Analyzing Quarantine Strategy for COVID-19 by SIR Model

양성진 (오현중학교 1학년)
지도교수: 김종훈 (제주대학교 초등컴퓨터교육전공)
보조강사: 오승탁 (신례초등학교)

Ⅰ. 서론

최근 코로나바이러스감염증-19(COVID-19)가 전 세계적으로 유행함으로써 많은 사람들이 금전적, 신체적 피해를 입고 있다. 이로 인해, 각 정부는 다양한 방역 전략을 준비해 바이러스 확산을 억제하려 노력하고 있다. 예를 들어, 대한민국은 확산을 막는 것에 중점을 두어 강력한 방역 절차를 행하고 있는 반면, 스웨덴 정부는 면역을 생성하는 것에 가중치를 두어 최소한의 방역을 행하고 있다. 그렇다면 COVID-19에 가장 효과적인 방역 수칙은 무엇일까?

많은 전문가들은 이를 연구하고자 수학적 모델을 활용하고 있다. 가령, 가장 간단하지만 실효가 있는 SIR Model부터 좀 더 세분화된 감염 경로를 모델링해주는 SEIR Model을 사용하고 있다. 또한 많은 학자들은 COVID-19에 대한 수학적 분석을 진행하고 있다[1].

반면, 이러한 모델을 사용, COVID-19에 대한 각 방역 전략에 따른 확산 분석은 이뤄진 바는 정부 발표자료 이외에는 찾아보기 어려웠다. 그리고 이 분석이 방역 전략에 대한 수치화된 평가를 줄 것으로 기대되어 방역 전략에 따른 확산 분석의 필요성을 느꼈다.

COVID-19에 대한 여러 방역 전략에 대한 충분한 통찰을 얻기 위해, 연구를 위한 사전 표본 및 분석 표본(시각화 자료)를 계획하였다. 사전 표본(확산을 예측한 표본)은 미래 시점을 기준으로 두기 때문에 기존 기록을 사용하기에는 부적절하며, 위에서 잠깐 다루었던 SIR 바이러스 확산 모델을 통해 생성된 가상 표본을 사용한다. 또한 SIR Model은 간단한 매개변수를 이용해 계획되었기에 표본 양에 따라 정확도가 낮아지게 되는 데, 이를 해결하기 위해 소규모 표본대상(제주도)을 기준으로 분석을 진행한다. 확산 모델의 초기 상태는 2021년 4월 22일 0시 제주특별자치도 발표를 기준으로 두며, 총 인구 수는 2019년 정부 발표를 기준으로 확산 모델을 디자인할 것이고, 확산 모델 예측 범위는 약 1년(360일)으로 설정한다(일정 시간 이후로는 상태가 변화하지 않을 것이다). 분석을 위한 시각화의 경우 단순 출력으로는 분석에 한계가 있어 Python 시각화 모듈 'matplotlib'으로 생성된 그래프를 이용한다. 이러한 시각화 작업이 마무리 되면 각 방역 전략에 따른 확산에 대해 최종 분석을 진행한다.

[1] arxiv.org에서 'COVID-19 infection model'로 검색 시 약 400개의 논문 검색됨

Ⅱ. 이론적 배경

1. Markov-Chain을 바탕으로 한 SIR Model

A. Matplotlib 모듈과 Pyplot API

바이러스 확산을 모델링한 가장 기초적인 수학적 모델은 Markov-Chain based SIR Model(SIR Model로 통칭)이다. SIR Model은 사람들을 Susceptible Individuals, Infected Individuals, Recovered or Removed Individuals와 같은 3가지 부류로 나누어 처리한다. SIR Model을 구현하기 위해, 다음과 같은 사전 정의가 필요하다.

