Stride Simulations

Cedric De Schepper
Cedric.DeSchepper@student.uantwerpen.be

Thanh Danh Le
ThanhDanh.Le@student.uantwerpen.be

Wanner Marynen
Wannes.Marynen@student.uantwerpen.be

Stijn Vissers
Stijn.Vissers@student.uantwerpen.be

June 19, 2019

-

1 Abstract

In this paper we'll discuss what to focus on when running simulations, what influences the simulations and how to interpret the results.

2 Simulation

2.1 Introduction

In this section, we'll explain why we have to account for stochasticity, make an important distinction between outbreaks and extinctions, and demonstrate how much different variables can influence the results.

2.2 Stan

The Stan (STochastic ANalysis) controller makes it possible to investigate the influence of stochasticity on simulation results. It runs a given number of simulations, for the same parameter configuration, but with different random number seeds. The output for the different runs is aggregated in a single .csv file. This file contains, for each random number seed (columns), the cumulative number of infected cases per time-step (rows). The first row specifies the random number seed used for each run.

The Stan controller can be addressed from the command line, in the following way:

```
$ ./bin/stride e sim c [CONFIG FILE ] stan [COUNT]
```

For this section, we used the configuration file stochastic_analysis.xml, which is present in the config folder of Stride. [COUNT] indicates the number of simulation runs that should be executed. You can choose the value of [COUNT] yourself.

This output file will be used in the following sections.

2.3 Stochastic variation

By using the results produced by the Stan controller, mentioned in the previous section, to plot the number of cumulative cases per time-step in Fig.1, it's very easy to see that there exists a large difference between the number of infected cases observed during every run. Either the number of infected cases is minimal, or the amount rises to a fairly constant number. Important to add is that the horizontal line with a value of around 0 infected cases isn't a single line but represents around 50% of all simulations. By limiting the y-axis to only 100 infected cases and the x-axis to 250 days as seen in Fig.2, this distinction becomes visible.

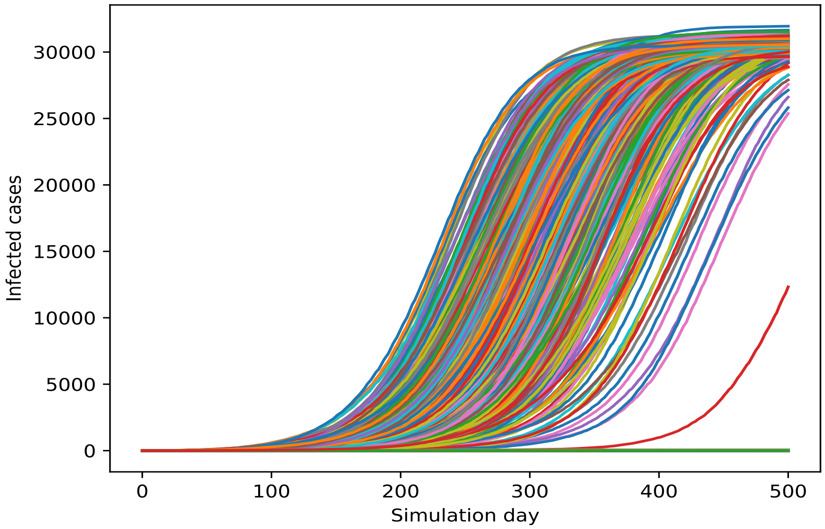


Figure 1: Cumulative cases of 1024 simulations with a period of 500 days

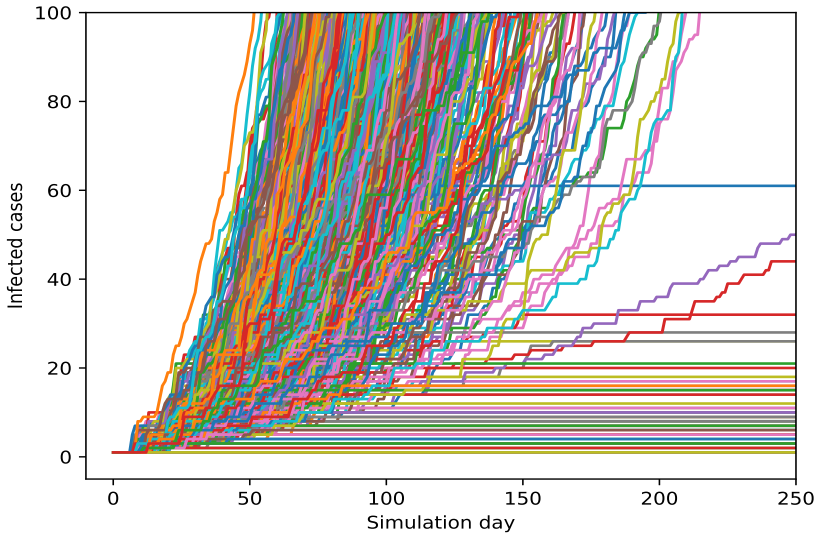


Figure 2: Cumulative cases of 1024 simulations with a period of 500 days

This distinction can also be seen when plotting the newly infected cases per day in Fig.3. The minimum and maximum values vary greatly, which again indicates the two different simulation results. By combining both Fig.1 and Fig.3, it becomes clear that chances does play an important factor in determining the outcome of a simulation. The stochasticity neatly divides the results into two separate cases: one where an outbreak occurs, and one where an outbreak is averted. This stochastic variation must be taken into account when analyzing result or it might influence the observed findings.

