Table of Contents

For this part of the case study the following scripts/functions were

used: base_sir_fit.m , siroutput_full.m, siroutput.m, COVIDdata.mat,

Policy_Base_sir.m, siroutput_full_policies.m

```
%%%The following code is taken from the base_sir_fit.m script
```

Model for the whole COVID data

```
load("COVIDdata.mat")

coviddata = table2array(COVID_STLmetro(:,5:6))/(100000 * STLmetroPop); % We
 changed the table to an array and only took out only the cases and deaths and
 made them into a percentage of the total population.
t = 798; % We made time to be the same as the number of rows in COVID_STLmetro

% The following line creates an 'anonymous' function that will return the cost
 (i.e., the model fitting error) given a set
% of parameters.  There are some technical reasons for setting this up in this
 way.
% Feel free to peruse the MATLAB help at
% https://www.mathworks.com/help/optim/ug/fmincon.html
% and see the sectiono on 'passing extra arguments'
```

```
% Basically, 'sirafun' is being set as the function siroutput (which you
% will be designing) but with t and coviddata specified.
sirafun= @(x)siroutput(x,t,coviddata);
```

set up rate and initial condition constraints

Set A and b to impose a parameter inequality constraint of the form A*x < b Note that this is imposed element-wise If you don't want such a constraint, keep these matrices empty.

```
A = [0,1,1,0,0,0,0];%k_fatality and k_recover must add up to one or less than
 one according to our matrix
b = 1;
```

set up some fixed constraints

Set Af and bf to impose a parameter constraint of the form Af*x = bf Hint: For example, the sum of the initial conditions should be constrained If you don't want such a constraint, keep these matrices empty.

```
Af = [0,0,0,1,1,1,1]; %the initial states must add up to one
bf = 1;
```

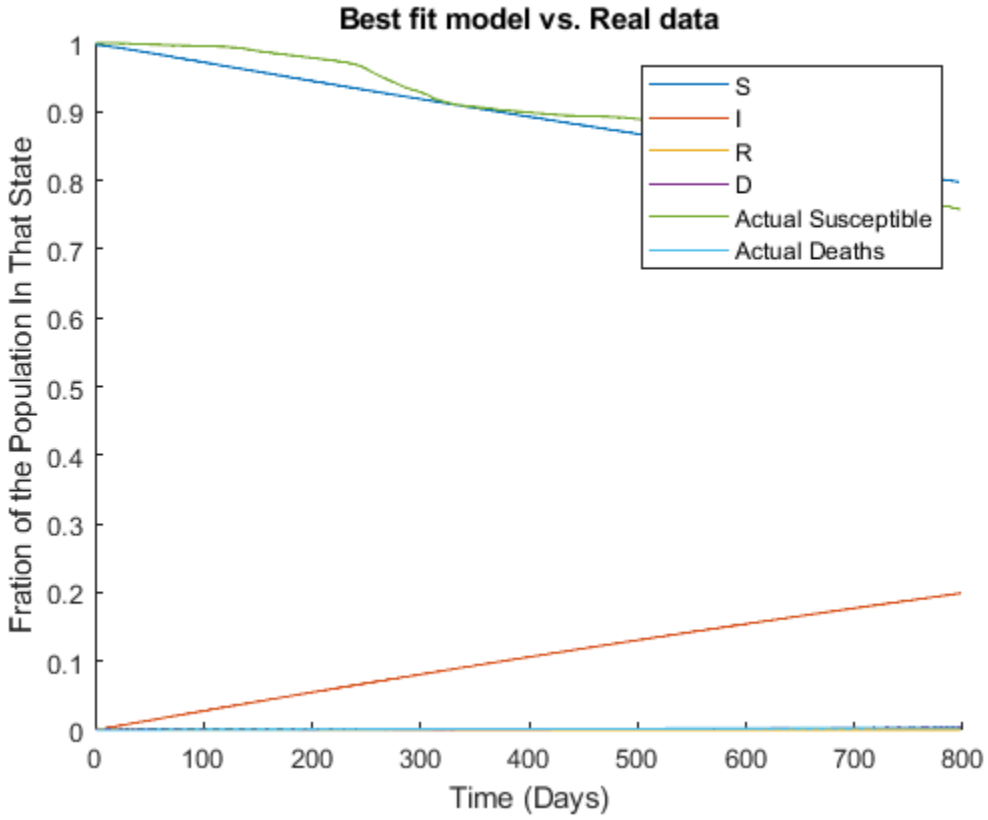
set up upper and lower bound constraints

Set upper and lower bounds on the parameters lb < x < ub here, the inequality is imposed element-wise If you don't want such a constraint, keep these matrices empty.

```
ub = [1,1,1,1,1,1,1]';
lb = [0,0,0,0,0,0,0]'; %all the rates and states must be between 0 and 1

% Specify some initial parameters for the optimizer to start from
x0 = [0.2,0.3,0.1,1,0,0,0]; %arbitrary starting values

% This is the key line that tries to opimize your model parameters in order to
% fit the data
% note tath you
x = fmincon(sirafun,x0,A,b,Af,bf,lb,ub);

%plot(Y);
%legend('S','I','R','D');
%xlabel('Time')

Y_fit = siroutput_full(x,t);

figure;
hold on
plot(Y_fit);
title("Best fit model vs. Real data")
ylabel("Fration of the Population In That State")
xlabel('Time (Days)')
plot(1-coviddata(:,1));
plot(coviddata(:,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
```

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than the value of the step size tolerance and constraints are satisfied to within the value of the constraint tolerance.



