Networks Dynamics and Learning: Homework 3

Valerio Zingarelli
Politecnico di Torino
Student id: s281586
s281586@studenti.polito.it

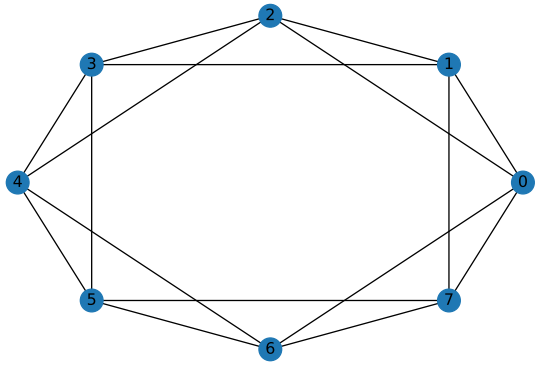

Fig. 1. Graph of exercise 1.1. In this case, to test the graph creation, a graph with n=8 and k=4 has been created

***Abstract*—In this report I will explain and explore the methods I adopted to solve proposed exercises and the results I got. The essay will be divided into sections, each one corresponding to one of the assigned problems. During the resolution of the exercises, Vito Palmisano and I have compared our results and provided advice to each other, especially in last point.**

I. Problem 1

The first task of the assigned homework is divided into 2 different parts. More specifically, we needed to "warm-up" in order to use the models we develop during this part to solve the successive ones.

A. Problem 1.1

In this part we were required to simulate an epidemic on a symmetric k-regular undirected graph with node set $\mathcal{V} = 1, ..., n$ where every node is directly connected to the $k = 4$ nodes whose index is closest to their own modulo n. In Fig,1, an example of the generation of the symmetric k-regular undirected graph has been provided. More specifically, in this case, we have $n_{nodes} = 8$ and $k = 4$. Concerning the epidemic, the model that should be adopted is SIR model. In this kind of epidemic simulation, the agents(indeed, the nodes of the graph), could have 3 different status: $X_i(t) \in \{S, I, R\}$ which represent a person susceptible to disease, infected and recovered respectively. Considering an agent i, given $\beta \in [0; 1]$ the probability of infection between an infected person and a susceptible one, we get that the probability of a susceptible node of being infected is:

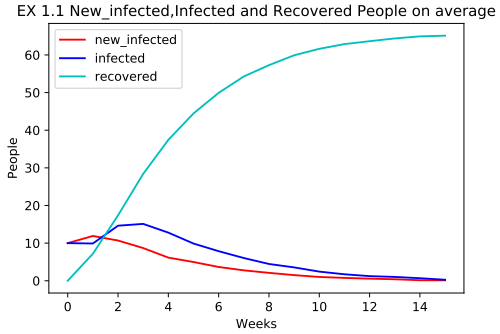

Fig. 2. An example of the dynamics for 1.1, an epidemic without vaccination on a 4-regular graph.

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

Where m is the number of infected neighbours for node i.

In addition to that, considering $\rho \in [0, 1]$ to be the probability of recovery, we have:

$$P(X_i(t+1) = R | X_i(t) = I) = \rho \tag{2}$$

Given these premises, the exercise required to simulate a SIR-model epidemic with $\beta = 0.3$ and $\rho = 0.7$ starting from 10 initially infected nodes out of 500. The graph chosen to be the basis of simulation is a k-regular undirected graph with $k = 4$. $N_{iter} = 100$ simulations have been performed and several average statistics have been computed. In Fig.2, some plots of new infected, infected, recovered and susceptible agents over time have been provided.

For practical reasons, my choice of implementation, for the entire homework, is to use a function that properly could be adapted to all the different tasks we were required to

accomplish. Let's briefly have a look at such a function: its name is *SIRFunction* and its purpouse, of course, is to simulate a SIR epidemic model. Its parameters are:

- β: the probability of infection between an infected person and a susceptible one
- ρ: the probability of recovery once an agent has been infected
- *n_steps*: the number of weeks to simulate
- *n_iter*: the number of iterations
- *initial_infected*: the number of agents that are infected in the first week
- *G*: the graph to use for simulation
- *n_agents*: number of nodes in the graph
- *vaccination*: a flag that indicates if we want to simulate an epidemic with or without vaccination
- *vaccine*: an array containing the percentages of people that should be vaccinated to the i-th week
- *to_print*: a flag that says if we want to print result or just store them into variables.