The graph in Fig.3 representing the mean value forms a bell-like curve and is a excellent visualization of the distribution of new cases per day. The number of transmissions per day first starts slowly. Then the disease spreads faster, until it reaches its full potential. After this point has been reached, it becomes less evident as before to find susceptible victims, resulting in the number of new cases to dwindle until the final amount of infected cases has been reached. The fluctuating nature of both the max and mean graph is caused by the weekends, causing people to come into contact with infected persons less often than during a week day.

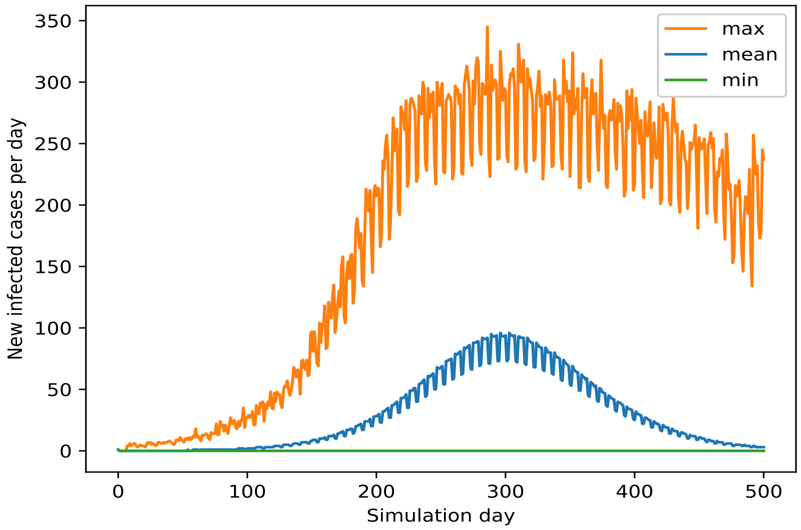


Figure 3: Newly infected cases per day of 1024 simulations with a period of 500 days

2.4 Determining an extinction threshold

As shown in the previous section, the introduction of an infected individual in a partially susceptible population does not always lead to an outbreak. In some cases, only a few or even no secondary cases are observed, while in other cases a large outbreak occurs. These situations with only a few or no secondary cases are called extinction cases. Sometimes we are only interested in one out of the two cases. If so, simulation runs of the other case could skew the results. For this purpose, it'd very helpful if we can determine a threshold to distinguish both cases from each other.

To do this, we'll use the configuration file *stochastic_analysis.xml* again to run a number of stochastic simulations. By plotting the frequencies of the final number of infected cases displayed in several bins, the distinction between outbreaks and distinctions once again becomes clear (See Fig.4). In the event of an extinction, the disease ends up infecting between zero to fifty people. This can be observed when looking at the horizontal lines in Fig.2. In the second case, an outbreak occurs. This results in the disease spreading among the population, infecting around thirty thousand people as seen before in Fig.1. Since we are dealing with a population of 600 000, this corresponds to approximately 5% of the total population. Since the vaccine_rate parameter is 0.9, 10% of the population was not vaccinated, which is a total of 60 000 persons. This means that around 50% of the unvaccinated population is infected.

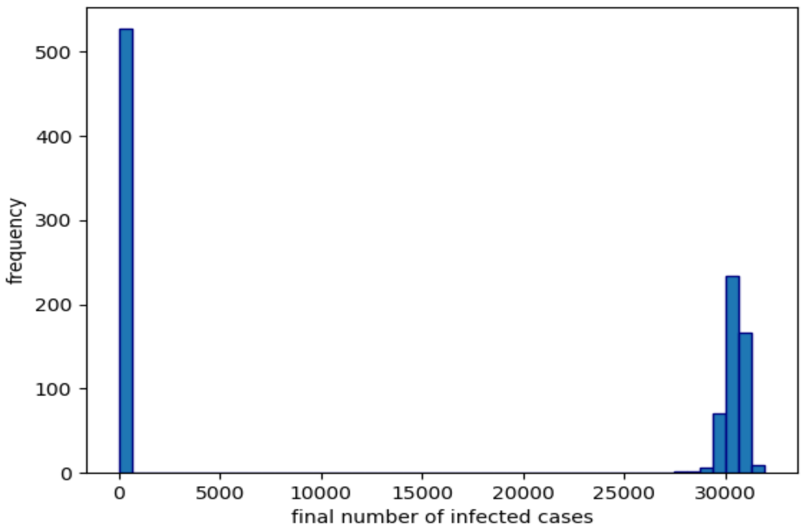


Figure 4: Frequencies of the final number of infected cases displayed in several bins showing a clear distinction between outbreaks and extinctions

Why do we see a 50-50 relation between extinctions and outbreaks? Here the stochastic variation impacts several simulation elements: the location of our ”patient zero”, the contact between people, transmission of the disease, etc. All this elements will impact the occurrence of an outbreak.

In real life, such a phenomenon can be observed, in particular when a community decides not to vaccinate themselves because of e.g. religious reasons. This results in all of the susceptible people in the community being infected when a case is introduced. On the other hand, when a case in introduced in a community where everyone is vaccinated, the disease has no chance of spreading, and thus outbreaks are prevented.

Now that we have proven the two possible outcomes, we can try to determine a threshold to distinguish them. By plotting the total infected cases of a large amount of simulation runs as seen in Fig.5, it might be possible to find the threshold for our configuration. For the 1024 simulation runs we did, we found a threshold of 50. When a run reached a higher infected count, it always ended up being an outbreak. However due to the stochastic nature of the simulation runs, it is impossible to deduce an exact value. Since we still do want to choose a threshold, we'll again look at Fig.4. There we can see that there exist no bins/simulations in the range of 5000 to 25000. As a threshold we can then simply choose a value in this range.

