Sample - Sample Correlations From Raw Data

Robert M Flight

There is an [Executive Summary](#executive-summary) at the end of this report.

## Purpose

To evaluate the correlation between various samples.

## Data

**Near Raw Data** provided by Justin. This data includes the intensity values for each peptide in each sample. What the PamGene instrument does, based on this data and the description of the instrument from the manual, is:

1. Wash sample onto the appropriate array on the chip (cycle).
2. Take intensity measurements over time (exposure time).
3. Repeat over multiple cycles and exposure times.

So this near raw data is median, background corrected intensities for each peptide in each sample in each array of each chip, over cycles and exposure times. The cycles and exposure times are treated as independent measures for each peptide for this analysis, giving us extra measures to use for correlation and log-fold-changes.

## Methods

We imported nearly raw data provided from the PamGene instrument that includes cycle and exposure time for each peptide on each array and chip. Inspection of the log-intensities determined that values less than 30 are likely to be unreliable. Before correlation analysis, values less than 30 were replaced with zero. Sample-sample correlations were calculated using information-content-informed Kendall-tau (Flight, Robert M and Moseley, Hunter NB [2021] 2021), treating zero as missing values. Only features that had a non-missing value (> 0) in at least one of the samples were included. Clustering of samples used 1 - ICI-Kt as the distance, and then the dendrogram was sorted using dendsort with min type sorting (Sakai et al. 2014). Means, absolute differences between largest and smallest, standard deviations (SD) and relative standard deviations (RSD) were calculated among the three technical replicates for each pooled sample. The RSD mode was calculated using the replicates where the average mean was greater than or equal to 30. All calculations were performed using R version 4.1.0 (R Core Team 2021). Heatmaps were generated using the ComplexHeatmap package version 2.10.0 (Gu, Eils, and Schlesner 2016).

library(tidygraph)

##   
## Attaching package: 'tidygraph'

## The following object is masked from 'package:stats':  
##   
## filter

library(dplyr)

##   
## Attaching package: 'dplyr'

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

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

library(ggraph)

## Loading required package: ggplot2

library(visualizationQualityControl)  
library(ComplexHeatmap)

## Loading required package: grid

## ========================================  
## ComplexHeatmap version 2.10.0  
## Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/  
## Github page: https://github.com/jokergoo/ComplexHeatmap  
## Documentation: http://jokergoo.github.io/ComplexHeatmap-reference  
##   
## If you use it in published research, please cite:  
## Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional   
## genomic data. Bioinformatics 2016.  
##   
## The new InteractiveComplexHeatmap package can directly export static   
## complex heatmaps into an interactive Shiny app with zero effort. Have a try!  
##   
## This message can be suppressed by:  
## suppressPackageStartupMessages(library(ComplexHeatmap))  
## ========================================

library(ggraph)  
library(flextable)  
theme\_set(cowplot::theme\_cowplot())  
knitr::opts\_chunk$set(fig.width = 8, fig.height = 8)  
source(here::here("scripts", "functions.R"))

raw\_data = read\_raw\_files()

## New names:  
## \* `#REF` -> `#REF...7`  
## \* `#REF` -> `#REF...8`  
## \* `#REF` -> `#REF...9`  
## \* `#REF` -> `#REF...10`  
## \* `#REF` -> `#REF...11`  
## \* ...  
## New names:  
## \* `#REF` -> `#REF...7`  
## \* `#REF` -> `#REF...8`  
## \* `#REF` -> `#REF...9`  
## \* `#REF` -> `#REF...10`  
## \* `#REF` -> `#REF...11`  
## \* ...  
## New names:  
## \* `#REF` -> `#REF...7`  
## \* `#REF` -> `#REF...8`  
## \* `#REF` -> `#REF...9`  
## \* `#REF` -> `#REF...10`  
## \* `#REF` -> `#REF...11`  
## \* ...  
## New names:  
## \* `#REF` -> `#REF...7`  
## \* `#REF` -> `#REF...8`  
## \* `#REF` -> `#REF...9`  
## \* `#REF` -> `#REF...10`  
## \* `#REF` -> `#REF...11`  
## \* ...

