An evaluation of effect of preventive measures on COVID-19

Trilokinath Modi

March 28, 2021

1 Model

Considering a spread of COVID-19 in a population with four compartments s, e, i, r, describing the fraction of population that is susceptible, exposed but not infective, infective and recovered respectively. Assuimg a spatially homogenous spread of the virus and the population to be constant. From [1], the dynamics can then be written as equation 1.

$$\frac{ds}{dt} = -\beta si \tag{1}$$
$$\frac{de}{dt} = \beta si - \gamma e \tag{2}$$
$$\frac{di}{dt} = \gamma e - \alpha i \tag{3}$$
$$\frac{dr}{dt} = \alpha i \tag{4}$$

1.1 Dimensionless model

Given dimensionless parameters, time τ, susceptibles σ, exposed ϵ, infected η and removed ρ. Let these dimensionless parameters are defined by equation 5.

$$\tau = \frac{t}{t_0}, \quad \sigma = \frac{s}{s_0}, \quad \epsilon = \frac{e}{e_0}, \quad \eta = \frac{i}{i_0}, \quad \rho = \frac{r}{r_0} \tag{5}$$

The given SEIR model can then be written as equation 6.

$$\begin{aligned} \frac{d\sigma}{d\tau} &= -\beta t_0 i_0 \eta\sigma \\ \frac{d\epsilon}{d\tau} &= \beta s_0 i_0 \frac{t_0}{e_0}\eta\sigma - \gamma t_0 \epsilon \\ \frac{d\eta}{d\tau} &= \gamma \frac{t_0}{i_0} e_0 \epsilon - \alpha t_0 \eta \\ \frac{d\rho}{d\tau} &= \alpha \frac{t_0}{r_0} i_0 \eta \end{aligned} \tag{6}$$

Now, let,

$$s_0 = e_0 = i_0 = \frac{\gamma}{\beta}, \quad t_0 = \frac{1}{\gamma}, \quad r_0 = \frac{\alpha}{\beta} \tag{7}$$

Using equation 7 in equaiton 6, we get equation 8.

$$
\begin{aligned}
\frac{d\sigma}{d\tau} &= -\eta\sigma \\
\frac{d\epsilon}{d\tau} &= \eta\sigma - \epsilon \\
\frac{d\eta}{d\tau} &= \epsilon - \frac{\alpha}{\gamma}\eta = \epsilon - A\eta \\
\frac{d\rho}{d\tau} &= \eta
\end{aligned} \tag{8}
$$

Hence, the dimensionless parameter $A = \frac{\alpha}{\gamma}$.

1.2 Steady states

At steady state,

$$
\frac{ds}{dt} = \frac{de}{dt} = \frac{di}{dt} = \frac{dr}{dt} = 0 \tag{9}
$$

$$
\begin{aligned}
\frac{ds}{dt} &= 0 \implies i^* = 0 \text{ OR } s^* = 0 \\
\frac{de}{dt} &= 0 \implies s^* i^* = \frac{e^*\gamma}{\beta} \\
\frac{di}{dt} &= 0 \implies e^* = \frac{\alpha}{\gamma} i^* \\
\frac{dr}{dt} &= 0 \implies i^* = 0 \\
\frac{d(e+i)}{dt} &= 0 \implies \beta s^* i^* = \alpha i \implies s^* = \frac{\alpha}{\beta} \text{ OR } i^* = 0
\end{aligned}
$$

Hence, due to $\frac{dr}{dt} = 0$, the $i* = 0$ has to be true in all steady states. The $s*$ can be both 0 or $\frac{\alpha}{\beta}$.

Since, $\frac{ds}{dt} + \frac{de}{dt} + \frac{di}{dt} + \frac{dr}{dt} = 0$, the population is bounded by a constant number N or fraction 1. Hence, $\frac{dr}{dt}$ can be found using $1 - others$. The stability matrix of the remaining 3-dimensional system can be written as equation 10. The Jacobian is written using dimensionless SEIR model.

$$
J = \begin{pmatrix} -\eta & 0 & -\sigma \\ \eta & -1 & \sigma \\ 0 & 1 & -A \end{pmatrix} \tag{10}
$$

Now at steady state $I^* = 0 \implies \eta^* = 0$,

$$
\begin{aligned}
tr(J)\Big|_{\eta^*} &= -(\eta + 1 + A) = -(1 + A) < 0 \\
Det(J)\Big|_{\eta^*} &= A > 0
\end{aligned}
$$

Hence, all steady states with $I* = 0$ are stable. The eigen value of stability matrix can be computed using equation 11.

$$
\begin{vmatrix} -\eta^* - \lambda & 0 & -\sigma^* \\ \eta^* & -1 - \lambda & \sigma^* \\ 0 & 1 & -A - \lambda \end{vmatrix} = 0 \implies \tag{11}
$$

