CSE 8803 EPI: Data Science for Epidemiology, Fall 2023

Lecturer: B. Aditya Prakash
Scribe: Sudarshan Anand, Harshini Reddy Vummadi

September 17, 2024
Lecture # : 9 Inference II

---

1 Summary

As data collection will never be a perfect process, we continue to discuss ways that missing data can be inferred from what data we do have. In the first half of the class, we wrapped up the discussion on the different ways to calibrate Agent Based Models (ABMs) and discussed how Neural Networks with some epidemiological knowledge helps in this feat.

Then we moved onto the Inference II powerpoint and discussed two main topic: how to find Patient Zero and how to infer missing infections. In the former, several methods are given to inituitively track down the center of an infection graph. Then in the latter, we discussed 4 methods in detail for finding the infections that have slipped through the crack. Using these methods in tandem, we can observe a more complete view of an epidemiological history and better understand how to contain current and future epidemics.

2 Finding Patient Zero

Aim: Given the unlabelled contact network, can we find which node was the first to get infected?

Data collection is not perfect and will miss some cases. In some real-world disease analyses, such as with Tuberculosis (CDC, 2007) or AIDS, the source was able to be predicted through reconstructing the likely transmission path to the first case, Patient Zero [1]. Finding Patient Zero not only tells us the original causes of an epidemic, but can also tell us how to intervene.

Visually looking at the network, humans could spot likely candidates for patient zero based on identifying clusters and based on past trends. For example, consider the grid network in Figure 1. Intuitively we could say that the most likely patient zero candidates are the centers of the two blobs. However, how do we get the candidate(s) algorithmically?

We first started with the familiar MLE (Maximum Likelihood Estimation) [2, 4]. As per this, we claim the following:

"The best source is the one that maximizes the data likelihood in the SI model."

$$\hat{v} = argmax P(G_N | v^* = v)$$

where G_N is the infected subgraph and v is an observed node in G_N.

We use the concept of Rumor centrality to solve this MLE problem.

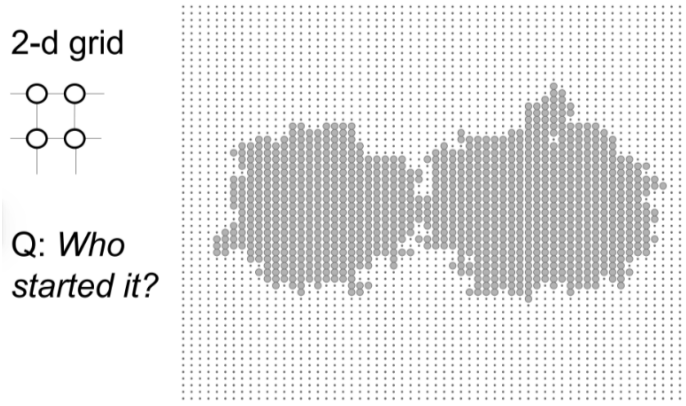


Figure 1: *An example contact network. Visually, it seems obvious that the centers of the blobs should be the most likely sources of the infection (patient zero candidates). But how do we represent this algorithmically?*

2.1 Rumor Centrality

Given an infection snapshot (the network state at a particular time instant), a *propagation ripple* shows the possible path of disease spread considering a node v as patient zero.
This process sums up the likelihood of all ripples across different time snapshots of a network [11, 12].

$$P(G_N|v^* = v) = \sum_{\text{all ripples}} R_i$$

This quantity is hard to compute: #P-hard[1] (Since there are so many different ways the disease could propagate)!
However, probabilities of different cascades can be computed efficiently for k-regular trees assuming it's an SI model. We can determine the most likely Patient Zero from trees by utilizing these probabilities. Now, we could just extend this idea to general graphs by extracting a tree from the graph such as a Minimum spanning tree that covers all nodes in the graph or use breadth-first search to generate a subgraph. This can serve as a quick and dirty heuristic.

2.2 Finding Number and Identity of Sources

Looking back at Figure 1, it is highly likely that there are 2 sources, one at the center of each cluster. However, how can we determine the number of sources for more complicated scenarios?
When an infected node is surrounded by many uninfected nodes, the chances of the infected node being a source are slim. This is called ***exoneration***. Since the first infected node (the source) has the longest time frame to infect neighbors, one that still has uninfected neighbors is less likely to be the first.
We arrive at a two part solution: use ***minimum description length*** for various numbers of seeds, and for that seed number, calculate exoneration as the sum of centrality and penalty.