names(raw\_data) = c("PTK\_median", "PTK\_saturation", "STK\_median", "STK\_saturation")

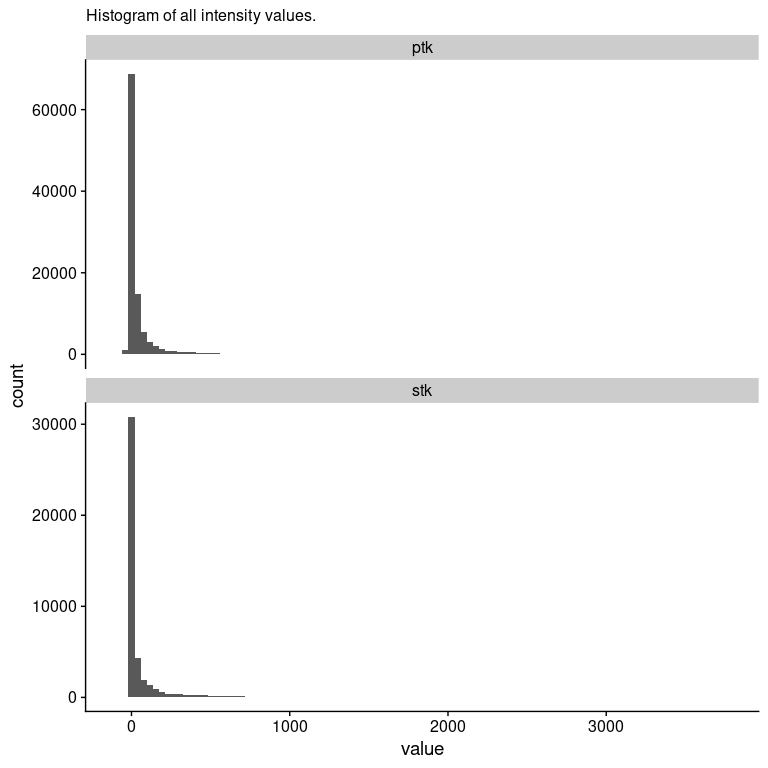
### Transform for Correlation

We will start working with the median values and see what happens.

ptk\_data = transform\_raw\_data(raw\_data$PTK\_median)  
ptk\_info = ptk\_data$sample\_info  
ptk\_info = ptk\_info %>%  
 dplyr::mutate(sample\_id = paste0(comment, "\_", replicate))  
ptk\_raw = ptk\_data$data %>%  
 mutate(kinases = "ptk")  
  
stk\_data = transform\_raw\_data(raw\_data$STK\_median)  
stk\_info = stk\_data$sample\_info  
stk\_raw = stk\_data$data %>%  
 mutate(kinases = "stk")  
  
all\_raw = rbind(ptk\_raw, stk\_raw)

First, we need to check what the distribution of intensity values looks like.

ggplot(all\_raw, aes(x = value)) +  
 geom\_histogram(bins = 100) +  
 facet\_wrap(~ kinases, ncol = 1, scales = "free\_y") +  
 labs(subtitle = "Histogram of all intensity values.")

