ImpactTB/BAA: Standard Operating Procedures for Data Analysis

Colorado State University Coding Team

2022-08-08

Contents

Chapter 1

Overview

1.1 About the project

The objective of the Immune Mechanisms of Protection against Mycobacterium tuberculosis (IMPAc-TB) program is to get a thorough understanding of the immune responses necessary to avoid initial infection with *Mycobacterium tuberculosis (Mtb)*, formation of latent infection, and progression to active TB illness. To achieve these goals, the National Institute of Allergy and Infectious Diseases awarded substantial funding and established multidisciplinary research teams that will analyze immune responses against *Mtb* in animal models (mice, guinea pigs, and non-human primates) and humans, as well as immune responses elicited by promising vaccine candidates. The contract awards establish and give up to seven years of assistance for IMPAc-TB Centers to explain the immune responses required for *Mtb* infection protection.

The seven centers that are part of the study are (in alphabetical order):

1. Colorado State University
2. Harvard T.H. Chan School of Public Health
3. Seattle Children Hospital
4. [more]

Colorado State University Team and role of each member:

- Dr. Marcela Henao-Tamayo: Principal Investigator
- Dr. Brendan Podell: Principal Investigator
- Dr. Andres Obregon-Henao: Research Scientist-III
- Dr. Taru S. Dutt: Research Scientist-I
- [more]

1.2 About this book

The aim of this book is to provide data protocols and data collection templates for key types of data that are collected over the course of this project. By using standard templates to record data, as well as starting from defined pipelines to process and analyze the data, we aim to standardize the collection and processing of data across this project.

Here, we have built a comprehensive guide to wet lab data collection, sample processing, and computational tool creation for robust and efficient data analysis and dissemination.

Chapter 2

Experimental metadata

Metadata for an experiment:

- species
- start_date
- end_date
- experimental_groups

Chapter 3

Initial mouse characteristics

At the start of each experiment with a mouse model, we record several measurements or characteristics of each mouse. We record these measurements along with an identifier for each mouse (for example, based on tags or ear notches), so that we can later link the initial characteristics of each mouse with later measurements on the same mouse.

The values that we initially record for each mouse include:

- `group`: An identifier for the experimental group to which the mouse is assigned (e.g., "Control", "Group 1")
- `group_detail`: A longer description of the mouse's treatment group (e.g., "Vaccinated with vaccine candidate A at 4 and 8 weeks")
- `notch_id`: The ear notch pattern of the mouse (e.g., "0" for no notch, "1R" for one notch in the right ear)
- `mouse_number`: A number that corresponds with the mouse's ear notch patter; together with the mouse's group number, this provides a unique identifier for each mouse in the experiment
- `cage_number`: The number of the cage to which the mouse is first assigned. This may change over the course of the experiment, as mice might be removed from a cage due to fighting, etc. Any of these later changes of cage will be recorded [where]
- `sex`: Whether the mouse is male ("m") or female ("f")
- `age`: Age
- `strain`: The strain of the mouse (e.g., "C57BL/6J" for Black 6, "C3HeB/FeJ" for Kramnik)

We have created a spreadsheet template that can be used to record these data, which you can download by clicking here. This template currently includes example data (colored in red to help you remember that it's only there as an

example). To Use this template, take a look at the example data, then delete it and replace with the real data for your experiment.

Here is an example of how the first rows of this template might look once it's filled out:

This template should be used at the initial time when mice are brought into the experiment. The file format is an Excel file, so you can use it by saving it to your computer and then opening and recording data with Excel. Later code in this chapter will read in a file in this template format to provide basic summaries of the data. Later code will read in these files to record the data in a project-wide database, which will allow us to integrate it with other data collected over the course of the experiment.

[Rules for naming the file. Include experiment name / study ID?]

Chapter 4

Mouse Weights

4.0.1 Overview

Extreme weight loss and loss of muscle mass, also known as cachexia, typically presents along side chronic inflammatory illnesses like Tuberculosis disease @ [baazim2022interplay]. We now recognize that cachexia is part of a systemic response to inflammation, and has been linked to upregulation of pro-inflammatory cytokines such as TNF, IL-6, and IFNg in humans @ [baazim2022interplay]. Additionally, studies support the role of key immune cell populations such as CD8+ T-cells that, when depleted, counteract muscle and fat deterioration @[baazim2019cd8], and suggest that CD8+ T-cells may metabolically reprogram adipose tissue.

In recognition of cachexia related illnesses and diseases, we tracked the progression of weight loss over the course of this study, as is done with many TB-mouse studies @ [smith2022host], @ [segueni2016controlled]. These data is also useful when correlating to CFU count as well as expression of cytokines and other biological markers @ [smith2022host]. Here, mice are weighed in grams weekly to monitor clinical status as TB patients frequently display weight loss as clinical symptom associated with disease progression.

The following contains information about how the data was collected, organized, and curated for analysis in RStudio.

4.0.2 Parameters

Weights are recorded in an excel worksheet.

Column titles are as follows: who_collected date_collected sex dob notch_id mouse_number weight unit cage_number group notes

Groups included are: bcg, saline, bcg+id93, saline+id93, saline+noMtb

The notes column contains information regarding clinical observations.

good reference: https://elifesciences.org/articles/74419#s4

```
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.1.2
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## Warning: package 'readr' was built under R version 4.1.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

4.0.3 Read in data

Data is stored in one excel sheet, each week is one sheet named as the date -> return vector for each sheet name

4.0.4 Can also use rio to read in the data, more streamlined

4.0.5 Clean data

```
dataset <- data$before_vaccination %>%
 select("sex", "mouse_number", "weight", "cage_number", "group")
```

```
# combining columns mouse_number and cage_number

dataset$mouse_id <- paste(dataset$mouse_number, "-", dataset$cage_number)
```

4.0.6 Body weight over time graph and statistics

4.0.7 Weight loss over time graph and statistics

4.0.8 Weight vs CFU

4.0.9 Weight vs ELISA results

4.0.10 Weight vs lesion burden

Chapter 5

Colony forming units to determine bacterial counts

5.1 Data description

The data are collected in a spreadsheet with multiple sheets. The first sheet (named "[x]") is used to record some metadata for the experiment, while the following sheets are used to record CFUs counts from the plates used for samples from each organ, with one sheet per organ. For example, if you plated data from both the lung and spleen, there would be three sheets in the file: one with the metadata, one with the plate counts for the lung, and one with the plate counts for the spleen.

The metadata sheet is used to record information about the overall process of plating the data. Values from this sheet will be used in calculating the bacterial load in the original sample based on the CFU counts. This spreadsheet includes the following columns:

- `organ`: Include one row for each organ that was plated in the experiment. You should name the organ all in lowercase (e.g., "lung", "spleen"). You should use the same name to also name the sheet that records data for that organ for example, if you have rows in the metadata sheet for "lung" and "spleen", then you should have two other sheets in the file, one sheet named "lung" and one named "spleen", which you'll use to store the plate counts for each of those organs.
- `prop_resuspended`: In this column, give the proportion of that organ that was plated. For example, if you plated half the lung, then in the "lung" row of this spread sheet, you should put 0.5 in the `prop_resuspended` column.

- `total_resuspended_uL`: This column contains an original volume of tissue homogenate. For example, raw lung tissue is homogenized in 500 uL of PBS in a tube containing metal beads.
- `og_aliquot_uL`: 100 uL of th total_resuspended slurry would be considered an original aliquot and is used to peform serial dilutions.
- `dilution_factor`: Amount of the original stock solution that is present in the total solution, after dilution(s)
- `plated_uL`: Amount of suspension + diluent plated on section of solid agar

5.2 Read in data

```
library(readxl)
library(dplyr)
library(purrr)
library(tidyr)
library(stringr)
library(tidyverse)
library(gridExtra)
library(ggplot2)
library(ggpubr)

