Visualizing and standardizing data

2025-05-22

Introduction

The purpose of this document is to show how specific samples are determined to seropositive or seronegative using external information about a particular pathogen and antigen (e.g., an internationally recognized threshold of protective immunity). This document then describes how to aggregate serostatus information about each sample to calculate a population-level seroprevalence. Upon completing this lab you should be able to:

- Read in control data if available
- Visualize MFI distributions on the appropriate scale
- Calculate serostatus for each sample using a predetermined cutoff
- Calculate a population seroprevalence.

General housekeeping

Before we start, let's navigate to the appropriate working directory. You can accomplish this by navigating to the "Session" tab of RStudio, and choosing "Set Working Directory" –> "Choose Directory" and using your file browser to navigate to the `seroanalytics_workshop` folder. Alternatively, you can modify the code below as appropriate for your files to get to the `seroanalytics_workshop` folder.

```
# set my_path to be the working directory location of where
# the *seroanalytics_workshop* folder is stored on your
# computer
my_path <- "/OneDrive-JohnsHopkins/seroanalytics_workshop"

# source the functions file from the directory using the
# my_path location the functions file is saved in the
# Source/ folder within the working directory
source(paste(my_path, "Source/utils.R", sep = "/"))
```

Reading in data

Note this is the same procedure as in Lab 3.

```
control_data <- read.csv(paste(my_path, "Data/simulated_control_long_training_data.csv",
    sep = "/"))
sample_data <- read.csv(paste(my_path, "Data/simulated_sample_wide_training_data.csv",
    sep = "/"))
```

```r
# this converts sample_data from a wide to a long
# dataframe. edit the column names to your data.
sample_long <- reshape(sample_data, varying = setdiff(names(sample_data),
    c("id", "age", "sex")), v.names = "mfi", timevar = "antigen",
    times = setdiff(names(sample_data), c("id", "age", "sex")),
    idvar = "id", direction = "long")
rownames(sample_long) <- NULL
```