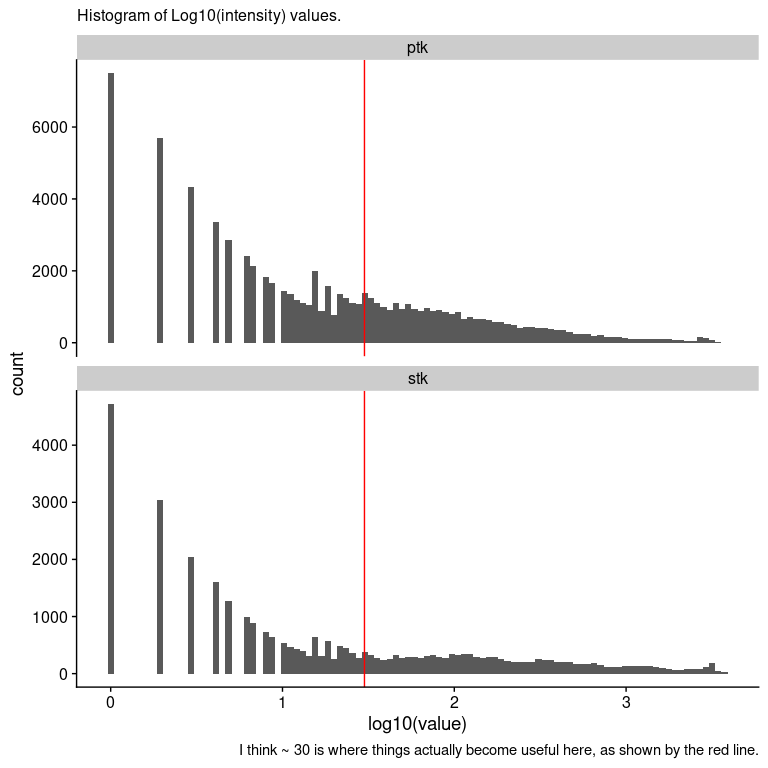


This definitely looks log-normal, so we will transform, and use log10 so we can easily tell where to apply any kind of cutoff.

ggplot(all\_raw, aes(x = log10(value))) +  
 geom\_histogram(bins = 100) +  
 geom\_vline(xintercept = log10(30), color = "red") +  
 facet\_wrap(~ kinases, ncol = 1, scales = "free\_y") +  
 labs(subtitle = "Histogram of Log10(intensity) values.",  
 caption = "I think ~ 30 is where things actually become useful here, as shown by the red line.")

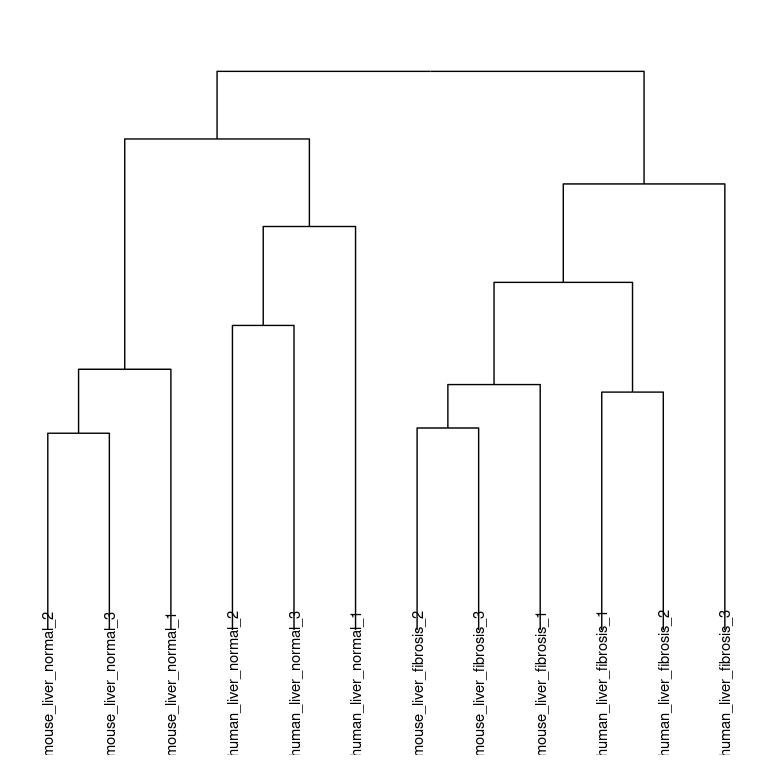
## Warning in FUN(X[[i]], ...): NaNs produced  
  
