A Stochastic Differential Equation Framework for Guiding Information Diffusion

Yichen Wang⋄, Evangelos Theodorou∗, Apurv Verma⋄, and Le Song⋄

⋄College of Computing, Georgia Institute of Technology
∗School of Aerospace Engineering, Georgia Institute of Technology

{yichen.wang,apurvverma, evangelos.theodorou}@gatech.edu, lsong@cc.gatech.edu

Abstract

The information content of a node in information networks is influenced by its neighbors in the networks. Recently there have been many works on modeling information diffusion, but few have integrated these models for online decision making. A framework for guiding information diffusion is critically important for understanding the vulnerabilities of these networks and designing good policies to suppress rumors and misinformation. Here, we propose a unified stochastic differential equation framework for modeling information diffusion over networks and designing the control policy to guide such diffusion. Our framework can handle noisy data, networks with time-varying edges and node birth processes. Using both synthetic and real world networks, we showed that our framework is robust, able to steer both stable and unstable information diffusion systems to desired states with faster convergence, less variance and lower control cost than alternatives.

1 Introduction

Online social and information platforms have brought to the public a new style of social lives: people use these platforms, such as Facebook and Twitter, to receive information, express their opinions and influence their peers. Nowadays, large-scale online user activity data with fine temporal resolutions are becoming increasingly available, which has fueled the increasing efforts in developing realistic representations and models as well as learning and inference algorithms to understand, predict and distill knowledge from information diffusion over networks [21, 9, 15].

However, most existing works focus on modeling and predicting information diffusion [9, 10, 14], and few have integrated these models for guiding such diffusion in a close-loop feedback fashion. In this case, a decision maker seeks a policy to determine the best external intervention such that the information contents can be guided towards a target state. This is significantly more challenging than traditional influence maximization or activity shaping tasks [21, 13] where the intervention is determined before the stochastic process unfolds, and they do not take into account the actual instantiation of the process. A framework for doing this is critically important for understanding the vulnerabilities of information networks and designing policies to suppress the spread of undesired contents. For instance, a network moderator may want to effectively contain the spread of rumors and misinformation, and a government agent may want to effectively suppress the spread of terrorist propaganda.

Intuitively, an effective framework for guiding information diffusion should take into account the current status, and integrate such feedback into the intervention policy, as shown in Figure 1. However, most models are based on probabilistic graphical models or point processes [33, 13, 11, 14]. It is not clear how to take into account feedback in such models and compute the best intervention policy based on these models. The problem of how to guide information diffusion in an optimal way remains largely unsolved [16].
In this paper, we provide a novel view of information diffusion models based on point processes, and reformulate them into stochastic differential equations. The novel reformulation allows us to significantly generalize existing information diffusion models, and plays a critical role in connecting the task of guiding information diffusion to that of stochastic optimal control often used in robotics and finance. As a result, we can bring in lots of tools from stochastic optimal control literature to address sequential decision making problems in information diffusion.

Interestingly, the information diffusion setting also introduces new challenges for the stochastic optimal control framework. Previously, stochastic optimal control has been mostly studying systems driven by Wiener processes and/or Poisson processes. Information diffusion problems require us to consider more advanced processes such as, multivariate Hawkes processes which are models for long term memory process and mutual exciting phenomena in social interactions [13, 14], and time-varying networks and node birth processes which are models for real world evolving networks [14]. Thus many of the technical results from stochastic optimal controls needs to be developed for information diffusion problems. Therefore, besides building important connection between information diffusion and stochastic optimal control, our paper also makes the following technical contributions:

- We provide a general way to formulate the information diffusion processes as Stochastic Differential Equations (SDEs) driven by Hawkes processes and Wiener processes, which captures the mutual excitation phenomena, and is robust to noisy data and stochastic disturbances.
- We extend the stochastic optimal control theory and derive generalized Ito’s lemma and HJB equation for SDEs driven by multi-dimensional point processes.
- We propose an efficient stochastic control algorithm which can guide information diffusion dynamics towards a target state while reducing the variance of the state. The algorithm can also deal with the challenging cases of time-varying networks and networks with node births.

Finally, using both synthetic and real world networks, we showed that our framework is robust with faster convergence, less variance than alternatives.

2 Backgrounds and Preliminaries

Related work. Most previous works in influence maximization [21, 4, 3, 5, 2] focus on selecting the nodes of information sources to maximize the spread of information in infinite time window, which is a different setting.
First, the state of each user is often assumed to be binary, either adopting a product or not. Second, there is typically no quantitative prescription on how much incentive should be provided to each user. Moreover, in most cases, a finite time window must be considered. For example, a politician would like to maximize his support by a million people in one week, rather than in fifty years. Recently, [12, 24] studied the control of epidemic process [27] that is modeled by a continuous and deterministic differential equation. However, these works neither consider the influence of discrete abrupt event nor the stochasticity in the system, which is important in social applications. The recent Hawkes process based information diffusion models [13, 7] overcome the limitation and consider how to design the baseline intensity to achieve desired steady state behavior for stable systems. However, they do not consider system feedback, and the variance of the dynamics can still be very large. In contrast, our method can incorporate system feedback and adaptively determine the control policy. As for related work in stochastic optimal control, the jump diffusion SDE [17] is closely related. However, most of the works [17, 26] focus on SDEs driven by Poisson processes. Significant generalizations are needed for our information diffusion setting.

**Point processes.** Our framework builds on mathematical tools from point processes, stochastic differential equations and stochastic optimal controls. Here we first provide necessary concepts of temporal point process [1, 6]. A temporal point process is a random process whose realization consists of a list of discrete events localized in time, \( \{t_i\} \) with \( t_i \in \mathbb{R}^+ \) and \( i \in \mathbb{Z}^+ \). Many different types of data produced in online social networks can be represented as temporal point processes, such as the times of opinion posting and tweeting. A temporal point process can be equivalently represented as a counting process, \( N(t) \), which records the number of events before time \( t \).

An important way to characterize temporal point processes is via the conditional intensity function \( \lambda(t) \), a stochastic model for the time of the next event given all the times of previous events. Let \( \mathcal{H}(t) = \{t_i | t_i < t\} \) be the history of events happened up to \( t \). Formally, \( \lambda(t) \) is the conditional probability of observing an event in a small window \([t, t+dt]\) given the history \( \mathcal{H}(t) \), i.e.,

\[
\lambda(t)dt := \mathbb{P}\{\text{event in } [t, t+dt]|\mathcal{H}(t)\} = \mathbb{E}[dN(t)|\mathcal{H}(t)],
\]

where one typically assumes that only one event can happen in a window of \( dt \), i.e., \( dN(t) \in \{0,1\} \). The function form of the intensity \( \lambda(t) \) is often designed to capture the phenomena of interests. Some commonly used forms in modeling information diffusion are [6, 13, 14, 23]:

(i) **Poisson processes.** The intensity \( \lambda(t) \) is independent of the history \( \mathcal{H}(t) \), but can be a constant, \( \lambda \) (homogeneous), or a time-varying function, i.e., \( g(t) \geq 0 \) (inhomogeneous).

(ii) **Hawkes processes.** The intensity models an excitation effect between events, i.e., \( \lambda(t) = \mu + \alpha \sum_{t_i \in \mathcal{H}(t)} \kappa_\omega(t-t_i) \)

\[
dN(t) = \mu + \alpha \sum_{t_i \in \mathcal{H}(t)} \kappa_\omega(t-t_i),
\]

where \( \kappa_\omega(t) := \exp(-\omega t)[t \geq 0] \) is an exponential triggering kernel, \( \mu \geq 0 \) is a baseline intensity independent of the history. Here, the occurrence of each historical event increases the intensity by a certain amount determined by the kernel and the weight \( \alpha \geq 0 \), making the intensity history dependent and a stochastic process by itself. We will focus on the exponential kernel. However, other functional forms, such as log-logistic function with long tail influence, can also be used.

(iii) **Multivariate Hawkes processes.** Instead of modeling a single user, we can also model interdependency or mutual excitation between the events from a collection of \( U \) users. Then the intensity \( \lambda_i(t) \) of a user \( i \) will depend on events from the other users

\[
\lambda_i(t) = \mu_i + \sum_{j=1}^{U} \alpha_{ij} \kappa_\omega(t-t_j) \]

\[
dN_j(t) = \mu_j + \sum_{j=1}^{U} \alpha_{ij} \sum_{t_j \in \mathcal{H}_j(t)} \kappa_\omega(t-t_j),
\]

where \( \mu_i \) is the base intensity for node \( i \), \( N_j(t) \) is the counting process representing the historical events \( \mathcal{H}_j(t) \) from user \( j \), and \( \alpha_{ij} \geq 0 \) models the strength of influence from user \( j \) to user \( i \).

Given the intensity functions, one can generate the time stamps of events using the Ogata’s thinning algorithm [25]. The algorithm is essentially a rejection sampling algorithm where samples are first proposed from a homogeneous Poisson process and then samples are kept according to the ratio between the actual intensity and that of the Poisson process.
3 Stochastic Differential Equations for Information Diffusion

We will first propose a novel point process based information diffusion model, and reformulate it as a SDE in Theorem 1. Then we show its application in two examples, including the Hawkes process and opinion diffusion model.

**Theorem 1.** Consider a network with $U$ users and the information content $x_i(t)$ for node $i$ is modeled as:

$$
x_i(t) = b_i + \sum_{j=1}^{U} \alpha_{ij} \kappa_{ij}(t) \star (h(x_j(t)))dN_j(t)
$$

$$
= b_i + \sum_j \alpha_{ij} \sum_{t_j \in \mathcal{H}_j(t)} \kappa_{ij}(t-t_j)h(x_j(t))
$$

where $b_i$ is the base content, $\alpha_{ij}$ is the influence weight from node $j$ to $i$, $h$ is a problem-specific function, $N_j(t)$ is the point process, $\mathcal{H}_j(t)$ is the history of events up to time $t$ for user $j$, function $h(x_j(t))$ is an application-dependent function, $\kappa_{ij}(t) = \exp(-\omega t)$ is an exponential triggering kernel, and $\star$ is the convolution operator. The SDE form is:

$$
\text{d}x_i(t) = \omega(b_i - x_i(t))\text{d}t + \sum_j \alpha_{ij} h(x_j(t))\text{d}N_j(t)
$$

To prove it, we apply the differential operator to (3) and use the property of convolution operator. Appendix A contains details. Next we explain the generative model and SDE in detail. First, the model in (3) is a generalization of point process models, since it not only models the on-off temporal behavior of user $i$, but also models the information content (activity feature) $x_i(t)$. For example, $x_i(t)$ can be the user’s scalar opinion, or a feature vector of interested topics.

