Université catholique de Louvain

LINMA2472 : Algorithms in data science

Homework 4 : Epidemiological SIR model over a network

Students : Vincent Cammarano - 5391 21 00
Louis Parys - 7256 17 00
Mattias Van Eetvelt - 1660 18 00

Group : 6

Professor : Jean-Charles Delvenne
Gautier Krings
Estelle Massart
Rémi Delogne
Bastien Massion
Brieuc Pinon

December 2022

Contents

1 Introduction

For our last project we wanted to create an SIR simulation on a network. The first part of the work was to build the network from the famous book from Homer, the Iliad. Since we already did that ,although not with the same book, the first homework, this part was pretty straight forward. Then, we analysed our network with help of certain metrics since those metrics would be useful for the second part of the project.
Finally, we implemented an SIR model on our network with an stochastic method and compared it with an SIR model build by solving the differential equations. We also varied the node from which the epidemic started from to find out how the spread would be affected by it.

2 Context

First in order to better understand the network, a short resume of the Iliad is presented.

Iliad is an ancient Greek epic poem written by the poet Homer in the 8th century BC. The poem tells the story of the last year of the Trojan War, a conflict between the city of Troy and the Greek armies led by Agamemnon, king of Mycenae.

The central character of the poem is Achilles, a Greek hero who is the greatest warrior of the age. The story begins with a dispute between Achilles and Agamemnon over a captured woman, Briseis, which leads to Achilles withdrawing from the war and refusing to fight. This decision has devastating consequences for the Greek army, as the Trojans, led by Hector, begin to gain the upper hand.

As the war rages on, the gods take an active role in the conflict, with Zeus and the other deities intervening on both sides. Eventually, Achilles decides to return to the fight and, with the help of his friend Patroclus, he is able to defeat Hector in single combat. However, Patroclus is killed in the process, and Achilles, consumed by grief and anger, seeks revenge by killing Hector.

The poem ends with the death of Hector and the eventual fall of Troy, but the consequences of the war continue to reverberate throughout the rest of the story.

3 Data preprocessing

3.1 Removing wrong names

A characters list was generated from the book using the NLTK library. Since the methods used are not perfect, some words that are not characters are included in the list. Therefor we first had to manually delete the words that do not represent any character.

3.2 Occurrence matrix

In order to be able to construct a network with the list of characters, we made a matrix of occurrence by looking through each paragraph of the book and link characters if they appeared in the same one paragraph. Once the matrix was created, we were able to make the dot product between the matrix and her transpose to get the adjacency matrix on which we will build the graph.

4 Network analysis

4.1 Visualization

At first, we created a graph using the NetworkX library, by extracting the nodes and edges with the previously computed adjacency matrix.

In order to have a better understanding of the network, the graph is plotted using a color gradient representing the degree of each node. The width of the edges represent the weight of a given edge. For graphical reasons, the weights were normalized.

Figure 1 shows that the center of the network is filled with nodes with the high degree while the edges of the graph is composed of low degree nodes. It is easy to understand that those highly connected nodes are the principal characters such as Achilles or Hector.

