Stochastic Models for Epidemics

Student name: Alice Hankin
Supervisor names: Simon Harris/Jesse Goodman
Host Unit: Department of Statistics
Programme: BSc(Hons)

How The Scholarship Has Furthered My Career Development

This scholarship has been a wonderful opportunity. The most valuable thing for me was getting experience in research. I will be doing my honours year in 2021 and it has been really helpful to know what to expect when undertaking a research project. I have also developed a better understanding of which areas in statistics I am interested in pursuing in the future. I have learned practical skills in computing languages LaTeX, R and Python. In particular, the use of R Shiny to create web applications is completely new to me, but will surely be beneficial experience for future data visualisation projects.

Summary of Research and Significance

Branching processes can be used to model situations such as the population of animal species over time or the passing down of surnames through generations. In particular, they can be used to model epidemics. Parameters - such as the length of the infectious period or the number of people infected by one individual in one day - determine what happens to the spread of the virus over time. In simulating the effect of a virus on a population, we gain some insight into which parameters are most important in keeping infection low, and we are able to predict how a population might be affected.

Abstract

This report details the development of multiple web applications written in R and R Shiny that can be used to model branching processes in various ways. The apps do as follows:

(1) Outputs a random tree derived from some offspring distribution. It allows for detection at a particular generation, or a user-inputted detection probability.

(2) Plots population over time for multiple random trees, each with the same offspring distribution, and a fixed detection probability. It also outputs histograms for both population detected and time detected, as well as what we would expect to see theoretically.

(3) Uses a multi-type model, with both infectious and non-infectious individuals, and outputs the population over time.

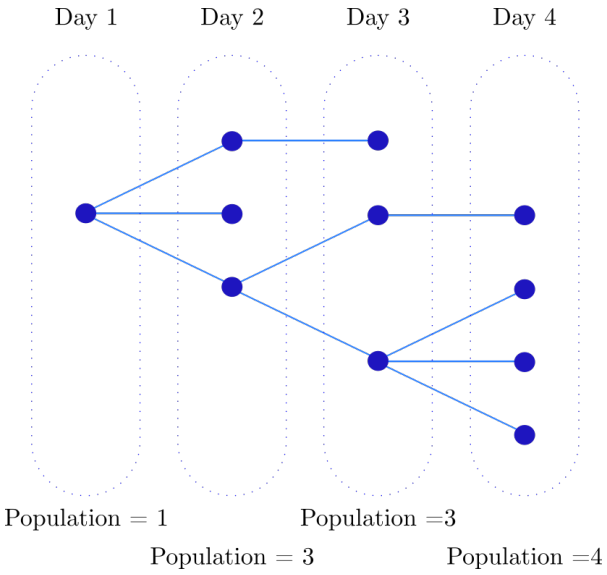
(4) Imitates a continuous-time branching process, and also plots the population over time.

(5) Gives a simulated distribution of the most recent common ancestor for two random individuals in a given generation of a branching process, and compares it to the theoretical result.

Report

Let's suppose we have a population where one individual is infected with a virus. Every day, we record how many people are infected and who infected them.

We can represent this process by drawing a tree, such as the one below.



Each dark blue nodes represents one infected person and the light blue lines show who infected them. One person is infected on day 0, then three on day 1 and so on.

We think of this as replacing each infected person on day n with an independent random number of newly infected people on day $n + 1$. These random numbers are taken from a probability distribution we call the offspring distribution.

This process is called a branching process.

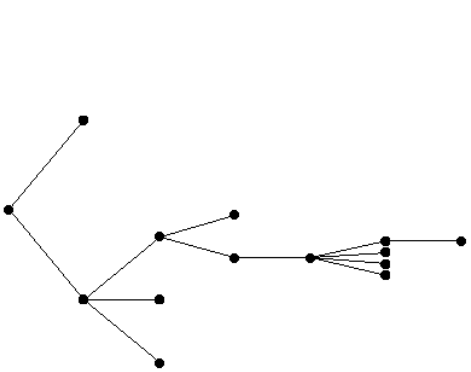
In this report I discuss the ways in which we can understand branching processes through simulation. However, there is a lot of research that has been done on the theory of branching processes. I have left links to some of the research that helped me in the references at the end of this report.

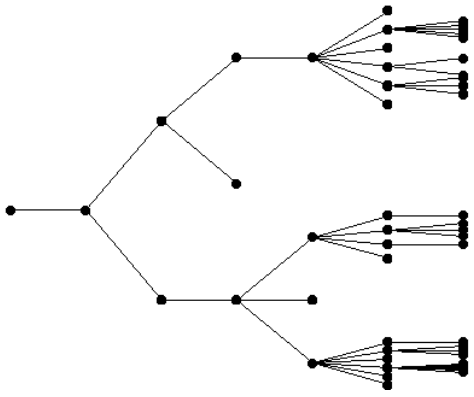
All code I have written can be found at https://github.com/alicemhankin/srs-2021

Tree Simulation

I started by writing a simple program. The user would input the geometric parameter of the offspring distribution, and the program would output a random tree.

