# Modeling and Applications for Temporal Point Processes
## - Part I

**Hongteng Xu**

[1]Infinia ML, Inc.
[2]Department of ECE, Duke University

August 4, 2019

# Outline

- **Part I: Basics and typical models for TPPs**
  1. **Real-world event sequences**
  2. Temporal point processes and intensity functions
  3. Classic learning strategies
  4. Simulation and prediction
  5. Hawkes processes
  6. Open source packages
- Part II: Deep networks for temporal point processes
- Part III: Temporal point processes in practice

# Event sequences in real world: Earthquakes

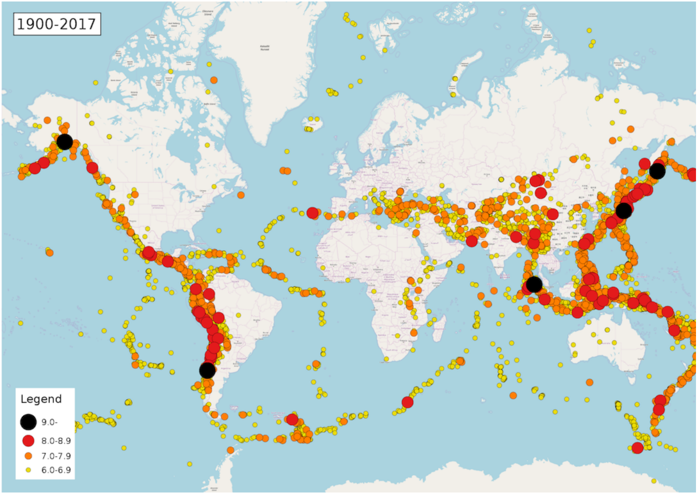



Figure 1: The locations and the intensities of the earthquakes from 1900 to 2017 [Ogata(1988)].

# Event sequences in real world: Social Networks

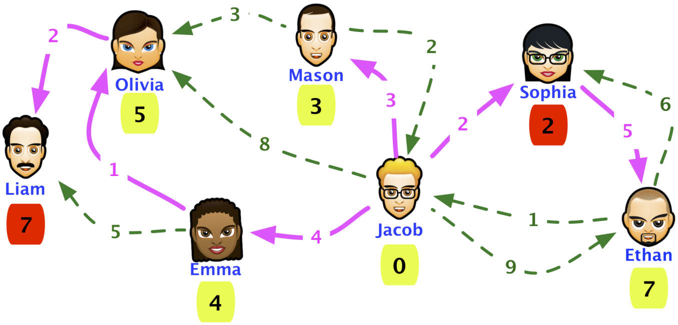



Figure 2: User behaviors on nets [Farajtabar et al.(2015), Zhao et al.(2015)].

# Event sequences in real-world: Patient Flows

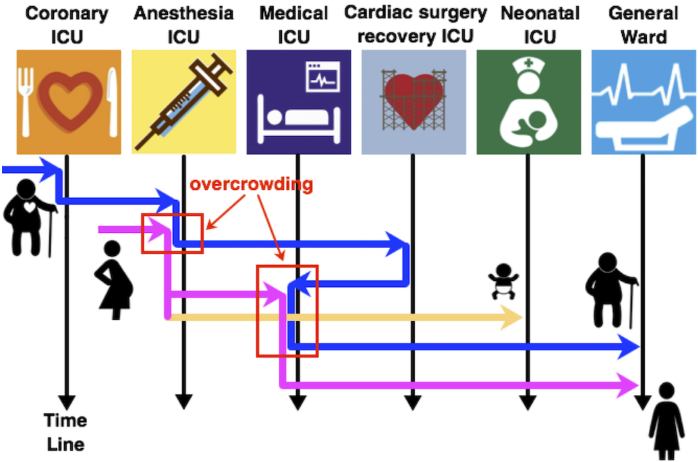



Figure 3: The transition behaviors of patients among different care units [Xu et al.(2016)a].

# Event sequences in real world: Conflicts

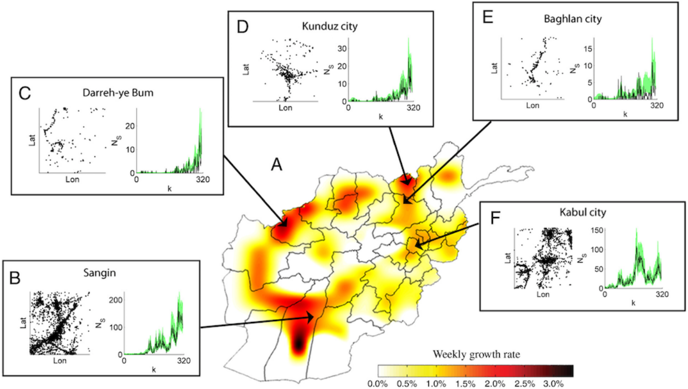



Figure 4: The Afghan war diary (AWD) in 320 weeks [Zammit et al.(2012)].

# Event sequence modeling

- Earthquakes
- Social networks
- Patient flow
- Conflicts
- Financial trades
- Taxi transports
- Online shopping
- ...

# Event sequence modeling

- Earthquakes
- Social networks
- Patient flow
- Conflicts
- Financial trades
- Taxi transports
- Online shopping
- ...

Asynchronous and interdependent event sequences: $\boldsymbol{s} = \{(t_i, d_i, f_i)\}_{i=1}^{I}$

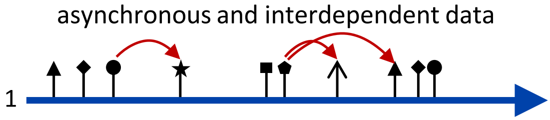
- Time stamps: $t_i \in [0, T]$.
- Entities (event types): $d_i \in \mathcal{D} = \{1, ..., D\}$.
- Optional Marks (features): $f_i \in \mathbb{R}^D$.

# Event sequence modeling

- Earthquakes
- Social networks
- Patient flow
- Conflicts
- Financial trades
- Taxi transports
- Online shopping
- ...

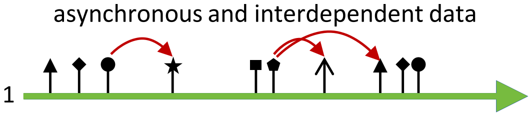
Asynchronous and interdependent event sequences: $\boldsymbol{s} = \{(t_i, d_i, f_i)\}_{i=1}^{I}$

- Time stamps: $t_i \in [0, T]$.
- Entities (event types): $d_i \in \mathcal{D} = \{1, ..., D\}$.
- Optional Marks (features): $f_i \in \mathbb{R}^D$.



