

Parameter Identification and SIR Epidemic Model.

In this lesson we consider the development of an epidemic, which is similar to measles. The population will be partitioned into three disjoint groups of susceptible, infected and recovered. One would like to precisely determine what parameters would control or prevent the epidemic. The populations will be modeled by three differential equations. The Matlab command ode45 can be used to solve such systems of differential equations.

Make the following basic assumptions. Assume the population is a disjoint union

$$\text{susceptible} = S(t),$$

$$\text{infected} = I(t) \text{ and}$$

$$\text{recovered} = R(t).$$

So, the total population = $S(t) + I(t) + R(t)$. Assume all infected eventually recover.

Assume all recovered are not susceptible. Assume the increase in the infected is proportional to

the change in time, the number of infected and the number of susceptible.

The change in the infected population will increase from the susceptible group and will decrease into the recovered group.

$I(t+\Delta t) - I(t) = \Delta t a S(t) I(t) - \Delta t b I(t)$ where a reflects how contagious or infectious the epidemic is and b reflects the rate of recovery. Now divide by Δt and let it go to zero to get the differential equation for I , $I' = a S I - b I$. The differential equations for S and R are obtained in a similar way.

SIR Epidemic Model:

$$S' = -a S I$$

$$S(0) = S_0 \text{ and } a = \text{contagious constant}$$

$$I' = a S I - b I$$

$$I(0) = I_0 \text{ and } b = \text{recovery constant}$$

$$R' = b I, R(0) = R_0.$$

Note, $(S + I + R)' = S' + I' + R' = 0$ so that $S + I + R = \text{constant}$ and

$$S(0) + I(0) + R(0) = S_0 + I_0 + R_0.$$

Note, $I'(0) = (a S(0) - b) I(0) > 0$ if and only if

$$a S(0) - b > 0 \text{ and } I(0) > 0.$$

The Matlab command `ode45` is used to approximate the solution of our system of differential equation. This command is a robust implementation for systems of differential equations, which uses a variable step size method and the fourth and fifth order Runge-Kutta method.

Since there are three unknowns and three differential equations and we wish to use Matlab's `ode45` scheme, the file, `ypsir.m`, must contain the three right sides of the differential equations where $y(1) = S$, $y(2) = I$ and $y(3) = R$. The choice of a and b in `ypsir.m` will determine whether or not the $I(t)$ will increase; if $I'(0) = (a S(0) - b) I(0) > 0$, then $I(t)$ will increase. The `sir.m` file contains the call to `ode45` and the graphs of the three population groups are generated by the Matlab command `plot`. The initial populations are $S(0) = 99$, $I(0) = 1$ and $R(0) = 0$.

m-files ypsir.m

```

function ypsir = ypsir(t,y)
a = .01;
b = .1;
ypsir(1) = -a*y(1)*y(2);
ypsir(2) = a*y(1)*y(2) - b*y(2);
ypsir(3) = b*y(2);
ypsir = [ypsir(1) ypsir(2) ypsir(3)];

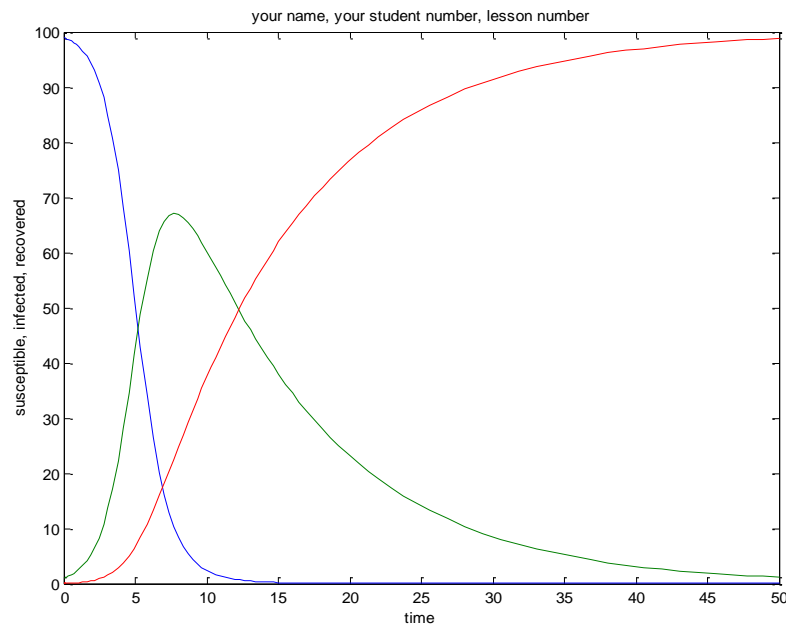
```