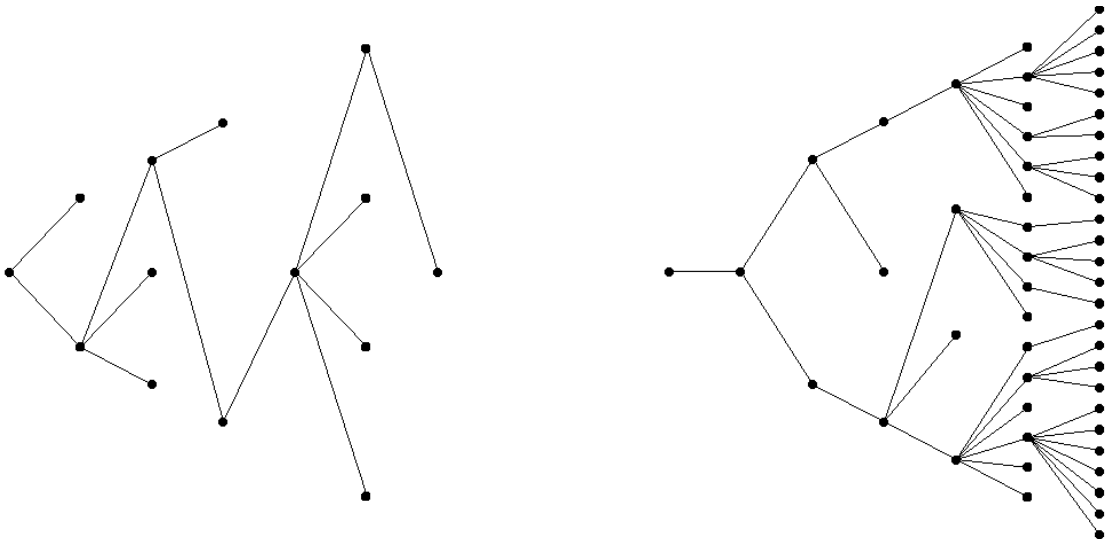
For example, with parameter $p = 0.35$, the program might output either of the following plots:



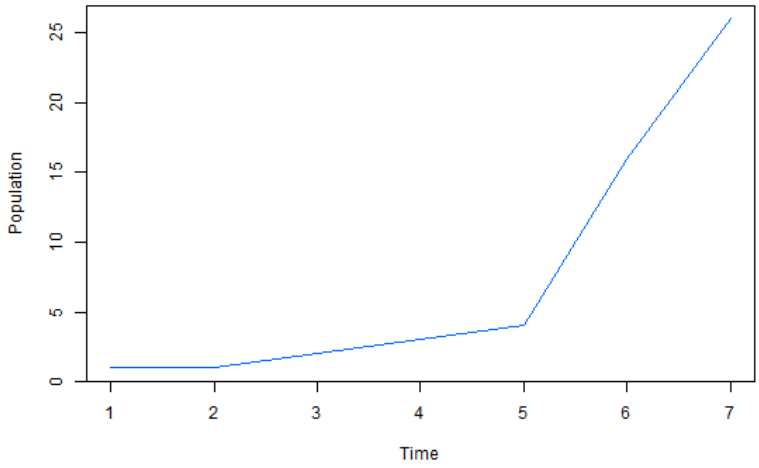


These plots detail the structure of the tree, but it is apparent, especially for bigger trees, that there can be overprinting where the nodes start to overlap.

To combat this, I included an option for the user to see the same tree, but with the nodes evenly spaced. The trees above could also be drawn as follows:



It is also useful to plot the number of infected people over time. For example, the second tree above corresponds to a graph that looks like:



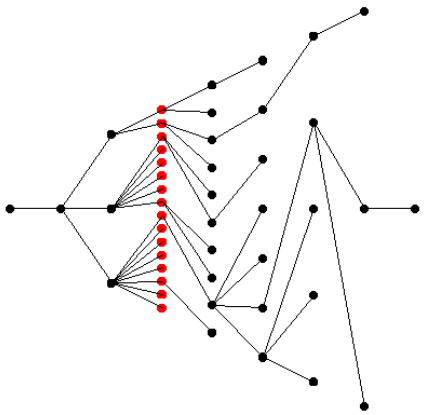
It is intuitive that if the offspring mean (this is the average number of people who get infected by one individual per time-step) is above 1, then the population is likely to grow each generation. Similarly, if the offspring mean is below 1, then the population is likely to drop each generation.

