Case study of mutliplex salivary immune data

Michelle Byrne - michelle.byrne@gmail.com (mailto:michelle.byrne@gmail.com)

June 1, 2019

This script includes code for you to synthesise your own multiplex data from your original dataset if you wish to later share it with others without actually sharing your original dataset.

Or, you can use our own synthesized dataset and jump straight to exploring correlations, distributions, etc.

It is a companion to the book chapter by Riis, et al.

- This code assumes you start with *.csv data in wide fomat (i.e., one row per subject, with multiple analytes, each in their own column).
- Also, it assumes you do not have any completely missing (i.e., no sample collected) data, other than values that were determined by the assay to be out of range (OOR). If you do have missing samples, for the purposes of this exercise only, I recommend running a single (EM) imputation. In reality, you would want to check your missing data mechanism and then address missingness using an appropriate method.
- The steps for cleaning the data in this code are for exploratory purposes only in order to see how different cleaning steps affect the data.

Import the raw data

```
# import data
library(readr)
depechemode <- read_csv("smhh_data.csv")

# Idk why there are extra rows in my data, but this removes them. You may need to visually inspe
ct your own csv file if it's doing something funny.
depechemode <- depechemode[-c(37:40), ]

View(depechemode)
```

Options for out of range assay values. These choices will be dependent on your laboratory, the nature of the study (e.g., clinical design, age of sample, etc.). The intention here is to provide a way to examine how these choices affect the data.

Left-censored (i.e., non-detectables, too low for assay to detect)

- OPTION 1: Replace the nondetectables (left-censored) with 0
- OPTION 2: Replace them with the lower limit of assay detection (LLD)
- OPTION 3: Replace them with half the LLD

Right-censored (i.e., too high for assay to detect)

- OPTION 1: Replace with the max of that analyte's distribution (essentially winsorizing)
- OPTION 2: Replace with the upper limit of assay detection
- OPTION 3: Keep as a missing data point, especially if you believe the value was an error rather than a "true" high value. NOTE: In this example, this will delete cases listwise which is probably not appropriate but depends on your missing data mechanism and general philosophy of missing data…

Please note: For the example given here, the assaying software indicated right or left censored data with OOR > or <. Check your output and replace with other text if necessary.

OPTION 1: Replace left-censored with 0 and right-censored with the max of the analyte's distribution

```
depechemode_1 <- depechemode

depechemode_1[depechemode_1=="OOR <"]<-0.001 # I chose 0.001 instead of 0 so as not to cause tro
uble with log transforming later, as above.

depechemode_1$Sal_A2M[depechemode_1$Sal_A2M=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_A2M)))
depechemode_1$Sal_Hapt[depechemode_1$Sal_Hapt=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_Hapt)))
depechemode_1$Sal_CRP[depechemode_1$Sal_CRP=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_CRP)))
depechemode_1$Sal_SAP[depechemode_1$Sal_SAP=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_SAP)))
depechemode_1$Sal_IL2[depechemode_1$Sal_IL2=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_IL2)))
depechemode_1$Sal_IL4[depechemode_1$Sal_IL4=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_IL4)))
depechemode_1$Sal_IL6[depechemode_1$Sal_IL6=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_IL6)))
depechemode_1$Sal_IL8[depechemode_1$Sal_IL8=="OOR >"] <- max(na.omit(as.numeric(depechemode_1$Sa
l_IL8)))
depechemode_1$Sal_IL10[depechemode_1$Sal_IL10=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_IL10)))
depechemode_1$Sal_IL12p70[depechemode_1$Sal_IL12p70=="OOR >"] <- max(na.omit(as.numeric(depechem
ode_1$Sal_IL12p70)))
depechemode_1$Sal_IL13[depechemode_1$Sal_IL13=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_IL13)))
depechemode_1$Sal_IL17[depechemode_1$Sal_IL17=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_IL17)))
depechemode_1$Sal_IFNg[depechemode_1$Sal_IFNg=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_IFNg)))
depechemode_1$Sal_TNFa[depechemode_1$Sal_TNFa=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_TNFa)))
depechemode_1$Sal_IL1a[depechemode_1$Sal_IL1a=="OOR >"] <- max(na.omit(as.numeric(depechemode_1
$Sal_IL1a)))
depechemode_1$Sal_IFNa2[depechemode_1$Sal_IFNa2=="OOR >"] <- max(na.omit(as.numeric(depechemode_
1$Sal_IFNa2)))

depechemode_1$Sal_A2M <- as.numeric(depechemode_1$Sal_A2M)
depechemode_1$Sal_Hapt <- as.numeric(depechemode_1$Sal_Hapt)
depechemode_1$Sal_CRP <- as.numeric(depechemode_1$Sal_CRP)
depechemode_1$Sal_SAP <- as.numeric(depechemode_1$Sal_SAP)
depechemode_1$Sal_IL2 <- as.numeric(depechemode_1$Sal_IL2)
depechemode_1$Sal_IL4 <- as.numeric(depechemode_1$Sal_IL4)
depechemode_1$Sal_IL6 <- as.numeric(depechemode_1$Sal_IL6)
depechemode_1$Sal_IL8 <- as.numeric(depechemode_1$Sal_IL8)
depechemode_1$Sal_IL10 <- as.numeric(depechemode_1$Sal_IL10)
depechemode_1$Sal_IL12p70 <- as.numeric(depechemode_1$Sal_IL12p70)
depechemode_1$Sal_IL13 <- as.numeric(depechemode_1$Sal_IL13)
depechemode_1$Sal_IL17 <- as.numeric(depechemode_1$Sal_IL17)
depechemode_1$Sal_IFNg <- as.numeric(depechemode_1$Sal_IFNg)
depechemode_1$Sal_TNFa <- as.numeric(depechemode_1$Sal_TNFa)
```

```
depechemode_1$Sal_IL1a <- as.numeric(depechemode_1$Sal_IL1a)
depechemode_1$Sal_IFNa2 <- as.numeric(depechemode_1$Sal_IFNa2)
```

Option 2: Replace left and right-censored with lower and upper limits of detection

```
depechemode_2 <- depechemode

#Left-censored:
depechemode_2$Sal_A2M[depechemode_2$Sal_A2M=="OOR <"] <- 0.33 # Change this to YOUR lower limit
 of assay detection for each of your specific immune markers
depechemode_2$Sal_Hapt[depechemode_2$Sal_Hapt=="OOR <"] <- 0.03
depechemode_2$Sal_CRP[depechemode_2$Sal_CRP=="OOR <"] <- 0.01
depechemode_2$Sal_SAP[depechemode_2$Sal_SAP=="OOR <"] <- 0.01
depechemode_2$Sal_IL2[depechemode_2$Sal_IL2=="OOR <"] <- 0.95
depechemode_2$Sal_IL4[depechemode_2$Sal_IL4=="OOR <"] <- 0.22
depechemode_2$Sal_IL6[depechemode_2$Sal_IL6=="OOR <"] <- 1.18
depechemode_2$Sal_IL8[depechemode_2$Sal_IL8=="OOR <"] <- 1.71
depechemode_2$Sal_IL10[depechemode_2$Sal_IL10=="OOR <"] <- 1.71
depechemode_2$Sal_IL12p70[depechemode_2$Sal_IL12p70=="OOR <"] <- 2.21
depechemode_2$Sal_IL13[depechemode_2$Sal_IL13=="OOR <"] <- 1.97
depechemode_2$Sal_IL17[depechemode_2$Sal_IL17=="OOR <"] <- 1.31
depechemode_2$Sal_IFNg[depechemode_2$Sal_IFNg=="OOR <"] <- 1.87
depechemode_2$Sal_TNFa[depechemode_2$Sal_TNFa=="OOR <"] <- 4.62
depechemode_2$Sal_IL1a[depechemode_2$Sal_IL1a=="OOR <"] <- 1.55
depechemode_2$Sal_IFNa2[depechemode_2$Sal_IFNa2=="OOR <"] <-  0.31

#right-censored:
depechemode_2$Sal_A2M[depechemode_2$Sal_A2M=="OOR >"] <- 5341 # Change this to YOUR upper limit
 of assay detection for each of your specific immune markers
depechemode_2$Sal_Hapt[depechemode_2$Sal_Hapt=="OOR >"] <- 478
depechemode_2$Sal_CRP[depechemode_2$Sal_CRP=="OOR >"] <- 101
depechemode_2$Sal_SAP[depechemode_2$Sal_SAP=="OOR >"] <- 172
depechemode_2$Sal_IL2[depechemode_2$Sal_IL2=="OOR >"] <- 15605
depechemode_2$Sal_IL4[depechemode_2$Sal_IL4=="OOR >"] <- 3628
depechemode_2$Sal_IL6[depechemode_2$Sal_IL6=="OOR >"] <- 19412
depechemode_2$Sal_IL8[depechemode_2$Sal_IL8=="OOR >"] <- 27965
depechemode_2$Sal_IL10[depechemode_2$Sal_IL10=="OOR >"] <- 28093
depechemode_2$Sal_IL12p70[depechemode_2$Sal_IL12p70=="OOR >"] <- 36141
depechemode_2$Sal_IL13[depechemode_2$Sal_IL13=="OOR >"] <- 32271
depechemode_2$Sal_IL17[depechemode_2$Sal_IL17=="OOR >"] <- 21505
depechemode_2$Sal_IFNg[depechemode_2$Sal_IFNg=="OOR >"] <- 30646
depechemode_2$Sal_TNFa[depechemode_2$Sal_TNFa=="OOR >"] <- 75618
depechemode_2$Sal_IL1a[depechemode_2$Sal_IL1a=="OOR >"] <- 25358
depechemode_2$Sal_IFNa2[depechemode_2$Sal_IFNa2=="OOR >"] <- 5159

depechemode_2$Sal_A2M <- as.numeric(depechemode_2$Sal_A2M)
depechemode_2$Sal_Hapt <- as.numeric(depechemode_2$Sal_Hapt)
depechemode_2$Sal_CRP <- as.numeric(depechemode_2$Sal_CRP)
depechemode_2$Sal_SAP <- as.numeric(depechemode_2$Sal_SAP)
depechemode_2$Sal_IL2 <- as.numeric(depechemode_2$Sal_IL2)
depechemode_2$Sal_IL4 <- as.numeric(depechemode_2$Sal_IL4)
depechemode_2$Sal_IL6 <- as.numeric(depechemode_2$Sal_IL6)
depechemode_2$Sal_IL8 <- as.numeric(depechemode_2$Sal_IL8)
depechemode_2$Sal_IL10 <- as.numeric(depechemode_2$Sal_IL10)
depechemode_2$Sal_IL12p70 <- as.numeric(depechemode_2$Sal_IL12p70)
depechemode_2$Sal_IL13 <- as.numeric(depechemode_2$Sal_IL13)
depechemode_2$Sal_IL17 <- as.numeric(depechemode_2$Sal_IL17)
depechemode_2$Sal_IFNg <- as.numeric(depechemode_2$Sal_IFNg)
```

```
depechemode_2$Sal_TNFa <- as.numeric(depechemode_2$Sal_TNFa)
depechemode_2$Sal_IL1a <- as.numeric(depechemode_2$Sal_IL1a)
depechemode_2$Sal_IFNa2 <- as.numeric(depechemode_2$Sal_IFNa2)
```

OPTION 3: Replace left-censored with half the LLD and right-censored keep as missing data points.

```
depechemode_3 <- depechemode

#Left-censored:
depechemode_3$Sal_A2M[depechemode_3$Sal_A2M=="OOR <"] <- (0.5*0.33) # Change this to YOUR lower
 limit of assay detection for each of your specific immune markers
depechemode_3$Sal_Hapt[depechemode_3$Sal_Hapt=="OOR <"] <- (0.5*0.03)
depechemode_3$Sal_CRP[depechemode_3$Sal_CRP=="OOR <"] <- (0.5*0.01)
depechemode_3$Sal_SAP[depechemode_3$Sal_SAP=="OOR <"] <- (0.5*0.01)
depechemode_3$Sal_IL2[depechemode_3$Sal_IL2=="OOR <"] <- (0.5*0.95)
depechemode_3$Sal_IL4[depechemode_3$Sal_IL4=="OOR <"] <- (0.5*0.22)
depechemode_3$Sal_IL6[depechemode_3$Sal_IL6=="OOR <"] <- (0.5*1.18)
depechemode_3$Sal_IL8[depechemode_3$Sal_IL8=="OOR <"] <- (0.5*1.71)
depechemode_3$Sal_IL10[depechemode_3$Sal_IL10=="OOR <"] <- (0.5*1.71)
depechemode_3$Sal_IL12p70[depechemode_3$Sal_IL12p70=="OOR <"] <- (0.5*2.21)
depechemode_3$Sal_IL13[depechemode_3$Sal_IL13=="OOR <"] <- (0.5*1.97)
depechemode_3$Sal_IL17[depechemode_3$Sal_IL17=="OOR <"] <- (0.5*1.31)
depechemode_3$Sal_IFNg[depechemode_3$Sal_IFNg=="OOR <"] <- (0.5*1.87)
depechemode_3$Sal_TNFa[depechemode_3$Sal_TNFa=="OOR <"] <- (0.5*4.62)
depechemode_3$Sal_IL1a[depechemode_3$Sal_IL1a=="OOR <"] <- (0.5*1.55)
depechemode_3$Sal_IFNa2[depechemode_3$Sal_IFNa2=="OOR <"] <-  (0.5*0.31)

#right-censored:
depechemode_3$Sal_A2M[depechemode_3$Sal_A2M=="OOR >"] <- NA

depechemode_3$Sal_A2M <- as.numeric(depechemode_3$Sal_A2M)
depechemode_3$Sal_Hapt <- as.numeric(depechemode_3$Sal_Hapt)
depechemode_3$Sal_CRP <- as.numeric(depechemode_3$Sal_CRP)
depechemode_3$Sal_SAP <- as.numeric(depechemode_3$Sal_SAP)
depechemode_3$Sal_IL2 <- as.numeric(depechemode_3$Sal_IL2)
depechemode_3$Sal_IL4 <- as.numeric(depechemode_3$Sal_IL4)
depechemode_3$Sal_IL6 <- as.numeric(depechemode_3$Sal_IL6)
depechemode_3$Sal_IL8 <- as.numeric(depechemode_3$Sal_IL8)
depechemode_3$Sal_IL10 <- as.numeric(depechemode_3$Sal_IL10)
depechemode_3$Sal_IL12p70 <- as.numeric(depechemode_3$Sal_IL12p70)
depechemode_3$Sal_IL13 <- as.numeric(depechemode_3$Sal_IL13)
depechemode_3$Sal_IL17 <- as.numeric(depechemode_3$Sal_IL17)
depechemode_3$Sal_IFNg <- as.numeric(depechemode_3$Sal_IFNg)
depechemode_3$Sal_TNFa <- as.numeric(depechemode_3$Sal_TNFa)
depechemode_3$Sal_IL1a <- as.numeric(depechemode_3$Sal_IL1a)
depechemode_3$Sal_IFNa2 <- as.numeric(depechemode_3$Sal_IFNa2)
```

