Exploring HPA Consensus Data

Robert M Flight

Purpose

How exactly should we be using expression cutoffs in the Human Protein Atlas (HPA) consensus data set?

```
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(ggplot2)
library(UpSetR)
source(here::here("scripts", "functions.R"))
```

Data

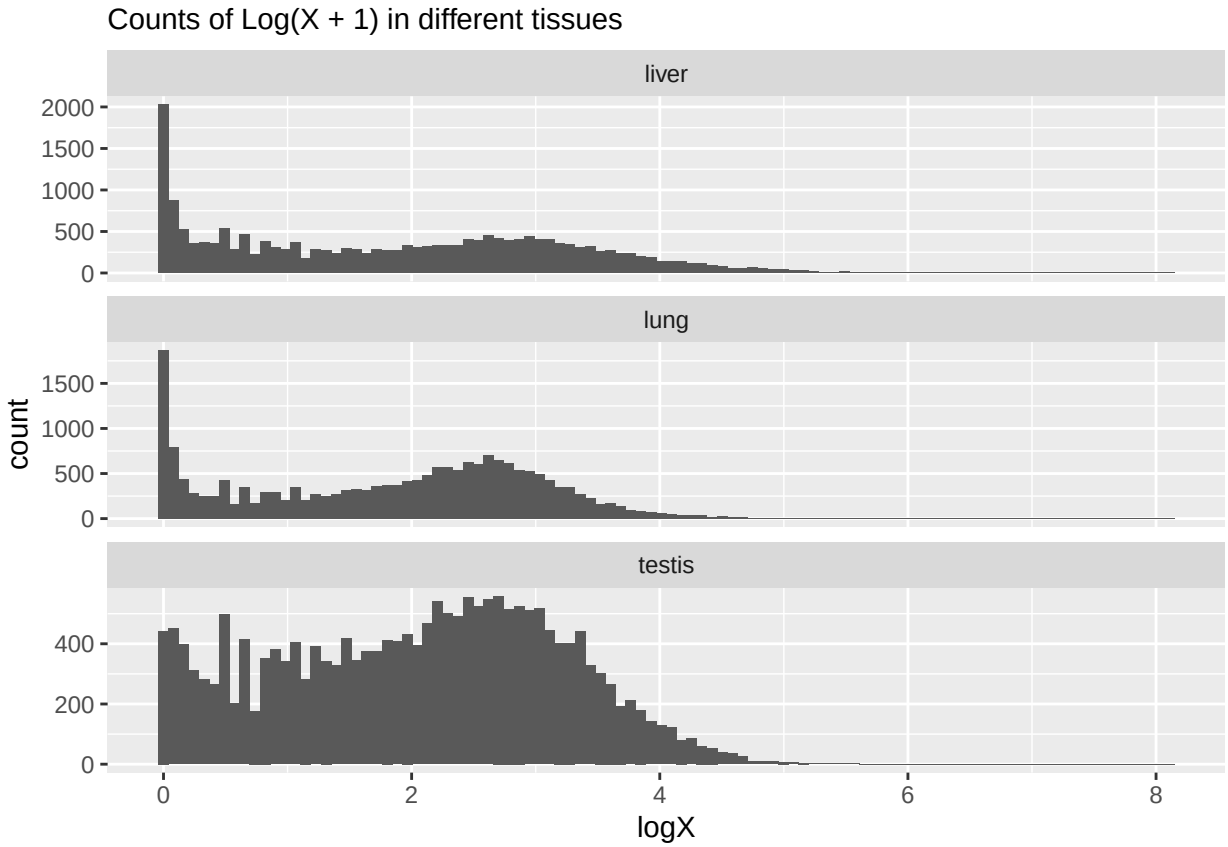
HPA consensus data downloaded on 2021-09-27.

```
hpa_expression = read.table(here::here("data", "inputs", "hpa", "rna_consensus.tsv"),
                            header = TRUE, sep = "\t")
hpa_expression = hpa_expression %>%
  dplyr::mutate(logX = log1p(NX))
```

Explore

What does tissue specific expression look like?

```
hpa_expression %>%
  dplyr::filter(Tissue %in% c("liver", "lung", "testis")) %>%
  ggplot(aes(x = logX)) +
  geom_histogram(bins = 100) +
  facet_wrap(~ Tissue, scales = "free_y", ncol = 1) +
  labs(subtitle = "Counts of Log(X + 1) in different tissues")
```



OK, that's kinda cool. Looks like each tissue might have the same set of genes, just with zero's if there is no expression.

```
hpa_expression %>%
  dplyr::group_by(Tissue) %>%
  dplyr::summarise(n = n()) %>%
  dplyr::pull(n) %>%
  unique()
```

```
## [1] 19670 19304 17717 18816
```

Not quite. But close.