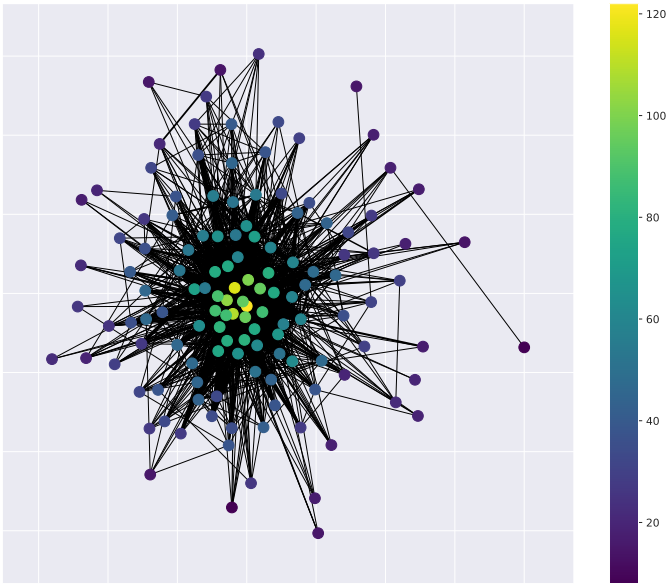


Figure 1: Basic representation of the network

4.2 Metrics on the network

4.2.1 Degree distribution of the nodes

On figure 1, we can see the distribution of the degree of the nodes in the network. There is some super connected characters with 100 and plus connected nodes, but the majority of the nodes have a degree between 20 and 60.

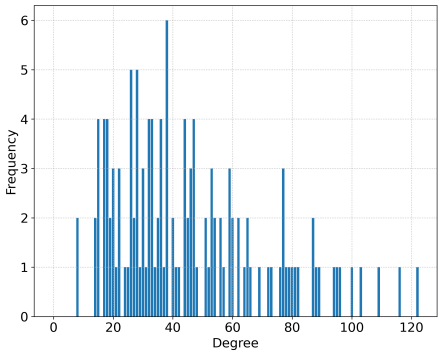


Figure 2: Degree distribution of the nodes

4.2.2 Degree centrality

Degree centrality assigns an importance score based simply on the number of links held by each node. As expected the principal characters have a higher degree centrality

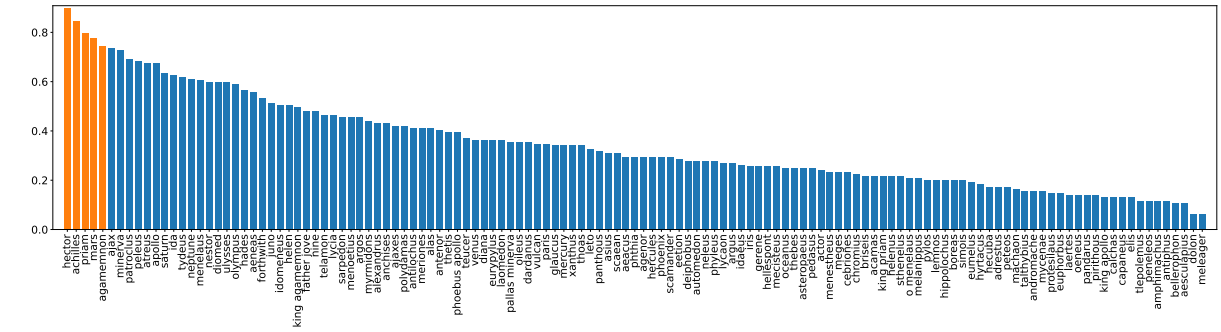


Figure 3: Score of the degree centrality on the network of characters

4.2.3 Betweenness Centrality

The betweenness centrality measure the number of time a node lies on the shortest path of other node. That means that if a node get a high score for this metric, it actually is a bridge between parts of the network.

We can see that character like Hector or Achilles get high scores, and of course they are main characters in the stories. It is then possible to say that they are gateway between some communities.

