Epidemic Spread on Weighted Networks

Miguel Alcón Doganoc
Universitat Politècnica de Catalunya
Barcelona, Spain
miguel.alcon@est.fib.upc.edu

Roger Pujol Torramorell
Universitat Politècnica de Catalunya
Barcelona, Spain
roger.pujol.torramorell@est.fib.upc.edu

1 INTRODUCTION

Understanding how infectious diseases spread, has public health and ecological implications. The contact structure between hosts is known to have a key influence on disease spread. However, most studies assume that all types of contacts are identical, when in reality some individuals interact more strongly than others. This can be clearly seen in, for example, sexual-contact networks, where the number of sex acts is not equal for all partners. These differences between weights among partners, generates an heterogenous network and this heterogeneity can affect by speeding up or slowing down an epidemic spread depending on how strongly connected the hosts are.

In this project we are going to generate various networks that try to imitate the real world, changing the heterogeneity of the networks and seeing how two different infection models (SIR and SIS) spread on those. We replicate the simulation process done in the paper [1], where they generate different networks and simulate the SIR epidemics model with it. Furthermore, want to extend with the simulation of the SIS model.

2 METHODOLOGY

2.1 Probability distributions

As explained before, the main goal of this project is simulating the SIR and SIS epidemic models into networks. Random numbers and, so, probability distributions are the basis of the project. We use 3 different distributions among all the paper: Poisson, power law and uniform. Except the second one, all the distributions used in this work are the ones implemented in the well-known *random* standard library of C++. If the distribution is not specified somewhere in the paper, random numbers follow a uniform distribution then.

For the power law, we created ourselves the number generator that follows the distribution. Given a real random number y between 0 and 1, the number generator returns the following:

$$[(x_1^{1-\lambda} - (x_1^{1-\lambda} - x_0^{1-\lambda})) \cdot y]^{\frac{1}{1-\lambda}}$$

Where x_0 and x_1 are the maximum and minimum possible numbers, respectively, and λ the exponent.

We tested the generator generating 10000 numbers with it and checking whether them follow a power law or not. You can see the results in figure 1.

2.2 Network Generation

Networks are obtained by first generating 10000 nodes with k_i stubs ($0 \leq i < n$). Each k_i is selected randomly using a Poisson or a power law distribution. The l_i interaction events a node has are then distributed among its k_i stubs. Again, each l_i is randomly picked using a Poisson or power law distributions, or they are just $\delta \cdot k_i$. We refer to this as the delta distribution, though it is just a multiplication of k_i. Moreover, the interactions of a node are distributed among the stubs in the following way:

(1) Generate one real number r_j between 0 and 1 for each stub ($0 \leq j < k_i$).
(2) Calculate the total sum.

$$total = \sum_{j=0}^{k_i} r_j$$

(3) Each stub is then equal to

$$\lfloor \frac{(l_i - k_i) \cdot r_j}{total} \rfloor$$

With this, all (or almost all) the interactions are distributed.
(4) Distribute randomly the remaining interactions, if any.

This process assigns a weight to each stub of a node according to its l_i.

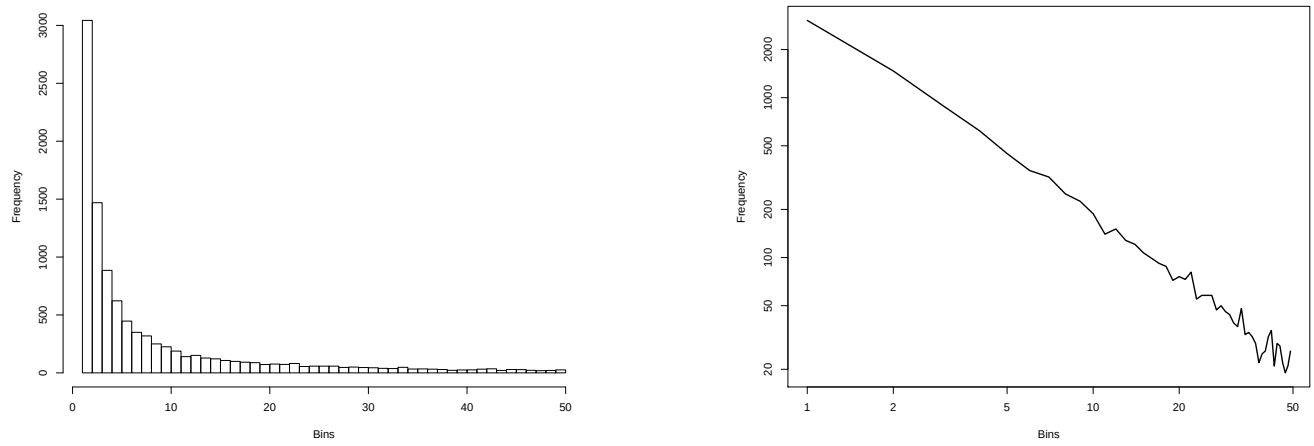
After this, the adjacency matrix of the network is generated following this process:

(1) **Match**. Stubs are randomly matched together. To do this, we put all the stubs in a vector and we shuffle it. Then, each stub in a even position within the resultant vector is matched to the stub at its right, if there is one. If the stubs in a match are from the same node, we randomly select another stub of the vector and we try to swap it with one of the others. We repeat this process until all matched stubs are from different nodes.
(2) **Reject**. A match can be rejected if the weights of the stubs differ by more than one interaction event. In addition, matches are rejected if they differ by more than 10% of the smaller weight involved to avoid biases in nodes with few links. If stubs with non-identical weighs are matched, i.e. they differ by 1, we assign randomly one of them to the weight of the connection.
(3) **Connect** For all accepted match, we sum its weight to the corresponding position of the adjacency matrix. We sum instead of assign the weight to the matrix in order to avoid checking that there not exist any match between the same nodes.

Since this way of matching the nodes can lead to a lot of rejections, this process is repeated until it does not match any stub more in two consecutive iterations. Finally, we generate the adjacency list from the matrix to reduce the memory size, and because we can iterate through it faster.

2.3 Infection Simulation

To simulate the epidemic infection spread through the generated networks, we tested two different models: SIR and SIS.



(a) Normal scale

(b) Log scale (line histogram)

Figure 1: Generated numbers with our power law number generator. We divided all the possible values into 50 bins.

2.3.1 SIR. The initials for this models stands for: Susceptible, Infected and Recovered. This simulate a disease that infects with probability μ_{SI}, where this probability is dependent from how many contacts with infected hosts it is having. Basically, every node in susceptible state, can be infected by each connection with an infected host by a probability β. Then an infected host can recover with probability μ_{IR}, where this probability is constant for all the nodes, in our case we set $\mu_{IR} = \gamma$. Finally when the host has ben recovered, it can't get infected again. So basically each host follows the state machine shown in figure 2.

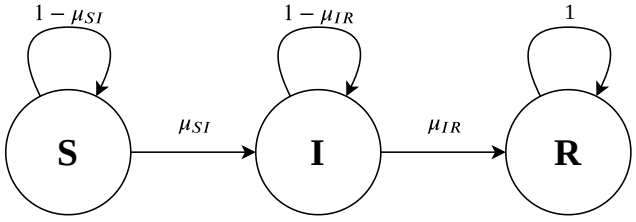


Figure 2: SIR model representation.

Assuming we have an adjacency list W, where each position is equal to the weight between the two nodes. In order to simulate this in an optimal way, at each time step, we go through all nodes of the network and do the following:

- **If node i is Susceptible:**
  $n = \sum_{j \in I} W_i, j$
  $\mu_{SI} = 1 - (1 - \beta)^n$
  if $\mu_{SI} > real\_random(0, 1)$: i is Infected
  else: i stays Susceptible
- **If node i is Infected:**
  if $\gamma > real\_random(0, 1)$: i is Recovered
  else: i stays Infected
- **If node i is Recovered:**
  i stays Recovered

Note that $j \in I$ means that j is an infected node and the function $real\_random(0, 1)$ gives a random value with an uniform distribution between 0 and 1.

2.3.2 SIS. The initials of this model stands for: Susceptible, Infected, Susceptible. This means that there is no Recovered state where the node is immune. Essentially the model to move from state S to I is exactly the same as in the SIR model, but this time when an infected node recovers, it simply become again a Susceptible node (see figure 3).