## Warning in FUN(X[[i]], ...): NaNs produced

## Warning: Removed 38742 rows containing non-finite values (stat\_bin).



Based on this, I actually wouldn’t trust any intensity values < 30 (1.5) in the data for correlation. Given that, we will take all the intensity values < 30 and set them to 0, and treat them as missing for the ICI-Kendall-tau correlation calculation.

all\_raw = all\_raw %>%  
 dplyr::mutate(measure\_id2 = case\_when(  
 kinases %in% "ptk" ~ paste0("p.", measure\_id),  
 kinases %in% "stk" ~ paste0("s.", measure\_id)  
 ))  
sample\_wise = all\_raw %>%  
 dplyr::select(sample\_name2, value, measure\_id2) %>%  
 tidyr::pivot\_wider(id\_cols = measure\_id2,  
 names\_from = sample\_name2,  
 values\_from = value)  
  
sample\_matrix = sample\_wise %>%  
 dplyr::select(-measure\_id2) %>%  
 as.matrix()  
sample\_matrix[sample\_matrix < 30] = 0  
is\_1 = apply(sample\_matrix, 1, function(.x){  
 sum(.x > 0) > 0  
})  
sample\_matrix = sample\_matrix[is\_1, ]  
sample\_cor\_data = ICIKendallTau::ici\_kendalltau(t(sample\_matrix), global\_na = c(NA, 0), perspective = "global")  
sample\_cor = sample\_cor\_data$cor  
  
sample\_order = similarity\_reorder(sample\_cor, transform = "sub\_1")  
sample\_dendrogram = tidygraph::as\_tbl\_graph(sample\_order$dendrogram)  
ggraph(sample\_dendrogram, 'dendrogram', height = height) +  
 geom\_edge\_elbow() +  
 geom\_node\_text(aes(label = label, filter = leaf), angle = 90,  
 nudge\_y = -0.02) +  
 scale\_y\_continuous(expand = expansion(mult = 0.1, 0))



Cool! Everything seems to cluster together by liver status. Note that this only happens after combining both the **PTK** and **STK** data together, and replacing values < 30, and removing entries that are < 30 across all the samples, leaving us with 4256.

Let’s look at the full correlation matrix and verify that this clustering is correct.

ptk\_info = dplyr::left\_join(data.frame(sample\_id = colnames(sample\_cor)), ptk\_info, by = "sample\_id")  
ptk\_info = ptk\_info %>%  
 dplyr::mutate(organism = case\_when(  
 grepl("mouse", comment) ~ "mouse",  
 grepl("human", comment) ~ "human"  
 ),  
 condition = case\_when(  
 grepl("normal", comment) ~ "control",  
 grepl("fibrosis", comment) ~ "fibrosis"  
 ),  
 type = paste0(organism, "\_", condition))  
  
col\_map = circlize::colorRamp2(seq(0.7, 1, length.out = 20), viridis::viridis(20))  
annote\_cols = c("human\_fibrosis" = "#C00000",  
 "mouse\_fibrosis" = "#404040",  
 "human\_control" = "#E6B9B8",  
 "mouse\_control" = "#BFBFBF")  
#names(annote\_cols) = unique(ptk\_info$type)  
col\_annotation = HeatmapAnnotation(df = ptk\_info[, "type", drop = FALSE], which = "column", col = list(type = annote\_cols), show\_annotation\_name = FALSE)  
  
row\_annotation = HeatmapAnnotation(df = ptk\_info[, "type", drop = FALSE], which = "row", col = list(type = annote\_cols), show\_annotation\_name = FALSE, show\_legend = FALSE)  
  
  
sample\_sample\_heatmap = Heatmap(sample\_cor, col = col\_map, "ICI-Kt", cluster\_rows = sample\_order$dendrogram,  
 cluster\_columns = sample\_order$dendrogram, top\_annotation = col\_annotation,  
 left\_annotation = row\_annotation,  
 show\_column\_names = FALSE,  
 show\_row\_names = FALSE)  
draw(sample\_sample\_heatmap, merge\_legend = TRUE)

