RumorFlow Dynamics: Modeling the Propagation of Rumors with Differential Equations

George Liu, Daniel Bao, Annika Joshi

1 Abstract

Rumors are an integral part of social interactions, spreading across various groups and online media. Through rumors, misinformation flows through society, influencing public opinion on issues varying in importance. This paper utilizes systems of ordinary differential equations (ODEs) to model the propagation of rumors over time. To do this, we determined that a population can be divided into four categories, based upon a person's knowledge of an existing rumor. Then, ODEs can be developed to represent the interactions between categories and, thus, effectively model rumor dynamics. Additionally, our technique acknowledges that rumors have varied relevance to society. To mimic this trait, we included an "importance value" between 0 to 1. Overall, our method successfully demonstrated the spread of rumors within a society. It also confirmed that more important rumors spread more quickly and reached a steady state sooner. This model can be implemented to analyze the dissemination of misinformation, underscoring its relevance in politics, sociology, and more.

2 Introduction

Misinformation, frequently disguised as rumors, has much sway over public discourse and societal decision-making. It rapidly infiltrates society - going beyond geographical and cultural barriers - whether it disseminates through online platforms or traditional word-of-mouth; such is the power of misinformation. Hence, the issue, particularly amidst social upheaval or political turmoil, wherein such propagation not only exacerbates tensions but also sows discord among communities; its contagious nature functions much like infectious diseases. Hence, in an effort to tackle this issue, researchers have pivoted towards mathematical modeling, drawing a parallel between the propagation of rumors and epidemiological models employed for disease spread analysis. For instance, deterministic models scrutinize the conditions that govern the spread of rumors; they examine factors like initial conditions and social network connectivity. With the use of ordinary differential equations (ODEs), the models can then illustrate how variables related to rumor dissemination develop over time. Thus, through performing stability and sensitivity analyses, we can identify crucial drivers of spreading rumors and evaluate intervention strategies. Other examples include stochastic models that introduce randomness: they mirror the uncertainty of real-world rumor propagation scenarios. Alternatively, complex network analyses focus on understanding how the structure of social networks influences the spread and persistence of rumors within communities. Ultimately, mathematical models, by illuminating the complex elements that propel the spread of rumors, provide valuable understanding to curb misinformation's detrimental impacts on society. These understandings equip policymakers: social media platforms and community leaders with tools for crafting targeted interventions – thus cultivating a populace ingrained in knowledge and resilience.

3 The Model

3.1 Rules of Interaction

We formulated a deterministic mathematical model to describe the dynamics of the spread of rumors. In the model, the total population is divided into four epidemiological classes of unaware $U(t)$, rumor spreaders $S(t)$, rumor repressors $R(t)$ and indifferent people $I(t)$. Unaware people $U(t)$ are people that have not heard of the rumor. Rumor spreaders $S(t)$ are people that are actively spreading the rumor. Rumor repressors $R(t)$ are people that are actively opposing the rumor and trying to stop its spread. Indifferent people are people that no longer care about the rumor $I(t)$.

The interactions between these classes of people are based on several underlying assumptions:
1. When unaware people interact with spreaders, they will either become spreaders or become indifferent.
2. When unaware people interact with repressors, they will either become repressors or become indifferent.
3. When spreaders interact with repressors, they will either stay a spreader, become a repressor, or become indifferent.
4. When repressors interact with spreaders, they will either stay a repressor, become a spreader, or become indifferent.
5. Spreaders and repressors can become indifferent after interacting with eachother. This rate is based on the importance of the rumor. A more important rumor means that people are less likely to become indifferent. Spreaders and repressors have the same probability of becoming indifferent.
6. Indifferent people will never go out of their way to sway someone to either direction.
7. People are generally less likely to be repressors than spreaders.

3.2 Variables and Parameters

We can use the above assumptions to construct variables and parameters to use to model the rumor spreading population dynamics.

