Linking Micro Event History to Macro Prediction in Point Process Models

Yichen Wang Georgia Tech
Xiaojing Ye Georgia State University
Haomin Zhou Georgia Tech
Hongyuan Zha Georgia Tech
Le Song Georgia Tech

Abstract

User behaviors in social networks are microscopic with fine grained temporal information. Predicting a macroscopic quantity based on users' collective behaviors is an important problem. However, existing works are mainly problem-specific models for the microscopic behaviors and typically design approximations or heuristic algorithms to compute the macroscopic quantity. In this paper, we propose a unifying framework with a jump stochastic differential equation model that systematically links the microscopic event data and macroscopic inference, and the theory to approximate its probability distribution. We showed that our method can better predict the user behaviors in real-world applications.

1 Introduction

Online social platforms generate large-scale event data with fine-grained temporal information. The microscopic data captures the behavior of individual users. For example, the activity logs in Twitter contain the detailed timestamps and contents of tweets/retweets, and a user's shopping history on Amazon contains detailed purchasing times. With the prevalent availability of such data, a fundamental question is to inference the collective outcome of each user's behavior and make macroscopic predictions (Givon et al., 2004).

Macroscopic prediction has many applications. For example, in user behavior modeling, a practical question is to predict the expected popularity of a song, movie, or a post. It is important both to understand the information diffusion and to inform content creation and feed design on social services. For example, the business merchants may want to popularize new products to boost sales. Social media managers may want to highlight new posts that are more likely to become popular, while users may want to learn from properties of popular tweets to personalize their own. Furthermore, predicting the evolution of social groups, such as Facebook groups, can also help handle the threat of online terrorist recruitment. In summary, the macro prediction task focuses on computing the expected value $\mathbb{E}[x(t)]$ of a macro population quantity $x(t)$ at time t.

Existing works on macro prediction are in two classes. The first class of approaches is in the majority. They typically fit the micro events by learning parameterized point processes, and then design problem-specific algorithms to predict each individual behavior (Du et al., 2012; Yang & Zha, 2013; Du et al., 2013; Yu et al., 2015; Zhao et al., 2015; Gao et al., 2015). They overcome the limitation of feature based classification/regression works (Cheng et al., 2014; Shulman et al., 2016) that typically ignore event dynamics and require laborious feature engineering. The second class of work is recently proposed to solve the influence estimation problem (Chow et al., 2015). It only models and predicts in macro scope, and directly models the probability distribution of the macro quantity using a problem-specific Fokker-Planck equation.

The limitations of the first class is that they typically use sampling based approach to approximate individual user behaviors and estimate $\mathbb{E}[x(t)]$. Hence the accuracy is limited due to the approximations and heuristic corrections. The problem with the second class is that the exact equation coefficients are not computationally tractable in general and hence various approximation techniques need to be adopted. Hence the prediction accuracy heavily depends on the approximations of these coefficients. Furthermore, all prior works are problem-specific and methods for popularity prediction (Yu et al., 2015; Zhao et al., 2015; Gao et al., 2015) are not applicable to influence prediction (Du et al., 2013; Chow et al., 2015). Hence here is lack of a unifying framework to link micro data and models to macro predictions.

In this paper, we propose a unifying framework that links micro event data to macro prediction task. It consists of three main stages:

- *Micro model.* We first fit the intensity function of micro point process models to temporal event data using convex optimization algorithms.
- *Linkage Jump SDE.* We then formulate the jump stochastic differential equation (SDE) to link the fitted micro

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

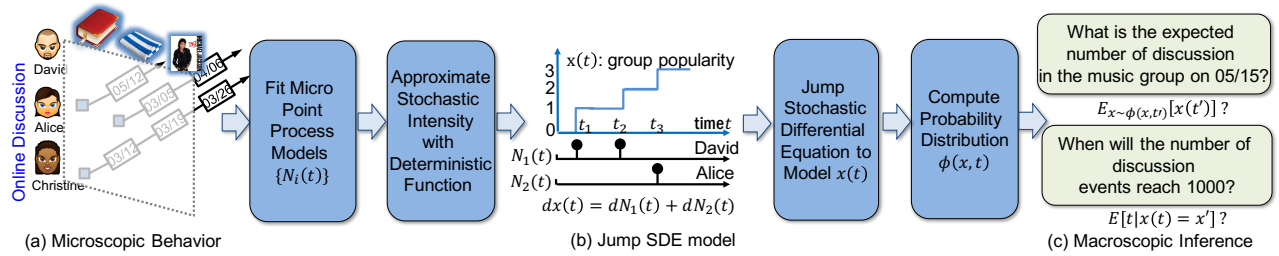


Figure 1: Framework illustration on group popularity prediction. (a) Micro events: users discuss in groups (book, shopping, music) at different times. (c) Macro inference tasks: predicting the group popularity $x(t)$, defined as # events in that group. Our work bridges the micro event data and the macro inference systematically with four steps in four blocks. (b) The proposed Jump SDE model for group popularity. Each black dot is the event triggered by the point process $N_i(t)$, which captures the users' discussion behaviors. $x(t)$ increases by 1 when David or Alice posts in the group.

models with the macro quantity of interest.

- *Macro inference.* Finally, we approximate the stochastic intensity function of point process with deterministic functions, and propose and solve a differential equation for the probability distribution of the macro quantity over time, and use it for inference.

Figure 1 summarizes the workflow of our framework. Specifically, we make the following contributions:

(i) Our framework is general and there is no need to design problem-special algorithms, hence avoiding the limitations of existing methods based only on micro or macro modeling. Our framework also links all the existing models for micro event to the macro prediction.

(ii) We propose an efficient convex optimization to fit the micro models, and a scalable Runge-Kutta algorithm for inference tasks.

(iii) Our framework has superior accuracy and efficiency performance in diverse real-world problems, outperforming problem-specific state-of-arts.

2 Background

Micro event data. We denote the collection of event data as $\{\mathcal{H}_i(\tau)\}$ in the time window $[0, \tau]$, where $\mathcal{H}_i(\tau) = \{t_k^i | t_k^i \leqslant \tau\}$ is the collection of history events triggered by user i, and the time t_k^i denotes the event timestamp.

