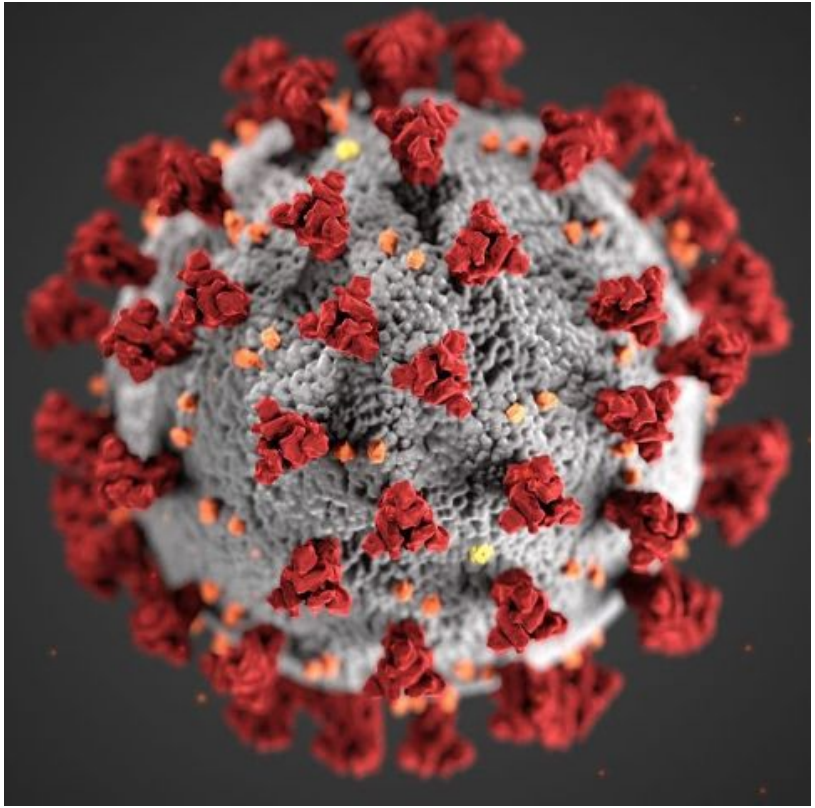
Agent-Based Mutations

Jun Han, Taekyu Kim, Jiahao Xu, Michael Quintieri
CMSE 202, Section 002
https://docs.google.com/presentation/d/1YxcXpd4M4tZzCpJQrh0iQfxla_vDx6y_xenJEEvgrm8/edit?usp=sharing

Question

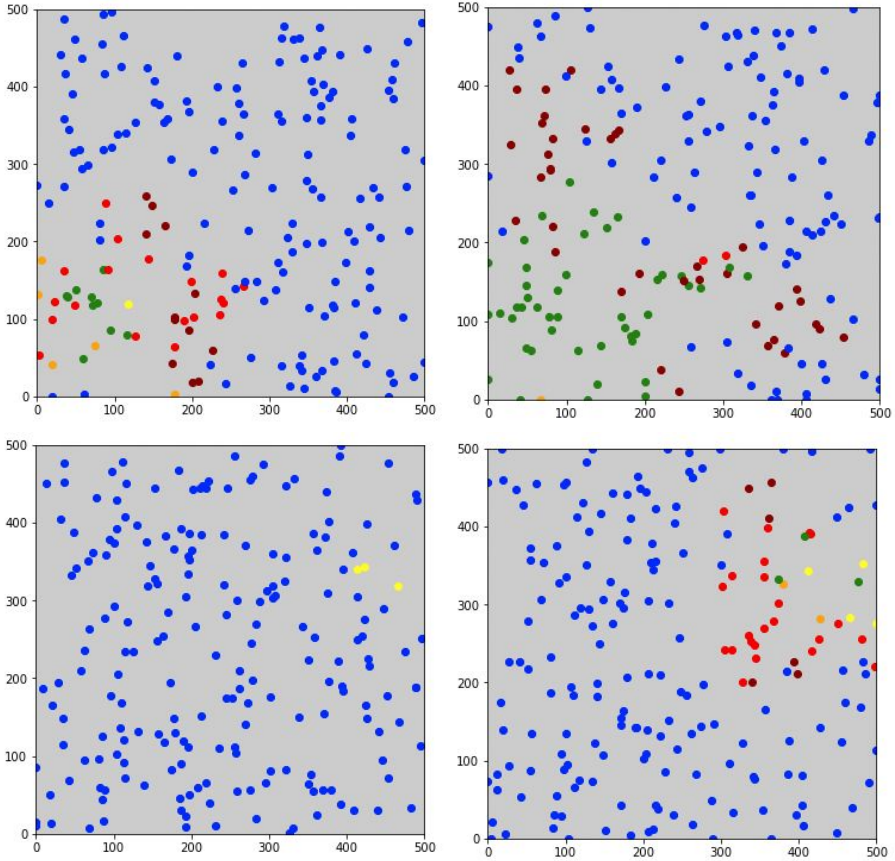
- **How well can an agent-based model represent a mutating pathogen?**
- Real-world viruses mutate frequently, upon transmission between host cells.
- More successful mutations tend to be passed on more frequently, evolving the virus.



