Pertussis Resurgence MP

Vina Nguyen

2023-06-12

Mini-Project

Investigating Pertussis Resurgence

Pertussis (more commonly known as whooping cough) is a highly contagious respiratory disease caused by the bacterium Bordetella pertussis. People of all ages can be infected leading to violent coughing fits followed by a high-pitched intake of breath that sounds like "whoop". Infants and toddlers have the highest risk for severe complications and death. Recent estimates from the WHO suggest that ~16 million cases and 200,000 infant deaths are due to pertussis annually 1.

Bordetella pertussis attacks cells lining the airways. The rope-like structures shown are cilia, that typically sweep away inhaled dirt and foreign objects. In a pertussis infection, the bacteria use adhesive proteins to stick whilst releasing toxins that damage cells, trigger inflammation and increase mucus production leading to uncontrollable violent coughing.

1. Investigating pertussis cases by year

The United States Centers for Disease Control and Prevention (CDC) has been compiling reported pertussis case numbers since 1922 in their National Notifiable Diseases Surveillance System (NNDSS). We can view this data on the CDC website here: https://www.cdc.gov/pertussis/surv-reporting/cases-by-year.html

> Q1. With the help of the R "addin" package datapasta assign the CDC pertussis case number data to a data frame called cdc and use ggplot to make a plot of cases numbers over time.

```
cdc <- data.frame(
                          Year = c(1922L,
                                   1923L,1924L,1925L,1926L,1927L,1928L,
                                   1929L,1930L,1931L,1932L,1933L,1934L,1935L,
                                   1936L,1937L,1938L,1939L,1940L,1941L,
                                   1942L,1943L,1944L,1945L,1946L,1947L,1948L,
                                   1949L,1950L,1951L,1952L,1953L,1954L,
                                   1955L,1956L,1957L,1958L,1959L,1960L,
                                   1961L,1962L,1963L,1964L,1965L,1966L,1967L,
                                   1968L,1969L,1970L,1971L,1972L,1973L,
                                   1974L,1975L,1976L,1977L,1978L,1979L,1980L,
                                   1981L,1982L,1983L,1984L,1985L,1986L,
                                   1987L,1988L,1989L,1990L,1991L,1992L,1993L,
                                   1994L,1995L,1996L,1997L,1998L,1999L,
                                   2000L,2001L,2002L,2003L,2004L,2005L,
                                   2006L,2007L,2008L,2009L,2010L,2011L,2012L,
                                   2013L,2014L,2015L,2016L,2017L,2018L,
                                   2019L,2020L,2021L),
  No..Reported.Pertussis.Cases = c(107473,
                                   164191,165418,152003,202210,181411,
                                   161799,197371,166914,172559,215343,179135,
                                   265269,180518,147237,214652,227319,103188,
                                   183866,222202,191383,191890,109873,
                                   133792,109860,156517,74715,69479,120718,
                                   68687,45030,37129,60886,62786,31732,28295,
                                   32148,40005,14809,11468,17749,17135,
                                   13005,6799,7717,9718,4810,3285,4249,
                                   3036,3287,1759,2402,1738,1010,2177,2063,
                                   1623,1730,1248,1895,2463,2276,3589,
                                   4195,2823,3450,4157,4570,2719,4083,6586,
                                   4617,5137,7796,6564,7405,7298,7867,
                                   7580,9771,11647,25827,25616,15632,10454,
                                   13278,16858,27550,18719,48277,28639,
                                   32971,20762,17972,18975,15609,18617,6124,
                                   2116)
)
```

```
library(ggplot2)
```

```
x <- ggplot(cdc) +
  aes(Year, No..Reported.Pertussis.Cases) +
  geom_point() +
  geom_line() +
  labs(title = "Pertussis Casses by Year (1922-2019)")
```

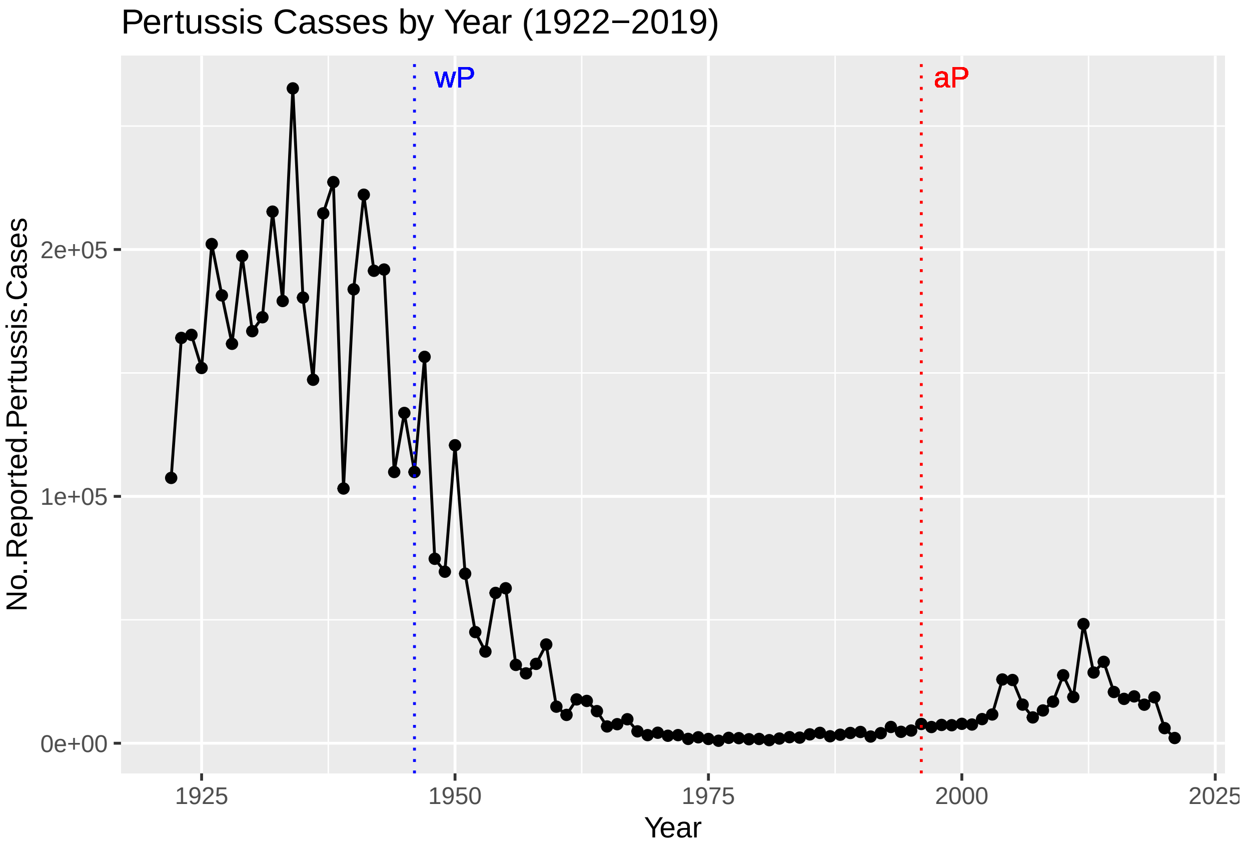
2. A tale of two vaccines (wP & aP)

Two types of pertussis vaccines are currently available: whole-cell pertussis (wP) and acellular pertussis (aP). The first vaccines were composed of 'whole cell' (wP) inactivated bacteria. The latter aP vaccines use purified antigens of the bacteria (the most important pertussis components for our immune system). These aP vaccines were developed to have less side effects than the older wP vaccines and are now the only form administered in the United States.