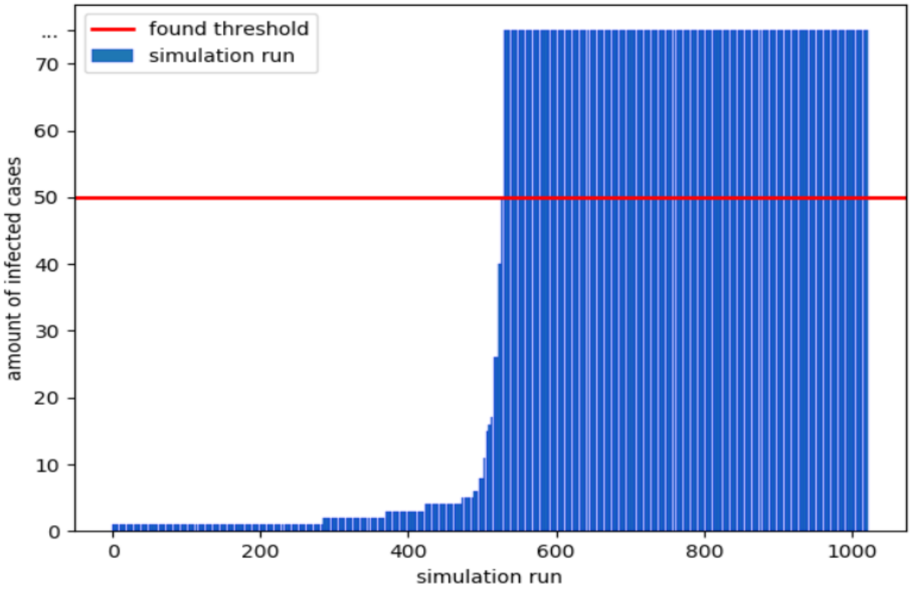


Figure 5: Final number of infected cases for 1024 runs

2.5 Estimating the immunity level

Suppose we have data about the evolution of a recent measles outbreak. Over the course of 50 days, data was collected on the number of new cases that was observed each day. The result can be seen in Figure 6. We already have a good estimate of most relevant parameters. These are described in the configuration file *outbreak*_2019_*estimates.xml*. However, we do not have any information about the number of individuals that were immune to measles prior to the recorded outbreak. Using Stride, we could try to estimate the immunity level of this population. Vary the parameter that determines the immunity level in the population, 2 and try to estimate which was the case in the original population. Remember that: Stochasticity plays a role in Stride The data we have are from an outbreak of measles It is useful to use the PyStride environment for this assignment. However, it is not required.

We have to determine the immunity level based on the limited data provided in the picture. The main data we needed to use to determine the immunity level is the cases per day. After testing the complete range of immunity's we determined that the immunity of the real population will be around 70%.

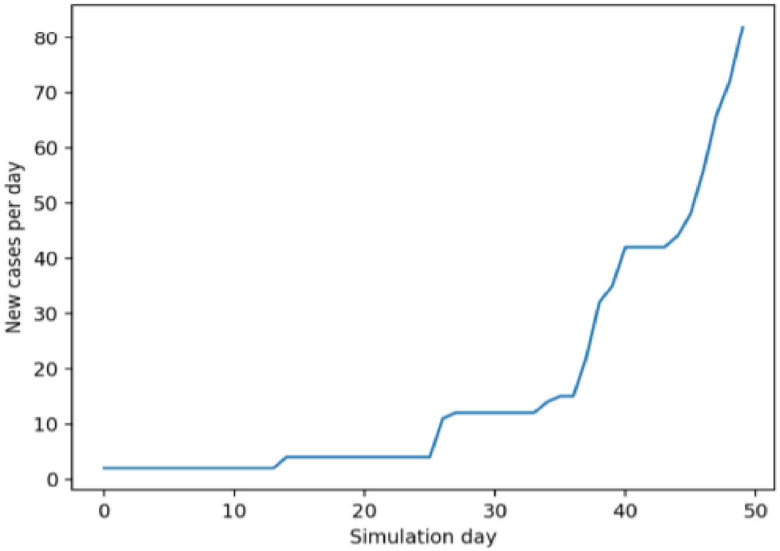


Figure 6: new cases from original data

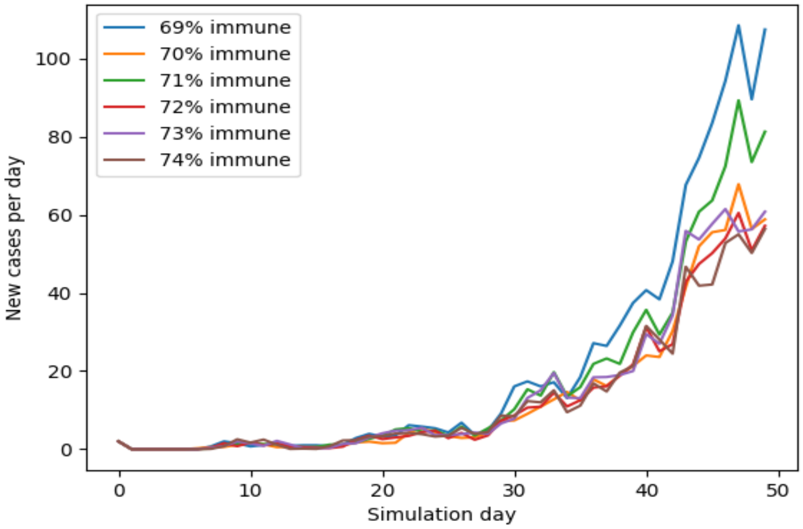


Figure 7: new cases simulated data

To estimate a closer percentage we use a smaller range, from 69% to 75%. The latter test shows us that the real percentage will be roughly 71%.