Variable	Description
$U(t)$	Population size of unaware people
$S(t)$	Population size of rumor spreaders
$R(t)$	Population size of rumor repressors
$I(t)$	Population size of indifferent people
$N(t)$	Total population size
λ_S	Probability of unaware person turning into spreader
λ_R	Probability of unaware person turning into repressor
σ_S	Probability of repressor turning into spreader
ω_R	Probability of spreader turning into repressor
ψ	Importance of the rumor (Probability of indifference)

3.3 Equations

With the defined variables, we can then model each of the changes in the populations $U(t)$, $S(t)$, $R(t)$, and $I(t)$.

$$U + S + R + I = N \tag{3.3.1}$$

$$\frac{dU}{dt} = -US - UR \tag{3.3.2}$$

$$\frac{dS}{dt} = \lambda_S US + (\sigma_S - \omega_R - \psi)SR \tag{3.3.3}$$

$$\frac{dR}{dt} = \lambda_R UR + (\omega_R - \sigma_S - \psi)SR \tag{3.3.4}$$

$$\frac{dI}{dt} = (1 - \lambda_S)US + (1 - \lambda_R)UR + 2\psi SR \tag{3.3.5}$$

4 Interpreting the Model

4.1 Base Values for the Model

Variable	Value	Source
$U(t)$	1950	Assumed
$S(t)$	40	Assumed
$R(t)$	10	Assumed
$I(t)$	0	Assumed
$N(t)$	2000	Assumed
λ_S	0.5	[1]
λ_R	0.1	Assumed
σ_S	0.1	[2]
ω_R	0.1	[2]
ψ	Varied	

We modeled a population of total size 2000. The parameters are based on research from sources [1] and [2]. With these parameters, we will vary the importance value of the rumor and determine its effects on the populations. In this context, the closer the importance value is to 0, the more likely people are to stay invested, and the more important the rumor is considered. For example, a very polarizing political rumor would have less people being indifferent to it, resulting in a more important rumor.

4.2 Testing the Model

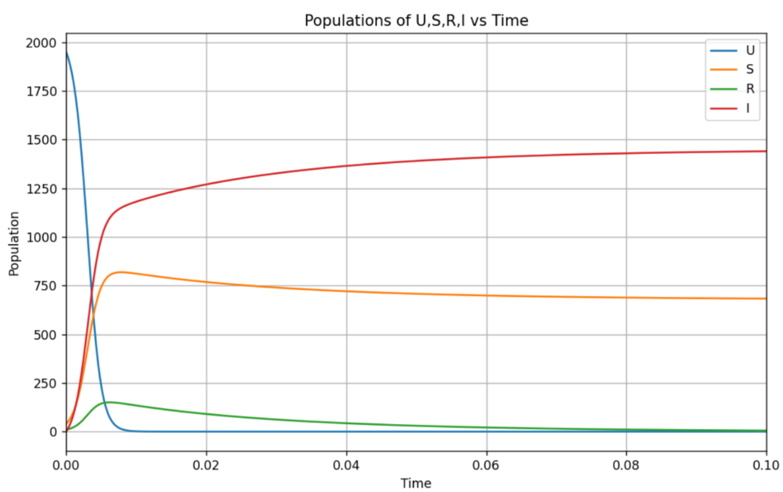

Figure 4.2.1: Graph of a population system with an important rumor ($\psi = 0.05$)