If something is not completely clear, I refer to the code for more details.

B. Problem. 1.2

In the second point of exercise 1, we were required to create and implement an algorithm to model a random graph with Preferential Attachment approach. Starting from a complete graph with $k+1$ nodes it's needed to add one node at a time and to connect it to the already existing nodes according to some stochastic rules. More specifically, at time t, the degree of the new added node will be:

$$w(t) = k/2 = c \tag{3}$$

After that, we should decide where to add the links proportionally to the current degree of each node:

$$P(W_{n_t i} = W_i n_t = 1) = \frac{w_i(t-1)}{\sum_{j \in \mathcal{V}_{t-1}} w_j(t-1)} \tag{4}$$

where $W(t)$ is the adjacency matrix for the next time-step t and $w_i(t1)$ is the degree of node i prior to adding the new node. Of course, if k is odd, then $c = k/2 \notin \mathbb{N}$ and the solutions to this issue could be to alternate between adding $int(k/2)$ and $int(k/2)+1$ new links when adding a new node in order to mantain the same average degree. In other words, this means to add $\lfloor \frac{k}{2} \rfloor$ and $\lceil \frac{k}{2} \rceil$ new links respectively.

In the code, two examples with k even and odd have been provided and it's possible to notice that in both cases the obtained graph presents the desired average degree. Same algorithm used in this point has been used again in the following requests. Let's now have a brief look at the parameters used for the function generating the random graph with Preferential Attachment approach:

- *k*: the average degree of the nodes of the obtained random graph
- *G1*: starting graph
- *n_nodes*: number of nodes if the final graph

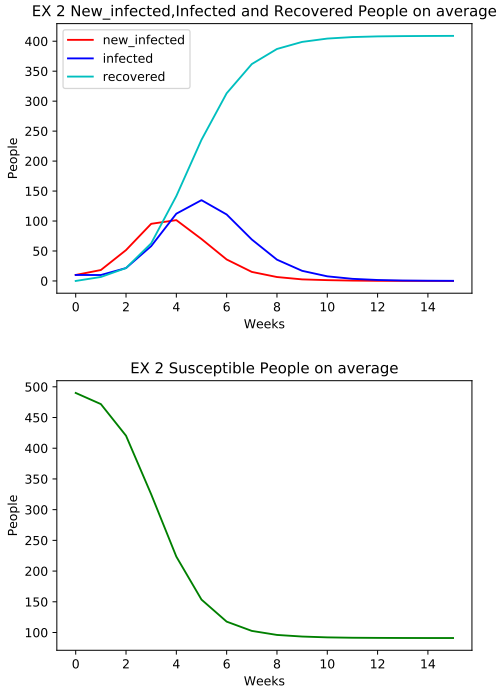

Fig. 3. An example of the dynamics for Exercise 2, an epidemic without vaccination. In this case, the graph used for the model is a random graph with $k = 6$.

- *to_print*: a flag that says if we want to print the average of the resulting graph or just store it into a variable.

In addition to that, other possible implementations of random graphs creation algorithm have been proposed; in particular, Erdos-Renyi algorithm and Small World model have been implemented and tested as I will discuss in Problem 5 section.

II. Problem 2

The second problem is basically an application of the two algorithms created during points 1.1 and 1.2. So firstly we were required to create a random graph with $k = 6$ using the algorithm developed during point 1.2 and then, to apply the SIR epidemic's model to the graph we had just created. In Fig 3 an example of the epidemic's dynamics is provided.

It's interesting to notice that in this second case, the epidemics is much faster then it was in point 1.1 and this can be due to the higher average degree of the agents. In order to understand if this change in the evolution of epidemics is due to the higher average degree or random graph itself, a simulation with $k = 4$ has been tested. This, of course, has been done in order to compare obtained results with the previous point. In Fig. 4, an example of the evolution of epidemics on a random graph with $k = 4$ has been provided. It's interesting to notice how the usage of a random graph makes the number of infected people much higher then the usage of a symmetric graph.

