UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

MASTER THESIS No. 1286

Detectability of Patient Zero Depending on its Position in the Network

Iva Miholić

Zagreb, June 2016.

Umjesto ove stranice umetnite izvornik Vašeg rada.

Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`*.*

CONTENTS

List of Figures

List of Algorithms

1. Introduction

A *network* is a set of items with connections between them. The Internet, the World Wide Web, social networks like genealogical trees, networks of friends or co-workers, biological networks like epidemiological networks, networks of citations between papers, distribution systems like postal delivery routes: they all take a form of networks. Most social, biological and technological networks have specific structural properties. Such networks are referred to as *complex networks*. An example of a complex network is represented on Figure 1.1.

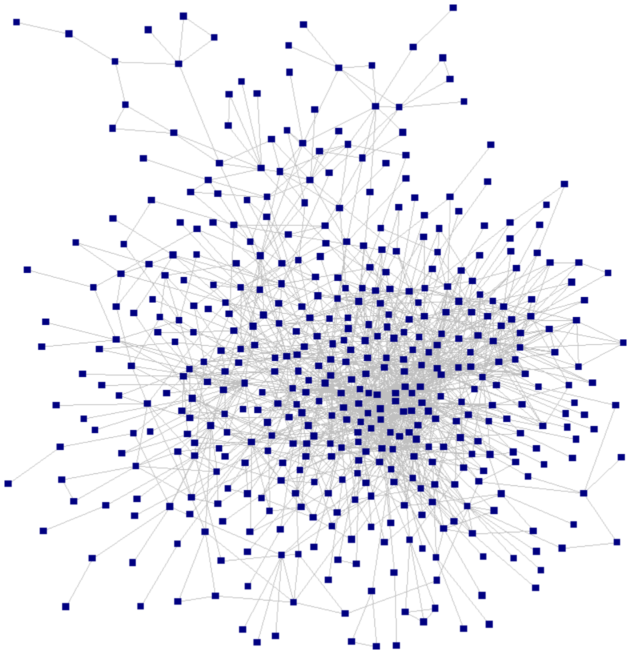

Figure 1.1: A network graph of Paul Erdős and his collaborators, courtesy of Krebs [1]. The nodes represent mathematicians and the edges represent the relationship "wrote a paper with".

A network structure or a topology can be mathematically modelled as a graph with set of vertices (or nodes) representing the items of the network. The network structure can then be analysed using graph theory. An edge between two nodes represents a connection between the two corresponding items. Edges can be directed or undirected, depending on the nature of the connection. To better mimic the real-world (complex) network structure, it is common to add attributes to nodes and/or edges or to have both directed and undirected edges on the same graph.

For large-scaled complex networks that have millions or billions of vertices, the study in the form of traditional graph theory is not sufficient or sometimes possible. When this is the case, the statistical methods for quantifying large complex networks are used.

The ultimate goal of the study of complex network structure is to understand and explain the workings of systems built upon network such as spreading of disease or information propagation.

After statistical properties analysis, the model of the system or a process is created. The model can help us understand the meaning of statistical properties - how they came to be as they are and how they relate to the behaviour of a networked system. Based on the statistical properties and using the right model, the behaviour of networked systems can be determined and predicted.

The basis of the complex network theory – the structure analysis and the process modelling – can be found in Newman [3].

1.1. Epidemic processes in complex networks

The models for stochastic processes such as disease spreading are categorized as homogeneous or heterogeneous mixing frameworks. The former assume that all individuals in a population have an equal probability of contact and different equations can be applied to understand epidemic dynamics. Since such models fail to describe the realistic scenario of disease spreading, heterogeneity is introduced by using a network structure.

There is an extremely close relationship between epidemiology and network theory since the connections between individuals (or group of individuals) allowing an infectious disease to propagate naturally define a contact network. Simplest epidemic dynamics consider a system with fixed total population consisting of N individuals modelled with undirected contacting network. We define the contact network as an undirected and non-weighted graph $G(N, L)$ with fixed set of nodes N and fixed set of links L. A link (u, v) between two nodes exists if the two corresponding members were in contact during the epidemic time.

The structure of the network has profound impact on the contagnion dynamics but in order to understand the evolution of the epidemic over time we have to define the basic individual-level processes that govern the epidemic spreading. Complementary to the network, epidemic modelling describes the dynamical evolution of the contagion process within a population. The state of the art results on epidemic modelling in complex networks can be found in Pastor-Satorras et al. [4].

Classic epidemic models generally assume the network is static during epidemic process while the population can be divided into different classes or compartments depending on the stage of the disease, such as susceptible (those who can contract the infection), infectious (those who contracted the infection and are contagious), recovered, removed or immune. The model defines the basic processes that govern the transition of individuals from one compartment to another. Each member of population can be a part of exactly one compartment at once.

Understanding the structure of the transmission network along with choosing the right epidemic model allows us to predict the distribution of infection and to simulate the full dynamics in order to control disease or plan immunization. In this thesis we will focus on SIR model for epidemic spreading and its modification, the ISS model for modelling rumour diffusion.

1.2. Finding patient zero

The inverse problem of estimating the initial epidemic conditions like localizing the source of an epidemic, commonly known as the patient zero problem, has only recently been formulated.

In the patient zero problem the source(s) of an epidemic or information diffusion propagation are determined based on limited knowledge of network structure or partial history of the propagation. The survey of methods for identifying the propagation source in networks can be found in Jiang et al. [5].

In the case of the SIR model there are three different approaches. Zhu and Ying [6] proposed a simple path counting approach and prove that the source node minimizes the maximum distance (Jordan centrality) to the infected nodes on infinite trees. Lokhov et al. [7] used a dynamic message-passing algorithm and estimate the probability that a given node produces the observed snapshot using a mean-field approch and an assumption of a tree-like contact network.

Antulov-Fantulin et al. [8] introduce analytical combinatoric, as well as Monte-Carlo based methods for source detection problem. These methods produce exact and approximate source probability distribution for any network topology based on a snapshot of the epidemic at known discrete time T. The provided benchmark results show

Monte-Carlo based MAP estimators outperforming previous results on a grid network for the SIR model. Additionally, these methods are applicable to many heterogeneous mixing models (SIR, IS, ISS) and are able to introduce uncertainty in the epidemic starting time, as well as uncertainty of temporal ordering of interactions. Even though the introduced Monte Carlo methods assume the epidemic started from a single source, one can also discriminate such hypothesis using Kolmogorov-Smirnov test [8].

1.3. Effects of network topology on epidemic spreading and detectability of patient zero

Complex networks show various levels of correlations in their topology which can have an impact on dynamical processes running on top of them. Real-world network of relevance for epidemic spreading are different from regular lattices. Networks are hierarchically organized with a few nodes that may act as hubs and where the vast majority of nodes have few direct connections. Although randomness in the connection process of nodes is always present, organizing principles and correlations in the connectivity patterns define network structures that are deeply affecting the evolution and behavior of epidemic and contagion process. These network's complex features often find their signature in statistical distributions which are generally heavy tailed and skewed.

Antulov-Fantulin et al. [8] have also introduced a metric for source detectability based on the entropy of estimated source probability distribution. The detectability of source node differs based on models parameters concerning the rate of disease spreading. Since topological properties of the network have profound impact on epidemic dynamic, the detectability of source node depending on its topological properties is an interesting analytical problem.

2. Complex network structure

Most of real networks in social and biological systems are characterized by the similar topological properties: small average path length, high clustering coefficients, fat tailed scale-free degree distributions, degree correlations and local network structure observable in the presence of communities.

2.1. Measures and metrics

Since larger networks can be difficult to envision and describe only by the graph G, we observe more detailed insights of the structure of these networks with various metrics.

Degree distribution

The degree distribution $P(k)$ defines the probability that a vertex in the networks interacts with exactly k other vertices. That is, $P(k)$ is the fraction of nodes with degree k under a degree distribution P.

The *scale-free* power-law degree distribution of the form $P(k) = Ak^{-\gamma}$ where $2 < \gamma < 3$ appears in a wide variety of complex networks. The networks with such property are referred to as *scale-free networks*. This feature is a consequence of two generic mechanisms: networks expand continuously by the addition of new vertices and new vertices attach preferentially to sites that are already well connected [9]. It is often said the scale-free distributions have "fat tails" since there tends to be many more nodes with very large degree compared to a Poisson degree distribution in a network with links formed completely independently.

Geodesic path

A path in a network is defined as an arbitrary sequence of vertices in which each pair of adjacent vertices is directly connected in the graph.

A geodesic path is the shortest path between two vertices. The small world network property observable in complex netowrk is considered to be present when average shortest path length is comparable to the logarithm of its network size.

Centrality

Centrality measures compare nodes and say something about how a given node relates to the overall network.

Degree centrality describes how connected a node is in terms of direct connections. For a vertex v in a network with n vertices it is defined as $\frac{deg(v)}{n-1}$. Since the degree centrality captures only centrality in terms of direct connections, it doesn't measure node's marginal contribution to the network when it has relatively few links but lies in a critical location in the network.

Closeness centrality describes how close a given vertex is to any other vertex. Let d_{ij} denote the length of geodesic path from vertex i to vertex j. For vertex v closeness centrality C_v is defined as harmonic mean between the distances of geodesic paths from vertex v to all others:

$$C_v = \frac{1}{n-1} \sum_{j \neq i} \frac{1}{d_{vj}}. \tag{2.1}$$

Betweenness centrality describes how well situated a vertex is in terms of the paths it lies on. Let σ_{st} be the number of geodesic paths between pairs of vertices v_s and v_t and let $\sigma_{st}(v_i)$ be the number of the geodesic paths σ_{st} which pass through the vertex v_i. The betweenness centrality is than defined as

$$C(v_i) = \sum_{st} \frac{\sigma_{st}(v_i)}{\sigma_{st}}. \tag{2.2}$$

Neighbours characteristics like eigenvector centrality measure how important, central or influential nodes neighbours are and capture a concept the vertex is more important if it has more important neighbours.

Let's define the adjacency matrix A of network G with N nodes as a matrix of size $N \times N$ that contains non-zero element A_{ij} if there exist an edge between vertices i and j. For an unweighed network all non-zero elements of A are equal to one. Note the adjacency matrix is symmetric for undirected graphs and generally asymmetric for directed graphs.

For the given vertex v, **eigenvector centrality** C_v [10] is proportional to the sum of centralities of its neighbours:

$$\lambda C_v = \sum_k A_{vk} C_k. \tag{2.3}$$

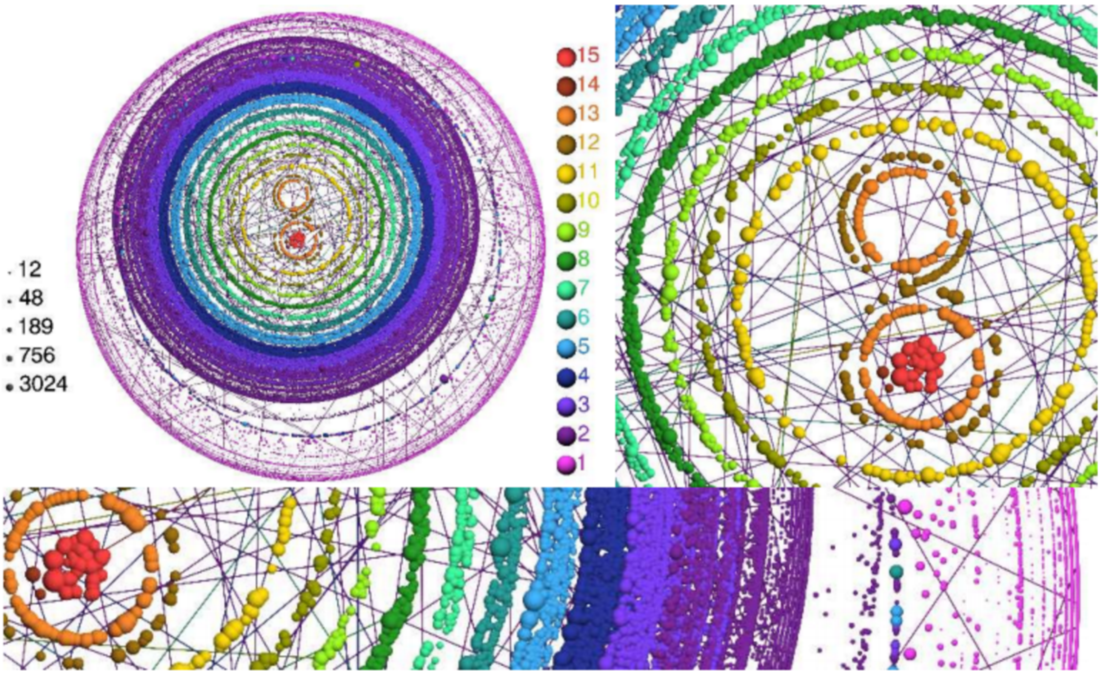

Figure 2.1: Graphical representation of a fraction of the .fr domain of Web, courtesy of Alvarez-Hamelin et al. [2]. The vertices of the same closeness are represented with the same color.

Consequently, C_v is eigenvector of adjacency matrix A corresponding to eigenvalue λ. The standard convention is to use the eigenvector associated with the largest eigenvalue for eigenvector centrality.

K-core

A k-core of undirected graph G is a maximal connected subgraph of G in which all vertices have degree at least k. The k-core is a measure of how sparse the graph is. The k-core can be obtained in $O(|L|)$ by iteratively removing all vertices of degree less than k from the graph.

A vertex u has coreness c if it belongs to a c-core but not to $(c+1)$-core. The k-core decomposition refers to a process of determining the coreness of each node and grouping the nodes according to their coreness. The concept of k-core (decomposition) was introduced to study the clustering structure of social networks and to describe the evolution of random graphs. K-core decomposition of complex networks reveals rich k-core architectures (Figure 2.1).

2.2. Modelling global network structure

2.2.1. Erdős Rényi graph model

Traditionally, networks of complex topology have been described with the random graph theory of Erdős and Rényi [11], but in the absence of data on large networks, the predictions of the ER theory were rarely tested in the real world.

This random graph model assumes we start with N vertices and connect each pair of vertices with probability p. The formation is independant across links so the probability of generating a network with exactly m links is equal to $p^m(1-p)^{\frac{N(N-1)}{2}-m}$ and the expected number of links (or the average degree) is $\langle d \rangle = pN(N-1)/2$.

The degree distribution of the generated random network is

$$P(d) = \binom{N-1}{d} p^d (1-p)^{n-1-d}. \tag{2.4}$$

For large n, the degree distribution follows a Poisson distribution $P(d) = e^{-\lambda}\lambda^d/d!$, where

$$\lambda = N\binom{N-1}{d} p^d (1-p)^{N-1-d}.$$

In Erdős-Rényi model, maximum coreness is related to the average degree $\langle d \rangle$. Since the topology is very homogeneus, it is also expected most vertices will belong to the same k-core that is also the highest.

While random network can observe features like diameters small relative to the network size, they lack certain features that are prevalent among complex networks, such as the high clustering and presence of communities.

2.2.2. Barabási-Albert graph model

Barabási-Albert model is the model of evolving a scale-free network, which uses a preferential attachment property and thus creating a heterogeneous topology. The preferential attachment mechanism is one of the generating mechanisms of scale-free networks [9] and refers to building the network gradually where each new vertex tends to connect with the old vertices that are already well connected within the old network.

The Barabási-Albert graph is generated starting from m_0 isolated vertices. At each time step new vertices with m edges are added to the network $m < m_0$. The new vertex will create an edge to the existing node v_i with the probability proportional to its degree k_i.

The Barabási-Albert graph model produces power law distribution $P(k) \approx k^{-3}$ in the limit of time. The average geodesic path increases logarithmically with the size of the network.