More specifically, the model captures both exogenous and endogenous properties of social networks. The exogenous term is the base information content $b_i$, and the endogenous term captures the phenomena that one’s information content, e.g., opinion or a vector of interested topics, is influenced by his neighbors. The adjacency matrix $A = (\alpha_{ij})$ measures the strength of such influence, the exponential kernel $\kappa_{ij}(t)$ captures the decay of influence of past events over time, $h(x)$ captures the form of influence. Typical forms include $h = 1$ and $h(x) = x$. We will provide examples in the next subsections. Hence the summation term captures how the information content of each history event from others influences that of user $i$. The form of influence is represented by the convolution operator $\star$ defined as: $f(t) \star \text{d}N(t) = \int_{0}^{t} f(t-s)\text{d}N(s) = \sum_{t_i < t} f(t - t_i)$.

The SDE form of the model contains a drift and jump part.

**The drift term.** The change rate of user $i$’s information content, $\text{d}x_i(t)/\text{d}t$, is proportional to the negative of his current information content $x_i(t)$. For example, if $x_i$ represents opinion, such negative feedback stabilizes the stochastic process which models the phenomena that people’s opinion tends to stabilize over time. Moreover, if ignoring the jump term, the expected value of $x_i(t)$ will converge to $b_i$ as time goes by. This can be seen by setting $\mathbb{E}[\text{d}x_i(t)] = 0$.

**The jump term.** It captures the fact that the total change of $i$’s information content is a weighted summation of his neighbors’ event influence. $\alpha_{ij}$ ensures only the user’s neighbor influence will be considered and $\text{d}N_j(t) \in \{0, 1\}$ captures whether an event happens or not.

Our novel SDE formulation for modeling information diffusion is important. First, it has a nature interpretation of the original generative model. Furthermore, the SDE opens the new gate to stochastic optimal theory to solve many important social problems such as least square opinion guiding, opinion influence maximization. Without this novel connection to SDE, it will not be easy to design optimal stochastic control algorithms.

Besides transforming an existing model to a SDE, we can also directly design the SDE model to take account of many other factors, e.g., one can add the Wiener diffusion process to enrich the SDE by capturing the Gaussian noise in the system. Next, we show two applications of Theorem 1.
3.1 SDE for Multivariate Hawkes Processes

The Multivariate Hawkes process [23] efficiently captures users behavior pattern, as shown in many works on information diffusion [33, 13, 14, 31]. \( N_i(t) \) counts the number of events generated by user \( i \) up to time \( t \) and its intensity function \( \lambda_i(t) \) models interdependency and mutual excitation between the events from a collection of \( U \) users as follows.

\[
\lambda_i(t) = \eta_i + \sum_{j=1}^{U} \alpha_{ij} \kappa_{\omega}(t) * dN_j(t) \tag{5}
\]

where \( \eta_i \) is the base intensity for node \( i \), \( N_j(t) \) is the counting process representing the historical events \( H_j(t) \) from user \( j \), and \( \alpha_{ij} > 0 \) models the strength of influence from user \( j \) to user \( i \). Here, the occurrence of each historical event from one’s neighbors increases the intensity by a certain amount determined by the kernel and the weight \( \alpha_{ij} \). The intensity describes the probability that a user takes an action in small time window \([t, t + dt]\) and one can generate the time stamps of events using the Ogata’s thinning algorithm [25].

**SDE for Hawkes intensity.** \( h(x_j(t)) = 1 \) in this case, and we can use Theorem 1 to reformulate (5) as a SDE:

\[
d\lambda_i(t) = \omega(\eta_i - \lambda_i(t))dt + \sum_j \alpha_{ij} dN_j(t) \tag{6}
\]

How to design the optimal policy to influence the baseline part such that the user activity achieves a desired level? To do this, the SDE model is important.

3.2 SDE for Opinion Diffusion Model

The recent opinion diffusion model considers both the timing and content of each event [7, 18]. It assigns each user \( i \) an intensity function \( \lambda_i(t) \) using the Hawkes process, and an opinion process \( x_i(t) \) where \( x_i(t) = 0 \) corresponds to neutral opinion, and large positive/negative values correspond to extreme opinions. The users are connected according to an adjacency matrix \( A = (\alpha_{ij}) \). The model captures how opinion of each user can be influenced by neighbors. More specifically, the opinion of user \( i \) is modified as a temporally discounted average of the neighbor opinion as follows.

\[
x_i(t) = \eta_i + \sum_{j=1}^{U} \alpha_{ij} \kappa_{\omega}(t) \cdot (x_j(t) \cdot dN_j(t)) \tag{7}
\]

This model has superior performance in learning and predicting opinions [7, 18]. Furthermore, [7] presented a relation between the stationary opinion \( \mathbb{E}[x_i(t)] \) and model parameters. One can then control the parameters such that \( \mathbb{E}[x_i(t)] \) will converge to a target. However, the algorithm does not take into account the intermediate state of the opinion dynamics, and hence can result in trajectories with large variance. Given this generative model, it is not clear whether it can be used to design effective feedback policies to guide the opinion dynamics more precisely to some target states.

**SDE for opinion diffusion.** Directly applying Theorem 1 yields (4) with \( h_j(x_j) = x_j \). We further generalize it and add one noise term in (4):

\[
dx_i(t) = \omega(h_i - x_i(t))dt + \beta dw_i(t) + \sum_j \alpha_{ij} x_j(t)dN_j(t)
\]

The noise (Wiener) process \( dw_i(t) \) term captures the normal fluctuations in the dynamics due to unobserved factors such as activity outside the social platform and unexpected real life events. Next, we show how to optimally control the SDE to guide the information diffusion.

4 Convex Information Guiding Framework

We will first formally define the novel information guiding problem and then show its application to two examples. For simplicity of notation, we set \( \mathbf{x}(t) = \mathbf{x}; \mathbf{u}(t) = \mathbf{u} \).
**Definition 2** (Information Guiding Problem). The SDE of information content \( x(t) \in \mathbb{R}^U \) in (4) is reformulated with an extra control policy term \( u(x, t) \in \mathbb{R}^U \) as follows.

\[
dx = (f(x) + u)dt + g(x)dw(t) + h(x)dN(t) \tag{8}
\]

Given initial data \((x_0, t_0)\), we aim to find an optimal control policy \( u^* \) on the finite window \([t_0, T]\) that minimizes the following convex objective function:

\[
V(x_0, t_0) := \min_u \mathbb{E}\left[ \phi(x(T)) + \int_{t_0}^T \mathcal{L}(x, u, t)dt \right] \tag{9}
\]

where \( V \) is called the Value Function that summarizes the optimal expected cost if optimal control \( u^* \) is executed from time \( t_0 \) to \( T \). It is a function of the initial state, since the optimal value depends on the initial state. The expectation \( \mathbb{E} \) is over stochastic processes \( \{w(t), N(t)\} \) for \( t \in [t_0, T] \). \( \phi \) is terminal cost and \( \mathcal{L} \) is instantaneous cost.

**Control policy.** \( u(x, t) \) is a feedback control, which means that it can depend on the information content \( x(t) \) at time \( t \) and can be designed adaptively based on current state. For example, if \( x \) models the opinion, the practical meaning of \( u_i(x, t) \) is that it determines how fast the opinion needs to be changed for user \( i \).

For example, a network moderator may request the user to change his opinion from \(-3\) to \(1\) in one day, and \( u_i \) quantifies the amount of change that needs to be made in a unit amount of time.

**Terminal cost \( \phi \).** It is the cost at final time \( T \) and a necessary term. We discuss several cases as follows:

- **Least Square Guiding.** The goal is to guide the expected state to achieve the pre-specified target \( a \) at final time \( T \). For opinion diffusion, the goal can be to ensure nobody believes the rumor at the terminal time \( T \). Mathematically, we set \( \phi = \|x(T) - a\|^2 \). To influence users’ Hawkes intensity of taking actions, one can also set the desired level of the intensity function to be at a high level \( a \) and conduct activity guiding: \( \phi = \|\lambda(t) - a\|^2 \).

- **Information Influence Maximization.** The goal is to maximize the information content of all users, e.g., the goal for an educator is to maximize the students’ recognition of the value of education. Mathematically, we set \( \phi = -\sum_u x_u(T) \) to maximize each user’s positive opinion. Moreover, to improve the overall activity level in social platforms, the goal is to maximize everyone’s Hawkes process intensity to take actions: \( \phi = -\sum_u \lambda_u(T) \).

**Instantaneous cost \( \mathcal{L} \).** This is the cost at any time \( t \in [t_0, T] \) and in the form of \( \mathcal{L} = q(x) + \rho c(u) \). The state cost \( q(x) \) is optional and a control cost \( c(u) \) is always necessary. We set \( q = 0 \) if the state cost only occurs at final time \( T \). Otherwise \( q \) is necessary since it is the cost at any intermediate time, i.e., maximizing the positive opinion influence such as students’ recognition of the value of education over consecutive weeks and not just on one day. Its form is typically same as the terminal cost, i.e., \( q = \phi \). The control cost is \( c(u) = \|u(t)\|^2 \). It models the scenarios that the policy typically costs money or human efforts. The scalar \( \rho \) controls the trade-off between the control cost and state cost. Large \( \rho \) means small control budget and setting \( \rho \to \infty \) leads to the case without control. Finally, we sum up all the instantaneous costs by taking integration on \([t_0, T] \).

**Uniqueness.** Our information guiding problem is much more general and challenging than traditional influence maximization and activity shaping problem [21, 4, 13, 7] due to the novel definition of instantaneous cost and the feedback control policy. First, we can not only consider the cost just at the terminal time, but can also control the system during a period with intermediate cost. Controlling the system during a period is much more challenging than only at terminal time. The forms of the cost functions also contain a wide range of applications. Most importantly, we seek to find a control policy that is updated according to the feedback of current state over time. It is more challenging than prior works that fix the policy only at the initial time. Next, we show two examples, where the cost functions can be least square guiding or information maximization.
Guiding user engagement. How to control the baseline intensity of the Hawkes process in (5) by providing incentives to users to steer the community to a desired level? We can formulate it as an information guiding problem by adding a control policy \( u_i(\lambda_i, t) \) to the SDE in (6) as:

\[
d\lambda_i(t) = \omega(\eta_i + u_i(\lambda_i, t) - \lambda_i(t))dt + \sum_j \alpha_{ij} dN_j(t)
\]

where \( \eta_i + u_i(\lambda_i, t) \) is the exogenous intensity and the practical meaning of \( u_i \) is that it captures the amount of additional influence to change the baseline intensity \( \eta_i \) for user \( i \).

