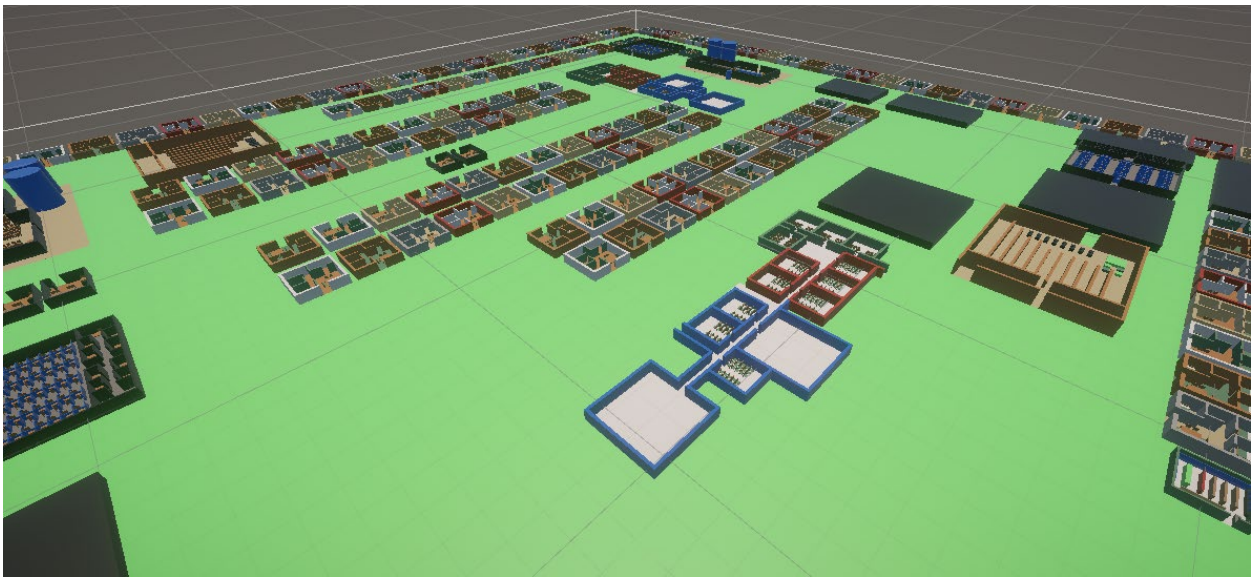


The Influence of Reduction Measures in Mitigating the Spread of Disease in an Agent-Based Artificial Society Simulation Model

Steven Chan

James Langevin

University of Victoria

CSC 446 – Operations Research: Simulation

Dr. Kui Wu

April 11, 2023

Table of Contents

Abstract

Compartment models are the traditional method used in epidemiology to study and predict the spread of infectious diseases. While such methods can roughly estimate the rates of infection and recovery, it does have numerous flaws, specifically with regards to how it treats interactions between members of the population. With the advent of increasingly powerful computers, it has become feasible to instead model the spread of diseases via agent-based simulation methods and artificial society constructions. With the proper parameters, such methods can simulate human interaction and contact, and thus simulate the spread of disease by taking in human factors into account. This paper details the construction of such a simulation, and the results thereof.

Project Description

An epidemic is an event in which a disease rapidly spreads throughout a population. Throughout mankind's vast history, disease has been one of the leading causes of death. The infamous Black Death (the bubonic plague) was reported to have killed between 30 and 50% of Europe's population when it spread through in the 14th century, claiming the lives of 75 to 200 million people (Shipman, 2014). Even today, in less developed parts of the world, diseases kill millions annually. And of course, we will be dealing with the aftermath of the COVID-19 pandemic that shuttered the world in 2020 and 2021 for decades to come.

We have always tried to understand the world around us, especially things that can harm us. Epidemiology is the study of the spread of diseases within populations. One of the most popular techniques for modeling the spread of a disease is via the use of compartment models, paired with mathematical models that use differential equations to determine the flow of people between each compartment (Beckley, et al., 2013). These equations often take the form:

$$\begin{aligned} \frac{dS}{dt} &= -\alpha SI \\ \frac{dI}{dt} &= \alpha SI - \beta I \\ \frac{dR}{dt} &= \beta I \end{aligned}$$

Figure 1: Differential equations for the SIR model. (Beckley, et al., 2013)

These equations, while popular, have some disadvantages. They assume that a population is totally homogenous: all agents are fungible and interact with all other agents with equal probability. This assumption does not hold up in the real world. People have friends and family, lovers and confidants, colleagues and classmates. They are more likely to interact with those physically and emotionally close to them

than strangers and people across the city. People are at specific locations for a reason; they do not exist as a cloud of probability in an environment. Today, with computer-aided modeling, it is possible to simulate all of these parameters and variables in software, building a virtual society in which agents live and interact. This is the premise behind agent-based simulation and artificial societies. By simulating agents, locations, and the interaction between them, it is possible to better represent the myriad aspects of the human condition than a set of differential equations.

A simulation allows us to investigate how effective a specific measure is in reducing the spread of a disease without having to put real populations at risk. In this study, we build an agent-based artificial society, in which agents simulate tasks such as going to school or work, interacting with colleagues and classmates, eating and talking with family, and sleeping. We create sufficient housing, working, and schooling facilities to support a population of 2,000 people. We introduce infectious agents into the simulated environment with simulation parameters that aim to accurately reflect a real, generally non-lethal respiratory virus (such as the flu or common cold) and investigate how well three different types of mask policy work in combating the spread of this disease. We show the fallacy of partial reduction measures and how full compliance with reduction measures is necessary in fully stamping out a disease before the entire population becomes infected.

We will investigate the following policies:

1. A population of that does not wear masks.
2. A population that will wear masks upon showing symptoms.
3. A population that will conform to wearing masks from an increasing mask compliance level.

Simulation Model

In contrast with traditional compartment models, our simulation models are based on *Agents*, each of which represent an individual in their environment, or *society*. Our model makes use of the usual susceptible-infectious-recovered (SIR) compartments. Each agent can be in one of three states:

- Susceptible agents are individuals within a society who have not yet been infected with the simulated disease and may become infected via prolonged exposure to infected individuals.
- Infectious individuals are agents in their society who have become afflicted with the simulated disease. These agents may start exhibiting symptoms within a few days after they become infectious, but regardless, they can still transmit the disease even before they become symptomatic.
- Recovered individuals are those who have fought off the disease. They are immune to further infections for the duration of the simulation and will not transmit the disease to susceptible agents.

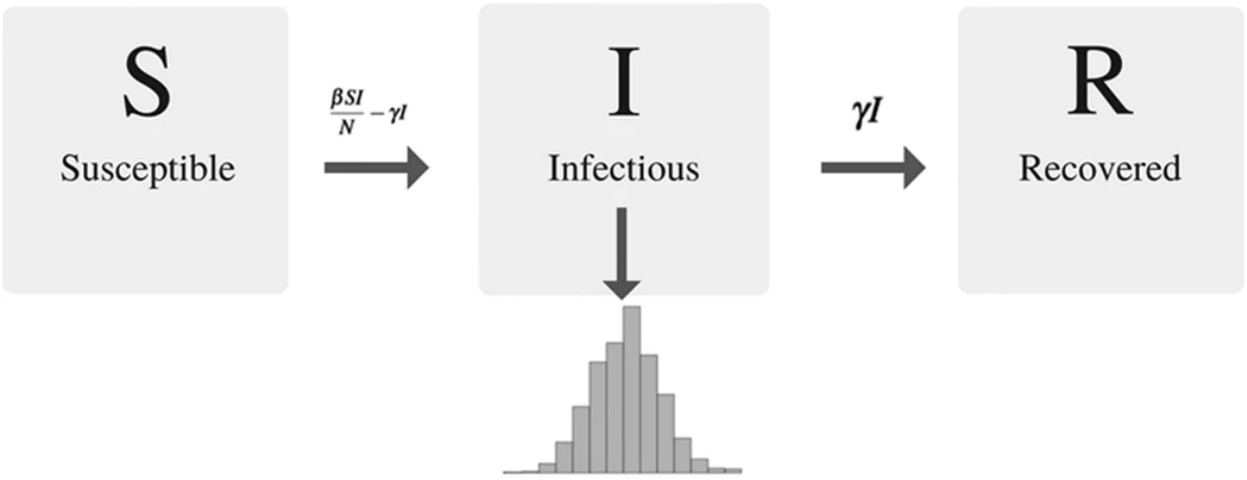


Figure 2: A depiction of the traditional susceptible-infectious-recovered compartment model (Daughton, Generous, Priedhorsky, & Deshpande, 2017). While our simulation uses an agent-based artificial society approach rather than the more common approach of mathematical modeling via differential equations, we kept the stages the same because it generalizes quite well to many illnesses in the short term.

The simulated disease does not correspond to any specific real-world disease, but the modeling is designed to approximate a generally non-lethal respiratory virus such as a common cold or a seasonal flu. Various parameters, such as transmissibility and range falloff, are simulation parameters that we will discuss in the Simulation Parameters section. The simulated disease does not kill any agents. Given that no agents are born for the duration of the simulation, nor do any agents enter or exit the society, the simulation is essentially a closed environment, which is a typical setup for this type of experiment.

Agents interact with each other and their environment. This is the driving force for the spread of the disease. Agents will go to sleep, eat, hang out with family, go to school, or work, and interact with nearby agents while at these activities. Between activities, agents will walk, via pathfinding, from one location to another, providing another avenue of disease transmission. Agents will take measures, such as wearing masks, to avoid becoming infected by the disease. Some agents, as in the real world, will refuse to comply with such measures for political or ideological reasons, and our simulation will explore how well compliance with such reduction measures will influence the spread of disease in a small town.

Simulation Parameters

Our simulation model exposes a wide variety of parameters that can be adjusted to create different scenarios. Some parameters are numeric or Boolean in type, while others are created via a combination of linear and cubic Hermite splines and then sampled to obtain a value.

Parameter	Value
Infection Range Curve	(curve)
Transmission Curve	(curve)
Infection Range Curve (Masked)	(curve)
Min Infection Points	80
Max Infection Points	120
Min Symptom Time	1
Max Symptom Time	2.5
Min Recovery Time	3
Max Recovery Time	6
Falloff	40
Initial Infected Agents	1
Mask Compliance Level	30
Maskless	(Scenario dependent)
Sick Mask	(Scenario dependent)
All Mask	(Scenario dependent)

Figure 3: Table of the value-adjustable parameters for the simulation. This does not include parameters like environment layouts. See below for more information.

We selected these specific values as a compromise between simulation speed and realism. While none of the values are meant to represent any specific respiratory virus, most of the values are like those of the common cold. The values are:

- Infection Range Curve: The infection range curve is a function defined by a cubic Hermite spline, defined on the interval $[0, 6]$ and maps the distance to an infected agent with the number of infection points conferred to the susceptible agent. This value is then multiplied by the Transmission Curve, which acts as a multiplier for the infectiousness of the virus over time. While it's impossible to fully map out how range influences transmission due to, among other factors, ethical concerns, the Center for Disease Control and Prevention (CDC) believes that the common flu can easily spread via aerosol droplets out to two meters (Center for Disease Control and Prevention, 2022). However, some particles could reach as far as seven meters (Setti, et al., 2020). We interpolated these values such that it falls off relatively rapidly to about two meters, then slowly decays to zero at six meters. A value of 100 (at zero meters) means that extremely close contact with an infected individual will cause certain infection within about an hour.