To improve the accuracy of the simulations we ran each simulation 20 times and then we used the mean value of the new cases per day of the 20 simulations.

2.6 Estimating R_0

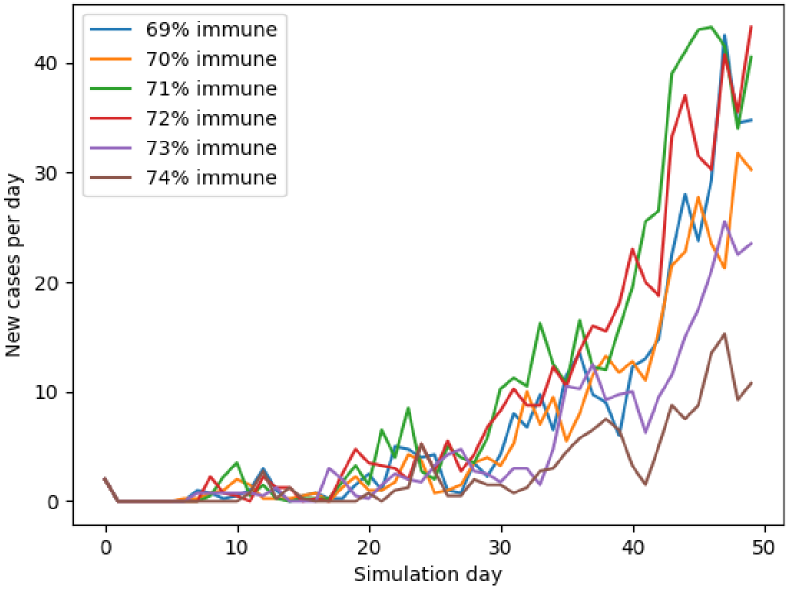


Figure 8: new cases per day with an R0=12

When we check the conclusion of the last question with other R0's we see that it only holds for values around 15 which was the R0 that was used in the simulation for the previous assignment.

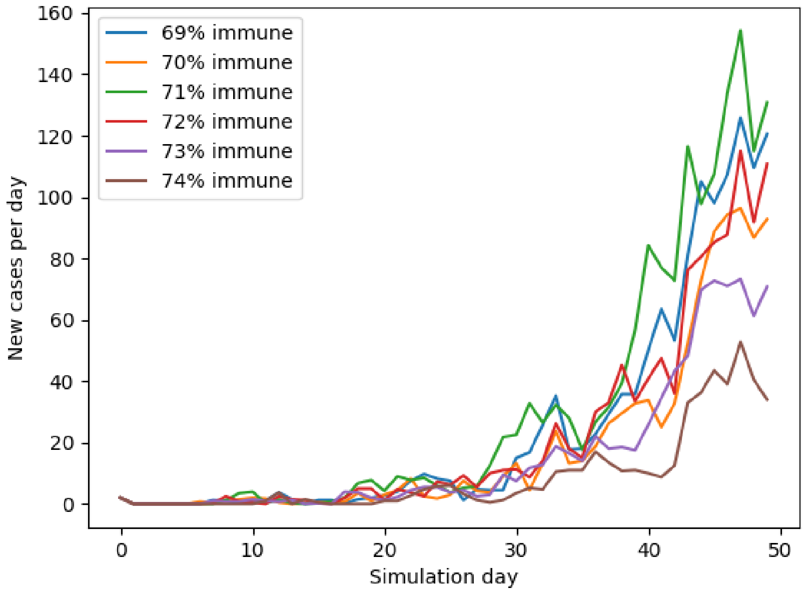


Figure 9: new cases per day with an R0=18

For an R0 value of 16 it is clear that the value of the immunity level will be around 74%. This increase is logical because the disease is more infectious so with the higher number of infectious contacts per day a smaller percentage of people need to be infected to keep the same amount of new cases per day.

3 Population generation

In the next sections we will investigate what happens when certain parameters for a simulation change.

3.1 Investigating the influence of demography on epidemics

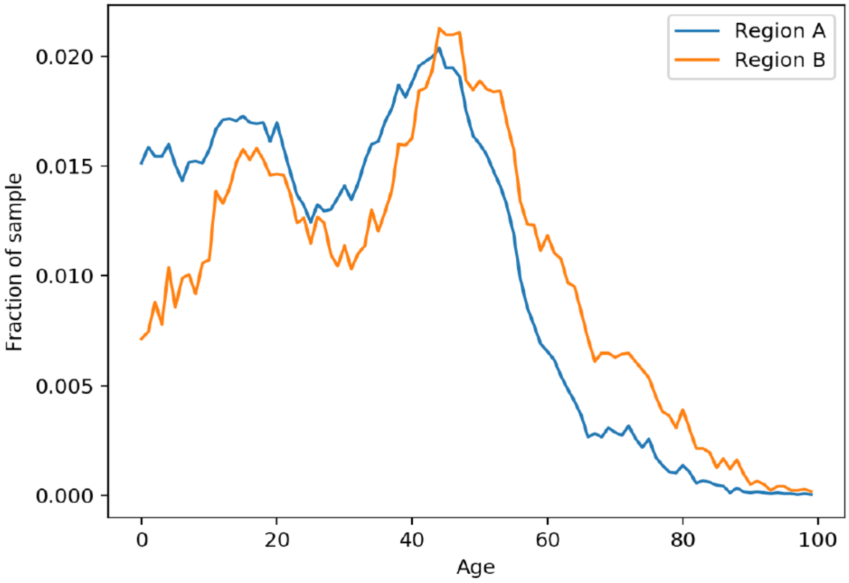


Figure 10: Comparison of age distributions in household samples from region A and region B.