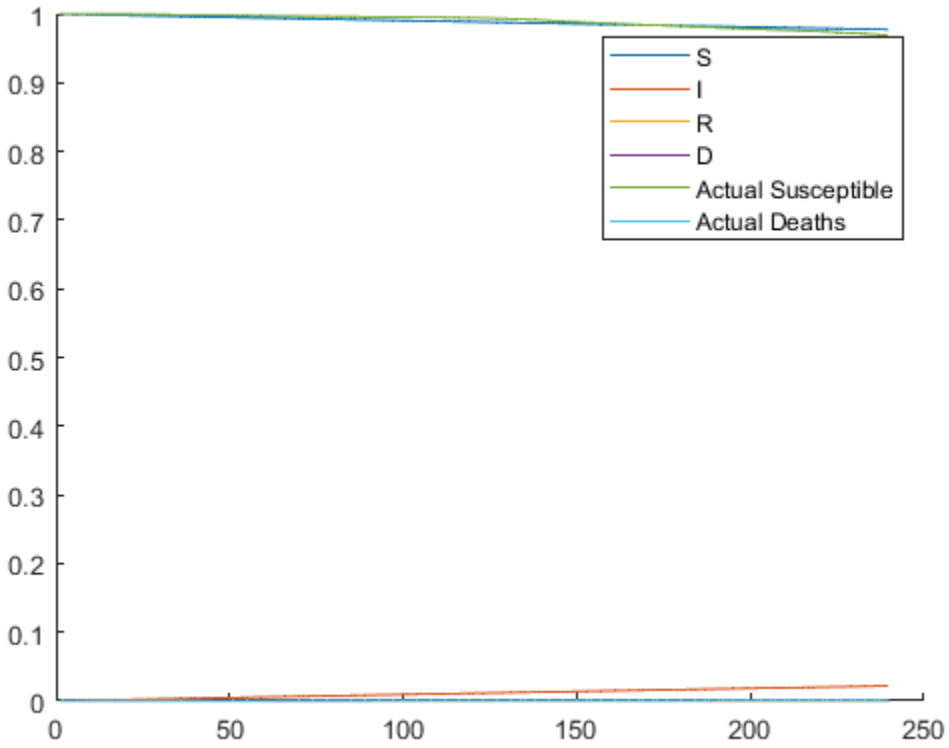
Models to fit different phases based on actual data; Phase 1, 0<t<240

```
%Looking at the model from the previous graph, coviddata seems to have 5
%distinct phases
t = 240;
sirafun= @(x)siroutput(x,t,coviddata(1:240,:));
A = [0,1,1,0,0,0,0];%k_fatality and k_recover must add up to one or less
% than one according to our matrix
b = 1;
Af = [0,0,0,1,1,1,1]; %the initial states must add up to one
bf = 1;
ub = [1,1,1,1,1,1,1]';
lb = [0,0,0,0,0,0,0]'; %all the rates and states must be between 0 and 1
x0 = [0.2,0.3,0.1,1,0,0,0]; %arbitrary starting values
x1 = fmincon(sirafun,x0,A,b,Af,bf,lb,ub);
Y_fit1 = siroutput_full(x1,t);
```

```
figure;
hold on
plot(Y_fit1);
plot(1-coviddata(1:240,1));
plot(coviddata(1:240,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
```

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than the value of the step size tolerance and constraints are satisfied to within the value of the constraint tolerance.



