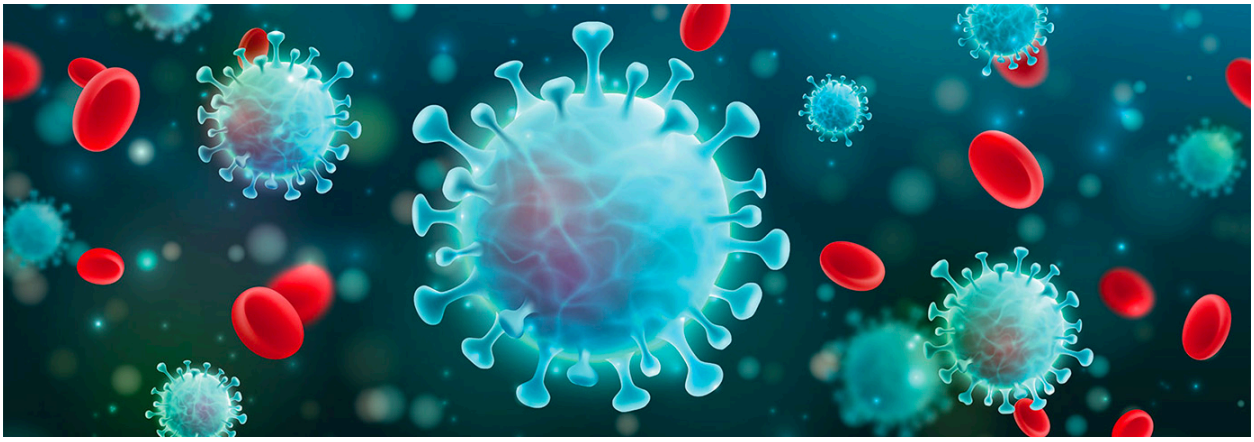
VIRUS TRANSMISSION SIMULATION

INFO6205: Final Project



Oliver Rodrigues (001566167)
Madhurima Chatterjee (001003806)

Introduction:

COVID-19 is a highly transmissible disease that was declared a global pandemic on March 11, 2020, by the World Health Organization. The disease is caused by a strain of coronavirus, namely, severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), and is highly infectious, resulting in more than 56 million cases worldwide as of November 19, 2020. While originating in Wuhan, China, in December 2019, the disease has spread to more than 200 countries of the world by now [3]. Countries have taken non-clinical restrictive measures (e.g., lockdown) as the primary approach to contain the virus's outbreak since effective clinical measures are yet to be found. This disease has heavily affected the economies of most countries, leading to the global coronavirus recession or the Great Shutdown

The SARS-CoV-2 virus spreads through close contacts, i.e., when a susceptible person inhales droplets coming from an infected individual through coughing or sneezing. It can also infect people if they touch their eyes, nose, or mouth after having physical contact with contaminated surfaces. This nature of transmission has given rise to various preventive practices, referred to as non-pharmaceutical interventions (NPI), such as wearing masks, personal protective equipment (PPE), washing hands frequently, staying home, and avoiding gatherings.

To model the transmission of this disease, many mathematical models have been reported in the literature such as Susceptible-Infected-Recovered (SIR) epidemic model and Susceptible-Infected-Susceptible (SIS) epidemic model. In parallel to the above-mentioned models and variants thereof, there are attempts to develop agent-based models (ABM) in the literature with different aims and goals. Agent-based models have been used to model various diseases for a long time. Thus, agent-based models have become rather popular to model the spread of COVID-19 and analyze various ways to approach the issue.

Aim of the Project:

Simulate the spread of SARS-COV-2, the virus responsible for the Covid-19 pandemic, and compare that with SARS-COV-1 the virus responsible for the 2002 SARS epidemic.

We are taking the following factors into consideration
- R factor of the virus to be simulated
- Usage and effectiveness of masks
- Prevalence of testing and contact tracing
- Availability and efficacy of the vaccine
- Barriers to entry
- Three different environments (Open area, Central hub, and Multiple Communities)

Project Details:

We have used Python for building the simulation model. The GUI was built using Tkinter while the movement of objects was simulated using turtle library. Charts are built using Matplotlib. We are using the agent-based method of modeling the spread of the virus.

Agents are artificial individuals programmed to perform predetermined actions. Each of these agents can have their behavior, they can collaborate and compete with other agents. This method can be very powerful while simulating the spread of a virus as it can help us model complex behavior and patterns. Although each of these agents performs simple actions, all of them combined can help us build a real-life situation

Usually, an agent-based model involves a set of agents A having a certain behavior. A set of rules are applied to change its position, state, or relationship with other agents. These rules consider a relation of conditions imposed by other agents (specific agents) or local influences (neighbor agents). This process is repeated until a determined stop criterion has been reached.

Model description:

Our proposed model has 3 different kinds of agents, Susceptible, Infected and Recovered.
Susceptible agents are uninfected agents but are susceptible to be infected
Infected agents are those that are already infected
Recovered agents represent those infected individuals who have either recovered or are dead.
Agents are infected randomly based on the initial infection percentage entered by the user. Each infected agent has a certain probability of infecting a susceptible agent based on multiple factors like R0 or base R factor of the virus, vaccination, usage of masks, what stage of infection the person is at(The initial few days being highly contagious in case of covid-19). So each virus has a base infection probability which will be used to calculate successive probabilities based on the above-mentioned factors

Probability:

Probability is the core tenet upon which this simulation is built upon. Every event or interaction has a probability which is used to take a simple yes or no decision. We use a pseudo-random number generating function producing a uniform distribution of values from 0 to 1, random.random() in the case of python which helps us compare with the probability of our event to take these yes/no decisions. When we combine these multiple yes-no decisions we can simulate a complex behavioral pattern.

Random Movement of Agents:

We use three different rules to move each agents
Rule 1: Each agent has a certain displacement probability with values ranging between 0 and 1. Based on this probability we decide whether the agent move or not

Rule 2: Once the decision has been made regarding moving the agent we have two options
A: Move short distance
B: Move long distance
Now this is calculated by using what we call social distancing factor which is provided via the config. It has values ranging between 0 and 1. A social distancing factor of 0 means that all agents which are probable to be displaced will move long distance and social distancing factor with value of 1 means that all agents will only move short distance. This is occurs when lockdown is imposed.
So if **sd_factor** is the value of social distancing factor **PA or** probability of moving short distance is **sd_factor** and **PB** or probability of moving long distance is **1-sd_factor**

Environments for Random Movement:

We have three different locations for simulating the movement
Open Space - As the name suggests this is just open enviroment where agents move freely. We have used open space to calibrate the value of probability commensurate with R0 of the virus
Central Hub - Usually the movement of the people in a city is based around a central hub or a business district where people visit
Communities - In this scenario we are assuming that people mostly move in their communities and certain amount of people from each community travel to other communities

Base infection probability:

As mentioned above each of our viruses's has a base infection probability which is calibrated using the R0 factor for the virus. The **R0** value for covid-19 is about 2.5 and about 2.2 for SAR-COV-1. This is the base reproduction number of the virus and it can increase or decrease based on multiple factors including vaccination, masks, social distancing, contact tracing etc. **Rt** is used to denote effective value of R. Rt>1 means that the infection is still growing. Rt<1 means that the infection has a negative growth. In addition the contagiousness behavior also defers from virus to virus. For SARS-COV-2 or covid 19 virus a person is most **contagious 2-3 days** before the incubation period ends. The incubation period is between 2-14 days. We are assuming 7 days as

incubation period on average. From there on it goes decreasing. Hence our model simulates this probability in a manner that a Covid 19 infected person has the highest infection probability 3 days before end of incubation. From there on it decreases by 5**%** everyday.

