The 2017 plague outbreak in Madagascar & the Madagascar Plague Model

true true

June 15th, 2022

Abstract

During this experiment the plague that occurred in Madagascar in 2017 was used as a means to understand plague transmission as real-time data was collected from official reports, described the outbreak's characteristics and estimated transmission parameters using statistical and mathematical models.

Contents

Introduction

The goal of this experiment was to learn more about the behaviour and the way the plague spreads and then modelling our findings about the plague. Due to the fact that Madagascar has annual plague outbreaks, it was decided to use their outbreak information/data as data for this experiment.

The plague has once been one of the deadliest natural disasters in human history, referred to as the Black Death, attributed to the gram-negative bacterium 'Yersinia pestis'. (Boire et al., 2014), it's life-cycle typically starts within an insect vector, in our case: fleas. After which transmission occurs to a mammalian host, typically rodents or other wild animals. Humans are then only affected as incidental hosts. Although the disease was widespread in ancient times, nowadays, plague epidemics still pose a threat to humans, as there are reports of continues annual infections in the following 5 countries: Madagascar, Tanzania, Vietnam, China & the USA. (Boire et al., 2014; Dennis et al., 1999) The disease can manifest in different clinical forms of plague: bubonic, pneumonic and septic, as a result of being bitten by infected fleas. Additionally, direct contamination with infective materials can be an alternative transmission route (Boire et al., 2014). Patients who contract bubonic plague develop a sudden onset of fever, headache, chills, tender and painful lymph nodes (Robert L. Dillard; Andrew L. Juergens, 2022). While plague can be successfully treated with

antibiotics, if left untreated, the bacteria can disseminate from the lymph nodes into the bloodstream causing secondary septicemic plague. Which in addition to the symptoms already present in bubonic plague, patients with septicemic plague undergo abdominal pain and most likely bleeding into the skin and other organs, next to that skin and other tissues may undergo necrosis, especially the fingers, toes and the nose (Centers for Disease Control and Prevention, 2005). However, the most deadly form of the disease is driven by the pneumonic plague and is the only form of plague that can be spread from person to person by infectious droplets caused by sneezing, coughing or being together in an enclosed space. This form of disease has an average incubation time of 4 days and progresses rapidly and is almost always fatal without sufficient antibiotic treatment (Mead, 2017).

One of the goals set for this experiment is to recreate the model and results that the researchers gained from their experiments. To remake the model, the formula's used in the document have been used and adapted to R code, so they can be used with the Desolve package in R. To help with this, the researchers published their code on github, although their code was written in python and had to be adapted to R. Due to that, we also used the same parameters & data the researchers used.

Methods

Outbreak data The outbreak data was manually inputted from separate reports of WHO (World Health Organization), including the cumulative total numbers of clinical cases.

Temperature & Precipation data Data used is the same as the researchers requested from the National Centers for Environmental Information.

Plague transmission model (PTM) The plague transmission model used is a modified SEIR (Susceptible-Exposed-Infectious-Removed) model, it contains 2 additional components reflecting the seasonal from infected rat fleas and the imperfect effects of public health interventions.

Formulas

The following formulas were used to model the PTM.

$$\frac{dS_b}{dt} = -S_b * \alpha f_{irf} - \beta S_b \frac{I_p}{N} f_{itv,h}$$

$$\frac{dS_p}{dt} = -\beta S_p \frac{I_p}{N} f_{itv,h}$$

$$\frac{dE_b}{dt} = S_b \alpha f_{irf} f_{irv,f} - \gamma_b E_b$$

$$\frac{dE_p}{dt} = \beta (S_b + S_p) \frac{I_p}{N} f_{itv,h} - \gamma_p E_p$$

$$\frac{dI_b}{dt} = \gamma_b E_b - \epsilon I_b - \delta_b I_b$$

$$\frac{dI_p}{dt} = \gamma_p E_p - \epsilon I_b - \delta_p I_p$$

Herein the S, E, I describe Susceptible, Exposed, and Infectious and the subscripts b & p denote the bubonic and pneumonic form. The model assumes that there is a fixed population of size N, where only a small part $S_b = pN$ is exposed to infected rat fleas. The transmission rates of the flea-to-human and human-to-human are denoted by α and β. Next to that, the flea density fluctuates due to the season's temperature and