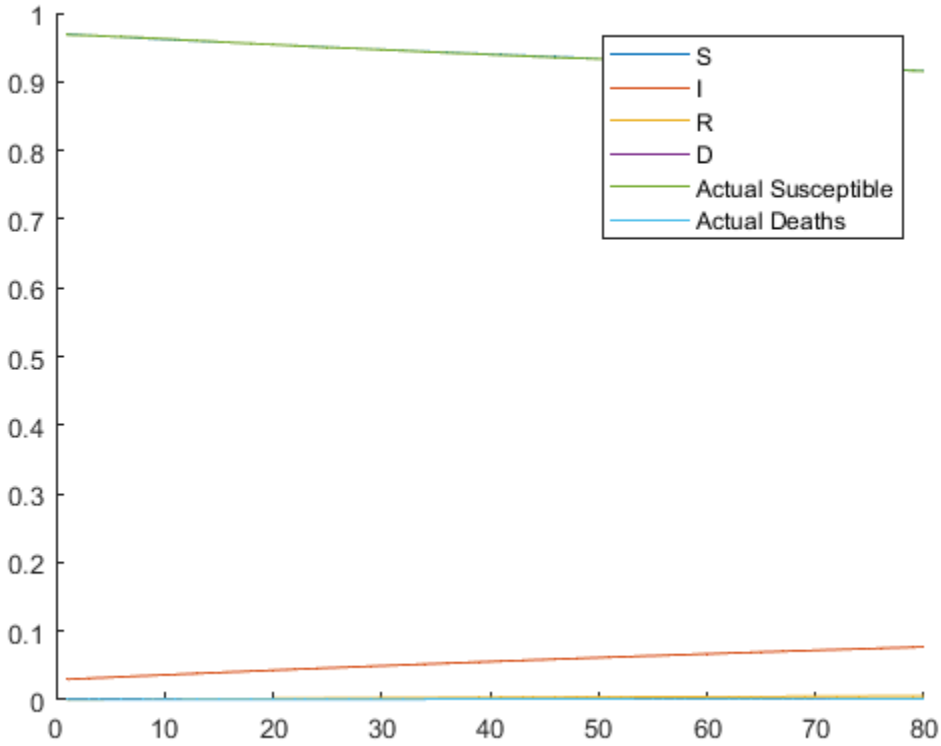
Phase 2, 240<t<320

```
t = 80;
sirafun= @(x)siroutput(x,t,coviddata(241:320,:));
A = [0,1,1,0,0,0,0];%k_fatality and k_recover must add up to one or less
% than one according to our matrix
b = 1;
Af = [0,0,0,1,1,1,1]; %the initial states must add up to one
bf = 1;
ub = [1,1,1,1,1,1,1]';
lb = [0,0,0,0,0,0,0]'; %all the rates and states must be between 0 and 1
```

```
x0 = [x1(1,1:3),Y_fit1(240,:)]; %Same rates at previous phase with initial
% conditions the same as the last day in previous phase
x2 = fmincon(sirafun,x0,A,b,Af,bf,lb,ub);
Y_fit2 = siroutput_full(x2,t);
figure;
hold on
plot(Y_fit2);
plot(1-coviddata(241:320,1));
plot(coviddata(241:320,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
```

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than the value of the step size tolerance and constraints are satisfied to within the value of the constraint tolerance.



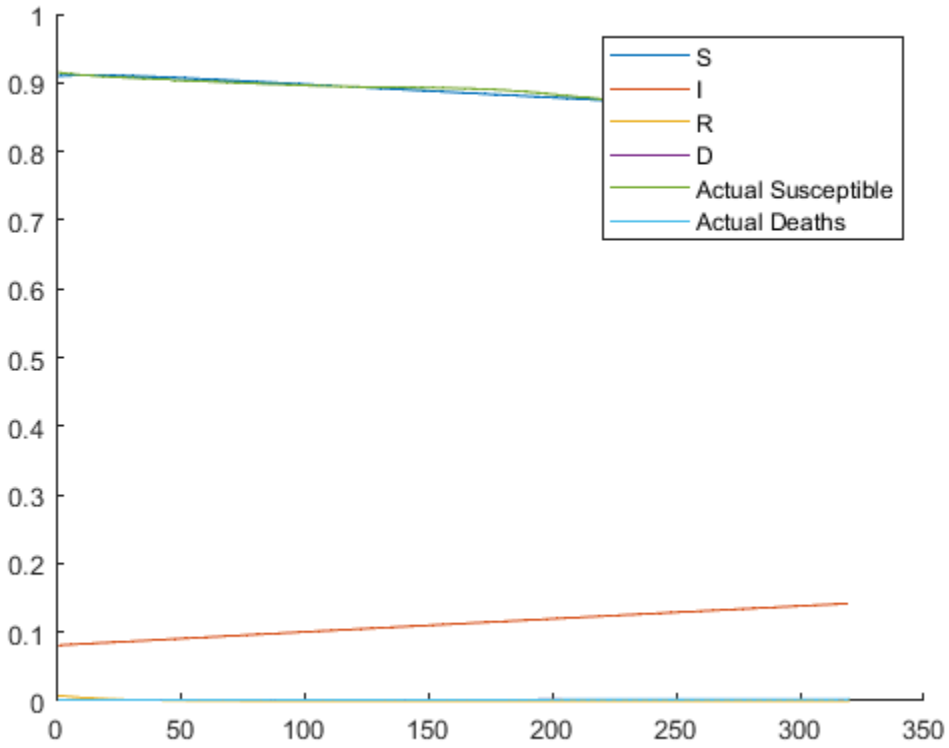
Phase 3, 320<t<640

```
t = 320;
sirafun= @(x)siroutput(x,t,coviddata(321:640,:));
A = [0,1,1,0,0,0,0];%k_fatality and k_recover must add up to one or less
% than one according to our matrix
b = 1;
```

```
Af = [0,0,0,1,1,1,1]; %the initial states must add up to one
bf = 1;
ub = [1,1,1,1,1,1,1]';
lb = [0,0,0,0,0,0,0]'; %all the rates and states must be between 0 and 1
x0 = [x2(1,1:3),Y_fit2(80,:)]; %Same rates at previous phase with initial
% conditions the same as the last day in previous phase
x3 = fmincon(sirafun,x0,A,b,Af,bf,lb,ub);
Y_fit3 = siroutput_full(x3,t);
figure;
hold on
plot(Y_fit3);
plot(1-coviddata(321:640,1));
plot(coviddata(321:640,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
```

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than the value of the step size tolerance and constraints are satisfied to within the value of the constraint tolerance.



