DisCease
A Complex Systems And Networks Project

Akhil Devarashetti

January 24, 2021

Abstract

I tried to create a model that simulates the spread of a disease that does not have a medicine. I observed the effects of varying parameters, then introduced a concept of deflections which mimic social distancing and social gatherings. I finally made an attempt to evolve these deflections based on a performance metric.

1 Introduction

During February 2020, the outbreak of COVID-19 was starting to become a pandemic. I recognized that this percolation is a complex systems problem. I started this project with the goal of developing a model that simulates COVID-19-like epidemics and study the behaviour of the agents in order to determine the best behaviours to mitigate the spread of the disease, hence, the name DisCease.

I created a modified SIR model[2] and considered it as the base model. I observed the outbreak patterns and then applied various techniques to the model which helps to reduce the spread of the disease. These techniques are listed below and explored in the Methods section.

- Length based social distancing.
- Perfect deflections with overlap and without overlap.
- Social distancing triggered after 20% of the population gets diseased.
- Social distancing only between unlike-agents.
- Evolved deflections.

I observed that some of these techniques yielded better results than the base model. The project is deployed at `https://discease.akhilez.com/` and I made the entire code open sourced on GitHub at `https://github.com/Akhilez/DisCease`

2 Background

Some ideas that inspired this project include the SIR Model[2], Herd Immunity[2], and Genetic Algorithms.[5]

2.1 SIR model

The Sir Model is a popular model created to predict the number of infections, recoveries and deaths caused by a certain disease. The SIR model consists of three sets of people which act as dependent variables. The groups of people being monitored include S(t), I(t) and R(t), which are the number of susceptible, infected, and recovered people with respect to time (t), which is an independent variable. The number of Infections increases whenever the Susceptible and the Infected have contact, and the person infected moves from the Susceptible group into the Infected group. For some value β, the rate of new infections is βSI. This results in the first differential equation

$$\frac{\partial S}{\partial t} = -\beta IS \tag{1}$$

In addition to this change in state, an Infected person can also be moved into the Recovered set at a rate of γI. This results in two different equations

$$\frac{\partial I}{\partial t} = IS - I \tag{2}$$

$$\frac{\partial R}{\partial t} = I \tag{3}$$

By using all of the equations above, I can summarize the total population as

$$\frac{\partial S}{\partial t} + \frac{\partial I}{\partial t} + \frac{\partial R}{\partial t} = -\beta IS + (IS - I) + I = 0 \tag{4}$$

By choosing different values for the three different unknowns, and plotting the number of diseased, Susceptible, and Infected I can monitor and predict how a disease can spread over time.

2.2 Herd Immunity

Herd immunity[2] occurs when a large portion of the population has gotten and has become immune to a disease, making it harder for a virus to spread. Herd Immunity is a way to protect the vulnerable people within the population. To reach the goal of herd immunity, a large portion of the population must cooperate. Since depending on people catching and recovering from the disease is not a reliable way of building immunity, vaccinations are the popular spread of immunization. Once a certain portion of the population is immunized, viruses can be eradicated.

3 Methods

3.1 System Description

Our modified SIR model consists of a 2D box-like environment with circular agents running around. I chose 100 agents to be good enough for the selected size of the environment. Adding more agents to the environment made it overcrowded and difficult to not catch the disease. Each agent has a location, velocity and an acceleration vector[4]. The dynamics of the system change the acceleration of the agent only, but not its velocity or location, resulting in higher fluidity in its motion. The environment adds drag to the moving agents. Each side of the environment is walled, which makes the agents bounce off when hit. All agents are set to overlap with each other, meaning, they do not collide, but pass through. I later change this behaviour to observe its effect.