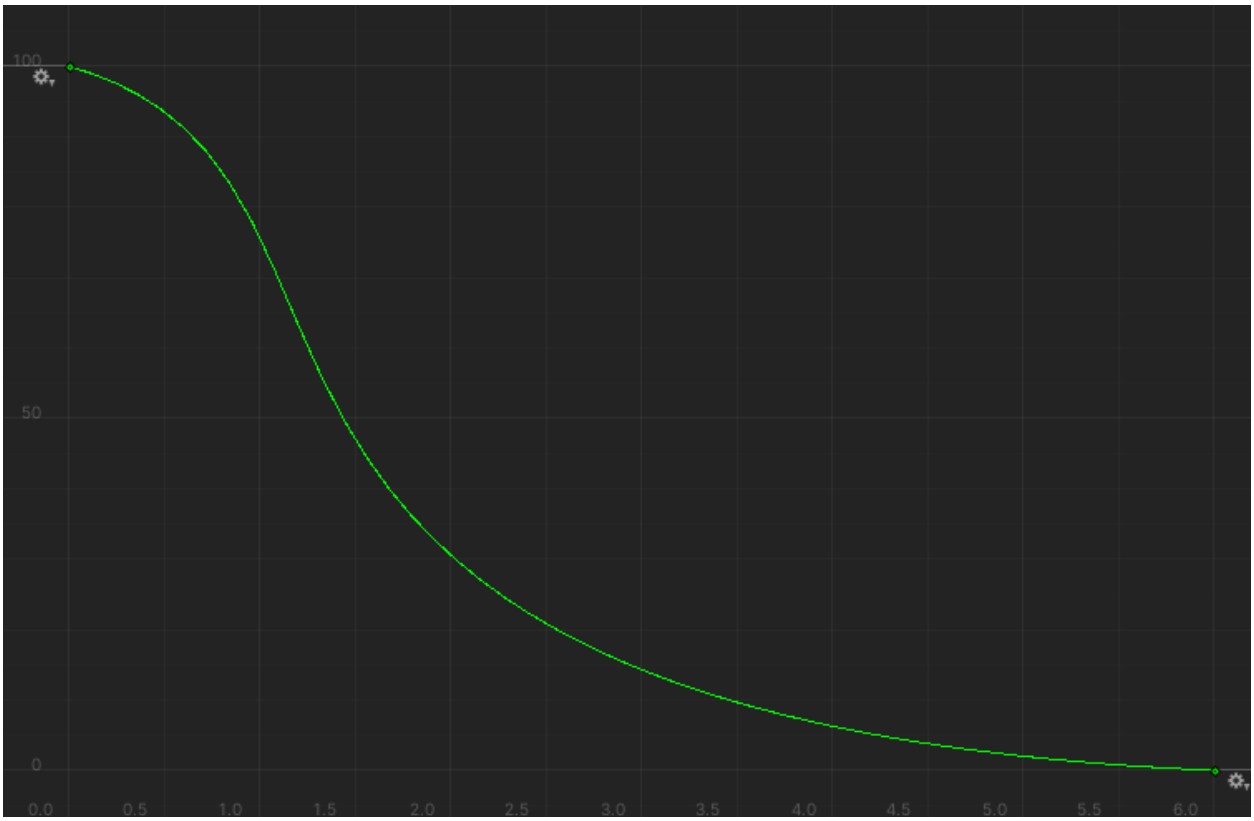


Figure 4: Infection range curve, mapping distance (in meters) to infection points gain per hour. Agents need 80 to 120 points to become infected, simulating the diversity of immune systems and other stochastic factors.

- Transmission Curve: The transmission curve maps a multiplier to the infection range curve with respect to the time since the agent has been infected. Viruses take time to multiply in a host before becoming fully infectious, and its ability to infect others falls off as the immune system fights off the virus. In the simulation, this effect is captured by the transmission curve. The CDC states that the odds of transmission are greatest when a person becomes symptomatic and falls off from there (Center for Disease Control and Prevention, 2022). The H1N1 influenza, as well as possibly other viruses, have sample population infection rates that adhere to a Weibull distribution (Duan, et al., 2013); we modeled the transmission curve after this data.

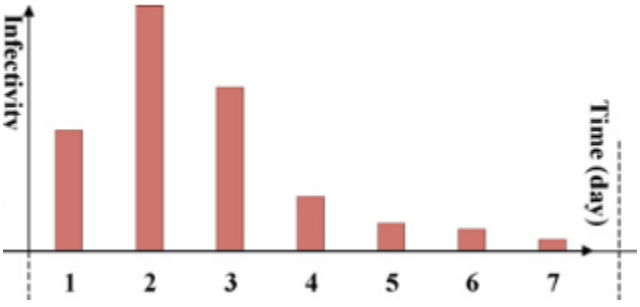


Figure 5: Duan et al.'s findings on infectivity of H1N1 over time

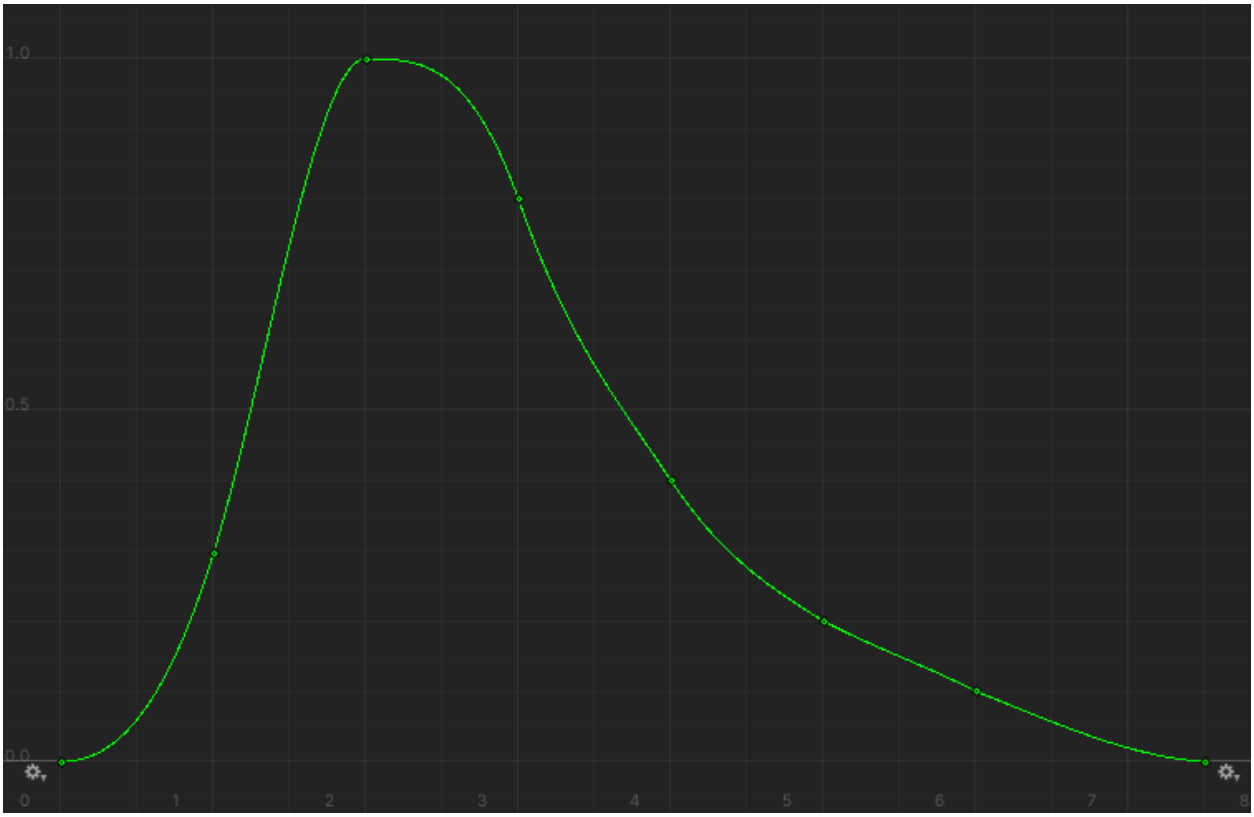


Figure 6: The transmission curve used in the simulation. This correlates roughly with Duan et al.'s chart.

- Infection Range Curve (Masked): If an agent decides to don a mask in order to reduce the risk of their being infected, then this curve will be sampled instead of the regular infection range curve. This curve is created by modifying the original infection range curve such that the infection points conferred is roughly 3.61% to 4% of the original curve. These values are based on the average percentage of droplets of airborne sneeze particles that pass through a two-layered cloth mask (Akhtar, et al., 2020).

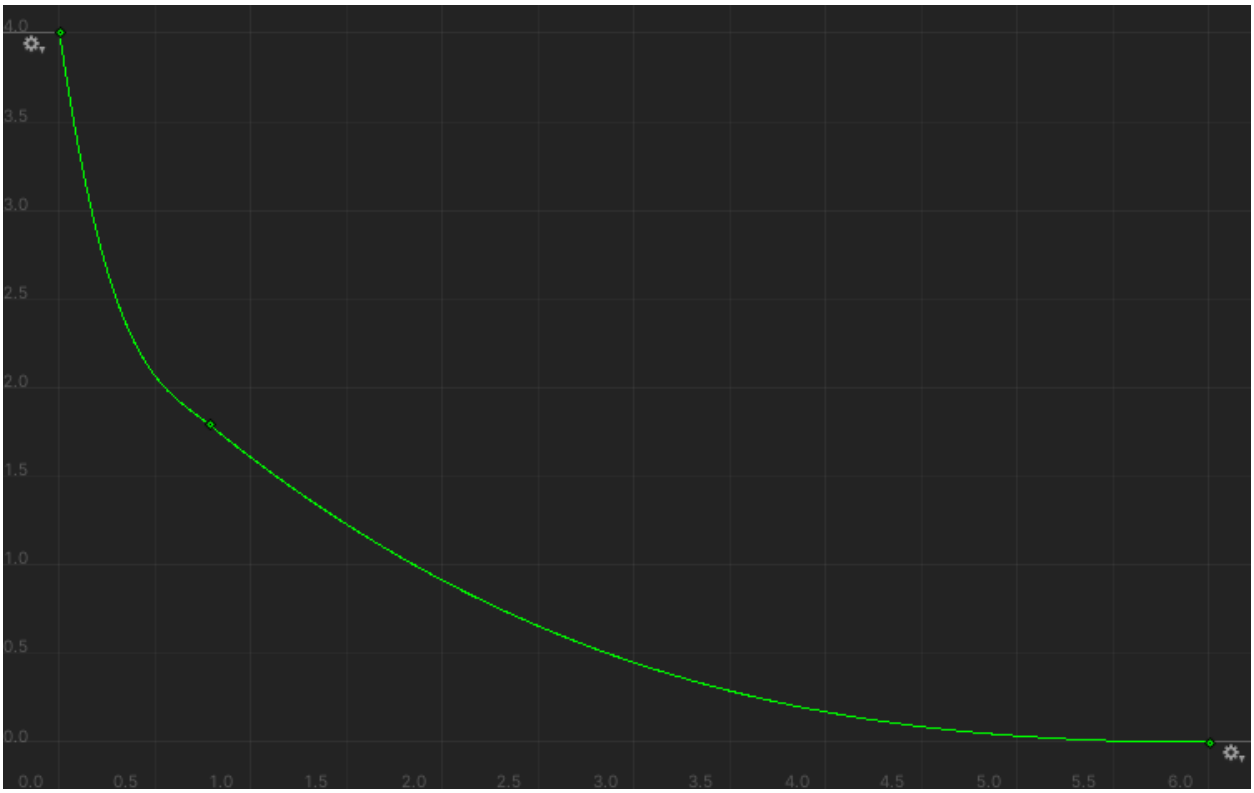


Figure 7: The infection range curve used for agents who wear a mask. While the shape is fairly similar to the original, the scale has been reduced to about 4% of the original.