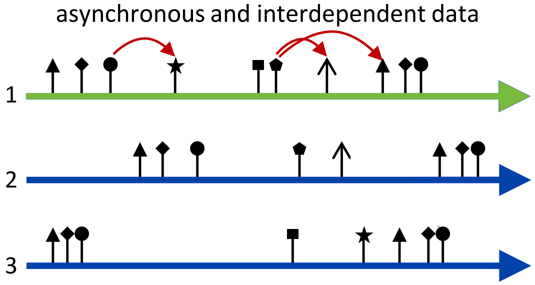
# Event sequence modeling

***Prob 1:***
***Learn triggering pattern (or called Granger causality) among events***



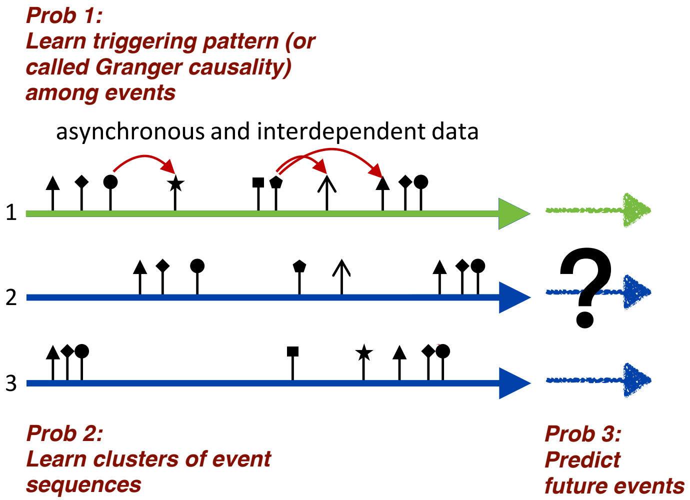
# Event sequence modeling

***Prob 1:***
***Learn triggering pattern (or called Granger causality) among events***



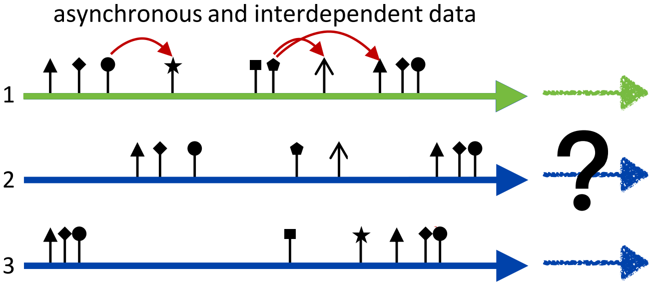

***Prob 2:***
***Learn clusters of event sequences***

# Event sequence modeling



# Event sequence modeling

***Prob 1:***
***Learn triggering pattern (or called Granger causality) among events***




***Prob 2:***
***Learn clusters of event sequences***

***Prob 3:***
***Predict future events***

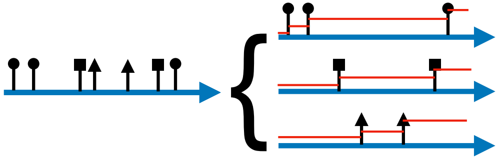
**How to describe/represent event sequences quantitatively?**

# Outline

- **Part I: Basics and typical models for TPPs**
  1. Real-world event sequences
  2. **Temporal point processes and intensity functions**
  3. Classic learning strategies
  4. Simulation and prediction
  5. Hawkes processes
  6. Open source packages
- Part II: Deep networks for temporal point processes
- Part III: Temporal point processes in practice

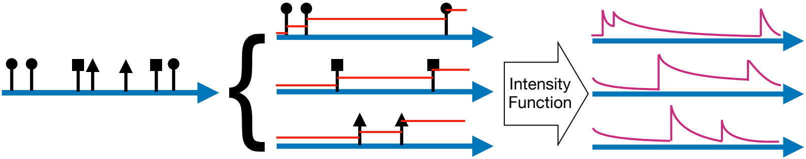
# Temporal point processes: Intensity functions

- **Event sequence:** $\boldsymbol{s} = \{(t_i, d_i)\}_{i=1}^{I}$, $d_i \in \mathcal{D} = \{1, ..., D\}$.
- *D*-**dimensional counting processes:** $N = \{N_d(t)\}_{d=1}^{D}$.
  $N_d(t)$ is the number of type-$d$ events occurring till time $t$.



# Temporal point processes: Intensity functions

- **Event sequence:** $\boldsymbol{s} = \{(t_i, d_i)\}_{i=1}^{I}$, $d_i \in \mathcal{D} = \{1, ..., D\}$.
- ***D*-dimensional counting processes:** $N = \{N_d(t)\}_{d=1}^{D}$. $N_d(t)$ is the number of type-$d$ events occurring till time $t$.




- **Intensity function:** The expected instantaneous happening rate of type-$d$ events given historical observations.

$$\lambda_d(t) = \frac{\mathbb{E}[\mathrm{d}N_d(t)|\mathcal{H}_{t_{last}}]}{\mathrm{d}t}, \ \mathcal{H}_{t_{last}} = \{(t_i, d_i)|t_i \leq t_{last}, d_i \in \mathcal{D}\}.$$

# Intensity functions and conditional probabilities

- **Intensity function:** The expected instantaneous happening rate of type-$u$ event given the history $\mathcal{H}_{t_{last}}$.

$$\lambda_d(t) = \frac{\mathbb{E}[\mathrm{d}N_d(t)|\mathcal{H}_{t_{last}}]}{\mathrm{d}t} = \frac{p(t, d|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})}.$$

- $p(t, d|\mathcal{H}_{t_{last}})$: the conditional probability density function (pdf) that type-$d$ event happens at time $t$ given history.
- $F(t|\mathcal{H}_{t_{last}})$: the conditional probability that there is at least one event happening in $(t_{last}, t]$ given history.

# Intensity functions and conditional probabilities

The overall intensity is

$$\lambda(t) = \sum_{d=1}^{D} \lambda_d(t)$$

# Intensity functions and conditional probabilities

The overall intensity is

$$\begin{aligned}\lambda(t) &= \sum_{d=1}^{D} \lambda_d(t) \\ &= \sum_{d=1}^{D} \frac{p(t, d|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} = \frac{p(t|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})}\end{aligned}$$

# Intensity functions and conditional probabilities

The overall intensity is

$$\begin{aligned}
\lambda(t) &= \sum_{d=1}^{D} \lambda_d(t) \\
&= \sum_{d=1}^{D} \frac{p(t, d|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} = \frac{p(t|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} \\
&= \frac{\frac{\mathrm{d}F(t|\mathcal{H}_{t_{last}})}{\mathrm{d}t}}{1 - F(t|\mathcal{H}_{t_{last}})} = -\frac{\mathrm{d}}{\mathrm{d}t} \log(1 - F(t|\mathcal{H}_{t_{last}})).
\end{aligned} \tag{1}$$

# Intensity functions and conditional probabilities

The overall intensity is

$$
\begin{aligned}
\lambda(t) &= \sum_{d=1}^{D} \lambda_d(t) \\
&= \sum_{d=1}^{D} \frac{p(t, d|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} = \frac{p(t|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} \qquad (1) \\
&= \frac{\frac{\mathrm{d}F(t|\mathcal{H}_{t_{last}})}{\mathrm{d}t}}{1 - F(t|\mathcal{H}_{t_{last}})} = -\frac{\mathrm{d}}{\mathrm{d}t} \log(1 - F(t|\mathcal{H}_{t_{last}})).
\end{aligned}
$$

Therefore we have

$$F(t|\mathcal{H}_{t_{last}}) = 1 - \exp\left(-\int_{t_{last}}^{t} \lambda(s)\mathrm{d}s\right), \qquad (2)$$

$$p(t|\mathcal{H}_{t_{last}}) = \lambda(t) \exp\left(-\int_{t_{last}}^{t} \lambda(s)\mathrm{d}s\right), \qquad (3)$$