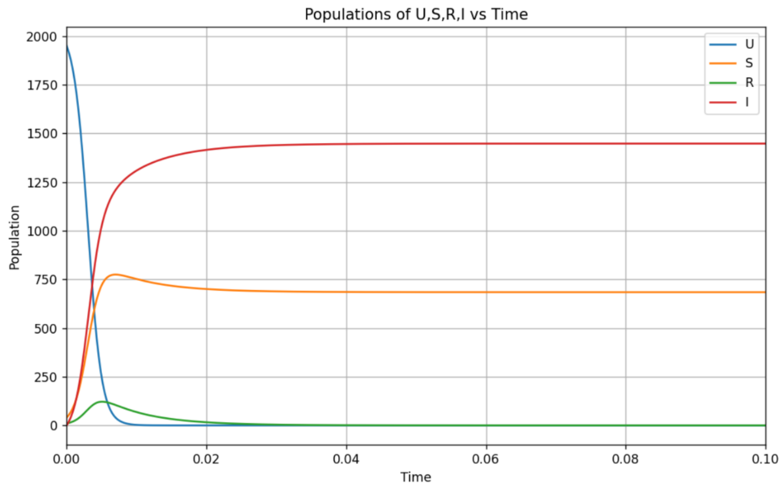

Figure 4.2.2: Graph of a population system with a moderately important rumor ($\psi = 0.4$)

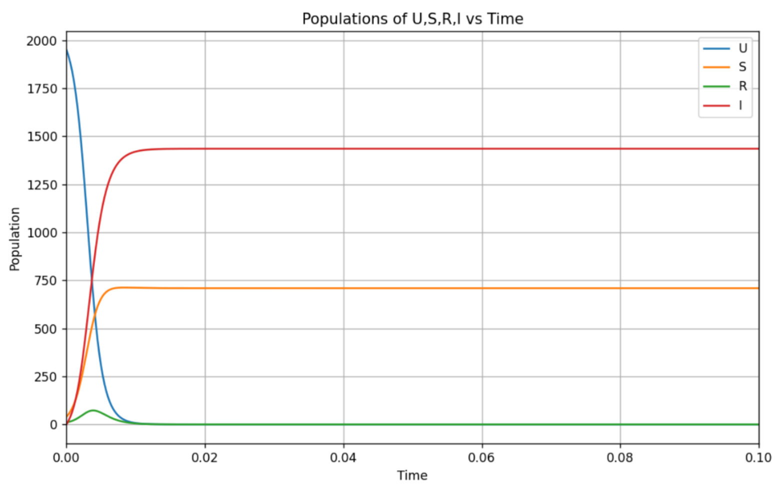

Figure 4.2.3: Graph of a population system with a trivial rumor ($\psi = 0.95$)

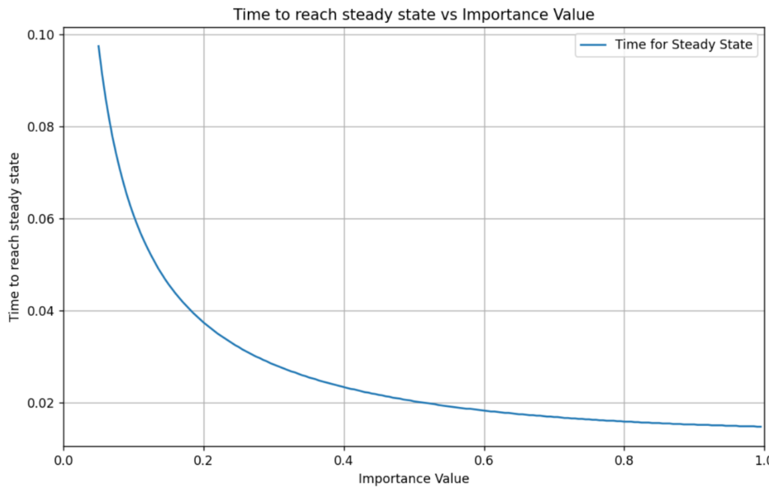

Figure 4.2.4: Time for system to reach steady state vs importance value

We implemented Euler's method with the Python script titled "eulers.py" provided in appendix 7.1 to estimate the different population sizes over time and generate figures 4.2.1-4.2.3 by setting the importance value to 0.05, 0.4, and 0.95 respectively. We then used the Python script titled "steady_state_importance.py" provided in the appendix to graph the time taken for the population to reach a steady state in relation with the rumor's importance value in figure 4.2.4.