[1] #P (sharp-P) is the problem of finding the number of solutions to a problem in NP. #P-hard problems are those where counting the number of solutions is hard.

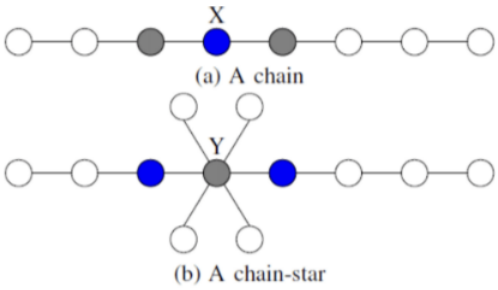


Figure 2: *Effect of network geometry on the Maximum Likelihood Estimation. Since X is in the middle of two nodes and has infected both, it is highly likely to be the source of the infections. However, since Y has managed to infect only two of its six neighbors, it is less likely to have caused the outbreak.*

The **MDL principle**, from information theory, states that given a limited set of observed data, the best explanation is the one that permits the greatest compression of data [3]. This principle is very useful in model selection. In other words, the simplest and the most elegant explanation is the best one (Ockham's razor).
The first part of the solution involves utilizing the above ***MDL principle*** to determine the number of seed nodes that best explains the infection spread network [10]. For the grid network in Figure 1, the network having two sources is the simplest explanation given that there are two clusters. Claiming just one seed node would either require us to explain the infection crossing between clusters (if seed in one of the clusters) or why infection spread only to the left and right (if seed node between clusters).
The running time of this two-part solution is linear and can be optimized using NetSleuth [10]. Seedset Scoring:

$$\mathcal{L}(S) = \mathcal{L}_{\mathbb{N}}(|S|) + log\binom{N}{|S|}$$

where $|S|$ is the encoding integer and the log term is the number of possible $|S|$-sized sets. Big O Runtime: $O(k * (E_1 + E_F + V_I))$, meaning it is linear!

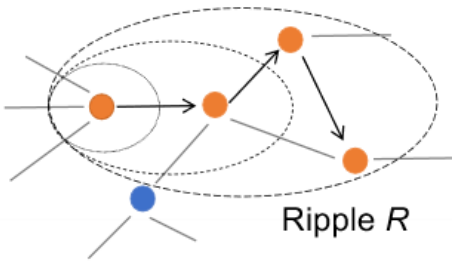


Figure 3: *Propagation ripple: Possible traces of infection spread in a given snapshot of the infected network. Here, the leftmost yellow node is assumed to spread the infection. The dotted-circles show the frontier expansion as the ripple propagates.*

Here, in our two-part MDL solution, we encode both the model and the data together. For the model selection, we utilize the above seed set scoring function and the data for this are the propagation ripples (Figure 3). The method to optimize the score is first we identify

the high-quality node set given k, and then given the nodes we must optimize the ripple R (Figure 3). The ripple cost of each additional one added is:

$$\mathcal{L}(R|\mathcal{S}) = \mathcal{L}_{\mathbb{N}}(T) + \sum_{t}^{T} \mathcal{L}(F^t)$$

where $\mathcal{L}_{\mathbb{N}}(T)$ represents how long a ripple is and the summed term describes how the "frontier" advances.

Overall, the total MDL cost is

$$\mathcal{L}(G_I, \mathcal{S}, R) = \mathcal{L}(\mathcal{S}) + \mathcal{L}(R|\mathcal{S})$$

Some extensions of The MDL Algorithm that have been published include:

- Different models
- Temporal networks
- More rigorous results on specific graphs
- Noisy input
- Using graph neural networks

2.2.1 GNNs

2.2.2 GNN preliminaries

Graph Neural Networks are tools utilized to handle unstructured data that do not have a fixed order of traversal like text and audio, which are traversed from left to right or images that can be traversed pixel by pixel. The key mechanisms GNNs utilize are the *message-passing mechanism* and *aggregation* [8]. Let us first get an intuitive idea of this:

Think about when you and your friends were booking the tickets for Six Flags. When all of you go into the ticket portal to decide on a time for the bus, imagine each person calling their friends to inform the time slot they have booked. Now everyone has received information about a possible availability in one of the buses. So, each person aggregates information received from their friends to decide on a time slot for the bus.
This is very similar to how message-passing works in GNNs. In the network, each node u has an initial state (in above example, bus time slot is equivalent to the node's state), and at each layer, each node u sends a "message" to all of its neighbors about its current state (*message-passing*), and updates its state based on the "messages" from its neighbors about their states (*aggregation*).
The layers in a GNN are something like rolling out the network, such that the message pass can be thought to be sent from one layer to the next.