#Replace w/ path to CFU sheet
path <- c("DATA/Copy of baa_cfu_sheet.xlsx")

sheet_names <- excel_sheets(path)
sheet_names <- sheet_names[!sheet_names %in% c("metadata")]

merged_data <- list()

for(i in 1:length(sheet_names)){

  data <- read_excel(path, sheet = sheet_names[i]) %>%
    mutate(organ = paste0(sheet_names[i]))

  data <- data %>%
    #mutate(missing_col = NA) %>%
    mutate_if(is.double, as.numeric) %>%
    mutate_if(is.numeric, as.character) %>%
    pivot_longer(starts_with("dil_"), names_to = "dilution",
                 values_to = "CFUs") %>%
    mutate(dilution = str_extract(dilution, "[0-9]+"),
           dilution = as.numeric(dilution))
```

```
  merged_data[[i]] <- data


}

all_data <- bind_rows(merged_data, .id = "column_label") %>%
    select(-column_label)

head(merged_data)
```

```
## [[1]]
## # A tibble: 342 x 8
##    count_date           who_plated who_counted groups mouse organ dilution CFUs
##    <chr>                <chr>      <chr>       <chr>  <chr> <chr>    <dbl> <chr>
##  1 "\"February 21 2022~ BK         BK          group~ A     lung         0 TNTC
##  2 "\"February 21 2022~ BK         BK          group~ A     lung         1 TNTC
##  3 "\"February 21 2022~ BK         BK          group~ A     lung         2 TNTC
##  4 "\"February 21 2022~ BK         BK          group~ A     lung         3 53
##  5 "\"February 21 2022~ BK         BK          group~ A     lung         4 9
##  6 "\"February 21 2022~ BK         BK          group~ A     lung         5 4
##  7 "\"February 21 2022~ BK         BK          group~ A     lung         6 2
##  8 "\"February 21 2022~ BK         BK          group~ A     lung         7 1
##  9 "\"February 21 2022~ BK         BK          group~ A     lung         8 0
## 10 "\"February 21 2022~ BK         BK          group~ B     lung         0 TNTC
## # ... with 332 more rows
##
## [[2]]
## # A tibble: 112 x 8
##    count_date           who_plated who_counted groups  mouse organ dilution CFUs
##    <chr>                <chr>      <chr>       <chr>   <chr> <chr>    <dbl> <chr>
##  1 "\"April 25 2022\"" JR         JR          group_1 A     sple~        0 TNTC
##  2 "\"April 25 2022\"" JR         JR          group_1 A     sple~        1 TNTC
##  3 "\"April 25 2022\"" JR         JR          group_1 A     sple~        2 53
##  4 "\"April 25 2022\"" JR         JR          group_1 A     sple~        3 9
##  5 "\"April 25 2022\"" JR         JR          group_1 A     sple~        4 4
##  6 "\"April 25 2022\"" JR         JR          group_1 A     sple~        5 2
##  7 "\"April 25 2022\"" JR         JR          group_1 A     sple~        6 1
##  8 "\"April 25 2022\"" JR         JR          group_1 A     sple~        7 0
##  9 "\"April 25 2022\"" JR         JR          group_1 B     sple~        0 TNTC
## 10 "\"April 25 2022\"" JR         JR          group_1 B     sple~        1 TNTC
## # ... with 102 more rows
```

```
head(all_data)
```

```
## # A tibble: 6 x 8
##   count_date            who_plated who_counted groups mouse organ dilution CFUs
##   <chr>                 <chr>      <chr>       <chr>  <chr> <chr>    <dbl> <chr>
## 1 "\"February 21 2022\~ BK         BK          group~ A     lung         0 TNTC
## 2 "\"February 21 2022\~ BK         BK          group~ A     lung         1 TNTC
## 3 "\"February 21 2022\~ BK         BK          group~ A     lung         2 TNTC
## 4 "\"February 21 2022\~ BK         BK          group~ A     lung         3 53
## 5 "\"February 21 2022\~ BK         BK          group~ A     lung         4 9
## 6 "\"February 21 2022\~ BK         BK          group~ A     lung         5 4
```

5.3 Example one

5.4 Exploratory analysis and quality checks

5.5 Exploratory analysis

Dimensions of input data:

Based on the input data, data were collected for the following organ or organs:

The following number of mice were included for each:

The following number of replicates were recorded at each count date for each experimental group:

The following number of dilutions and dilution level were recorded for each organ:

People who plated and collected the data. Date or dates of counting:

Based on the input data, the plates included in these data were counted by the following person or persons: Based on the input data, the plates included in these data were counted on the following date or dates:

```
all_data %>%
  select(organ, who_plated, who_counted, count_date) %>%
  distinct()
```

```
## # A tibble: 3 x 4
##   organ  who_plated who_counted count_date
##   <chr>  <chr>      <chr>       <chr>
## 1 lung   BK         BK          "\"February 21 2022\""
```

```
## 2 lung   BK         BK          "\"April 18 2022\""
## 3 spleen JR         JR          "\"April 25 2022\""
```

```
head(all_data)
```

```
## # A tibble: 6 x 8
##   count_date            who_plated who_counted groups mouse organ dilution CFUs
##   <chr>                 <chr>      <chr>       <chr>  <chr> <chr>    <dbl> <chr>
## 1 "\"February 21 2022\~ BK         BK          group~ A     lung         0 TNTC
## 2 "\"February 21 2022\~ BK         BK          group~ A     lung         1 TNTC
## 3 "\"February 21 2022\~ BK         BK          group~ A     lung         2 TNTC
## 4 "\"February 21 2022\~ BK         BK          group~ A     lung         3 53
## 5 "\"February 21 2022\~ BK         BK          group~ A     lung         4 9
## 6 "\"February 21 2022\~ BK         BK          group~ A     lung         5 4
```

Distribution of CFUs at each dilution:

Here's a plot that shows how many plates were too numerous to count at each dilution level:

Here is a plot that shows how the CFU counts were distributed by dilution level in the data:

5.6 Identify a good dilution for each sample

```
# Make all_data into tidy data and filter for CFUs between 10-75

tidy_cfu_data <- all_data %>%
  mutate(dilution = str_extract(dilution, "[0-9]+"),
         dilution = as.numeric(dilution)) %>%
  filter((CFUs >= 5 & CFUs <= 95) | groups == "control") %>%
  mutate(CFUs = as.numeric(CFUs))


head(tidy_cfu_data)
```

```
## # A tibble: 6 x 8
##   count_date            who_plated who_counted groups mouse organ dilution  CFUs
##   <chr>                 <chr>      <chr>       <chr>  <chr> <chr>    <dbl> <dbl>
## 1 "\"February 21 2022\~ BK         BK          group~ A     lung         3    53
## 2 "\"February 21 2022\~ BK         BK          group~ A     lung         4     9
## 3 "\"February 21 2022\~ BK         BK          group~ C     lung         5     8
## 4 "\"February 21 2022\~ BK         BK          group~ D     lung         3    53
## 5 "\"February 21 2022\~ BK         BK          group~ A     lung         2    92
## 6 "\"February 21 2022\~ BK         BK          group~ A     lung         4     7
```

5.7 Calculate CFUs from best dilution/Estimate bacterial load for each sample based on good dilution

```
# Calculating CFU/ml for every qualifying replicate between 10-75 CFUs. Column binding
meta <- read_excel(path, sheet = "metadata")

tidy_cfu_meta_joined <- inner_join(meta, tidy_cfu_data) %>%
  group_by(groups) %>%
  mutate(CFUs_per_ml = (CFUs * (dilution_factor^dilution) *
                          (total_resuspension_mL/volume_plated_ul) * 1000)) %>%
  select(organ, count_date, who_plated, who_counted, groups,  mouse, dilution,
         CFUs, CFUs_per_ml) %>%
  ungroup()
```