- Min and Max Infection Points: To handle various factors such as the fact that multiple infected agents nearby will produce more viral particles, we opted to use a points system rather than a probability model as used by some other simulations like Duan et al.'s. With this model, agents are given a random points threshold, uniformly generated between the ranges specified in the min and max infection points parameter. If the agent exceeds this points threshold, it becomes infected. With the aforementioned infection range and transmission curve, close exposure to a fully infected agent will cause a new infection in under one hour of contact. Agents require 80 to 120 points to become infected; the threshold is generated on simulation startup.
- Min and Max Symptom Time: Not everybody exhibits symptoms at the same time. According to the Mayo Clinic, it takes roughly one to three days from exposure to show symptoms (Mayo Clinic, 2021). We use 1 to 2.5 days for this parameter, shaving half a day off the tail to speed up the simulation slightly without sacrificing accuracy.
- Min and Max Recovery Time: Just as there is variation in the time from infection to symptoms, there is a variation in the time it takes for an infection to clear up. This is because factors, such as immune system strength, rest, and health all play significant factors. It takes about 3 to 7 days for a cold to resolve itself (DerSarkissian, 2022). We chose to use 3 to 6 days as an optimization measure: according to Duan et al., by the seventh day, the infectiousness drops of to under 5%, and we consider it highly unlikely an agent will be able to confer an illness at such a low ratio.

- Falloff: Below a certain threshold, the immune system can fight off the virus before it becomes capable of infecting an agent. Thus, numerous short bursts of contact will have a lower chance of infecting an agent. In the simulation, this is modeled by the falloff parameter, which subtracts this many points from the infection points threshold (per hour) if no infected agents are within six meters of the agent. With a value of 40, a night's rest is sufficient to reset the points, as long the agent did not become infected.
- Initial Infected Agents: This sets the number of infected agents at the beginning of the simulation. In the case of novel viruses, this is usually 1; a virus is introduced into a population from one person, usually dubbed "patient zero." We chose to use a value of 1 for this purpose, but other values would have also sufficed, with likely steeper infection curves.
- Maskless: If this option is selected, agents will never wear a mask.
- Sick Mask: If this option is selected, agents who become symptomatic will wear a mask.
- All Mask: If this option is selected, agents who become symptomatic will wear a mask. Additionally, all agents have a chance to decide to wear a mask, dependent on their inherent tolerance for risk and the amount of time that has passed since the infections began.

Values have been chosen to be mimic those of real-world parameters, in order to better closely simulate the spread of diseases in human communities.

Additionally, the scene itself can be freely modified to create different scenarios. For example, it is possible to add more agents to the scene by simply adding more home prefabs into the scene or add more of any number of types of shops, offices, or factories to provide for the town. In our simulation, we have 250 houses. With a capacity of 8 agents per household, we have 2,000 agents. According to Statistics Canada, about 21% of Canadians are under the age of 20. Rounding to the nearest eighth, we decided on having 2 out of 8 agents in each household be "children" for the purposes of this simulation, because children go to school, and adults work during the morning hours. To simplify the simulation, the schools are K-12, a relatively common practice in smaller communities.

Figure 8: Agents exist within their environment, called a society. Houses, stores, factories, and more can be added and moved around to create unique environments for the agents.

Methodology

The simulation is built on top of the Unity engine and was developed with Visual Studio 2022 and JetBrains Rider. The project consists of about two thousand lines of C# source code across thirty-seven files, as well as over forty other assets. An asset can be anything from an agent, a scenario configuration, a building, all the way to entire housing clusters. Some buildings, like schools, may contain many hundreds of objects like chairs and tables.

Initialization

When the simulation starts, control is passed to the Simulation Manager (in *SimulationManager.cs*). One of the many duties the manager handles is simulation startup. On initialization, the manager will search the scene (the environment the agents will interact within), cataloguing houses, schools, and workplaces found. For each house, the manager will request a list of beds in that house, and for each bed, an agent will be assigned to it. The agent will spawn, initialized with data such as its bed and house. If the agent is an adult, then the manager will find a workplace and randomly assign a job to the agent, as long as there is room at that workplace. If the agent is a child, then an algorithm is run to assign a school and create a school schedule for that child. This algorithm takes room occupancy into account, so no student will have time conflicts, and classroom capacities will be respected. Despite the complexity, this mechanism improves realism, and provides opportunities for agents to move around, interact, and infect other agents. The manager is also the primary access point of the Scenario object, a data structure that keeps track of many of the simulation parameters that make up unique scenario. This allows new

scenarios to be created and swapped out with ease. After work or school assignments have been laid out, the manager will randomly infect an agent (or more, depending on the value of the initial infected agents parameter), and the simulation is allowed to proceed.

Day-to-Day

Agents follow a schedule with regards to what task they are accomplishing, depending on the time of day. The Time Controller (*TimeController.cs*) is an object that manages the concept of time within the simulation, as well as allowing us to modify the time scale. Agents wake up at around 6:00 and have breakfast and handle household tasks until about 8:30. At 8:30, children go to school and adults to their place of employment. For adults, they will work in their office or factory line until about 15:00, when they get off. Children have four classes to attend according to their schedule. Their schedules are:

Block	Time
1	8:30 to 10:30
2	10:30 to 12:00
3	12:00 to 13:30
4	13:30 to 15:00

After 15:00, children will also head home. Agents will tend to the home, as well as have dinner, until 22:00, at which point they will head to bed. The cycle then repeats. At each point, agents can have contact with their families, as well as colleagues and classmates/friends, providing vectors for infection via close contact. As their day-to-day tasks are finished, various systems operate in the background, creating new tasks, allowing the agent to navigate the world and interact, simulating infections via the many parameters available to it, and logging the state of the simulation.

Navigation System

Many simulations that use mathematical methods lack any notion of space, and some, such as Duan et al.'s experiment, simply have agents instantly transport themselves from one place to another. Our model implements a navigation system, as do those by some other researchers in agent-based transportation models (Hackl & Dubernet, 2019). Unity provides a mechanism for generating *navigation meshes*, a data structure that holds polygons representing terrain that agents can navigate within. The mesh is generated based on the geometry of the scene. Our simulation runs a custom navigation component, using the popular A* algorithm to generate paths to the goal, and a custom vector projection technique to ensure that the agents do not overshoot their targets. For improved realism, most objects in the simulation have similar scales to those of real-world equivalents: one meter is one meter, and one second is one second. To this end, agents move at a speed comparable to that of a human at a normal pace of 1.42 meters per second (Browning, Baker, Herron, & Kram, 2006).

Infection System

Each agent has a component responsible for handling infections regarding itself. When the simulation is initialized and the agent is spawned, the infection point threshold, time to show symptoms, and time to recover are computed based on the simulation parameters. Agents will also have a 10% chance of absolutely refusing to comply with all mask mandates. The system uses sphere colliders to efficiently determine when two agents are within infection range of one another, and are added to a list when they are within range (and removed when out of range). Roughly every three minutes (in simulation time), an update cycle is run to determine the next state of the system. The agent will check if its current number of infection points exceeds the threshold. If so, the agent will be marked as infected, and the time of infection is logged. Alternatively, if an agent is infectious and the recovery time has elapsed, then the agent will be marked as recovered.

After that, the system will check whether the agent should wear a mask. How this works depends on the mask behaviour system parameter:

- If the parameter is set to no mask, the agent will never wear a mask.
- If the parameter is set so that infectious agents wear masks, then if an agent is symptomatic and does not refuse mask compliance, they will wear a mask upon becoming symptomatic.
- If the parameter is set so that all agents wear masks, then the above behaviour is executed. Additionally, susceptible agents may also wear a mask if they do not refuse mask compliance. Their odds of doing so is modulated by time: the longer the simulation runs and the more agents fall sick, the greater the odds of them wearing one is. Additionally, each agent will also have a risk factor; some agents will only wear a mask when the infection rates are very high, others will proactively protect themselves.

If an agent is infectious, then it will check all nearby susceptible agents. If there is line-of-sight to the agent, then the infection range and transmission curves are evaluated, and points are added to each nearby susceptible agent.

Finally, if no agents are nearby, and the agent is not infected, then it will subtract infection points based on the falloff system parameter.

The interactions between this infection system and the infection systems of other agents are how the simulated disease spreads through the population. Given the sheer number of agents and the cost of detecting nearby agents, caching and fudging factors are used to help accelerate the program. We have not observed any noticeable behavioural changes arising from these optimizations.

Task System

Agents in the simulation are driven by tasks. It is the force behind agent interactions with each other and the environment and provides the impetus for the viral spread among the population. Agents have a task queue, and agents will execute tasks according to a FIFO discipline. As seen in queuing simulations, additional tasks are generated as current ones are finished. If an agent finishes all of its tasks, it will raise

an event that will force the Task Maker (*TaskMaker.cs*) to generate a new task. The task generated is dependent on the simulation's time of day – see the Day-to-Day section for more details. Each task consists of a series of task instructions. A task is considered completed when all of its instructions are completed. Task instructions include:

- Go To: This task takes a point in Euclidean coordinates. When executed, the instruction instructs the agent's navigation system to navigate to those coordinates and waits until the agent is at that position before advancing to the next instruction.
- Wait: This task takes a time, in seconds. When executed, the task will halt further task execution for the specified number of seconds.
- Wait Until: This task takes a date and time (in simulation time). When executed, the task will halt further task execution until the specified date and time has elapsed.
- Release Claim: This task takes a claim and releases it. This will be explained in a subsection.

When the task is complete, the agent will pull the next task in the queue, or request a task from the Task Maker, if necessary.

Claims System

To ensure two agents don't attempt to claim the same work spot, classroom chair, or seat at the dinner table, these objects are marked as *claimable*. When a task is given to an agent and added to its queue, the task contains a list of claimable objects that will be claimed as long as the task exists in the agent's queue. Claimed objects cannot be claimed by others, and the Task Maker will ensure that a generated task does not impose on another agent's claims. When an agent completes the task, all claims it made for that task are unclaimed, freeing it for other agents to use. Sometimes, such as in the case of class rotations, it may be wise to unclaim seats before class ends, so that next class's students can be guaranteed to have seats (otherwise, the next class may fail to find enough seats because the last class has not left yet). In that case, a Release Claim task instruction can be used to release a specific claim before the task finishes.

Simulation Setup

Throughout the process of building the simulator, various mechanisms have been incorporated to help debug and ensure that the system performs as we expect. Custom code has been written to extend the Unity editor, providing the ability to manually infect agents, dumping logs to disk, draw agent pathfinding, writing task queues to console, and debug toggles for enabling breakpoints at critical code sections.

For the data collection runs, we set the parameters as described in the Simulation Parameters section. Because it is not possible to ask a virus to modify its own parameters to better benefit humans, the only real practical means of defeating epidemics is human intervention. As such, the parameters that constitute different scenarios is the mask behaviour. In real societies, this would be implemented by

public health agencies providing recommendations for when and how people should wear masks to prevent the spread of disease (as in the COVID-19 pandemic over the last three years). In the simulation, this is controlled via setting the mask behaviour between none, infectious individuals only, or infectious individuals plus time-varying susceptible individuals. For more details on these behaviours, please read the Infection System section. We conduct five runs of each type, for a total of fifteen runs. See the Results and Analysis section for more information.