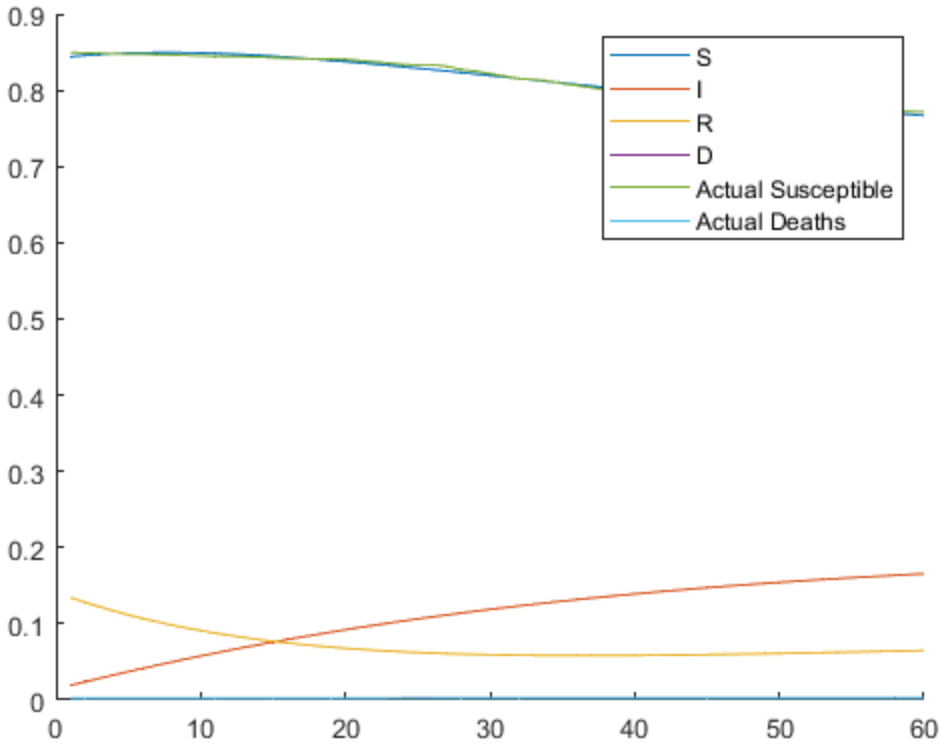
Phase 4, 640<t<700

```
t = 60;
```

```
sirafun= @(x)siroutput(x,t,coviddata(641:700,:));
A = [0,1,1,0,0,0,0];%k_fatality and k_recover must add up to one or less
% than one according to our matrix
b = 1;
Af = [0,0,0,1,1,1,1]; %the initial states must add up to one
bf = 1;
ub = [1,1,1,1,1,1,1]';
lb = [0,0,0,0,0,0,0]'; %all the rates and states must be between 0 and 1
x0 = [x3(1,1:3),Y_fit3(320,:)]; %Same rates at previous phase with initial
% conditions the same as the last day in previous phase
x4 = fmincon(sirafun,x0,A,b,Af,bf,lb,ub);
Y_fit4 = siroutput_full(x4,t);
figure;
hold on
plot(Y_fit4);
plot(1-coviddata(641:700,1));
plot(coviddata(641:700,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
```

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than the value of the step size tolerance and constraints are satisfied to within the value of the constraint tolerance.