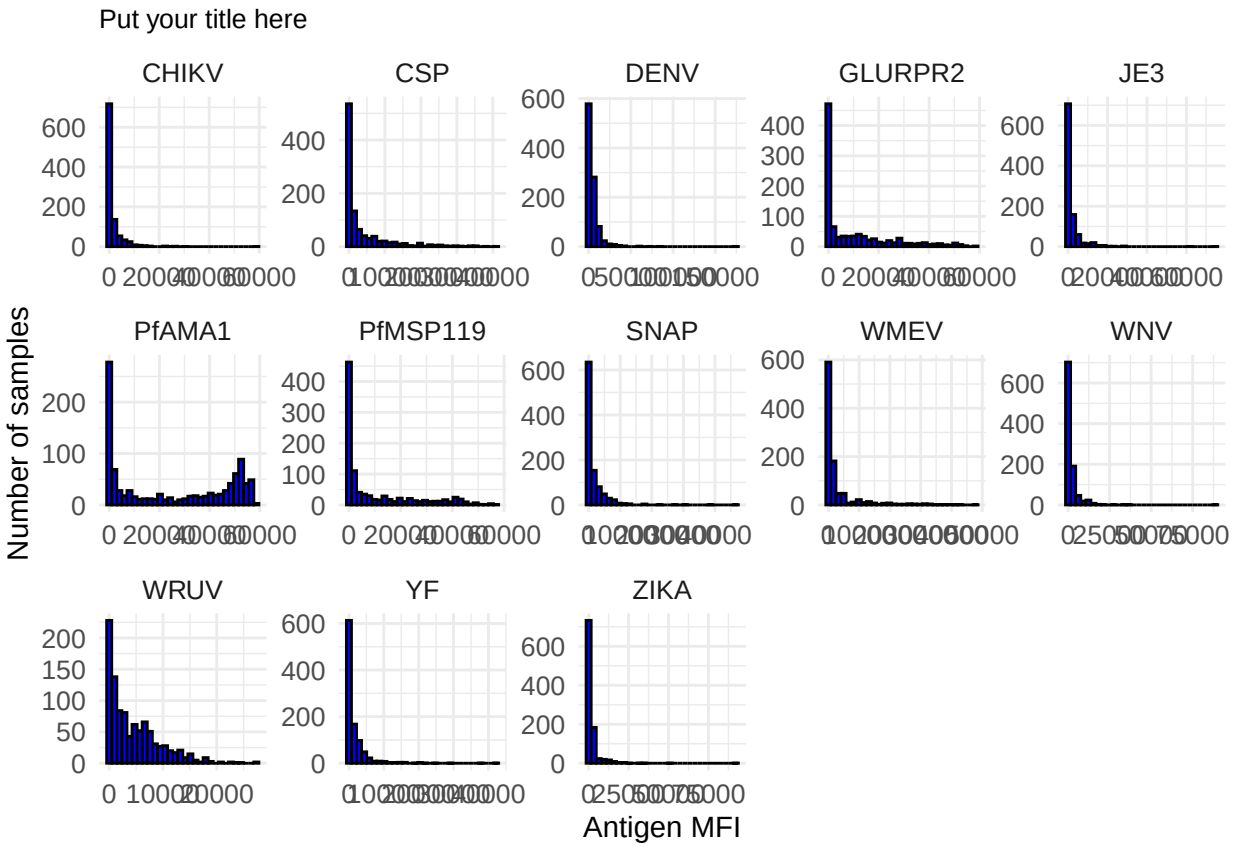
Visualizing your controls

General visalization

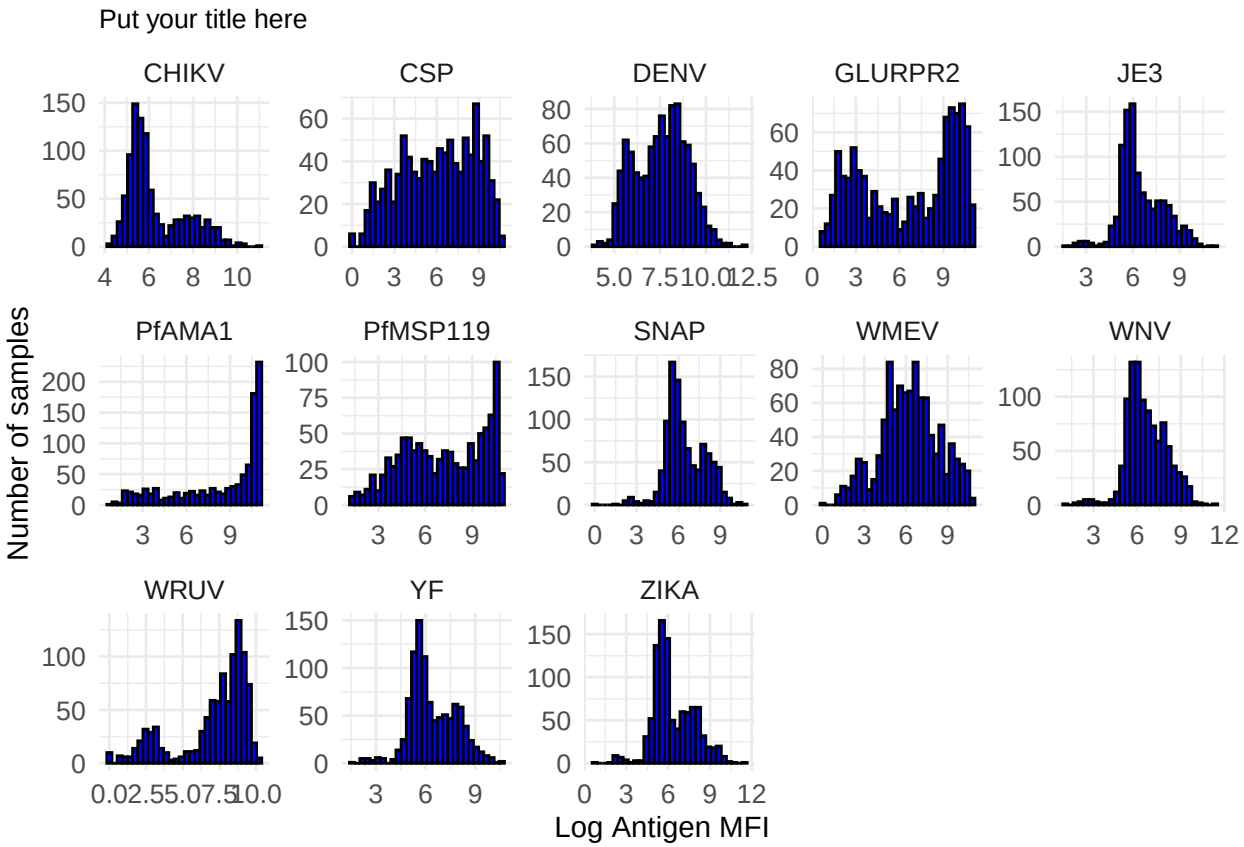
1. Adjust the code below to make a histogram of sample MFI values in an untransformed and log scale.