Logging

The Epidemic Tracker (*EpidemicTracker.cs*) is responsible for periodically logging the number of susceptible, infected, and recovered agents in the simulation. In our simulation, it is configured to log ten times per simulation hour (once per six simulation minutes). When logging, the current simulation time is logged, as well as the number of each type of agent. This is packed into a record (*AgentRecord*) and stored as a list. At any point (such as when the simulation is finished), a key binding can be pressed to dump the contents to disk. When called, the records are exported to a comma-separated values (.csv) file, allowing further examination.

Results and Analysis

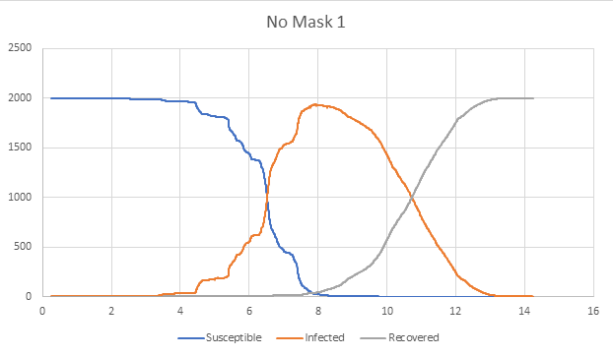
This section contains the graphs of each simulation with respect to time, for a simulated population of 2,000 agents. We did 3 separate tests with 5 different seeds for each test, for a total of 15 tests. These tests include:

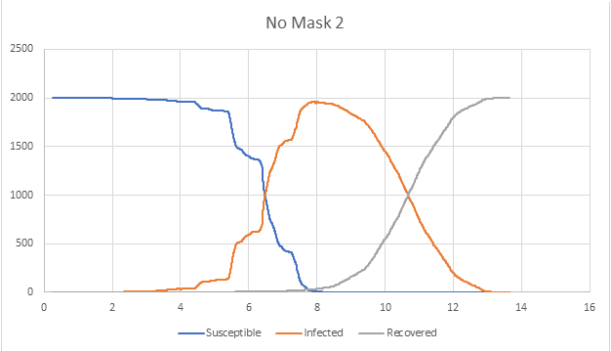
1. Agents who didn't wear masks.
2. Agents who wore masks after becoming symptomatic.
3. Agents who wore masks after becoming symptomatic or if they followed the mask mandate over time.

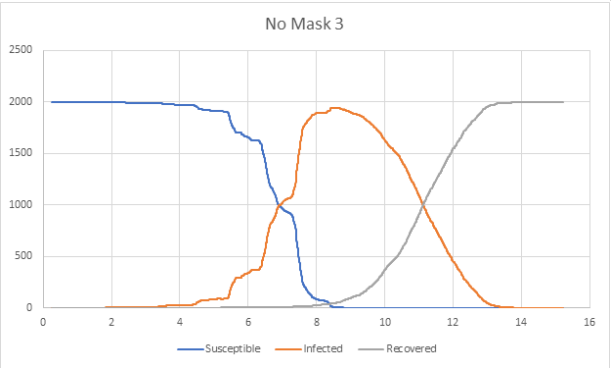
Note that this simulation was run in a town with workplaces and schools in which agents were in close contact for most of their daily lives.

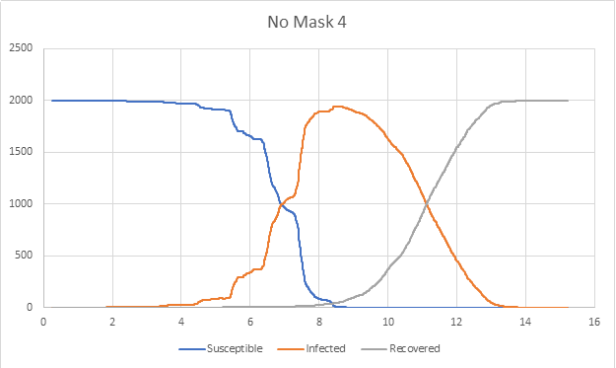
While the numbers of each test varied, they each followed the same general curve. However, there were cases where some agents would become infected faster due to the tasks that agent had and where that agent lived, thus reducing the chance of infecting other agents.

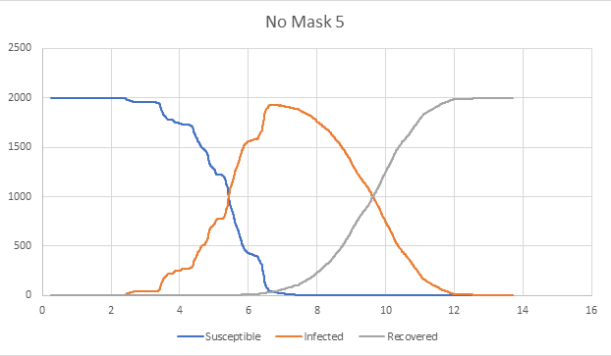
Below are the graphs of the simulations, in the order of No Mask, Sick Mask, then All Mask, matching the order of the tests above.

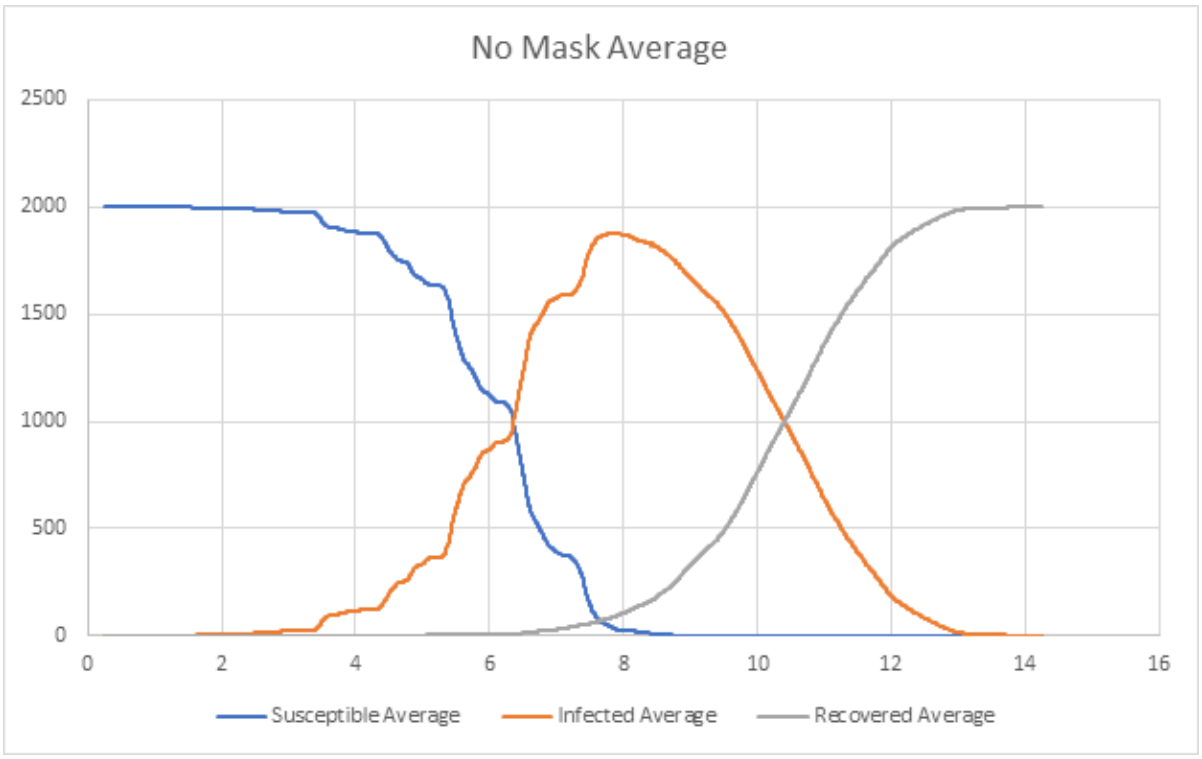

No Mask 1
2500
2000
1500
1000
500
0
0 2 4 6 8 10 12 14 16
Susceptible
Infected
Recovered


No Mask 2
2500
2000
1500
1000
500
0
0 2 4 6 8 10 12 14 16
Susceptible
Infected
Recovered


No Mask 3
2500
2000
1500
1000
500
0
0 2 4 6 8 10 12 14 16
Susceptible
Infected
Recovered

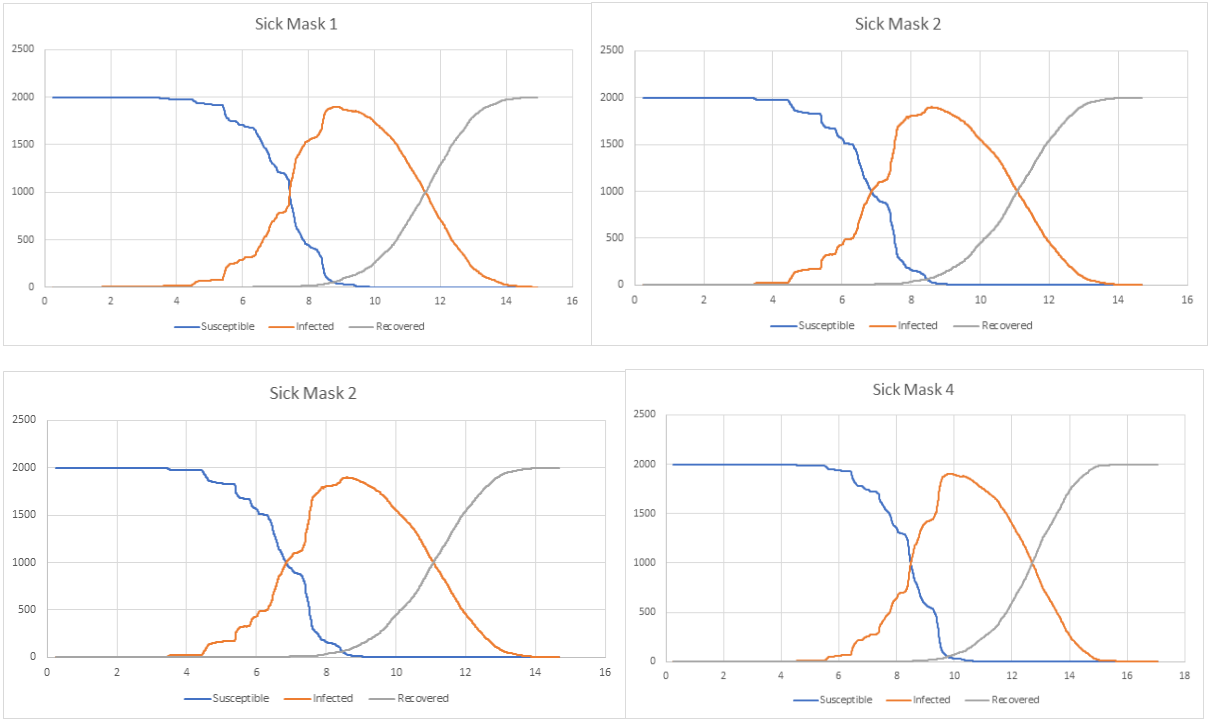

No Mask 4
2500
2000
1500
1000
500
0
0 2 4 6 8 10 12 14 16
Susceptible
Infected
Recovered