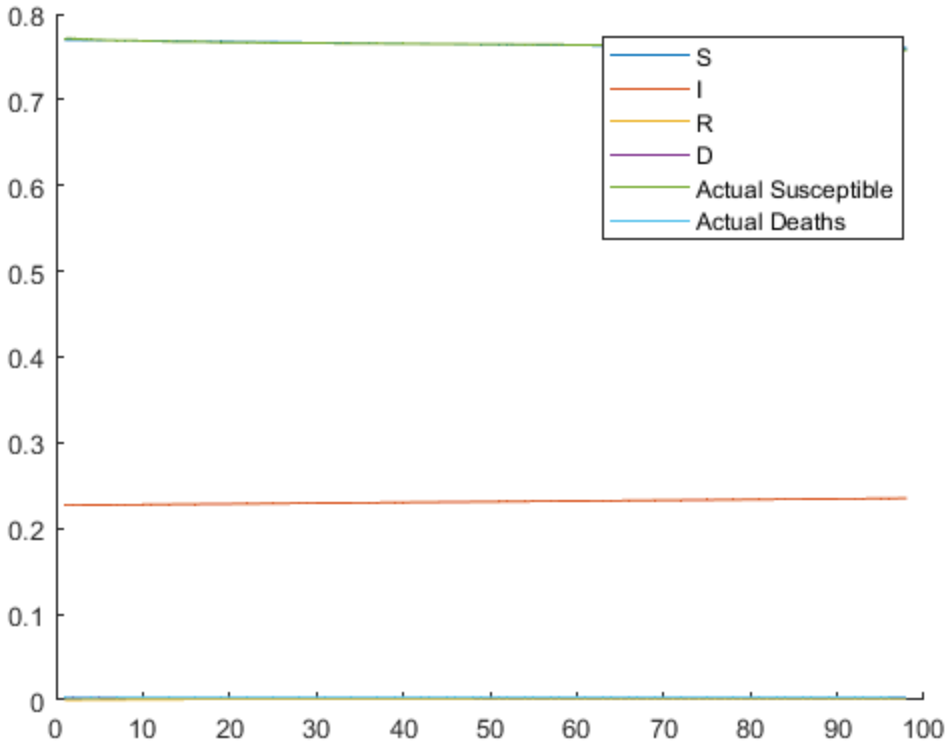


Phase 5, 700<t<798

```
t = 98;
sirafun= @(x)siroutput(x,t,coviddata(701:798,:));
A = [0,1,1,0,0,0,0];%k_fatality and k_recover must add up to one or less
% than one according to our matrix
b = 1;
Af = [0,0,0,1,1,1,1]; %the initial states must add up to one
bf = 1;
ub = [1,1,1,1,1,1,1]';
lb = [0,0,0,0,0,0,0]'; %all the rates and states must be between 0 and 1
x0 = [x4(1,1:3),Y_fit4(60,:)]; %Same rates at previous phase with initial
% conditions the same as the last day in previous phase
x5 = fmincon(sirafun,x0,A,b,Af,bf,lb,ub);
Y_fit5 = siroutput_full(x5,t);
figure;
hold on
plot(Y_fit5);
plot(1-coviddata(701:798,1));
plot(coviddata(701:798,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
```

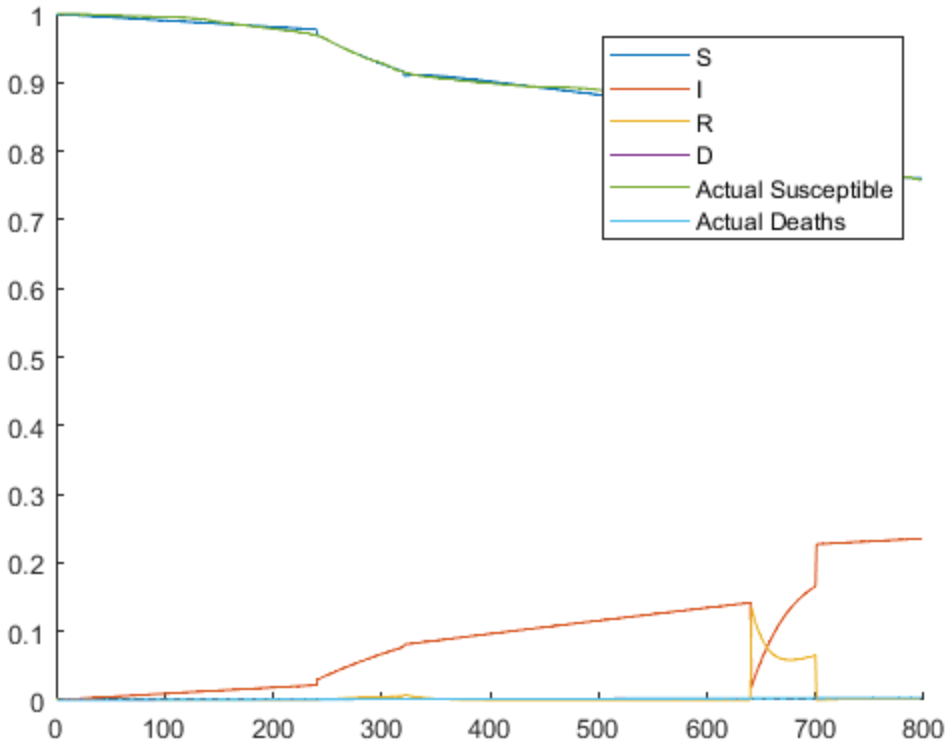
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in feasible directions, to within the value of the optimality tolerance, and constraints are satisfied to within the value of the constraint tolerance.