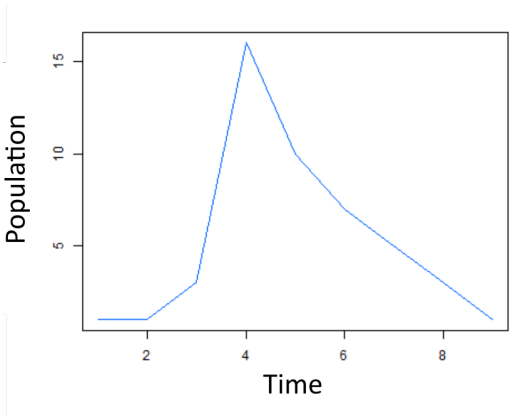
In fact, we expect the population of infected people in generation n to be μ^n, where μ is the mean of the offspring distribution.

Change of parameter

Let's suppose that the offspring mean is high (above 1). This is representative of an infection growing exponentially. Then lets say that at some generation n, the virus gets detected. The population starts practising good hygiene, wearing masks and so on. Then, the offspring mean drops to below 1.

Let's see what happens when we simulate this. Let's suppose the mean changes from 6 to 0.5 in generation 4.

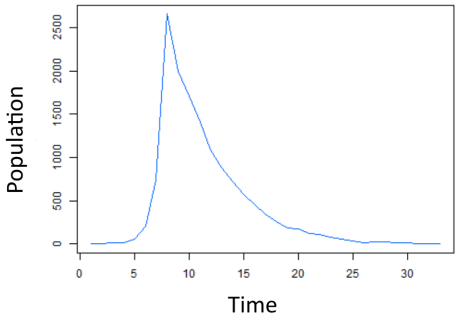




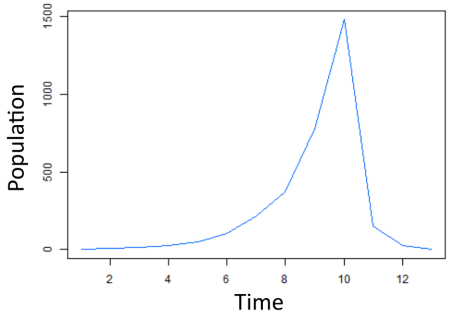
Here, the red nodes show where the switch happens. We can see that the population/time graph grows approximately exponentially up until generation 4, and then it decays exponentially after generation 4.

Changing the offspring mean before and after the switch, as well as in what generation the switch happens changes how the population graph looks:

Switch in gen 8, first mean 3.5, second mean 0.8



Switch in gen 10, first mean 2, second mean 0.1

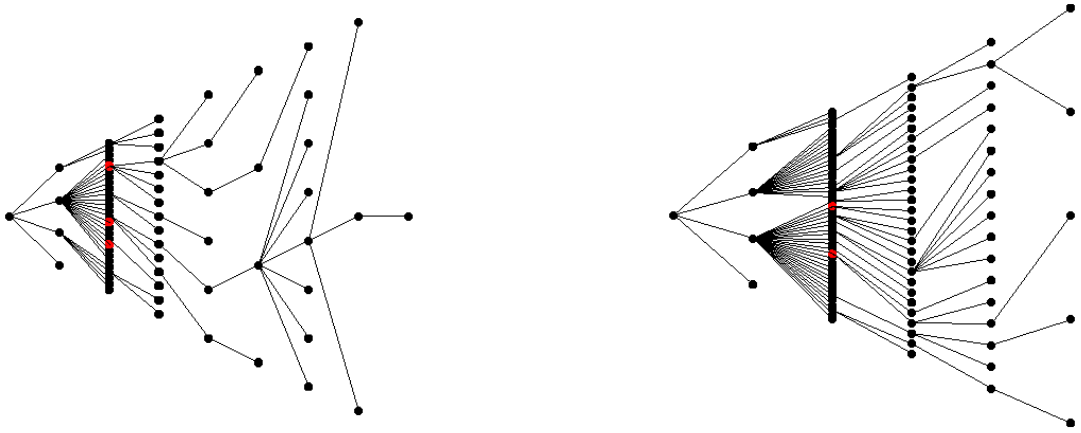


Detection

Now, artificially choosing a time for when the "switch" should happen might not be realistic. In real life, if the population of infected people is high, then there is a high probability that the virus is detected. If the population is small, there is a low probability the virus is detected. Another way to put this is that the probability of detection is proportional to the population of infected people.

We say that each person has some probability (the "detection probability") of being detected. The generation in which the first detection event occurs triggers the mean to change from high to low.

On the following page are two simulations with detection probability 0.05 that change from mean 3.5 to mean 0.3. The red dots represent when the virus gets detected for the first time.