https://www.cdc.gov/coronavirus/2019-ncov/variants/variant-classifications.html

https://cov-lineages.org/lineage_list.html

Models



- Agent-based modeling is the strategy we used to represent this situation.
- This involves agents which act with a degree of randomness, interacting with an environment.
- Three classes have been defined as parts of this model.
- The Environment class holds information about the virus as well as numbers of various types of people.
- The Person class is our main agent, moving around the environment, being infected, recovering, and dying.
- The Virus class holds information about recovery time, mortality rate, and transmission rate. It has several numerical characteristics, which it mutates by changing as the simulation runs.

A Brief Look at Some Code

```
class Virus:
    """
    This is the virus object for the simulation.
    The Virus object spreads between Person objects, and can mutate over time as it spreads.
    """

#-==-==-==-==-==-==-==-==-==-==-==-==-==-

    # This is a typical init function, containing the parameters which will be used in other functions.
    def __init__(self, virus_type, recovery_days, mortality_rate, transmission_rate, infection_distance):
        """
        This function initializes the virus.
        Inputs:
            virus_type: The type of the virus, specially defined.
            recovery_days: The amount of time it takes to recover from the virus.
            mortality_rate: The virus' mortality rate.
            transmission_rate: The virus' transmission rate.
            infection_distance: The distance over which the virus can be transmitted.
        Outputs:
            Packs the inputs into the virus object.
        """
        self.virus_type = virus_type
        self.recovery_days = recovery_days
        self.mortality_rate = mortality_rate
        self.transmission_rate = transmission_rate
        self.infection_distance = infection_distance

#-==-==-==-==-==-==-==-==-==-==-==-==-==-
```

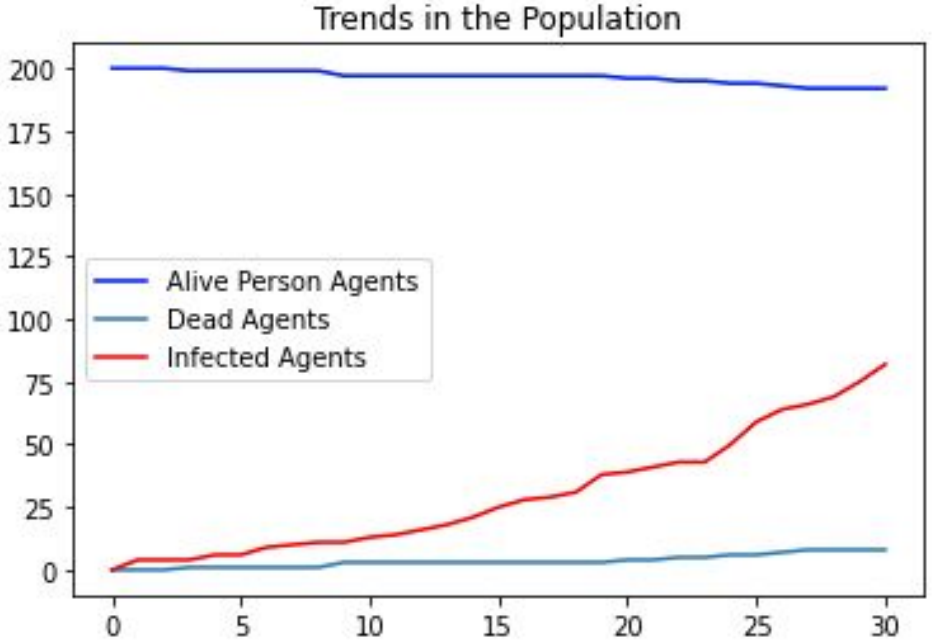
```
class Person:
    """
    This is the Person object, the agent for our agent-based model. The methods here handle the behavior of the agents
    """
#-==-==-==-==-==-==-==-==-==-==-==-==-==-

    # This is a standard init function, with various parameters to be used in the other functions as described therein.
    def __init__(self, x, y, infected = False, virus_type = None, immune = False, xmax = 100, ymax = 100):
        """
        This function initializes the Person.
        Inputs:
            x: The x value of the person
            y: The y value of the person.
            infected: A boolean describing whether the Person is infected.
            virus_type: Stores the type of Virus contained within the Person.
            immune: A boolean describing whether the Person is immune to infection.
            xmax: The maximum x-value for the Person object.
            ymax: The maximum y-value for the Person object
        Outputs:
            Packs the inputs into the Person object.
        """
        self.x = x
        self.y = y
        self.infected = infected
        self.virus_type = virus_type
        self.alive = True
        self.immune = immune
        self.days_infected = 0
        self.xmax = xmax
        self.ymax = ymax

#-==-==-==-==-==-==-==-==-==-==-==-==-==-
```

```
class Environment:
    """
    This is the environment for the agent-based model. The majority of the simulation is handled in the methods here, calli
    """

#-==-==-==-==-==-==-==-==-==-==-==-==-==-

    # Once again, a standard init function.
    # Note that the standard size of the environment is now 100 by 100.
    # This is the code that actually establishes the environment.
    def __init__(self, xmax = 100, ymax = 100):
        """
        This function initializes the environment.
        Inputs:
            xmax and ymax: Maximum x and y values for the environment. Default to 100 by 100.
        Outputs:
            Sets up various quantities needed for the simulation, including starting with no virus, no deaths or infections,
        """
        self.xmax = xmax
        self.ymax = ymax
        self.people = []
        self.virus = None
        self.dead_count = 0
        self.infect_count = 0
        self.environ = np.zeros((self.xmax, self.ymax, 3))
        self.environ[:, :, 0] = 0.8
        self.environ[:, :, 1] = 0.8
        self.environ[:, :, 2] = 0.8

#-==-==-==-==-==-==-==-==-==-==-==-==-==-
```

```
# Necessary inputs; nothing too exotic or unexpected.
import random
import matplotlib.pyplot as plt
import numpy as np
import time
from IPython.display import clear_output
```