the breeding pattern of the rats. Due to that, the density of infected rat fleas is described as a sinusoidal function $f_{irf} = A + b\sin(2\pi/12t) + C\cos(2\pi/12t)$ that follows the temperature fluctuations. Due to this, the flea-to-human transmission parameter α incorporates a scaling factor from temperature to rat fleas.

Due to there going to be interventions that would reduce flea-to-human and human-to-human transmission rates via rodent and flea control. However, these effects have been considered imperfect and has a logistical form as $f_{itv,h,f} = 2 - 2/[1 + \theta + \exp(\tau_h, f - t)]$, where $\tau > 0$ denotes the time at which the interventions reached half maximum effect in controlling human-to-human ($/tau_h$) and flea-to-human (τ_f) transmission. The reduction from a perfect intervention is defined by $\theta \geq 0$, where $\theta = 0$ means a perfect scenario. The infected cases are assumed to recover and die with the total rate of removal from the infected pool being δ_b and δ_p for bubonic and pneumonic cases.

Libraries • pander -This library is used to print out data/R-code in an alternative way, usually easier to read.

- deSolve -This library contains the ODE that was used to recreate the model in the R language.

Results

```
# get the working directory
working_directory <- getwd()
# get the directory for the data_file
data_file_bugfix <- paste(working_directory, "/Project_madagascar/Data/", sep = "")
data <- read.csv("C://Users//matsp//Documents//Thema08//Thema_08_intro_to_system_bio//Project_madagasca
# options(digits = 15)

# Run a test to see if the data imported correctly
pander(head(data))
```

Sb	Sy	Eb	Ey	Ib	Ip	tijd
1651	25569244	0.3886	0.217	0.9754	0.9726	0.005
1650	25569244	0.7683	0.4225	0.9602	0.9521	0.02
1650	25569243	1.139	0.6185	0.9539	0.9378	0.045
1649	25569243	1.502	0.8064	0.9561	0.9295	0.08
1649	25569243	1.856	0.9878	0.9663	0.9266	0.125
1649	25569243	2.203	1.164	0.9842	0.9289	0.18