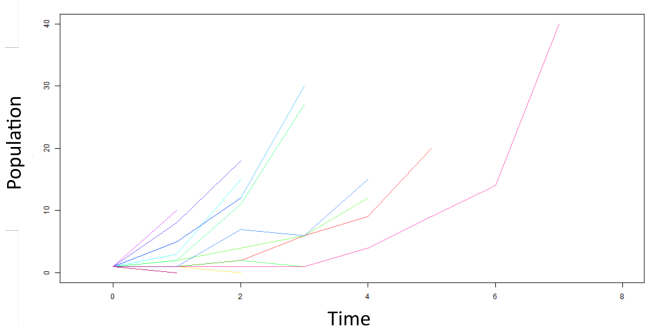


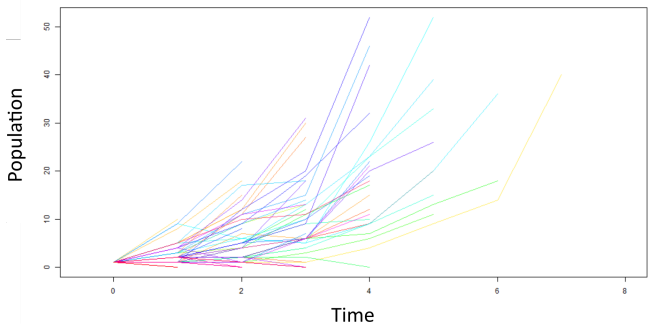
The simulation can be found at https://alicemh.shinyapps.io/tree/

Histogram

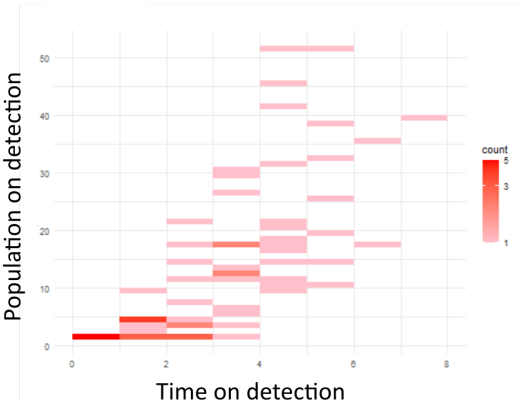
What happens if we run the simulation over and over again? Let's first exclude the tree diagrams, and only keep the population/time graphs. Let's also forget what happens after the virus gets detected.

Below is what we get after running the simulation for 20 repeats (with detection probability 0.05 and geometric parameter 0.33) and then 120 repeats . The lines end either when the population hits zero, or when the first detection event occurs (whichever happens first).

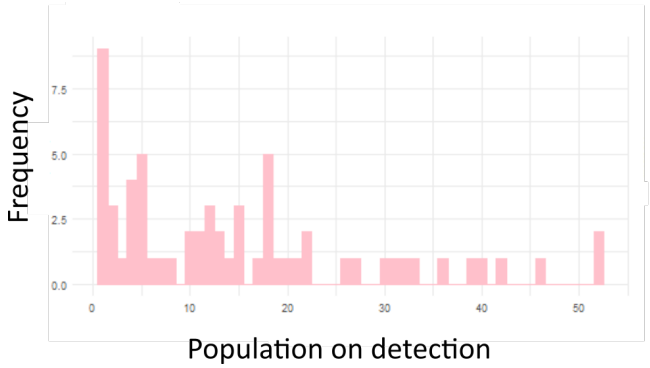


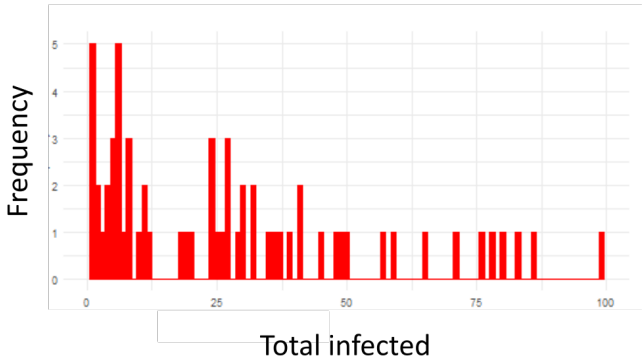


The line graph alone tells us a lot, but we can make it easier to visualise with a scatter plot. For all the trees that got detected (before the population hit zero), we can plot the time of of detection against the population at detection.