Micro models. Temporal point processes are widely applied to model micro event data (Daley & Vere-Jones, 2007; Aalen et al., 2008; Zhou et al., 2013a,b; Yang & Zha, 2013; Farajtabar et al., 2014, 2015; He et al., 2015; Lian et al., 2015; Du et al., 2015, 2016; Wang et al., 2016b,c,d). Each user's behavior can be modeled as a point process. It is a random process whose realization consists of a list of discrete events localized in time, $\{t_k\}$ with $t_k \in \mathbb{R}^+$. It can be equivalently represented as a counting process, $N(t)$, which records the number of events before 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 history events. Let $\mathcal{H}(t^-) = \{t_k | t_k < t\}$ be the history of events happened up to but not including t. Formally, $\lambda(t) := \lambda(t|\mathcal{H}(t^-))$ is the conditional probability of observing an event in a window $[t, t + \mathrm{d}t)$ given $\mathcal{H}(t^-)$, *i.e.*,

$$\lambda(t)\mathrm{d}t = \mathbb{P}[\text{event in } [t, t + \mathrm{d}t)|\mathcal{H}(t^-)] = \mathbb{E}[\mathrm{d}N(t)|\mathcal{H}(t^-)]$$

where two events coincide with probability 0, *i.e.*, $\mathrm{d}N(t) \in \{0, 1\}$. The functional form of the intensity characterizes the point process. Commonly used forms include:

- Poisson process: Its intensity function is *deterministic* and independent of history.
- Hawkes process (Hawkes, 1971): It captures the mutual excitation phenomena between events:

$$\lambda(t) = \eta + \alpha \sum_{t_k \in \mathcal{H}(t^-)} \kappa(t - t_k) \tag{1}$$

  where $\eta \geqslant 0$ captures the long-term incentive to generate events. $\kappa(t) \geqslant 0$ models temporal dependencies, and $\alpha \geqslant 0$ quantifies how the influence from each past event evolves over time, making the intensity function depend on the history $\mathcal{H}(t^-)$ and a *stochastic process itself*.
- Self correcting process (Isham & Westcott, 1979). It seeks to produce regular event patterns with the inhibition effect of history events:

$$\lambda(t) = \exp\left(\eta t - \sum_{t_k \in \mathcal{H}(t^-)} \alpha\right) \tag{2}$$

  where $\eta, \alpha > 0$. The intuition is while the intensity increases steadily, when a new event appears, it is decreased by a constant factor $e^{-\alpha} < 1$. Hence the probability of new points decreases after an event has happened recently.

The key rationale of using point process for micro behaviors is that it models time as a continuous random variable. The event data is *asynchronously* since users can interact with each other at any time and there may not be any synchronization between events. Hence it is more appropriate to capture the uncertainty in real world than discrete-time models.

Macro prediction. Set $x(t)$ to be the *macro* population quantity of interest, such as # influenced users and # discussion events in a group, at time $t > \tau$. The task is to predict $\mathbb{E}[x(t)]$ given event data $\mathcal{H}(\tau)$ from all users.

This is a challenging problem since $x(t)$ is the result of collective behaviors, but the dynamics of each user is driven by different micro models with different stochasticity. Most existing works learn the point process models and predict each *individual* user's behavior. This introduces information loss, such as approximating the expectation (Zhao et al., 2015), sampling events (Du et al., 2013), and ignoring the stochasticity (Gao et al., 2015).

A more rigorous way for prediction is to compute or approximate the probability distribution of $x(t)$ given the history events $\mathcal{H}(t^-)$. However, there is no mathematical model that links the micro models to the macro quantity inference. Hence the existing works are specially designed for a specific problem due to the *missing* link.

In the next three sections, we will present the three stages of our inference framework. We first present how to fit the micro models in section 3, then in section 4 we approximate the stochastic intensity with deterministic functions, and present a jump SDE model and a differential equation that builds up the micro-macro connection. Finally, in section 5 we present an efficient numerical algorithm to compute the probability distribution and discuss different inference tasks.

3 Fitting Event Data to Microscopic Models

In this section, we will show the efficient convex optimization framework to fit micro point processes from event data. Specifically, we set $N_i(t)$ to be the point process user i. Its intensity function $\lambda_i(t, p_i)$ is parameterized with $p_i \in \mathcal{P}$ to capture the phenomena of interests, such as online discussions (Du et al., 2015; Wang et al., 2016a) and influence diffusion (Zhao et al., 2015; Chow et al., 2015; Gao et al., 2015). Given the event data $\mathcal{H}(\tau)$, we use maximum likelihood estimation (MLE) to learn parameters $\{p_i\}$.

From survival analysis theory Aalen et al. (2008), given a time t', for point process $N_i(t)$, the conditional probability that no event happens during $[t', t)$ is $S(t|\mathcal{H}(t')) = \exp\left(-\int_{t'}^{t} \lambda_i(\tau, p_i)\,\mathrm{d}\tau\right)$ and the conditional density $f(t|\mathcal{H}(t'))$ that an event occurs at time t as $f(t|\mathcal{H}(t')) = \lambda(t)S(t|\mathcal{H}(t'))$. Then given events $\mathcal{H}_i(\tau)$, we express its log-likelihood as:

$$\ell_i(p_i) = \sum\nolimits_k \log\left(\lambda_i(t_k^i, p_i)\right) - \int_0^T \lambda_i(t, p_i)\,\mathrm{d}t \qquad (3)$$

Hence the likelihood from the observed events $\mathcal{H}(\tau)$ generated by all point processes is just the summation of the likelihood of each point process: $\ell(\{p_i\}) = \sum_{\mathcal{H}_i \in \mathcal{H}} \ell_i(p_i)$. Furthermore, p_i is typically linear (convex) in the intensity function (Du et al., 2013; Zhou et al., 2013a; Yang & Zha, 2013; Farajtabar et al., 2014, 2015; Du et al., 2015; Wang et al., 2016b), and the objective function $\max_{p_i \in \mathcal{P}} \ell(\{p_i\})$ is concave. This is because the $\log(\cdot)$ function and the negative integration of the intensity are concave, and ℓ is nondecreasing w.r.t. the intensity. Hence it can be solved efficiently with many optimization algorithms, such as Quasi-Newton algorithm (Schmidt et al., 2009). In Section 6 we will discuss different parameterizations of intensity functions in diverse applications.

4 Linking Microscopic Models to Macroscopic Inference

In this section, we will first present the method to approximate the stochastic intensity function of point process. Then we present the jump SDE model, which links many works in micro information diffusion and user behavior modeling to a macro quantity. We then derive the the stochastic calculus rule for the jump SDE, and finally present the equation for computing the distribution of point processes with deterministic intensity.

4.1 Approximating Stochastic Intensity with Deterministic Functions

We propose to approximate the stochastic intensity function of each point process by only conditioning on its given history $\mathcal{H}(\tau)$, instead of $\mathcal{H}(t^-)$, which is unknown.

