Pandemic Simulator

Cezar-Alexandru Tulceanu

1. Introduction

The purpose of this project is to analyze the evolution of an infectious disease in a small town, according to the following rules:

1. The town has N inhabitants, of whom, initially, K are infected with a disease.
2. Every morning, any two people meet exactly once. If one is infected and the other is not, the latter has a probability of $p\%$ of becoming infected.
3. People can recover. Every evening, a sick person has a probability of $q\%$ of recovering.
4. We want to find the expected value of the number of infected people after Z days.

2. Theoretical Calculation of the Value

The algorithm described below calculates the exact expected value of the number of people who are infected after Z days, using only the problem's parameters.

First, the algorithm calculates the probability of a person becoming infected on a morning when i people are infected, using the following formula:

```
infection_chance[i]=1-(1-p/100)**i
```

Now, the algorithm calculates the values of the matrix $inf[i, j]$, which represents the probability that if we start the day with i infected people, we will have j infected people by noon, using the following formula:

```
inf=np.zeros((n+1,n+1),dtype=float)
for i in range(1,n+1):
    for j in range(1,n+1):
        if j<i:
            inf[i,j]=0
        else:
            inf[i,j]=scipy.special.comb(n-i, j-i)*(
               infection_chance[i]**(j-i))*((1-infection_chance[i
               ])**(n-j))
```

The idea behind the formula is that we are looking for a subset of $j - i$ healthy people and calculating the probability that these specific people get sick, and that no one else does. The binomial coefficient $\binom{n-i}{j-i}$ appears because we can choose this subset in $\binom{n-i}{j-i}$

ways. Now that we have chosen the subset, we must multiply by $infection\_chance[i]$ for each of the $j - i$ people in the subset and multiply by $1 - infection\_chance[i]$ for every healthy person not in the subset.

Next, the algorithm calculates the values of the matrix $heal[i, j]$, which represents the probability that if we have i infected people at noon, we will have j infected people in the evening, using the following formula:

```
healing_prob=q/100
heal=np.zeros((n+1,n+1),dtype=float)
for i in range(1,n+1):
    for j in range(1,n+1):
        if i<j:
            heal[i,j]=0
        else:
            heal[i,j]=scipy.special.comb(i, i-j)*(healing_prob**(
               i-j))*((1-healing_prob)**j)
```

The idea behind the formula is that we are looking for a subset of $i - j$ sick people and calculating the probability that these specific people recover, and that no one else does. The binomial coefficient $\binom{i}{i-j}$ appears because we can choose this subset in $\binom{i}{i-j}$ ways. Now that we have chosen the subset, we must multiply by $healing\_prob$ for each of the $i - j$ people in the subset and multiply by $1 - healing\_prob$ for every sick person not in the subset.

Now that we have the matrices inf and $heal$, we want to calculate the matrix $dp[i][j]$, which represents the probability that if we start the day with i infected people, we will have j infected people in the evening, using the following formula:

```
dp=np.matmul(inf,heal)
```

The matrix dp is simply the matrix product of inf and $heal$ because the formula

$$dp[i][j] = \sum_{k=1}^{n} \text{inf}[i][k] \cdot \text{heal}[k][j]$$

considers every case where we have k infected people at noon, and this formula is precisely the definition of matrix multiplication.

Now, the algorithm calculates the values of the matrix $final[i, j]$, which represents the probability that if we had i infected people on the morning of the first day, we will have j infected people on the evening of the Z-th day, according to the formula:

```
days=Z
final=np.identity(n+1,dtype=float)
for i in range(1,days+1):
    final=np.matmul(final,dp)
```

The $final$ matrix is simply the dp matrix raised to the power of Z. This is because, analogous to the previous case, matrix multiplication handles every intermediate case of having x_i infected people after the i-th day, for each $i \in \{1, 2, 3, ..., Z - 1\}$.

Finally, the algorithm calculates and displays the sought-after expected value using the formula:

```
initial_cases=K
ans=0.0
for i in range(0,n+1):
    ans+=i*final[initial_cases][i]
print(ans)
```

3. Finding the Required Number of Simulations

To find the number R of simulations required to have an error of ϵ with an accuracy of $a\%$, we will use the following formula derived from Chebyshev's inequality:

$$R = \left\lfloor \frac{\mathbb{V}ar[X]}{\epsilon^2} \cdot \frac{1}{1 - \frac{a}{100}} \right\rfloor + 1$$

where X is the random variable that takes the value i with probability $final[K][i]$ for each $i \in \{0, 1, 2, 3, ..., N\}$.

