Imperial College London – Zambart

Workshop on *Analysing and modelling epidemic data*

Practical 2 Part 1: Deriving model parameters from data.

Ruth McCabe and Dr Pablo N Perez-Guzman

Background

When a new infectious disease emerges, policymakers are faced with a range of unknowns that they have to respond to quickly. In the early stages of an outbreak, mathematical modelling is a particularly useful tool to answer questions such as: whether hospitals will have the capacity to treat all patients requiring care; how many people will die from the disease; how long it will take for infections to peak. We can use mathematical models to assist with these questions, but we need to be able to parameterise them using relevant data.

We consider a situation in which there is a new disease and patients are being admitted to hospital. You are tasked with estimating the average time it takes for an infected person to either recover or die.

We will be doing our analysis in R studio. A script is provided with all code also provided in boxes at the relevant question in this handout.

Objectives

- Estimate the mean time to recovery and death of patients.
- Understand the impact on estimates if using truncated data.
- Correct estimates derived from truncated data.

Getting started

Navigate to the odin interface https://shiny.dide.ic.ac.uk/infectiousdiseasemodels-lusaka-2022/ in Chrome or Safari.

In the Tuesday section, “Deriving model parameters from data”, click on “Download script.R”.

Example 1: Estimating the mean time to recovery

A doctor from the local hospital has recorded the number of days that it has taken the first 100 patients to recover from the disease:

1, 12, 6, 7, 10, 9, 21, 9, 8, 7, 3, 9, 2, 8, 25, 12, 30, 3, 3, 6,
9, 8, 4, 13, 7, 6, 4, 13, 37, 6, 4, 78, 6, 12, 10, 5, 21, 7, 5, 15,
7, 4, 23, 13, 7, 19, 8, 2, 5, 4, 1, 22, 3, 22, 3, 59, 3, 11, 20, 8,
4, 5, 16, 2, 23, 4, 2, 17, 3, 3, 16, 5, 2, 10, 4, 9, 2, 5, 9, 1,
2, 7, 12, 8, 8, 15, 8, 8, 5, 4, 7, 4, 4, 10, 16, 12, 4, 11, 11, 10

We want to know, on average, how long it takes patients to recover.

1. Read the data into R using the following code:

```
recovery_data <- c(1, 12, 6, 7, 10, 9, 21, 9, 8, 7, 3, 9, 2, 8, 25, 12, 30, 3, 3, 6, 9,
        8, 4, 13, 7, 6, 4, 13, 37, 6, 4, 78, 6, 12, 10, 5, 21, 7, 5, 15, 7, 4,
        23, 13, 7, 19, 8, 2, 5, 4, 1, 22, 3, 22, 3, 59, 3, 11, 20, 8, 4, 5, 16,
        2, 23, 4, 2, 17, 3, 3, 16, 5, 2, 10, 4, 9, 2, 5, 9, 1, 2, 7, 12, 8, 8,
        15, 8, 8, 5, 4, 7, 4, 4, 10, 16, 12, 4, 11, 11, 10)
```

2. Use a histogram to look at the distribution of the data. What continuous probability distribution best describes them? (Eg. Normal, Exponential, Gamma, …)

```
hist(recovery_data, freq = FALSE,
  breaks = 20, bty = "n", xlab = "Time to recovery (days)", main="")
```

3. What is the mean recovery time?

```
mean(recovery_data)
```

4. Consider how the mean recovery time changes as the sample size increases. Do you notice any convergence?

```
n_sample <- 1: length(recovery_data)

estimator_over_n <- cumsum(recovery_data) / n_sample

plot(estimator_over_n,

    xlab = "Sample size",

    ylab = "Estimated mean time to recovery",

    bty = "n")

abline(h = mean(recovery_data), lty = 2)
```

Example 2: Estimating the mean times to recovery or death

In Example 1, all of the initial patients recovered. Unfortunately, this is extremely unlikely to be the case in reality when faced with an emerging infectious disease. There are competing risks of whether an individual will recover or die.

We are now going to flash forward to the end of the first wave of this outbreak. Colleagues in a neighbouring state have analysed all of the patients admitted to their hospitals in order to estimate the mean time to recovery or to death (the outcome). We don't get to see their data, but we can see that they modelled them using an exponential distribution.

Let us consider an example for the time to recovery, noting that the procedure is identical for the time to death.

Let $x_1, x_2, x_3, \ldots, x_n$ denote the n observed times for each patient to recover.

Recalling that if a random variable X has an exponential distribution, then $f(X;\lambda) = \lambda \exp(-\lambda x)$is the probability density function.

