Homework 2: Models, Dynamics, Construction, Inference

Released: Sept 13, 2022; Due: **5pm ET**, Sept 27, 2022

Georgia Tech
College of Computing
B. Aditya Prakash

CSE 8803 EPI Fall 2022
Student NAME: XXXXXXXX
Student GTID: XXXXXXXX

Reminders:

1. *Out of 100 points. 4 Questions. Contains 6 pages.*
2. *If you use Late days, mark how many you are using (out of maximum 4 available) at the top of your answer PDF.*
3. *There could be more than one correct answer. We shall accept them all.*
4. *Whenever you are making an assumption, please state it clearly.*
5. *You will submit a solution pdf* `LASTNAME.pdf` *containing your answers and the plots as well as a tar-ball* `LASTNAME.tgz` *that contains your code and any output files.*
6. *Please type your answers either in LaTeXdocument or in a separate file like a Word document and then convert it into a pdf file. Typed answers are strongly encouraged. Illegible handwriting may get no points, at the discretion of the grader. Only drawings may be hand-drawn, as long as they are neat and legible.*
7. *Additionally, you will submit one tar-ball* `LASTNAME.tgz` *that contains your code and any results files. Code and results for each question should be contained in a separate sub-directory (Eg:* `Q1`*) and there should be a* `README.txt` *file for each sub-directory explaining any packages to install, command to run the code files and location of the expected output. Please follow the naming convention* ***strictly****.*
8. *If a question asks you to submit code please enter the file path (Eg:* `Q1/Q-1.3.1.py`*) in the solution pdf.*
9. *You can download all the datasets needed for this homework from canvas files, you can check the information about the datasets in the* `README.txt` *file.*

1. (24 points) Metapopulation Model

We will model the SIR metapopulation model briefly introduced in Lecture 4 (Slide 5-6).

Let there be M regions with total populations $\{N_i\}_{i=1}^{M}$. We let $S_i(t), I_i(t), R_i(t)$ be the total population in respective component for region i at time t.

Assume that we σ_{ij} be the population flow from region i to j. Also, we assume the parameters β, γ are same across all regions.

Q 1.1 (6 points) Similar to Lecture 4 Slide 5, write down the equations to derive effective population sizes $S_i^{eff}(t), I_i^{eff}(t), R_i^{eff}(t)$ at time t from $\{S_j(t), R_j(t), R_j(t)\}_{j=1}^{M}$ and flow parameters σ_{ij}.

Solution:

Since SIR parameters β, γ are same across all regions we can write the change in S, I, R compartment sizes in 1 time-step as:

$$S_i(t+1) = S_i(t) - \beta S_i^{eff}(t) \sum_{j=1}^{M} \frac{I_j^{eff}(t)}{N_j}$$

$$I_i(t+1) = I_i(t) + \beta S_i^{eff}(t) \sum_{j=1}^{M} \frac{I_j^{eff}(t)}{N_j} - \gamma I_i^{eff}(t)$$

$$R_i(t+1) = R_i(t) + \gamma I_i^{eff}(t)$$

Q 1.2 (6 points) Instead of 1 time-step, derive equations for change in Δt time; i.e., derive the formulas for $S_i(t+\Delta t) - S_i(t), I_i(t+\Delta t) - I_i(t), R_i(t+\Delta t) - R_i(t)$. *Hint:* This is similar in nature to Q1.2 in HW 1.

Solution:

Q 1.3 (2 points) Set $\Delta t \to 0$ and derive the ODE equations for the metapopulation model.

Solution:

Q 1.4 (10 points) Implement the ODE of metapopulation model. The code should take the parameters β, γ, the flow parameters $\{\sigma_{i,j}\}_{i,j\in[1,M]\times[1,M]}$, total population $\{N_i\}_{i=1}^{M}$ and the initial population of each compartment for all regions $\{S_i(0), I_i(0), R_i(0)\}_{i=1}^{M}$ and `max_time` and output the population sizes of $\{S_i(t), I_i(t), R_i(t)\}_{i=1}^{M}$ till `max_time`.

We have provided a starter code in python script `metapop.py` that implements the model. You have to fill in the missing parts of the method `def ode(self, times, init, parms):` that involves calculating the effective population.

Once you have completed the method, run the script to generate the SIR plots for each population. Submit the completed code and the plots.

Note: You don't need to change other functions in the script, only fill in the code in the space indicated.

Note: In case you are not comfortable with python, you may translate the python script into your language of choice and complete the implementation of metapopulation model.

Solution:

2. (30 points) Viral propagation You are managing a network of devices inside your company connected to each other via a local intranet connection. Occasionally, some of the computers in the network get infected with a virus which can spread across the network as users share information with each other. You job is to simulate the spread of the virus under different assumptions on the diffusion mechanism.

Network dataset: Your dataset contains multiple networks $\mathcal{G} = \{G_1, G_2, \ldots, G_9\}$ between routers (this dataset is sourced from actual internet router dataset [1]. Each of the networks are snapshots of the connections made across routers on some day collected over 3 months. The network files are named as `network1.txt, network2.txt,..., network9.txt`.