a. Consider how many bins to use (edit bins = 30 to see what data looks like with different numbers of bins).
b. Describe the distribution (untransformed and log scale).
c. Are there any outliers or anything unusual about your data?

```r
#natural scale
faceted_natural_scale <- ggplot(sample_long, aes(x = mfi)) +
  geom_histogram(bins = 30, color = "black", fill = "blue") +
  facet_wrap(~ antigen, scales = "free", ncol = 5) +  # <- Set 5 columns per row
  labs(
    title = "Put your title here",
    x = "Antigen MFI",
    y = "Number of samples"
  ) +
  theme_minimal() +
  theme(
    strip.text = element_text(size = 10),  # Smaller font for facet labels
    axis.text = element_text(size = 10),  # Smaller font for axis text
    plot.title = element_text(size = 10)  # Smaller font for the plot title
  )
faceted_natural_scale
```

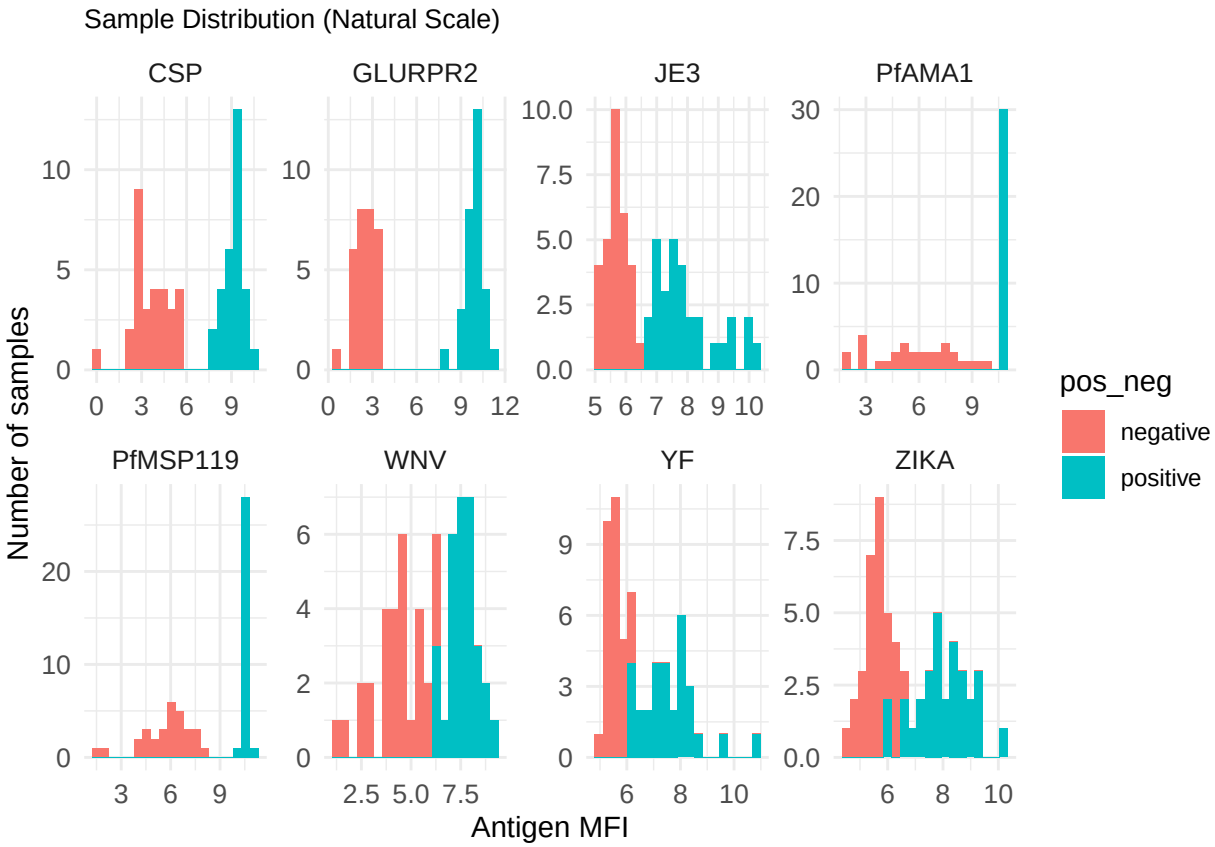



```
#log scale
faceted_log_scale <- ggplot(sample_long, aes(x = log(mfi))) +
  geom_histogram(bins = 30, color = "black", fill = "blue") +
  facet_wrap(~ antigen, scales = "free", ncol = 5) +  # <- Set 5 columns per row
  labs(
    title = "Put your title here",
    x = "Log Antigen MFI",
    y = "Number of samples"
  ) +
  theme_minimal() +
  theme(
    strip.text = element_text(size = 10),  # Smaller font for facet labels
    axis.text = element_text(size = 10),  # Smaller font for axis text
    plot.title = element_text(size = 10)  # Smaller font for the plot title
  )
faceted_log_scale
```

