A Simulation of Infection Suppression Across Multiple Delivery Methods

Tiara Barlow, Jared Dewey, & Kyle Mitchell

William and Mary

Random Walks in Mathematics and Biology
Professor Shaw
May 6, 2025

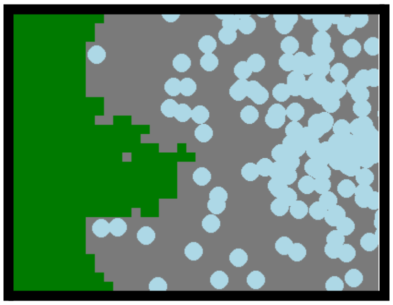
Abstract

This simulation compares how three drug delivery methods: injection, inhalation, and oral ingestion, affect the spread and control of infections. Using a 2D grid and random walks on Python, we model how medicine diffuses through tissue to reach a target area (infected cells). The simulation removes the immune system to focus on how the method of delivery influences the outcome. Each method is tested by changing how much medicine is released and how fast it starts to work. This resulted in injection showing to be the most effective in reducing infection quickly. Inhalation often fails to control the spread of the infection, but it does give immediate relief to a localized area. Pills work much slower than the other two but tend to succeed over time. These differences demonstrate how different delivery methods can affect the outcome of infection spread.

Background

The effectiveness of any medication heavily relies on the delivery method to the body. While the actual product that is entering the body is crucial, the route it takes to reach infected tissues or cells can quickly change how effective the medication is. Different delivery methods, such as injection, inhalation, or oral ingestion, are used based on the urgency, target location, and other physiological barriers that need to be considered.

Each delivery method engages with the body in different ways. Injections, as Ford Versypt et al. (2013) note in their review on drug delivery through degradable microspheres, “emergency or chronic conditions where sustained releases are less critical than immediate systemic availability” (p.30). In life-threatening situations, like sepsis or rabies, typically the primary concern is the speed of the application of the medicine. Injection drugs bypass many of the body's natural filtration systems, like the liver's metabolism or the breakdown of enzymes in the digestive tract. Afterwards, they enter directly into the bloodstream or muscle tissue, allowing them to take effect within minutes. In our simulation, we look at very fast acting scenarios of injection. Injected drugs, having the ability to reach most parts of the body quickly, make this method one of the most effective in suppressing infections that would have spread uncontrollably past the early development stages.



In contrast, oral ingestion requires the medication to traverse the gastrointestinal tract, and once in the stomach, diffuse into the bloodstream. Nichols and Bassingthwaighte (2008) talk about the barrier effects of intestinal epithelial cells and transporter proteins that can delay compounds from reaching the bloodstream at the predicted rate. This delay is not something small, like we talked about in the injection section; suppressing the development at an early stage can severely lessen the effects of infections on the body. Even with their cons, oral drugs remain the most common form of everyday medicine. They are the most non-invasive, easily distributable, and most effective drug for long-term management of diseases and infections like ear infections, strep throat, and urinary tract infections.

Inhalation occurs in more niche scenarios, it is designed for localized treatment within the lungs. Inhaled drugs often fail to reach deep into various systems, disregarding very unique bioengineered examples. This method, while very fast and convenient, falls short in long-term effect and overall impact on infections. Inhaled drugs are normally used as a stopgap for chronic scenarios like asthma, while also treating illnesses like bronchitis.

Another factor that we looked into was spatial modeling. The crucial factor of drug delivery is not just how fast or potent the medication is, but also how accessible the target area is. Highly vascular tissues like muscle and liver allow for a much faster diffusion, which is why when injecting into muscle tissue, it will normally reach its target area much faster; while fatty tissues can greatly slow the diffusion of medication. The physical structure of a tissue influences how fast either a pathogen or a drug can travel, our model will simplify this, allowing both infection and medication to travel a 2D grid. In a real-world model various barriers could come into play, such as fibrotic tissues, fat, or bone that could slow or entirely block diffusion. Though our model will lack that complexity, it will still allow for a strong foundation for further spatial aspects to be added, while giving us preliminary data that can be drawn from.

Methods / Simulation

For our simulation, we coded purely in Python - using the Pygame library to render frames and MatplotLib to create graphs. This tech stack was chosen due to us having more experience with it, and therefore being more comfortable with it. We were able to focus more on the high-level aspects of the model without worrying about the low-level specifics.