Synthesise

Synthpop synthesises the original dataset (depechemode) and creates a list (kraftwerk), which includes the synthesised dataset (neworder). I've set the seed so I can replicate everything exactly each time I run this (you can't because you don't have the original data), but if you use this with your own data, and you want a new dataset

every time, remove the my.seed stuff. Also, if you are using your own data, make sure to change the OOR values first, as above.

```
library("synthpop")
my.seed <- 1337
kraftwerk_1 <- syn(depechemode_1, seed = my.seed)
```

```
## syn  variables
## 1    Sex Group Age at Ax Sal_A2M Sal_Hapt Sal_CRP Sal_SAP Sal_IL2 Sal_IL4 Sal_IL6
##      Sal_IL8 Sal_IL10 Sal_IL12p70 Sal_IL13 Sal_IL17 Sal_IFNg Sal_TNFa Sal_IL1a Sal_IFNa2
```

```
neworder_1 <- kraftwerk_1$syn

kraftwerk_2 <- syn(depechemode_2, seed = my.seed)
```

```
## syn  variables
## 1    Sex Group Age at Ax Sal_A2M Sal_Hapt Sal_CRP Sal_SAP Sal_IL2 Sal_IL4 Sal_IL6
##      Sal_IL8 Sal_IL10 Sal_IL12p70 Sal_IL13 Sal_IL17 Sal_IFNg Sal_TNFa Sal_IL1a Sal_IFNa2
```

```
neworder_2 <- kraftwerk_2$syn

kraftwerk_3 <- syn(depechemode_3, seed = my.seed)
```

```
## syn  variables
## 1    Sex Group Age at Ax Sal_A2M Sal_Hapt Sal_CRP Sal_SAP Sal_IL2 Sal_IL4 Sal_IL6
##      Sal_IL8 Sal_IL10 Sal_IL12p70 Sal_IL13 Sal_IL17 Sal_IFNg Sal_TNFa Sal_IL1a Sal_IFNa2
```

```
neworder_3 <- kraftwerk_3$syn
```

Transforming immune variables

- Normally we would check the skew and kurtosis, and visually inspect the data as we do later, BEFORE transforming anything. See [github link coming soon] for code on exploring and reporting this. For example, you may not need to transform anything if it is not skewed, or if your immune markers are not the outcome (dependent) variables. Other types of transformations, such as square root, may be more appropriate than log transformation.
- In this example, we'll just go ahead and transform everything so we can see how it changes correlations and distributions.
- Also, another thing we are NOT doing here is correcting for flow rate. Some markers (like SIgA) require this.

```
# First, in case your results had any values rounded down to zero, you'll need to replace these
 (I chose 0.001) in order to log transform them:
neworder_1$Sal_A2M[neworder_1$Sal_A2M == 0] <- 0.001
neworder_1$Sal_Hapt[neworder_1$Sal_Hapt == 0] <- 0.001
neworder_1$Sal_CRP[neworder_1$Sal_CRP == 0] <- 0.001
neworder_1$Sal_SAP[neworder_1$Sal_SAP == 0] <- 0.001
neworder_1$Sal_IL2[neworder_1$Sal_IL2 == 0] <- 0.001
neworder_1$Sal_IL4[neworder_1$Sal_IL4 == 0] <- 0.001
neworder_1$Sal_IL6[neworder_1$Sal_IL6 == 0] <- 0.001
neworder_1$Sal_IL8[neworder_1$Sal_IL8 == 0] <- 0.001
neworder_1$Sal_IL10[neworder_1$Sal_IL10 == 0] <- 0.001
neworder_1$Sal_IL12p70[neworder_1$Sal_IL12p70 == 0] <- 0.001
neworder_1$Sal_IL13[neworder_1$Sal_IL13 == 0] <- 0.001
neworder_1$Sal_IL17[neworder_1$Sal_IL17 == 0] <- 0.001
neworder_1$Sal_IFNg[neworder_1$Sal_IFNg == 0] <- 0.001
neworder_1$Sal_TNFa[neworder_1$Sal_TNFa == 0] <- 0.001
neworder_1$Sal_IL1a[neworder_1$Sal_IL1a == 0] <- 0.001
neworder_1$Sal_IFNa2[neworder_1$Sal_IFNa2 == 0] <- 0.001

neworder_2$Sal_A2M[neworder_2$Sal_A2M == 0] <- 0.001
neworder_2$Sal_Hapt[neworder_2$Sal_Hapt == 0] <- 0.001
neworder_2$Sal_CRP[neworder_2$Sal_CRP == 0] <- 0.001
neworder_2$Sal_SAP[neworder_2$Sal_SAP == 0] <- 0.001
neworder_2$Sal_IL2[neworder_2$Sal_IL2 == 0] <- 0.001
neworder_2$Sal_IL4[neworder_2$Sal_IL4 == 0] <- 0.001
neworder_2$Sal_IL6[neworder_2$Sal_IL6 == 0] <- 0.001
neworder_2$Sal_IL8[neworder_2$Sal_IL8 == 0] <- 0.001
neworder_2$Sal_IL10[neworder_2$Sal_IL10 == 0] <- 0.001
neworder_2$Sal_IL12p70[neworder_2$Sal_IL12p70 == 0] <- 0.001
neworder_2$Sal_IL13[neworder_2$Sal_IL13 == 0] <- 0.001
neworder_2$Sal_IL17[neworder_2$Sal_IL17 == 0] <- 0.001
neworder_2$Sal_IFNg[neworder_2$Sal_IFNg == 0] <- 0.001
neworder_2$Sal_TNFa[neworder_2$Sal_TNFa == 0] <- 0.001
neworder_2$Sal_IL1a[neworder_2$Sal_IL1a == 0] <- 0.001
neworder_2$Sal_IFNa2[neworder_2$Sal_IFNa2 == 0] <- 0.001

neworder_3$Sal_A2M[neworder_3$Sal_A2M == 0] <- 0.001
neworder_3$Sal_Hapt[neworder_3$Sal_Hapt == 0] <- 0.001
neworder_3$Sal_CRP[neworder_3$Sal_CRP == 0] <- 0.001
neworder_3$Sal_SAP[neworder_3$Sal_SAP == 0] <- 0.001
neworder_3$Sal_IL2[neworder_3$Sal_IL2 == 0] <- 0.001
neworder_3$Sal_IL4[neworder_3$Sal_IL4 == 0] <- 0.001
neworder_3$Sal_IL6[neworder_3$Sal_IL6 == 0] <- 0.001
neworder_3$Sal_IL8[neworder_3$Sal_IL8 == 0] <- 0.001
neworder_3$Sal_IL10[neworder_3$Sal_IL10 == 0] <- 0.001
neworder_3$Sal_IL12p70[neworder_3$Sal_IL12p70 == 0] <- 0.001
neworder_3$Sal_IL13[neworder_3$Sal_IL13 == 0] <- 0.001
neworder_3$Sal_IL17[neworder_3$Sal_IL17 == 0] <- 0.001
neworder_3$Sal_IFNg[neworder_3$Sal_IFNg == 0] <- 0.001
neworder_3$Sal_TNFa[neworder_3$Sal_TNFa == 0] <- 0.001
neworder_3$Sal_IL1a[neworder_3$Sal_IL1a == 0] <- 0.001
neworder_3$Sal_IFNa2[neworder_3$Sal_IFNa2 == 0] <- 0.001
```

```
# Then, just create new variables in neworder that are all the analytes, natural log transformed
neworder_1$ln_a2m <- lapply(neworder_1$Sal_A2M, log)
neworder_1$ln_hapt <- lapply(neworder_1$Sal_Hapt, log)
neworder_1$ln_crp <- lapply(neworder_1$Sal_CRP, log)
neworder_1$ln_sap <- lapply(neworder_1$Sal_SAP, log)
neworder_1$ln_il2 <- lapply(neworder_1$Sal_IL2, log)
neworder_1$ln_il4 <- lapply(neworder_1$Sal_IL4, log)
neworder_1$ln_il6 <- lapply(neworder_1$Sal_IL6, log)
neworder_1$ln_il8 <- lapply(neworder_1$Sal_IL8, log)
neworder_1$ln_il10 <- lapply(neworder_1$Sal_IL10, log)
neworder_1$ln_il12p70 <- lapply(neworder_1$Sal_IL12p70, log)
neworder_1$ln_il13 <- lapply(neworder_1$Sal_IL13, log)
neworder_1$ln_il17 <- lapply(neworder_1$Sal_IL17, log)
neworder_1$ln_ifng <- lapply(neworder_1$Sal_IFNg, log)
neworder_1$ln_tnfa <- lapply(neworder_1$Sal_TNFa, log)
neworder_1$ln_il1a <- lapply(neworder_1$Sal_IL1a, log)
neworder_1$ln_ifna2 <- lapply(neworder_1$Sal_IFNa2, log)

neworder_2$ln_a2m <- lapply(neworder_2$Sal_A2M, log)
neworder_2$ln_hapt <- lapply(neworder_2$Sal_Hapt, log)
neworder_2$ln_crp <- lapply(neworder_2$Sal_CRP, log)
neworder_2$ln_sap <- lapply(neworder_2$Sal_SAP, log)
neworder_2$ln_il2 <- lapply(neworder_2$Sal_IL2, log)
neworder_2$ln_il4 <- lapply(neworder_2$Sal_IL4, log)
neworder_2$ln_il6 <- lapply(neworder_2$Sal_IL6, log)
neworder_2$ln_il8 <- lapply(neworder_2$Sal_IL8, log)
neworder_2$ln_il10 <- lapply(neworder_2$Sal_IL10, log)
neworder_2$ln_il12p70 <- lapply(neworder_2$Sal_IL12p70, log)
neworder_2$ln_il13 <- lapply(neworder_2$Sal_IL13, log)
neworder_2$ln_il17 <- lapply(neworder_2$Sal_IL17, log)
neworder_2$ln_ifng <- lapply(neworder_2$Sal_IFNg, log)
neworder_2$ln_tnfa <- lapply(neworder_2$Sal_TNFa, log)
neworder_2$ln_il1a <- lapply(neworder_2$Sal_IL1a, log)
neworder_2$ln_ifna2 <- lapply(neworder_2$Sal_IFNa2, log)

neworder_3$ln_a2m <- lapply(neworder_3$Sal_A2M, log)
neworder_3$ln_hapt <- lapply(neworder_3$Sal_Hapt, log)
neworder_3$ln_crp <- lapply(neworder_3$Sal_CRP, log)
neworder_3$ln_sap <- lapply(neworder_3$Sal_SAP, log)
neworder_3$ln_il2 <- lapply(neworder_3$Sal_IL2, log)
neworder_3$ln_il4 <- lapply(neworder_3$Sal_IL4, log)
neworder_3$ln_il6 <- lapply(neworder_3$Sal_IL6, log)
neworder_3$ln_il8 <- lapply(neworder_3$Sal_IL8, log)
neworder_3$ln_il10 <- lapply(neworder_3$Sal_IL10, log)
neworder_3$ln_il12p70 <- lapply(neworder_3$Sal_IL12p70, log)
neworder_3$ln_il13 <- lapply(neworder_3$Sal_IL13, log)
neworder_3$ln_il17 <- lapply(neworder_3$Sal_IL17, log)
neworder_3$ln_ifng <- lapply(neworder_3$Sal_IFNg, log)
neworder_3$ln_tnfa <- lapply(neworder_3$Sal_TNFa, log)
neworder_3$ln_il1a <- lapply(neworder_3$Sal_IL1a, log)
neworder_3$ln_ifna2 <- lapply(neworder_3$Sal_IFNa2, log)
```

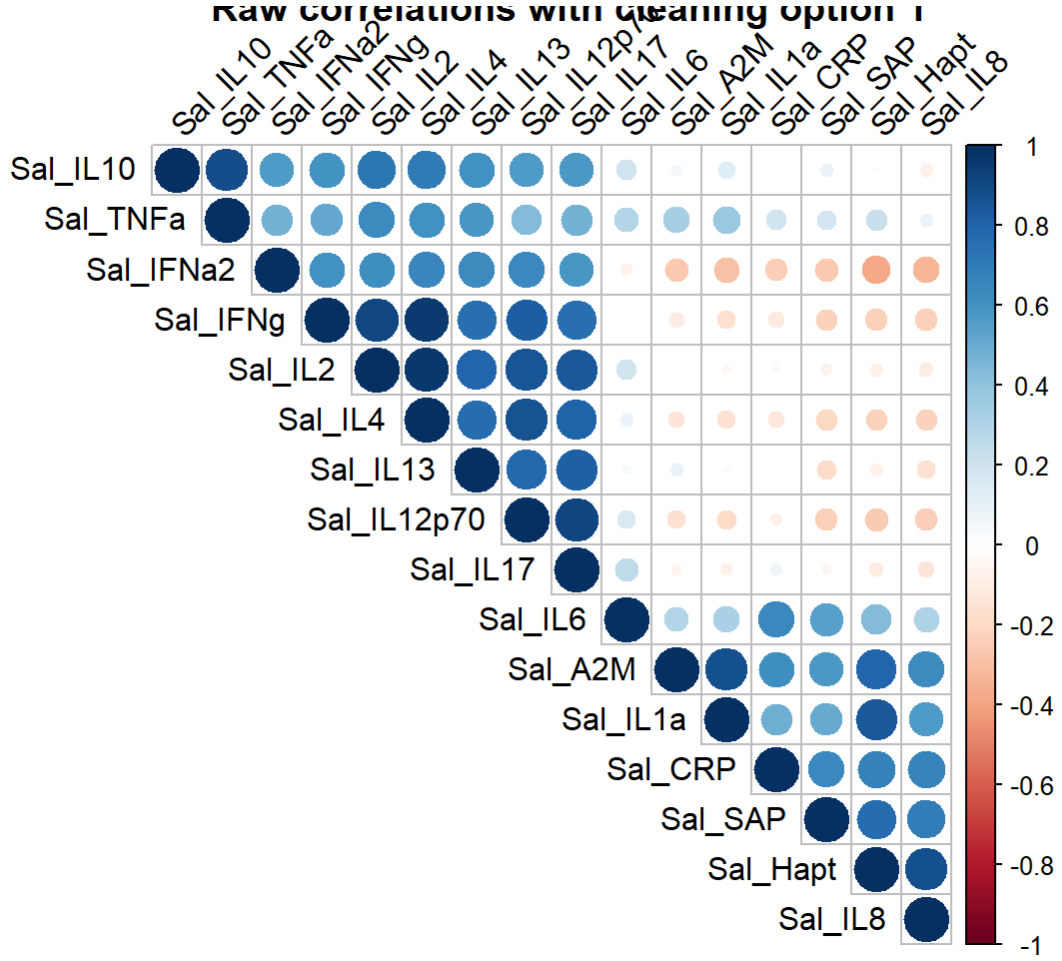
Now have fun exploring the synthesised dataset of multiplex data (neworder)

First explore correlations amongst raw immune markers (and sex, age, and group). I've set the corrplot so you can easily see which markers cluster together (hclust):

- You'll probably find that many of your multiplex analytes are very colinear. What should this mean for our a priori hypotheses and/or data reduction?
- How do the correlations change depending on the cleaning option?

