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.

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 2.8.0  
## 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)  
theme\_set(cowplot::theme\_cowplot())  
knitr::opts\_chunk$set(fig.width = 8, fig.height = 8)  
source(here::here("kea3\_functions.R"))  
save\_loc = here::here("kea3\_enrichment\_2021-09-29")

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")

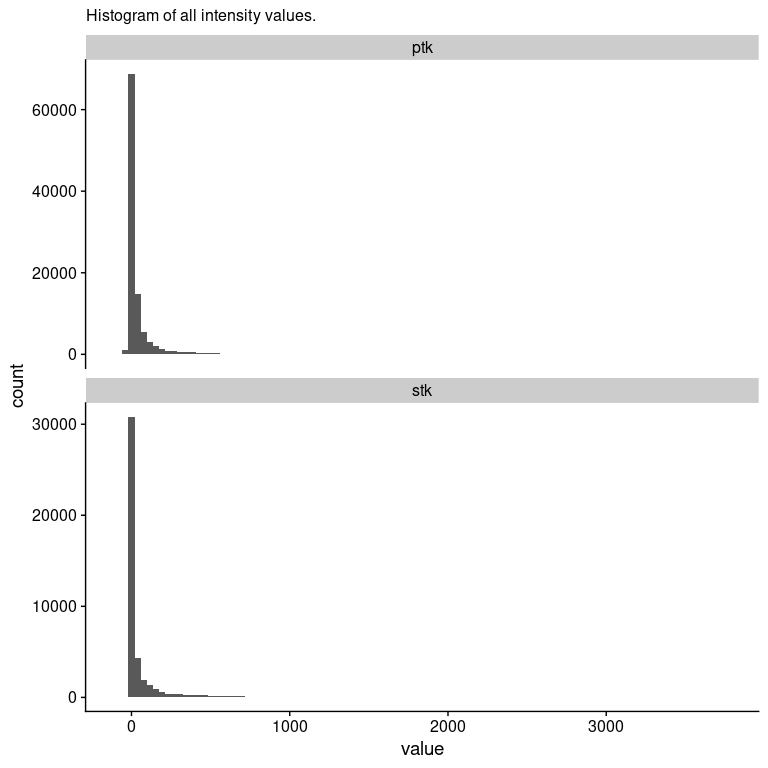
### Transform for Correlation

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.")

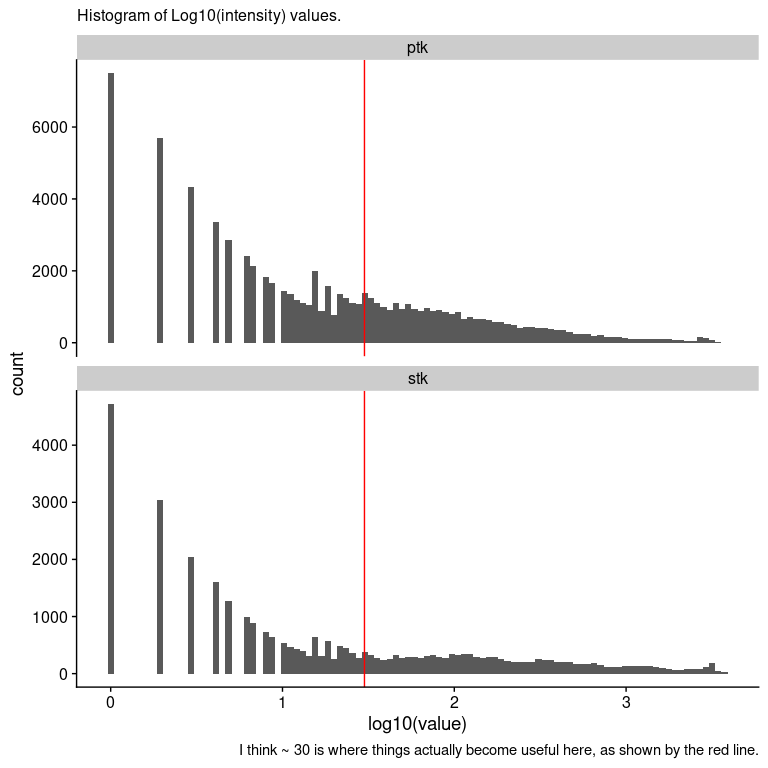


This definitely looks log-normal, so we will transform, and use log10 so we can easily tell where to apply any kind of cutoff.

