Type your solution using Word or LaTeX, and submit a PDF document on canvas.

Handwritten solutions will not be graded.

1 Introduction

The purpose of the final project is to have a research experience in a very active area of technology. It is not expected that the algorithm that you implement in your project works perfectly. This is the very nature of research. In order to have a high grade on the project, you need not have the best algorithm (see grading section). I will, however, reward ingenuity and creativity.

2 What is expected from me, and how can I get a high grade?

1. Read this document carefully
2. Download the data from Canvas, familiarize yourself with the data.
3. Submit the different parts of your project report by the corresponding deadlines (see next sections).
4. Present your project in class during the week of April 29, 2019. You will have 20 minutes and 5-10 additional minutes to answer questions.

2.1 Deliverable and schedule

The following sections of your project report should be submitted on canvas:

- Part I by April 12, 2019
- Part II by April 26, 2019

Your group will need to prepare and give a 20 minutes class presentation during the week of April 29, 2019. The presentation will be in front of the other students. This is a unique chance to learn how to present scientific and technical material in front of an audience. There will be time (5-10 minutes) for questions after each presentation.

2.2 Grading policy

1. Each part (I & II) counts for 40% of the project grade. Please note that 5% of each report grade will be for organization. This category allows us to give you points if your lab report is well organized, and pleasant to read. While this category may sound subjective, objective criteria include:

- figures with captions, title, labels on both axes with units of measurements, error bars, and a legend;
- using a standard variable-width font, in 12 pt (e.g., Times New Roman, Palatino, etc.) with 1-inch margins;
- proper spelling and grammar.

2. Present your project in class during the final exam time period. The presentation should clearly demonstrate the work that has been accomplished. You need to describe the work that you have done, even if your idea did not work as well as expected. [20 % of the grade].

3 Mission challenge

In this project, you will train to be a data scientist: you will help a researcher working at the Center for Disease Control (CDC) predict the spread of epidemics using face-to-face contact networks. Alternatively, the same data science principles can be used to detect, monitor and predict the outcome of Internet worms and viruses using darknet traffic data.

Massive physical, operational, or communication networks have become ubiquitous today. Such networks have the following structure: a massive number of nodes are functionally connected through physical or virtual links (e.g., online social network). Each node is described by a set of time-dependent attributes, or features. The same abstraction can be used to model the spread of rumors on a social network, or the propagation of malware on a computer network. Such very large networks cannot be directly observed. One must resort to the monitoring of these networks on a much smaller scale: a succinct subset of nodes and associated edges are extracted from the original network.

Since *massive graphs* can no longer be analyzed by directly computing properties of the corresponding adjacency matrix $\boldsymbol{A}$, the properties of the original graph need to be estimated from a much smaller subgraph, $G_s = (V_s, E_s)$, defined by a subset of nodes $V_s \subset V$, and a subset of edges $E_s \subset E$ of the original graph.

The sampling of the graph can be formalized as the sampling of the entries of the (inaccessible) adjacency matrix $\boldsymbol{A}$. In practice, several sampling strategies, which can combine deterministic and random sampling of edges and vertices, have been considered. We describe in Section 6 the sampling algorithms that you will implement in this project.

4 The Data

This project is using six datasets collected by English, French and Italian researchers. The data are available at the SocioPatterns website. These datasets are well known and are used to benchmark algorithms in network science and graph theory.

The data provide temporal information about face-to-face contacts between individuals in several environments. Each individual was wearing a sensor capable to detect face-to-face close range proximity (1.5 m) between the sensors. In addition to this dynamic face-to-face contact network, the researchers also recorded the location – albeit with a much coarser resolution – of the participants using RFID readers located at various locations inside the environment wherein the participants interacted. There are therefore two datasets, sampled at the same exact instants,

name	location	number of nodes n
InVS13	French Institute for Public Health Surveillance	92
InVS15	French Institute for Public Health Surveillance	232
LH10	Hospital ward (Lyon, France)	81
LyonSchool	Primary school (Lyon, France)	242
SFHH	2009 French Society for Hospital Hygiene Conference	403
Thiers13	High School (Marseilles, France)	326

Table 1: Name, origin, and size (number of nodes) of the datasets

1. a time series of face-to-face contact events between individuals,
2. a time series of co-presence events between individuals.