# Intensity functions and conditional probabilities

The overall intensity is

$$\lambda(t) = \sum_{d=1}^{D} \lambda_d(t)$$
$$= \sum_{d=1}^{D} \frac{p(t, d|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} = \frac{p(t|\mathcal{H}_{t_{last}})}{1 - F(t|\mathcal{H}_{t_{last}})} \tag{1}$$
$$= \frac{\frac{\mathrm{d}F(t|\mathcal{H}_{t_{last}})}{\mathrm{d}t}}{1 - F(t|\mathcal{H}_{t_{last}})} = -\frac{\mathrm{d}}{\mathrm{d}t} \log(1 - F(t|\mathcal{H}_{t_{last}})).$$

Therefore we have

$$F(t|\mathcal{H}_{t_{last}}) = 1 - \exp\left(-\int_{t_{last}}^{t} \lambda(s)\mathrm{d}s\right), \tag{2}$$
$$p(t|\mathcal{H}_{t_{last}}) = \lambda(t) \exp\left(-\int_{t_{last}}^{t} \lambda(s)\mathrm{d}s\right), \tag{3}$$
$$p(t, d|\mathcal{H}_{t_{last}}) = \lambda_d(t) \exp\left(-\int_{t_{last}}^{t} \lambda(s)\mathrm{d}s\right), \tag{4}$$
$$p(d|t, \mathcal{H}_{t_{last}}) = \frac{\lambda_d(t)}{\lambda(t)}. \tag{5}$$

# Outline

- **Part I: Basics and typical models for TPPs**
  1. Real-world event sequences
  2. Temporal point processes and intensity functions
  3. **Classic learning strategies**
  4. Simulation and prediction
  5. Hawkes processes
  6. Open source packages
- Part II: Deep networks for temporal point processes
- Part III: Temporal point processes in practice

# Learning TPPs

- The key of learning a temporal point process $\{N_d\}_{d=1}^{D}$ is parametrizing and estimating its intensity functions, *i.e.*, $\{\lambda_d(t;\theta)\}_{d=1}^{D}$.

# Learning TPPs

- The key of learning a temporal point process $\{N_d\}_{d=1}^{D}$ is parametrizing and estimating its intensity functions, *i.e.*, $\{\lambda_d(t;\theta)\}_{d=1}^{D}$.
- Given a TPP model $\{\lambda_d(t;\theta)\}_{d=1}^{D}$, the common learning strategies include:
  - Maximum likelihood estimation.
  - Least-square estimation.
  - Discriminative learning.
- The convergence of MLE and that of LS are guaranteed. They can achieve unbiased estimation of intensity function.

# Learning TPPs

- The key of learning a temporal point process $\{N_d\}_{d=1}^{D}$ is parametrizing and estimating its intensity functions, *i.e.*, $\{\lambda_d(t;\theta)\}_{d=1}^{D}$.
- Given a TPP model $\{\lambda_d(t;\theta)\}_{d=1}^{D}$, the common learning strategies include:
  - Maximum likelihood estimation.
  - Least-square estimation.
  - Discriminative learning.
- The convergence of MLE and that of LS are guaranteed. They can achieve unbiased estimation of intensity function.
- Recently, the reinforcement learning of temporal point processes is considered in [Li et al.(2018)].

# Learning TPPs: MLE

Given an event sequence, *i.e.*, $\boldsymbol{s} = \{(t_i, u_i)\}_{i=1}^{I}$, we can write the likelihood function as

$$
\begin{aligned}
L(\boldsymbol{s}; \{\lambda_d\}_{d=1}^{D}) &= \prod_{i=1}^{I_n} p(t_i, d_i | \mathcal{H}_{t_{i-1}}) \times (1 - F(T|\mathcal{H}_{t_I})) \\
&\overset{\text{Eqs.(2,4)}}{=} \prod_{i=1}^{I} \lambda_{d_i}(t_i) \exp\left(-\int_{t_{i-1}}^{t_i} \lambda(s)\mathrm{d}s\right) \times \exp\left(-\int_{t_I}^{T} \lambda(s)\mathrm{d}s\right) \qquad (6) \\
&= \prod_{i=1}^{I} \lambda_{d_i}(t_i) \times \exp\left(-\int_{0}^{T} \lambda(s)\mathrm{d}s\right).
\end{aligned}
$$

# Learning TPPs: MLE

Given an event sequence, *i.e.*, $\boldsymbol{s} = \{(t_i, u_i)\}_{i=1}^{I}$, we can write the likelihood function as

$$\begin{aligned}
L(\boldsymbol{s}; \{\lambda_d\}_{d=1}^{D}) &= \prod_{i=1}^{I_n} p(t_i, d_i | \mathcal{H}_{t_{i-1}}) \times (1 - F(T|\mathcal{H}_{t_I})) \\
&\overset{\text{Eqs.(2,4)}}{=} \prod_{i=1}^{I} \lambda_{d_i}(t_i) \exp\left(-\int_{t_{i-1}}^{t_i} \lambda(s) \mathrm{d}s\right) \times \exp\left(-\int_{t_I}^{T} \lambda(s) \mathrm{d}s\right) \quad (6) \\
&= \prod_{i=1}^{I} \lambda_{d_i}(t_i) \times \exp\left(-\int_{0}^{T} \lambda(s) \mathrm{d}s\right).
\end{aligned}$$

Accordingly, given a set of event sequences $\mathcal{S} = \{\boldsymbol{s}_n\}_{n=1}^{N}$, we can learn the TPP model $\{\lambda_d(t)\}_{d=1}^{D}$ by maximum likelihood estimation (MLE) [Zhou et al.(2013), Xu et al.(2016)]:

$$\min_{\{\lambda_d\}_{d=1}^{D}} -\sum_{\boldsymbol{s} \in \mathcal{S}} \log L(\boldsymbol{s}; \{\lambda_d\}_{d=1}^{D}) + \alpha R(\{\lambda_d\}_{d=1}^{D}), \quad (7)$$

# Learning TPPs: Least-Square (LS) Estimation

The idea of least-square estimation is very straightforward — fitting the observed counting processes via the integral of intensity functions [Wang et al.(2016)]:

$$\min_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \sum_{d=1}^{D} \Big[ \hat{N}_d(t_i) - \int_0^{t_i} \lambda_d(s) \mathrm{d}s \Big]^2. \tag{8}$$

# Learning TPPs: Least-Square (LS) Estimation

The idea of least-square estimation is very straightforward — fitting the observed counting processes via the integral of intensity functions [Wang et al.(2016)]:

$$\min_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \sum_{d=1}^{D} \Big[ \hat{N}_d(t_i) - \int_0^{t_i} \lambda_d(s) \mathrm{d}s \Big]^2. \tag{8}$$

Because the variance $\mathbb{V}[N_d(t) - \int_0^t \lambda_d(s) ds] \sim \mathcal{O}(t^2)$, the work in [Xu et al.(2017)b] further modifies the objective function as

$$\min_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \sum_{d=1}^{D} \frac{1}{t_i^2} \Big[ \hat{N}_d(t_i) - \int_0^{t_i} \lambda_d(s) \mathrm{d}s \Big]^2. \tag{9}$$