```
## Joining, by = "organ"
```

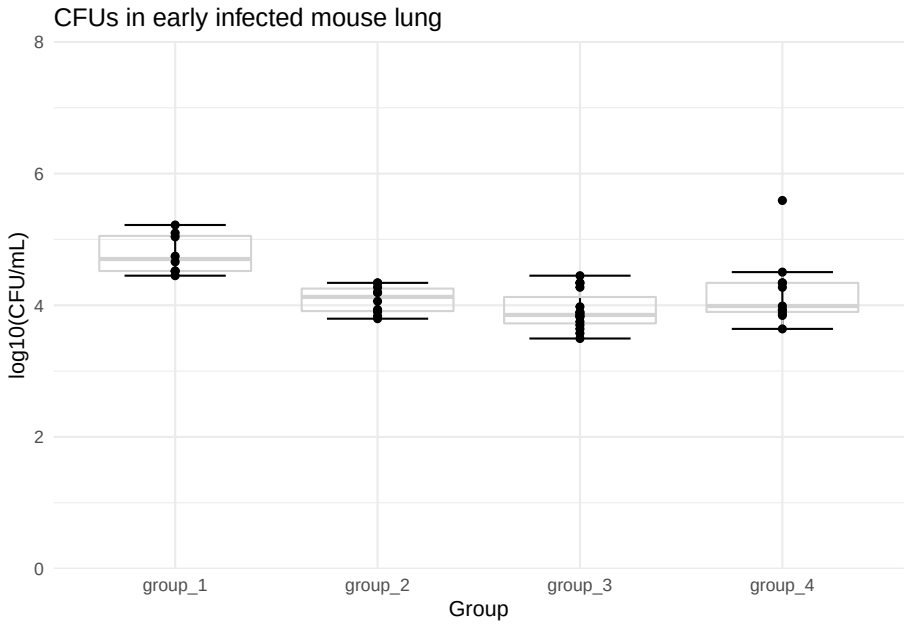
```
head(tidy_cfu_meta_joined)
```

```
## # A tibble: 6 x 9
##   organ count_date            who_plated who_counted groups mouse dilution  CFUs
##   <chr> <chr>                 <chr>      <chr>       <chr>  <chr>    <dbl> <dbl>
## 1 lung  "\"February 21 2022\~ BK         BK          group~ A            3    53
## 2 lung  "\"February 21 2022\~ BK         BK          group~ A            4     9
## 3 lung  "\"February 21 2022\~ BK         BK          group~ C            5     8
## 4 lung  "\"February 21 2022\~ BK         BK          group~ D            3    53
## 5 lung  "\"February 21 2022\~ BK         BK          group~ A            2    92
## 6 lung  "\"February 21 2022\~ BK         BK          group~ A            4     7
## # ... with 1 more variable: CFUs_per_ml <dbl>
```

5.8 Create initial report information for these data

```
tidy_lung_cfu_plot <- tidy_cfu_meta_joined %>%
  filter(organ == "lung") %>%
  mutate(group = fct_relevel(groups, "group_1", "group_2", "group_3", "group_4")) %>%
  ggplot(aes(x = groups, y = log10(CFUs_per_ml), fill = groups))+
  stat_boxplot( aes(x = groups, y = log10(CFUs_per_ml)),
    geom='errorbar', linetype=1, width=0.5)+
```

```
  geom_boxplot(aes(group = groups), fill = NA, show.legend = FALSE, color = "lightgrey")+
  geom_point(show.legend = FALSE)+
  labs(title = paste0("CFUs in early infected mouse lung"), x = "Group", y = "log10(CFU/mL)",
       color = "Group")+
  guides(shape = "none")+
  theme_minimal()+
  stat_compare_means(label = "p.signif", method = "t.test", ref.group = "group_1") +
  scale_y_continuous(expand = c(0, 0), limits = c(0, 8))

tidy_lung_cfu_plot
```



5.9 Sample ANOVA

```
cfu_stats <- tidy_cfu_meta_joined %>%
  group_by(organ) %>%
  nest() %>%
  mutate(aov_result = map(data, ~aov(CFUs_per_ml ~ groups, data = .x)),
         tukey_result = map(aov_result, TukeyHSD),
         tidy_tukey = map(tukey_result, broom::tidy)) %>%
  unnest(tidy_tukey, .drop = TRUE) %>%
  separate(contrast, into = c("contrast1", "contrast2"), sep = "-") %>%
  select(-data, -aov_result, -tukey_result, -term, -null.value)# %>%
```

```
## Warning: The `.drop` argument of `unnest()` is deprecated as of tidyr 1.0.0.
## All list-columns are now preserved.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated
```

```
  # filter(adj.p.value <= 0.05)