By repetitively connecting each new node to the previous graph with exactly m edges, we obtain a graph where any subgraph has a vertex of degree at most m and the k-core of the graph is m.

3. Epidemic process modelling

In the focus of this thesis are heterogeneous epidemic models on the contact network formed by connections between single contacting individuals with transitions of individuals between compartments happening in discrete time steps.

3.1. SIR model

Wide range of diseases that provide immunity to the host can be successfully modelled on a network whose members take one of three possible roles at a time: susceptible (S), infected (I) or recovered (R). The diffusion of disease takes place between infected nodes and their susceptible neighbours. An infectious node may also recover from the disease. The recovery grants permanent immunity effectively erasing the member from the contacting network. The possible events can be represented as

$$S + I \xrightarrow{p} 2I, \quad I \xrightarrow{q} R. \tag{3.1}$$

In the SIR model, the infection and recovery process completely determine the epidemic evolution. The transitions (3.1) occur spontaneously and independently in each time step. In discrete-time formulation an infected individual when meeting susceptible will infect the neighbouring susceptible with probability p at each time step. The recovery probability q is the probability the infected individual will recover for each time step. The transition probabilities p and q are often assumed constant and equal for all nodes in the same epidemic process.

3.1.1. Simulating the discrete SIR epidemic

For the contacting network represented by $G(N, L)$ and SIR parameters p and q, we are able to simulate one time step of discrete SIR process. Let s_t, i_t and r_t denote sets of nodes that are respectively susceptible, infected and recovered after time step t. At time step t all previously infected nodes i_{t-1} will try to infected their susceptible neighbours independently of each other and at the same time. Afterwards the passive recovery process will try to turn them to recovering nodes, each with probability q.

This process can be simulated with NaiveSIR algorithm [13] by putting all the initially infected nodes in the queue. While traversing the nodes, we try to infect each neighbouring node. When the new node gets infected, it gets pushed to the queue. SIR simulation of one time step t is described by algorithm 1.

For the algorithm complexity analysis standard big-O notation is used (asymptotic upper bound within a constant factor) [14]. The average case running time of the NaiveSIR algorithm is equal to $O(\frac{E[X]\langle d\rangle}{q})$ where $E[X]$ denotes total expected number of infected nodes and $\langle d\rangle$ denotes the average node degree [13]. The space complexity of NaiveSIR algorithm with respect to the number of links L is equal to $O(L)$ since the memory holds a contact network G in a form of adjacency list ($O(L)$), queue of infected nodes Iq ($O(N)$) and indicators of each compartment that are best implemented as a bitset data structure ($O(1)$).

Algorithm 1: One time step NaiveSIR simulation on graph **G**.

```
Data: G - network, $(p, q)$ -parameters of the SIR model, $Iq$ - queue of infected
      nodes, $I$ - bitset of infected nodes, $S$ - bitset of susceptible nodes, $R$ -
      bitset of recovered nodes
infected_size = size($Iq$)
for $k = 1$ to infected_size do
    if $Iq$ is empty then
        break
    dequeue(u, Iq)
    foreach $v \in nei(u)$) do
        if $v \in S$ then
            let transmission $u \rightarrow v$ occur with probability $p$
            if $u \rightarrow v$ occured then
                update($I(v)$ and $S(v)$)
    let transmission $u \rightarrow v$ occur with probability $q$;
    if $u \rightarrow v$ occured then
        update($I(u)$ and $R(u)$)
    else
        enqueue(u, Iq)
return {S, I, R}
```

3.1.2. Probability of one time step transition

Probability of one time step transition can be easily evaluated. Let $nei(v)$ indicate a set of all neighbours of node v, $nei(V)$ a set of all neighbours of all nodes in set V and $nei_V(v) = nei(v) \cap V$, a set of all neighbours of v that are also in V. After k-th time step of the SIR process, the resulting i_k and r_k were given. At time step k, only initially active nodes i_{k-1} and their neighbours $nei(i_{k-1})$ actively participate in the epidemic process. For each node v in $i_{k-1} \cup nei(i_{k-1})$, one of four independent events may have happened during time step k and they are easily detectable based on i_{k-1}, r_{k-1}, i_k and r_k:

- E_1 : **if** $v \notin i_{k-1}$ and $v \in i_k$
  node v was infected with probability $1 - (1-p)^{nei_{i_{k-1}}(v)}$
- E_2 : **if** $v \notin i_{k-1}$ and $v \notin i_k$
  node v was not infected with probability $(1-p)^{nei_{i_{k-1}}(v)}$
- E_3 : **if** $v \in i_{k-1}$ and $v \in r_k$
  node v was recovered with probability q
- E_4 : **if** $v \in i_{k-1}$ and $v \notin r_k$
  node v was not recovered with probability $1 - q$

Since all the events $E_1 - E_4$ are independent and the sets of nodes corresponding to each event are disjoint while completely covering the set of active nodes $i_{k-1} \cup nei(i_{k-1})$, the conditional probability of one time step SIR transition $P(i_k, r_k | i_{k-1}, r_{k-1})$ can be calculated as

$$\begin{aligned} P(i_k, r_k | i_{k-1}, r_{k-1}) = & \left[\Pi_{v \in E_1} (1 - (1-p)^{nei_{i_{k-1}}(v)})\right] \left[\Pi_{v \in E_2} (1-p)^{nei_{i_{k-1}}(v)}\right] \\ & \cdot \left[\Pi_{v \in E_3} q\right] \left[\Pi_{v \in E_4} (1-q)\right]. \end{aligned} \tag{3.2}$$

$nei_{i_{k-1}}(v)$ denotes the set of all neighbours of v that were also infected at the beginning of time step k (set $nei(v) \cap i_{k-1}$).

3.2. Epidemic models as social contagion processes

Even though infectious diseases represent the central focus of epidemic modelling, the model where an individual is strongly influenced by the interaction with its peers is present in several other domains, especially in social context in the diffusion of information, the propagation of rumour and adoption of innovation or behaviours. Since the social contacts can in these domains generate epidemic-like outbreaks, simple models for information diffusion are epidemic models modified to specific features of social contagion. The crucial difference to pathogen spreading is that transmission of information involves intentional acts by both the sender and the receiver and it is often beneficial for both participants.

3.2.1. Rumour spreading with ISS model

The need to study rumour spreading presents itself in a number of important technological and commercial applications where it is desirable to spread the "epidemic" as fast and as efficient as possible. In examples such as rumour based protocols for resource discovery and marketing campaigns that use rumour like strategies (viral marketing) the problem translates to design of an epidemic algorithm in such a way that the given information reaches as much nodes as possible, similarly to a rumour.

Models for rumour spreading are variants of the SIR model in which the recovery process does not occur spontaneously, but rather is a consequence of interactions. The modification mimics the idea it is worth spreading the rumour as long as it is novel for the recipient. This process can be formalized as a model where each of N members of the contacting network can be a part of one of three compartments: **ignorant (S), spreader (I) and stifler (R)**. Ignorants have not heard the rumour and are susceptible to being informed. Spreaders are actively spreading the rumour, while stiflers know about the rumour but they're not spreading it.

The spreading process evolves by direct contacts of spreaders with others in the population. When a spreader meets an ignorant, the latter turns into a new spreader with probability a. When a spreader meets another spreader or a stifler, the former spreader turns into stifler with probability b and the latter remains unchanged. This model is known as the ISS model (Ignorant-Spreader-Stifler) [15]. The possible events can be represented as

$$S + I \overset{\alpha}{\to} 2I, \quad R + I \overset{\beta}{\to} 2R, \quad 2I \overset{\beta}{\to} R + I. \tag{3.3}$$

Since we are examining the spreading process in discrete time, at each time step the current spreaders try to interact with their neighbours. A modification of the NaiveSIR algorithm for rumour spreading simulation of one time step t is described by algorithm 2.

Algorithm 2: One time step of ISS simulation with modified NaiveSIR algorithm on graph **G**.

```
Data: G - network, $(a, b)$ - parameters of the ISS model, $Iq$ - priority queue of
      spreader nodes, $I$ - bitset of spreader nodes, $S$ - bitset of ignorant nodes,
      $R$ - bitset of stifler nodes
stifler_size = size($Iq$)
for $k = 1$ to stifler_size do
    if $Iq$ is empty then
        break
    dequeue(u, Iq)
    foreach $v \in nei(u)$ do
        if $v \in S$ then
            let transmission $u \to v$ occur with probability $a$
            if $u \to v$ occured then
                update($I(v)$ and $S(v)$)
        else
            let transmission $u \to v$ occur with probability $b$
            if $u \to v$ occured then
                update($I(u)$ and $R(u)$)
    if $u \in$ I then
        enqueue(u, Iq)
return {S, I, R}
```

4. Patient zero – single source epidemic detection

In accordance with [8], we will focus on a patient zero problem given snapshot of population at time T and complete knowledge of underlying contacting network modelled by G with the assumption the epidemic has started from a single source node and that it is governed by the SIR process with known p and q. The estimators proposed in [8] will be presented in this chapter, while the newly proposed estimators based on importance sampling technique will be presented in the next chapter.

Let random vector $\vec{S} = (S(1), \ldots, S(N))$ indicate the nodes that got infected up to a predefined temporal threshold T with SIR(p, q) epidemic process on network G with N nodes. $S(i)$ is a Bernoulli random variable with the value 1 if the node i got infected before time T from the start of the epidemic process. We observe one realization $\vec{s}_*$ of $\vec{S}$ (the snapshot at time T). The finite set of possible source nodes Θ is determined by realization $\vec{s}_*$. We want to infer which nodes from the set of infected or recovered nodes $\Theta = \{\theta_1, \theta_2, \ldots, \theta_m\}$ are most likely to be the source of the epidemic process.

A maximum aposteriori probability estimate (MAP) is the node with the highest probability for being the source of the epidemic spread for given target realization $\vec{s}_*$:

$$\hat{\theta}_{MAP} = \text{arg max}_{\theta_i \in \Theta} P(\Theta = \theta_i | \vec{S} = \vec{s}_*) \tag{4.1}$$

By applying the Bayes theorem with equal apriori probabilities $P(\Theta = \theta_i)$, probability in (4.1) can be expressed as

$$\begin{aligned} P(\Theta = \theta_i | \vec{S} = \vec{s}_*) &= \frac{P(\vec{S} = \vec{s}_* | \Theta = \theta_i) P(\Theta = \theta_i)}{\sum_{\theta_k \in \Theta} P(\vec{S} = \vec{s}_* | \Theta = \theta_k) P(\Theta = \theta_k)} \\ &= \frac{P(\vec{S} = \vec{s}_* | \Theta = \theta_i)}{\sum_{\theta_k \in \Theta} P(\vec{S} = \vec{s}_* | \Theta = \theta_k)}. \end{aligned} \tag{4.2}$$

4.1. Direct Monte Carlo estimator

The integration problem

$$\mathbf{E_f}[h(X)] = \int_X h(x)f(x)dx \tag{4.3}$$

can be estimated using Monte Carlo technique with n samples $X_1, \ldots, X_n$ generated from the density f as the empirical average

$$h_n = \frac{1}{n}\sum_{j=1}^{n} h(X_j). \tag{4.4}$$

The convergence of h_n towards $\mathbf{E_f}[h(X)]$ is assured by the Strong Law of Large Numbers.

Inferring the probability $P(\vec{S} = \vec{s}_*|\Theta = \theta_i)$ up to multiplicative constant is an integration problem equivalent to expectation of Kronecker delta function $\delta(\vec{S}) = 1\{\vec{S} = \vec{s}_*\}$ where $\vec{S}$ is a random variable governed by probability distribution $P(\vec{S}|\Theta = \theta_i)$. Let m_i denote estimation of expected number of hits for a fixed source θ_i estimated using Monte Carlo technique:

$$m_i = \sum_{j=1}^{n} 1\{\vec{S}_i = s_*\} \tag{4.5}$$

where $\vec{S}_i$ are drawn from $P(\vec{S}|\Theta = \theta_i)$.

The estimate m_i is obtained using Direct Monte Carlo estimator by simulating epidemic process up to time T starting from a single infected node θ_i and checking whether the generated realization $\vec{S}_i$ coincides with $\vec{s}_*$. Since m_i is estimation of $P(\vec{S} = \vec{s}_*|\Theta = \theta_i)$ up to multiplicative constant $1/n$ for all $\theta_i \in \Theta$, we derive Direct Monte Carlo MAP estimator based on the estimation of probability $P(\Theta = \theta_i|\vec{S} = \vec{s}_*)$ combining (4.5) with (4.2):

$$\hat{P_i}^n = \hat{P}(\Theta = \theta_i|\vec{S} = \vec{s}_*) = \frac{m_i}{m} \tag{4.6}$$

where $m = \sum_{j=1}^{n} m_j$.

If the size of realization $\vec{s}_*$ is big, the number of simulations required to obtain reliable estimations can be large. Since the estimations for different source node candidates are independent, the computations can be parallelised.

Additionally for the SIR model, a prunning mechanism can be incorporated. If a sampling simulation infects a node that was not infected during the target epidemic represented by the realization s_*, it is safe to stop the sampling simulation prior to ending time T and call the partial sample unequal to target realization s_*.

The accuracy of direct Monte Carlo estimation is controlled by convergence conditions. Upon estimating two source PDF's $\hat{P}_i^n$ and $\hat{P}_i^{2n}$ with n and $2n$ independent simulations respectively, the distribution estimation is said to converge when the following conditions are satisfied:

$$|\hat{P}_i^{2n} - \hat{P}_i^n|/\hat{P}_{2n} \leq c,\ |\hat{P}_i^{2n} - \hat{P}_i^n| \leq c \quad \forall \theta_i \in \Theta. \tag{4.7}$$

Algorithm 3: Direct Monte Carlo estimation of expected number of realizations completely corresponding to $\vec{s}_*$ after T time steps for a fixed source θ_i.

```
Data: G - network, $(p, q)$ - parameters of the SIR process, $\vec{s}_*$ - target
      realization, $T$ - temporal threshold, $\theta_i$ - proposed source node, $n$ - number
      of simulations
$m_i = 0$
for $d = 1$ to $n$ do
    for $t = 1$ to $T$ do
        Run one SIR simulation $(p, q, \theta_i)$ for time step $t$ and obtain $\vec{S}_t^{(d)}$
        if $\exists j \in N : (S_t(j) == 1$ and $s_*(j) == 0)$ then
            break
    if $\vec{S}_T^{(d)}$ equals $\vec{s}_*$ then
        $m_i = m_i + 1$
return $m_i$
```

4.2. Soft Margin estimator

Let $\vec{S}_\theta^{(j)}$ denote j-th sample (outcome) obtained by Monte Carlo simulation of contagion process with source node θ and duration of T time steps. $\vec{S}_\theta^{(j)}$ is one realization of random binary vector $\vec{S}_\theta$ that describes the outcome of an epidemic process. A similarity measure $\varphi : (\vec{S}_\theta \times \vec{S}_\theta) \to [0, 1]$ can be defined between any two realizations of $\vec{S}_\theta$. For example, φ can be defined as the Jaccard similarity function:

$$\varphi(\vec{s}_1, \vec{s}_2) = \frac{\vec{s}_1 \cap \vec{s}_2}{\vec{s}_1 \cup \vec{s}_2} = \frac{\sum_{j=1}^{N}(s_1(j) = 1 \text{ and } s_2(j) = 1)}{\sum_{j=1}^{N}(s_1(j) = 1 \text{ or } s_2(j) = 1)}. \tag{4.8}$$

Moreover, we can define a discrete random variable $\varphi(\vec{s}_*, \vec{S}_\theta)$ that measures the similarity between fixed realization $\vec{s}_*$ and random realization from $\vec{S}_\theta$. Let PDF of that random variable be $f_\theta(x)$ where $x = \varphi(\vec{s}_*, \vec{S}_\theta)$. Unbiased estimator for the PDF can be obtained with Monte Carlo method from n samples as

$$f_\theta(x) = \int_0^1 p_k \delta(x - x_k) dx \approx \frac{1}{n} \sum_{i=1}^{n} \delta(x - \varphi(\vec{s}_*, \vec{S}_\theta^{(i)})) \tag{4.9}$$

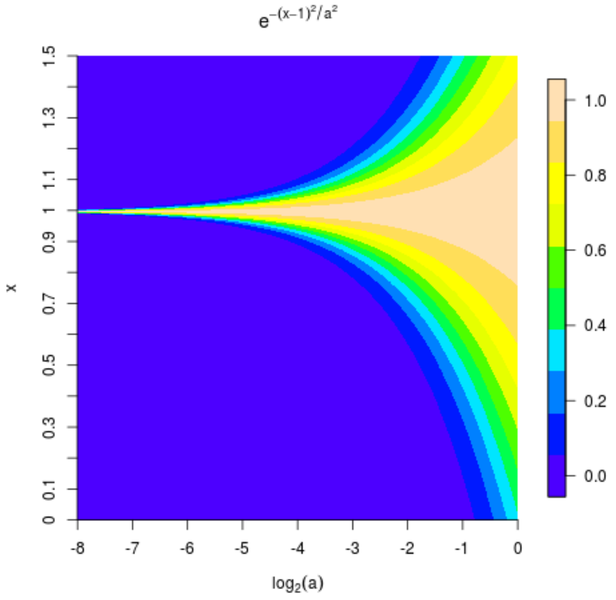

Figure 4.1: Contour plot of Gaussian weighting function $w_a(x) = e^{-(x-1)^2/a^2}$.

where $\delta(x)$ denotes the Dirac delta function. In the integral definition we observe a series of probabilities $p_1, p_2, \ldots, p_d$ corresponding to each realization of discrete random variable $\varphi(\vec{s}_*, \vec{S}_\theta)$. With Monte Carlo method, we take the PDF definition as an integration problem (4.3) and sample from this discrete distribution $\{p_1, \ldots, p_d\}$ to obtain the PDF estimate.