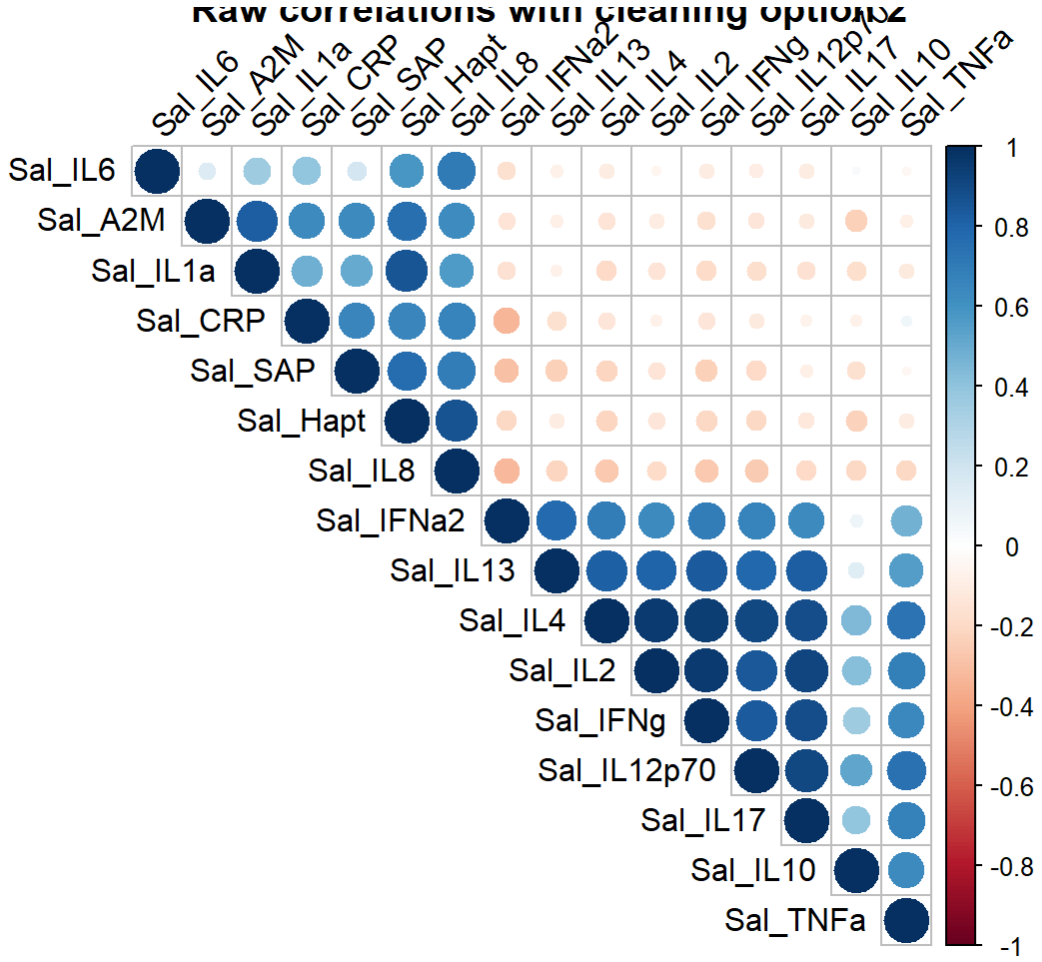
OPTION 1 Correlations (Left-censored as 0 and right-censored as max)

```
# This first corrplot is only going to compare the raw salivary markers - in my dataset, columns
4:19 (you'll have to change this in your own dataset)

library(Hmisc)
library(corrplot)

laroux_1 <- rcorr(as.matrix(neworder_1[ , c(4:19)])) # change to the columns that have the raw m
arkers
corrplot(laroux_1$r, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, title = "Raw correlations with cleaning option 1")
```



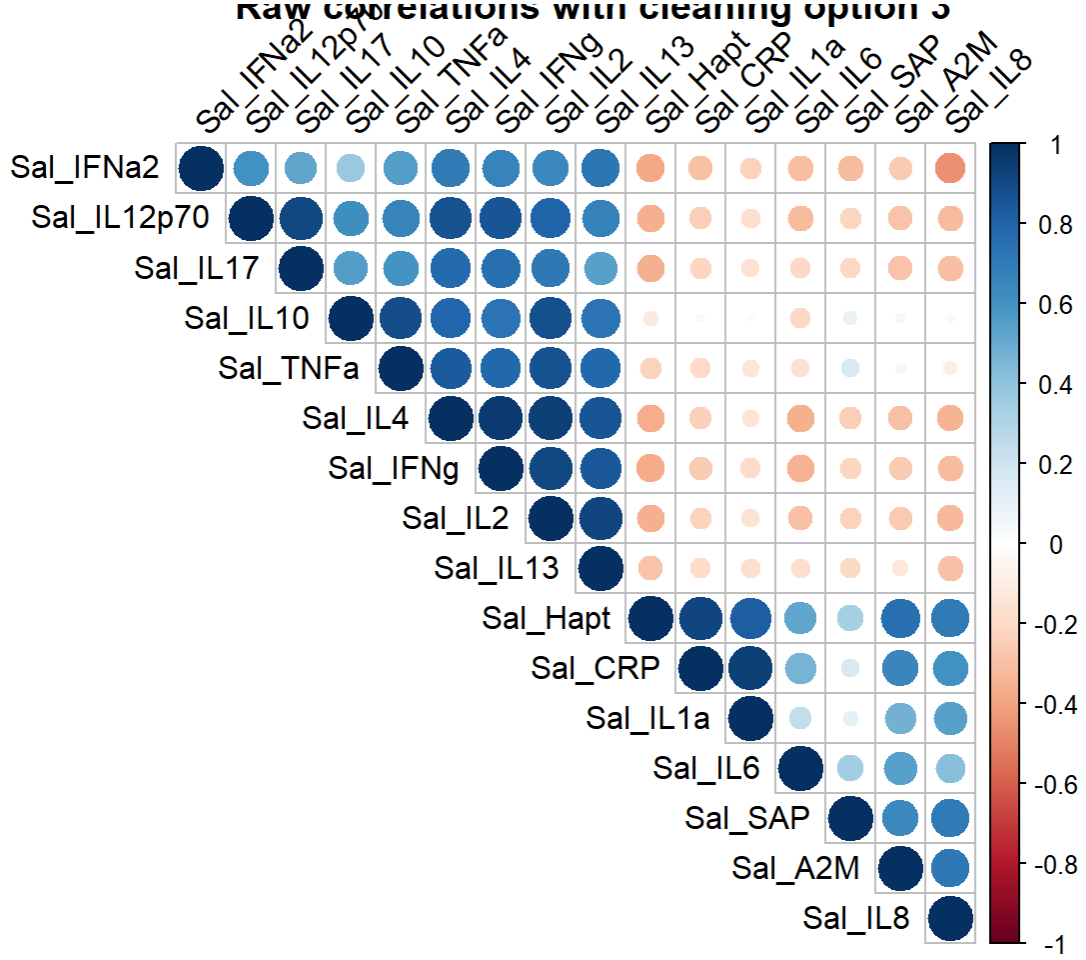
OPTION 2 Correlations (Left-censored as LLD and right-censored as ULD)

```
laroux_2 <- rcorr(as.matrix(neworder_2[ , c(4:19)])) # change to the columns that have the raw m
arkers
corrplot(laroux_2$r, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, title = "Raw correlations with cleaning option 2")
```



OPTION 3 Correlations (Left-censored as 1/2 LLD and right-censored missing)

```
laroux_3 <- rcorr(as.matrix(neworder_3[ , c(4:19)])) # change to the columns that have the raw m
arkers
corrplot(laroux_3$r, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, title = "Raw correlations with cleaning option 3")
```

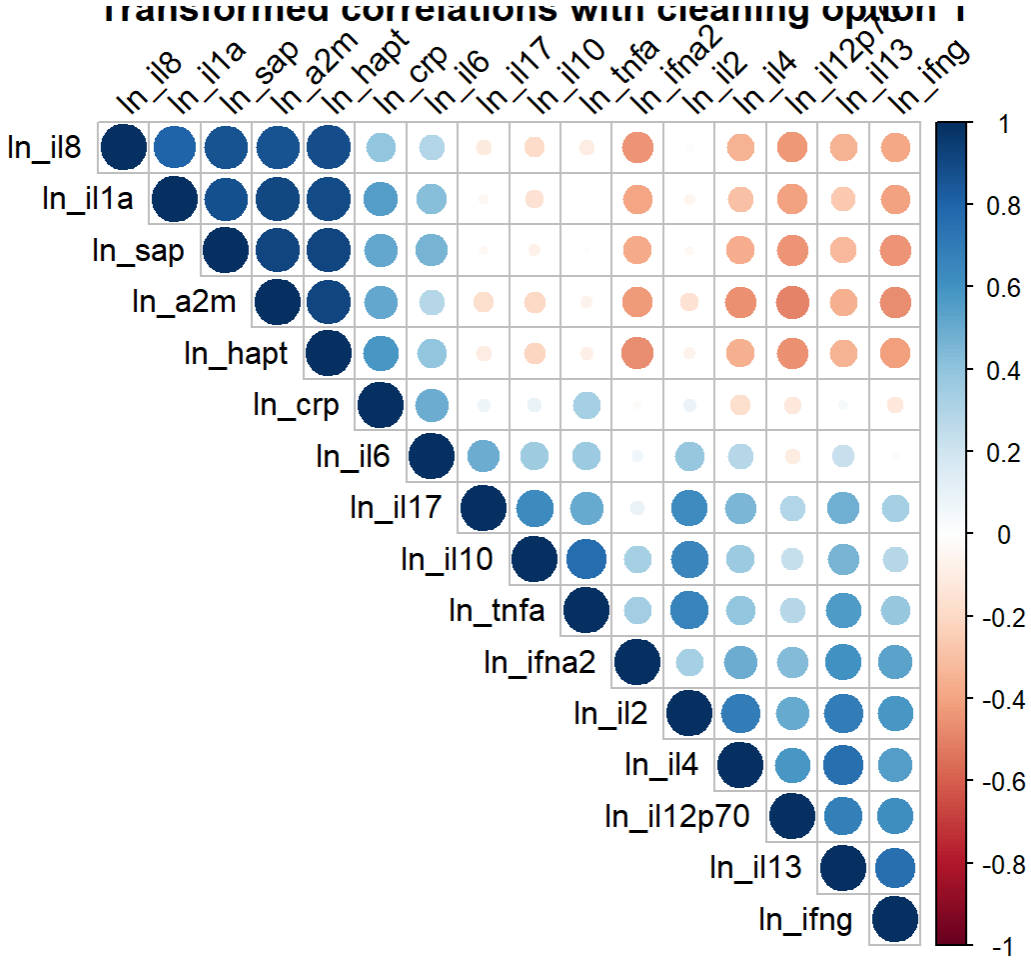


Now see how transforming the immune markers changes the correlations.

- What do these tell us about how we should be structuring our models/hypotheses with multiplex data? Should we be wary of multiple comparisons?

