iGEM Intro to Math Modeling Demo

Compiled by Wangui Mbuguiro

PART 1: WHAT IS MATH MODELING?

Mathematical modeling is the process of describing a system using variables and equations in order to improve our understanding and predict behavior (Wikipedia overview). You have likely already had experience building mathematical models, such as linear regressions (which are statistical models) and the Game of Life (which is a mechanistic model). In biomedical research, math modeling has aided in improving experimental design, optimizing therapeutics, and investigating the underlying mechanisms of diseases and treatments. For iGEM, we will build mechanistic models, which focus on modeling the interactions between components in a biological system.

PART 2: SIR ROLE PLAYING GAME

One application of mechanistic modeling is the Susceptible-Infected-Recovered **(SIR)** used in epidemiology to study the spread of infectious disease. We will play a role-playing game to demonstrate the spread of the flu, then model it in MATLAB using the equations described and code provided in parts 3 and 4.

To start, all except for one of you will be labelled as susceptible. You will move throughout the room until indicated to pause. Upon pausing, you will be asked to interact with your nearest neighbor. Two things may occur:

1. If your neighbor is not infected, then you will remain susceptible.
2. If your neighbor is infected, then you will each flip a coin. If you both land on the same side of the coin, you will also become infected.

If you become infected, report it to the calculators at the board. Over time, random people will be selected to become recovered. If you are recovered, you will still continue in the game, but you can no longer infect others or become infected upon interaction. Note, we are ignoring the effects of vaccination, death, birth, quarantine, etc to simplify the game and model.

The calculators at the board will be filling in the follow table and produce a plot for the data.

Title	SIR Data from BUGSS iGEM		
Time Step	Susceptible_	Infected_I	Recovered_R
0	9	1	0
1			
2			

⇒ What do you expect to happen to each population over time?

After looking at the resulting plot, consider these questions:

⇒ How does this flu compare to others?
⇒ What is the most effective way to stop the spread of flu?

In order to explore these questions, we will create a model, simulate the data, and then conduct computational analyses.

PART 3: SIR MODEL & SIMULATION

Approach to mechanistic model of SIR
To model this system, we start by considering the processes that the susceptible, infected and recovered groups are a part of. We will represent them here as chemical reactions, where an arrow represents reactants going to products.

Process	**Reaction**
Infection	$S + I \xrightarrow{b} I + I$
Recovery	$I \xrightarrow{k} R$

Here, **b & k** represent the **rate constants** for infection and recovery, which we will talk more about as we simulate this system.

Simulating the SIR model
We will be using MATLAB, a matrix-based language and programming platform. Open the **SIR_main.m** file in MATLAB and start exploring it. The **SIR_main.m** is where we specify our parameters. Parameters can include how many susceptible, infected, and resistant people we start with, we call these our initial conditions **(S, I, R).** We can also define how fast each process is going, also known as the rate constants **(b and k).** You can also specify the time for the simulation **(Tspan)**. Try changing these values.

```
% SIR_main.m
% Code to model Susceptible, Immune, and Recovered groups during epidemic

% Define Initial Conditions - (ADJUST THIS SECTION)
S = 9;
I = 1;
R = 0;
y0 = [S, I, R];

% Define Rate Constants - (ADJUST THIS SECTION)
b = 1/4; % Infection rate constant
k = 1/5; % recovery rate constant
p = [b, k];

% Define Simulation Time - (ADJUST THIS SECTION)
Tspan = 25;
```

Note, although you can specify the rate constants, the rate that each process is occurring will still depend on the number of people who are susceptible, infected, and recovered, thus it will change over time (e.g. more infected → faster infection rate). We describe the rate of change in the number of people in each group over time using **ordinary differential equations** (ODE's) in the **SIR_eqns.m** file.

