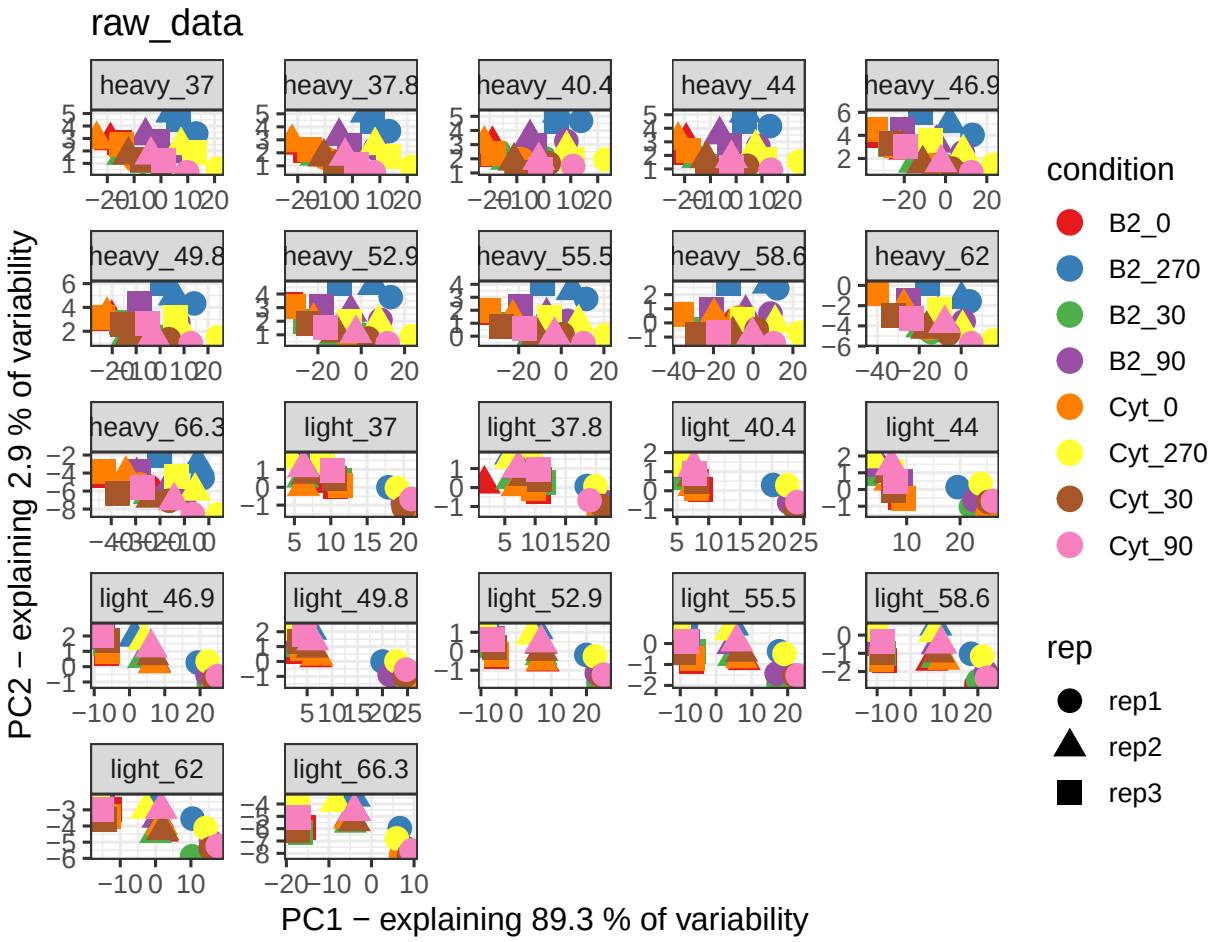



```
rm(s)
```

#PCA - principal component analysis

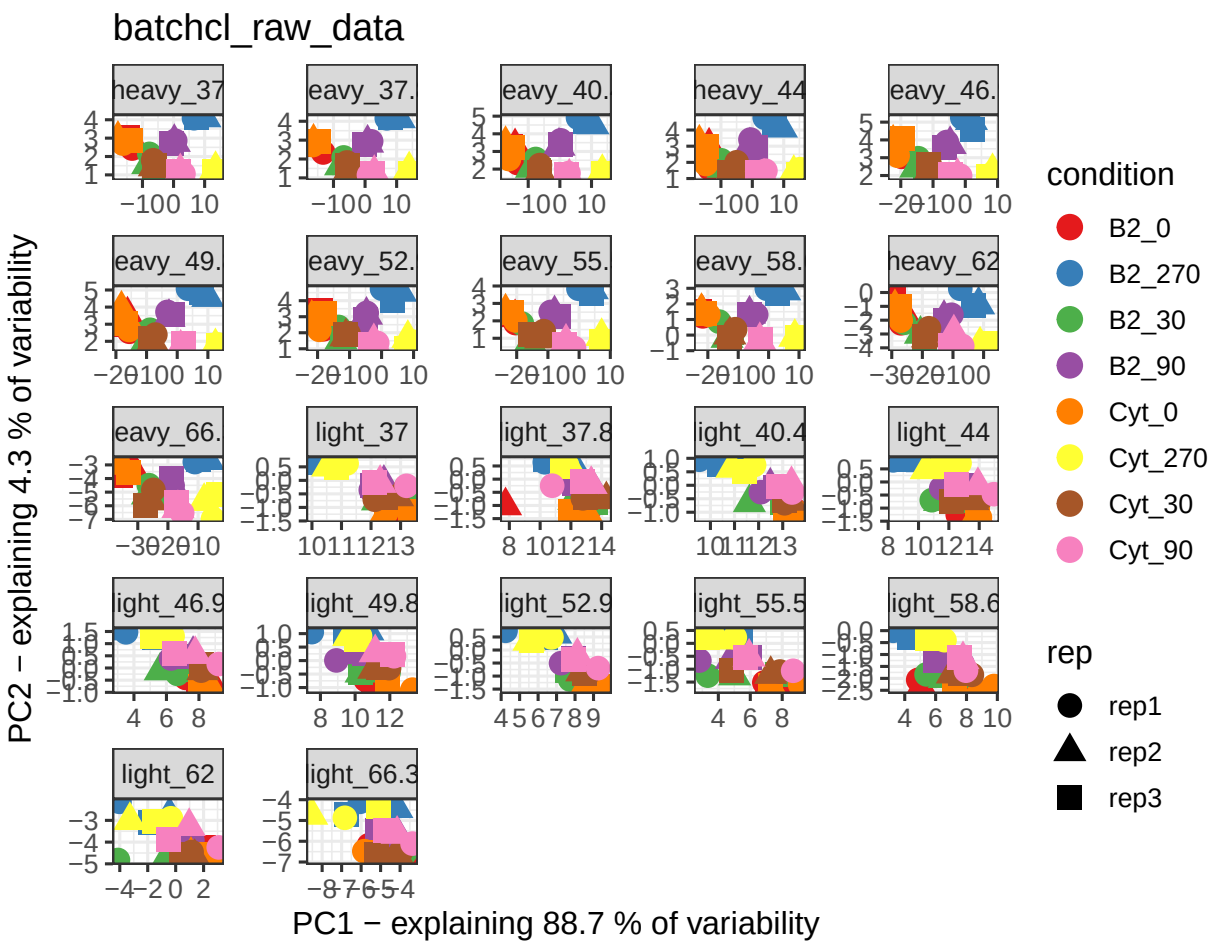
```
#PCA of raw_data
raw_PCA_m <- t(na.omit(exprs(raw_dataE)))
if (length(raw_PCA_m) > 50)
{
  raw_PCA <- prcomp(raw_PCA_m, scale = FALSE)
  perc_var <-
    round(100 * raw_PCA$sdev ^ 2 /
            sum(raw_PCA$sdev ^ 2), 1)
  PCA_raw_data <-
    data.frame(PC1 = raw_PCA$x[, 1],
               PC2 = raw_PCA$x[, 2],
               rep = pData(raw_dataE)$rep,
               condition = pData(raw_dataE)$condition,
               sample = pData(raw_dataE)$sample)
  print(ggplot(data = PCA_raw_data, aes(PC1, PC2)) +
          geom_point(aes(color = condition, shape = rep), size = 4) +
          guides(size = FALSE) +
          customPlot +
          ggtitle("raw_data") +
          xlab(paste("PC1 - explaining", perc_var[1], "% of variability")) +
          ylab(paste("PC2 - explaining", perc_var[2], "% of variability")) +
          facet_wrap(~ sample, scale = "free"))
  ggsave(file.path(dir_save, paste0("PCA_raw_data_", script.version, ".pdf")),
```

```
        width = 15, height = 12)
  rm(raw_PCA, perc_var, PCA_raw_data)
}
```



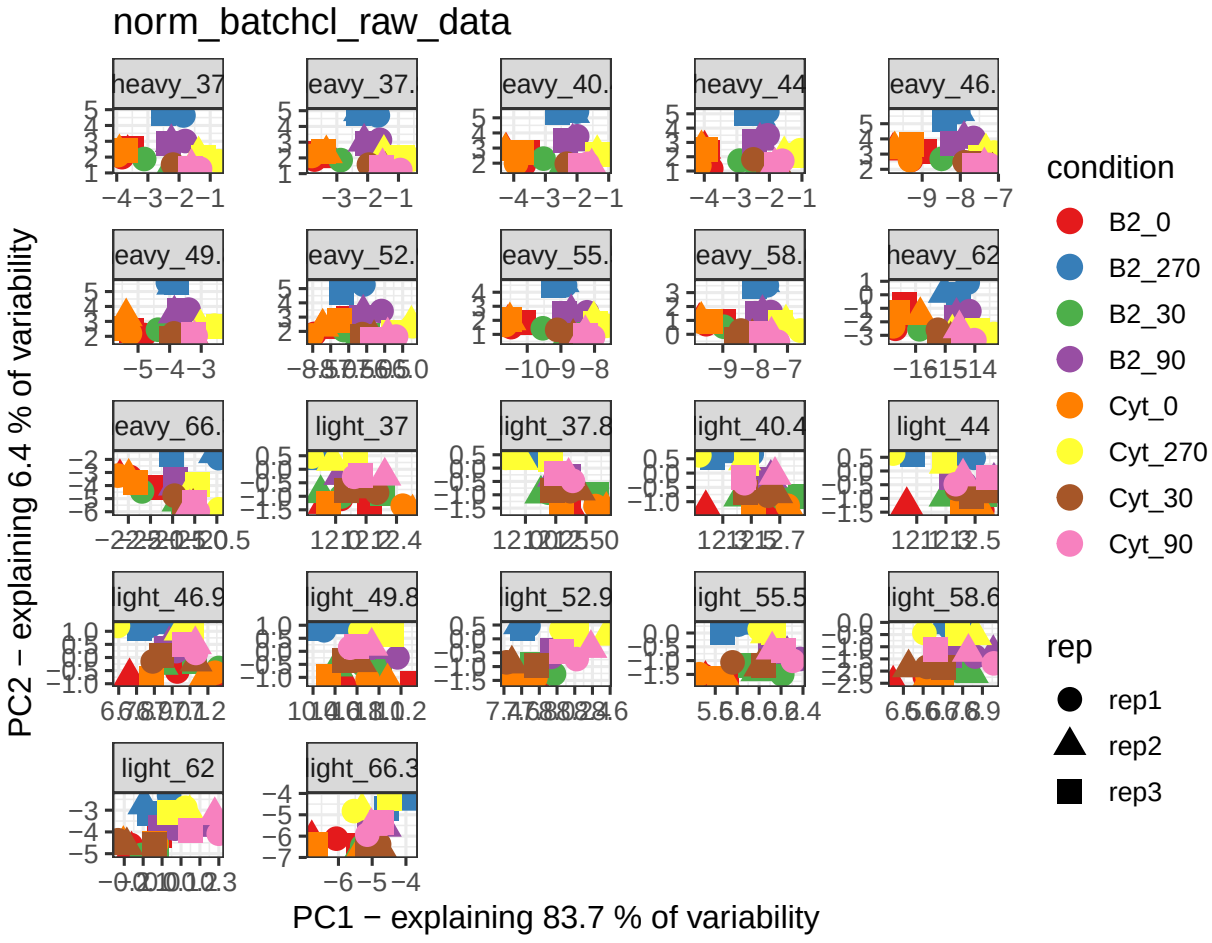

```
rm(raw_PCA_m)

#PCA of batchcl_raw_data
batchcl_raw_PCA_m <- t(na.omit(exprs(batchcl_raw_dataE)))
if (length(batchcl_raw_PCA_m) > 50)
{
  batchcl_raw_PCA <- prcomp(batchcl_raw_PCA_m, scale = FALSE)
  perc_var <-
    round(100 * batchcl_raw_PCA$sdev ^ 2 /
            sum(batchcl_raw_PCA$sdev ^ 2), 1)
  PCA_batchcl_raw_data <-
    data.frame(PC1 = batchcl_raw_PCA$x[, 1],
               PC2 = batchcl_raw_PCA$x[, 2],
               rep = pData(batchcl_raw_dataE)$rep,
               condition = pData(batchcl_raw_dataE)$condition,
               sample = pData(batchcl_raw_dataE)$sample)
  print(ggplot(data = PCA_batchcl_raw_data, aes(PC1, PC2)) +
          geom_point(aes(color = condition, shape = rep), size = 4) +
          guides(size = FALSE) +
```

```
          customPlot +
          ggtitle("batchcl_raw_data") +
          xlab(paste("PC1 - explaining", perc_var[1], "% of variability")) +
          ylab(paste("PC2 - explaining", perc_var[2], "% of variability")) +
          facet_wrap(~ sample, scale = "free"))
  ggsave(file.path(dir_save, paste0("PCA_batchcl_raw_data_", script.version, ".pdf")),
         width = 15, height = 12)
  rm(batchcl_raw_PCA, perc_var, PCA_batchcl_raw_data)
}
```




```
rm(batchcl_raw_PCA_m)

#PCA of norm_batchcl_raw_data
norm_batchcl_raw_PCA_m <- t(na.omit(exprs(norm_batchcl_raw_dataE)))
if (length(norm_batchcl_raw_PCA_m) > 50)
{
  norm_batchcl_raw_PCA <- prcomp(norm_batchcl_raw_PCA_m, scale = FALSE)
  perc_var <-
    round(100 * norm_batchcl_raw_PCA$sdev ^ 2 /
            sum(norm_batchcl_raw_PCA$sdev ^ 2), 1)
  PCA_norm_batchcl_raw_data <-
    data.frame(PC1 = norm_batchcl_raw_PCA$x[, 1],
               PC2 = norm_batchcl_raw_PCA$x[, 2],
```

```
                rep = pData(norm_batchcl_raw_dataE)$rep,
                condition = pData(norm_batchcl_raw_dataE)$condition,
                sample = pData(norm_batchcl_raw_dataE)$sample)
  print(ggplot(data = PCA_norm_batchcl_raw_data, aes(PC1, PC2)) +
          geom_point(aes(color = condition, shape = rep), size = 4) +
          guides(size = FALSE) +
          customPlot +
          ggtitle("norm_batchcl_raw_data") +
          xlab(paste("PC1 - explaining", perc_var[1], "% of variability")) +
          ylab(paste("PC2 - explaining", perc_var[2], "% of variability")) +
          facet_wrap(~ sample, scale = "free"))
  ggsave(file.path(dir_save, paste0("PCA_norm_batchcl_raw_data_", script.version, ".pdf")),
         width = 15, height = 12)
  rm(norm_batchcl_raw_PCA, perc_var, PCA_norm_batchcl_raw_data)
}
```




```
rm(norm_batchcl_raw_PCA_m)
```

#Merge modified data back into 'cdata'

```
cdata_i <- as.data.frame(2^exprs(batchcl_raw_dataE))
names(cdata_i) <- paste0("batchcl_raw_signal_sum_", names(cdata_i))
cdata_i$gene_name <- rownames(cdata_i)
cdata <- left_join(cdata, cdata_i)
```

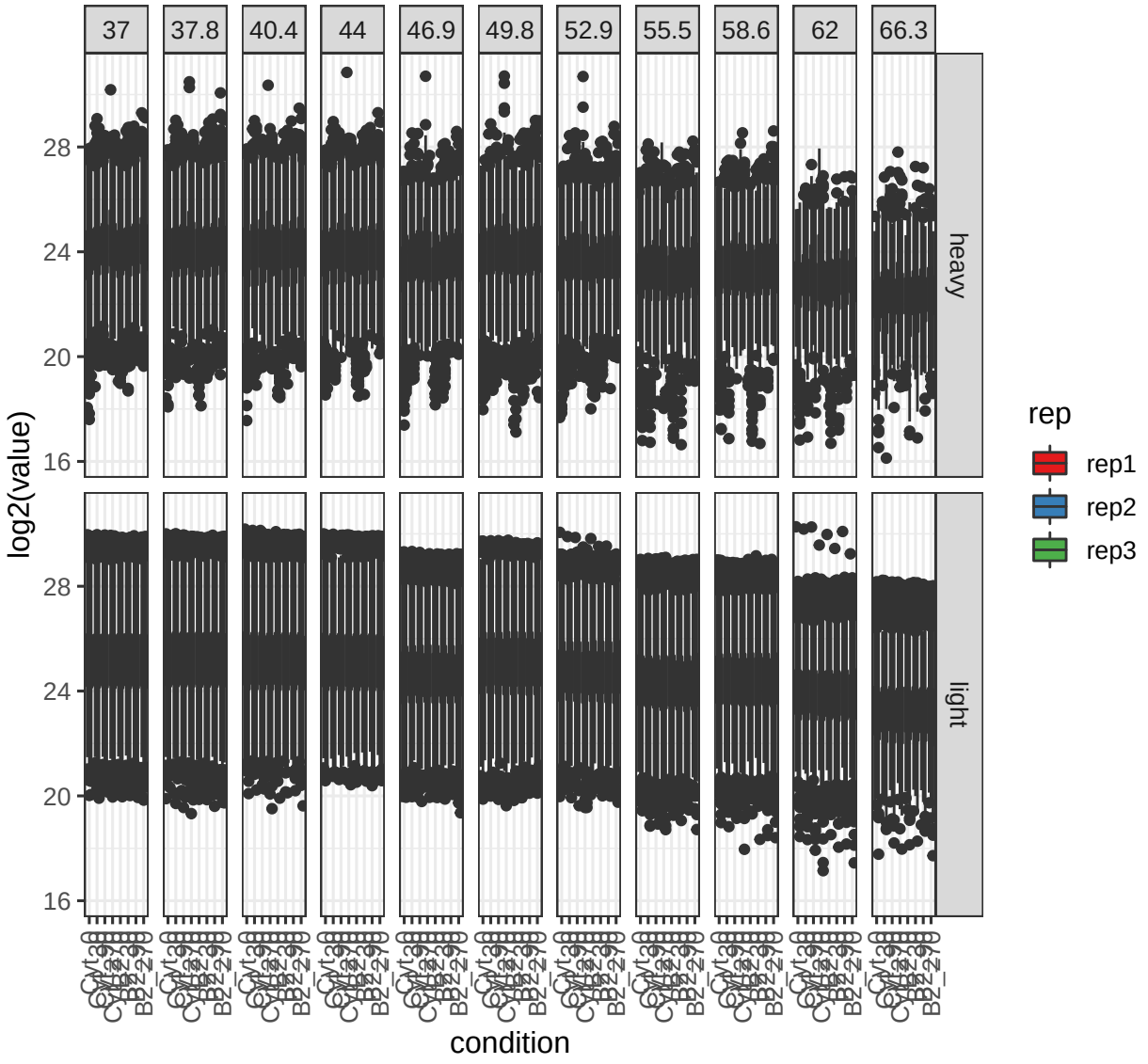
```
rm(cdata_i)

cdata_i <- as.data.frame(2^exprs(norm_batchcl_raw_dataE))
names(cdata_i) <- paste0("norm_batchcl_raw_signal_sum_", names(cdata_i))
cdata_i$gene_name <- rownames(cdata_i)
cdata <- left_join(cdata, cdata_i)
rm(cdata_i)

write.csv(cdata, file = file.path(dir_save, paste0("Full_dataset_", script.version, ".csv")))
```