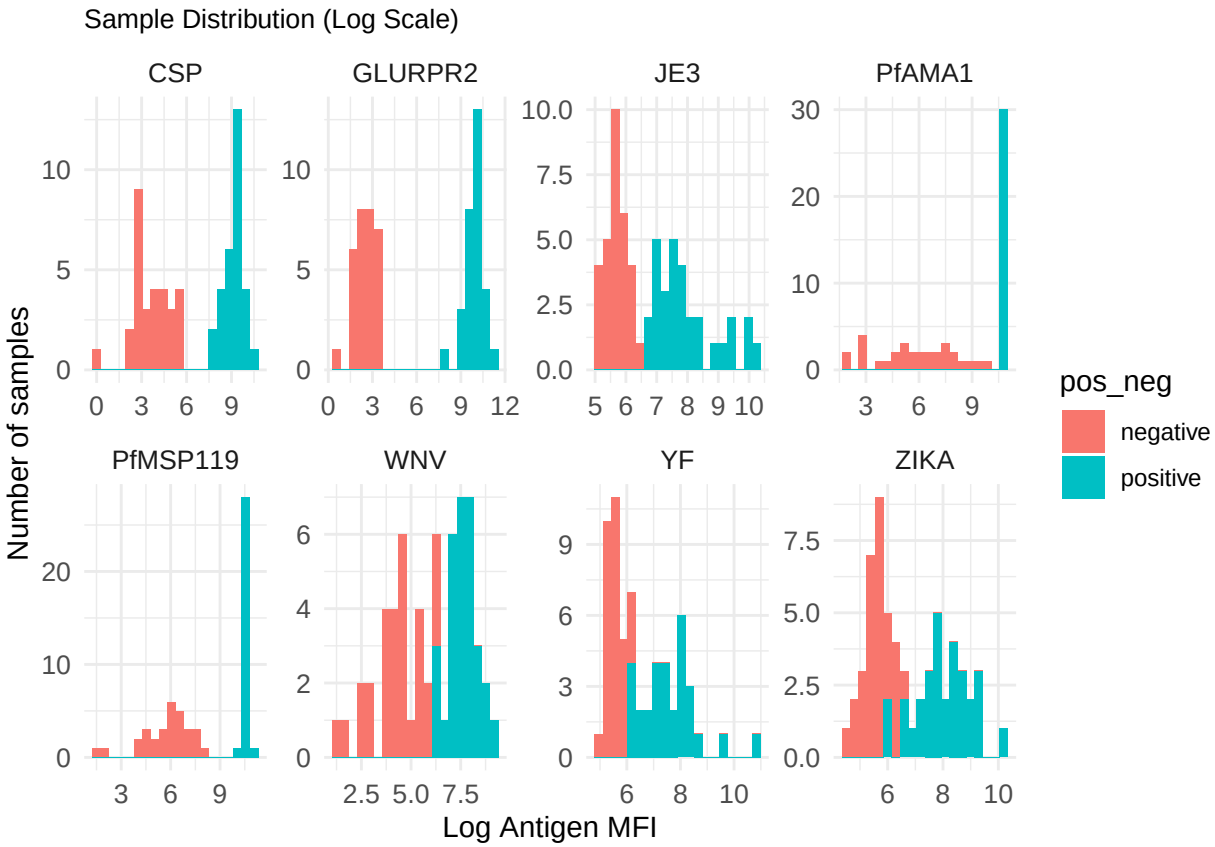


2. Make a histogram of control MFI values. Color the histogram by positive and negative controls. Is there overlap between your positive and negative controls?

```
neg_controls_natural_scale <- ggplot(control_data, aes(x = log(mfi),
    fill = pos_neg)) + geom_histogram(bins = 20) + facet_wrap(~antigen,
    scales = "free", ncol = 4) + labs(title = "Sample Distribution (Natural Scale)",
    x = "Antigen MFI", y = "Number of samples") + theme_minimal() +
    theme(strip.text = element_text(size = 10), axis.text = element_text(size = 10),
        plot.title = element_text(size = 10))
neg_controls_natural_scale
```




```
neg_controls_log_scale <- ggplot(control_data, aes(x = log(mfi), fill = pos_neg)) +
  geom_histogram(bins = 20) +
  facet_wrap(~ antigen, scales = "free", ncol = 4) +  # <- Set 5 columns per row
  labs(
    title = "Sample Distribution (Log Scale)",
    x = "Log Antigen MFI",
    y = "Number of samples"
  ) +
  theme_minimal() +
  theme(
    strip.text = element_text(size = 10),  # Smaller font for facet labels
    axis.text = element_text(size = 10),  # Smaller font for axis text
    plot.title = element_text(size = 10)  # Smaller font for the plot title
  )
neg_controls_log_scale
```



Establishing and applying a cutoff

3. Some antigens have an established standard cutoff (e.g., based on a correlate of protection). For example, for Measles (wmev antigen), a cutoff value typically used on Luminex is 153 IU/mL. Therefore, those samples at or above 153 IU/mL are seropositive and thought to be protected from future measles infection, and those samples below 153 IU/mL are seronegative and thought to be not protected from future measles infection.