**Guiding opinion diffusion.** We can also guide the opinion SDE with an extra control policy \( u_i(x_i, t) \) as follows.

\[
dx_i = \omega(b_i + u_i - x_i)dt + \beta dw_i + \sum_j \alpha_{ij} x_j dN_j \tag{10}
\]

where the dependency on \( t \) is omitted. \( u_i(x_i, t) \) determines the amount of opinion that needs to be changed.

Solving the information guiding problem is challenging, since the objective involves taking expectation over complex information diffusion process. Furthermore, it is a functional optimization problem since the optimal policy is a function of both state and time. Fortunately, the SDE formulations allow us to connect the problem to that of stochastic dynamic programming methods. As a result, we can extend lots of tools in stochastic optimal control to address sequential decision making in information diffusion.

5 Finding the Optimal Control Policy

In this section we will find the optimal control posed in (9). Prior work in control theory all focus on controlling the SDE where the jump term is a Poisson process [26, 17]. However, in our model, the jump term is driven by more complex processes (e.g., Hawkes process). Thus significant generalizations, both in theory and algorithms, are needed. The theory developed here can be of independent interests to other applications with complex processes. We first derive the HJB equation for a deterministic system, then extend the procedure to our stochastic information guiding problem.

5.1 HJB Equation for Deterministic Systems

To obtain the optimal control, we need to compute the value function \( V \) in (9) subject to the constraint of the SDE in (8). A typical way to solve this sequence of optimization problems is to break down the complex optimization into simpler subproblems [17]. First, the fixed initial condition \( x(t_0) \) needs to be replaced by a more arbitrary start, \( x(t) \), so that the start can be analytically manipulated and we obtain a time-varying objective \( V(x, t) \) amenable to analysis.

Next, since the value function consists of an integral term, we break the integral into \([t, t+dt] \) and \([t+dt, T] \). If the system is deterministic, we can further split (9) as:

\[
V(x, t) = \min_u \left[ \phi + \int_t^{T-} L d\tau + \int_t^{t+dt} L d\tau \right] \tag{11}
\]

The first term is the optimal cost starting from \( t + dt \) and the second term is the optimal cost on \([t, t+dt] \). Hence the form of (11) follows the structure of a dynamic programming and we can solve the problem recursively: If we know \( V(x(t+dt), t+dt) \), we only need to proceed optimally on \([t, t+dt] \) to compute \( V(x, t) \) backward.

To further simplify (11), we perform deterministic Taylor expansion up to second order of the first term on right-hand side as \( V(x(t+dt), t+dt) := V(x, t) + dV(x, t) \), where

\[
dV = V_t dt + V_x^\top dx + \frac{1}{2} dx^\top V_{xx} dx + dx^\top V_{xt} dt + \frac{1}{2} V_{tt} dt^2
\]
Then we can cancel $V(x, t)$ on both sides of (11), divide it by $dt$, and take the limit as $dt \to 0$. Since $dx = x'(t)dt$, all the second order term in $dV$ goes to 0. Hence we obtain Hamilton-Jacobi-Bellman (HJB) equation: 
$$-V_t = \min_u \{L(x, u, t) + V_x^T x'\}.$$ 

However, our system is stochastic and the above procedure needs to be generalized significantly. While the HJB equation and stochastic dynamic programming has been studied in SDEs driven by Poisson processes [32, 17], the generalization to general point process such as Hawkes process is not existent, and we are the first to provide the theory.

5.2 HJB Equation for Information Diffusion Model

To derive the HJB equation for our SDE model in (8), we need to solve two challenges: (i) compute the stochastic Taylor expansion $dV$ under our SDE, which is not a simple form as in the deterministic case; and (ii) take the expectation of the stochastic terms in (11) before minimization. Hence our SDE formulation of information diffusion model is central for obtaining the optimal control policy.

First, to compute $dV$, we derive the novel Theorem 3, which applies to general SDE driven by any point processes and significantly generalizes that of [17], which is only for Poisson process driven SDE. See appendix B for the proof.

**Theorem 3** (Generalized Ito’s Lemma). Given the SDE in (8), let $V(x, t)$ be a twice-differentiable function in $x$ and once in $t$, then we have:
$$dV = \left\{ V_t + \frac{1}{2} \text{tr}(V_{xx} gg^T) + V_x^T (f + u) \right\} dt + V_x^T g dw + (V(x + h, t) - V(x, t)) dN(t)$$ \hspace{1cm} (12)

Next, we handle the expectation challenge and derive the stochastic HJB equation. See appendix C for the proof.

**Theorem 4.** The HJB equation for the Information Guiding Problem in (9) is:
$$-V_t = \min_u \left[ L + \frac{1}{2} \text{tr}(V_{xx} gg^T) + V_x^T (f + u) + \sum_{j=1}^{U} \lambda_j(t) (V(x + h_j(x), t) - V(x, t)) \right]$$ \hspace{1cm} (13)

where $h_j(x)$ is the $j$-th column of $h(x)$.

The HJB equation is a key and simplified representation of the value function in (9), since the expectation over the complex system has been handled. It is a partial differential equation and its solution is the value function. Next we show under the optimal parameterization of the value function, the HJB equation is solved efficiently.

5.3 Optimal Parameterization of Value Function

Solving the HJB equation (13) leads to the optimal control. To do this, we first explore the structure of $V$.

**Proposition 5.** If the SDE (8) is linear in the state $x$, and the terminal and instantaneous cost are quadratic/linear in $x$, then the value function $V$ in (9) must be quadratic/linear.

The argument is intuitive since the $V$ is the optimal value of the summation of quadratic/linear functions. See appendix D for details. Two classes of important information guiding problems fall into this category:

- **Least Square Guiding.** We have $\phi = \frac{1}{2} \|x(T) - a\|^2$, $L = \frac{1}{2} \|x(t) - a\|^2 + \frac{E}{2} \|u(t)\|^2$, hence $V$ is quadratic.
- **Information maximization.** We have $\phi = -\sum u x_u(T)$, $L = -\sum u x_u(t) + \frac{E}{2} \|u(t)\|^2$, hence $V$ is linear.

Here we show derivations for the quadratic case. See appendix F for complete derivations for the maximization problem. We set $V(x, t)$ to be quadratic in $x$ with unknown coefficients $v_1(t) \in \mathbb{R}^U$, $v_{11}(t) \in \mathbb{R}^{U \times U}$ and $v_0(t) \in \mathbb{R}$:
$$V(x, t) = v_0(t) + v_1(t)^T x + \frac{1}{2} x^T v_{11}(t) x$$ \hspace{1cm} (14)
Algorithm 1 Optimal Control

1: Input: target state $a$, model parameters $\{A, \mu, \beta, b\}$, timestamps $\{\tau_k\}_{k=1}^m$, events $\{t_i\}_{i=1}^n$
2: Output: $v_{11}(\tau_k), v_1(\tau_k), k = 1, \ldots, m$
3: for $k = 1$ to $m$ do
4:   Compute $\lambda_u(\tau_k) = \mu_u + \sum_{j:t_j<\tau_k} \alpha_{uu}, \kappa(\tau_k - t_i), \Lambda(\tau_k) = \sum_{j=1}^U \lambda_j(\tau_k) B^j$ for each user $u$
5: end for
6: Compute $v_{11}(\tau_k), v_1(\tau_k)$ using Ode45 solver. Then compute $u(\tau_k)$ from (15).

To find the optimal control, we substitute (14) to HJB equation and take the gradient of the right-hand side of (13) with respect to $u$ and set it to 0. This yields the optimal feedback control policy:

$$u^*(x,t) = -\frac{1}{\rho} V_x = -\frac{1}{\rho} \left( \underbrace{v_1(t)}_{\text{feedforward}} + \underbrace{v_{11}(t)x}_{\text{feedback}} \right)$$  \quad (15)

It moves to the negative partial gradient $-V_x$. It is intuitive since it minimizes the objective function. Moreover, since $\rho$ controls the tradeoff between control and state cost, setting $\rho \to \infty$ means low budget for control hence $u^* \to 0$. The optimal policy consists of a feedforward and feedback term.

- The feedforward term is state-independent and captures the baseline control policy: how one should control the system as time goes by.
- The feedback term is linear in the current information content. It captures how one should adaptively change the policy based on the current state $x(t)$ at time $t$. This is appealing since we update the policy according to the current status in the optimal sense. Hence we can make use of the valuable information efficiently, while prior works can only fix the policy at initial time and not able to adaptively change it. In the experiments, we will show our feedback policy has much faster convergence rate to the optimal cost with less variance than alternatives.

5.4 Stochastic Optimal Control Algorithm

The final step is to find $\{v_1(t), v_{11}(t)\}$ to obtain $u^*$. We begin with substituting $u^*$ in (15) back to the HJB equation. Since $V(x,t)$ is quadratic, we can separate the HJB equation into purely quadratic in $x$, linear, and scalar terms. Grouping coefficients of these terms leads to a set of matrix Ordinary Differential Equations (ODEs) as follows. See appendix E for the complete derivations.

**Update $v_{11}(t)$**. Setting $\Lambda(t) = \sum_{j=1}^U \lambda_j(t) B^j$, where matrix $B^j$ has the $j$-th column to be $(\alpha_{1j}, \ldots, \alpha_{Uj})^\top$ and zero elsewhere, we solve the ODE with $v_{11}(T) = I$:

$$-v'_{11} = I + 2v_{11}(-1 + \Lambda) - \frac{v_{11}v_{11}}{\rho} + \sum_{j=1}^U \lambda_j B^j v_{11} B^j$$

**Update $v_1(t)$**. We solve the ODE with $v_1(T) = -a$:

$$-v'_1(t) = -a + (-1 + \Lambda^\top - v_{11}(t)/\rho)v_1(t) + v_{11}(t)b$$

**Online updating**. The ODEs can be solved offline. We use the numerical Runge-Kutta algorithm [8] and partition $(t_0, T]$ to equally-spaced timestamps $\{\tau_k\}$. Then we obtain values of $v_{11}, v_1$ at these timestamps. We use the Ode45 Solver in MATLAB. Finally given the learned functions, we use (15) to update the policy online and adaptively to current state. Algorithm 1 summarizes the procedure.

**Summary**. Finding the optimal policy consists of 4 steps: First, obtain the HJB equation in (13); second, parameterize $V$ with unknown coefficients e.g., quadratic parameterization in (14); third, substitute $V$ to HJB equation to obtain the optimal control policy in (15). Finally, substitute (15) back to the HJB equation to obtain the unknown coefficients.
6 Extensions

Here we first extend our control framework to control over time-varying network and node birth network. Finally we present the first framework for min-max robust control for Jump Diffusion SDEs.