Let's return to our CDC data plot and examine what happened after the switch to the acellular pertussis (aP) vaccination program.

> Q2. Using the ggplot geom_vline() function add lines to your previous plot for the 1946 introduction of the wP vaccine and the 1996 switch to aP vaccine (see example in the hint below). What do you notice?

```
x + geom_vline(xintercept = 1946, linetype="dotted", color="blue") + geom_vline(xintercept = 1996, line
  geom_text(x=1950, y=270000, label="wP", color="blue") +
  geom_text(x=1999, y=270000, label="aP", color="red")
```



> Q3. Describe what happened after the introduction of the aP vaccine? Do you have a possible explanation for the observed trend?

After the introduction of the aP vaccine, there was a increase of pertussis cases. We can observe a gradual increase followed by a few spikes of cases. This increase of cases may be due to vaccination hesitancy or bacterial evolution.

> Additional points for discussion: How are vaccines currently approved? - Typically we examine 'Correlates of protection' and need to conclude a study in finite time. For the aP vaccine there is an induction of pertussis toxin (PT) antibody titers in infants at equivalent levels to those induced by the wP vaccine. The aP vaccines also had less side effects (reduction of sore arms, fever and pain). - It is impossible to discover a effect 10 years post vaccination in the current trial system. - It is unclear what differentiates people that have been primed with aP vs. wP long term. - It is unclear what differentiates people that have been primed with aP vs. wP long term.

FDA Review Staff -> FDA Inspection -> Biologics License Application(BLA)-> Vaccines and Related Biological Products Advisory Committee (VRBPAC) -> FDA Approval -> Prescribing Information and Labeling -> Vaccine Safety Surveillance -> Lot Release -> FDA Regulatory Research

FDA Inspection: The FDA inspects manufacturing facilities to examine and evaluate the process used to make the vaccine (expert investigators will determine if the process is compliant with FDA requirements).

FDA Approval: The FDA evaluates more data and manufacturing information as part of a BLA. Determines whether or not the vaccine is safe and effective for its intended use. If so, FDA will approve and license the vaccine.

3. Exploring CMI-PB data

Why is this vaccine-preventable disease on the upswing? To answer this question we need to investigate the mechanisms underlying waning protection against pertussis. This requires evaluation of pertussis-specific immune responses over time in wP and aP vaccinated individuals.

The new and ongoing CMI-PB project aims to provide the scientific community with this very information. In particular, CMI-PB tracks and makes freely available long-term humoral and cellular immune response data for a large number of individuals who received either DTwP or DTaP combination vaccines in infancy followed by Tdap booster vaccinations. This includes complete API access to longitudinal RNA-Seq, AB Titer, Olink, and live cell assay results directly from their website: https://www.cmi-pb.org/

The CMI-PB API returns JSON data The CMI-PB API (like most APIs) sends responses in JSON format. Briefly, JSON data is formatted as a series of key-value pairs, where a particular word ("key") is associated with a particular value. An example of the JSON format for Ab titer data is shown below:

{ "specimen_id":1, "isotype":"IgG", "is_antigen_specific":true, "antigen":"PT", "ab_titer":68.5661390514946, "unit":"IU/ML", "lower_limit_of_detection":0.53 }

To read these types of files into R we will use the read_json() function from the jsonlite package. Note that if you want to do more advanced querys of APIs directly from R you will likely want to explore the more full featured rjson package. The big advantage of using jsonlite for our current purposes is that it can simplify JSON key-value pair arrays into R data frames without much additional effort on our part.

```r
# Allows us to read, write and process JSON data
library(jsonlite)
```

Let's now read the main subject database table from the CMI-PB API. You can find out more about the content and format of this and other tables here: https://www.cmi-pb.org/blog/understand-data/

```r
subject <- read_json("https://www.cmi-pb.org/api/subject", simplifyVector = TRUE)
```

Key-point: The subject table provides metadata about each individual in the study group. For example, their infancy vaccination type, biological sex, year of birth, time of boost etc.

```r
head(subject, 3)
```

```
##   subject_id infancy_vac biological_sex              ethnicity  race
## 1          1          wP         Female Not Hispanic or Latino White
## 2          2          wP         Female Not Hispanic or Latino White
## 3          3          wP         Female                Unknown White
##   year_of_birth date_of_boost      dataset
## 1    1986-01-01    2016-09-12 2020_dataset
## 2    1968-01-01    2019-01-28 2020_dataset
## 3    1983-01-01    2016-10-10 2020_dataset
```

> Q4. How may aP and wP infancy vaccinated subjects are in the dataset?

```r
table(subject$infancy_vac)
```

```
## 
## aP wP 
## 47 49
```

> Q5. How many Male and Female subjects/patients are in the dataset?

```
sum(subject$biological_sex=="Female")
```

```
## [1] 66
```

```
sum(subject$biological_sex=="Male")
```

```
## [1] 30
```

> Q6. What is the breakdown of race and biological sex (e.g. number of Asian females, White males etc…)?

```
table(subject$biological_sex, subject$race)
```

```
##
##           American Indian/Alaska Native Asian Black or African American
##   Female                             0    18                         2
##   Male                               1     9                         0
##
##           More Than One Race Native Hawaiian or Other Pacific Islander
##   Female                  8                                         1
##   Male                    2                                         1
##
##           Unknown or Not Reported White
##   Female                      10    27
##   Male                         4    13
```

Side-Note: Working with dates

Two of the columns of subject contain dates in the Year-Month-Day format. Recall from our last mini-project that dates and times can be annoying to work with at the best of times. However, in R we have the excellent lubridate package, which can make life allot easier. Here is a quick example to get you started:

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

What is today's date (at the time I am writing this obviously).

```
today()
```

```
## [1] "2023-06-12"
```

How many days have passed since new year 2000?

```r
today() - ymd("2000-01-01")
```

```
## Time difference of 8563 days
```

What is this in years?

```r
time_length( today() - ymd("2000-01-01"),  "years")
```

```
## [1] 23.44422
```

Note that here we are using the ymd() function to tell lubridate the format of our particular date and then the time_length() function to convert days to years.

