Exercise 3: High-dimensional data

Éric Marty

May 23, 2022

Early in the COVID-19 pandemic, researchers needed to be able to understand how this outbreak might differ from previous major viral outbreaks. Quantities such as the disease's *incubation period*, *infectious period*, and *basic reproduction number* can be estimated from early epidemiological data.

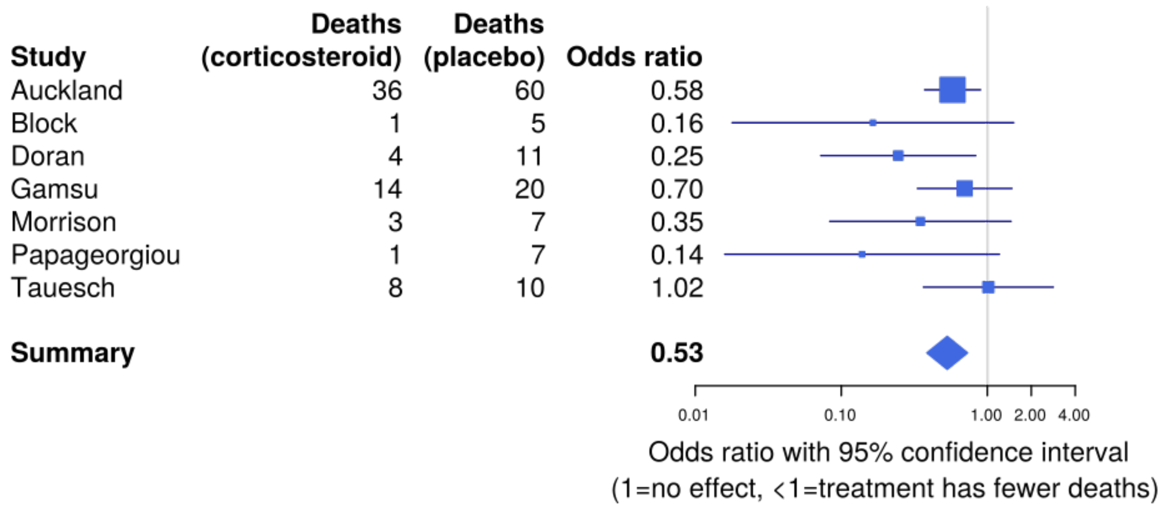
Researchers in the CEID Coronavirus Working Group conducted a literature review and collected parameter estimates for COVID-19 and seven earlier outbreaks of MERS, SARS, Ebola, and influenza. The data are provided in the file `outbreakparams.csv`.

Original data as well as data descriptions can be found here.

```r
library(tidyverse)
data <- read_csv('../data/outbreakparams.csv')  # read in csv as a "tibble" dataframe.
```

For each disease outbreak, there are multiple estimates from multiple publications of each parameter. A "forest plot" can be used to visualize the results of several studies in a meta-analysis in a single plot.

Here is an example of a forest plot made with the `forestplot` function from the `rmeta` package:



We can adapt the basic idea of a forest plot using functions from base R, and from there build up tools to deal with the high dimensional nature of our data (multiple parameters, multiple outbreaks.)

First, let's filter our data to look at just a single parameter for a singe outbreak, across all studies.

```
outbreaks <- data$outbreak %>% unique()
noutbreaks <- length(outbreaks)

plotdata <- data %>%
  filter(parameter %in% c("R0")) %>%
  filter(outbreak == "2019-2020 global nCoV outbreak") %>%
  select(outbreak, parameter, estimate, lowerBound, upperBound)
plotdata
```

```
## # A tibble: 17 x 5
##    outbreak                       parameter estimate lowerBound upperBound
##    <chr>                          <chr>        <dbl>      <dbl>      <dbl>
##  1 2019-2020 global nCoV outbreak R0            2.2        1.4        3.9
##  2 2019-2020 global nCoV outbreak R0            2.2        1.4        3.8
##  3 2019-2020 global nCoV outbreak R0            2.68       2.47       2.86
##  4 2019-2020 global nCoV outbreak R0            2.9        2.1        4.5
##  5 2019-2020 global nCoV outbreak R0            2.1       NA         NA
##  6 2019-2020 global nCoV outbreak R0           NA          1.3        3.2
##  7 2019-2020 global nCoV outbreak R0            6.3        3.3       11.3
##  8 2019-2020 global nCoV outbreak R0            4.7        2.8        7.6
##  9 2019-2020 global nCoV outbreak R0            6.6        4         10.5
## 10 2019-2020 global nCoV outbreak R0            4.9        3.3        7.2
## 11 2019-2020 global nCoV outbreak R0            7.05       6.11       8.18
## 12 2019-2020 global nCoV outbreak R0            3.24       3.16       3.32
## 13 2019-2020 global nCoV outbreak R0            2.28       2.06       2.52
## 14 2019-2020 global nCoV outbreak R0            1.4        1.04       1.85
## 15 2019-2020 global nCoV outbreak R0            2.17       1.69       2.76
## 16 2019-2020 global nCoV outbreak R0            1.58       1.29       1.92
## 17 2019-2020 global nCoV outbreak R0           NA          2.76       3.25
```