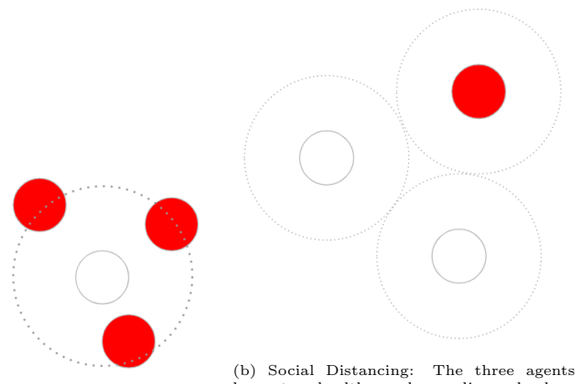
In our base model, each agent will be initialized with a random location and acceleration vector. They continue in a straight line for a short distance and change their acceleration vector, flinging them in a new direction. This behaviour makes more sense in this simulation than a simple random walk.

3.2 Life-Cycle of An Agent

An agent can be in one of 4 states - healthy, diseased, recovered and dead. Every agent starts out healthy. Each agent has a circular field of vision around it with radius greater than its own body as seen in Figure 3.1a. Every other agent that lies inside this field of vision is the agent's neighbour. When the agent is healthy and its neighbour diseased, there exists a probability proportional to the contagionRate with which the agent gets infected. As the number of diseased neighbours increase, this probability gets accumulated and it becomes more likely for the agent to get infected. When an agent is diseased, it recovers from the disease with a probability proportional to the recoveryRate, it also may die with a probability proportional to deathRate. When an agent is dead, it has no effect on the dynamics of the rest of the system. When an agent is recovered, it will become healthy/susceptible with a probability proportional to the recoveryLossRate. Also note that when an agent is recovered, it cannot get diseased until it becomes healthy again.

3.3 Social Distancing

Next, I introduced a concept of social-distancing[3]. Here, agents define a circular boundary around them with radius less than their visual field as seen in Figure 3.1b. No other agent can move inside this boundary. This ensures that the agents always maintain a certain distance from each other. This is a loose approximation of social distancing.



(a) Visual field of an agent.

(b) Social Distancing: The three agents here, two healthy and one diseased, show their social distance boundary.

Figure 3.1

3.4 Deflections

The social distancing approach discussed above is too strict and not flexible to simulate realistic social-distancing. So, I introduced a concept called deflections that enable more flexible social distancing behaviours. When an agent sees a neighbour, it can take an action of moving towards it, or an action of moving away from it. I termed these actions as deflections. Deflections are rules specific to each agent which dictate whether to move towards or away from another agent in its vicinity and how strongly. A repulsive force is represented as a negative scalar while the attractive force is represented as a positive scalar.

An agent can act differently based on the state of its neighbour. For example, the agent would like to move towards a healthy agent and away from a diseased agent. Likewise, the agent can have different behaviour based on its own state. For example, a diseased agent might want to move away from a recovered neighbour and get attracted towards another diseased agent. Finally, I end up with a force value for each pair of states except the dead state i.e. {(healthy, diseased), (healthy, recovered), ...}, a total of 9 scalar values. They are represented in a 3x3 matrix where each row represents the state of the agent and each column representing states of its neighbour. I can see these deflection matrix in the Figure 4.4. When there are more than one neighbour in the visual field, then the deflection force vectors from all the neighbours are summed and normalized for the final deflection force.

One can imagine how perfect deflections for an agent might look like. I defined perfect deflections as attraction on the pairs: {(healthy, healthy), (healthy, recovered), (recovered, recovered), (diseased, diseased)} and repulsion on pairs: {(healthy, diseased), (recovered, diseased)}. This deflection matrix can be seen in Figure 4.4c. This ensures that no healthy or recovered agent tries to stick around with a diseased agent, thus, reducing the spread of the disease.

3.5 Herd Immunity

I modeled herd immunity[2] by changing the deflection of (recovered, recovered) to repulsion in the perfect deflections. With this, the recovered agents try to spread around in the environment. Because recovered agents cannot get diseased directly, it lowers the overall probability of disease percolation.