For example, for the Hawkes intensity in (1), we set $\mathcal{H}(t^-) \approx \mathcal{H}(\tau)$, and approximate its stochastic intensity only with events before time τ:

$$\lambda(t) \approx \eta + \alpha \sum\nolimits_{t_k \in \mathcal{H}(\tau)} \kappa(t - t_k)$$

Similarly, for the self-correcting process in (2), we approximate its stochastic intensity as follows:

$$\lambda(t) \approx \exp\left(\eta t - \sum\nolimits_{t_k \in \mathcal{H}(\tau)} \alpha\right)$$

Note that if the prediction time t is not far from τ, this approximation scheme works well, as shown in our experiment. In the following presentations of our method, we assume the point processes $\{N_i(t)\}$ are all *approximated with deterministic intensities*, and we use $\{\lambda_i(t)\}$ to denote the corresponding intensity computed using $\{\mathcal{H}_i(\tau)\}$.

4.2 Jump Stochastic Differential Equations

We formally define the class of point process driven stochastic differential equations (SDEs) as follows.

Definition 1. *The jump stochastic differential equation is a differential equation driven by one or more point processes.*

$$\mathrm{d}x(t) = \sum\nolimits_{i=1}^{m} h_i(x(t))\mathrm{d}N_i(t) \qquad (4)$$

where $x(t) : \mathbb{R}^+ \to \mathbb{N}$ is the state of the stochastic process. $N_i(t)$ is the i-th point process with deterministic intensity $\lambda_i(t)$. The jump amplitude function $h_i(x) : \mathbb{N} \to \mathbb{N}$ captures the influence of the point process. It has two parametric forms: $h_i = \pm 1$ or $h_i(x) = -x + b_i$, where $b_i \in \mathbb{N}$ is a constant state.

The jump SDE is a *continuous-time discrete-state* stochastic process. The state $x(t)$ is the *macro* quantity of interest. The point process $N_i(t)$ governs the generation of *micro* events for user i with the intensity $\lambda_i(t)$. We assume that there can only be at most one event triggered by one of the m point processes, *i.e.*, if $\mathrm{d}N_i(t) = 1$, $\mathrm{d}N_j(t) = 0$ for $j \neq i$. Hence each time only one point process will influence the state. The function $h_i(x)$ captures the influence with two forms.

(i) $h_i = \pm 1$. This term captures the *smooth* change and is the most common type of influence. It means the point process increases or decreases $x(t)$ by 1. For example, if a node is influenced by the information, # influenced users is increased by 1. Figure 1 shows an example.

(ii) $h_i = -x + b_i$. This term captures the *drastic* change. It is the special case where the point process changes current state to a fixed state b_i, *i.e.*, if $\mathrm{d}x = -x + b_i$, $x(t + \mathrm{d}t) = \mathrm{d}x(t) + x(t) = b_i$. For example, the population can suddenly decrease to a certain level due to the outbreak of a severe disease.

Moreover, the summation in (4) captures the collective influence of all point processes. Hence the jump SDE is the first model that systematically links all micro models to the macro quantity. It is general since one can plug in any micro point process models to (4). For simplicity of notation, we use x and $x(t)$ interchangeably.

4.3 Stochastic Calculus for Jump SDEs

Given the jump SDE, we derive the coppresponding stochastic calculus rule, which describes the differential of a function $g(x)$ with x driven by the jump SDE. It is fundamentally different from that for continuous-state SDEs.

Theorem 2. *Given the Jump SDE in (4), the differential form of function $g(x(t)) : \mathbb{N} \to \mathbb{R}$ is:*

$$\mathrm{d}g(x) = \sum\nolimits_{i=1}^{m} \big(g(x + h_i(x)) - g(x)\big)\mathrm{d}N_i(t) \qquad (5)$$

Theorem 2 describes the fact that $\mathrm{d}g(x)$ is determined by whether there is jump in $\mathrm{d}x$, which is modulated by each point process $N_i(t)$. Specifically, its influence in $g(x)$ is $g(x + h_i(x)) - g(x)$ and this term is modulated by its coefficient $\mathrm{d}N_i(t) \in \{0, 1\}$. Appendix A contains the proof.

Now we can derive the expectation of $g(x(t))$ as follows:

Corollary 3. *Given $x(\tau) = x_\tau$ and $\{\mathcal{H}_i(\tau)\}$, the expectation of $g(x(t))$, for $t > \tau$ satisfies the following equation:*

$$\mathbb{E}[g(x(t))] = \mathbb{E}\Big[\int_{\tau}^{t} \mathcal{A}[g](x(s))\mathrm{d}s\Big] + g(x_\tau)$$

where the functional operator $\mathcal{A}[g]$ is defined as:

$$\mathcal{A}[g]\big(x(t)\big) = \sum\nolimits_{i=1}^{m} \big(g(x + h_i) - g(x)\big)\lambda_i(t), \qquad (6)$$

and $\lambda_i(t)$ is the deterministic intensity.

Proof sketch. This corollary directly follows from integrating both sides of (5) on $[\tau, t]$ and taking the expectation. Appendix B contains proof details.

4.4 Probability Distribution

We are now ready to present the result that describes the time evolution of $\phi(x, t) := \mathbb{P}[x(t) = x]$, which is the probability distribution of $x(t)$ at time t. Specifically, we will derive a differential equation as follows.

Theorem 4. *Let $\phi(x, t)$ be the probability distribution for the jump process $x(t)$ in (4) given $\{\mathcal{H}_i(\tau)\}$, $\{\lambda_i(t)\}$ are deterministic intensities, then it satisfies the equation:*

$$\phi_t = -\sum\nolimits_{i=1}^{m} \lambda_i(t)\phi(x, t) + \sum\nolimits_{\mathcal{I}'} \delta(x - b_{i'})\lambda_{i'}(t) \qquad (7)$$
$$+ \sum\nolimits_{i^+ \in \mathcal{I}^+} \lambda_{i^+}(t)\phi(x - 1, t) + \sum\nolimits_{\mathcal{I}^-} \lambda_{i^-}(t)\phi(x + 1, t)$$

where $\phi_t := \frac{\partial \phi(x,t)}{\partial t}$, $\mathcal{I}' = \{i' | h_{i'}(x) = -x + b_{i'}\}$, $\mathcal{I}^+ = \{i | h_i = 1\}$. $\mathcal{I}^- = \{i | h_i = -1\}$. $\delta(x)$ is the delta function.