OPTION 1: Left-censored as 0 and right-censored as max

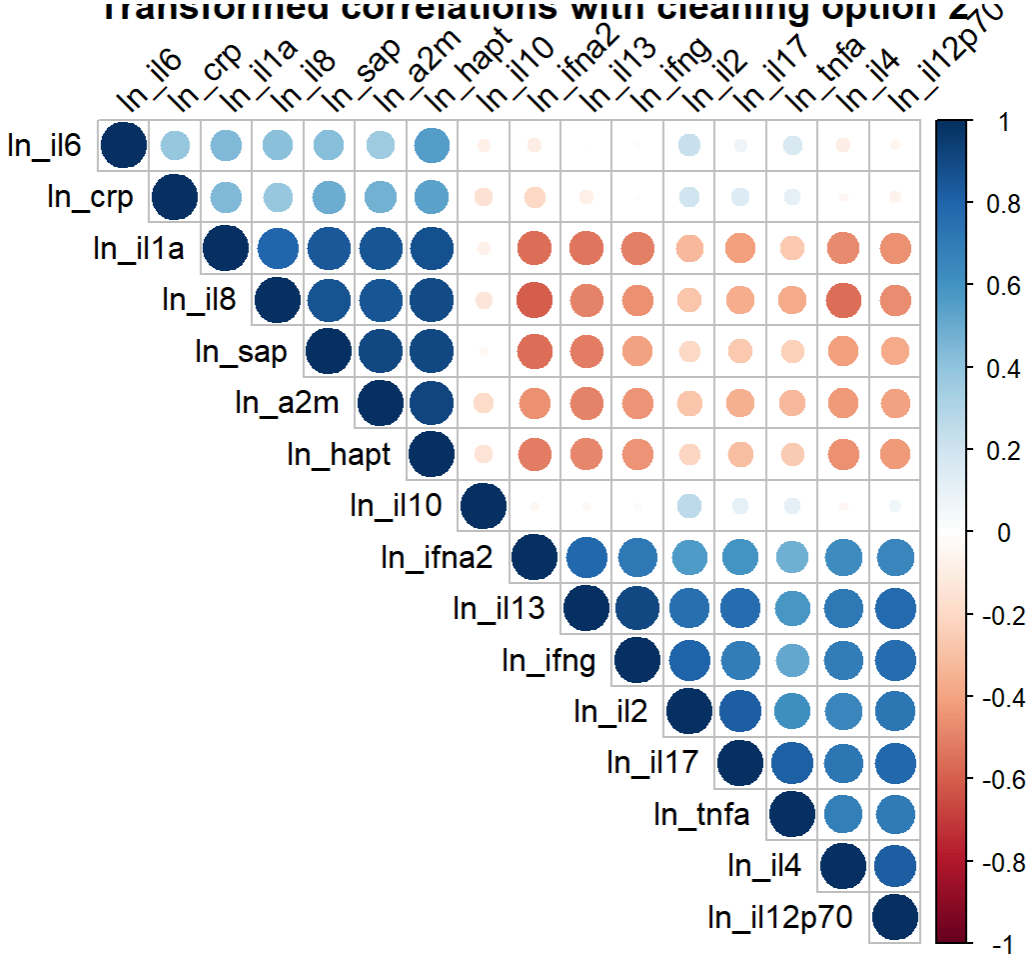
```
eurythmics_1 <- rcorr(as.matrix(neworder_1[ , c(20:35)])) # change to the columns that have your transformed markers
corrplot(eurythmics_1$r, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, title = "Transformed correlations with cleaning option 1")
```




```
# maybe some smart person can find a way to put the markers in the same order as the above so yo
u can easily see which correlations changed (I could also just keep them alphabetical instead of
clustering them)
```

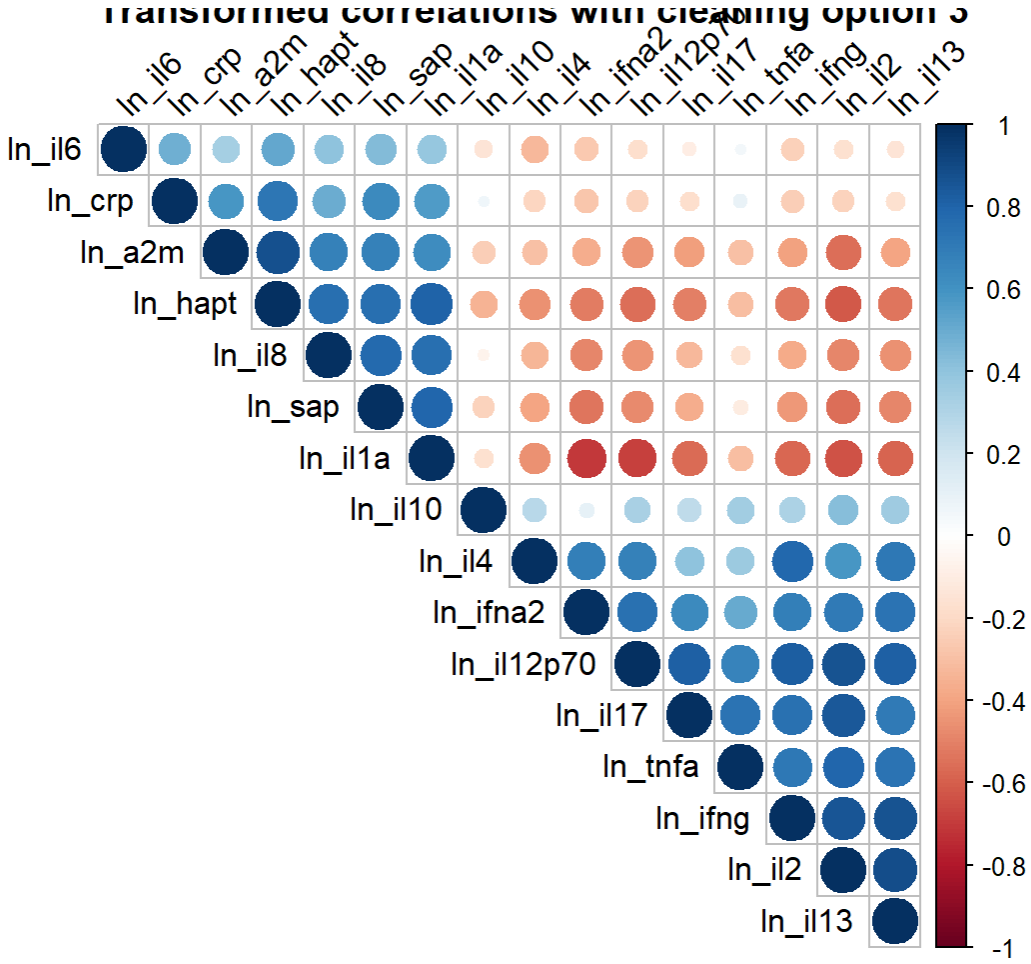
OPTION 2: Left-censored as LLD and right-censored as ULD

```
eurythmics_2 <- rcorr(as.matrix(neworder_2[ , c(20:35)])) # change to the columns that have your
transformed markers
corrplot(eurythmics_2$r, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, title = "Transformed correlations with cleaning option
 2")
```



OPTION 3: Left-censored as 1/2 LLD and right-censored as missing

```
eurythmics_3 <- rcorr(as.matrix(neworder_3[ , c(20:35)])) # change to the columns that have your transformed markers
corrplot(eurythmics_3$r, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45, title = "Transformed correlations with cleaning option 3")
```



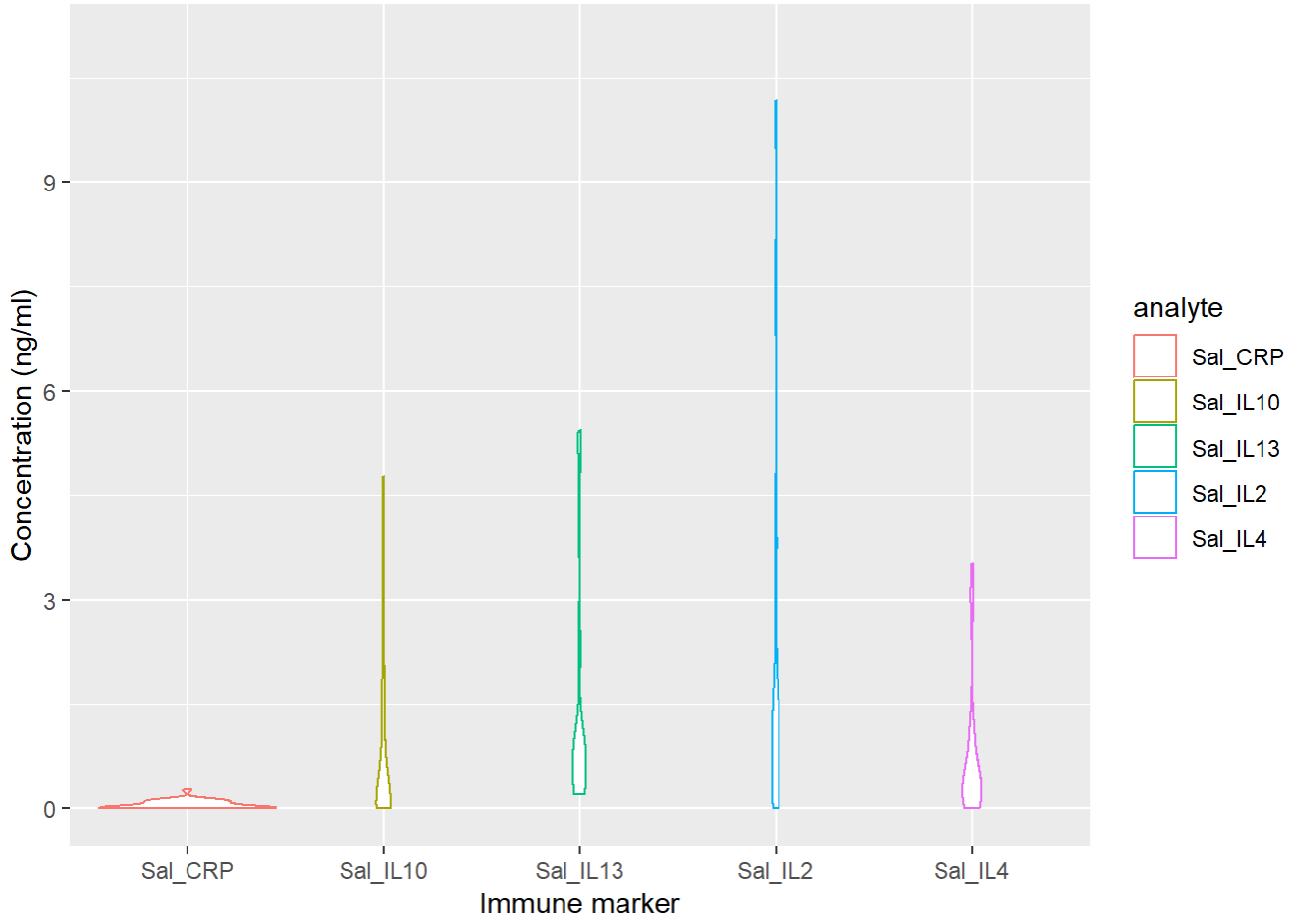
Explore the distribution of (some of the) raw immune markers and compare them using violin plots (https://medium.com/@bioturing/5-reasons-you-should-use-a-violin-graph-31a9cdf2d0c6)

OPTION 1: Left-censored as 0 and right-censored as max

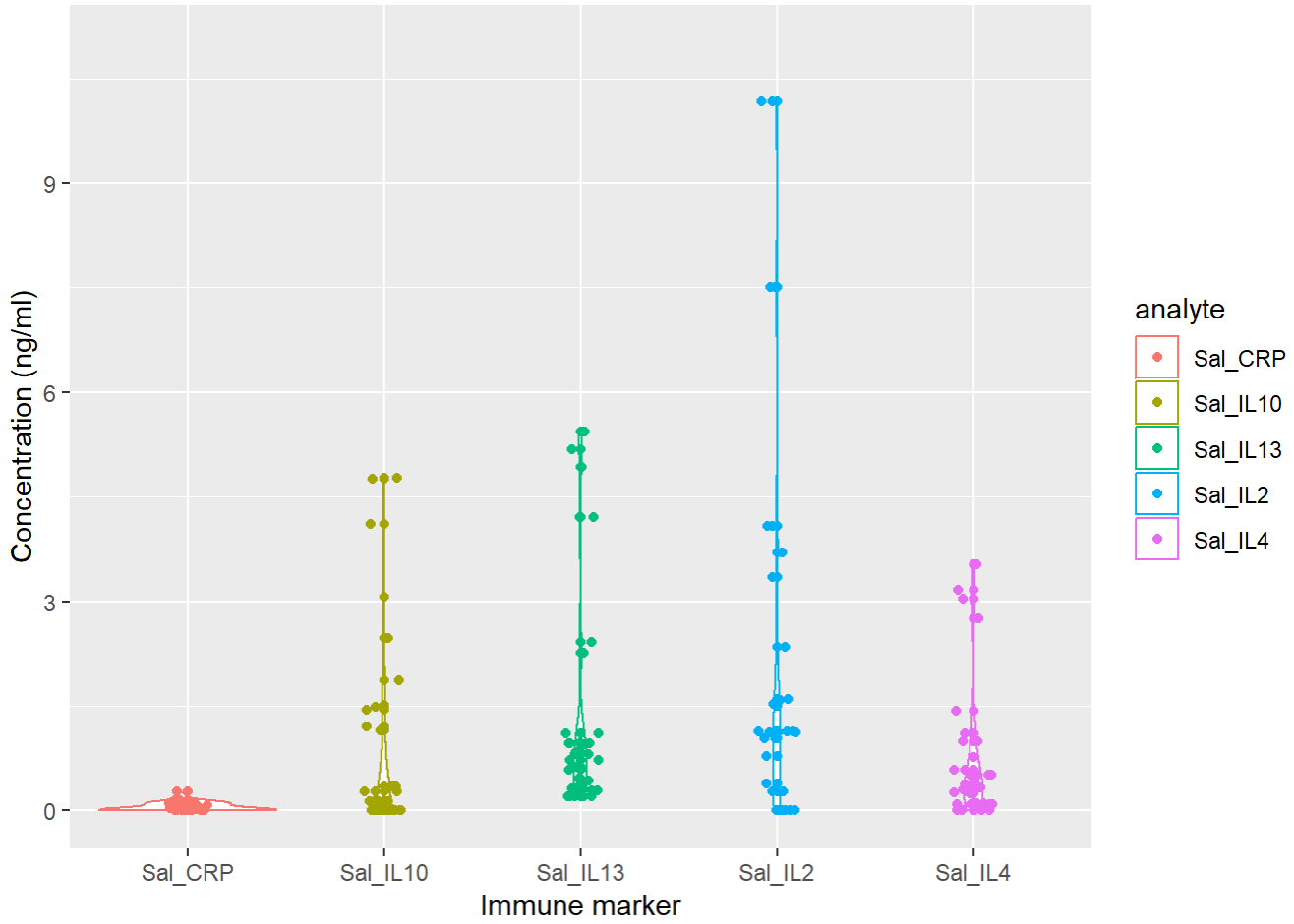
```
library(tidyverse)
neworder_1 <- tibble::rowid_to_column(neworder_1, "ID")

# first do the distributions for the raw immune markers (but only a few because the distribution
s are terrible)

# change to long format with only two columns (after demographics): Analyte (which analyte it i
s) and Concentration (the value in ng/ml, or whatever yours is in)
# Here I have only included a FEW analytes as an example because the distributions for some were
too extreme, see comment below.
petshop_1 <- gather(neworder_1, "analyte", "concentration", Sal_CRP, Sal_IL10, Sal_IL13,Sal_IL2,
Sal_IL4)
petshop_1$analyte <- as.factor(petshop_1$analyte)
levels(petshop_1$analyte)
```

```
## [1] "Sal_CRP"  "Sal_IL10" "Sal_IL13" "Sal_IL2"  "Sal_IL4"
```

```
# now finally some violin plots.
# NOTE - you may have to play around to set the ylim. If you have one or two markers with a coup
le extreme raw values, you won't see anything.
# Don't worry, next we are going to clean (transform) the data a bit and then examine the distri
butions o ALL the analytes side by side

# First, plain violin plots:
ggplot(petshop_1, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0, 11) +
  geom_violin() +
  labs(
    x="Immune marker", y = "Concentration (ng/ml)")
```




```
# With data superimposed:
ggplot(petshop_1, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0, 11) +
  geom_violin() +
  geom_jitter(height = 0, width = .1) +
  geom_point() +
  labs(
    x="Immune marker", y = "Concentration (ng/ml)")
```