6.1 Control networks with time-varying edge

Real world social network changes over time since users can follow or unfollow each other as time goes by [14]. Our control framework can be easily extended to the time-varying network case. For the control with fixed network, the conditional expectation is over the stochastic pair \{w(t), N(t)\} when using Bellman’s optimality to derive the HJB equation. Now the network adds one more stochasticity to the system and we also need to take the expectation of the network topology $A(t)$ to derive the HJB equation. It leads to the expectation over the jump term $h_j(x)$ in the HJB equation (13):

$$\sum_j \lambda_j(t)(V(x + h_j(x), t) - V(x, t)) \rightarrow \sum_j \lambda_j(t)(V(x + \mathbb{E}[h_j(x)], t) - V(x, t))$$

(16)

where $\mathbb{E}[h_j(x)] = \mathbb{E}[h_{ij}]$, and $h_{ij} = \mathbb{E}[x_{ij}]$. Setting $\mathbb{E}[A(t)] = \mathbb{E}[A_{ij}(t)]_{i,j=1}^{U}$, we will use the expected value of the network adjacency matrix, $\mathbb{E}[A(t)]$ instead of $A$ as input to the algorithm.

Next, we demonstrate how to compute $\mathbb{E}[A(t)]$ for the link creation process. We model link creation $v \rightarrow s$ as $A_{vs}(t)$, with $A_{vs}(t) = 1$ meaning a link is created at time $t$ and 0 otherwise. We model the link creation process as a survival process [1, 14], which only has one event for an instantiation of the point process. More specifically, its intensity is defined as:

$$\sigma_{vs}(t) = (1 - A_{vs}(t))\gamma_v$$

(17)

where $\sigma_{vs}(t)$ denotes the intensity to create link $v \rightarrow s$, and $1 - A_{vs}(t)$ effectively ensures a link is created only once, and intensity is set to 0 after that. The term $\gamma_v \geq 0$ denotes the Poisson process intensity, which models the node $v$’s own initiative to follow and create link to others. Since the link creation process (survival process) is a special case of the temporal point process, from (1) we have:

$$\text{d}\mathbb{E}[A_{vs}(t)] = \sigma_{vs}(t)\text{d}t = (1 - \mathbb{E}[A_{vs}(t)])\gamma_v\text{d}t$$

(18)

with initial condition $\mathbb{E}[A_{vs}(0)] = 0$ if $s$ and $v$ are not connected. Intuitively, (18) measures the probability that a link $v \rightarrow s$ happens between small interval $[t, t + \text{d}t)$. We can easily solve the first order matrix differential equation (18) and obtain :

$$\mathbb{E}[A_{vs}(t)] = 1 - \exp(-\gamma_v t)$$

With the expected value network adjacency matrix as input, we can then use Algorithm 1 to compute the optimal control.

6.2 Control networks with node birth

The network dimension can also grow as time goes by since more people will join the network. A most challenging problem is that the dimension of the system is changing, as compared with the previous case where the number of users are fixed. It remains unknown how to derive HJB equation if the network’s dimension is not fixed.

Here we provide an efficient and novel way to handle this problem by establishing the connection to the time-varying edge case. The main idea is to transform the stochasticity of node birth process, which changes the network’s dimension, to the stochasticity of link creation process with fixed network dimension. More specifically, we have the following observation.
Observation. The process of adding a new user \( v \) to the existing network \( A \in \mathbb{R}^{(N-1) \times (N-1)} \) and connects to user \( s \) is equivalent to link creation process of setting \( A(t) \in \mathbb{R}^{N \times N} \) to be the existing network and letting \( A_{vs} = 1 \).

With this observation, instead of having a sequence of size-growing matrix \( A(t) \), a memory efficient way is to fix its dimension to be the maximum number of nodes at terminal time and adds a link whenever a user joins the network. Moreover, it enables us to transform the stochasticity of node birth process, which changes the network’s dimension, to the stochasticity of link creation process with fixed network dimension. Hence the problem boils down to the time varying network.

Finally, a key difference between the controls for time-varying and node birth network is: for the time-varying edge case, the optimal control is on every node. For the node birth case, we will not control the node until it joins the network.

7 Experiments

We focus on two tasks: least square opinion guiding and opinion influence maximization, and evaluate our framework using synthetic and real world data. We compare with state-of-arts in reinforcement learning and heuristics.

- **Cross Entropy [30]**: It samples controls from a Gaussian distribution, sorts the samples in ascending order w.r.t. the cost and recomputes the distribution parameters based on the first \( K \) elite samples. Then returns to the first step with new distribution, until costs converge.

- **Finite Difference [29]**: It approximates the gradient of the expected cost with respect to the control.

- **Greedy**: It controls the system when the local state cost is high. Specifically, we divide the time horizon into \( n \) state cost observation timestamps. At each timestamp, Greedy computes state cost and controls the system based on pre-specified control rules if current cost is more than \( k \) times of the optimal cost of our method. It will stop if it reaches the current budget upper bound. We vary \( k \) from 1 to 5, \( n \) from 1 to 100 and report the best performance.

- **Slant [7]**: It sets the control policy only at the initial time to achieve the target state.

7.1 Experiments on Synthetic Data

**Experimental setup.** We consider a network with 1000 users. We simulate the opinion SDE on the observation window \([0, 10]\) by applying Euler forward method to compute its difference form. The observation window is divided into 100 timestamps, with interval \( \Delta t = 0.1 \). We set the baseline opinion uniformly at random, \( b_i \sim U[-1, 1] \), noise level \( \beta = 0.2 \), adjacency matrix \( (\alpha_{ij}) \) with sparsity of 0.001 and \( \alpha_{ij} \sim U[0, 0.01] \), initial opinion \( x_i(0) = -10 \), and \( \omega = 1 \) for the kernel \( \kappa_\omega \). See appendix G for details. We repeat simulation of the SDE for ten times and report average performance. We set the tradeoff (budget level) parameter \( \rho = 10 \), and our results generalize beyond this value.

**Total cost.** Figure 2 (a) shows our method performs the best consistently. For LSOG, we set target opinion \( a_i = 1 \) and the total cost per user measures the difference between each person’s opinion and the target. We computed it by dividing the value function \( V(x_0, t_0) \) by \#. For OIM, there is no target state. Since the goal is to maximize each user’s positive opinion, the larger the opinion, the better. We compute the total opinion per user by dividing the negative value function by \#. On both tasks, our method has around 2.5× improvement over CrossEntropy and 4× improvement over FiniteDifference, which are state-of-art reinforcement learning algorithms. CrossEntropy assumes the control is sampled from a Gaussian distribution, and FiniteDifference approximates the gradient. However, our method does not have such restriction on control policy or approximation of the gradient, and our policy exactly minimizes the cost. Hence it has the best performance.
Least Square Guiding

Opinion Maximization

(a) Total cost/opinion  (b) Instantaneous cost/opinion  (c) Our trajectory  (d) CrossEntropy trajectory

Figure 2: Experiments on least square opinion guiding (LSOG) and opinion influence maximization (OIM). (a) total cost for LSOG and total opinion for OIM per user. Error bar is the variance; (b) instantaneous cost/opinion per user over time. Line is the mean and pale region is the variance; (c) and (d) sample opinion trajectory of five users.

**Instantaneous cost & trajectory.** Figure 2 (b) shows the instantaneous cost over time per user, which is computed by dividing the cost $\mathcal{L}$ in (9) by $\#$ users at each time $t$. Our method converges to the optimal cost fastest and the cost is much lower than competitors at each time. Moreover, our method has the lowest variance and is quite stable despite multiple runs and the stochasticity in the SDE. (c) and (d) compare the opinion trajectories. The jumps in the opinion correspond to the opinion posting events that are modulated by the Hawkes process in the SDE. For LSOG, the opinion converges from initial value to target much faster. For OIM, our method maximizes the opinion from the negative initial value quickly: around time 2.5 all users’ opinion are positive in (c), compared to time 5 in (d). Moreover, our method can achieve the largest opinion value, e.g., 20, while that of CrossEntropy is less than 10.

**Robustness.** In real world, we need to learn parameters of the model. Typically, error exists between the estimated and ground truth. Our framework still works efficiently with inaccurate model parameters. See appendix G for details.

### 7.2 Experiments on Real-world Data

**Experimental setup.** We evaluate our framework over two node birth networks. We focus on the least square opinion guiding task. Twitter [14] contains nearly 550,000 tweet, retweet and link creation events from around 280,000 users. We use all events from Sep 21 to Sep 30 2012 and use the data before Sep 21 to construct the initial social network. We consider the links created in the second 10-day period to be the node birth. MemeTracker [22] contains online social media activities from Aug 2008 to Apr 2009. Users track the posts of others and the network growth is captured by hyperlinks of comments on one site to others. In particular, we extract 11,321,362 posts among 5000 nodes. We use the data in Aug 2008 to construct the initial network and use the LIWC [28] toolbox to extract opinions from posts.

We use two evaluation procedures. First, similar to the synthetic network case, we now have a real network and parameters are learned from data, hence we can evaluate the total cost. We set $\rho = 10$. However, a more interesting evaluation scheme would entail carrying real policy in a social platform. Since it is very challenging, we mimic such procedure. We partition the data into ten intervals and use one interval for training and other intervals for testing:
Least Square Guiding

815
5187
6798
7589
8601
0
2500
5000
7500

Methods

Total opinion

Methods

Our
CrossEntropy
FiniteDifference
Greedy
SLANT

1. We estimate model parameters using data in interval 1.

2. Given the learned parameters, we compute the optimal policy and the optimal opinion trajectory $x_i^*$, for all other intervals, $i = 2, \cdots, 10$. Then we sort the real opinion trajectory $x_i$ according to the similarity to $x_i^*$ using Euclidean distance $||x_i^* - x_i||$ in descending order.

3. Compute total cost on each of the other intervals and sort intervals according to total cost in descending order.

4. Finally we compute prediction accuracy by dividing the number of pairs with consistent ordering in step 2 and 3 by total number of pairs. See appendix G for details.

Ideally, the ordered pairs by step 2 and 3 by total number of pairs. See appendix G for details.

Figure 3: Experiments on least square guiding (LSOG) and opinion influence maximization (OIM) over node birth networks.