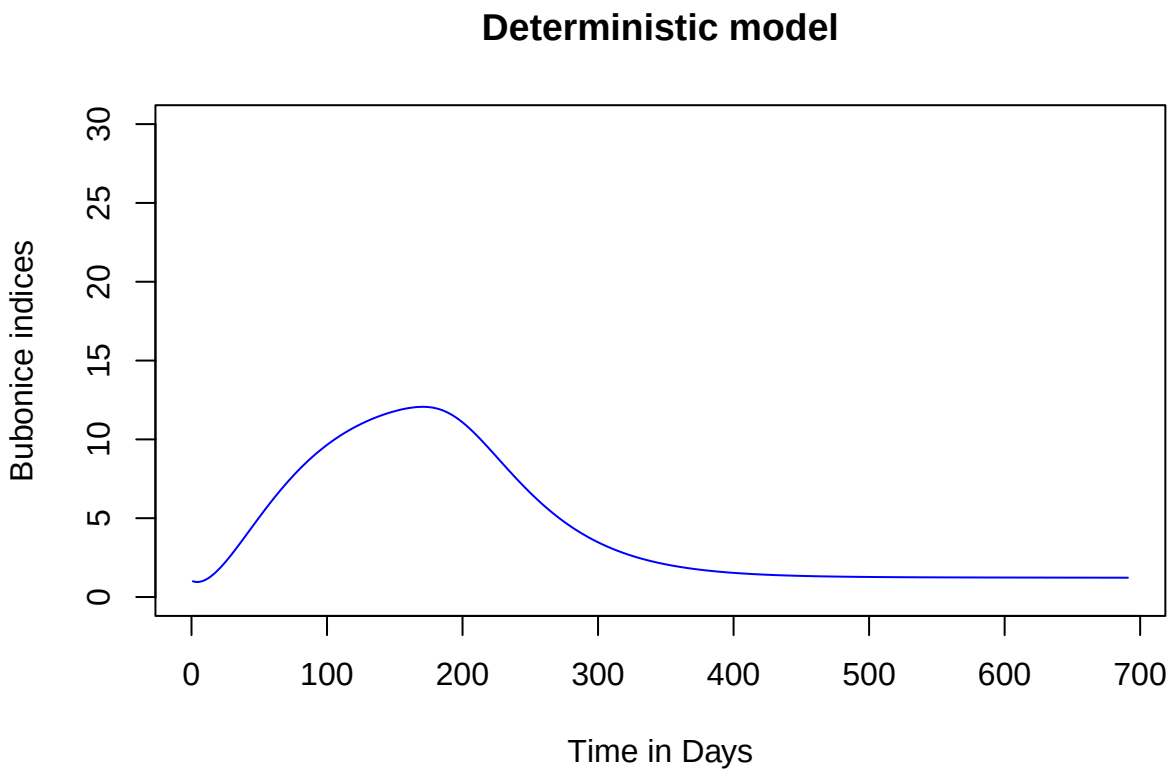
In the table above, the imported data is visible.

```
# TODO fixe logical bug

# Define the variables, parameters, state & time.
tau_p <- 8.893084e+00
tau_b <- 1.793090e+01

A <- 1.15
B <- 0.08
C <- 0.1


parameters <- c(p       = 6.459494e-05, # fraction susceptible to bite
```