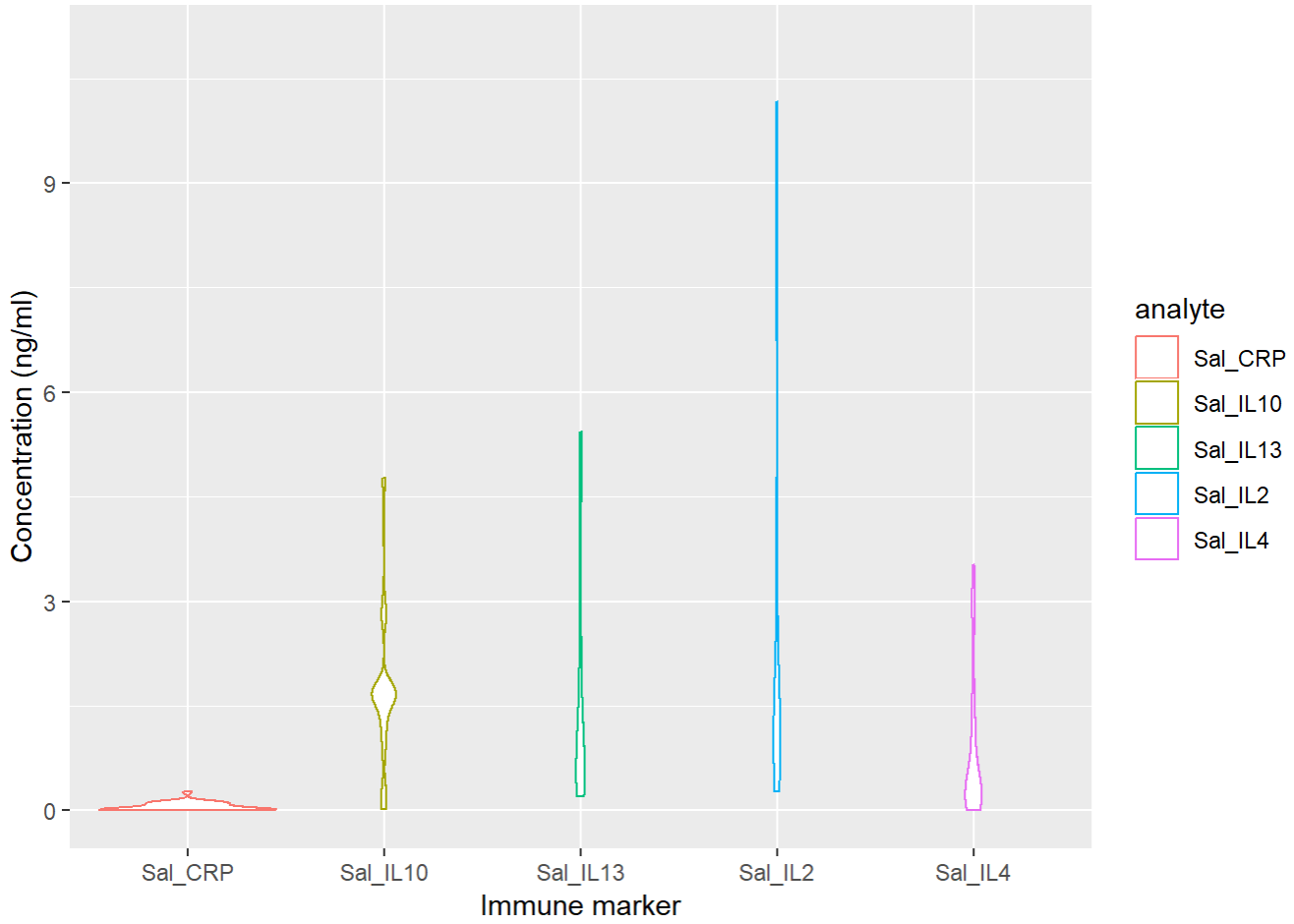


OPTION 2: Left-censored as LLD and right-censored as ULD

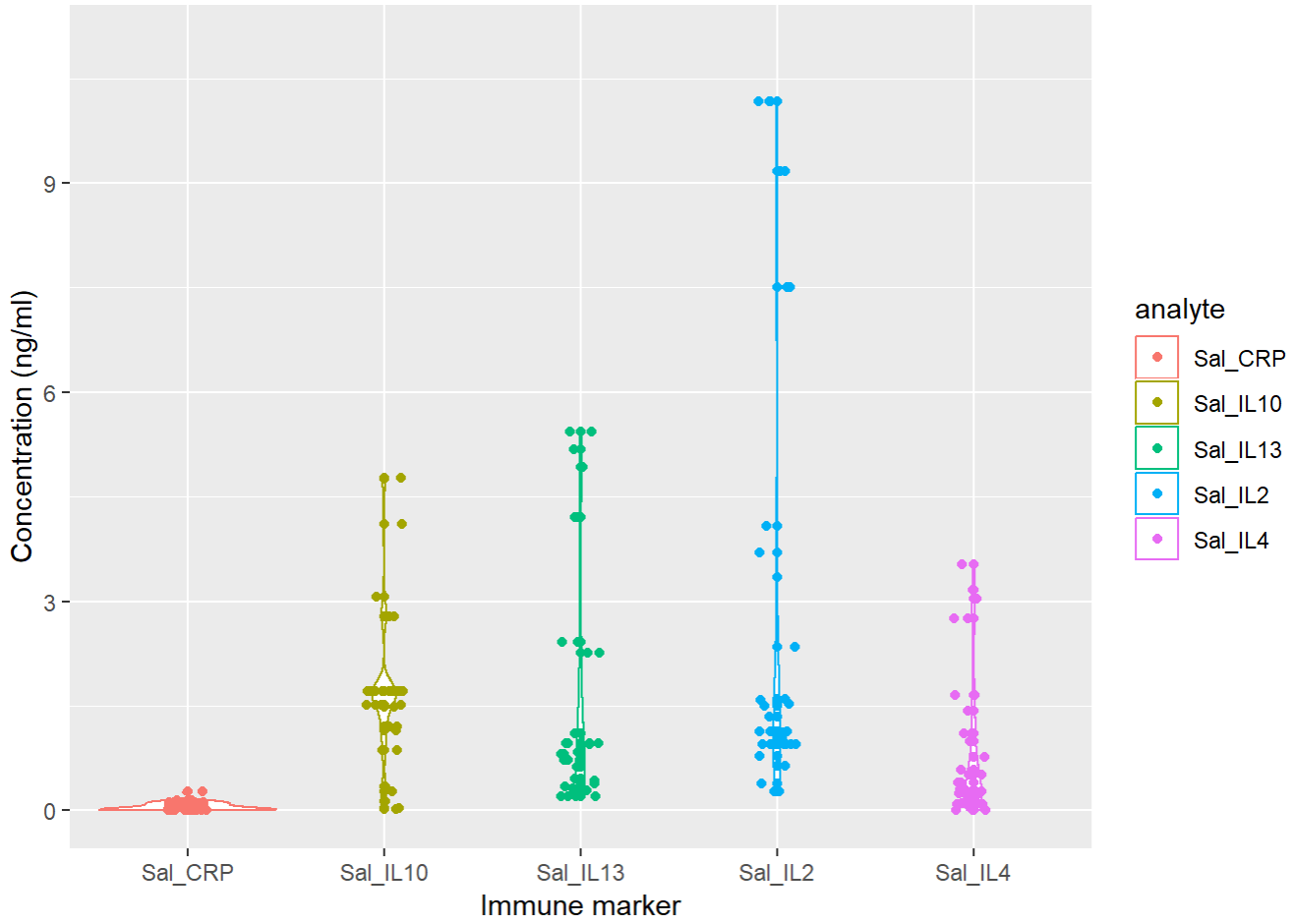
```
# add IDs
neworder_2 <- tibble::rowid_to_column(neworder_2, "ID")

# change to long format with only two columns (after demographics):
petshop_2 <- gather(neworder_2, "analyte", "concentration", Sal_CRP, Sal_IL10, Sal_IL13,Sal_IL2,
Sal_IL4)
petshop_2$analyte <- as.factor(petshop_2$analyte)
levels(petshop_2$analyte)
```

```
## [1] "Sal_CRP"  "Sal_IL10" "Sal_IL13" "Sal_IL2"  "Sal_IL4"
```

```
# Plain violin plots:
ggplot(petshop_2, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0, 11) +
  geom_violin() +
  labs(
    x="Immune marker", y = "Concentration (ng/ml)")
```




```
# With data superimposed
ggplot(petshop_2, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0, 11) +
  geom_violin() +
  geom_jitter(height = 0, width = .1) +
  geom_point() +
  labs(
    x="Immune marker", y = "Concentration (ng/ml)")
```