The goal of the simulation is to model the arrival of medicine to an infected area of the body and to monitor its impact on the amelioration of disease. Now, real world healing involves interactions between disease, medication, and the immune system. Each of these alone constitutes an entire branch of research, so, to keep this situation tractable, some assumptions and simplifications need to be made:

1) Disease is a state of being. There are no viral particles floating around nor bacteria interlaced throughout the cell grid. A cell is either in an infected state or not, and that is all there is to disease.
2) There is no immune system. There are no phagocytes consuming infected cells nor neutrophils exploding in the infected area.
3) Cells form a perfect, uniform grid. The size of the cell grid is in theory variable, however for all our simulations we just used the size 40x30. This was chosen due it being large enough to give the infection and medicine room to move while not being too slow to run reliably.
4) Disease has no refractory or incubation period. An infected cell can infect a number during one simulation tick, then infect another the next frame. The same goes for freshly infected cells: they can immediately start spreading disease to their neighbors.

The Simple Model

For our initial simulation model we also added the following assumptions:

1) There is no spontaneous self recovery in cells. Once infected, a cell will remain so until a medication molecule reaches it.
2) Infection only implies the spread of infection. There is no cell death nor other negative side effects.

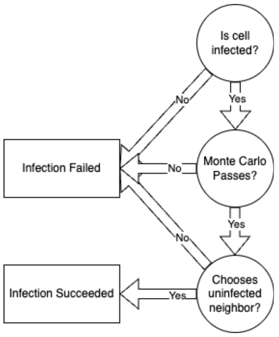
These assumptions will be relaxed in a later simulation, but were initially made to reduce the programmatic complexity of the simulation. Our first simulation will from here on out be referred to as the “simple model” and our second will be referred to as the “SID model.”

The simple model follows a straightforward schedule:

1) Attempt to spawn medication
2) Allow disease to spread
3) Move medication around and clean cells

Ironically, the arrival of medication was the last part of this simulation to be implemented. There are a few parts involved in this, but it is still rather simple. As the model runs, it keeps track of how many molecules have arrived at the site. During the arrival phase, a Monte Carlo test is run for each remaining molecule to see if it will arrive. If it is successful, then the molecule will be spawned at a predetermined location (which is the same for each).

Next, the model tries to spread the disease. Having cells arranged in a simple MxN grid, we used a simple double buffer to represent the grid (the implications and motivations of this will be discussed later). Disease spread is handled in 2 simple parts. Initially a Monte Carlo test is run to see if the disease should attempt spreading. If it passes, then the cell will choose a random neighbor to attempt spreading to. If the neighbor is uninfected, then the infection is a success and it becomes infected. If it is already infected, then the infection fails.



Finally, the medication molecules actually diffuse around the infected area. This is handled with a random walk. A theta is chosen at random, then the molecule travels a fixed step length in that direction. Then, the molecule scans all cells it is touching and disinfects them.

Parameters / Notes on Simple Simulation

Despite only having 3 parts, there are many parameters at play. These are stored in a Scenario object separate from the actual simulation. This separation of concerns allowed us to more easily play around with the actual simulation or the parameters without worrying about affecting the other.

These parameters were held constant across each delivery type:

● Ticks Before Medication Release:	0
● Disease Spread Chance:	0.125
● Grid Dimensions:	40x30
● Medication Spawn Location:	(40, 15) (center right)

These parameters were changed for each delivery type:

- Maximum Medications Released
- Medication Spawn Chance

The maximum amount of medication released corresponds to the amount of medication in the delivery type for obvious reasons. More interesting is how the medication spawn chance corresponds to arrival speed. A lower spawn chance means that a molecule is less likely to appear at any individual frame, so the probability of it arriving at a later time increases.

Another point to note is the difference in bounds between the medication and disease. The medication molecules store a “world position” and thus are able to exit the bounds of the simulation while the disease stores a “grid position” and thus is stuck in the bounds. This has the effect of medication slowly and naturally diffusing out of the system for essentially no extra computation cost!

The SID Model

While the additional assumptions made by the simple model were helpful for quickly building a cohesive and easily expandable framework, they also limited what we could simulate. Essentially, we could test different delivery types and different levels of infectiousness in disease, but nothing more.

By relaxing these restrictions, we are better able to simulate different types of disease. Now, cells have three states: Susceptible, Infected, Dead. They can both recover on their own or die due to infection. Everything in Parameters / Notes on Simple Simulation still holds, however there are just two new parameters in play:

- Mortality Rate
- Self Recovery

Just like Maximum Medication Released and Medication Spawn Chance, one of these parameters clearly needs much more justification than the other. While cells don't display true spontaneous self recovery, they do exhibit some level of disease defense with interferons (Immunology). This also gives us a way to control an immune defense without needing to simulate moving cells.