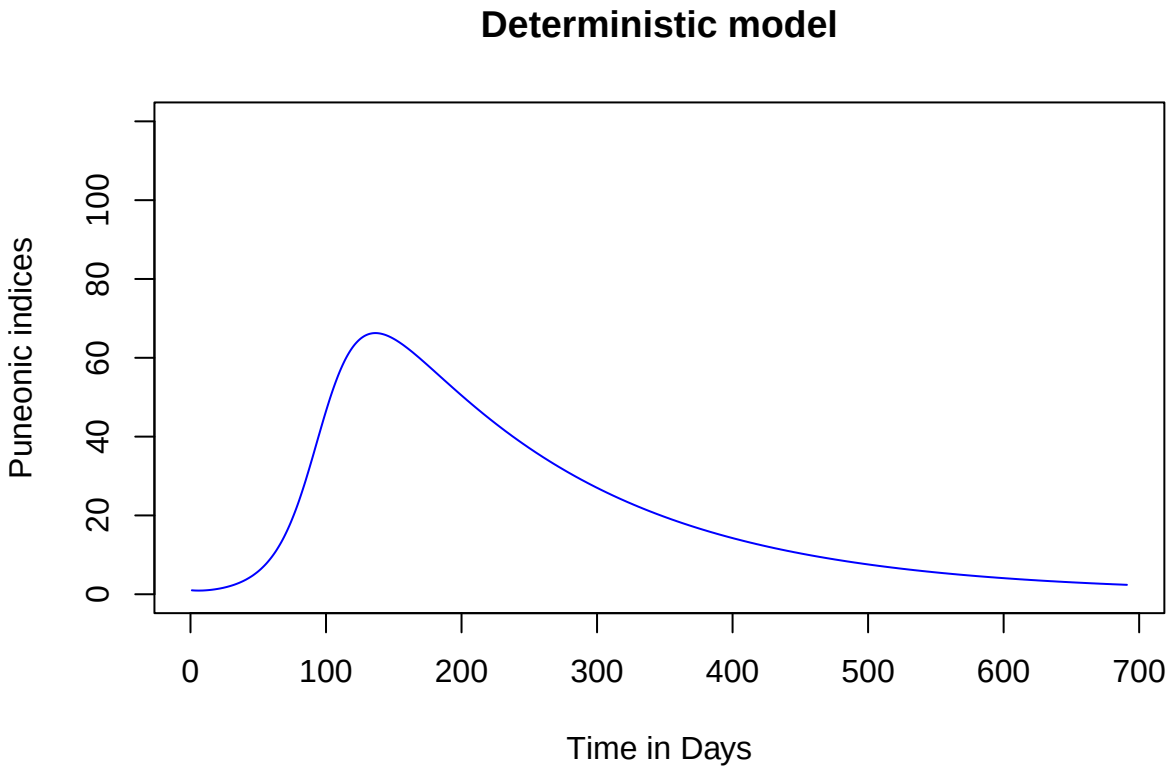
```
                alpha   = 1.904753e-03, # rate of bites
                beta    = 2.233049e+00, # between-host infection
                gamma_b = 2.293953e-01, # E to I
                gamma_p = 2.863875e-01, # E to I
                delta_b = 2.626527e-01, # recovery + death
                delta_p = 3.413748e-01, # recovery + death
                epsilon = 3.204494e-02, # I_b to I_p
                kk      = 1.105443e+00, # Imperfect intervention
                N       = 25570895     # population size
                )

S_b_0 <- 1651
S_p_0 <- 25569244

state <- c(Sb = S_b_0, Sy = S_p_0, Eb = 0, Ey =  0, Ib = 1, Ip = 1)

times <- seq(0, 70 - 1, by = 0.1)

# Define the flea function for the flea population
flea <- function (t){
  x <- A + B * sin((pi/180.) * t) + C * cos((pi/180.) * t )

  return (x)
}
# Define the intervention functions of human intervention on the disease
intervention_p <- function (t){
  return (1 - 1/(as.numeric(parameters["kk"]) + exp(tau_p - t)))
}
intervention_b <- function (t){
  return (1 - 1/(as.numeric(parameters["kk"]) + exp(tau_b - t)))
}
```

```
# Define the function for the model of the deterministic data runs
deterministic_model <- function(t, y, parms){
  with(as.list(c(parms, y, t)),{
    flea_t <- flea(t)
    intervention_b1 <- intervention_b(t)
    intervention_p1 <- intervention_p(t)

    delta.Sb <- -alpha * flea_t * intervention_b1 * Sb - -beta * intervention_p1 * Sb * Ip/N
    delta.Sy <- -beta * intervention_p1 * Sy * Ip/N
    delta.Eb <- alpha * flea_t * intervention_b1 * Sb - gamma_b * Eb
    delta.Ey <- beta * intervention_p1 * (Sb+Sy) * Ip/N - gamma_p * Ey
    delta.Ib <- gamma_b * Eb - epsilon * Ib - delta_b * Ib
    delta.Ip <- gamma_p * Ey + epsilon * Ib - delta_p * Ip

        return(list(c(delta.Sb, delta.Sy, delta.Eb, delta.Ey, delta.Ib, delta.Ip)))
       })
}

# Run the ode model and save the output to the out variable
out <- ode(times = times, y = state,   parms = parameters, func = deterministic_model, method = "lsoda"
```

```
plot(as.data.frame(out)$Ib, type = "l", xlab = "Time in Days", ylab = "Bubonice indices", col = "blue",
```