Using Stride, a number of simulations can be run for each population. Before running a simulation, a few parameters must be set accordingly to produce adequate results.
The duration of a simulation will be set to 365 days which equals a year. This gives room to observe what happens after the peak of an outbreak. Next, the initial number of infected cases is set to equal 1 person. This is done by setting the seed to 0.00000167. An outbreak must start with someone after all.

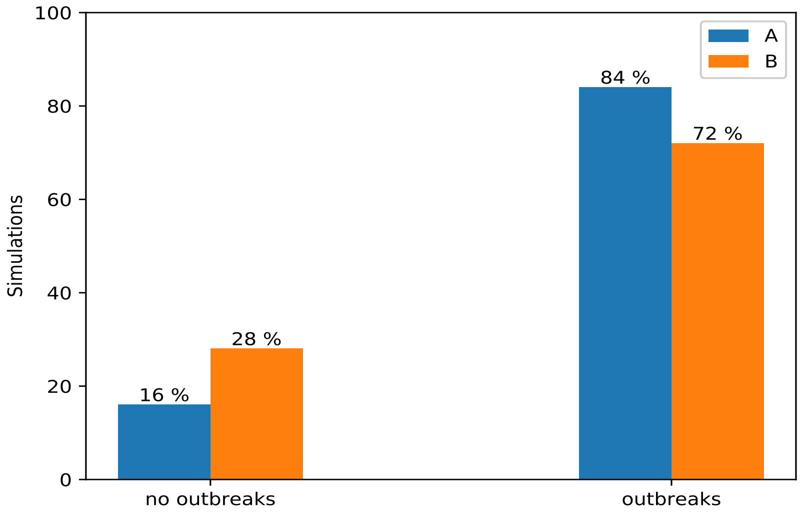


Figure 11: Percentage of outbreaks in region A and B after 100 simulations

As seen in Fig 11, region A will have a higher chance of outbreaks. Although the difference is roughly 10% between the two populations, this does not mean that the chances of outbreaks in region B are low. We consider an outbreak if the threshold determined in section 2.4 is exceeded. However these two graphs don't really show how an outbreak transpires in either of the populations.

Fig 12 and Fig 13 display the newly infected cases per day over the simulation period. Here, it is noticeable that the peak of an outbreak in region A can be reached earlier than in region B. Thus, in region A an outbreak can die out earlier too. Next, it can be noted that the peaks in region A can reach a higher number of newly infected cases per day. A Hypothesis could be that because region A has a younger population, There would be more commuting which means more people come into contact with each other. This could then potentially lead to more infected cases per day.

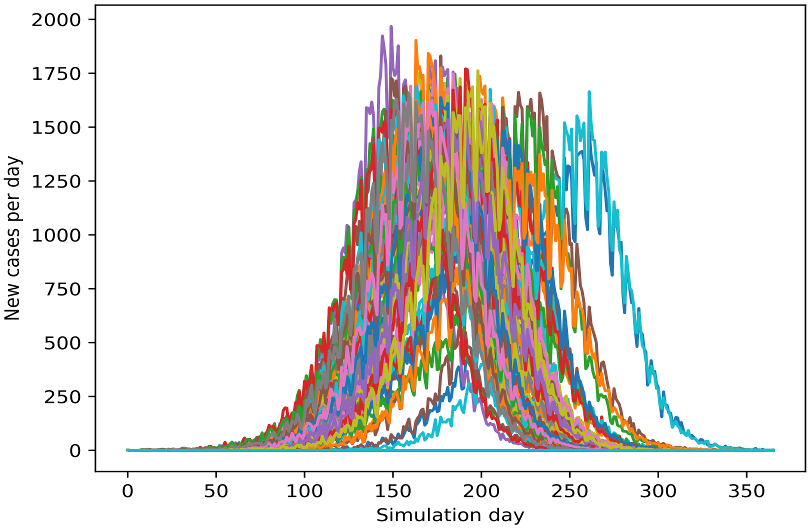


Figure 12: New cases of infected per day for region A of 100 simulations over a period of 365 days

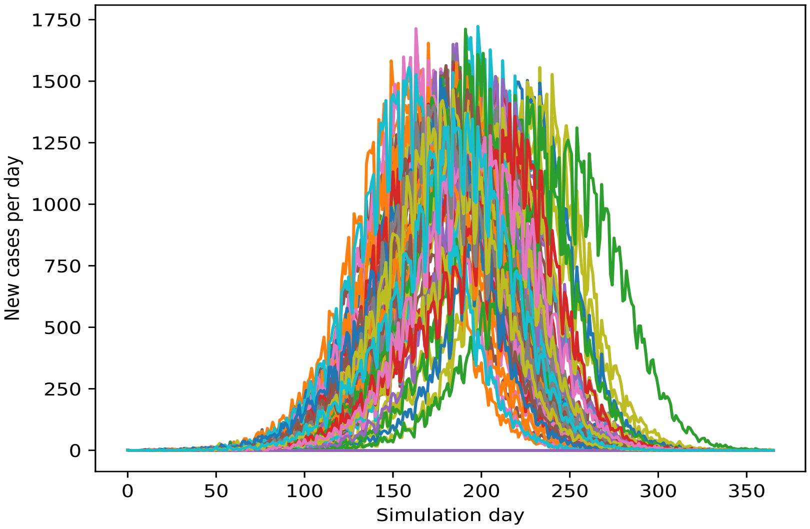


Figure 13: New cases of infected per day for region B of 100 simulations over a period of 365 days.

