Intro to Math Modeling Final Project

Yi Gong

December 22, 2021

1 Introduction

SARS-CoV-2 (COVID-19) is a coronavirus leading to serious respiratory illness with high transmitting capability. The COVID-19 pandemic has been a worldwide event posing severe challenges to all human health. With limited data till now (May 2020), the concern of this math modeling project will be focused on predictions based on different preventive procedures.

In this project, I will utilize the classic SIR model as theory base to predict the future trends of COVID-19 infections in five U.S. states. Python OOP will be administered as the abstraction of SIR model, featured by ODE functions. By performing a time series evaluation of the data, I will use otimization with Nelder–Mead method to calculate the proper β: infection propability, thus calculating the R_0 value.

The new parameter, $nContact \in [2, 5]$, as to specify different procedures encountering COVID-19. $nContact$ refers to the number of unprotected people contacted (susceptible) by any one infected patient. We can briefly quantify to what degree the government policy took place in the imaginative scenario, featured by different values of $nContact$.

Data Source: U.S. CDC Data Racial Data

(*NOTE: This project was originally completed in May 2020, but has undergone updates as of Dec 2021 (Issuing health disparity), datasets/methods unchanged.)

1.1 Data Visualization

By defining two ploting functions, we can show the original data features as displayed below.

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.optimize import minimize
from scipy.integrate import odeint
import seaborn as sns
```

[2]:
```
def plot_num_t(y,x_name,y_name,label,color_name):
    plt.plot(y,color = color_name, label=label)
    plt.grid(True)
    plt.xlabel(f'{x_name}')
    plt.ylabel(f'{y_name}')
def general_plot(df, state_sum, state_name):
```

```
    plt.figure()
    plt.title(state_name, fontsize=15)
    plot_num_t(df['deathIncrease'].values[::
↪-1],'Time','Number','daily_death','red')
    plot_num_t(df['positiveIncrease'].values[::
↪-1],'Time','Number','daily_infected','orange')
    plt.legend()
    plt.show()