#Create tidy data

```
mdata <- NULL

mdata_i <- tidy(raw_dataE)
mdata_i <- mdata_i %>%
  mutate(value = 2 ^ value)
names(mdata_i)[1] <- "gene_name"
names(mdata_i)[2] <- "ID"
mdata_i <- left_join(mdata_i, conditions)
mdata_i$measurement <- "raw_signal_sum"
mdata <- bind_rows(mdata, mdata_i)
rm(mdata_i)

mdata_i <- tidy(batchcl_raw_dataE)
mdata_i <- mdata_i %>%
  mutate(value = 2 ^ value)
names(mdata_i)[1] <- "gene_name"
names(mdata_i)[2] <- "ID"
mdata_i <- left_join(mdata_i, conditions)
mdata_i$measurement <- "batchcl_raw_signal_sum"
mdata <- bind_rows(mdata, mdata_i)
rm(mdata_i)

mdata_i <- tidy(norm_batchcl_raw_dataE)
mdata_i <- mdata_i %>%
  mutate(value = 2 ^ value)
names(mdata_i)[1] <- "gene_name"
names(mdata_i)[2] <- "ID"
mdata_i <- left_join(mdata_i, conditions)
mdata_i$measurement <- "norm_batchcl_raw_signal_sum"
mdata <- bind_rows(mdata, mdata_i)
rm(mdata_i)

mdata$condition <- factor(mdata$condition, ordered = TRUE, levels = c("Cyt_0", "Cyt_30", "Cyt_90", "Cyt
conditions$condition <- factor(conditions$condition, ordered = TRUE, levels = as.character(unique(condi
mdata$measurement <- factor(mdata$measurement, ordered = TRUE, levels = c("raw_signal_sum", "batchcl_ra
```

#Data transformation overview

```
ggplot(data = subset(mdata, grepl("norm_batchcl_raw", measurement)), aes(condition, log2(value))) +
  geom_boxplot(aes(fill = rep)) +
  customPlot +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  facet_grid(SILAC ~ Temperature)
```




```
ggsave(file.path(dir_save, paste0("Normalization_overview_", script.version, ".pdf")), width = 25, heigh
```

#Calculation of fold changes

## Calculation of ctrl.ratios

```
fc.data <- as.data.frame(2^exprs(norm_batchcl_raw_dataE))
initial.names <- names(fc.data)
comps <- read.csv("stat_comparisons.csv")
fc.metadata <- NULL
for (i in seq_along(comps$ratio.name))
{
  IDs <- grep(paste0("[0-9]_", comps$nominator[i], "_rep"), conditions$ID, value = TRUE)
  if (length(IDs) == 0)
  {
    IDs <- grep(paste0(comps$nominator[i], "_rep"), fc.metadata$new.ID, value = TRUE)
  }
```

```
  for (j in seq_along(IDs))
  {
    nominator <- IDs[j]
    denominator <- gsub(comps$nominator[i], comps$denominator[i], nominator)
    ratio.name <- paste0(gsub(comps$nominator[i], comps$ratio.name[i], nominator),
                         ".ctrl.ratio")
    if (nominator %in% conditions$ID & denominator %in% conditions$ID)
    {
      fc.metadata <- bind_rows(fc.metadata,
                               data.frame(ID = nominator,
                                          new.ID = ratio.name,
                                          condition = comps$ratio.name[i]))
    }else
    {
      fc.metadata <- bind_rows(fc.metadata,
                               data.frame(ID = subset(fc.metadata, new.ID == nominator)$ID,
                                          new.ID = ratio.name,
                                          condition = comps$ratio.name[i]))
    }
    if (denominator %in% names(fc.data))
    {
      fc.data[, ratio.name] <-
        fc.data[, as.character(nominator)] /
        fc.data[, as.character(denominator)]
    }else
    {
      print(paste(denominator, "was not a column name in fc.data"))
      stop()
    }
  }
  rm(j, IDs)
}
rm(i)
fc.data <- fc.data %>%
  dplyr::select(ends_with(".ctrl.ratio"))
fc.data$gene_name <- rownames(fc.data)
cdata <- left_join(cdata, fc.data)
fc.metadata <- left_join(fc.metadata,
                         conditions[, c("ID",
                                        "rep",
                                        "Temperature",
                                        "construct",
                                        "time",
                                        "SILAC",
                                        "sample",
                                        "col.name",
                                        "file")])
fc.metadata$ID <- fc.metadata$new.ID
fc.metadata$new.ID <- NULL
mfc.data <- fc.data %>%
  group_by(gene_name) %>%
  gather(key = ID, value = value, -gene_name) %>%
  mutate(measurement = "ctrl.ratio")
```

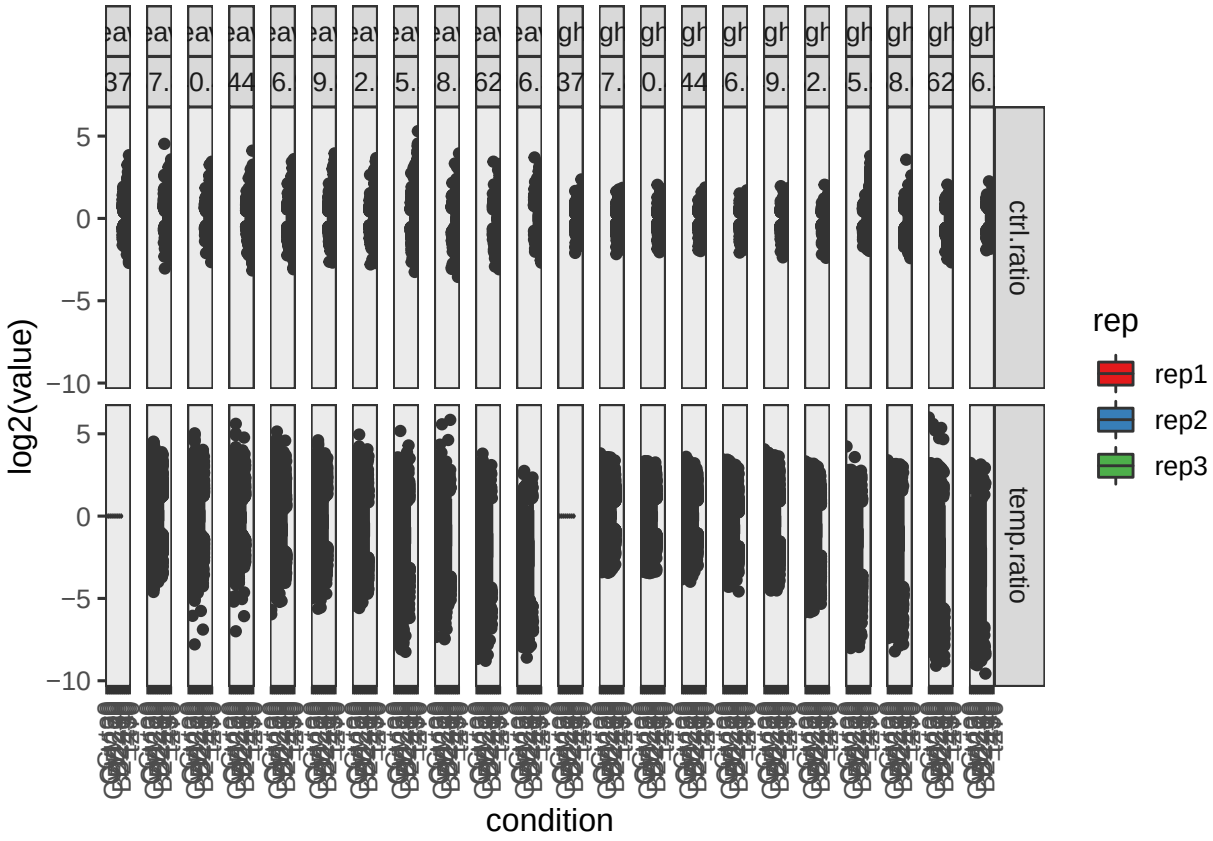
```
mfc.data <- left_join(mfc.data, fc.metadata)
mfc.data <- mfc.data[, names(mdata)]
mdata <- bind_rows(mdata, mfc.data)
rm(fc.data, mfc.data, fc.metadata)
```

### Calculation of temp.ratios

```
fc.data <- as.data.frame(2^exprs(norm_batchcl_raw_dataE))
for (i in seq_along(conditions$ID))
{
  sort(conditions$Temperature)[1]
  fc.data[, paste0(conditions$ID[i], ".temp.ratio")] <-
    fc.data[, as.character(conditions$ID[i])] /
    fc.data[, as.character(gsub(conditions$Temperature[i],
                                sort(conditions$Temperature)[1],
                                conditions$ID[i]))]
}
rm(i)
fc.data <- fc.data %>%
  dplyr::select(ends_with(".temp.ratio"))
fc.data$gene_name <- rownames(fc.data)
cdata <- left_join(cdata, fc.data)
fc.metadata <- data.frame(ID = gsub(".temp.ratio", "", names(fc.data)),
                          new.ID = names(fc.data))
fc.metadata <- left_join(fc.metadata, conditions)
fc.metadata$ID <- fc.metadata$new.ID
fc.metadata$new.ID <- NULL
mfc.data <- fc.data %>%
  group_by(gene_name) %>%
  gather(key = ID, value = value, -gene_name) %>%
  mutate(measurement = "temp.ratio")
mfc.data <- left_join(mfc.data, fc.metadata)
mfc.data <- mfc.data[, names(mdata)]
mdata <- bind_rows(mdata, mfc.data)
rm(fc.data, mfc.data, fc.metadata)
```

#Factorize tidy data variables

```
mdata$condition <- factor(mdata$condition, ordered = TRUE, levels = unique(append(as.character(levels(c
mdata$measurement <- factor(mdata$measurement, ordered = TRUE, levels = c("raw_signal_sum", "batchcl_ra

ggplot(data = subset(mdata, grepl("ratio", measurement)), aes(condition, log2(value))) +
  geom_boxplot(aes(fill = rep)) +
  customPlot +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  facet_grid(measurement ~ SILAC + Temperature)
```




```
ggsave(file.path(dir_save, paste0("Normalization_overview_ratios_", script.version, ".pdf")), width = 40
```