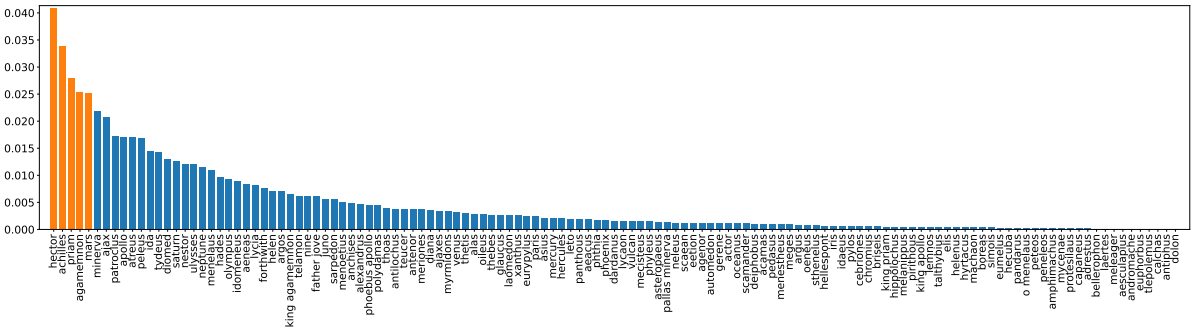


Figure 4: Score of the betweeness centrality on the network of characters

4.2.4 Eigen Centrality

The eigen centrality is an extension of the degree of a node. In fact, it first computes the degree of a node but it also computes the degree of the linked one. So if a node get a high score of eigen centrality we can say that he is highly connected to nodes that have a lot of influence.

Figure 5 shows that Hector and Achilles have high scores, meaning that they have a lot of influences and that they are bridges on highly connected parts of the network.

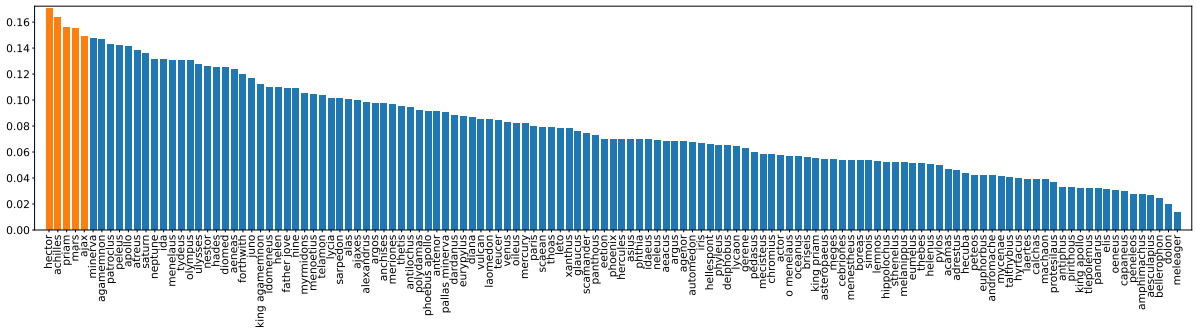


Figure 5: Score of the eigen centrality on the network of characters

4.3 Link analysis with PageRank algorithm

PageRank is an algorithm developped by Google that can be used to rank the importance of individual within the network. It works by assigning a numerical weight to each individual, based on the number and importance of the connections they have to other individuals. The higher the PageRank score, the more influential the individual is considered to be within the network.

This methods allows to highlight the principal characters. The top 3 nodes with the highest PageRank scare are Jove (Zeus) closely followed by Hector and Achilles. Achilles and Hector are expected results however Jove is quite surprising, especially since he is not in any of the top 10 centrality metrics. Another interesting observation is that those 3 characters have PageRank scores that are really high compared to the others characters. Agamemnon has the fourth highest score which is 0.027. This almost half the score of Hector, 0.051

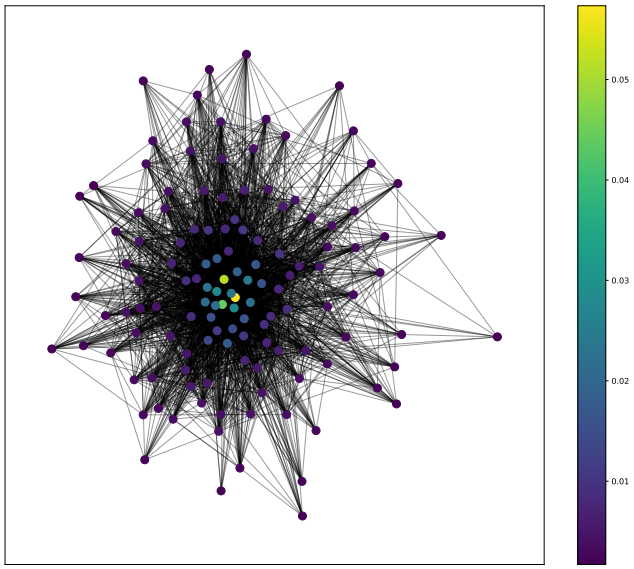


Figure 6: PageRank

5 Analysis of the communities

The goal of this section is to determine if there are distinct communities in the book. Their pertinence is studied and discussed using various algorithms.