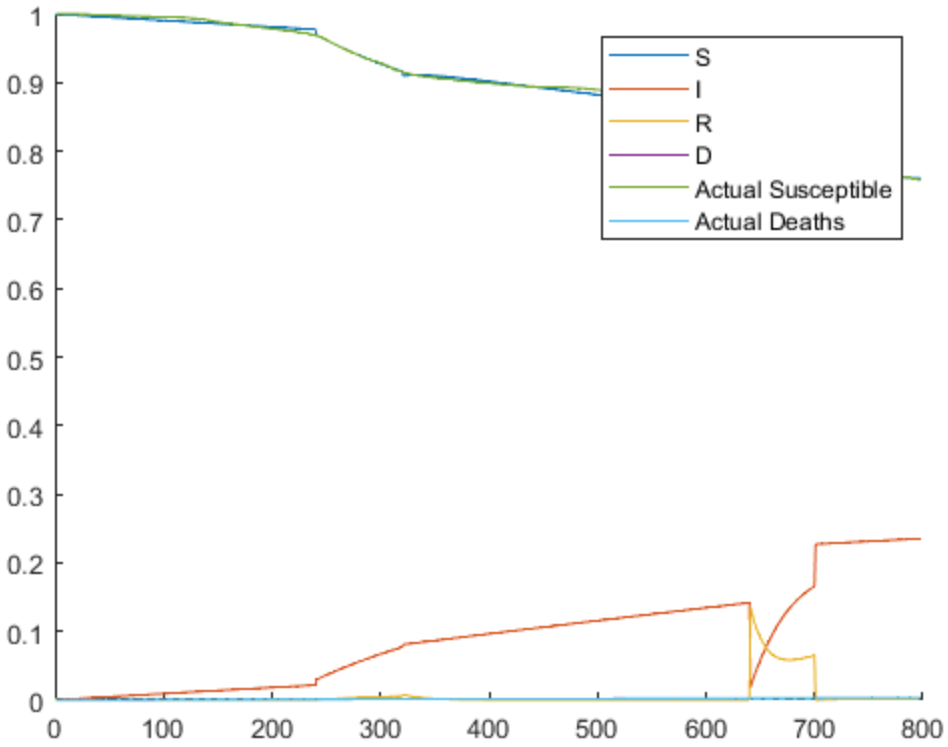
Phases Combined

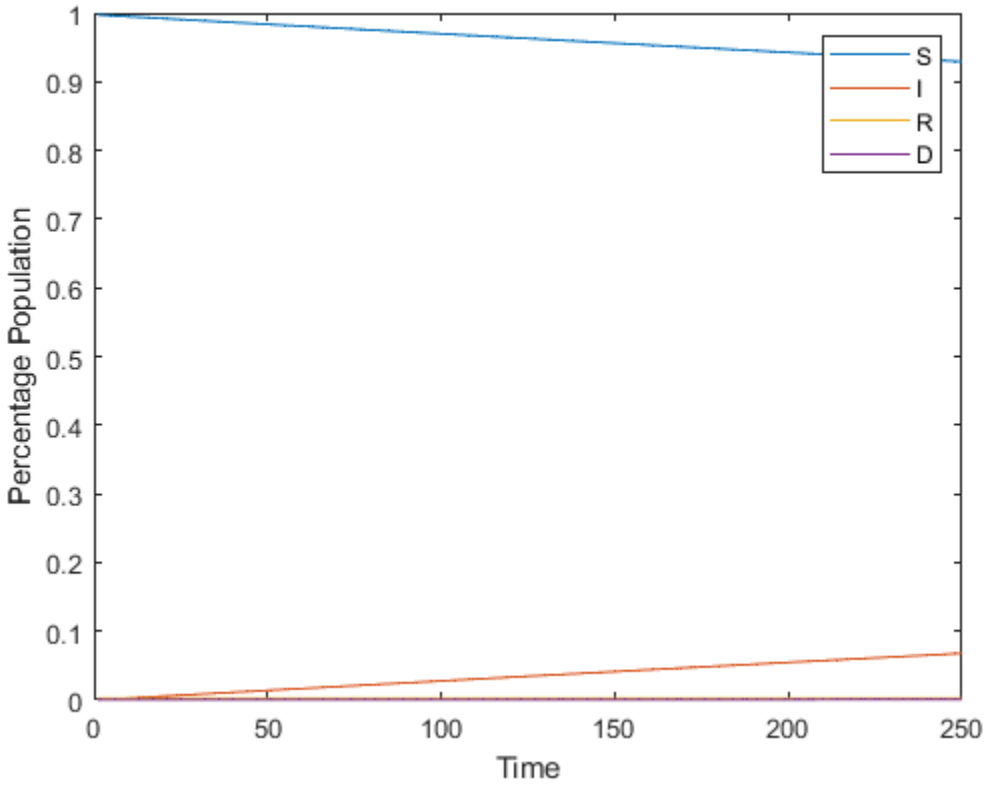
```
Y_fits_combined=cat(1,Y_fit1,Y_fit2,Y_fit3,Y_fit4,Y_fit5);
figure;
hold on
plot(Y_fits_combined);
plot(1-coviddata(:,1));
plot(coviddata(:,2));
legend('S','I','R','D','Actual Susceptible', 'Actual Deaths');
% fmincon makes the susceptible and death data match the actual data but it
% doesn't prioritize the infection data and recovery data since we weren't
% given actual data for those fractions, which is
% why there are some inconsistencies for those lines

%%%The following code is taken from the Policy_Base_sir.m script
```