cfu_stats
```

```
## # A tibble: 9 x 7
## # Groups:   organ [2]
##   organ  contrast1 contrast2 estimate conf.low conf.high adj.p.value
##   <chr>  <chr>     <chr>        <dbl>    <dbl>     <dbl>       <dbl>
## 1 lung   group_2   group_1    -60953. -138742.    16836.      0.171
## 2 lung   group_3   group_1    -63903. -135699.     7893.      0.0963
## 3 lung   group_4   group_1    -26214. -102416.    49987.      0.793
## 4 lung   group_3   group_2     -2950.  -69900.    64000.      0.999
## 5 lung   group_4   group_2     34739.  -36915.   106393.      0.569
## 6 lung   group_4   group_3     37689.  -27410.   102787.      0.417
## 7 spleen group_2   group_1     -6565   -13529.      399.      0.0656
## 8 spleen group_3   group_1     -7310   -13341.    -1279.      0.0178
## 9 spleen group_3   group_2      -745.   -6776.     5286.      0.943
```

5.10 Save processed data to database

5.11 Example two

Chapter 6

Enzyme-linked immunosorbest assay (ELISA)

ELISA is a standard molecular biology assay for detecting and quantifying a variety of compounds, including peptides, proteins, and antibodies in a sample. The sample could be serum, plasma, or bronchoalveolar lavage fluid (BALF).

6.0.0.1 Importance of ELISA

An antigen-specific reaction in the host results in the production of antibodies, which are proteins found in the blood. In the event of an infectious disease, it aids in the detection of antibodies in the body.ELISA is distinguishable from other antibody-assays in that it produces quantifiable findings and separates non-specific from specific interactions by serial binding to solid surfaces, which is often a polystyrene multiwell plate.

In IMPAc-TB project, it is crucial to evaluate the if the vaccine is eliciting humoral immunity and generating antibodies against vaccine antigen. ELISA will be used to determine the presence of Immunoglobulin (Ig) IgG, IgA, and IgM in the serum different time points post-vaccination.

6.0.0.2 Principle of ELISA

ELISA is based on the principle of antigen-antibody interaction. An antigen must be immobilized on a solid surface and then complexed with an enzyme-linked antibody in an ELISA. The conjugated enzyme's activity is evaluated

by incubating it with a substrate to yield a quantifiable result, which enables detection. There are four basic steps of ELISA:

1. Coating multiwell plate with antigen/antibody: This step depends on what we want to detect the sample. If we need to evaluate the the presence of antibody, the plate will be coated with the antigen, and vice versa. To coat the plate, a fixed concentration of antigen (protein) is added to a 96 well high-binding plate (charged plate). Plate is incubated over night with the antigen at 4 degree celsius (as proteins are temperature sensitive) so that antigens are completely bound to the well.

2. Blocking: It is possible that not each and every site of the well is coated with the targeted antigen, and there could be uncovered areas. It is important to block those empty spaces so that primary antibody (which we will add to the next step) binds to these spaces and give us false positive results. For this, microplate well surface-binding sites are blocked with an unrelated protein or other substance.Most common blocking agents are bovine serum albumin, skim milk, and casein. One of the best blocking agents is to use the serum from the organism in which your secondary (detection antibody) is raised. For example, if the secondary antibody is raised in goat, then we can use goat serum as a blocking agent.

3. Probing: Probing is the step where we add sample containing antibodies that we want to detect. This will be the primary antibody. If the antibodies against the antigen (which we have coated) are present in the sample, it will bind to the antigen with high affinity.

4. Washing: After the incubation of sample containing primary antibody, the wells are washed so that any unbound antibody is washed away. Washing solution contains phosphate buffer saline + 0.05% tween-20 (a mild detergent). 0.05% tween-20 washes away all the non-specific interactions as those are not strong, but keeps all the specific interaction as those are strong and cannot be detached with mild detergent.

5. Detection: To detect the presence of antibody-antigen complex, a secondary antibody labelled with an enzyme (usually horseradish peroxidase) is added to the wells, incubated and washed.

6. Signal Measurement: Finally to detect "if" and "how much" of the antibody is present, a chromogenic substrate (like 3,3',5,5'-Tetramethylbenzidine) is added to the wells, which can be cleaved the the enzyme that is tagged to the secondary antibody. The color compund is formed after the addition of the substrate, which is directly proportional to the amount of antibody present in the sample. The plate is read on a plate reader, where color is converted to numbers.

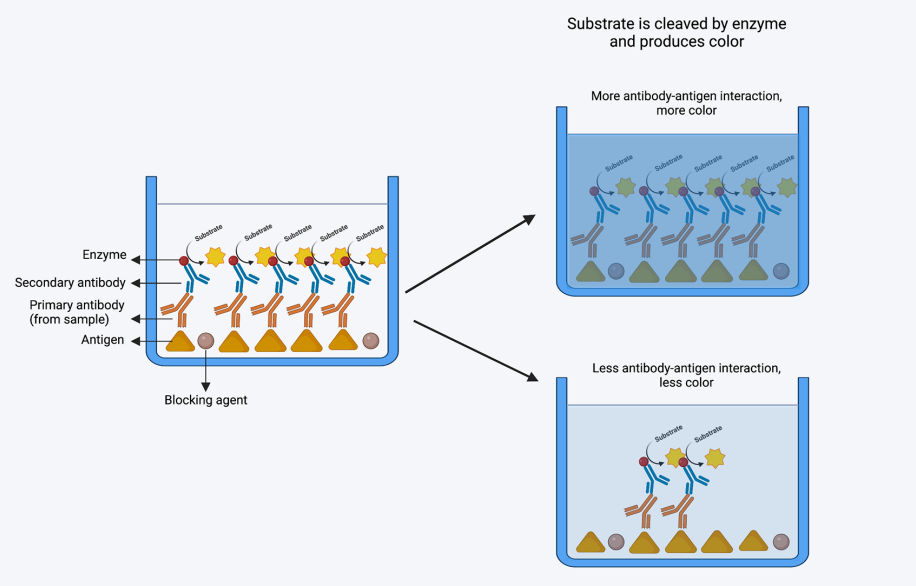


Figure 6.1: A caption

6.1 Read in data from excel file

```
library(readxl)
library(tidyverse)
library(minpack.lm)
```

```
## Warning: package 'minpack.lm' was built under R version 4.1.2
```

```
library(broom)
```

```
## Warning: package 'broom' was built under R version 4.1.2
```

```
library(purrr)
```

```
elisa_raw_data <- read_excel("DATA/elisa_s1_07-25-20.xlsx", sheet = "S1", col_names = FALSE,  ran
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
```

```
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
```

```
head(elisa_raw_data)
```

```
## # A tibble: 6 x 12
##   ...1          ...2  ...3 ...4   ...5  ...6 ...7   ...8  ...9 ...10 ...11 ...12
##   <chr>        <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl> <dbl> <chr> <dbl> <dbl>
## 1 5.199999999~ 0.05  0.069 6.3E~ 0.061 0.122 0.16~ 0.145 0.135 6.80~ 0.053 0.05
## 2 7.900000000~ 0.098 0.069 6.80~ 0.115 0.202 5.89~ 0.134 0.069 0.106 0.05  0.075
## 3 8.899999999~ 0.133 0.119 OVRF~ 3.87  2.32  OVRF~ 3.85  2.12  OVRF~ 3.21  1.02
## 4 OVRFLW       3.46  1.16  OVRF~ 3.80  2.36  OVRF~ 3.70  1.49  OVRF~ 3.68  1.63
## 5 3.815999999~ 1.82  0.446 3.89~ 3.42  1.13  OVRF~ 2.33  0.608 OVRF~ 3.41  1.10
## 6 OVRFLW       3.69  1.43  OVRF~ 3.66  1.27  3.839 1.74  0.444 2.49~ 0.637 0.704
```

6.1.1 Tidy the data

```
# Convert all columns to numeric

elisa_raw_data_numeric <- elisa_raw_data %>%
  mutate_if(is.character, as.numeric)
```

```
## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion

## Warning in mask$eval_all_mutate(quo): NAs introduced by coercion
```

```
# pivot longer the data

elisa_raw_data_tidy <- pivot_longer(data = elisa_raw_data_numeric, cols = "...1":"...12

# remove "..." from the first column
```

```
elisa_raw_data_tidy$well_id <- str_replace(elisa_raw_data_tidy$well_id, "...", "")

# Add new column to the data_frame

elisa_raw_data_tidy_new <- elisa_raw_data_tidy %>%
  mutate(name = rep(LETTERS[1:8], each = 12))

elisa_raw_data_tidy_new <- elisa_raw_data_tidy_new %>%
  mutate(well_id = paste0(name, well_id)) %>%
  select(-name)

head(elisa_raw_data_tidy_new)
```

```
## # A tibble: 6 x 2
##   well_id od_450nm
##   <chr>      <dbl>
## 1 A1         0.052
## 2 A2         0.05
## 3 A3         0.069
## 4 A4         0.063
## 5 A5         0.061
## 6 A6         0.122
```

6.1.2 Read in second data set

```
elisa_label_data <- read_excel("DATA/elisa_s1_07-25-20.xlsx", sheet = "S1", col_names = FALSE,  r
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
```

```
head(elisa_label_data)
```

```
## # A tibble: 6 x 12
##   ...1        ...2   ...3  ...4  ...5  ...6  ...7  ...8  ...9  ...10 ...11 ...12
##   <chr>       <chr>  <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 blank       secon~ naïv~ 1A-1~ 1A-1~ 1A-1~ 1A-2~ 1A-2~ 1A-2~ 1A-3~ 1A-3~ 1A-3~
## 2 1A-4 (1/250 1A-4 ~ 1A-4~ 1B-1~ 1B-1~ 1B-1~ 1B-2~ 1B-2~ 1B-2~ 1B-3~ 1B-3~ 1B-3~
## 3 1B-4 (1/250 1B-4 ~ 1B-4~ 2A-1~ 2A-1~ 2A-1~ 2A-2~ 2A-2~ 2A-2~ 2A-3~ 2A-3~ 2A-3~
## 4 2B-1 (1/250 2B-1 ~ 2B-1~ 2B-2~ 2B-2~ 2B-2~ 2B-3~ 2B-3~ 2B-3~ 2B-4~ 2B-4~ 2B-4~
## 5 3A-1 (1/250 3A-1 ~ 3A-1~ 3A-2~ 3A-2~ 3A-2~ 3A-3~ 3A-3~ 3A-3~ 3A-4~ 3A-4~ 3A-4~
## 6 3B-1 (1/250 3B-1 ~ 3B-1~ 3B-2~ 3B-2~ 3B-2~ 3B-3~ 3B-3~ 3B-3~ 3B-4~ 3B-4~ 3B-4~
```