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).

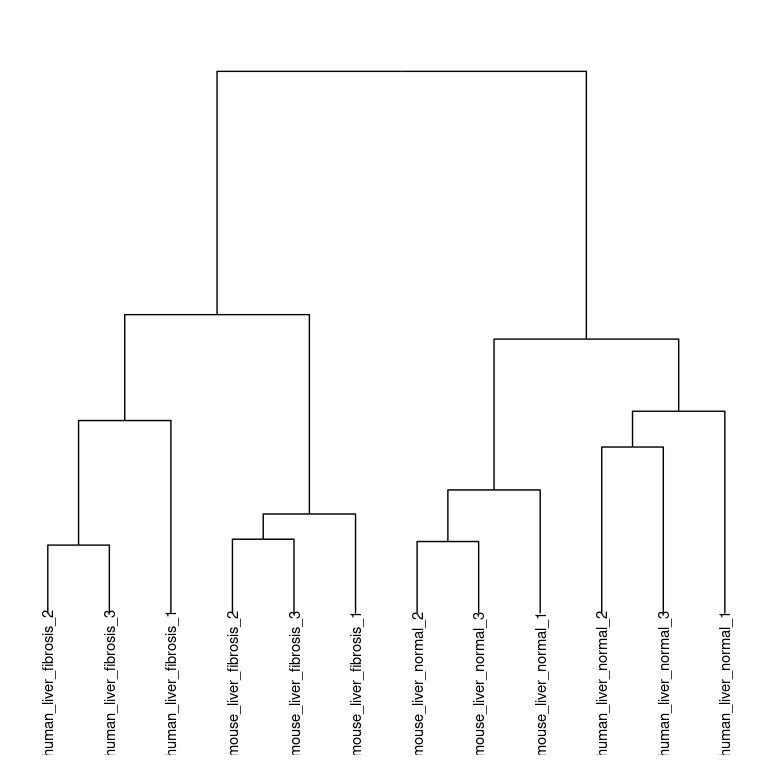


Based on this, I actually wouldn’t trust any intensity values < 30 (1.5) in the data for correlation. Given that, we will take all the intensity values < 30 and set them to 0, and treat them as missing for the ICI-Kendall-tau correlation calculation.

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  
sample\_cor = ICIKendallTau::ici\_kendalltau(t(sample\_matrix), global\_na = c(NA, 0), perspective = "global")$cor

## Warning: UNRELIABLE VALUE: Future ('<none>') unexpectedly generated random  
## numbers without specifying argument 'seed'. There is a risk that those random  
## numbers are not statistically sound and the overall results might be invalid.  
## To fix this, specify 'seed=TRUE'. This ensures that proper, parallel-safe random  
## numbers are produced via the L'Ecuyer-CMRG method. To disable this check, use  
## 'seed=NULL', or set option 'future.rng.onMisuse' to "ignore".