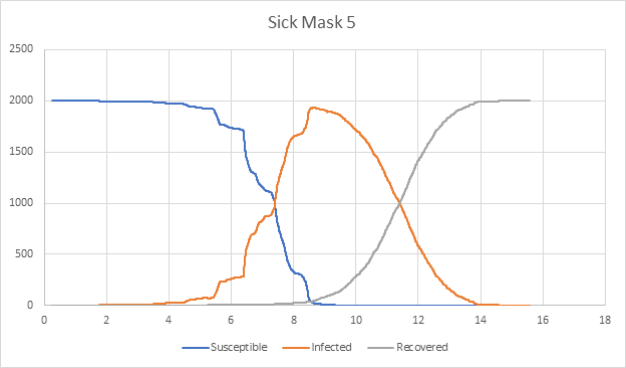

No Mask 5
2500
2000
1500
1000
500
0
0 2 4 6 8 10 12 14 16
Susceptible
Infected
Recovered

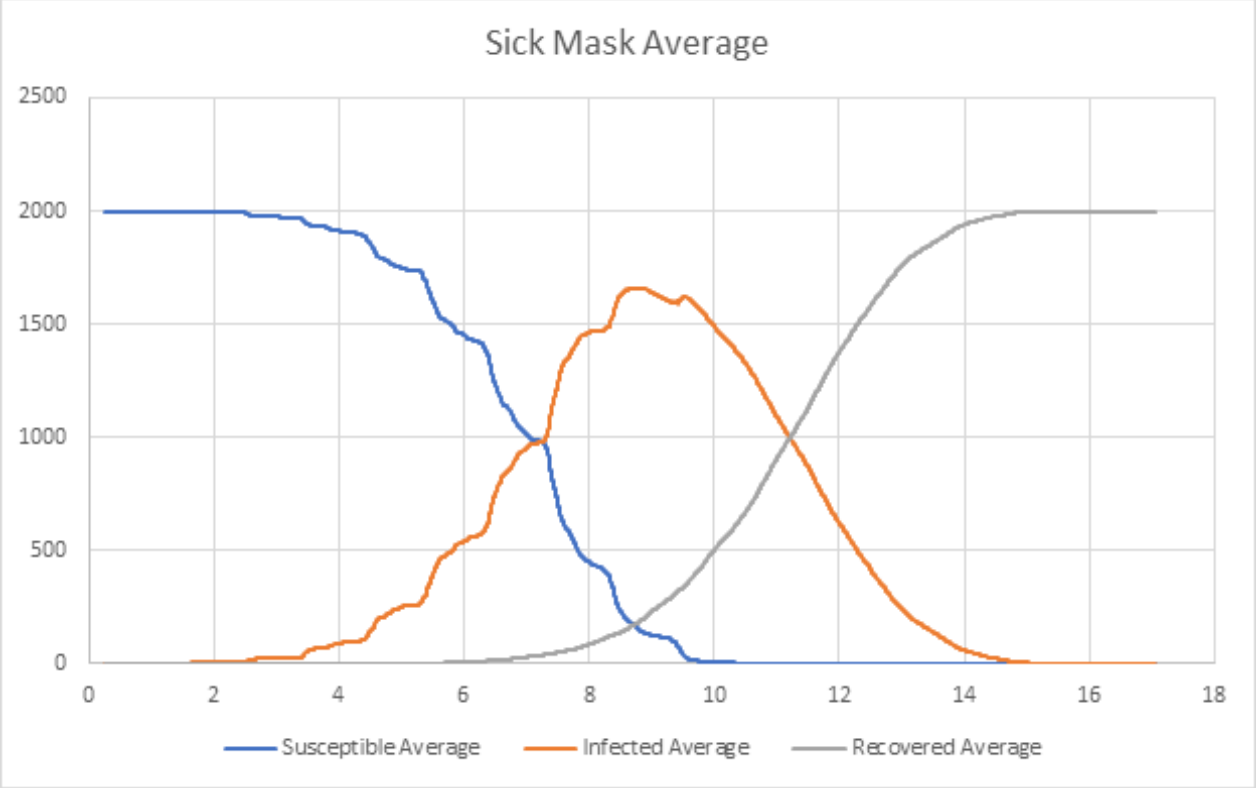


As seen from the results, when agents did not wear masks, the infection spread rapidly.

When running tests on a population that wore masks upon becoming symptomatic, we obtain the following results:

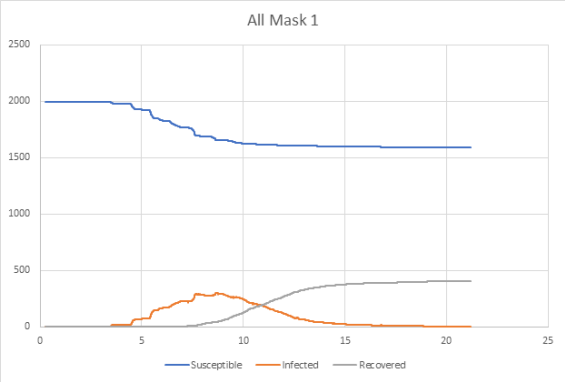


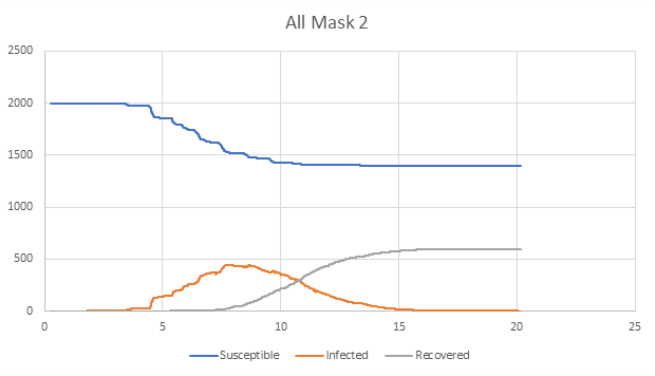




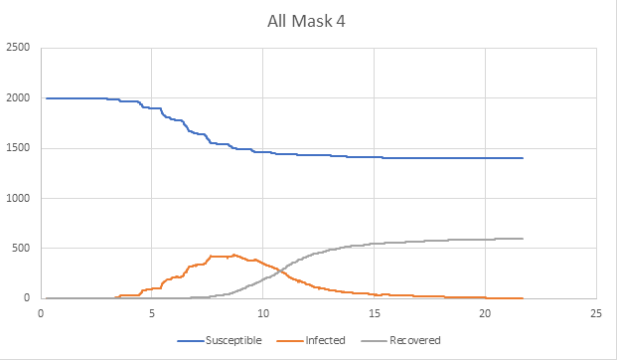
While the infection didn't spread as fast as when agents didn't wear masks, we can see that wearing a mask only when becoming symptomatic will not stop the spread of the infection, and only modestly impede the spread of the virus.

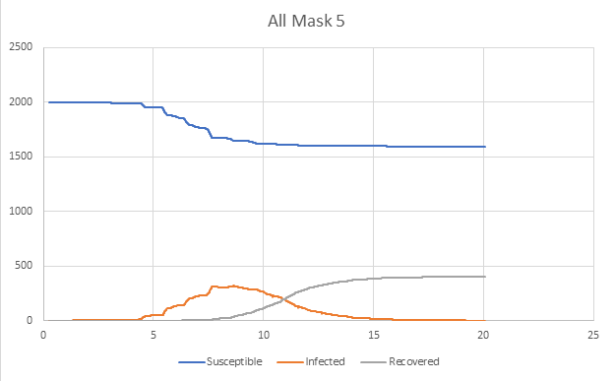
When we ran tests on when agents would wear masks according to an increasing mask compliance rate, we got the following results:

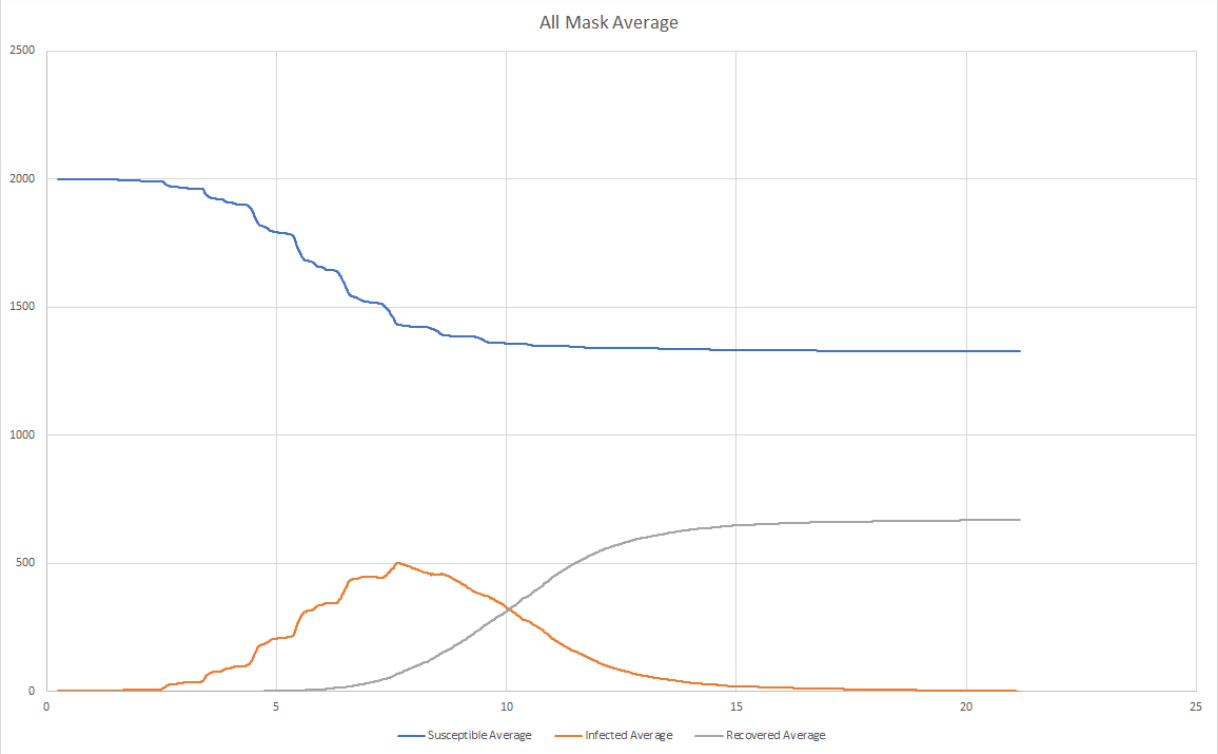

All Mask 1
2500
2000
1500
1000
500
0
0
5
10
15
20
25
Susceptible
Infected
Recovered


All Mask 2
2500
2000
1500
1000
500
0
0
5
10
15
20
25
Susceptible
Infected
Recovered


All Mask 3
2500
2000
1500
1000
500
0
0
2
4
6
8
10
12
14
16
Susceptible
Infected
Recovered


All Mask 4
2500
2000
1500
1000
500
0
0
5
10
15
20
25
Susceptible
Infected
Recovered