Phase 1 No Mandates

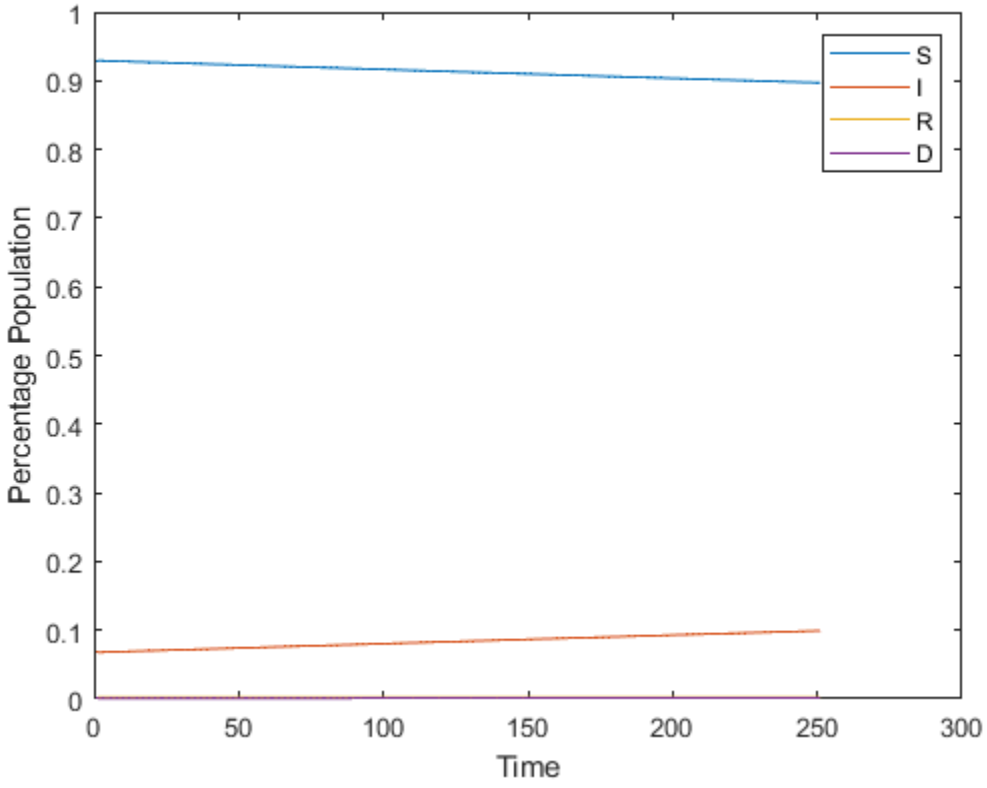
```
Phase1=siroutput_full_policies([x(1),x(2),x(3),x(4),x(5),x(6),x(7)], 250);
% We keep the rates and initial conditions as they originally are since no
% mandates were enforced during this time
figure;
plot(Phase1);
legend('S','I','R','D');
xlabel('Time')
ylabel('Percentage Population');
Phase2_starting_values=Phase1(250,:);
Phase1=Phase1(1:249,:);
```


S
I
R
D
Actual Susceptible
Actual Deaths
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
100
200
300
400
500
600
700
800


S
I
R
D
Percentage Population
Time
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
50
100
150
200
250

Phase 2 Mask Mandates

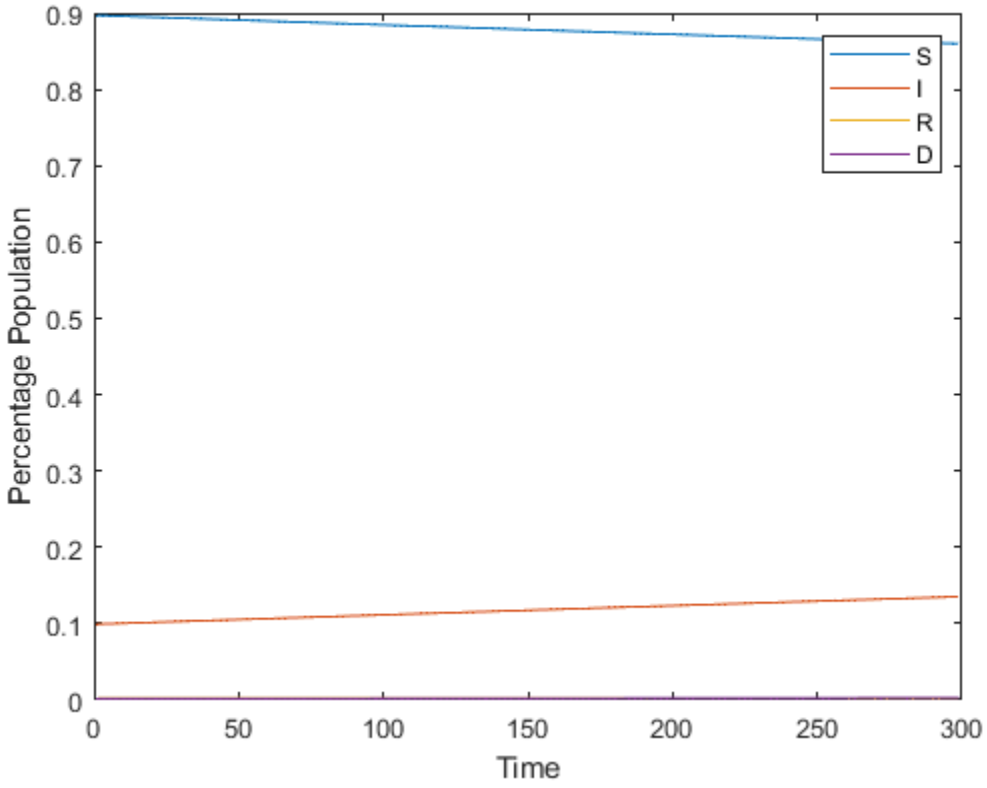
```
Phase2=siroutput_full_policies([x(1)*0.5,x(2),x(3),Phase2_starting_values(1:4)],
 251);
% We mades the initial conditions start from the last state percentages in
% the previous phase and decreased the infection rate by a factor of 0.5 due
 to
% the mask mandate while keeping the other rates the same as the previous
 phase
figure;
plot(Phase2);
legend('S','I','R','D');
xlabel('Time')
ylabel('Percentage Population');
Phase3_starting_values=Phase2(251,:);
Phase2=Phase2(1:250,:);
```