# Learning TPPs: Least-Square (LS) Estimation

Or, we can define a contrast function [Bacry et al.(2017)a]:

$$C(\{\lambda_d\}) = \sum_{d=1}^{D} \int_0^T \lambda_d^2(s)ds - 2\int_0^T \lambda_d(s)\mathrm{d}\hat{N}_d(s), \tag{10}$$

and learn the TPP by minizing the expectation of the contrast function (fitting the empirical intensity function directly under $L^2$ error) [Bacry et al.(2017)a, Eichler et al.(2017)]:

$$\begin{aligned}&\arg\min_{\{\lambda_d\}_{d=1}^{D}} \mathbb{E}[C(\{\lambda_d\})]\\ =&\arg\min_{\{\lambda_d\}_{d=1}^{D}} \sum_{d=1}^{D}\mathbb{E}[(\lambda_d(t)-\hat{\lambda}_d(t))^2],\end{aligned} \tag{11}$$

The empirical intensity function is the differential of discretized counting process:

$$\hat{\lambda}_d(t) = \frac{\hat{N}_d(t+\Delta t) - \hat{N}_d(t)}{\Delta t}, \tag{12}$$

# Learning TPPs: Discriminative Learning

Sometimes, the data are insufficient to estimate likelihood and the main task is predict event types given timestamps, we can consider the discriminative learning of TPPs — maximizing the conditional probability $p(d|t, \mathcal{H}_{t_{last}})$ given observations.

$$\begin{aligned}&\max_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \log p(d_i|t_i, \mathcal{H}_{t_{i-1}})\\ =&\max_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \log \frac{\lambda_{d_i}(t_i)}{\lambda(t_i)}\end{aligned} \tag{13}$$

# Learning TPPs: Discriminative Learning

Sometimes, the data are insufficient to estimate likelihood and the main task is predict event types given timestamps, we can consider the discriminative learning of TPPs — maximizing the conditional probability $p(d|t, \mathcal{H}_{t_{last}})$ given observations.

$$\begin{aligned}&\max_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \log p(d_i|t_i, \mathcal{H}_{t_{i-1}})\\=&\max_{\{\lambda_d\}_{d=1}^{D}} \sum_{i=1}^{I} \log \frac{\lambda_{d_i}(t_i)}{\lambda(t_i)}\end{aligned} \tag{13}$$

When $\lambda_d(t) = \exp(f_d(t))$, where $f_d(t)$ is an arbitrary function (*e.g.*, a neural network), Eq. (13) corresponds to a softmax regression problem [Xu et al.(2016)a].

# Gradient-based learning

- All the learning strategies above are rely on gradient-based learning.
- For some typical TPP models like Hawkes processes, the MLE can be achieved by an EM algorithm, which corresponds to projected gradient descent, and the LS estimation have closed form solutions.

# Gradient-based learning

- All the learning strategies above are rely on gradient-based learning.
- For some typical TPP models like Hawkes processes, the MLE can be achieved by an EM algorithm, which corresponds to projected gradient descent, and the LS estimation have closed form solutions.
- When the observed event sequences are independent, we can apply min-batch optimization.
- When the intensity function at time $t$ is mainly influenced by the historical events in $[t - \Delta t, t)$, which is common in practice, we can apply a sliding window to each sequence, and define min-batch on the corresponding sub-sequences.

# Outline

- **Part I: Basics and typical models for TPPs**
  1. Real-world event sequences
  2. Temporal point processes and intensity functions
  3. Classic learning strategies
  4. **Simulation and prediction**
  5. Hawkes processes
  6. Open source packages
- Part II: Deep networks for temporal point processes
- Part III: Temporal point processes in practice

# Simulation of TPPs: Ogata's modified thinning algorithm

- Given a predefined or pre-trained TPP $\{\lambda_d\}_{d=1}^{D}$, we can simulate new sequences and predict future behaviors.
- At time $t$, we need to find out where to place the next point $t_i > t$ and which type $d_i \in \mathcal{D}$ it is.

# Simulation of TPPs: Ogata's modified thinning algorithm

- Given a predefined or pre-trained TPP $\{\lambda_d\}_{d=1}^{D}$, we can simulate new sequences and predict future behaviors.
- At time $t$, we need to find out where to place the next point $t_i > t$ and which type $d_i \in \mathcal{D}$ it is.
- **Ogata's modified thinning algorithm** [Ogata(1981)] has been widely used to simulate sequences.
- The basic idea is
  1. Simulate a homogeneous Poisson process on some interval $[t, t + L(t)]$ for some chosen distance function $L(t)$. The intensity of the Poisson process satisfies $m(t) \geq \sup_{s \in [t, t+L(t)]} \lambda(s)$.
  2. Thin out the points that are too many according to the real $\lambda(t)$, *e.g.*, keep a point at $t_i$ with probability $\frac{\lambda(t_i)}{m(t)}$.

# Simulation of TPPs: Ogata's modified thinning algorithm

Given a TPP model $\{\lambda_d\}_{d=1}^{D}$, we can simulate an event sequence in $[0, T]$ using the following steps:

1. Set $t = 0$, $i = 0$
2. Repeat till $t > T$:
   - Compute $L(t)$ and a constant intensity $m(t)$ in $[t, t + L(t)]$.
   - Simulate a Poisson process: $\Delta t \sim \exp(m(t))$, $u \sim \text{Unif}[0, 1]$.

# Simulation of TPPs: Ogata's modified thinning algorithm

Given a TPP model $\{\lambda_d\}_{d=1}^{D}$, we can simulate an event sequence in $[0, T]$ using the following steps:

1. Set $t = 0$, $i = 0$
2. Repeat till $t > T$:
   - Compute $L(t)$ and a constant intensity $m(t)$ in $[t, t + L(t)]$.
   - Simulate a Poisson process: $\Delta t \sim \exp(m(t))$, $u \sim \text{Unif}[0, 1]$.
   - If $\Delta t < L(t)$ and $t + \Delta t < T$ and $\underbrace{u \leq \frac{\lambda(t + \Delta t)}{m(t)}}_{\textit{thinning criterion}}$:

     $i = i + 1$,
     $t_i = t + \Delta t$. (a new time stamp)
     $d_i \sim [\frac{\lambda_1(t_i)}{\lambda(t_i)}, ..., \frac{\lambda_D(t_i)}{\lambda(t_i)}]$. (a new event type)
   - $t = t + \min(\{L(t), \Delta t\})$.
3. Output $\boldsymbol{s} = \{(t_i, d_i)\}_{i=1}^{I}$.

# Simulation of TPPs: Prediction

Given a TPP model $\{\lambda_d\}_{d=1}^{D}$ and its observations in $[0, T]$, we can make predictions for the events in the future, $(T, T + \Delta t]$.

- If $\Delta t$ is very small, we can make instantaneous predictions on the probability of type-$d$ event:

$$p(d|T + \Delta t, \mathcal{H}_T) = \frac{\lambda_d(T + \Delta t)}{\lambda(T + \Delta t)}. \tag{14}$$

# Simulation of TPPs: Prediction

Given a TPP model $\{\lambda_d\}_{d=1}^{D}$ and its observations in $[0, T]$, we can make predictions for the events in the future, $(T, T + \Delta t]$.