Table 1 provides the name, origin, and size (number of nodes) of the datasets. In each case, we have access to two matrices that encode face-to-face contacts and co-presence. Each matrix has three columns and a large number of rows. Each row contains a time index t and two node indices i and j. The presence of the row indicates that nodes i and j were in face-to-face (co-presence) contact at time t. The data are sampled with a temporal resolution of 20s. Time is measured in seconds and expressed in UNIX `ctime`.

A ZIP archive of the six datasets can be retrieved here: click to download. You simply need to load the dataset in MATLAB. For instance,

```
>> load tij_LyonSchool.dat
>> whos
  Name                    Size              Bytes  Class     Attributes

  tij_LyonSchool      125773x3            3018552  double
```

The first contact happens between node $i = 1558$ and node $j = 1567$ at time $t = 31220$. The last contact happens between node $i = 1913$ and node $j = 1922$ at time $t = 148120$.

```
>> tij_LyonSchool(1:1,:)

ans = 31220        1558        1567
>> tij_LyonSchool(end:end,:)

ans = 148120        1913        1922
```

Assignment [70 = 6 x 2 x 5 + 10]

1. For each of the six datasets, and for each type of contact (face-to-face and co-presence), build the static, temporally aggregated network, where each entry of the adjacency matrix of the corresponding network is given by

$$A(i,j) = \text{total number of contact events between nodes } i \text{ and } j. \tag{1}$$

We can represent each participant as a node of a graph. If two individuals i and j are in contact (face-to-face or co-presence), then we connect i and j with an edge. The weight $A(i,j)$ along the edge provides a natural mechanism to quantify the strength of the face-to-face or co-presence contact between the individuals i and j. If $A(i,j) = 0$, then nodes i and j were never in contact, and there is no edge between i and j.

In this document, all graphs are defined with respect to the temporally aggregated matrices, $\boldsymbol{A}$, defined by (1).

Definition 1 *For each of the six datasets, $i = 1, \ldots, 6$, defined in Table 1, we denote by $\boldsymbol{A}_f^{(i)}$ and by $\boldsymbol{A}_p^{(i)}$ the i^{th} face-to-face and co-presence contact networks respectively.*

2. You should observe that the co-presence graph has many more edges, with larger weights, than the face-to-face contact graph. Please explain this phenomenon.

In the rest of the project, we will sample the denser co-presence graph and compare it to the thinner face-to-face contact graph.

5 How do we compare the original graph to the reduced graph?

Throughout this project, you will evaluate the performance of several algorithms to reduce the size of a large graph $G = (V, E)$. When sampling very large graphs, one is confronted with the following fundamental question: can the large scale structural properties of the graph G be recovered from the sampled subgraph $G_s = (V_s, E_s)$?

The evaluation of the performance of the various sampling methods has focused so far on the ability to recover from the subgraph G_s the local statistics of the original graph, such as vertex degree distribution, average or global clustering coefficient, etc. We describe in the following these statistics.

5.1 Step 1: Global scale graph properties

The first step is used to check whether the large scale properties of the graph have changed.

Given a weighted graph $G = (V, E, \boldsymbol{w})$, you will compute the following statistics.

1. size = number of vertices = $n \stackrel{\text{def}}{=} |V|$
2. $m \stackrel{\text{def}}{=}$ number of edges
3. volume = sum of the weights along all the edges $\stackrel{\text{def}}{=} \sum_{e \in E} w(e)$
4. density = ratio of the number of edges over the maximum possible number of edges (given the number of nodes)

$$\text{density} \stackrel{\text{def}}{=} \frac{2m}{n(n-1)}. \tag{2}$$

5.2 Step 2: Local scale graph properties

The second step is used to check whether the local scale properties of the graph have changed.