Phase 3 Vaccinations

```
Phase3=siroutput_full_policies([x(1)*0.5,x(2)*0.8,x(3)*1.5,Phase3_starting_values(1:4)],
 299);
% We mades the initial conditions start from the last state percentages in
% the previous phase. We decreased the death rate by a factor of 0.8 and
% increased the recovery rate by a factor of 1.5 due to vaccines being
% rolled out during this period. We kept the other rates the same as the
```

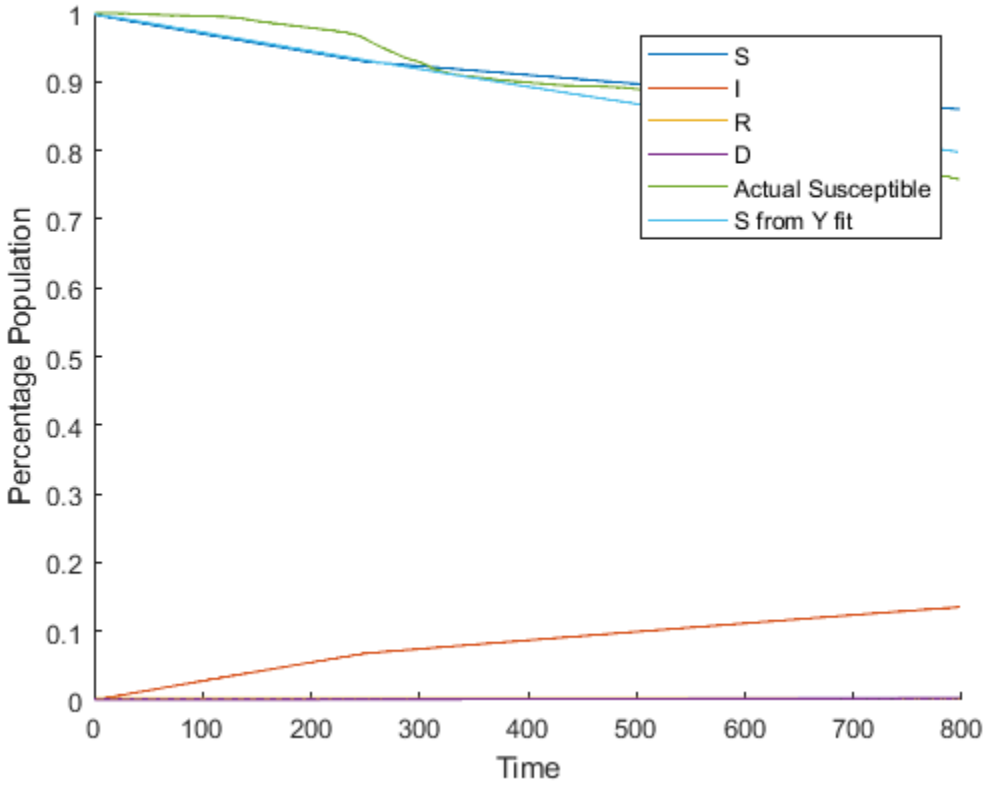
```
% previous phase.
figure;
plot(Phase3);
legend('S','I','R','D');
xlabel('Time')
ylabel('Percentage Population');
```



Phase Combined and Percent Change

```
Phase_combined=cat(1, Phase1, Phase2, Phase3); %Combined phases into 1 array
figure;
hold on
plot(Phase_combined);
plot(1 - coviddata(:,1))
plot(Y_fit(:,1))
legend('S','I','R','D','Actual Susceptible','S from Y fit');
xlabel('Time')
ylabel('Percentage Population');

Case_percentage_change=1-
(Phase_combined(798,2)+Phase_combined(798,3)+Phase_combined(798,4))/
(Y_fit(798,2)+Y_fit(798,3)+Y_fit(798,4));
%This calculates the percent change by dividing the last case
%percentages (sum of infected, deceased, and recovered fraction) in
 Phase_combined by the last state percentages in Y_fit
%(array without any mandates) and subtracting the quotient from 1
```

```
Deaths_percentage_change=1-(Phase_combined(798,4)/Y_fit(798,4));
%This calculates the percent change by dividing the last death percentage
%in Phase_combined by the last death percentage in Y_fit and subtracting
%the quotient from 1.
```



Published with MATLAB® R2022a