Considering $\eta^* = 0$, we get condition equation 12

$$-\lambda(1+\lambda)(A+\lambda) = -\sigma^*\lambda \implies \lambda = 0 \text{ OR } (1+\lambda)(A+\lambda) = \sigma* \tag{12}$$

The eigen values must be negative. Hence, evaluating the condition 12.

$$\begin{aligned}
(1+\lambda)(A+\lambda) &= \sigma* \\
\implies \lambda^2 + (A+1)\lambda + A - \sigma^* = 0 \implies \lambda &= \frac{-(A+1) \pm \sqrt{(A+1)^2 - 4(A-\sigma^*)}}{2} < 0 \\
\implies \lambda &= \frac{-(A+1) \pm \sqrt{(A-1)^2 + 4\sigma^*}}{2} < 0
\end{aligned}$$

On looking closely, the term under square root is always positive. Hence one of the λ is always negative. The other λ can be evaluated by equation **??**.

$$\begin{aligned}
\sqrt{(A-1)^2 + 4\sigma^*} &< A+1 \\
\implies \sigma^* &< A \\
\implies s^* &< \frac{\alpha}{\beta}
\end{aligned} \tag{13}$$

Hence, the required condition at which the steady state condition is marginally stable is $s^* < \frac{\alpha}{\beta}$.

The other steady state condition is when $I^* = 0, S* = 0$. Substituting these in eigen matrix eqn 11 we get $-\lambda(1+\lambda)(A+\lambda) = 0$ implying either $\lambda = 0$ or $\lambda^2 + (A+1)\lambda + A = 0$. This doesn't enforce any specific condition and all λ values are negative.

1.3 Expression $s(\infty)$

The proof is as follows.

$$\frac{dE}{dt}+\frac{dI}{dt}=\beta SI-\alpha I$$
$$\frac{d(E+I)}{dS}=-1+\frac{\alpha}{\beta S}$$
$$\frac{dS}{d(E+I}=\frac{\beta S}{\alpha-\beta S}$$
$$\frac{dS}{\frac{\beta S}{\alpha-\beta S}}=d(E+I)$$

Integrating both sides

$$\frac{\alpha}{\beta}log(S)-S=E+I+C$$
$$\implies \frac{s^{\frac{\alpha}{\beta}}}{e^S}=e^{E+I}C$$

Now at $t=0$ we get

$$C=\frac{S(0)^{\frac{\alpha}{\beta}}}{e^{S(0)+E(0)+I(0)}}$$

At $t=\infty, E(\infty)=I(\infty)=0$

$$\frac{s(\infty)^{\frac{\alpha}{\beta}}}{e^S(\infty)}=e^{E(\infty)+I(\infty)}\frac{S(0)^{\frac{\alpha}{\beta}}}{e^{S(0)+E(0)+I(0)}}$$

Let $RT=\frac{s(\infty)}{s(0)}$, then we get,

$$RT^{\frac{\alpha}{\beta}}=e^{s(\infty)-s(0)-E(0)-I(0)}$$
$$RT^{\frac{\alpha}{\beta}}=e^{s(0)(RT-\frac{E(0)-S(0)}{S(0)}})$$

Neglecting term $\frac{E(0)-S(0)}{S(0)}$ due to small fractions at $t=0$ and taking log on both sides.We get,

$$\frac{\alpha}{\beta}log(RT)=s(0)(RT-1)$$
$$\implies \frac{log(RT)}{RT-1}=\frac{\beta}{\alpha}S(0)$$
$$\implies \frac{log(\frac{S(\infty)}{S(0)})}{\frac{S(\infty)}{S(0)}-1}=\frac{\beta}{\alpha}S(0)$$
$$\implies \frac{S(\infty)}{S(0)}=\exp\left(\frac{\beta}{\alpha}S(0)(S(\infty)-1)\right) \tag{14}$$

The equation 14 is the required relation between the ratio of $\frac{S(\infty)}{S(0)}$ against $S(0)$.

1.3.1 Plot $\frac{s(\infty)}{s(0)}$ against $s(0)$

The plot for this equation is drawn in Mathematica using command $ploty=exp(2.5*x*(y-1))$. The resulting figure is shown in figure 1.

From the plot, one can say $\frac{S(\infty)}{S(0)}=1$ implies either the disease has never occurred or if the disease has occurred then $\frac{S(\infty)}{S(0)}\rightarrow<\frac{\alpha}{\beta}=\frac{2}{5}$. The shape do exhibit this behaviour.

