Lab: Epidemic Spread on Graphs

In this lab we will implement and visualize the spread of an epidemic on the 7-node network we saw in lecture. Note that a similar approach can be used for simulating the spread of a rumor in a social network.

For this lab, you will need files `adjacency.mat` and `spread.m`. If these files are not already on your Desktop, open the **ggmshared** folder and drag them from there to your Desktop. Next, open Matlab and change your directory to Desktop by navigating to **home/icermprime/Desktop**.

1. Create a Matlab code called `epidemic.m` with the code below:

```
clear all; close all;
load('adjacency.mat')%adjacency matrix
G = graph(A); %create graph G from adjacency matrix A
figure(1)%plot the graph
plot(G,'-o','Layout','force','Linewidth',2,'EdgeColor','k','MarkerSize
',10,'NodeColor','b');
```

   This code loads the adjacency matrix for the 7-node network, creates a graph object in Matlab, and plots the network. You do not need to copy any of the green text: this text that comes after the percentage signs represents comments that help us keep track of and understand our program, and is ignored by Matlab.
   After you run the program and plot the graph, determine which node has the highest degree = the largest number of edges (Note that you do not need to use Matlab for this task). Then write down this node number in the space below.

   The node with the highest degree is: _

2. We will now assume that the node you found in the previous problem is the first that gets infected with the disease. Add the lines of code below at the end of your `epidemic.m` program, and fill in the blank in the second line below with the node number from problem 1:

```
status = zeros(1,7); %vector of infection status: 1 for infected individuals
, 0 for healthy ones
index = _; %index of infected seed from previous problem
status(index)= 1; %the most central node is infected
inf_nodes = find(status == 1); %vector of infected nodes
```

Read (but no need to write down) the comments after the percentage signs to understand the commands above.

Run `epidemic`. What is the vector `status`? How about `inf_nodes`? Make sure you understand that `status` corresponds to the status of each individual in the network at a given time, while `inf_nodes` corresponds to the set of nodes that are infected at a given time.

3. Visualize the start of the infection by adding the following code at the end of `epidemic` and running the program:

```
figure(1)
h = plot(G,'-o','Layout','force','Linewidth',2,'EdgeColor','k','MarkerSize
',10,'NodeColor','b');
highlight(h,inf_nodes,'NodeColor','r');
pause(0.5);
```

Note that the healthy nodes are in blue, while the infected ones are in red.

4. Now we are ready to simulate disease spread on the network. At each time step, infected individuals may meet healthy ones that they are friends with in the network. To decide if these healthy people become infected, we assign a probability $p = 50\%$ that a person gets infected upon encounter with someone who is infected. Run the following code in the command line (double-chevron >>) in Matlab:

```
p = 0.5; %probability of spread of infection
```

Using the function `spread` provided, simulate what happens during a time step by running the following code in the command line:

```
[status,inf_nodes] = spread(status,inf_nodes,G,A,p);
```

What is the vector `status` now? How about `inf_nodes`?

Run the line above again in the command line to simulate another time step in disease transmission, and write down `status` and `inf_nodes`.

5. We will now simulate disease spread for more than one time step until all individuals become infected, and will count how many time steps this process takes. Add the following code at the end of `epidemic` and fill in the blank to accomplish this task:

```
p = 0.5;
count = 0;
for i = 1:50
[status,inf_nodes] = spread(status,inf_nodes,G,A,p);
count = count + 1;
pause(0.5);
if sum(status)== _
break;
end
end
display(['Spread takes ' num2str(count) ' time steps'])
```

   Note that we want to stop the code as soon as all individuals in the network became infected. The function `break` stops the `for` loop when the condition in the `if` statement is satisfied (in other words, when all individuals in the network are infected).

   Hint for the blank: The function `sum(status)` is calculating the sum of all elements of the vector `status`. Think about what this sum should be when all individuals in the network are infected.

   Run `epidemic` a couple of times. What do you notice? How many time steps does it take for the whole network to get infected?

6. Run the `epidemic` code again, with different values of the probability of infection p (try 10% and 90%, for example). What changes?

7. Run the `epidemic` code again, leaving $p = 50\%$ but now choosing a different initial infected individual in line `index = _;` of your program (try `index = 5`, for example). What changes, and why?

8. Discuss with people around you what could make this model of epidemic spread more realistic. Is it likely that everyone gets infected during disease spread? As an example, you can think of the disease being the flu.

9. **Challenge problem** If you finish the lab early, open `spread.m` which you used for the epidemic spread, and try to understand how this program implements the transmission of disease.

10. For the last $\sim 5 - 10$ minutes, check out the vaccination game on networks at `http://vax.herokuapp.com/game`. Try to eliminate nodes and discuss what you observe. Play with quarantine and difficulty levels! (and try not to get too addicted!)
    Also check out `http://vax.herokuapp.com/tour` for an overview of some strategies that may be effective during epidemics.