- If $\Delta t$ is very small, we can make instantaneous predictions on the probability of type-$d$ event:

$$p(d|T + \Delta t, \mathcal{H}_T) = \frac{\lambda_d(T + \Delta t)}{\lambda(T + \Delta t)}. \tag{14}$$

- If $\Delta t$ is large, we can make long-term predictions on the expected number of type-$d$ events in $(T, T + \Delta t]$ by simulation:

$$\frac{1}{K}\sum_{k=1}^{K}(\hat{N}_d^{(k)}(T + \Delta t) - N_d(T)). \tag{15}$$

# Outline

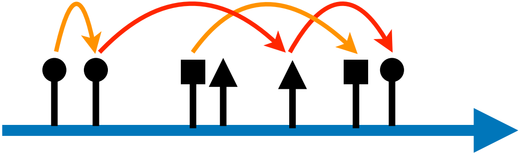
- **Part I: Basics and typical models for TPPs**
  1. Real-world event sequences
  2. Temporal point processes and intensity functions
  3. Classic learning strategies
  4. Simulation and prediction
  5. **Hawkes processes**
  6. Open source packages
- Part II: Deep networks for temporal point processes
- Part III: Temporal point processes in practice

# Hawkes process

Homogeneous Poisson process:

$$\lambda_d(t) = \mu_d \tag{16}$$

Simple, but memoryless...



**Hawkes process:** model the self- and mutually-triggering patterns hidden in event sequences explicitly [Hawkes(1971), Liniger(2009)].

## Hawkes process

The intensity functions of a $D$-dimensional Hawkes process, denoted as $HP(\boldsymbol{\mu}, \boldsymbol{\Phi})$, are

$$\begin{aligned}\lambda_d(t) &= \underbrace{\mu_d}_{exogenous} + \underbrace{\sum_{v=1}^{D} \int_0^t \phi_{dv}(t,s) dN_v(s)}_{endogenous\ \ triggering} \\ &= \mu_d + \sum_{t_i<t} \phi_{dd_i}(t, t_i)\end{aligned} \tag{17}$$

- $\boldsymbol{\mu} = [\mu_d] \geq \mathbf{0}$: **exogenous fluctuation** of the system.
- $\sum_{t_i<t} \phi_{dd_i}(t, t_i)$: **endogenous triggering term** caused the system's history.
- $\boldsymbol{\Phi} = [\phi_{dv}(t,s) \geq 0]$, $s \leq t$: **impact functions**, representing the influence of type-$v$ event at time $s$ on type-$d$ event at time $t$.
  - $\phi_{dd}(t,s)$: self-triggering pattern.
  - $\phi_{dv}(t,s)$, $d \neq v$: mutually-triggering pattern.

# Hawkes process: parametrization strategies

- We often assume that the impact functions are shift-invariant: $\phi_{dv}(t, s) = \phi_{dv}(t - s)$.
- The widely-used impact functions include:
  1. Exponential impact function [Zhou et al.(2013)]:

$$\phi_{dv}(t) = a_{dv} \exp(-wt). \tag{18}$$

  2. Basis representation [Xu et al.(2016)]:

$$\phi_{dv}(t) = \sum_{m=1}^{M} a_{dv}^{m} \kappa_m(t). \tag{19}$$

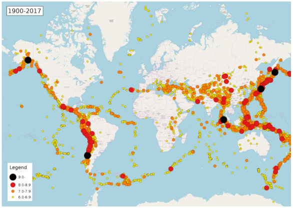
- Accordingly, the parameters of Hawkes process include the exogenous fluctuations $\boldsymbol{\mu} = [\mu_d]$ and the parameters of the impact functions $\boldsymbol{A} = [a_{dv}^{m}]$.

# Hawkes process

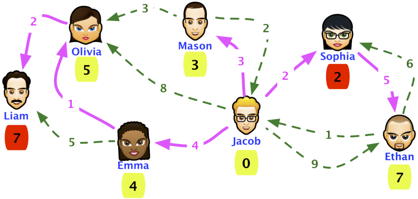
Hawkes process is important because

- Connections with real-world scenarios.
- Well-studied stationary properties.
- Explicit representation of Granger causality.
- High efficiency on learning.
- High efficiency on simulation.
- Superposition properties and robustness to data sparsity.

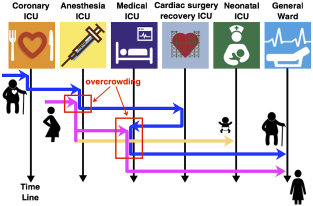
# Connections with real-world scenarios



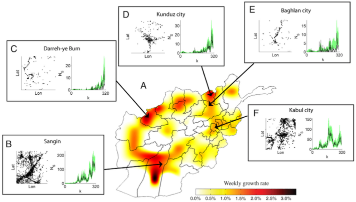

(a) Earthquakes




(b) Social networks




(c) Patient flow




(d) Conflicts

Figure 5: Illustrations of event sequences modeled by Hawkes processes.

# Explicit representation of Granger causality

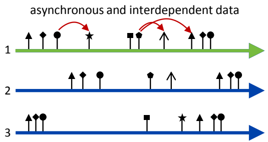
The impact functions not only decides the stationary of Hawkes processes but also provide us with an explicit representation of **Granger causality graph** of event types [Xu et al.(2016)].

| Scene | Entities | Sequences | Task |
|---|---|---|---|
| Patient admission | Diseases | Patients' admissions | Disease network |
| Job hopping | Companies | Employee's job history | Company network |
| Social network | Users | Users' interactions | User network |

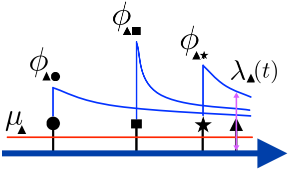
# Explicit representation of Granger causality

The impact functions not only decides the stationary of Hawkes processes but also provide us with an explicit representation of **Granger causality graph** of event types [Xu et al.(2016)].

| Scene | Entities | Sequences | Task |
|---|---|---|---|
| Patient admission | Diseases | Patients' admissions | Disease network |
| Job hopping | Companies | Employee's job history | Company network |
| Social network | Users | Users' interactions | User network |




(a) Observations

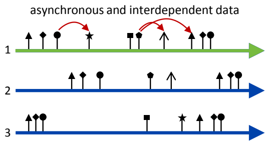



(b) Hawkes process

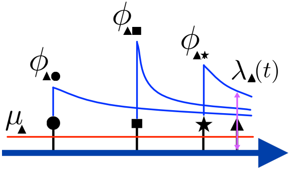
# Explicit representation of Granger causality

The impact functions not only decides the stationary of Hawkes processes but also provide us with an explicit representation of **Granger causality graph** of event types [Xu et al.(2016)].

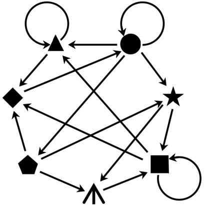
| Scene | Entities | Sequences | Task |
|---|---|---|---|
| Patient admission | Diseases | Patients' admissions | Disease network |
| Job hopping | Companies | Employee's job history | Company network |
| Social network | Users | Users' interactions | User network |




(a) Observations




(b) Hawkes process



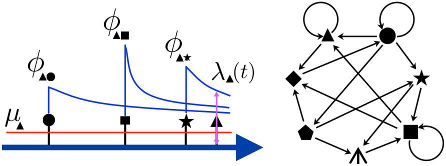
(c) Granger causality

Figure 6: Learning Granger causality graph based on Hawkes processes.

# Explicit representation of Granger causality

## Theorem ([Eichler et al.(2017)])

*For a Hawkes process,* $v \rightarrow d \notin \mathcal{E}$ *if and only if* $\phi_{dv}(t) \equiv 0$



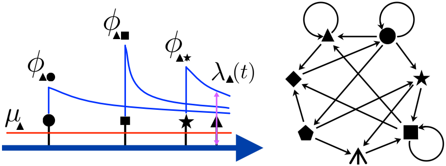

(a) Hawkes process (b) $G(\mathcal{D}, \mathcal{E})$

Figure 7: The sparsity of impact functions indicates $G(\mathcal{D}, \mathcal{E})$.