The type of deflections and their effects on the percolation is not limited to the experiments I have conducted, there may be more interesting permutations of deflections. One more interesting permutation includes perfect deflections with repulsion between healthy agents. This shows a social distancing phenomenon not only between different groups but also between the individuals of the same group.

3.6 Evolution

One major question I asked ourselves is "Can I evolve deflections?". I then made an attempt to test the hypothesis. I ran the simulation for 400 episodes where each episode is 1500 time-steps.

3.6.1 Genes

I considered the deflection matrix as the genes of the agents. The genes remain unchanged throughout the agent's lifetime. The system starts out with random deflections.

3.6.2 Performance Metric

After each episode, the agents' score is calculated from the formula below:

$$score = \frac{nEpisodes}{nDiseased + nSpread} \tag{5}$$

Here $nEpisodes$ is the number of episodes the agent survived. $nDiseased$ is the number of times the agent got diseased. And $nSpread$ is the number of neighbours the agent spread its disease to.

This formula penalizes the score of the agents that get diseased often or that spreads the disease often. It rewards the agents which survive the episode. This way, I can drive the evolution to maximize survival and minimize death.

3.6.3 Selection Mechanism

After each episode, I select the agents for next episode with the following criteria:

- 50% of the agents of the next episode are taken directly from the best half of the previous episode.

- 40% of the agents are mutated copies of the agents of the previous episode picked with a probability proportional to its score. This is also called roulette wheel probability. The mutation is performed by adding a small random value between in the range [-0.1, 0.1] to the deflections element-wise.
- 10% of the agents are new agents with random deflections.

The idea behind this approach is that as agents with genes that try to attract towards diseased agents will tend to get infected, this will result in a low score. But the agents that repel diseased agents will have lower probability of getting infected, scoring higher. In this way, the system will eventually leave out the agents with bad genes and propagate the fitter genes through the episodes. In practice, however, this kind of evolution is not monotonic. The following section contains further details on this.

4 Results

I ran the simulation and took observations for many parameter combinations. Due to a large number of possibilities in varying the parameters, I filtered out the most interesting combinations and assigned serial numbers (SN{1, 2, ...}) to them. I ran the simulation for 1500 time-steps. They are as follows:

- Base model

SN1: Vanilla base model
SN2: High death rate
SN3: High recovery rate
SN4: Low recovery rate
SN5: High contagion rate
SN6: High recovery loss rate
SN7: Social distancing

- Perfect deflections model

SN8: Vanilla perfect deflections
SN9: No overlap between the agents
SN10: Social distancing
SN11: Social distancing after 20% of the population gets diseased
SN12: Social distancing disabled for like-agents (with same state)
SN13: 3x increase in deflection force
SN16: Repulsion between recovered agents