OK, lets see how the counts of our genes look if we use a cutoff > 0.

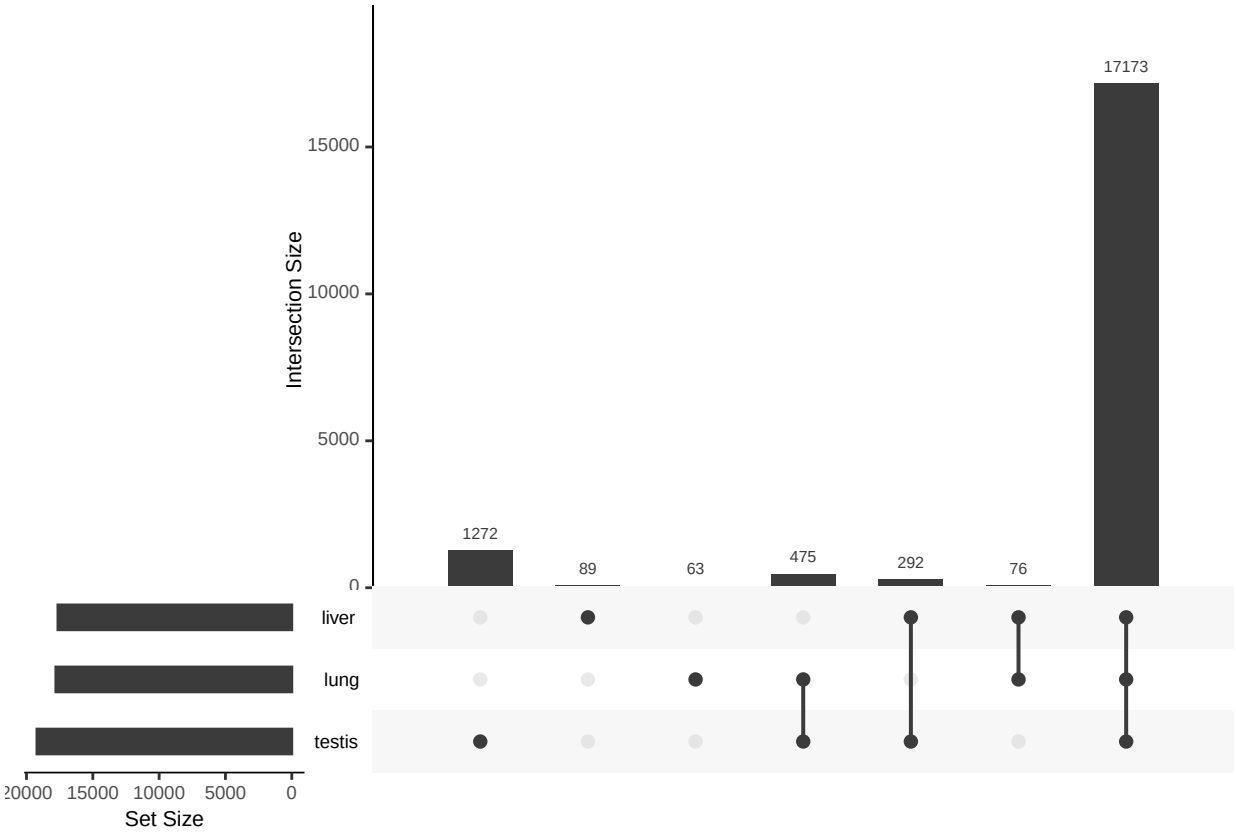
```
greater_0 = hpa_expression %>%
  dplyr::filter(Tissue %in% c("lung", "liver", "testis")) %>%
  split(., .$Tissue) %>%
  purrr::map(., filter_hpa_expression, tissue = NULL, cutoff = 0.01)
purrr::map_int(greater_0, ~ nrow(.x))
```

```
##  liver   lung testis
##  17643  17803  19228
```

```
min_values = purrr::map_dbl(greater_0, ~ min(.x$logX))
min_values
```

```
##      liver       lung     testis
## 0.09531018 0.09531018 0.09531018
```

```
g0_lists = purrr::map(greater_0, ~ unique(.x$Gene.name))
upset(fromList(g0_lists))
```



I've also tried using a tissue specific expression cutoff that tries to find the mode of the distribution, then uses the data on the high side to define a standard deviation (with mean of data replaced by the mode), and then apply X SDs in the negative direction to define a lower value cutoff for expression.