# Explicit representation of Granger causality

## Theorem ([Eichler et al.(2017)])

*For a Hawkes process, $v \to d \notin \mathcal{E}$ if and only if $\phi_{dv}(t) \equiv 0$*




(a) Hawkes process (b) $G(\mathcal{D}, \mathcal{E})$

Figure 7: The sparsity of impact functions indicates $G(\mathcal{D}, \mathcal{E})$.

Take MLE as an example [Zhou et al.(2013), Xu et al.(2016)]:

$$\phi_{dv} = a_{dv}\exp(-wt): \ \min_{\boldsymbol{\mu},\boldsymbol{A}\geq 0} -\sum_{\boldsymbol{s}\in\mathcal{S}} \log L(\boldsymbol{s};\boldsymbol{\mu},\boldsymbol{A}) + \alpha\|\boldsymbol{A}\|_1,$$
$$\phi_{dv} = \sum_m a_{dv}^m \kappa_m(t): \ \min_{\boldsymbol{\mu},\boldsymbol{A}\geq 0} -\sum_{\boldsymbol{s}\in\mathcal{S}} \log L(\boldsymbol{s};\boldsymbol{\mu},\boldsymbol{A}) + \alpha\|\boldsymbol{A}\|_{1,2},$$

# Explicit representation of Granger causality

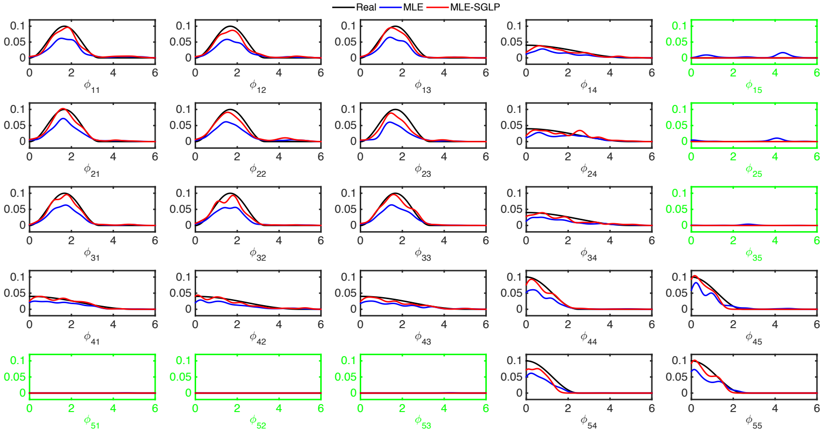



Figure 8: The regularizer imposes sparsity on impact functions.

# Explicit representation of Granger causality

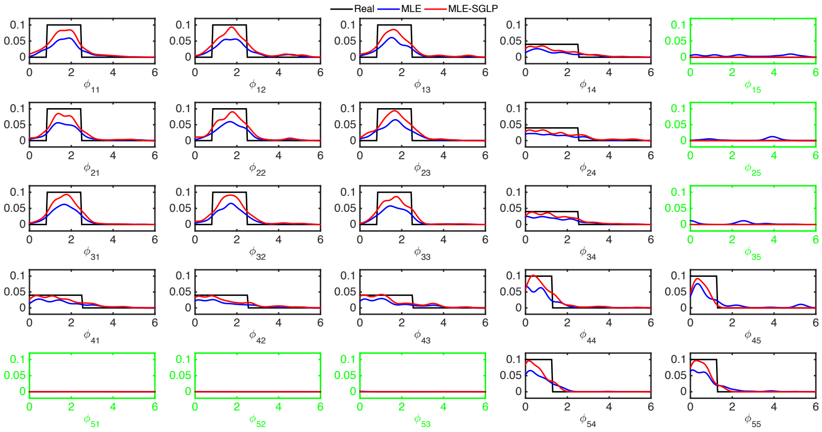



Figure 9: The learning of Granger causality graph is robust to model misspecficiation.

# High efficiency on learning

- For the Hawkes processes with $\phi_{dv}(t) = \sum_{m=1}^{M} a_{dv}^{m}\kappa_m(t)$, if $\{\kappa_m(t)\}_{m=1}^{M}$ are predefined. Both MLE and LS correspond to convex optimization.
- If $\{\kappa_m(t)\}_{m=1}^{M}$ are fast-decay functions, *e.g.*, exponential functions, we can truncate the history of each event and apply SGD on the batch of events.
- It is easy to impose structures on the impact functions, adding regularizers to the optimization problems.
- It is easy to take side information (features of events) into account, further parametrizing exogenous intensity and impact functions.

# Simulation: Acceleration of Ogata's thinning method

For some specific Hawkes processes, we can accelerate their simulations with the help of the **recursive representation of intensity functions**.

$$\lambda_d(t) = \mu_d + \sum\nolimits_{t_i<t} a_{dd_i} \exp(-w(t-t_i)) \tag{20}$$

If nothing happens in $(t, t+\Delta t]$:

$$\begin{aligned}
\lambda_d(t+\Delta t) &= \mu_d + \sum\nolimits_{t_i<t+\Delta t} a_{dd_i} \exp(-w(t+\Delta t-t_i)) \\
&= \mu_d + \exp(-w\Delta t) \sum\nolimits_{t_i<t} a_{dd_i} \exp(-w(t-t_i)) \\
&= \mu_d + \exp(-w\Delta t)(\lambda_d(t) - \mu_d)
\end{aligned}$$

# Simulation: Acceleration of Ogata's thinning method

For some specific Hawkes processes, we can accelerate their simulations with the help of the **recursive representation of intensity functions**.

$$\lambda_d(t) = \mu_d + \sum\nolimits_{t_i < t} a_{dd_i} \exp(-w(t - t_i)) \tag{20}$$

If nothing happens in $(t, t + \Delta t]$:

$$\begin{aligned}
\lambda_d(t + \Delta t) &= \mu_d + \sum\nolimits_{t_i < t + \Delta t} a_{dd_i} \exp(-w(t + \Delta t - t_i)) \\
&= \mu_d + \exp(-w\Delta t) \sum\nolimits_{t_i < t} a_{dd_i} \exp(-w(t - t_i)) \\
&= \mu_d + \exp(-w\Delta t)(\lambda_d(t) - \mu_d)
\end{aligned}$$

If there is one event $(t', d')$ happening in $(t, t + \Delta t]$:

$$\begin{aligned}
\lambda_d(t + \Delta t) &= \mu_d + \sum\nolimits_{t_i < t + \Delta t} a_{dd_i} \exp(-w(t + \Delta t - t_i)) \\
&= \mu_d + \exp(-w\Delta t)(\lambda_d(t) - \mu_d + a_{dd'} \exp(-w(t - t')))
\end{aligned}$$

# Simulation: Acceleration of Ogata's method

Recall Ogata's simulation method:

1. Set $t = 0$, $i = 0$
2. Repeat till $t > T$:
   - ~~Compute $L(t)$ and $m(t)$~~.
   - Simulate a Poisson process: $\Delta t \sim \exp(\lambda(t))$, $u \sim \text{Unif}[0, 1]$.
   - If $\Delta t < L(t)$ and $t + \Delta t < T$ and $u \leq \frac{\lambda(t+\Delta t)}{\lambda(t)}$:

     $i = i + 1$,

     $t_i = t + \Delta t$. (a new time stamp)

     $d_i \sim [\frac{\lambda_1(t_i)}{\lambda(t_i)}, ..., \frac{\lambda_D(t_i)}{\lambda(t_i)}]$. (a new event type)
   - $t = t + \Delta t$.
3. Output $\boldsymbol{s} = \{(t_i, d_i)\}_{i=1}^{I}$.