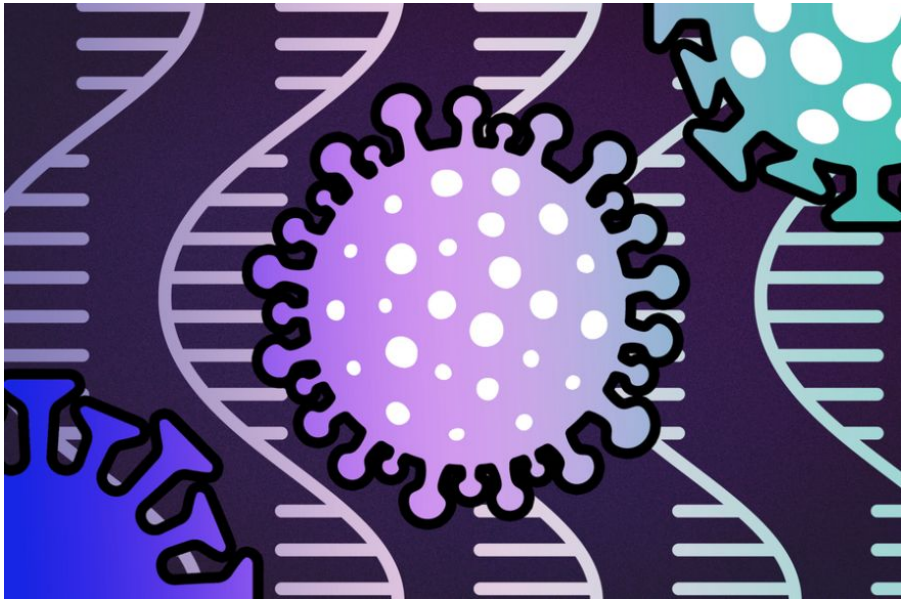
Computational Techniques

- Visualizations were produced showing the model's trends over time.
- Additionally, as a demonstration of statistical methods, several runs of the simulation were done.
- The results could be averaged to get an idea of how pathogens behave in a population.



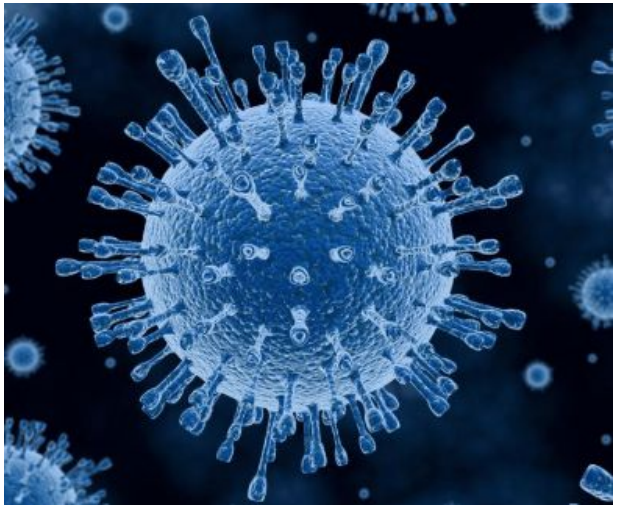
Difficulties and Complications

- Initially, there was some difficulty getting the agents to die or recover.
- The agents could move beyond the bounds of the graph.
- Implementing evolution was also a challenge because of the way the virus is defined in the model–this required simplification.
- There was some debate about the type of conclusions to get–implementing evolution vs. exploring different parameters in a non-evolving situation.