In contradiction to SARS-COV-2, SARS-COV-1 infected person is not contagious until the end of incubation period and is most contagious in the second week of infection. The incubation period is about 2-7 days. We are assuming that peak infection occurs on 12th day of the infection and from there on it starts decreasing by 5 percent everyday.

Infection Distance:

Along with infection probability another important factor is infection distance. For covid-19 it is 6 feet while for SAR-COV-1 it is 3 feet. We are assuming infection distance of 7 units for SARS-COV-2 and 3 units for SARS-COV-1

Usage and effectiveness of contact tracing and quarantine:

Contact tracing can be very effective when coupled with lockdown or restriction of movements. Contact tracing is the practice of tracing all the individuals an infected person has come in contact with. All these people are advised to quarantine themselves. In an ideal world all contacts will quarantine themselves and hence help us break the chain. However that's not what happen and many people evade quarantine. In order to take this factor into consideration we have introduced **quarantine_probability** to model people who do not quarantine. In addition an infected person won't be able to recall all of his contacts correctly. It's easier remember people we have met in indoor environments while we tend to forget those whom we've met outdoors. To simulate this scenarios we have introduced **contact_tracing_efficiency** factor. A contact_tracing_effiiciency of 0.8 mean about 80% of contacts will be traced

Asymptomatic patients and contact tracing:

One of the most difficult aspects of contact tracing is finding asymptomatic individuals. Since they don't have any symptoms, they do not get tested and hence freely roam around being contagious without having any knowledge of it. To model this factor we have introduced asymptomatic_testing_probability parameter to model percentage of symptomatic individuals who don't get themselves tested and act as super spreaders.

Availability and efficacy of vaccines:

Vaccines are the most effective tool against the virus. Different vaccines are available around the world with efficacy ranging from 64% to 95%. What this means is person getting a vaccine with efficacy of 64% is 64% less likely to get the virus. If a person is vaccinated his effective probability of getting the infection reduces based on the efficacy of vaccine. However we know that we are facing major supply issues for vaccines all

around the world. To simulate this issue we have introduce **vaccine_probability** factor to simulate percentage of people getting vaccinated.

Usage and effectiveness of masks:

From the beginning of the pandemic masks have been a very important tool in protecting ourselves from the virus. We are using **mask_probability_factor** to simulate mask usage. An infected person wearing a mask will on an average reduce **infection_distane** by half. There are three common scenarios when we think of simulating usage of masks.
S-1: Infected person wearing mask while susceptible doesn't
In this case if a susceptible person is outside half the infection radius he reduces his infection probability by half
S-2: Both infected and susceptible wearing masks
In this case if a susceptible person is outside half the infection radius he reduces his infection probability by 95%

S-3: Only susceptible wearing mask
In this case if a susceptible person is outside half the infection radius he reduces his infection probability by half

Time:

Each iteration corresponds to one unit of time. We are defining time_conversion_factor in config which has a value of 24. So 24 units of time or 24 iterations equal one day. We use this time conversion factor to convert time in the application to days

Implementation:

As mentioned above python was used for building the model. Below is description of algorithm used

1) Get inputs from GUI
2) Construct screens based on the environment selected
3) Create people objects based on the count specified and assign them random locations in the specified area. Assign x and y limits to person objects so that they don't move outside that
4) While runSimulation not equal false
5) Simulate movement:
    for person in population
      If probability of displacement:
        if probability of short distance:
          move short distance within specified x and y limits

        if probability of long distance:
          move long distance within specified x and y limits
    update infection status for person
6) Calculate infections based on newly assigned coordinates of people
    for people in susceptible:
      For people in infected:
        infection_status = calculate infection status based on effective probability
        if infection_status = true:
          infect susceptible
7) Filter infectious and susceptible from population
8) Calculate value of R if time % 7 == 0
9) If contact tracing true:
    Update close contacts based on proximity
    Trace contacts
10) Plot graph of SIR percentage stack plot and R value graph

The algorithm has quadratic worst case growth

Movement of objects:
We are using turtle for simulating movement of objects

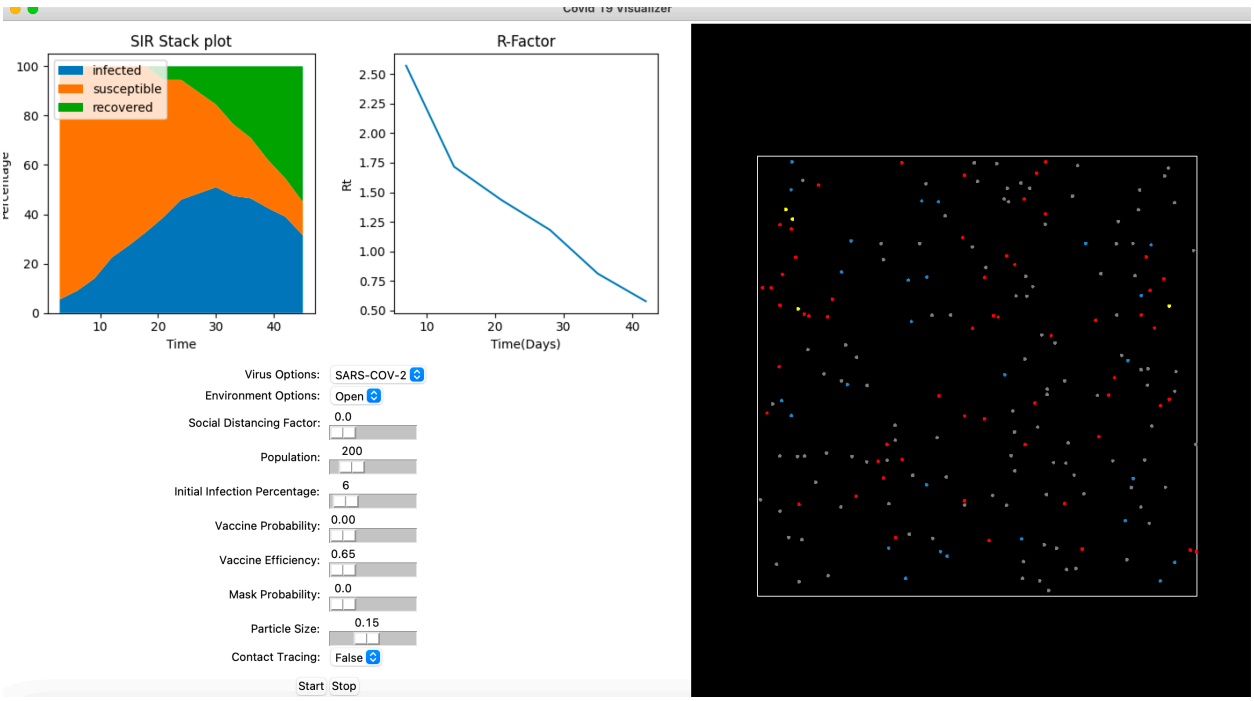
Turtle is a library provided by python
Each turtle object is created and assigned to person object
This object can be used to move the turtle to any desired coordinate based on our runtime calculations
We can also use this turtle object to change color of the turtle

Calculating Value of R:
R factor is the amount of people a person infects on an average

We calculate value of R every 7 days in a 28 day window
The value of R gets improved as we get more data(Typically after 28 days)
Below is our algorithm to calculate value of R:
1) For each infected person in population store the value of id of person it was infected by
2) Create of map of infected person and all the people they have infected
3) To calculate R for any time window select lower bound and upper bound of the window
4)for all infected people, if the person was infected within our time window then
infections++
if person has infected other people:
total infected + = people infected by the person