The likelihood of the observed data is:

$$L = \prod_{i=1}^{n} f(x_i;\lambda) = \prod_{i=1}^{n} \lambda \exp(-\lambda x_i) = \lambda^n \exp\left(-\lambda \sum_{i=1}^{n} x_i\right)$$

The log-likelihood is:

$$l = n\log(\lambda) - \lambda \sum_{i=1}^{n} x_i$$

To find the Maximum Likelihood Estimator (MLE), we differentiate the log-likelihood, set this equal to zero and solve for λ:

$$\frac{\partial l}{\partial \lambda} = \frac{n}{\lambda} - \sum_{i=1}^{n} x_i$$

$$\frac{\partial l}{\partial \lambda} = 0 \rightarrow \widehat{\lambda} = \frac{n}{\sum_{i=1}^{n} x_i}$$

By following this procedure, the researchers estimated that:

$$\widehat{\lambda_r} = \frac{1}{14}$$

$$\widehat{\lambda_d} = \frac{1}{7}$$

This means that, on average, patients who recover do so in an average of 14 days, and patients who die do so in average of 7 days.

We can use this information to simulate plausible times to recovery and death and then consider the differences in these two distributions.

1. Sample 1000 observations from each of these distributions. (Note: the set.seed() function ensures that we all get the same sample each time. Without this, there would be slight variability in the results.)

```
set.seed(2904)

data <- data.frame(

  recovery = round(rexp(1000, 1 / 14)),

  death = round(rexp(1000, 1 / 7)))
```

2. Use the head() function in R to look at the first 20 simulated times to recovery and death.

```
head(data, 20)
```

3. Using the formulae above, what are the MLEs of your two samples (time to recovery and time to death)?

```
length(data$recovery)/sum(data$recovery)

length(data$death)/sum(data$death)
```

4. What does this translate to in terms of the mean number of days until the outcome? Verify your calculation by using the mean() function.

```
1/(length(data$recovery)/sum(data$recovery))

1/(length(data$death)/sum(data$death))

mean(data$recovery)

mean(data$death)
```

5. Consider how the mean time to recovery and mean time to death change as the sample size increases. Do you notice any convergence?

```
n_sample <- 1:nrow(data)

estimator_recovery <- cumsum(data$recovery) / n_sample

estimator_death <- cumsum(data$death) / n_sample

plot(estimator_recovery,

    xlab = "Sample size",

    ylab = "Estimated mean time to outcome (days)",

    bty = "n",

    col = "blue",

    ylim = c(0, max(estimator_recovery)))

points(estimator_death,

     col = "red")

abline(h = mean(data$recovery), col = "blue", lty = 2)

abline(h = mean(data$death), col = "red", lty = 2)
```

Example 3: Analysing truncated data

During an outbreak, data involving the time to an event are often subject to survival bias. For example, in the first two weeks of an outbreak, you only know what has happened in those two weeks and not what will happen to patients in the future. It is possible that a patient alive on day 14 could die on day 15.

Figure: Example of data that could be missed when conducting an analysis early on in an outbreak. If we analyse data before the wave is over, we lose the three data points in the grey shaded box. Estimating parameters on the remaining subset of data biased our estimates of the time to recovery and death. Crosses indicate death and ticks indicate recovery.

Now imagine that your colleagues from Example 2 want to estimate the time to each outcome at the second week of the outbreak. We only know the outcomes of patients up to this time, and so we will remove everyone else from our analysis in this question. We want to investigate how this would change our estimates of the time to outcome.

1. Truncate the data generated in Example 2 to observations less than 14 days.

```
truncated_recov <- data$recovery[which(data$recovery <= 14)]

truncated_death <- data$death[which(data$death <= 14)]
```

2. How many observations are there, what is the MLE and what is the mean of:
    a. Time to recovery?

```
length(truncated_recov)
length(truncated_recov)/sum(truncated_recov)
mean(truncated_recov)
```

    b. Time to death?

```
length(truncated_death)
length(truncated_death)/sum(truncated_death)
mean(truncated_death)
```

3. Consider how the mean time to recovery and mean time to death change as the sample size increases. Do you notice any convergence?

```
n_recov <- 1:length(truncated_recov)
n_death <- 1:length(truncated_death)
biased_estimator_recov <- cumsum(truncated_recov) / n_recov
biased_estimator_death <- cumsum(truncated_death) / n_death
plot(biased_estimator_recov,
    xlab = "Sample size",
    ylab = "Estimated mean time to outcome (days)",
    bty = "n",
    col = "blue",
    ylim = c(0, max(estimator_recovery)))
points(biased_estimator_death,
     col = "red")
abline(h = mean(data$recovery), col = "blue", lty = 2)
abline(h = mean(data$death), col = "red", lty = 2)
```

Example 4: Adjusting for truncated data

We have seen that analysing data early on in the outbreak will be subject to bias, in that only events which happen up until that time are included in the analysis. As demonstrated above, this could substantially lower our estimates of the outcome time and thus give us a false picture of the dynamics of the disease.

We can adjust our estimation procedure to account for the fact that our data are truncated.

If a random variable X_T is distributed according to a truncated exponential distribution, then $f(X_T;\ \lambda\ ) = \frac{\lambda \exp(-\lambda x_T)}{1-\exp(\lambda\beta)}$

where β is the truncation time.

We can follow the steps as above to calculate the MLE, but this involves more complicated mathematics. We shall instead deploy a built-in function in R, uniroot(), to estimate the MLE under this different likelihood function.