m-files sir.m.

```

% your name, your student number, lesson number
clear;
to = 0;
tf = 50;
yo = [99 1 0];
[t y] = ode45('ypsir',[to tf],yo);
plot(t,y(:,1),t,y(:,2),t,y(:,3))
title('your name, your student number, lesson number')
xlabel('time')
ylabel('susceptible, infected, recovered')

```



Note, the three populations versus time give the output. Also the steady state populations are indicated by the right side of the graph where the three curves level off, that is, where the derivatives all approach zero.

The following code identifies the contagious and recovery parameters, a and b , from several observations of the percentage of infected and recovered in the populations. The system of differential equations is discretized to form an over-determined algebraic system. The least squares method is used to approximate the two parameters. The Matlab code `sir_parid.m` is as follows and lists the two subroutines `sirid()` and `ypsirid()`:

Matlab code `sir_parid.m`:

```
% This code uses least squares to identify two parameters in the
% SIR model:
%   S_t = -aSI; I_t = aSI - bI; R_t = bI where
%           a = "contagious" coefficient and
%           b = "recovery" coefficient.
% The data is given in the vectors Sd, Id and Rd,
% and they are adjusted by a random variable.
% The data is used in the finite difference approximation of the above:
%   (S_{i+1} - S_{i-1})/(2 dt) = -a S_i I_i
%   (I_{i+1} - I_{i-1})/(2 dt) = a S_i I_i - b I_i and
%   (R_{i+1} - R_{i-1})/(2 dt) = b I_i.
% Least squares is used to compute the linear polynomial coefficients.
% The first six data points are used.
%
%function [t y] = sirid
%   global olda oldb
%   yo = [99 1 0];
%   to = 0;
%   tf = 50;
%   [t y] = ode45('ypsirid',[to tf],yo);
%
%function ypsirid = ypsirid(t,y)
%   global olda oldb
%   ypsirid(1) = -olda*y(1)*y(2);
%   ypsirid(2) = olda*y(1)*y(2) - oldb*y(2);
%   ypsirid(3) = oldb*y(2);
%   ypsirid = [ypsirid(1) ypsirid(2) ypsirid(3)]';
%
clear; clf(figure(1))
global olda oldb
olda = 0.010; oldb = 0.100;
td = [0.00 1.077 2.136 3.092 4.171 5.372 6.695 8.020 9.015...
      10.293 11.271 15.039 20.590];
% Sd = [99.0 97.2 92.9 84.8 68.1 42.8 20.0 8.3 4.3 2.0 1.1 0.2 0.04];
Id = [1.00 2.59 6.39 13.65 28.20 48.76 64.00 66.93 64.38 ...
      58.85 54.15 37.88 21.85];
Rd = [0.00 0.28 0.63 1.55 3.74 8.37 15.97 24.76 31.30 39.19...
      44.72 61.92 78.11];
numdata = 13;
rvec = rand(1,numdata);
Id(2:numdata) = Id(2:numdata) + .1*rvec(1,2:numdata) - .05;
```

```

rvec = rand(1,numdata);
Rd(2:numdata) = Rd(2:numdata) + .1*rvec(1,2:numdata) - .05;
Sd = 100 - Id - Rd;
%
for i = 2:1:numdata-1
    ii = (i-1)*3;
    d(ii) = (Sd(i+1) - Sd(i-1))/(td(i+1) - td(i-1));
    d(ii+1) = (Id(i+1) - Id(i-1))/(td(i+1) - td(i-1));
    d(ii+2) = (Rd(i+1) - Rd(i-1))/(td(i+1) - td(i-1));
    A(ii,1) = -Sd(i)*Id(i); A(ii,2) = 0;
    A(ii+1,1) = Sd(i)*Id(i); A(ii+1,2) = -Id(i);
    A(ii+2,1) = 0.0; A(ii+2,2) = Id(i);
end
%
meas = 6;
m = 3*meas + 1;
x = A(2:m,:) \ d(2:m)';
%
[olda oldb]
[x(1) x(2)]
plot(td(1:1:meas+1),Sd(1:1:meas+1),'*',td(1:1:meas+1),Id(1:1:meas+1),'o'
,....
      td(1:1:meas+1),Rd(1:1:meas+1),'s',td,Sd,'x',td,Id,'x',td,Rd,'x')
olda = x(1);
oldb = x(2);
[t y] = sirid;
hold on
plot(t, y(:,1), 'b',t, y(:,2), 'g',t,y(:,3), 'r')