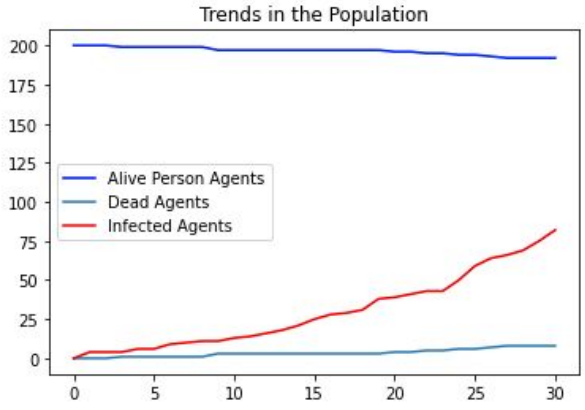
Simplifications and Assumptions

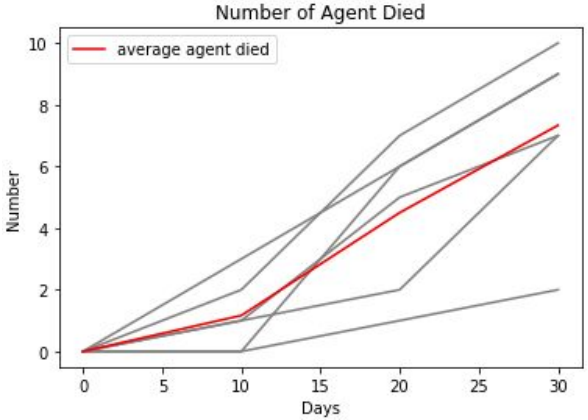
- Once recovered, an agent is forever 100% immune to all strands of the virus.
- Agents cannot leave the environment, and no new agents enter.
- No symptoms are expressed except reduced movement.
- Close proximity is necessary to spread the virus.
- Infected agents are not treated differently or isolated.
- All mutations change the way the virus is expressed.
- Real data for transmission rates is hard to find–assumptions had to be made in that regard.
- Only one strand of the virus exists in the model.
- The genes of real viruses are complicated. (See https://news.mit.edu/2021/map-sars-cov-2-genome-0511.)

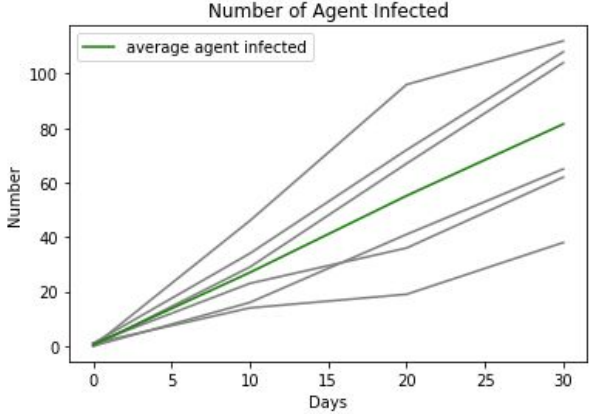


Conclusions

- In the end, a satisfactory simulation of a mutating pathogen was produced.
- This model was versatile enough to produce notably different results over multiple runs, which as seen below could be averaged for statistical purposes.
- The model did involve numerous simplifications, however.
- Given more time, this model could reasonably be further developed to reduce these simplifications and increase realism.







Two runs of the simulation...

https://drive.google.com/file/d/1kaOv2-WjaD9BUoixRiqcoB9Nzrm8S7bS/view?usp=sharing

https://drive.google.com/file/d/1a9B0CcbNxA_dtEx9Gi4pRcnURdRqq9JL/view?usp=sharing

(Some Sources)

- https://coronavirus.jhu.edu/data/mortality (<- I am using this data to simulate, Taekyu) US fatality rate of Covid : 1.1 percent
- https://virologyj.biomedcentral.com/articles/10.1186/s12985-021-01609-w <- I am using this as the transmission rate(1.12 percent)
- https://www.massgeneral.org/news/press-release/how-severe-is-the-sars-cov-2-ba2-subvariant-compared-to-earlier-subvariants#:~:text=Mortality%20rates%20were%200.7%25%20for%20Delta%2C%200.4%25%20for%20the,variant%20compared%20with%20Omicron%20BA. (Omicron, Omicron BA 2 and Delta mortality rate)
-