1. Run the function titled truncation_adjusted_MLE() using the code below.

```
truncation_adjusted_MLE <- function(data, trunc_cut_off = 14){
  dL <- function(lambda, n = length(data), sum_obs = sum(data)){
  n / lambda - sum_obs - ((n*trunc_cut_off*exp(-lambda*trunc_cut_off))/(1 - exp(-
lambda*trunc_cut_off)))
}
  f_zero <- function(lambda){
    dL(lambda, n = length(data),sum_obs = sum(data))
  }
   ML_sol <- uniroot(f_zero, interval = c(1e-6, 1e6))
    return(list("MLE" = ML_sol$root,
           "average_outcome" = 1/ML_sol$root))
}
```

2. Use the function to work out the MLE and average outcome time for recovery and death, and compare this to what we observed in Example 2.
    a. Time to recovery

```
truncation_adjusted_MLE(data = truncated_recov)
```

b. Time to death

```
truncation_adjusted_MLE(data = truncated_death)
```

Extension: Biased estimates under different truncation times

This is an optional extra for this practical for those who have time.

Using what we have learned so far in this practical, can you analyse the effect of different truncation times on the estimates of the mean outcome times?

```
truncation_adjusted_MLE_extension <- function(df = data, trunc_cut_off){

  # truncate original data

  data_recov_trunc <- df$recovery[which(df$recovery <= trunc_cut_off)]

  data_death_trunc <- df$death[which(df$death <= trunc_cut_off)]

  ## naive MLE and outcome time

  ##recovery

  MLE_naive_recov <- length(data_recov_trunc)/sum(data_recov_trunc)

  average_outcome_naive_recov <- 1/MLE_naive_recov

  #death

  MLE_naive_death <- length(data_death_trunc)/sum(data_death_trunc)

  average_outcome_naive_death <- 1/MLE_naive_death

  ### adjusted for truncation

  ## recovery

  dL_recov <- function(lambda, n = length(data_recov_trunc), sum_obs =
sum(data_recov_trunc)){

    n / lambda - sum_obs - ((n*trunc_cut_off*exp(-lambda*trunc_cut_off))/(1 - exp(-
lambda*trunc_cut_off)))

  }
```

```
  f_zero_recov <- function(lambda){
    dL_recov(lambda, n = length(data_recov_trunc),sum_obs = sum(data_recov_trunc))
  }
  ML_sol_recov <- uniroot(f_zero_recov, interval = c(1e-6, 1e6))
  ## death
  dL_death <- function(lambda, n = length(data_death_trunc), sum_obs =
sum(data_death_trunc)){
    n / lambda - sum_obs - ((n*trunc_cut_off*exp(-lambda*trunc_cut_off))/(1 - exp(-
lambda*trunc_cut_off)))
  }
   f_zero_death <- function(lambda){
    dL_death(lambda, n = length(data_death_trunc),sum_obs = sum(data_death_trunc))
  }
  ML_sol_death <- uniroot(f_zero_death, interval = c(1e-6, 1e6))
  return(c("trunc_cut_off" = trunc_cut_off,
    "MLE_naive_recov" = MLE_naive_recov,
          "average_outcome_naive_recov" = average_outcome_naive_recov,
          "MLE_naive_death" = MLE_naive_death,
          "average_outcome_naive_death" = average_outcome_naive_death,
          "MLE_adjust_recov" = ML_sol_recov$root,
          "average_outcome_adjust_recov" = 1/ML_sol_recov$root,
          "MLE_adjust_death" = ML_sol_death$root,
          "average_outcome_adjust_death" = 1/ML_sol_death$root))
}
```

```
trunc_times <- seq(5,21,1)
sens_analysis <- c()
for(i in trunc_times){
  output <- truncation_adjusted_MLE_extension(df = data, trunc_cut_off = i)
  sens_analysis <- rbind(sens_analysis,output)
}
sens_analysis <- data.frame(sens_analysis)
```

```
ylims <- c(0, 40)
plot(sens_analysis$trunc_cut_off, sens_analysis$average_outcome_naive_recov,
    col = "blue", bty = "n", type = "l", ylim = ylims,
    ylab = "Average time to recovery (days)", xlab = "Truncation time (days)")
lines(sens_analysis$trunc_cut_off, sens_analysis$average_outcome_adjust_recov,
     col = "red")
legend("topright", legend = c("Naive", "Adjusted"), col = c("blue", "red"),
      lty = 1, bty = "n")
```

```
ylims <- c(0, max(sens_analysis$average_outcome_adjust_death))
plot(sens_analysis$trunc_cut_off, sens_analysis$average_outcome_naive_death,
    col = "blue", bty = "n", type = "l", ylim = ylims,
    ylab = "Average time to death (days)", xlab = "Truncation time (days)")
lines(sens_analysis$trunc_cut_off, sens_analysis$average_outcome_adjust_death,
     col = "red")
legend("topright", legend = c("Naive", "Adjusted"), col = c("blue", "red"),
      lty = 1, bty = "n")
```