Open the **SIR_eqns.m** file. The ordinary differential equations written for this system look as follows:

Change in # susceptible over time: $\frac{dS}{dt} = -b * S * I$

Change in # infected over time: $\frac{dI}{dt} = +b * S * I - k * I$

Change in # recovered over time: $\frac{dR}{dt} = +k * I$

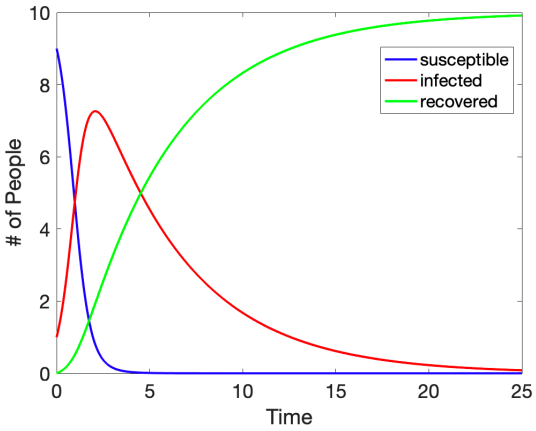
In the equations, the signs, + or -, represent whether a group gains or loses people. From these equations, you'll notice that certain rates repeat reciprocally. For instance, loss from the susceptible group is equal to gains in the infected group. The same relationship holds true for infected and recovered groups.

We use an ode solver **(ode45)** in **SIR_main.m** to solve these equations.

```
% Run Simulation
options = odeset('MaxStep',5e-2, 'AbsTol', 1e-5,'RelTol', 1e-5,'InitialStep', 1e-2);
[T,Y] = ode45(@SIR_eqns,[0 Tspan],y0,options,p);
```

Ode45 is a function that takes in your initial conditions (**y0**) and reaction rate constants (**p**) and solves your set of ordinary differential equations over time. Ode45 returns the value of your solutions (**Y**; # of susceptible, infected, and recovered) at each time step (**T**).

Once you've defined your parameters and run the **SIR_main.m** file, you should get the following output. Try adjusting b & k and see how the plot changes.



PART 4: ANALYSIS

Defining the equations and parameters is just the start. Once we have created the model, we can begin analyzing our system using computational methods. Two common computational methods include **optimization** and **sensitivity analysis**.

Optimization

In part 3, we are simulating our system using guessed values for the infection and recovery rate constants **(b & k).** Optimization is the process of estimating those values from observed data.

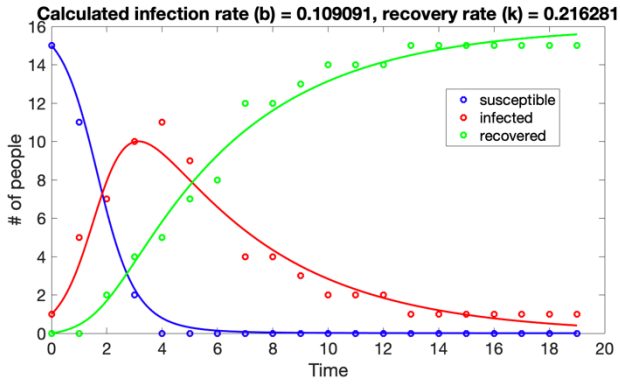
Open **SIR_optimization.m** file. The code for optimization uses parts from the simulation code (**SIR_main.m**) described above. You will learn more about the construction of this code during iGEM. For now, modify the observed data **(texp, Sexp, Iexp, Rexp)** using data collected in class.

```
% SIR_optimization.m
% Code to optimize parameters for model

GroupNames = {'Susceptible'; 'Infected'; 'Recovered'};

% Define experimental data (ADJUST THIS SECTION WITH YOUR DATA)
texp = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11,12,13,14,15,16,17,18,19]';
Sexp = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]';
Iexp = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]';
Rexp = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]';
yexp = [Sexp, Iexp, Rexp];
```

Run the **SIR_optimization.m** code and you should get an output similar to the following:



Results from optimizing data from a DrEAMM activity created by the Finley Lab at the University of Southern California. Experimental data are plotted as points, and the simulations result (using optimized rate constants) are plotted as solid lines.