> Q7. Using this approach determine (i) the average age of wP individuals, (ii) the average age of aP individuals; and (iii) are they significantly different?

```r
# Use today's date to calculate age in days
subject$age <- today() - ymd(subject$year_of_birth)
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
ap <- subject %>% filter(infancy_vac == "aP")

round( summary( time_length( ap$age, "years" ) ) )
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      23      25      26      26      26      27
```

```r
# wP
wp <- subject %>% filter(infancy_vac == "wP")
round( summary( time_length( wp$age, "years" ) ) )
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      28      32      35      37      40      55
```

> Q8. Determine the age of all individuals at time of boost?

```
subject$age <- ymd(subject$date_of_boost) - ymd(subject$year_of_birth)
```

```
ap <- subject %>% filter(infancy_vac == "aP")

round( summary( time_length( ap$age, "years" ) ) )
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##      19      20      20      20      21      23
```

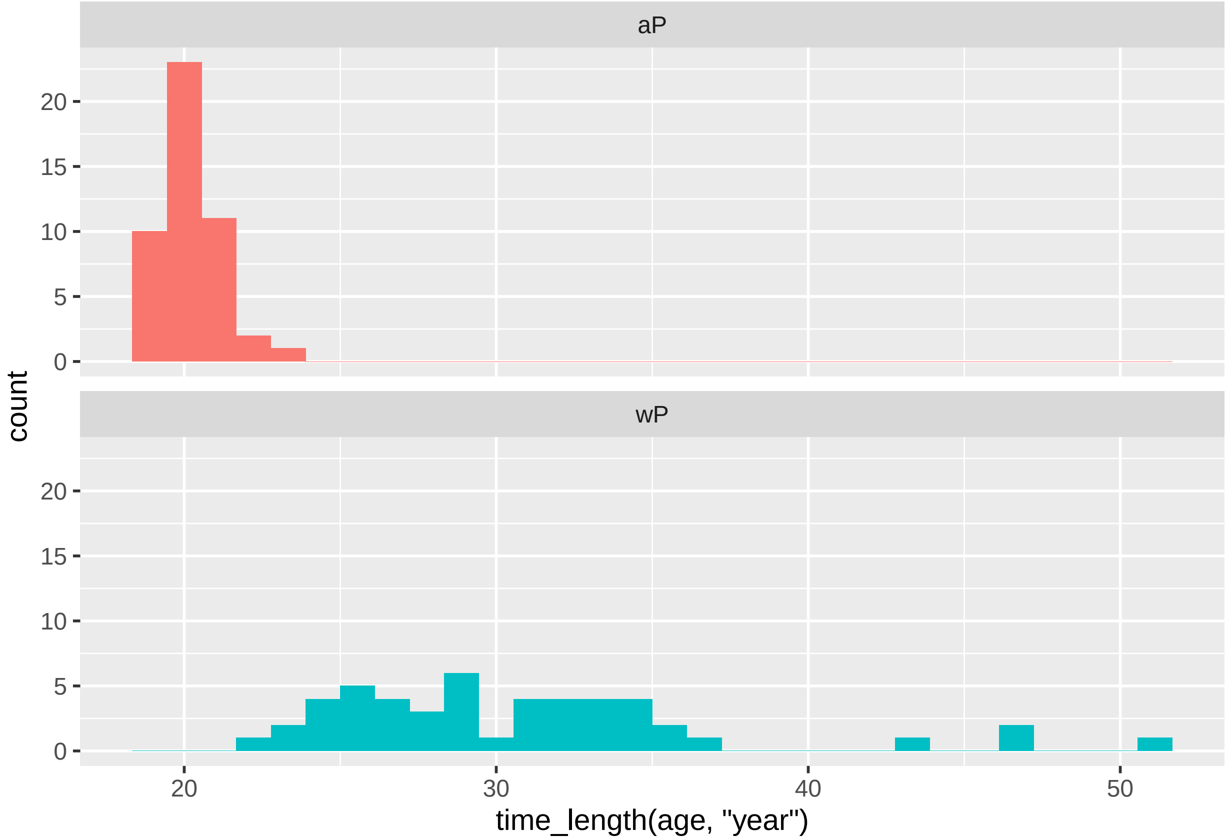
```
# wP
wp <- subject %>% filter(infancy_vac == "wP")
round( summary( time_length( wp$age, "years" ) ) )
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##      23      26      29      31      34      51
```

Q9. With the help of a faceted boxplot (see below), do you think these two groups are significantly different?

```
ggplot(subject) +
  aes(time_length(age, "year"),
      fill=as.factor(infancy_vac)) +
  geom_histogram(show.legend=FALSE) +
  facet_wrap(vars(infancy_vac), nrow=2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Yes these groups are significantly different since the medians do not overlap.

Joining multiple tables Read the specimen and ab_titer tables into R and store the data as specimen and titer named data frames.

```
# Complete the API URLs...
specimen <- read_json("https://www.cmi-pb.org/api/specimen", simplifyVector = TRUE)
titer <- read_json("https://www.cmi-pb.org/api/ab_titer", simplifyVector = TRUE)
```

To know whether a given specimen_id comes from an aP or wP individual we need to link (a.k.a. "join" or merge) our specimen and subject data frames. The excellent dplyr package (that we have used previously) has a family of join() functions that can help us with this common task:

> Q9. Complete the code to join specimen and subject tables to make a new merged data frame containing all specimen records along with their associated subject details:

```
meta <- inner_join(specimen, subject)
```

```
## Joining with `by = join_by(subject_id)`
```

```
dim(meta)
```

```
## [1] 729  14
```

```
head(meta)
```

```
##   specimen_id subject_id actual_day_relative_to_boost
## 1           1          1                           -3
## 2           2          1                          736
## 3           3          1                            1
## 4           4          1                            3
## 5           5          1                            7
## 6           6          1                           11
##   planned_day_relative_to_boost specimen_type visit infancy_vac biological_sex
## 1                             0         Blood     1          wP         Female
## 2                           736         Blood    10          wP         Female
## 3                             1         Blood     2          wP         Female
## 4                             3         Blood     3          wP         Female
## 5                             7         Blood     4          wP         Female
## 6                            14         Blood     5          wP         Female
##                ethnicity  race year_of_birth date_of_boost      dataset
## 1 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 2 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 3 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 4 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 5 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 6 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
##          age
## 1 11212 days
## 2 11212 days
## 3 11212 days
## 4 11212 days
## 5 11212 days
## 6 11212 days
```

> Q10. Now using the same procedure join meta with titer data so we can further analyze this data in terms of time of visit aP/wP, male/female etc.

```
abdata <- inner_join(titer, meta)
```

```
## Joining with ‘by = join_by(specimen_id)‘
```

```
dim(abdata)
```

```
## [1] 32675    21
```

> Q11. How many specimens (i.e. entries in abdata) do we have for each isotype?

```
table(abdata$isotype)
```

```
##
##  IgE  IgG IgG1 IgG2 IgG3 IgG4
## 6698 1413 6141 6141 6141 6141
```

> Q12. What do you notice about the number of visit 8 specimens compared to other visits?