R values = total infected / infection

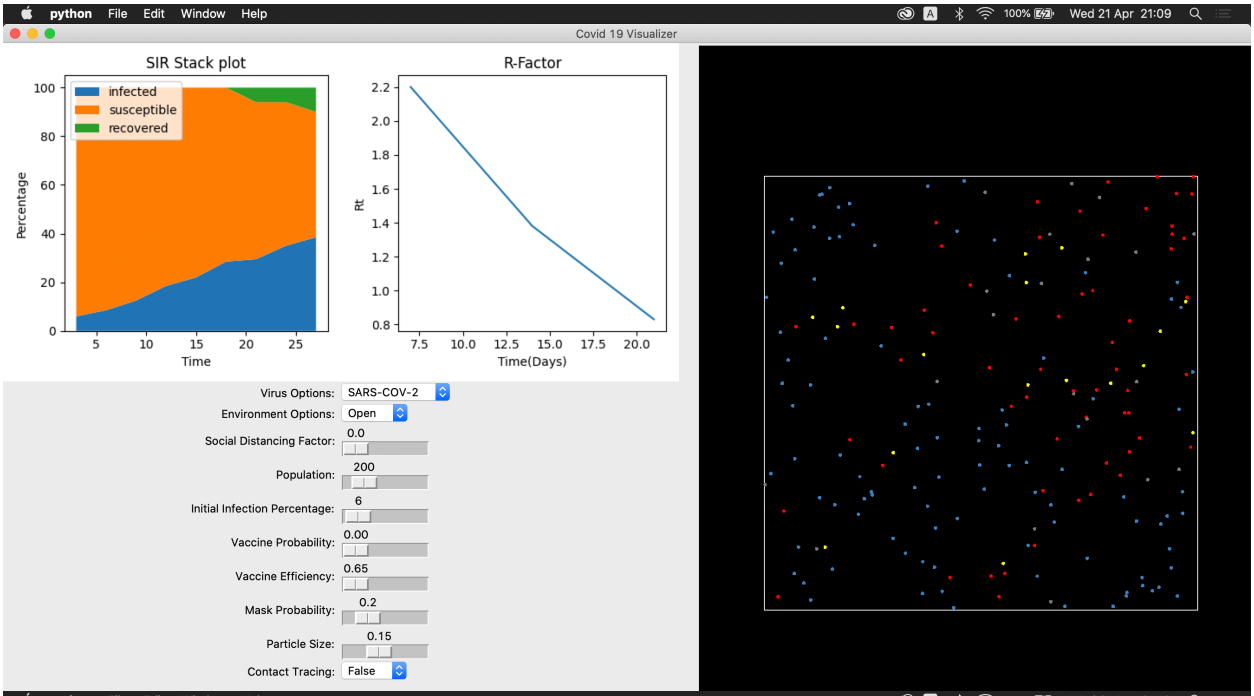
Charts:
We have used matplotlib for charting. We are plotting two plots. One is the SIR stack plot for percentages and the other is the effective value of R against time

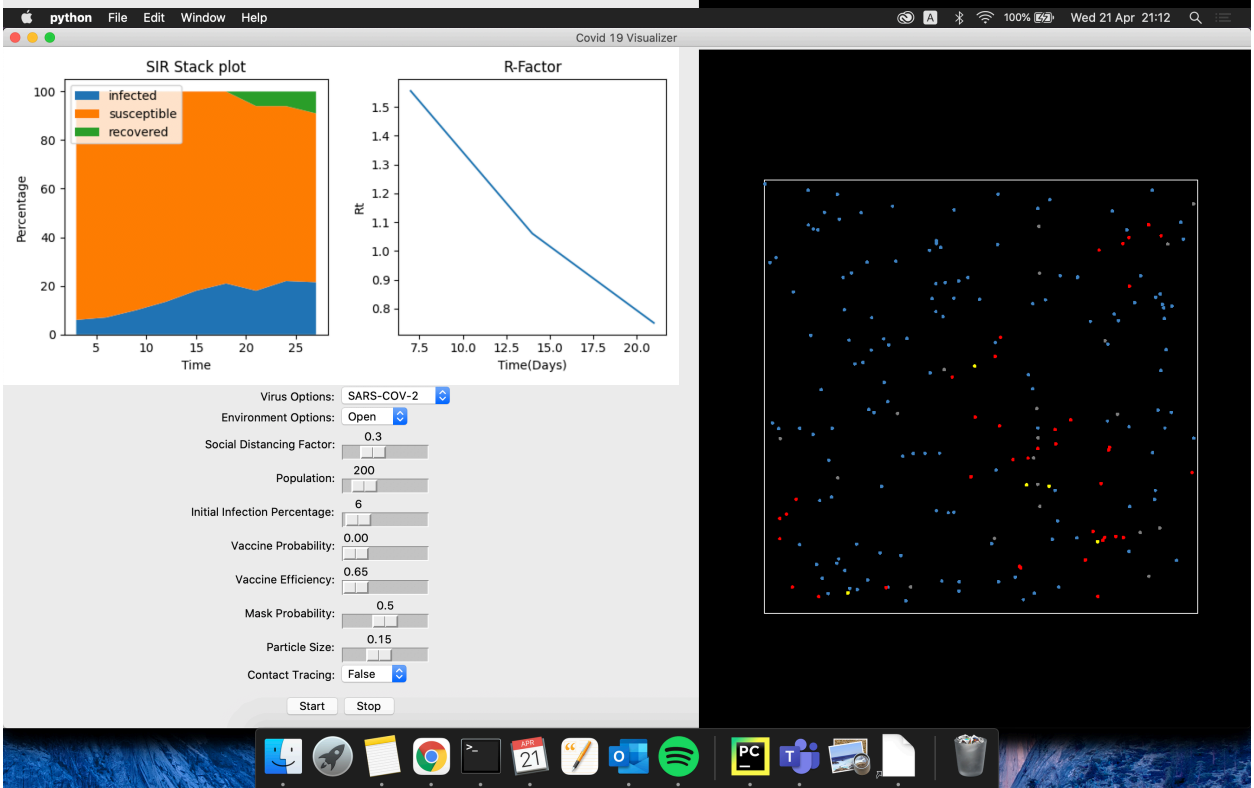


Output:

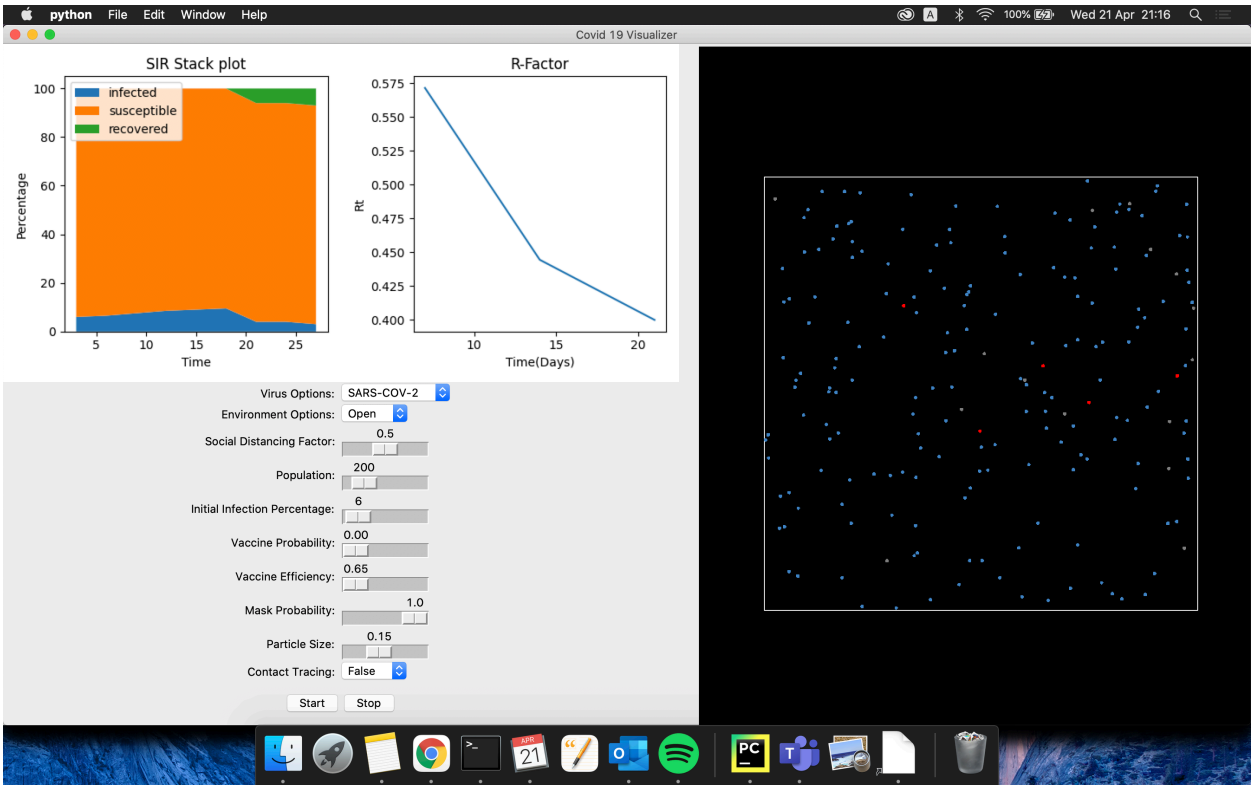
Graphs for different viruses and scenarios:

Virus: SARS-CoV-2 Open Environment with population 200, 20% mask probability, no social distancing and no vaccination.

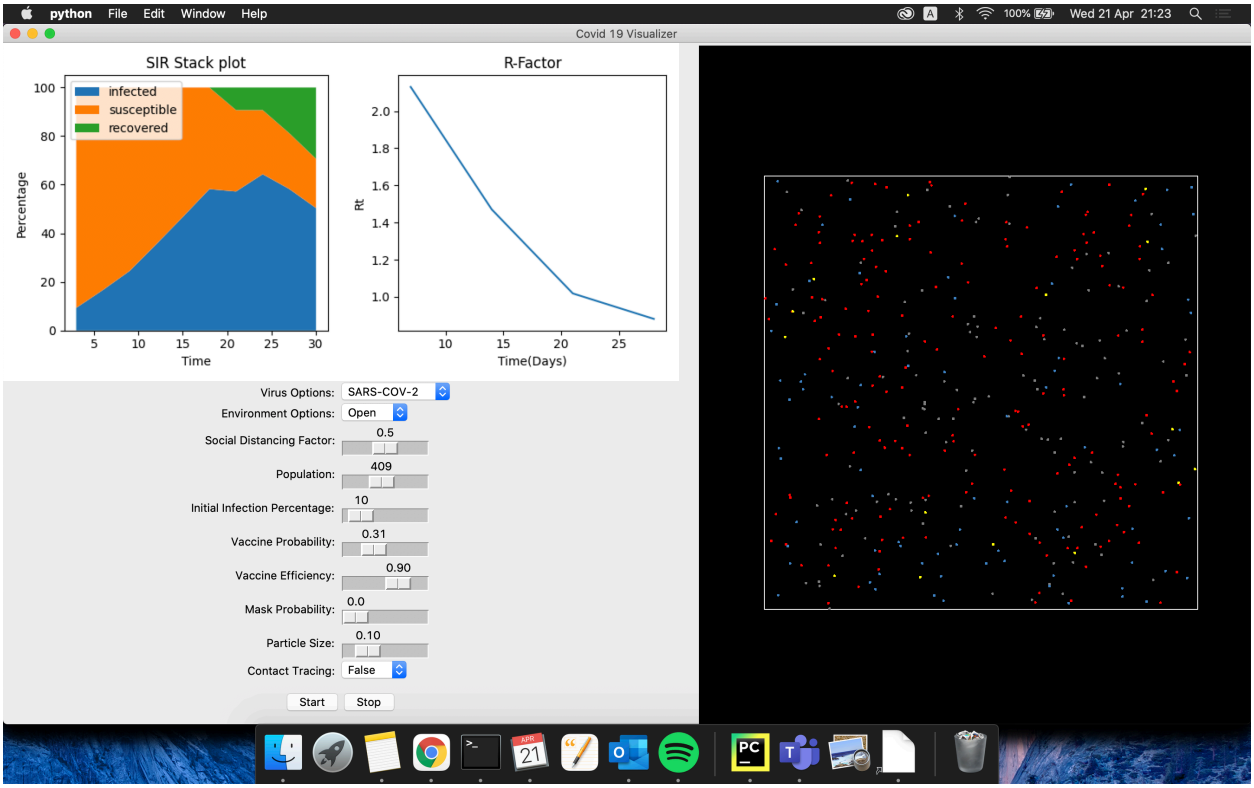




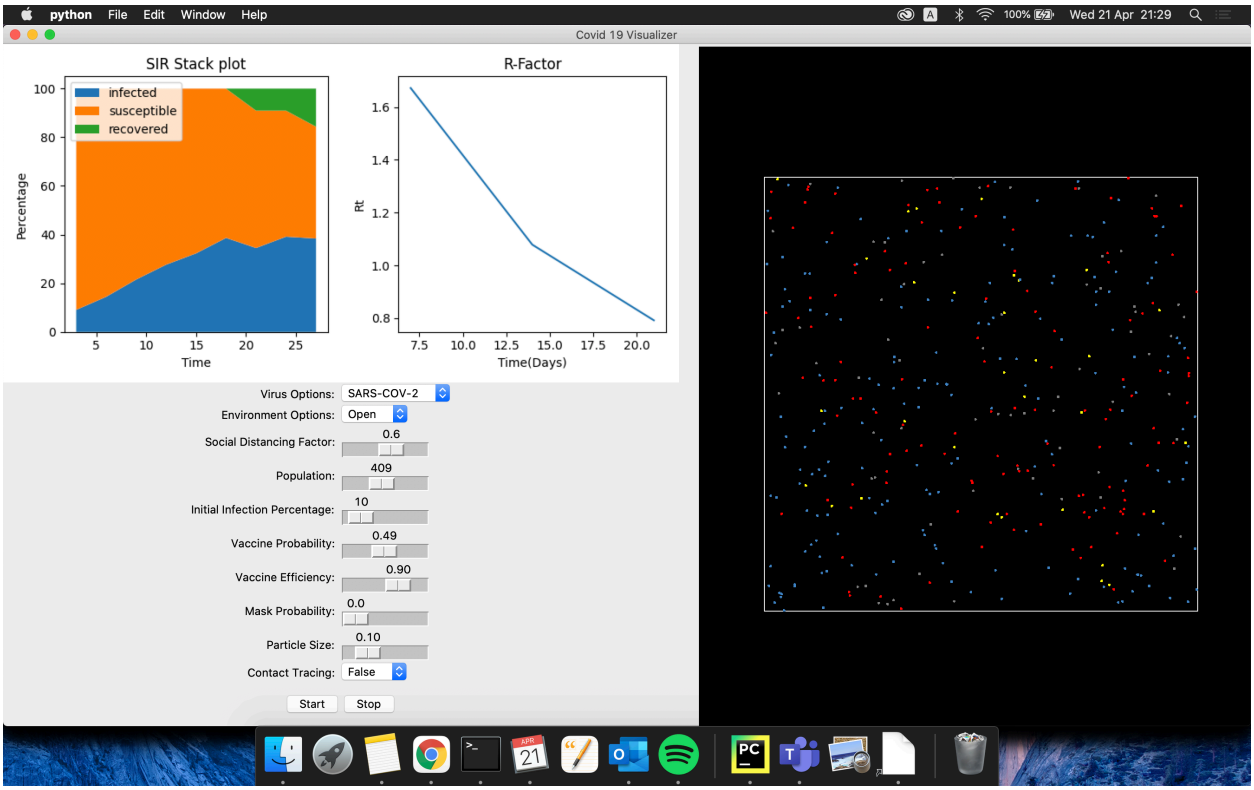
Virus: SARS-CoV-2 Open Environment with population 200, Social Distancing factor 0.3, 50% mask probability, and no vaccination.