- Evolutionary model

SN14: Beginning of the evolution
SN15: End of the evolution

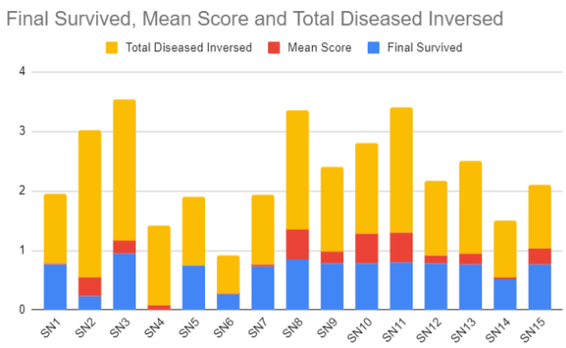
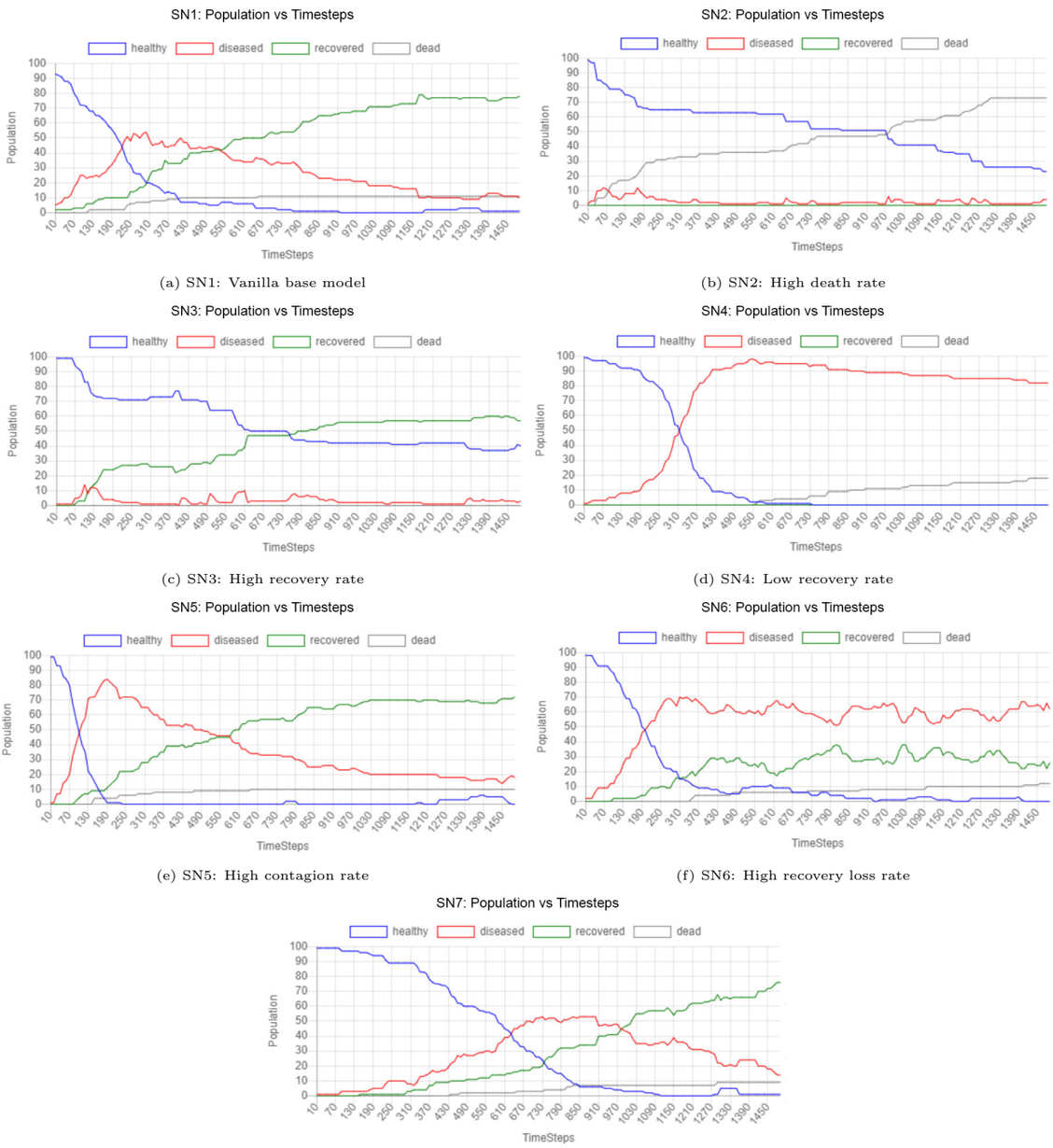