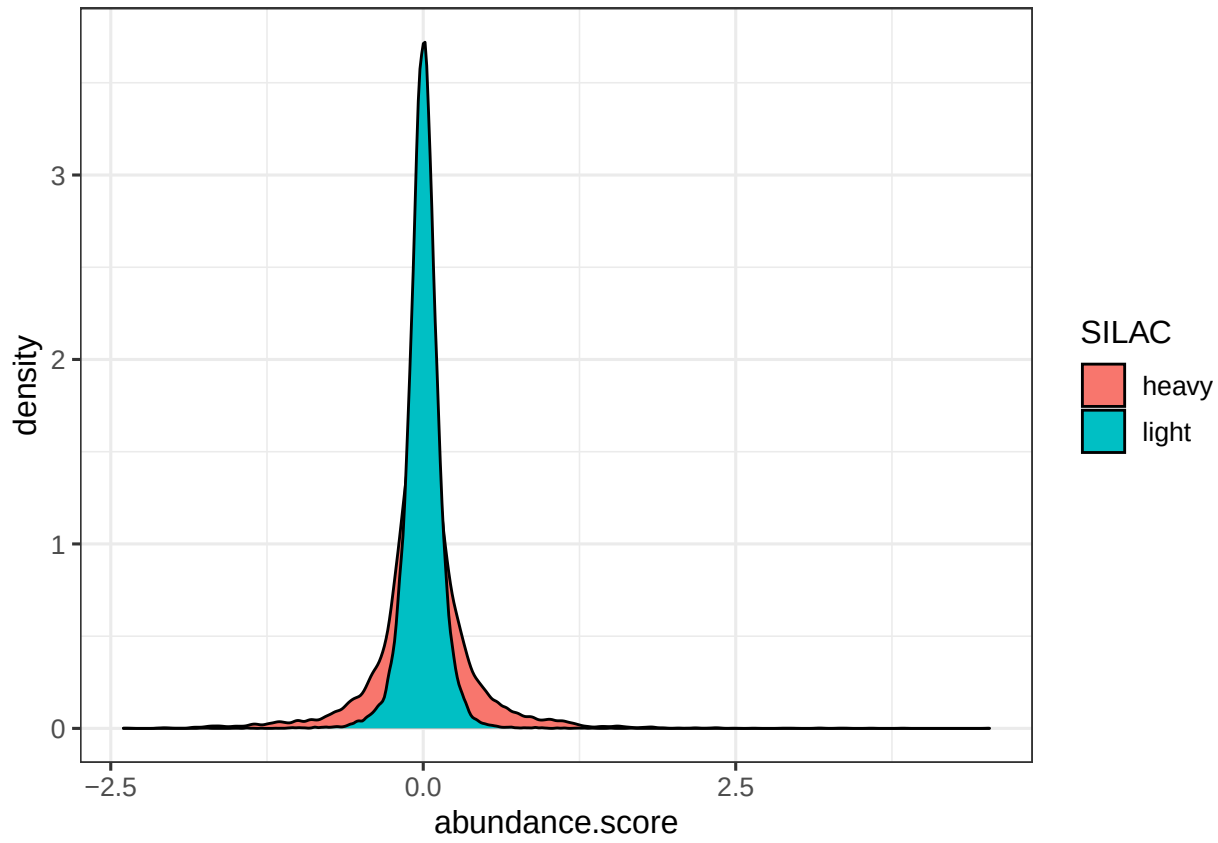
#LIMMA analysis

##Creating limma data set

```
limma.score.data <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio", !is.na(value)) %>%
  mutate(ratio = value) %>%
  dplyr::select(gene_name, condition, Temperature, SILAC, rep, ratio)
limma.score.data.length <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio",
                Temperature %in% sort(unique(conditions$Temperature))[1:2],
                !is.na(value)) %>%
  group_by(gene_name, SILAC, condition, rep) %>%
  summarise(No.of.abundance.Ts = n())
limma.score.data <- left_join(limma.score.data, limma.score.data.length)
rm(limma.score.data.length)
limma.score.data.length <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio",
                !is.na(value)) %>%
  group_by(gene_name, SILAC, condition, rep) %>%
  summarise(No.of.Ts.total = n())
limma.score.data <- left_join(limma.score.data, limma.score.data.length)
rm(limma.score.data.length)
rep.cut.off <- 2
#allowing only proteins that have at least 2/3 of all replicates present for at least one of the two lo
```

```
score.data.length <- limma.score.data %>%
  dplyr::filter(No.of.abundance.Ts >= 1,
                Temperature %in% sort(unique(conditions$Temperature))[1:2],
                !is.na(ratio)) %>%
  group_by(gene_name, SILAC, condition, Temperature) %>%
  summarise(rep = n()) %>%
  dplyr::filter(rep >= rep.cut.off) %>%
  group_by(gene_name, SILAC, condition) %>%
  summarise(Freq = n()) %>%
  dplyr::filter(Freq >= 1) %>%
  dplyr::select(gene_name, SILAC, condition) %>%
  mutate(abundance.score.valid = TRUE)
limma.score.data <- left_join(limma.score.data, score.data.length)
rm(score.data.length)
limma.score.data$abundance.score.valid[limma.score.data$No.of.abundance.Ts < 1] <- NA
limma.score.data$abundance.score.valid[is.na(limma.score.data$No.of.abundance.Ts)] <- NA
#allowing only Temperature for the stability score calculation for which data points of at least 2/3rd
score.data.length <- limma.score.data %>%
  dplyr::filter(No.of.abundance.Ts >= 1,
                abundance.score.valid,
                No.of.Ts.total >= 5,
                !is.na(ratio)) %>%
  group_by(gene_name, SILAC, condition, Temperature) %>%
  summarise(rep = n()) %>%
  dplyr::filter(rep >= rep.cut.off) %>%
  mutate(stability.score.valid = TRUE) %>%
  dplyr::select(gene_name, SILAC, condition, Temperature, stability.score.valid)
# score.data.length <- mdata %>%
#   dplyr::filter(measurement == "ctrl.ratio",
#                 !is.na(value)) %>%
#   group_by(gene_name, SILAC, condition, Temperature) %>%
#   summarise(rep = n()) %>%
#   dplyr::filter(rep >= rep.cut.off) %>%
#   mutate(stability.score.valid = TRUE) %>%
#   dplyr::select(gene_name, SILAC, condition, Temperature, stability.score.valid)

limma.score.data <- left_join(limma.score.data, score.data.length)
limma.score.data$stability.score.valid[limma.score.data$No.of.Ts.total < 5] <- NA
score.data.length <- limma.score.data %>%
  group_by(gene_name, condition, SILAC, rep) %>%
  dplyr::filter(abundance.score.valid,
                stability.score.valid) %>%
  summarise(n = n()) %>%
  dplyr::filter(n >= 5) %>%
  group_by(gene_name, condition, SILAC) %>%
  summarise(n = n()) %>%
  dplyr::filter(n >= 2) %>%
  dplyr::select(gene_name, condition, SILAC) %>%
  mutate(stability.score.valid.rep = TRUE)
limma.score.data <- left_join(limma.score.data, score.data.length)
rm(score.data.length)
limma.score.data$stability.score.valid <- limma.score.data$stability.score.valid & limma.score.data$stab
limma.score.data$stability.score.valid.rep <- NULL
```

```r
abundance.score <- limma.score.data %>%
  subset(Temperature %in% sort(unique(limma.score.data$Temperature))[1:2]) %>%
  subset(abundance.score.valid) %>%
  group_by(gene_name, SILAC, condition,rep) %>%
  summarise(abundance.score = mean(log2(ratio), na.rm = TRUE))
limma.score.data <- left_join(limma.score.data, abundance.score)
limma.score.data <- limma.score.data %>%
  mutate(stability = log2(ratio) - abundance.score)
limma.score.data$stability[!limma.score.data$stability.score.valid] <- NA
limma.score.data$stability[is.na(limma.score.data$stability.score.valid)] <- NA
stability.score <- limma.score.data %>%
  dplyr::filter(stability.score.valid  & abundance.score.valid) %>%
  group_by(gene_name, SILAC, condition, rep) %>%
  summarise(stability.score = sum(stability, na.rm = TRUE))
score.count.table <- limma.score.data %>%
  dplyr::select(gene_name, condition, SILAC, rep, No.of.abundance.Ts, No.of.Ts.total) %>%
  unique()
limma.score.data <- left_join(score.count.table, abundance.score)
limma.score.data <- left_join(limma.score.data, stability.score)
# limma.score.data <- left_join(abundance.score, stability.score)
limma.score.data <- limma.score.data %>%
  mutate(ID = paste0(SILAC, "_", condition, "_", rep))
rm(abundance.score,stability.score)
# limma.score.data <- limma.score.data %>%
#   group_by(gene_name, SILAC, condition, rep) %>%
#   dplyr::select(gene_name, SILAC, condition, No.of.abundance.Ts, No.of.Ts.total,
#                 abundance.score.valid, stability.score.valid,
#                 rep, abundance.score, stability.score)
# limma.score.data <- unique(limma.score.data)
# limma.score.data <- limma.score.data %>%
#   mutate(ID = paste0(SILAC, "_", condition, "_", rep)) %>%
#   group_by(gene_name, ID, SILAC, condition, rep) %>%
#   dplyr::select(gene_name, ID, SILAC, condition, rep, No.of.abundance.Ts, No.of.Ts.total,
#                 abundance.score.valid, stability.score.valid,
#                 abundance.score, stability.score) %>%
#   filter(No.of.abundance.Ts >= 1, No.of.Ts.total >= 4)
ggplot(data = limma.score.data, aes(abundance.score)) +
  geom_density(aes(fill = SILAC)) +
  theme_bw(base_size = 12)
```




```
limma.score.data$abundance.score[limma.score.data$SILAC == "heavy"] <- scale(limma.score.data$abundance
limma.score.data$stability.score[limma.score.data$SILAC == "heavy"] <- scale(limma.score.data$stability

limma.score.data$abundance.score[limma.score.data$SILAC == "light"] <- scale(limma.score.data$abundance
limma.score.data$stability.score[limma.score.data$SILAC == "light"] <- scale(limma.score.data$stability
write.csv(limma.score.data,file.path(dir_save, paste0("raw_score_data_",script.version,".csv")))
```

##Correlation of rep1 to rep2 to rep3

```
rep.cor.data <- limma.score.data %>%
  dplyr::filter(abundance.score.valid, stability.score.valid)
rep.cor.data <- rep.cor.data %>%
  ungroup() %>%
  select(gene_name, ID, SILAC, abundance.score, stability.score) %>%
  unique()
rep.cor.data$rep <- gsub(".+_rep","rep",rep.cor.data$ID)
rep.cor.data$ID <- gsub("_rep.+","",rep.cor.data$ID)

crep.cor.data <- rep.cor.data %>%
  group_by(gene_name, SILAC, ID, rep) %>%
  gather(key = score.name, value = score, -gene_name, -ID, -rep, -SILAC) %>%
  group_by(gene_name, SILAC, score.name) %>%
  spread(key = rep, value = score) %>%
  mutate(time = gsub("(heavy_)|(light_)", "", ID))

ggplot(data = crep.cor.data, aes(rep1, rep2)) +
  geom_abline() +
```

```
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point(alpha = 0.1) +
  facet_grid(ID ~ score.name, scale = "free") +
  geom_text(data = subset(crep.cor.data, abs(rep2 - rep1) > 3), aes(label = gene_name), size = 2,check_
  geom_point(data = subset(crep.cor.data, abs(rep2 - rep1) > 3), size = 2, colour = "red") +
  geom_smooth(method = "loess") +
  theme_bw(base_size = 12)
ggsave(file.path(dir_save, paste0("Rep1_vs_rep2_score_correlation_", script.version, ".pdf")),width = 1

ggplot(data = crep.cor.data, aes(rep1, rep3)) +
  geom_abline() +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point(alpha = 0.1) +
  facet_grid(ID ~ score.name, scale = "free") +
  geom_text(data = subset(crep.cor.data, abs(rep3 - rep1) > 3), aes(label = gene_name), size = 2,check_
  geom_point(data = subset(crep.cor.data, abs(rep3 - rep1) > 3), size = 2, colour = "red") +
  geom_smooth(method = "loess") +
  theme_bw(base_size = 12)
ggsave(file.path(dir_save, paste0("Rep1_vs_rep3_score_correlation_", script.version, ".pdf")),width = 1

ggplot(data = crep.cor.data, aes(rep2, rep3)) +
  geom_abline() +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point(alpha = 0.1) +
  facet_grid(ID ~ score.name, scale = "free") +
  geom_text(data = subset(crep.cor.data, abs(rep3 - rep2) > 3), aes(label = gene_name), size = 2,check_
  geom_point(data = subset(crep.cor.data, abs(rep3 - rep2) > 3), size = 2, colour = "red") +
  geom_smooth(method = "loess") +
  theme_bw(base_size = 12)
ggsave(file.path(dir_save, paste0("Rep2_vs_rep3_score_correlation_", script.version, ".pdf")),width = 1

write.csv(crep.cor.data,file.path(dir_save, paste0("Data_replicate_score_correlation_",script.version,"
```

##Constructing expression sets for abundance and stability data

```
##abundance
abundance.data <- limma.score.data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, ID, abundance.score) %>%
  unique() %>%
  spread(ID, abundance.score)
abundance.weights <- limma.score.data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, ID, No.of.abundance.Ts) %>%
  unique() %>%
  spread(ID, No.of.abundance.Ts)
abundance.weights <- base::as.data.frame(abundance.weights)
rownames(abundance.weights) <- abundance.weights$gene_name
abundance.weights$gene_name <- NULL
abundance.weights <- as.matrix(abundance.weights)
abundance.data <- base::as.data.frame(abundance.data)
```

```
rownames(abundance.data) <- abundance.data$gene_name
abundance.data$gene_name <- NULL
abundance.metadata <- data.frame(ID = names(abundance.data))
abundance.metadata <- left_join(abundance.metadata,
                                unique(limma.score.data[, c("ID", "SILAC", "condition", "rep")]))
abundance.metadata$condition <- paste(abundance.metadata$SILAC, abundance.metadata$condition, sep = "_")
rownames(abundance.metadata) <- abundance.metadata$ID
colnames(abundance.data) <- abundance.metadata$ID
abundance.data <- as.matrix(abundance.data)
abundance.dataE <-  ExpressionSet(assayData = abundance.data,
                                  phenoData = AnnotatedDataFrame(abundance.metadata))
validObject(abundance.dataE)
```

```
## [1] TRUE
```

```
rm(abundance.data, abundance.metadata)

##stability
stability.data <- limma.score.data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, ID, stability.score) %>%
  unique() %>%
  spread(ID, stability.score)
stability.weights <- limma.score.data %>%
  group_by(gene_name) %>%
  dplyr::select(gene_name, ID, No.of.Ts.total) %>%
  unique() %>%
  spread(ID, No.of.Ts.total)
stability.weights <- base::as.data.frame(stability.weights)
rownames(stability.weights) <- stability.weights$gene_name
stability.weights$gene_name <- NULL
stability.data <- base::as.data.frame(stability.data)
rownames(stability.data) <- stability.data$gene_name
stability.data$gene_name <- NULL
stability.metadata <- data.frame(ID = names(stability.data))
stability.metadata <- left_join(stability.metadata,
                                unique(limma.score.data[, c("ID", "SILAC", "condition", "rep")]))
stability.metadata$condition <- paste(stability.metadata$SILAC, stability.metadata$condition, sep = "_")
rownames(stability.metadata) <- stability.metadata$ID
colnames(stability.data) <- stability.metadata$ID
stability.data <- as.matrix(stability.data)
stability.dataE <-  ExpressionSet(assayData = stability.data,
                                  phenoData = AnnotatedDataFrame(stability.metadata))
validObject(stability.dataE)
```

```
## [1] TRUE
```

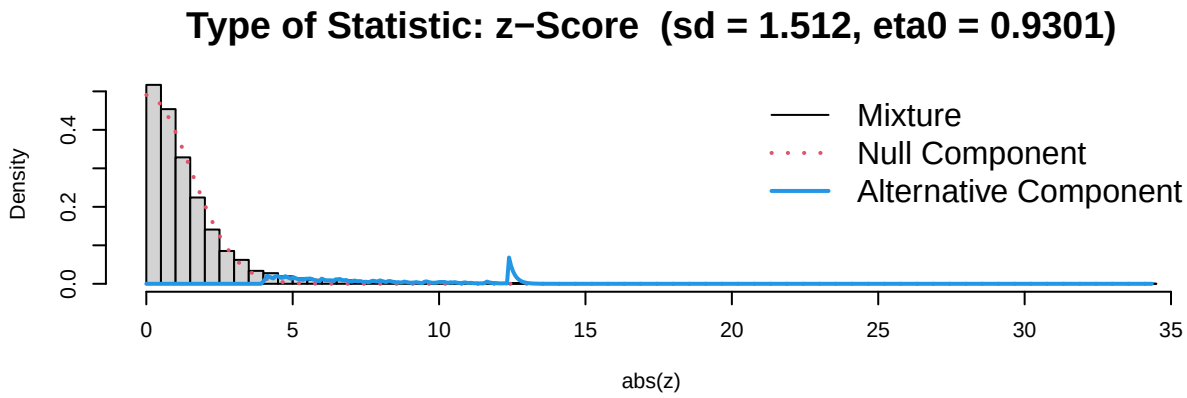
```
rm(stability.data, stability.metadata)
```

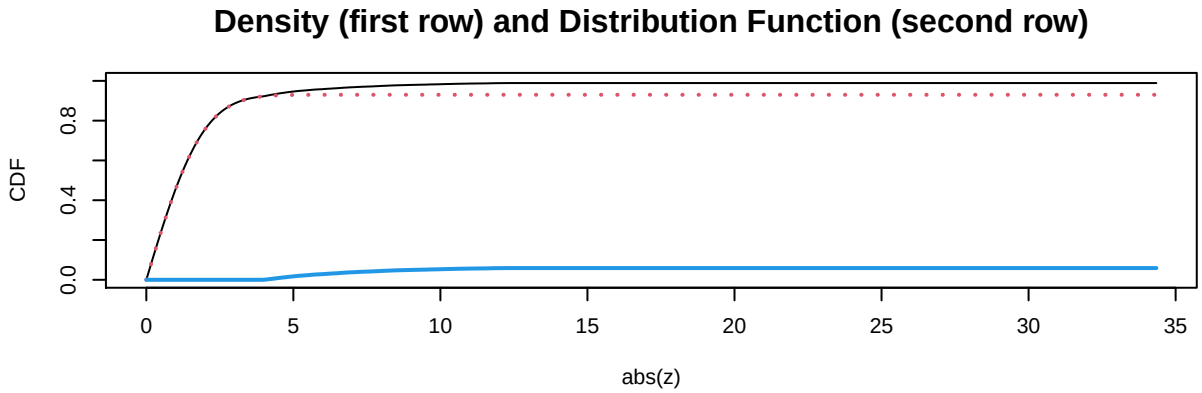
##Limma evaluation of stability and abundance scores

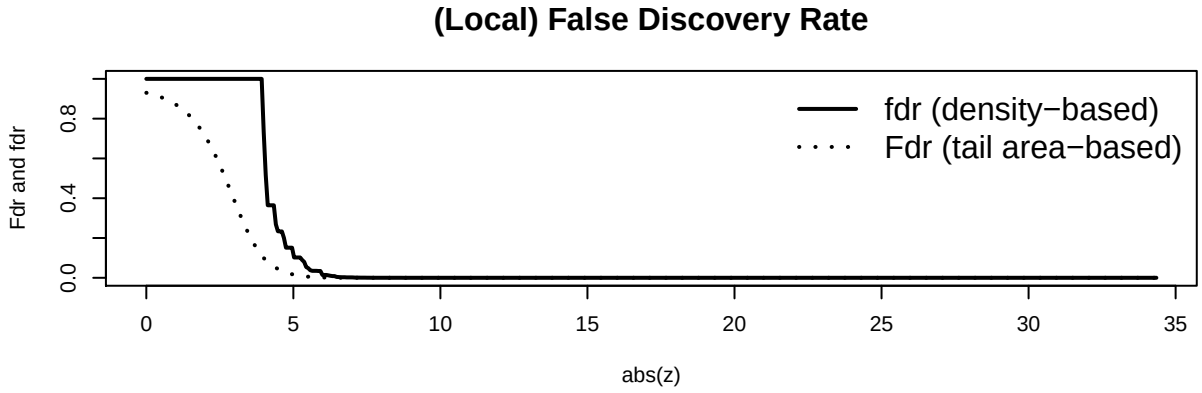
```
limma_results <- NULL
```

##Construction of design matrix and defining conditions that will be compared

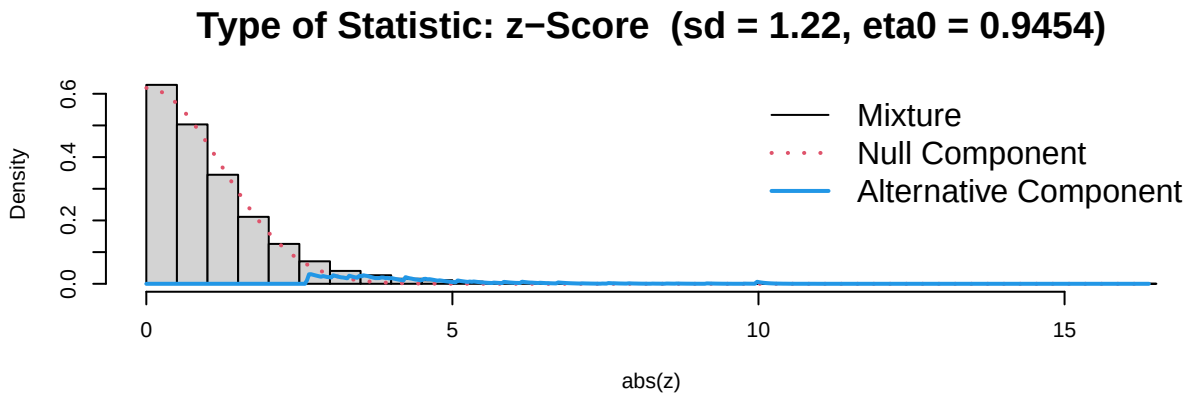
```
for (to_condition in unique(gsub("_rep.+","", limma.score.data$ID)))
{
  limma_data = abundance.dataE
  limma_data <- limma_data[, limma_data$condition == to_condition]
  condition <- factor(pData(limma_data)$condition, ordered = FALSE)
  replicate <- factor(pData(limma_data)$rep, ordered = FALSE)
  abundance.weights.sub <- abundance.weights[, grepl(paste0("^", to_condition, "_rep"),
                                                     colnames(abundance.weights))]
  mut_comparison <- eBayes(lmFit(limma_data, weights = abundance.weights.sub^2))
  res <- limma::topTable(mut_comparison, sort.by = "t",
                         coef = 1, number = Inf)
  res$gene_name <- rownames(res)
  names(res)[grep("P.Value", names(res))] <- "pvalue.limma"
  names(res)[grep("adj.P.Val", names(res))] <- "fdr.limma"
  res <- subset(res,!is.na(logFC))
  res$pvalue.fdrtool <- NA
  res$qval.fdrtool <- NA
  res$lfdr.fdrtool <- NA
  res$comparison <- paste0("abundance - ", to_condition)
  res$comparison2 <- to_condition
  res$score <- "abundance"
  limma_results <- rbind(limma_results, res)

  limma_data = stability.dataE
  limma_data <- limma_data[, limma_data$condition == to_condition]
  condition <- factor(pData(limma_data)$condition, ordered = FALSE)
  replicate <- factor(pData(limma_data)$rep, ordered = FALSE)
  condition_d <- model.matrix(~1)
  stability.weights.sub <- stability.weights[, grepl(paste0("^", to_condition,"_rep"),
                                                     colnames(stability.weights))]
  mut_comparison <- eBayes(lmFit(limma_data, weights = stability.weights.sub^2))
  res <- limma::topTable(mut_comparison, sort.by = "t",
                         coef = 1, number = Inf)
  res$gene_name <- rownames(res)
  names(res)[grep("P.Value", names(res))] <- "pvalue.limma"
  names(res)[grep("adj.P.Val", names(res))] <- "fdr.limma"
  res <- subset(res,!is.na(logFC))
  res$pvalue.fdrtool <- NA
  res$qval.fdrtool <- NA
  res$lfdr.fdrtool <- NA
  res$comparison <- paste0("stability - ", to_condition)
  res$comparison2 <- to_condition
  res$score <- "stability"
  limma_results <- rbind(limma_results, res)
}
rm(to_condition)

fdr_res_abundance <- NULL
try(fdr_res_abundance <- fdrtool(limma_results$t[limma_results$score == "abundance"], plot = TRUE, verbo
```