Given a weighted graph $G = (V, E, \boldsymbol{w})$, you will compute the following statistics.

4. degree distribution

$$\deg(v) \stackrel{\text{def}}{=} \sum_{u:(u,v) \in E} w_{uv}. \tag{3}$$

   Because many degree distribution decay as $1/k^{\gamma}$, one should consider displaying the histogram using a log-log plot: the logarithm of the number of nodes with degree k should be displayed as a function of the logarithm of the degree, $\log k$.

5. average degree

$$\overline{d} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{v \in V} \deg(v). \tag{4}$$

We define the neighborhood of a vertex v as the set of vertices that are directly connected to v by and edge. Please note that v is not in the neighborhood of v.

Definition 2 *Let $v \in V$. The neighborhood of v is defined by*

$$N(v) \stackrel{\text{def}}{=} \{w \in V \backslash \{v\} : (v, w) \in E\}. \tag{5}$$

6. clustering coefficient. This is the ratio of the actual number of triangles that include v as a node and for which the two other nodes are in the neighborhood of v, over the maximal possible number of such triangles (given the size of $N(v)$). The clustering coefficient at v can be computed as follows,

$$\mathrm{cc}(v) \stackrel{\text{def}}{=} \frac{2|\Delta(v)|}{\deg(v)\,(\deg(v)-1)} \tag{6}$$

where

$$\Delta(v) \stackrel{\text{def}}{=} \{(u,w)|u \in N(v), w \in N(v), (u,w) \in E\} \tag{7}$$

is the set of edges in the neighborhood of v, $N(v)$.

Because nodes tends to cluster in social networks (the friend of my friend is also my friend), the clustering coefficient can be used to quantify the "transitivity" of the connectivity.

7. average clustering coefficient

$$\overline{\mathrm{cc}} \stackrel{\text{def}}{=} \frac{1}{n}\sum_{v\in V}\mathrm{cc}(v). \tag{8}$$

5.3 Step 3: Advanced structural graph properties

The final step involves quantifying structural changes that happen at multiple scales. In section ?? we discuss sophisticated distances that can be used to capture these multiscale changes created by sampling.

Assignment [900 = 6 x 2 x 7 x 10 + 6 x 10]

3. For the 12 contact networks $\left(\boldsymbol{A}_f^{(i)}, \boldsymbol{A}_p^{(i)}\right), i = 1,\ldots,6$, evaluate the 7 statistics described in sections 5.1 and 5.2.

   When you need to compute a probability distribution (e.g., degree, clustering coefficient), please display an histogram. If the histogram (probability distribution) of a statistic (e.g., degree) appears to decay as a rational function of k, please try a log-log plot.

   You can use a table to display all the other statistics, where you only need to compute a scalar.

4. For each of the six datasets, compare the statistics of the face-to-face networks, with those of the co-presence networks. Explain your findings.

6 Sampling of Large Graphs

6.1 The Naive Approaches

We first describe naive approaches that either lead to unreasonable models of computation (e.g., the adjacency matrix will never fit in memory), or have very poor properties (e.g., the sampled graph is disconnected).

6.1.1 Node sampling: induced subgraph sampling

The node sampling algorithm is described in Algorithm 2. A subset $V_s \subset V$ of vertices are selected independently according to a probability distribution $\boldsymbol{p}$: a node v is selected with a probability $p(v)$. Once k vertices have been selected, the edges of the sampled graph, E_s, are added by considering the induced subgraph: $e = (v, w) \in E_s$ if and only if $v \in V_s$ and $w \in V_s$ (see Fig. 1). The probability distribution $\boldsymbol{p}$ can be

- uniform: all vertices are equally likely;
- proportional to the degree distribution: nodes with high degree are more likely to be selected

A variation of the node sampling involves selecting all the neighbors of the subset V_s. The subset of edges remain the subgraph induced by the subset of nodes and their neighbors.