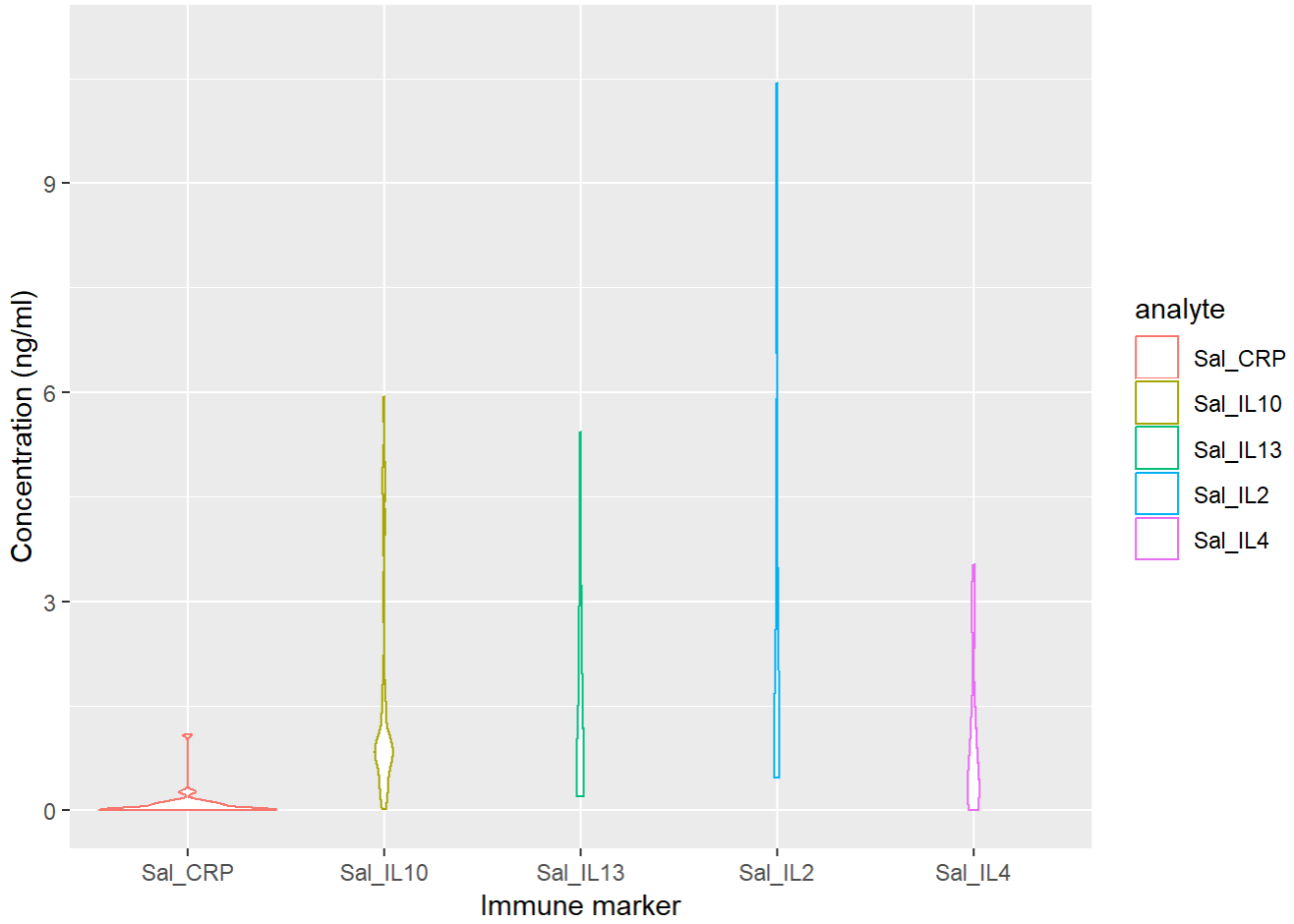


OPTION 3: Left-censored as 1/2 LLD and right-censored as missing

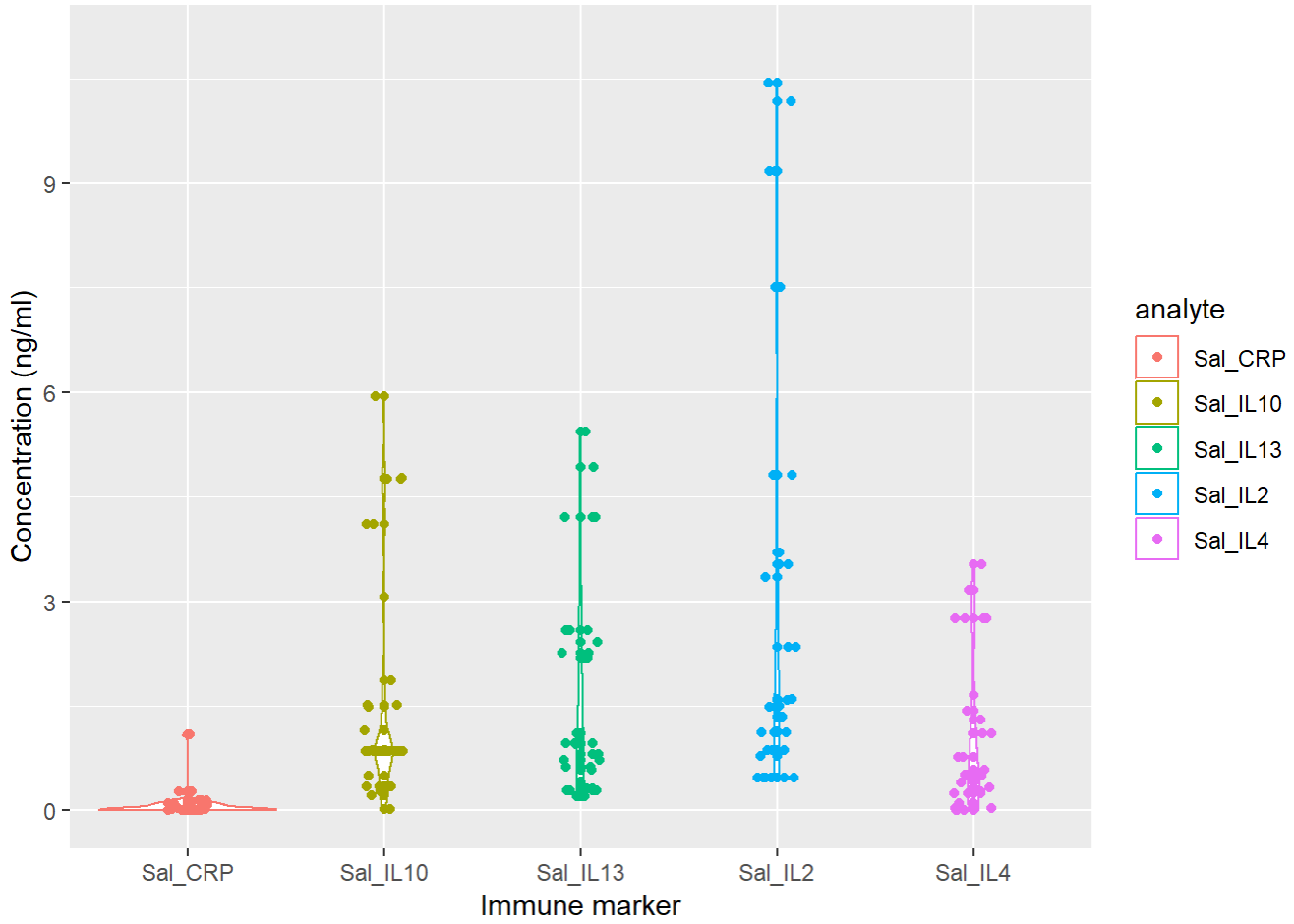
```
# add IDs
neworder_3 <- tibble::rowid_to_column(neworder_3, "ID")

# change to long format with only two columns (after demographics):
petshop_3 <- gather(neworder_3, "analyte", "concentration", Sal_CRP, Sal_IL10, Sal_IL13,Sal_IL2,
Sal_IL4)
petshop_3$analyte <- as.factor(petshop_3$analyte)
levels(petshop_3$analyte)
```

```
## [1] "Sal_CRP"  "Sal_IL10" "Sal_IL13" "Sal_IL2"  "Sal_IL4"
```

```
# Plain violin plots
ggplot(petshop_3, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0, 11) +
  geom_violin() +
  #geom_jitter(height = 0, width = .1) +
  #geom_point() +
  labs(
    x="Immune marker", y = "Concentration (ng/ml)")
```




```
# With data superimposed
ggplot(petshop_3, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0, 11) +
  geom_violin() +
  geom_jitter(height = 0, width = .1) +
  geom_point() +
  labs(
    x="Immune marker", y = "Concentration (ng/ml)")
```

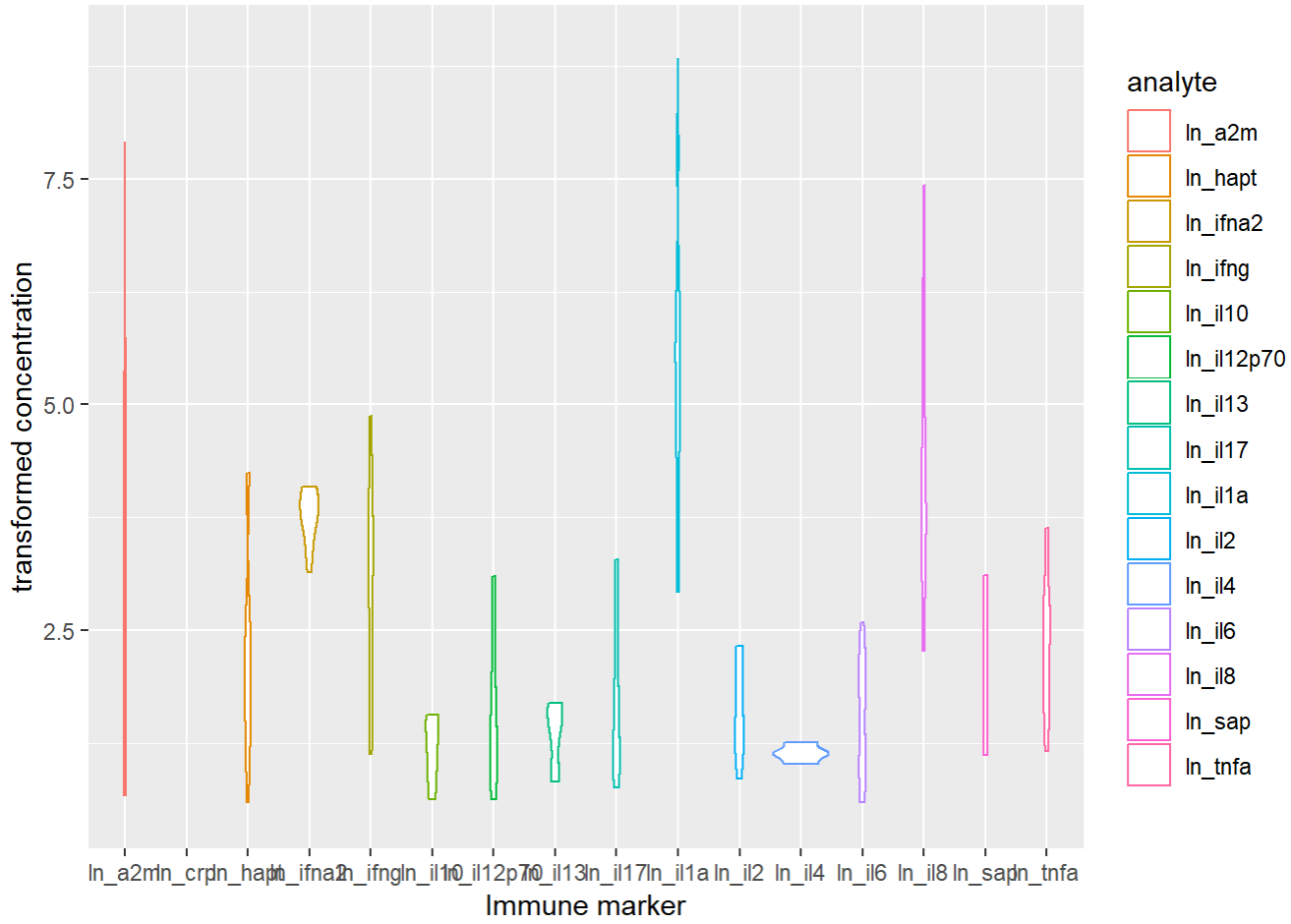


Next take a look at the distributions for the ln transformed markers. First, you'll notice that you can actually fit all of them in on the same plot because the scaling is more similar. First, we'll just do this for cleaning Option #1.

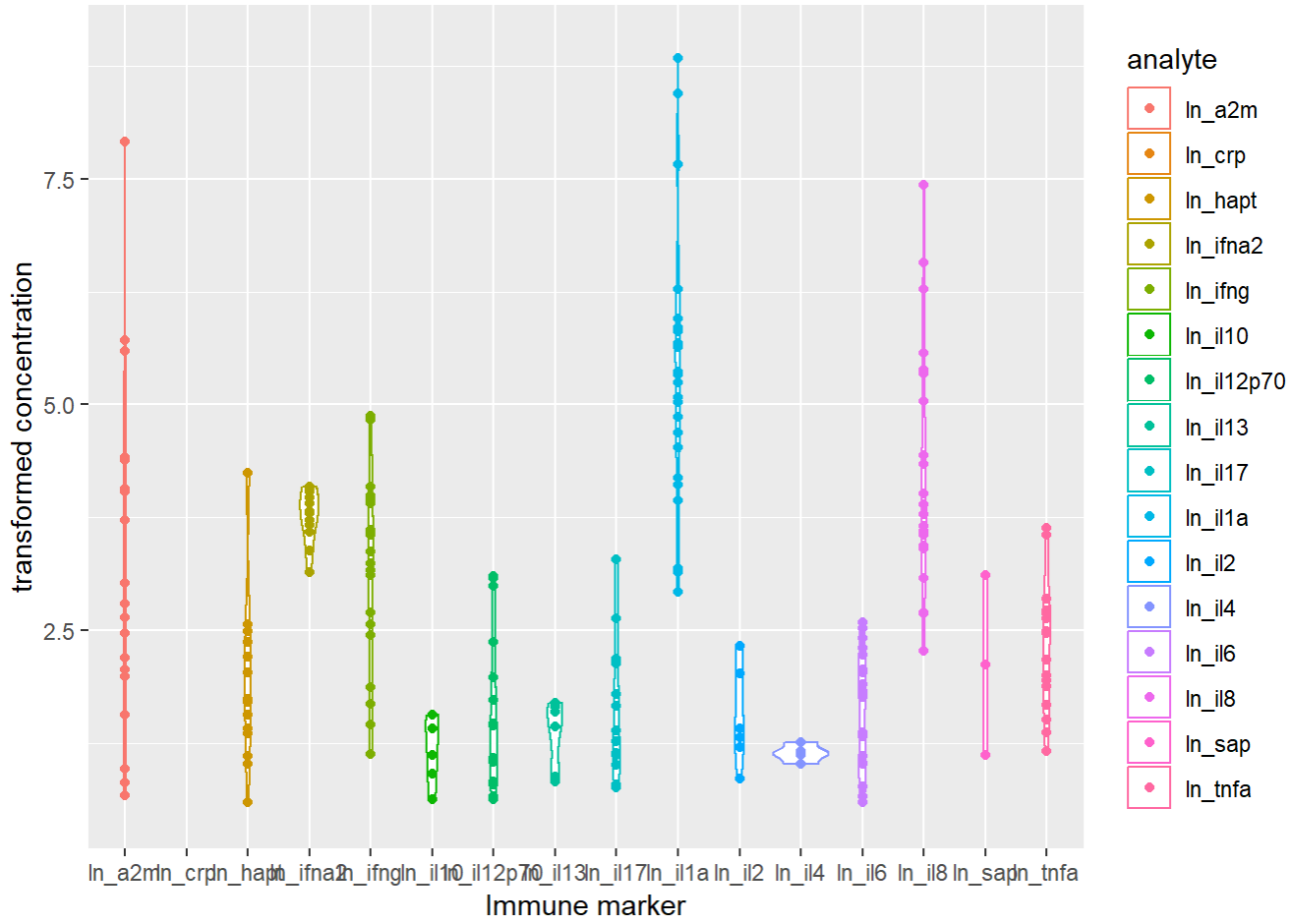
```
# Next, explore the distributions of the transformed immune markers
devo_0 <- gather(neworder_1, "analyte", "concentration", ln_a2m, ln_hapt, ln_crp, ln_sap, ln_il
2, ln_il4, ln_il6, ln_il8, ln_il10, ln_il12p70, ln_il13, ln_il17, ln_ifng, ln_tnfa, ln_il1a, ln_
ifna2)
devo_0$concentration <- as.numeric(devo_0$concentration)
devo_0$analyte <- as.factor(devo_0$analyte)
levels(devo_0$analyte)
```

```
##  [1] "ln_a2m"     "ln_crp"     "ln_hapt"    "ln_ifna2"   "ln_ifng"
##  [6] "ln_il10"    "ln_il12p70" "ln_il13"    "ln_il17"    "ln_il1a"
## [11] "ln_il2"     "ln_il4"     "ln_il6"     "ln_il8"     "ln_sap"
## [16] "ln_tnfa"
```

```
# Plain violin plots:
ggplot(devo_0, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  labs(
    x="Immune marker", y = "transformed concentration")
```




```
# With data superimposed:
ggplot(devo_0, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  #geom_jitter(height = 0, width = 1) +
  geom_point() +
  labs(
    x="Immune marker", y = "transformed concentration")
```