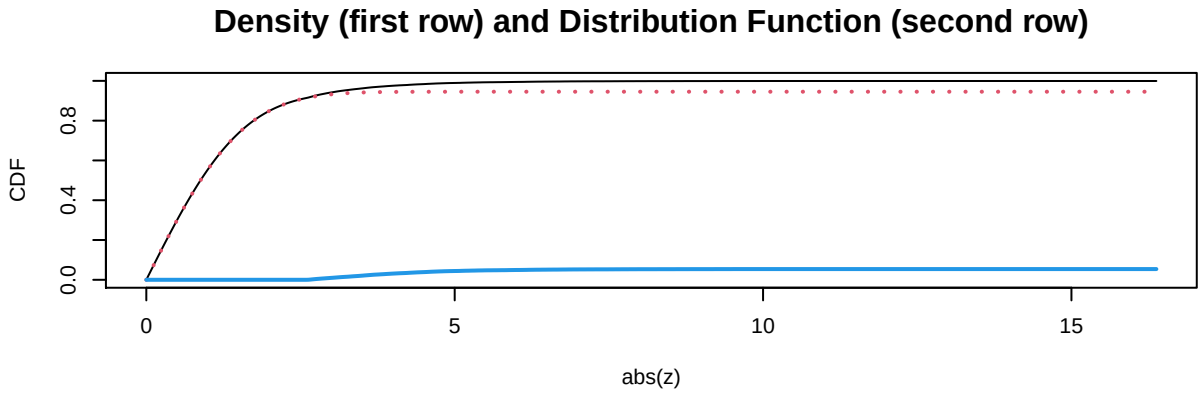


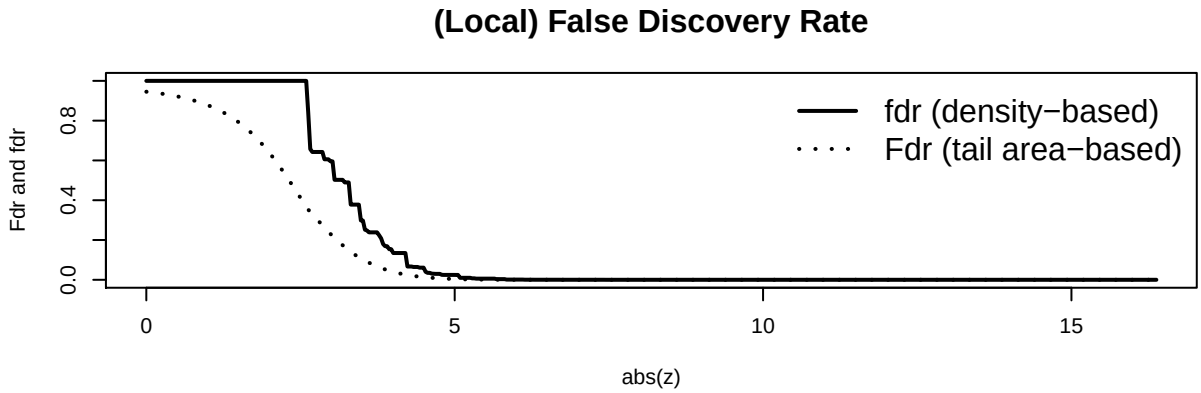




```
if (length(fdr_res_abundance) > 0)
{
  limma_results$pvalue.fdrtool[limma_results$score == "abundance"] <- fdr_res_abundance$pval
  limma_results$qval.fdrtool[limma_results$score == "abundance"] <- fdr_res_abundance$qval
  limma_results$lfdr.fdrtool[limma_results$score == "abundance"] <- fdr_res_abundance$lfdr
}
rm(fdr_res_abundance)
fdr_res_stability <- NULL
try(fdr_res_stability <- fdrtool(limma_results$t[limma_results$score == "stability"], plot = TRUE, verbo
```



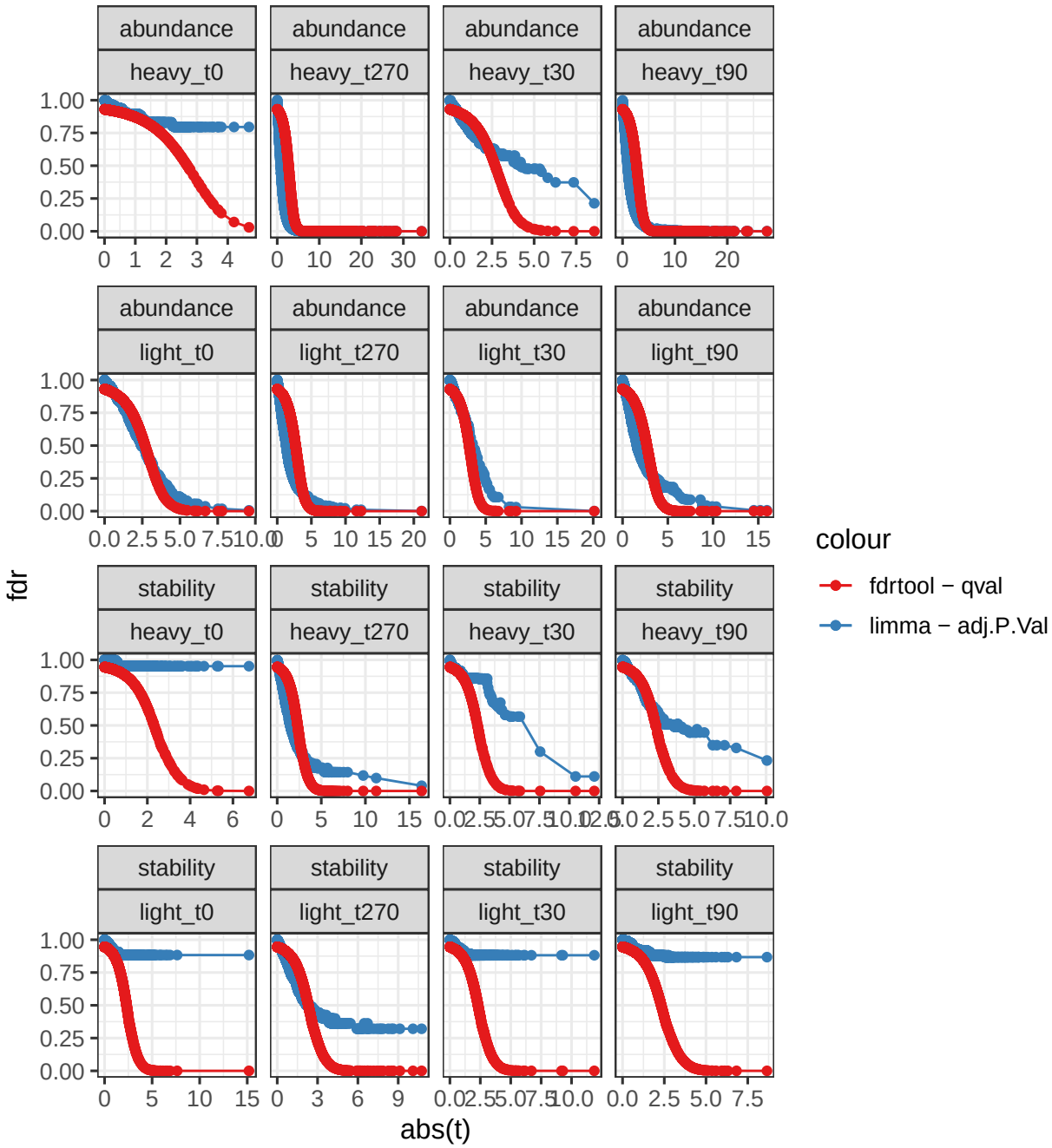




```
if (length(fdr_res_stability) > 0)
{
  limma_results$pvalue.fdrtool[limma_results$score == "stability"] <- fdr_res_stability$pval
  limma_results$qval.fdrtool[limma_results$score == "stability"] <- fdr_res_stability$qval
  limma_results$lfdr.fdrtool[limma_results$score == "stability"] <- fdr_res_stability$lfdr
}
rm(fdr_res_stability)
```

##Limma and fdrtool statistics comparison

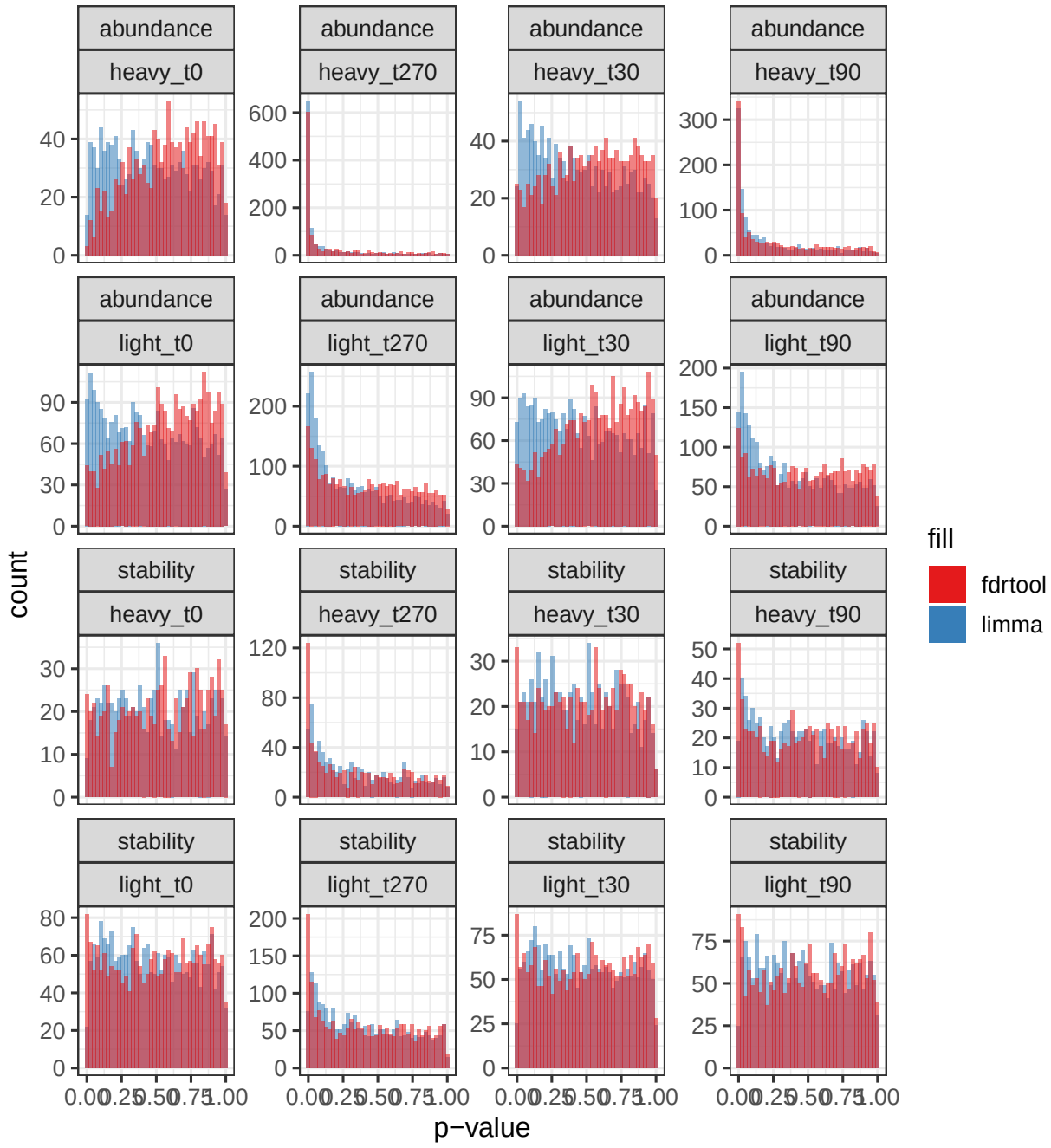
```
ggplot(data = limma_results, aes(abs(t))) +
  geom_line(aes(y = fdr.limma, colour = "limma - adj.P.Val")) +
```

```
  geom_line(aes(y = qval.fdrtool, colour = "fdrtool - qval")) +
  geom_point(aes(y = fdr.limma, colour = "limma - adj.P.Val")) +
  geom_point(aes(y = qval.fdrtool, colour = "fdrtool - qval")) +
  facet_wrap(score ~ comparison2, scale = "free_x") + ylab("fdr") +
  theme_bw(base_size = 12) +
  scale_colour_brewer(palette = "Set1")
```




```
ggsave(file.path(dir_save, paste0("t_vs_fdr_limma_vs_fdrtool_", script.version,".pdf")), width = 15, he

ggplot(data = limma_results) +
  geom_histogram(aes(pvalue.limma, alpha = 0.5, fill = "limma"), bins = 40) +
```

```
  facet_wrap(score ~ comparison2, scale = "free_y") +
  geom_histogram(aes(pvalue.fdrtool, alpha = 0.5, fill = "fdrtool"), bins = 40) +
  guides(alpha = FALSE) +
  xlab("p-value") +
  theme_bw(base_size = 12) +
  scale_fill_brewer(palette = "Set1")
```




```
ggsave(file.path(dir_save, paste0("p-value_histogram_limma_vs_fdrtool_", script.version,".pdf")), width
```

##Hit annotation

```
limma_results$score.value <- limma_results$logFC

limma_results$hit_annotation_method <- NA
fdr_hit_threshold <- 0.01
fdr_candidate_threshold = 0.05
score_hit_threshold <- 3
score_candidate_threshold <- 2
limma_results$pvalue <- NA
limma_results$fdr <- NA
for (comp in unique(limma_results$comparison))
{
  limma_hits <-
    nrow(limma_results[limma_results$comparison == comp &
                         limma_results$fdr.limma <= fdr_hit_threshold,])
  fdrtool_hits <-
    nrow(limma_results[limma_results$comparison == comp &
                         limma_results$qval.fdrtool <= fdr_hit_threshold,])
  if (limma_hits >= fdrtool_hits)
  {
    limma_results$hit_annotation_method[limma_results$comparison == comp] <- "limma"
  }
  if (fdrtool_hits > limma_hits)
  {
    limma_results$hit_annotation_method[limma_results$comparison == comp] <- "fdrtool"
  }
  rm(limma_hits,fdrtool_hits)
}
table(limma_results$hit_annotation_method)
```

```
##
## fdrtool   limma
##   26043    2464
```

```
limma_results$hit_annotation_method <- "fdrtool"

limma_results$pvalue[limma_results$hit_annotation_method == "limma"] <-
  limma_results$pvalue.limma[limma_results$hit_annotation_method == "limma"]
limma_results$fdr[limma_results$hit_annotation_method == "limma"] <-
  limma_results$fdr.limma[limma_results$hit_annotation_method == "limma"]

limma_results$pvalue[limma_results$hit_annotation_method == "fdrtool"] <-
  limma_results$pvalue.fdrtool[limma_results$hit_annotation_method == "fdrtool"]
limma_results$fdr[limma_results$hit_annotation_method == "fdrtool"] <-
  limma_results$qval.fdrtool[limma_results$hit_annotation_method == "fdrtool"]

limma_results$hit <- with(limma_results,
                          ifelse(fdr <= fdr_hit_threshold & abs(score.value) >=
                                   score_hit_threshold,T,F))
limma_results$hit_annotation <- with(limma_results,
                                     ifelse(fdr <= fdr_hit_threshold & abs(score.value) >=
                                              score_hit_threshold,"hit",
                                            ifelse(fdr <= fdr_candidate_threshold & abs(score.value) >=
                                                     score_candidate_threshold,
                                                   "candidate","no hit")))
```

```
limma_results$hit_annotation <- factor(limma_results$hit_annotation,ordered =  TRUE,
                                       levels = c("hit","candidate","no hit"))

with(limma_results,table(comparison,hit_annotation))
```

```
##                          hit_annotation
## comparison                 hit candidate no hit
##   abundance - heavy_t0       0         0   1232
##   abundance - heavy_t270    88       117   1027
##   abundance - heavy_t30      1         0   1231
##   abundance - heavy_t90      5        37   1190
##   abundance - light_t0       2        16   2765
##   abundance - light_t270    23        32   2728
##   abundance - light_t30      4        13   2766
##   abundance - light_t90      8        15   2760
##   stability - heavy_t0       1         4    830
##   stability - heavy_t270     3         9    824
##   stability - heavy_t30      0         6    830
##   stability - heavy_t90      1         2    833
##   stability - light_t0       1         9   2266
##   stability - light_t270     4        23   2249
##   stability - light_t30      1         3   2272
##   stability - light_t90      1         4   2271
```