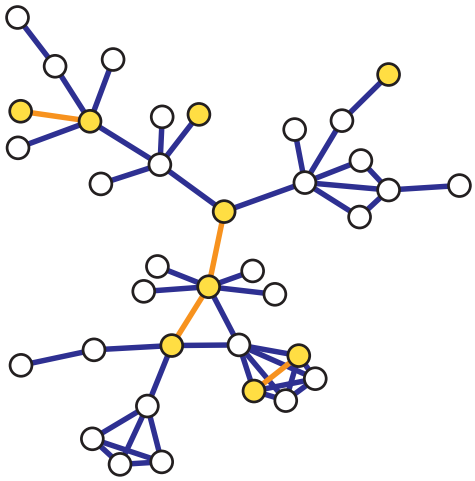

Figure 1: Sampled nodes are shown in yellow. Edges connecting the sampled nodes are in orange. Several nodes are isolated.

Algorithm 1 Induced Graph Sampling

```
1: procedure nodeSampling(G = (E, V), p, k)
   // Input: E, k, p
   // Output: V_s the reduced vertex set; A_s the adjacency matrix of the reduced edge set
   // Return a subset of k vertices V_s chosen with probability distribution p from V,
   // and the induced set of edges

   // Selection of the vertices
2:     V_s ← ∅
3:     for j ← 1, k do
4:         Select a vertex v ∈ V at random according to the probability distribution p(v)
5:         V_s ← V_s ∪ {v}
6:     end for

   // Selection of the edges
7:     for i ← 1, k do
8:         for j ← 1, k do
9:             v ← V_s(i)          // indices of the vertices in the original graph
10:            w ← V_s(j)
11:            A_s(i, j) ← A(v, w) // the reduced adjacency matrix
12:        end for
13:    end for

14: end procedure
```

6.1.2 Edge sampling: incident subgraph sampling

The edge sampling strategy involves selecting a subset of edges E_s at random from the set of edges according to a probability distribution $\boldsymbol{p}$. The subset of vertices is formed by collecting all the endpoints of the subset of edges E_s (see Fig. 2). Algorithm 2 describes the edge sampling algorithm.

Edge sampling and node sampling can be combined in a manner whereby a vertex v is selected at random, and an edge $e = (v, w)$ incident to v is then selected at random.

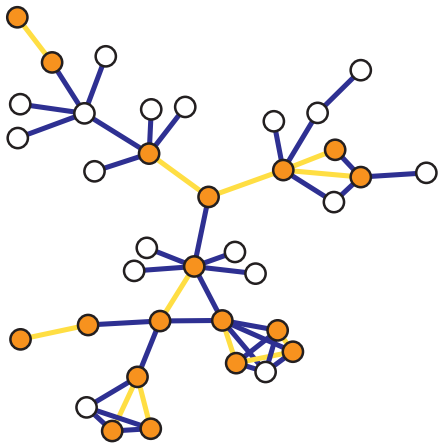

Figure 2: Sampled edges are shown in yellow. Vertices incident to the sampled edges are in orange. Several edges are isolated.

Algorithm 2 Edge Sampling

procedure `edgeSampling`$(G = (E, V), \boldsymbol{p}, k)$
// Input: $E, k, \boldsymbol{p}$
// Output: E_s the reduced edge set, V_s: the reduced vertex set
// return a subset of k edges from E chosen with probability distribution $\boldsymbol{p}$
    $E_s \leftarrow \emptyset$
    $V_s \leftarrow \emptyset$
    for $j \leftarrow 1, k$ **do**
        Select an edge $e \in E$ at random according to the probability distribution $p(e)$
        $E_s \leftarrow E_s \cup \{e\}$
        $V_s \leftarrow V_s \cup \{e^+, e^-\}$
    end for

end procedure

6.2 Sampling by random exploration

In contrast to the previous naive approaches, these sampling strategies can yield more faithful replica of the original graph, using computationally efficient algorithms. We note that many of the algorithms require that the neighborhood of a given node be readily accessible (fast random access in the main memory, as opposed to random access on disk, which is much slower).