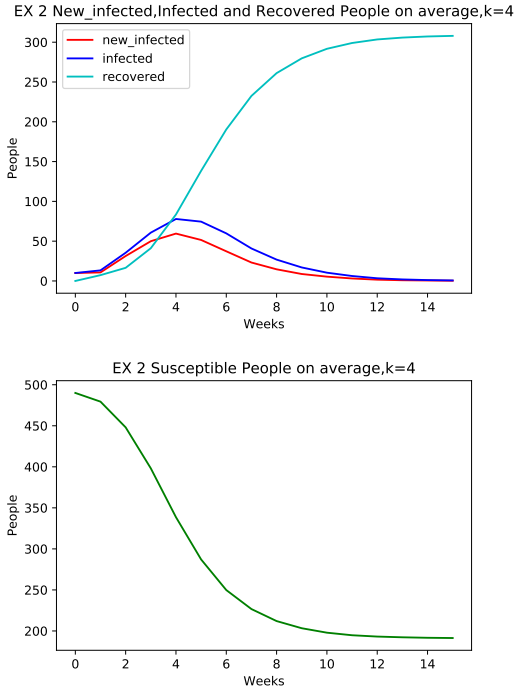

Fig. 4. An example of the dynamics for Exercise 2, an epidemic without vaccination. In this case, the graph used for the model is a random graph with $k = 4$.

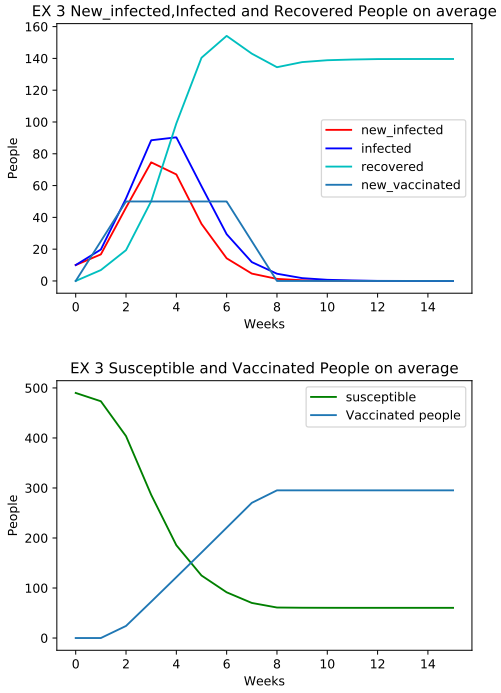

Fig. 5. An example of the dynamics for Exercise 3, an epidemic with vaccination. In this case, the graph used for the model is a random graph.

III. Problem 3

In this exercise, we have to deal with vaccinations. In fact, the request was to simulate a pandemic *with vaccination* according to the previously developed SIR discrete model. In particular, we were given :

$$V(t) = [0, 5, 15, 25, 35, 45, 55, 60, 60, 60, 60, 60, 60, 60, 60] \tag{5}$$

This array should be interpreted as: 55% of the population has received vaccination by week 7, and 5% received vaccination during week 7.

In Fig. 5 an example of the dynamics of epidemic with vaccination has been provided. In addition to that, another interesting statistics has been computed: the number of recovered people *because of vaccine*. This means, in other words

$$X_i(t-1) = I \rightarrow X_i(t) = V \tag{6}$$

In Fig. 6 an example of this dynamics is provided.

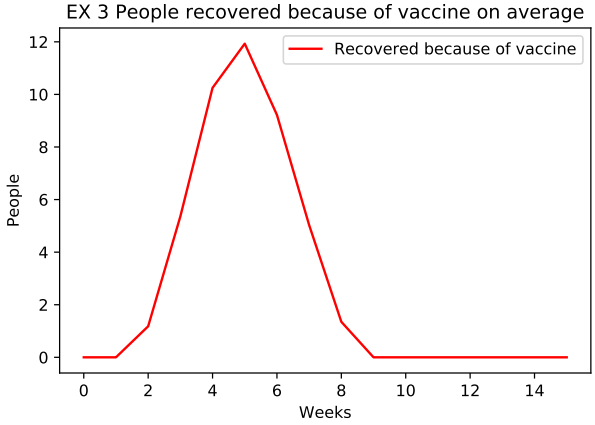

Fig. 6. Number of people recovered because of vaccine in EX3.