For the simplicity of explanation, we assume $\mathcal{I}' = \{i'\}$, $\mathcal{I}^+ = \{i^+\}$, $\mathcal{I}^- = \{i^-\}$, *i.e.*, there is only one entry in each index set. ϕ means the probability of state transition, the negative sign before ϕ in the first term of (7) means the transition starts from x, and the positive sign means the transition ends at x. the intensity $\lambda_i(t)$ is the transition rate.

This Theorem describes the transitions between current state x and two classes of states, including the smooth change to the general states $\{x - 1, x + 1\}$ and drastic change to a constant state $b_{i'}$. Next, we discuss each case in detail.

(i) $x \rightleftharpoons \{x - 1, x + 1\}$. λ_{i^+} captures the rate to jump by 1 from current state, hence it is the rate from $x - 1$ to x, and from x to $x + 1$. Similarly, λ_{i^-} captures the rate to decrease by 1 from current state.

(ii) $x \rightleftharpoons b_{i'}$. The delta function in (7) shows x drastically transits to $b_{i'}$ with rate $\lambda_{i'}$. The transition from $b_{i'}$ to x in the second row is a special case, and can only happen if $x = b_{i'}$. This is because of the transition rate is in the form of $\lambda_{i'}(t)\delta(x - b_{i'})$. It is only nonzero if $x = b_{i'}$. Hence the delta function is also a transition probability with probability 1 if $x = b_{i'}$ and 0 otherwise. This captures the case where some state $b_{i'}$ in the system can have *self transition*.

Proof sketch. We set the right-hand-side of (7) to be $\mathcal{B}[\phi]$. The main idea is to show $\sum_x g(x)\phi_t = \sum_x g(x)\mathcal{B}[\phi]$ holds for any test function g. Then $\phi_t = \mathcal{B}[\phi]$ follows from the Fundamental Lemma of Calculus of Variations (Jost & Li-Jost, 1998). To do this, we first show $\sum_x g(x)\phi_t = \sum_x \mathcal{A}[g]\phi$ using both Corollary 3 and the fact that each $\lambda_i(t)$ is a *deterministic intensity*. Then we show $\sum_x \mathcal{A}[g]\phi = \sum_x g(x)\mathcal{B}[\phi]$ by moving the operator $\mathcal{A}$ from the test function g to the distribution function ϕ. Appendix C contains proof details.

5 Macroscopic Prediction

In this section, we present an efficient algorithm to solve the differential equation, and show the flexibility of applying the probability distribution in many prediction tasks.

5.1 Equivalent Linear System of Equations

We first simplify the equation (7). It holds for each x, we show that it can be written as a system of differential equations. First, set the upper bound for x to be n, *e.g.*, in influence prediction, n is # users in the network. Hence the state space is $x = \{0, 1, 2, \cdots, n\}$. Next, we create a vector $\boldsymbol{\phi}(t)$ that collects the probability distribution of all possible states at t: $\boldsymbol{\phi}(t) = \big(\phi(0,t), \cdots, \phi(n,t)\big)^\top$

Hence $\boldsymbol{\phi}'(t) = \big(\phi_t(0,t), \cdots, \phi_t(n,t)\big)^\top$. To handle the case with constant state $b_{i'}$ and the delta function $\delta(x - b_{i'})$, $i' \in \mathcal{I}'$, we create a sparse vector $\boldsymbol{\mu}(t) \in \mathbb{R}^{n+1}$, with the $b_{i'}$-th entry as $\boldsymbol{\mu}(b_{i'}) = \lambda_{i'}(t)$ and 0 elsewhere. Specifically,

$$\boldsymbol{\mu}(t) = (0, \cdots, \underset{\text{the } b_{i'_1}\text{-th entry}}{\lambda_{i'_1}(t)}, \cdots, \underset{b_{i'_2}\text{-th entry}}{\lambda_{i'_2}(t)}, \cdots, 0)^\top$$

Now, we can express (7) as the following Ordinary Differential Equations:

$$\boldsymbol{\phi}'(t) = \boldsymbol{Q}(t)\boldsymbol{\phi}(t) + \boldsymbol{\mu}(t), \tag{8}$$

where $\boldsymbol{Q}(t)$ is a state transition matrix with $Q_{k,k}(t) = \sum_{i=1}^m \lambda_i(t), Q_{k,k+1}(t) = \sum_{i\in\mathcal{I}^-} \lambda_i(t), Q_{k,k-1}(t) = \sum_{i\in\mathcal{I}^+} \lambda_i(t)$ for $k = 1, \cdots, n+1$. The term $Q_{k,k}(t)$ on the diagonal is the rate from current state x to other states, $Q_{k,k-1}(t)$ is the rate from x to $x+1$, and $Q_{k,k+1}(t)$ is the rate from $x+1$ to x. $\boldsymbol{Q}(t)$ is a sparse matrix and it only has nonzero entries where there is state transition. It is tridiagonal and the number of nonzero entries is $O(n)$.

5.2 Numerical Algorithm

It is typically difficult to get an analytical solution to (8) since $\boldsymbol{Q}(t)$ is a function of time. We solve it numerically using the Runge-Kutta algorithm (Dormand & Prince, 1980). It is the classic method to solve ordinary differential equations. We first divide $[\tau, t]$ into timestamps $\{t_k\}_{k=0}^K$ with $\Delta t = t_k - t_{k-1}$, $t_0 = \tau$ and $t_K = t$. Then starting from $\boldsymbol{\phi}(\tau_0)$, the RK algorithm solves $\boldsymbol{\phi}(\tau_1)$, and use $\boldsymbol{\phi}(\tau_1)$ to solve $\boldsymbol{\phi}(\tau_2)$. We repeat such process until τ_K. This algorithm is a build-in ODE45 solver in MATLAB. Algorithm 1 summarizes the procedure.

Computation complexity. The main computation to obtain $\boldsymbol{\phi}(t_k)$ at each timestamp t_k is the matrix-vector product operation. Since $\boldsymbol{Q}$ is a sparse matrix with $O(n)$ nonzero entries, the complexity of this operation is $O(n)$, and our algorithm is quite scalable. For example, in influence estimation problem, our algorithm is efficient and only linear in the network size n.

Special case. If the intensity λ_i is a constant ($\boldsymbol{Q}$ is a constant matrix) and $\mathcal{I}' = \emptyset$ ($\boldsymbol{\mu} = \mathbf{0}$), the solution to (8) has an analytical form: $\boldsymbol{\phi}(t) = \exp(\boldsymbol{Q}t)\boldsymbol{\phi}(0)$, where $\exp(\boldsymbol{Q}t)$ is called the matrix exponential. See appendix for efficient algorithms with $O(n)$ complexity.

