FRTN30 Network Dynamics
Hand-in 4

Due: 2021-05-28

Instructions:

- You may implement your solutions in any language you see fit, but the TAs can only guarantee you support with MATLAB/Octave. Your code should be written in a general manner, i.e., if a question is slightly modified, it should only require slight modifications in your code as well. Upload your PDF report together with your code in a separate, runnable file into Canvas. Make sure to name the PDF file with your full name.
- The PDF file should read like a standard lab-report, including a description of what you are doing, a summary of the theory used, proper presentation of results (including readable figures with axis labels) and a short interpretation of the results. Also include your name and stid-id on the first page.
- Comment your code well. Clarity is more important than efficiency.
- Late submission is discouraged: you get 1 points in your exam (out of 25) for each on-time submission.
- Collaboration policy: collaboration such as exchange of ideas among students is encouraged, however, every student has to submit her/his own manuscript (in PDF format) and code, and specify whom she/he has collaborated with and on what particular part of the work.
- Up to five of the best hand-ins may be rewarded with an extra point.

The Influenza H1N1 2009 Pandemic in Sweden

During the fall of 2009 there was a large pandemic of the H1N1-virus, commonly known as the swine-flu. During this pandemic it is estimated that about 1.5 million people in Sweden were infected. As an attempt to stop the pandemic and reduce excess mortality the government issued a vaccination program beginning in week 40 of 2009. During the weeks that followed they vaccinated more than 60% of the Swedish population.

In this hand-in, you will simulate the pandemic with the goal of learning the network-structure characteristics and disease-dynamics parameters of the pandemic in Sweden 2009. This task will be divided into 4 parts where the focus of each part is to:

1. get started and learn how to:
   a. simulate a pandemic on a known graph;
   b. generate a random graph;
2. simulate the disease propagation on a random graph without vaccination;
3. simulate disease propagation on a random graph with vaccination;
4. estimate the network-structure characteristics and disease-dynamics parameters for the pandemic in Sweden during the fall of 2009.

All numbers regarding the H1N1 pandemic in Sweden during the fall of 2009 have been taken from the report[1] (available online) by the Swedish Civil Contingencies Agency (*Myndigheten för samhällsskydd och beredskap*, MSB) and the Swedish Institute for Communicable Disease Control (*Smittskyddsinstitutet*, SMI).

1 Preliminary parts

As a warm-up exercise we will start off by doing two preliminary parts. The first one will involve simulating an epidemic on a given graph, while the second part will be to generate a random graph with preferential attachment.

1.1 Epidemic on a known graph

In this part you will simulate an epidemic on a symmetric k-regular undirected graph with node set $\mathcal{V} = \{1, \ldots, n\}$ where every node is directly connected to the $k = 4$ nodes whose index is closest to their own modulo n. See Figure 1 for an example with 8 nodes. The graph that you will simulate the epidemic on will however contain $n = 500$ nodes.

[1] `https://www.msb.se/Upload/Nyheter_press/Bilaga_Influensa%20A(H1N1)%202009.pdf`

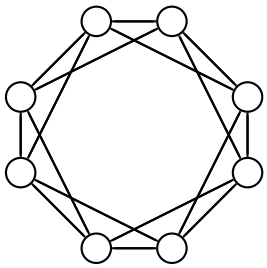

Figure 1: Symmetric k-regular graph.

The disease propagation model that you will use to simulate the epidemic is a discrete-time version of the SIR epidemic model. At any time $t = 0, 1, \ldots$ nodes are in state $X_i(t) \in \{\mathrm{S}, \mathrm{I}, \mathrm{R}\}$, where S is susceptible, I is infected and R is recovered. Let $\beta \in [0, 1]$ be the probability that the infection is spread from an infected individual to a susceptible one (given that they are connected by a link) during one time step. Assuming that a susceptible node i has m infected neighbors, this means that the probability that individual i does not get infected by any of the neighbors during one time step is $(1-\beta)^m$. Thus, the probability that individual i becomes infected by any of its neighbors is $1-(1-\beta)^m$. Furthermore, let $\rho \in [0, 1]$ be the probability that an infected individual will recover during one time step. The epidemic is driven by

$$\mathbb{P}\left(X_i(t+1) = \mathrm{I} \;\middle|\; X_i(t) = \mathrm{S}, \sum_{j \in \mathcal{V}} W_{ij} \delta^{\mathrm{I}}_{X_j(t)} = m \right) = 1 - (1-\beta)^m$$
$$\mathbb{P}\left(X_i(t+1) = \mathrm{R} \mid X_i(t) = \mathrm{I} \right) = \rho$$

where $\sum_{j \in \mathcal{V}} W_{ij} \delta^{I}_{X_j(t)}$ is the number of infected neighbors for node i.