a. For this antigen, plot the standard curve of IU/mL dilution by MFI for the 4 plates with raw data, and add a vertical line showing the cutoff. Add a rug plot to show where the observed values fall on the y-axis (MFI values).

```
## Read in data on standards
standard_data <- read.csv(paste(my_path, "Data/simulated_standard_curve_training_data.csv",
    sep = "/"))

## Which samples on the plates are standards?
std_one_plate <- standard_data[which(standard_data$Plate == 1),
    ]
std_all_plates <- data.frame(Sample = std_one_plate$Sample, Dilution = c(c(1/100,
1/200, 1/400, 1/800, 1/1600, 1/3200, 1/6400, 1/12800), std_one_plate$dilution_iu_ml[9:26]),
Location = c(c("65(1,A9)", "66(1,B9)", "67(1,C9)", "68(1,D9)",
        "69(1,E9)", "70(1,F9)", "71(1,G9)", "72(1,H9)"), c("73(1,A10)",
        "74(1,B10)", "75(1,C10)", "76(1,D10)", "77(1,E10)", "78(1,F10)",
```

```
        "79(1,G10)", "80(1,H10)"), c("81(1,A11)", "82(1,B11)",
        "83(1,C11)", "84(1,D11)", "85(1,E11)", "86(1,F11)", "87(1,G11)",
        "88(1,H11)", "89(1,A12)", "90(1,B12)")), Replicate = rep(1,
        nrow(std_one_plate)))

## We will do this for 1 plate at a time

## Plate 1
std_wmev_plate1 <- standard_data[which(standard_data$Antigen ==
    "wmev" & standard_data$Plate == 1), ]

## Plate 2
std_wmev_plate2 <- standard_data[which(standard_data$Antigen ==
    "wmev" & standard_data$Plate == 2), ]

## Plate 3
std_wmev_plate3 <- standard_data[which(standard_data$Antigen ==
    "wmev" & standard_data$Plate == 3), ]

## Plate 4
std_wmev_plate4 <- standard_data[which(standard_data$Antigen ==
    "wmev" & standard_data$Plate == 4), ]

## Combine the standards for the 4 plates
std_wmev_plate1to4 <- rbind(std_wmev_plate1, std_wmev_plate2,
    std_wmev_plate3, std_wmev_plate4)

## Clean up the sample data
plate_1_tidy <- read_and_tidy(file_name = paste(my_path, "Data/Raw Plate 1 dataset.csv",
    sep = "/"), plate_number = 1, num_wells = 96, antigen_names = c("SNAP",
    "WNV", "YF", "JE3", "ZIKA", "DENV", "CHIKV", "GLURPR2", "CSP",
    "PfAMA1", "PfMSP119", "WRUV", "WMEV"), control_samples = c("POS1",
    "POS2", "NEG1", "NEG2"), background_samples = c("BLANK1",
    "BLANK2"), standard_curve_values = std_all_plates, bead_threshold = 30)

plate_2_tidy <- read_and_tidy(file_name = paste(my_path, "Data/Raw Plate 2 dataset.csv",
    sep = "/"), plate_number = 2, num_wells = 96, antigen_names = c("SNAP",
    "WNV", "YF", "JE3", "ZIKA", "DENV", "CHIKV", "GLURPR2", "CSP",
    "PfAMA1", "PfMSP119", "WRUV", "WMEV"), control_samples = c("POS1",
    "POS2", "NEG1", "NEG2"), background_samples = c("BLANK1",
    "BLANK2"), standard_curve_values = std_all_plates, bead_threshold = 30)

plate_3_tidy <- read_and_tidy(file_name = paste(my_path, "Data/Raw Plate 3 dataset.csv",