Furthermore, to derive this formula, we used the following formula for the variance of S_n:

$$\mathbb{V}\text{ar}(S_n) = \frac{\mathbb{V}\text{ar}(X)}{n}$$

Where:

$$S_n = \frac{X_1 + X_2 + \cdots + X_n}{n}$$

Thus, by applying Chebyshev's inequality, the desired accuracy is guaranteed if we choose $n \geq R$:

$$P\left(|S_n - \mathbb{E}[S_n]| \geq \epsilon\right) \leq \frac{\mathbb{V}ar(S_n)}{\epsilon^2} = \frac{\mathbb{V}ar(X)}{n\epsilon^2}$$

Below is the code that displays this required number, R:

```
ans=0.0
for i in range(0,n+1):
    ans+=i*final[initial_cases][i]

expected_value=ans
expected_value_of_square=0.0

for i in range(0,n+1):
    expected_value_of_square+=i*i*final[initial_cases][i]

variation=expected_value_of_square-(expected_value*expected_value
   )
```

```
accuracy_percentage=95

epsilon=0.25

sim_number=((variation/(epsilon*epsilon))/(1-accuracy_percentage
   /100))+1
sim_number=int(sim_number)

print(sim_number)
```

4. Simulation Theory

The N people in the town are represented by a vector with N cells. The number in the i-th cell is equal to 1 if the i-th person is sick and is equal to 0 otherwise.

4.1 A Single Simulation

In an individual simulation, the algorithm follows these steps for each of the Z days:

1. All pairs of people are iterated through. If an infected person meets a healthy one, the latter becomes infected with a probability of $p\%$.
2. At the end of the day, the recovery process is simulated. Each sick person has a probability of $q\%$ of recovering.

4.2 Monte Carlo: The Complete Algorithm

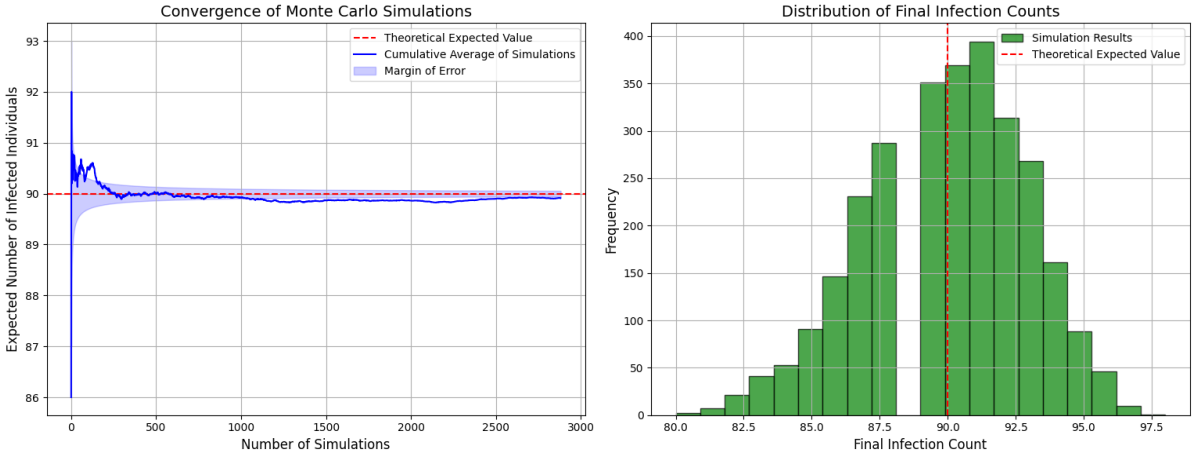
The Monte Carlo algorithm executes multiple simulations to calculate the expected value of the number of infected people. The main steps are:

1. Execute R independent simulations, where R is the number of simulations required to achieve the desired accuracy, given by the formula in the theoretical estimation algorithm described above.
2. For each simulation, the algorithm described in the immediately preceding section is applied.
3. After all simulations, the average value of the number of infected people at the end of the Z days is calculated.

After the R runs, the final result is the average number of infected people, which represents a fairly good estimate of the expected value.

5. Experimental Validation and Visualization

To empirically validate the theoretical model, a Monte Carlo simulation was executed. The results of this simulation are presented visually to offer a clearer understanding of the model's behavior. The analysis is twofold: firstly, to demonstrate the convergence of the simulation's average to the theoretical expected value, and secondly, to show the statistical distribution of the outcomes.



Visual Analysis of Monte Carlo Simulation Results. The graph on the left shows the running average of infected cases converging to the expected theoretical value as the simulation count increases. The graph on the right is a histogram showing the frequency distribution of the final results over thousands of runs.

The graphical analysis presented above confirms the robustness of both the theoretical and simulation models. The convergence plot (left) clearly illustrates that as the number of simulations increases, the experimental average progressively approaches the deterministic value calculated theoretically. This is a direct observation of the Law of Large Numbers in practice. The histogram (right) provides deeper insight into the stochastic nature of the process, revealing the probability distribution of the final number of infected individuals. It shows which outcomes are most likely and quantifies the variance around the expected value, offering a complete picture of the disease's potential impact after the specified period.