The Soft Margin estimator is defined as

$$\hat{P}_a(\vec{S} = \vec{s}_* | \Theta = \theta) = \int_0^1 w_a(x) \hat{f}_\theta(x) dx \tag{4.10}$$

where $w_a(x)$ is a weighting function and $f_\theta(x)$ is the PDF function of the random variable $\varphi(\vec{s}_*, \vec{S}_\theta)$. For $w_a(x)$ Antulov-Fantulin et al. [8] proposed a Gaussian weighting form $w_a(x) = e^{-(x-1)^2/a^2}$. In this way, the problem definition is altered to estimating the number of realizations with similarity to s_* in the interval around $\varphi = 1$ defined by Gaussian function $w_a(x)$ (Figure 4.1), as opposed to estimating the number of realizations with similarity strictly equal to $\varphi = 1$ with Direct Monte Carlo method. In the limit where $a \to 0$, unbiased direct Monte Carlo estimate is obtained.

The Soft Margin formula (4.10) can be further simplified by combining with (4.9):

$$\begin{aligned} \hat{P}_a(\vec{S} = \vec{s}_* | \Theta = \theta) &= \int_0^1 w_a(x) \hat{f}_\theta(x) dx \\ &= \int_0^1 w_a(x) \frac{1}{n} \sum_{i=1}^{n} \delta(x - \varphi(\vec{s}_*, \vec{S}_\theta^{(i)})) dx, \end{aligned} \tag{4.11}$$

and further by using the property of Dirac delta function $\int_{-\infty}^{\infty} f(x)\delta(x-b)dx = f(b)$:

$$\begin{aligned}\hat{P}_a(\vec{S} = \vec{s}_*|\Theta = \theta) &= \frac{1}{n}\sum_{i=1}^{n}\int_0^1 w_a(x)\delta(x - \varphi(\vec{s}_*, \vec{S}_\theta^{(i)}))dx \\ &= \frac{1}{n}\sum_{i=1}^{n} w_a(\varphi(\vec{s}_*, \vec{S}_\theta^{(i)})) \\ &= \frac{1}{n}\sum_{i=1}^{n} e^{\frac{(\varphi_i-1)^2}{a^2}}.\end{aligned} \tag{4.12}$$

Note that it's not needed to determine constant a in advance. The parameter a can be chosen as the infinum of the set of parameters for which the source probability distribution estimates $\hat{P}_a(\Theta = \theta_i|\vec{S} = \vec{s}_*)$ have converged under the convergence property (4.7).

Additionally, for a fixed number of simulations n, PDF's based on different parameters a can be estimated with one set of samples.

Algorithm 4: Soft Margin approximation of $P(\vec{S} = \vec{s}_*|\Theta = \theta_i)$ for a fixed source θ_i.

```
Data: G - network, $(p, q)$ - parameters of the SIR process, $\vec{s}_*$ - target realization,
      $T$ - temporal threshold, $\theta_i$ - proposed source node, $n$ - number of
      simulations, $a$ - Soft Margin parameter
for $i = 1$ to $n$ do
    Run SIR simulation $(p, q, \theta_i)$ for $T$ time steps and obtain $\vec{S}_T^{(i)}$
    Calculate and save $\varphi_i = \varphi(\vec{s}_*, \vec{S}_T^{(i)})$
Calculate $\hat{P}(\vec{S} = \vec{s}_*|\Theta = \theta_i) = \frac{1}{n}\sum_{i=1}^{n} e^{\frac{-(\varphi_i-1)^2}{a^2}}$
return $\hat{P}(\vec{S} = \vec{s}_*|\Theta = \theta_i)$
```

4.3. Time complexity of Direct Monte Carlo and Soft Margin estimators

The average run time complexity $\overline{RT}$ of source detection Monte Carlo estimators (Direct Monte Carlo and Soft Margin) is $\overline{RT} \propto m \times n \propto \overline{RT}_M$, where m denotes the number of potential sources in the observed realization, n the number of samples of the random variable $\vec{S}_\theta$ and $\overline{RT}_M$ denotes the average run-time complexity of sampling one realization from contagion process M [8].

Note that in the worst-case scenario the number of potential sources is proportional to the network size, but in reality we are mostly interested in source detection problems in which the number of potential sources is much smaller than the network size.

Additionally, different Monte Carlo estimators have different convergence properties with respect to the number of samples n. With $c = 0.05$ and convergence conditions from (4.7), the Soft Margin estimator converges for $n \in [10^4, 10^6]$ on the benchmark grid samples on which the Direct Monte Carlo needs $n \in [10^6, 10^8]$ simulations to converge.

5. Importance sampling based single source detection

5.1. Importance sampling

Importance sampling is a technique for estimating properties of a particular distribution with samples generated from a different distribution than the one of interest. The technique is used with Monte Carlo method as a variance reduction technique since we usually choose to sample from the distribution that is biased towards the realizations that have more impact on the parameters being estimated.

In other words, the method of importance sampling is estimation of integration problem (4.3) based on generating a sample $X_1, \ldots, X_m$ from a given biased distribution g when in fact the samples X_i come from the target distribution f:

$$\mathbf{E_f}[h(X)] = \int_X h(x)f(x)dx = \int_X h(x)\frac{f(x)}{g(x)}g(x)dx \approx \frac{1}{m}\sum_{j=1}^{m}\frac{f(X_j)}{g(X_j)}h(X_j). \quad (5.1)$$

By choosing to sample from the biased distribution g, we are left with the extra weight $w^{(i)} = \frac{f(X_j)}{g(X_j)}$ from the integral that corrects the bias of the estimation. The new estimator converges whatever the choice of distribution g, as long as $supp(g) \supset supp(f)$[1].

Note the estimation can be done with unbiased estimate,

$$\frac{1}{m}\sum_{i=1}^{m} w^{(i)}h(\mathbf{x}^{(i)}), \quad (5.2)$$

or with a weighted estimate

$$\frac{\sum_{i=1}^{m} w^{(i)}h(\mathbf{x}^{(i)})}{\sum_{i=1}^{m} w^{(i)}}. \quad (5.3)$$

When using the weighted estimate, we only need to know the ratio $f(\mathbf{x})/g(\mathbf{x})$ up to a multiplicative constant. Although inducing a small bias, the weighted estimate often has a smaller mean squared error than the unbiased one.

[1] $supp(g) = \{x|g(x) \neq 0\}$

5.1.1. Measuring the quality of importance distribution

By properly choosing $g(\cdot)$, one can reduce the variance of the estimate substantially. In order to make the estimation error small, one wants to choose $g(\mathbf{x})$ as close in shape to $f(\mathbf{x})h(\mathbf{x})$ as possible. The efficiency of such method is difficult to measure.

Effective sample size (ESS) is commonly used to measure how different the importance distribution is from the target distribution. Suppose we have m independent samples generated from $g(\mathbf{x})$. The ESS of this method is defined as

$$\text{ESS}(m) = \frac{m}{1 + var_g[w(\mathbf{x})]}. \tag{5.4}$$

The variance here is estimated as a square of the coefficient of variation of the weights:

$$cv^2 = \frac{\sum_{j=1}^{m}(w^{(j)} - \bar{w})^2}{(m-1)\bar{w}^2} \tag{5.5}$$

where $\bar{w}$ is sample average of the $w^{(j)}$. The ESS measure of efficiency can be partially justified by the delta method [16].

5.1.2. Rejection control and weighting

When applying importance sampling, one often produces random samples with very small importance weights because of a less than ideal importance density. The following technique for combining rejection and importance weighting can be used.

Suppose we have drawn samples $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$ from $g(\mathbf{x})$. Let $w^{(j)} = \frac{f(\mathbf{x}^{(j)})}{g(\mathbf{x}^{(j)})}$. We can conduct the following operation for any given threshold value $c > 0$:

Algorithm 5: Rejection Control (RC)

```
for $j = 1, \ldots, m,$ do
    accept $\mathbf{x}^{(j)}$ with probabilty $r^{(j)} = \min\left\{1, \frac{w^{(j)}}{c}\right\}$
    if $\mathbf{x}^{(j)}$ is accepted then
        weight $w^{(j)}$ is updated to $w^{(*j)} = q_c w^{(j)}/r^{(j)}$, where
            $$q_c = \int \min\left\{1, \frac{w^{(j)}}{c}\right\} g(\mathbf{x}) d\mathbf{x} \tag{5.6}$$
```

Since the constant q_c is the same for all accepted samples, it is not needed for the evaluation of the weighted estimate in (5.3). Nevertheless, it can be unbiasedly estimated [16] from the sample as

$$\hat{p}_c = \frac{1}{m} \sum_{j=1}^{m} \min \Big\{ 1, \frac{w^{(j)}}{c} \Big\}. \tag{5.7}$$

With this technique we are adjusting the importance density g in light of current importance weights. The new importance density $g^*(\mathbf{x})$ is expected to be close to the target distribution $f(\mathbf{x})$.

After applying rejection control, we will typically have fewer than N samples. More samples can be drawn from either $g(x)$ or $g^*(x)$ (via rejection control) to make up for the rejected samples.

5.2. Sequential importance sampling

Since it is not trivial to design a good importance sampling distribution, especially for high dimensional problems, one may build up the importance density sequentially. Suppose we can decompose $\mathbf{x}$ as $\mathbf{x} = (x_1, \ldots, x_d)$ where each of the x_j may be multidimensional. Then our importance distribution can be constructed as

$$g(\mathbf{x}) = g_1(x_1) g_2(x_2|x_1) g_3(x_3|x_1, x_2) \ldots g_d(x_d|x_1, \ldots, x_{d-1}) \tag{5.8}$$

by which we hope to obtain some guidance from the target density while building up the importance density. We can then rewrite the target density as

$$f(\mathbf{x}) = f_1(x_1) f_2(x_2|x_1) f_3(x_3|x_1, x_2) \ldots f_d(x_d|x_1, \ldots, x_{d-1}) \tag{5.9}$$

and the weights as

$$w(\mathbf{x}) = \frac{f_1(x_1) f_2(x_2|x_1) f_3(x_3|x_1, x_2) \ldots f_d(x_d|x_1, \ldots, x_{d-1})}{g_1(x_1) g_2(x_2|x_1) g_3(x_3|x_1, x_2) \ldots g_d(x_d|x_1, \ldots, x_{d-1})} \tag{5.10}$$

which suggests a recursive monitoring and computing of importance weight:

$$w_t(\mathbf{x}_t) = w_{t-1}(\mathbf{x}_{t-1}) \frac{f(x_t|\mathbf{x}_{t-1})}{g(x_t|\mathbf{x}_{t-1})}. \tag{5.11}$$

At the end, w_d is equal to $w(\mathbf{x})$. By using the recursive process we can stop generating further components of $\mathbf{x}$ if the partial weight derived from the sequentially generated partial sample is too small and we can take advantage of $f(x_t|\mathbf{x}_{t-1})$ in designing $g_t(x_t|\mathbf{x}_{t-1})$.

The sequential importance sampling method can then be defined as follows:

Algorithm 6: SIS Step

```
1. Draw $X_t$ from $g_t(x_t|\mathbf{x_{t-1}})$, and let $\mathbf{x}_t = (\mathbf{x}_{t-1}, x_t)$.
2. Compute $w_t(\mathbf{x}_t) = w_{t-1}(\mathbf{x}_{t-1})\frac{f(x_t|\mathbf{x}_{t-1})}{g(x_t|\mathbf{x}_{t-1})}$.
```

When we observe that w_t is getting too small, we can choose to reject the sample halfway and restart again.

5.2.1. Improving the SIS procedure with resampling

When the system grows, the variance of the importance weights w_t increases. After a certain number of steps, many of the weights become very small and a few very large. In that situation one may use a resampling strategy.

Suppose at step t we have a collection of m partial samples of length $t, S_t = \{\mathbf{x}_t^{(j)}, j = 1, \ldots, m\}$ which are properly weighted by the collection of weights $W_t = \{w_t^{(j)}, j = 1, \ldots, m\}$ with respect to the density g.

The resampling step is done on the existing partial sample set before expanding with the SIS step.

Simple random sampling

- Sample a new set of partial samples, S_t' from S_t according to the weights $w_t^{(j)}$.
- Assign equal weights, W_t/m, to the samples in S_t' where $W_t = w_t^{(1)} + \ldots + w_t^{(m)}$

Residual resampling

- Retain $k_j = [mw_t^{(*j)}]$ copies of $\mathbf{x}_t^{(j)}$ where $w_t^{(*j)} = w_t^{(j)}/W_t$ and $j = 1, \ldots, m$. Let $m_r = m - k_1 - k_2 - \ldots - k_m$.
- Obtain m_r draws from S_t with probabilities proportional to $mw_t^{(*j)} - k_j$, $j = 1, \ldots m$.
- Reset all the weights to W_t/m.

Residual sampling dominates the simple random sampling in having smaller Monte Carlo variance.

Resampling schedule

The resampling step tends to result in a better group of anecestors so as to produce better descendants. The success of resampling, however, relies heavily on the Markovian structure among the state variables $x_1, x_2, \ldots$. Given the realization of x_t, the next variable x_{t+1} is statistically independent of all the previous states $\mathbf{x}_{t-1}$. If the resampling from set $\{\mathbf{x}_{t-1}^{(j)}, j = 1, \ldots m\}$ is not equivalent to resampling from $\{x_{t-1}^{(j)}, j = 1, \ldots, m\}$, the set of the "current state" frequent resampling will rapidly impoverish diversity of the partial samples produced earlier. When no simple Markovian structure is present, frequent resampling generally gives bad results.

For this reason, it is desirable to prescribe a schedule for the resampling to take place. The resampling schedule can be either deterministic or dynamic. When the schedule is dynamic, some small bias may be introduced.