Algorithm 1 NUMERIC RUNGE KUTTA

1: **Input:** $\{\lambda_i(t)\}$, error tolerance ε, $\boldsymbol{\phi}_0$, time t,
2: **Output:** $\boldsymbol{\phi}(t)$
3: Discretize $[t_0, t]$ into $\{t_k\}$ with interval length Δt, $t_K = t$
4: Construct $\boldsymbol{Q}(t)$ from $\{\lambda_i(t)\}$ and $\boldsymbol{\mu}(t)$ from the model
5: $\{\boldsymbol{\phi}(t_k)\}$ = ODE45$([t_0, t], \boldsymbol{\phi}_0, \boldsymbol{Q}(t), \boldsymbol{\mu}, \Delta t, \varepsilon)$
6: $\boldsymbol{\phi}(t) = \boldsymbol{\phi}(t_K)$

5.3 Macroscopic Prediction Tasks

We discuss two prediction tasks as follows.

Size prediction. What is the expected value of x at time t'? We directly compute it using the definition of expectation:

$$\mathbb{E}[x(t')] = \sum\nolimits_x x\phi(x, t')$$

Time prediction. This is a new task and not considered in most prior works. What is the expected time when $x(t)$ reaches size x' on window $[\tau, t_f]$? We model the time as a random variable, $T := \inf\{t | x(t) = x'\}$.

We use $S(t)$ to denote the survival probability that size x' is not reached at time t. It is equal to the summation of probability $\phi(x,t)$ for each $x < x'$:

$$S(t) = \mathbb{P}[T > t] = \sum\nolimits_{x=0}^{x'-1} \phi(x, t)$$

Hence, from the theory of survival analysis (Aalen et al., 2008), the probability density of T is: $f(t) = -S'(t) = -\sum_{x=0}^{x'-1} \phi_t(x,t)$. Then $\mathbb{E}[T]$ is:

$$\mathbb{E}[T] = \int_\tau^{t_f} t f(t) \mathrm{d}t = -\int_\tau^{t_f} t \sum\nolimits_{x=0}^{x'-1} \phi_t(x,t) \mathrm{d}t$$

To compute this expectation, we set $t = t_f$ in Algorithm 1 and obtain $\boldsymbol{\phi}(t_k)$ for each timestamps t_k in the window $[\tau, t_f]$. Then the integral is computed as a Riemann sum:

$$\mathbb{E}[T] = -\sum\nolimits_k t_k \sum\nolimits_{x=0}^{x'-1} \phi_t(x, t_k)\Delta t$$

where $\boldsymbol{\phi}_t(t_k)$ is computed using (8). With the probability distribution, our work provides a unifying solution for these two inference tasks.

6 Applications

In this section, we show our framework unifies different applications. We will model event data using micro models, use the jump SDE model to link micro models to a macro quantity, and derive the differential equation.

Item Popularity Prediction. (Du et al., 2015) proposed to use Hawkes process to model users' recurrent behaviors, such as listening to a music or watching a TV program many

times. These repeated behaviors show the user's implicit interest to an item. This model has superior performance in recommending proper items to users at the right time compared with epoch based recommendation systems (Koren, 2009; Wang et al., 2015). Mathematically, this model use point process $N_{ui}(t)$ to model user u's interaction events to item i. Based on the model, we can further inference the item popularity $x(t)$, defined as the total number of events happened to the item up to t.

Micro model. This model parameterize the intensity function between user u and item i as follows:

$$\lambda_{ui}(t) = \eta_{ui} + \alpha_{ui} \sum\nolimits_{t_k^{u,i} \in \mathcal{H}^{u,i}} \kappa(t - t_k^{u,i}),$$

where $\eta_{ui} \geqslant 0$ is a baseline intensity and captures the users' inherent preference to items. $\kappa(t) = \exp(-t)$ is an exponential decaying triggering kernel, $\alpha_{ui} \geqslant 0$ is the magnitude of influence of each past event $t_k^{u,i}$, and $\mathcal{H}^{u,i}$ is the history events between user u and item i. Here, the occurrence of each historical event increases the intensity by a certain amount determined by the kernel and the weight. The parameters $(\eta_{ui}), (\alpha_{ui})$ are collected into user-by-item matrices and are assumed to be low rank, since users' behaviors and items' attributes can be categorized into a limited number of prototypical types. We follow (Du et al., 2015) and use the generalized conditional gradient algorithm to learn parameters by maximum likelihood estimation (MLE).

Jump SDE. For a item i, we set $x(t)$ to be the accumulative number of interaction events between each user u and i:

$$\mathrm{d}x(t) = \sum\nolimits_u \mathrm{d}N_{ui}(t)$$

Differential equation. From Theorem 4, we can derive the popularity distribution, $\phi^i(x,t)$ for item i as follows,

$$\phi_t^i = \sum\nolimits_u -\lambda_{ui}(t)\phi^i(x,t) + \lambda_{ui}(t)\phi^i(x-1,t)$$

Hence $\boldsymbol{Q}$ is a matrix with $Q_{kk}(t) = -\sum_u \lambda_{ui}(t)$ and $Q_{k,k-1}(t) = \sum_u \lambda_{ui}(t)$. Since at initial time there is no events with probability one, we set $\boldsymbol{\phi}(0) = (1, 0, \cdots, 0)^\top$.

Social Influence Prediction. The goal is to compute the expected number of nodes influenced by source nodes through information propagation over the network.

Micro model. Given a directed contact network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we use a continuous-time generative model for the information diffusion process (Du et al., 2013; Rodriguez et al., 2011). It begins with a set of infected source nodes, $\mathcal{S}(t_0)$, and the contagion is transmitted from the sources along their out-going edges to their direct neighbors. Each transmission through an edge entails a random spreading time, t_{ji}, drawn from a density over time, $f_{ji}(t_{ji})$. We assume transmission times are independent and possibly distributed differently across edges. Then, the infected neighbors transmit the contagion to their respective neighbors, and the process continues. We set $N_{ji}(t)$ to be the point process capturing the infection on the edge $j \to i$. Hence $\mathrm{d}N_{ji}(t) = 1$ means node i is infected by node j at time t. Set $\lambda_{ji}(t) = \alpha_{ji}$ to be the infection rate, then $f_{ji}(t_{ji})$ follows an exponential distribution, $f_{ji}(t_{ji}) = \alpha_{ji}\exp(-\alpha_{ji}t_{ji})$. We use the convex MLE algorithm (Rodriguez et al., 2011) to learn $\{\alpha_{ij}\}$.