    test_ratio = df['total'].values[::-1] / state_sum
    plt.figure()
    plt.plot(test_ratio, color='blue')
    plt.xlabel('time')
    plt.ylabel('Percentage of tested people')
    plt.grid(True)
    plt.show()
```

1.1.1 Data Importation

[3]:
```
NY_data, AK_data, CA_data, MA_data, TX_data = pd.read_csv('./Data/NY.csv'), pd.
↪read_csv('./Data/AK.csv'), pd.read_csv('./Data/CA.csv'), pd.read_csv('./Data/
↪MA.csv'), pd.read_csv('./Data/TX.csv')
NY_data.head()
```

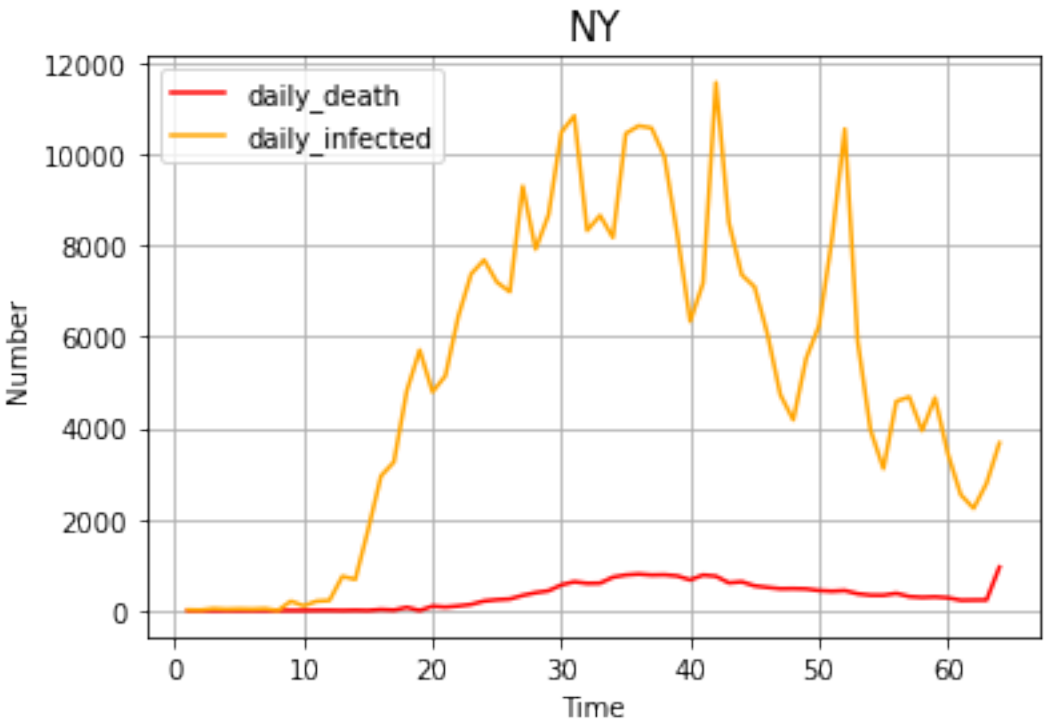
[3]:

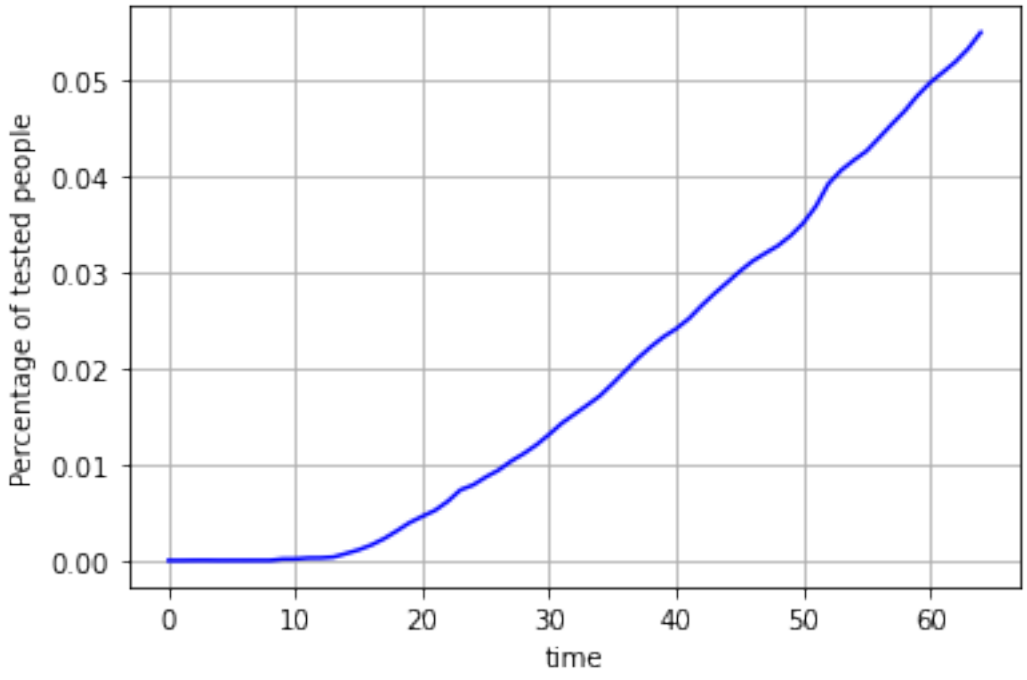
	date	state	positive	negative	recovered	death	total	\
0	20200507	NY	327649	762267	55547.0	20828.0	1089916	
1	20200506	NY	323978	731943	54597.0	19877.0	1055921	
2	20200505	NY	321192	707707	58950.0	19645.0	1028899	
3	20200504	NY	318953	688357	58950.0	19415.0	1007310	
4	20200503	NY	316415	669496	58950.0	19189.0	985911	

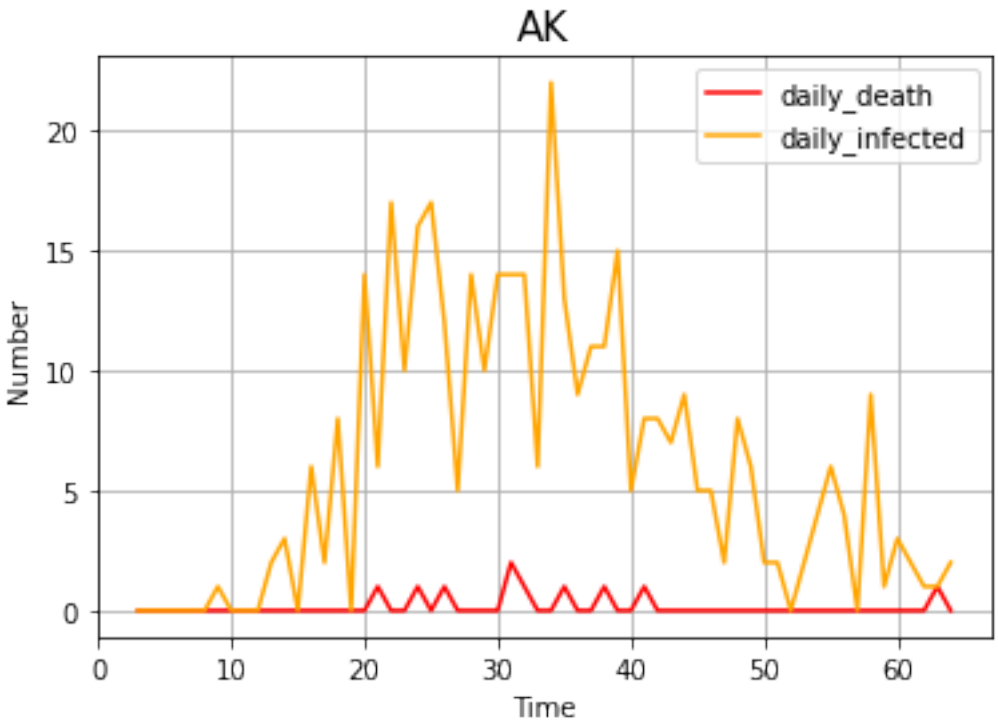
	deathIncrease	negativeIncrease	positiveIncrease	totalTestResultsIncrease
0	951.0	30324.0	3671.0	33995.0
1	232.0	24236.0	2786.0	27022.0
2	230.0	19350.0	2239.0	21589.0
3	226.0	18861.0	2538.0	21399.0
4	280.0	23402.0	3438.0	26840.0

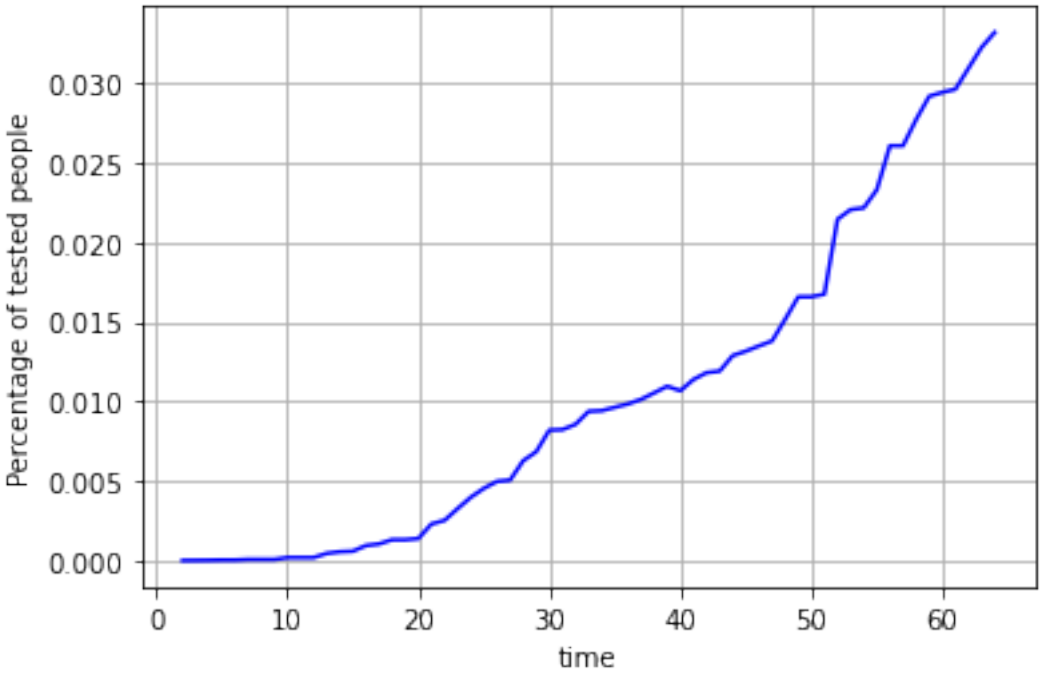
1.1.2 Time Series Visualization

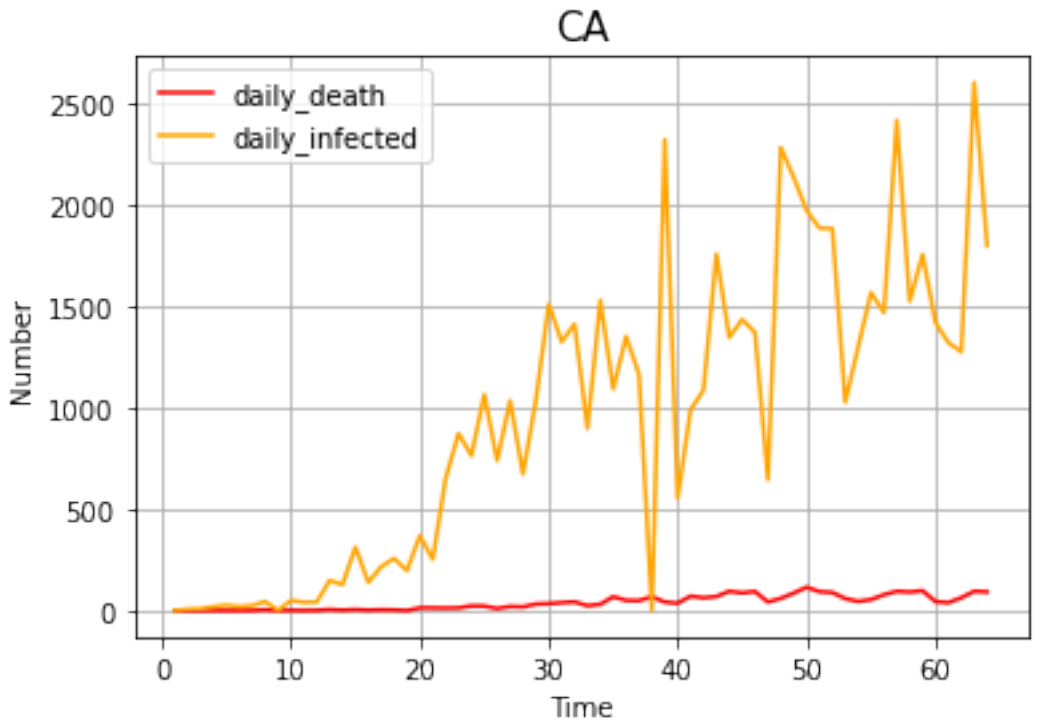
[4]:
```
NY_num, AK_num, CA_num = 19795791, 733391, 39770000
general_plot(NY_data,NY_num,'NY')
general_plot(AK_data,AK_num,'AK')
general_plot(CA_data,CA_num,'CA')
```

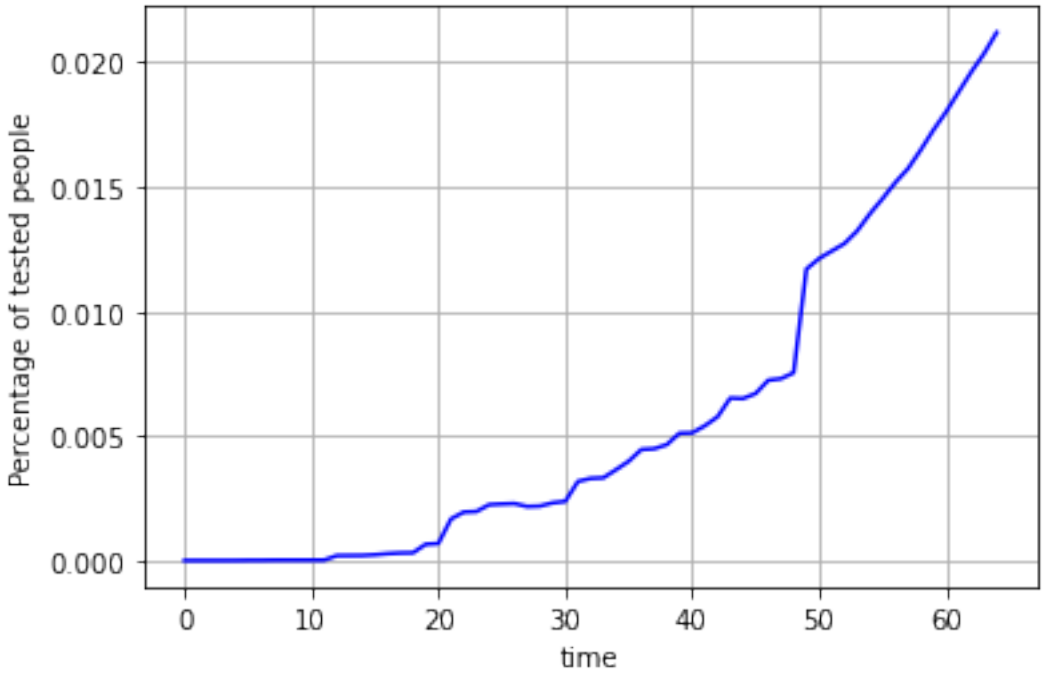

NY
daily_death
daily_infected
Number
Time
0
2000
4000
6000
8000
10000
12000
0
10
20
30
40
50
60


Percentage of tested people
time
0.00
0.01
0.02
0.03
0.04
0.05
0
10
20
30
40
50
60


AK
daily_death
daily_infected
Number
Time
0
5
10
15
20
0
10
20
30
40
50
60


Percentage of tested people
time
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
10
20
30
40
50
60


CA
daily_death
daily_infected
Number
Time
0
500
1000