With a deterministic schedule, we conduct resampling at time $t_0, 2t_0, \ldots,$ where t_0 is given in advance. In a dynamic schedule, a sequence of thresholds $c_1, c_2, \ldots,$ are given in advance. We monitor the coefficient of variation of the weights cv_t^2 and invoke the resampling step when event $cv_t^2 > c_t$ occurs. A typical sequence of c_t can be $c_t = a + bt^\alpha$.

Increasing c_t after each SIS step makes sense since it can be shown that as the system evolves, cv_t^2 increases stochastically [17].

Resampling scheme

- Check the weight distribution by performing one of the methods at time t. Resample if needed.
- Invoke an SIS step. Set $t = t + 1$.

5.3. Sequential Importance sampling source estimator

Given snapshot s_* that holds all infected nodes up to time T, we want to determine the probability of an epidemic starting in node θ_i, $P(\theta_i|\vec{S} = \vec{s}_*)$, where $\vec{S}$ is a random variable whose one realization is $\vec{s}_*$. Since all the apriori probabilities $P(\theta_i)$ are the same, we can approximate aposteriori probabilities $P(\vec{S} = \vec{s}_*|\theta_i)$ and use them to determine $P(\theta_i|\vec{S} = \vec{s}_*)$, as we did in 4.2. These aposteriori probabilities were estimated with Direct Monte Carlo and Soft Margin method up to a multiplicative constant. This can also be done using Sequential Importance Sampling technique.

First note the SIS step as defined in Algorithm 2 is based on the densities of a complete history of the process, or at time t, all the process steps up to time t. The target density is thus the join probability of all the steps taken in the process. Since we are only interested in the final realization, it makes sense to use target and importance probability distributions of the form

$$f(s_t) = f_1(i_1, r_1) f_2(i_2, r_2 | i_1, r_1) f_3(i_3, r_3 | i_2, r_2) \dots f_t(i_t, r_t | i_{t-1}, r_{t-1}) \tag{5.12}$$

$$g(s_t) = g_1(i_1, r_1) g_2(i_2, r_2 | i_1, r_1) g_3(i_3, r_3 | i_2, r_2) \dots g_t(i_t, r_t | i_{t-1}, r_{t-1}) \tag{5.13}$$

where i_t denotes a vector of infected nodes after time step t, and r_t denotes a vector of recovered nodes after time step t. Note that $i_t \cup r_t = s_t$ and $i_t \cap r_t = \emptyset$. Each adjacent element of the sequence $(i_1, r_1), (i_2, r_2), (i_3, r_3), \dots, (i_t, r_t)$ is connected with one SIR step.

5.3.1. Modelling the target distribution

We can evaluate the partial target density $f_k(i_k, r_k | i_{k-1}, r_{k-1})$ in closed form. This is exactly the probability of one time step SIR transition given with Formula 3.2.

5.3.2. Modelling the importance distribution

With our sequential sampling procedure we will try to estimate the number of realizations at time T that are equal to s_* for some fixed starting node θ_i. The importance density will be biased towards that goal. Since we are building the final densities sequentially, our biased sampling must sample reasonably enough at each step (it must not be to "slow" or too "fast"), especially since it is not certain what samples at mid steps are valuable too us as we might perform some sort of resampling or reduction procedure.

It is certain, however, we do not want to infect the nodes that were never infected in the snapshot s_* and we can safely use $SIR(p = 1, q)$ at the last SIS step. That leads us to the biased density similar to the one in 3.2 where only the nodes in s_* are eligible for events E_1 and E_2 and it holds $p = 1$ when $k = T$.

It may be reasonable to increase p at each step of SIS procedure but it is not clear when this should be done. Additionally, one might want to use a resampling or a rejection technique based on vc^2 for simulations with many SIS steps. This has to be done carefully too since our target event is rare and weights w are naturally small.

5.3.3. Building the algorithm

5.4. Soft Margin SIS estimator

5.5. Configurational bias Monte Carlo estimator

5.5.1. Metropolis-Hastings algorithm

5.5.2. Metropolized independence sampler

A very special choice of the proposal transition function $T(\mathbf{x}, \mathbf{y}$ is an independent trial density $g(\mathbf{y})$ in which the proposed sample $\mathbf{y}$ is generated from $g()$ independent of the previous state $\mathbf{x}$. This method is an alternative to rejection sampling and importance sampling strategies for Monte Carlo.

MIS scheme

Given the current state $\mathbf{x}^{(t)}$

- Draw $\mathbf{y} \sim g(\mathbf{y})$
- Simulate $u \sim$ Uniform$[0, 1$ and let

$$\mathbf{x}^{(t+1)} = \begin{cases} \mathbf{y} \quad \text{if} \quad u \leq \text{min}\Big\{1, \frac{w(\mathbf{y})}{\mathbf{x}^{(t)}}\Big\} \\ \mathbf{x}^{(t)} \quad \text{otherwise,} \end{cases}$$

where $w(\mathbf{x}) = f(\mathbf{x})/g(\mathbf{x})$ is the usual importance sampling weight.

Similarly to rejection control, the efficiency of MIS depends on how close the trial density $g(\mathbf{x})$ is to the target $f(\mathbf{x})$.

5.5.3. Configurational bias Monte Carlo algorithm

Configurational bias Monte Carlo algorithm can be viewed as SIS-based Metropolized independence sampler. Let $\mathbf{x}^{(t)}$ denote the full sample obtained at iteration t. We first start by obtaining $\mathbf{x}^{(0)}$ and corresponding $w^{(0)}$ from $g(\mathbf{x})$ via SIS strategy. Suppose we obtained $\mathbf{x}^{(t)}$ with weight $w(\mathbf{x}^{(t)}$ at iteration t. At the next iteration, we do the following:

- Independently generate new trial configuration (sample) $\mathbf{y}$ from $g()$ using SIS strategy and compute it's importance weight $w(\mathbf{y}) = \frac{f(\mathbf{y})}{g(\mathbf{y})}$.
- Let $\mathbf{x}^{(t+1)} = \mathbf{y}$ with probability min$\big\{1, \frac{w(\mathbf{y})}{w(\mathbf{x}^{(t)})}\big\}$ and let $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$ otherwise.

Note the sampling is done sequentially using the SIS procedure. Therefore, we may incorporate a stage-wise rejection decision. Suppose we have succesfully built $k-1$ steps of a sample and obtained $\mathbf{x}_{k-1}$. Then at the k-th step of SIS we accept $\mathbf{x}_k$ built from $\mathbf{x}_{k-1}$ with probability

$$p_k = \min\Big\{1, \Big\} \tag{5.14}$$

When rejected, we go back to the first stage to rebuild the whole configuration. Note, one does not need to perform this acceptance - rejection decision at every stage and the standard resampling scheduling can be applied.

5.5.4. Building CBMC source estimator

6. Analysis of source detection estimators on the benchmark dataset

Antulov-Fantulin et al. [8] provided a dataset of SIR realizations along with their estimations obtained with Direct Monte Carlo for 4 classes of SIR parameters: $A = (p = 0.3, q = 0.3, T = 0.5), B = (p = 0.3, q = 0.7, T = 0.5), C = (p = 07, q = 0.3, T = 5)$ and $D = (p = 0.7, q = 0.7, T = 5)$. The benchmark dataset contains 160 such realizations on the grid of size $30x30$. Their estimations obtained with Direct Monte Carlo were held under convergence condition $|P^{2n}_{ML} - P^{n}_{ML}|/P^{2n}_{ML}| \leq 0.05$ and $|P^{x}_{i} - P^{2x}_{i}| \leq 0.05$ for all other nodes.

6.1. Correctness of the Direct Monte Carlo implementation

6.2. Correctness of the Soft Margin implementation

For the Soft Margin estimator we use the following convergence condition:

$$|\hat{P}^{n}_{a}(\Theta = \theta_{MAP}|\vec{R} = \vec{r}_{*}) - \hat{P}^{2n}_{a}(\Theta = \theta_{MAP}|\vec{R} = \vec{r}_{*})|/\hat{P}^{2n}_{a}(\Theta = \theta_{MAP}|\vec{R} = \vec{r}_{*}) \leq 0.05$$

and

$$|\hat{P}^{n}_{a}(\Theta = \theta|\vec{R} = \vec{r}_{*}) - \hat{P}^{2n}_{a}(\Theta = \theta|\vec{R} = \vec{r}_{*})| \leq 0.05.$$

6.3. Sequential Importance Sampling results

Sequential importance sampling is done under importance sampling distribution with the following properties:

- parameter p is fixed in steps $t < 5$ and $p = 1$ in the last step $t = T = 5$,

- parameter q is fixed,
- at each step, only nodes that are in the given final simulation may be infected with probability p,
- nodes that are infected may be recovered with probability q.

The simulations are done under the same convergence condition as Direct Monte Carlo simulation from the benchmark dataset, starting from $n = 10000$ samples.

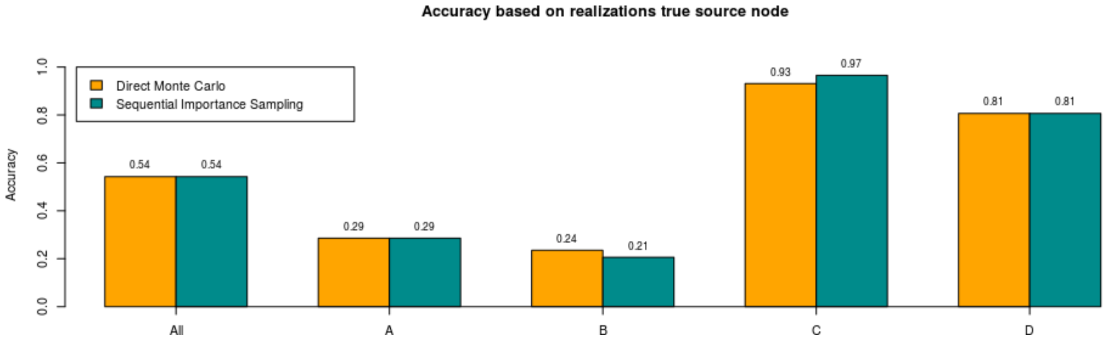

Figure 6.1

Figure 6.1 represents accuracies of estimations obtained by Direct Monte Carlo and Sequential Importance Sampling estimators w.r.t. the realizations true source node. In other words, they represent the portion of MAP estimations that correctly estimated the source node of the realization. When we observe low accuracy for Direct Monte Carlo estimator on average, we shouldn't expect such accuracy to be higher for the "inferior" Sequential Monte Carlo estimator. Accuracies for Direct Monte Carlo and Sequential Importance Sampling estimators follow similar pattern overall and for all the SIR parameter classes. For classes A and B they are low, and for classes C and D they are high.

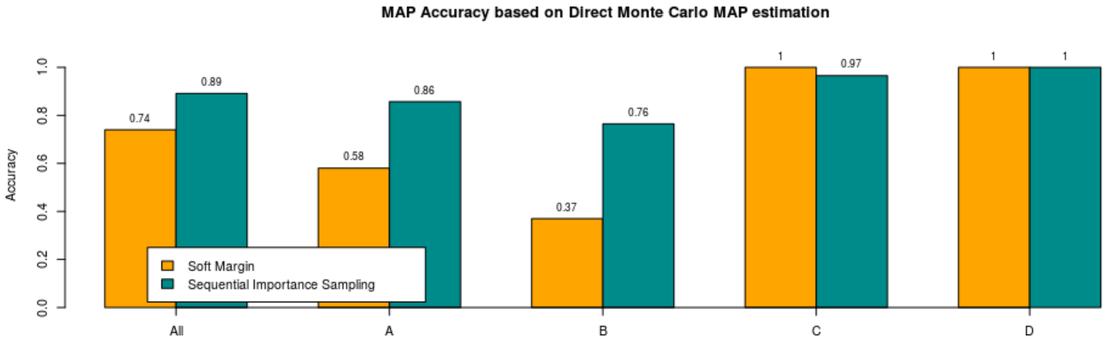

Figure 6.2

Figure 6.2 represents the accuracy used in [8] to compare range of estimators. This accuracy refers to the portion of MAP estimations that are equal to corresponding

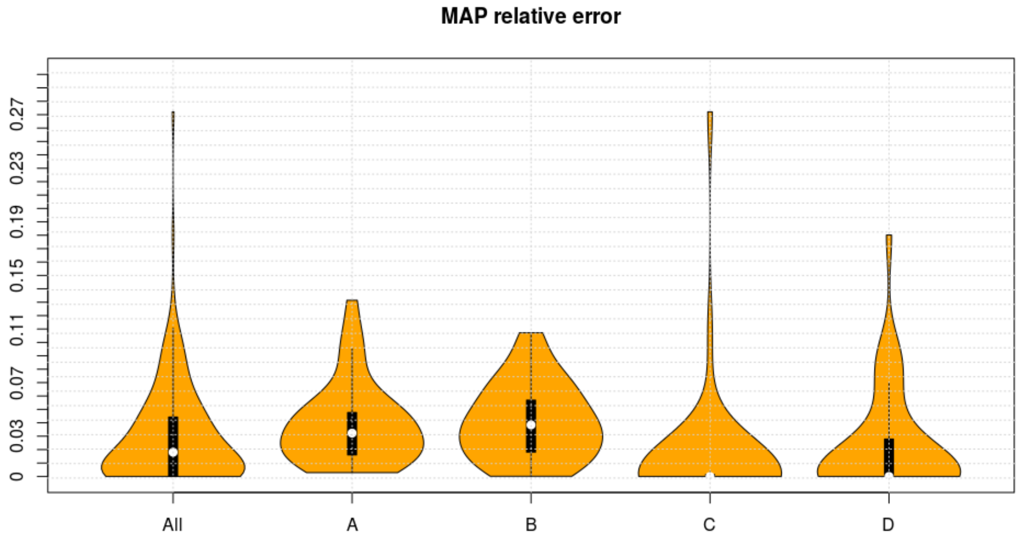

Figure 6.3

MAP estimations of Direct Monte Carlo estimator provided in the benchmark dataset. Soft Margin accuracies presented here are taken from [8].Those were calculate with fixed $a = 0.031$ and under the same convergence conditions as the benchmark Direct Monte Carlo solutions. Sequential Importance Sampling estimator for classes A and B outperforms SoftMargin. This only means its MAP estimations are more similar to Direct Monte Carlo estimations. Note that these classes also have low true source node accuracy and belong to low to medium detectability zone of parameters.

The similarity between estimations obtained with Sequential Importance Sampling and Direct Monte Carlo also presents itself as a low relative MAP error estimation w. r. t. Direct Monte Carlo probability across all classes of parameters, as presented in Figure 6.3.