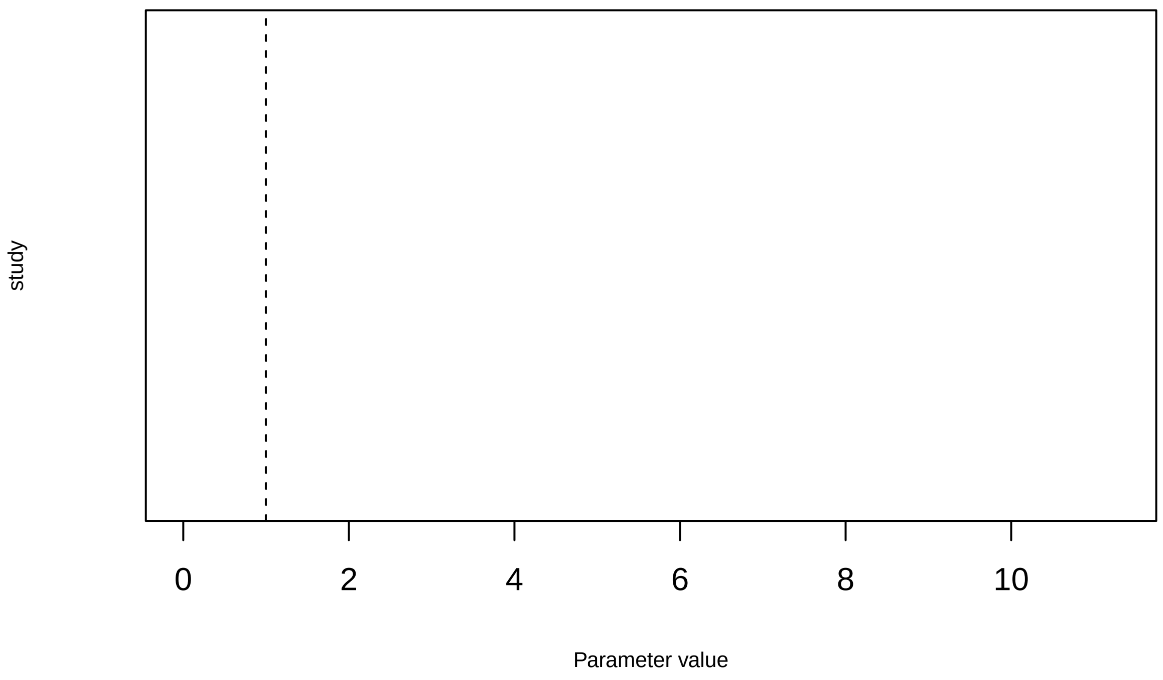
Next, we need some metadata to help us define the visual parameters of our visualization.

```
nest <- nrow(plotdata) # number of estimates
plotmax <- max(plotdata$upperBound, na.rm = TRUE) # maximum range value for plotting
plotmin <- 0 # minimum range value (we use zero since this is quantitative "ratio" data with zero as th
```

Now, we can build a basic forest plot. We start by defining and empty plot are encompassing the range of our data, and add axes.

Reproduction number is a measure of how many secondary infections arise on average from each infection. This measure has a critical value of 1. If the value is below 1, the disease will tend to die out. If the value is 1 or greater, an epidemic can be sustained or grow. We will add a vertical reference line at critical value of 1.