```
# pivot longer the data

elisa_label_data_tidy <- pivot_longer(data = elisa_label_data, cols = "...1":"...12", r

# remove "..." from the first column

elisa_label_data_tidy$well_id <- str_replace(elisa_label_data_tidy$well_id, "...", "")

# Add new column to the data_frame

elisa_label_data_tidy_new <- elisa_label_data_tidy %>%
  mutate(name = rep(LETTERS[1:8], each = 12))

elisa_label_data_tidy_new <- elisa_label_data_tidy_new %>%
  mutate(well_id = paste0(name, well_id)) %>%
  select(-name)

head(elisa_label_data_tidy_new)
```

```
## # A tibble: 6 x 2
##   well_id information
##   <chr>   <chr>
## 1 A1      blank
## 2 A2      secondary
## 3 A3      naïve (1/250)
## 4 A4      1A-1 (1/250
## 5 A5      1A-1 (1/1250
## 6 A6      1A-1 (1/6250
```

6.2 #Join the OD table with the information table

```
elisa_data = elisa_raw_data_tidy_new %>% inner_join(elisa_label_data_tidy_new, by="well_id")

head(elisa_data)
```

```
## # A tibble: 6 x 3
##   well_id od_450nm information
##   <chr>      <dbl> <chr>
## 1 A1         0.052 blank
## 2 A2         0.05  secondary
## 3 A3         0.069 naïve (1/250)
## 4 A4         0.063 1A-1 (1/250
## 5 A5         0.061 1A-1 (1/1250
## 6 A6         0.122 1A-1 (1/6250
```

6.2.1 Separate the information table into sample ID and dilution columns

```
tidy_elisa_data <- separate(elisa_data, col = "information", into = c("sample_id", "dilution"), s
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 2 rows [1, 2].
```

```
head(tidy_elisa_data)
```

```
## # A tibble: 6 x 4
##   well_id od_450nm sample_id   dilution
##   <chr>      <dbl> <chr>       <chr>
## 1 A1         0.052 "blank"     <NA>
## 2 A2         0.05  "secondary" <NA>
## 3 A3         0.069 "naïve "    1/250)
## 4 A4         0.063 "1A-1 "     1/250
## 5 A5         0.061 "1A-1 "     1/1250
## 6 A6         0.122 "1A-1 "     1/6250
```

```
tidy_elisa_data <- tidy_elisa_data %>%
  mutate(dilution = str_extract(dilution, "(/)[0-9]+"),
         dilution = str_replace(dilution, "/", ""),
```

```
         dilution = as.numeric(dilution))

tidy_elisa_data <- tidy_elisa_data %>%
  select(well_id, sample_id, dilution, od_450nm)

head(tidy_elisa_data)
```

```
## # A tibble: 6 x 4
##   well_id sample_id   dilution od_450nm
##   <chr>   <chr>          <dbl>    <dbl>
## 1 A1      "blank"           NA    0.052
## 2 A2      "secondary"       NA    0.05
## 3 A3      "naïve "         250    0.069
## 4 A4      "1A-1 "          250    0.063
## 5 A5      "1A-1 "         1250    0.061
## 6 A6      "1A-1 "         6250    0.122
```

6.3 ELISA data analysis optimization

ELISA data can be analyzed in different ways based on how the data is acquired. There are a a few examples of the type of ELISA data:

1. With standard curve: ELISA can be used to determine the concentrations of the antigen and antibody. This type of ELISA data usually have a standard curve with different concentrations of the known analyte and the concentration in the sample is determined by extrapolating the unknown values in the curve. This type of assay is straightforward, easy to interpret and are more robust.

2. Without standard curve: Usually vaccine studies involve investigating the presence of high-affinity (and novel) antibodies against the vaccine antigens. Therefore, plotting a standard curve is not feasible as there is no previous information available for antibody concentration or type of antibody. Also, because antibody response to a vaccine will differ depending on the individual, it is not practical to generate a calibration curve from which absolute concentrations can be extrapolated. For this type of ELISA, quantification of the antibody titers is performed using serial dilutions of the test samples, and analysis can be performed using the following three methods:

1. Fitting sigmoid model
2. Endpoint titer method 3: Absorbance summation method

Let's have a look at these methods, how we can apply these methods in our data, and R-based packages that we can utilize to perform tis analysis.

6.3.1 1. Fitting sigmoid model:

In this model, we will perform a 8-10 point serial dilution of our sample and plot a 4 parameter sigmoidal curve. Example data:

```
elisa_example_data <- read_excel("DATA/example_elisa_data.xlsx")

# separate 1/

elisa_example_data <- separate(elisa_example_data, col = "dilution", into = c("numerator", "denom

elisa_example_data <- elisa_example_data %>%
  mutate_if(is.character, as.numeric)

elisa_example_data$dilution <- elisa_example_data$numerator / elisa_example_data$denomenator

elisa_example_data <- elisa_example_data %>%
  mutate(log_dilution = log(dilution, base = 3))

head(elisa_example_data)
```

```
## # A tibble: 6 x 5
##   numerator denomenator absorbance dilution log_dilution
##       <dbl>       <dbl>      <dbl>    <dbl>        <dbl>
## 1         1          30       4    0.0333          -3.10
## 2         1          90       3.73 0.0111          -4.10
## 3         1         270       2.34 0.00370         -5.10
## 4         1         810       1.1  0.00123         -6.10
## 5         1        2430       0.51 0.000412        -7.10
## 6         1        7290       0.22 0.000137        -8.10
```

```
mod_1 <- nlsLM(absorbance ~ ((a-d)/(1+(log_dilution/c)^b)) + d,
data = elisa_example_data,
start = list (a = 4, d= 0, c = -5, b = 1))

# a = maximum absorbance
# d = minimum absobance
# c = point of maximum growith (dilition where the curve is straight line)
# b = slope at c

mod_1
```

```
## Nonlinear regression model
##   model: absorbance ~ ((a - d)/(1 + (log_dilution/c)^b)) + d
```

```
##    data: elisa_example_data
##        a        d        c        b
##  4.12406  0.04532 -5.31056  7.62972
##  residual sum-of-squares: 0.02221
##
## Number of iterations to convergence: 9
## Achieved convergence tolerance: 1.49e-08
```

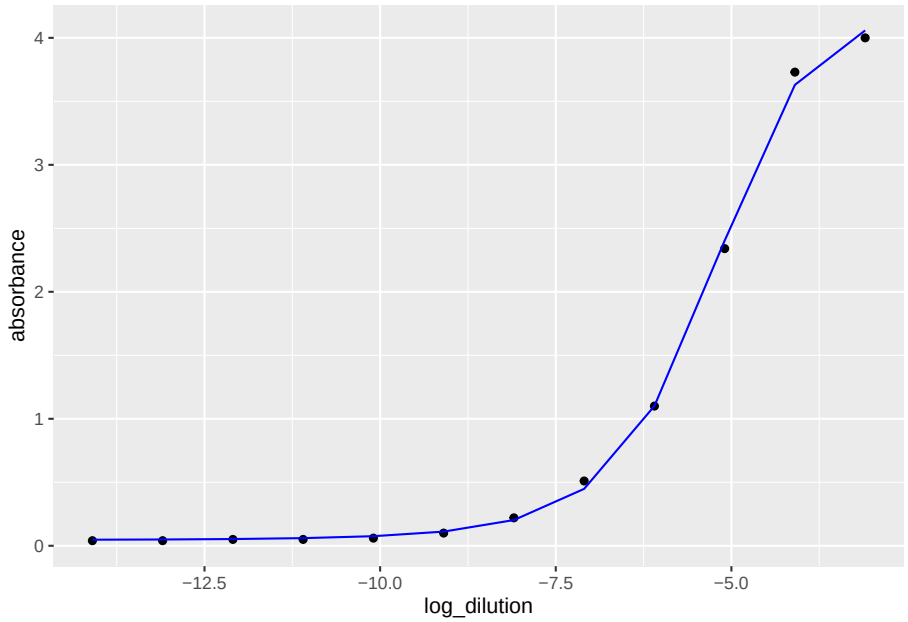
```
summary(mod_1)
```