The actual code to work these new parameters was quite simple. The Scenario objects were given the corresponding two extra parameters with default values set to 0 for backwards compatibility. The only function that actually saw any change was the one used for simulating disease.

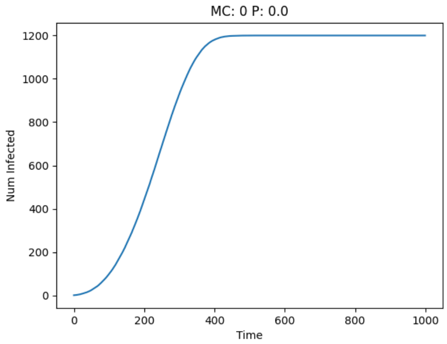
A new Monte Carlo simulation is now run attempting disease spread to see how the cell should evolve. This was decided because while mutually exclusivity is assured for cell death and recovery, it isn't necessarily true for them and the infection of a neighbor. If a cell is deemed recovered, then it is no longer infected and continues on as normal. If it is deemed dead, then it is no longer infected and immune to infection for the rest of the simulation. Running tallies are updated accordingly

Results

For all models, there were two parameters, medicine concentration (MC) and rate of medication arrival in the desired area (P), and drug delivery begins immediately at 0. Each model had their parameters set based on the method of drug delivery. All simple models were simulated over 50 runs, and all SID models had two other parameters, mortality rate and self-recovery rate.

Unmedicated Growth

After running the simple model of unmedicated cell growth, whose parameters exclude medication arrival in the system with no medicine concentration (MC) and no rate of arrival of medication (P), the graph displayed a logistic growth of the number of infected cells in the system.

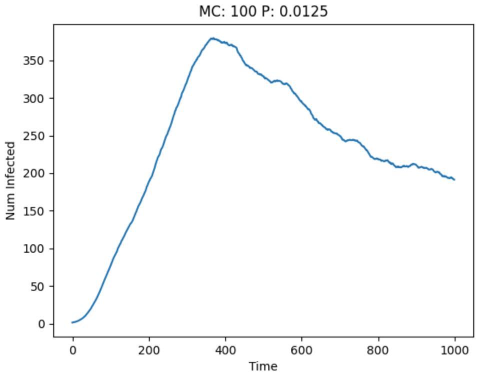


With the number of infected cells starting at zero, the lack of medication allows the infection to spread within the body quickly, and it rapidly increases to the maximum number of 1200 infected cells set in the simulation after around 400 arbitrary time units. The number of infected cells stays constant at 1200 after the rapid growth, since the lack of medication in the system does not allow the infected cells to be "cleaned".

The growth pattern suggests an initial exponential growth, then a slowdown as the number of infected cells reaches its maximum. This demonstrates how rapid infection can overwhelm the system. The lack of medicated recovery and self-recovery in the simulation shows that the body's natural resistance is either not present or ineffective. Biologically, the unmedicated scenario can simulate newborns with ineffective immune systems and how they are easily susceptible to infections. In the absence of proper care, disease can overwhelm the body, emphasizing the importance of early vaccines and medications to ensure long-term survival and immune system development.

Inhalation Method

Considering a simple model with new parameters of a medicine concentration of 100 and a rate of arrival of medication of 0.0125, we simulated the inhalation method of medicine delivery to the lungs. The graph shows a quick increase to a peak, then a slow decrease in infected cells.

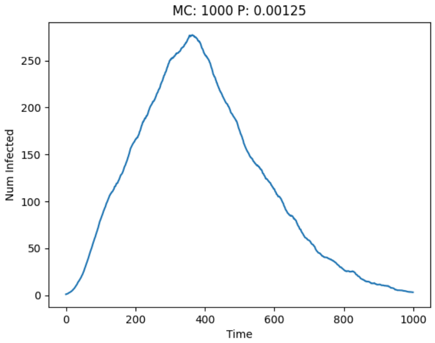


Though the number of infected cells starts to increase up to 350, it starts to decrease slightly as the medication hits the lungs. However, over a long period of time, the infected cells do not completely dissipate due to the low medicine concentration and low rate of arrival.

In a real-world situation with inhalers, the body would respond to a pump of aerosol medication with initial relief, but it would not fully eliminate the underlying inflammation. Since lung conditions like asthma do not fully recover over time with just short-term medication, symptoms are likely to return without long-term treatment or management.