Figure 4.1: Performance of all cases

SN1: The base vanilla model is tuned to match the standard SIR model as seen in Figure 4.2a. I will be comparing other cases with this base model to make judgements. **SN2:** From the Figures 4.1 and4.2b, I see that when death rate of a disease is high, then the disease doesn't spread

as much. This is good for the goal of this project, but many agents die in the process. **SN3:** A better outcome is seen in the next case where I increased the recovery rate to a high value. As in Figure 4.2c the spread is small because as agents get infected, the soon recover and do not further spread the disease. **SN4:** I then lowered the recovery rate and the outcome was disastrous as seen in Figure 4.2d. **SN5:** Having high contagion rate is not as bad as having low recovery rate, this is seen in the Figures 4.1 and Figure 4.2e. **SN6:** However, the worst case of all turned out to be the one where the recovered agents lose their immunity quickly. As seen in Figure 4.2f, the number of diseased agents does not drop throughout the run. **SN7:** When I enabled length based social distancing[3], the system took longer to reach outbreak (the point in the population graph where number of diseased is equal to the number of healthy) as seen in Figure 4.2g. Although social distancing helped prolong the spread of the disease, the system still had an outbreak. Increasing the social distance made the outbreak happen later in time.



(a) SN1: Vanilla base model

(b) SN2: High death rate

(c) SN3: High recovery rate

(d) SN4: Low recovery rate

(e) SN5: High contagion rate

(f) SN6: High recovery loss rate

(g) SN7: Social distancing

Figure 4.2: Population graphs of base model variants

SN8: When I introduced perfect deflections into the system, many interesting patterns emerged. All the healthy and recovered agents form small groups from randomness. They resist coming close to diseased agents, which form their own groups. Formation of these groups is analogous to forming social gatherings between healthy agents and restraining diseased agents into quarantine cells. Ideally, this type of behaviour will restrict the spread of the disease completely and that is what I see in some simulations shown in Figure 4.3a. However, there is one major flaw in this system. Whenever an agent inside a group of healthy ones gets infected, then almost all the agents in that group get infected. The diseased population suddenly bursts into a large number. I see this in Figure 4.3c. Sometimes, this system spreads the disease and follows a similar trend to the base model, this is seen in Figure 4.3b.

I then modified this model with a few variations. **SN9:** I removed the overlapping rule and made the agents collide with each other. I observed a pattern that is very similar to swarm or flock where a large group of agents move in sync and form a macro-organism. However, due to this change, the gap between two agents becomes smaller on the whole and the probability of being in the vicinity of a diseased agent rises. **SN10:** I also added length based social distancing among all agents which had a similar effect to the previous case. **SN11:** I then enabled social distancing after 20% of the population gets diseased[3]. This is better than the base model, but suffers the problems of SN9. **SN12:** Now, I enabled overlapping and added length based social distancing to agents of different state. This produced the most robust model I have seen in our experiments as seen in the Figure 4.1. In half of the cases, the disease does not even cause an outbreak, similar to the Figure 4.3a. **SN13:** Since the perfect deflection model is so good, I tried increasing the strength of the deflections by 3. The agents immediately become flock-like and form dense groups. The disease-burst problem gets worse in this scenario and majority of the population gets diseased at once.

(a) SN8: Vanilla perfect deflections, type 1

(b) SN8: Vanilla perfect deflections, type 2

(c) SN8: Vanilla perfect deflections, type 3

(d) SN9: No overlap between the agents

(e) SN10: Social distancing on perfect deflections

(f) SN11: Social distancing after 20%

Figure 4.3: Population graphs of deflection model variants

SN14 & SN15: Finally, I ran the evolutionary algorithm and the results were quite outstanding. The evolution started off with random genes as shown in Figure 4.4a. The population graph at the beginning of the evolution is seen in Figure 4.6a which is similar to the base model as expected. After 400 episodes, the model evolved genes shown in Figure 4.4b that are somewhat similar to the perfect deflections defined as Figure 4.4c. I also see in Figure 4.1 that the evolved model performs better than the same model at the beginning of the evolution and is slightly better than the base model, but much weaker than the model of perfect deflection (SN8). The model learned to maintain social distancing in order to prevent the disease. This shows that the evolutionary algorithm works. The population graph of the evolved model is similar to that of SN8, the system with perfect deflections. The mean score of all the agents in the system keeps increasing overtime as seen in the Figure 4.5 due to the selection process.