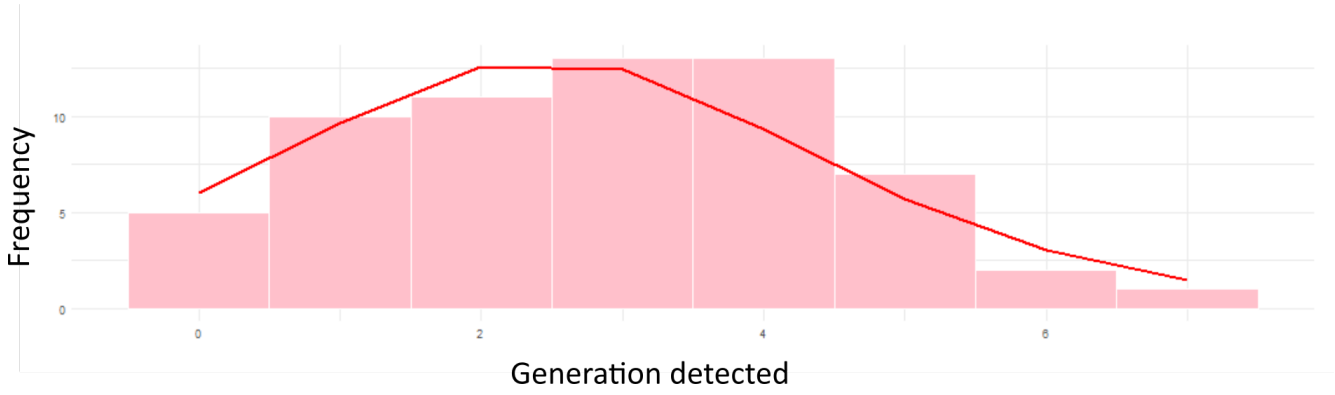


We can also plot only the population at detection, as well as the total number infected:



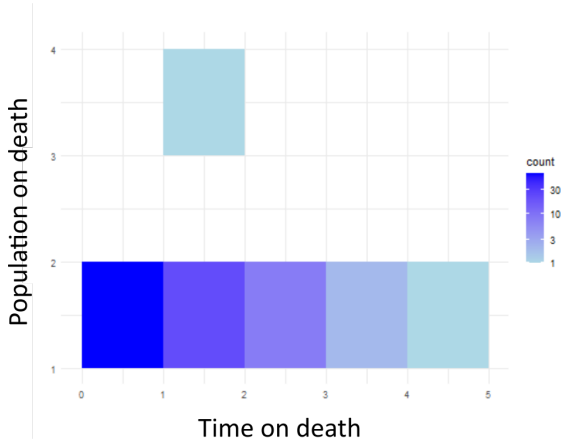


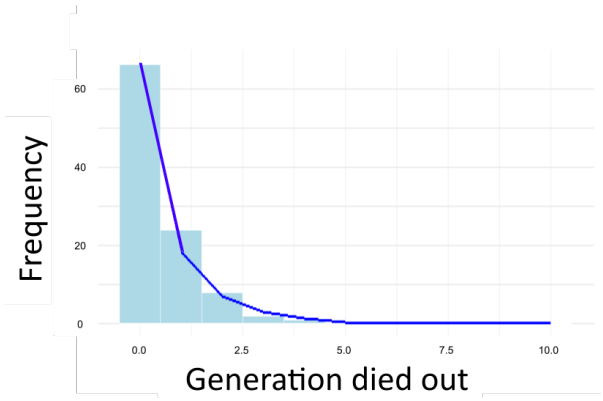
Or only the time of detection:

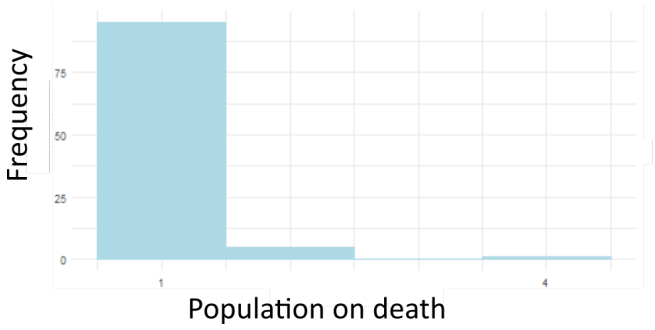


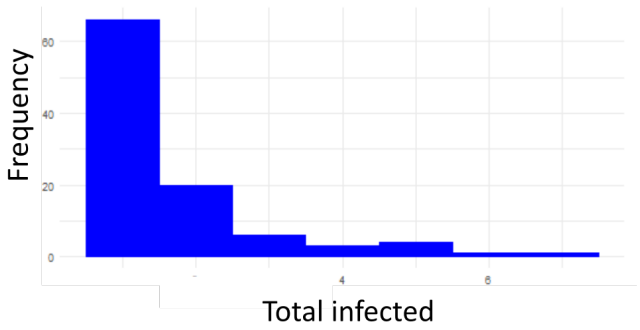
For the graph directly above, the pink histogram is the simulated results. The red line graph shows what we would expect to see, as calculated using generating functions.

We can also plot the same things for the trees in which a detection event did not occur. These trees ended when the population hit zero. Instead of plotting what happens at detection, we plot what happens when it dies.









The simulation can be found at: https://alicemh.shinyapps.io/histogram/

Multi-Type Branching Process

The multi-type model has two different types of people - those that are infectious and those that are not infectious.