The central idea of these algorithms is to construct the reduced graph by assembling several connected paths. Each path is initiated at some random vertices in the graph, and the algorithm explores the graph using a combination of deterministic and stochastic strategies. We describe in the following sections several sampling strategies based on the random exploration of the graph.

6.2.1 Metropolis-Hastings Random Walk

The random walk algorithm starts at a randomly chosen vertex v_0. The next vertex v_1 is chosen at random according the probability

$$p(v_1) = \frac{A_{v_0 v_1}}{\deg(v_0)} \tag{9}$$

where the degree of v_0 is computed using the formula for weighted graphs (3),

$$\deg(v_0) \stackrel{\text{def}}{=} \sum_{v \in N(v_0)} A_{v_0 v}. \tag{10}$$

The simple random walk is inherently biased: nodes with high degrees will be visited more often. The following sampling strategy, called the Metropolis-Hastings Random Walk corrects this bias. The idea is to always accept the move towards a node of smaller degree, and reject some of the moves towards higher degree nodes. This eliminates the bias towards high degree nodes.

Algorithm 3 Metropolis-Hastings Random Walk

```
1:  procedure metropolisHastingsRW(G = (E, V), m)
    // Input: V, E, m_s
    // Output: the reduced edge set E_s of size m_s
2:      S ← ∅
3:      v ← random (V))
4:      while |S| ⩽ m_s do
5:          Select new node w ∈ N(v) uniformly at random
            // choose p according to a uniform distribution on [0,1]
6:          p ← U[0, 1]
7:          if p ⩽ deg(v) / deg(w) then
8:              v ← w
9:          else
10:             Stay at v
11:         end if
12:     end while
13: end procedure
```

6.2.2 Frontier Sampling

Frontier sampling randomly selects a list of m seed vertices. The vertices are selected by uniformly sampling the set of vertices V. The list of seed vertices is visited randomly, with a probability proportional to the degree of each vertex. The seed vertex is replaced by one of its neighbors at random, and the exploration continues with the new vertex. The algorithm stops the exploration once its has sampled m_s edges. Algorithms 4 describes the Frontier Sampling procedure.

Algorithm 4 Frontier Sampling

procedure `frontierSampling`$(G = (E, V), m_s)$
// Input: E, m_s
// Output: E_s the reduced edge set
// return a subset of m_s edges from E chosen with probability distribution $\boldsymbol{p}$
    $k \leftarrow 0$ // k is the number of iterations
    Initialize $L = \{v_1, \ldots, v_m\}$ with m randomly chosen vertices (uniformly)
    repeat
        Select $u \in L$ with probability $p(u) = \deg(u) / \sum_{v \in L} \deg(v)$
        Select an outgoing edge of (u, v) uniformly at random
        Replace u with v in L
        $E_s \leftarrow E_s \cup \{(u, v)\}$
        $V_s \leftarrow V_s \cup \{u, v\}$
        $k \leftarrow k + 1$
    until $k \leqslant m_s$
end procedure

6.2.3 Snowball Sampling

Snowball sampling randomly selects a set V_0 of vertices. The vertices are selected by uniformly sampling the set of vertices V. For each vertex v in V_0, we consider the neighborhood of v formed by the vertices connected to v. We extend this definition to any subset S of vertices.

Definition 3 *Let $S \subset V$. The neighborhood of S is defined by*

$$N(S) \stackrel{def}{=} \{w \in V \backslash S : \exists v \in S, (v, w) \in E\}. \tag{11}$$

The snowball sampling creates a series of waves. During the first wave, V_0 is extended to $V_1 \stackrel{\text{def}}{=} N(V_0) \cap (V \backslash V_0)$. During the second wave, V_1 is extended to $V_2 \stackrel{\text{def}}{=} N(V_1) \cap (V \backslash (V_0 \cup V_1))$ (see Fig. 3). The reduced graph is formed by collecting the vertices sampled during each wave, $V_s \stackrel{\text{def}}{=} V_0 \cup V_1 \cup \ldots \cup V_k$. The edges are the incident edges, created by the connection between the successive waves.