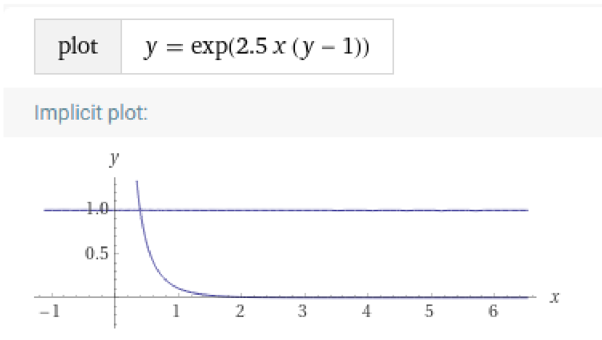
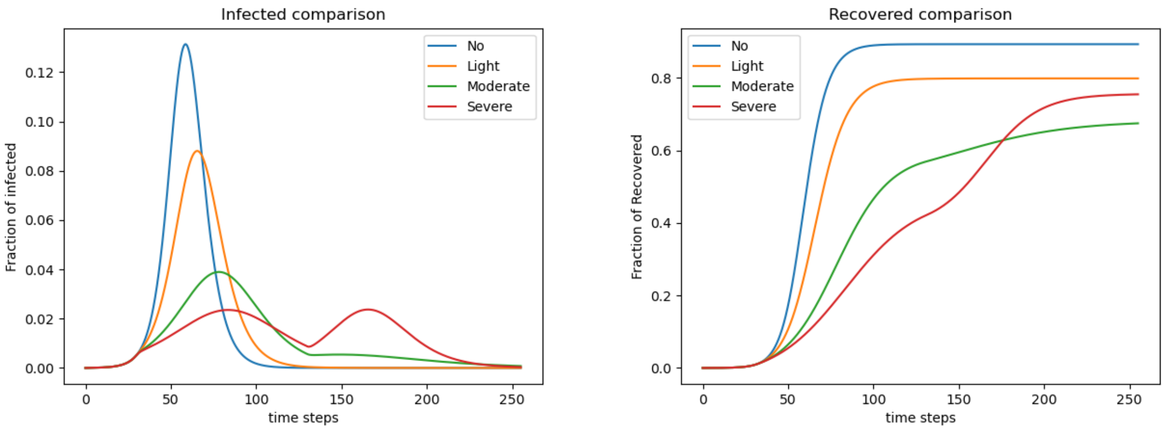


Figure 1: The plot represents the ratio $\frac{S(\infty)}{S(0)}$ in Y-axis and $S(0)$ in x-axis.

1.4 Effect of preventive measures

The ratio from the simulation turned out to be $\frac{S(\infty)}{S(0)} = 0.10735$. From the equation 14, the ratio for the gven parameter setting comes to 0.10738. The value is computed in mathematica using the command $2.49975 * (x - 1) = log(x)$, where 2.49975 is obtained by $\frac{\beta}{\alpha} S(0)$. This explaines the theoretical value and simulation for no preventive measure.

The two plots for the evaluation of preventive measure by parameter β are shown in the figure 2.



(a) (b)

Figure 2: Plots (a)(b) portrays the rates of infection and recoveries respectively. In each of the two plot, x- axis is time steps, whereas y-axis is the fraction. Each curve represents severity level. The severity level is modified in days 30 to 130.

A second wave is observed in the severe prevention case due to presence of two peaks. In case of moderate there is a increase in infection fraction but is not significant to call it second wave. In other two there is no second wave.

Table 1 represents the advantages and disadvantages.

Table 1: Advantages and disadvantages

Severity	**Adantage**	**Disadvantage**
No	Economical impact are low and infection spreads so quickly that it has earliest downfall in fection rate	Highest infection rate
Light	Negligible impact on economy and infection reaches a peak and falls quickly, but slower than severe.	Much lower infection rate compared to no severity
Moderate	Lesser infection spread as compared to lower severity	Longer time for infection rates to fall completely implies economic impact
Severe	Lowest infection rates	Second wave and longest infection rate containment.

References

[1] Tom Britton, Frank Ball, and Pieter Trapman. "A mathematical model reveals the influence of population heterogeneity on herd immunity to SARS-CoV-2". In: *Science* 369.6505 (2020), pp. 846–849. ISSN: 0036-8075. DOI: 10.1126/science.abc6810. eprint: https://science.sciencemag.org/content/369/6505/846.full.pdf. URL: https://science.sciencemag.org/content/369/6505/846.