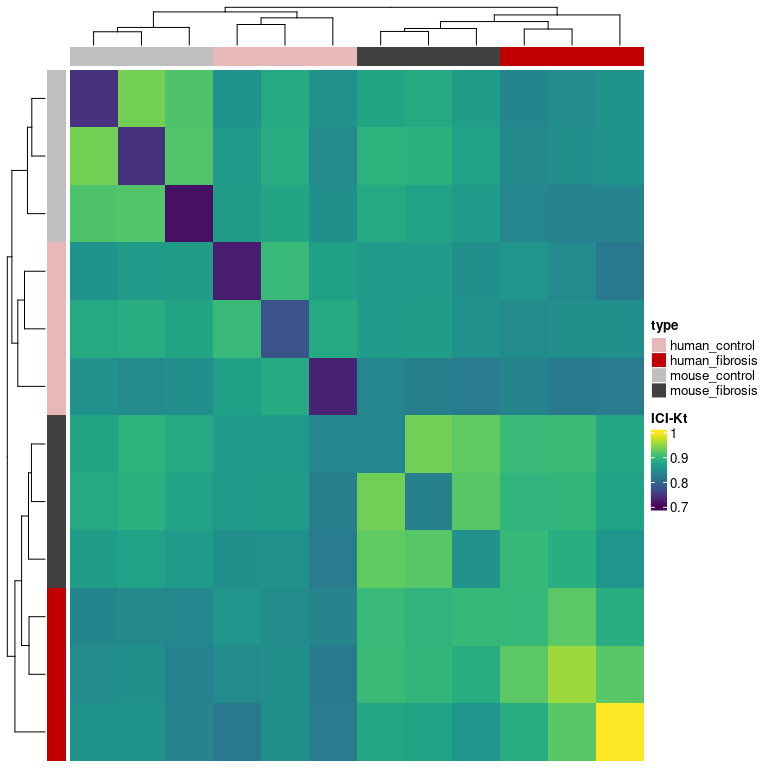


Figure X. Sample-sample ICI-Kt correlation heatmap. Diagonal values represent the number of non-missing value in the sample.

Note that **all** correlation p-values are < 2.2x10^16.

What is the correlation within and between the groups of samples?

group\_cor = median\_class\_correlations(sample\_cor, ptk\_info$type)  
group\_cor\_text = format(group\_cor, digits = 2)  
group\_cor\_text[lower.tri(group\_cor\_text, diag = FALSE)] = ""  
  
group\_cor\_table = knitr::kable(group\_cor\_text, caption = "Median correlations within and between organism and liver state. Within liver state correlations across organisms are lower than within a liver state / organism combination, but higher than between liver state.")  
  
group\_cor\_table

Median correlations within and between organism and liver state. Within liver state correlations across organisms are lower than within a liver state / organism combination, but higher than between liver state.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | mouse\_fibrosis | mouse\_control | human\_fibrosis | human\_control |
| mouse\_fibrosis | 0.88 | 0.82 | 0.85 | 0.84 |
| mouse\_control |  | 0.92 | 0.84 | 0.90 |
| human\_fibrosis |  |  | 0.92 | 0.88 |
| human\_control |  |  |  | 0.93 |