Jump SDE. Since the infection can only happen if node j is already infected, we keep track of the set $\mathcal{S}(t)$ that includes the nodes that have been infected at t and $\mathcal{V} \setminus \mathcal{S}$ is the set of non-activated nodes. Denote $x(t)$ as the number of influenced nodes, then only the edges satisfying the condition $\mathcal{C}(t) = \{(j,i) \in \mathcal{E} | j \in \mathcal{S}(t), i \in \mathcal{V} \setminus \mathcal{S}(t)\}$ will be potentially influenced:

$$\mathrm{d}x(t) = \sum\nolimits_{(j,i)\in\mathcal{C}(t)} \mathrm{d}N_{ji}(t)$$

Differential equation. Applying Theorem 4 yields:

$$\phi_t = \sum\nolimits_{(j,i)\in\mathcal{C}(t)} -\alpha_{ji}\phi(x,t) + \alpha_{ji}\phi(x-1,t)$$

Hence $\boldsymbol{Q}$ is also a bi-diagonal matrix. Since initially all source nodes are activated, we set $\boldsymbol{\phi}(|\mathcal{S}|, 0) = 1$ and other components are 0.

7 Experiments

We evaluate our method, MIC2MAC (Micro to Macro), and show it leads to both accuracy and efficiency improvement on different problems: item popularity and influence prediction. For different problems, we compare with different competitors, which are problem specific. However, MIC2MAC is generic and works across different applications.

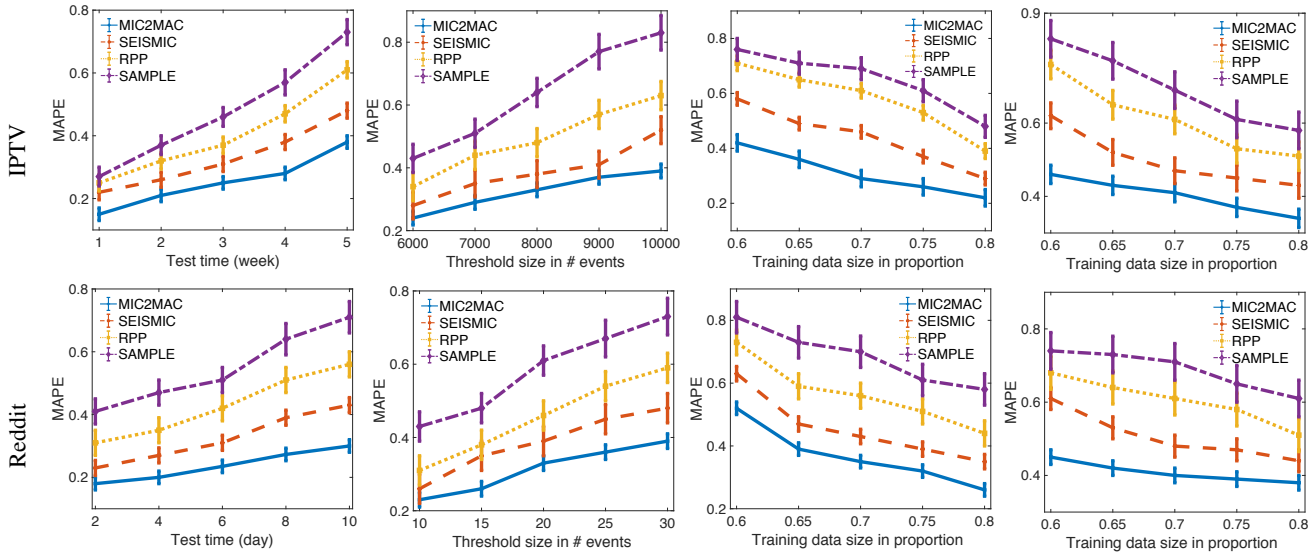
Evaluation scheme. We focus on the task: Given the users' micro behavior, can we forecast the future evolution of a macro quantity $x(t)$? We use the following metrics.

1. **Size prediction.** In the test data, we compute the mean absolute percentage error (MAPE) between estimated size $\hat{x}(t')$ and ground truth $x(t')$ at time t': $|\hat{x}(t') - x(t')|/x(t')$. For the item popularity task, the size is # events happened to the item. For influence estimation, the size is # infected nodes in the network.
2. **Time prediction.** We also predict when $x(t)$ reaches a threshold size x', and report the MAPE.

7.1 Experiments on Item Popularity Prediction

Datasets. Our datasets are obtained from two different domains including the TV streaming services (IPTV) and the online media services (Reddit). IPTV contains 7,100 users' watching history of 436 TV programs in 11 months, with 2,392,010 events. Reddit contains the online discussions of 1,000 users in 1,403 groups, with a total of 10,000 discussion events. We cleaned all bots' posts on Reddit. The code and data will be released once published.

Competitors. The state-of-arts use different point processes to model micro behaviors and different approximations or heuristics for inference. The parameters of these models are



(a) Size MAPE vs. test time (b) Time MAPE vs. threshold (c) Size MAPE vs. train size (d) Time MAPE vs. train size

Figure 2: Experiments on item popularity prediction. (a) predict the popularity (size) at different test times, which are the relative times after the end of train time; (b) predict the time when the size reaches different thresholds. The train data is fixed with 70% of total data for (a) and (b); (c) predict the size at final time by varying train data; (d) predict the time to reach the size of 8,000 (IPTV) and 20 (Reddit) by varying train data. Results are averaged over all items.

learned using MLE from training data. SEISMIC (Zhao et al., 2015) defines a self-exciting process with a post infectiousness factor and use the branching property for inference. It introduces several heuristics to correction factors to account for a long term decay. RPP (Gao et al., 2015) adds a reinforcement coefficient to Poisson process that depicts the self-excitation phenomena and discard the stochasticity in the system to make predictions. We also compare with a simple heuristic, SAMPLE that makes inference by simulating future events using Ogata's thinning algorithm (Ogata, 1981), we take the sample average of 1000 simulations to compute the expected size.

Prediction results. We use all events with $p \in (0, 1)$ proportion as the training data to learn parameters of all methods, and the rest as testing. The prediction performances on depends on different variates: (i) items, (ii) training data sizes, (iii) testing times t' for the size prediction, and (iv) threshold x' for time prediction. Hence we make predictions for each item and report the averaged results and vary MAPE as a function of (ii)-(iv). Figure 2 shows MIC2MAC significantly and consistently outperforms state-of-arts in different datasets on different prediction tasks.