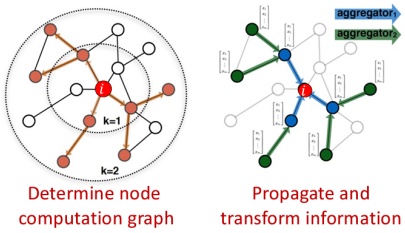


Figure 4: *The idea of GNNs [6]*

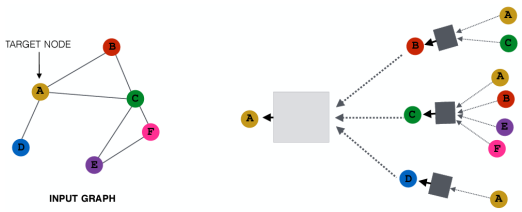


Figure 5: *An example showing the input graph and the computation graph. Here, the GNN has 2-layers. So, it includes nodes that are at most 2-hops away from the target node A (nodes E, F) [6]*

2.2.3 GNNs for finding Patient Zero

A final method to detect patient zero is using graph neural networks. In [7], the authors propose a framework for the same:

- Utilize observed data with a Variational Auto-Encode (VAE), a deep generative model, to learn the source distribution (possible candidates for patient-zero) and model the uncertainty.
- The diffusion process can be complex here given that it depends on the network structure, so the authors train a GNN, which is good at capturing relational structure in addition to individual node (edge) features, to learn this process and use it to solve the forward problem of ascertaining the probability of a given node u being the source (patient-zero) in the given network (inputs: network and node; output: the probability).
- Finally, they combine the two ideas to jointly solve the forward problem of giving probability as well as the inverse problem of finding the source.

2.3 Reconstructing the COVID-19 Pandemic Epicenter

How was Hunan Seafood Wholesale Market in Wuhan guessed as COVID ground zero? Researchers used spatial relative risk analysis done by compiling data from many sources: Weibo cases, CCDC sequencing PCR report, population density data, animal sales records, and mobility data. Investigators were then able to pinpoint the source to vendors selling live mammals as seen in Figure 6 [14].

3 Finding Missing Infections

A large number of COVID-19 infections went and still are continuing to go unreported. When there was only 23 reported infections in 5 major American cities in March 1st, there may have been greater than 28,000 in actuality. The difficulty in estimating the unreported infections allowed it to spread more quickly in the US and worldwide, and when severity is slowed so too is the remedial action that follows. Potential reasons for missing infections vary from a lack of testing to asymptomatic infections.

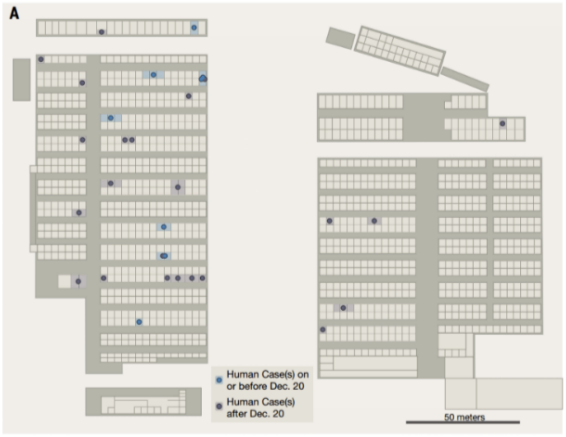


Figure 6: *This visual model shows how we can use physical spaces to model real-world interactions and make inferences about the kind of underlying networks that might be at play. The red dots show that new infections mostly happen within the vicinity of the old infection, indicating the efficacy of some kind of MLE model.*

To identify the true reported rate, consider a serological methodology (test everyone for infection). This is considered the gold standard for disease reporting, It tests if the person's body has the required antibodies for the disease, indicating that they contracted the disease. Hence, it can give a very accurate picture of the people who contracted the infection. However, this is an expensive method and is also a delayed process and we might not obtain the information for analysis in time. Another method uses the ILI system, where researchers compared COVID with influenza to predict actual rates of infection. However, this method suffers from ad-hoc corrections and confounding factors as every disease has its own unique characteristics that effects the way it spreads. Given these methods and their drawbacks, we turn to epidemiological models to better determine missing infections.