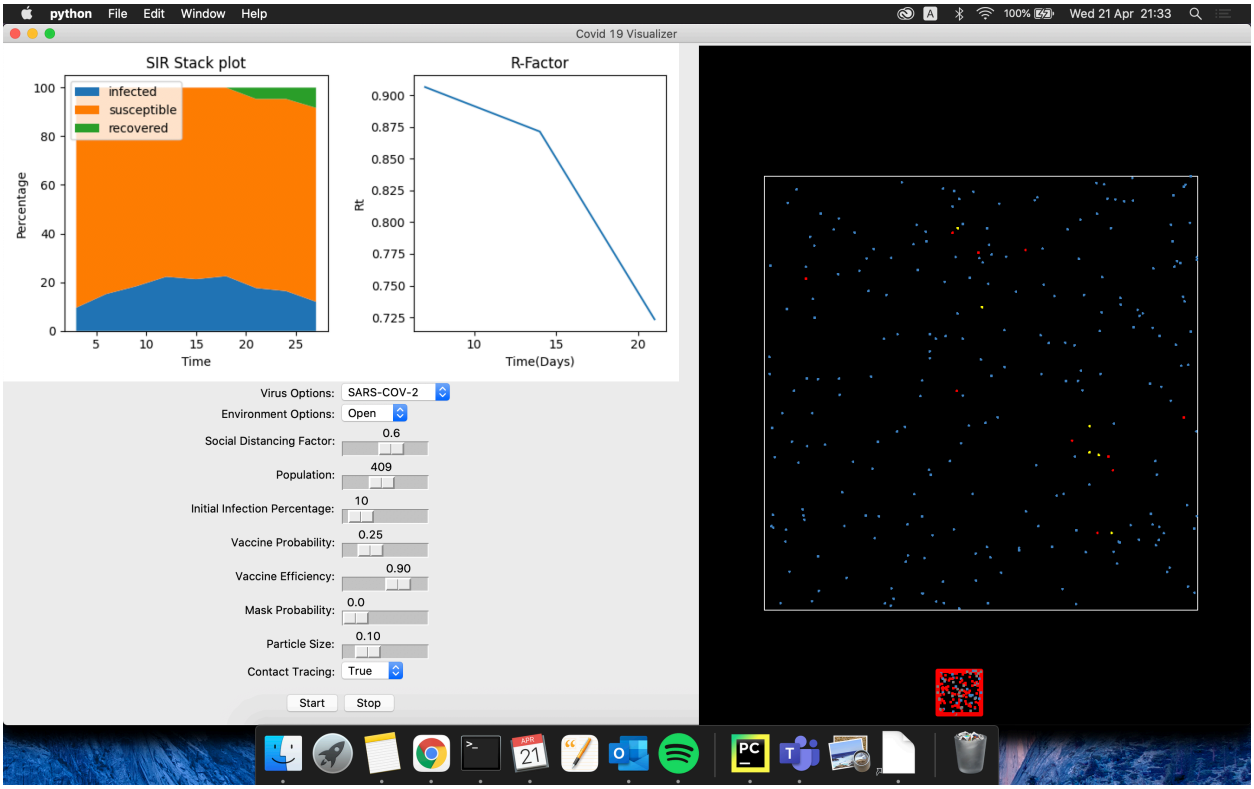
Virus: SARS-CoV-2 Open Environment with population 200, 100% mask probability, 50% social distancing and no vaccination.



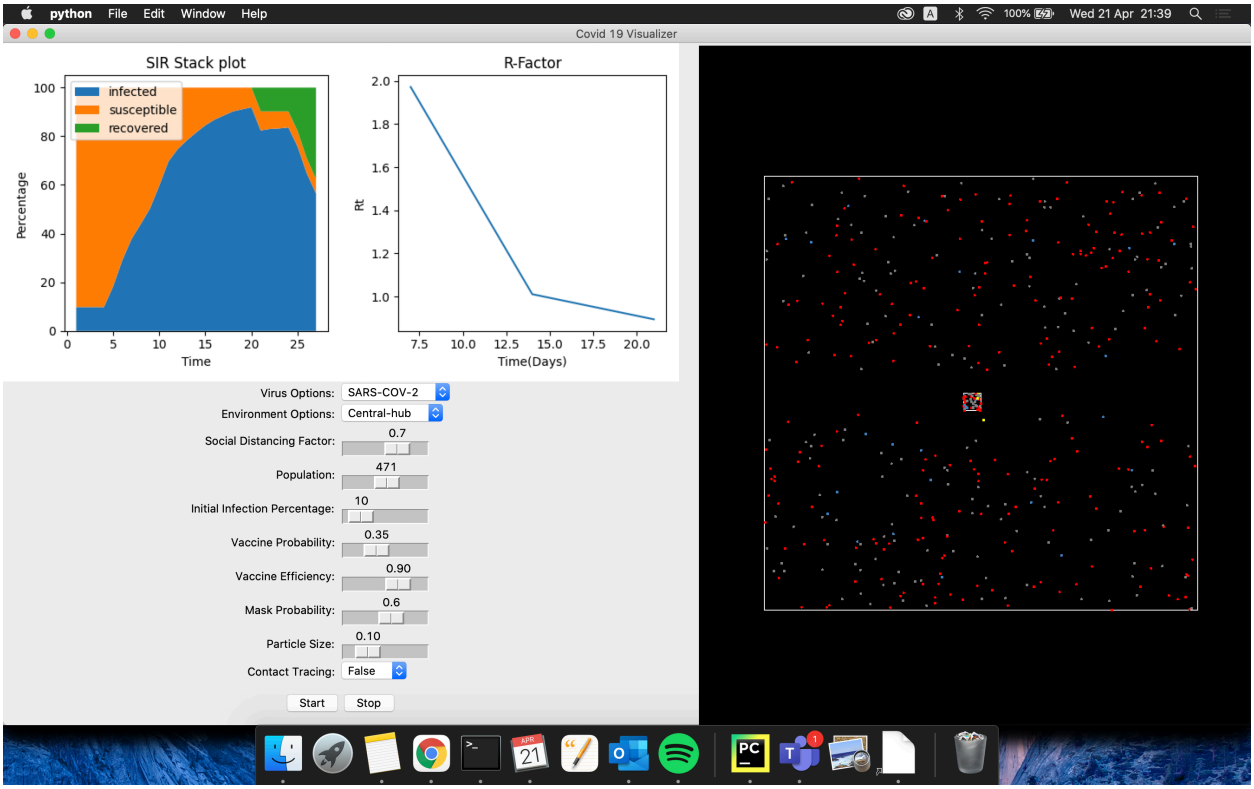
Virus: SARS-CoV-2 Open Environment with population 409, 100% mask probability, 50% social distancing and 31% vaccination with 90% efficiency.