(a) Cost/opinion on Meme (b) Cost/opinion on Twitter (c) Prediction on Meme (d) Prediction on Twitter

1. We estimate model parameters using data in interval 1.

2. Given the learned parameters, we compute the optimal policy and the optimal opinion trajectory $x_i^*$, for all other intervals, $i = 2, \cdots, 10$. Then we sort the real opinion trajectory $x_i$ according to the similarity to $x_i^*$ using Euclidean distance $||x_i^* - x_i||$ in descending order.

3. Compute total cost on each of the other intervals and sort intervals according to total cost in descending order.

4. Finally we compute prediction accuracy by dividing the number of pairs with consistent ordering in step 2 and 3 by total number of pairs. See appendix G for details.

Ideally, the ordered pairs by step 2 (predicted) and step 3 (real) should be exactly the same, which leads to the accuracy of 1. We report prediction accuracy over ten runs by choosing each different interval for training once.

Total cost. Figure 3 (a) and (b) show that our method consistently performs the best for the two time-varying networks. In terms of total cost per user, it achieves around 6× improvement on LSOG over CrossEntropy and 3× on OIM, compared with the 2.5× improvement for fixed networks. Hence controlling the SDE over time-varying networks is a challenging problem for traditional reinforcement learning algorithms. Moreover, the total costs of all methods for Twitter are higher than that of Memetracker. This is because Twitter has a much higher frequency of node birth, i.e., users join the network in the timescale of minute-to-minute rather than day-to-day in Memetracker. Hence it is more challenging to control due to the high stochasticity.

Prediction accuracy. Figure 3 (c) and (d) further show that our method performs the best. On both tasks and two networks, our method achieves more than 0.4+ improvement over CrossEntropy. It means that our method accommodates 40% more of the total realizations correctly. Accurate prediction means that if applying our control policy, we will achieve the objective better than alternatives. Moreover, the two reinforcement learning algorithms only achieve the accuracy around 30%, which means that their policies do not achieve the optimal cost as our method. Hence it is hard for these methods to succeed on time-varying networks.

Budget sensitivity. Figure 4 further shows our method performs best consistently as budget level decreases. Large value of the cost tradeoff parameter $\rho$ means small budget.
Figure 4: Prediction accuracy as a function of $\rho$ on Twitter.

8 Conclusion

We have proposed a novel SDE model for information diffusion and a novel information guiding problem which builds a new bridge between information diffusion and stochastic optimal control theory. Moreover, in the experiments we have shown that it is important to incorporate the system status information to design a feedback control policy, which will achieve a lower cost with faster speed. To our knowledge, our method is the first efficient method to guide diffusion over time-varying networks. There are many extensions we can imagine along this direction. For instance, the jump term in the model can be extended to marked point processes [20]. Nonlinear SDE can also be modeled.
References

[1] O. Aalen, O. Borgan, and H. Gjessing. *Survival and event history analysis: a process point of view*. Springer, 2008.

[2] W. Chen, A. Collins, R. Cummings, T. Ke, Z. Liu, D. Rincon, X. Sun, Y. Wang, W. Wei, and Y. Yuan. Influence maximization in social networks when negative opinions may emerge and propagate. In *SDM*, 2012.

[3] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1029–1038. ACM, 2010.

[4] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *KDD*, 2009.

[5] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE, 2010.

[6] D. Daley and D. Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*, volume 2. Springer, 2007.

[7] A. De, I. Valera, N. Ganguly, S. Bhattacharya, and M. G. Rodriguez. Learning opinion dynamics in social networks. *arXiv preprint arXiv:1506.05474*, 2015.

[8] J. R. Dormand and P. J. Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[9] N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, 2013.

[10] N. Du, L. Song, H. Woo, and H. Zha. Uncover topic-sensitive information diffusion networks. In *Artificial Intelligence and Statistics (AISTATS)*, 2013.

[11] N. Du, Y. Wang, N. He, and L. Song. Time sensitive recommendation from recurrent user activities. In *NIPS*, 2015.

[12] R. S. Epanchin-Niell and J. E. Wilen. Optimal spatial control of biological invasions. *Journal of Environmental Economics and Management*, 63(2):260–270, 2012.

[13] M. Farajtabar, N. Du, M. Gomez-Rodriguez, I. Valera, H. Zha, and L. Song. Shaping social activity by incentivizing users. In *NIPS*, 2014.

[14] M. Farajtabar, Y. Wang, M. Gomez-Rodriguez, S. Li, H. Zha, and L. Song. Coevolve: A joint point process model for information diffusion and network co-evolution. In *NIPS*, 2015.

[15] M. Gomez-Rodriguez, J. Leskovec, et al. Modeling information propagation with survival theory. In *ICML*, pages 666–674, 2013.

[16] M. Gomez Rodriguez, L. Song, and B. Schölkopf. Open problem: Finding good cascade sampling processes for the network inference problem. In *COLT*, 2014.

[17] F. B. Hanson. *Applied stochastic processes and control for Jump-diffusions: modeling, analysis, and computation*, volume 13. Siam, 2007.

[18] X. He, T. Rekatsinas, J. Foulds, L. Getoor, and Y. Liu. Hawkestopic: A joint model for network inference and topic modeling from text-based cascades. In *ICML*, pages 871–880, 2015.
[19] S. M. Iacus. *Simulation and inference for stochastic differential equations: with R examples*, volume 1. Springer Science & Business Media, 2009.

[20] M. Jacobsen. *Point process theory and applications*. Springer, 2006.

[21] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *SIGKDD*, pages 137–146. ACM, 2003.

[22] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, 2009.

[23] T. J. Liniger. *Multivariate Hawkes Processes*. PhD thesis, Swiss Federal Institute of Technology Zurich, 2009.

[24] C. Nowzari, V. M. Preciado, and G. J. Pappas. Analysis and control of epidemics: A survey of spreading processes on complex networks. *IEEE Control Systems*, 2016.

[25] Y. Ogata. On lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, 1981.

[26] B. K. Oksendal and A. Sulem. *Applied stochastic control of jump diffusions*, volume 498. Springer, 2005.

[27] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925, 2015.

[28] J. W. Pennebaker, R. L. Boyd, K. Jordan, and K. Blackburn. The development and psychometric properties of liwc2015. *UT Faculty/Researcher Works*, 2015.

[29] J. Peters and S. Schaal. Policy gradient methods for robotics. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2219–2225. IEEE, 2006.

[30] F. Stulp and O. Sigaud. Path integral policy improvement with covariance matrix adaptation. In *ICML*, 2012.

[31] Y. Wang, B. Xie, N. Du, and L. Song. Isotonic hawkes processes. In *ICML*, 2016.

[32] J. Yong and X. Y. Zhou. *Stochastic controls: Hamiltonian systems and HJB equations*, volume 43. Springer Science & Business Media, 1999.

[33] K. Zhou, H. Zha, and L. Song. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *AISTAT*, 2013.
A Proof of Theorem 1

Theorem 1. Consider a network with $U$ users and the information content $x_i(t)$ for node $i$ is modeled as:

$$x_i(t) = b_i + \sum_{j=1}^{U} \alpha_{ij} \kappa_{\omega}(t) * (h(x_j(t))dN_j(t))$$  \hspace{1cm} (19)

where $b_i$ is the base content, $\alpha_{ij}$ is the influence weight from node $j$ to $i$, $h$ is a problem-specific function, $N_j(t)$ is the point process, $H_j(t)$ is the history of events up to time $t$ for user $j$, function $h(x_j(t))$ is an application-dependent function, $\kappa_{\omega}(t) = \exp(-\omega t)$ is an exponential triggering kernel, and $*$ is the convolution operator. The SDE form is:

$$dx_i(t) = \omega(b_i - x_i(t))dt + \sum_j \alpha_{ij} h(x_j(t))dN_j(t)$$  \hspace{1cm} (20)

**Proof.** The convolution operator $*$ of any two functions $f(t)$ and $g(t)$ is defined as:

$$f(t) * g(t) = \int_{0}^{t} f(t-s)g(s)ds$$  \hspace{1cm} (21)

Next, we will apply the differential operator $d$ to $x_i(t)$, and we need the following two properties:

- $d\kappa_{\omega}(t) = -\omega\kappa_{\omega}(t)dt$ for $t \geq 0$ and $\kappa_{\omega}(0) = 1$.

- The differential of the convolution of two functions is: $d(f * g) = f(0)g + g * df$.

Set $f = \kappa_{\omega}(t)$, and $g = \sum_j \alpha_{ij} h(x_j) dN_j(t)$, then take the differential of $x_i(t)$ and use the above two properties, we have:

$$dx_i(t) = d(f * g) = \sum_{j=1}^{U} \alpha_{ij} h(x_j) dN_j(t) - \omega \left( \sum_{j=1}^{U} \alpha_{ij} \kappa_{\omega}(t) * (h(x_j) \cdot dN_j(t)) \right) dt$$  \hspace{1cm} (22)

$$= \sum_{j=1}^{U} \alpha_{ij} h(x_j) dN_j(t) - \omega(x_i(t) - b_i) dt$$  \hspace{1cm} (23)

$$= \omega(b_i - x_i(t))dt + \sum_{j=1}^{U} \alpha_{ij} h(x_j(t))dN_j(t)$$  \hspace{1cm} (24)

This completes the proof.
B Proof of Theorem 3

Theorem 3 (Generalize Ito Lemma). Given the SDE in (8), let \( V(x,t) \) be a twice-differentiable function in \( x \) and once in \( t \), then we have:

\[
\frac{dV}{dt} = \left\{ V_t + \frac{1}{2} \text{tr}(V_{xx}g^2) + V_x^T (f + u) \right\}dt + V_x^T gdw + (V(x + h, t) - V(x, t))dN(t)
\]

(25)

To prove the theorem, we will first provide some background and useful formulas as follows.

\[
(dt)^2 = 0, dtN(t) = 0, dtw(t) = 0, dw(t)dtN(t) = 0, dw(t)dw(t)^T = dtI
\]

(26)

All the above equations hold in the mean square limit sense. The mean square limit definition enables us to extend the calculus rules for deterministic functions and properly define stochastic calculus rules such as stochastic differential and stochastic integration for stochastic processes. See [17] for the proof of these equations. In the proof below, we will directly acknowledge and use these equations.

Proof. We first restate the SDE in (8) as follows.

\[
dx = (f(x) + u)dt + g(x)dw(t) + h(x)dN(t)
\]

(27)

where we have set \( F(x) = (f(x) + u)dt + g(x)dw(t) \).

Hence, \( F(x) \) denotes the continuous part of the SDE, while \( hN(t) \) denotes the discontinuous part. For notation simplicity we set \( F(x) = F \) and \( h(x) = h \) and omit the dependency on \( x \).