3.1 Approach 1: Real-World Calibration

The idea is to use real-world data to provide some insight on the proportion of cases that aren't being reported. Then fit I_r to reported infections to estimate the unknown parameters such as the rate of reporting α. However, they suffer greatly from ad-hoc modeling assumptions. This means altering α slightly can drastically alter forecasts.
Overparameterization is a big problem in epidemiology and you can have many combinations of parameters that will fit your data. When trying to determine what model to use given a similar α, it is often best to go with the simpler of the two. This ties back to the principles of MDL, as the simpler model will generalize better and avoid issues with overparameterization over the more complex model.

3.2 Approach 2: MDLInfer

In this approach the number of reported infections is known: $D_{reported}$. If we are then given the correct count of total infections D, the model can be calibrated with both to find a better fit of $D_{reported}$. Put simply, we want to find the D^* that fits the $D_{reported}$ curve the best. Now we use MDL for a clear problem formulation:

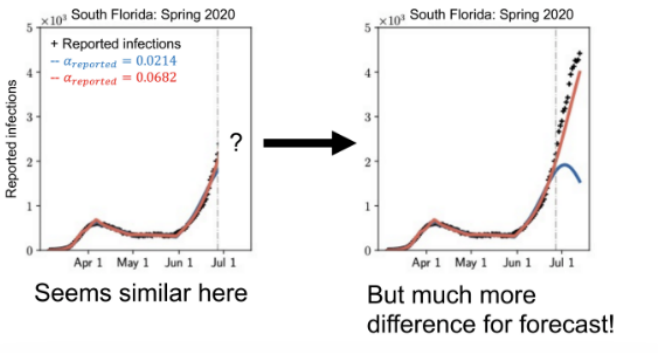


Figure 7: *With two different values of α, the model matches the training data exceedingly well. However, one model performs very well on holdout data, while the other performs very poorly. This illustrates the need for parameter optimization that goes beyond how how well a model matches training data.*

$$D^* = \underset{D}{\operatorname{argmin}} L\left(D_{\text{reported}} \mid D\right) + L(D)$$

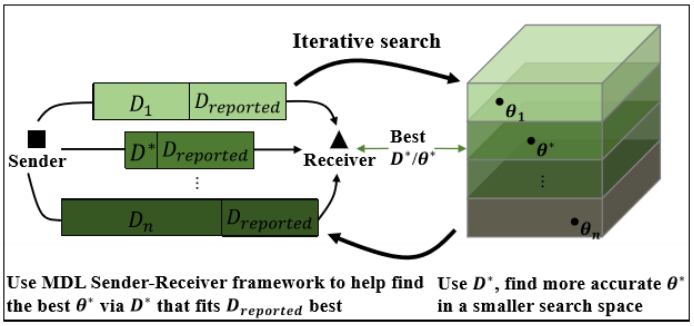


Figure 8: *This illustrates how the MDLInfer algorithm works; it relies on an iterative search through the parameter space and communication between a sender and receiver of information. This helps penalize the information of the data given the model and the information encoded in the model.*

Governing Concepts:

- **Minimum Description Length (MDL):** This is largely driven by concepts from Information Theory that aim to find the best set of parameters to a model that both minimizes the amount of information needed to both describe the model and the residual errors between observed and model-predicted values.
- **Likelihood Function**: This is key in quantifying how well a model describes the observed data. In the context of epidemiological models, this function often involves terms for susceptible, exposed, infected, and recovered individuals (S, E, I, R).
- **Regularization**: A penalty term is introduced to the likelihood function to prevent overfitting. The MDLInfer algorithm favors simpler models unless a more complex

model significantly improves the fit to the data. This is achieved by encoding the number of bits used in the model. A neural network would have a *lot* of bits because there are so many parameters, but an SIR model would only have a few because there are so few parameters.

This says that the information of the data ($D_{reported}$) given the model (D) plus the information actually encoded in the model is the quantity of bits encoded [4]. This is the idea of Occam's razor mentioned earlier which says that the best solution is the simplest solution because we can find many models that seem to fit well (Figure 5). While the approach is simplistic, MDLInfer was found to have great performance when predicting the future.