- $S(t)$ is the number of susceptible individuals;
- $I(t)$ is the number of infected individuals; and
- $R(t)$ is the number of recovered(or removed) individuals
- β is the count of contacts per day that are sufficient to spread the disease
- γ is the number of days are required to recover or die

SIR Model의 핵심 원리는 $S(t), I(t), R(t)$를 다음과 같은 Differential Formula

$$\frac{dS(t)}{dt} = -\beta S(t)I(t)$$
$$\frac{dI(t)}{dt} = \beta S(t)I(t) - \gamma I(t)$$
$$\frac{dR(t)}{dt} = \gamma I(t)$$

를 이용해 $S(t+1), I(t+1), R(t+1)$를 구하는 것이다. 그렇다면 100일 후, 200일 후의 $S(t), I(t), R(t)$ 는 어떻게 구해야 할까. 이를 위해 SIR Model에서는 Markov-Chain라는 확률과정[2]을 사용한다. Markov-Chain이란 시간에 따른 계의 상태의 변화를 나타낼 수 있는 확률 과정이며, SIR Model에서는 [Figure 1]와 같이 작용한다.

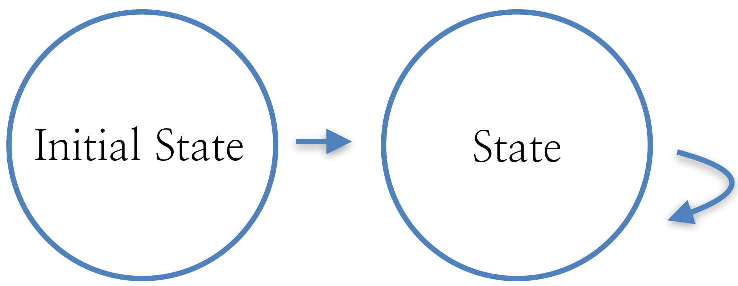

Figure 1

[2] 시간의 진행에 대해 확률적인 변화를 가지는 구조

2. Matplotlib을 이용한 데이터 시각화 처리

B. Matplotlib 모듈과 Pyplot API

Matplotlib 모듈은 정적, 동적, 상호작용 가능한 시각화 자료를 제작하는 포괄적 시각화 도구이다. 파이썬 패키지 도구인 PyPl를 통해 다음과 같은 명령어로 설치가 가능하다

```
pip install matplotlib
```

또한, Matplotlib 모듈은 그래프와 같은 2차원 시각화 자료에 관한 추가적인 도구를 가지고 있다. Pyplot(matplotlib.pyplot)이라 불리는 도구인데, MATLAB과 같이 그래프는 물론 점, 선의 형태를 꾸밀 수 있는 동/정적 시각화 도구이다. 이를 이용해 간단한 그래프를 그리는 코드는 다음과 같다.

```
from matplotlib import pyplot as plt
x_value = [1, 2, 3, 4]
y_value = [1, 4, 9, 16]
plt.plot(x_value, y_value)
plt.show()
```

실행 결과는 [Figure 2]와 같다.

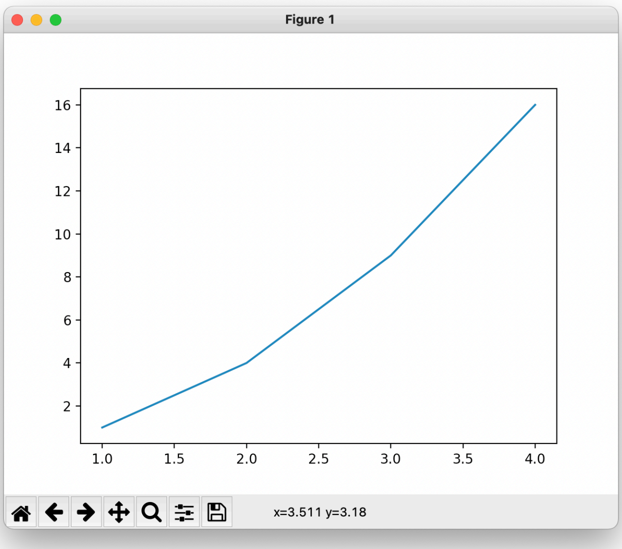

Figure 2

Ⅲ. 이론적 배경

1. 연구 설계

서론에서 밝힌 대로, 'COVID-19에 대한 각 방역 전략에 따른 확산 분석'을 위해 사전 표본을 먼저 추출한 후 시각화 하여 2차원 그래프를 제작할 것이다. 사전 표본의 경우 SIR Model의 구현이 가장 먼저 이루어져야 할 것인데, SIR 모델은 $S(t), I(t), R(t)$의 변화를 반환하기 때문에 Markov-Chain Update를 구현해야 한다. $S(t), I(t), R(t)$와 $S(t+1), I(t+1), R(t+1)$ 사이의 관계가 다음과 같은 식과 같이 존재한다는 것에 유의하자.

```
update():
      S -= beta * S * I / N
      I += beta * S * I / N - gamma * I
      R += gamma * I
```