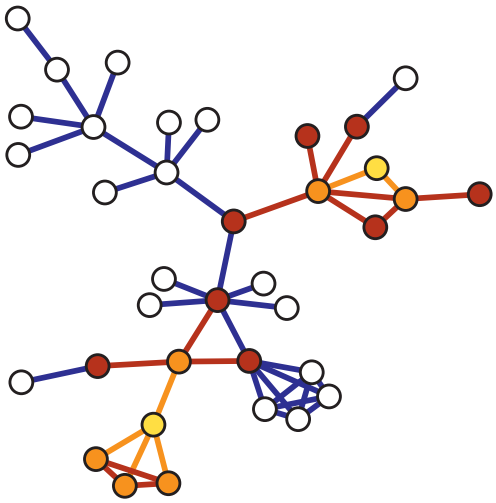

Figure 3: Snowball sampling. The initial set of vertices V_0 is shown in yellow. Vertices reaching to the second wave, V_1, are shown in orange. The second wave of nodes, V_1, is shown in orange as well. The third waves of nodes is shown in maroon.

6.2.4 Snowball Expansion Sampling

The Snowball Expansion sampler is a variation of the snowball sampler that maximizes the expansion of the graph between the successive waves.

Algorithm 5 Snowball Expansion Sampling

procedure `snowBallExpansion`($G = (E, V), m_s$)
// Input: V, E, m_s
// Output: V_s the reduced vertex set
    $S \leftarrow \varnothing$
    $v =$ random $(V))$
    $S \leftarrow S \cup \{v\}$
    while $|S| \leqslant m_s$ **do**
        Select new node $v \in N(S)$ based on maximization of $|N(\{v\})\backslash(N(S) \cup S)|$
        $S \leftarrow S \cup \{v\}$
    end while
end procedure

Assignment [490 = 6 x 7 x 10 + 7 x 10]

Implement the seven sampling algorithms: Algorithms 1-7.

5. For each sampling algorithm, and for the six contact networks $\boldsymbol{A}_p^{(i)}$, create five subsampled networks, using five sampling ratios, $f = 0.9, 0.8, 0.7, 0.6, 0.5$. The sampling ratio f measures the proportion of edges that remain in the sampled network, irrespective of their locations,

$$f \stackrel{\text{def}}{=} \frac{|E_s|}{|E|}. \tag{12}$$

   For each sampled network, you will evaluate the 7 statistics described in sections 5.1 and 5.2.

   When you need to compute a probability distribution (e.g., degree, clustering coefficient), please display an histogram. If the histogram (probability distribution) of a statistic (e.g., degree) appears to decay as a rational function of k, please try a log-log plot.

   You can use a table to display all the other statistics, where you only need to compute a scalar.

6. For each sampling strategy, compare the seven statistics computed from the original face-to-face contact network $\boldsymbol{A}_f^{(i)}$ with those computed from the sampled co-presence network $\boldsymbol{A}_p^{(i)}$.

   Propose some simple strategies to remedy some of the biases introduced by the sampling algorithms. For instance, you may assume that you have access to global statistics: size, volume of the graph, etc. to correct for a bias in the degree distribution.

7 Spread of Epidemics

We now consider a simple model of an infectious disease spreading on the network that is transmitted when two individuals are in contact (either face-to-face or co-presence). Historic examples of such epidemics include the Great Plague of London (17th century), the 1918 influenza pandemic, the Severe Acute Respiratory Syndrome (SARS) outbreak of 2003, the 2009 influenza A (H1N1) epidemic, etc.

Similar mathematical models can be used to model gossips on social networks, or the propagation of computer viruses in the context of cybersecurity.