    sep = "/"), plate_number = 3, num_wells = 96, antigen_names = c("SNAP",
    "WNV", "YF", "JE3", "ZIKA", "DENV", "CHIKV", "GLURPR2", "CSP",
    "PfAMA1", "PfMSP119", "WRUV", "WMEV"), control_samples = c("POS1",
    "POS2", "NEG1", "NEG2"), background_samples = c("BLANK1",
    "BLANK2"), standard_curve_values = std_all_plates, bead_threshold = 30)

plate_4_tidy <- read_and_tidy(file_name = paste(my_path, "Data/Raw Plate 4 dataset.csv",
    sep = "/"), plate_number = 4, num_wells = 96, antigen_names = c("SNAP",
    "WNV", "YF", "JE3", "ZIKA", "DENV", "CHIKV", "GLURPR2", "CSP",
    "PfAMA1", "PfMSP119", "WRUV", "WMEV"), control_samples = c("POS1",
```

```
    "POS2", "NEG1", "NEG2"), background_samples = c("BLANK1",
    "BLANK2"), standard_curve_values = std_all_plates, bead_threshold = 30)

## Filter out low bead counts
plate_1_filt <- filter_low_beads(plate_1_tidy)
plate_2_filt <- filter_low_beads(plate_2_tidy)
plate_3_filt <- filter_low_beads(plate_3_tidy)
plate_4_filt <- filter_low_beads(plate_4_tidy)

## Background removal
plate_1_bg <- rm_background(plate_1_filt, method = "subtraction")
plate_2_bg <- rm_background(plate_2_filt, method = "subtraction")
plate_3_bg <- rm_background(plate_3_filt, method = "subtraction")
plate_4_bg <- rm_background(plate_4_filt, method = "subtraction")

## Select just WMEV antigen
plate_1_bg_WMEV <- plate_1_bg[which(plate_1_bg$Antigen == "WMEV"),
    ]

plate_2_bg_WMEV <- plate_2_bg[which(plate_2_bg$Antigen == "WMEV"),
    ]

plate_3_bg_WMEV <- plate_3_bg[which(plate_3_bg$Antigen == "WMEV"),
    ]

plate_4_bg_WMEV <- plate_4_bg[which(plate_4_bg$Antigen == "WMEV"),
    ]


## Standardize (i.e., convert sample MFIs to IU/mL's) Note
## that the plate_df_norm input should only include the
## serosurvey samples and the relevant standard curve
## (here, measles)

plate_1_standard <- get_concentration_FlexFit(
       plate_df_norm = plate_1_bg_WMEV[which(plate_1_bg_WMEV$Sample_Type==
       "TestSample" | plate_1_bg_WMEV$Sample %in% std_wmev_plate1$Sample[1:8]),
       ], std_curve_values = std_all_plates[17:26, ], input = "MFI")


plate_2_standard <- get_concentration_FlexFit(
       plate_df_norm = plate_2_bg_WMEV[which(plate_2_bg_WMEV$Sample_Type==
       "TestSample" | plate_2_bg_WMEV$Sample %in% std_wmev_plate2$Sample[1:8]),
       ], std_curve_values = std_all_plates[17:26, ], input = "MFI")


plate_3_standard <- get_concentration_FlexFit(
       plate_df_norm = plate_3_bg_WMEV[which(plate_3_bg_WMEV$Sample_Type==
       "TestSample" | plate_3_bg_WMEV$Sample %in% std_wmev_plate3$Sample[1:8]),
       ], std_curve_values = std_all_plates[17:26, ], input = "MFI")


plate_4_standard <- get_concentration_FlexFit(
       plate_df_norm = plate_4_bg_WMEV[which(plate_4_bg_WMEV$Sample_Type==
```