```

Use five measured data points:

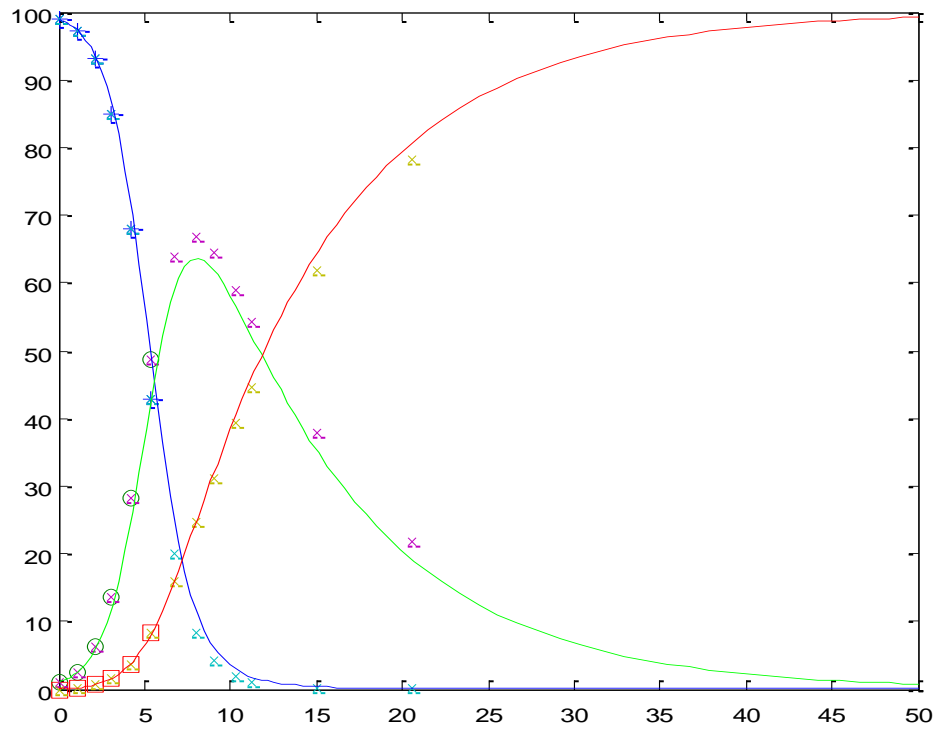
```
>> sir_parid
```

```
ans =
```

```
0.0100    0.1000
```

```
ans =
```

```
0.0096    0.1112
```



Use six measured data points:

```
>> sir_parid
```

```
ans =
```

```
0.0100    0.1000
```

```
ans =
```

```
0.0096    0.0986
```