1500
2000
2500
0
10
20
30
40
50
60


Percentage of tested people
time
0.000
0.005
0.010
0.015
0.020
0
10
20
30
40
50
60

2 SIR Modeling

The SIR model is composed of three crucial parameters: $S(t)$, $I(t)$, $R(t)$

$S(t)$: Number of susceptibles at time t.
$I(t)$: Number of infections at time t.
$R(t)$: Number of recoveries at time t.

(*t interval set as one day)

Parameters β and γ help to establish the relationships among $S(t)$, $I(t)$, $R(t)$

β: The transmission rate
γ: The mortality (or recovery) rate
R_0: The basic reproduction number is defined as the expected number of secondary cases produced by a single (typical) infection in a completely susceptible population. (Jones, 2020)

ODE Functions with Mass-Action Law:

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

R_0 is simplified as :

$$R_0 = \frac{\beta}{\gamma}$$

Since COVID-19 diseases have a relatively low mortality rate, we will confer it to the recovery rate instead. From CDC advice and common experiences in terms of quarantine length as 14 days, we set $\gamma = \frac{1}{14}$. In order to estimate a proper β for each state, we use the convex optimization model. The optimal function can be described as:

$$min \sum_{t=1}^{T} (e^{(\beta-\gamma)t}) - I(t))^2$$

where T refers to obersvation time step. Use Scipy to calculate β and γ. The Nelder-Mead method iteratively generates a sequence of simplices to approximate an optimal point of (1.1). At each iteration, the vertices of the simplex are ordered according to the objective function values (Gao & Han, 2010):

$$f(x_1) \leq f(x_2) \leq \cdots \leq f(x_{n+1})$$

We also define a new variable *nContact*, representing the average susceptibles resulting in every infected person. This variable assits the analysis to evaluate how social distancing (reduced *nContact*) can help to inhibit the spread of COVID-19.

[5]:
```
class SIRModel():