IV. Problem 4

In this exercise, we were required to find the best parameters k,β,ρ to model a simplified version of the H1N1 epidemics occurred in Sweden in 2009. In this case, the number of nodes of the graph we have to deal with is $|\mathcal{V}| = 934$. In order to prove the algorithm is doing right, the real evolution of newly infected people has been provided:

$$I_0 = [1, 1, 3, 5, 9, 17, 32, 32, 17, 5, 2, 1, 0, 0, 0, 0] \tag{7}$$

During the weeks we were asked to simulate, vaccine has been received by people according to the following:

$$V(t) = [5, 9, 16, 24, 32, 40, 47, 54, 59, 60, 60, 60, 60, 60, 60, 60] \tag{8}$$

Of course, this array should be read exactly like the previous one from (5). In order to find the best parameters, provided algorithm has been followed. In particular starting from a random guess k_0, β_0 and ρ_0 and for each configuration in parameter spaces $k \in [k_0\Delta k, k_0, k_0 + \Delta k]$, $\beta \in [\beta_0\Delta\beta, \beta_0, \beta_0 + \Delta\beta]$, $\rho \in [\rho_0\Delta\rho, \rho_0, \rho_0 + \Delta\rho]$:

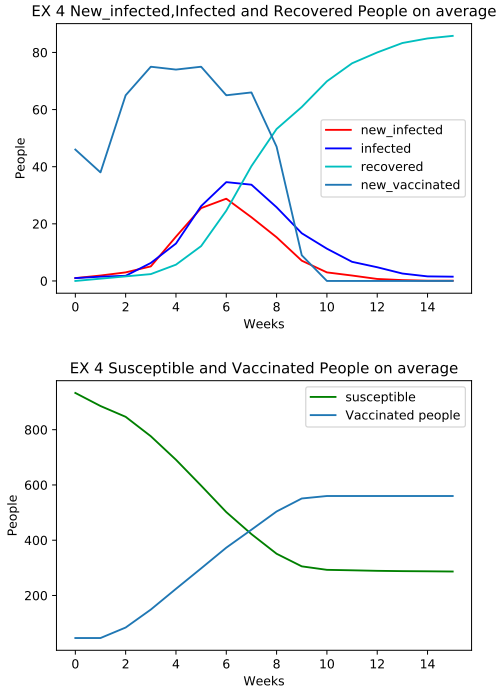

Fig. 7. An example of the dynamics for Exercise 4, an epidemic with vaccination. In this case, the graph used for the model is a random graph. The plots refer to the best configuration, indeed, the one that manages to reach the best RMSE value.

1) Create a random graph $\mathcal{G}$ with $|\mathcal{V}| = 934$ and $degree = k$
2) Simulate the pandemic on $\mathcal{G}$
3) Compute RMSE between obtained results and I_0
4) Update k_0 ρ_0 β_0 with the best found parameters.
5) if the same ρ or β get founds 2 consecutive times, then $\Delta\beta$ or $\Delta\rho$ (or both in case both ρ or β are the same of the previous "best" config) gets divided by 2. This is not done for Δk since it is 1.

Let's now analyze the function which I implemented in terms of code. It takes the following input parameters:

- k_0: the initial random guess value of k
- ρ_0: the initial random guess value of ρ
- $beta_0$: the initial random guess value of β
- Δk: the value of the steps for k's interval
- $\Delta\beta$: the value of the steps for β's interval.
- I_0: an array containing the values of real epidemics. It's the value I compare the results of the simulation to.
- *algorithm*: algorithm to random graph's generation(Preferential Attachment or Small World).

In this way it's possible to detect the best configuration and explore its neighbourhood in order to find a better one.

In Fig. 7 an example of an epidemics evolution has been provided. Of course, the one in the figure is the evolution based on the parameters that fitted data in a better way. In addition to that, an interesting comparison between I_0 and the newly infected generated by simulation has been presented in Fig.8.