```
table(abdata$visit)
```

```
## 
##    1    2    3    4    5    6    7    8 
## 5795 4640 4640 4640 4640 4320 3920   80
```

4. Examine IgG1 Ab titer levels

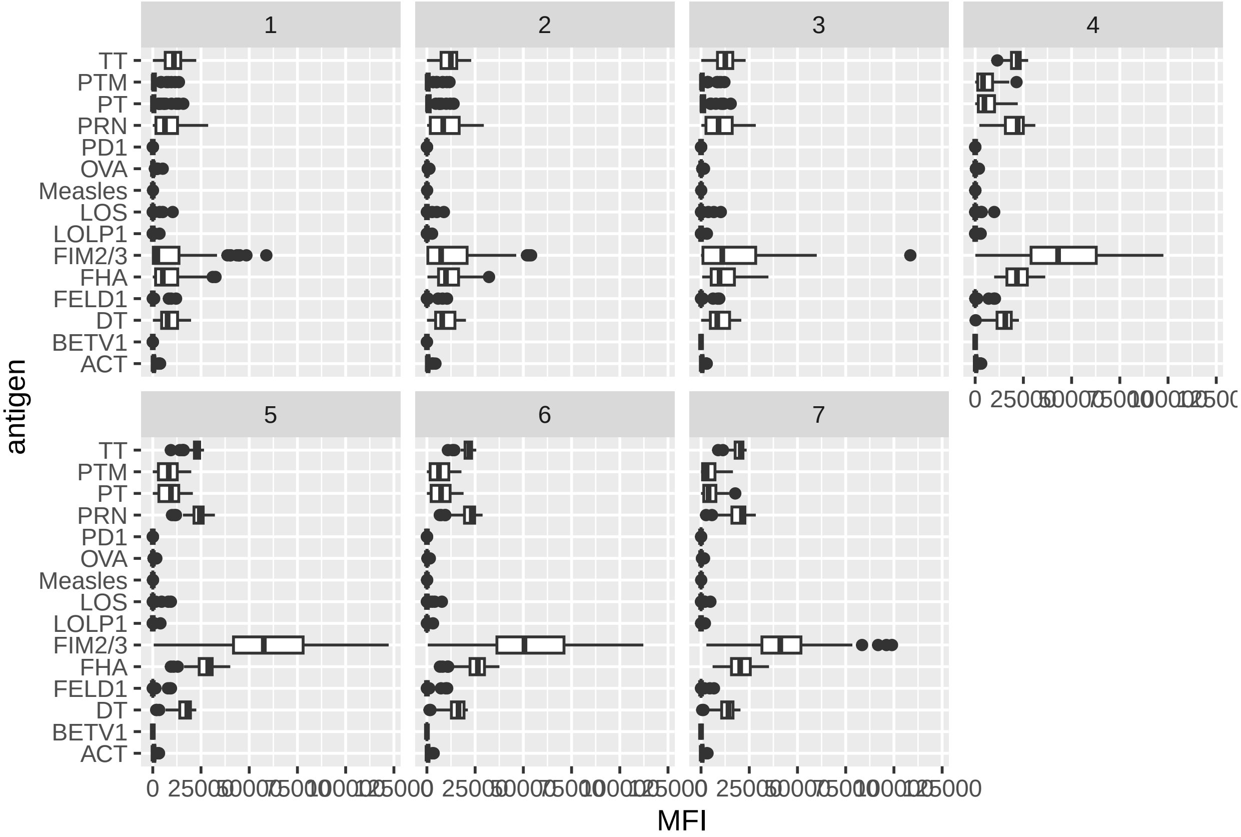
Now using our joined/merged/linked abdata dataset filter() for IgG1 isotype and exclude the small number of visit 8 entries.

```
ig1 <- abdata %>% filter(isotype == "IgG1", visit!=8)
head(ig1)
```

```
##   specimen_id isotype is_antigen_specific antigen        MFI MFI_normalised
## 1           1    IgG1                TRUE     ACT 274.355068      0.6928058
## 2           1    IgG1                TRUE     LOS  10.974026      2.1645083
## 3           1    IgG1                TRUE   FELD1   1.448796      0.8080941
## 4           1    IgG1                TRUE   BETV1   0.100000      1.0000000
## 5           1    IgG1                TRUE   LOLP1   0.100000      1.0000000
## 6           1    IgG1                TRUE Measles  36.277417      1.6638332
##    unit lower_limit_of_detection subject_id actual_day_relative_to_boost
## 1 IU/ML                 3.848750          1                           -3
## 2 IU/ML                 4.357917          1                           -3
## 3 IU/ML                 2.699944          1                           -3
## 4 IU/ML                 1.734784          1                           -3
## 5 IU/ML                 2.550606          1                           -3
## 6 IU/ML                 4.438966          1                           -3
##   planned_day_relative_to_boost specimen_type visit infancy_vac biological_sex
## 1                             0         Blood     1          wP         Female
## 2                             0         Blood     1          wP         Female
## 3                             0         Blood     1          wP         Female
## 4                             0         Blood     1          wP         Female
## 5                             0         Blood     1          wP         Female
## 6                             0         Blood     1          wP         Female
##                ethnicity  race year_of_birth date_of_boost      dataset
## 1 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 2 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 3 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 4 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 5 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
## 6 Not Hispanic or Latino White    1986-01-01    2016-09-12 2020_dataset
##          age
## 1 11212 days
## 2 11212 days
## 3 11212 days
## 4 11212 days
## 5 11212 days
## 6 11212 days
```

Q13. Complete the following code to make a summary boxplot of Ab titer levels for all antigens:

```
ggplot(ig1) +
  aes(MFI, antigen) +
  geom_boxplot() +
  facet_wrap(vars(visit), nrow=2)
```

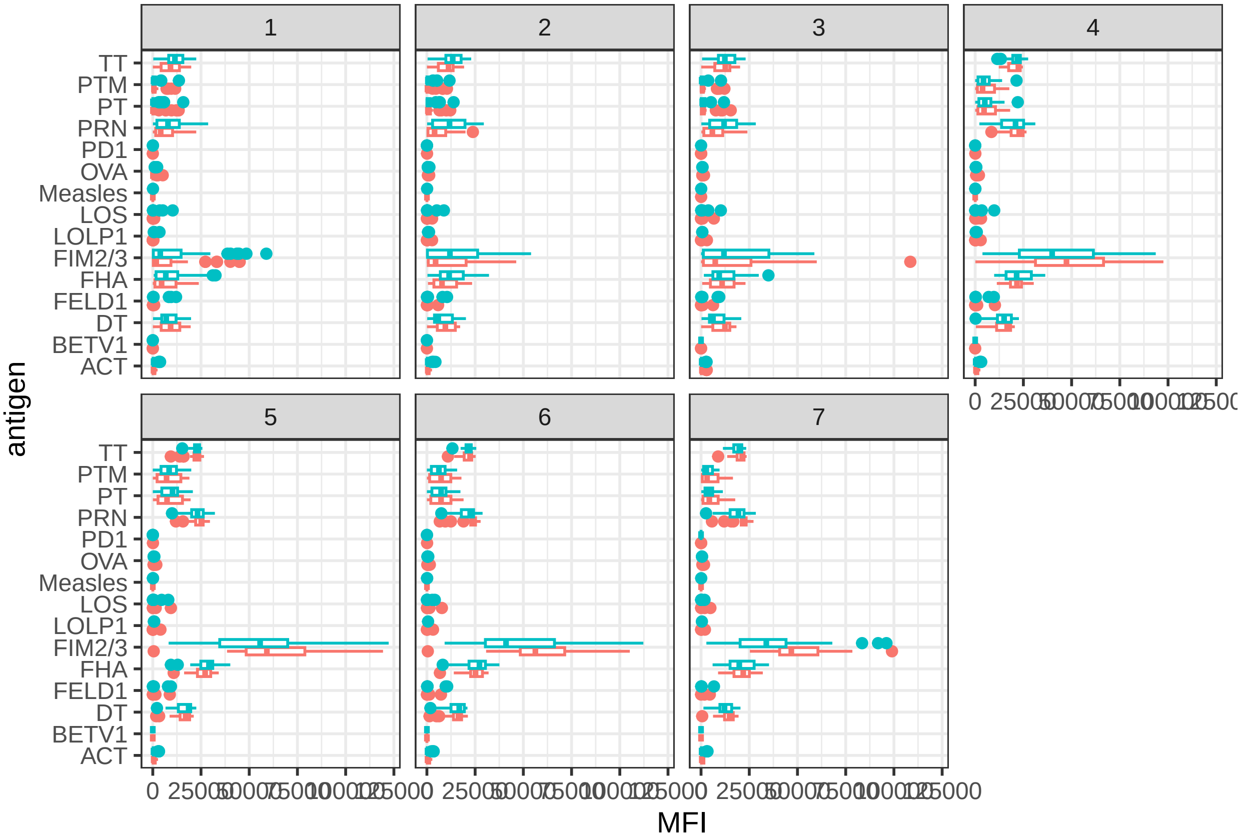


Q14. What antigens show differences in the level of IgG1 antibody titers recognizing them over time? Why these and not others?

FIM2/3 and FHA. Might take long because antibody molecule slowly contacts antigen.

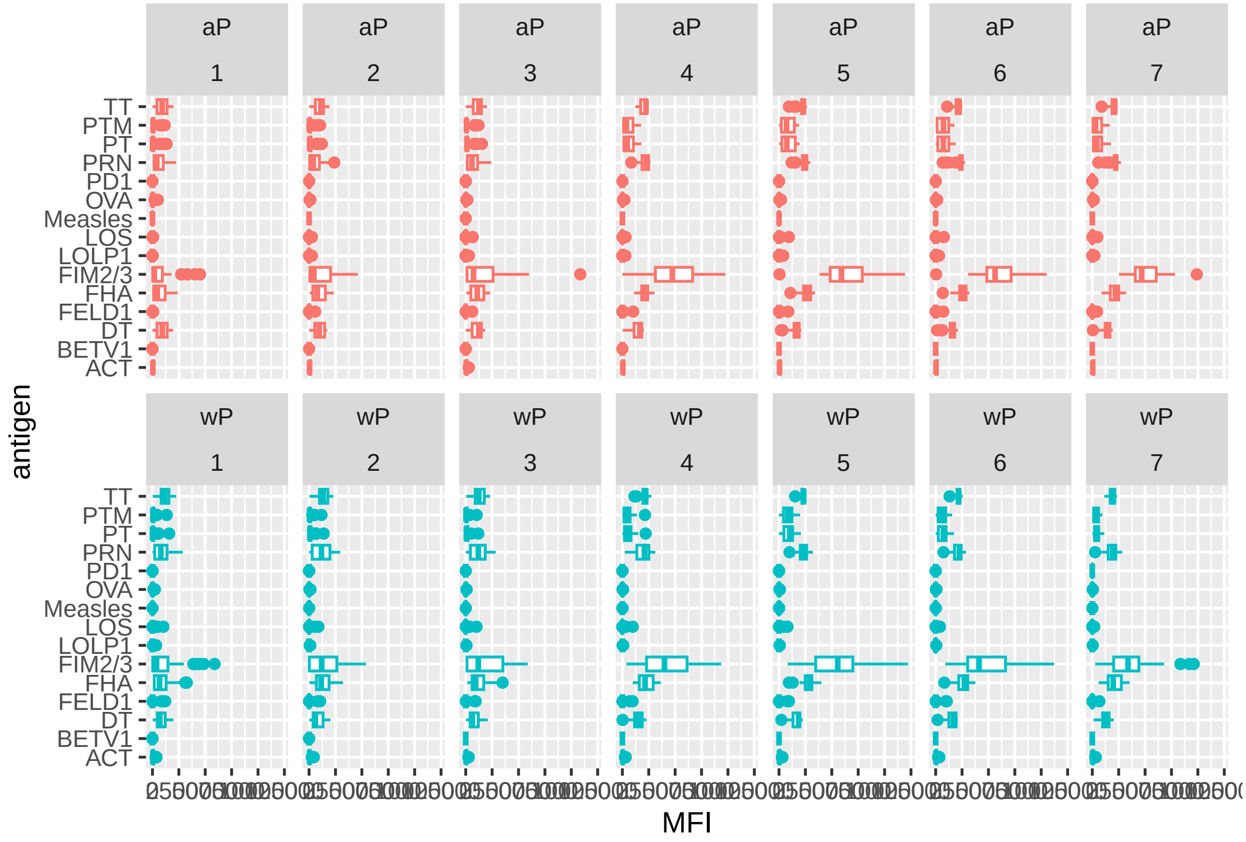
We can attempt to examine differences between wP and aP here by setting color and/or facet values of the plot to include infancy_vac status (see below). However these plots tend to be rather busy and thus hard to interpret easily.