The estimations for infection and recovery rate constants **(b & k)** are included in the plot title. These rate constants can be useful in comparing scenarios (different flu's or interventions, etc) as well as simulating additional scenarios (future time, interventions like quarantine, etc).

In your code, there was also data provided from a DrEAMM activity created by the Finley Lab at the University of Southern California (results pictured above). Try comparing the two flu's in terms of infectiveness and recovery.

Sensitivity Analysis

Sensitivity analysis is a method that predicts which inputs (e.g. initial population sizes, infection and recovery rate constants) have the largest impact on certain outputs (e.g. maximum # infected, time until max infected) of our model. Sensitivity analysis works by comparing the output(s) for our base case simulation (the case that uses our defined inputs) to several perturbation cases (cases that vary the inputs provided). This is another computational method you will learn about through iGEM.

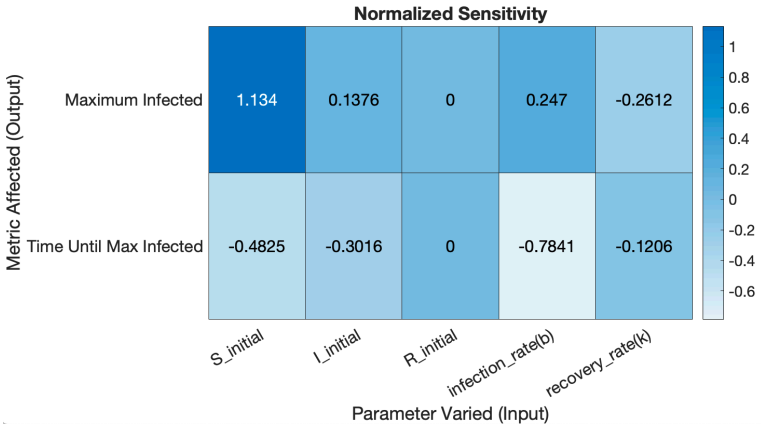
Open **SIR_sensitivity.m** file. Set the initial conditions and reaction rate constants. This time you can use the rate constants optimized to your data!

```
% SIR_sensitivity.m
% Code to test sensitivity of Susceptible, Immune, and Recovered model to
% various parameters

% Define Initial Conditions (ADJUST THIS SECTION)
S = 0;
I = 0;
R = 0;
y0 = [S, I, R];

% Define Rate Constants  (ADJUST THIS SECTION)
b = 0; % Infection rate constant
k = 0; % recovery rate constant
p = [b, k];
```

When you run this file, you should get an output similar to the following:



This heat map shows the result of varying one parameter at a time and comparing each case to the base case (also known as a "local univariate sensitivity analysis"). Outputs are on the y-axis and inputs are on the x-axis. The shade of each box represents a *normalized* result which indicates how much varying each input affected the output (magnitude) and in what direction (sign).

Sensitivity analysis is useful in hypothesis generation to start answering the questions such as "what is the most effective way to stop the flu?" We can also build upon our model, adding terms to represent processes such as vaccination or isolation, to better reflect our system or predict the effects of these interventions.

HELPFUL RESOURCES

Activity concept and data taken from a DrEAMM activity created by Dr. Stacey Finley at the University of Southern California which was presented at IMAG 2019.
https://www.imagwiki.nibib.nih.gov/sites/default/files/Finley%202019-03-06_MSM%20K12%20Dissemination.pdf

SIR Math explanations:
http://www.stat.columbia.edu/~regina/research/notes123.pdf

David Smith and Lang Moore, "The SIR Model for Spread of Disease - The Differential Equation Model," Convergence (December 2004)
https://www.maa.org/press/periodicals/loci/joma/the-sir-model-for-spread-of-disease-the-differential-equation-model