If one is not infectious, every time-step, they either stay alive and non-infectious OR they become infectious.

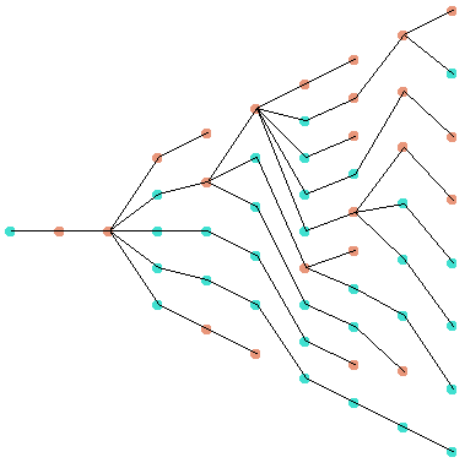
If one is infectious, each time-step, they either die, OR they stay infectious and spread the infection to a random number of people (these people are non-infectious).

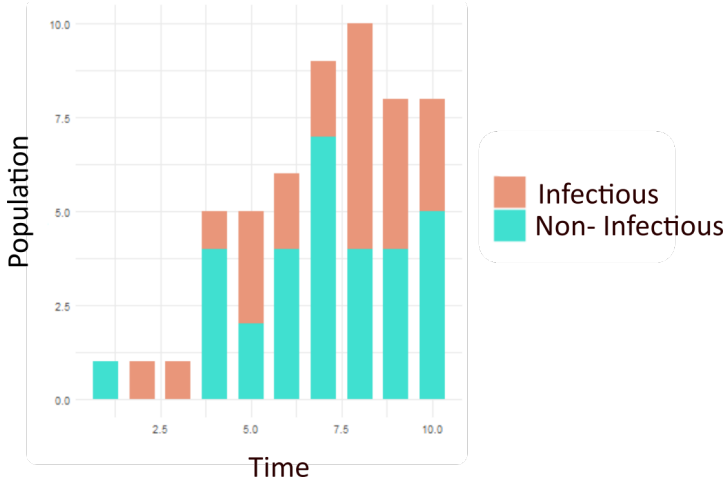
We start off with one non-infectious person.

The user can input:

- The probability that a non-infectious person becomes infectious (p)
- The probability that an infectious person dies (q)
- The geometric parameter for the spread of infection (r) both before and after it changes
- The generation when the parameter r changes

The program will output the tree, and the population over time.





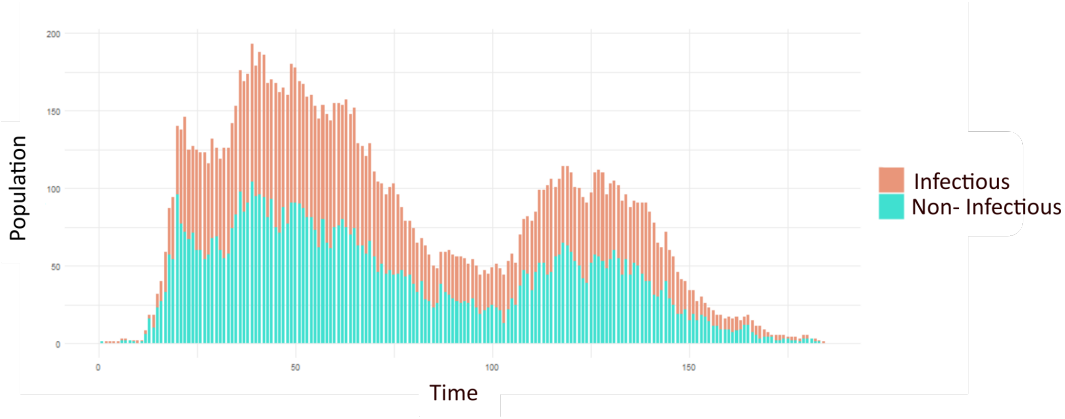
The above plots have $p = q = 1/3$ and $r = 0.35$ both before and after the change.

The problem with this is that, if we want the maximum number of generations to be big, the graphs of the trees start to get visually cluttered.