```
##
## Formula: absorbance ~ ((a - d)/(1 + (log_dilution/c)^b)) + d
##
## Parameters:
##   Estimate Std. Error  t value Pr(>|t|)
## a  4.12406    0.05820   70.860 1.75e-12 ***
## d  0.04532    0.02268    1.998   0.0808 .
## c -5.31056    0.03933 -135.037 1.01e-14 ***
## b  7.62972    0.35854   21.280 2.50e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05269 on 8 degrees of freedom
##
## Number of iterations to convergence: 9
## Achieved convergence tolerance: 1.49e-08
```

6.3.1.1 Fitted model curve

```
tidy_params <- mod_1 %>% tidy()

a <- tidy_params$estimate[tidy_params$term == "a"]
b <- tidy_params$estimate[tidy_params$term == "b"]
c <- tidy_params$estimate[tidy_params$term == "c"]
d <- tidy_params$estimate[tidy_params$term == "d"]

elisa_example_data <- elisa_example_data %>%
  mutate(fitted = predict(mod_1))

elisa_example_data <- elisa_example_data %>%
  mutate(fitted = predict(mod_1))

elisa_example_data %>%
```

```
  ggplot(aes(x = log_dilution, y = absorbance)) +
  geom_point() +
  geom_line(aes(y=fitted), color = "blue")
```



6.3.2 2. Endpoint titer method

The endpoint titer approach chooses an absorbance value just above the background noise (or the lower asymptotic level). **The highest dilution with an absorbance greater than this predetermined value is the endpoint titer.** This method is based on the assumption that a sample with a higher protein concentration will require a higher dilution factor to achieve an absorbance just above the level of background noise.

```
endpoint_titer <- c * (((a - d) / (0.2 - d)) - 1) ^ (1 / b)

summary(endpoint_titer)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -8.113  -8.113  -8.113  -8.113  -8.113  -8.113
```

```
endpoint_titer
```

```
## [1] -8.113285
```

6.3.3 Absorbance summation method

In this model of data analysis, we sum all the absorbance values from each sample to obtain one value. This value is termed as absorption summation (AS). Using the above data, the AS will be calculated as below:

```
AS = 0.04 + 0.04 + 0.05 + 0.05 + 0.06 + 0.1 + 0.22 + 0.51 + 1.1 + 2.34 + 3.73 + 4.0

AS
```

```
## [1] 12.24
```

6.4 Use the the data analysis models in our data

The presented data is from a mouse study. In this data, presence of IgG antibody has been evaluated against receptor binding domain (RBD) of SARS-CoV-2 virus in two different groups of mice. We need to elucidate which group has higher concentration of the antibodies.

6.4.1 read in the data

```
elisa_data <- read_excel("DATA/elisa_data_serial_dilution.xlsx")
```

6.4.2 make the tidy data

```
elisa_data <- pivot_longer(data = elisa_data, cols = "Mouse_1":"Mouse_5", names_to = "m

elisa_data
```

```
## # A tibble: 100 x 4
##    Groups  Dilution mouse_id absorbance
##    <chr>   <chr>    <chr>         <dbl>
##  1 Group 1 1/50     Mouse_1         4.1
##  2 Group 1 1/50     Mouse_2         3.9
##  3 Group 1 1/50     Mouse_3         4.3
##  4 Group 1 1/50     Mouse_4         4.2
##  5 Group 1 1/50     Mouse_5         4
##  6 Group 1 1/100    Mouse_1         3.9
##  7 Group 1 1/100    Mouse_2         3.8
```

```
##  8 Group 1 1/100    Mouse_3         3.6
##  9 Group 1 1/100    Mouse_4         3.7
## 10 Group 1 1/100    Mouse_5         3.8
## # ... with 90 more rows
```

```
# separate dilution column and convert it to log2

elisa_data <- separate(elisa_data, col = "Dilution", into = c("numerator", "denomenator"), sep =

elisa_data <- elisa_data %>%
  transform(numerator = as.numeric(numerator),
            denomenator = as.numeric(denomenator))

elisa_data <- elisa_data %>%
  mutate(dilution = elisa_data$numerator / elisa_data$denomenator)

elisa_data <- elisa_data %>%
  mutate(log_dilution = log2(dilution))

elisa_data
```

```
##      Groups numerator denomenator mouse_id absorbance    dilution log_dilution
## 1   Group 1         1          50  Mouse_1       4.10 2.00000e-02    -5.643856
## 2   Group 1         1          50  Mouse_2       3.90 2.00000e-02    -5.643856
## 3   Group 1         1          50  Mouse_3       4.30 2.00000e-02    -5.643856
## 4   Group 1         1          50  Mouse_4       4.20 2.00000e-02    -5.643856
## 5   Group 1         1          50  Mouse_5       4.00 2.00000e-02    -5.643856
## 6   Group 1         1         100  Mouse_1       3.90 1.00000e-02    -6.643856
## 7   Group 1         1         100  Mouse_2       3.80 1.00000e-02    -6.643856
## 8   Group 1         1         100  Mouse_3       3.60 1.00000e-02    -6.643856
## 9   Group 1         1         100  Mouse_4       3.70 1.00000e-02    -6.643856
## 10  Group 1         1         100  Mouse_5       3.80 1.00000e-02    -6.643856
## 11  Group 1         1         200  Mouse_1       3.42 5.00000e-03    -7.643856
## 12  Group 1         1         200  Mouse_2       3.35 5.00000e-03    -7.643856
## 13  Group 1         1         200  Mouse_3       3.56 5.00000e-03    -7.643856
## 14  Group 1         1         200  Mouse_4       3.28 5.00000e-03    -7.643856
## 15  Group 1         1         200  Mouse_5       3.29 5.00000e-03    -7.643856
## 16  Group 1         1         400  Mouse_1       2.89 2.50000e-03    -8.643856
## 17  Group 1         1         400  Mouse_2       2.98 2.50000e-03    -8.643856
## 18  Group 1         1         400  Mouse_3       2.64 2.50000e-03    -8.643856
## 19  Group 1         1         400  Mouse_4       2.78 2.50000e-03    -8.643856
## 20  Group 1         1         400  Mouse_5       2.91 2.50000e-03    -8.643856
## 21  Group 1         1         800  Mouse_1       2.12 1.25000e-03    -9.643856
## 22  Group 1         1         800  Mouse_2       2.18 1.25000e-03    -9.643856
## 23  Group 1         1         800  Mouse_3       2.17 1.25000e-03    -9.643856
```