One striking difference that I see in the evolved genes is that the deflection between the recovered agents is repulsive. This is exactly what I did to achieve herd immunity. It was surprising to see the evolution figure that out.

(a) Mean genes before evolution

(b) Mean genes after evolution

(c) Perfect deflections

Figure 4.4: Mean Genes

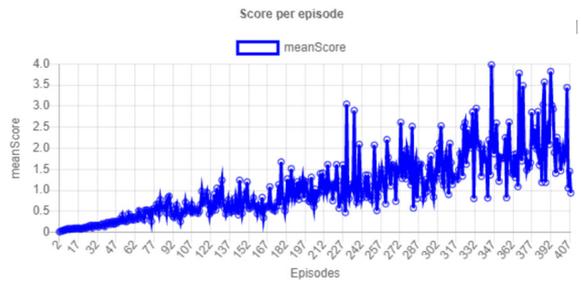
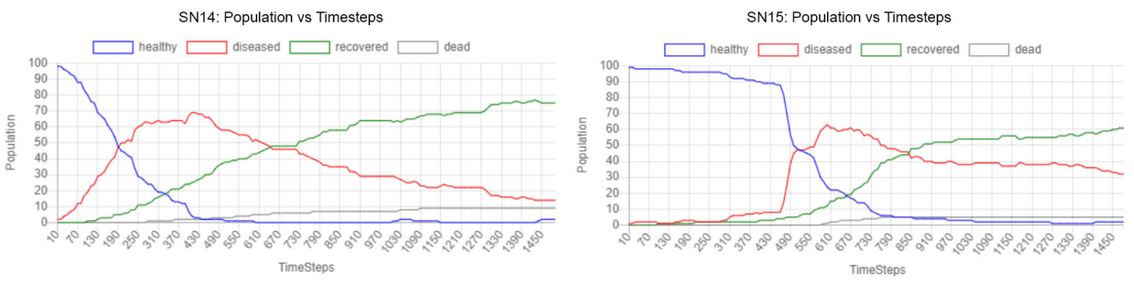


Figure 4.5: Scores for each episode in the evolution



(a) SN14: Beginning of the evolution

(b) SN15: End of the evolution

Figure 4.6: Population graphs of evolutionary model.

5 Conclusion

I have successfully demonstrated that social distancing is a key factor in preventing the percolation of a disease in a highly active and dynamic system. I also demonstrated that herd immunity helps in this process. Furthermore, I evolved the system itself to mitigate the spread and it resulted in a system with social distancing and herd immunity. One issue with the evolutionary process I found is that it follows preferential attachment[1] where the agents of the higher score will always remain in the system even if they have a few bad deflections in their genes. Another issue is that the evolved system will always have agents with random deflections due to the selection process. These limitations will never let evolution reach 100% perfection.

References

[1] Albert-Laszlo Barabasi and Reka Albert. "Albert, R.: Emergence of Scaling in Random Networks. Science 286, 509-512". In: *Science (New York, N.Y.)* 286 (Nov. 1999). DOI: 10.1126/science.286.5439.509.

[2] Joanna Nicho. "The SIR Epidemiology Model in Predicting Herd Immunity". In: May 2010.

[3] Grant Sanderson. *Simulating an epidemic*. 2020. URL: https://youtu.be/gxAaO2rsdIs (visited on 04/29/2020).

[4] D. Shiffman, S. Fry, and Z. Marsh. *The Nature of Code*. D. Shiffman, 2012. ISBN: 9780985930806. URL: https://books.google.com/books?id=hoK6lgEACAAJ.

[5] Anita Thengade and Rucha Dondal. "Genetic Algorithm – Survey Paper". In: *IJCA Proc National Conference on Recent Trends in Computing, NCRTC* 5 (Jan. 2012).