Aggregating all networks: We aggregate all the 9 snapshots of the networks into a single network $G(\rho)$. The aggregation process is simple: For each pair of nodes (u, v) if at least ρ networks in $\mathcal{G}$ have an edge between them, we add an edge (u, v) to $G(\rho)$. Here ρ is a parameter of our model.

We also assume an SIS network model with usual parameters β, γ on the graph $G(\rho)$.

Q 2.1 (10 points) Decreasing ρ will increase the number of edges in the graph and make it more denser and connected. In class we saw that λ, the highest eigenvalue of the

[1] Collected by `https://service.uoregon.edu/TDClient/2030/Portal/KB/ArticleDet?ID=53820` and downloaded from `https://snap.stanford.edu/data/Oregon-1.html`

adjacency matrix is a good proxy for the connectedness of the graph. Let us observe this empirically for this dataset.

For each value of ρ in $\{1, 2, 3, \ldots, 9\}$, write code to construct $G(\rho)$. Find the largest eigenvalues λ of the adjacency matrix for each graph $G(\rho)$. Submit the plot between ρ and γ. How does λ change with increase in ρ?

Hint: Since the graphs have over 10,000 nodes, please consider using sparse representations of adjacency matrix and efficient built in methods to compute eigenvalues on the sparse graph. For example `scipy.sparse.linalg` module has efficient implementations to compute eigenvalues that enable you to perform the computations within fractions of a second.

Solution:

Q 2.2 (8 points) Set $\rho = 1$. Use the EoN package [2] to set up an SIS model on $G(\rho)$. Specifically use the `EoN.fast_SIS` function. Set $\gamma = 0.08, \beta = 0.001$ and `max_time` $T = 10$. Set node 1 as the only infected node at $t = 0$.

Now plot a curve with $I(t)$ on y-axis vs t on x-axis for $0 \leq t \leq T$ after simulating the model 50 times (similar to HW1 Q1.3). Now set $\beta = 0.0001$ and plot the $I(t)$ vs t curve. Do you notice any difference? Explain.

Solution:

Q 2.3 (10 points) Now set $\beta = 0.001, \gamma = 0.16$ and $\rho = 1$. We will try to reduce the effective strength of the graph by another method of iteratively removing nodes using the following rule:

Find the node with the largest degree among all nodes (in case of a tie, choose the node with the highest index). Then remove the node.

Using this rule find the number of node removals required to reduce the effective strength of the graph to less than 1. Also produce a plot of number of nodes removed in x-axis and effective strength on y-axis till effective strength reduces to less than 1.

Solution:

Q 2.4 (2 points) We will denote as K the number of nodes you had to remove in Q2.3 before effective strength decreased below 1. Consider the two graphs G_a and G_b where G_a is the initial graph you constructed by removing all K nodes and G_b the graph you constructed by removing $K - 1$ nodes (hence, effective strength is > 1). Simulate the SIS model on G_a and G_b with the same parameters by randomly choosing a single node to be infected at $t = 0$. Perform 100 simulations for both G_a and G_b for $T = 10$ steps. Plot $I(t)$ vs t for both the graphs.

Solution:

3. (30 points) Finding Culprits

Let's use the k-regular Cayley tree we used in the previous HW.

Q 3.1 (10 points) Let's consider the $k = 3$ tree with 22 nodes (see Figure 1). You can use edge list from `cayley.txt` to construct the graph.

[2] `https://epidemicsonnetworks.readthedocs.io/en/latest/EoN.html`

Implement a simple SI model using *EoN.fast_SI* or equivalent package in your language of choice. Set $\beta = 0.2$ in the SI model and run it on this graph. Let only the central node 0 be infected at $t = 0$. Run the SI model for 10 steps and record the set of nodes I which are infected at the end of 10 steps.

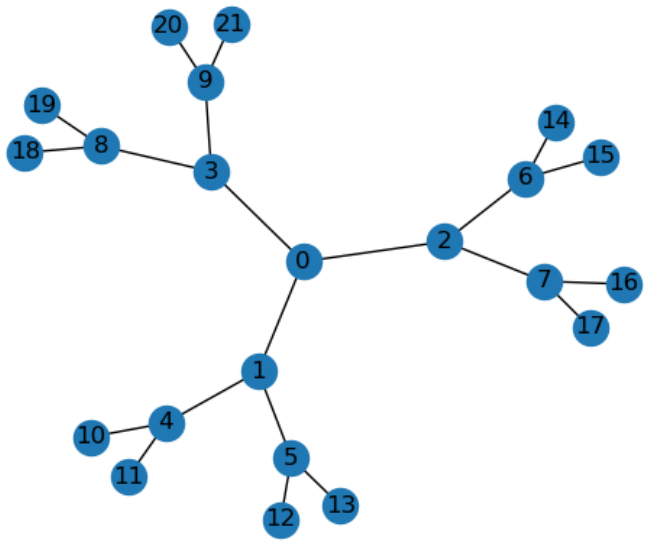


Figure 1: Cayley Tree

Repeat this for 100 times. Record these 100 sets of nodes as $X = \{I_1, I_2, \ldots, I_{100}\}$ in a file.

