COVID-19 Program Suite Description and Instructions

Program Overview

The COVID-19 suite contains four Matlab programs, one that implements the mathematical model, one that runs that implementation for an individual scenario, and two that use it to set up virtual experiments.

covid19_sim.m is a function program that accepts input data called from a keyboard command or a script file and produces a table of results for the class counts in each day. The input parameters include δ, p_c, p_{ca}, and t_2, along with parameters that indicate the initial fractions of the hospitalized and immune classes.

covid19_sim_test.m is a script that runs one simulation using covid19_sim.m. A scenario is specified using the six input parameters for covid19_sim. One plot shows the susceptible, latent, combined infectious, and combined removed fractions of the population for the given scenario. Others show the numbers of US deaths in thousands and the new cases and hospitalizations per hundred thousand people. Additional outputs are the final percent susceptible, the final death count, both as thousands in the US and as a percentage of the total population, and the maximum numbers of simultaneous hospitalizations and daily new infections.

COVID19_simplot.m is a script that uses covid19_sim.m to prepare a set of plots that show the hospitalizations per hundred thousand people, thousands of US deaths, and percent susceptible for a set of scenarios specified by varying one of the parameters. Additional outputs are the final susceptible percentages, the final death totals, and the maximum hospitalizations.

COVID19_paramstudy.m is a script that uses covid19_sim.m to prepare a set of plots that show how certain key quantities change when one of the parameters is varied:

1. The maximum number hospitalized per hundred thousand people;
2. Total US deaths in thousands;
3. The final percentage still at risk at the end of the outbreak;
4. The number of days to reach the maximum of new infections and a target threshold taken as the end of the outbreak.

Getting Started

We assume here that the reader already knows how to use the Matlab editor to create and execute a script file, which is a set of instructions to be executed according to a prescribed flow of control. No significant programming experience is required. Users should start with covid19_sim_test.m, which is designed to ensure that the main program is functioning correctly and to provide an easy entry into Matlab programming.

covid19_sim_test is a simple program, with no complicated programming structures such as nested loops or conditionals. The program is organized into a structure that facilitates use and modification by novice programmers. This structure is implemented by a set of section dividers (lines that start with %%), with additional help provided by comments (lines that start with %), and blank lines.

The SCENARIO DATA section contains statements that assign values to the six parameters that get sent to covid19_sim. These can be changed to observe the effect they have on the results.

The INITIALIZATION section contains statements that set up the plots that will come later. It is always a good idea to start a graph with "clf" and "hold on", which clear out any old graph and tell Matlab not to erase any subsequent plots without special instructions to do so. The "opengl" statement avoids some graphics bugs that occur on some computers. Some programs have data structures that are initialized in this section as well.

The COMPUTATION section is primarily just the function call to covid19_sim, where the real work is done out of sight. There are also a couple of statements that use the daily change in the susceptible population to make a vector of new infection counts.

The OUTPUT section contains a lot of graphics statements and a few statements that print results in the Command Window. Matlab graphics are designed so that most of the graph formatting is done automatically, although these details can be specified. An example is the "legend" statement. Usually Matlab makes a good choice of location without the programmer's help, but it is often best to specify where you want the legend using direction names such as "Northeast".

covid19_sim_test.m can be used to run any scenario for the COVID19 epidemic model simply by changing the values in the scenario data section. Sometimes you will want to tinker with graphics statements to improve the appearance and/or design of a plot.

Using COVID19_simplot and COVID19_paramstudy

These scripts are constructed so that they can be applied to different experiments with a minimum of changes. The changes are identified in the comments of the script files, and are described here in more detail.

The logical structure is similar to that of covid19_sim and covid19_sim_test; however, the user needs to understand some of the details in order to adapt the program for different experiments.

1. The scripts begin with some brief documentation that describes the program and how to use it and, most importantly, identifies the particular version of covid19_sim that it has been tested on. The script may work with newer or older versions, depending on what changes were made, but one should not expect this.

2. The first programming section contains default scenario data. Not all of the default values will be used, depending on the experiment.

3. The next section contains the data for the experiment parameter, described in the program as the "Independent Variable". The program is designed to require minimal changes for different experiments; most of those changes are in this section.

    (a) First, the set of parameter values to be used for the independent variable must be specified with a generic name. The actual parameter you have chosen will be identified elsewhere.

        i. The values are listed in COVID19_simplot under the generic name 'xvals' and are specified individually using Matlab syntax for a row vector or list. Each of the parameter values in this list will generate its own set of curves in the three-panel plot. Generally a set of 3 to 8 values is best.

ii. In COVID19_paramstudy, the independent variable values are used as the horizontal coordinate on each plot, so a much larger set is necessary. The user specifies the first value, the last value, and the count of values, keeping in mind that the end point values are both counted. For example, if you want the values 0, 0.05, 0.1, and so on up to 1.0, you choose `first=0`, `last=1`, and `N=21`, dividing the interval from 0 to 1 into 20 equal parts.