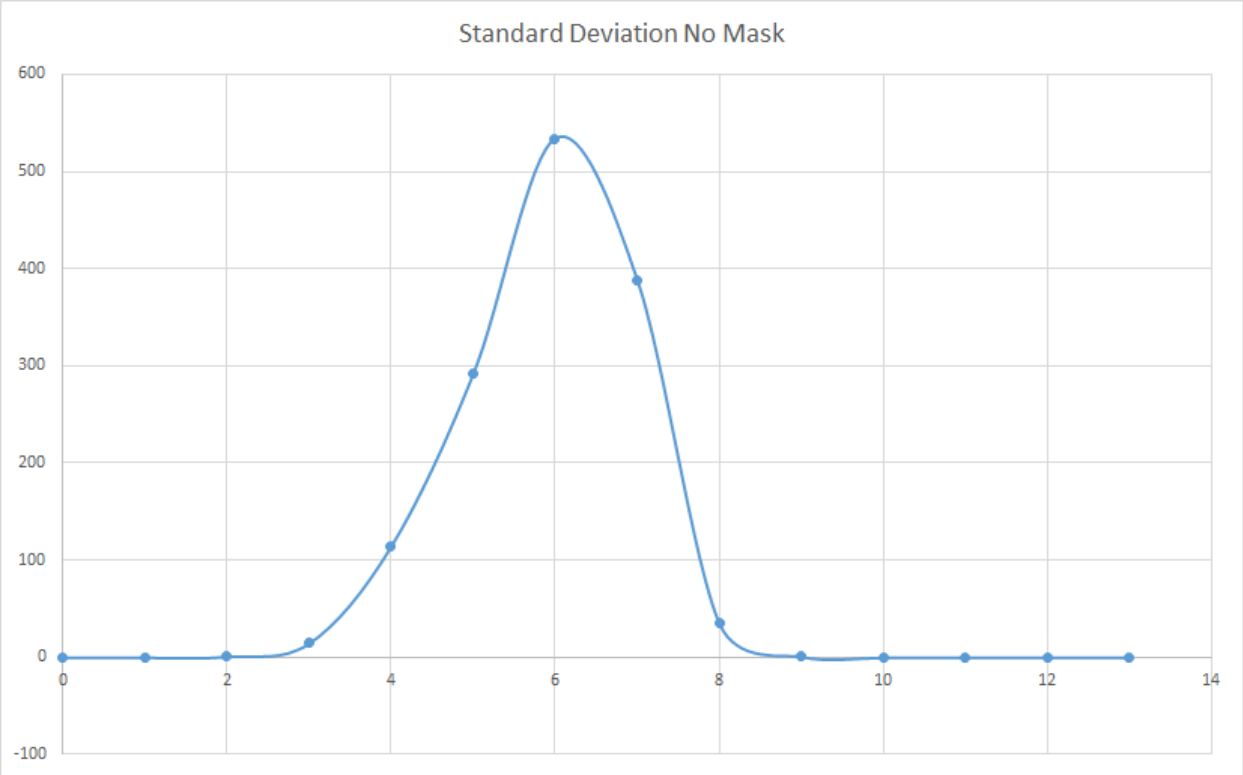

All Mask 5
2500
2000
1500
1000
500
0
0
5
10
15
20
25
Susceptible
Infected
Recovered

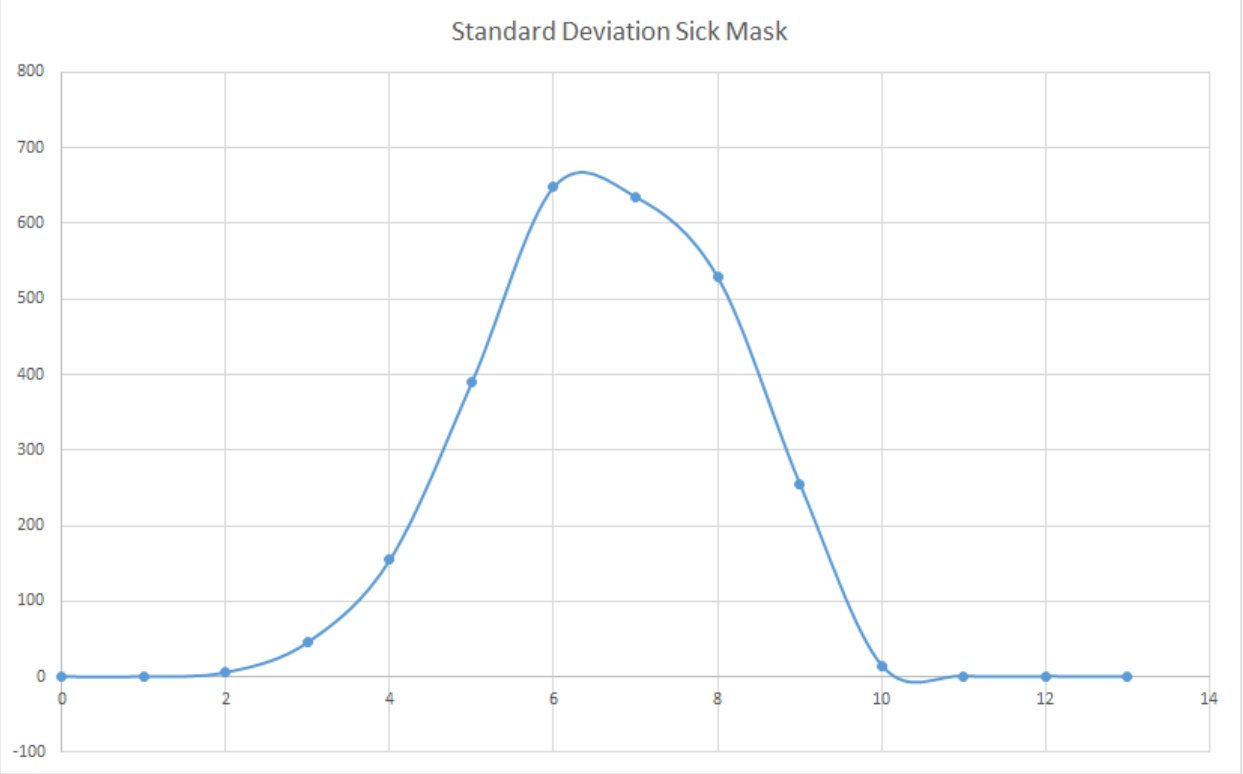


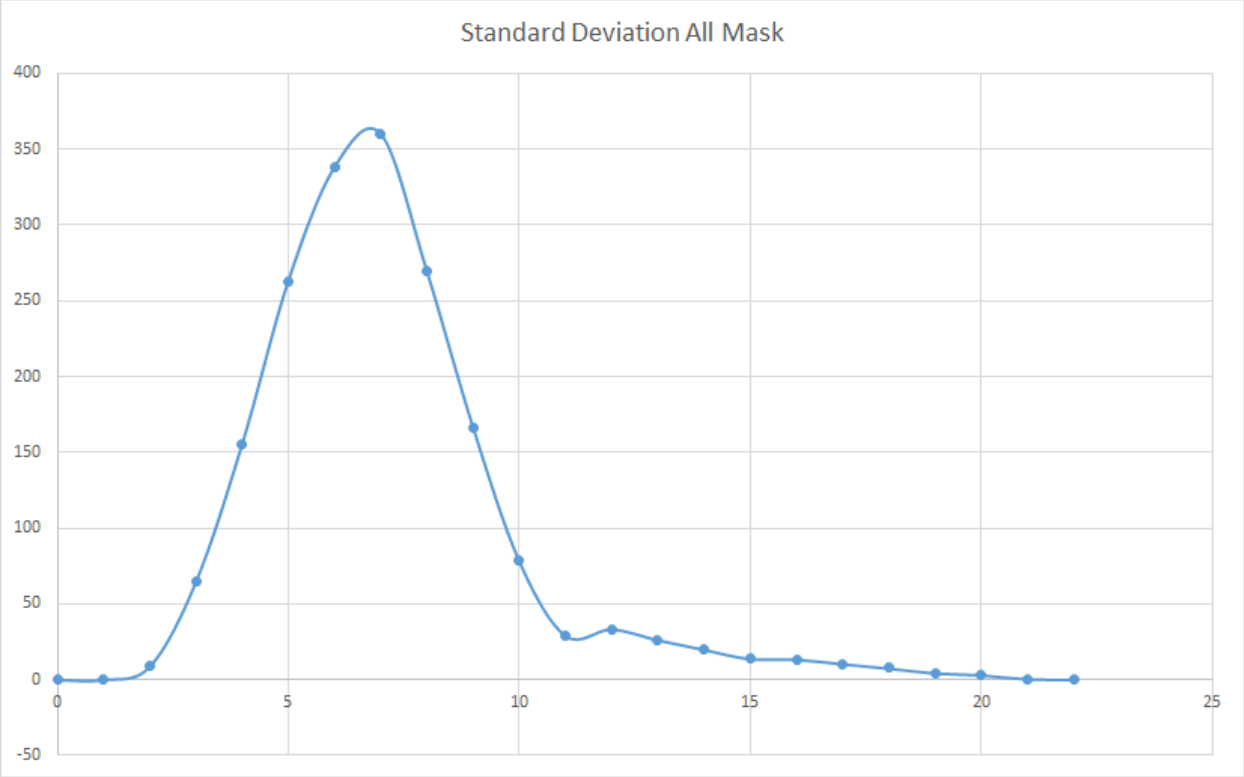
As can be seen from the results, the infection dies out after about 15 days, when roughly 90% of the agents are wearing masks. More significantly, a majority of the population remains uninfected by the end of the viral spread.

Standard Deviation

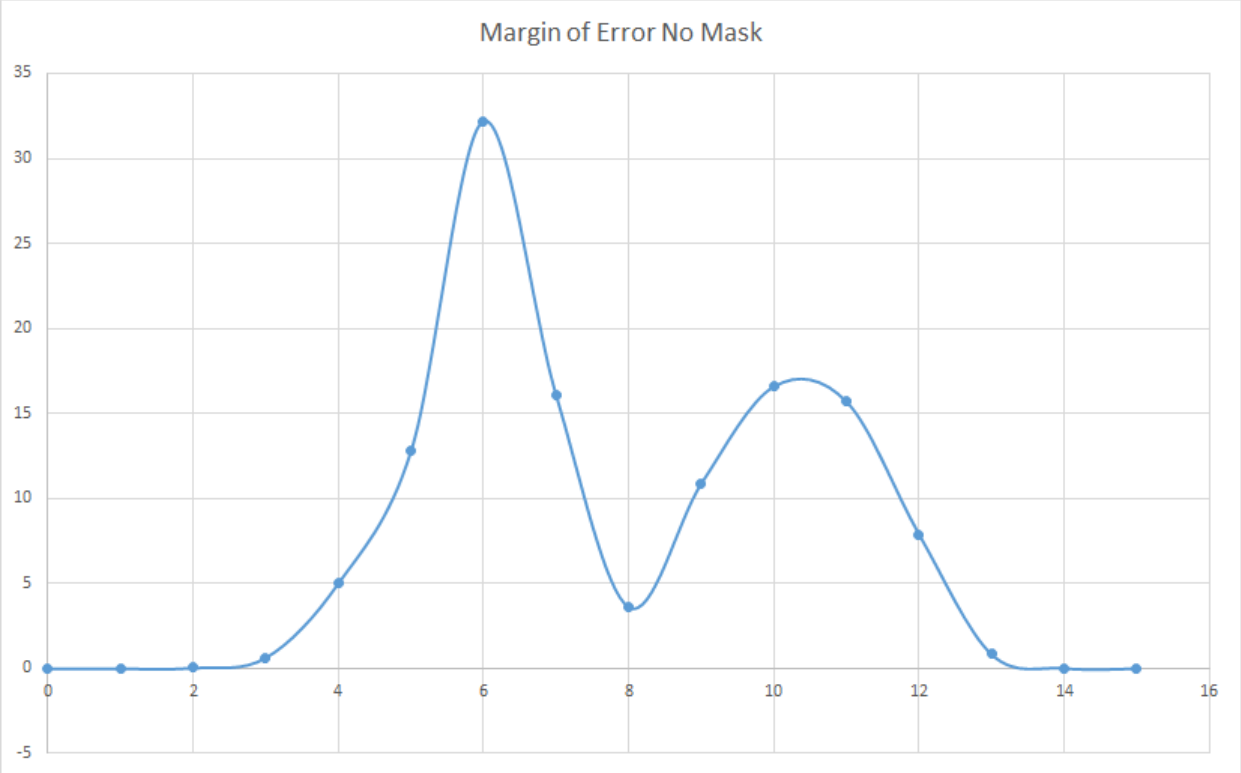
Standard deviation of those that are infected (Note that this goes down because people recover from illness, and there will eventually be 0 infected individuals).