3.2 Vaccinating on campus

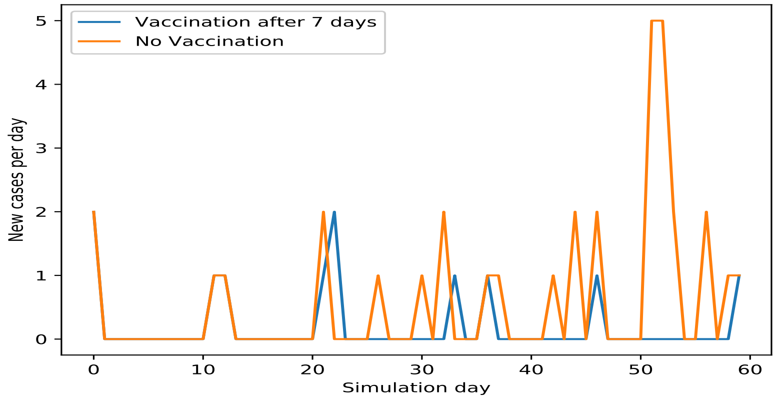


Figure 14: Newly infected cases per day when vaccination was done after 7 days and when not.

Now as seen in Fig 15, when making sure that college students between 18 and 26 are vaccinated, it can be noted that vaccinating does have a clear effect. Looking at Fig 15, the effect of vaccinating after a week can be clearly seen after day 20. From then the period between new infected cases, becomes longer and the amount of new infected cases per day does not seem to go higher than when no vaccination took place.

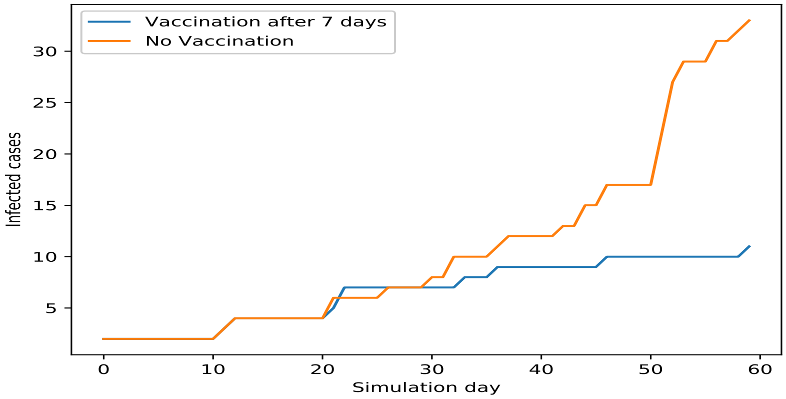


Figure 15: Cumulative cases when vaccination was done after 7 days and when not.

3.3 Is commuting to work important for disease spread?

Using stride we tested this assumption and ran simulations for commuter percentages from 0% to a 100%. These simulations confirm our assumptions as it shows that the 0% commuting population takes longer to reach its peak and has a longer period where it is infecting a large amount of people. While the 100% commuting population peaks quicker and is shorter as infectious.

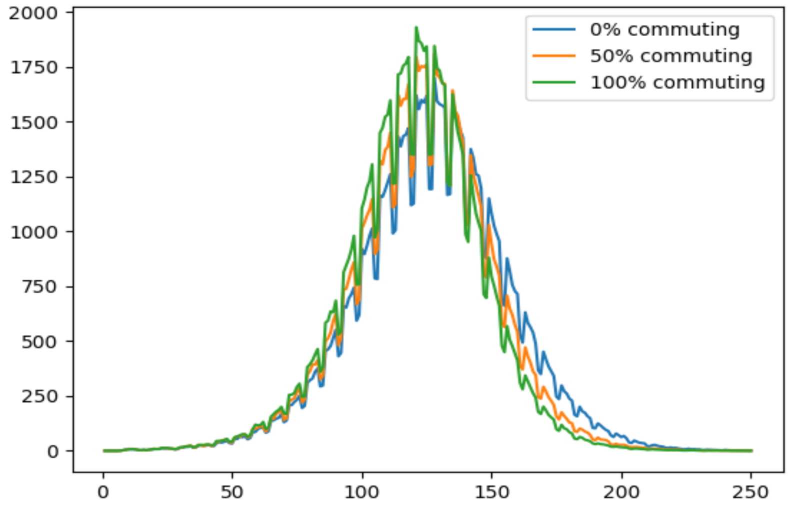


Figure 16: The means new cases per day of 16 simulation runs per commuting level

4 Performance profiling of sequential code

For the last part of this paper, we analyzed the running times of stride using the *GProf* tool.
To be able to do this, we first had to recompile the stride project with the extra `-pg` flag for the `CXX` compiler. After doing so, our stride executable will now dump some output when executed. Afterwards, the *GProf* tool uses this output to analyse the performance.
This output is not readable at all due to the size of the stride project. Because of this, we used another tool called *gprof2dot*, which is able to create a dot representation of the call graph.
This representation is much easier to analyse, and the results of our analysis can be found below under the form of bar charts. In these charts, the wall-clock time (in seconds, using 1 thread) of each procedure will be plotted and compared to the default case.

4.1 Default case

Before we start varying the different parameters, we first established a default scenario to compare our results with.
In Figure 17 the results of our default scenario can be seen. On the left side of the figure the time it took to create the population and run the simulation, when reading the population from a file, is shown. On the right hand side, you can see the data when the population is generated instead of read from a file.