```
##  24  Group 1         1          800  Mouse_4       2.14 1.25000e-03    -9.643856
##  25  Group 1         1          800  Mouse_5       2.21 1.25000e-03    -9.643856
##  26  Group 1         1         1600  Mouse_1       1.89 6.25000e-04   -10.643856
##  27  Group 1         1         1600  Mouse_2       1.98 6.25000e-04   -10.643856
##  28  Group 1         1         1600  Mouse_3       1.76 6.25000e-04   -10.643856
##  29  Group 1         1         1600  Mouse_4       1.69 6.25000e-04   -10.643856
##  30  Group 1         1         1600  Mouse_5       1.82 6.25000e-04   -10.643856
##  31  Group 1         1         3200  Mouse_1       0.97 3.12500e-04   -11.643856
##  32  Group 1         1         3200  Mouse_2       0.85 3.12500e-04   -11.643856
##  33  Group 1         1         3200  Mouse_3       0.88 3.12500e-04   -11.643856
##  34  Group 1         1         3200  Mouse_4       0.84 3.12500e-04   -11.643856
##  35  Group 1         1         3200  Mouse_5       0.91 3.12500e-04   -11.643856
##  36  Group 1         1         6400  Mouse_1       0.52 1.56250e-04   -12.643856
##  37  Group 1         1         6400  Mouse_2       0.48 1.56250e-04   -12.643856
##  38  Group 1         1         6400  Mouse_3       0.42 1.56250e-04   -12.643856
##  39  Group 1         1         6400  Mouse_4       0.48 1.56250e-04   -12.643856
##  40  Group 1         1         6400  Mouse_5       0.47 1.56250e-04   -12.643856
##  41  Group 1         1        12800  Mouse_1       0.26 7.81250e-05   -13.643856
##  42  Group 1         1        12800  Mouse_2       0.21 7.81250e-05   -13.643856
##  43  Group 1         1        12800  Mouse_3       0.27 7.81250e-05   -13.643856
##  44  Group 1         1        12800  Mouse_4       0.26 7.81250e-05   -13.643856
##  45  Group 1         1        12800  Mouse_5       0.28 7.81250e-05   -13.643856
##  46  Group 1         1        25600  Mouse_1       0.12 3.90625e-05   -14.643856
##  47  Group 1         1        25600  Mouse_2       0.16 3.90625e-05   -14.643856
##  48  Group 1         1        25600  Mouse_3       0.17 3.90625e-05   -14.643856
##  49  Group 1         1        25600  Mouse_4       0.16 3.90625e-05   -14.643856
##  50  Group 1         1        25600  Mouse_5       0.14 3.90625e-05   -14.643856
##  51  Group 2         1           50  Mouse_1       3.20 2.00000e-02    -5.643856
##  52  Group 2         1           50  Mouse_2       3.40 2.00000e-02    -5.643856
##  53  Group 2         1           50  Mouse_3       3.60 2.00000e-02    -5.643856
##  54  Group 2         1           50  Mouse_4       3.20 2.00000e-02    -5.643856
##  55  Group 2         1           50  Mouse_5       3.70 2.00000e-02    -5.643856
##  56  Group 2         1          100  Mouse_1       2.90 1.00000e-02    -6.643856
##  57  Group 2         1          100  Mouse_2       2.80 1.00000e-02    -6.643856
##  58  Group 2         1          100  Mouse_3       2.70 1.00000e-02    -6.643856
##  59  Group 2         1          100  Mouse_4       2.80 1.00000e-02    -6.643856
##  60  Group 2         1          100  Mouse_5       2.80 1.00000e-02    -6.643856
##  61  Group 2         1          200  Mouse_1       2.32 5.00000e-03    -7.643856
##  62  Group 2         1          200  Mouse_2       2.35 5.00000e-03    -7.643856
##  63  Group 2         1          200  Mouse_3       2.56 5.00000e-03    -7.643856
##  64  Group 2         1          200  Mouse_4       2.28 5.00000e-03    -7.643856
##  65  Group 2         1          200  Mouse_5       2.29 5.00000e-03    -7.643856
##  66  Group 2         1          400  Mouse_1       1.89 2.50000e-03    -8.643856
##  67  Group 2         1          400  Mouse_2       1.98 2.50000e-03    -8.643856
##  68  Group 2         1          400  Mouse_3       1.64 2.50000e-03    -8.643856
##  69  Group 2         1          400  Mouse_4       1.78 2.50000e-03    -8.643856
```

```
## 70  Group 2         1          400  Mouse_5       1.91 2.50000e-03    -8.643856
## 71  Group 2         1          800  Mouse_1       1.12 1.25000e-03    -9.643856
## 72  Group 2         1          800  Mouse_2       1.18 1.25000e-03    -9.643856
## 73  Group 2         1          800  Mouse_3       1.17 1.25000e-03    -9.643856
## 74  Group 2         1          800  Mouse_4       1.14 1.25000e-03    -9.643856
## 75  Group 2         1          800  Mouse_5       1.21 1.25000e-03    -9.643856
## 76  Group 2         1         1600  Mouse_1       0.89 6.25000e-04   -10.643856
## 77  Group 2         1         1600  Mouse_2       0.98 6.25000e-04   -10.643856
## 78  Group 2         1         1600  Mouse_3       0.76 6.25000e-04   -10.643856
## 79  Group 2         1         1600  Mouse_4       0.69 6.25000e-04   -10.643856
## 80  Group 2         1         1600  Mouse_5       0.82 6.25000e-04   -10.643856
## 81  Group 2         1         3200  Mouse_1       0.47 3.12500e-04   -11.643856
## 82  Group 2         1         3200  Mouse_2       0.45 3.12500e-04   -11.643856
## 83  Group 2         1         3200  Mouse_3       0.48 3.12500e-04   -11.643856
## 84  Group 2         1         3200  Mouse_4       0.44 3.12500e-04   -11.643856
## 85  Group 2         1         3200  Mouse_5       0.41 3.12500e-04   -11.643856
## 86  Group 2         1         6400  Mouse_1       0.20 1.56250e-04   -12.643856
## 87  Group 2         1         6400  Mouse_2       0.28 1.56250e-04   -12.643856
## 88  Group 2         1         6400  Mouse_3       0.22 1.56250e-04   -12.643856
## 89  Group 2         1         6400  Mouse_4       0.28 1.56250e-04   -12.643856
## 90  Group 2         1         6400  Mouse_5       0.27 1.56250e-04   -12.643856
## 91  Group 2         1        12800  Mouse_1       0.16 7.81250e-05   -13.643856
## 92  Group 2         1        12800  Mouse_2       0.11 7.81250e-05   -13.643856
## 93  Group 2         1        12800  Mouse_3       0.17 7.81250e-05   -13.643856
## 94  Group 2         1        12800  Mouse_4       0.16 7.81250e-05   -13.643856
## 95  Group 2         1        12800  Mouse_5       0.18 7.81250e-05   -13.643856
## 96  Group 2         1        25600  Mouse_1       0.02 3.90625e-05   -14.643856
## 97  Group 2         1        25600  Mouse_2       0.06 3.90625e-05   -14.643856
## 98  Group 2         1        25600  Mouse_3       0.04 3.90625e-05   -14.643856
## 99  Group 2         1        25600  Mouse_4       0.08 3.90625e-05   -14.643856
## 100 Group 2         1        25600  Mouse_5       0.06 3.90625e-05   -14.643856
```

```
# converting data into dataframe

elisa_data_df <- elisa_data %>%
  group_by(Groups, mouse_id) %>%
  summarize(log_dilution = log_dilution, absorbance = absorbance)
```

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

