Mathematical & Computational Modeling of Infectious Diseases **D.B. Larremore**
Homework 3 **2025**

Instructions:

- Please turn in a single PDF file.
- Please share a link to your code, but do not attach the actual code.
- Handwritten math (scanned and included in a PDF) is fine, but please watch out for 10MB+ file sizes!

Ben Aoki-Sherwood

code: `https://github.com/aoki-sherwoodb/CSCI-5897-InfectiousDiseases`

1. The goal of this problem is to explore the preferential depletion of susceptibles.

   Consider a population with four equal-sized groups, numbered $1, 2, 3, 4$. Suppose that the contact structure in the population is fully mixed (i.e. $c_{ij} = \bar{c}$ for all i, j), that $\gamma_i = 3$ for all i, and that $R_0 = 1.5$, under SIR dynamics. Finally, suppose that the susceptibility for group 1 is $p_1 = 1$, the susceptibility for group 2 is $p_2 = 2$, with $p_3 = 3$ and $p_4 = 4$.

   a. In terms of $\bar{c}$, s_1, s_2, s_3, and s_4 (and constants), what is the next-generation matrix for this system?

$$
\begin{aligned}
G &= (D_p D_s C D_\omega^{-1}) D_\gamma^{-1} \\
&= \frac{\bar{c}\gamma\omega}{D}{}_{p} D_s \\
&= \frac{4\bar{c}}{3}\begin{bmatrix} s_1 p_1 & 0 & \dots & \\ 0 & s_2 p_2 & \dots & \\ 0 & 0 & s_3 p_3 & \dots \\ 0 & 0 & 0 & s_4 p_4 \end{bmatrix} \\
&= \frac{4\bar{c}}{3}\begin{bmatrix} s_1 & 0 & 0 & 0 \\ 0 & 2s_2 & 0 & 0 \\ 0 & 0 & 3s_3 & 0 \\ 0 & 0 & 0 & 4s_4 \end{bmatrix}
\end{aligned}
$$

   b. To ensure that $R_0 = 1.5$, what must $\bar{c}$ be equal to?

   The next-generation matrix when the entire population is susceptible (i.e., $s_i = 1$ for all i)

is

$$G_0 = \frac{\bar{c}}{3}\begin{bmatrix} p_1 & 0 & 0 & 0 \\ 0 & p_2 & 0 & 0 \\ 0 & 0 & p_3 & 0 \\ 0 & 0 & 0 & p_4 \end{bmatrix}$$

$$= \frac{\bar{c}}{3}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

The largest eigenvalue of a diagonal matrix is the largest entry on the diagonal, so we must have that

$$1.5 = \frac{4\bar{c}}{3}$$

$$\bar{c} = \frac{9}{8} = 1.125$$

c. Using these parameters, code up a version of your model with initial conditions where 99.9% of people in each group are susceptible, and the other 0.1% are infected.[1] Simulate an epidemic wave using an appropriate timestep Δt and appropriate maximum simulation time to capture the wave. Create a plot of the four populations' I compartments vs time, showing $i_1(t)$, $i_2(t)$, $i_3(t)$, and $i_4(t)$. Color these curves in a single hue, but with varying levels of light/dark or saturation, such that the boldest and darkest line is the most susceptible group, and the faintest and lightest line is the least susceptible group.

[1] You'll have to modify your code to cope with the fact that there are now 4 groups!

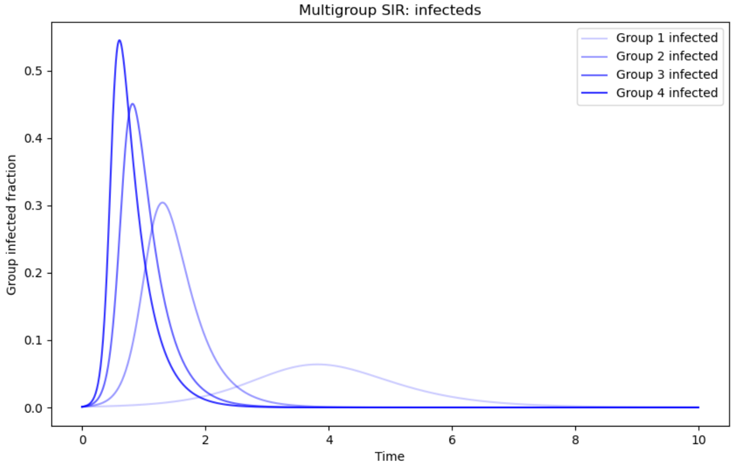

Figure 1: **Infection dynamics for four groups with different susceptibilities. Population subgroups are shaded according to their susceptibilities, with group 4 (susceptibility 4) shown in the darkest shade and group 1 (susceptibility 1) shown in the lightest shade.**

d. Define the average relative susceptibility among the susceptibles at any point in time $\bar{p}(t)$ as

$$\bar{p}(t) = \sum_{i=1}^{4} p_i \, s_i(t) \Bigg/ \sum_{i=1}^{4} s_i(t) \ .$$

Note that this is simply a weighted average of the susceptibilities of the susceptibles, by adding up the susceptibilities in the numerator and dividing by the number of susceptibles in the denominator. Over the same time window as your previous plot, create two addition figures: First, show $s_i(t)$ for each $i = 1, 2, 3, 4$ using the same color scheme as before. Second, show $\bar{p}(t)$ in black.