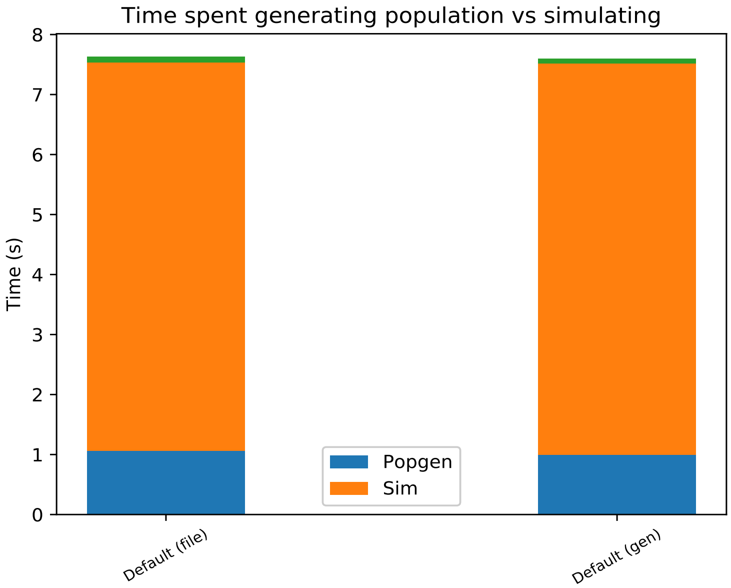
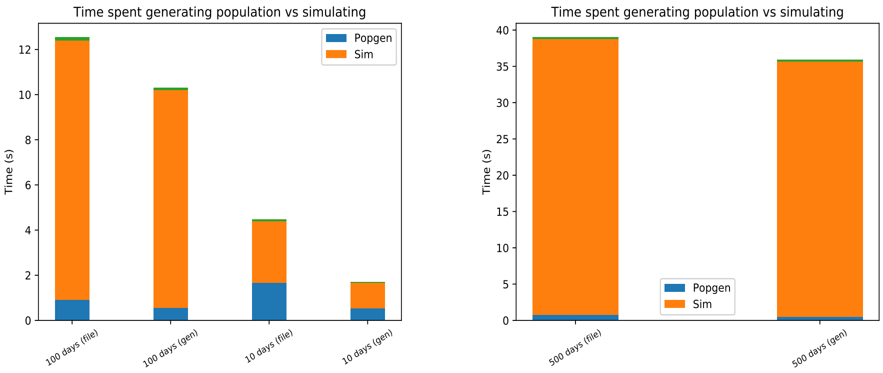


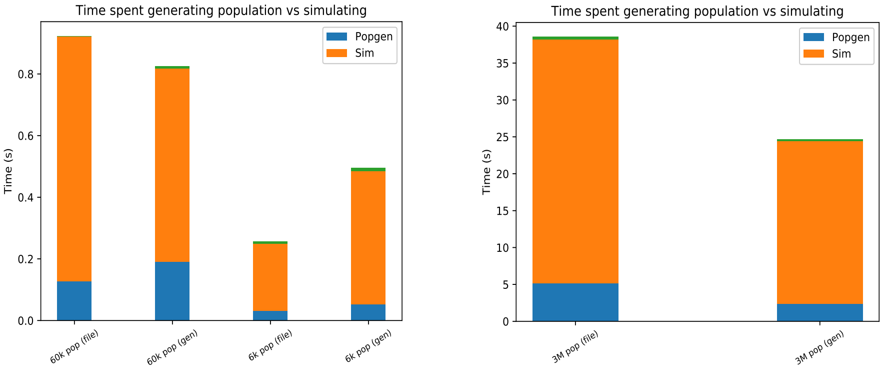
Figure 17: Analysis of the default scenario.



(a) Simulation during 100 and 10 days

(b) Simulation during 500 days

Figure 18: Impact of total days of simulation



(a) Population of 60 and 6 thousand

(b) Population of 3 million

Figure 19: Impact of population size

4.2 Number of days

As you can see in Figure 18, the number of simulation days clearly play a role in the distribution of the weight of the program, and in the total runtime. It is no big surprise that when the total days of simulation is higher, more time is spent simulating.

4.3 Population size

For this parameter, we tested the simulator with very small, and much larger populations. The result of varying this parameter is according to our expectations. The larger our population, the higher our total runtime, as can be seen in Figure 19.

4.4 Immunity rate

Varying the immunity rate doesn't affect the simulator that much, as no great differences can be observed when varying this parameter. The results of our runs can be found in Figure 20

4.5 Seeding rate

According to our results, visible in Figure 21, the seeding rate does affect the total time of simulations. A higher seeding rate, will result in longer simulation times.

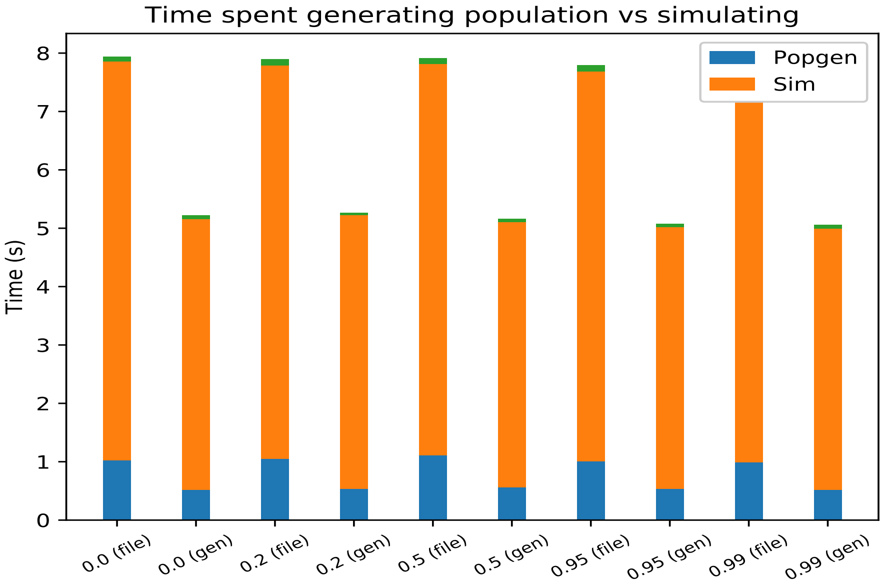


Figure 20: Analysis of the impact of the immunity rate.