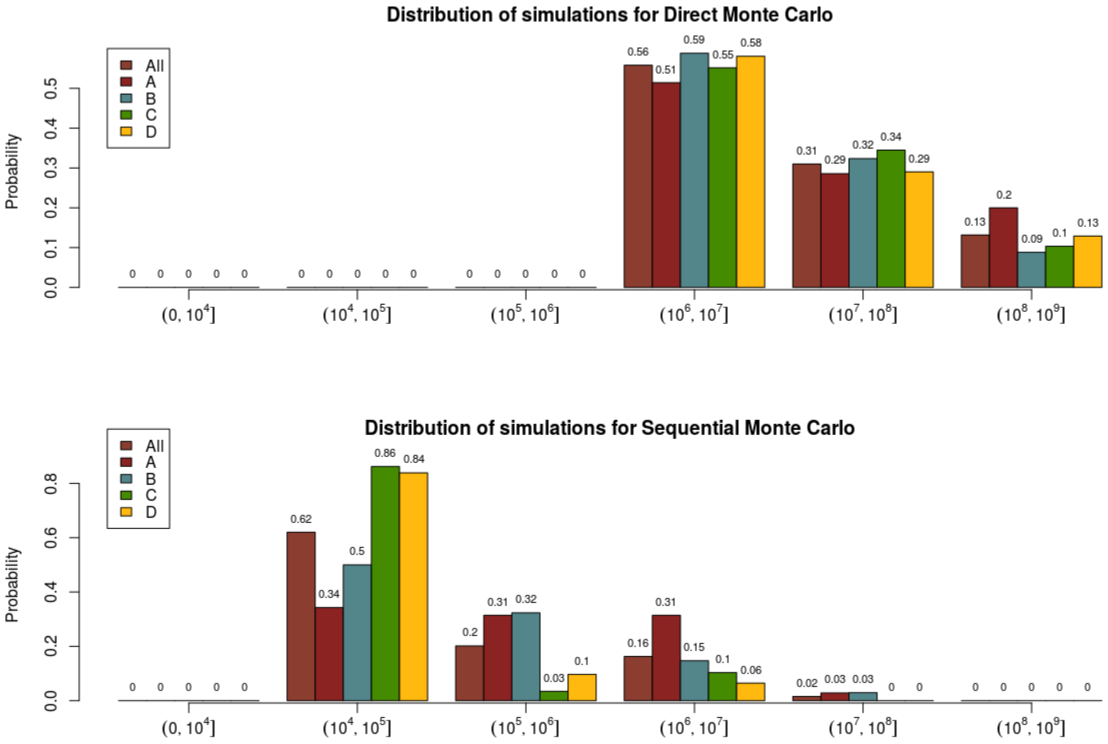

Figure 6.4

In Figure 6.4 distributions of number of simulations (samples) for which the estimators converged are presented. For Sequential Importance Sampling estimator we observe 10^5 samples are needed for SIR parameters in classes C and D in more than 80% of benchmark realizations. However, some simulations, observably mostly those in classes A and B, require more than 10^6 samples for convergence. The impact on the accuracies and the results presented here when the number of samples is capped by 10^6 is yet to be analysed.

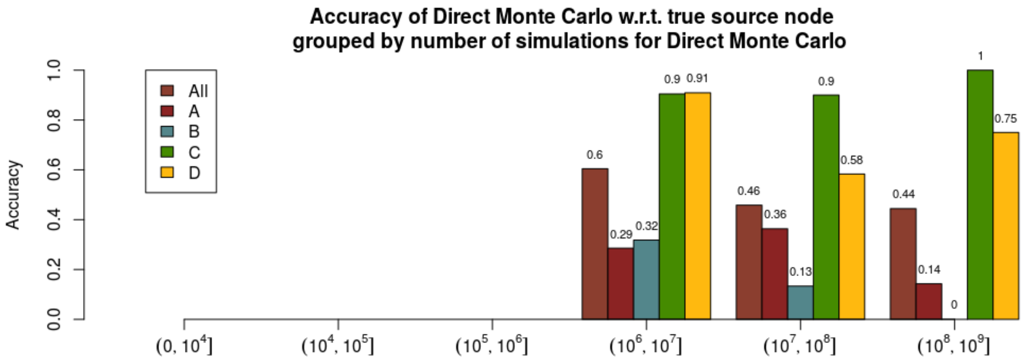

Figure 6.5

Figure 6.5 presents accuracy w.r.t. true source node of Direct Monte Carlo and Sequential Importance Sampling based estimations grouped by number of simulations required to obtain Direct Monte Carlo estimation for the coresponding benchmark sample.

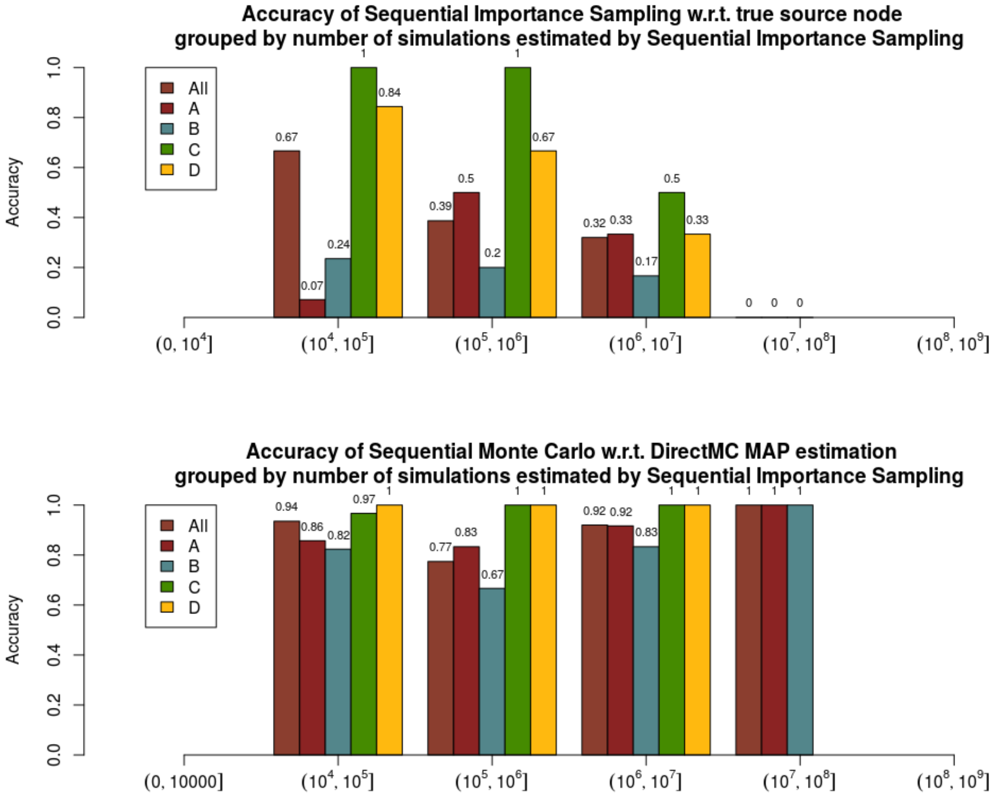

Figure 6.6

Figure 6.6 presents accuracies for Sequential Importance Sampling. Note the benchmark samples that required more than 10^7 simulations are the ones Direct Monte Carlo estimator also failed to estimate correctly.

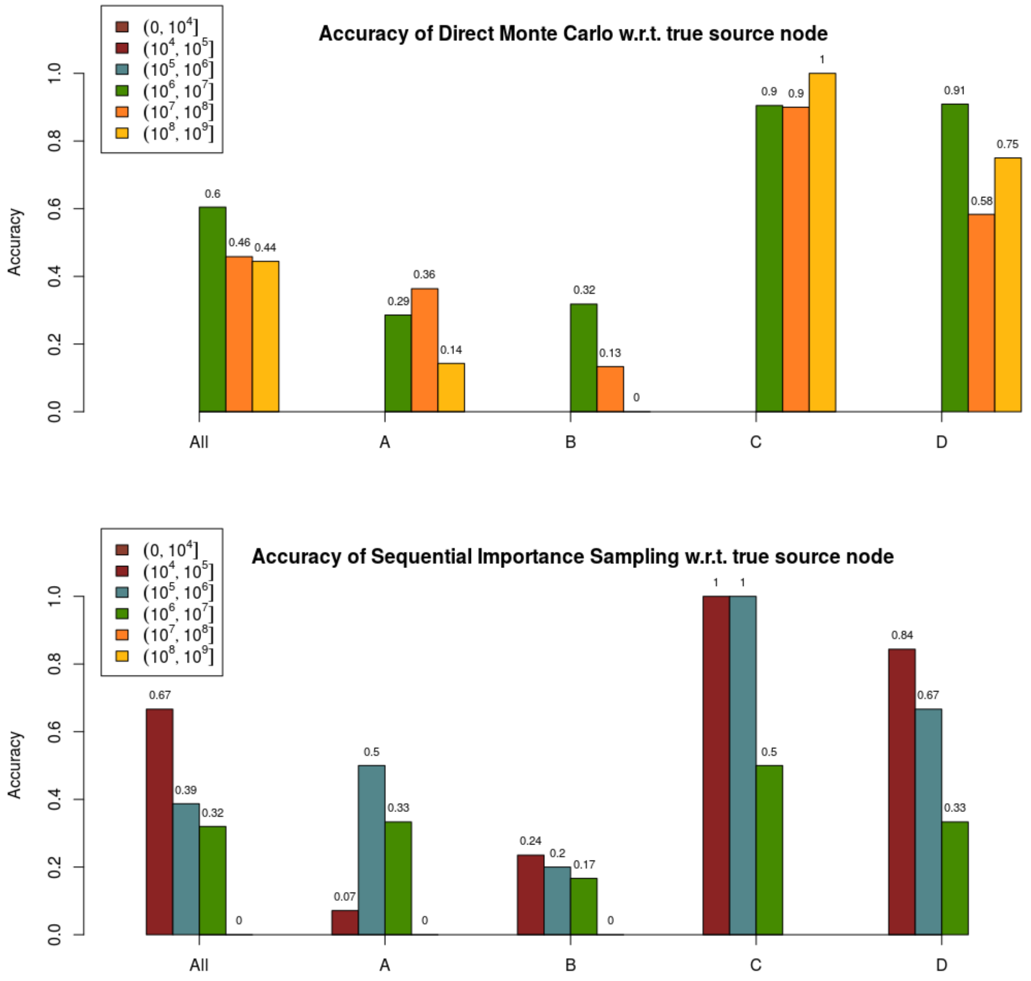
Accuracy of Direct Monte Carlo w.r.t. true source node
(0, 10^4]
(10^4, 10^5]
(10^5, 10^6]
(10^6, 10^7]
(10^7, 10^8]
(10^8, 10^9]
Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
0.6
0.46
0.44
0.29
0.36
0.14
0.32
0.13
0
0.9
0.9
1
0.91
0.58
0.75
All
A
B
C
D
Accuracy of Sequential Importance Sampling w.r.t. true source node
(0, 10^4]
(10^4, 10^5]
(10^5, 10^6]
(10^6, 10^7]
(10^7, 10^8]
(10^8, 10^9]
Accuracy
0.0
0.2
0.4
0.6
0.8
1.0
0.67
0.39
0.32
0
0.07
0.5
0.33
0
0.24
0.2
0.17
0
1
1
0.5
0.84
0.67
0.33
All
A
B
C
D

Figure 6.7

6.4. Sequential Importance Sampling with resampling

6.4.1. Simple Random Sampling

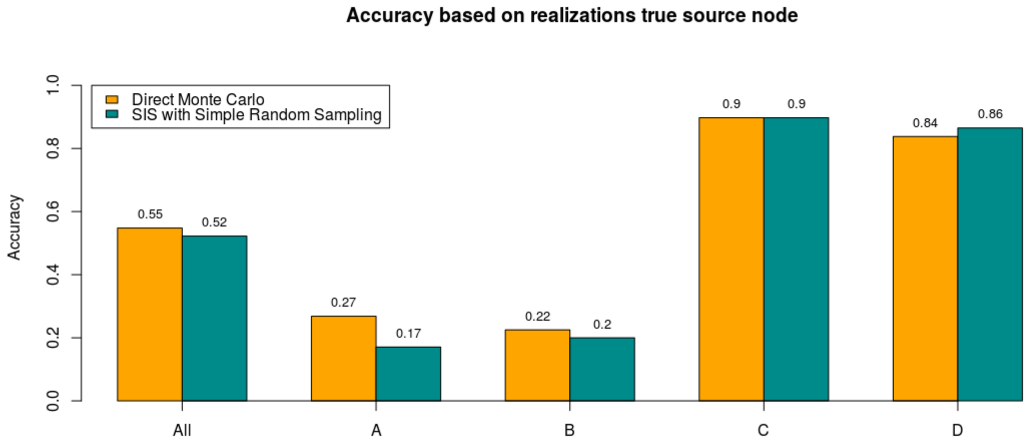

Figure 6.8

6.4.2. Residual Sampling

6.5. Sequential Importance Sampling and Soft Margin

6.6. Configurational Bias Monte Carlo

7. Detectability of patient zero

The source detectability $D(\vec{r_*}) = 1 - H(\vec{r_*})$ is characterized via Shannon entropy H (normalized by entropy of uniform distribution) of the estimated source probability distribution $P(\Theta = \theta_i | \vec{R} = \vec{r_*})$.

7.1. Detectability based on parameters of the SIR model

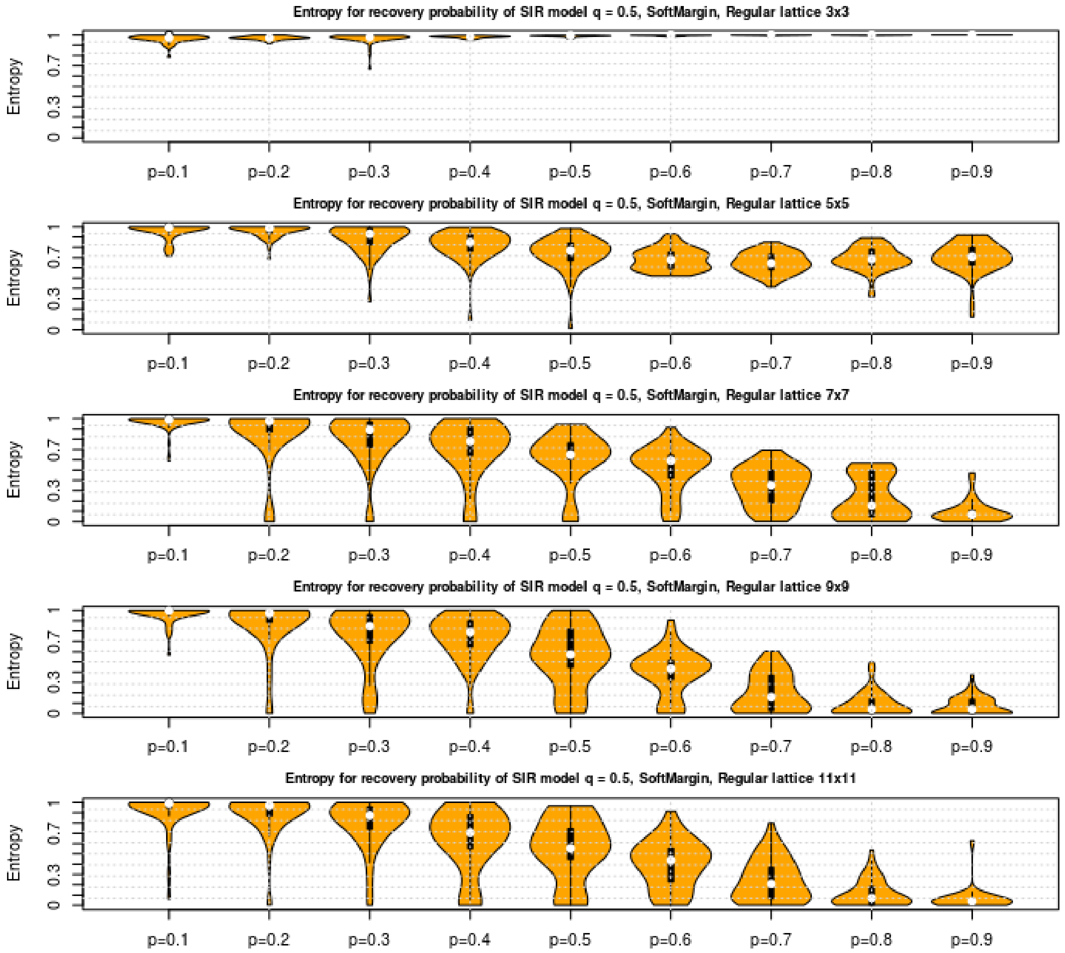

Figure 7.1: Box plots of estimated entropy density for entropy of source probability distributions of single source candidates on the 4-connected lattice of different sizes estimated with Soft Margin method with $10^4 - 10^6$ simulations with adaptive a chosen from $\{1/2^3, 1/2^4, \ldots, 1/2^{15}\}$. Estimation is done under SIR model with different parameters p in range $0.1 - 0.9$, fixed $q = 0.5$ and $T = 5$. The source node in each experiment is the central node of lattice. Each entropy density is estimated with 50 experiments containing realizations with more than 1 node.