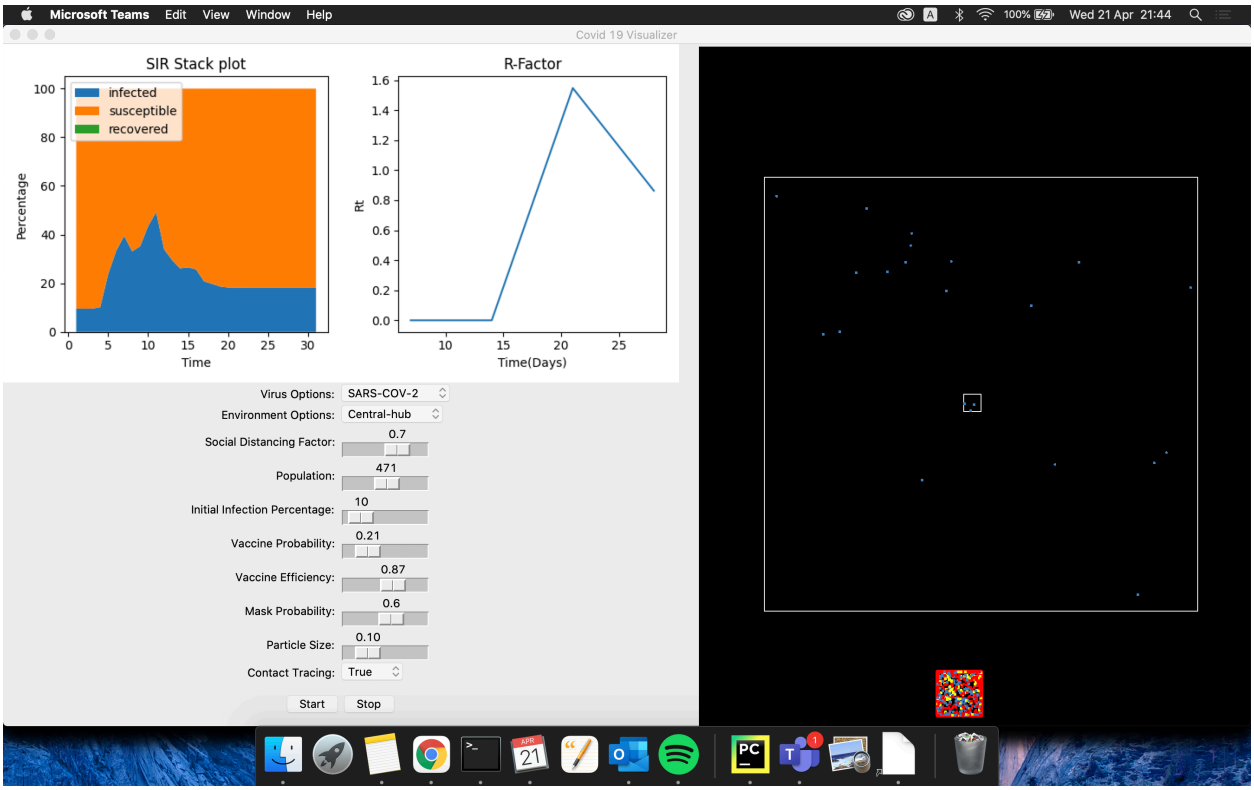
Virus: SARS-CoV-2 Open Environment with population 409, 0% mask probability, 60% social distancing and 49% vaccination with 90% efficiency.



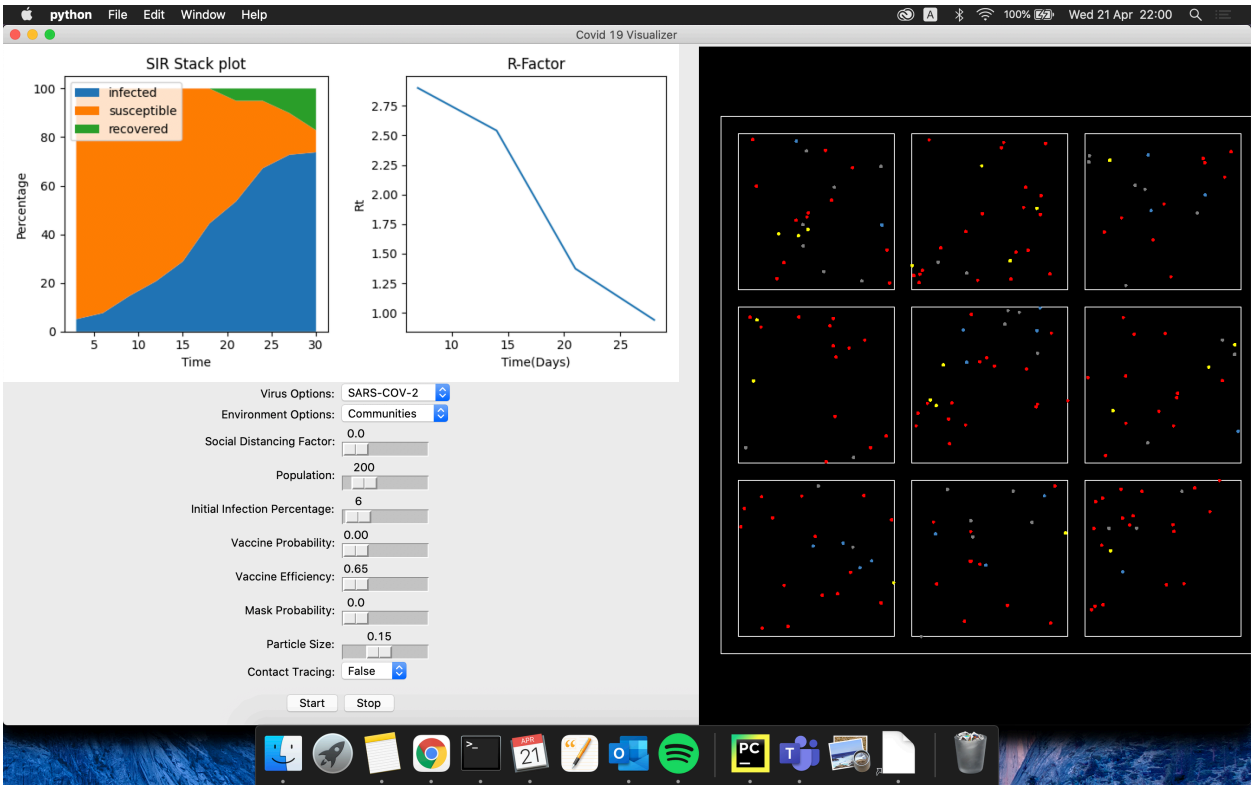
Virus: SARS-CoV-2 Open Environment with population 409, 0% mask probability, 60% social distancing, 25% vaccination with 90% efficiency and contact tracing.