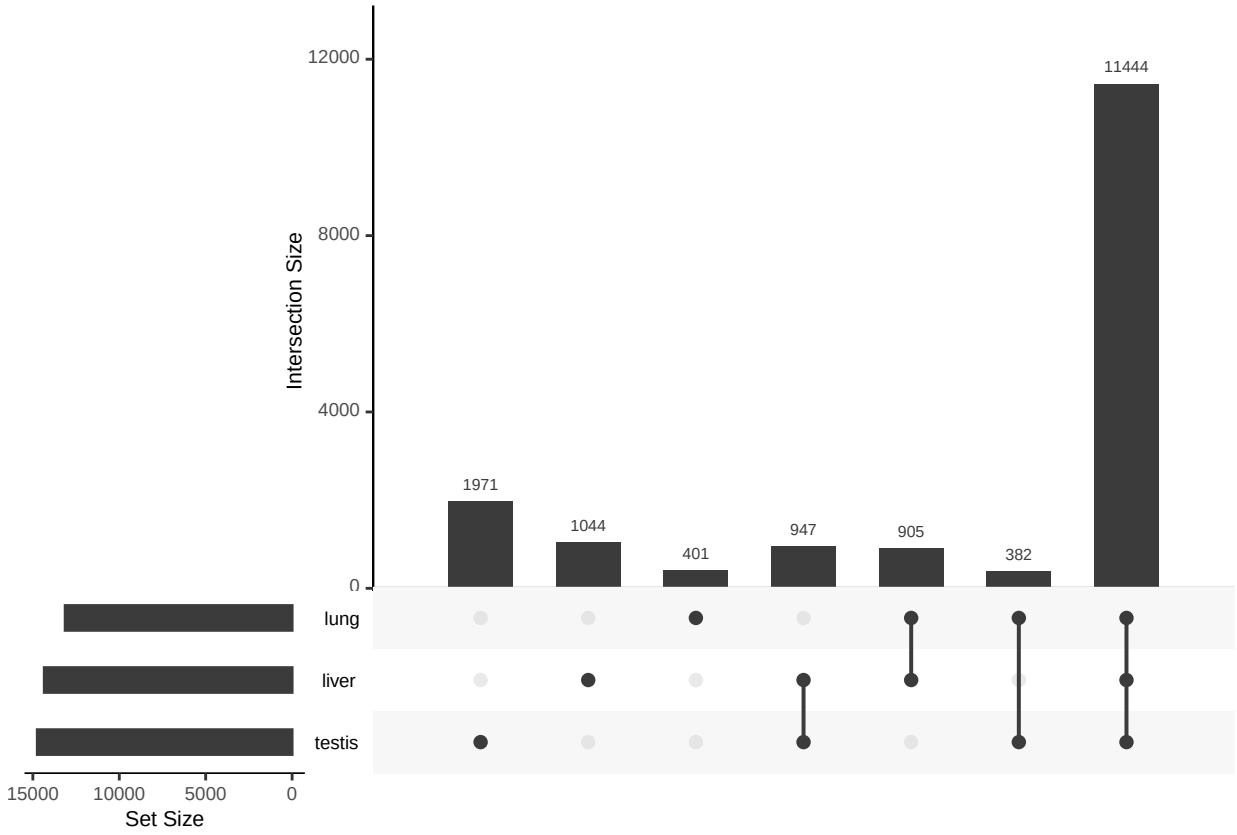
```
greater_2sd = hpa_expression %>%
  dplyr::filter(Tissue %in% c("lung", "liver", "testis")) %>%
  split(., .$Tissue) %>%
  purrr::map(., filter_hpa_expression, tissue = NULL, n_sd = 2)
purrr::map_int(greater_2sd, ~ nrow(.x))
```

```
##  liver   lung testis
##  14351  13137  14753
```

```
g2sd_lists = purrr::map(greater_2sd, ~ unique(.x$Gene.name))
g2sd_min = purrr::map_dbl(greater_2sd, ~ min(.x$logX))
g2sd_min
```

```
##     liver      lung    testis
## 0.6418539 1.2527630 1.1314021
```

```
upset(fromList(g2sd_lists))
```




```
greater_3sd = hpa_expression %>%
  dplyr::filter(Tissue %in% c("lung", "liver", "testis")) %>%
  split(., .$Tissue) %>%
  purrr::map(., filter_hpa_expression, tissue = NULL, n_sd = 3)
purrr::map_int(greater_3sd, ~ nrow(.x))
```