(b) The set of values is the only thing that needs to be defined in the INDEPENDENT VARIABLE DATA section of COVID19_simplot. COVID19_paramstudy also requires the user to specify the name of the parameter to be used as the horizontal axis label. This specification only applies to the graph; so far the program does not know which of the parameters is to take on the values predefined using 'first', 'last', and 'N'.

4. Next comes the INITIALIZATION section, which does basic housekeeping tasks such as setting up the plot windows and defining any data structures needed for the data.

5. The programs conclude with sections for computation and output, with the graphs being created in the computation section if they occur inside a loop or in an output section if they are only produced after all the data is collected.

6. The key to the program structure is that each parameter has a default value that will be used in all scenarios, except that one of the parameters will instead use the values specified either as 'xvals' or using 'first', 'last', and 'N'. The function call requires one value to be passed in for each of the parameters. The first line inside the main program loop identifies the name of the parameter whose values were defined in the Independent Variable section. This line overwrites the previous value each time through the loop; meanwhile, the other parameters have only been given default values, so these are also used. The default value for the independent variable parameter is never used, but having it specified in the data section allows for a minimum of changes as the program is repurposed for a new experiment.

7. As with covid19_sim_test, the user might want to tinker with plot details, perhaps by overriding Matlab's choice of axis limits, modifying a legend, or adding text.

Changes for the Limited Lockdown Experiment

If you are asked to do the Limited Lockdown experiment, you will need to make some changes to the program suite. These changes introduce a parameter called 'lockdays' that indicates the amount of time δ will be kept at its default value before reverting to $\delta = 1$ at the end of the lockdown. This of course assumes that there will be no further social distancing or mask usage.

1. Save a copy of COVID19_simplot.m as COVID19_simplot2.m. Then change the variable association statement in line 73 to

   ```
   lockdays = xvals(n);
   ```

   and the function call in line 75 to

   ```
   [S,~,~,~,H,~,D,R0] = covid19_sim2(lockdays,delta,pc,pca,t2,H0,V);
   ```

2. Save a copy of covid19_sim.m as COVID19_sim2.m. Change the first line of the program to match the function name and argument list that are called in COVID19_simplot2.m.

3. The COMPUTATION section needs some significant changes:

   (a) Some code needs to be inserted that sets delta to 1, either because there is no lockdown or because the lockdown has elapsed. These are two different triggers, so they must be implemented separately.

   (b) Code needs to be inserted so that the check for the end condition (the 'if-else' construction at the end of the 'for' loop) is only checked after the lockdown.

   Implement these changes by adding a short 'if' statement at the beginning of the section and editing the long 'for' loop that is already there. The resulting code is provided for you here.

```
if lockdays==0
    delta = 1;
end

for t=1:maxdays
    \% y is a column vector, Y^T
    y = rk4(1,y);
    Y = y';
    results(t+1,:) = Y;
    if t==lockdays
        delta = 1;
        summ = sum(Y(2:6));
    end
    if t>lockdays
        if sum(Y(2:6))>min(target,summ)
            summ = sum(Y(2:6));
        else
            results = results(1:(t+1),:);
            break;
        end
    end
end
```

Changes for the Herd Immunity Experiment

If you are asked to do the Herd Immunity experiment, you may need to make some changes to COVID19_paramstudy. These changes introduce an outcome called 'infected' that computes the fraction of the initially susceptible who are ultimately infected. This outcome replaces the time plots in the lower right panel.

1. Save a copy of COVID19_paramstudy.m as COVID19_paramstudy2.m.

2. At the end of the INITIALIZATION section, add the line

```
infected = zeros(1,N);
```

3. At the end of the loop in the COMPUTATION section, add the line

```
infected(n) = ...;
```

where '...' stands for the correct formula. This formula uses entries in the susceptible list 'S' and calculates the fraction of the initial susceptibles who get sick during the scenario. Note that you can access the last element of the list using 'S(end)'.

4. The statements for subplot(2,2,4) in the OUTPUT section need to be changed.
   (a) Replace 'maxnewtime' with 'infected' in the first 'plot' statement after the first 'subplot(2,2,4)'.
   (b) Delete the two subsequent 'plot' statements and the 'legend' statement.
   (c) Change the 'ylabel' statement in the last line from 'days' to 'fraction infected'.