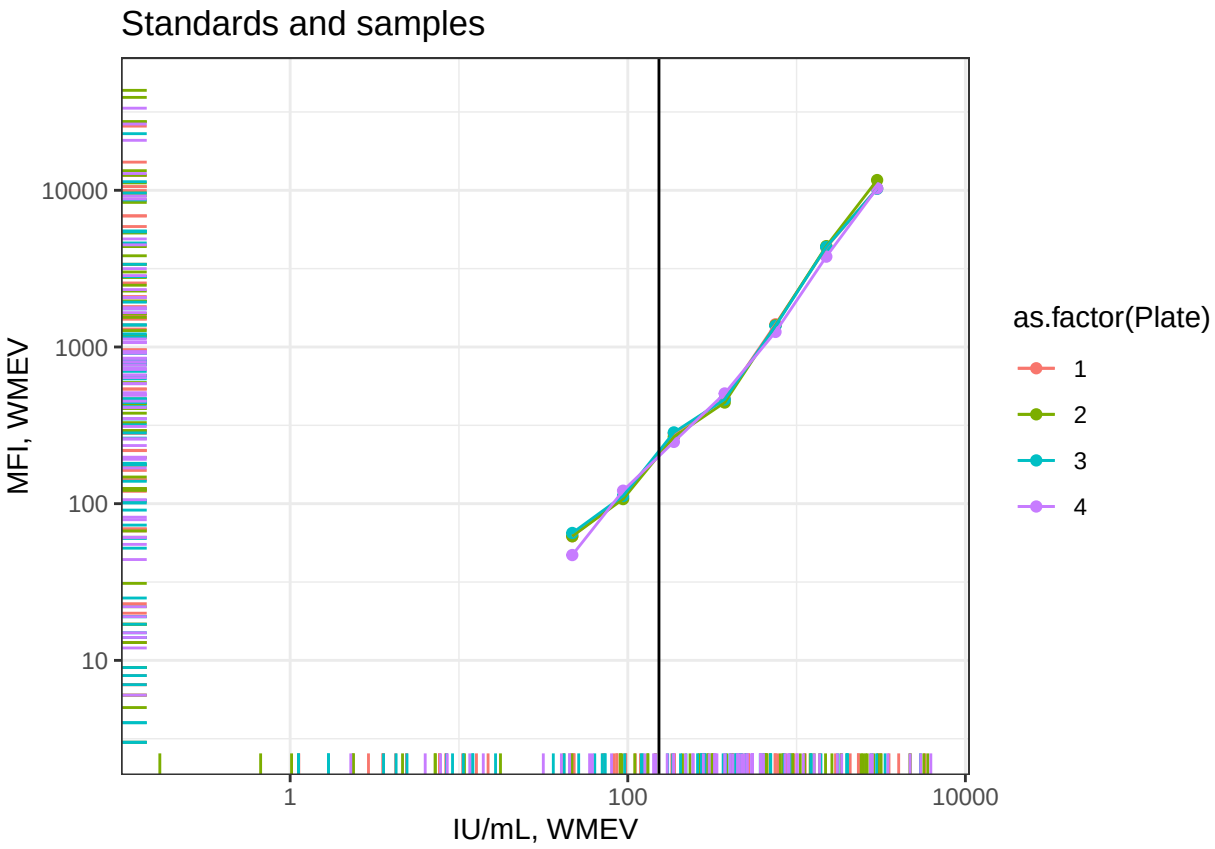
```
      "TestSample" | plate_4_bg_WMEV$Sample %in% std_wmev_plate4$Sample[1:8]),
      ], std_curve_values = std_all_plates[17:26, ], input = "MFI")



## Combine data on test samples only
data_final <- rbind(plate_1_standard[which(plate_1_standard$Sample_Type ==
    "TestSample"), ], plate_2_standard[which(plate_2_standard$Sample_Type ==
    "TestSample"), ], plate_3_standard[which(plate_3_standard$Sample_Type ==
    "TestSample"), ], plate_4_standard[which(plate_4_standard$Sample_Type ==
    "TestSample"), ])

## Compute measles IU/mL
data_final$meas_IU_mL <- exp(data_final$Log_Conc_bg)

## Plot the 4 standard curves and the samples and the
## cutoff For plotting, drop the 2 lowest standards that
## have negative MFI_BG
ggplot(std_wmev_plate1to4[which(std_wmev_plate1to4$dilution_iu_ml >
    23.4), ]) + geom_point(aes(x = dilution_iu_ml, y = WMEV,
    group = as.factor(Plate), colour = as.factor(Plate))) + geom_line(aes(x = dilution_iu_ml,
    y = WMEV, group = as.factor(Plate), colour = as.factor(Plate))) +
    scale_x_log10() + scale_y_log10() + xlab("IU/mL, WMEV") +
    ylab("MFI, WMEV") + theme_bw() + geom_rug(data = data_final,
    aes(x = meas_IU_mL, y = MFI, colour = as.factor(Plate))) +
    ggtitle("Standards and samples") + geom_vline(xintercept = 153)
```