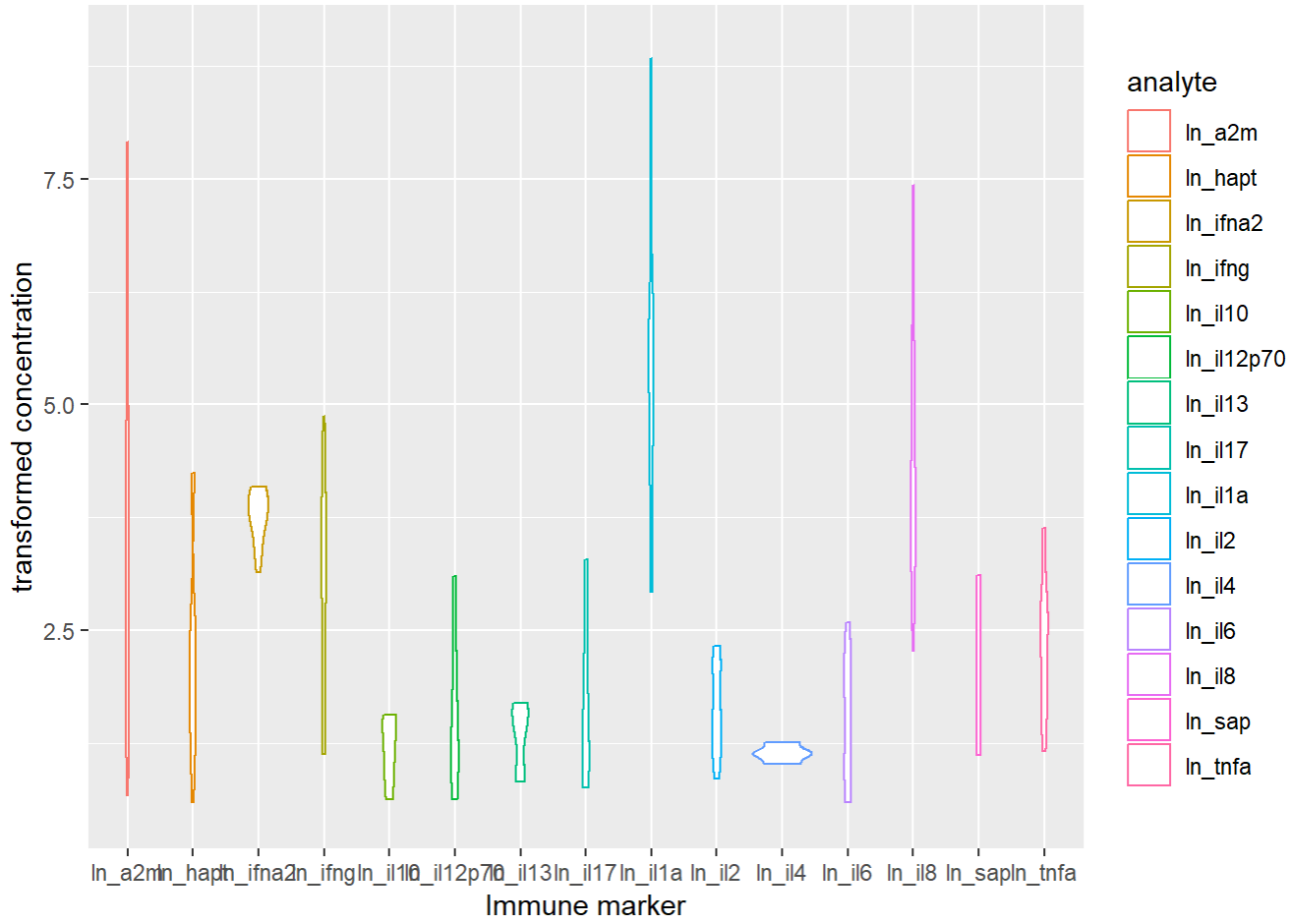
It's still not super useful, though, because CRP is so fat. Take that one out and then have a look, for all cleaning options.

OPTION 1: Left-censored as 0 and right-censored as max

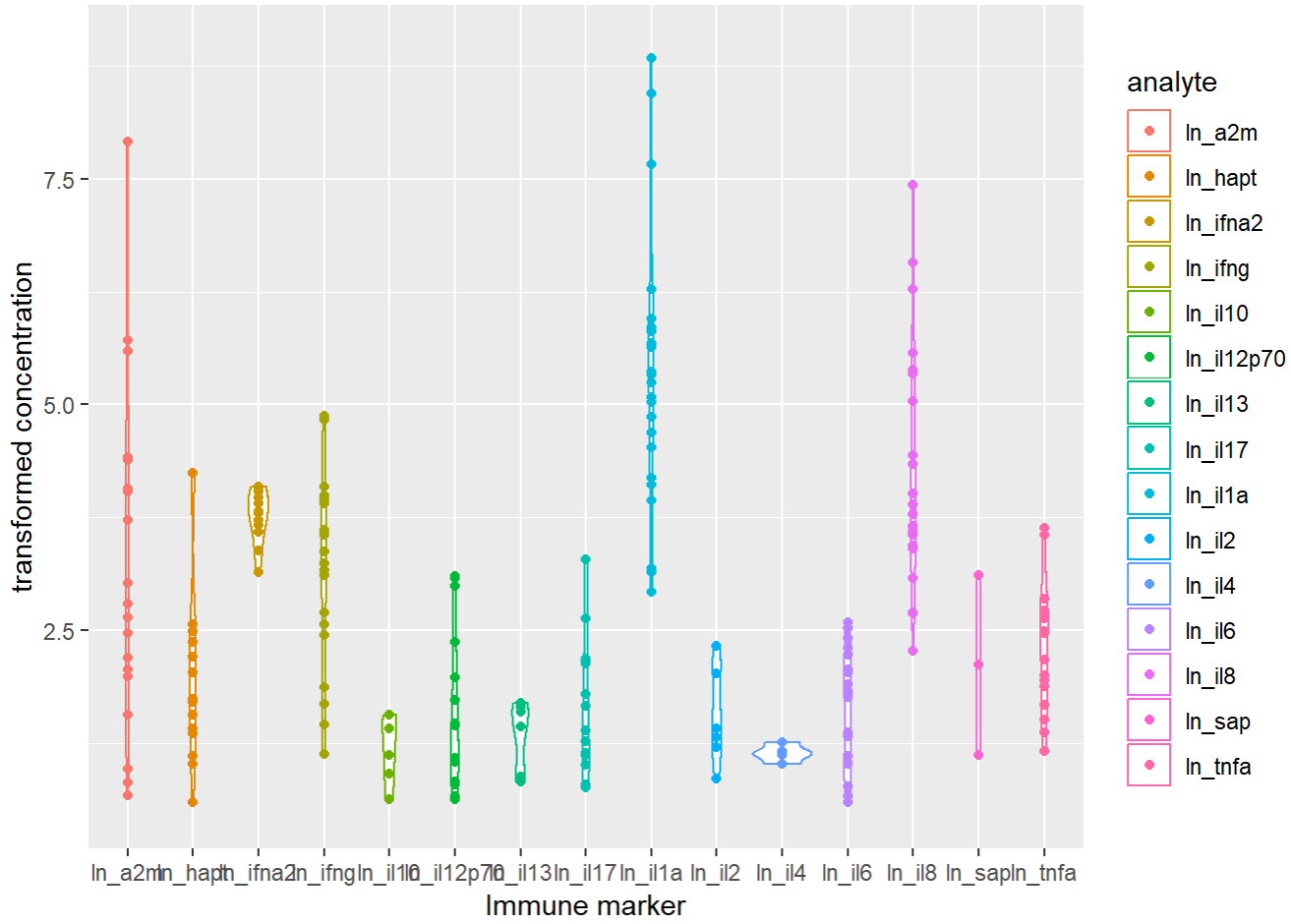
```
devo_1 <- gather(neworder_1, "analyte", "concentration", ln_a2m, ln_hapt, ln_sap, ln_il2, ln_il
4, ln_il6, ln_il8, ln_il10, ln_il12p70, ln_il13, ln_il17, ln_ifng, ln_tnfa, ln_il1a, ln_ifna2)
devo_1$concentration <- as.numeric(devo_1$concentration)
devo_1$analyte <- as.factor(devo_1$analyte)
levels(devo_1$analyte)
```

```
##  [1] "ln_a2m"     "ln_hapt"    "ln_ifna2"   "ln_ifng"    "ln_il10"
##  [6] "ln_il12p70" "ln_il13"    "ln_il17"    "ln_il1a"    "ln_il2"
## [11] "ln_il4"     "ln_il6"     "ln_il8"     "ln_sap"     "ln_tnfa"
```

```
# Plain violin plots
ggplot(devo_1, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  labs(
    x="Immune marker", y = "transformed concentration")
```




```
# With data superimposed
ggplot(devo_1, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  #geom_jitter(height = 0, width = 1) +
  geom_point() +
  labs(
    x="Immune marker", y = "transformed concentration")
```

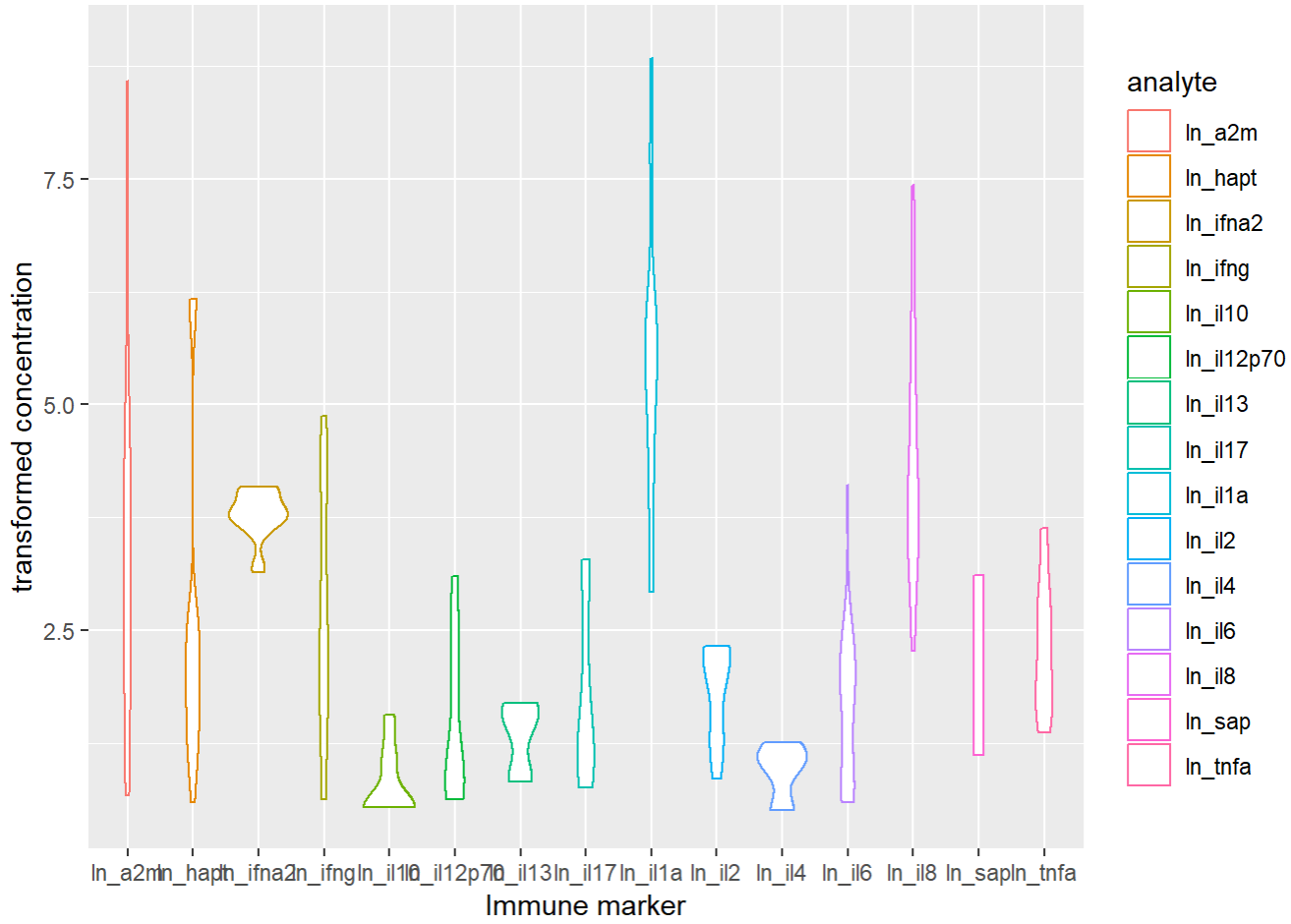


OPTION 2: Left-censored as LLD and right-censored as ULD

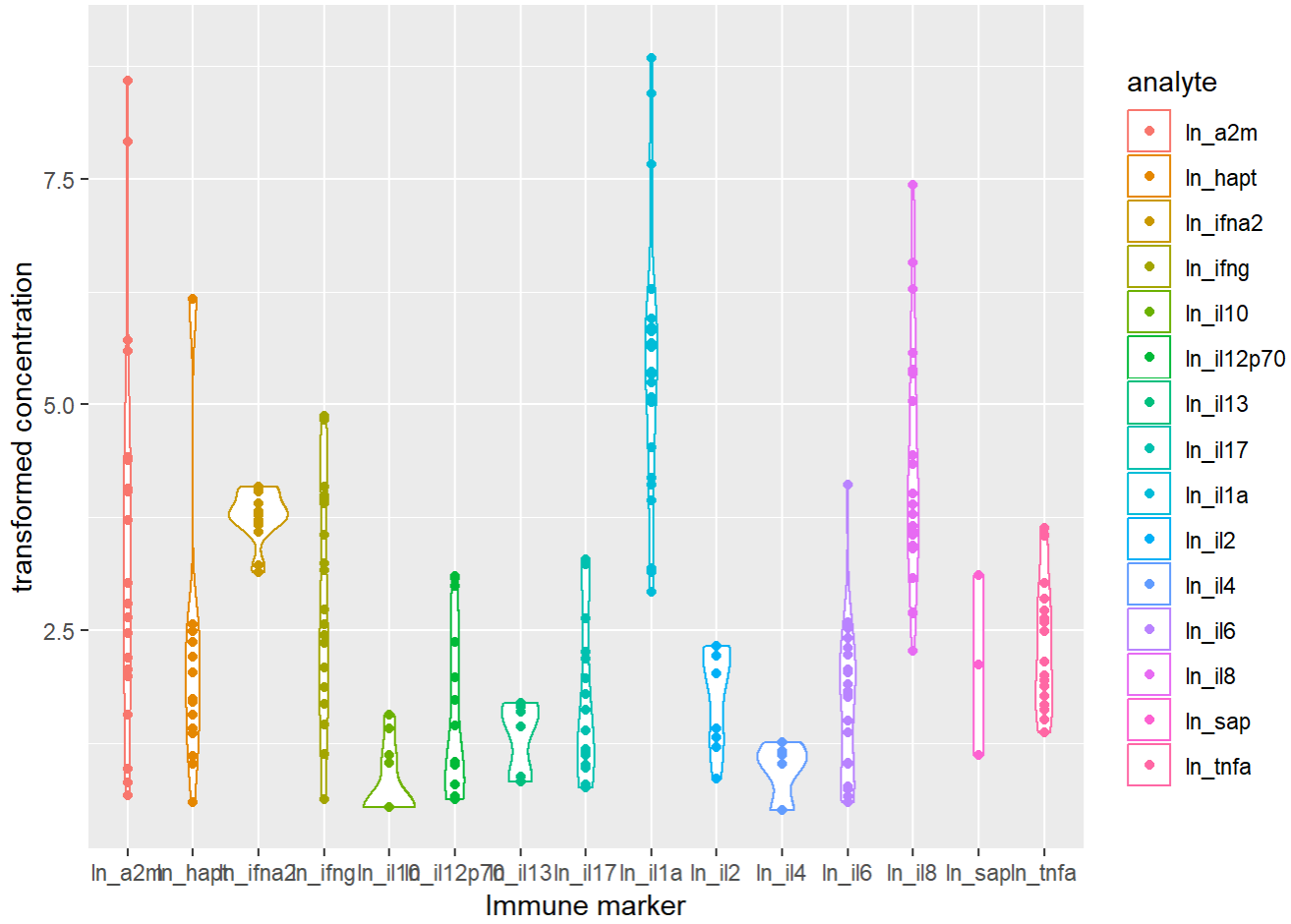
```
devo_2 <- gather(neworder_2, "analyte", "concentration", ln_a2m, ln_hapt, ln_sap, ln_il2, ln_il
4, ln_il6, ln_il8, ln_il10, ln_il12p70, ln_il13, ln_il17, ln_ifng, ln_tnfa, ln_il1a, ln_ifna2)
devo_2$concentration <- as.numeric(devo_2$concentration)
devo_2$analyte <- as.factor(devo_2$analyte)
levels(devo_2$analyte)
```

```
##  [1] "ln_a2m"     "ln_hapt"    "ln_ifna2"   "ln_ifng"    "ln_il10"
##  [6] "ln_il12p70" "ln_il13"    "ln_il17"    "ln_il1a"    "ln_il2"
## [11] "ln_il4"     "ln_il6"     "ln_il8"     "ln_sap"     "ln_tnfa"
```

```
# Plain violin plots:
ggplot(devo_2, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  labs(
    x="Immune marker", y = "transformed concentration")
```




```
# With data superimposed:
ggplot(devo_2, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  #geom_jitter(height = 0, width = 1) +
  geom_point() +
  labs(
    x="Immune marker", y = "transformed concentration")
```

