Homework III

- The deadline for on time submission is January 19, 2025. The maximum grade in case of late submission is 5 points, and the deadline for late submission is 5 days before the oral exam. The solutions have to be uploaded on the course website under the name Homework3.

- You may implement your solutions in any language you see fit, but the teacher assistant can only guarantee you support with MATLAB and Python. Your code should be written in a quite general manner, i.e., if a question is slightly modified, it should only require slight modifications in your code as well. Upload a PDF lab-report together with your code.

- The PDF should read like a standard lab-report, including a description of what you are doing, and proper presentation of results (including readable figures with axis labels, if any). Writing the report in Latex is strongly encouraged. Clarity of the presentation (especially if the report is hand-written) and ability to synthesize are part of the evaluation of the homework.

- Comment your code well. Clarity is more important than efficiency.

- Collaboration such as exchange of ideas among students is encouraged. You can work in group of up to 5 students and write a unique PDF report. However, every student has to submit her/his copy of the final PDF report and code, and specify whom she/he has collaborated with and on what particular part of the work.

---

1   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 homework, 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 a report by the Swedish Civil Contingencies Agency (*Myndigheten för samhällsskydd och beredskap*, MSB) and the Swedish Institute for Communicable Disease Control (*Smittskyddsinstitutet*, SMI).

1.1   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.

1.1.1   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 $k = 6$ 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.
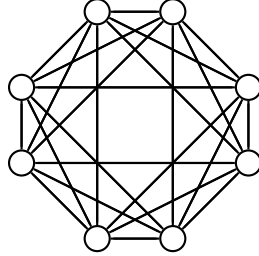

Figure 1: Symmetric k-regular graph.

The disease propagation model that you will use to simulate the epidemic is a discrete-time simplified 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 the following transition probabilities

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

$$\mathbb{P}(X_i(t+1) = \mathrm{R} \mid X_i(t) = \mathrm{I}) = \rho$$

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 $k = 6$. See Figure 1 for an example with $n = 8$ nodes and $k = 6$. Let $\beta = 0.25$ and $\rho = 0.6$. 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 are* susceptible/infected/recovered at each week (on the average).

Hint: 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.

1.1.2   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 be k. 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.

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 very easy to change the average degree. It should be possible to change the average degree by only changing the value of k in your algorithm. This algorithm will then be used in Section 1.4.

1.2   Simulate a pandemic without vaccination

In this part you will be using the graph generated in Section 1.1.2 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 1.1.1.

Problem 2: Using the methods developed in Section 1.1, 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 $\beta = 0.25$ and $\rho = 0.6$. 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.

1.3   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 according to:

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

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 Vacc(t). You should then find individuals to vaccinate. These individuals should be selected uniformly at random from the population that *has not yet received vaccination*. This means that *an infected 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 $\beta = 0.25$ and $\rho = 0.6$. With one week being one unit of time, simulate the epidemic *with vaccination* for 15 weeks, using the vaccination scheme 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.

1.4   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

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:

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

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:

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

where the first entry is the initial number of infected. 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, β, ρ) 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 1.1.2. 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 1.3 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 1.3 multiple times before updating k_0, β_0, ρ_0.

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

1.5   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   Coloring

In this part, we will study graph coloring as an application of distributed learning in potential games. The aim of graph coloring is to assign a color to each node in a given undirected graph, such that none of the neighbors of a node have the same color as that node. We will begin with a simple line graph to illustrate the distributed learning algorithm, and then look at a more general example, which can be seen as a distributed solution approach to assign non-interfering channels to wifi access points.

a) In this example, study a line graph with 10 nodes. Denote the i-th node state by $X_i(t)$ and the set of possible states by $\mathcal{C} = \{\text{red}, \text{green}\}$. At initialization, each node is red, i.e, $X_i(t) = \text{red}$ for all $i = 1, \dots, 10$. Every discrete time instance t, one node $I(t)$, chosen uniformly at random, wakes up and updates its color.

The new color (resulting from a node's update), is chosen from a probability distribution given by

$$P(X_i(t+1) = a \mid X(t), I(t) = i) = \frac{e^{-\eta(t) \sum_j W_{ij} c(a, X_j(t))}}{\sum_{s \in \mathcal{C}} e^{-\eta(t) \sum_j W_{ij} c(s, X_j(t))}},$$

where the cost is given by

$$c(s, X_j(t)) = \begin{cases} 1 & \text{if } X_j(t) = s \\ 0 & \text{otherwise.} \end{cases}$$

In the above expression, $\eta(t)$ is the inverse of the noise. To decide upon a good choice of $\eta(t)$, some heuristics are required, but it is preferable to have it increasing in time so that the noise is decreasing. For this exercise you can start with

$$\eta(t) = \frac{t}{50}.$$

To study how close to a solution the learning algorithm is, we consider the potential function, which is given by

$$U(t) = \frac{1}{2} \sum_{i,j \in \mathcal{V}} W_{ij} \, c(X_i(t), X_j(t)),$$

where $\mathcal{V}$ is the set of nodes. If the potential is zero, there are no conflicting nodes and a solution is found.

Your task is to simulate the learning dynamics described above. Include plots of the potential function in your report and briefly comment on it.

b) Next, we use the coloring algorithm for the problem of assigning wifi-channels to routers. The adjacency matrix of a network of 100 routers is given in `wifi.mat` and the routers' coordinates are given in `coord.mat`. Here, a link between two nodes means that the two routers are able to interfere with each other. The set of possible states is $\mathcal{C} = \{1 : \text{red}, 2 : \text{green}, 3 : \text{blue}, 4 : \text{yellow}, 5 : \text{magenta}, 6 : \text{cyan}, 7 : \text{white}, 8 : \text{black}\}$, where colors represent frequency bands, and the cost function is

$$c(s, X_j(t)) = \begin{cases} 2 & \text{if } X_j(t) = s, \\ 1 & \text{if } |X_j(t) - s| = 1, \\ 0 & \text{otherwise.} \end{cases}$$

The cost function $c(s, X_j(t))$ symbolizes that routers that are close by should not use channels with the same frequency band or a frequency band right next to each other.

Use $\eta(t) = t/50$ and verify that a near-zero potential solution is found after a sufficient number of iterations (you can use an arbitrary initial condition). Include plots of the potential functions for some different tested cases and an illustration of the node coloring corresponding to the smallest potential function (that is, the best obtained solution) in the report. Comment briefly on the obtained results.

c) *Optional:* Evaluate what happens for different choices of $\eta(t)$, i.e., constant (with small and large values), or other increasing functions $\eta(t)$ etc.

Comment on what you observe.