Standard Deviation No Mask
600
500
400
300
200
100
0
-100
0
2
4
6
8
10
12
14


Standard Deviation Sick Mask
800
700
600
500
400
300
200
100
0
-100
0
2
4
6
8
10
12
14

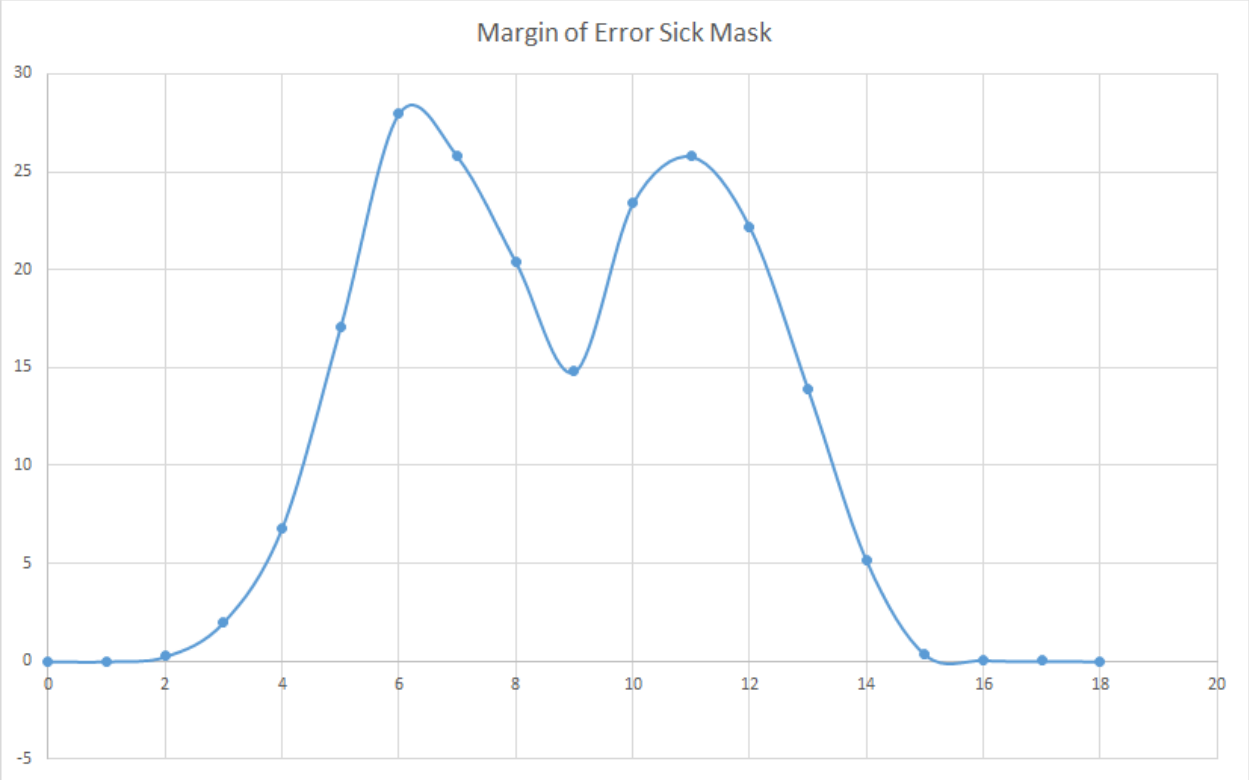


Margin of Error and Confidence



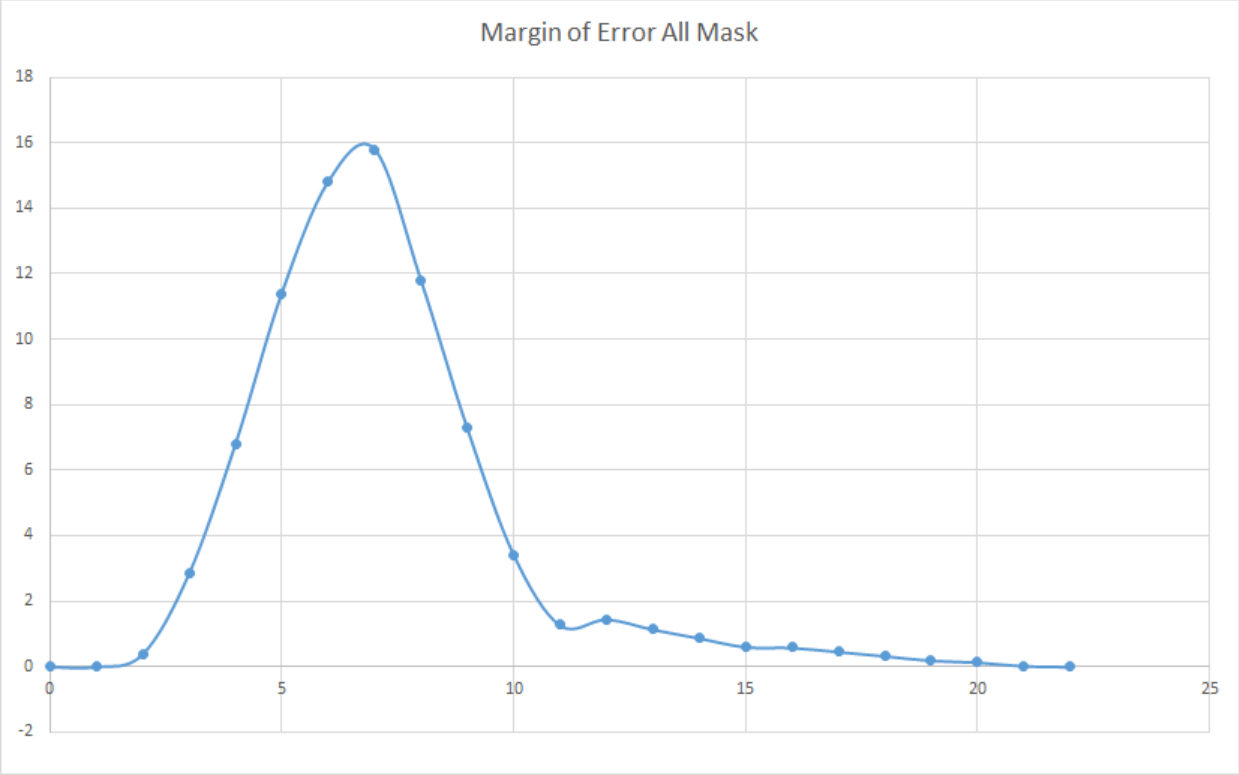
No Mask Margin of Error and Confidence Interval

Days	Point Estimate	Margin of Error	Confidence Interval (95%)
0	1	0	0
1	1	0	0
2	3.2	0.0367	3.16 - 3.24
3	20.6	0.643	20 – 21.2
4	113.8	5.03	108.77 - 118.83
5	336.2	12.8	323.4 - 349
6	871.2	23.2	847.8 – 894.2
7	1579.0	16.05	1563 - 1595
8	1870.0	3.62	1866.4 - 1873.6
9	1667.2	10.9	1656.3 - 1678.1
10	1233.6	16.6	1216.4 - 1249.6
11	636.4	15.7	620.7 - 652.1
12	183.6	7.89	175.7 - 191.5
13	15.4	0.833	14.6 - 16.2
14	0.4	0.0237	0.376 - 0.424
15	0	0	0



Sick Mask Margin of Error and Confidence Interval

Days	Point Estimate	Margin of Error	Confidence Interval (95%)
0	1	0	0
1	1	0	0
2	5.0	0.25	4.75 - 5.25
3	25.2	2.01	23.2 - 27.2
4	86.2	6.82	79.4 - 93.0
5	251.6	17.1	235.5 - 269.7
6	520.8	28.0	512.8 - 568.8
7	953.6	25.8	927.8 - 979.4
8	1464.4	20.4	1444.0 - 1484.8
9	1640.2	14.8	1625.4 - 1655.0
10	1491.8	23.4	1468.4 - 1515.2
11	1093.0	25.8	1067.2 - 1118.8
12	624.0	22.2	601.8 - 646.2
13	243.0	13.9	229.1 - 256.9
14	60.4	5.15	55.25 - 65.55
15	3.8	0.35	3.45 - 4.15
16	0.6	0.059	0.541 - 0.659
17	0.2	0.0196	0.1804 - 0.2196
18	0	0	0



All Mask Margin of Error and Confidence Interval

Days	Point Estimate	Margin of Error	Confidence Interval (95%)
0	1	0	0
1	1	0	0
2	6.4	0.383	6.02 - 6.78
3	33.6	2.83	30.8 - 36.4
4	91.0	6.79	84.2 - 97.8
5	205.2	11.4	19.8 - 216.6
6	337.0	14.8	322.2 - 352.8
7	446.0	15.8	430.2 - 461.8
8	479.6	11.8	267.8 – 491.4
9	423.8	7.28	416.5 – 431.1
10	326.8	3.42	323.4 – 330.2
11	205.6	1.27	204.3 - 206.9
12	112.6	1.45	111.2 - 114.1
13	60.4	1.14	59.3 - 61.5
14	33.2	0.877	32.3 - 34.1
15	19.0	0.605	18.4 - 19.6
16	13.6	0.583	13 – 14.2
17	9.4	0.458	8.94 - 9.86
18	6.8	0.331	6.47 - 7.13
19	4.2	0.192	4.01 - 4.39
w20	2.0	0.135	1.86 - 2.13
21	0.4	0.024	0.376 - 0.424
22	0	0	0

Time and System Constraints

Due to the scope of the project, there were several limitations to the simulation. This includes the number of agents in the simulation, as our computers were not powerful enough to have more than 2,000 simulated agents doing tasks at a speed that was greater than 32 times the regular simulation speed (which was 1 minute per second). This was because each agent had to keep track of other agents within range, which led to significant performance loss at times, such as when agents left for work or school.

Additionally, the agents have an infection radius of 3 meters, adding up to 6 meters total when colliding with another agent's sphere collider. In reality, this radius can be affected by many things, such as wind, AC units, air circulation in a building, sanitation, and other sorts of environmental aspects. However, for the purposes of the simulation, we will not consider these factors, as such techniques often require complex fluid dynamics models to be implemented. These were not implemented due to time constraints and to keep the overall system load within a manageable state.

Conclusion

The versatility of Unity for the purpose of simulations surpassed what we had initially imagined. Plastic SCM allowed us to work on the Unity project with ease, encountering very few issues throughout the project. While there were limitations in Unity due to the sheer number of lists that had to be kept track of, it ran well enough even with thousands of lists being updated at a time.