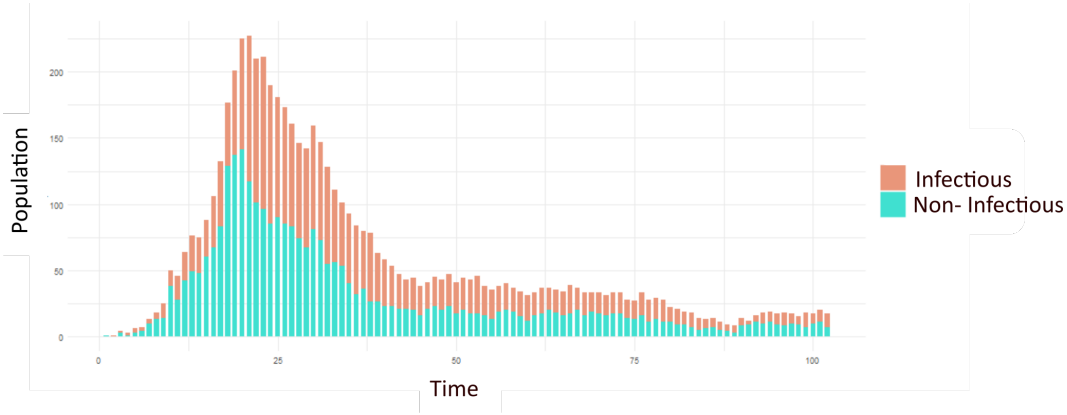
So, I created an alternate version of the app where the trees are omitted. This allows the maximum number of iterations to be in the hundreds, rather than around 20.

This gives really interesting results such as:

(a) Parameter change from 2.5 to 0.5 in generation 20



(b) Parameter change from 2.5 to 0.3 in generation 20



(both of the above examples do have $p = q = 1/3$

The simulation with no trees but high maximum time can be found at:
https://alicemh.shinyapps.io/multitype/

The simulation with trees but low maximum time can be found at:
https://alicemh.shinyapps.io/multitype2/

Continuous time

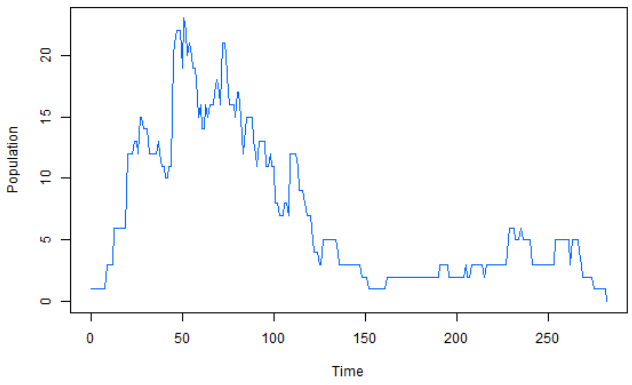
Instead of using a continuous time framework, we can modify the existing discrete-time code in order to model a continuous time branching process.

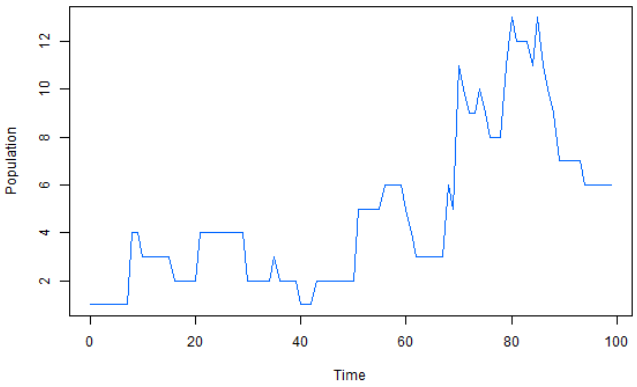
Here, we have a model as follows. At each time-step, each infected person:

- Remains infected but doesn't spread the infection - with probability q (close to 1)
- Otherwise, spreads the infection to a geometrically distributed (parameter r) random number of people

The amount of time before the infection spreads is approximately exponential.

Here are some population/time plots (these use $r = 1$):