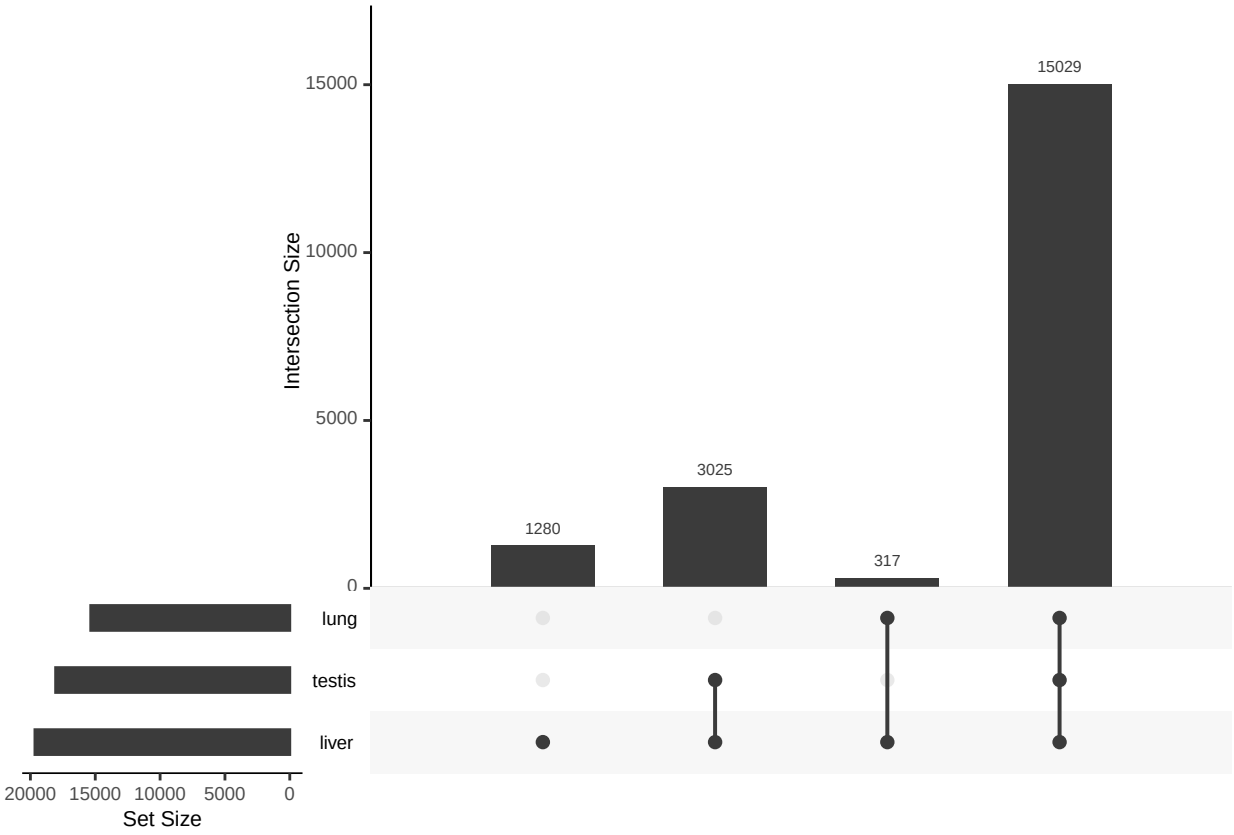
```
##  liver   lung testis
##  19670  15355  18070
```

```
g3sd_min = purrr::map_dbl(greater_3sd, ~ min(.x$logX))
g3sd_min
```

```
##     liver      lung    testis
## 0.0000000 0.5877867 0.3364722
```

```
g3sd_lists = purrr::map(greater_3sd, ~ unique(.x$Gene.name))

upset(fromList(g3sd_lists))
```



So using 0, results in very little tissue specificity, using 2 SD is **much more** specific, and 3 SD is somewhere in between. Probably 3 SD is reasonable.