As evident in the numerical solution graphs detailing the relationships between the time of rumor propagation and the proportion of populations allocated (unaware persons, rumor spreaders, rumor repressors, indifferent persons), we see that the importance of rumor is directly correlated to the longevity of impact of the rumor, that is to say, as the rumor is perceived to be more significant, its effective period of influence on the population proportion is greater, taking longer to reach a steady state (we assigned an arbitrary steady state value of $\frac{dI}{dT} < 400$). We define the steady state as the point in which the population stays relatively stable, and thus the proportion of populations indifferent as well. Thus, when the steady state is reached, the rumor is dead. This much resembles real-life expectations of the rumor life cycle: it begins with unverified news, grows while occasionally being fact-checked, and ultimately reaches a point of stability where the rumor can be referred to as being dead, with some lasting effects on the population of course.

Looking at figures 4.2.1-4.2.3, we can see that there is a general inverse relationship between the importance value and the lifespan of the rumor. Trivial rumors ($\psi = 0.95$) have a lifespan of 0.0150t, whereas important rumors ($\psi = 0.05$) have a lifespan of 0.0974t. In general, there is a negative exponential relationship between importance value and time taken to reach the steady state which we can see on figure 4.2.4. Notably, this negative exponential relationship results in scenarios in which even trivial rumors have problematic lifespans, with the lifespan of 0.0150t, almost $\frac{1}{6}$ of the lifespan of the important rumor, still being a relatively notable quantity.

In real-life applications, this finding holds implications across a span of unalike domains. In the context of social media platforms, breeding grounds for rumor propagation, platforms can use this insight into the importance of rumors in duration to ascertain their prioritizations of rumor mitigation efforts, focusing more on resources on debunking or containing rumors of high or moderate importance. Similarly, in emergency situations, where rumors can massively, and detrimentally, impact decision-making and responses, understanding this insight can inform communication strategies, ensuring accurate information and mitigating potential misinformation harm.

5 Conclusion

Rumors are a staple of human nature, lasting for millennia; as such addressing its propagation spillovers into addressing tensions in societal decisions and public discourse among others. Moreover, in interpersonal interactions and community settings, awareness of the importance of rumors in their longevity can guide interventions aimed at promoting critical thinking and skepticism, greatly aiding social coercion. Overall, this insight provided by the deterministic models underscores the importance of targeted and proactive approaches in combating misinformation. Our study does have limitations that can be addressed in future research. For example, certain people, such as celebrities and major athletes, have more influence over society and could convert unaware people into spreaders or repressors at a higher rate. Additionally, different groups may be more prone to spreading rumors. However, overall, by addressing the underlying factors driving the longevity of rumors, we, as a community, can better equip ourselves to safeguard against such and foster a more resilient community.

6 References

[1] Roberto, J. and Piqueira, C. (2010) Rumour Propagation Model: An Equilibrium Study. Mathematical Problems in Engineering, 2010, Article ID: 631357. https://doi.org/10.1155/2010/631357

[2] Chen, X., Wang, N. Rumor spreading model considering rumor credibility, correlation and crowd classification based on personality. Sci Rep 10, 5887 (2020). https://doi.org/10.1038/s41598-020-62585-9