##Save limma results

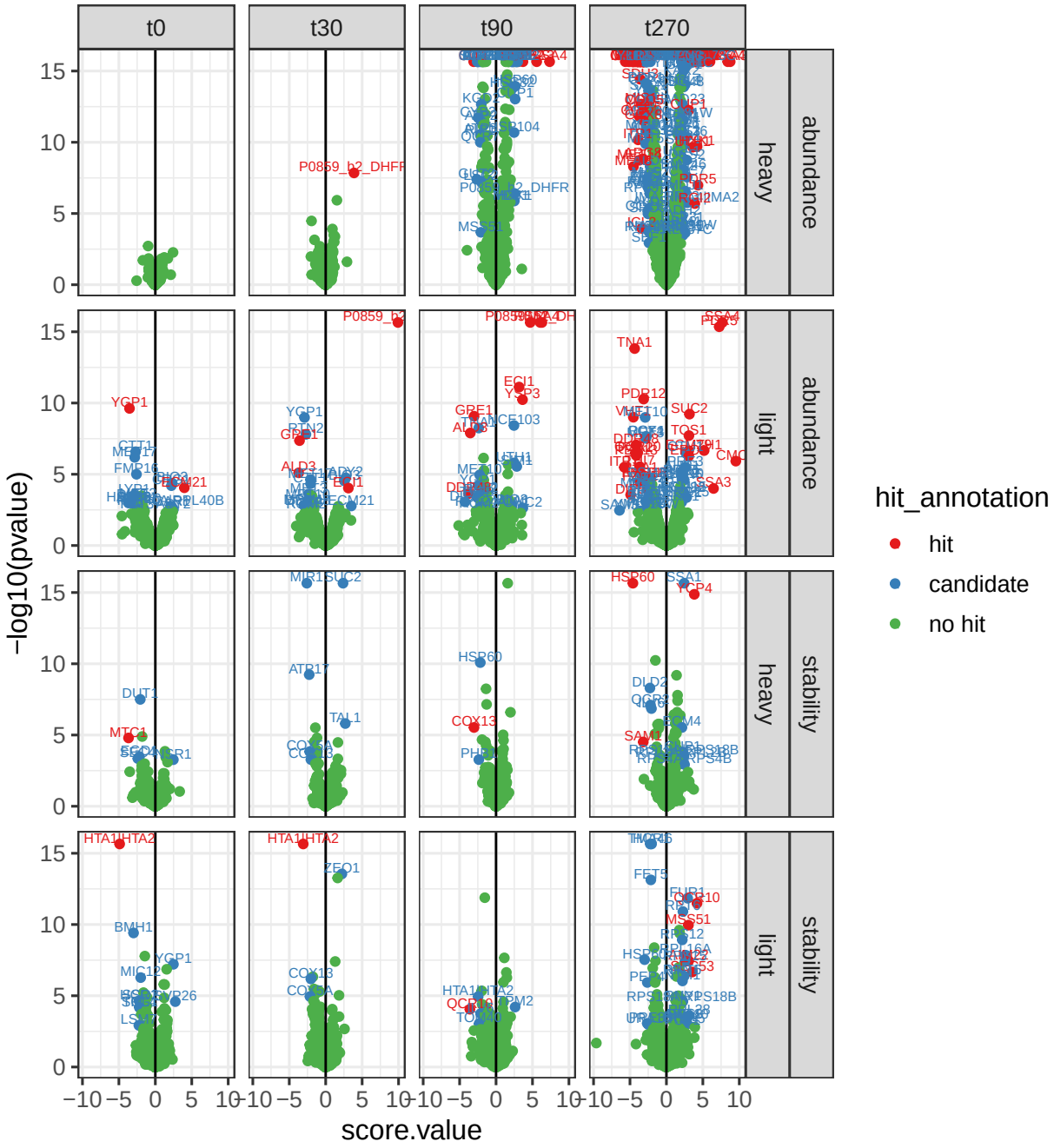
```
write.csv(limma_results, file.path(dir_save, paste0("Limma_results_", script.version, ".csv")), row.nam
```

##Merge limma significances with raw score data

```
limma.cp.data <- limma.score.data %>%
  group_by(gene_name, ID, SILAC, condition, rep) %>%
  gather(key = score, value = score.value, -gene_name, -ID, -No.of.abundance.Ts,
         -No.of.Ts.total,
         -condition, -rep, -SILAC)
limma.sub <- limma_results[, c("gene_name", "fdr", "comparison2", "score")] %>%
  unique()
limma.sub$score <- paste0(limma.sub$score, ".score")
limma.sub$condition <- limma.sub$comparison2
limma.sub$comparison2 <- NULL
limma.sub$SILAC <- gsub("_.+", "", limma.sub$condition)
limma.sub$condition <- gsub(".+_", "", limma.sub$condition)
limma.cp.data <- left_join(limma.cp.data, limma.sub)
limma.cp.data$fdr.star <-
  as.character(cut(as.numeric(as.character(limma.cp.data$fdr)),
                   breaks = c(0,0.001,0.01,0.05,1),
                   include.lowest = TRUE))
limma.cp.data$fdr.star[is.na(limma.cp.data$fdr.star)] <- ""
limma.cp.data$fdr.star <- recode(limma.cp.data$fdr.star,
                                 "(0.05,1]" = "",
                                   "(0.01,0.05]" = "*",
                                   "(0.001,0.01]" = "**",
                                   "[0,0.001]" = "***")
```

##Volcano-Plot

```
limma_results$sample <- gsub(" - .+", "", limma_results$comparison)
limma_results$comparison <- gsub(".+ - ", "", limma_results$comparison)
limma_results$time <- gsub(".+_", "", limma_results$comparison)
limma_results$time <- factor(limma_results$time, ordered = T, levels = c("t0", "t30", "t90", "t270"))
limma_results$SILAC <- gsub("_.+", "", limma_results$comparison)
qplot(score.value, -log10(pvalue), data = limma_results, colour = hit_annotation) +
  geom_vline(aes(xintercept = 0)) +
  facet_grid(sample + SILAC ~ time) +
  geom_text(aes(label = gene_name),
            data = subset(limma_results, hit_annotation != "no hit"),
            vjust = 0, nudge_y = 0.1, size = 2, check_overlap = FALSE) +
  theme_bw(base_size = 12) +
  scale_colour_brewer(palette = "Set1")
```




```
ggsave(file.path(dir_save, paste0("Volcano_plot_", script.version, ".pdf")), width = 15, height = 15)


pdf(file.path(dir_save, paste0("Volcano_plot_single_pages_", script.version, ".pdf")), width = 12, heigh
for (comp in unique(limma_results$comparison2))
{
  print(comp)
  sub <- subset(limma_results, comparison2 == comp)
  print(qplot(score.value, -log10(pvalue), data = sub, colour = hit_annotation) +
  geom_vline(aes(xintercept = 0)) +
  facet_grid(comparison2 ~ score + hit_annotation_method) +
```

```
  geom_text(aes(label = gene_name),
            data = subset(sub, hit_annotation != "no hit"),
            vjust = 0, nudge_y = 0.1, size = 2, check_overlap = FALSE) +
  theme_bw(base_size = 12) +
  scale_colour_brewer(palette = "Set1"))
}
```

```
## [1] "light_t0"
```

```
## [1] "heavy_t0"
```

```
## [1] "light_t30"
```

```
## [1] "heavy_t30"
```

```
## [1] "light_t90"
```

```
## [1] "heavy_t90"
```

```
## [1] "light_t270"
```

```
## [1] "heavy_t270"
```

```
dev.off()
```

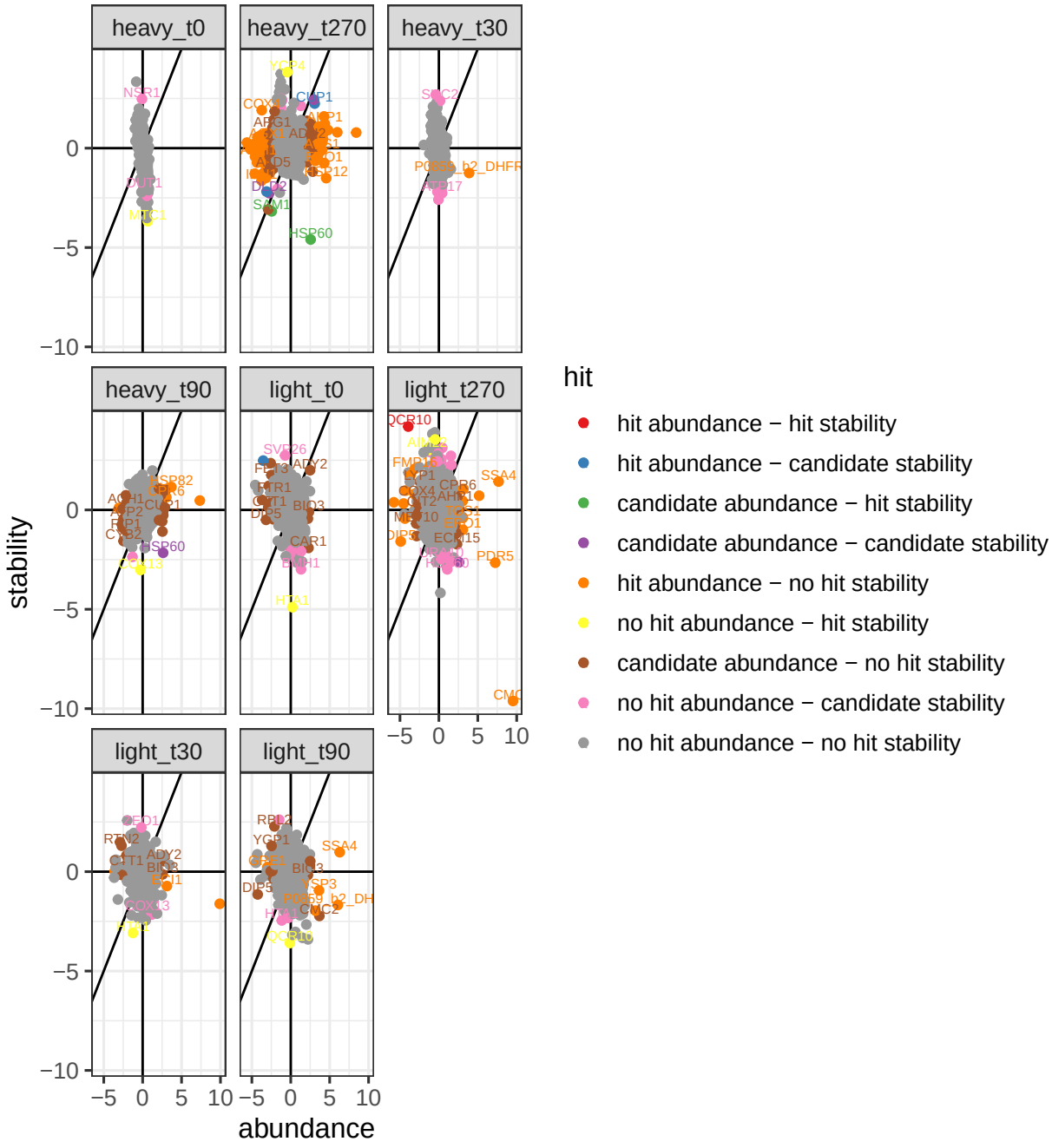
```
## pdf
##   2
```

```
rm(sub)
```

##Correlation of limma results - abundance vs stability

```
cor.data <- limma_results %>%
  group_by(gene_name, comparison2) %>%
  dplyr::select(gene_name, comparison2, score.value, score) %>%
  spread(key = score, value = score.value)
cor.data_i <- limma_results %>%
  group_by(gene_name, comparison2) %>%
  dplyr::select(gene_name, comparison2, hit_annotation, score) %>%
  mutate(hit_annotation = gsub("enriched ", "", hit_annotation)) %>%
  spread(key = score, value = hit_annotation)
names(cor.data_i)[-c(1, 2)] <- paste0("hit_", names(cor.data_i)[-c(1, 2)])
cor.data <- left_join(cor.data, cor.data_i)
rm(cor.data_i)
cor.data$comparison <- cor.data$comparison2
cor.data$comparison2 <- NULL
cor.data <- cor.data %>%
  group_by(gene_name, comparison)
cor.data <- na.omit(cor.data)
cor.data$hit <- paste(cor.data$hit_abundance, "abundance -", cor.data$hit_stability, "stability")
cor.data$hit <- factor(cor.data$hit, ordered = TRUE,
                       levels = c("hit abundance - hit stability",
                                  "hit abundance - candidate stability",
                                  "candidate abundance - hit stability",
                                  "candidate abundance - candidate stability",
                                  "hit abundance - no hit stability",
                                  "no hit abundance - hit stability",
                                  "candidate abundance - no hit stability",
                                  "no hit abundance - candidate stability",
```

```
                                    "no hit abundance - no hit stability"))
no.of.comparisons <- length(unique(cor.data$comparison))
gr.width <- 6 + ceiling(sqrt(no.of.comparisons)) * 7
gr.height <- floor(sqrt(no.of.comparisons)) * 7

ggplot(data = cor.data, aes(abundance, stability, colour = hit)) +
  geom_abline() +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point() +
  facet_wrap(~ comparison) +
  geom_text(aes(label = gsub("[|].+", "", gene_name)),
            data = subset(cor.data, hit != "no hit abundance - no hit stability"),
            vjust = 0, nudge_y = 0.1, size = 2, check_overlap = TRUE) +
  customPlot
```




```
ggsave(file.path(dir_save, paste0("Fold_change_correlation_abundance_vs_stability_", script.version, ".p

pdf(file.path(dir_save, paste0("Fold_change_correlation_abundance_vs_stability_single_pages_", script.ve
for(comp in unique(cor.data$comparison))
{
  print(comp)
  sub <- subset(cor.data, comparison == comp)
  print(ggplot(data = sub, aes(abundance, stability, colour = hit)) +
          geom_abline() +
          geom_vline(xintercept = 0) +
          geom_hline(yintercept = 0) +
```

```
          geom_point() +
          facet_wrap(~ comparison) +
          geom_text(aes(label = gsub("[|].+", "", gene_name)),
                    data = subset(sub, hit != "no hit abundance - no hit stability"),
                    vjust = 0, nudge_y = 0.1, size = 2, check_overlap = TRUE) +
          customPlot)
}
```

```
## [1] "heavy_t0"
## [1] "heavy_t270"
## [1] "heavy_t30"
## [1] "heavy_t90"
## [1] "light_t0"
## [1] "light_t270"
## [1] "light_t30"
## [1] "light_t90"
```

```
dev.off()
```

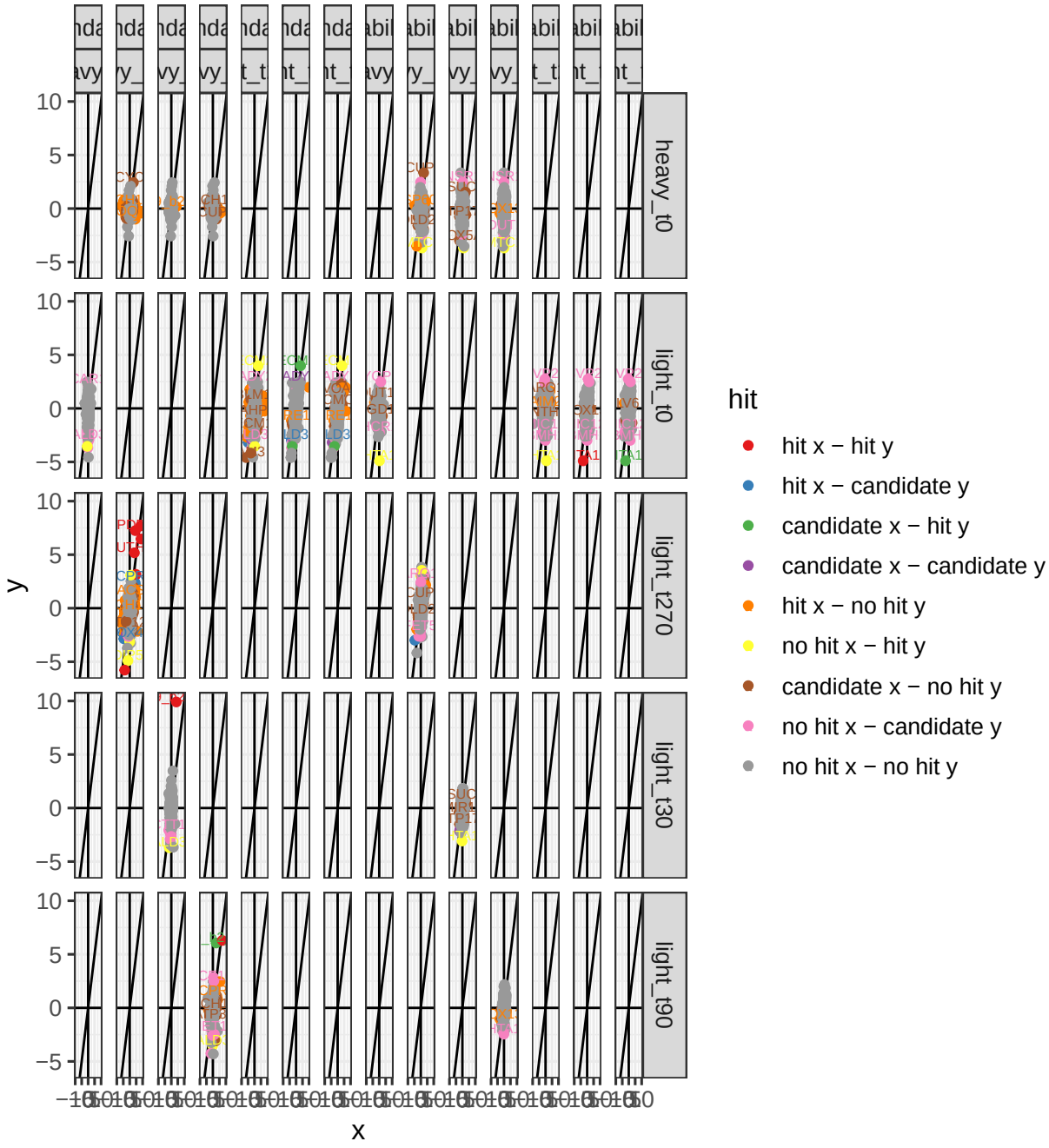
```
## pdf
##   2
```

```
rm(sub, comp)

write.csv(cor.data, file = file.path(dir_save, paste0("Fold_change_correlation_data_abundance_vs_stabili
```