Next, we expand \( dV \) according to its definition:

\[
dV(x,t) = V(x(t + dt), t + dt) - V(x, t)
\]

(29)

With the definition \( x(t + dt) = x(t) + dx \), we can further expand \( V(x(t + dt), t + dt) \) using Taylor expansion on variable \( t \) as follows.

\[
V(x(t + dt), t + dt) = V(x + dx, t + dt)
\]

(30)

\[
= V(x + dx, t) + V_t(x, t)dt + \frac{1}{2} V_{xx}(x, t)dt^2
\]

(31)

Next, we expand \( V(x + dx, t) \) as follows.

\[
V(x + dx, t) = V(x + F + hdtN(t), t)
\]

(32)

\[
= \left( V(x + F + h, t) - V(x + F, t) \right) dN(t) + V(x + F, t)
\]

(33)

\[
= \left[ V(x + h, t) + V_x(x + h)^T F + \frac{1}{2} FV_{xx}(x + h)F^T \right] dN(t) - \left[ V(x, t) + V_x^T F + \frac{1}{2} FV_{xx}F^T \right] dN(t)
\]

(34)

Taylor expansion 1

\[
\cdots + V(x, t) + V_x^T F + \frac{1}{2} FV_{xx}F^T
\]

Taylor expansion 2

(35)

\[
= \left( V(x + h, t) - V(x, t) \right) dN(t) + \left( V_x(x + h) - V_x \right)^T F dN(t)
\]

(36)

\[
+ V(x, t) + V_x^T F + \frac{1}{2} FV_{xx}F^T + \left( \frac{1}{2} FV_{xx}(x + h)F^T - \frac{1}{2} FV_{xx}(x)F^T \right) dN(t)
\]

(37)

Next, we show the reasoning from (32) to (35). First, since \( dN(t) \in \{0, 1\} \), there are two cases for (32). If jump happens, i.e., \( dN(t) = 1 \), then (32) is equivalent to \( V(x + F(x) + h(x), t) \). If there is no jump, i.e., \( dN(t) = 0 \), (32) is equivalent to \( V(x + F(x), t) \). Hence (33) is equivalent to (32).
Second, from (33) to (35), we have used the following Taylor expansions.

**Taylor expansion 1.** For \( V(x + F + h, t) \), we expand it around \( V(x + h, t) \) on the \( x \)-dimension:

\[
V(x + F + h, t) = V(x + h, t) + V_x(x + h)F + \frac{1}{2} FV_{xx}(x + h)F^\top
\]

**Taylor expansion 2.** For \( V(x + F, t) \), we expand it around \( V(x, t) \) along the \( x \) dimension:

\[
V(x + F, t) = V(x, t) + V_x^\top F + \frac{1}{2} FV_{xx}F^\top
\]

Next, we simplify each term in (36) and (37). We keep the first term and expand the second term, \( (V_x(x + h) - V_x) F dN(t) \) as:

\[
(V_x(x + h) - V_x)^\top F dN(t) = (V_x(x + h) - V_x)^\top ((f + u)dt + gdw(t))dN(t)
\]

\[
= (V_x(x + h) - V_x)^\top ((f + u)dt + gdw(t))dN(t)
\]

\[
= 0
\]

where we have used the equations: \( dt dN(t) = 0 \) and \( dw(t) dN(t) = 0 \) in the Ito mean square limit sense from (26).

We keep the third term and expand the fourth term, \( V_x^\top F \), as:

\[
V_x^\top F = V_x^\top (f + u)dt + V_x^\top gdw(t)
\]

The fifth term, \( \frac{1}{2} FV_{xx}F^\top \), is expanded as:

\[
\frac{1}{2} FV_{xx}F^\top = \frac{1}{2} ((f + u)dt + gdw(t))V_{xx}((f + u)dt + gdw(t))^\top
\]

\[
= \frac{1}{2} (f + u) V_{xx} (f + u)^\top (dt)^2 + 2(f + u)dt V_{xx} (g dw(t))^\top + (gdw(t)) V_{xx} (gdw(t))^\top
\]

\[
= \frac{1}{2} (0 + 0 + \text{tr}(V_{xx}gg^\top))dt
\]

\[
= \frac{1}{2} \text{tr}(V_{xx}gg^\top)dt,
\]

where we have used the property that \( (dt)^2 = 0 \), \( dt dw = 0 \), and \( dw(t) dw(t)^\top = dtI \) from (26).

Finally, the last term is expressed as:

\[
\left( \frac{1}{2} FV_{xx}(x + h)F^\top - \frac{1}{2} FV_{xx}(x)F^\top \right) dN(t) = \frac{1}{2} \text{tr}(V_{xx}(x + h)gg^\top) dt dN(t) - \frac{1}{2} \text{tr}(V_{xx}gg^\top) dt dN(t) = 0 - 0 = 0
\]

Substituting (38), (39), (40), and (41) to (36) and (37), we have:

\[
V(x + dx, t) = \left( V(x + h, t) - V(x, t) \right) dN(t) + V_x^\top (f + u) dt + V_x^\top g dw(t) + V(x, t) + \frac{1}{2} \text{tr}(V_{xx}gg^\top) dt
\]

Plugging (42) to (31), we have:

\[
V(x(t + dt), t + dt) = \left( V(x + h, t) - V(x, t) \right) dN(t) + V_x^\top (f + u) dt + V_x^\top g dw(t)
\]

\[
+ V(x, t) + \frac{1}{2} \text{tr}(V_{xx}gg^\top) dt + V_t(x, t) dt
\]

19
Hence after simplification, we have:

\[
dV = V(x(t + dt), t + dt) - V(x(t), t)
= \left\{ V_t + \frac{1}{2} \text{tr}(V_{xx}gg^\top) + V_x^\top (f + u) \right\} dt + V_x^\top gdw + (V(x + h, t) - V(x, t))dN(t)
\]

This completes the proof.
Proof of Theorem 4

Theorem 4. The HJB equation for the Information Guiding Problem in (9) is:

\[-V_t = \min_u \left[ \mathcal{L} + \frac{1}{2} \text{tr}(V_{xx}gg^\top) + V_x^\top (f + u) \right. \right.
\left. \left. + \sum_{j=1}^U \lambda_j(t)(V(x + h_j(x), t) - V(x, t)) \right] \]

where \(h_j(x)\) is the \(j\)-th column of \(h(x)\).

Proof. First, similar to the deterministic case in (11), the value function \(V\) can be written as:

\[ V(x, t) \]

\[ = \min_u \mathbb{E} \left[ V(x(t + dt), t + dt) + \int_t^{t+dt} \mathcal{L} \, d\tau \right] \]

\[ = \min_u \mathbb{E} \left[ V(x, t) + dV + \mathcal{L} \, dt \right] \]

\[ = \min_u \mathbb{E} \left[ V(x, t) + \left\{ V_t + \frac{1}{2} \text{tr}(V_{xx}gg^\top) + V_x^\top (f + u) \right\} dt + V_x^\top g \, dw + (V(x + h, t) - V(x, t)) \, dN(t) + \mathcal{L} \, dt \right] \]

\[ = \min_u \left[ V(x, t) + \left\{ V_t + \mathcal{L} + \frac{1}{2} \text{tr}(V_{xx}gg^\top) + V_x^\top (f + u) \right\} dt + \sum_{j=1}^U \lambda_j(t)(V(x + h_j(x), t) - V(x, t)) dt \right] \]

where (46) to (47) follows from Theorem 3. (47) to (48) follows from the properties of Wiener process and Point process: \(\mathbb{E}[dw] = 0\) and \(\mathbb{E}[dN(t)] = \lambda(t) dt\).

Finally, we cancel \(V(x, t)\) on both sides of (48) and divide both sides by \(dt\). This yields (43).
D Proof of Proposition 5

For the quadratic cost case, i.e., the opinion least square guiding problem, we have: \( \phi = \frac{1}{2}\|x(T) - a\|^2 \), \( \mathcal{L} = \frac{1}{2}\|x(t) - a\|^2 + \frac{\rho}{2}\|u(t)\|^2 \). Since the instantaneous cost \( \mathcal{L} \) is quadratic in \( x \) and \( u \), and terminal cost \( \phi \) is quadratic in \( x \), if the control \( u \) is a linear function of \( x \), then the value function \( V \) must be quadratic in \( x \), since it is the optimal value of the summation of quadratic functions.

Moreover, the fact that \( u \) is linear in \( x \) is because our SDE model for information diffusion is linear in both \( x \) and \( u \). Since \( V(T) = \phi(T) \) is quadratic, as illustrated in [17], one can show by induction that when computing the value of \( V \) backward in time, \( u \) is always linear in \( x \).

Similarly, one can show for the opinion maximization cost, i.e., \( \phi = -\sum x_u(T), \mathcal{L} = -\sum x_u(t) + \frac{\rho}{2}\|u(t)\|^2 \), the value function \( V \) is linear in the state \( x \).
E  Optimal Control Policy for Least Square Opinion Guiding

In this section, we will provide detailed derivations of the optimal control policy for the opinion SDE defined in (10) with the least square opinion guiding cost. First, we choose $\omega = 1$ and restate the controlled SDE in (10) as follows.

$$dx_i(t) = (b_i + u_i(x, t) - x_i(t))dt + \beta dw_i(t) + \sum_{j=1}^{U} \alpha_{ij} x_j(t) dN_j(t)$$

Putting it in the vector form, we have:

$$dx(t) = (b - x + u)dt + \beta dw(t) + h(x)dN(t)$$  \hspace{1cm} (49)

where the $j$-th column of $h(x)$ captures how much influence that $x_j$ has on all other users and is defined as $h_j(x) = B^j x$, where the matrix $B^j \in \mathbb{R}^{U \times U}$ and has the $j$-th column to be $(\alpha_{1j}, \ldots, \alpha_{Uj})^\top$ and zero elsewhere. We substitute $f = b - x(t) + u(t)$, $g = \beta$ and $h$ to (13) and obtain the HJB equation as:

$$-\frac{\partial V}{\partial t} = \min_u \left\{ \mathcal{L}(x, u, t) + \frac{\beta^2}{2} \text{tr}(V_{xx}(x, t)) + V_x(x, t)\top (b - x(t) + u(t)) + \sum_{j=1}^{U} \lambda_j(t) (V(x + h_j(x), t) - V(x, t)) \right\}$$ \hspace{1cm} (50)

For the least square guiding problem, the instantaneous and terminal cost are defined as:

$$\mathcal{L}(x, u, t) = \frac{1}{2} ||x - a||^2 + \frac{1}{\rho} ||u||^2, \hspace{0.5cm} \phi(T) = \frac{1}{2} ||x(T) - a||^2$$  \hspace{1cm} (51)

Hence we assume that value function $V$ is quadratic in $x$ with unknown coefficients $v_1(t) \in \mathbb{R}^U$, $v_{11}(t) \in \mathbb{R}^{U \times U}$ and $v_0(t) \in \mathbb{R}$:

$$V(x, t) = v_0(t) + v_1(t)\top x + \frac{1}{2} x\top v_{11}(t) x$$ \hspace{1cm} (52)

To find the optimal control, we substitute (52) to HJB equation and take the gradient of the right-hand side of the HJB equation (50) with respect to $u$ and set it to 0. This yields the optimal feedback control policy:

$$u^*(x, t) = -\frac{1}{\rho} V_x = -\frac{1}{\rho} \left( v_1(t) + v_{11}(t) x \right)$$ \hspace{1cm} (53)

Substitute $u^*$ in (15) to the HJB equation, we first compute the four terms on the right side of the HJB equation. Note that the minimization is reached when $u = u^*$. In the following computation, we will use the property that $v_{11} = \frac{v_{11}}{1}$ and $a\top b = b\top a$ for any vector $a$ and $b$.