```
ggplot(ig1) +
  aes(MFI, antigen, col=infancy_vac ) +
  geom_boxplot(show.legend = FALSE) +
  facet_wrap(vars(visit), nrow=2) +
  theme_bw()
```



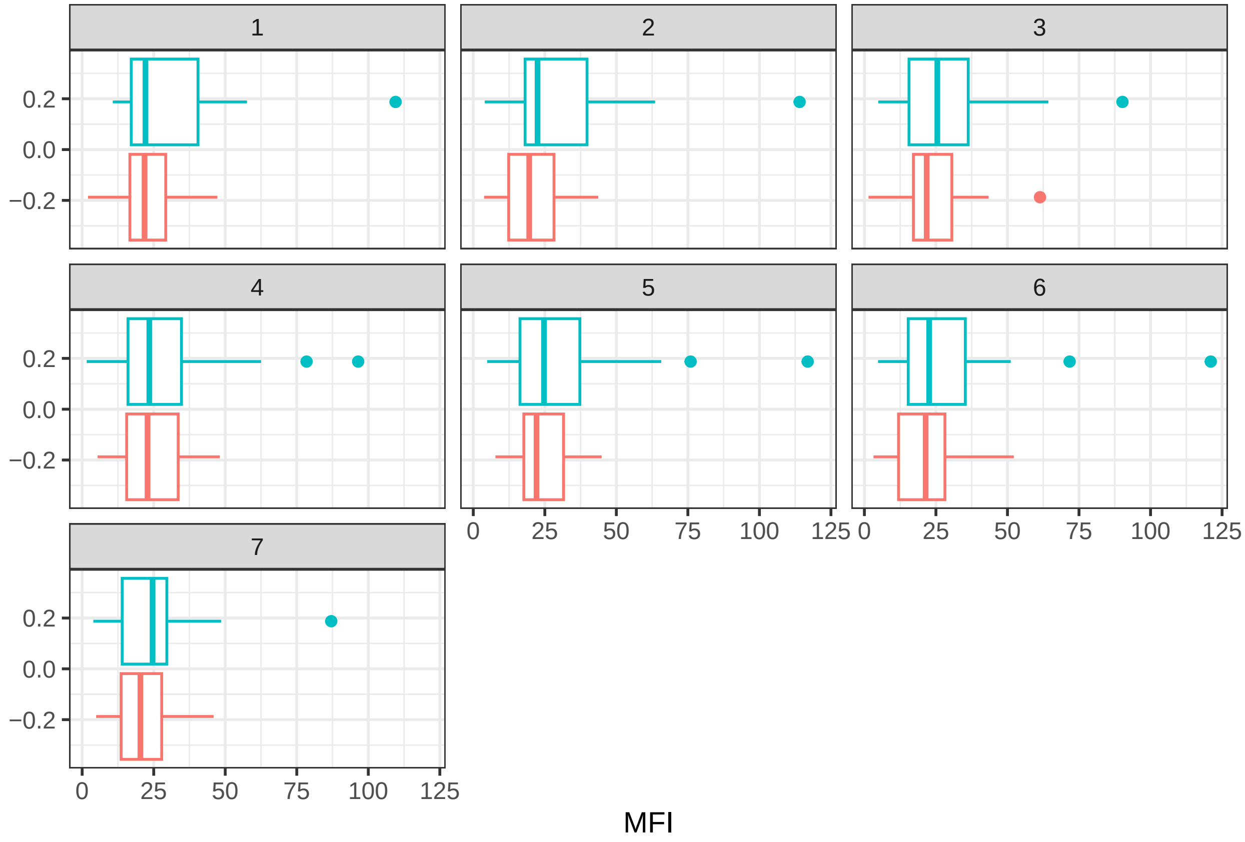
Another version of this plot adding infancy_vac to the faceting:

```
ggplot(ig1) +
  aes(MFI, antigen, col=infancy_vac ) +
  geom_boxplot(show.legend = FALSE) +
  facet_wrap(vars(infancy_vac, visit), nrow=2)
```



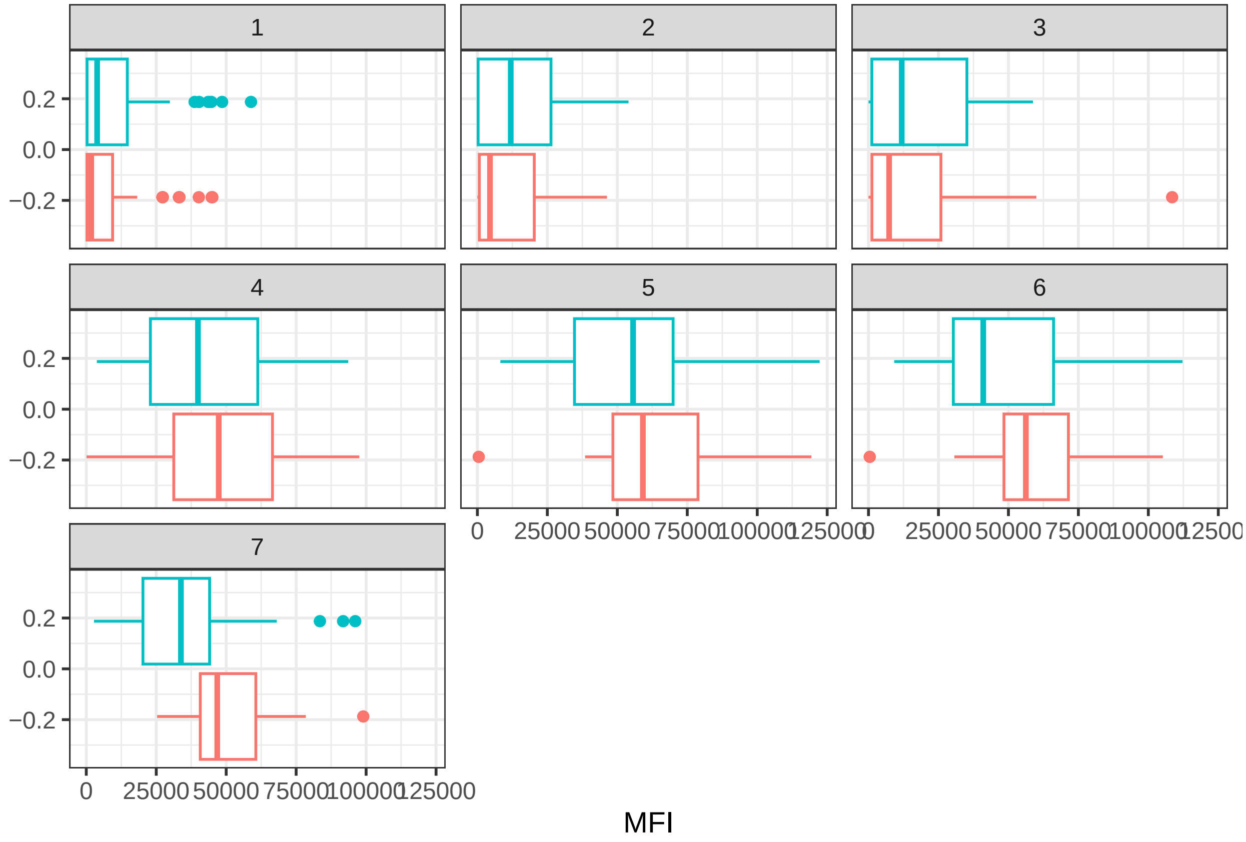
Q15. Filter to pull out only two specific antigens for analysis and create a boxplot for each. You can chose any you like. Below I picked a "control" antigen ("Measles", that is not in our vaccines) and a clear antigen of interest ("FIM2/3", extra-cellular fimbriae proteins from B. pertussis that participate in substrate attachment

```
filter(ig1, antigen=="Measles") %>%
  ggplot() +
  aes(MFI, col=infancy_vac) +
  geom_boxplot(show.legend = FALSE) +
  facet_wrap(vars(visit)) +
  theme_bw()
```



and the same for antigen==“FIM2/3”

```
filter(ig1, antigen=="FIM2/3") %>%
  ggplot() +
  aes(MFI, col=infancy_vac) +
  geom_boxplot(show.legend = FALSE) +
  facet_wrap(vars(visit)) +
  theme_bw()
```



Q16. What do you notice about these two antigens time course and the FIM2/3 data in particular?

FIM2/3 increases over time and exceed those of Measles.

Q17. Do you see any clear difference in aP vs. wP responses?

aP (in red) increases faster than wP (in blue).

5. Obtaining CMI-PB RNASeq data

For RNA-Seq data the API query mechanism quickly hits the web browser interface limit for file size. We will present alternative download mechanisms for larger CMI-PB datasets in the next section. However, we can still do "targeted" RNA-Seq querys via the web accessible API.

For example we can obtain RNA-Seq results for a specific ENSEMBLE gene identifier or multiple identifiers combined with the & character:

For example use the following URL https://www.cmi-pb.org/api/v2/rnaseq?versioned_ensembl_gene_id=eq.ENSG00000211896.7

The link above is for the key gene involved in expressing any IgG1 antibody, namely the IGHG1 gene. Let's read available RNA-Seq data for this gene into R and investigate the time course of it's gene expression values.