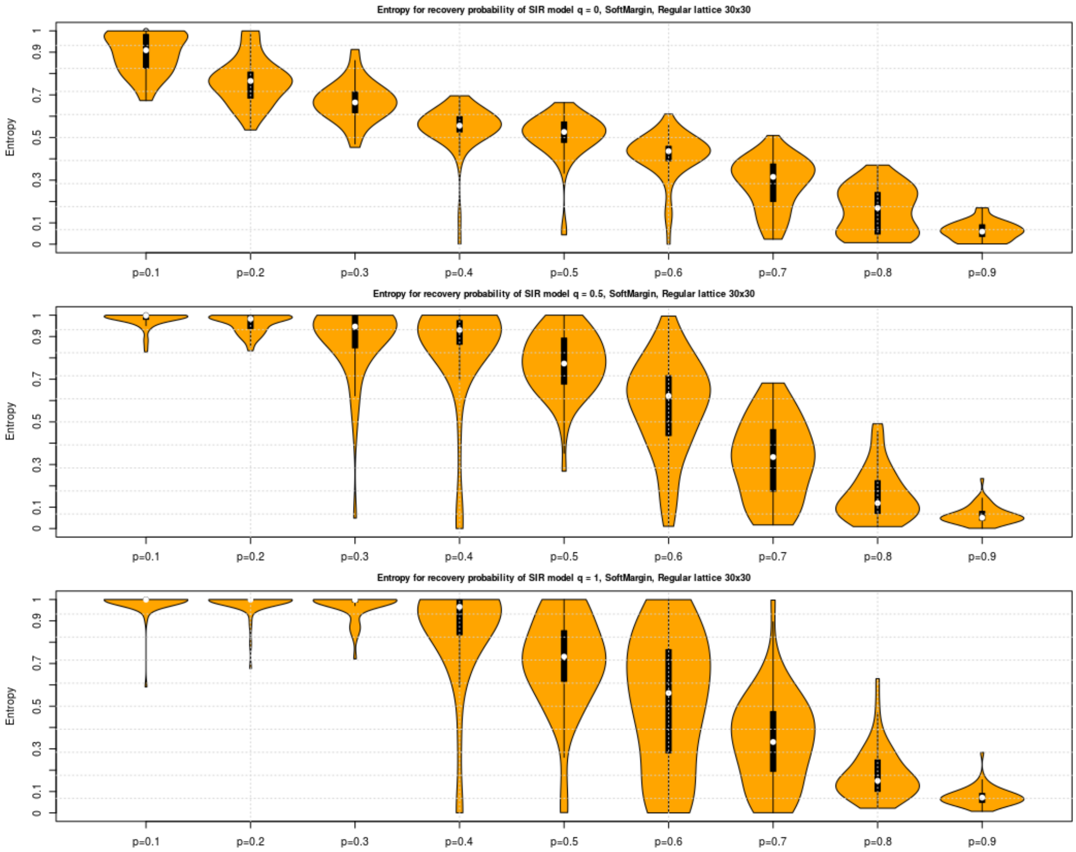

Figure 7.2: Box plots of estimated entropy density for entropy of source probability distributions of single source candidates on the 4-connected lattice 30 × 30 estimated with Soft Margin method with $10^4 - 10^6$ simulations with adaptive a chosen from $\{1/2^3, 1/2^4, \ldots, 1/2^{15}\}$. Estimation is done under SIR model with different parameters p in range $0.1 - 0.9$, and $q = \{0, 0.5, 1\}$ with $T = 5$. The source node in each experiment is the central node of lattice. Each entropy density is estimated with 50 experiments containing realizations with more than 1 node.

In Figures 7.1 and 7.2 the results of [8] are reproduced. The existence of different detectability regimes is shown in Figure 7.2 as well as a similar detectability behaviour for SIR models with the same parameter p across different values of parameter q. Three entropy regions are observed: low detectability-high entropy region $(p < 0.2)$, intermediate detectability - intermediate entropy region $(0.2 < p < 0.7)$ and high detectability-low entropy region $(p > 0.7)$.

In a regime where network size restricts the epidemic spreading but not the epidemic itself, the entropy is high as the realizations from different sources are almost identical (Figure 7.1).

7.2. Detectability based on parameters of the ISS model

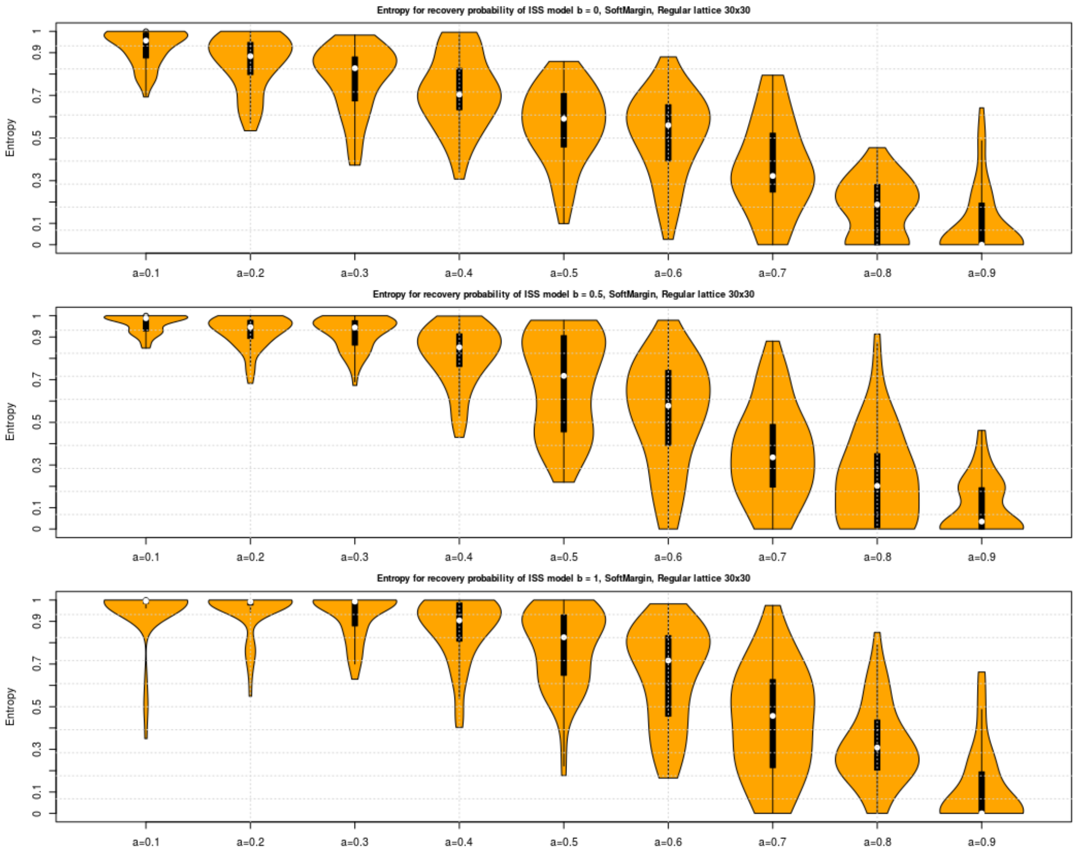

Figure 7.3: Box plots of estimated entropy density for entropy of source probability distributions of single source candidates on the 4-connected lattice 30×30 estimated with Soft Margin method with $10^4 - 10^6$ simulations with adaptive a chosen from $\{1/2^3, 1/2^4, \dots, 1/2^{15}\}$. Estimation is done under ISS model with different parameters a in range $0.1 - 0.9$, and $b = \{0, 0.5, 1\}$ with $T = 5$. The source node in each experiment is the central node of lattice. Each entropy density is estimated with 50 experiments containing realizations with more than 1 node.