# Simulation: Acceleration of Ogata's method

Recall Ogata's simulation method:

1. Set $t = 0$, $i = 0$
2. Repeat till $t > T$:
   - ~~Compute $L(t)$ and $m(t)$~~.
   - Simulate a Poisson process: $\Delta t \sim \exp(\lambda(t))$, $u \sim \text{Unif}[0, 1]$.
   - If $\Delta t < L(t)$ and $t + \Delta t < T$ and $u \leq \frac{\lambda(t+\Delta t)}{\lambda(t)}$:

     $i = i + 1$,

     $t_i = t + \Delta t$. (a new time stamp)

     $d_i \sim [\frac{\lambda_1(t_i)}{\lambda(t_i)}, ..., \frac{\lambda_D(t_i)}{\lambda(t_i)}]$. (a new event type)
   - $t = t + \Delta t$.
3. Output $\boldsymbol{s} = \{(t_i, d_i)\}_{i=1}^{I}$.

**For the Hawkes processes with exponential impact functions, the intensity always decays when nothing happens**. Therefore, we have

- $L(t)$ can be $\infty$, and $m(t) = \sup_{s \in [t, t+L(t)]} \lambda(t) = \lambda(t)$.

# Simulation: Hawkes process and branch process

Furthermore, Hawkes process can be viewed as a branch process [Møller et al.(2006), Farajtabar et al.(2014)], whose intensity functions can be represented as **the superposition of Poisson processes' intensity functions**.

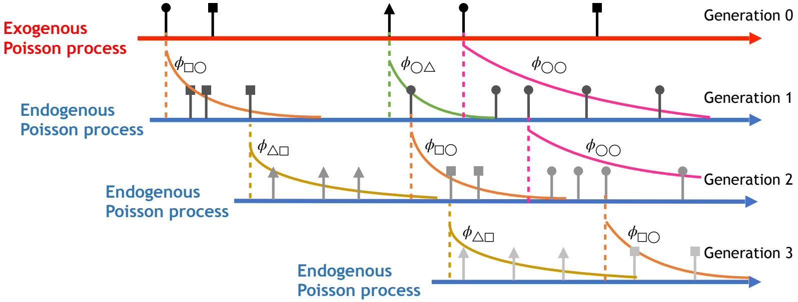



Figure 10: Hawkes process and branch process.

# Simulation based on branch clustering method

For the Hawkes process with $\lambda_d(t) = \mu_d + \sum_{t_i<t} \phi_{dd'}(t - t_i)$:

1. Simulate $\mathcal{S}^0 = \{(t_i^0, d_i^0)\}_{i=1}^{I_0}$ via a $D$-dimensional homogeneous Poisson process Poisson($\{\mu_d\}_{d=1}^{D}$) in $[0, T]$.
2. Set $\mathcal{S} = \mathcal{S}^0$.

# Simulation based on branch clustering method

For the Hawkes process with $\lambda_d(t) = \mu_d + \sum_{t_i<t} \phi_{dd'}(t - t_i)$:

1. Simulate $\mathcal{S}^0 = \{(t_i^0, d_i^0)\}_{i=1}^{I_0}$ via a $D$-dimensional homogeneous Poisson process Poisson$(\{\mu_d\}_{d=1}^D)$ in $[0, T]$.
2. Set $\mathcal{S} = \mathcal{S}^0$.
3. For the $k$-th generation, $k = 1, ..., K$:
   - Set $\mathcal{S}^k = \emptyset$.
   - For $(t_i^{k-1}, d_i^{k-1}) \in \mathcal{S}^{k-1}$:
     - Simulate a sequence $\boldsymbol{s}$ via a $D$-dimensional inhomogeneous Poisson process Poisson$(\{\phi_{dd_i^{k-1}}(t)\}_{d=1}^D)$ in $[t_i^{k-1}, T]$.
     - $\mathcal{S}^k = \mathcal{S}^k \cup \boldsymbol{s}$.
   - $\mathcal{S} = \mathcal{S} \cup \mathcal{S}^k$.
4. Output $\mathcal{S}$.

# Simulation: Comparisons

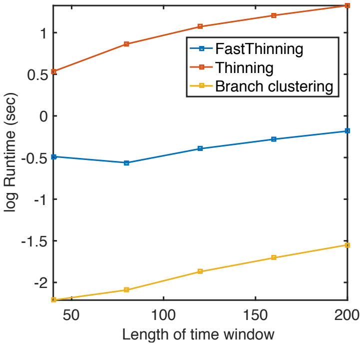



Figure 11: Comparisons for different simulation methods on runtime.

# Superposition property and its benefits

Given $N^k(t) \sim HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})$, $k = 1, ..., K$, how to $\boldsymbol{\Phi} = [\phi_{dv}(t)]$?

- **Multi-source+MHP:** Treat observed sequences as independent samples and learn $\{HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})\}_{k=1}^K$ accordingly.

# Superposition property and its benefits

Given $N^k(t) \sim HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})$, $k = 1, ..., K$, how to $\boldsymbol{\Phi} = [\phi_{dv}(t)]$?

- **Multi-source+MHP:** Treat observed sequences as independent samples and learn $\{HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})\}_{k=1}^K$ accordingly.

## Theorem (Superposition property [Xu et al.(2017)b])

*For $K$ independent Hawkes processes, i.e., $N^k(t) \sim HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})$, $k = 1, ..., K$, their superposition is still a Hawkes process, where $N(t) = \sum_{k=1}^K N^k(t)$ and $N(t) \sim HP(\sum_{k=1}^K \boldsymbol{\mu}^k, \boldsymbol{\Phi})$.*

# Superposition property and its benefits

Given $N^k(t) \sim HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})$, $k = 1, ..., K$, how to $\boldsymbol{\Phi} = [\phi_{dv}(t)]$?

- **Multi-source+MHP:** Treat observed sequences as independent samples and learn $\{HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})\}_{k=1}^K$ accordingly.

## Theorem (Superposition property [Xu et al.(2017)b])

*For $K$ independent Hawkes processes, i.e., $N^k(t) \sim HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})$, $k = 1, ..., K$, their superposition is still a Hawkes process, where $N(t) = \sum_{k=1}^K N^k(t)$ and $N(t) \sim HP(\sum_{k=1}^K \boldsymbol{\mu}^k, \boldsymbol{\Phi})$.*

- **Superposition+HP:** Superpose observed sequences and learn a single $HP(\boldsymbol{\mu}, \boldsymbol{\Phi})$.

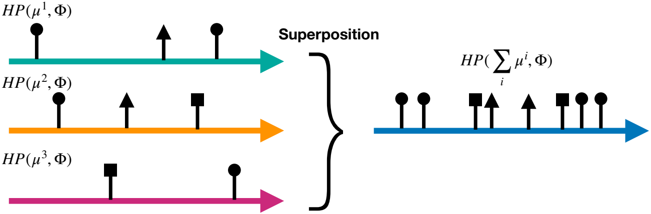



Figure 12: Learning superposed Hawkes processes.