```
url <- "https://www.cmi-pb.org/api/v2/rnaseq?versioned_ensembl_gene_id=eq.ENSG00000211896.7"

rna <- read_json(url, simplifyVector = TRUE)
```

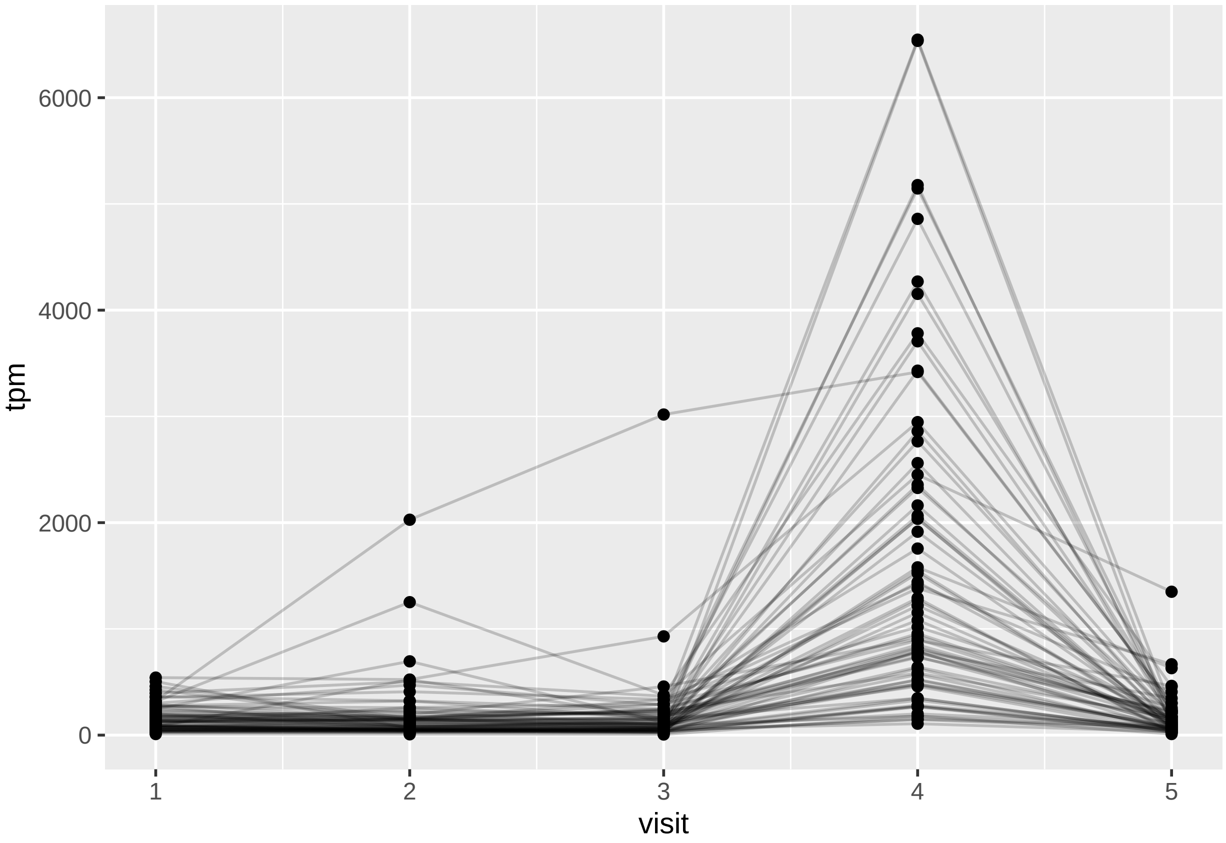
To facilitate further analysis we need to "join" the rna expression data with our metadata meta, which is itself a join of sample and specimen data. This will allow us to look at this genes TPM expression values over aP/wP status and at different visits (i.e. times):

```
#meta <- inner_join(specimen, subject)
ssrna <- inner_join(rna, meta)
```

```
## Joining with ‘by = join_by(specimen_id)‘
```

> Q18. Make a plot of the time course of gene expression for IGHG1 gene (i.e. a plot of visit vs. tpm).