7 Appendix

7.1 Code

Below is the file eulers.py which carries out euler's method to model the population dynamics. It also graphs each of the populations in respect to time.

```
#eulers.py
import numpy as np
import matplotlib.pyplot as plt
import sys

# Define parameters
a = 0.5 #lambda s
b = 0.1 #lambda r
c = 0.1 #sigma s
d = 0.1 #omega r

x = 0.95 #importance value

# Define initial conditions
U0 = 1950
S0 = 40
R0 = 10
I0 = 0

# Define time parameters
t0 = 0
T = 0.30 # Total time
dt = 0.0001 # Time step
num_steps = int(T / dt) # Number of time steps

# Initialize arrays to store results
U = np.zeros(num_steps)
S = np.zeros(num_steps)
R = np.zeros(num_steps)
I = np.zeros(num_steps)
time = np.zeros(num_steps)
```

```
# Set initial conditions
U[0] = U0
S[0] = S0
R[0] = R0
I[0] = I0
time[0] = t0

# Euler's method
for i in range(1, num_steps):
    dU_dt = -U[i-1]*S[i-1] - U[i-1]*R[i-1]
    dS_dt = a*U[i-1]*S[i-1] + (c-d-x)*S[i-1]*R[i-1]
    dR_dt = b*U[i-1]*S[i-1] + (d-c-x)*S[i-1]*R[i-1]
    dI_dt = (1-a)*U[i-1]*S[i-1] + (1-b)*U[i-1]*R[i-1] +
        (x+x)*S[i-1]*R[i-1]

    if (dI_dt < 400):
        print(i*dt)
        sys.exit()

    U[i] = U[i-1] + dt * dU_dt
    S[i] = S[i-1] + dt * dS_dt
    R[i] = R[i-1] + dt * dR_dt
    I[i] = I[i-1] + dt * dI_dt
    time[i] = time[i-1] + dt

# Plot results
plt.figure(figsize=(10, 6))
plt.plot(time, U, label='U')
plt.plot(time, S, label='S')
plt.plot(time, R, label='R')
plt.plot(time, I, label='I')
plt.xlabel('Time')
plt.ylabel('Population')
plt.title('Populations of U,S,R,I vs Time')
plt.legend()
plt.grid(True)

ax = plt.gca()
ax.set_xlim([0, 0.30])
plt.show()
```

Below is the file steady_state_importance.py which is used to calculate the time taken to reach the steady state at various importance values. It also graphs the time taken to reach the steady state in relation to importance value.

```
#steady_state_importance.py
import numpy as np
import matplotlib.pyplot as plt
```

```
importanceArr = []
steadyStateArr = []

for importance in np.arange(0.05, 1, 0.005):

    importanceArr.append(importance)

    # Define parameters
    a = 0.5 #lambda s
    b = 0.1 #lambda r
    c = 0.1 #sigma s
    d = 0.1 #omega r

    x = importance #importance value

    # Define initial conditions
    U0 = 1950
    S0 = 40
    R0 = 10
    I0 = 0

    # Define time parameters
    t0 = 0
    T = 0.10 # Total time
    dt = 0.0001 # Time step
    num_steps = int(T / dt) # Number of time steps

    # Initialize arrays to store results
    U = np.zeros(num_steps)
    S = np.zeros(num_steps)
    R = np.zeros(num_steps)
    I = np.zeros(num_steps)
    time = np.zeros(num_steps)

    # Set initial conditions
    U[0] = U0
    S[0] = S0
    R[0] = R0
    I[0] = I0
    time[0] = t0

    # Euler's method
    for i in range(1, num_steps):
        dU_dt = -U[i-1]*S[i-1] - U[i-1]*R[i-1]
        dS_dt = a*U[i-1]*S[i-1] + (c-d-x)*S[i-1]*R[i-1]
        dR_dt = b*U[i-1]*S[i-1] + (d-c-x)*S[i-1]*R[i-1]
        dI_dt = (1-a)*U[i-1]*S[i-1] + (1-b)*U[i-1]*R[i-1] +
            (x+x)*S[i-1]*R[i-1]
```

```
        if (dI_dt < 400):
            steadyStateArr.append(i*dt)
            break

        U[i] = U[i-1] + dt * dU_dt
        S[i] = S[i-1] + dt * dS_dt
        R[i] = R[i-1] + dt * dR_dt
        I[i] = I[i-1] + dt * dI_dt
        time[i] = time[i-1] + dt

# Plot results
plt.figure(figsize=(10, 6))
plt.plot(importanceArr, steadyStateArr, label='Time for Steady State')
plt.xlabel('Importance Value')
plt.ylabel('Time to reach steady state')
plt.title('Time to reach steady state vs Importance Value')
plt.legend()
plt.grid(True)

ax = plt.gca()
ax.set_xlim([0, 1.0])
plt.show()
```