Program

```
import numpy as np
import scipy.integrate
import matplotlib.pyplot as plt
import pandas as pd
import os

cwd = os.getcwd()

beta = 5/8
alpha = 1/4
gamma = 1/3
N = 10000
s_inf = list()
s_0 = np.arange(N)

class Q3d():
    def __init__(self, beta, betaL, betaM, betaS):
        self.beta = beta
        self.betaL = betaL
        self.betaM = betaM
        self.betaS = betaS
        self.alpha = 1/4
        self.gamma = 1/3
        self.delta = 1/10000
        self.s_0 = 1 - self.delta
        self.e_0 = self.delta
        self.i_0 = 0
        self.r_0 = 0
        self.remI = list()
        self.remR = list()
        self.no_preventive_solve()
        # self.plott()
        self.preventive_solve(self.betaL)
        # self.plott()
        self.preventive_solve(self.betaM)
        # self.plott()
        self.preventive_solve(self.betaS)
        # self.plott()
        self.pplot()
        self.pplot_r()

    def compute(self, y, t, alpha, beta, gamma):
        S, E, I, R = y
        self.ds = -beta * S * I
        self.de = beta * S*I - gamma *E
        self.di = gamma * E - alpha * I
        self.dr = alpha * I

        return [self.ds, self.de, self.di, self.dr]
```

```
def no_preventive_solve(self):
    self.t = np.linspace(0, 255, 25500)
    self.solution = scipy.integrate.odeint(self.compute, [self.s_0, self.
        ↪ e_0, self.i_0, self.r_0], self.t, args=(self.alpha, self.beta,
        ↪ self.gamma))
    self.solution = np.array(self.solution)
    self.remI.append(self.solution[:, 2].copy())
    self.remR.append(self.solution[:, 3].copy())
    print(self.solution[-1,0]/self.s_0)

def preventive_solve(self, betaNew):
    self.betaNew = betaNew
    self.t = np.linspace(0, 30, 3000)
    self.solution = scipy.integrate.odeint(self.compute, [self.s_0, self.
        ↪ e_0, self.i_0, self.r_0], self.t,
                                           args=(self.alpha, self.beta, self.
                                               ↪ gamma))
    self.solution = np.array(self.solution)
    self.plotDataFrame = pd.DataFrame(self.solution, columns=None)

    self.t = np.linspace(30, 130, 10000)
    self.solution = scipy.integrate.odeint(self.compute, [self.solution[-1,
        ↪  0], self.solution[-1, 1], self.solution[-1, 2], self.solution
        ↪ [-1, 3]], self.t,
                                           args=(self.alpha, self.betaNew, self
                                               ↪ .gamma))
    self.solution = np.array(self.solution)
    self.plotDataFrame = self.plotDataFrame.append(pd.DataFrame(self.
        ↪ solution, columns=None))

    self.t = np.linspace(130, 255, 12500)
    self.solution = scipy.integrate.odeint(self.compute,
                                           [self.solution[-1, 0], self.solution
                                               ↪ [-1, 1], self.solution[-1,
                                               ↪ 2],
                                            self.solution[-1, 3]], self.t,
                                           args=(self.alpha, self.beta, self.
                                               ↪ gamma))
    self.solution = np.array(self.solution)
    self.plotDataFrame = self.plotDataFrame.append(pd.DataFrame(self.
        ↪ solution, columns=None))

    self.solution = self.plotDataFrame.to_numpy()
    self.t = np.linspace(0, 255, 25500)

    self.remI.append(self.solution[:, 2].copy())
    self.remR.append(self.solution[:, 3].copy())
```

```
    # def plott(self):
    # plt.figure()
    # plt.plot(self.t, self.solution[:, 0], label="S(t)")
    # plt.plot(self.t, self.solution[:, 1], label="E(t)")
    # plt.plot(self.t, self.solution[:, 2], label="I(t)")
    # plt.plot(self.t, self.solution[:, 3], label="R(t)")
    # plt.legend()
    # plt.show()

    def pplot(self):
        plt.figure()
        figNameP = cwd + "\\EXAM_Q3d_I.png"
        plt.plot(self.t, self.remI[0], label="No")
        plt.plot(self.t, self.remI[1], label="Light")
        plt.plot(self.t, self.remI[2], label="Moderate")
        plt.plot(self.t, self.remI[3], label="Severe")
        plt.legend()
        plt.xlabel("time steps")
        plt.ylabel("Fraction of infected")
        plt.title("Infected comparison")
        plt.savefig(figNameP)

    def pplot_r(self):
        plt.figure()
        figNameP = cwd + "\\EXAM_Q3d_R.png"
        plt.plot(self.t, self.remR[0], label="No")
        plt.plot(self.t, self.remR[1], label="Light")
        plt.plot(self.t, self.remR[2], label="Moderate")
        plt.plot(self.t, self.remR[3], label="Severe")
        plt.legend()
        plt.xlabel("time steps")
        plt.ylabel("Fraction of Recovered")
        plt.title("Recovered comparison")
        plt.savefig(figNameP)

obj = Q3d(5/8, 1/2, 3/8, 1/3)
```