##Correlation of limma results

```
cor.data <- limma_results %>%
  group_by(gene_name, score) %>%
  dplyr::select(gene_name, comparison2, score.value, score) %>%
  spread(key = comparison2, value = score.value)
cor.data_i <- limma_results %>%
  group_by(gene_name, score) %>%
  dplyr::select(gene_name, comparison2, hit_annotation, score) %>%
  mutate(hit_annotation = gsub("enriched ", "", hit_annotation)) %>%
  spread(key = comparison2, value = hit_annotation)
names(cor.data_i)[-c(1, 2)] <- paste0("hit_", names(cor.data_i)[-c(1, 2)])
cor.data <- left_join(cor.data, cor.data_i)
cor.data <- as.data.frame(cor.data)
rm(cor.data_i)
conds <- gsub("hit_", "", names(cor.data)[grep("hit_", names(cor.data))])
comparisons <- data.frame(x = c("heavy_t0", "heavy_t30", "heavy_t90", "heavy_t270", "light_t30", "light_
                          y = c("light_t0", "light_t30", "light_t90", "light_t270", "light_t0", "light_
mcor.data <- NULL
for (i in seq_along(comparisons$x))
{
  mcor.data_i <- data.frame(gene_name = cor.data$gene_name,
                            score = cor.data$score,
                            x = cor.data[, as.character(comparisons$x[i])],
                            y = cor.data[, as.character(comparisons$y[i])],
                            hit_x = cor.data[, paste0("hit_", comparisons$x[i])],
```

```
                            hit_y = cor.data[, paste0("hit_", comparisons$y[i])],
                            x.name =  as.character(comparisons$x[i]),
                            y.name =  as.character(comparisons$y[i]),
                            comparison = paste(comparisons$x[i], "vs", comparisons$y[i]))
  mcor.data <- bind_rows(mcor.data, mcor.data_i)
  rm(mcor.data_i)
}
cor.data <- mcor.data
rm(mcor.data)
cor.data <- na.omit(cor.data)
cor.data$hit <- paste(cor.data$hit_x, "x -", cor.data$hit_y, "y")
cor.data$hit <- factor(cor.data$hit, ordered = TRUE,
                       levels = c("hit x - hit y",
                                  "hit x - candidate y",
                                  "candidate x - hit y",
                                  "candidate x - candidate y",
                                  "hit x - no hit y",
                                  "no hit x - hit y",
                                  "candidate x - no hit y",
                                  "no hit x - candidate y",
                                  "no hit x - no hit y"))
no.of.comparisons <- length(unique(paste(cor.data$score, cor.data$comparison)))
gr.width <- 6 + ceiling(sqrt(no.of.comparisons)) * 7
gr.height <- floor(sqrt(no.of.comparisons)) * 7

ggplot(data = cor.data, aes(x, y, colour = hit)) +
  geom_abline() +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0) +
  geom_point() +
  facet_grid(y.name ~ score + x.name) +
  geom_text(aes(label = gsub("[|].+", "", gene_name)),
            data = subset(cor.data, hit != "no hit x - no hit y"),
            vjust = 0, nudge_y = 0.1, size = 2, check_overlap = TRUE) +
  customPlot
```




```
ggsave(file.path(dir_save, paste0("Fold_change_correlation_", script.version, ".pdf")), width = gr.width

pdf(file.path(dir_save, paste0("Fold_change_correlation_single_pages_", script.version, ".pdf")), width
for(comp in unique(cor.data$score))
{
  print(comp)
  sub <- subset(cor.data, score == comp)
  print(ggplot(data = sub, aes(x, y, colour = hit)) +
          geom_abline() +
          geom_vline(xintercept = 0) +
          geom_hline(yintercept = 0) +
```

```
          geom_point() +
          facet_grid(y.name ~ score + x.name) +
          geom_text(aes(label = gsub("[|].+", "", gene_name)),
                    data = subset(sub, hit != "no hit x - no hit y"),
                    vjust = 0, nudge_y = 0.1, size = 2, check_overlap = TRUE) +
          customPlot)
}
```

```
## [1] "abundance"
```

```
## [1] "stability"
```

```
dev.off()
```

```
## pdf
##   2
```

```
rm(sub, comp)

write.csv(cor.data, file = file.path(dir_save, paste0("Fold_change_correlation_data_", script.version,
```

#Create Excel-Sheet for data browsing

```
data.sum <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio") %>%
  mutate(condition = paste(SILAC, condition, sep = "_")) %>%
  group_by(gene_name, condition, Temperature) %>%
  summarise(median.ratio = mean(value, na.rm = TRUE)) %>%
  spread(key = condition, value = median.ratio)
data.sum_i <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio") %>%
  mutate(condition = paste(SILAC, condition, sep = "_")) %>%
  group_by(gene_name, condition, Temperature) %>%
  mutate(log2.value = log2(value)) %>%
  summarise(log2.range = diff(range(log2.value, na.rm = TRUE))) %>%
  group_by(gene_name, Temperature) %>%
  mutate(condition = paste0("log2.range_", condition)) %>%
  spread(key = condition, value = log2.range)
data.sum <- left_join(data.sum, data.sum_i)
data.sum_i <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio", !is.na(value)) %>%
  group_by(gene_name, SILAC, Temperature) %>%
  dplyr::select(gene_name, SILAC, Temperature, rep) %>%
  unique() %>%
  na.omit() %>%
  summarise(found.in.replicates = n()) %>%
  ungroup() %>%
  spread(key = SILAC, value = found.in.replicates)
names(data.sum_i)[c(3,4)] <- paste0("found.in.replicates_", names(data.sum_i)[c(3,4)])
data.sum <- right_join(data.sum_i, data.sum)
data.sum$found.in.replicates_heavy[is.na(data.sum$found.in.replicates_heavy)] <- 0
data.sum$found.in.replicates_light[is.na(data.sum$found.in.replicates_light)] <- 0
data.sum_i <- mdata %>%
  dplyr::filter(measurement == "ctrl.ratio", !is.na(value)) %>%
  group_by(gene_name, SILAC) %>%
  dplyr::select(gene_name, SILAC, Temperature) %>%
```

```
  unique() %>%
  na.omit() %>%
  summarise(found.in.Temperatures = n()) %>%
  spread(key = SILAC, value = found.in.Temperatures)
names(data.sum_i)[c(2,3)] <- paste0("found.in.Temperatures_", names(data.sum_i)[c(2,3)])
data.sum <- right_join(data.sum_i, data.sum)
feature_data <- fData(raw_dataE)
data.sum <- right_join(feature_data[, c("gene_name", "protein_id", "description", "found.in.ms.runs")],
                       data.sum)
data.sum_i <- feature_data %>%
  dplyr::select(gene_name, starts_with("max.qupm_")) %>%
  group_by(gene_name) %>%
  gather(key = max.qupm.tag, value = max.qupm, -gene_name) %>%
  mutate(Temperature = gsub(".+_", "", max.qupm.tag)) %>%
  mutate(Temperature = as.numeric(as.character(Temperature))) %>%
  dplyr::select(gene_name, Temperature, max.qupm)
data.sum <- left_join(data.sum, data.sum_i)
data.sum_i <- feature_data %>%
  dplyr::select(gene_name, starts_with("average.top3_")) %>%
  group_by(gene_name) %>%
  gather(key = average.top3.tag, value = average.top3, -gene_name) %>%
  mutate(Temperature = gsub(".+_", "", average.top3.tag)) %>%
  mutate(Temperature = as.numeric(as.character(Temperature))) %>%
  dplyr::select(gene_name, Temperature, average.top3)
data.sum <- left_join(data.sum, data.sum_i)
data.sum$path <- file.path(".","plots_combined",
                       paste0(gsub("\\|","-" ,data.sum$gene_name),".pdf"))
write.csv(data.sum, file.path(dir_save, paste0("Final_results_", script.version, ".csv")),
          row.names = FALSE)

hit_classifications <- limma_results[,c("gene_name", "comparison2", "score", "hit_annotation")]
hit_classifications <- hit_classifications %>%
  group_by(gene_name) %>%
  mutate(key = paste0(score, "_", comparison2)) %>%
  dplyr::select(gene_name, key, hit_annotation) %>%
  spread(key = key, value = hit_annotation)
data.sum <- left_join(data.sum, hit_classifications)
write.csv(data.sum, file.path(dir_save, paste0("Final_results_classified_", script.version, ".csv")),
          row.names = FALSE)

limma_spread <- limma_results[, c("gene_name", "comparison2", "score", "score.value", "fdr", "hit_annota
  mutate(comparison = comparison2) %>%
  dplyr::select(-comparison2) %>%
  group_by(gene_name, comparison, score) %>%
  gather(key = key, value = value, -gene_name, -comparison, -score) %>%
  mutate(key = gsub(".value", "", key)) %>%
  mutate(new.key = paste0(key, "_", comparison)) %>%
  group_by(gene_name, score) %>%
  dplyr::select(gene_name, score, new.key, value) %>%
  spread(key = new.key, value = value)
limma_spread <- limma_spread[, c("gene_name", "score",
                                 grep("^score_", names(limma_spread), value = TRUE),
                                 grep("^fdr_", names(limma_spread), value = TRUE),
```

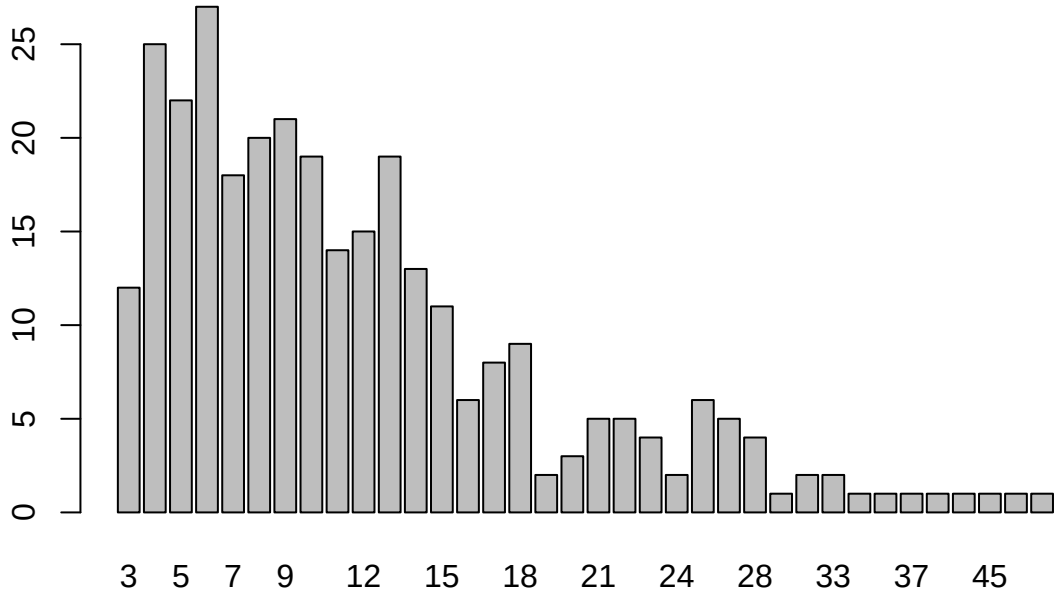
```
                                    grep("^hit_annotation_", names(limma_spread), value = TRUE))]
limma_spread <- right_join(
  unique(data.sum[, c("gene_name", "protein_id", "description", "found.in.ms.runs", "path")]),
  limma_spread)

write.csv(limma_spread, file.path(dir_save, paste0("Limma_results_classified_", script.version, ".csv")),
          row.names = FALSE)
```

#Clustering of data

##Preparation of cluster data

```
# stability hits
dir.hits <- "data_analysis_results_V1/stab_hits_categories/"
```

```
hits <- unique(subset(limma_results, hit_annotation %in% c("hit", "candidate"))$gene_name)
# hits <- unique(subset(limma_results, hit_annotation %in% c("hit", "candidate") & score == "stability")
barplot(table(cdata$max.qupm[cdata$gene_name %in% hits]))
```

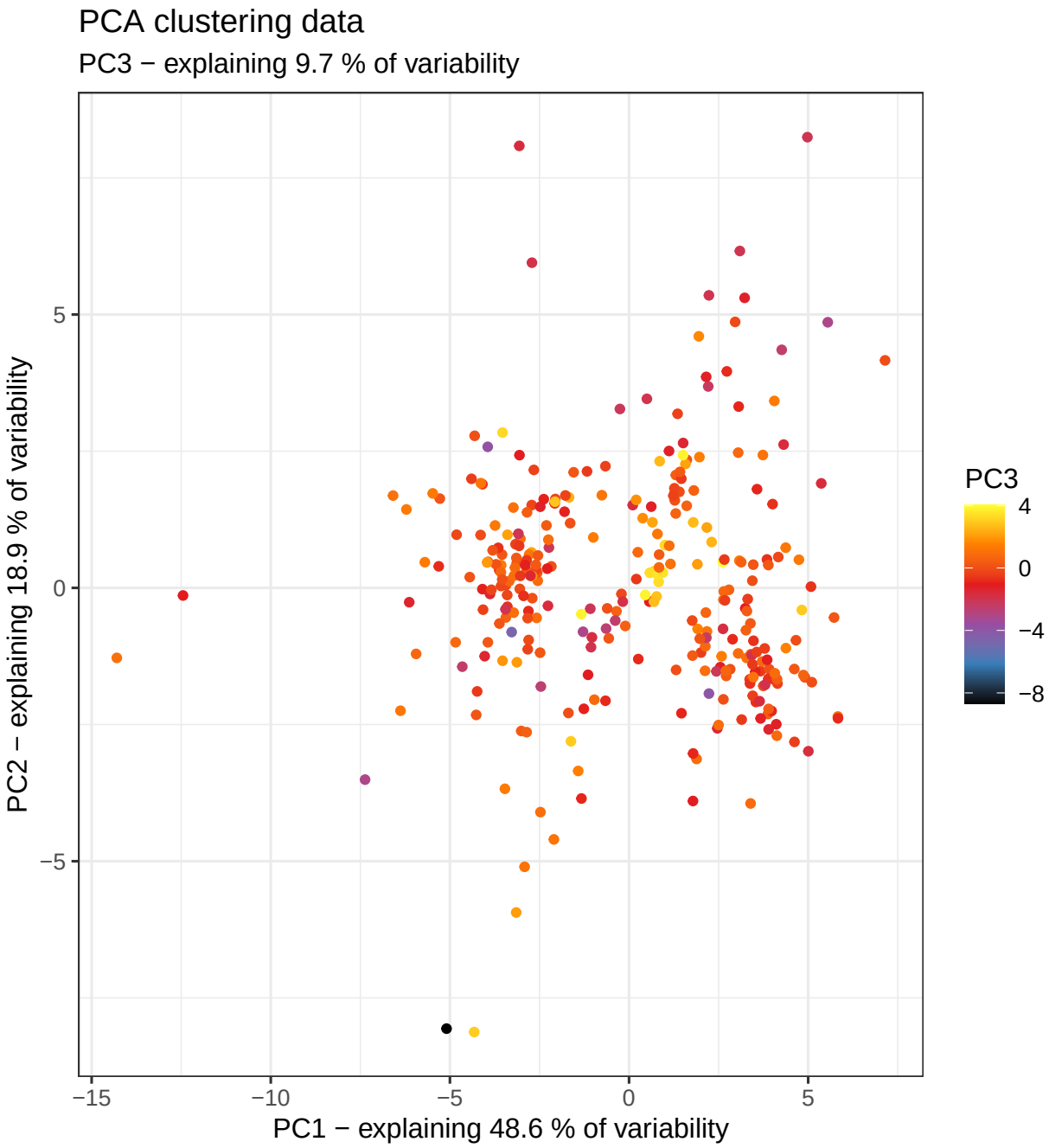



```
# hits <- hits[hits %in% cdata$gene_name[cdata$max.qupm >= 10]]

# old.data <- read.csv("old_full_data.csv")
# old.data <- subset(old.data, gene_name %in% mdata$gene_name)
# old.data$median.value <- log2(old.data$median.value)
# hits <- unique(append(hits, as.character(unique(old.data$gene_name))))

m_clust.data <- limma_results %>%
  mutate(condition = comparison2) %>%
```

```
  dplyr::select(-comparison2) %>%
  group_by(gene_name, score, condition) %>%
  dplyr::filter(gene_name %in% hits) %>%
  summarise(median.value = median(score.value, na.rm = TRUE))
# m_clust.data <- bind_rows(m_clust.data, old.data)
clust.data_m <- m_clust.data %>%
  mutate(key = paste(score, condition, sep = "_")) %>%
  ungroup() %>%
  dplyr::select(gene_name, key, median.value) %>%
  group_by(gene_name) %>%
  spread(key = key, value = median.value) %>%
  as.data.frame()
rownames(clust.data_m) <- clust.data_m$gene_name
m_clust.data <- m_clust.data %>%
  dplyr::filter(gene_name %in% clust.data_m$gene_name)
clust.data_m$gene_name <- NULL
clust.data_m <- clust.data_m %>%
  as.matrix()
clust.data_m[is.na(clust.data_m)] <- 0
m_clust.data$SILAC <- gsub("_.+", "", m_clust.data$condition)
m_clust.data$condition <- gsub(".+_", "", m_clust.data$condition)
m_clust.data$condition <- factor(m_clust.data$condition, ordered = TRUE, levels = c("t0", "t30", "t90",
```

##PCA of clustering data

```
clust.data_PCA <- prcomp(clust.data_m, scale = FALSE)
perc_var <-
  round(100 * clust.data_PCA$sdev ^ 2 /
          sum(clust.data_PCA$sdev ^ 2), 1)
PCA_clust.data <-
  data.frame(PC1 = clust.data_PCA$x[, 1],
             PC2 = clust.data_PCA$x[, 2],
             PC3 = clust.data_PCA$x[, 3])
ggplot(data = PCA_clust.data, aes(PC1, PC2, colour = PC3)) +
  geom_point() +
  theme_bw(base_size = 12) +
  scale_colour_gradientn(colours = c("black", "#377eb8", "#984ea3",
                                     "#e41a1c", "#ff7f00", "#ffff33")) +
  ggtitle("PCA clustering data",
          subtitle = paste("PC3 - explaining", perc_var[3], "% of variability")) +
  xlab(paste("PC1 - explaining", perc_var[1], "% of variability")) +
  ylab(paste("PC2 - explaining", perc_var[2], "% of variability"))
```




```
ggsave(file.path(dir_save, paste0("PCA_clustering_data_", script.version, ".pdf")),
       width = 7, height = 5)
rm(clust.data_PCA, perc_var, PCA_clust.data)
```

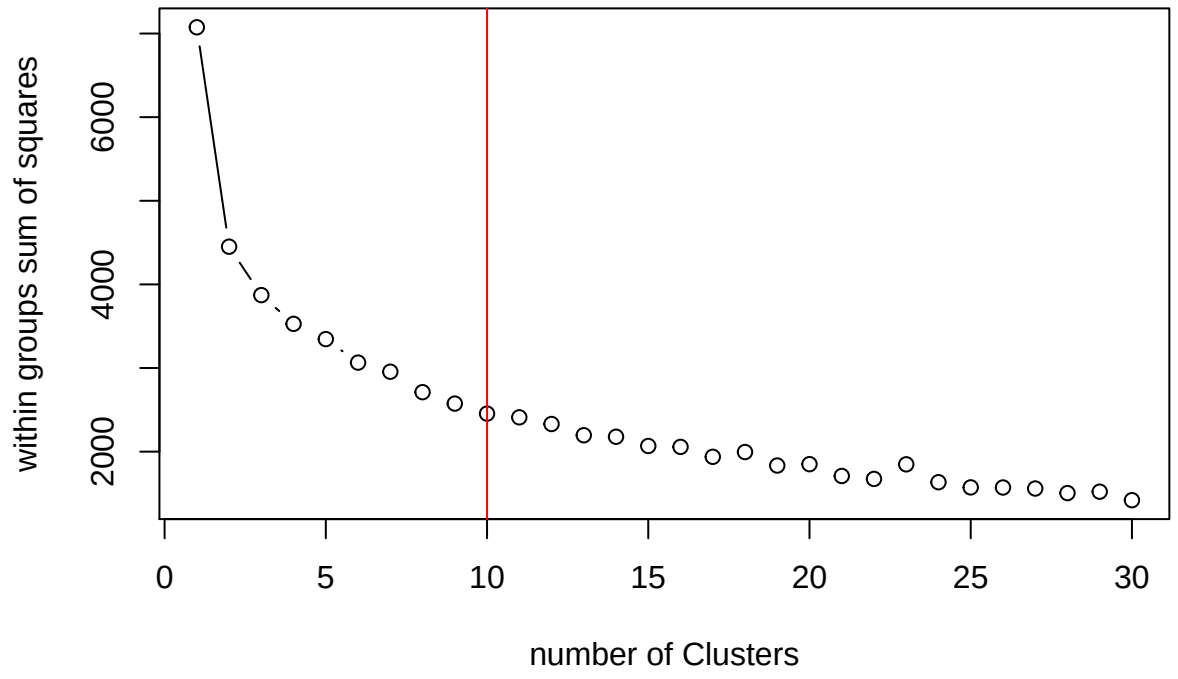
##Set number of clusters

```
cluster.number <- 10
```