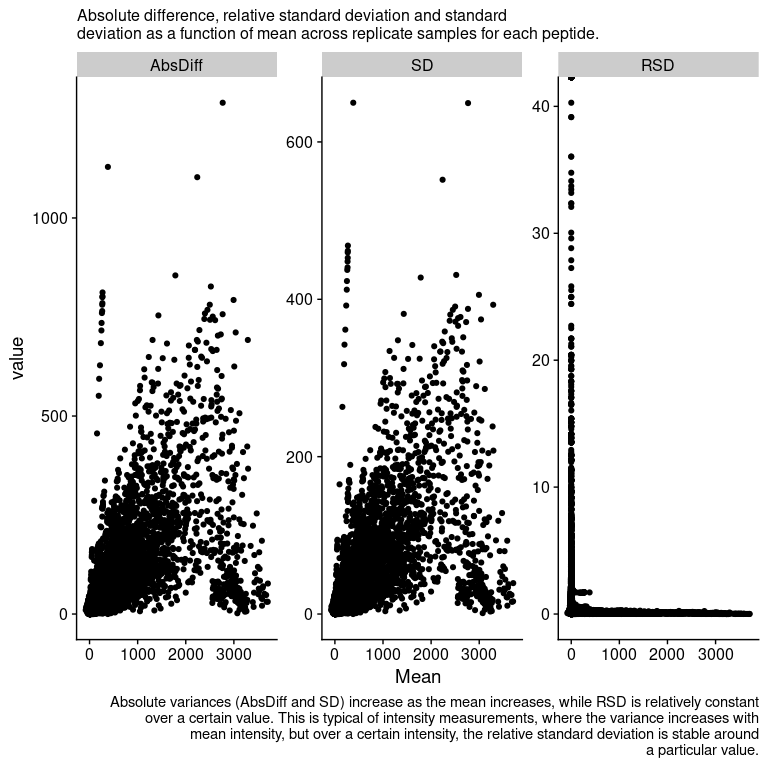
### Variance Measures

variance\_measures = all\_raw %>%  
 dplyr::select(comment, measure\_id2, value) %>%  
 dplyr::group\_by(comment, measure\_id2) %>%  
 dplyr::summarise(Mean = mean(value),  
 AbsDiff = abs(max(value) - min(value)),  
 SD = sd(value),  
 RSD = abs(SD / Mean),  
 n = n())

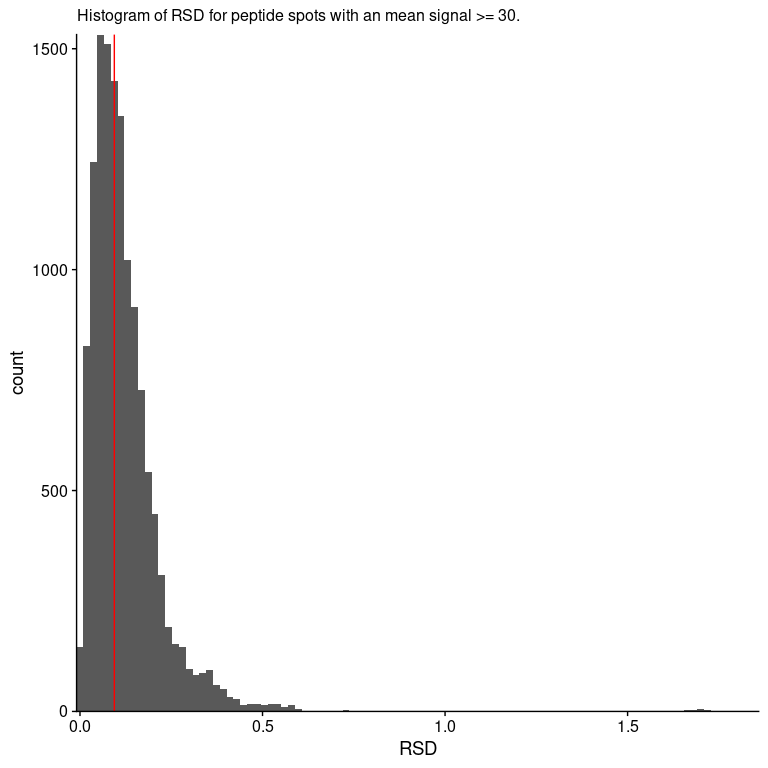
## `summarise()` has grouped output by 'comment'. You can override using the `.groups` argument.

variance\_long = variance\_measures %>%  
 tidyr::pivot\_longer(cols = c(-comment, -measure\_id2, -Mean, -n),   
 names\_to = "summary",  
 values\_to = "value")  
variance\_long$summary = factor(variance\_long$summary, levels = c("AbsDiff", "SD", "RSD"))  
  
ggplot(variance\_long, aes(x = Mean, y = value)) +  
 geom\_point() +  
 facet\_wrap(vars(summary), scales = "free\_y") +  
 labs(subtitle = "Absolute difference, relative standard deviation and standard\ndeviation as a function of mean across replicate samples for each peptide.",   
 caption = "Absolute variances (AbsDiff and SD) increase as the mean increases, while RSD is relatively constant\nover a certain value. This is typical of intensity measurements, where the variance increases with\nmean intensity, but over a certain intensity, the relative standard deviation is stable around\na particular value.")