5.1 Louvain algorithm

One of the tool we used for the detection of community, is the Louvain algorithm. The result is presented in figure 7. As it can be observed, it can't clearly be said that the method found distinct communities. Even if this is close to finding communities there is too much noise inside the clusters of color to be able to find distinct groups of characters. Lastly, the communities seem to be well distributed, there is roughly the same number of characters in each community. The 130 characters are distributed inside the communities as followed : 28-40-23-39.

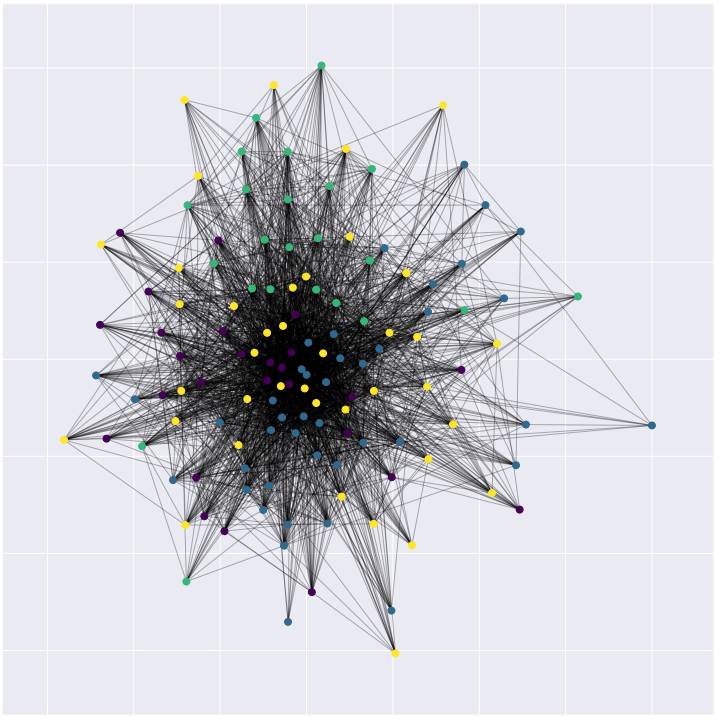


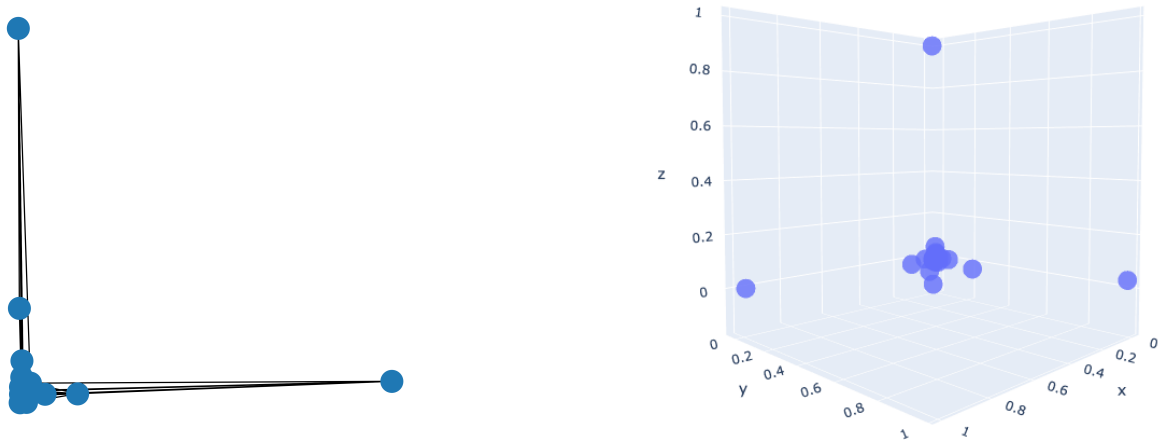
Figure 7: Louvain algorithm on the network of characters