##K-means clustering

```
# plot within groups sum of squares for different cluster numbers
wss <- (nrow(clust.data_m) - 1) * sum(apply(clust.data_m, 2, var))
for (i in 2:(cluster.number*3)) wss[i] <- sum(kmeans(clust.data_m,
```

```
                                                    centers = i)$withinss)
plot(1:(cluster.number*3), wss, type = "b", xlab = "number of Clusters",
     ylab = "within groups sum of squares")
abline(v = cluster.number, col = "red")
```



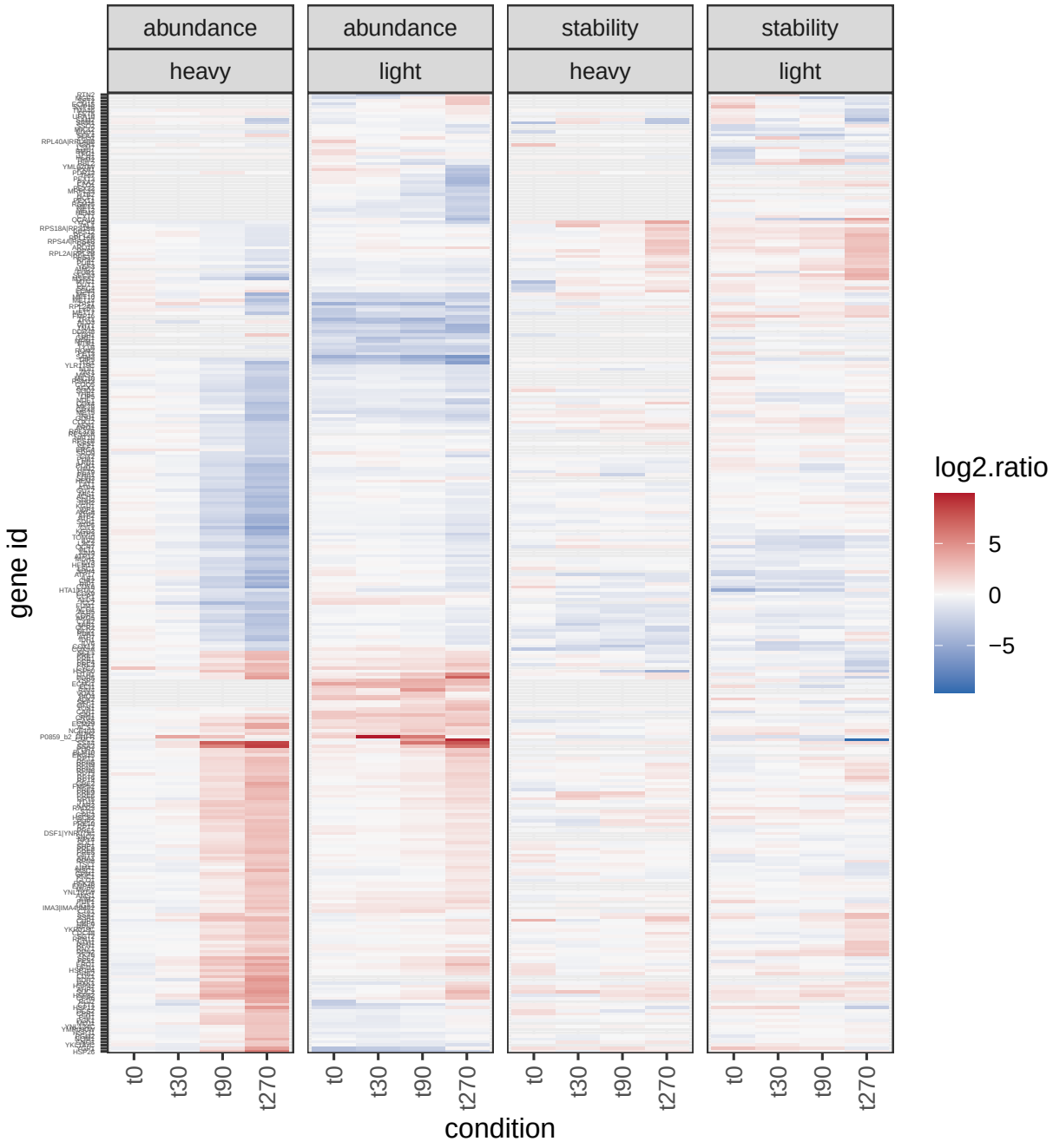

```
kmeans.fit <- kmeans(clust.data_m, cluster.number) # 5 cluster solution
cluster.groups <- data.frame(gene_name = names(kmeans.fit$cluster),
                             kmeans.cluster.group = kmeans.fit$cluster)
rm(wss, kmeans.fit)
```

##Hierarchical clustering

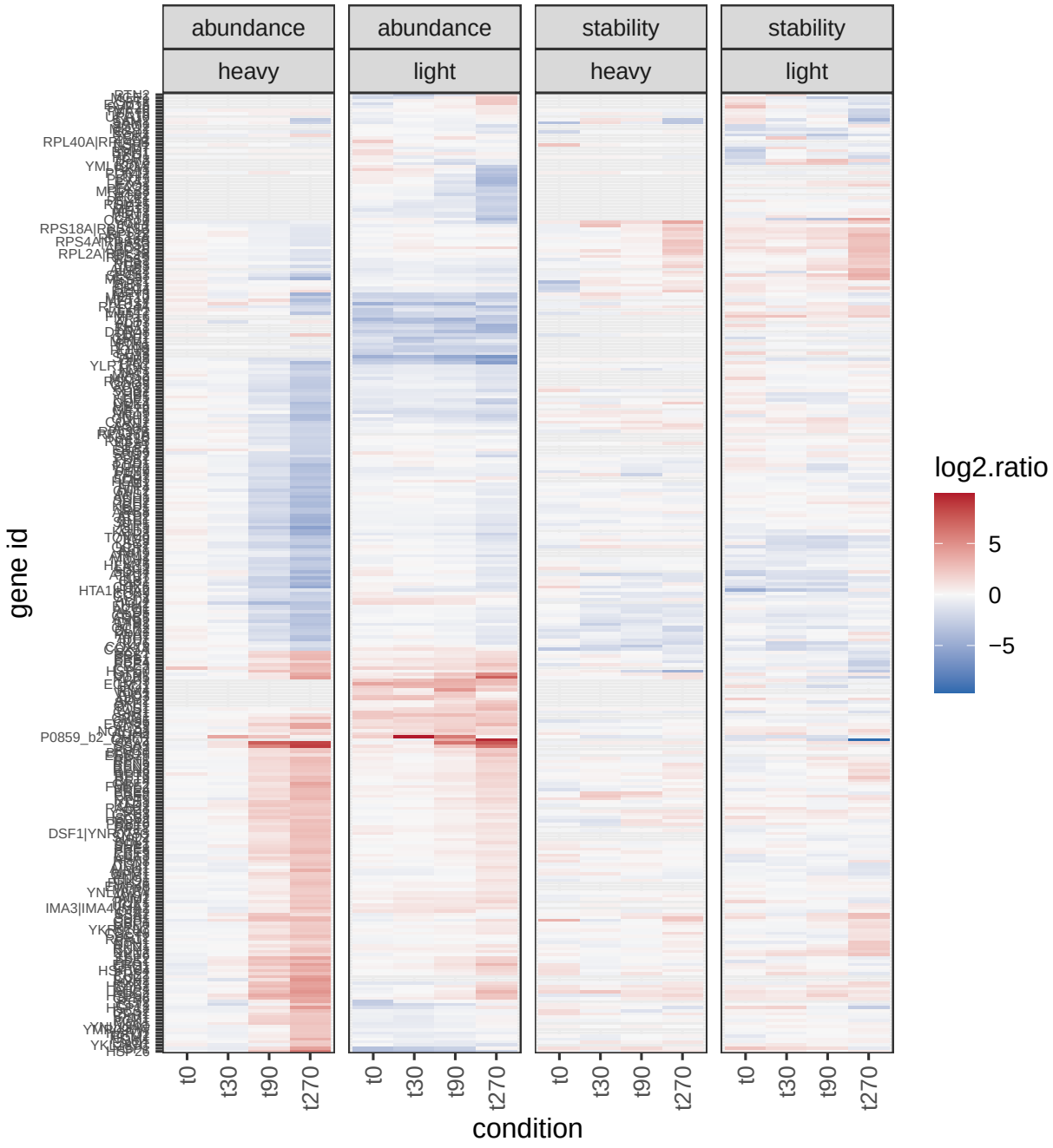
```
d <- dist(clust.data_m, method = "euclidean")
hclust.fit <- hclust(d, method = "ward.D2")
cluster.groups_hclust <-
  data.frame(gene_name = names(cutree(hclust.fit, k = cluster.number)),
             hclust.cluster.group = cutree(hclust.fit, k = cluster.number))
cluster.groups <- left_join(cluster.groups, cluster.groups_hclust)
rm(cluster.groups_hclust)
m_clust.data <- left_join(m_clust.data, cluster.groups)
m_clust.data$gene_name <- factor(m_clust.data$gene_name, ordered = TRUE,
                                 levels = hclust.fit$labels[hclust.fit$order])
rm(d, hclust.fit)
```

##Plot clustering results

```
gr.width <- 2 + ncol(clust.data_m) * 0.1 + max(nchar(rownames(clust.data_m))) * 0.1 + 4
gr.height <- 3 + nrow(clust.data_m) * 0.03
ggplot(data = m_clust.data, aes(condition, gene_name)) +
  geom_tile(aes(fill = median.value)) +
  scale_fill_gradient2(low = "#2166ac", high = "#b2182b", midpoint = 0,
                       mid = "#f7f7f7", name = "log2.ratio") +
  facet_grid(~ score + SILAC, scale = "free_x") +
  theme_bw(base_size = 12) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5),
        axis.text.y = element_text(size = 3)) +
  ylab("gene id")
```

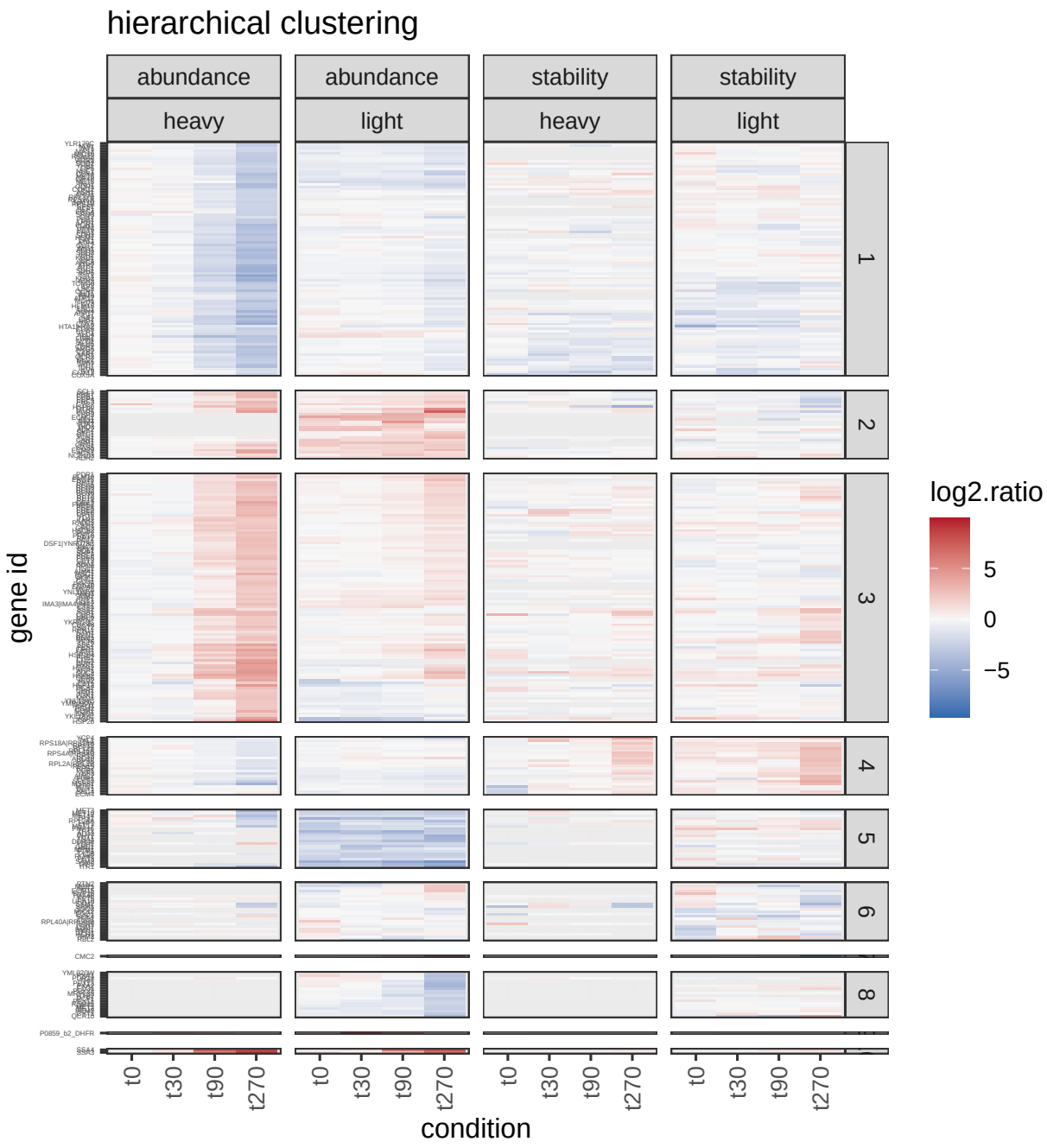



```
ggsave(file.path(dir_save, paste0("Heatmap_hits_", script.version, ".pdf")), width = gr.width, height =

ggplot(data = m_clust.data, aes(condition, gene_name)) +
  geom_tile(aes(fill = median.value)) +
  scale_fill_gradient2(low = "#2166ac", high = "#b2182b", midpoint = 0,
                       mid = "#f7f7f7", name = "log2.ratio") +
  facet_grid(~ score + SILAC, scale = "free_x") +
  theme_bw(base_size = 12) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5),
        axis.text.y = element_text(size = 6)) +
  ylab("gene id")
```



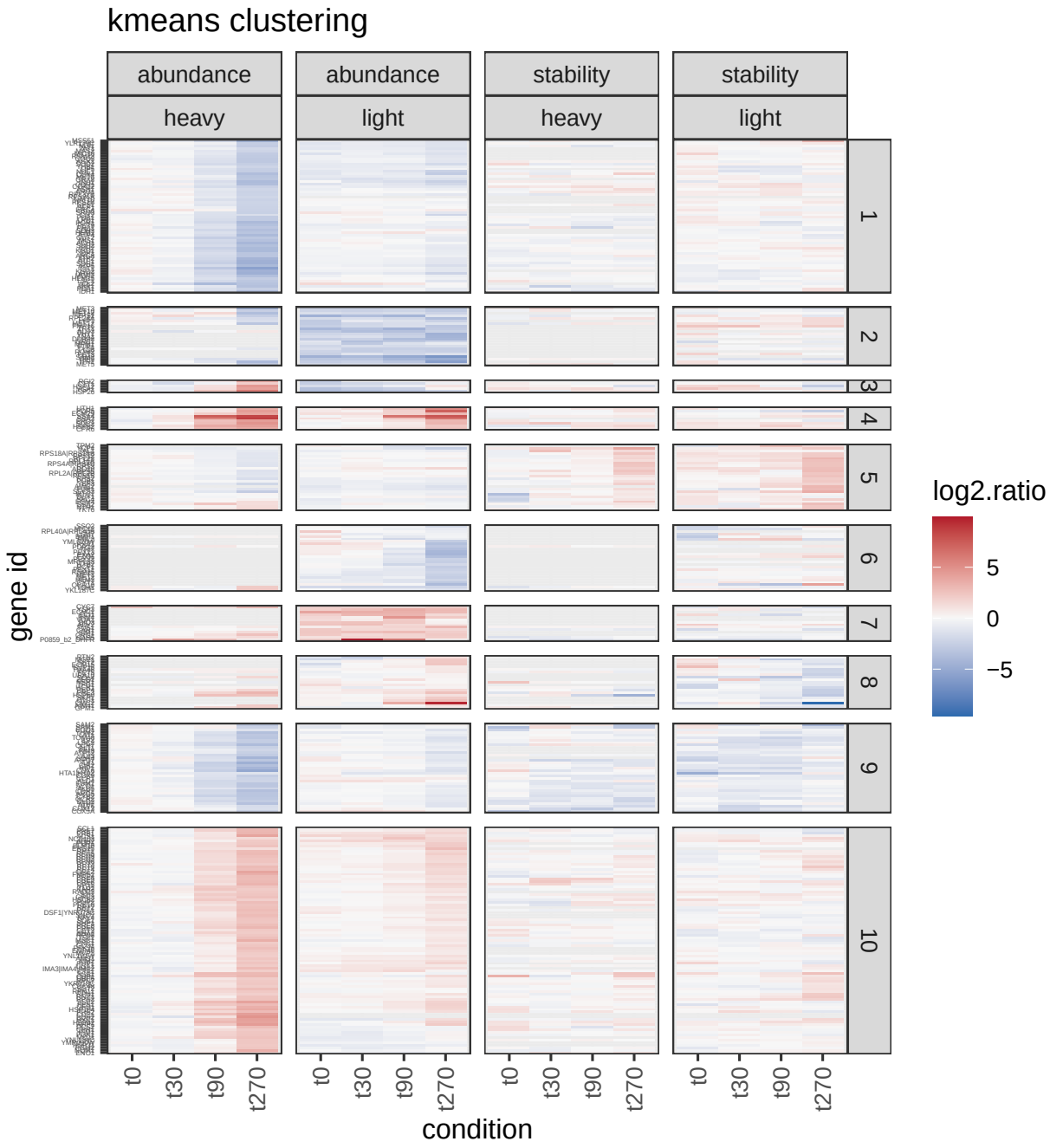

```
ggsave(file.path(dir_save, paste0("Stability_hits_Heatmap_hits_", script.version, ".pdf")), width = gr.w

#Clustering heatmaps
gr.height <- 2.5 + nrow(clust.data_m) * 0.04 + cluster.number * 0.1
ggplot(data = m_clust.data, aes(condition, gene_name)) +
  geom_tile(aes(fill = median.value)) +
  scale_fill_gradient2(low = "#2166ac", high = "#b2182b", midpoint = 0,
                       mid = "#f7f7f7", name = "log2.ratio") +
  facet_grid(hclust.cluster.group ~ score + SILAC, scale = "free", space = "free") +
  theme_bw(base_size = 12) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5),
```

```
        axis.text.y = element_text(size = 3)) +
  ylab("gene id") +
  ggtitle("hierarchical clustering")
```