$$F(t+1) = F(t) + \frac{\Delta F(t)}{\Delta t} \text{ (단, } F \in \{S, I, R\})$$

이를 활용하면, Markov-Chain Update의 구현체는 다음과 같이 정의될 것이다.

또한, update()가 n회 시행될 경우 S, I, R의 값은 n일 후의 상태에 대한 값으로 변경된다. 그렇다면 이 SIR Model은 어떻게 방역 전략에 따른 수치적 변화를 보여줄 수 있을까? 답은 방역 강도가 β와 반비례 관계를 가지게 된다는 것이다. 이는 β는 감염자가 하루 당 확산 가능한 접촉의 번수를 의미하는데, 방역 강도가 강해지면 강해질 수록 접촉의 수가 줄어드는 것은 자명하기 때문이다. 연구에는 방역 강도를 다섯 단계로 나누었다.

- No Quarantine: $\beta = 1$, 하루 당 평균 1명 감염 전파
- Loose Quarantine: $\beta = \frac{1}{2}$, 2일에 평균 1명 감염 전파
- Normal Quarantine: $\beta = \frac{1}{5}$, 5일에 평균 1명 감염 전파
- Korean Quarantine: $\beta = \frac{1}{10}$, 10일에 평균 1명 감염 전파 ($RO \approx 2$)
- Lockdown Quarantine: $\beta = \frac{1}{20}$, $RO \leq 1$

위의 값은 방역이 전혀 이뤄지지 않는 상황을 1로 가정, 이에 따른 비율을 β로 책정하였다. 또한

이 SIR Model을 통해 추출된 데이터들을 Array에 저장함으로써 matplotlib을 활용한 시각화가 가능하게 하였다.

2. 연구 과정

가장 먼저 SIR Model를 Python으로 구현하자[3]. 구현 코드는 [D]의 자료를 바탕으로 제작하였다.

```
1  class SIR:
2      def __init__(self, S, I, R, beta, gamma):
3          self.N = S + I + R
4          self.S = S
5          self.I = I
6          self.R = R
7          self.beta = beta
8          self.gamma = gamma
9
10     def update(self):
11         self.S -= self.beta * self.S * self.I / self.N
12         self.I += self.beta * self.S * self.I / self.N -
   self.gamma * self.I
13         self.R += self.gamma * self.I
```

__init__ Method(2행~8행)에서는 $S(0), I(0), R(0), \beta, \gamma$를 매개변수로 받고 있으며 update Method(10행~13행)에서는 Markov-Chain과 Differential Formula를 활용해 S, I, R를 Update 하고 있다. 다음은 이 SIRModel Class를 활용해 각 방역 전략에 해당되는 표본을 추출, 시각화하는 코드 중 일부(No Quarantine 전략에 대한 코드)이다. 전체 코드는 [E]에서 확인 수 있다.

```
from SIRmodel import SIR
import matplotlib.pyplot as plt

# STD: April 22, 2021
N = 674635
I = 687
R = 660 + 25018 + 1 # recovered + vaccination + dead
S = N - I

def simulateWithNoQuarantine():
    model = SIR(S, I, R, 1, 1./30)
    ds = [[], [], [], []] # [date, S, I, R]

    for i in range(360):
        model.update()
        ds[0].append(i)
        ds[1].append(model.S)
        ds[2].append(model.I)
        ds[3].append(model.R)

    plt.title("Simulation with no quarantine")
    plt.ylabel("Number of People")
```

[3] matplotlib과의 상호작용이 가능할 수 있게 하기 위한 조치

```python
    plt.xlabel("Date")
    plt.plot(ds[0], ds[1], label="Susceptible")
    plt.plot(ds[0], ds[2], label="Infectible")
    plt.plot(ds[0], ds[3], label="Recovery/Removed")

    plt.legend()
    plt.show()
```

이 코드를 실행시키면 다음과 같은 시각화 자료를 얻게 된다.