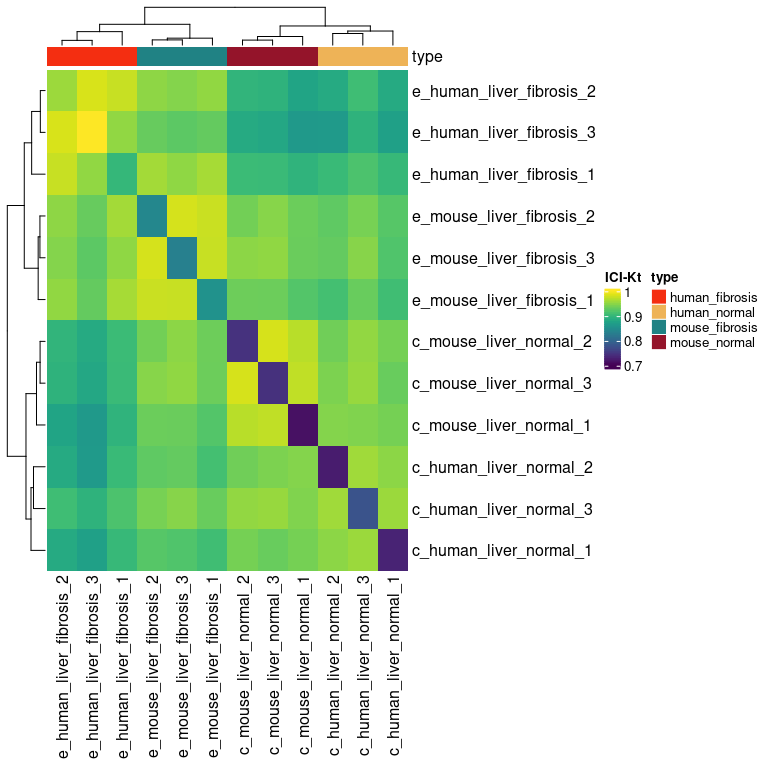
sample\_order = similarity\_reorder(sample\_cor, transform = "sub\_1")  
sample\_dendrogram = tidygraph::as\_tbl\_graph(sample\_order$dendrogram)  
ggraph(sample\_dendrogram, 'dendrogram', height = height) +  
 geom\_edge\_elbow() +  
 geom\_node\_text(aes(label = label, filter = leaf), angle = 90,  
 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

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) ~ "normal",  
 grepl("fibrosis", comment) ~ "fibrosis"  
 ),  
 type = paste0(organism, "\_", condition))  
  
col\_map = circlize::colorRamp2(seq(0.7, 1, length.out = 20), viridis::viridis(20))  
col\_annotation = HeatmapAnnotation(df = ptk\_info[, "type", drop = FALSE], which = "column")  
  
  
Heatmap(sample\_cor, col = col\_map, "ICI-Kt", cluster\_rows = sample\_order$dendrogram,  
 cluster\_columns = sample\_order$dendrogram, top\_annotation = col\_annotation)



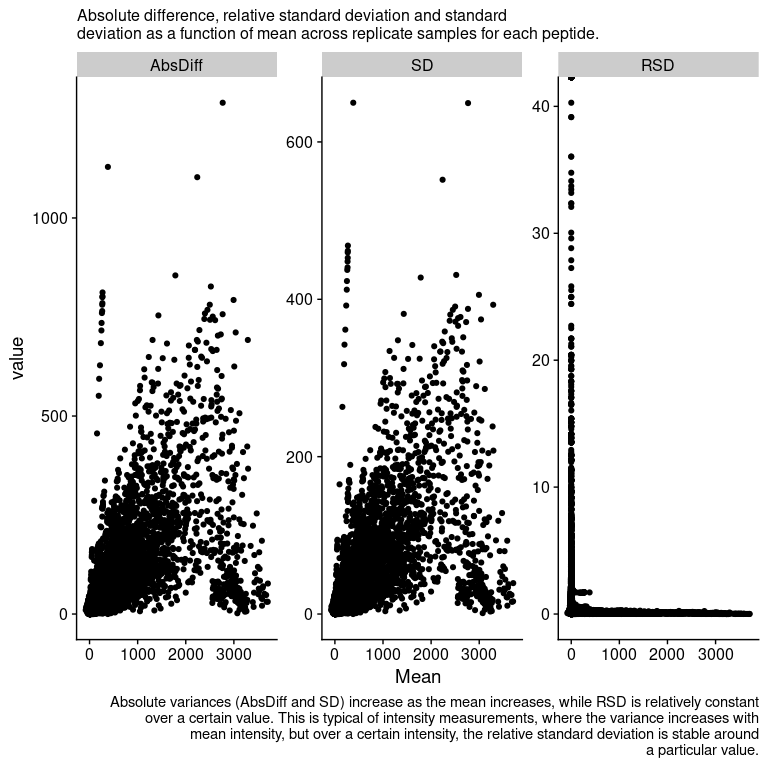
### Variance Measures

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())