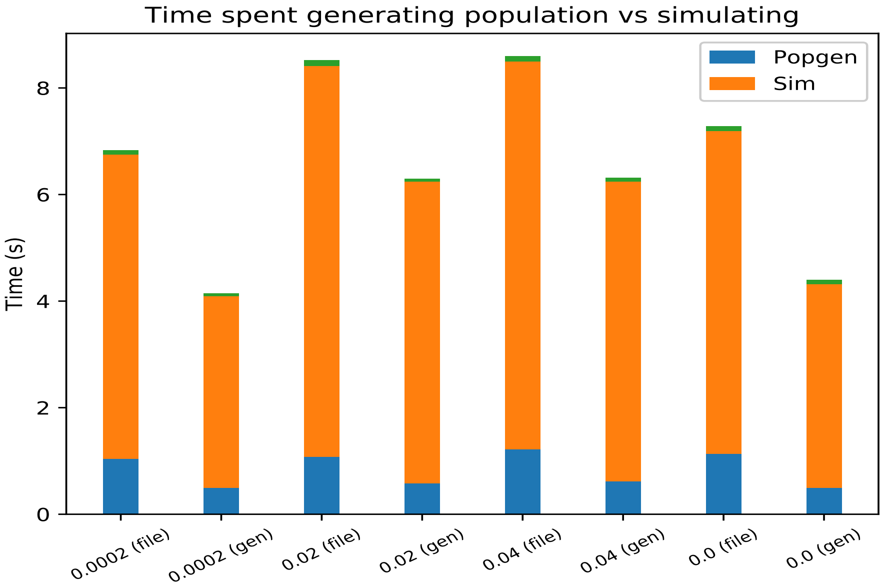


Figure 21: Analysis of the impact of the seeding rate.

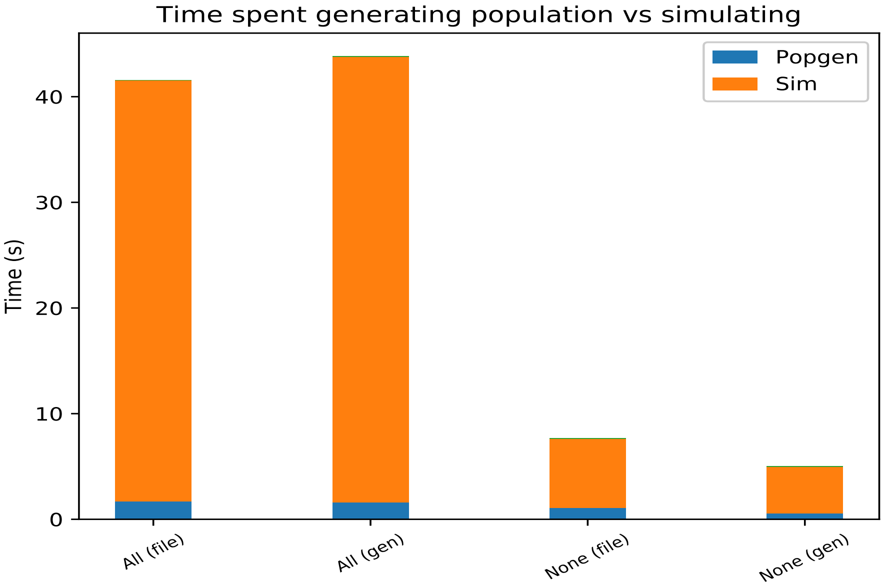


Figure 22: Analysis of the impact of the stride log mode.

4.6 Contact log mode

Out of the data from Figure 22, we can see that the log mode definitely has an impact on the total runtime of the simulation.
There is little difference between the default log mode (Transmission) and no log mode (None). On the other hand, it is obvious that when you log all contacts, the simulation is slowed down enormously.

4.7 Conclusion

One must be careful when altering the different parameters for simulations in stride. While some parameters have little impact on the total runtime or distribution of time of the program, others can greatly impact both.

Conclusion: when using extreme values or intensive log modes, one must accept that simulation times can skyrocket.

5 Discussion

Solving these assignments has learned us a lot about stride. Apart from the fact that we now can properly use the different components in the stride project, we also learned about the inner workings of stride.

The main thing to remember for the future is to check the small details. It has become clear that, for most cases, small changes or errors can greatly impact the results of the simulator. That is why it is important to verify our source data, to ensure that stride will be correctly simulating the outside world.

On the other hand, we also observed that our results are not always in compliance with our hypotheses. E.g. when checking if the work commuters contribute to the spread of diseases, we found results that contradict our intuitive thoughts. This does not always mean that we have introduced an error into our source data, but it can just be the result of a normal simulation. It is important that we examine both possibilities before rejecting a hypothesis.

References

[1] Kuylen, E.: Social Contact Patterns in an Individual-based Simulator for the Transmission Infectious Diseases. ScienceDirect (2017)

[2] BA Project Simulation Assignments (2019)