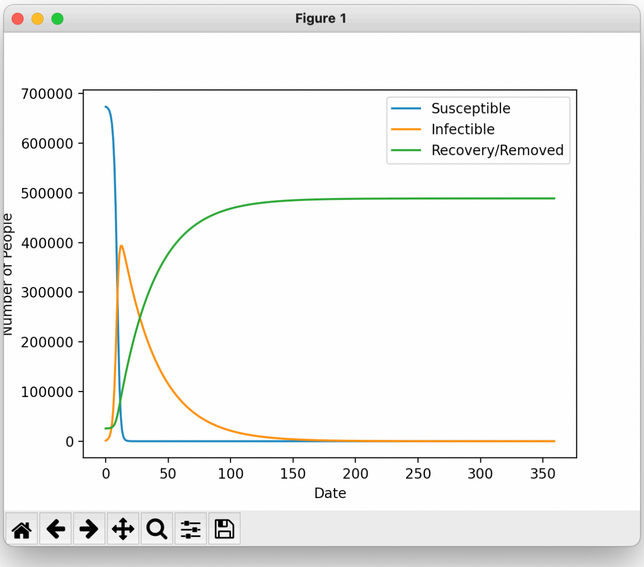

Figure 3. With No Quarantine

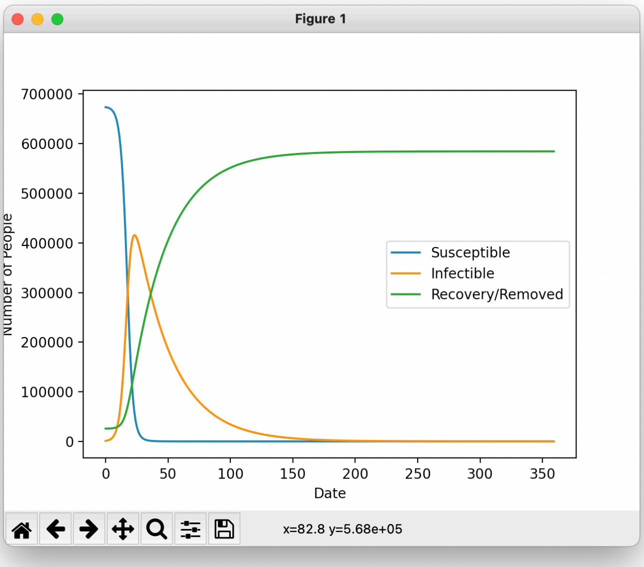

Figure 4. With Loose Quarantine

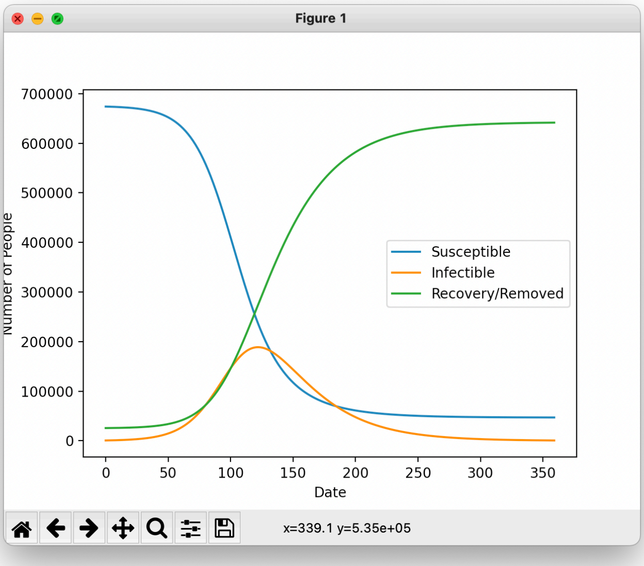

Figure 5. With Normal Quarantine & Korean Quarantine

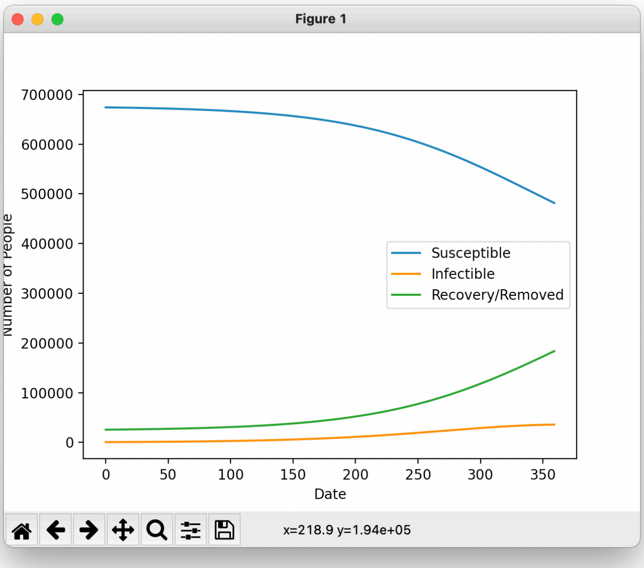

Figure 6. With Lockdown

3. 연구 결과

방역 전략에 따른 S, I, R의 변화를 나타내는 시각화 자료를 비교해보자. [Figure 3]과 [Figure 4], 그리고 [Figure 5]와 [Figure 6]과의 큰 차이점은 $I(t)$의 극점(vertex)의 위치이다. 전자는 극점이 원점에 가까이(대유행이 굉장히 빨리 온 경우)있고, 수많은 사망자 및 완치자가 한 순간에 폭증하는 것이 관찰된다. 이 SIR Model에서는 의료 체계 붕괴와 같은 특별한 상황이 고려되지 않기 때문에 실제로 이러한 방역 전략을 선택할 경우 자칫 완치자보다 사망자가 늘어나게 되는 최악의 상황에 도달할 수 있게 된다. 하지만, 대유행이 지나간 이후에는 확진자가 0명에 가까울 정도의 이상적 상황에 다다르게 된다. 즉, 대유행을 버티게 되면 집단면역에 빠르게 도달하게 되는 것이다.