Ingestion Method

To simulate medicine delivery by oral ingestion, we set the parameters of the medicine concentration to 1000 and the rate of arrival to 0.00125. The simple model of the ingestion method shows a peak after a gradual increase, then a gradual decrease back to 0 infected cells:

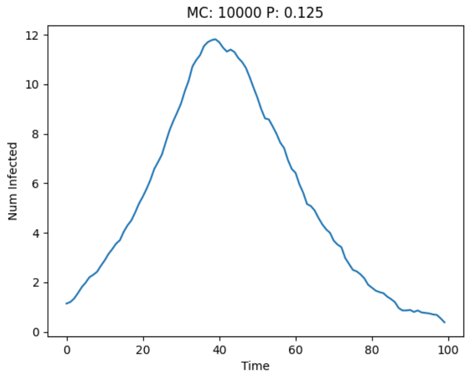


This simulation takes approximately the same time period as the inhalation delivery method. However, because of the higher medicine concentration and lower rate of arrival, it takes more time for the infection recovery to take effect. As the pill moves through the body and slowly diffuses medicine over time, the infection gradually disappears.

For an infection in a certain area of the body, such as strep throat, oral medicine delivery taken orally would be a delayed but successful process in getting rid of the infection. They do not provide instant relief like inhalation, but the infection clears after several days of sustained dosing, just as the model demonstrates.

Injection Method

The last simple model of the injection method has parameters set to a medicine concentration of 10,000 and a rate of arrival of 0.125. The graph resembles a similar increase and decrease pattern as the ingestion method, with a shorter time period:

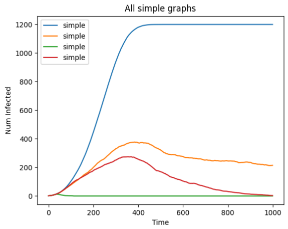


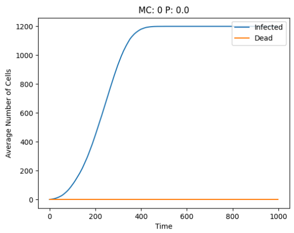
Because the parameters are much higher than the inhalation and ingestion methods, this method takes a shorter amount of time to deliver medicine to the desired areas. The number of infected cells is less than the number from the other delivery methods as well, which shows how fast-acting this method is.

Once the drug enters the system, the infected cells are quickly cleaned off, which is an optimal delivery method for conditions that are highly contagious such as rabies. Before the infection reaches the brain and starts to rapidly multiply, taking vaccines for rabies gets rid of the infection fairly quickly.

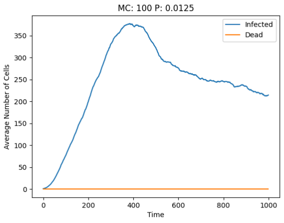
Simple Infection SID Model (No Recovery or Mortality)

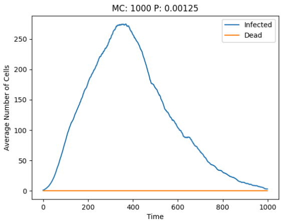
All graphs below show the simple infection models, which shows dead cells as the orange line and infected cells as the blue line (only on graphs 2-5).

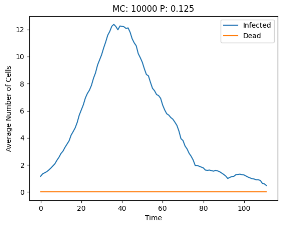




The graph displays, respectively: simple model comparison, unmedicated disease growth, inhalation against disease growth, ingestion against disease growth, and injection against disease growth.



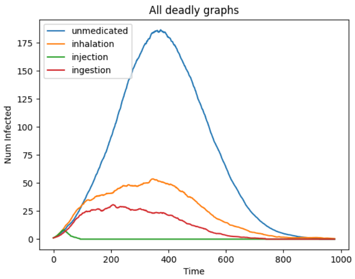


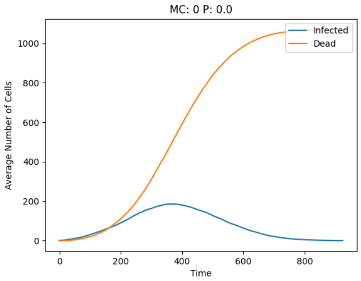


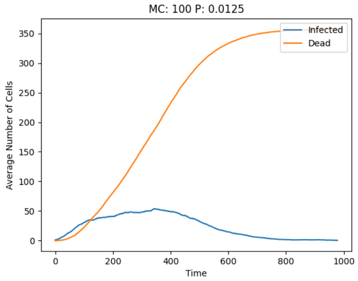
Each graph produced the same outcome as the graphs from the simple model. Since the simple model is essentially the SID model with these parameters set, this demonstrates we did not mess up the model.