7.1 The susceptible-infective-recovered (SIR) model on a network

We describe the Kermack-McKendrick epidemic model, also known as the susceptible-infective-recovered (SIR) model. In this model the total number of nodes n is divided into three subsets:

$$n = S(t) + I(t) + R(t), \tag{13}$$

where

1. $S(t) =$ number of susceptible individuals who have not been infected, but have no immunity and thus can be infected if exposed to the disease;
2. $I(t) =$ number of infected individuals who can transmit the infection to susceptible individuals by contact;
3. $R(t) =$ number of recovered (or immune) individuals who were previously infected, but have since then recovered and gained immunity against the infection. These individuals can neither become infected, nor transmit the disease.

The random variables $S(t), I(t)$ and $R(t)$ evolve according to the following dynamics.

- If v is susceptible at time t and it is the neighbor of an infected node u, then v can become infected at time $t + dt$ with a probability proportional to

$$\beta A_{uv} dt, \tag{14}$$

  where A_{uv} is the entry in the adjacency matrix of the contact network, which accounts for the amount of contacts that happened between u and v.

- If v is infected at time t, then v can recover at time $t + dt$ with a probability proportional to

$$\mu dt. \tag{15}$$

- If v is recovered (immune) at time t, then its status no longer changes.

The parameter β (measured in inverse of time units) controls the infection rate: if v is the neighbor of an infected node, then the average time interval for v to become infected is $1/\beta$.

Similarly, the parameter μ (also measured in inverse of time units) is the recovery rate: it takes a time interval of $1/\mu$ for node v to recover once it has been infected, or equivalently the mean infectious period is $1/\mu$.

To understand the roles of β and μ and their effect on the dynamic of the infection, let us consider the situation where all the vertices are initially healthy but susceptible.

Once a single node is infected, it will infect about $\beta\langle d\rangle dt$ nodes over the time step dt, where $\langle d\rangle$ is the average degree of the graph G. If we integrate this number of infections over the mean infection period, we obtain about $\beta\langle n\rangle/\mu$ nodes infected by node v.

Now, each infected node will itself infect about $\beta\langle n\rangle/\mu$ nodes during the mean infectious period. The number

$$\rho_0 = \frac{\beta}{\mu}\langle n\rangle \tag{16}$$

is known as the basic reproduction number. ρ_0 determines if an epidemic will occur: if $\rho_0 < 1$, then the infection will die out, whereas if $\rho_0 > 1$ the number of infected vertices will increase exponentially.

In this model, an epidemic will always come to an end when sufficiently many nodes are protected because they have recovered, and gained immunity.

7.2 Numerical simulation of the SIR model on a network

We consider the problem of simulating an epidemic on a graph $G = (V, E)$ with adjacency matrix $\boldsymbol{A}$. To carry out the simulations we need the following variables:

- β: infection rate
- μ: recovery rate
- dt: time step
- T: time interval for the simulation
- $\boldsymbol{A}$: adjacency matrix

The pseudo-code in Algorithm 6 describes the simulation of the SIR infection on the graph $G = (V, E)$. The source of the infection is a random node v_0.

All epidemics are initiated with a source node chosen at random uniformly amongst the set of vertices. The simulation stops at $t = t_\infty$ when the set of infectious nodes is empty, to wit all vertices are either immune (recovered) or were never infected (susceptible).

The impact of the epidemic is quantified using the fraction of the number of vertices n_r that recovered (and therefore were infected),

$$n_r \stackrel{\text{def}}{=} \frac{R(t_\infty)}{n}. \tag{17}$$

Specifically, for each simulation of an epidemic, we assess whether $n_r > 0.2$, and if this is the case we record n_r. Such simulations where at least 20% of the population was infected are considered to be large outbreaks.