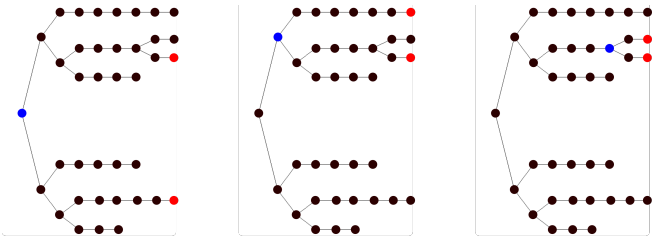


The simulation can be found at https://alicemh.shinyapps.io/exptree/

Common Ancestors

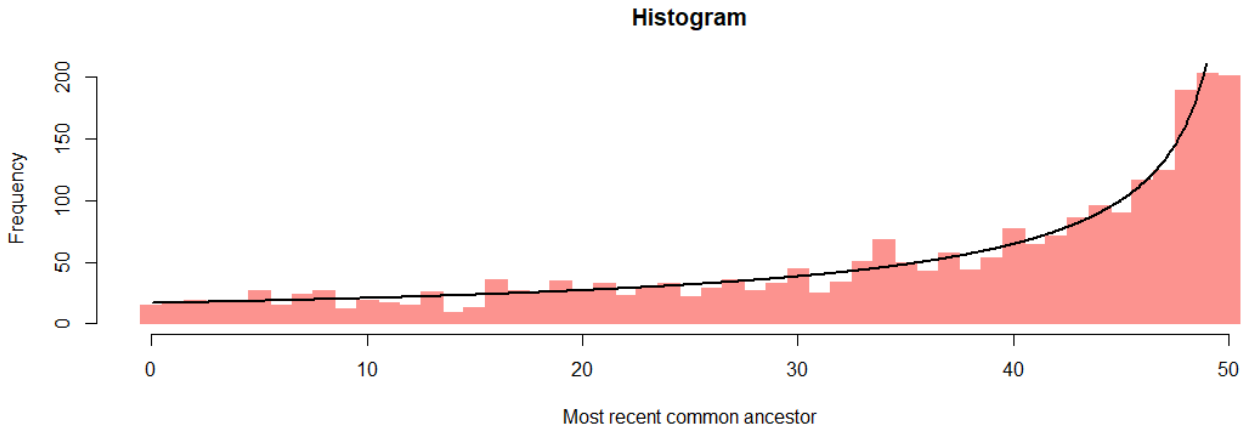
Let's suppose we have a branching process with offspring mean 1. We run a simulation until we have a tree with at least n generations. If we pick two points at random in generation n, can we then predict in what generation their most recent common ancestor is?

The image to the left shows two points in generation $n = 8$ in red, and their most recent common ancestor in blue.



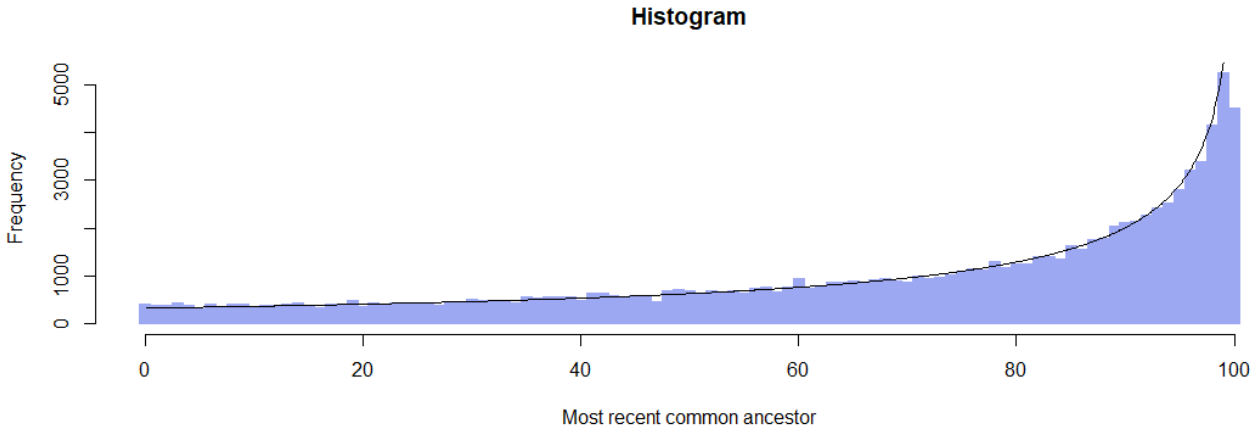
In *The Coalescent Structure Of Continuous-Time Galton–Watson Trees* (Harris, Johnston & Roberts), it was discovered that, as a proportion of n, the distribution for the generation of the most recent ancestor looks like: $\frac{-4x-2(x-2)\ln\left(\frac{1}{1-x}\right)}{x^3}$

I can run a simulation to corroborate this. The coloured histogram is the simulated results and the black line is what we would expect to see.



The above simulation has 500 trees, 5 samples per tree, and we are sampling at generation $n = 50$

We can increase each of these values to get even closer to the black line:



This has 10,000 trees, 10 samples per tree, and we are sampling at generation $n = 100$

References

Athreya, K. B., & Ney, P. E. (2004). *Branching Processes (Dover Books on Mathematics)* (Illustrated ed.). Dover Publications.

Durrett, R. (2010). *Probability: Theory and examples.* Cambridge: Cambridge Univ. Press. https://www.math.ucla.edu/∼ biskup/275a.1.20f/PDFs/Durrett-v5.pdf

Grimmett, G., & Welsh, D. A. (2003). *Probability: An introduction.* Oxford: Oxford.

Harris, Simon C., Johnston, Samuel G. G & Roberts, Matthew I. (2020). *The Coalescent Structure Of Continuous-Time Galton–Watson Trees.* The Annals of Probability. https://doi.org/10.1214/19-AAP1532

Lyons, R., Pemantle, R., & Peres, Y. (1995). *Conceptual Proofs of L Log L Criteria for Mean Behavior of Branching Processes.* The Annals of Probability. https://doi.org/10.1214/aop/1176988176

Probability generating functions. (2002). Centre for Theoretical Atomic, Molecular and Optical Physics. http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf