Writeup of Model Polsci427s

Garrett Allen, Emil Zakarian, Kishan Gandham

November 17, 2021

1 Abstract

The spread of misinformation across communities and platforms (Twitter, Facebook, etc.) has garnered considerable attention in recent years due to the insidious consequences of fake news. Beyond the grand threats that such misinformation can and has had to real world instances such as COVID denialism and hysteria surrounding election interference, studying the impact of a hoax across a set network can be of paramount importance to understanding spread. In this study, we use existing literature on epidemic spread to model the spread of a hoax across a network of set agents with a planted misinformed seed. By measuring the change in 'misinformation scores' from start to model convergence, our analysis attempts understand how a singular seed misinformation spread model alters edge/node dynamics and network structure.

2 Background

This model was created with the goal of simulating the spread of misinformation from a centralized source through a network of agents. The motivation for this project is rooted in the recent increase in discussion of misinformation spread and how social media has eased this spread. With political polarization at an all-time high according to the Pew Research Center [1], the spread of strongly polarized misinformation becomes extremely important in making political extremes irreconcilable. Nikolov et al. [2] find that partisanship is highly correlated with sharing and engaging in more misinformation; both on the American political left and right, but especially so on the right. Misinformation is also inherently contagious. An MIT study found that false news stories are 70 percent more likely to be retweeted on Twitter than true stories. Understanding what factors might affect misinformation contagion could give us a basis to combat this spread [3].

More critically, the spread of misinformation has become an existential threat to truth in the digital age. With platforms such as Twitter, Facebook, Instagram, and Reddit at the forefront of online social interactivity, they have become conduits of information spread and, as a result, have become sources of misinformation spread as well. Modeling the spread of misinformation in an abstract setting can help us determine how to prevent the spread of falsehoods on these and other platforms.

3 Existing Models

3.1 Information Diffusion

There are a couple dominant theories towards discussing the spread of information across a network. Kempe et. all discuss the two main conceptions [4]. First and foremost, is the Independent Cascade Model which features a recursion in which agents in the network with the virus or hoax recursively infect their neighbors. Key to this model is that in each discrete step an agent in the model with said virus or hoax will attempt to 'infect' one of its connections in the network. Beyond this, there also exists a linear threshold model which features a random threshold needed to get infected for each node that is based upon the weighted sum of the edges or connections to that node. With no standard for computational modeling of misinformation spread, understanding the different mechanisms for misinformation spread is critical towards developing an effective simulated model of spread.

3.2 Epidemic Modeling

There have been myriad substantive efforts towards modeling the spread of misinformation in online social networks (OSNs). One such effort has been epidemic modeling where the spread of misinformation and fake news across an OSN has been equated to the spread of a disease across a community. Daley and Kendall (1964) were some of the first to propose an information cascading model (the DK Model) where they grouped the agents in a network into three categories: ignorant, spreader, and stifle — those who are unaware or do not care, those who spread the hoax, and those who are attempting to stamp out the misinformation [5]. Following the DK Model, there have been other attempts towards mathematically constructing a spread. Pastor-Satorras and Vespignani (2000) [6] describe misinformation spread in a structured scale-free network — networks which follow a power law related to their nodes and edges. Other such models have attempted to assert a difference based upon connectivity differentials in the nodes and edges: Eguiluz and Klemm (2002) [7] highlight the difference between randomly wired networks where even a relatively impotent hoax or virus still has some degree of spread and compare these networks to networks with local structure. The former tends to display behavior where the intrinsic quality of the virus or hoax has to pass some finite threshold in order to take hold in the network. The SIR model provides more foundational basis —

it presumes that individuals all have approximately the same connections and exist in three states: susceptible to contraction, infected, or removed (dead and out of the network). While wildly assumptive, the SIR model, as postulated by Reed and Frost in the 1920s, and explained by Newman (2002) [8] gets at characteristics of human behavior that are categorically difficult to model in spread situations. There are those who are susceptible to misinformation, those that have such misinformation, those that are spreading or starting said hoax, those that know such information is false and are willing to break connections with other individuals in their network because of it, and those that actively try and fight back against the hoax.

3.3 OSN incompatibility with computational study

While fundamental to the study of misinformation spread, such models prove out of date with the modern spread phenomena on OSNs. Algorithmic-driven spread features built in factors that are unknown to the public or are being discovered at present. Empirical analysis, as aforementioned, highlights how fake news and falsehood is more likely to spread than truth perhaps due to the novelty of the information [3]. Nonetheless, existing epidemic models of spread seemingly did not align with how OSNs might work because of their opacity. While topological data analysis and studies on these platforms themselves with API access might prove useful, due to a lack of suitable information disclosure between these platforms and the public/research, computational modeling has become all the more challenging. Therefore, rather than attempting to model spread in an online situation, analysis of viral hoax spread across a community proves more fruitful and makes the existing models more useful to study.

4 Model Design and Assumptions of Model

4.1 Assumptions

Our model operated under a few key assumptions about how people react under the spread of misinformation. Below, we list each, along with reasoning as to why we thought these assumptions made sense.

1. Individuals are more likely to believe information told to them if they are closer to the person that tells them it is true. (research here).

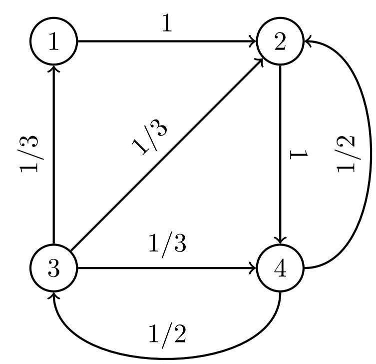
2. There exists some threshold where if two people differ in ideology/beliefs too much, they stop interacting with each other. We thought that this is pretty reasonable, as while some people are friends with those very ideologically different than them, these examples are typically very rare. Additionally, as people grow apart in beliefs, it becomes harder for them to be close friends, largely due to differing interests and an increasing lack of respect for the other person's beliefs.

3. Individuals find some hoaxes more believable than others, particularly information that is novel. (research).

These assumptions guided the design of our model heavily, and we believe that each claim is quite reasonable to assume for our model.

4.2 Mathematical Setup of Model

Our model can be thought of as a weighted graph with agents $P_1, P_2, \ldots, P_n$, where each P_i has a Weightlist W_i that gives the weights of the connections between all other P_j in the model. Each W_{ij} can be thought of as how "close" P_i is to P_j, with $W_{ij} = 0$ denoting that the two individuals do not know each other (and thus have no edge) and $W_{ij} = 1$ indicating a very close relationship between P_i and P_j The Weightlists are symmetric (that is, $W_{ij} = W_{ji}$ and each W_{ij} exists in $[0, 1]$. For convenience, $W_{ii} = 0$, so that later our rule for updating misinformation scores is more convenient. A picture of the possible distribution of weights is given below, where no edge represents a 0 weight.



Additionally, there exists a list S which keeps track of the misinformation score S_i for each individual in the model. Each S_i exists in $[0, 1]$, with $S_i = 0$ denoting that the individual P_i has never heard of the hoax, and a value of $S_i = 1$ denoting that they fully believe it to be true, and they spread it to others. FInally, there exists several hyperparameters that have a high degree of control over the model. p is the believability of hoax, which a value of 1 indicating that the hoax is very likely to be believed, and a value of 0 indicating that the hoax is unbelievable by anybody in the model other than the seed. $pZero$ is the probability of a connection $W_{ij} = 0$, and in essence controls how "sparse" our network is. High values of $pZero$ result in networks where very few people know each other (such as a group of individuals in a Facebook group), while low values of $pZero$ represent more tightly connected networks (such as a friend/acquaintance group) where everyone knows each other to some degree. There also exists a closeness parameter $c > 0$, which gives us control over how high the values in W_i are by applying the inverse CDF method on an exponential distribution. High values of closeness result in close networks where everybody knows each other, while low values of closeness result in networks filled with loose connections.

With all of the variables defined, we can now proceed with how the model runs. At the start of the model, the Weightlists W_i are initialized, and one P_i is selected to be the seed. We will denote this person P_s, and their value $S_s = 1$ at the start of the model; they represent the person who fully believes the hoax in our model, and seeks to spread it to other people in this network. All other values S_i are initially set to zero, indicating that they have never heard the hoax before.

Each run of the model, we call the "updateS" function, which updates each person's S_m as follows

$$S_{m*} = \frac{\sum_{i=1}^{n} W_i * S_i * p}{(n-1) * S_m} + S_m$$

The maximum update to S_{m*} is p if $W_i = 1, S_i = 1, p = 1$ for all i. This idea encapsulates the basic idea of our model; specifically, that whether or not someone believes a hoax is a function of how close they are to the person they are hearing it from, how believable the hoax is, and how willing other people are to spread it to them. If $S_{m*} > 1$, we manually set the value equal to 1 so that the S_i's remain bounded in the unit interval. This update runs every round of the model, and is the core of how misinformation spreads in our model.

After 3 runs of *updateS* running, we add in the function $cutTies$, which decides whether or not it is appropriate to cut ties between two individuals if they have very differing misinformation scores. This represents how people will stop talking to people who are very ideologically different than themselves, particularly if the difference is caused by misinformation. The 2-round "burn-in" period is used to ensure that everyone doesn't instantly cut ties with P_s, as we thought it was realistic that people would tolerate some amount of hoax-spreading before deciding whether or not to cut ties.

To decide whether to two individuals in the model should cut ties, we use a simple metric: If $W_{ij} > |S_i - S_j|$, then we do not cut ties between the two individuals. Otherwise, $W_{ij} = W_{ji} = 0$, and the two individuals no longer contribute to each other's updates. This is interpreted as two individuals ceasing to interact because they have become too far apart in their beliefs.

The model stops running when the difference between the $S_i^{k+1} - S_i^k < .01$ for all i in $1, \ldots, n$, where S^k denotes the S-list for run k. In short, the model stops when the change between everyone's S-score has become less than .01.

In summary, our model functions at its core by initializing the weights of everyone in the model (how close everyone is) and then picking one agent at random to be the seed. We then let the seed spread the hoax for 3 rounds before agents decide to start cutting ties between other agents, depending on if their difference in their beliefs is worth more than their friendship. The model stops when everyone has largely settled in their beliefs, either by believing the hoax entirely, never hearing about it at all, or changing so little that their beliefs have stagnated.

5 Results

Our results were, unfortunately, not very interesting. Almost all runs of the model produced outcomes where $S_i = 1$ for all i, or $S_s = 1$ and all other values of S_i are extremely low or zero. As p increased (the believability of the hoax) the model converged faster, but this is to be expected, as p just increases how fast S_i's increase. Our other results are listed below, as well as some of the limitations of our model.

6 Discussion

6.1 Absence of a Decay Mechanism

The lack of a decay mechanism for the S_i;s in our design likely resulted in the model converging at one of two extremes. Our design of the model did not feature a mechanism for the S_i's to reduce over time. With this mechanism lacking, misinformation scores, which our model was coded to make 0.0 at the time of model start either increased throughout the course of the model's run or stayed the same throughout the course of the mode's run. What our results revealed after significant trial and error with the model was that without a decay mechanism, the model approached one of two extrema. Either the majority of the connections in the model would rapidly cut ties with the seed (because p or $pZero$ were too low, resulting in slow spread) or the seed would exert significant influence over the connections in the network, and misinformation scores would rise until they reached 1. These results, while not very accurate to many scenarios, embody two specific real-life instances well: either initial spreaders are dramatically effective, causing many people to believe said hoax, or they are dramatically ineffective, resulting in loss of trust and cut ties.

6.2 Longer Convergence Time

The second interesting finding from our model had to do with pZero, our variable representing the overall connectedness of our network. Alterations in the overall connectedness of the network had abundantly clear differences in resultant misinformation scores and resultant cut ties. However, another important result is that when the probability of non-connection throughout the overall network goes up, the model takes a longer time to reach convergence. To determine convergence, our model had a built in threshold to end its run. When

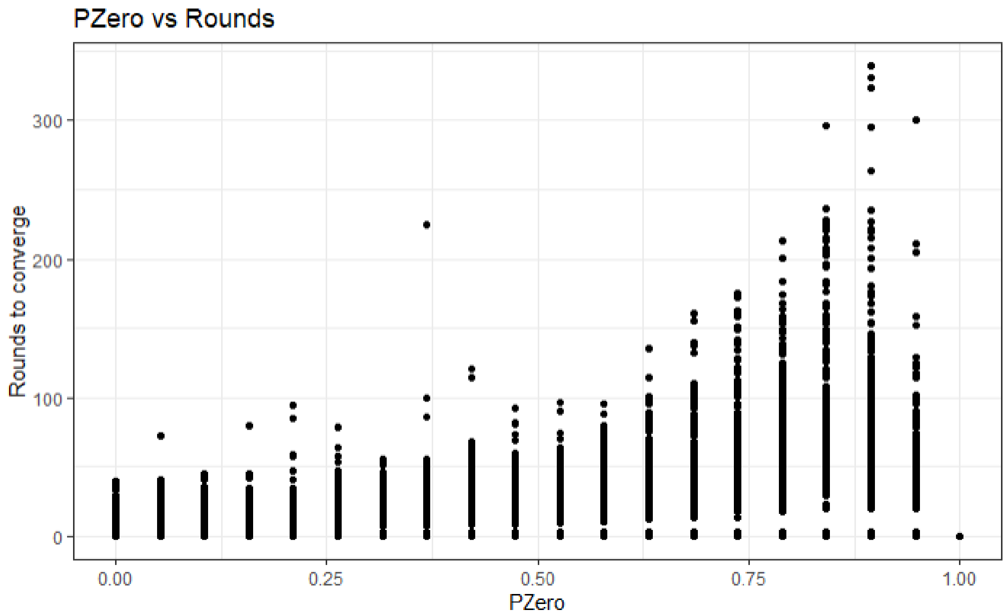


Figure 1: As pZero increases, the time to converge decreases

none of the misinformation scores of the agents in the network increased by 0.01 in a given iteration, the model was set to shut off. Therefore, the less connected the overall network is, the more changes continually happen in the misinformation scores before reaching the threshold and, importantly, the longer clustering takes to occur. This implies a quite reasonable real world result; specifically, sparse networks make it more difficult for hoaxes to spread. While our model is by no means complete, this seems intuitive and reasonable, and is a good indicator that our model is capturing some aspects of real world misinformation spread.

6.3 Variance in Average Misinformation Score

The final interesting result that we observed from our model was about the variance in the average misinformation score. As pZero increases, we found that

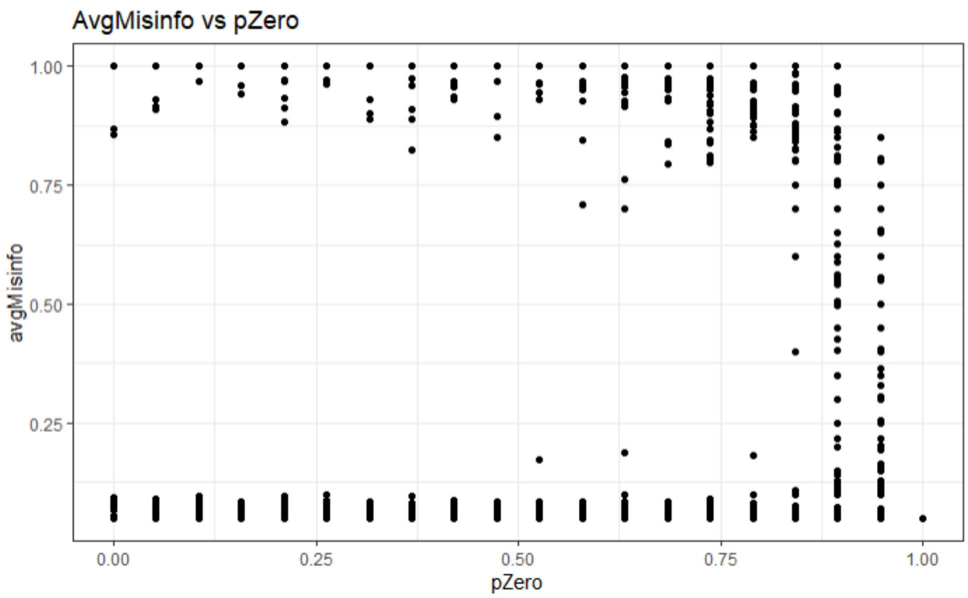


Figure 2: As pZero increases, avgMisinfo becomes more variable

the average misinformation was more variable than at low values of pZero. This implies that, in sparse networks (where people don't know each other well) how many people are infected with the hoax becomes more random, as it becomes harder for misinformation to travel between different clusters of people. This results in situations where it is impossible to actually infect some people given various starting positions, as there may exist clusters of people that are totally disconnected. While this result isn't very interesting in terms of our model, it does make sense in the real world; the spread of hoaxes is more unpredictable the less people are able to communicate with each other.

7 Model Limitations

Aside from the lack of a misinformation decay mechanism, there are some inherent limitations to our model. Firstly, agents in a real network would likely have some inner capacity to believe/disbelieve misinformation in isolation from how close they are to the source of misinformation. While hoaxes might have some inherent believability due to their proximity to real information, our model is limited in regards to the individual psychology and belief process of agents. Our model also did not allow for decay in connections outside of ties fully being cut between agents. A more sophisticated version would allow for both a natural decrease in misinformation in the absence of reinforcement, as well as a decrease but not full cutting of connections due to large differences in misinformation access.

8 Conclusion and Further Work

While our model did not accurately model the phenomena we wanted to study, it is a good start for creating a more fleshed out, realistic computational model for this subject. There are a few different things we could do to try to improve the model. For one, rather than cutting ties completely, we would make weights between people "decay" over time. This would be more realistic, as it would model that people become less close to individuals who they differ ideologically from. Additionally, we could make the misinformation scores S_i decay over time, such that S_i does not monotonically increase, but can vary over time. Finally, we could try to adjust our threshold for cutting ties and our methods for updating S_i scores, as it may be the case that the functions we are using are not well-suited to complex behavior. All of these would be interesting to explore in the future in order to make a model that is more accurate to the real world.

9 Appendix

9.1 Code

```
import math

import numpy

import igraph

import csv

from itertools import zip_longest


#independents

 #weight list for each agent

n = 150 # number of agents


def cdfExponential(x,scale):

  return 1 - math.exp(-(1/scale)*x)


#-math.log(.01)/closeness

def createWeights(n,pZero,closeness):

  personWeights = []

  for i in range (0,n):

    personWeights.append([0] * n)

  for i in range (0,n):

    for j in range (0,n):

      m = numpy.random.uniform(low = 0, high = 1)

      if(m  < pZero): #used for setting 0 values in open

        r = 0

      else:
```

```
        x = numpy.random.uniform(low = 0, high = 100)
        r =  round(1- cdfExponential(x,closeness),6)
        #random closeness value with less links of high closeness
      if(j != i):
        personWeights[i][j] = r #making weights symmetric
        personWeights[j][i] = r

      else:
        personWeights[i][i] = 0 #1 connection w/ self.
  return personWeights


#plotting initial graph from adjacency matrix
#G = igraph.Graph.Weighted_Adjacency(personWeights)
#layout = G.layout("lgl")
#igraph.plot(G,layout=layout)

#initalize sList

#setting initial believer of misinformation
#print("seed = " + str(seed))
#print("sList:" + str(-1) + str(sList))

def updateS(weightList, sList, p):
  updateList = sList.copy() #used for pointer stuff
  for i in range(0,len(weightList)):
    influenceFromContacts = 0
```

```
    for j in range (0, len(weightList)):

      influenceFromContacts+=sList[j]*p*weightList[i][j]



    if(sList[i] >= 1):
      sList[i] = 1
      updateList[i] = sList[i]
    else:
      update = influenceFromContacts/((n-1)) + sList[i]
      updateList[i] = round(update,6)
      if(updateList[i] >=1):
        updateList[i] = 1
  return updateList



def cutTies(weightList,sList,numCutTies):
  for i in range(0,len(weightList)):
    for j in range(0,len(weightList)):
      difference = abs(sList[i] - sList[j])



      if(difference >  weightList[i][j]
      and weightList[i][j] != 0):



        weightList[i][j] = 0
```

```
        weightList[j][i] = 0
        numCutTies[i]+=1
        numCutTies[j]+=1



#initialize stuff we care about


def percentOverThreshold(t, sList):
  overThresh = 0
  for i in range(0,len(sList)):
    if(sList[i] > t):
      overThresh +=1
  return overThresh / len(sList)


def avgMisinfo(sList):
  avgMisinfo = 0
  for i in range(0,len(sList)):
    avgMisinfo += sList[i]
  return avgMisinfo/len(sList)


def greaterThanValue(t, currSList, prevSlist):
  counter = 0
  for i in range(0,len(sList)):
    if(currSList[i] - prevSlist[i] < t):
      counter+=1
  if(counter == len(sList)):
    return True
```

```
  else:
    return False



t = .5
pZeroList = numpy.linspace(0,1,100)
closenessList = numpy.linspace(1,1000,100)
pList = numpy.linspace(0,1,20)

with open('misinfo.csv', 'w', encoding="ISO-8859-1",newline='') as file:
  write = csv.writer(file)
  write.writerow(("pZero","Closeness",
  "p","percThreshold","avgMisinfo","Clusters","Rounds"))

  for z in pZeroList:
    for c in closenessList:
      for p in pList:
        personWeights = createWeights(n,z,c)
        stopRounds = 1000
        seed = numpy.random.randint(0, n)
        sList = [0]*n
        sList[seed] = 1
        allSLists = []
        allSLists.append(sList)
        numCutTies = [0]*n
```

```
        for i in range (0,stopRounds):
          if(i > 2):
            cutTies(personWeights, sList,numCutTies)
          sList = updateS(personWeights,sList,p)
          allSLists.append(sList)

          if(greaterThanValue(.001, allSLists[i + 1],allSLists[i])):
            stopRounds = i
            break
        print([z,c,p])

        G = igraph.Graph.Weighted_Adjacency(personWeights)
        percThreshold = percentOverThreshold(.5,sList)
        avgM = avgMisinfo(sList)
        components = len(G.components())



        output = [z,c,p,percThreshold,avgM,components,stopRounds]
        write.writerow(output)
  print("done!")
        #write data to CSV as row: [0]
```

References

[1] "Political Polarization in the American Public." Pew Research Center - U.S. Politics amp; Policy, Pew Research Center, 12 June 2014, https://www.pewresearch.org/politics/2014/06/12/political-polarization-in-the-american-public/.

[2] Nikolov, D., Flammini, A., Menczer, F. (2021). Right and left, partisanship predicts (asymmetric) vulnerability to misinformation. Harvard Kennedy School (HKS) Misinformation Review. https://doi.org/10.37016/mr-2020-55

[3] Vosoughi, Soroush Roy, Deb Aral, Sinan. (2018). The spread of true and false news online. Science. 359. 1146-1151. 10.1126/science.aap9559.

[4] Kempe, David, et al. "Maximizing the Spread of Influence through a Social Network." Theory of Computing, Apr. 2014, pp. 105–147.

[5] Pittel, B. "On a Daley-Kendall Model of Random Rumours." Journal of Applied Probability, vol. 27, no. 1, Applied Probability Trust, 1990, pp. 14–27, https://doi.org/10.2307/3214592.

[6] Pastor-Satorras, Romualdo, and Alessandro Vespignani. "Epidemic Spreading in Scale-Free Networks." Physical Review Letters, 2001.

[7] Eguiluz, Victor M., and Konstantin Klemm. "Epidemic Threshold in Structured Scale-Free Networks." Physical Review Letters, 2002.

[8] Newman, M. E. J. "Spread of Epidemic Disease on Networks." Physical Review, 2002.