반면, [Figure 5]와 [Figure 6]는 반대의 상황으로 흘러가게 된다. 극점이 원점에서 멀리 떨어져 있기 때문에 대유행이 늦게 도달하게 된다. 이는 달리 말해 백신과 같은 대응 체계를 설립할 시간이 더 주어진다는 의미이다. 이러한 강력한 방역 전략은 의료 체계의 붕괴를 막을 수 있다는 장점이 있지만, 자연적 집단면역이 힘들다는 단점이 존재한다.

Ⅳ. 결론 및 제언

COVID-19에 따른 여러 방역 전략에는 각기 다른 장단점이 존재한다. 약한 방역 체계를 채용하는 경우 대유행이 굉장히 빠르게 나타날 수 있기 때문에 사회의 의료 체계가 탄탄하거나 인구수가 적은 경우 선호되지만, 사회 인구수가 매우 크거나 위생 수준, 의료 체계가 비약한 경우에는 바이러스로 인한 대재앙은 물론 사회의 붕괴에 치닫을 수 있음에 유의해야 할 것이다. 반대로, 강한 방역 체계는 사회 구성원들의 불만 또는 불복종 경향이 커지기 때문에 사회적 비난을 감수할 수 있을 때 시행되어야 할 것이다.

■ 참고 문헌

A. https://www.maa.org/press/periodicals/loci/joma/the-sir-model-for-spread-of-disease-the-differential-equation-model

B. https://pypi.org/project/matplotlib/

C. https://matplotlib.org/stable/api/index.html

D. https://scipython.com/book/chapter-8-scipy/additional-examples/the-sir-epidemic-model/

E. https://github.com/RedoC-github/Gifted-Information-2021/blob/master/system.py