7.3. Detectability based on network topology for the SIR model

7.3.1. Barabassi graph

Degree centrality

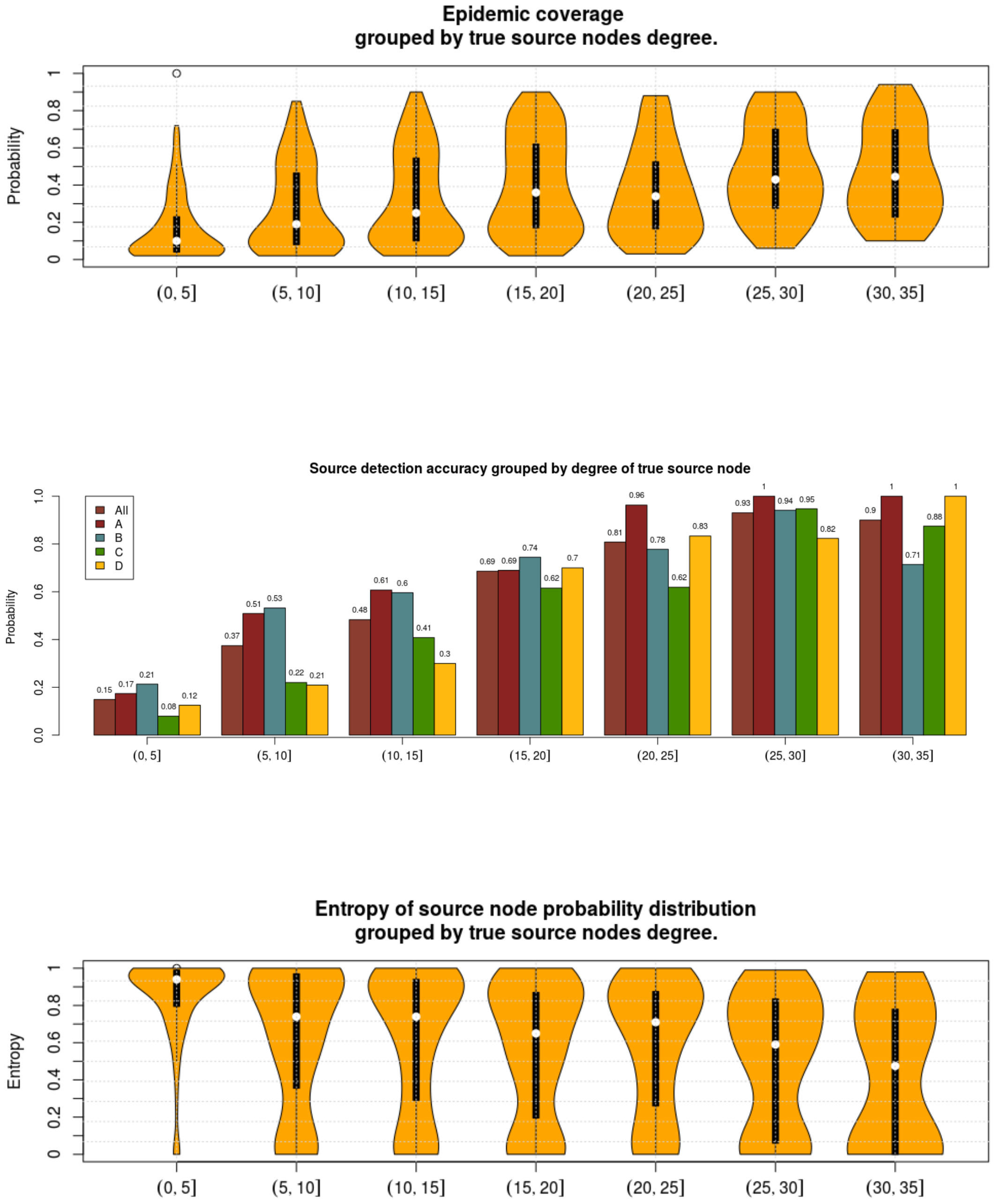

Figure 7.4: Kepsn

Closeness centrality

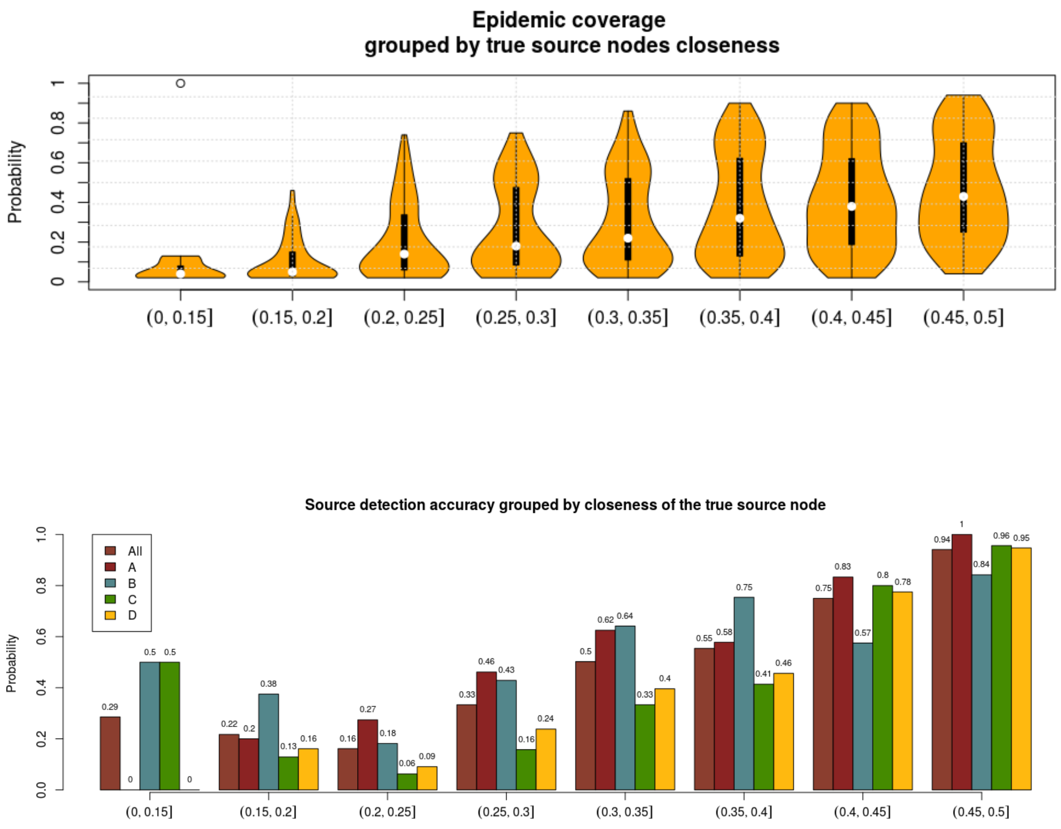

Figure 7.5: Kepsn

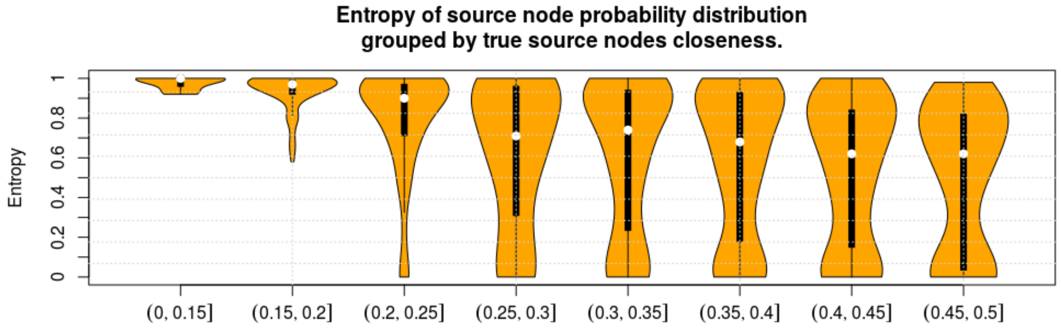

Figure 7.6: Kepsn

Betweenness centrality

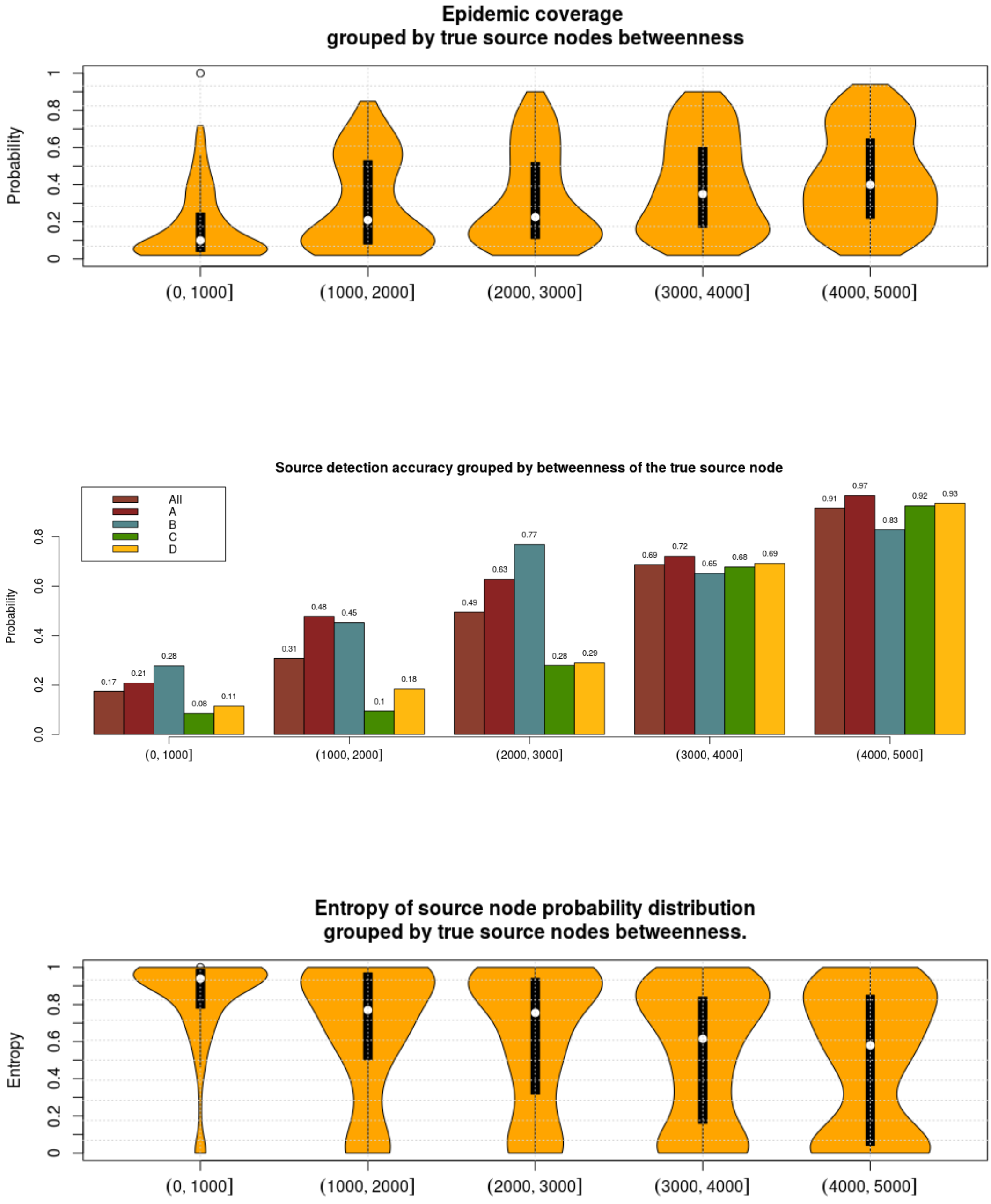

Figure 7.7: Kepsn

Eigenvector centrality

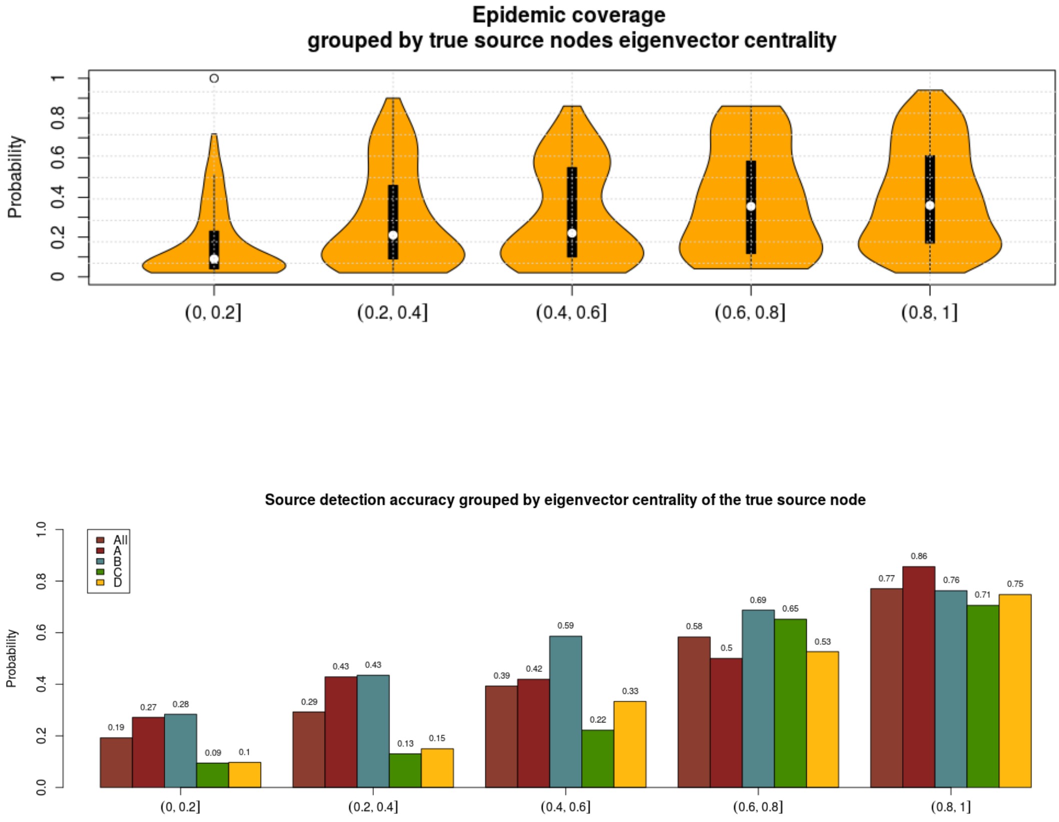

Figure 7.8: Kepsn

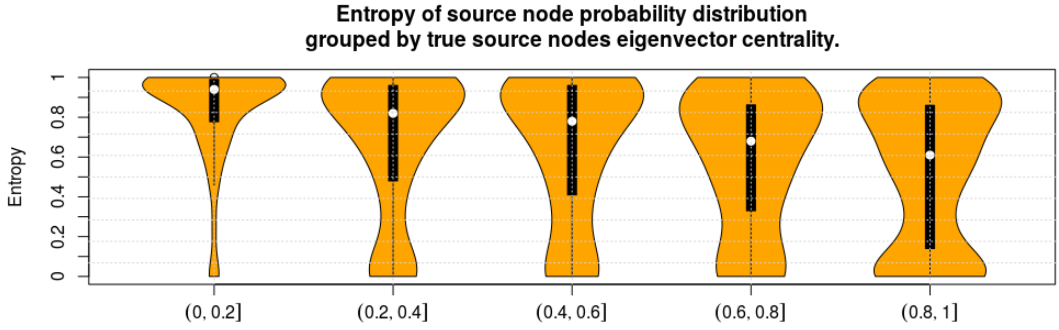

Figure 7.9: Kepsn

7.3.2. Erdős-Rényi graph

Degree centrality

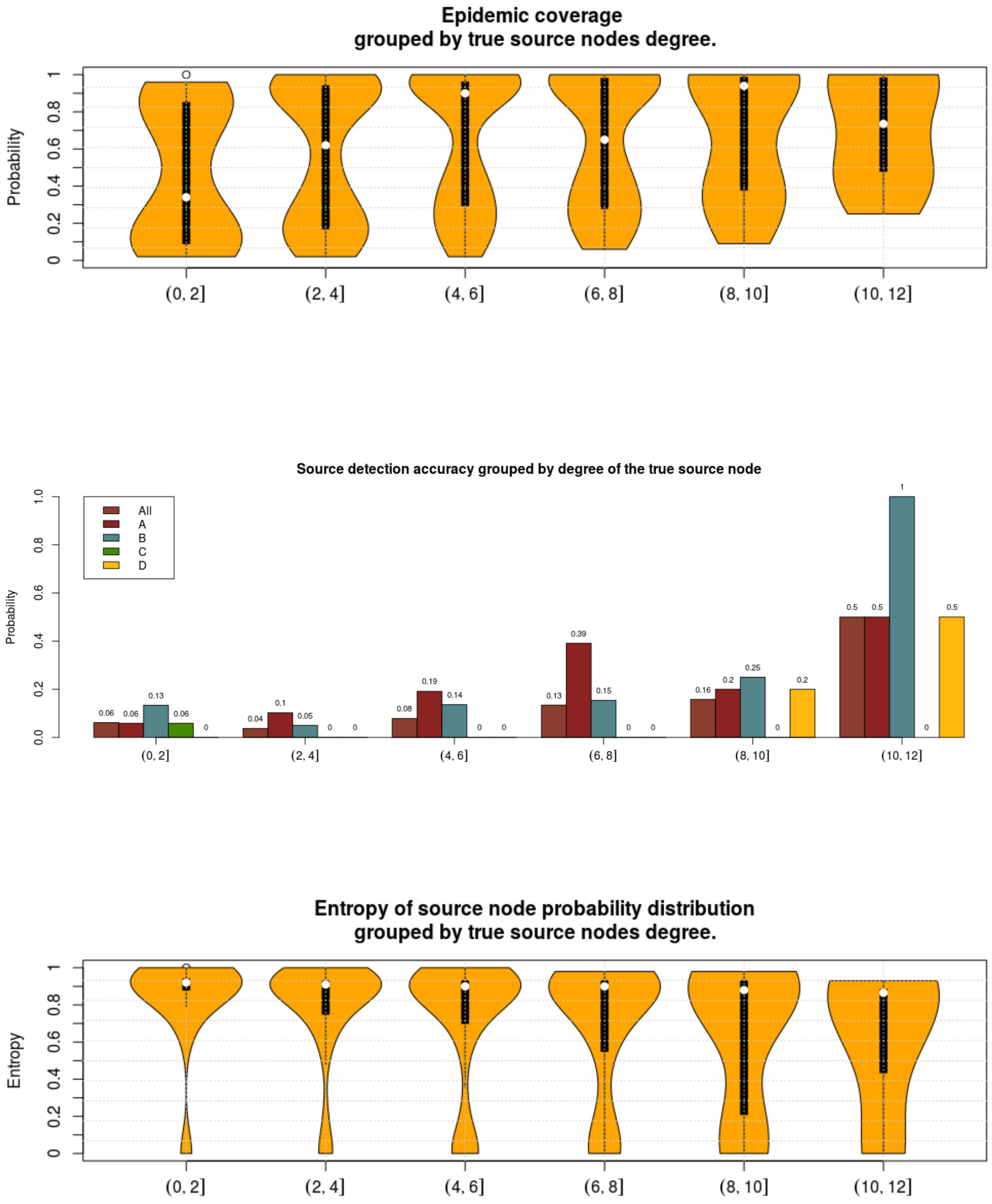

Figure 7.10: Kepsn

Closeness centrality

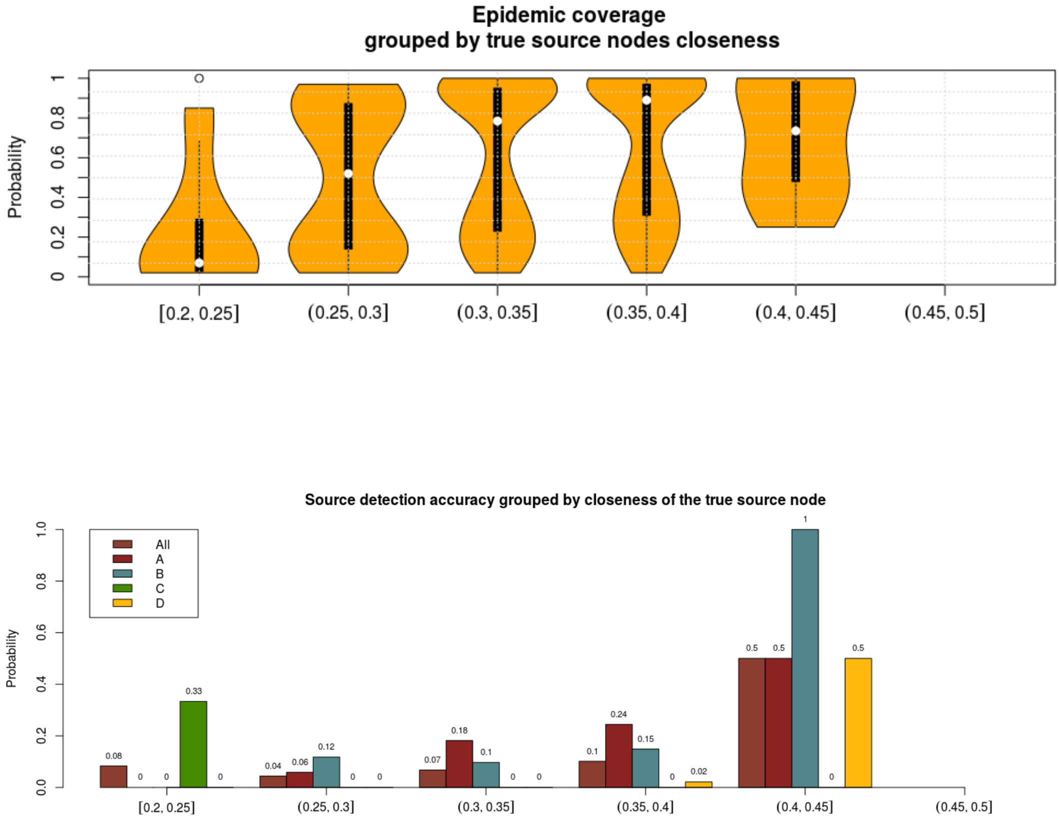

Figure 7.11: Kepsn

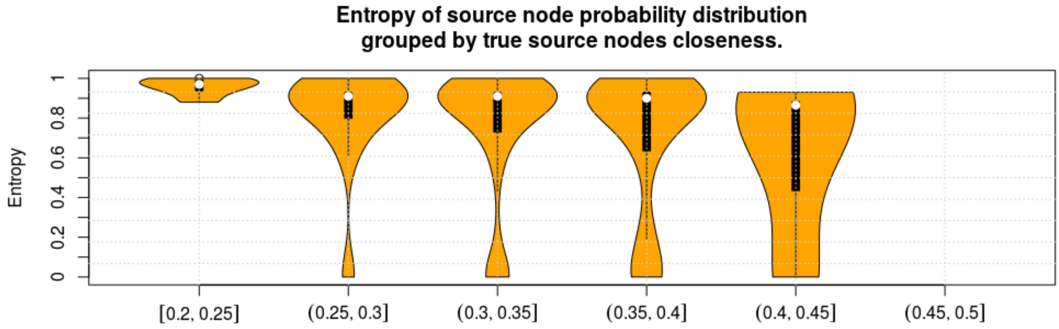

Figure 7.12: Kepsn

Betweenness centrality

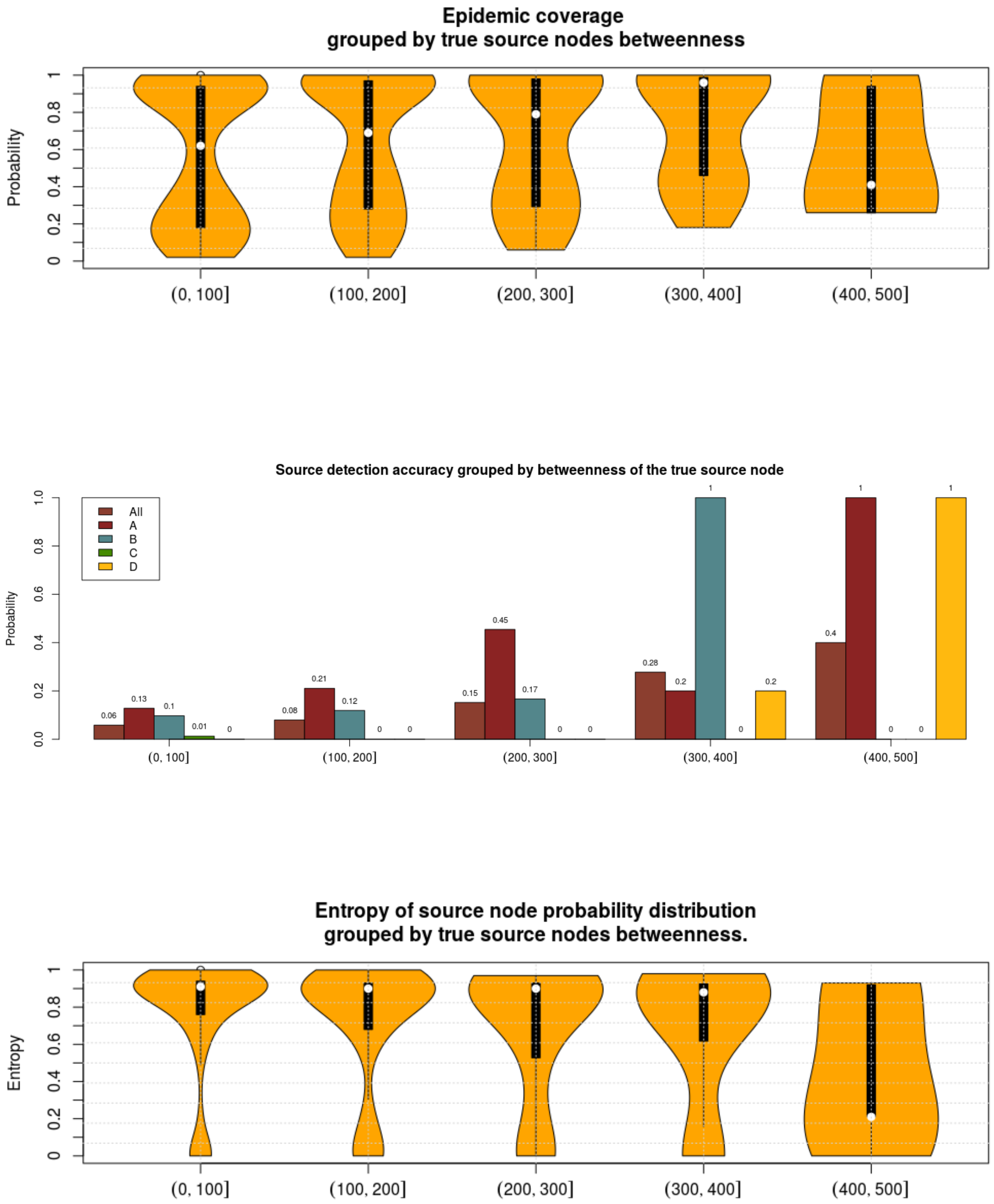

Figure 7.13: Kepsn

Eigenvector centrality

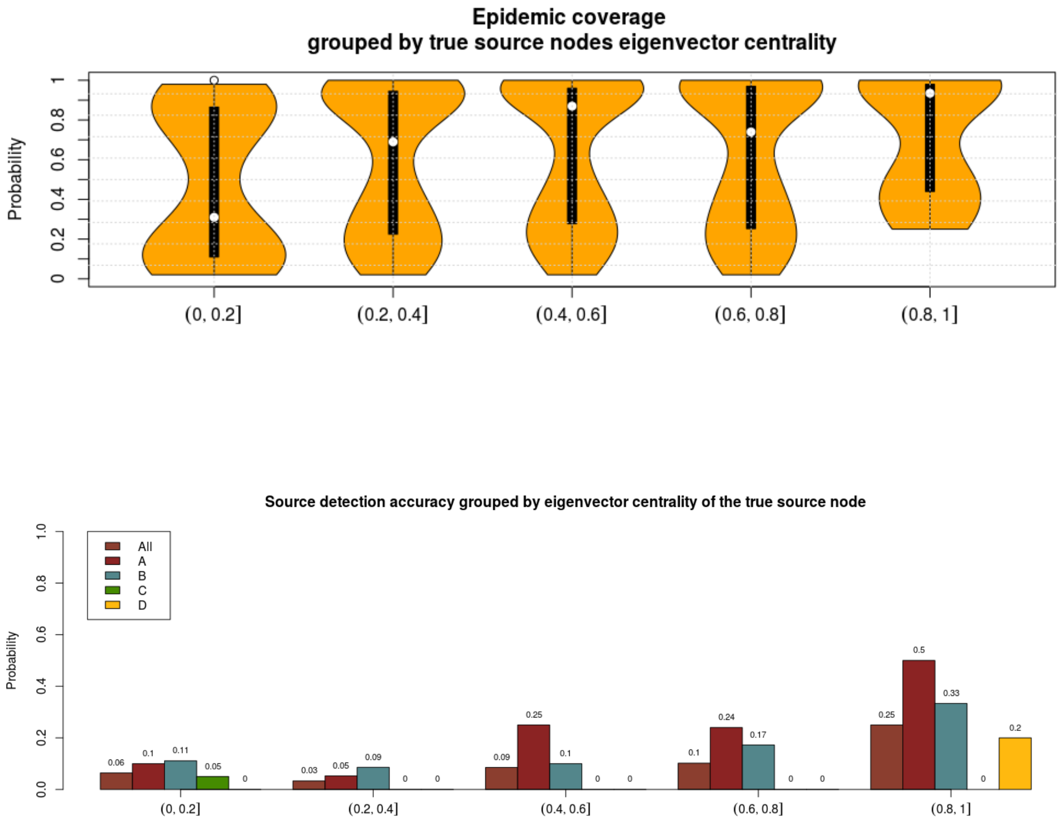

Figure 7.14: Kepsn

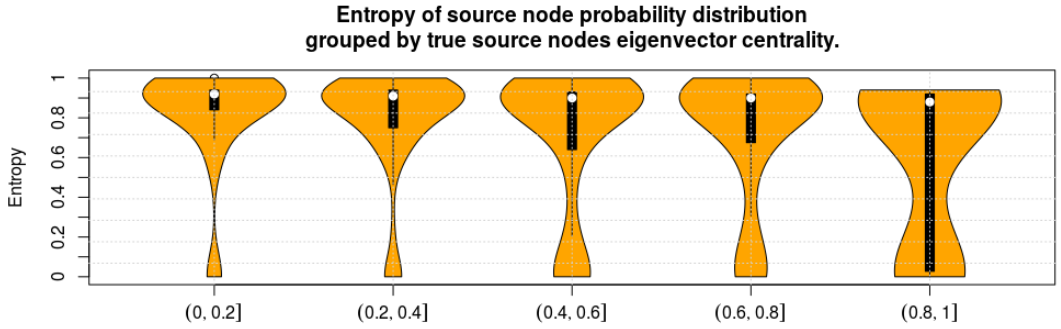

Figure 7.15: Kepsn

Degree coreness

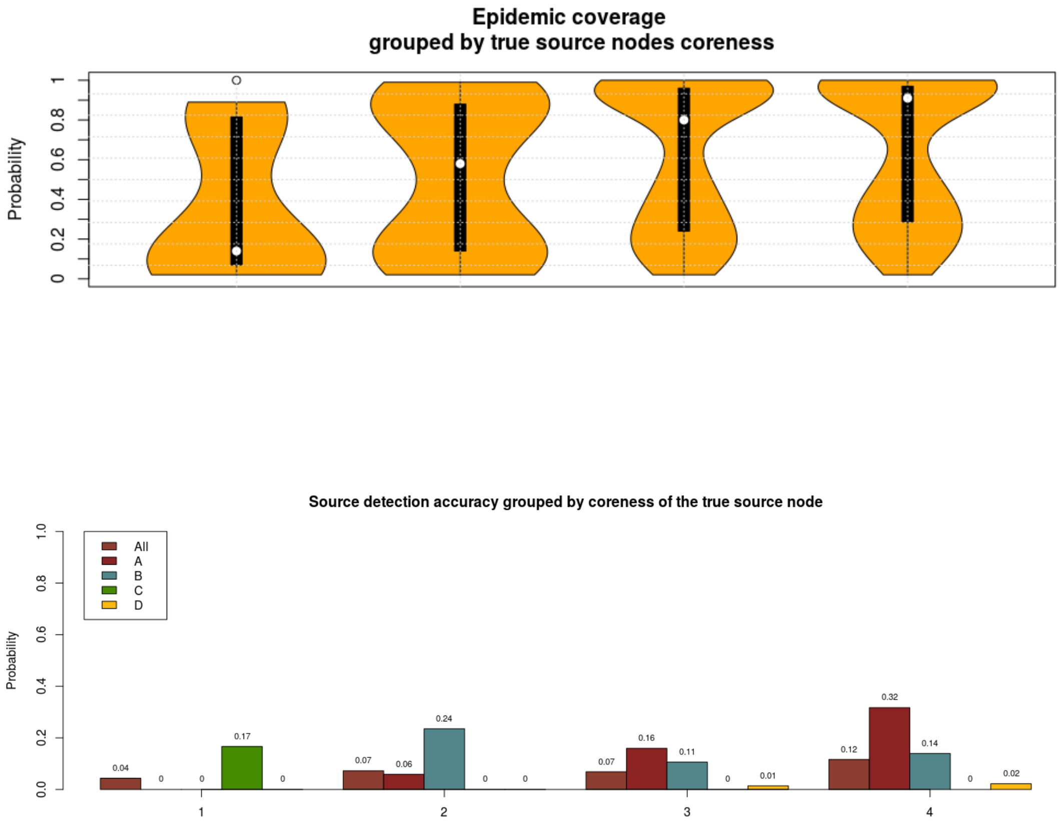

Figure 7.16: Kepsn

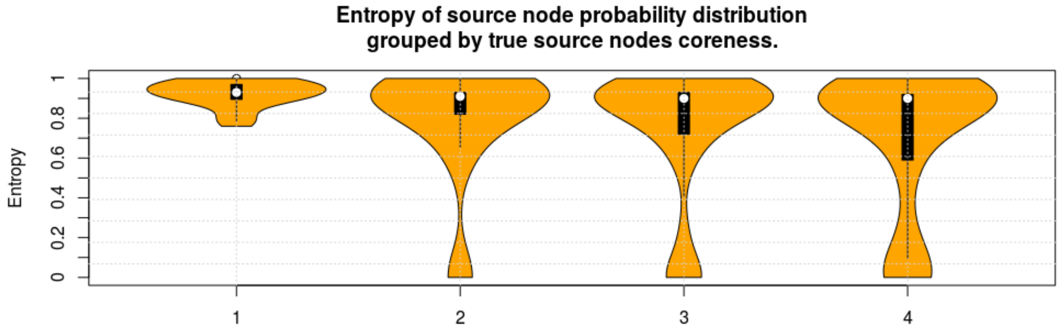

Figure 7.17: Kepsn

8. Conclusion

Zaključak.

BIBLIOGRAPHY

[1] Valdis Krebs. The social graph of a facmous mathematician. http://www.orgnet.com/Erdos.html, 2014.

[2] Ignacio Alvarez-Hamelin, Alain Barrat, and Ro Vespignani. k-core decomposition: a tool for the visualization of large scale networks. Technical report.

[3] M. E. J. Newman. The structure and function of complex networks. *SIAM REVIEW*, 45:167–256, 2003.

[4] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Rev. Mod. Phys.*, 87:925–979, Aug 2015. doi: 10.1103/RevModPhys.87.925. URL http://link.aps.org/doi/10.1103/RevModPhys.87.925.

[5] Jiaojiao Jiang, Sheng Wen, Shui Yu, Yang Xiang, Wanlei Zhou, and Ekram Hossain. Identifying propagation sources in networks: State-of-the-art and comparative studies. *IEEE Communications Surveys and Tutorials, accepted*, 17(9), 2014.

[6] Kai Zhu and Lei Ying. Information source detection in the SIR model: A sample path based approach. *CoRR*, abs/1206.5421, 2012. URL http://arxiv.org/abs/1206.5421.