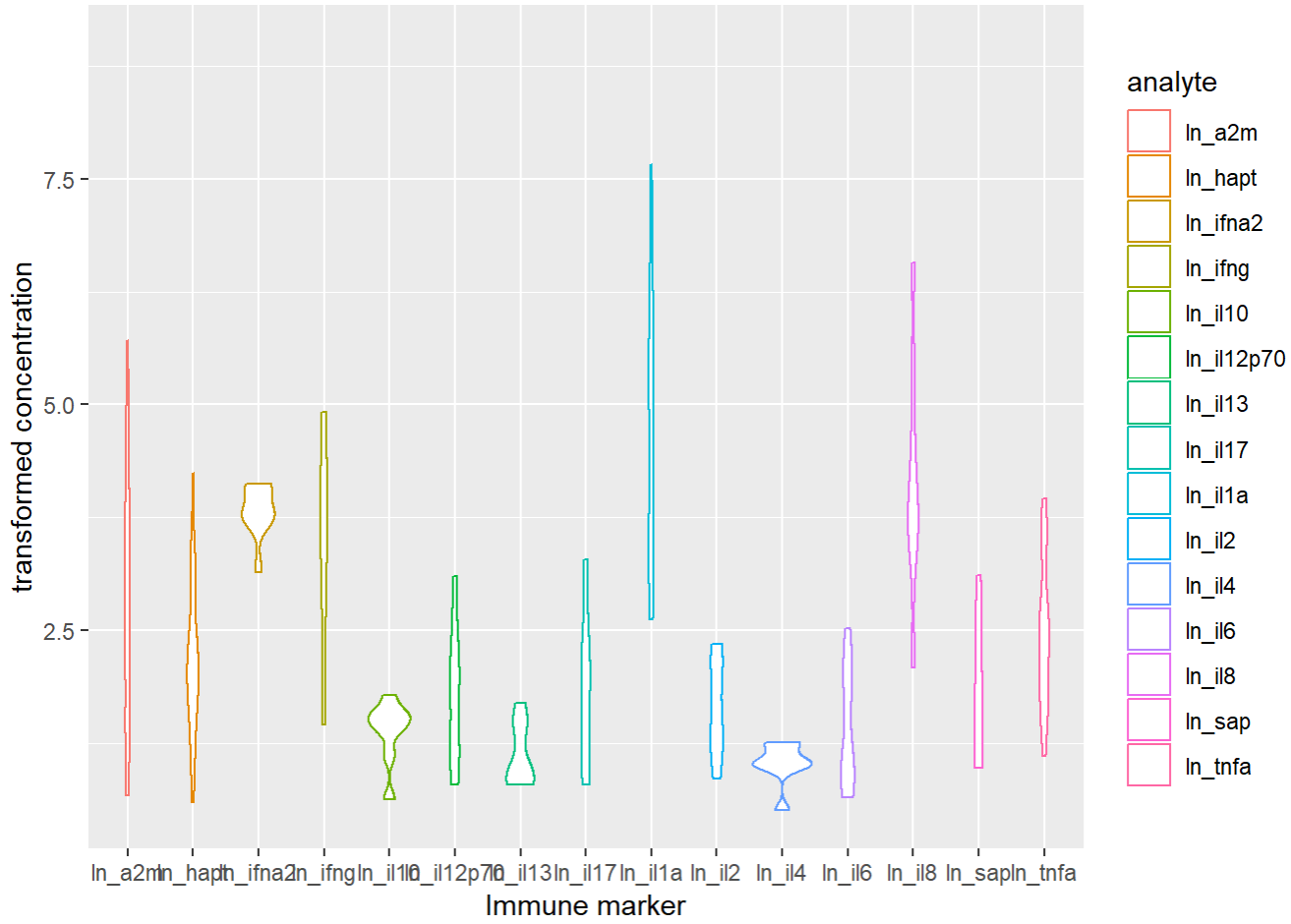


OPTION 3: Left-censored as 1/2 LLD and right-censored as missing

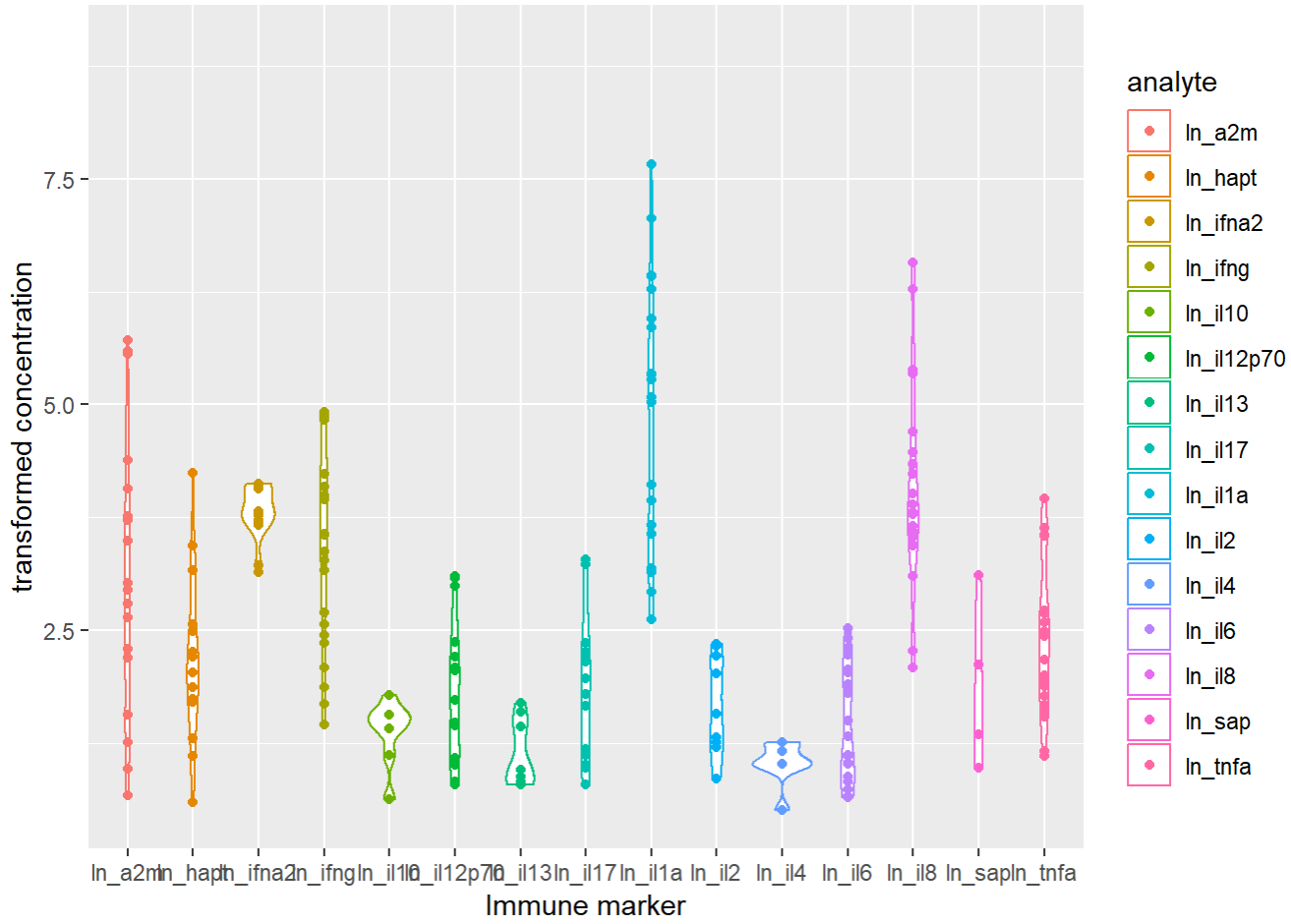
```
devo_3 <- gather(neworder_3, "analyte", "concentration", ln_a2m, ln_hapt, ln_sap, ln_il2, ln_il
4, ln_il6, ln_il8, ln_il10, ln_il12p70, ln_il13, ln_il17, ln_ifng, ln_tnfa, ln_il1a, ln_ifna2)
devo_3$concentration <- as.numeric(devo_3$concentration)
devo_3$analyte <- as.factor(devo_3$analyte)
levels(devo_3$analyte)
```

```
##  [1] "ln_a2m"     "ln_hapt"    "ln_ifna2"   "ln_ifng"    "ln_il10"
##  [6] "ln_il12p70" "ln_il13"    "ln_il17"    "ln_il1a"    "ln_il2"
## [11] "ln_il4"     "ln_il6"     "ln_il8"     "ln_sap"     "ln_tnfa"
```

```
ggplot(devo_3, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  labs(
    x="Immune marker", y = "transformed concentration")
```




```
# With data superimposed:
ggplot(devo_3, aes(x = analyte, y = concentration, color = analyte))+
  ylim(0.5, 9) +
  geom_violin() +
  #geom_jitter(height = 0, width = 1) +
  geom_point() +
  labs(
    x="Immune marker", y = "transformed concentration")
```



Extra: Report your means and ranges * We are not yet sure what the normal ranges are for salivary immune data, so collecting this data from real data would be useful. Although doing this for the simulated data (as is shown here), we may soon include the capability to calculate these on your original data and upload the means and ranges without uploading the individual data.

* Don't forget to also report your age mean and ranges if you have that variable.

```
units_age = "years" #CHANGE IF DIFFERENT (e.g., months)
units_imm = "ng/ml" #CHANGE THIS IF IT IS DIFFERENT (e.g., pg/ml, which is actually standard. Mi
ne is weird)
summary(neworder_1[4:20])
```

```
##    Age at Ax        Sal_A2M           Sal_Hapt         Sal_CRP
##  Min.   :13.35   Min.   :   1.29   Min.   : 0.530   Min.   :0.00100
##  1st Qu.:15.68   1st Qu.:   4.78   1st Qu.: 1.688   1st Qu.:0.00775
##  Median :16.18   Median :  15.15   Median : 5.600   Median :0.03500
##  Mean   :16.24   Mean   : 268.51   Mean   :14.259   Mean   :0.05414
##  3rd Qu.:16.95   3rd Qu.:  80.21   3rd Qu.:12.050   3rd Qu.:0.10000
##  Max.   :18.44   Max.   :2740.25   Max.   :69.370   Max.   :0.27000
##     Sal_SAP          Sal_IL2           Sal_IL4          Sal_IL6
##  Min.   : 0.030   Min.   : 0.0010   Min.   :0.0010   Min.   : 0.840
##  1st Qu.: 0.140   1st Qu.: 0.2102   1st Qu.:0.1050   1st Qu.: 1.745
##  Median : 0.430   Median : 1.1200   Median :0.4600   Median : 2.890
##  Mean   : 2.077   Mean   : 1.9989   Mean   :0.7204   Mean   : 4.471
##  3rd Qu.: 1.123   3rd Qu.: 1.7875   3rd Qu.:0.8250   3rd Qu.: 6.935
##  Max.   :22.320   Max.   :10.1700   Max.   :3.5300   Max.   :13.250
##     Sal_IL8           Sal_IL10      Sal_IL12p70        Sal_IL13
##  Min.   :   9.64   Min.   :0.001   Min.   : 0.001   Min.   :0.200
##  1st Qu.:  35.97   1st Qu.:0.001   1st Qu.: 0.310   1st Qu.:0.380
##  Median :  52.02   Median :0.130   Median : 2.550   Median :0.810
##  Mean   : 217.38   Mean   :0.834   Mean   : 4.838   Mean   :1.466
##  3rd Qu.: 209.05   3rd Qu.:1.270   3rd Qu.: 4.655   3rd Qu.:1.397
##  Max.   :1696.08   Max.   :4.770   Max.   :22.040   Max.   :5.430
##     Sal_IL17         Sal_IFNg          Sal_TNFa         Sal_IL1a
##  Min.   : 0.001   Min.   :  3.080   Min.   : 0.220   Min.   :  18.53
##  1st Qu.: 1.290   1st Qu.:  6.218   1st Qu.: 3.190   1st Qu.:  85.50
##  Median : 2.885   Median : 24.595   Median : 7.175   Median : 209.34
##  Mean   : 6.385   Mean   : 34.837   Mean   :10.547   Mean   : 673.29
##  3rd Qu.: 6.600   3rd Qu.: 40.013   3rd Qu.:14.090   3rd Qu.: 383.92
##  Max.   :26.620   Max.   :130.550   Max.   :37.780   Max.   :6887.88
##    Sal_IFNa2
##  Min.   : 0.001
##  1st Qu.: 0.001
##  Median : 0.001
##  Mean   :18.594
##  3rd Qu.:42.025
##  Max.   :59.810
```

```
units_age
```

```
## [1] "years"
```

```
units_imm
```

```
## [1] "ng/ml"
```