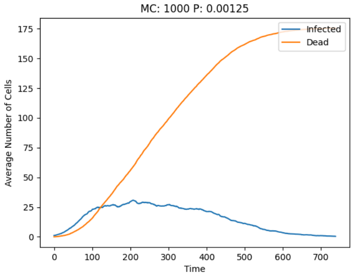
Deadly Infection (Mortality Rate Only)

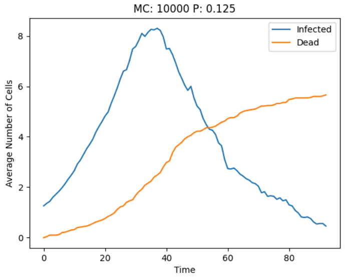
The graphs below show each delivery method from above, respectively, but now with the mortality rate parameter set to half of the infection rate of each condition











The second graph shows the unmedicated disease growth, which shows the infected cells with a short bell-curve like line and the dead cells in an S-shaped curve. Because the cells now have an added mortality rate, the infected cells decrease after 400 arbitrary time units. This is not because they have recovered, but it is because those cells have died from the infection.

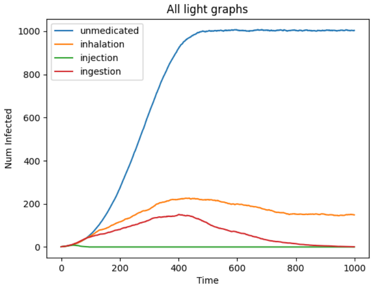
The third graph shows the inhalation delivery method against disease growth. As mentioned before, since the inhalation method is not as effective as the other two methods of drug delivery from low medicine concentration and rate of arrival, the cells will die in a similar way as the unmedicated growth does.

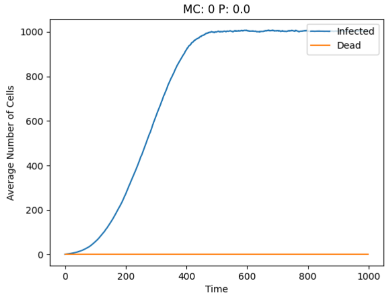
The fourth graph shows the oral ingestion method against disease growth. Because the rate of medicine arrival is extremely low, the infected cells die faster than the inhalation method as no immediate relief is provided from pills.

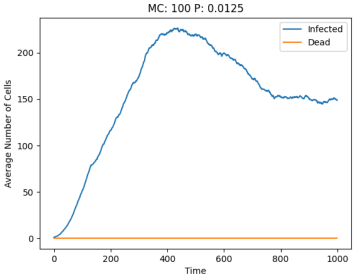
The fifth graph shows the injection method against disease growth. With a high medicine concentration and rate of arrival, the number of both infected and dead cells at their peaks are much lower than the other two methods because the direct and immediate delivery of medicine helps the body recover from infection faster.

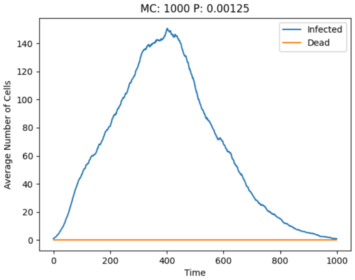
Light Infection (Recovery Rate Only)

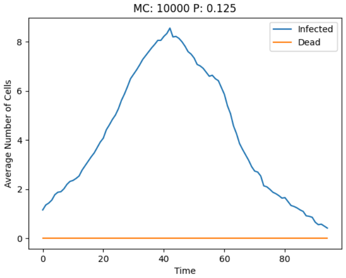
Removing the mortality rate from the other graphs, we added the recovery rate, which is half the infection rate, to each graph:











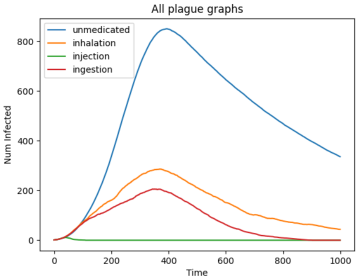
All graphs, excluding the comparison graph, display the dead cells constantly at 0. Since the mortality rate was removed and the recovery rate was added, it was easier for the cells to recover from the disease, thus fewer cells died.

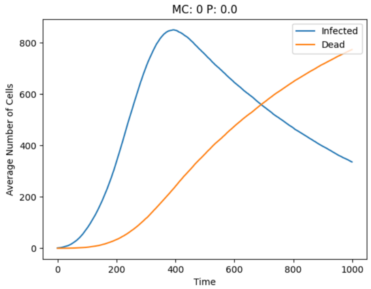
The second graph shows the unmedicated disease growth, and since there is no mortality rate, it is extremely similar to the simple model with the S-shaped curve.