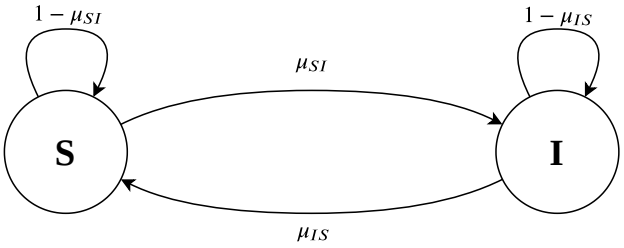


Figure 3: SIS model representation.

To simulate this we do like in SIR but, at each time step, we go through all nodes of the network and do the following:

- **If node i is Susceptible:**
  $n = \sum_{j \in I} W_i, j$
  $\mu_{SI} = 1 - (1 - \beta)^n$
  if $\mu_{SI} > real\_random(0, 1)$: i is Infected
  else: i stays Susceptible
- **If node i is Infected:**
  if $\gamma > real\_random(0, 1)$: i is Recovered
  else: i stays Infected

2.4 Experiments

In our experiments, we test 6 different networks:

- P_k Poisson - P_l Poisson
- P_k power law - P_l Poisson
- P_k Poisson - P_l power law
- P_k power law - P_l power law
- P_k Poisson - P_l delta
- P_k power law - P_l delta

The parameters are:

- P_k Poisson: $< k >= 4$
- P_l Poisson: $< l >= 8$
- P_k power law: $\lambda_k = 1.4$, $x_0 = 1$ and $x_1 = 22$
- P_l power law: $\lambda_l = 0.89$, $x_0 = k$ and $x_1 = 22$
- P_l delta: $\delta = 2$

We generated 20 networks for each of the mentioned configurations and in them we simulated 100 times each simulation model (SIR and SIS). We used the same parameters in both models, $\beta = 0.01$ and $\gamma = 0.004$.

3 RESULTS

In this section we present the results of the experiments explained in section 2.4. Figures 4 and 5 show the variaton of infected and recovered hosts (y axis) over the time (x axis). Black and green lines represent the average variation of infected and recovered number of hosts (respectively) over time for each network type. Red and blue dashed lines represent the variation of infected nodes over the time in the network with the higher and lower peak (respectively) of infected hosts. Figure 4 is the result of the experiments for the SIR epidemics and figure 5 for the SIS.

4 DISCUSSION

Knowing that Poisson distribution generates homogeneous values and Power law generates heterogeneous values, we can see how it affects to the infection spread.

In figure 4 we can see how the Poisson-Poisson network is the one that get more hosts infected (and consequentially more hosts recovered), but it starts slower than other networks. This is logical, since it is the most homogeneous network, the disease spreads at first slow because the nodes infected doesn't reach a huge amount of nodes, until many nodes are infected. If we compare it with any network with a power law anywhere, we see that the others grow faster at start, but they also start to fade with a smaller amount of nodes infected. This is because in heterogeneous networks, once the nodes with many connections recover, then the disease loses most of the infecting probability. Also in heterogeneous networks, there are more nodes with few connections that are harder to infect, which results in more nodes that never get infected. Then the networks with constant P_l are the ones with the highest total infected hosts and also some of the fastest ones to spread. Here we can see directly how the homogeneity of P_k affects to the speed, because as we previously stated, in the power law plot we can see how the spread speed is maximum but the total infected is better when it is homogeneous. Finally, comparing how the distribution of P_l affects, we can see that it happens almost the opposite to P_k, more homogeneous results in more speed, but this time it still increase the max amount of infected nodes.

In figure 5 we can see a similar behavior than before. Again, if P_k is more homogeneous, the number of infected nodes grows slower but more, and if P_l is more homogeneous, it grows faster and more. With the SIS model we can see more clearly the rate of infection against the rate of recovery by looking the height where there seems to be a horizontal asymptote. If it is higher, then the recover rate is lower or the infection rate is higher. This asymptote seems to be related to the previous max amount of infected, since it is increased by homogeneous P_k and P_l.

5 RELATED WORK

As commented in section 1, our work is based on [1]. In this paper, the authors present a novel framework to estimate key epidemiological variables from joint probability distributions of k and l. The framework also allows for a derivation of the full time course of epidemic prevalence and contact behavior, which are validated with simulations of the SIR epidemic spread on networks. We replicated these simulations, using the same parameters to validate our results on SIR. With this, we extended their experiments to the SIS epidemics.