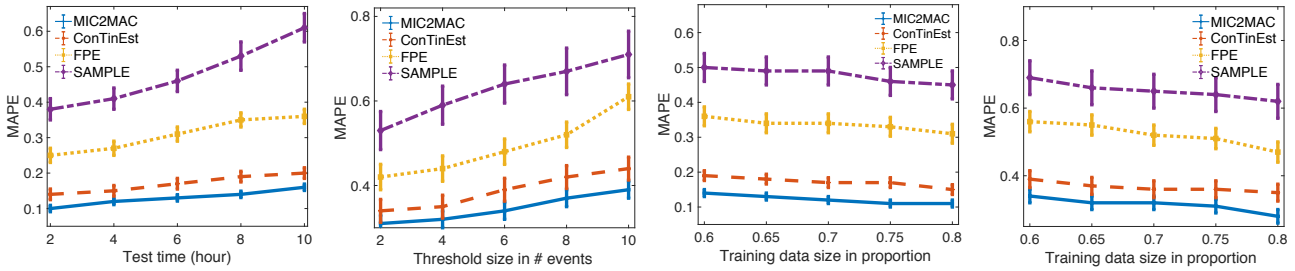
Size MAPE vs. test time. Figure 2 (a) shows that MAPE increases as test time increases. Since we fix the training data, and the farer the future, the more stochasticity and harder to predict. However, MIC2MAC has the smallest slope of MAPE vs. time, showing its robustness. Moreover, MIC2MAC has 10% accuracy improvement than SEISMIC. These two methods use different approximations, and the accuracy improvement suggests that the heuristic scheme in SEISMIC is less accurate compared with our intensity approximation. Moreover, RPP discard the stochasticity in prediction hence is less accurate than SEISMIC.

Time MAPE vs. threshold. The time prediction is especially novel and the competitors are not designed to predict time. For a fair comparison, we use the intensity function of SEISMIC, RPP, SAMPLE to simulate events and record the time that the size reaches the threshold. This simulation is repeated for 50 times for each threshold and the averaged time is reported. Figure 2 (b) shows the time MAPE increases as threshold increases. This is because train data size compared with the threshold is becoming smaller as the threshold increases. However, MIC2MAC is robust to the change of the threshold and the error only changes around 10% when the threshold changes from 6000 to 10000 on IPTV, while SEISMIC changes 20%. MIC2MAC is also 10% more accurate than SEISMIC. All competitors do not perform well since they use the sample average for prediction.

Size & Time MAPE vs. train size. Figure 2 (c) and (d) show that as the training data increases, MAPE decreases since more data leads to more accurate parameters. Our work also consistently performs the best with different training data.

7.2 Experiments on Influence Prediction

Dataset. We use the MemeTracker dataset (Leskovec et al., 2009). It contains information flows captured by hyperlinks between online media sites with timestamps. A site posts a piece of information and uses hyperlinks to refer to the same or closely related information posted by other sites. Hence a cascade is a collection of hyperlinks between sites that refer to the closely related information. In particular, we extract 321,362 hyperlink cascades among 1000 nodes.



(a) Size MAPE vs. test time (b) Time MAPE vs. threshold (c) Size MAPE vs. train size (d) Time MAPE vs. train size

Figure 3: Experiments on influence prediction. (a-b) training data fixed with 70% of total data. (c) predict size at final cascade time. (d) predict the time to reach threshold size of 4.

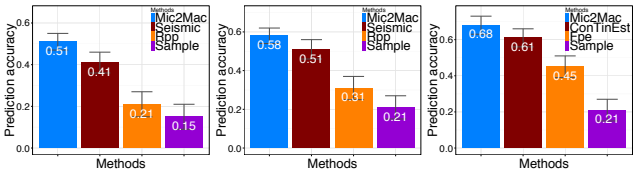
Competitors. CONTINEST (Du et al., 2013) is the state-of-art and uses kernel function to compute the number of infected nodes using Monte-Carlo sampling. It learns the model parameters using NETRATE (Rodriguez et al., 2011) with exponential transmission functions. FPE (Chow et al., 2015) is macroscopic method that directly computes the probability distribution, but it learns model parameters heuristically. We also add SAMPLE as a baseline.

Prediction results. To estimate influence on test set, we set $\mathcal{C}(u)$ to be the set of cascades in which u is the source node. Then # distinct nodes infected before t quantifies the real influence of node u. We also split the train and test data by proportion p. The results are averaged over all test cascades.

Size prediction. Figure 3 (a) shows that MIC2MAC has around 5% accuracy improvement over CONTINEST, and 20% improvement over FPE. This is important considering the collective influence sources. The individual improvement leads to significant improvement overall since the error accumulates considering all source nodes.

Time prediction. CONTINEST is not designed for this task. We collect its size output with different times as input and choose the one when the threshold is reached. FPE uses the same way as our method. SAMPLE predicts in the same way as the popularity experiment. Fig. 3 (b) shows our method is around $2\times$ more accurate than FPE. It highlights the importance of formulating the jump SDE model and using MLE to learn model parameters. Although FPE also computes the probability distribution, it learns the parameters heuristically without looking into the micro dynamics. Hence the less accurate parameters lead to less accurate prediction. Fig. 3 (c,d) further show that our method performs the best. The typical length of a cascade is small and around 4 nodes, the change of train data is also small, hence the curves are flat.

Rank prediction on two problems. Since MIC2MAC can predict the popularity of all items and the influence of all nodes, we also evaluate the rank prediction at the final time. Note that for the popularity problem, the final time for each item is the same, and is the universal final time of the dataset. For the influence problem, since each node has different start time of the infection, the final time is different for each node. Specifically, for all items we obtain two lists of ranks $\mathcal{L}$ and



(a) IPTV (b) Reddit (c) Memetracker

Figure 4: Rank prediction in different datasets.

$\hat{\mathcal{L}}$ according to the true and estimated size. Then the accuracy is evaluated by the Kendall-τ rank correlation (Kendall, 1938) between the two lists. A high value means the predicted and true sizes are strongly correlated. We vary the train size p from 0.6 to 0.8, and the error bar represents the variance over different sets. Figure 4 (a,b) show MIC2MAC performs the best, with accuracy more than 50% and consistently identifies 10% items more correctly than SEISMIC on the popularity prediction problem. (c) shows that it achieves accuracy of 68% with 7% improvement over CONTINEST on the influence prediction problem.