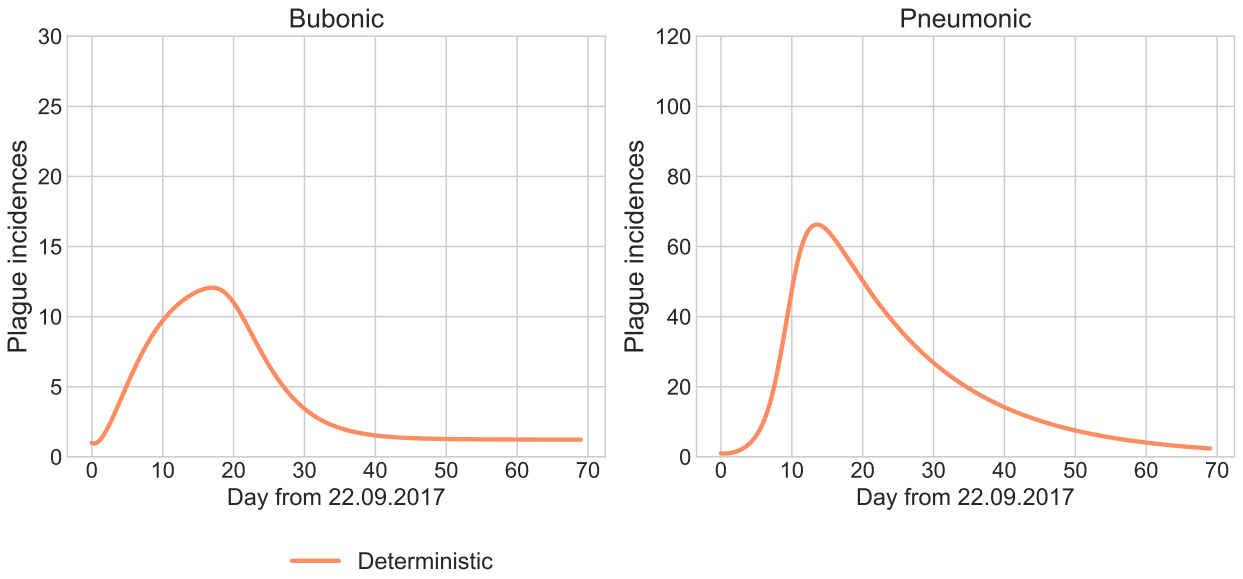



```
plot(as.data.frame(out)$Ip, type = "l", xlab = "Time in Days", ylab = "Puneonic indices", col = "blue",
```