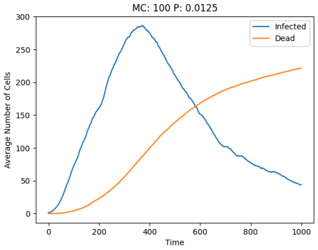
The third, fourth, and fifth graph shows the infected cells being treated with the inhalation, ingestion, and injection treatments. These models are extremely similar to their simple models but with less sharp peaks and valleys, which is most likely due to the recovery rate. Recovery isn't an immediate process, so a less steep graph makes the recovery process more realistic.

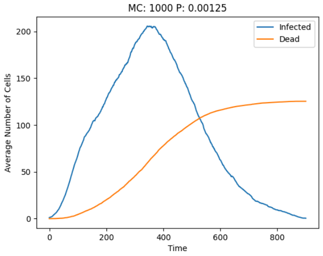
Plague (Low Recovery, Medium Mortality)

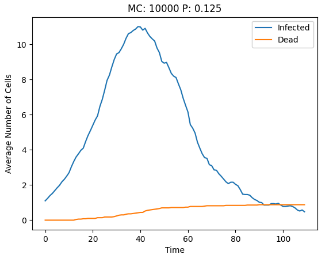
To simulate a plague-like infection, we simulated with a low recovery rate and a medium mortality rate:









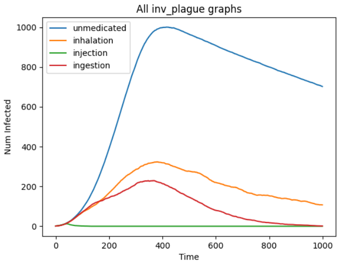


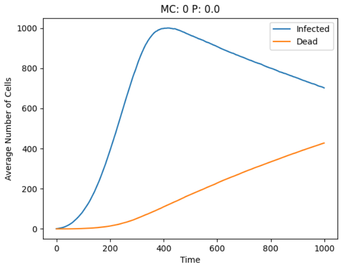
The first graph shows all the graphs together for comparison.

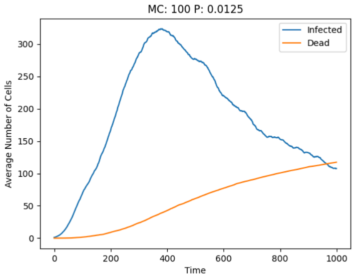
The unmedicated disease growth, inhalation, and ingestion graph shows a peak of infected cells and a decrease due to those cells dying from the disease. However, for the injection graph, there is only a minimal amount of cells dying from infection because the delivery method was efficient enough to stop disease growth, even at a medium mortality rate.

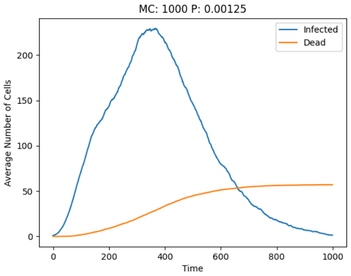
Inverse Plague (Medium Recovery, Low Mortality)

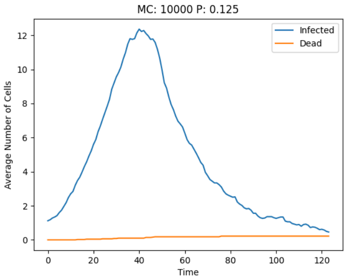
For our last simulation, we switched the recovery and mortality parameters.