8 Conclusions

We have proposed a generic framework with a MLE algorithm to fit point process models to event data, a jump SDE model to link the micro behaviors to a macro quantity, and an equation for the probability distribution of the macro quantity. It has improved accuracy performance in diverse applications, and outperforms the state-of-arts that are specifically designed only for that application.

We point out the limitations of our method: for point process with stochastic intensity function, we use deterministic functions to approximate the intensity, which might be undesirable if (i) the prediction time is very far into the future, (ii) the intensity function is highly stochastic, or (iii) the model has intertwined stochasticities, such as the model that captures the co-evolution of information diffusion and network topology (Farajtabar et al., 2015). It remains as future work to consider all the stochasticity in point process models and develop efficient algorithms.

Acknowledgements. This project was supported in part by NSF DMS 1620342, NSF IIS 1639792, NSF DMS 1620345, NSF/NIH BIGDATA 1R01GM108341, ONR N00014-15-1-2340, NSF IIS-1218749, and NSF CAREER IIS-1350983.

References

Aalen, Odd, Borgan, Ornulf, and Gjessing, Hakon. *Survival and event history analysis: a process point of view*. Springer, 2008.

Brémaud, Pierre. Point processes and queues. 1981.

Cheng, Justin, Adamic, Lada, Dow, P Alex, Kleinberg, Jon Michael, and Leskovec, Jure. Can cascades be predicted? In *WWW*, 2014.

Chow, Shui-Nee, Ye, Xiaojing, Zha, Hongyuan, and Zhou, Haomin. Influence prediction for continuous-time information propagation on networks. *arXiv preprint arXiv:1512.05417*, 2015.

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

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

Du, Nan, Song, Le, Smola, Alexander J., and Yuan, Ming. Learning networks of heterogeneous influence. In *NIPS*, 2012.

Du, Nan, Song, Le, Gomez-Rodriguez, Manuel, and Zha, Hongyuan. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, 2013.

Du, Nan, Wang, Yichen, He, Niao, and Song, Le. Time sensitive recommendation from recurrent user activities. In *NIPS*, 2015.

Du, Nan, Dai, Hanjun, Trivedi, Rakshit, Upadhyay, Utkarsh, Gomez-Rodriguez, Manuel, and Song, Le. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, 2016.

Farajtabar, Mehrdad, Du, Nan, Gomez-Rodriguez, Manuel, Valera, Isabel, Zha, Hongyuan, and Song, Le. Shaping social activity by incentivizing users. In *NIPS*, 2014.

Farajtabar, Mehrdad, Wang, Yichen, Gomez-Rodriguez, Manuel, Li, Shuang, Zha, Hongyuan, and Song, Le. Coevolve: A joint point process model for information diffusion and network co-evolution. In *NIPS*, 2015.

Gao, Shuai, Ma, Jun, and Chen, Zhumin. Modeling and predicting retweeting dynamics on microblogging platforms. In *WSDM*, 2015.

Givon, Dror, Kupferman, Raz, and Stuart, Andrew. Extracting macroscopic dynamics: model problems and algorithms. *Nonlinearity*, 17(6):R55, 2004.

Hawkes, Alan G. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

He, Xinran, Rekatsinas, Theodoros, Foulds, James, Getoor, Lise, and Liu, Yan. Hawkestopic: A joint model for network inference and topic modeling from text-based cascades. In *ICML*, pp. 871–880, 2015.

Isham, V. and Westcott, M. A self-correcting pint process. *Advances in Applied Probability*, 37:629–646, 1979.

Jost, Jürgen and Li-Jost, Xianqing. *Calculus of variations*, volume 64. Cambridge University Press, 1998.

Kendall, Maurice G. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

Koren, Y. Collaborative filtering with temporal dynamics. In *KDD*, 2009.

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

Lian, Wenzhao, Henao, Ricardo, Rao, Vinayak, Lucas, Joseph E, and Carin, Lawrence. A multitask point process predictive model. In *ICML*, 2015.

Moler, Cleve and Van Loan, Charles. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review*, 45(1):3–49, 2003.

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

Rodriguez, Manuel, Balduzzi, David, and Schölkopf, Bernhard. Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the International Conference on Machine Learning*, 2011.

Schmidt, M., van den Berg, E., Friedlander, M. P., and Murphy, K. Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm. In *AISTAT*, 2009.

Shulman, Benjamin, Sharma, Amit, and Cosley, Dan. Predictability of popularity: Gaps between prediction and understanding. In *ICWSM*, 2016.

Wang, Xin, Donaldson, Roger, Nell, Christopher, Gorniak, Peter, Ester, Martin, and Bu, Jiajun. Recommending groups to users using user-group engagement and time-dependent matrix factorization. In *AAAI*, 2016a.

Wang, Yichen, Chen, Robert, Ghosh, Joydeep, Denny, Joshua C, Kho, Abel, Chen, You, Malin, Bradley A, and Sun, Jimeng. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *KDD*, 2015.

Wang, Yichen, Du, Nan, Trivedi, Rakshit, and Song, Le. Coevolutionary latent feature processes for continuous-time user-item interactions. In *NIPS*, 2016b.

Wang, Yichen, Theodorou, Evangelos, Verma, Apurv, and Song, Le. A stochastic differential equation framework for guiding online user activities in closed loop. *arXiv preprint arXiv:1603.09021*, 2016c.

Wang, Yichen, Xie, Bo, Du, Nan, and Song, Le. Isotonic hawkes processes. In *ICML*, 2016d.

Xue, Jungong and Ye, Qiang. Computing exponentials of essentially non-negative matrices entrywise to high relative accuracy. *Mathematics of Computation*, 82(283): 1577–1596, 2013.

Yang, Shuang-Hong and Zha, Hongyuan. Mixture of mutually exciting processes for viral diffusion. In *ICML*, pp. 1–9, 2013.

Yu, Linyun, Cui, Peng, Wang, Fei, Song, Chaoming, and Yang, Shiqiang. From micro to macro: Uncovering and predicting information cascading process with behavioral dynamics. In *ICDM*, 2015.

Zhao, Qingyuan, Erdogdu, Murat A., He, Hera Y., Rajaraman, Anand, and Leskovec, Jure. Seismic: A self-exciting point process model for predicting tweet popularity. In *KDD*, 2015.

Zhou, Ke, Zha, Hongyuan, and Song, Le. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *AISTAT*, 2013a.

Zhou, Ke, Zha, Hongyuan, and Song, Le. Learning triggering kernels for multi-dimensional hawkes processes. In *ICML*, 2013b.