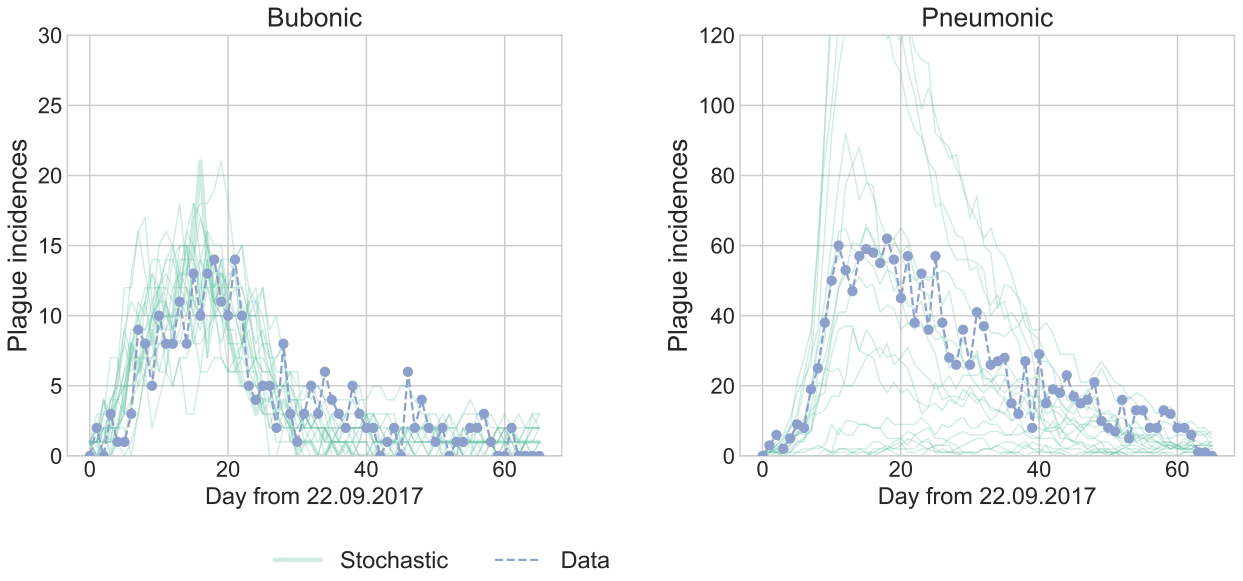

```
# python code figures
knitr::include_graphics("C://Users//matsp//Documents//Thema08//Thema_08_intro_to_system_bio//Project_ma
```

```
## Warning in knitr::include_graphics("C://Users//matsp//Documents//Thema08//
## Thema_08_intro_to_system_bio//Project_madagascar//Fig1.pdf"): It is highly
## recommended to use relative paths for images. You had absolute paths:
## "C://Users//matsp//Documents//Thema08//Thema_08_intro_to_system_bio//
## Project_madagascar//Fig1.pdf"
```




```
knitr::include_graphics("C://Users//matsp//Documents//Thema08//Thema_08_intro_to_system_bio//Project_ma
```

```
## Warning in knitr::include_graphics("C://Users//matsp//Documents//Thema08//
## Thema_08_intro_to_system_bio//Project_madagascar//Fig2.pdf"): It is highly
## recommended to use relative paths for images. You had absolute paths:
## "C://Users//matsp//Documents//Thema08//Thema_08_intro_to_system_bio//
## Project_madagascar//Fig2.pdf"
```




```
# to test sample diffreance with a simple T test
print("bubonic comparison")
```

```
## [1] "bubonic comparison"
```

```
(test_ib <- t.test(as.data.frame(out)$Ib,data$Ib))
```

```
##
##  Welch Two Sample t-test
##
## data:  as.data.frame(out)$Ib and data$Ib
## t = -0.022567, df = 1379, p-value = 0.982
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.4111354  0.4017839
## sample estimates:
## mean of x mean of y
##  4.226284  4.230960
```

```
cat("the P value", test_ib$p.value, "> alpha 0,05")
```

```
## the P value 0.9819993 > alpha 0,05
```

```
print("Puneonic comparison")
```

```
## [1] "Puneonic comparison"
```

```
(test_ip <- t.test(as.data.frame(out)$Ip,data$Ip))
```

```
##
##  Welch Two Sample t-test
##
## data:  as.data.frame(out)$Ip and data$Ip
## t = -0.027568, df = 1379, p-value = 0.978
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.140998  2.081656
## sample estimates:
## mean of x mean of y
##  21.47405  21.50372
```

```
cat("the P value", test_ip$p.value, "> alpha 0,05")
```

```
## the P value 0.9780105 > alpha 0,05
```