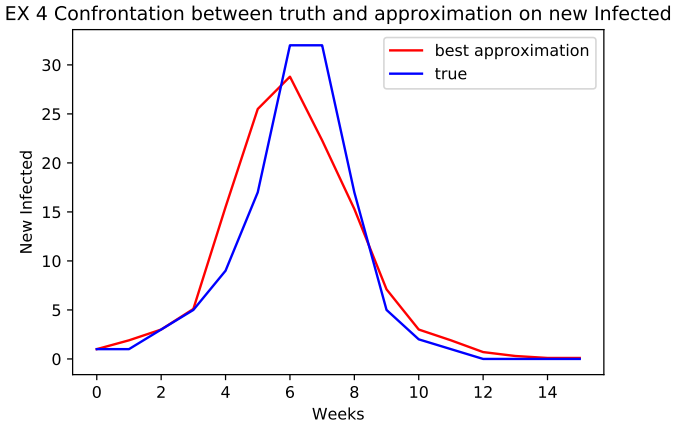

Fig. 8. A comparison between best approximation of I_0 and I_0 itself in Problem 4.

The algorithm has been launched 10 times in order to reduce the variability of the results and the average best obtained $RMSE$, with $algorithm =' PreferentialAttachment'$ is

$$\mathbb{E}[RMSE_{best}] \approx 3.942 \tag{9}$$

Because of the random nature of the exercise, there is a lot of variability in the results and the best configuration changes a little bit from one iteration to another. That's why a set of parameters that identifies the best configuration has not been reported.

V. Problem 5-Possible Improvements

The last task required to find a better way of both creating the random graph and find the best parameters to model Sweden's H1N1 epidemics' diffusion. Different improvements both in graph creation and in best parameters finding have been tested and, in the following, they will be discussed one at a time.

A. Graph creation

Concerning random graph creation algorithms, another algorithm have been tested. In fact, Preferential Attachment approach is not the best option to choice to model a situation like the one we have to deal with.

Small World

Empirical observations suggest that real world graphs have $smalldiameter$ and $highclustering$. That's why Small World models for random graphs have been introduced. The main idea is to start with a simple graph where every node is connected to k closest nodes modulo n. Then, add to the existing graph l additional undirected links, where

$$I \approx Bin(\frac{nk}{2}, p) \tag{10}$$

and the new links connect pairs of nodes chosen independently and uniformly at random. In this way it's possible to combine models for geographical proximity and some rare long distance

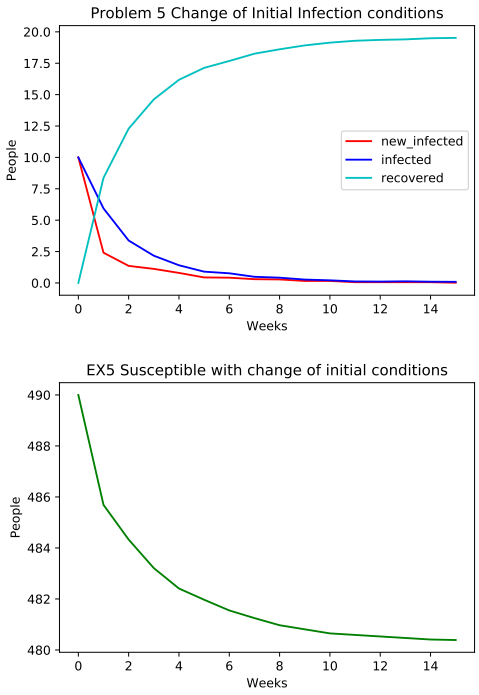

Fig. 9. An example of the dynamics in Problem 1.1 but changing the initial conditions. In this case an epidemic without vaccination has been simulated. Looking at $infected$ and $new_i nfected$ curves, it's easy to notice how the epidemics is very very slow and after few weeks it basically disappears.

links(e.g. journeys or something similar). A function called $SmallWorld$ has been implemented using the following input parameters:

- *n_nodes*: the number of the nodes the final random graph should have
- *k*: the k for generating the starting graph where every node is connected to k closest nodes modulo n.
- *to_print*: a flag saying if we want to print the average degree, the diameter of the graph and the clustering coefficient. If it is set to be False, just the obtained graph is returned
- *p*: the probability of the Binomial Random Variable I