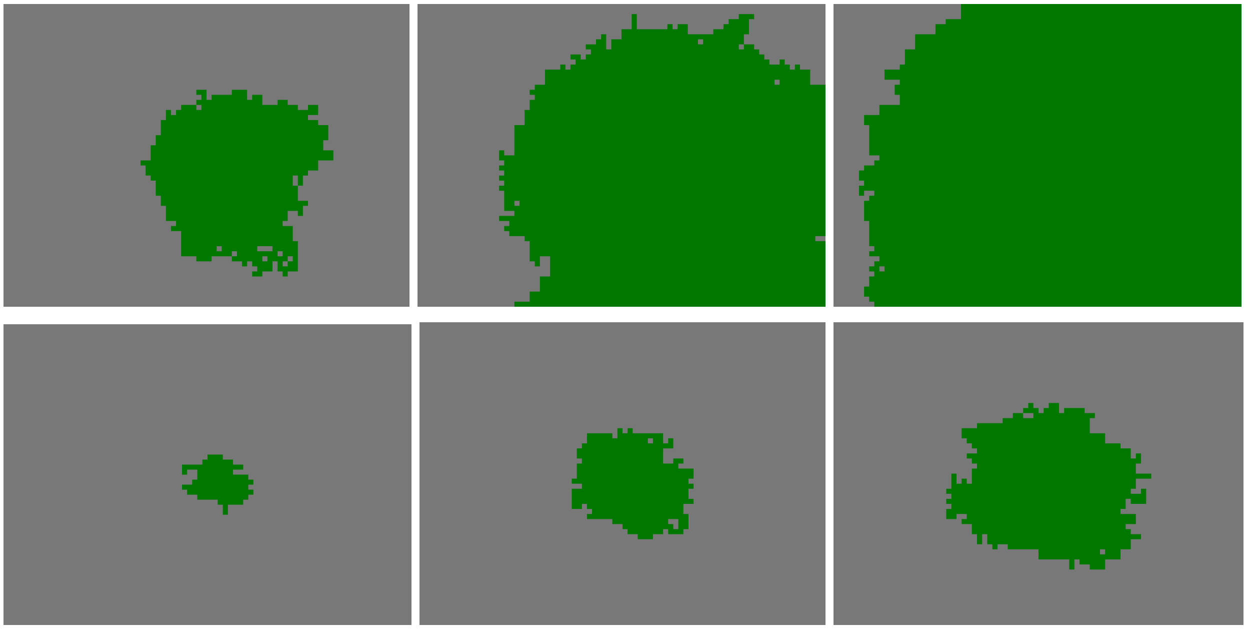


The first graph shows each inverse plague graph together for comparison.

The unmedicated disease growth and the inhalation delivery method show a peak and an increase in dead cells, but because of the higher recovery rate than the normal plague-like simulation, both graphs seem to have a faster rate of decreasing infected cells after the peak. The ingestion and injection methods both have similar infected cell lines to their simple models.

Complications

The simulation ran as expected as we developed it save for two major hiccups: disease initially spread to the bottom right of the grid much more than expected and the count of dead cells grew inanely.



On top is a representation of the unfixed growth and the bottom is one of the fixed growth for (25, 50, 75) ticks. Without the context of the disease growing much quicker to the bottom right than expected, the most obvious difference is the size of the infected area between the different versions. This greatly hints at the cause of the bug, but we lacked this knowledge of a properly run simulation at the time.

Initially we suspected something was going wrong with infection direction selection. Much time and effort went into researching how Python’s random library works, in particular the choice() function. We even ran tests to make sure the expected output probability matched a uniform distribution in both an external environment and within the simulation itself. All of this showed that there was no significant difference in probability of travelling in any direction.

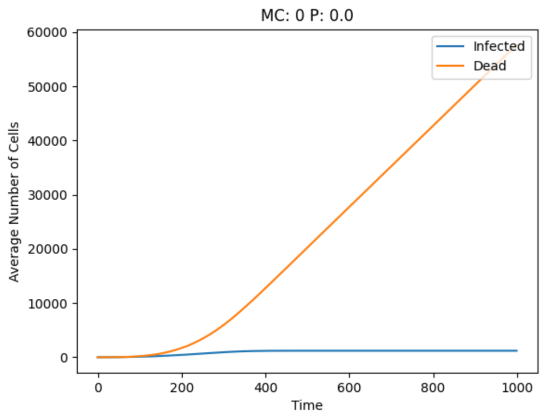
What was going wrong with our simulation requires more context on the initial structure. All cells were stored in a singular grid. The disease spread search would start in the top left corner, first going right then downwards to see if the infection would spread. When infection spread to a new cell, that cell was updated immediately.

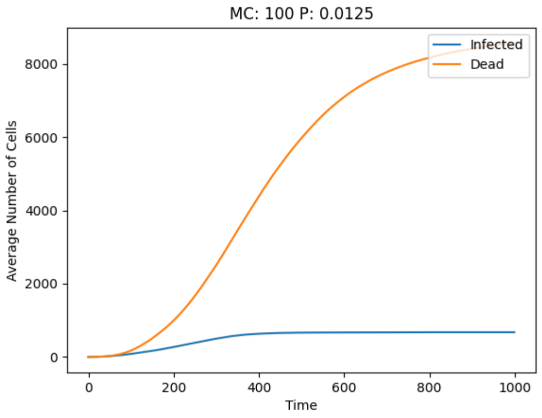
With all that stated, it is quite an obvious issue: disease would spread downwards or rightwards, then that newly infected cell would immediately start infecting its neighbors. When a cell is infected from below itself or from its right, it would not be checked again, and thus wouldn't get a chance to spread again. This was a major oversight that would have greatly altered how quickly disease spread. We certainly wouldn't have noticed it if the spread had been uniform, so in a way we were lucky it was disastrous.