Strangely enough, the two principal characters are not in the same community. Indeed, Hector is in community 3 with his brother Paris and his father Priam. On the other hand, Achilles is in community 2 with Patroclus, his army the myrmidons and his mother. Even if the Louvain algorithm does not reassemble the characters by importance, it classify them well by region. Hector is classed only with Trojans and Achilles in classified only with Grecs.

5.2 Spectral clustering

Another method to detect communities is the spectral clustering on the Laplacian. This is done using the *spectral layout* function from the NetworkX library.

From figure 8a, it can be observed that there are 3 communities. There is a big cluster as well, regrouping almost all of the nodes. However, when comparing to figure 8b, a fourth community appears. It was not visible using a 2D visualization. It is now safe to say that there are 4 communities. The Louvain algorithm computes 4 communities, but they are better distributed. In this case, 3 of the 4 communities only have one character, which seems a bit odd. What could have been expected was communities based on the importance of the characters or based on their respective regions, as with the Louvain algorithm.



(a) 2D Laplacian clustering

(b) 3D Laplacian clustering

Figure 8: Laplacian clustering visualization

5.3 K-means algorithm

The K-means algorithm is used to compare the results obtained with the Louvain algorithm. The graph is computed using the Kmeans function from the scikit-learn library where the numbers of communities is passed to 4.

As for the Laplacian clustering, there is one large community regrouping all the characters except 3 of them. Each one of these character represents one community. This results does not seem relevant. One possible explanation is that the network is too much connected. Hence, the data is not well suited for the K-means algorithm.

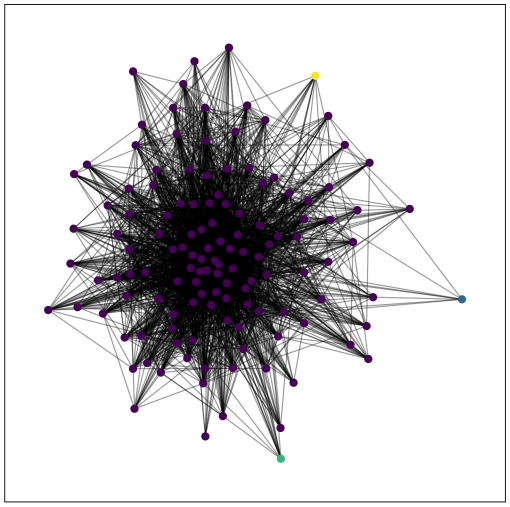


Figure 9: K-means clustering

5.4 Girvan-Newman algorithm

The Girvan-Newman algorithm works by iteratively removing the edges with the highest betweenness centrality until the desired number of communities is reached.

Again, there is a big community reassembling all the characters except Meleager. This clustering seems odd. However the modularity of the partitioned graph is $-4.2e^{-7}$ which can be approximated by 0. This means that output graph is undivided and it is the best that can be achieved.