The first term is:

$$\mathcal{L}(x, u^*, t) = \frac{1}{2} x\top x - a\top a + \frac{1}{2}\rho u^\top u^*$$

$$= \frac{1}{2} x\top x - a\top a + \frac{1}{2\rho} (v_1 + v_{11}x)\top (v_1 + v_{11}x)$$

$$= \frac{1}{2} x\top x - a\top a + \frac{1}{2\rho} v_1\top v_1 + \frac{1}{\rho} v_1\top v_{11}x + \frac{1}{2\rho} x\top v_{11}v_{11}x$$

$$= \frac{1}{2\rho} v_1\top v_1 + \frac{1}{\rho} x\top (v_{11} - a) + \frac{1}{2} x\top (v_{11} + I)x$$ \hspace{1cm} (54)

Note that in line 1 of the expansion of $\mathcal{L}$, we dropped the constant term $\frac{1}{2} a\top a$.  

23
The second term is a scalar: \( \text{tr}(V_{xx}(x,t)) = \frac{\beta^2}{2}\text{tr}(v_{11}) \). The third term is:

\[
V_x^\top (b - x + u^*) = (v_1 + v_{11}x)^\top (b - x - u^*) = (v_1 + v_{11}x)^\top (b - x - \frac{1}{\rho}(v_1 + v_{11}x))
\]

\[
= (v_1^\top b - \frac{1}{\rho}v_1^\top v_1) - (v_1^\top x + \frac{1}{\rho}v_1^\top v_{11}x + \frac{1}{\rho}v_{11}^\top v_1 x - b^\top v_{11} x) - x^\top v_{11}^\top x - \frac{1}{\rho}x^\top v_{11} v_{11}^\top x
\]

\[
= (v_1^\top b - \frac{1}{\rho}v_1^\top v_1) - x^\top (v_1 + \frac{2}{\rho}v_{11}v_1 - v_{11}b) - \frac{1}{2}\frac{\rho}{\rho}x^\top (2v_{11} + \frac{2}{\rho}v_{11}v_{11}^\top) x
\]

The fourth term is:

\[
\sum_{j=1}^U \lambda_j(t)(V(x + h_j(x),t) - V(x,t)) = \sum_{j=1}^U \lambda_j(t)(v_1^\top B^j x + \frac{1}{\rho}x^\top B^j^\top v_{11}(t) B^j x + \frac{1}{2}x^\top 2v_{11} B^j x)
\]

\[
= x^\top \sum_{j=1}^U \lambda_j(t) B^j + \frac{1}{2}x^\top \left( \sum_{j=1}^U \lambda_j(t) B^j B^j^\top + 2v_{11}^\top \Lambda \right) x
\]

where \( \Lambda(t) = \sum_{j=1}^U \lambda_j(t)B^j \). Next, we compute the left side of HJB equation as:

\[
-V_i = -v'_i(t) - x^\top v'_i(t) - \frac{1}{2}x^\top v''_i(t)x
\]

Then by comparing the coefficients for the scalar, linear and quadratic terms in both left-hand-side and right-hand-side of the HJB equation, we obtain three ODEs.

First, only consider all the coefficients quadratic in \( x \):

\[
-v''_i(t) = I + 2v_{11}(t)(-1 + \Lambda(t)) + \sum_{j=1}^U \lambda_j(t)B^j v_{11}(t)B^j - \frac{1}{\rho}v_{11}(t)v_{11}(t)
\]

Second, consider the linear term:

\[
-v'_i(t) = -a + (-1 + \Lambda^\top(t) - \frac{1}{\rho}v_{11}(t))v_i(t) + v_{11}(t)b
\]

Third, consider the scalar term:

\[
-v'_i(t) = b^\top v_i(t) + \frac{\beta^2}{2}\text{tr}(v_{11}(t)) - \frac{1}{2\rho}v_1^\top(t)v_1(t)
\]

Finally, we compute the terminal condition for the three ODEs by \( V(x(T),T) = \phi(x(T),T) \):

\[
V(X(T),T) = v_0(T) + x(T)^\top v_1(T) + \frac{1}{2}x(T)^\top v_{11}(T)x(T)
\]

\[
\phi(x(T),T) = -x(T)^\top a + \frac{1}{2}x(T)^\top x(T)
\]

Hence \( v_0(T) = 0 \), \( v_1(T) = -a \) and \( v_{11} = I \). Note here we drop the constant term \( \frac{1}{2}a^\top a \) in terminal cost \( \phi \).

Finally, we just need to use Algorithm 1 to solve the ODEs in (58) and (59) to obtain \( v_{11}(t) \) and \( v_1(t) \). Substituting \( v_{11}, v_1 \) to (53) leads to the optimal control policy.
F  Optimal Control Policy for Opinion Influence Maximization

We will provide the algorithm to solve the opinion influence maximization problem. The solving scheme is similar to the least square opinion shaping cost. The exact form of derivation is different due to the difference cost functions.

First, we choose $\omega = 1$ and restate the controlled opinion SDE in (10) as:

$$dx_i(t) = (b_i + u_i(x, t) - x_i(t))dt + \beta dw_i(t) + \sum_{j=1}^{U} \alpha_{ij} x_j(t) dN_j(t)$$

Putting it in the vector form, we have:

$$dx(t) = (b - x + u)dt + \beta dw(t) + h(x) dN(t)$$  \hspace{1cm} (63)

where the $j$-th column of $h(x)$ captures how much influence that $x_j$ has on all other users and is defined as $h_j(x) = B_j x$, where the matrix $B_j \in \mathbb{R}^{U \times U}$ and has the $j$-th column to be $(\alpha_{1j}, \cdots, \alpha_{Uj})^T$ and zero elsewhere. We substitute $f = b - x$, $g = \beta$ and $h$ to (13) and obtain the HJB equation as:

$$-\frac{\partial V}{\partial t} = \min_u \left\{ \mathcal{L}(x, u, t) + \frac{\beta^2}{2} \text{tr}(V_{xx}(x, t)) + V_x(x, t)^T (b - x(t) + u(t)) + \sum_{j=1}^{U} \lambda_j(t) (V(x + h_j(x), t) - V(x, t)) \right\}$$

For opinion influence maximization, we define the cost as follows. Suppose the goal is to maximize the opinion influence at each time on $[0, T]$, the instantaneous cost $\mathcal{L}$ is defined as:

$$\mathcal{L}(x, u, t) = -\sum_{j=1}^{U} x_i(t) + \frac{1}{2} \| u(t) \|^2 = -x(t)^T 1 + \frac{1}{2} \| u(t) \|^2$$

where $1$ is the column vector with each entry to be one. For the terminal cost, we have: $\phi(T) = -x(T)^T 1$.

Following the similar reasoning as the least square opinion guiding problem. Since the terminal cost $\phi$ is linear in the state $x$, the value function must be linear in $x$, since it is the optimal value of a linear function. Hence we set the value function $V(x, t)$ to be a linear function in $x$ with unknown coefficients $v_1(t) \in \mathbb{R}^U$ and $v_0(t) \in \mathbb{R}$:

$$V(x, t) = v_0(t) + v_1(t)^T x$$  \hspace{1cm} (65)

To find the optimal control, we substitute (65) to (64) and take the gradient of the right-hand-side of (64) with respect to $u$ and set it to $0$. This yields the optimal control policy:

$$u^*(t) = -\frac{1}{\rho} V_x = -\frac{1}{\rho} v_1(t)$$  \hspace{1cm} (66)

Next, we just need to compute $v_1(t)$ to find $u^*$. Substitute $u^*$ in (66) to the HJB equation, we will compute the four terms on the right side of the HJB equation and derive the ODEs by comparing the coefficients. Note that the minimization is reached when $u = u^*$.

First, $\mathcal{L}(x, u^*, t)$ is expanded as:

$$\mathcal{L}(x, u^*, t) = -x^T 1 + \frac{1}{2} \| u^* \|^2 = \underbrace{\frac{1}{2\rho} v_1^T v_1}_{\text{scalar}} - \underbrace{x^T 1}_{\text{linear}}$$

Since $V$ is linear in $x$, $V_{xx} = 0$. The third term is:

$$V_x^T (b - x + u^*) = v_1^T (b - x - \frac{1}{\rho} v_1) = v^T b - \frac{1}{\rho} v^T v_1 - x^T v_1$$

$$= \underbrace{v^T b}_{\text{scalar}} - \underbrace{\frac{1}{\rho} v^T v_1}_{\text{linear}} - \underbrace{x^T v_1}_{\text{linear}}$$
The fourth term is:
\[ \sum_{j=1}^{U} \lambda_j(t)(V(x + h_j(x), t) - V(x, t)) = \sum_{j=1}^{U} \lambda_j(t)v_1^\top h_j(x) = x^\top \Lambda^\top v_1 \] (69)

where \( \Lambda(t) = \sum_{j=1}^{U} \lambda_j(t)B^j \). Next, we compute the left-hand-side of HJB equation as:
\[ -V_t = -v_0'(t) - x^\top v_1'(t) \] (70)

Then by comparing the coefficients for the scalar and linear terms in both left side and right side of the HJB equation, we obtain two ODEs.