b. Apply cutoff. How many people are seropositive according to this cutoff, and what proportion of people are seropositive?

```
# Define cutoff based on IU/mL
cutoff <- 153

# Applying this cutoff: ifelse function, if first statement
# is true, then outcome is set to 1, and if first statement
# is false, then outcome is 0 since the first statement is
# a vector, then outcome will be a vector of 1's and 0's.
seropositivity <- ifelse(data_final$meas_IU_mL >= cutoff, 1,
    0)  #1 indicates seropositive, and 0 indicates seronegative

# number of people seropositive and seronegative
cat("Table of number seronegative and seronegative", "\n")
```

```
## Table of number seronegative and seronegative
```

```
table(seropositivity, useNA = "always")
```

```
## seropositivity
##    0    1 <NA>
##   88  168    0
```

```r
# percent of people seropositive and seronegative
cat("Table of percent seronegative and seronegative", "\n")
```

```
## Table of percent seronegative and seronegative
```

```r
round(prop.table(table(seropositivity, useNA = "always")), 3) *
    100
```

```
## seropositivity
##    0    1 <NA>
## 34.4 65.6  0.0
```

c. Calculate the confidence interval for this seroprevalence

```r
# Set up to calculate the confidence interval

# this is the number seropositive you get the number
# seropositive in your sample from the seropositive table
# above
x <- 168

# this is the total number of samples in your data note in
# these data we saw above there were no NA values, but if
# you do have NA values be sure to exclude them from this
# count
n <- nrow(data_final)

# this is the confidence interval and 95% is a standard CI
# but you can adjust this if you want
conf <- 0.95

# this uses the exact interval and an epitools function
ci <- binom.exact(x, n, conf.level = conf)

cat("CI lower", round(ci$lower, 4) * 100, "%, CI upper", round(ci$upper,
    4) * 100, "%")
```

```
## CI lower 59.46 %, CI upper 71.43 %
```

d. For this specific antigen, how would you interpret this seroprevalence and confidence interval?

e. What do you think about using this cutoff method for this antigen? What are the assumptions that went into this cutoff method?