Problem 1.1: You should simulate an epidemic on a symmetric k-regular graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = 500$ nodes and $k = 4$. See Figure 1 for an example with $n = 8$ nodes. Let $\beta = 0.3$ and $\rho = 0.7$. With one week being one unit of time, simulate the epidemic for 15 weeks. You can choose an initial configuration with 10 infected nodes selected at random from the node set $\mathcal{V}$, or make a different choice of initial configuration (in the latter case, please briefly discuss your motivation).

Do this $N = 100$ times and plot the following:

- The average number of newly infected individuals each week. In other words, you should plot how many people *become* infected each week (on the average).
- The average total number of susceptible, infected, and recovered individuals at each week. In other words, you should plot how many individuals *in total that are* susceptible/infected/recovered at each week (on the average).

Hint 1: It is important that your simulation code here is correct, as you will reuse it in the later parts. To help you check its validity, your average total number of susceptible, infected, newly infected and recovered individuals at week 15 should be approximately

$$\#\overline{S}(15) \approx 440, \quad \#\overline{I}(15) \approx 0.3, \quad \#\overline{I}_{new}(15) \approx 0.2, \quad \#\overline{R}(15) \approx 59.5$$

Hint 2: Since we use a fairly large amount of nodes for this simulation it is a good idea to use *sparse matrices* for this and the following problems. In `MATLAB` you could generate the sparse adjacency matrix W for this problem as:

```
n = 500;
W = zeros(n);
W = W + diag(ones(n-1,1),1); % add ones on the +1 off-diagonal
W = W + diag(ones(n-1,1),-1); % add ones on the -1 off-diagonal
W = W + diag(ones(n-2,1),2); % add ones on the +2 off-diagonal
W = W + diag(ones(n-2,1),-2); % add ones on the -2 off-diagonal
W = W + diag(ones(1,1),n-1); % add ones on the +n-1 off-diagonal
W = W + diag(ones(1,1),1-n); % add ones on the -n+1 off-diagonal
W = W + diag(ones(2,1),n-2); % add ones on the +n-2 off-diagonal
W = W + diag(ones(2,1),2-n); % add ones on the -n+2 off-diagonal
W = sparse(W); % transform it into a sparse matrix
G = graph(W); % convert W into a graph (might not work on all MATLAB versions)
plot(G) % probably best to do on a smaller graph...
```

Furthermore, it might also be useful to store the states of the different nodes in a good way. One possible way to do this would be to create a state-matrix X of size $(n, 1)$ and then let the value 0 correspond to susceptible, 1 correspond to infected, and 2 correspond to recovered:

```
X = zeros(n,1);
S = 0;
I = 1;
R = 2;
```

If you then happen to be interested in finding all the recovered individuals you could do this with:

```
recovered = (X == R); % recovered(j) = 1 if X(j) = R, and zero otherwise
```

Another `MATLAB` function that might be useful is `find`.

1.2 Generate a random graph

In this part you will generate a random graph according to the *preferential attachment model*. The goal is to have a randomly generated graph with average degree close to k. The idea is

the following: at time $t = 1$ we start with an initial graph $\mathcal{G}_1$, that is *complete* with $k + 1$ nodes. Then at every time $t \geq 2$, create a new graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ by adding a new node to $\mathcal{G}_{t-1}$ and connect it to some of the existing nodes $\mathcal{V}_{t-1}$ of $\mathcal{G}_{t-1}$ chosen according to some stochastic rule.