The fix to this issue was pretty simple, though it had a ripple effect through the rest of our code: we had to store changes separate from the current frame data. Every time the grid data was accessed, it had to be either in the "current" version with frame updates on it or from the unaltered frame data. Setting this up early on saved a lot of time trying to fix bugs due to the inevitable improper accesses we would have set up if we switched to this system instead of starting with it.

We were not so lucky when we migrated our code to the SID Model. For tracking deaths, we added a simple boolean flag to cells to denote whether or not they are alive. If a cell is marked dead, it cannot be spread to or from. Simple enough, right?

Well, we found some interesting results:





This was run with the “very deadly” scenario - a name very fitting for a disease that killed over 50 times the number of present cells. It is interesting to note that medication still helped to fight the illness even though it greatly outgrew the simulation bounds.

The fix to this was simple, just tedious. Whenever a cell is killed, it is also counted as “cleaned” as it is no longer infected. In addition to this, every time a cell’s state is checked, we had to also add a check for whether it is alive.

Possible Improvements

Concentration gradients also drive the movement of drugs. In real biological systems medications often migrate from areas of high concentration (for example the injection site) to areas of low concentration (the target area, like an inflammation or infected tissue). While our current model uses an unbiased random walk, a future model could have particles that are more likely to move toward infection clusters, simulating chemotaxis or guided diffusion.

Particle size and solubility also dictate a drug's ability to move through tissues. Large biological molecules require direct delivery because of how poorly they tend to diffuse through membranes, where smaller lipid soluble compounds can passively diffuse throughout cells. A future change could model different particle sizes and how they might affect the movement through a cell.

Finding a way to incorporate an immune system response into our simulation could obviously increase the accuracy tenfold. Seeing as how crucial the way our body reacts to infections alone is, incorporating that with medicine as well could make our simulations while much more confusing, also a lot more accurate.

Another factor that can greatly affect outcomes is patient variability. Things like age, body mass, metabolic rate, and other preexisting health conditions greatly affect how a drug behaves. In most pharmaceutical companies this is avoided by adjusting dosages or how the drug is administered. This could potentially be modeled by incorporating variability into the cell's recovery or susceptibility rate. This could model how medicine could be more successful in different parts of the body dependent on a patient's existing ailments.

All infections are not the same, so another variable we contemplated adding was infection variety. Since some infections have varying resistance to drugs, spread rates, and also the ability to regrow and adapt after treatment. While this would provide a more accurate representation of infections we felt it would over complicate our data.

A mix of multiple medicines was another idea that could be incorporated into our simulation. Seeing as a lot of infections are treated with a mix of medications, this could give a closer look into how an early injection could provide suppression, while at the same time a pill administered could continue to treat the infection over a long period of time.

Changing the dosages was one of the final changes we contemplated, while on a surface level one of the delivery methods wasn’t working, increasing the dosage could end up changing the outcome of the method, making one a lot more applicable than we would have previously thought.

References

Curtin, S., Tajeda-Vera, B., & Bastian, B. (2023). Deaths: Leading causes for 2021 (National Vital Statistics Reports, Vol. 73, No. 4). Centers for Disease Control and Prevention. https://www.cdc.gov/nchs/data/nvsr/nvsr73/nvsr73-04.pdf

Ford Versypt, A. N., Pack, D. W., & Braatz, R. D. (2013). Mathematical modeling of drug delivery from autocatalytically degradable PLGA microspheres—A review. Journal of Controlled Release, 165(1), 29–37. https://doi.org/10.1016/j.jconrel.2012.10.015

Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry, 81(25), 2340–2361. https://doi.org/10.1021/j100540a008

Lai, P., & Liang, Y. (2020). Mathematical modeling of drug diffusion and reaction kinetics in biological systems. Pharmaceutical Research, 37(8), 112–128. https://doi.org/10.1007/s11095-020-02845-3

Nichols, J. W., & Bassingthwaighte, J. B. (2008). Tissue barriers solute transport. Comprehensive Physiology, 5, 1135–1193. https://doi.org/10.1002/cphy.c080006

Santo, L., & Kang, K. (2022). National Ambulatory Medical Care Survey: 2019 national summary tables. Centers for Disease Control and Prevention. https://doi.org/10.15620/cdc:123251

British Society for Immunology. (n.d.). Immune responses to viruses. https://www.immunology.org/public-information/bitesized-immunology/pathogens-disease/immune-responses-viruses