3.3 Approach 3: Steiner Trees

This is a datamining method that learns the tree with minimum weight in the graph that connects all of the given terminals. In our case, the terminals would be known cases and the minimum tree built to include each of those would implicate many other nodes that may be the unreported infections. These are determined with dynamic solvers [5].

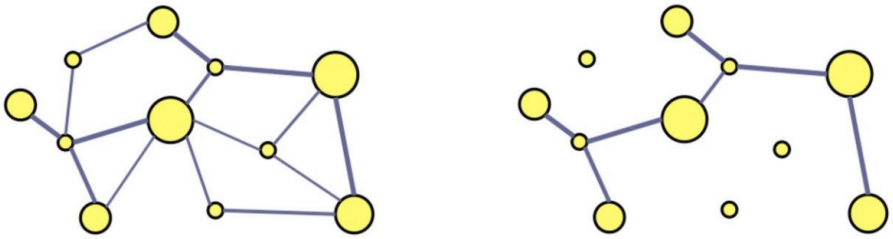


Figure 9: *Where larger circles are reported infections, we can clearly see the construction of a minimum spanning tree requires a few of the smaller circles to connect infected ones. So, the small circles in the Steiner tree of the network could give us possible missing infections.*

The Steiner tree method can be extended to temporal networks by converting it to one static graph. It can also be used to learn node weights from features, thus allowing us to estimate high-danger patients, for instance.

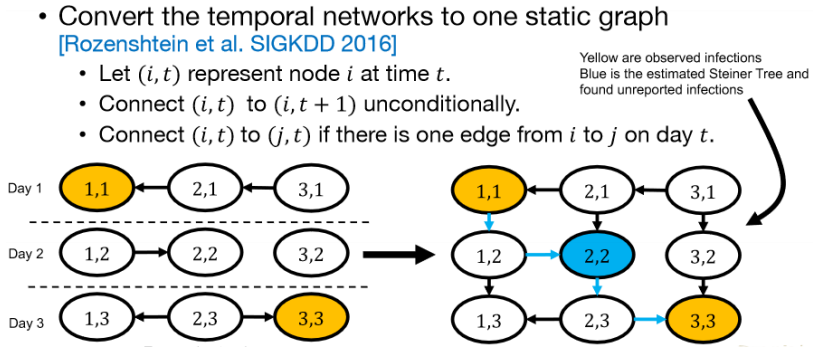


Figure 10: *Steiner Trees can be extended to temporal networks to find unreported infections.*

Steiner Trees allow epidemiologists to recognize patterns and identify what kinds of people are being unreported. Such a problem is common in hospitals.

3.4 Approach 4: NetFill

Missing infections can sometimes be derived based on seeds as shown in Figure 11. Because of this, it intuitively makes sense that our seeds are impacted by missing infections and can depend on whether we can recognize missing infections.