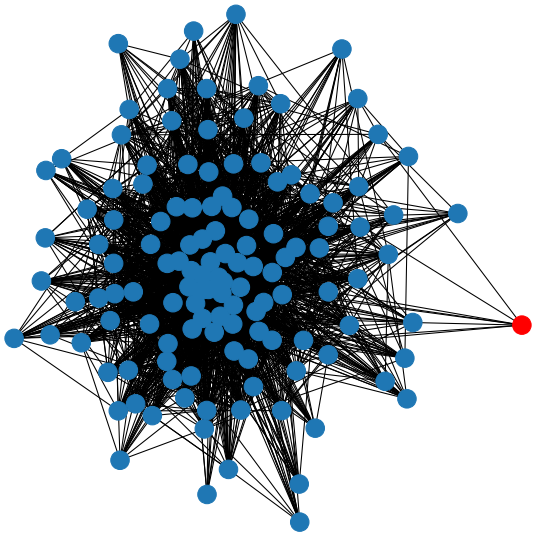


Figure 10: Girvan-Newman algorithm. The red node is Meleager

5.5 Discussion

As a conclusion concerning the community analysis, it can be said that the *Iliad* network is hard to divide in relevant communities. The best result is obtained using the Louvain algorithm with four well distributed communities. However, this result is not repeated using three others methods. For this reason, the spread inside the communities with not be studied in section 6, as it is judged to be irrelevant.

6 SIR Model

6.1 Stochastic modelization

The SIR model is a mathematical model commonly used in epidemiology to study the spread of infectious diseases. It is based on the assumption that a population can be divided into three compartments:

- Susceptible (S): individuals who are capable of being infected by the disease, but are not currently infected.
- Infected (I): individuals who are currently infected by the disease and can transmit it to others.
- Recovered (R): individuals who have recovered from the disease and are now immune to it.

The SIR model on a network describes the flow of individuals between these compartments over time, based on the following assumptions:

- A node can be infected only if infected nodes are linked to it.
- The probability of infection depends on first, the rate of infection (β parameter) and second, the number of infected nodes linked to the node in question. A node with a lot of infected nodes linked to it having a higher chance to get infected then a one poorly connected.
- Recovered individuals remain immune to the disease and do not move back to the susceptible compartment.

The first part of this section was dedicated to implement the SIR model on our network to see how the disease would spread trough it. Since SIR in Python is a common topic, we were able to find a library that was modeling the epidemiology for us. We used EoN (Epidemics on Networks) which is a library used for the simulation of epidemics on networks and solving ODE models of disease spread. The parameters used are $\beta = 0.0561215$ (infection rate) and $\gamma = 0.0455331$ (removal rate of infectives) of the covid from this paper [1].

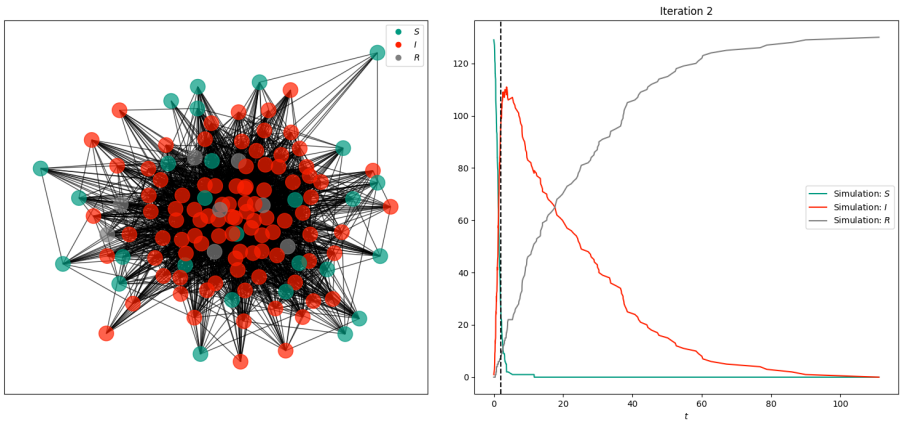


Figure 11: SIR modelization: First infected = Hector

Here is one frame of the SIR modelization, a .gif of the simulation has been provided in the .zip file of the homework for better visualization. It can be observed that the spread of the disease (in red) is very fast. This is due to the fact that the first infected node was chosen to be Hector. Since Hector is the node with the highest degree, betweenness and eigen centrality it makes him a super-spreader.