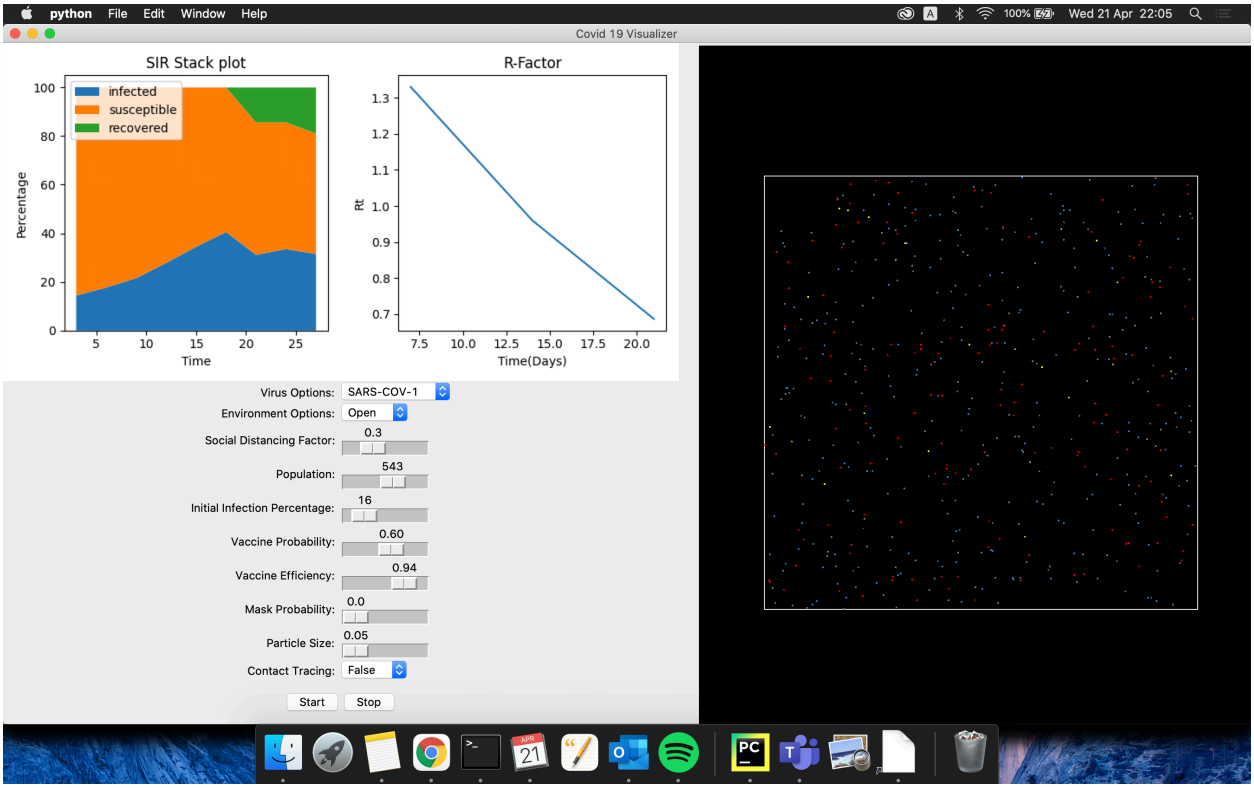
Virus: SARS-CoV-2 Central Hub Environment with population 471, 60% mask probability, 70% social distancing and 35% vaccination with 90% efficiency.



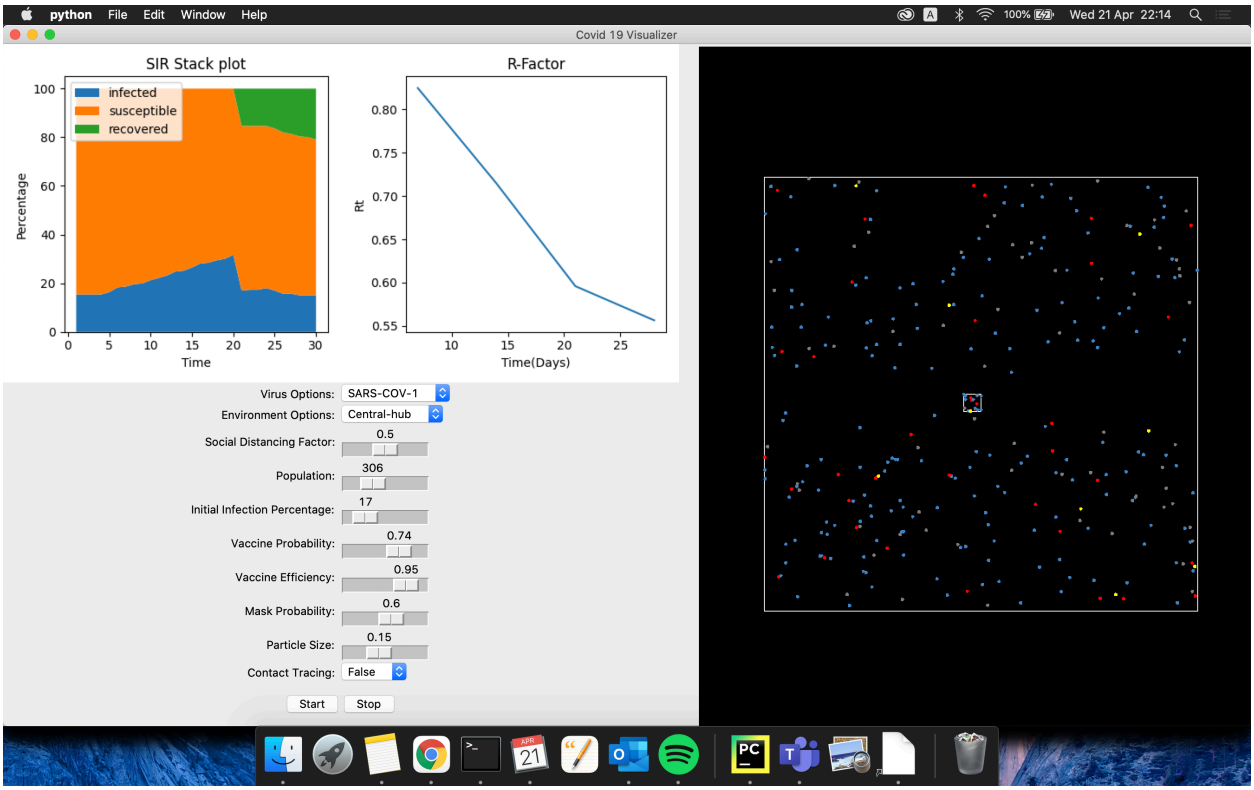
Virus: SARS-CoV-2 Central Hub Environment with population 471, 60% mask probability, 70% social distancing, 21% vaccination with 87% efficiency and contact tracing.