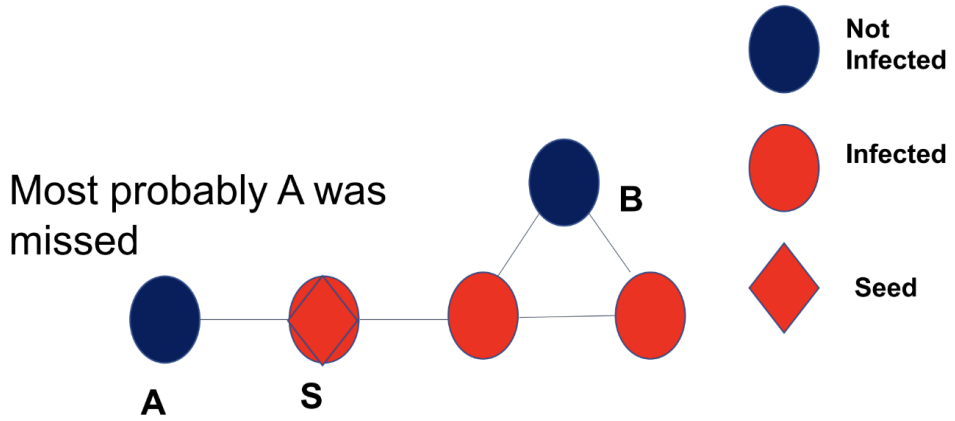
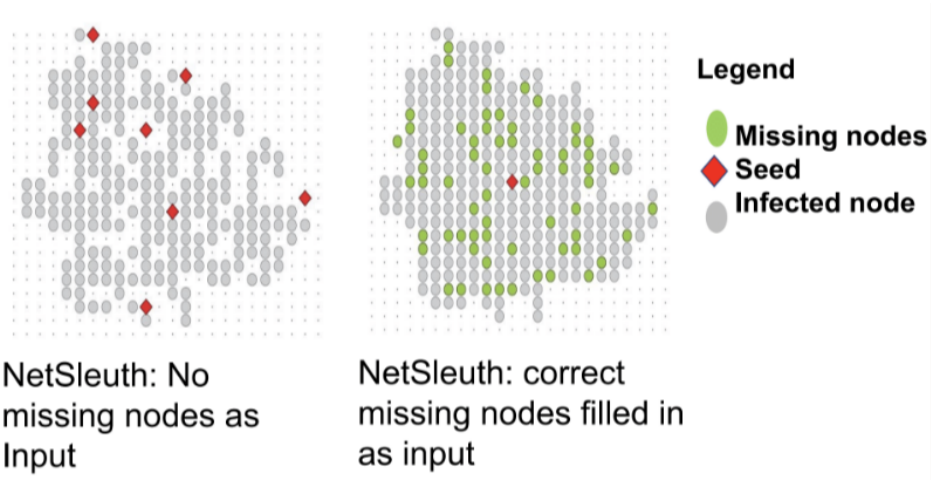


Figure 11: *We can see that missing infections can be detected based on their relationship to seeds. The seed S is directly connected to A and was able to infect both neighbors of B. Therefore, it is likely that A was a missed infection.*

NetFill is the idea of filling in gaps within a network to accurately trace back the seed of a network. This is done in the following steps:

1. Find starting points given missing nodes
2. Find missing nodes given starting points
3. Iterate above steps until convergence

The two ideas directly feed into one another and can be done in the NetSleuth algorithm [10].



4 Approach 5: Using Graphical Methods

Graphical methods may serve as a robust tool for modeling the dynamics of an epidemic, especially in situations where there exists a high probability of missing infections or "latent spreaders" within a population. Essentially, these models use graphical representations such as networks or graphs to model the interactions between various entities to infer infection, recovery, exposure and susceptibility rates.

Latent Spreaders and Activation Probability

One of the key hurdles in epidemiological modeling is accounting for latent spreaders, or people who are infected but asymptomatic or do not get tested and, hence, are not included in official counts. In graphical models, latent spreaders can be represented as nodes with particular attributes that indicate their potential for spreading the infection [9].

The probability of activation in the presence of latent spreaders can be examined through the PALS model (Refer to Figure 12). The model analyzes each individual infection state through two metrics: their susceptibility, which is often the individuals specific characteristics such as age and medical status, and exposure status, which must be indirectly inferred through contact networks. Each neighbor in the model has their latent spreader state modeled as a Bernoulli random variable. An individuals probability of being exposed to the infection is encoded by a Beta variable parameterized by the contagious status of its neighbors. These parameters get used to determine the individuals exposure state and infection state [9].

Putting the entire process together for individual i:

- For each neighbor $j \in n(i)$

  Draw the spreader state: $z_j \sim Bernoulli(\sigma(u^T x_j))$
- Draw the probability of exposure:

  $\theta i | z_i \sim Beta(1 + \sum_{n(i)} z_j, 1 + \sum_{n(i)} 1 - z_j)$
- Draw the exposure state: $\eta_i | \theta_i \sim Bernoulli(\theta_i)$
- Draw the infection state: $y_i | x_i, \eta_i \sim Bernoulli(\sigma(w^T x_i))$

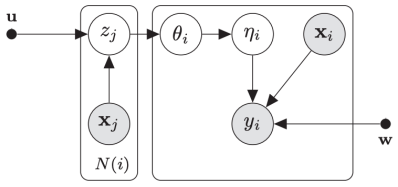


Figure 12: *PALS model depicting an individual i probability of infection. Probability is directly influenced by patient's characteristics and their neighbors' spreader states.*

By using the PALS model, we can gather key insights into how a susceptible individual became infected when interacting with a latent spreader.

