Dynamical Systems on Networks: Project Report

Athreya Chandramouli (2018121002)

Apr 2021

1 Introduction

In a lot of cases, it is sufficient to analyze dynamics in one or two dimensions. In these situations there are tools available to analyze trajectories and time evolutions. Here the general form of the ODEs is $\dot{x} = f(x, y)$ and $\dot{y} = g(x, y)$. These are interesting when f and g are functions of both x and y, i.e. x and y are "coupled". However, for more complex systems, there are multiple variables. It would be wasteful to assume that they are all coupled. To avoid this we can introduce an underlying network $G(V, E)$ given by an $n \times n$ (weighted) adjacency matrix A and model the dynamics as a combination of a self driving force and a sum of interaction forces over neighbours. More formally-

$$\dot{x_i} = f(x_i) + \sum_{j \in V} A_{ij} g(x_i, x_j)$$

Here $f(\cdot)$ and $g(\cdot, \cdot)$ are some general, possibly nonlinear, functions. For example, this can be used to model an extension of the Lotka-Volterra model to include multiple species. Another application that is very relevant today is to model an epidemic spread. The state at each vertex represents the probability of infection and there is an underlying contact network that affects how fast the rate grows.

2 Epidemic model

Here we'll explore the epidemic model in a bit more detail. The general form for the equation can be written as [1]-

$$\dot{x}_i = -Bx_i + \sum_j A_{ij} R(1 - x_i) x_j$$

Consider a simple interaction network of 4 people given in the figure. Here

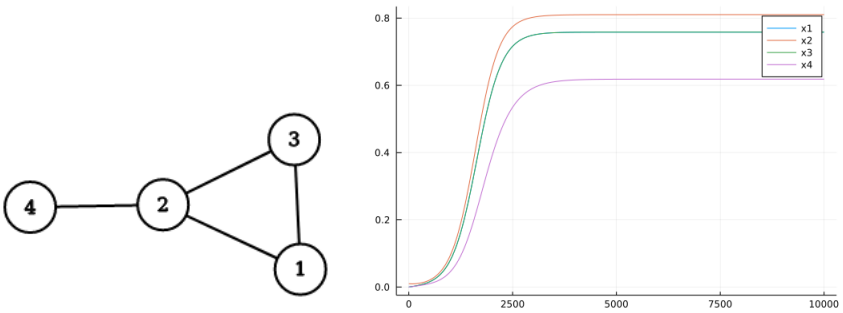

Figure 1: A simple interaction network

person 2 can be interpreted as sort of a mutual friend. This gives the equations:

$$\dot{x}_1 = -Bx_1 + A_{1,2}R(1-x_1)x_2 + A_{1,3}R(1-x_1)x_3$$

$$\dot{x}_2 = -Bx_2 + A_{2,1}R(1-x_2)x_1 + A_{2,3}R(1-x_2)x_3 + A_{2,4}R(1-x_2)x_4$$

$$\dot{x}_3 = -Bx_3 + A_{3,2}R(1-x_3)x_2 + A_{3,1}R(1-x_3)x_1$$

$$\dot{x}_4 = -Bx_4 + A_{4,2}R(1-x_4)x_2$$

where A is a parameter matrix representing the influence of each person on his neighbours, B is a scaling parameter representing self-influence and R is a scaling parameter for the influence of neighbours. The plot in the figure represents the time evolution of the probability of infection for parameters:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$B = 0.1$$

$$R = 0.2$$

at the initial condition

$$x_1 = x_3 = x_4 = 0, x_2 = 0.01$$

Thus we can see even if there is a low probability that one of the nodes are infected, if that node is in contact with other nodes the system converges to a steady state where all the nodes have more than 0.5 probability of getting infected.

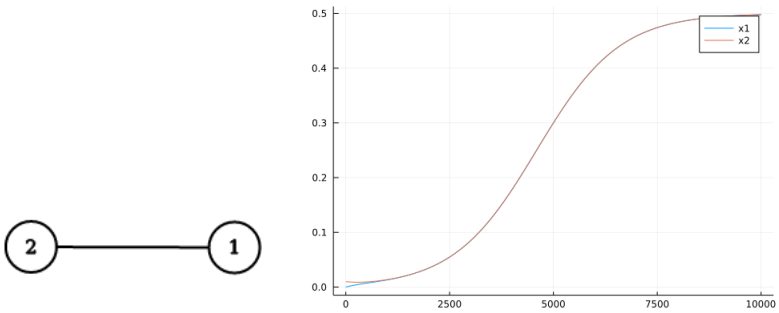

Figure 2: Incomplete interaction network

3 Incomplete Information

While such systems are difficult to solve by hand, they're still easy to solve with computers. The issue is that in practice, these networks may be extremely large and it is unrealistic to be able to model the entire network. In such a situation we assume a subgraph $G^{(s)}(V^{(s)}, E^{(s)})$ is known and given by the adjacency matrix $A^{(s)}$, where $V^{(s)} \subset V$ and $E^{(s)} \subset E$. Our task is to estimate true steady states x^* for vertices in $V^{(s)}$ despite only seeing $G^{(s)}$.

3.1 Naive method

The naive method to do this is to consider the contributions of the existing nodes in the network. For the previous example suppose we had only nodes 1 and 2 as in Figure 2. Using the naive method gives us the following equations:

$$\dot{x}_1 = -Bx_1 + A_{1,2}R(1 - x_1)x_2$$

$$\dot{x}_2 = -Bx_2 + A_{2,1}R(1 - x_2)x_1$$

Looking at the time evolution we see that this system converges to a steady state where both x_1 and x_2 are 0.5. Whereas in the original system x_1 converged to 0.758282 and x_2 converged to 0.810244. Thus we need a better way to approximate the dynamics.

3.2 Mean Field Approximation

According to [1], the missing vertex can be accounted for by adding one vertex to the subgraph, with value fixed to an estimate x_{eff} which is connected to each

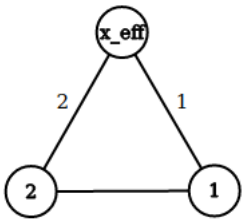

Figure 3: Mean Field approximation by adding an extra vertex

vertex with degree-weighted edges. This gives rise to ODEs of the form:

$$\dot{x}_i = f(x_i) + \sum_{j \in V^{(s)}} A_{ij}^{(s)} g(x_i, x_j) + (\delta_i - \delta_i^{(s)}) g(x_i, x_{\text{eff}})$$

The next step is to estimate x_{eff}. The basic approach is outlined in [2], and expanded on in [1]. Following their approach gives a general form

$$\dot{x}_{\text{av}} = f(x_{\text{av}}) + \beta g(x_{\text{av}}, x_{\text{av}})$$

where x_{eff} is the steady state of the above system. Here β is a parameter known as the resilience of the network. For undirected graphs it is given by $\beta = \sum_i \delta_i^2 / \sum_i \delta_i$ where δ_i is the degree of node i. Thus to approximate the steady states of the system, x_{eff} is derived by solving the above ODE and then the value of x_{eff} is plugged in to each of the ODEs for x_i.

The last piece therefore is to approximate β. If we know the actual network we can obtain the exact value. In fact [1] show that the estimate in the case of k-regular networks (where each vertex has degree k) is exact! They also provide ways to estimate the value of β when the method of sampling from the actual network used is known.

For our example, we get the ODEs:

$$\dot{x}_1 = -Bx_1 + A_{1,2} R(1 - x_1) x_2 + R(1 - x_1) x_{\text{eff}}$$

$$\dot{x}_2 = -Bx_2 + A_{2,1} R(1 - x_2) x_1 + 2R(1 - x_2) x_{\text{eff}}$$

To estimate x_{eff} we need to find the steady state of:

$$\dot{x}_{\text{av}} = -Bx_{\text{av}} + \beta R(1 - x_{\text{av}}) x_{\text{av}}$$

For simplicity, lets assume that the degree sequence of the original network is known. Hence we can find that $\beta = \frac{2^2+3^2}{2+3} = 2.6$. Solving the system with a program yields $x_{\text{eff}} = 0.8076925671636033$. Using this value in the above ODEs and solving them computationally we get $x_1^* = 0.765715$ and $x_2^* = 0.826455$.