The file should contain 100 lines and each line i should contain the list of nodes infected I_i which are separated by a space.

Solution:

Q 3.2 (10 points) Now let's assume you are given the set X and you want to calculate the possible starting point of the epidemic. (we know it is node 0 but assume we don't know). Let's use a simple algorithm for this purpose. Let the inferred starting point be the centroid of the infected subgraph: $u_i = \text{Centroid}(I_i)$. You may use `networkx.center` function.

Find the accuracy of this algorithm i.e., find the fraction of time the algorithm infers node 0 as the center of subgraph I_i over $X = \{I_1, I_2, \ldots, I_{100}\}$.

Solution:

Q 3.3 (6 points) Repeat the same steps as Q3.2 and Q3.3 except make node 1 (a point at depth 1) as the starting point. Compute the accuracy of the algorithm.

Solution:

Q 3.4 (4 points) Does the accuracy of the estimator change (comparing the answer you get in Q3.2 and Q3.3)? Show your result and give the answer. If so, what does it tell you about the estimator on this type of graph (strengths and weaknesses; this is slightly

an open-ended question). Do you think the rumor centrality metric we studied in class will probably perform better?

Solution:

4. (16 points) Steiner trees for Missing Infections

In this question we will explore the use of Steiner trees for finding missing infections. Steiner trees are classic objects used for many 'facility location' problems. As we know many infections go unrecorded in real-life, and it is helpful to understand which other nodes may be infected but not recorded (the missing nodes). In class we saw other types of methods for this problem. Here we will use the idea of minimum Steiner tree over the known infected nodes as a way to approximate the unknown infected nodes in the graph.

if $G = (V, E)$ is our undirected social network with edge weights, and a set of terminals $I \subseteq V$, then a Steiner tree is a tree in G that spans I (i.e., contains at least all nodes in I; it might contain extra nodes from V not in I too, but at least it has to span all of I).

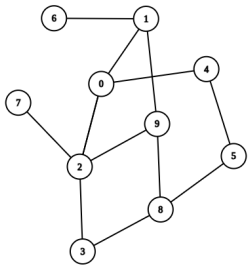
The minimum weighted Steiner tree, is the Steiner tree for a given I that has the least sum of the weights of all the edges included in it. There are several algorithms for finding such a Steiner tree given the set of terminals I and graph G. See this link for more details on the Steiner tree problem `https://en.wikipedia.org/wiki/Steiner_tree_problem`.

For the following question, assume that all the edges have equal weights.

Q 4.1 (6 points) Let's try to compute a couple of minimum Steiner trees by hand. Feel free to use whatever method you want (except of course just using a software package). These can be done with intuition as well. Just draw and show us the final answer.

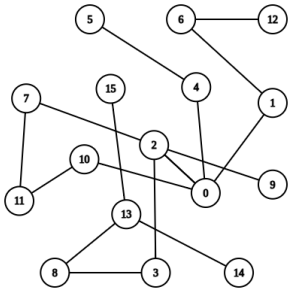
Q 4.1.1 (3 points) Terminal set = $\{7, 9, 3, 6\}$ for graph below.

Solution:



Q 4.1.2 (3 points) Terminal set = $\{12, 14, 9, 10\}$ for graph below.

Solution:



Q 4.2 (8 points) In the missing infections problem you are given a set of observed infections (the *known* set) K. For using Steiner trees to compute missing infections, the idea is very simple. Set K as the terminal nodes set I in the minimum weighted Steiner tree problem over the graph G. The resulting minimum weighted Steiner tree is a good rough approximation of how the disease might have spread. In particular the *extra* nodes you find in your min. Steiner tree are your missing nodes.

First generage the graph G as using the following function [3]:

`G = networkx.random_graphs.extended_barabasi_albert_graph(50, 1, 0.2, 0.1, seed=10)`.

Obtain an approximate minimum Steiner tree using this existing Networkx function [4].

Submit both a visualization and the adjacency list of the minimum Steiner tree for each of the following initial known infected nodes.

1. $K = \{10, 14, 3\}$
2. $K = \{10, 14, 3, 4, 2\}$
3. $K = \{10, 14, 3, 4, 2, 31, 49\}$
4. $K = \{10, 14, 3, 4, 2, 31, 49, 21, 25, 36, 43\}$

Plot the visualization in the pdf and submit the adjacency file as a .npy 2-D numpy array file.

Solution:

Q 4.3 (2 points) Briefly in 1-2 lines, what maybe the pros and cons in using a min. Steiner tree to identify missing infections? Suggest improvements for the same. This is an open-ended question so feel free to give intuitive answers.

Solution:

[3] Checkout `https://en.wikipedia.org/wiki/Barabasi-Albert_model` for more details on the network model.

[4] See documentation at: `https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.approximation.steinertree.steiner_tree.html`.