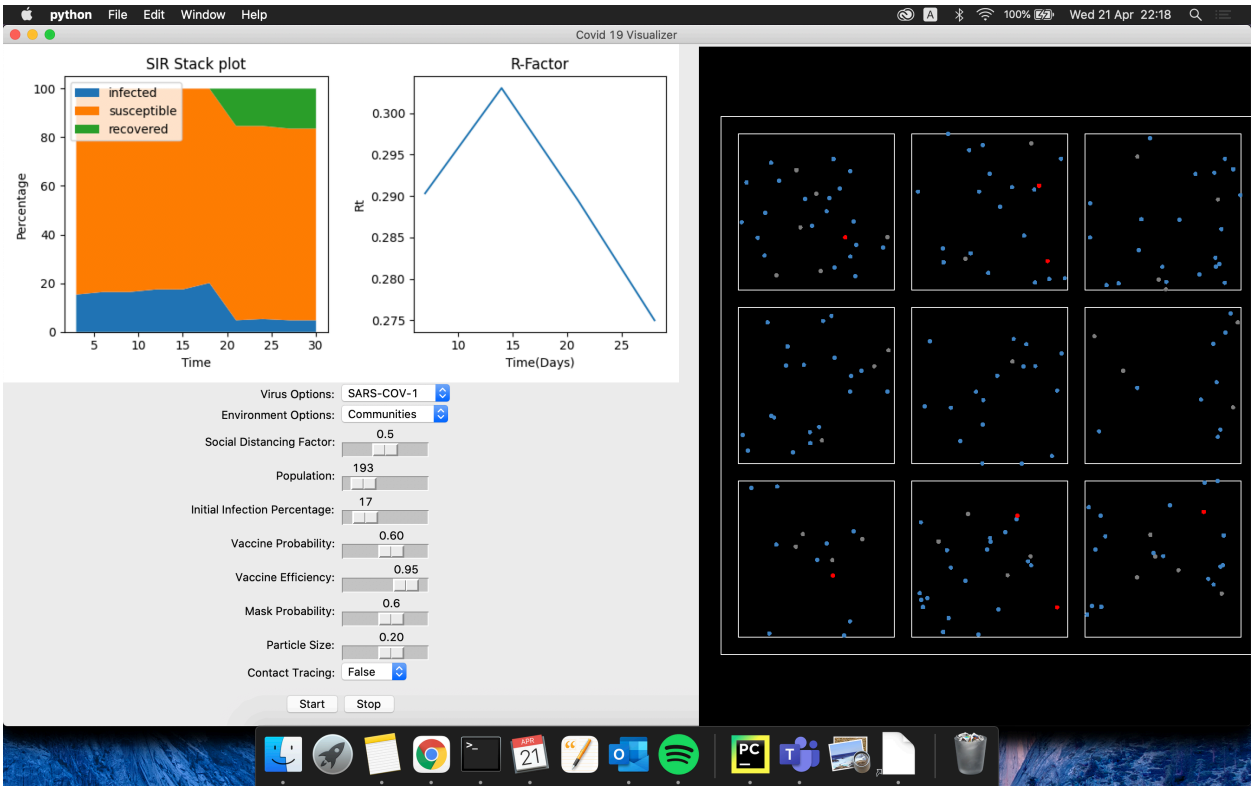
Virus: SARS-CoV-2 Communities Environment with population 200, 0% mask probability, 0% social distancing and 0% vaccination.



Virus: SARS-CoV-1 Open Environment with population 543, 0% mask probability, 30% social distancing and 60% vaccination with 94% efficiency.



Virus: SARS-CoV-1 Central-hub Environment with population 306, 60% mask probability, 50% social distancing and 74% vaccination with 95% efficiency.



Virus: SARS-CoV-2 Communities Environment with population 193, 60% mask probability, 50% social distancing and 60% vaccination with 95% efficiency.

Unit tests Running:

```
Terminal:  Local  +
(Covid-19-Visualizer) oliverrodrigues@Olivers-MacBook-Air Covid-19-Visualizer % coverage run -m --source src unittest discover
..0.15
......0.29924999999999996
...{0: <Mock name='mock.Person()' id='140728640498464'>, 1: <Mock name='mock.Person()' id='140728640498464'>, 2: <Mock name='mock.Person()' id='140728640498464'>, 3: <Mock nam
e='mock.Person()' id='140728640498464'>}
.150
.499
.102
.....lol {23: True} {}
lol {} {23: True}
{45: True} {} <Mock id='140728640596960'>
...../Users/oliverrodrigues/Programming/Covid-19-Visualizer/tests/test_screens.py:13: DeprecationWarning: Please use assertTrue instead.
  self.assert_(draw_pen.fd.callcount, 4)
..
----------------------------------------------------------------------
Ran 26 tests in 0.025s

OK
(Covid-19-Visualizer) oliverrodrigues@Olivers-MacBook-Air Covid-19-Visualizer % coverage report -m
```

Coverage by Line

```
OK
(Covid-19-Visualizer) oliverrodrigues@Olivers-MacBook-Air Covid-19-Visualizer % coverage report -m
Name               Stmts   Miss  Cover   Missing
-----------------------------------------------
src/__init__.py        0      0   100%
src/constants.py      15      0   100%
src/helpers.py       171     54    68%   4-42, 45-56, 191-197
src/location.py       21      0   100%
src/movement.py       79      1    99%   62
src/person.py         41     41     0%   2-45
src/screens.py        35      0   100%
-----------------------------------------------
TOTAL                362     96    73%
(Covid-19-Visualizer) oliverrodrigues@Olivers-MacBook-Air Covid-19-Visualizer %
```

Coverage by Branch

```
OK
(Covid-19-Visualizer) oliverrodrigues@Olivers-MacBook-Air Covid-19-Visualizer % coverage report -m
Name               Stmts   Miss Branch BrPart  Cover   Missing
-------------------------------------------------------------
src/__init__.py        0      0      0      0   100%
src/constants.py      15      0      0      0   100%
src/helpers.py       171     54    110      6    67%   4-42, 45-56, 77->73, 88->86, 89->86, 92->86, 96->94, 142->exit, 191-197
src/location.py       21      0      6      0   100%
src/movement.py       79      1     40      6    94%   51->53, 53->exit, 58->exit, 62, 71->83, 92->87
src/person.py         41     41      0      0     0%   2-45
src/screens.py        35      0      2      0   100%
-------------------------------------------------------------
TOTAL                362     96    158     12    73%
(Covid-19-Visualizer) oliverrodrigues@Olivers-MacBook-Air Covid-19-Visualizer %
```

Mathematical Analysis:

The algorithm we have used has a quadratic worst case growth. Since we have to compare susceptible and infected people
We are using python dictionary or map data structure for storing people objects as well as infected, susceptible and recovered

Invariants: We can tweak values of other factors while keeping one factor constant. So any our parameters can act as invariants while other parameters are changed

R Factor Calculation:

As mentioned above R factor is calculated for a particular time window.
Lets says a time window starts at t1 and ends at t2
Let there be n individuals who are infectious within this time range
Let Ki be the Toal people infected by each infectious individual
Then R factor for time window t2-t1 is
Rt = (Summation(Ki) from 1 to n) / n

As we can see from the above graphs in output section Rt decreases as we add different factors like vaccines, masks, social distancing

Since R factor is depended on average number of infections by each person, if the overall case count is reduced obviously R will reduce.

Consider a scenario where the infectious person has base infection probability of 0.3
Now if he is wearing a mask then the probability of infecting someone reduces to 0.15 even when the susceptible isn’t wearing mask and is within 6 feet distance.

Conclusion:

The most important conclusion from the simulation is that the R factor is not very effective when we try to compare two viruses. In our case, both SARS 2002 and Covid 19 had similar R factors but as we know Covid has turned out to be far more dangerous than SARS 2002. R factor is a good number when we want to look at the big picture. It tells us whether an infection is increasing or decreasing. However, a more effective way to control the virus is to consider the **K factor or the measure of overdispersion**. K factor for covid is 0.1 while that for SARS 2002 is 0.16. What that means is for covid 80% of infections are caused by 10% of people. This explains various super spreading events we've had with covid. If we want to control the virus we need to control these super spreading events. For example, if a community or environment has a low value of k then we can impose a lockdown so that no one goes in or comes out. This can help stop super spreading. Having said that we can still build a certain estimate of virus spread by looking at these R simulations. Using Agent-based modeling has helped us model complex behavior and this model coupled with the K factor can give us a better estimate compared to a compartmental model.