The rule by which the new node add links to the nodes of $\mathcal{G}_{t-1}$ is *preferential attachment*. This means that at every time-step $t \geq 2$, every new node added at time t will have a degree $w_t(t) = c = k/2$. Hence, it should add c undirected links to the existing graph $\mathcal{G}_{t-1}$. It decides which of the nodes in $\mathcal{V}_{t-1}$ it should connect to based on some probability that is proportional to the current degree of the node it is connecting to. In other words, if we denote the new node n_t, the probability that there will be a link between node n_t and node $i \in \mathcal{V}_{t-1}$ is:

$$\mathbb{P}(W_{n_t,i}(t) = W_{i,n_t}(t) = 1 \mid \mathcal{G}_{t-1} = (\mathcal{V}_{t-1}, \mathcal{E}_{t-1})) = \frac{w_i(t-1)}{\sum_{j \in \mathcal{V}_{t-1}} w_j(t-1)}, \qquad i \in \mathcal{V}_{t-1},$$

where $W(t)$ is the adjacency matrix for the next time-step t and $w_i(t-1)$ is the degree of node i prior to adding the new node. Some care should be taken here so that you do not add multiple links to the same node.

You should also note that if k is odd, it is a bit trickier to generate the random graph such that the average degree will become k (see the hint below).

Problem 1.2: Your goal is to, by using preferential attachment, generate a random graph of a large size (at least 900 nodes) with average degree $k \in \mathbb{Z}^+$. Let the initial graph $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$ be a complete graph with $|\mathcal{V}_1| = k_0 = k + 1$ nodes.

Note that the goal here is to implement a fairly general algorithm where it is easy to change the average degree. It should be possible by simply changing the value of k in your algorithm.

Optional: Can you do this with k being any real number larger than 2?

Hint: If you have a graph $\mathcal{G}_{t-1} = (\mathcal{V}_{t-1}, \mathcal{E}_{t-1})$ and wish to add c links using preferential attachment (without adding multiple links to the nodes) you could try the following algorithm in `MATLAB`:

```
k0; % we assume the current size of the graph is k0 > c
W; % W is the adjacency matrix of the graph prior to adding the new
   % node
c; % the number of links we wish to add from the new node (assumed
   % to be an integer here)
w = sum(W,2); % the degree-vector of the graph
P = w./sum(w); % probability vector for adding the links
for j = 1:c
  % randsample() will draw one random sample from the population
```

```
    % 1:(k0+1). The probability that it draws a certain individual is
    % proportional to the weights in the vector P.
    % Note that randsample() use replacement, so we have
    % to remove the drawn individual from the population.
    neighbor = randsample(1:k0+1, 1, true, full(P));
    P(neighbor) = 0;
    W(k0+1,neighbor) = 1;
    W(neighbor,k0+1) = 1;
  end
```

If k happens to be an odd number, then $c = k/2$ will not be an integer. However, it is still possible to achieve an average degree of k when you add a large number of nodes. This can be done by alternating between adding $\lfloor k/2 \rfloor$ and $\lceil k/2 \rceil$ links when adding a new node to the graph.

2 Simulate a pandemic without vaccination

In this part you will be using the graph generated in Section 1.2 and then simulate an epidemic on it. The disease propagation model is again the discrete-time version of the SIR epidemic model used in Section 1.1.

Problem 2: Using the methods developed in Section 1, generate a preferential attachment random graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $|\mathcal{V}| = 500$ nodes. The average degree should be $k = 6$. Let $\beta = 0.3$ and $\rho = 0.7$. With one week being one unit of time, simulate the epidemic for 15 weeks. You can choose an initial configuration with 10 infected nodes selected at random from the node set $\mathcal{V}$, or make a different choice of initial configuration (in the latter case, please briefly discuss your motivation).

Do this $N = 100$ times and plot the following:

- The average number of newly infected individuals each week. In other words, you should plot how many individuals *become* infected each week.
- The average total number of susceptible, infected, and recovered individuals at each week. In other words, you should plot how many individuals *in total that are* susceptible/infected/recovered at each week.

Hint: Remember to continue using sparse matrices for this and following problems.

3 Simulate a pandemic with vaccination

In this part you will essentially do the same thing as before, but you will also try to take some action to slow down the epidemic. This is normally done using vaccination. Therefore, during each week, some parts of the population will receive vaccination. *Once a person is vaccinated it cannot be infected.* Furthermore, *the vaccination is assumed to take effect immediately once given*, i.e. if person a is vaccinated in week 10, then a is no longer susceptible during that week, and can therefore not infect any other individual.