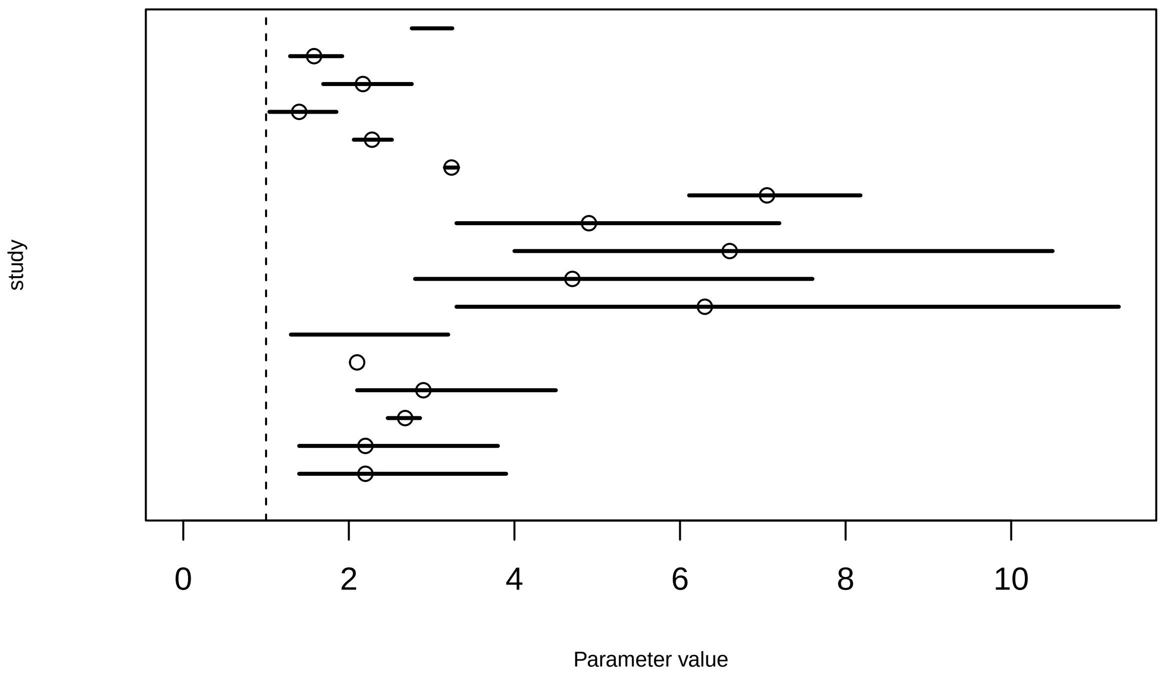
```
# Begin with an empty plot with a dummy data
plot(x=0, y=0, type = "n", # dummy data
     xlim=c(plotmin,plotmax), # axis limits
     ylim = c(0,nest), # axis limits
     xlab='Parameter value', # axis labes
     ylab='study', yaxt='n', xaxt='n', cex.lab = .65) # axis labels
axis(1) # add an x axis
abline(v=1.0, lty=2) # add a vertical dotted reference line at R_0 = 1
```



Next, let's add point estimates and a range line for each estimate.

```
# point estimates
points(x=plotdata$estimate, y = 1:nest)

# range lines
for(i in 1:nest) {
  lines(x = c(plotdata$lowerBound[i], plotdata$upperBound[i]), # endpoints (x)
        y = rep(i, each=2), # endpoints (y)
        lwd = 2) # line width
}
```