    def __init__(self, N, beta, gamma, k, city_data, prob, name):
        self.beta, self.gamma, self.N = beta, gamma, N
        self.city_data = city_data
        self.nContact = k
        self.prob = prob
        self.name = name
        self.t = np.linspace(0, 180, 181)
        self.setInitCondition(N, city_data)

    def odeModel(self, population, t):
        diff = np.zeros(3)
        s,i,r = population
        diff[0] = - self.beta * s * i / self.N
        diff[1] = self.beta * s * i / self.N - self.gamma * i
        diff[2] = self.gamma * i
        return diff

    def setInitCondition(self, N, city_data):
        tmp_i, tmp_r = city_data['positive'].values[0], city_data['recovered'].
↪values[0] + city_data['death'].values[0]
        self.populationInit = [N - tmp_i - tmp_r, tmp_i, tmp_r]

    def solve(self):
        self.solution = odeint(self.odeModel,self.populationInit,self.t)

    def plot(self, N, city_data, prob, name):
        plt.plot(self.solution[:,1],color = 'red',label = 'Infection')
        plt.plot(self.solution[:,2],color = 'green',label = 'Recovery')
        plt.plot(N - self.solution[:,2]- self.solution[:,1],color =␣
↪'orange',label = 'Suscepible')
        plt.plot(city_data['positive'].values[::
↪-1],color='blue',linestyle='--',label='True Infection')
        plt.title('SIR Model( '+'InfectProb:'+ str(np.round(prob,4)) + f'␣
↪nContact:{self.nContact} ' + 'State:' + self.name +' )')
        plt.legend()
        plt.xlabel('Day')
        plt.ylabel('Number of people')
        plt.grid(True)
        plt.show()
```

3 Results and Analysis

By setting the variable *nContact* between 2 to 5, we can evaluate the effects of government policies regarding the control of crowding and social distancing. A higher *nContact* level, in our case a value between 4-5, refers to minor or no restrictions on human activity. A lower *nContact* level,

in our case a value between 2-3, indicates higher-leveled regulations regarding quanrantine, social distancing, and non-medical government procedures.

From the following sample presentation, we can interpret a workflow from data to infection probability, and finally to the output modeling. With optimization under Nelder-Mead method, the loss function is optimized through time steps as shown before.

From the following graph, we can see that along with the increasing value of *nContact* (i.e. worsening control), the peak of virus infection is coming to a later data. Even without obvious peak, the smaller the *nContact* value is, the smaller the peak value will be. This is in line with our understanding that the better the virus isolation policy is, the less severe the situation would be. Blue line indicates that in real-time scenario, the government responded quickly, yet it could also be due to limited data efficacy.

3.1 Sample Presentation (NY)

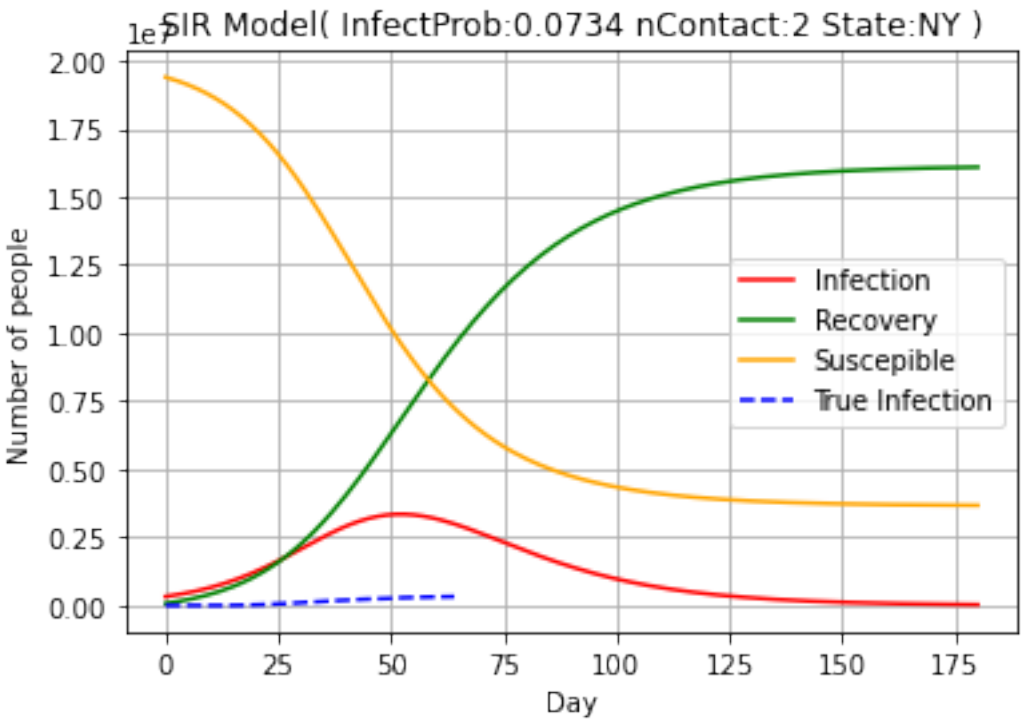
3.1.1 Method: Optimization with Nelder–Mead method

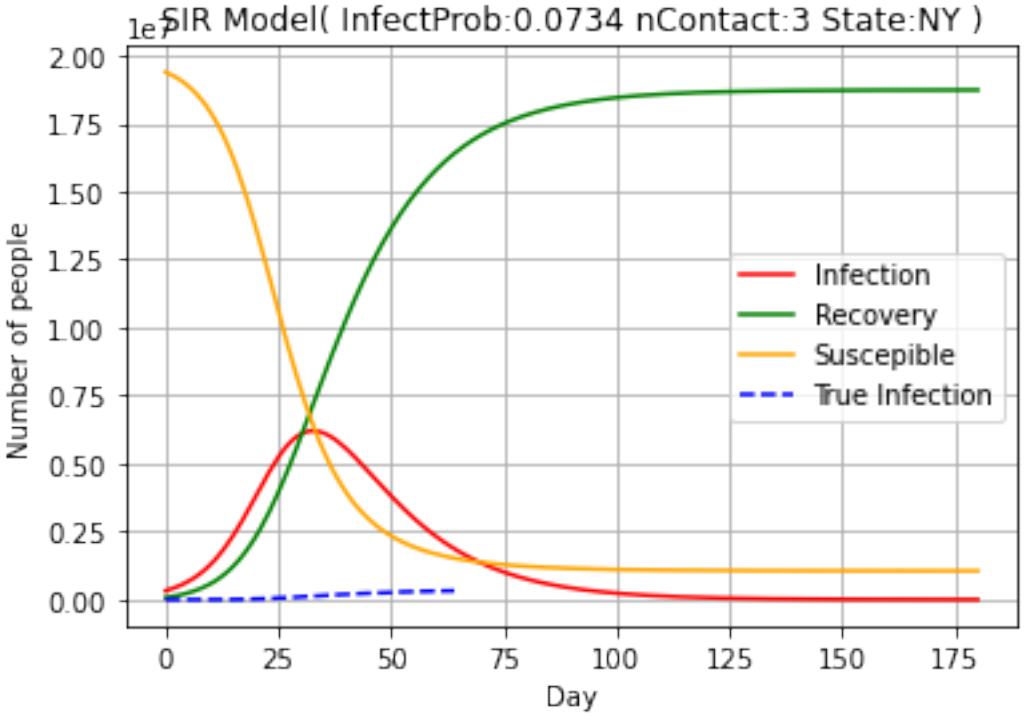
[6]:
```
nContact, gamma = int(5), 1/14
time_step, x0 = 40, 0.04