Missing Infections via Contact Tracing

Using **Hidden Markov Models (HMM)**, we can model an individuals likelihood of being infected. HMM is a statistical model that depicts the progress of external events (the 'symbol') based on internal factors (the 'state') [15]. These internal states are not observable. The model uses two stochastic processes, the hidden states processed behind the scenes and the observable events processed at the forefront [15].

Identifying missing infections can be modeled using a HMM where each individuals likelihood of infection needs to be modeled at each time step. These time steps will be defined by the infection probabilities, the belief state, and their contact network history. The contact history includes observations on test results on part of the community, the presence of symptoms and physical connections between individuals. These observations are stored in a compressed matrix format to avoid privacy issues and memory constraints. Using the observations and given conditional probabilities, each individuals infection probabilities (the external 'state') are updated. A rate of recovery is also determined and those probabilities are applied to the population [13].

$$p_i^{t+1} = 1 - \prod_j \left(1 - \sum_k p_j^t \gamma^k \gamma^k C_{t-k}[i,j](1 - p_i^t \gamma^k)\right)$$
$$= 1 - \prod_j \left(1 - p_j^t \left(\sum_k \left(\gamma^{2k} C_{t-k}[i,j]\right) - p_i^t \sum_k \left(\gamma^{3k} C_{t-k}[i,j]\right)\right)\right)$$

Figure 13: *Probability of infection, where γ is the rate of decay and C is the decayed contact history*

Through this approach shown in Figure 13, the HMM is able to accurately find missing nodes in a population and predict the progression of the infection through the network.

References

[1] N. Bock, P. Jensen, B. Miller, and E. Nardell. Tuberculosis infection control in resource-limited settings in the era of expanding HIV care and treatment. 196:S108–S113.

[2] S. R. Cole, H. Chu, and S. Greenland. Maximum likelihood, profile likelihood, and penalized likelihood: A primer. 179(2):252–260.

[3] P. D. Grüwald. *The Minimum Description Length Principle*. The MIT Press, 2007.

[4] M. H. Hansen and B. Yu. Model selection and the principle of minimum description length. 96(454):746–774. Publisher: Taylor & Francis _eprint: https://doi.org/10.1198/016214501753168398.

[5] F. K. Hwang and D. S. Richards. Steiner tree problems. 22(1):55–89. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230220105.

[6] J. Leskovec. Cs224w machine learning with graphs— home, 9 2023.

[7] C. Ling, J. Jiang, J. Wang, and Z. Liang. Source localization of graph diffusion via variational autoencoders for graph inverse problems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 1010–1020, New York, NY, USA, 2022. Association for Computing Machinery.

[8] Z. Liu, G. Wan, B. A. Prakash, M. S. Lau, and W. Jin. A review of graph neural networks in epidemic modeling. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6577–6587, 2024.

[9] M. Makar, J. Guttag, and J. Wiens. Learning the probability of activation in the presence of latent spreaders. 32(1). Number: 1.

[10] B. A. Prakash, J. Vreeken, and C. Faloutsos. Spotting culprits in epidemics: How many and which ones? In *2012 IEEE 12th International Conference on Data Mining*, pages 11–20. ISSN: 2374-8486.

[11] D. Shah and T. Zaman. Rumor centrality: a universal source detector. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '12, pages 199–210. Association for Computing Machinery.

[12] D. Shah and T. Zaman. Rumors in a network: Who's the culprit? 57(8):5163–5181. Conference Name: IEEE Transactions on Information Theory.

[13] G. Sharon, J. Ault, P. Stone, V. Kompella, and R. Capobianco. Multiagent epidemiologic inference through realtime contact tracing. *Autonomous Agents and Multiagent Systems*, 2021.

[14] M. Worobey, J. I. Levy, L. Malpica Serrano, A. Crits-Christoph, J. E. Pekar, S. A. Goldstein, A. L. Rasmussen, M. U. G. Kraemer, C. Newman, M. P. G. Koopmans, M. A. Suchard, J. O. Wertheim, P. Lemey, D. L. Robertson, R. F. Garry, E. C. Holmes, A. Rambaut, and K. G. Andersen. The huanan seafood wholesale market in wuhan was the early epicenter of the COVID-19 pandemic. 377(6609):951–959. Publisher: American Association for the Advancement of Science.

[15] B.-J. Yoon. Hidden markov models and their applications in biological sequence analysis. *Current genomics*, 10(6):402–415, 2009.