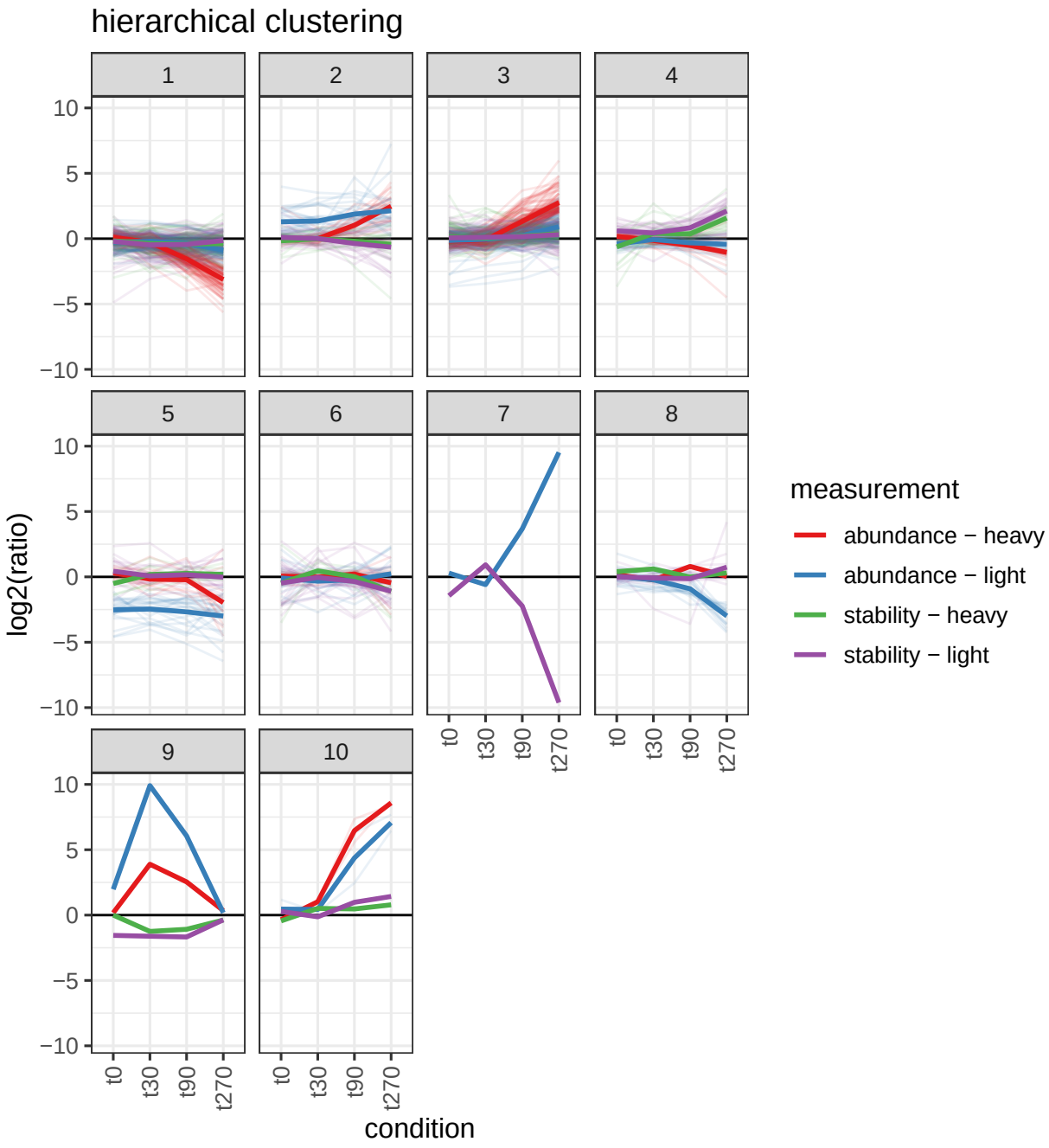

```
ggsave(file.path(dir_save, paste0("Clustering_heatmap_hits_hclust_", cluster.number, "_cluster_", script
       width = gr.width, height = gr.height)

ggplot(data = m_clust.data, aes(condition, gene_name)) +
  geom_tile(aes(fill = median.value)) +
  scale_fill_gradient2(low = "#2166ac", high = "#b2182b", midpoint = 0,
                       mid = "#f7f7f7", name = "log2.ratio") +
```

```
  facet_grid(kmeans.cluster.group ~ score + SILAC, scale = "free", space = "free") +
  theme_bw(base_size = 12) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5),
        axis.text.y = element_text(size = 3)) +
  ylab("gene id") +
  ggtitle("kmeans clustering")
```



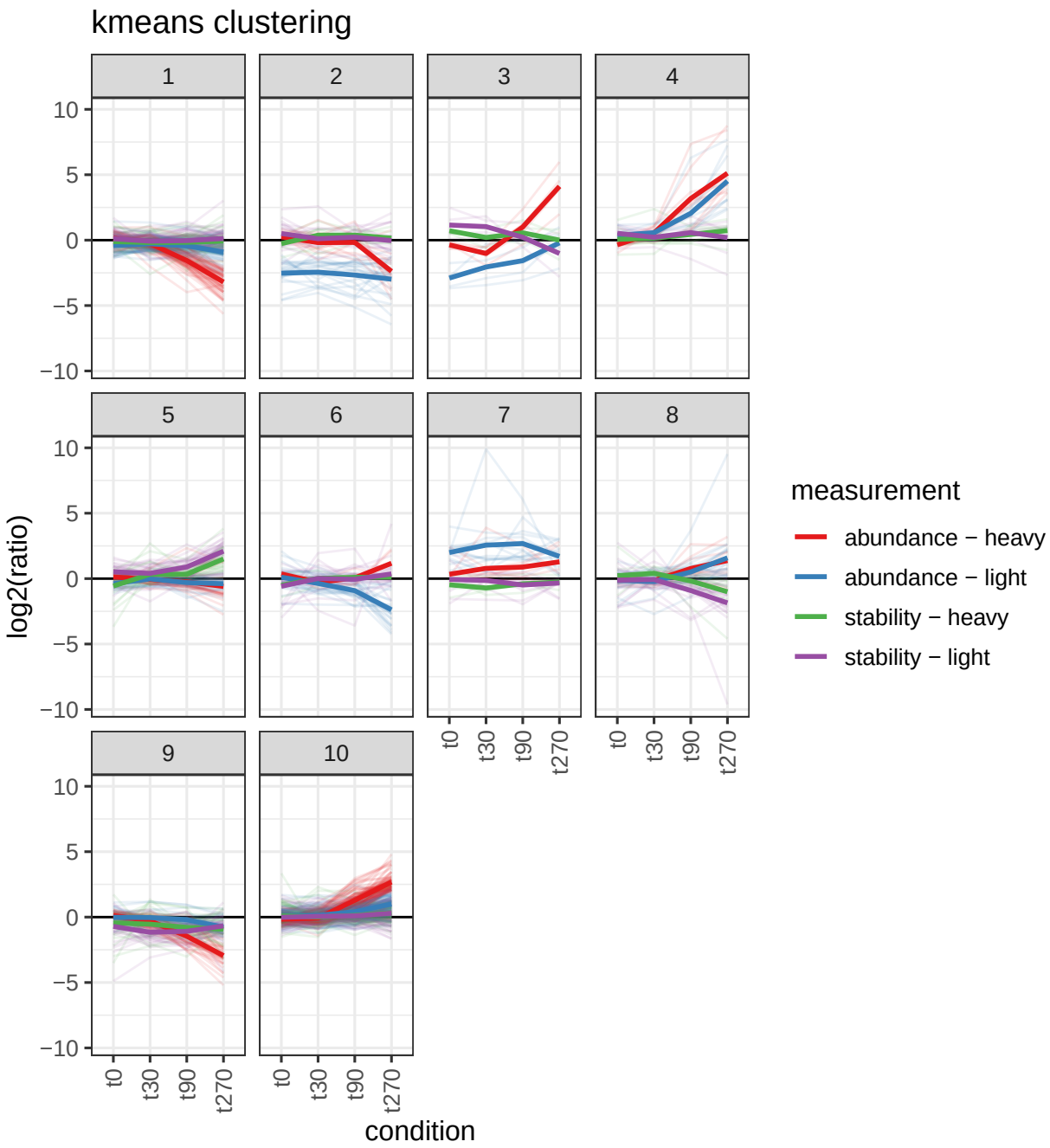

```
ggsave(file.path(dir_save, paste0("Clustering_heatmap_hits_kmeans_", cluster.number, "_cluster_", scrip
       width = gr.width, height = gr.height)

#Line plots
```

```
gr.width <- 4 + ceiling(sqrt(cluster.number)) * 5
gr.height <- 3 + floor(sqrt(cluster.number)) * 5
m_clust.data$measurement <- paste(m_clust.data$score, m_clust.data$SILAC, sep = " - ")
ggplot(data = m_clust.data, aes(condition, median.value, colour = measurement, fill = measurement)) +
  geom_hline(yintercept = 0) +
  geom_line(aes(group = paste(gene_name, score, hclust.cluster.group, SILAC)), alpha = 0.1) +
  stat_summary(fun.data = "mean_se", geom = "smooth", aes(group = paste(hclust.cluster.group, score, SI
  facet_wrap(~ hclust.cluster.group) +
  customPlot +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ggtitle("hierarchical clustering") +
  ylab("log2(ratio)")
```




```
ggsave(file.path(dir_save, paste0("Clustering_line_plot_hclust_", cluster.number, "_cluster_", script.v
       width = gr.width, height = gr.height)

ggplot(data = m_clust.data, aes(condition, median.value, colour = measurement, fill = measurement)) +
  geom_hline(yintercept = 0) +
  geom_line(aes(group = paste(gene_name, score, kmeans.cluster.group, SILAC)), alpha = 0.1) +
  stat_summary(fun.data = "mean_se", geom = "smooth", aes(group = paste(kmeans.cluster.group, score, SI
  facet_wrap(~ kmeans.cluster.group) +
  customPlot +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5)) +
  ggtitle("kmeans clustering") +
```

```
  ylab("log2(ratio)")
```




```
ggsave(file.path(dir_save, paste0("Clustering_line_plot_kmeans_", cluster.number, "_cluster_", script.v
       width = gr.width, height = gr.height)
write.csv(cluster.groups, file.path(dir_save, paste0("Cluster_results_",cluster.number,"cluster_", scri
rm(gr.width, gr.height)
```

#Save workspace

```
write.csv(mdata, file = file.path(dir_save, paste0("Tidy_data_all_", script.version, ".csv")))
write.csv(limma.cp.data, file = file.path(dir_save, paste0("Tidy_score_data_", script.version, ".csv"))
```

```
save.image(file.path(dir_save, paste0("Workspace_", script.version, ".RData")))
# load(file.path(dir_save, paste0("Workspace_", script.version, ".RData")))
```

#Save individual plots for each protein

```
library(gridExtra)
dir_save.plots = file.path(dir_save, "plots_combined")
if (!dir.exists(file.path(getwd(), dir_save.plots)))
{
  dir.create(file.path(getwd(), dir_save.plots))
}
h=15
w=15
cols <- c("#e41a1c","#377eb8","#4daf4a","#984ea3","#ff7f00","#ffff33","#a65628",
          "#f781bf","#999999","#1b9e77","#d95f02","#7570b3","#e7298a","#66a61e",
          "#e6ab02","#a6761d")

limits.limma = quantile(limma.cp.data$score.value,probs = c(0.001,0.999),na.rm = TRUE)
for (gene in sort(as.character(unique(subset(cdata)$gene_name))))
{
  gene.name <- gsub("\\|", "-", gene)
  f.name <- file.path(dir_save.plots, paste0(gene.name, ".pdf"))
  if (!file.exists(f.name))
  {
    pdf(width = w, height = h, file = f.name)
    sub <- subset(mdata, gene_name == gene)
    sub.cdata <- subset(cdata, gene_name == gene)
    average.top3 = sub.cdata$average.top3
    max.qupm <- sub.cdata$max.qupm
    ptitle <- paste(gene, "-", "average.top3:", average.top3,"-", "max.qupm:", max.qupm)
    subtitle <- paste(sub.cdata$protein_id, "-", sub.cdata$description)

    try(p1 <- ggplot(data = subset(sub, measurement == "norm_batchcl_raw_signal_sum"), aes(factor(Tempe
          geom_tile(aes(fill=log2(value))) +
          scale_fill_gradientn(colours = c("#377eb8",
                                           "#984ea3",
                                           "#e41a1c",
                                           "#ff7f00",
                                           "#ffff33"), name = "log2(norm signal_sum)") +
          xlab("Temperature") +
          facet_grid(SILAC ~ rep, scale = "free_x", space = "free") +
          theme_bw(base_size = 12) +
          theme(legend.position="bottom") +
          theme(axis.text.x = element_text(angle = 90, hjust = 1,vjust = 0.5)))
    try(p2 <- ggplot(data = subset(sub, measurement == "norm_batchcl_raw_signal_sum"), aes(Temperature,
          stat_summary(fun.data = "mean_se",geom = "smooth") +
          theme_bw(base_size = 12) +
          theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
          scale_fill_manual(values = cols) +
          scale_colour_manual(values = cols) +
          facet_grid(time ~ SILAC) +
          ylab("log2(norm signal_sum)") +
          stat_summary(fun.data = "mean_se",geom = "line", size = 2))
    try(p3 <- ggplot(data = subset(sub, measurement == "ctrl.ratio"),
```

```
                  aes(factor(Temperature), condition)) +
      geom_tile(aes(fill=log2(value))) +
      scale_fill_gradient2(low = "#2166ac", high = "#b2182b", midpoint = 0, mid = "#f7f7f7", name =
      xlab("Temperature") +ylab("condition") +
      facet_grid(SILAC ~ rep, scale = "free_x", space = "free") +
      theme_bw(base_size = 12) +
      theme(axis.text.x = element_text(angle = 90, hjust = 1,vjust = 0.5), legend.position="bottom"
try(p4 <- ggplot(data = subset(sub, measurement == "temp.ratio"), aes(Temperature, log2(value),fill=
      stat_summary(fun.data = "mean_se",geom = "smooth") +
      theme_bw(base_size = 12) +
      theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
      ylab("log2(temp.ratio)") +
      scale_fill_manual(values = cols) +
      facet_grid(time ~ SILAC) +
      scale_colour_manual(values = cols) +
      stat_summary(fun.data = "mean_se", geom = "line", size = 2))

if (gene%in%limma.cp.data$gene_name)
{
  sub <- subset(limma.cp.data,gene_name ==gene)
  sub$condition <- factor(sub$condition, ordered = TRUE, levels = levels(mdata$condition))
  new.limits <- limits.limma
  if (min(sub$score.value,na.rm = TRUE)<new.limits[1])
  {
    new.limits[1] <- min(sub$score.value,na.rm = TRUE)
  }
  if (max(sub$score.value,na.rm = TRUE)>new.limits[2])
  {
    new.limits[2] <- max(sub$score.value,na.rm = TRUE)
  }
  # try(p5 <- ggplot(data = subset(sub, grepl("_vs_control", condition) & !grepl("^control_", condi
  #       geom_tile() +
  #       coord_polar() +
  #       scale_fill_gradient2(low = "#2166ac",high = "#b2182b",
  #                            midpoint = 0,mid = "#f7f7f7",
  #                            limits = new.limits) +
  #       theme_bw(base_size = 12) +
  #       xlab("replicate") +
  #       geom_text(aes(label = fdr.star)))
  try(p6 <- ggplot(data = subset(sub, !grepl("^control", condition)), aes(x=condition,score.value,
        geom_boxplot(fill = "darkgrey") +
        geom_point(aes(shape = rep)) +
        theme_bw(base_size = 12) +
        scale_fill_brewer(palette = "Set1") +
        coord_cartesian(ylim = new.limits) +
        geom_text(aes(label = fdr.star,y = new.limits[2]),show.legend = FALSE) +
        facet_grid(. ~ score + SILAC) +
        geom_hline(yintercept = 0) +
        theme(axis.text.x = element_text(angle = 90, hjust = 1,vjust = 0.5)))
  # try(suppressWarnings( print(grid.arrange(p1, p2, p3, p4, p6, ncol = 2, top = ptitle, bottom = s
  try(suppressWarnings( print(grid.arrange(p1, p2, p3, p4, p6, layout_matrix = rbind(c(1,2), c(3,4)
}else{
  try(suppressWarnings( print(grid.arrange(p1, p2, p3, p4, ncol = 2, top = ptitle, bottom = subtitl
```

```
    }
    dev.off()
    rm(sub,min.p.value,new.limits,max.qupm,average.top3)
  }
}
```