def CostFunction(infectionProb):
    pos = NY_data['positive'].values[::-1]
    loss = np.array(np.exp((infectionProb * nContact - gamma) * time_step) -␣
 ↪pos)
    beta = (loss**2).sum()/time_step
    return beta
# Use optimization tools to discover the infection probability
solution = minimize(CostFunction, x0, method='nelder-mead', options={'xtol':␣
 ↪1e-8, 'disp': True})
NY_prob = solution.x[0]
print('\nBeta: ', NY_prob)
print('R0: ', NY_prob / gamma)
```

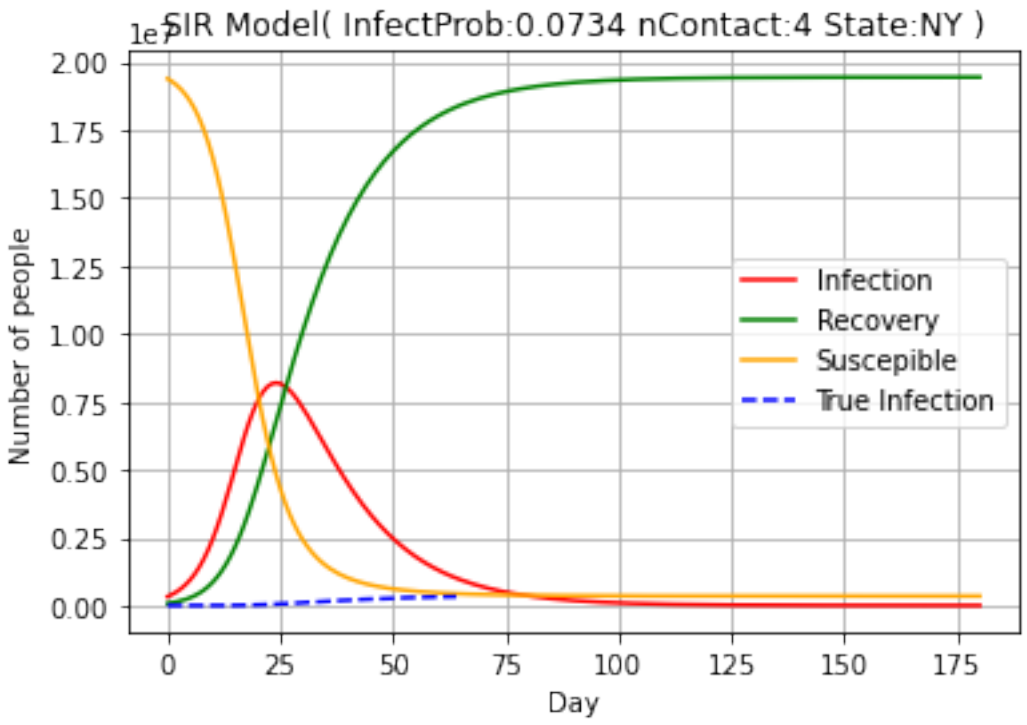
```
Optimization terminated successfully.
         Current function value: 23454993281.051792
         Iterations: 29
         Function evaluations: 58