## Warning: Removed 398 rows containing missing values (geom\_point).



p\_rsd = variance\_measures %>%  
 dplyr::filter(Mean >= 30) %>%  
 ggplot(aes(x = RSD)) +   
 geom\_histogram(bins = 100)  
  
rsd\_density = variance\_measures %>%  
 dplyr::filter(Mean > 2) %>%  
 dplyr::pull(RSD) %>%  
 density()  
rsd\_mode = rsd\_density$x[which.max(rsd\_density$y)]  
  
p\_rsd +   
 geom\_vline(xintercept = rsd\_mode, color = "red") +  
 coord\_cartesian(expand = FALSE) +  
 geom\_text(label = paste0("Mode: ", format(rsd\_mode, digits = 2)), x = 4, y = 3000) +  
 labs(subtitle = "Histogram of RSD for peptide spots with an mean signal >= 30.")



So this is nice, the **relative standard deviation** is fairly small, only 0.094.

### Translation to Log-Fold-Changes

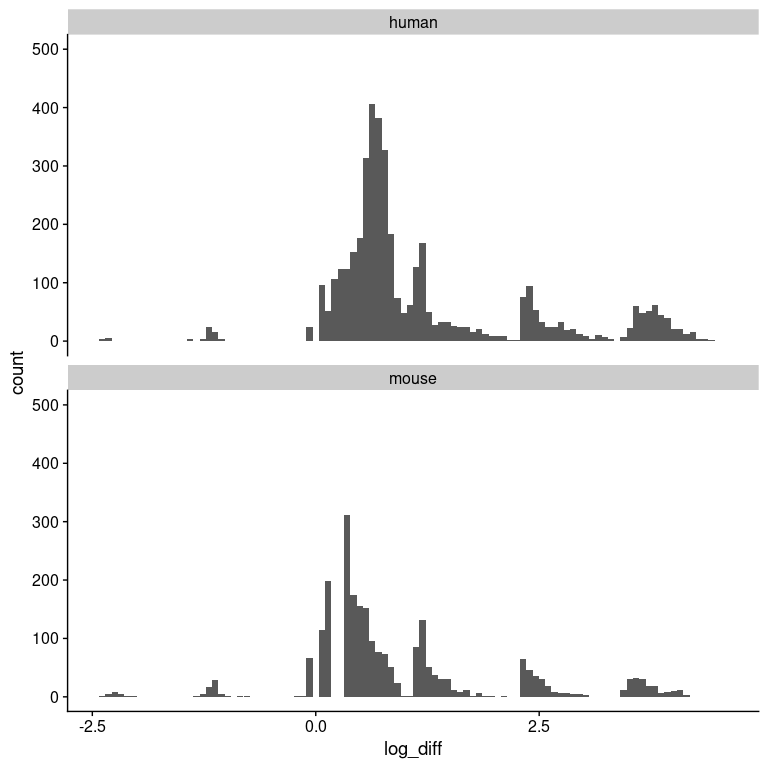
Can we translate these to fold-changes?

log\_raw = all\_raw %>%  
 mutate(hi\_value = case\_when(  
 value < 30 ~ 0,  
 TRUE ~ value  
 )) %>%  
 mutate(log\_value = log(hi\_value + 1))  
  
avg\_raw = log\_raw %>%  
 dplyr::select(comment, measure\_id2, log\_value) %>%  
 dplyr::group\_by(comment, measure\_id2) %>%  
 dplyr::summarise(mean = mean(log\_value)) %>%  
 tidyr::pivot\_wider(names\_from = comment, values\_from = mean)