REFERENCES

[1] Kamp, C., Moslonka-Lefebvre, M., and Alizon, S. Epidemic spread on weighted networks. *PLOS Computational Biology 9*, 12 (12 2013), 1–10.

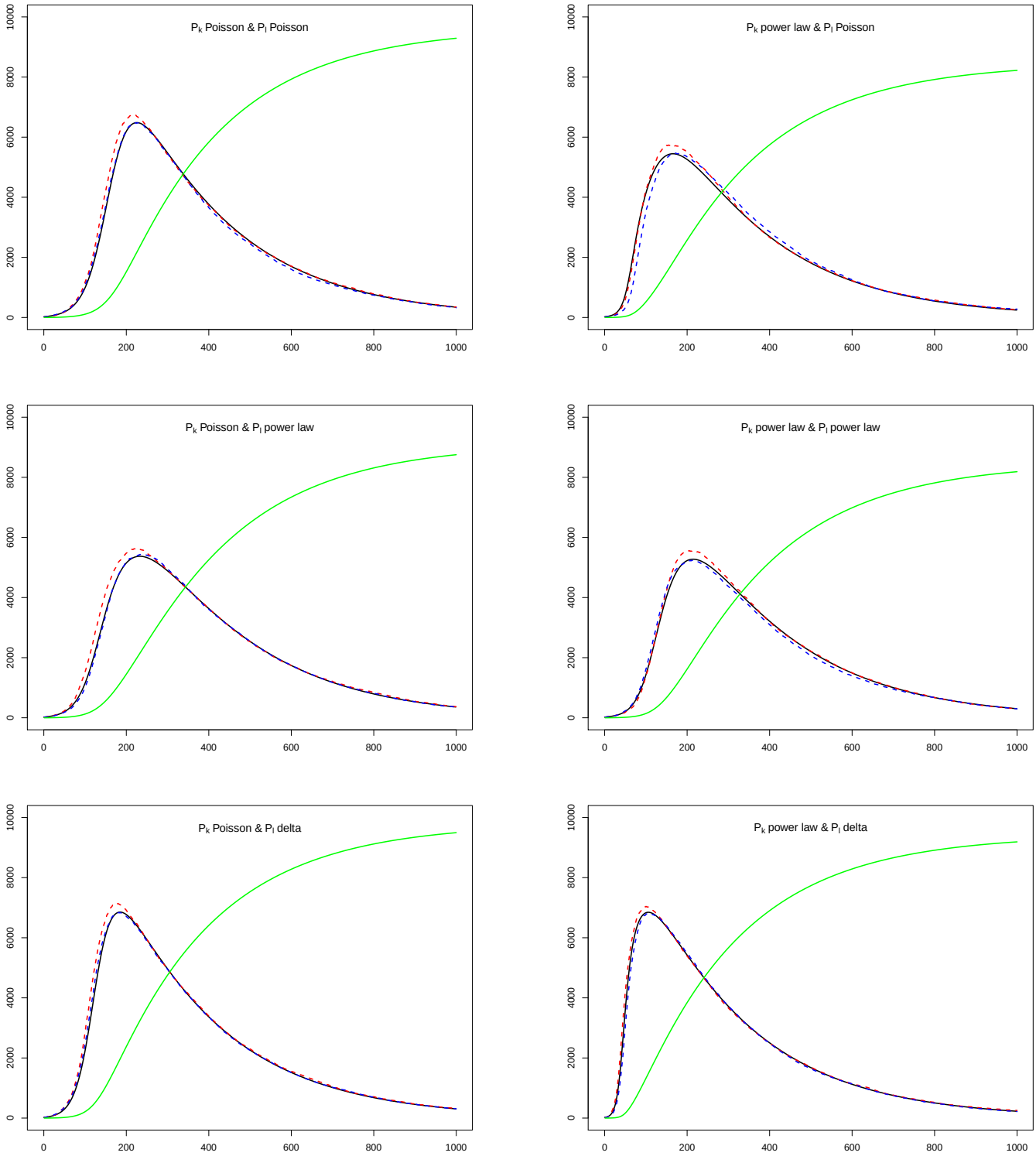


Figure 4: Dynamics of the number of infected and recovered hosts during the SIR epidemic spreading on different types of networks.