Beta:  0.07343043422698979
R0:  1.028026079177857
```

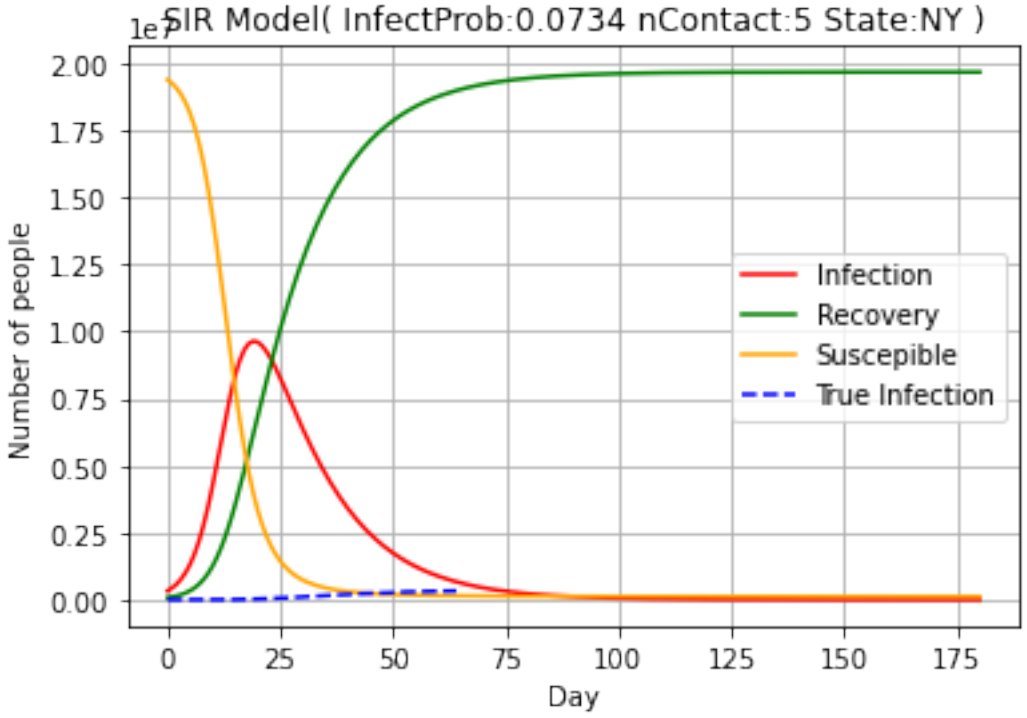
3.1.2 Model Graphing

[7]:
```
for k in range(2,6):
    beta, gamma, N = k * NY_prob, 1/14, NY_num
    SIRModel_NY = SIRModel(N, beta, gamma, k, NY_data, NY_prob, 'NY')
    SIRModel_NY.solve()
    SIRModel_NY.plot(N, NY_data, NY_prob, 'NY')
```


SIR Model(InfectProb:0.0734 nContact:2 State:NY)
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of people
0
25
50
75
100
125
150
175
Day
Infection
Recovery
Suscepible
True Infection


SIR Model(InfectProb:0.0734 nContact:3 State:NY)
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of people
0
25
50
75
100
125
150
175
Day
Infection
Recovery
Suscepible
True Infection


SIR Model(InfectProb:0.0734 nContact:4 State:NY)
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of people
0
25
50
75
100
125
150
175
Day
Infection
Recovery
Suscepible
True Infection


SIR Model(InfectProb:0.0734 nContact:5 State:NY)