First, only consider all the coefficients linear in \( x \):
\[ v_1'(t) = 1 + v_1(t) - \Lambda^\top v_1(t) \] (71)

Second, consider the linear term:
\[ v_0'(t) = -\frac{1}{2\rho}v_1^\top v_1 - v_1^\top b + \frac{1}{\rho}v_1^\top v_1 = -v_1(t)^\top b + \frac{1}{2\rho}v_1(t)^\top v_1(t) \] (72)

Hence we just need to solve the ODEs (70) to obtain \( v_1 \) and then compute the optimal control \( u^*(t) \) from (66).

Finally we derive the terminal conditions for the above two ordinary differential equations. First, \( V(T) = \phi(T) = -x(T)^\top 1 \) holds from the definition of the value function. Moreover, from the function form of \( V \), we have \( v_0(T) + x^\top v_1(T) \). Hence by comparing the coefficients, we have \( v_0(T) = 0 \) and \( v_1(T) = -1 \).

With the above terminal condition and (71), we will use Algorithm 1 to solve for \( v_1(t) \) and obtain the optimal control policy.
G Additional Experiments

G.1 Synthetic Experiments

Synthetic experimental setup. We consider a network with 1000 users and simulate the opinion SDE on the observation window $[0, 10]$ by applying Euler forward method to compute the difference form of (10) with $\omega = 1$:

$$x_i(t_{k+1}) = x_i(t_k) + (b_i + u_i(t_k) - x_i(t_k)) \Delta t + \beta \Delta w_i(t_k) + \sum_{j=1}^{U} \alpha_{ij} x_j(t_k) \Delta N_j(t_k)$$

where the observation window is divided into 100 time stamps $\{t_k\}$, with interval $\Delta t = 0.1$. The Wiener increments $\Delta w_i$ is sampled from the normal distribution $\mathcal{N}(0, \sqrt{\Delta t})$ and the Hawkes increments $\Delta N_j(t_k)$ is computed by counting the number of events on $[t_k, t_{k+1})$ for user $j$. The events for each user is simulated by the Otaga’s thinning algorithm [25]. The thinning algorithm is essentially a rejection sampling algorithm where samples are first proposed from a homogeneous Poisson process and then samples are kept according to the ratio between the actual intensity and that of the Poisson process. We set the baseline opinion uniformly at random, $b_i \sim \mathcal{U}[-1, 1]$, noise level $\beta = 0.2$, adjacency matrix $(\alpha_{ij})$ with sparsity of 0.001 and $\alpha_{ij} \sim \mathcal{U}[0, 0.01]$, initial opinion $x_i(0) = -10$, and $\omega = 1$ for the exponential triggering kernel $\kappa_\omega$. We repeat simulation of the SDE for ten times and report average performance. We set the tradeoff (budget level) parameter $\rho = 10$, and our results generalize beyond this value.

Network visualization. We conduct control over four 1000-user networks with different initial and target states. Figure 5 shows our framework works efficiently.

Robustness. To guide the opinion diffusion, we need to learn the parameters of the diffusion model. Typically error exists between estimated parameters and ground truth. To investigate how our framework performs with this discrepancy, we generate data with 10 and 100 events per user and learn parameters by maximum likelihood estimation (see appendix H for the learning algorithm). Then we compare with CROSSENTROPY using both true and learned parameters, in terms of instantaneous cost and trajectory. Figure 6 (a) and (c) show that as the training data size increases, the parameters are close to ground truth. More importantly, even with parameters learned with 10 and 100 events per user, our instantaneous cost and trajectories are close to these of ground-truth, indicating robustness. (b) and (d) show that CROSSENTROPY has high variance due to inaccurate parameters.

G.2 Real-World Experiments

Rationale of the accuracy prediction scheme. We essentially evaluate the prediction performance on test datasets. Given different trajectories in the test data, we predict which one will reach the objective function better. This is done by measuring how similar the real trajectory is to the optimal trajectory.

Our evaluation scheme can also be understood as follows. If the control policy by a method is applied, we evaluate how well the predicted outcome coincides with the reality in the test set. We make prediction using both true and learned parameters, in terms of instantaneous cost and trajectory. Figure 7 shows our method can steer the opinion of the user once he joins the network. In Twitter, user6-8 join the network sequentially. First, user6 joins around the 16th minute. Before this time, his opinion is not controlled. He creates a link to user2, since there is an immediate drop in the opinion of user2. Next, user7 joins and connects to user6 around the 22nd minute and the opinion of user6 drops due to the negative influence of user7. Finally our framework steers their opinions to target state quickly.
Figure 5: Controlled opinion of four networks with 1,000 users. The first column is the description of opinion change. The second column shows the opinion value per user over time. The three right columns show three snapshots of the opinion polarity in the network with 50 sub-users at different times. Yellow means positive and blue means negative polarity. Since the controlled trajectory converges fast, we use time range of \([0, 5]\).

Parameters are same except for different initial and target state: Set index I to denote user 1-500 and II to denote the rest. 1st row: \(x_0 = -10, a = 10\). 2nd row: \(x_0 = -10, a(I) = -5, a(II) = 10\). 3rd row: \(x_0\) sampled uniformly from \([-10, 10]\) and sorted in decreasing order, \(a(I) = -10, a(II) = 5\). 4th row: \(x_0\) is same as (c), \(a = 10\).
Figure 6: Comparison when model parameters are learned with different sample sizes. (a) and (b) instantaneous cost; (c) and (d) opinion trajectory for one randomly sample user.

Figure 7: Experiments on node birth networks on LSOG task. Opinion trajectories of eight randomly sampled users by our method. Four users are in the network and other four join sequentially. Time unit is day for Meme and minute for Twitter.
H Parameter Estimation

To control the opinion dynamics, we first need to learn the parameters of the uncontrolled dynamics in (10). In this section, we present the efficient framework convex parameter learning. There are two sets of parameters: i) \( \{b, \beta\} \) are the coefficients of drift and diffusion processes in (10), and ii) \( \{\eta, A\} \) are the coefficients of jump process, i.e., the Hawkes process. The observed data is in the form of \( \mathcal{T} = \{(t_i, x_{v_i}, v_i)\}_{i=1}^{n} \). Each triplet means user \( v_i \) posts his opinion \( x_{v_i} \) at time \( t_i \) in the network.

Now we introduce our learning framework. Following the derivation for Vasicek process [19], a classic diffusion SDE that is similar to (10) but without the jump term, we will compute the marginal conditional density \( p(x_{v_{i+1}}|x_{v_i}) \) for each sample \( (t_i, x_{v_i}, v_i) \). In order to so, we first derive the close form solution of \( x(t_{i+1}) \) given \( x(t_i) \) and obtain Gaussian conditional density:

**Gaussian conditional density.** Set \( V(x, t) = x \exp(t) \), apply the Generalized Ito’s lemma in Theorem 3, and integrate the opinion dynamics (10) on \([t_i, t_{i+1}]\), then we have

\[
    x(t_{i+1}) = b + (x(t_i) - b) \exp(-\Delta_i) + \beta \int_{t_i}^{t_{i+1}} \exp(-(t_{i+1} - s))dw(s) \\
    + h(x(t_i))(N(t_{i+1}) - N(t_i)) \exp(\Delta_i)
\]

where \( \Delta_i = t_{i+1} - t_i \). From the conditional law [19], since \( dw(s) \) follows Gaussian distribution, we can see \( x(t_i) \) has the Gaussian conditional density with mean to be the sum of drift and jump term, and variance to be integral of the diffusion:

\[
    \mathbb{E}[x(t_{i+1})|x(t_i)] = b + (x(t_i) - b) \exp(-\Delta_i) + h(x(t_i))(N(t_{i+1}) - N(t_i)) \exp(\Delta_i) \quad (73)
\]

\[
    \text{Var}[x(t_{i+1})|x(t_i)] = \beta^2(1 - \exp(-2\Delta_i))I \quad (74)
\]

Next, we derive the marginal density of the above conditional Gaussian distribution at dimension \( v_{i+1} \). Since only one event happened at time \( t_{i+1} \) to user \( v_{i+1} \), the counting difference vector \( N(t_{i+1}) - N(t_i) \) has value 1 at the \( v_{i+1} \)-th dimension and zero elsewhere.

Hence the \( v_{i+1} \)-th dimension of \( h(x(t_i))(N(t_{i+1}) - N(t_i)) \) is \( A(v_i, v_{i+1})x_{v_i} \), where \( A \) is the network adjacency matrix. Substitute it to (73), we have marginal conditional density as:

\[
    \mathbb{E}[x_{v_{i+1}}|x_{v_i}] = b_{v_i} + (x_{v_i} - b_{v_i}) \exp(-\Delta_i) + A(v_i, v_{i+1})x_{v_i} \\
    \text{Var}[x_{v_{i+1}}|x_{v_i}] = \beta^2(1 - \exp(-2\Delta_i))
\]

With these two sufficient statistics, we can compute \( p(x_{v_{i+1}}|x_{v_i}) \) from these two statistics since it is a conditional Gaussian density.

**Point process likelihood.** Having computed the probability for opinion transition from \( x_{v_i} \) to \( x_{v_{i+1}} \), now we compute the probability that this opinion happens at \( v_{i+1} \) at \( t_{i+1} \) and no event happens between \( t_i \) and \( t_{i+1} \) from theory of survival analysis [1]:

\[
    P(N_{v_{i+1}}(t_{i+1}) - N_{v_{i+1}}(t_i) = 1) = \lambda_{v_{i+1}}(t_{i+1}) \prod_{v=1}^{U} \exp\left(-\int_{t_i}^{t_{i+1}} \lambda_v(t)dt\right)
\]

Finally, combining the Gaussian conditional density and the point process likelihood, the complete likelihood function for all samples is:

\[
    \ell(T) = \prod_{i=1}^{n} p(x_{v_{i+1}}|x_{v_i}) \lambda_{v_i}(t_i) \prod_{v=1}^{U} \exp\left(-\int_{0}^{T} \lambda_v(t)dt\right)
\]
For Hawkes process, we parametrize $\lambda(t)$ in (5). The parameters can be estimated by maximizing the likelihood, i.e., $\max_{\lambda \geq 0, \eta \geq 0, \beta \geq 0, b} \ell(T)$.

We can see the likelihood function is nicely decomposed into two parts. The Gaussian density part corresponds to Gaussian distribution using the property of Wiener process. The point process part is the likelihood for Hawkes processes. Moreover, since parameters $\{\eta, A\}$ are linear in $\lambda(t)$ and parameters $\{b, \beta\}$ are linear in the mean and standard deviation of Gaussian density, the overall objective is concave, and the global optimum can be found by many algorithms. In our experiments, we adapt the efficient algorithm developed for Hawkes process in previous work [33] to update $\{\eta, A\}$ and projected gradient descent algorithm to update $\{b, \beta\}$. 