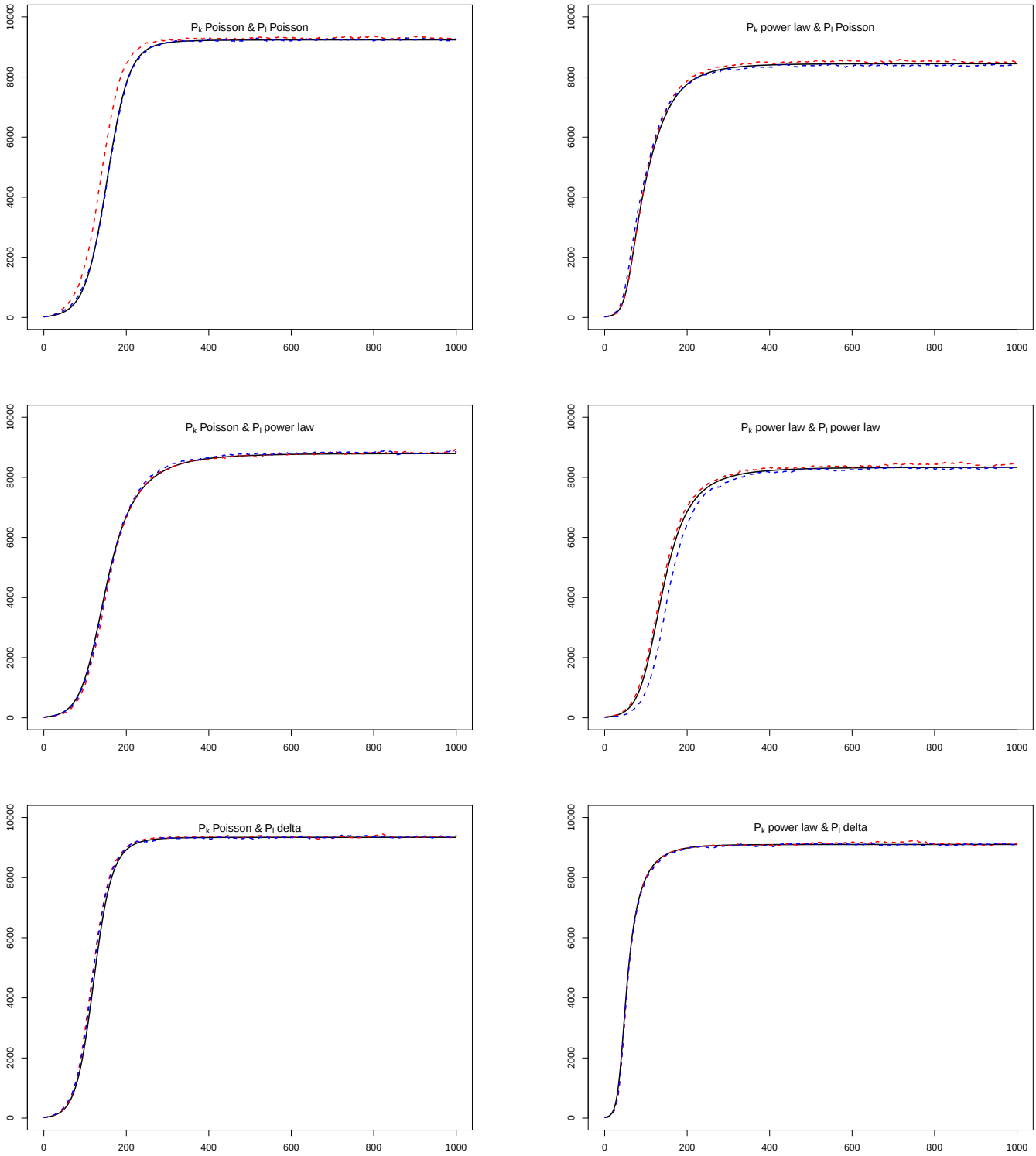


Figure 5: Dynamics of the number of infected hosts during the SIS epidemic spreading on different types of networks.