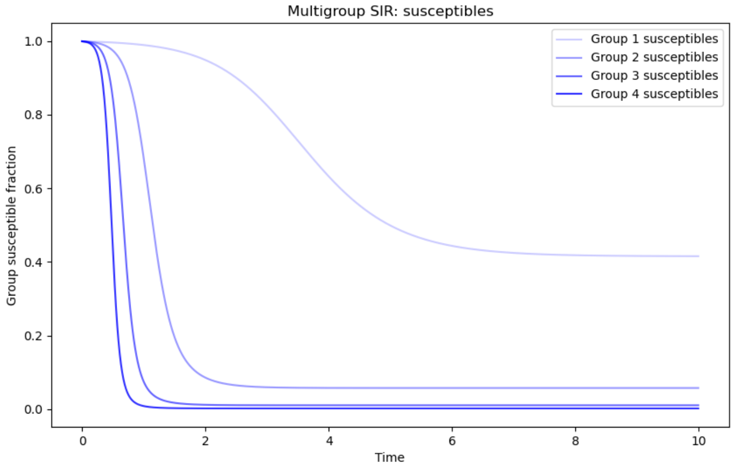

Figure 2: **Susceptible dynamics for four groups with different susceptibilities. Population subgroups are shaded according to their susceptibilities, with group 4 (susceptibility 4) shown in the darkest shade and group 1 (susceptibility 1) shown in the lightest shade.**

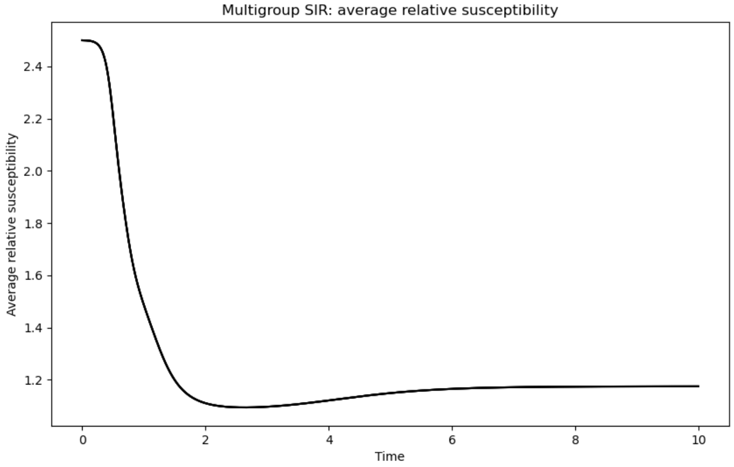

Figure 3: **Average relative susceptibility over time.**

e. Comment on what you observe in the plots, and explain the reason for the patterns in words that a high school student could understand. **In the plot of infecteds over time, we see that decreasing susceptibility pushes the epidemic peak later and flattens the epidemic curve. This is also reflected in the susceptible curves, where the most susceptible group (group 4) drops off and is depleted the fastest, and the least susceptible group (group 1) drops off slowest and reaches a higher final steady-state value around 0.5. The reason that the epidemic peaks earliest and highest for the most susceptible group is that they are more likely to become infected upon contact with an infectious person. This causes the proportion of infecteds in group 4 to spike quickly and then fall off again, as all of the susceptibles in that group become infected and then recovered, when they can no longer become infected or infect others. The third plot shows that the average relative susceptibility drops off quickly as the susceptibles in the higher-susceptibility groups (2-4) are depleted. This leaves mostly susceptibles in group 1, which are not very susceptible—so the average susceptibility is very low. Then, as these group 1 susceptibles finally become infected during the later group 1 peak, the average relative susceptibility rises again slightly. This is because the share of susceptibles that are in group 1 (lowest susceptibility) decreases as they get infected.**

Grad/EC Reflect on these plots in the context of the COVID-19 pandemic. What lessons are there to be drawn from the relationship between an epidemic wave and different groups with different susceptibilities?

In a well-mixed population, the waves of the pandemic will be staggered in both time and intensity according to the susceptibility of different groups. So, for example, if older adults are more susceptible to COVID-19 than younger people, we would expect to see an earlier and larger peak of infections in this group than in younger population groups. Also, susceptibility may also be related to disease severity. In this example, older adults are also more likely to have severe outcomes and require hospitalization or intense care for COVID-19, so it is important to have enough healthcare capacity to handle the large peak in this group.

2. The goal of this problem is to explore branching processes, and how superspreading can, perhaps surprisingly, increase the likelihood that an outbreak never grows to a large size.