Based on our 15 tests around wearing a mask and not wearing one, the results were as expected, although our simulation based around wearing a mask upon becoming symptomatic were surprising. It appears that wearing a mask upon becoming symptomatic is not enough to stop the spread of a virus. In fact, we found that it only slightly delayed the whole population from getting sick. However, if both the infected and the susceptible agents wore masks, we found that the infection stopped spreading relatively quickly as more people began to wear masks. In the first two scenarios, eventually everybody became infected, but with agents proactively protecting themselves, even only when the infection is underway in the population, on average only about one third of the population will become infected. A little under 1,500 agents of the 2,000 remains unharmed.

Additionally, we learn that as the infection progresses, stochastic factors naturally mean that the margin of error, and thus the confidence interval, increases. At the start, and for the first day, we can predict the number of infected individuals with perfect accuracy, but as the infection spreads, prediction becomes more difficult. Interestingly enough, despite the more complex agent behaviors implemented for the third test, it is actually easier to predict the infection outcomes; the margin of errors for each day is correspondingly lower than that of the other tests, with the largest margin of error being 15.8 for day 7, as compared to 25.8 for the second test.

Considering the infection rates of individuals during the outbreak of COVID-19, our numbers are relatively higher. However, if we are to consider the close proximity in which our agents work and live together, these rates are not unreasonable.

The versatility of the simulation means it is easy to add new features or improve on existing ones. For example, while we have put considerable effort into optimizing the simulation via caching and staggering computations over time, further effort can be invested into making it easier to run large simulations. One potential method would be to make more effective use of multithreading to allow the system to fully utilize the resources on a computer. It is also possible to improve on the AI that powers the agents. While we have opted for a simple schedule-based mechanism, further work could focus on adding events and interactions that can allow agents to act beyond a simple scheduling system. For example, an agent could spontaneously decide to host a party with friends. Such activities were popular during the 2020 lockdowns, but our simulation does not emulate such behaviors. In general, we suggest that improving the code that manages how agents interact, by adding more human-like interactions, could improve the realism of the simulation.

References

Akhtar, J., Garcia, A. L., Saenz, L., Kuravi, S., Shu, F., & Kota, K. (2020, December 1). *Can face masks offer protection from airborne sneeze and cough droplets in close-up, face-to-face human interactions?—A quantitative study.* Retrieved from National Library of Medicine: https://doi.org/10.1063%2F5.0035072

Beckley, R., Weatherspoon, C., Alexander, M., Chandler, M., Johnson, A., & Bhatt, G. S. (2013, June 21). *Modeling epidemics with differential equations.* Retrieved from Tennessee State University: https://www.tnstate.edu/mathematics/mathreu/filesreu/GroupProjectSIR.pdf

Browning, R. C., Baker, E. A., Herron, J. A., & Kram, R. (2006, February 1). *Effects of obesity and sex on the energetic cost and preferred speed of walking.* Retrieved from Journal of Applied Physiology: https://doi.org/10.1152/japplphysiol.00767.2005

Center for Disease Control and Prevention. (2022, September 20). *How Flu Spreads.* Retrieved from Center for Disease Control and Prevention: https://www.cdc.gov/flu/about/disease/spread.htm

Center for Disease Control and Prevention. (2022, August 11). *Understanding Exposure Risks.* Retrieved from Center for Disease Control and Prevention: https://www.cdc.gov/coronavirus/2019-ncov/your-health/risks-exposure.html

Daughton, A. R., Generous, N., Priedhorsky, R., & Deshpande, A. (2017, June 19). *An approach to and web-based tool for infectious disease outbreak intervention analysis.* Retrieved from Scientific Reports: https://doi.org/10.1038/srep46076

DerSarkissian, C. (2022, January 22). *6 Reasons Why Your Cold Won't Go Away.* Retrieved from Web MD: https://www.webmd.com/cold-and-flu/your-cold-wont-go-away

Duan, W., Cao, Z., Wang, Y., Zhu, B., Zeng, D., Wang, F.-Y., . . . Wang, Y. (2013, September 1). *An ACP Approach to Public Health Emergency Management: Using a Campus Outbreak of H1N1 Influenza as a Case Study.* Retrieved from IEEE Transactions on Systems, Man, and Cybernetics: Systems: https://doi.org/10.1109/TSMC.2013.2256855

Hackl, J., & Dubernet, T. (2019, April 8). *Epidemic Spreading in Urban Areas Using Agent-Based Transportation Models.* Retrieved from Future Internet: https://doi.org/10.3390/fi11040092

Mayo Clinic. (2021, June 11). *Common cold.* Retrieved from Mayo Clinic: https://www.mayoclinic.org/diseases-conditions/common-cold/symptoms-causes/syc-20351605

Setti, L., Passarini, F., Gennaro, G. D., Barbieri, P., Perrone, M. G., Borelli, M., . . . Miani, A. (2020, April 17). *Airborne Transmission Route of COVID-19: Why 2*

Meters/6 Feet of Inter-Personal Distance Could Not Be Enough. Retrieved from Investigative Journal of Environmental Research and Public Health: https://doi.org/10.3390%2Fijerph17082932

Shipman, P. L. (2014, November). *The Bright Side of the Black Death.* Retrieved from American Scientist: https://www.americanscientist.org/article/the-bright-side-of-the-black-death

Appendix

This section contains various images of the environment for the simulation model described in this paper.

Figure 9: Bird's eye view of the town. The top faces North.

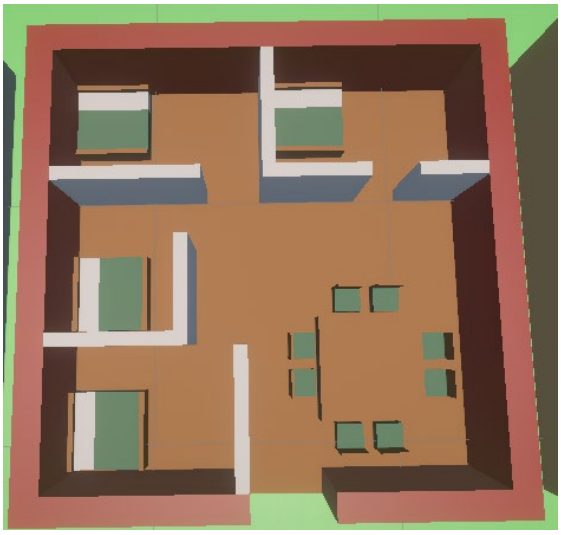


Figure 10: A house. Each house has enough beds and seating arrangements for eight agents.

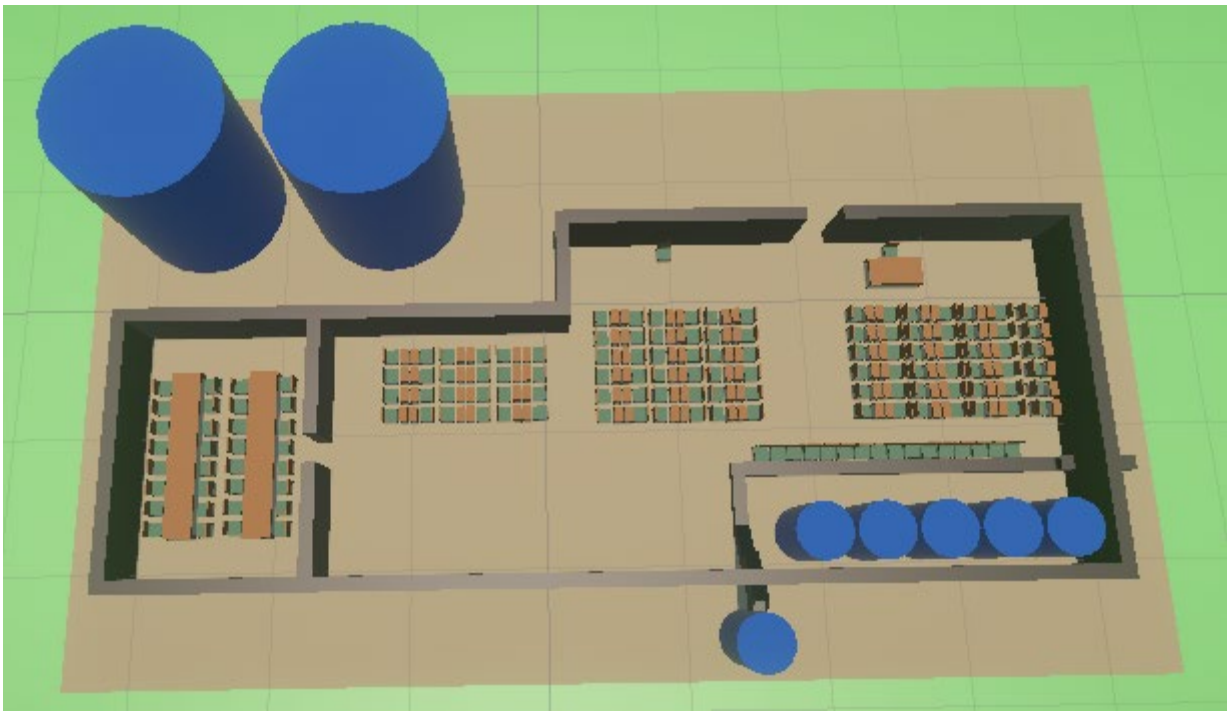


Figure 11: A factory. Adult agents work at one of various places, such as offices, stores, and factories.

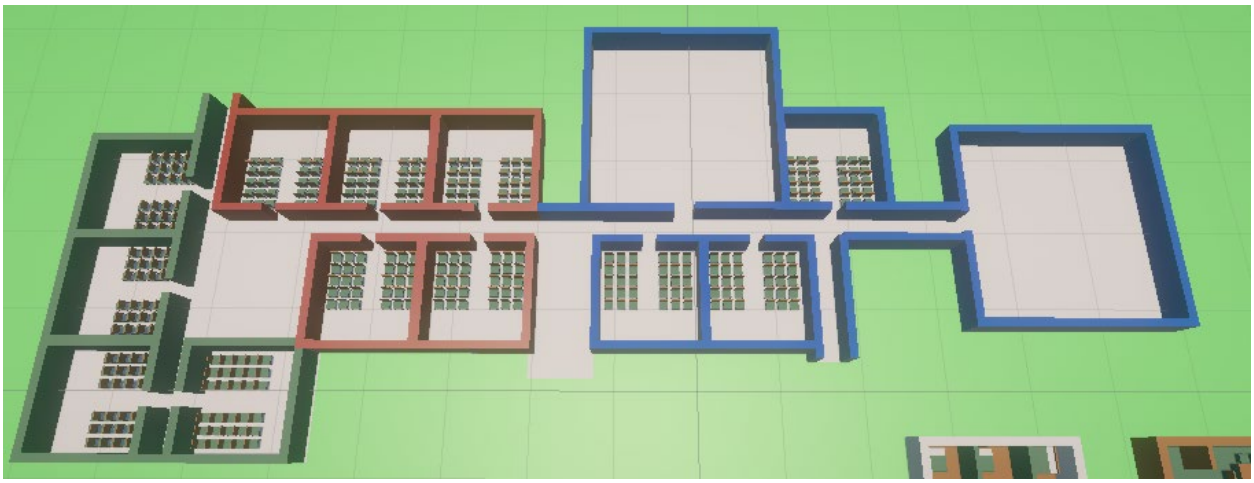


Figure 12: Children go to school during the day. They will switch classes for each of the four school blocks.