Regarding the last parameter, p, we need to check if its value makes sense. Given I the number of the new links, as we told before, we want that the expected value of I should be at least one. To obtain that we must impose:

$$\mathbb{E}[I] = \frac{npk}{2} \geq 1 \tag{11}$$

$$p \geq \frac{2}{nk} \tag{12}$$

Of course, if p doesn't satisfy the constraint, the function gives back an Error to the user. This kind of model has been tested 10 times in the conditions of Problem 4 and the following $\mathbb{E}[RMSE_{best}]$ has been computed:

$$\mathbb{E}[RMSE_{best}] \approx 6.02 \tag{13}$$

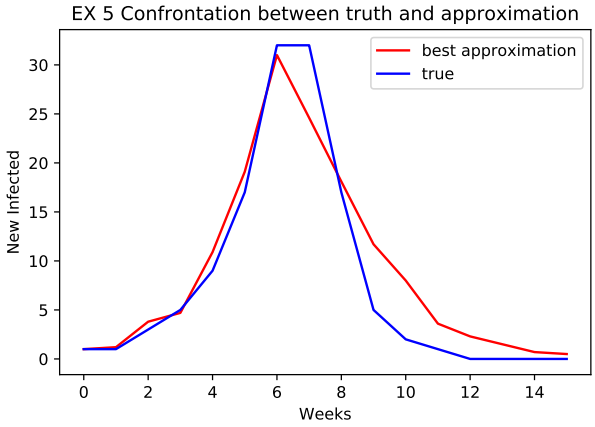

Fig. 10. A comparison between best approximation of I_0 and I_0 itself in Problem 5.

B. Algorithm improvement

The idea behind the "new" algorithm is to explore all the space of probability for rho and Beta and, concerning k, the idea is to explore the interval $[0, 2k]$. Indeed, if we are close to a local optimum, the algorithm proposed in the previous point just stops, for example: if we have a local optimum for $k = 10$ but a global optimum for $k = 13$, the solution with $k = 13$ will never be explored because the algorithm developed in problem 4 stops before. Using this approach, $\Delta\beta$ and $\Delta\rho$ are the steps used to define the range of values for β and ρ respectively. Even in this case, if the values of β and ρ are the same in 2 consecutive "best" configurations, the value of $\Delta\beta$ and/or $\Delta\rho$ gets divided by 2. In Fig.10 it's possible to notice a comparison between best approximation of I_0 and I_0 itself. It's simple to notice how this new approximation fits the data in a very better way with respect to the algorithm of the previous Problem(Fig.8). In this case, the algorithm has been executed several times and the average best obtained RMSE is:

$$\mathbb{E}[RMSE_{best}] \approx 3.2 \tag{14}$$

It's simple to notice that this kind of algorithm can obtain better performances with respect to the one proposed in Problem 4. Of course, on the other side, the check of the configurations in this case is much more demanding in terms of computational resources and computational time with respect to the previous algorithm.

C. Changing the initial infected choice

In previous problems, the choice of initial infected people was totally randomly made. This is a good choice if the epidemics is "imported" from another country but could be a little unrealistic if the epidemics starts in the country under analysis. In fact, it's quite improbable that 10(or more) people get infected in different part of a country simultaneously. Indeed, if I get infected by a virus and I am the first one(patient zero), I will pass the infection to people that I meet everyday and not to a guy which I don't know and lives in a completely

different part of the country. So a good idea to improve the creation of initially infected people could be the infection of one agent(patient zero) and his/her n neighbours. This could lead to a more realistic evolution of the epidemics. In order to prove this kind of idea, an implementation of it has been performed for the same task of Problem 1.1 and the obtained results have been provided in Fig.9. Notice that, changing initial conditions in the way that was exposed before, the epidemics is much slower with respect to Problem 1.1 and to Problems 2 and 3. Especially, if we look at $new_infected$ and $infected$ curves, they basically decrease really fast. This could be because of the usage of a sort of "cluster" of friends/relatives as a starting point for epidemics could help to "mitigate" its effect because it's more circumscribed and, as a consequence, limited.

D. Lockdown

An interesting improvement of the simulation could be the insertion of a $lockdown$. Indeed, if the number of newly infected each week is higher then a certain threshold, we can simulate the same measures adopted by governments to fight against Covid19. More specifically, we could cut randomly a percentage of the links(for example 80%) and simulate the remaining weeks on the new graph we have just created this way. After some weeks if the number of newly infected decreases, we substitute the "cut graph" with the original one and see what happens. For time reasons, the implementation of the lockdown simulation has not been performed.