You should once again simulate the disease propagation for 15 weeks, but you should now also distribute vaccination to the population. This should be done such that the total fraction of population that has received vaccination by each week is:

$$\text{Vacc}(t) = [0,\ 5,\ 15,\ 25,\ 35,\ 45,\ 55,\ 60,\ 60,\ 60,\ 60,\ 60,\ 60,\ 60,\ 60].$$

$\text{Vacc}(t)$ should be interpreted as: 55% of the population has received vaccination *by* week 7, and 5% received vaccination *during* week 7.

To simulate the actual vaccination you should, at the beginning of each week, find the correct number of individuals to vaccinate according to $\text{Vacc}(t)$. You should then find individuals to vaccinate. These individuals should be selected uniformly at random from the entire population that *has not yet received vaccination.* This means that *an infected or recovered individual might receive vaccination* as well. The reason behind this is that some people were not able to tell whether they had the H1N1-virus or just the common cold. If an infected individual becomes vaccinated it is assumed that she will not be able to infect another individual. In other words, we assume that regardless of the state of an individual prior to the vaccination, *once vaccinated the individual will not be able to become infected nor infect any other individuals.*

Problem 3: Using the method developed in the previous section, generate a random graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $|\mathcal{V}| = 500$ nodes. The average degree should be $k = 6$. Let $\beta = 0.3$ and $\rho = 0.7$. With one week being one unit of time, simulate the epidemic *with vaccination* for 15 weeks, using the vaccination scheme $\text{Vacc}(t)$ above. You can choose an initial configuration with 10 infected nodes selected at random from the node set $\mathcal{V}$, or make a different choice of initial configuration (in the latter case, please briefly discuss your motivation).

Do this $N = 100$ times and plot the following:

- The average number of *newly infected* and *newly vaccinated* individuals each week.
- The average total number of susceptible, infected, recovered and vaccinated individuals at each week.

4 The H1N1 pandemic in Sweden 2009

In this part you will use all the previous parts in order to estimate the social structure of the Swedish population and the disease-spread parameters during the H1N1 pandemic.

As mentioned before, during the fall of 2009 about 1.5 million people out of a total population of 9 million were infected with H1N1, and about 60% of the population received vaccination. Figure 2 illustrates the number of newly infected (and vaccinated) individuals each week.

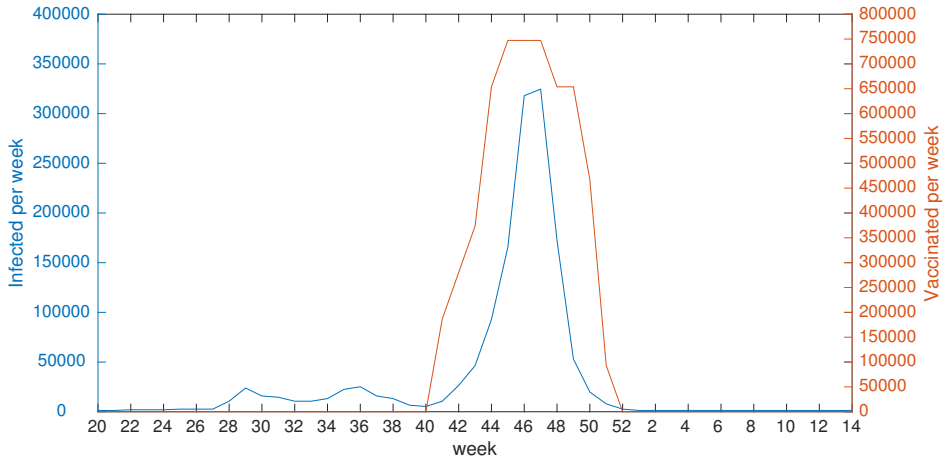

Figure 2: H1N1 Pandemic in Sweden during the fall of 2009.

We will simulate the pandemic between week 42, 2009 and week 5, 2010. During these weeks, the fraction of population that had received vaccination was:

$$\text{Vacc}(t) = [5,\ 9,\ 16,\ 24,\ 32,\ 40,\ 47,\ 54,\ 59,\ 60,\ 60,\ 60,\ 60,\ 60,\ 60,\ 60].$$

In order to not spend too much time running simulations, we will scale down the population of Sweden by a factor of 10^4. This means that the population during the simulation will be $n = |\mathcal{V}| = 934$. For the scaled version, the number of newly infected individuals each week in the period between week 42, 2009 and week 5, 2010 was:

$$I_0(t) = [1,\ 1,\ 3,\ 5,\ 9,\ 17,\ 32,\ 32,\ 17,\ 5,\ 2,\ 1,\ 0,\ 0,\ 0,\ 0]$$

The following algorithm will do a gradient-based search over the parameter space of k, β, and ρ in order to find the set of parameters that best matches the real pandemic.