   This problem introduces the **negative binomial** (NB) distribution. The distribution can be parameterized a few different ways, but for our purposes, it will be convenient to specify a mean and a dispersion. In the context of transmission chains and branching processes, drawing the number of secondary infections from a negative binomial requires that the mean be R_0. However, the dispersion parameter allows us flexibility, and importantly, allows us to model superspreading.

   When $k \to \infty$, the negative binomial distribution converges to a Poisson distribution. When $k = 1$, the negative binomial is equivalent to a geometric distribution. A key difference between these is that the mode of a Poisson—its most common value—is around its mean, while the mode of a Geometric is zero. In the context of branching processes, this means that a high k will lead to more similar numbers of secondary infections for each primary infection. In contrast, a low k will lead to many instances where there are just a few (or zero) secondary infections, and rare instances with a large number of secondary infections. When k is low, we often talk about "superspreaders," people whose infections lead to an exceptionally larger number of secondary infections.

   Note: You can find Python code to draw from a negative binomial with the R_0 and k parameterization provided on the next page. For those writing in other languages, be careful to use the correct parameterization (mean and dispersion).

   a. Write code for a branching process that, starting from a single infection, draws G generations, with each infection creating $NB(R_0, k)$ additional infections. Use your code to estimate q the probability[2] that an epidemic dies in finite time, for $R_0 = 3$ and $k = 0.1, 0.5, 1.0, 5.0$, and 10.0.[3] Provide your answers in a table, out to 3 decimal places.

      I estimate q by running 1000 branching processes for each value of k and counting the number of processes that terminate. Here are the estimates for q:

k	$\hat{q}$
0.1	0.856
0.5	0.493
1.0	0.327
5.0	0.125
10.0	0.097

   b. How does k affect q? Explain what this means in terms of the relationship between p (i.e., $1 - q$) and superspreading.

[2] See class notes.

[3] Hint: When we estimate a probability from a stochastic process like this, a convenient way to do this is Monte Carlo: simulate, say, $100,000$ branching processes and take note of how many cease before growing large.

As k increases, q decreases. Low k means more overdispersion, or superspreading. So as superspreading increases, the probability that an outbreak goes on forever (p) decreases.

Grad/EC How large do finite outbreaks get before they die out? For the parameters above, and for only the *finite* outbreaks, plot a histogram of $100,000$ finite outbreaks for your choice or choices of k, and $R_0 = 3$. What do you observe?

See figure below with finite outbreak size histograms for $k = 0.1, 1, 5$. Most outbreaks that die out (have finite length) die out after the first generation; the mode of all three histograms is at outbreak size 1. As k increases, the distribution of outbreak sizes becomes less skewed, more very small outbreaks and fewer large outbreaks. For very small k, there are a few outbreaks that infect over 100 people but still die out. This reflects the inverse relationship between k and q.

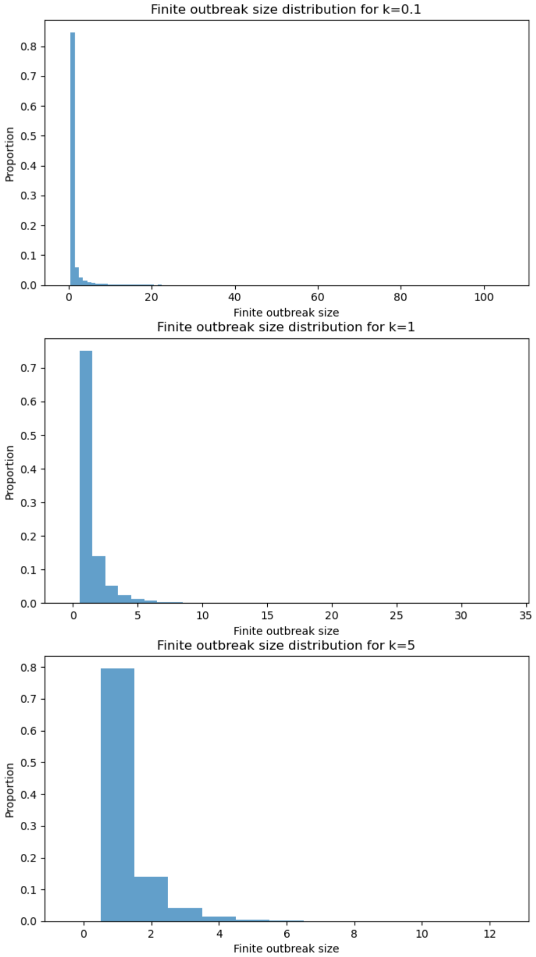

Figure 4: **Histogram of finite outbreak sizes (total number of infected people) for $R_0 = 3$ and $k = 0.1, 1, 5$.**

Here is some Python code to draw from $NB(R_0, k)$:

```
from scipy.stats import nbinom

k = 10000 # Dispersion Parameter k
R0 = 3 # Mean R0

mean = R0
variance = mean + (mean**2)/k
p = mean/variance
n = mean**2 / (variance - mean)

draw = nbinom.rvs(n=n,p=p)
draws = nbinom.rvs(n=n,p=p,size=10)
```