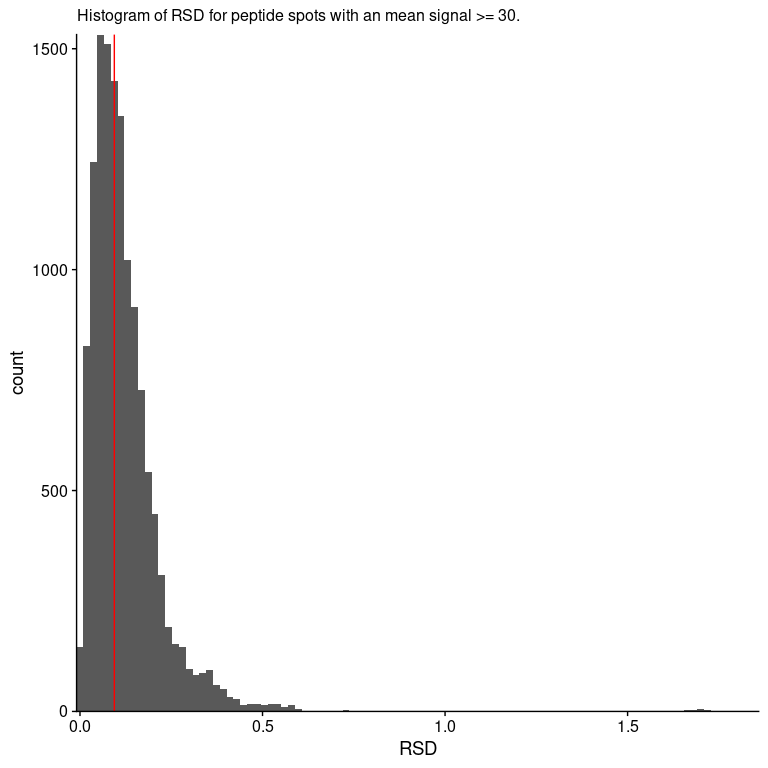
## `summarise()` has grouped output by 'comment'. You can override using the `.groups` argument.

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

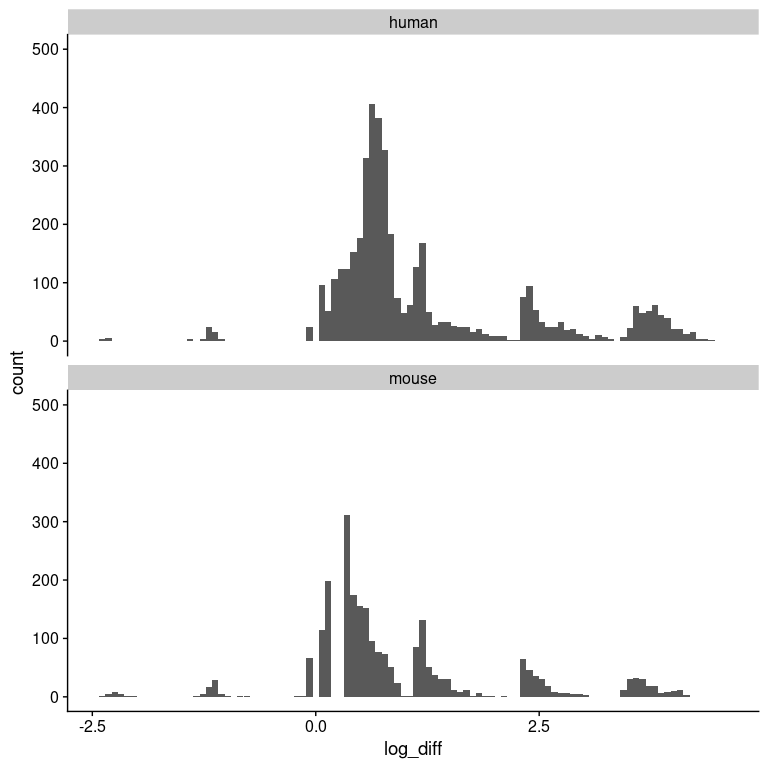
Can we translate these to fold-changes?

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.