Algorithm: Start with an initial guess of the parameters, k_0, β_0, and ρ_0 (here one could use $k_0 = 10$, $\beta_0 = 0.3$, $\rho_0 = 0.6$ as an initial guess) along with some Δk, $\Delta\beta$, and $\Delta\rho$ (here one might use $\Delta k = 1$, $\Delta\beta = 0.1$, and $\Delta\rho = 0.1$).

1. For each set of parameters $(k,\, \beta,\, \rho)$ in the parameter-space $k \in \{k_0 - \Delta k,\, k_0,\, k_0 + \Delta k\}$, $\beta \in \{\beta_0 - \Delta\beta,\, \beta_0,\, \beta_0 + \Delta\beta\}$, and $\rho \in \{\rho_0 - \Delta\rho,\, \rho_0,\, \rho_0 + \Delta\rho\}$:

   a) Generate a random graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ using the preferential attachment model developed in Section 1.2. The average degree should be k, and there should be $|\mathcal{V}| = 934$ nodes in the graph.

   b) Starting from week 42, simulate the pandemic for 15 weeks on $\mathcal{G}$. You should use the method developed in Section 3 with the vaccination scheme described above. Do this $N = 10$ times, and compute the *average* number of newly infected individuals each week, $I(t)$.

   c) Compute the root-mean-square error (RMSE) between the simulation and the real pandemic:

$$\text{RMSE} = \sqrt{\frac{1}{15}\sum_{t=1}^{15}(I(t) - I_0(t))^2},$$

   where $I(t)$ is the average number of newly infected individuals each week in the simulation and $I_0(t)$ is the true value of newly infected individuals each week.

2. Update k_0, β_0, and ρ_0 to the set of parameters yielding the lowest RMSE. If the result was the same set of parameters, the algorithm should stop.

Problem 4: Using the algorithm above, estimate the average degree k and the disease-spread parameters β and ρ for the pandemic.

Once you have found the best estimate, report what parameters you got. You should also show the following plots:

- The average number of *newly infected* individuals each week according to the model (with your best parameters) compared to the true value of newly infected individuals each week.
- The total number of susceptible, infected, recovered and vaccinated individuals at each week according to the model.

Hint: The algorithm will be somewhat slow to execute as it needs to run the method developed in Section 3 multiple times before updating k_0, β_0, ρ_0. As a reference, the 27 evaluations before updating k_0, β_0, ρ_0 should take about 5 - 20 seconds to complete, depending on implementation and hardware. Slower is fine, but it might be frustrating to work with.

If you feel that your implementation takes too long to run, you should start by trying to speed up the execution time of the code for the method developed in Section 3. Check that

your matrices are sparse, and that you are not using `for`-loops unnecessarily where you could have used e.g. `find` or similar. To measure execution time of code in `MATLAB` you can use e.g.

```
tic
    % %
    % The code you would want to measure
    % %
toc
```

For more information on performance improvements in `MATLAB`, this site [2] can be interesting to check.

It might also be a good idea to plot the infection curve (the number of newly infected individuals each week, $I(t)$) every time you find a new set of "best parameters" giving you a new "best error". It might also be a good idea to plot this infection curve against the true value $I_0(t)$, as well as the infection curves for the previous "best parameters". In this way you will get a sense of the progress of the algorithm.

Further, it might be a good idea to play around with the values of Δk, $\Delta\beta$, and $\Delta\rho$. For instance, perhaps start with large values and then decrease them when you cannot find a better set of parameters, perhaps reduce them by half.

5 Challenge (optional)

Try to find a better random graph (i.e. one that does not use preferential attachment) to represent the network for the pandemic. Try to also find a better algorithm to estimate the parameters.

[2] `https://se.mathworks.com/help/matlab/matlab_prog/techniques-for-improving-performance.html`