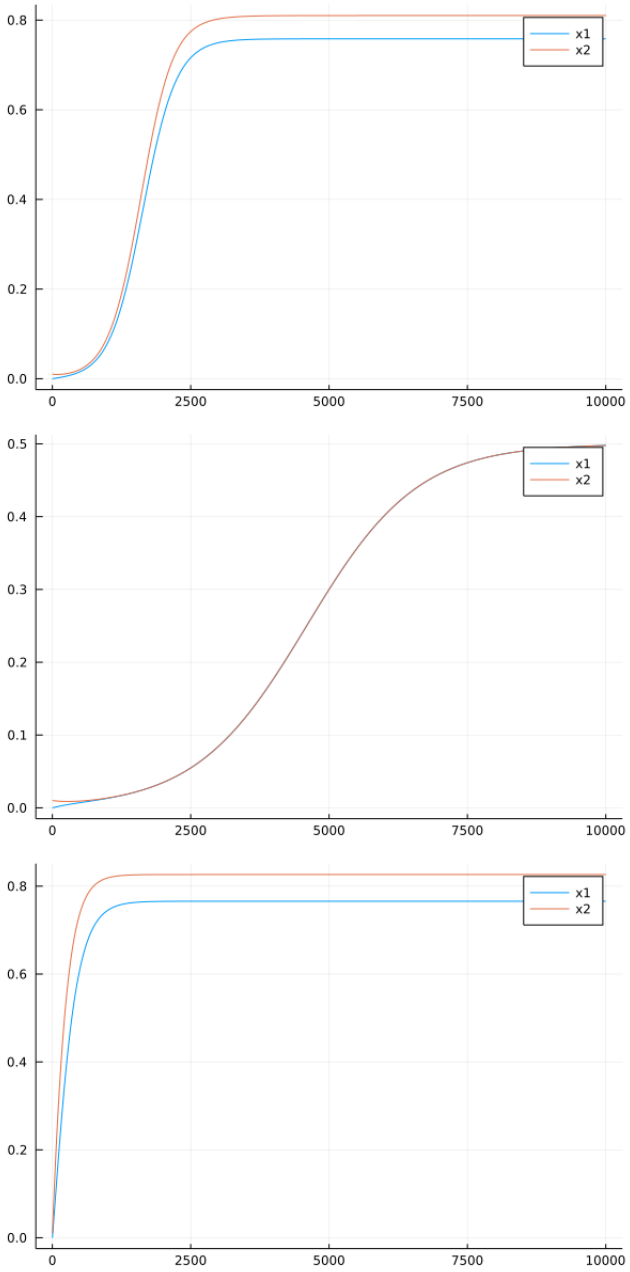

Figure 4: Top: Evolution of the actual system, Middle: Evolution of the naive approximation, Bottom: Evolution given by Mean Field Approximation

This estimate has a squared error of 0.00031808537082434965. The comparison between the time evolutions can be seen in Figure 4. Thus we can see that the mean field approximation does a decent job in estimating the steady state of the system.

4 Extensions

This section has some potential extensions to the network model. While these concepts are well explored individually, I am not sure if they've been applied to analysing nonlinear dynamics on networks.

4.1 Hypergraphs

Lets look at the Prey-Predator model once again. Consider the following situation: there is a single predator P that preys on two species A and B. Here if P, A and B are present in the system, the inhibiting influence of P on B's population is actually lower due to the presence of A than if the system had only P and B. Such details cannot be captured using a network. For such a situation we propose modelling the system using a hypergraph.

A hypergraph (V, E) is a set of vertices V and hyper-edges E where each $e \in E$ is a subset of the vertex set. Let $w : E \to \mathbb{R}$ be the weight function that returns the weight of each edge. Define $E_{i,k} = \{e \in E : x_i \in e,\ |e| = k\}$. Let x_{e_i} be the ith vertex in edge e. Then the general form for an ODE in such a system is of the form:

$$\dot{x}_i = f(x_i) + \sum_{k=1}^{n} \sum_{e \in E_{i,k}} w(e) g_k(x_{e_1}, \cdots, x_{e_k})$$

Here f and g_k(for $k = 0, \cdots, n$), are some general, possibly nonlinear, functions. For instance, in the previous example, a size two edge between P and B might have $w(\{P, B\}) = 1$ but a size three edge between P, A, and B might have $w(\{P, A, B\}) = 0.8$.

4.2 Other Graph properties

In the current model we only consider the influence of neighbours. Thus this model is inherently local. However there are other macro properties in the network. For instance in the case of the epidemic model, we might be interested in seeing how close a person is to other people in the network in terms of network distance. We also might be interested in people who might not only be high risk but have the potential to drastically affect other people in the

network. Such statistics are captured by graph centrality metrics. For instance, Closeness Centrality measures the former quantity. It is defined as the reciprocal of the sum of shortest network distances from the vertex to each vertex in the component:

$$C(v) = \frac{1}{\sum_y d(v, y)}$$

The latter can be captured by a metric called Betweenness Centrality. It is defined as the number of shortest paths passing through the vertex:

$$B(v) = \sum_{s \neq r \neq v \in V} \frac{\sigma_{s,r}(v)}{\sigma_{s,r}}$$

where $\sigma_{s,r}(x)$ is the number of shortest paths from s to r passing through x and $\sigma_{s,r}$ are the number of shortest paths from s to r. More sophisticated metrics can be designed for particular usecases. For example, in the epidemic model it is important to capture vertices that lie on the shortest paths from infected vertices. For this [3] proposed Percolation Centrality. It is a weighted version of Betweenness Centrality where the weights are dependent on the percolation of the source and destination nodes:

$$PC(v) = \sum_{s \neq r \neq v \in V} \frac{\sigma_{s,r}(v)}{\sigma_{s,r}} w_{v,s,r}$$

where,

$$w_{v,s,r} = \frac{R(x_s - x_r)}{\sum_{v \neq s \neq r} R(x_s - x_r)}$$

Accounting for such measures while computing the steady state of the system might help in understanding the dynamics better.

References

[1] Chunheng Jiang, Jianxi Gao, and Malik Magdon-Ismail. True nonlinear dynamics from incomplete networks, 2020.

[2] Jianxi Gao, Baruch Barzel, and Albert-László Barabási. Universal resilience patterns in complex networks. *Nature*, 530(7590):307, 2016.

[3] Mahendra Piraveenan, Mikhail Prokopenko, and Liaquat Hossain. Percolation centrality: Quantifying graph-theoretic impact of nodes during percolation in networks. *PLOS ONE*, 8(1):1–14, 01 2013.