[7] Andrey Y. Lokhov, Marc Mézard, Hiroki Ohta, and Lenka Zdeborová. Inferring the origin of an epidemic with a dynamic message-passing algorithm. *Phys. Rev. E*, 90:012801, Jul 2014. doi: 10.1103/PhysRevE.90.012801. URL http://link.aps.org/doi/10.1103/PhysRevE.90.012801.

[8] Nino Antulov-Fantulin, Alen Lančić, Tomislav Šmuc, Hrvoje Štefančić, and Mile Šikić. Identification of patient zero in static and temporal networks: Robustness and limitations. *Phys. Rev. Lett.*, 114:248701, Jun 2015. doi: 10.1103/PhysRevLett.114.248701. URL http://link.aps.org/doi/10.1103/PhysRevLett.114.248701.

[9] Albert lászló Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 1999.

[10] Phillip Bonacich. Power and centrality: A family of measures. *American Journal of Sociology*, 92(5):1170–1182, 1987.

[11] P. Erdős and A. Rényi. On random graphs. I. *Publ. Math. Debrecen*, 6:290–297, 1959.

[12] Samukhin Dorogovtsev, Mendes. Structure of growing networks with preferential linking. *Physical Review Letters*, 85, 11 2000. doi: 10.1103/physrevlett.85.4633. URL `http://gen.lib.rus.ec/scimag/index.php?s=10.1103/physrevlett.85.4633`.

[13] Nino Antulov-Fantulin, Alen Lancic, Hrvoje Stefancic, and Mile Sikic. Fastsir algorithm: A fast algorithm for simulation of epidemic spread in large networks by using SIR compartment model. *CoRR*, abs/1202.1639, 2012. URL `http://arxiv.org/abs/1202.1639`.

[14] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1994. ISBN 0201558025.

[15] Yamir Moreno, Maziar Nekovee, and Amalio F. Pacheco. Dynamics of rumor spreading in complex networks. *Phys. Rev. E*, 69:066130, Jun 2004. doi: 10.1103/PhysRevE.69.066130. URL `http://link.aps.org/doi/10.1103/PhysRevE.69.066130`.

[16] Jun S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer Publishing Company, Incorporated, 2008. ISBN 0387763694, 9780387763699.

[17] Wing Hung Wong Augustine Kong, Jun S. Liu. Sequential imputations and bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994. ISSN 01621459. URL `http://www.jstor.org/stable/2291224`.

Detectability of Patient Zero Depending on its Position in the Network

Sažetak

Sažetak na hrvatskom jeziku.

Ključne riječi: Ključne riječi, odvojene zarezima.

Title

Abstract

Abstract.

Keywords: Keywords.