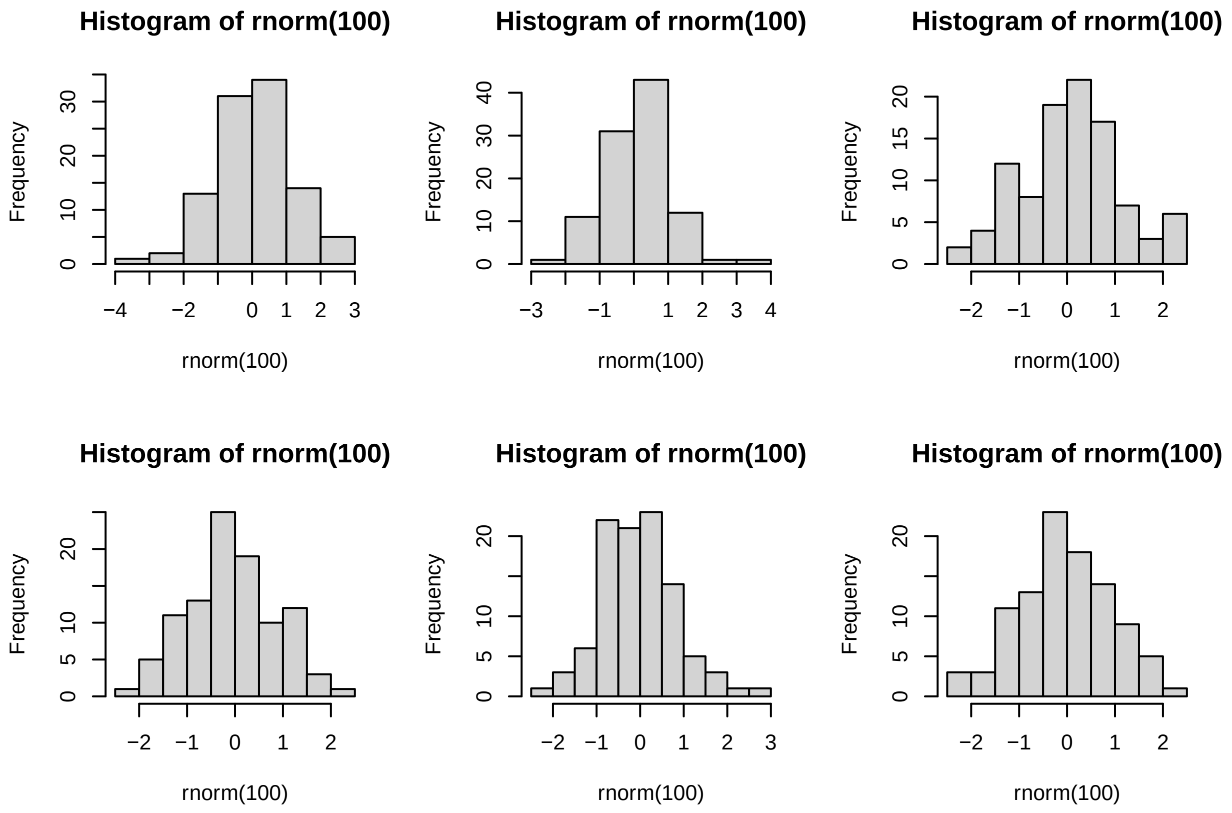
Now, we have an idea of what our data look like.

Exercise 1a.

Write a function to generate a small multiples figure based on the above forest plot. Edit the basic forest plot code to add appropriate labels and annotations, including reference lines for the critical values (if any).

You can use `par(mfrow=c(nrows,ncols))` to define a plotting matrix with n rows and n columns. Each subsequent plot will be drawn in a matrix cell, filling the matrix by row. (If you need to fill by column instead, you can use `par(mfcol=c(nrows,ncols))`).

```
par(mfcol=c(2,3))  # rows, columns
hist(rnorm(100))
hist(rnorm(100))
hist(rnorm(100))
hist(rnorm(100))
hist(rnorm(100))
hist(rnorm(100))
```



Use your function to generate a small multiples plot with panels for the parameters `R0`, `days_presymptomatic`, `days_infectious`, and `serial_interval_sym`.

You will still need to filter the data to include only this subset of measures. This could be done inside the function, or or you coudl prefilter the data before calling the function.

You will also need to determine an optimal way to sort the data within each plot. You can use the `arrange` function from the `tidyverse` package for this. For example:

```
plotdata <- plotdata %>%
  arrange(estimate) # sort by estimate
```

Exercise 1b.

1. Add a summary measure to each sub plot. Consider what an appropriate summary measure would be. Should it simply be the mean of the estimates?

2. Imagine you had a measure of uncertainty for the summary measure. How would you represent the uncertainty? The `forestplot` function from the `rmeta` package has a solution for this. Is it optimal?

3. Imagine you had a measure of power of each estimate. How would you represent that? The `forestplot` function from the `rmeta` package has a solution for this, too. Is it optimal?

Exercise 1c.

The dataset includes a column for virus type. If you have time, encode the additional variable `virusType`. For example, you could change the shape of the estimate point (using the argument `pch` of the `points` function), or color the lines and points using the `col` argument of the `points` and `lines` functions.

You can compare your solutions with the ones developed by the Coronavirus Working Group using the `rplotly` package: https://www.covid19.uga.edu/context

See https://bookdown.org/ndphillips/YaRrr/arranging-plots-with-parmfrow-and-layout.html for more ways to build small multiples plots in R.