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Number of people
0
25
50
75
100
125
150
175
Day
Infection
Recovery
Suscepible
True Infection

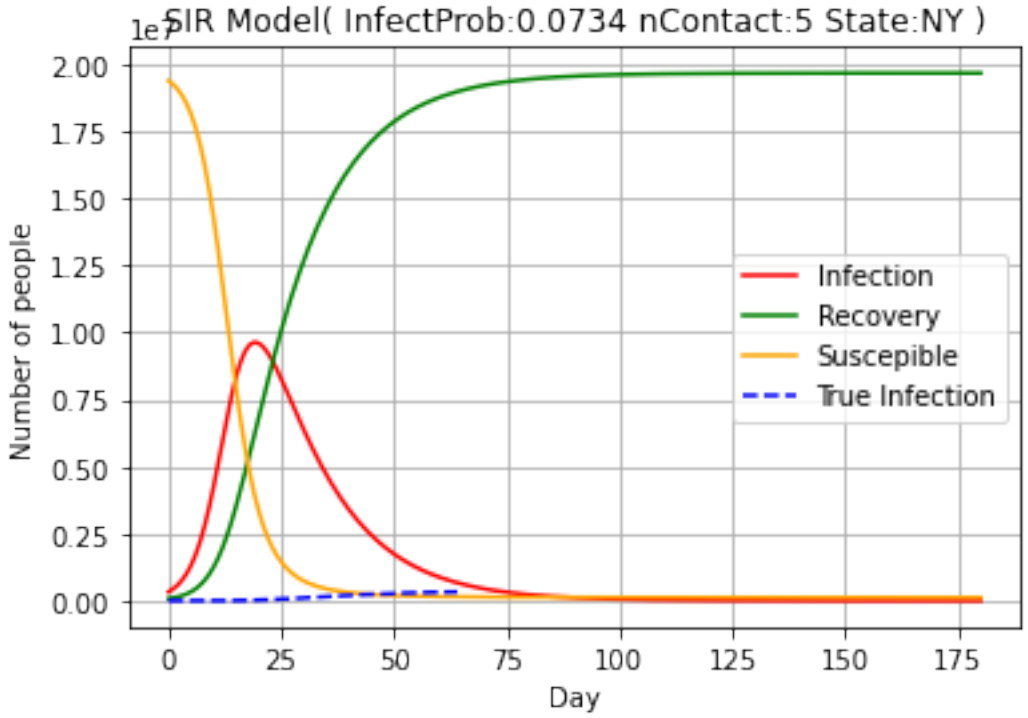
3.2 Horizontal Analysis

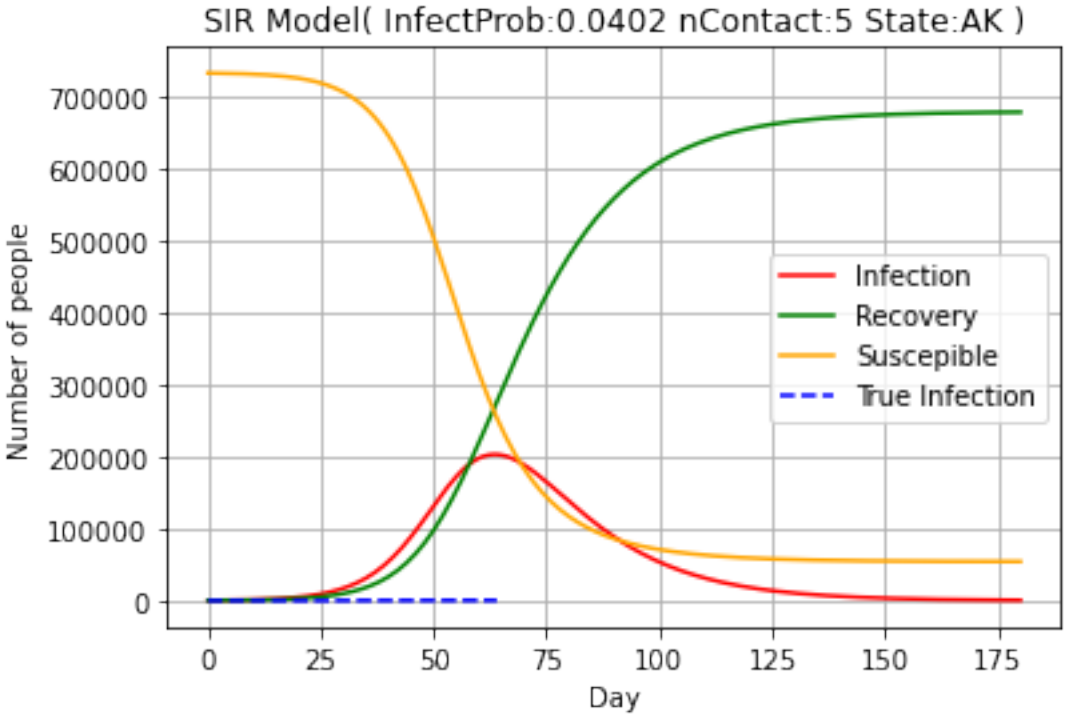
3.2.1 Paramters

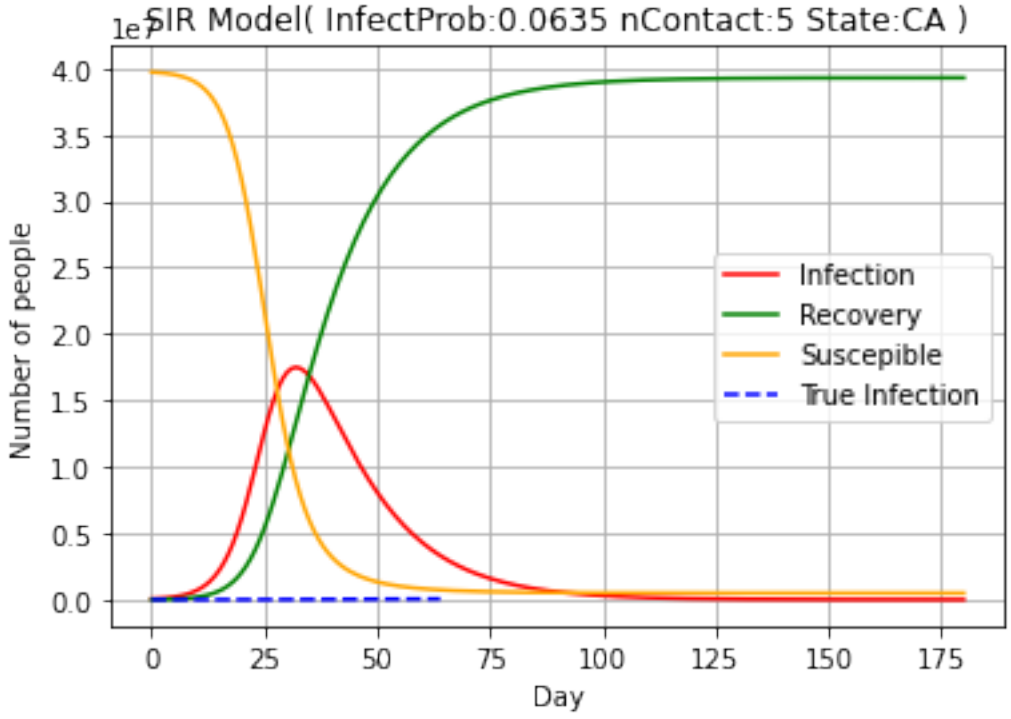
Parameter/ State	NY	AK	CA	MA	TX
γ	1/14	1/14	1/14	1/14	1/14
β	0.0734	0.0402	0.0635	0.0323	0.0608
nContact	5	5	5	5	5
R_0	1.028	0.563	0.889	0.452	0.851
N	19795791	733391	39770000	6593587	24782302

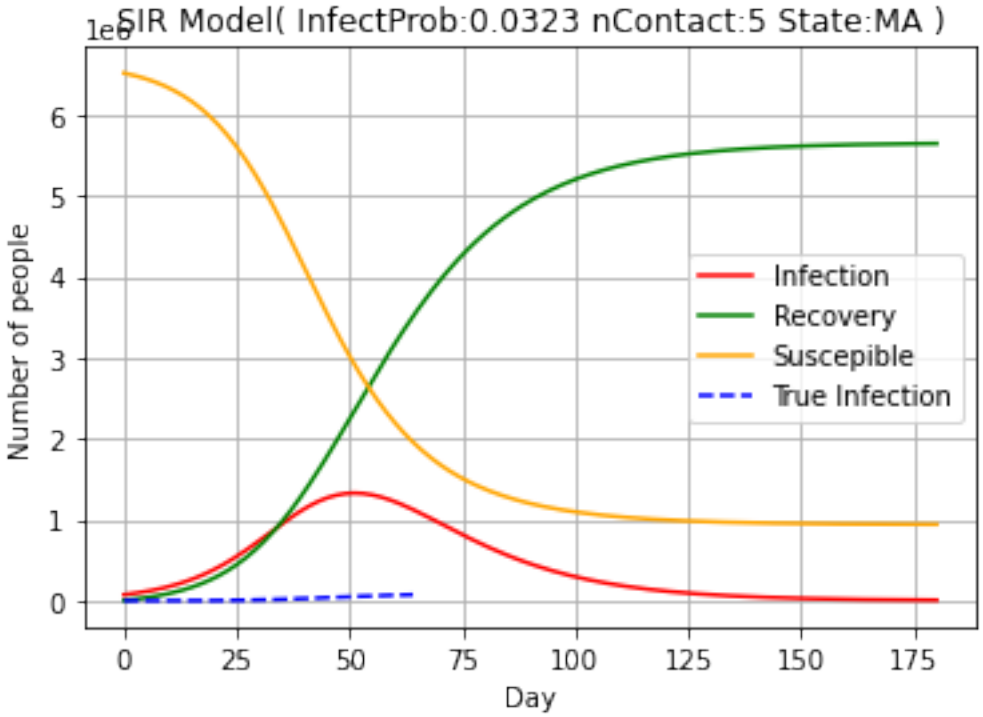
[8]:
```
state_name = ['NY','AK','CA','MA','TX']
data_list = [NY_data, AK_data, CA_data, MA_data, TX_data]
beta_list = [0.0734,0.0402,0.0635,0.0323,0.0608]
pp_list = [19795791,733391,39770000,6593587,24782302]
```

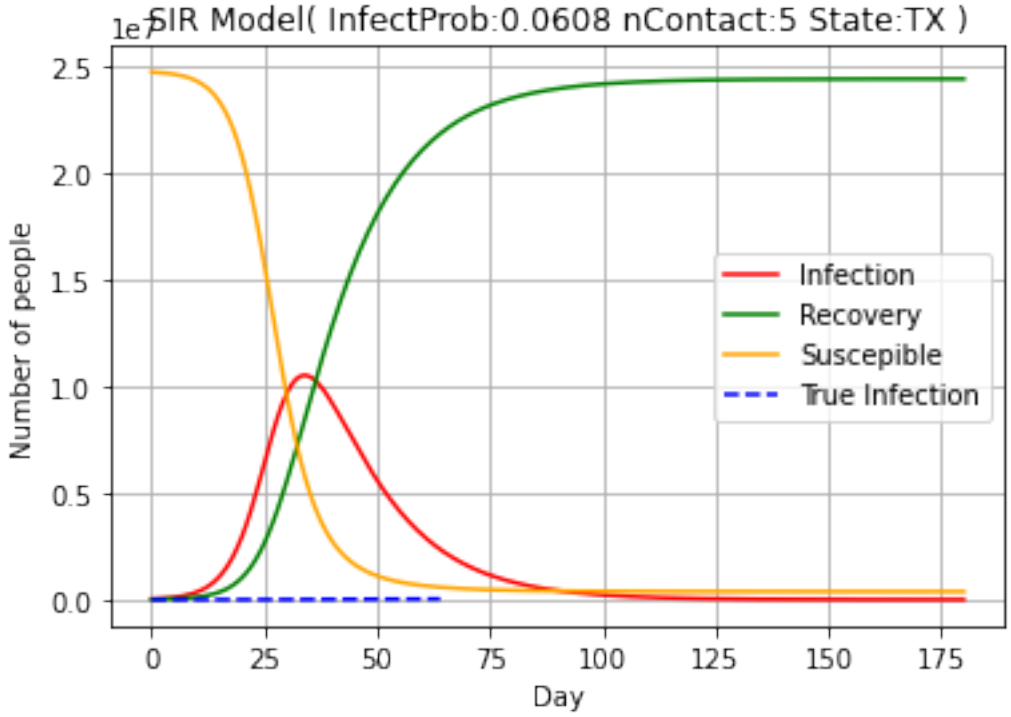
[9]:
```
for i in np.arange(5):
    prob = beta_list[i]
    path = data_list[i]
    name = state_name[i]
    beta, gamma, N = 5 * beta_list[i], 1/14, pp_list[i]
    plt.figure()