```
elisa_data_df
```

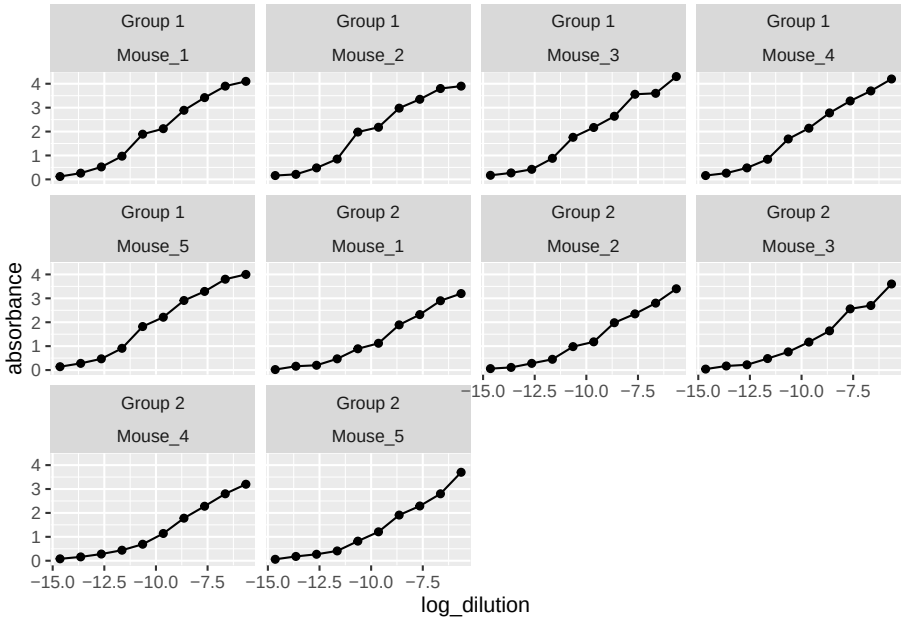
```
## # A tibble: 100 x 4
```

```
## # Groups:   Groups, mouse_id [10]
##    Groups  mouse_id log_dilution absorbance
##    <chr>   <chr>           <dbl>      <dbl>
##  1 Group 1 Mouse_1         -5.64       4.1
##  2 Group 1 Mouse_1         -6.64       3.9
##  3 Group 1 Mouse_1         -7.64       3.42
##  4 Group 1 Mouse_1         -8.64       2.89
##  5 Group 1 Mouse_1         -9.64       2.12
##  6 Group 1 Mouse_1        -10.6        1.89
##  7 Group 1 Mouse_1        -11.6        0.97
##  8 Group 1 Mouse_1        -12.6        0.52
##  9 Group 1 Mouse_1        -13.6        0.26
## 10 Group 1 Mouse_1        -14.6        0.12
## # ... with 90 more rows
```

```
elisa_data_nested <- elisa_data %>%
  group_by(Groups, mouse_id) %>%
  nest()

elisa_data_nested
```

```
## # A tibble: 10 x 3
## # Groups:   Groups, mouse_id [10]
##    Groups  mouse_id data
##    <chr>   <chr>    <list>
##  1 Group 1 Mouse_1  <tibble [10 x 5]>
##  2 Group 1 Mouse_2  <tibble [10 x 5]>
##  3 Group 1 Mouse_3  <tibble [10 x 5]>
##  4 Group 1 Mouse_4  <tibble [10 x 5]>
##  5 Group 1 Mouse_5  <tibble [10 x 5]>
##  6 Group 2 Mouse_1  <tibble [10 x 5]>
##  7 Group 2 Mouse_2  <tibble [10 x 5]>
##  8 Group 2 Mouse_3  <tibble [10 x 5]>
##  9 Group 2 Mouse_4  <tibble [10 x 5]>
## 10 Group 2 Mouse_5  <tibble [10 x 5]>
```

```
# plot the curves to evaluate the a, d, c, and b

elisa_data %>%
  ggplot(aes(x = log_dilution, y = absorbance)) +
  geom_point() +
  geom_line() +
  facet_wrap(Groups ~ mouse_id)
```



Based on the curve, the values are:

a = 4, d = 0 c = 2 b = 1

6.5 Creating a function for fitting model

```
# Creating a function for fitted model

fitted_model_elisa <- function(df_elisa, start_a, start_d, start_c, start_b) {
  mod_1 <- nlsLM(absorbance ~ ((a-d)/(1+(log_dilution/c)^b)) + d,
data = df_elisa,
start = list(a = start_a, d = start_d, c = start_c, b = start_b))
  return(mod_1)
}
```

6.5.1 Fitting the model into the data

```
fitted_model_elisa(elisa_data_nested$data[[1]], start_a = 4, start_d = 0, start_c = -8, start_b =
```

```
## Nonlinear regression model
```

```
##   model: absorbance ~ ((a - d)/(1 + (log_dilution/c)^b)) + d
##    data: df_elisa
##        a        d        c        b
##   4.3070  -0.6009 -10.2577   5.2893
##  residual sum-of-squares: 0.1199
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 1.49e-08
```

6.5.2 Apply the fitted model function to the whole dataframe

```
elisa_fitted_data <- elisa_data_nested %>%
  mutate(fitted_data = purrr::map(data, ~ fitted_model_elisa(.x, start_a = 4, start_d =

elisa_fitted_data
```

```
## # A tibble: 10 x 4
## # Groups:   Groups, mouse_id [10]
##    Groups  mouse_id data              fitted_data
##    <chr>   <chr>    <list>            <list>
##  1 Group 1 Mouse_1  <tibble [10 x 5]> <nls>
##  2 Group 1 Mouse_2  <tibble [10 x 5]> <nls>
##  3 Group 1 Mouse_3  <tibble [10 x 5]> <nls>
##  4 Group 1 Mouse_4  <tibble [10 x 5]> <nls>
##  5 Group 1 Mouse_5  <tibble [10 x 5]> <nls>
##  6 Group 2 Mouse_1  <tibble [10 x 5]> <nls>
##  7 Group 2 Mouse_2  <tibble [10 x 5]> <nls>
##  8 Group 2 Mouse_3  <tibble [10 x 5]> <nls>
##  9 Group 2 Mouse_4  <tibble [10 x 5]> <nls>
## 10 Group 2 Mouse_5  <tibble [10 x 5]> <nls>
```

6.5.3 Take out the summary of the data

```
elisa_fitted_data_summary <- elisa_fitted_data %>%
  mutate(fitted_data_summary = purrr::map(fitted_data, broom::glance))

#unnested_elisa_fitted_data <- elisa_fitted_data_summary %>%
#  unnest(elisa_fitted_data_summary) %>%
#  ungroup() %>%
#  dplyr::select(Groups, mouse_id, fitted_data)
```

```
# unnested_elisa_fitted_data$fitted_data[[1]]

# Another way

elisa_fitted_data_summary %>%
  unnest(fitted_data_summary)
```

```
## # A tibble: 10 x 13
## # Groups:   Groups, mouse_id [10]
##    Groups  mouse_id data     fitted_data  sigma isConv     finTol logLik     AIC
##    <chr>   <chr>    <list>   <list>       <dbl> <lgl>       <dbl>  <dbl>   <dbl>
##  1 Group 1 Mouse_1  <tibble> <nls>       0.141  TRUE      1.49e-8   7.93  -5.86
##  2 Group 1 Mouse_2  <tibble> <nls>       0.188  TRUE      1.49e-8   5.10  -0.208
##  3 Group 1 Mouse_3  <tibble> <nls>       0.204  TRUE      1.49e-8   4.25   1.49
##  4 Group 1 Mouse_4  <tibble> <nls>       0.132  TRUE      1.49e-8   8.65  -7.30
##  5 Group 1 Mouse_5  <tibble> <nls>       0.134  TRUE      1.49e-8   8.50  -6.99
##  6 Group 2 Mouse_1  <tibble> <nls>       0.0838 TRUE      1.49e-8  13.2  -16.3
##  7 Group 2 Mouse_2  <tibble> <nls>       0.118  TRUE      1.49e-8   9.71  -9.41
##  8 Group 2 Mouse_3  <tibble> <nls>       0.154  TRUE      1.49e-8   7.07  -4.14
##  9 Group 2 Mouse_4  <tibble> <nls>       0.0430 TRUE      1.49e-8  19.8  -29.7
## 10 Group 2 Mouse_5  <tibble> <nls>       0.126  TRUE      1.49e-8   9.11  -8.23
## # ... with 4 more variables: BIC <dbl>, deviance <dbl>, df.residual <int>,
## #   nobs <int>
```

6.5.4 Creating a function for endpoint titer model and fitting the model into the data

Chapter 7

Flow cytometry