Except that for liver, there are some really extreme values that seem to make the SD bigger than we might expect. So maybe 2 SD if we want to not just use > 0 for liver.

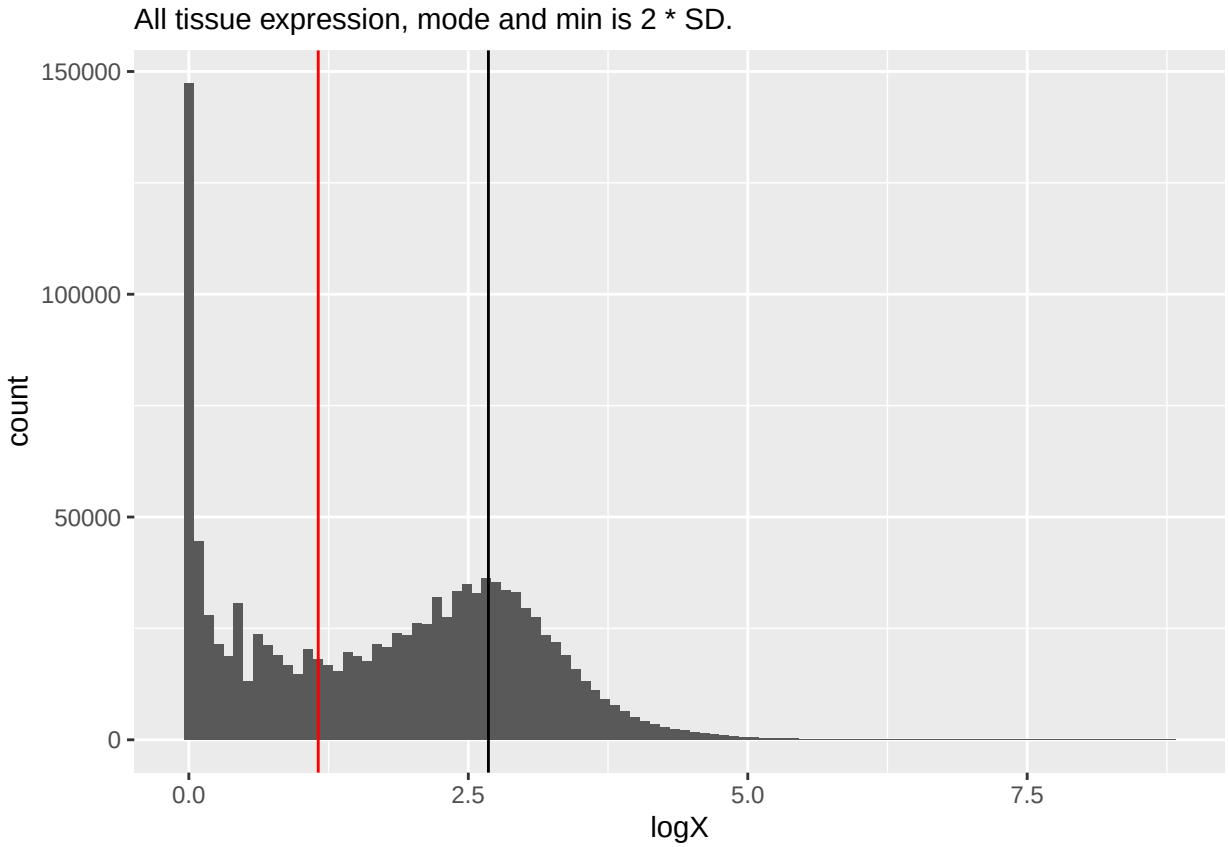
Alternatively, what if we just used all the tissue data combined?

```
mode_sd_all = hpa_expression %>%
  dplyr::filter(logX > 1) %>%
  dplyr::pull(logX) %>%
  calculate_mode_sd()

all_min = mode_sd_all[1] - 2*mode_sd_all[2]
names(all_min) = NULL
all_min
```

```
## [1] 1.157528
```

```
ggplot(hpa_expression, aes(x = logX)) +
  geom_histogram(bins = 100) +
  geom_vline(xintercept = mode_sd_all[1]) +
  geom_vline(xintercept = all_min, color = "red") +
  labs(subtitle = "All tissue expression, mode and min is 2 * SD.")
```



This, this looks extremely robust!