    SIR_Model = SIRModel(N, beta, gamma, 5, path, prob, name)
    SIR_Model.solve()
    SIR_Model.plot(N,path,prob, name)
```




SIR Model(InfectProb:0.0402 nContact:5 State:AK)
Number of people
Day
Infection
Recovery
Suscepible
True Infection
0
100000
200000
300000
400000
500000
600000
700000
0
25
50
75
100
125
150
175


SIR Model(InfectProb:0.0635 nContact:5 State:CA)
1e7
Number of people
Day
Infection
Recovery
Suscepible
True Infection
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
25
50
75
100
125
150
175


SIR Model(InfectProb:0.0323 nContact:5 State:MA)
1e6
Number of people
Day
Infection
Recovery
Suscepible
True Infection


SIR Model(InfectProb:0.0608 nContact:5 State:TX)
1e7
Number of people
Day
Infection
Recovery
Suscepible
True Infection

4 Health Disparities

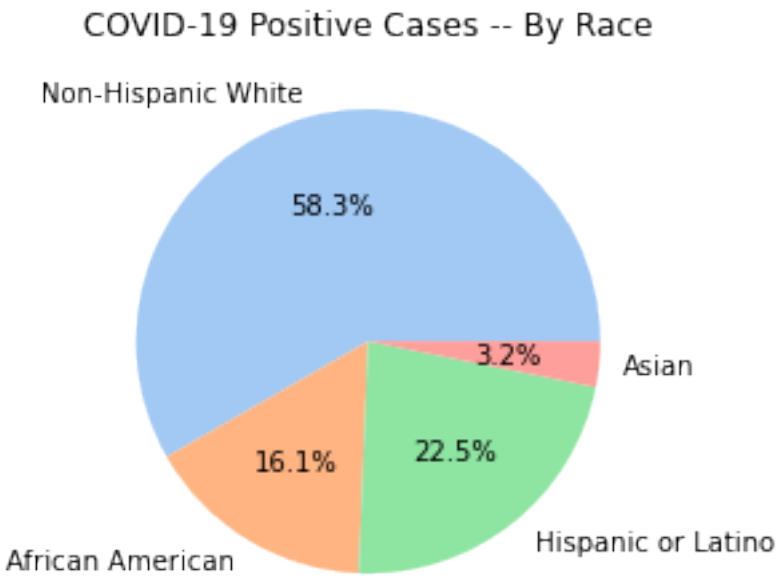
Prospective research informed me that racial/ economic minorities received disproportionately negative impacts in the pandemic. Every person behind the plain data is crying for help: their poverty or inaccessibility to medical resources. Unfortunately, we cannot see since such inequalities neutralized in bytes. By adding a separate section in my project, I included these underrepresented data in my analysis. Although not directly related to the final model, these behavioral perspectives are way more significant than the analysis itself. Healthcare is as much a political issue of social welfare as a scientific inquiry to balance equity and efficiency. Data-driven solution to health disparities is the "what" to share through analytics: a mission to address social-economic, racial and ethnic inequalities in the healthcare system.

[10]:
```
df = pd.read_csv('./Data/CRDT Data.csv')
race_data = df.iloc[:, :7]
race_data.head()
```

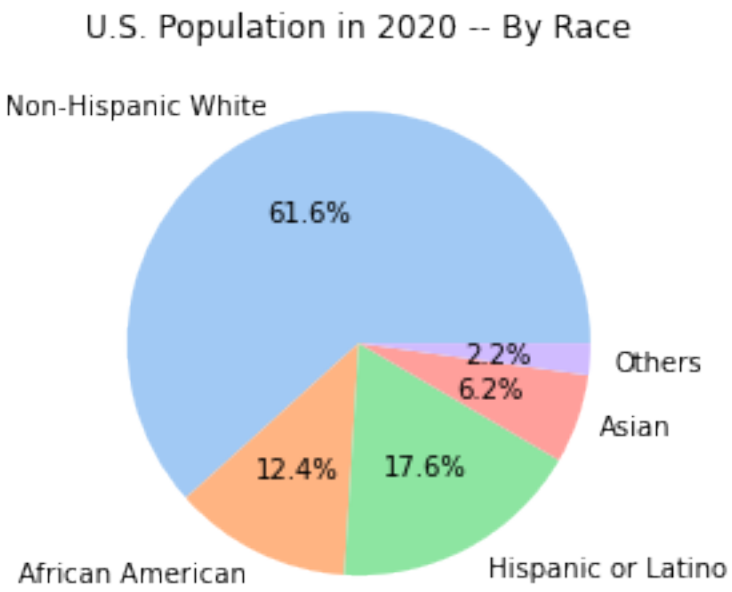
[10]:

	Date	State	Cases_Total	Cases_White	Cases_Black	Cases_Latinx	Cases_Asian
0	20210307	AK	59332.0	18300.0	1499.0	NaN	2447.0
1	20210307	AL	499819.0	160347.0	82790.0	NaN	2273.0
2	20210307	AR	324818.0	207596.0	50842.0	NaN	2913.0
3	20210307	AS	NaN	NaN	NaN	NaN	NaN
4	20210307	AZ	826454.0	308453.0	25775.0	244539.0	11921.0

[11]:
```
sum_data = [race_data['Cases_White'].sum(),race_data['Cases_Black'].
 ↪sum(),race_data['Cases_Latinx'].sum(),race_data['Cases_Asian'].sum()]
labels = ['Non-Hispanic White','African American',' Hispanic or Latino','Asian']
colors = sns.color_palette('pastel')[0:5]
plt.title('COVID-19 Positive Cases -- By Race')
plt.pie(sum_data, labels = labels, colors = colors, autopct='%.1f%%')
plt.show()
```



[12]:
```
pop_data = [61.6,12.4,17.6,6.2,2.2]
pop_labels = ['Non-Hispanic White','African American',' Hispanic or␣
↪Latino','Asian','Others']
colors = sns.color_palette('pastel')[0:5]
plt.title('U.S. Population in 2020 -- By Race')
plt.pie(pop_data, labels = pop_labels, colors = colors, autopct='%.1f%%')
plt.show()
```



5 References

Gao, Fuchang, and Lixing Han. "Implementing the Nelder-Mead Simplex Algorithm with Adaptive Parameters." Computational Optimization and Applications, vol. 51, no. 1, 2010, pp. 259–277., https://doi.org/10.1007/s10589-010-9329-3.

Jones, James Holland, et al. "Transmission-Dynamics Models for the SARS Coronavirus-2." American Journal of Human Biology : the Official Journal of the Human Biology Council, John Wiley & Sons, Inc., Sept. 2020, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7536961/.