## `summarise()` has grouped output by 'comment'. You can override using the `.groups` argument.

human\_diff = avg\_raw %>%  
 mutate(log\_diff = e\_human\_liver\_fibrosis - c\_human\_liver\_normal, organism = "human")  
mouse\_diff = avg\_raw %>%  
 mutate(log\_diff = e\_mouse\_liver\_fibrosis - c\_mouse\_liver\_normal,  
 organism = "mouse")  
all\_diff = rbind(human\_diff, mouse\_diff)  
  
ggplot(all\_diff, aes(x = log\_diff)) +   
 geom\_histogram(bins = 100) +  
 facet\_wrap(~ organism, ncol = 1) +  
 scale\_y\_continuous(limits = c(0, 500))

## Warning: Removed 4 rows containing missing values (geom\_bar).



Remember, these are not from the values spit out by PamGene, but based on the background-corrected, median values for each cycle and exposure, and then values < 30 imputed to zero and log-transformed. We average the values across the replicates, and then take the fibrotic - normal.

set\_flextable\_defaults(digits = 2)  
big\_border = fp\_border\_default(width = 2)  
group\_cor\_df = as.data.frame(group\_cor) %>%  
 tibble::rownames\_to\_column("type") %>%  
 flextable() %>%  
 colformat\_double() %>%  
 vline(border = big\_border, j = 1, part = "all") %>%  
 autofit()  
   
plot\_list = list(correlation = sample\_sample\_heatmap,  
 table = group\_cor\_df)  
export\_plots(plot\_list, "sample\_sample\_raw\_correlation.pptx")

## Executive Summary

* Used the *nearly raw* data from PamGene for correlations. This contains the median, background corrected intensity data for each cycle and exposure for each peptide in both the STK and PTK samples. This results in 4256 features for sample-sample correlation.
* Clean separation by liver condition and then by species.
* All sample-sample correlations were significant, and below the machine precision of the computer (2.2x10^-16).
* It’s also clear that kinase activity is elevated in the liver fibrosis vs normal (see diagonal of heatmap). There are more missing features in the normal samples (lover correlation values on the diagonal).

group\_cor\_df

| type | mouse\_fibrosis | mouse\_control | human\_fibrosis | human\_control |
| --- | --- | --- | --- | --- |
| mouse\_fibrosis | 0.88 | 0.82 | 0.85 | 0.84 |
| mouse\_control | 0.85 | 0.92 | 0.84 | 0.90 |
| human\_fibrosis | 0.87 | 0.84 | 0.92 | 0.88 |
| human\_control | 0.86 | 0.88 | 0.87 | 0.93 |

## Exported captions

### Heatmap Figure

Sample-sample correlation heatmap of replicate samples from human and mouse control and fibrotic liver samples. Diagonal entries represent the percentage of non-missing values in each sample. Correlations were calculated using ICI-Kt using features that had non-zero values in at least one sample. Feature values < 30 were treated as missing.

### Table Caption

Median ICI-Kt correlations among samples within a type (diagonal) and between types (off diagonal).

## References

Flight, Robert M, and Moseley, Hunter NB. (2021) 2021. *ICIKendallTau* (version 0.1.8). C++. <https://github.com/MoseleyBioinformaticsLab/ICIKendallTau>.

Gu, Zuguang, Roland Eils, and Matthias Schlesner. 2016. “Complex Heatmaps Reveal Patterns and Correlations in Multidimensional Genomic Data.” *Bioinformatics (Oxford, England)*.

R Core Team. 2021. *R: A Language and Environment for Statistical Computing* (version 4.1.0). Vienna, Austria. <https://www.R-project.org/>.

Sakai, Ryo, Raf Winand, Toni Verbeiren, Andrew Vande Moere, and Jan Aerts. 2014. “dendsort: modular leaf ordering methods for dendrogram representations in R.” *F1000Research* 3: 177. <https://doi.org/10.12688/f1000research.4784.1>.