```
ggplot(ssrna) +
  aes(visit, tpm, group=subject_id) +
  geom_point() +
  geom_line(alpha=0.2)
```



> Q19.: What do you notice about the expression of this gene (i.e. when is it at it's maximum level)?

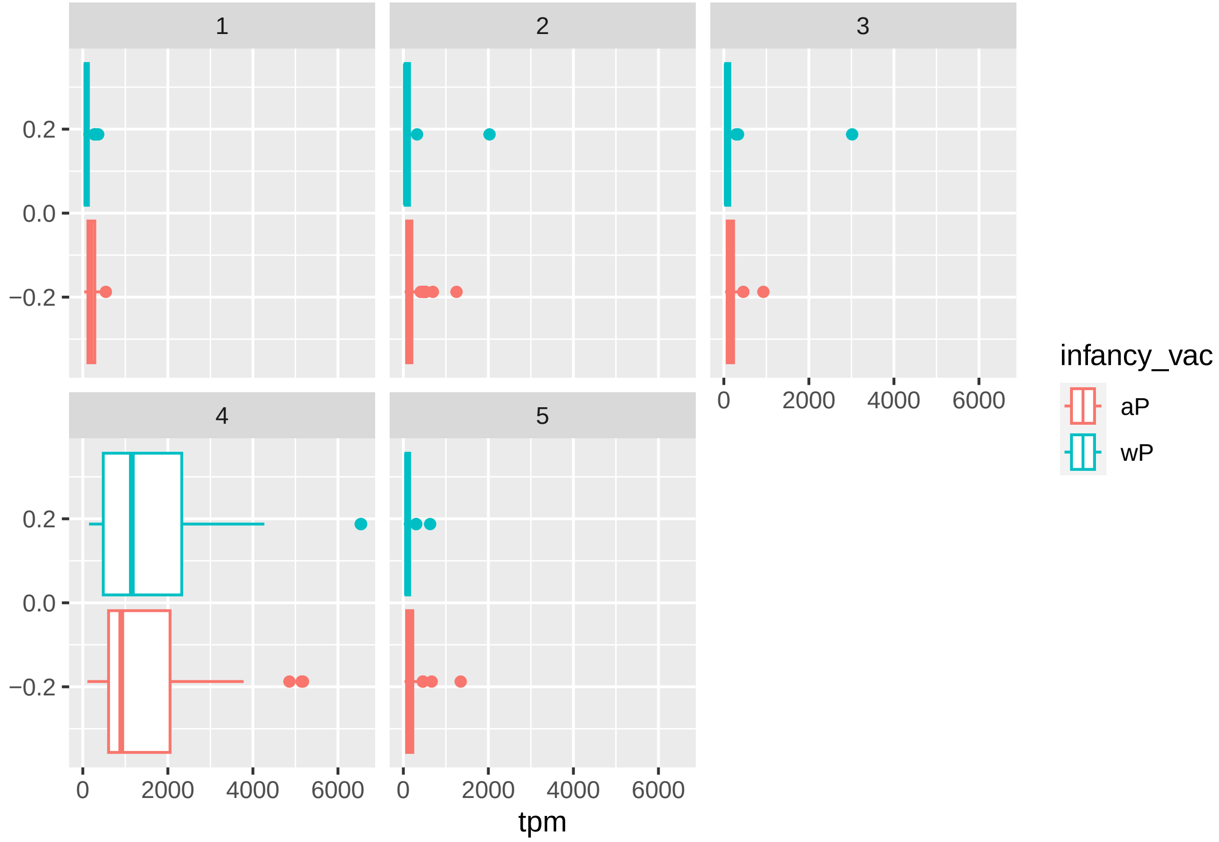
Tpm is low and similar between the 1st to 3rd visit. However there is a spike reaching max level at the fourth visit which drops at the 5th.

> Q20. Does this pattern in time match the trend of antibody titer data? If not, why not?

Yes because cells make antibodies that are long-lived.

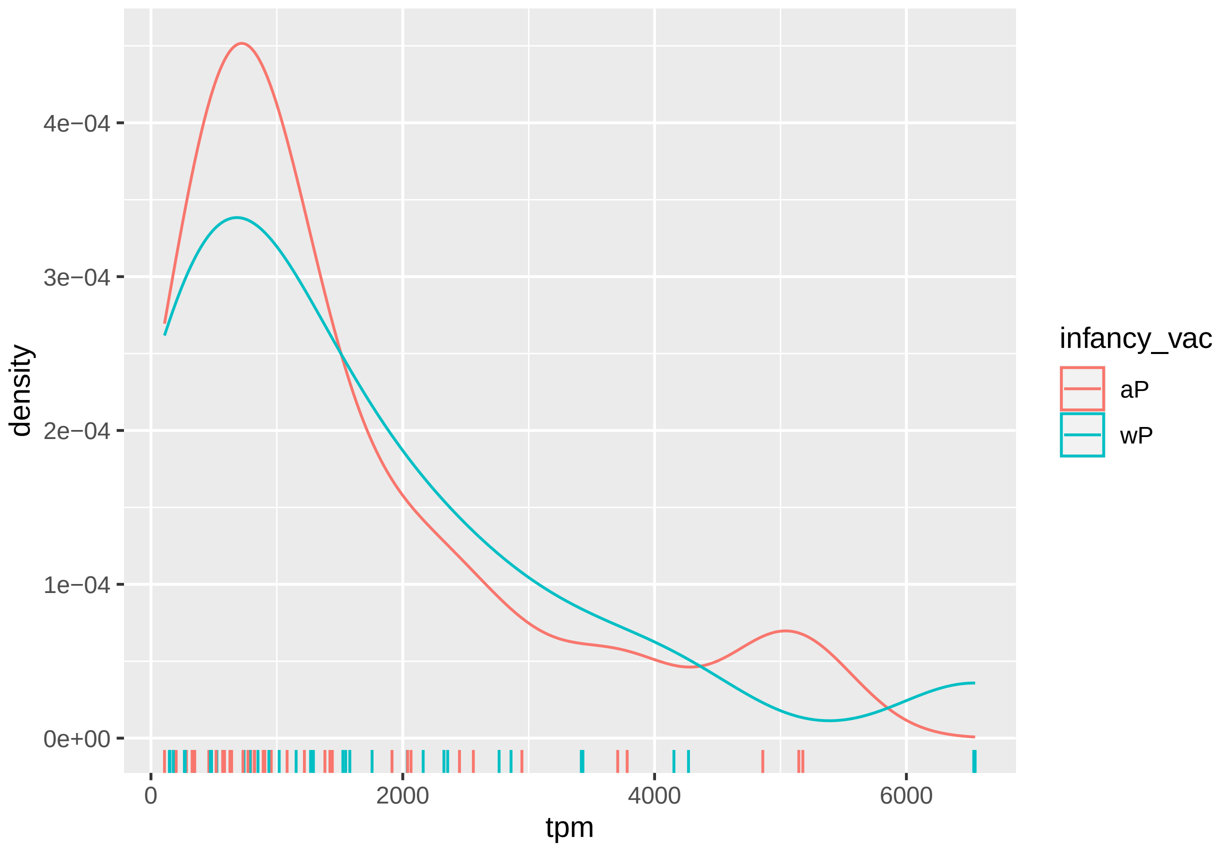
We can dig deeper and color and/or facet by infancy_vac status:

```
ggplot(ssrna) +
  aes(tpm, col=infancy_vac) +
  geom_boxplot() +
  facet_wrap(vars(visit))
```



There is no obvious wP vs. aP differences here even if we focus in on a particular visit:

```
ssrna %>%
  filter(visit==4) %>%
  ggplot() +
    aes(tpm, col=infancy_vac) + geom_density() +
    geom_rug()
```



```
sessionInfo()
```

```
## R version 4.2.3 (2023-03-15 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19044)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] dplyr_1.1.2     lubridate_1.9.2 jsonlite_1.8.4  ggplot2_3.4.2
##
## loaded via a namespace (and not attached):
##  [1] pillar_1.9.0      compiler_4.2.3   highr_0.10       tools_4.2.3
##  [5] digest_0.6.31     evaluate_0.21    lifecycle_1.0.3  tibble_3.2.1
##  [9] gtable_0.3.3      timechange_0.2.0 pkgconfig_2.0.3  rlang_1.1.0
## [13] cli_3.6.1         rstudioapi_0.14  yaml_2.3.7       xfun_0.39
## [17] fastmap_1.1.1    withr_2.5.0      knitr_1.43       generics_0.1.3  
## [21] vctrs_0.6.2      grid_4.2.3       tidyselect_1.2.0 glue_1.6.2      
## [25] R6_2.5.1         fansi_1.0.4      rmarkdown_2.22   farver_2.1.1    
## [29] magrittr_2.0.3   scales_1.2.1     htmltools_0.5.5  colorspace_2.1-0
## [33] labeling_0.4.2   utf8_1.2.3       munsell_0.5.0
```