In the graph(s), we can see the plot of the deterministic data model.

```
state_sto <- c(state[['Sb']],state[['Sy']],state[['Eb']],state[['Ey']],state[['Ib']],state[['Ip']])
D <- 6
# Vectorize the flea & intervention functions
# v_flea <- vectorize(flea)
# v_itv_p <- vectorize(intervention_p)
# v_itv_b <- vectorize(intervention_b)
```

```r
# Save all the flea/intervention data based on each piece of time
f_irf <- flea(times)
f_itv_b <- intervention_b(times)
f_itv_p <- intervention_p(times)

unit_vector <- function (i, D){
  out <- rep(0, D)
  out[i+1] <- 1
  return (out)
}



transition_rates <- function (state, f_irf, f_itv_p, f_itv_b) {

  return (array(c(parameters[["alpha"]] * f_irf * f_itv_b * state[["Sb"]],
                  parameters[["beta"]] * f_itv_p * state[["Sb"]] * state[["Ip"]]/parameters[["N"]],
                  parameters[["beta"]] * f_itv_p * state[["Sy"]] * state[["Ip"]]/parameters[["N"]],
                  parameters[["gamma_b"]] * state[["Eb"]],
                  parameters[["gamma_p"]] * state[["Ey"]],
                  parameters[["delta_b"]] * state[["Ib"]],
                  parameters[["delta_p"]] * state[["Ip"]],
                  parameters[["epsilon"]] * state[["Ib"]])))

}

nu <- array(c(unit_vector(2, D) - unit_vector(0, D), unit_vector(3, D) - unit_vector(0, D),
              unit_vector(3, D) - unit_vector(1, D), unit_vector(4, D) - unit_vector(2, D),
              unit_vector(5, D) - unit_vector(3, D), -unit_vector(4, D), -unit_vector(5, D),
              unit_vector(5, D) - unit_vector(4, D)), dim = c(6,8))

change_func <- function(n, firings, nu) {
  prechange <- rep(0,6)

  for (i in seq(1,n)) {
    #print(i)
    prechange <- prechange + (firings[i]*nu[,i])

  }
  return(prechange)
}

plague_tau <- function (parms, t, f_irf, f_itv_p, f_itv_b, nu) {
  x_t <- data.frame(matrix(ncol = 6, nrow = 0))
  colnames(x_t)<- c("Sb", 'Sy', 'Eb', 'Ey', 'Ib', "Ip")
  for (i in seq(0,length(t))){
    x_t[nrow(x_t) + 1,] <- state_sto
    rates <- transition_rates(state, f_irf[i], f_itv_p[i], f_itv_b[i])
    n <- length(rates)
    firings <- rpois(n, max(rates, rep(0, n)*0.1))
    change <- change_func(n, firings, nu)

    state_sto <- state_sto + change
```

```
  }
  x_t <- (x_t)
  return(x_t)
}

# Set the seed for the random number generator
set.seed(10042018)
runs <- 20

I_blist <- list()
I_Plist <- list()

runs <- 1
repeat {
  y <- plague_tau(parms, times, f_irf, f_itv_p, f_itv_b, nu)


  I_b <- (y$Ib)
  I_P <- (y$Ip)

  I_blist <- append(I_blist, list(I_b))
  I_Plist <- append(I_Plist, list(I_P))


  if(runs == 20) {
      break
   }
  runs <- runs +1
}
```

Conclusion

Looking at the deterministic models from both Python and R, it shows that the graphs are almost similar. Although, due to there being differences in how both coding-styles work under the hood. The graphs are different as is shown in the comparison between the deterministic models of R & python.

Discussion

While translating the python model to R. We found that there are some minor internal rounding differences when calculating floats in R and python, these differences where isolated and removed, simply by comparing each outcome of the r and python equation and the outcome of each equation used for calculating the delta value was exactly the same. Furthermore, we rewrote the R code back to python code to see if there were any differences or mistakes in our code, compared to the document's code. In the end we only managed to find one mistake between the code. Although even after fixing the error, the results in R are different from python, even between our own code.

This let to that we found that DeSolve ode function and the scipy odeint have minor differences in calculating the deltas, that with a given time create a snowball effect. This results in the R model having a higher peak curve compared to the Python 3 model.

Also, while converting the stochastic model from python to R. It was apparent that we had to fully rewrite the code that they have used as it wasn't easily use-able in R itself. This has lead to some complications in writing the code in an R-styled manner.

Sources

Main article: https://www.sciencedirect.com/science/article/pii/S1755436518300070

Boire et al., 2014; https://www.longdom.org/open-access/lessons-learned-from-historic-plague-epidemics-the-relevance-of-an-ancient-disease-in-modern-times-2329-8731.1000114.pdf Dennis et al., 1999; https://apps.who.int/iris/bitstream/handle/10665/66010/WHO_CDS_CSR_EDC_99.2.pdf?sequence=1&isAllowed=y Robert L. Dillard; Andrew L. Juergens, 2022; https://www.ncbi.nlm.nih.gov/books/NBK549855/ Centers for Disease Control and Prevention, 2005; https://www.cdc.gov/plague/symptoms/index.html Mead, 2017; https://www.nejm.org/doi/10.1056/NEJMp1713881