[1] `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7570398/`

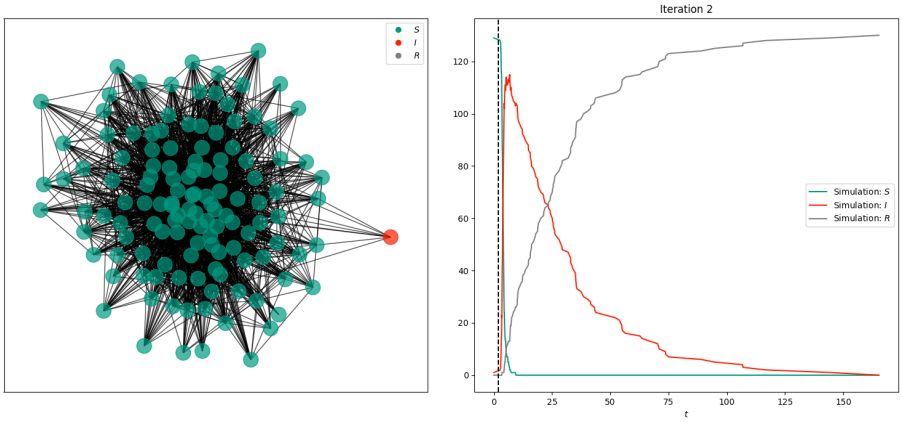


Figure 12: SIR modelization: First infected = Meleager

Here is the same model the picture has been taken at the same time as the one before but with Meleager as the first infected node. Meleager is the complete opposite of Hector since he has the lowest highest degree, betweenness centrality and eigen centrality. However, we can see that although the epidemic takes a bit more time to start it quickly spreads throughout the network still due to the high everage eigen centrality of our nodes.

6.2 Mathematical modelization

In this section we modeled the epidemic by solving the differential equations of the SIR model.
Here are the three equations of the model solving the number of susceptible (S), infected (I) and recovered (R) characters in the simulation.

$$\frac{dS}{dt} = -\beta SI \qquad \frac{dI}{dt} = \beta SI - \gamma I \qquad \frac{dR}{dt} = \gamma I$$

With β and γ being respectively the infection rate and removal rate as mentioned in section 6.1.

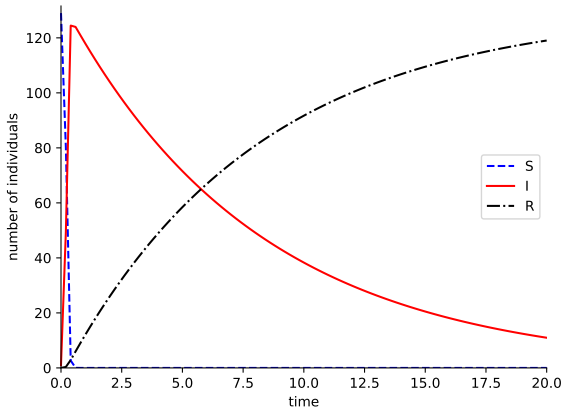


Figure 13: SIR modelization using differential equations

We can notice that the theoretical approach gives relatively similar outputs than the SIR made on our network. This is probably again due to the fact that the nodes of the network have a high degree, betweenness

and eigen centrality and that the network has a lack of clusters, due to the fact that it is highly connected. This shows that the graph has no singularities and that a disease spreads rapidly. If the network was less connected and had distinctive communities, the spread could be restrained to a specific community thus slowing down the spread and avoiding an epidemic.

7 Conclusion

In this work we first created a network based on the Homer book the Iliad. We then checked a series of metrics on it (degree distribution, degree, betweenness and eigen centrality as well as the PageRank algorithm). This part showed that our network was highly connected and showed to have an importance in the SIR section of the work. We then checked via different algorithm (louvain algorithm, spectral clustering, K-means algorithm and Girvan-Newman algorithm) for potential clusters in our network. Only to conclude that there were probably none due to the lack of repeatability in the results of the different algorithms. This did not allow us the check the impact of potential clusters on the SIR modelization conducted in the last part of the report. Finally, we modelled an SIR epidemic on our network using stochastic and a mathematical method. Those two methods showed significantly the same results probably due to the characteristics of our graph.