Finally, in all simulations, we use $\beta = 4 \times 10^{-4}$, and we vary μ so that

$$\rho \stackrel{\text{def}}{=} \frac{\beta}{\mu} \in [1, 256]. \tag{18}$$

Algorithm 6 numerical simulation of the SIR model

```
1:  procedure SIR(A, β, μ, dt, v_0, V)

        // Initialize all the variables
2:      nTimeSteps ← T/dt
3:      q := μdt

4:      susceptibleNodes ← V
5:      infectiousNodes ← {v_0}
6:      recoveredNodes ← ∅

        // main loop
7:      for t := 1, nTimeSteps do

8:          for all v ∈ infectiousNodes do

                // neighbors of v become infected with probability p
9:              for all u ∈ neighbors(v) do
10:                 if u ∈ susceptibleNodes then
11:                     p := βA(u, v)dt
12:                     infectiousNodes ← v_0 with probability p
13:                 end if
14:             end for

                // v becomes recovered with probability q
15:             recoveredNodes ← v_0 with probability q
16:         end for
17:     end for
18: end procedure
```

In Section 6 you have compared the precise face-to-face contact datasets to subsampled versions of the co-presence datasets. Face-to-face contacts might be difficult to obtain; conversely co-presence information is easily available.

The comparison of network statistics is interesting, but does not necessarily yield reliable information about the possibility of using the co-presence datasets as surrogates for the face-to-face contact. In this last part of the project, you study the numerical simulation of an epidemic on the detailed aggregated face-to-face contact, and compare it to a simulation obtained on a sampled version of the co-presence network. Furthermore, you will study if vaccinations policies devised using the coarser co-presence data can be as effective as those that are designed using the precise face-to-face contact network data.

Assignment [720 = 6 x 3 x 20 x 2]

For each of the six face-to-face datasets, perform 100 simulations of the SIR epidemic, using the followings parameters:

- the unique vertex that is the source of the infection is chosen uniformly at random;
- $\beta = 4 \times 10^{-4}$;
- $\mu = \frac{\beta}{k}, k \in \{1, 4, 16, 64, 256\}$

7. Plot as a function of ρ_0 (given by (16)) the distributions of recovered nodes.
8. Plot as a function of ρ_0 the fraction of epidemics where the final fraction of recovered nodes, n_r, given by (17), is greater than 20 %.
9. Plot as a function of ρ_0 the average number of recovered nodes for those epidemics where the final fraction of recovered nodes is greater than 20 %.

Assuming that you can keep 80% of the edges ($f = 0.8$ in (12)), use the best network sampling method to construct six sampled co-presence networks, for each of the six networks. You will perform 100 simulations of the SIR epidemic on the sampled networks using the same parameters as for the face-to-face network.

10. Generate the three plots described in questions 7-9 and compare them to the corresponding face-to-face plots. It will be useful to generate a unique figure with both plots (original face-to-face and downsampled co-presence).

7.3 Effect of Vaccination

We study a simple model of vaccination where a subset of nodes acquire an immunity to the infection. These nodes cannot transmit the disease and are never infected. You will study two different public health strategies:

1. vaccination of 20 nodes at random;
2. vaccination of the 20 nodes with the highest degrees.

Assignment [720 = 6 x 3 x 20 x 2]

For each of the six face-to-face datasets, perform 10 simulations of the SIR epidemic, using the followings parameters:

- the unique vertex that is the source of the infection is chosen uniformly at random;
- $\beta = 4 \times 10^{-4}$ and $\mu = \beta$;
- 20 nodes are vaccinated at random or
- the 20 nodes with the highest degrees are vaccinated.

11. Plot the ratio between the fraction of the total number of outbreaks that reach at least 20% of the population with and without vaccination.
12. Plot the ratio between the median outbreak size (that reach at least 20% of the population) with and without vaccination.
13. Comment on the best vaccination policy.

Assuming that you can keep 80% of the edges ($f = 0.8$ in (12)), use the best network sampling method to construct six sampled co-presence networks, for each of the six networks. You will perform 100 simulations of the SIR epidemic on the sampled networks using the same parameters as for the face-to-face network.

14. Generate the three plots described in questions 11-13 and compare them to the corresponding face-to-face plots. It will be useful to generate a unique figure with both plots (original face-to-face and downsampled co-presence).