# Superposition property and its benefits

1. **Multi-source+MHP:** Treat observed sequences as independent samples and learn $\{HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})\}_{k=1}^{K}$ accordingly.
2. **Superposition+HP:** Superpose observed sequences and learn a single $HP(\mu, \boldsymbol{\Phi})$.

## Theorem ([Xu et al.(2017)b])

*For $K$ $D$-dimensional Hawkes processes with $\phi_{dv}(t) = \sum_m a_{dv}^m \kappa_m(t)$, i.e., $HP(\boldsymbol{\mu}^k, \boldsymbol{A})$, $k = 1, ..., K$, suppose that*

- *Each observed sequence has $I$ events;*
- *The parameters are bounded as $\|\boldsymbol{\mu}^k\|_2^2 \leq B_\mu$ and $\|\boldsymbol{A}\|_F^2 \leq B_A$;*
- *The upper bound of $\|\sum_{k=1}^{K} \boldsymbol{\mu}^k\|_2^2$ is denoted as $B_{\Sigma\mu}$.*

*The bound on the excess risk of* **Superposition+HP** *is tighter if*

$$\begin{aligned} B_{\Sigma\mu} \leq & KB_\mu + D(K+D)B_\mu \log\Big(1 + \frac{KI}{D(K+D)}\Big) \\ & - D(1+D)B_\mu \log\Big(1 + \frac{KI}{D(1+D)}\Big). \end{aligned} \tag{21}$$

# Typical Cases

For $N^k(t) \sim HP(\boldsymbol{\mu}^k, \boldsymbol{\Phi})$, $k = 1, ..., K$

## Lemma (Typical Infeasible Condition)

*If* $\boldsymbol{\mu}^1 = \boldsymbol{\mu}^2 = ... = \boldsymbol{\mu}^K$*, the* **Multi-source+MHP** *strategy has a tighter bound of excess risk.*

## Lemma (Typical Feasible Condition)

*If* $\langle \boldsymbol{\mu}^k, \boldsymbol{\mu}^{k'} \rangle = 0$ *for all* $k \neq k'$*, the* **Superposition+HP** *strategy has a tighter bound of excess risk.*

# Benefits from superposed Hawkes processes

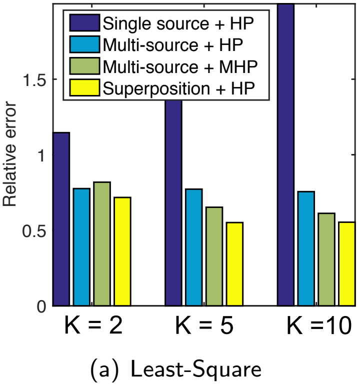



Figure 13: Comparisons based on LS and MLE, respectively.

Using superposition-based learning strategy, we can enhance the robustness to the problem of data insufficiency.

# Outline

- **Part I: Basics and typical models for TPPs**
  1. Real-world event sequences
  2. Temporal point processes and intensity functions
  3. Classic learning strategies
  4. Simulation and prediction
  5. Hawkes processes
  6. **Open source packages**
- Part II: Deep networks for temporal point processes
- Part III: Temporal point processes in practice

# Open source packages

Some toolboxes have been developed for TPPs.

- **Tick** [Bacry et al.(2017)b]
  `https://x-datainitiative.github.io/tick/index.html`
- **THAP** [Xu and Zha(2017)b]
  `https://github.com/HongtengXu/Hawkes-Process-Toolkit`
- **PoPPy** [Xu (2018)]
  `https://github.com/HongtengXu/PoPPy`

# Tick

A machine learning library for Python 3.

- The core functions are implemented by C language.
- Linear models, point processes, survival analysis.
- Integrate some classic Hawkes process models.
- Implement many optimization solvers
- Support multi-CPU computation

# THAP

THAP: A MATLAB **T**oolboxes for **HA**wkes **P**rocesses and its variants.

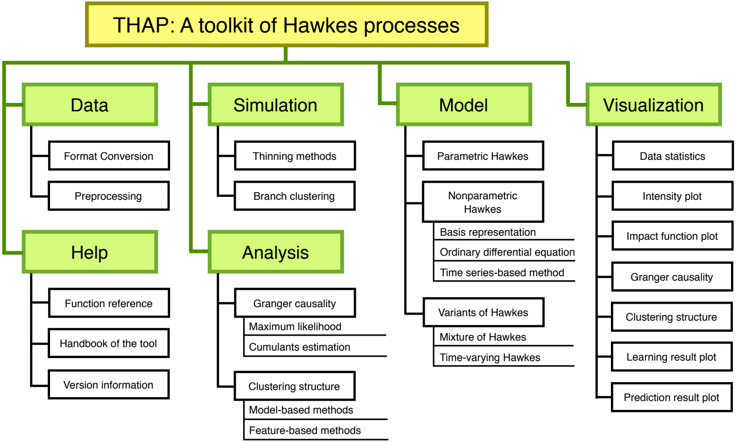



Figure 14: The architecture of THAP.

# THAP: Functions and Applications

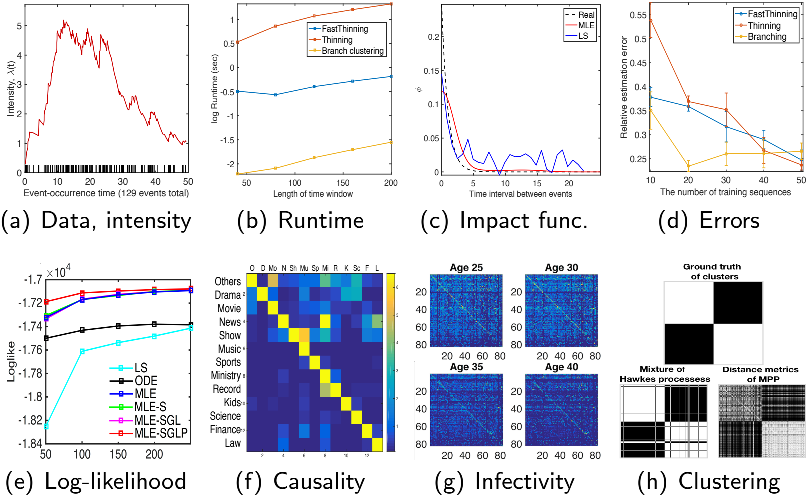



Figure 15: Visualization of typical functions achieved by *THAP*

# PoPPy

PoPPy: A **Po**int **P**rocess **Py**Torch Toolbox

- It is an extension of THAP.
- **Rich Functionality:** data operations, learning, prediction, simulation, visualization, ...
- **High Flexibility:** modular design of model, multiple loss functions, regularizers, support numerical and categorical features, ...
- **High Scalability:** support GPU computations

# PoPPy: Flexible model design

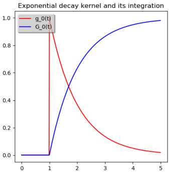
Intensity function:

$$\begin{aligned}\lambda_d(t) &= g_\lambda\left(\mu(d, \boldsymbol{f}_d, \boldsymbol{f}_s) + \sum_{t_i<t}\phi(t, t_i, d, d_i, \boldsymbol{f}_d, \boldsymbol{f}_{d_i})\right)\\ &= g_\lambda\left(\mu(d, \boldsymbol{f}_d, \boldsymbol{f}_s) + \sum_{t_i<t}\sum_{m=1}^{M} a_m(d, d_i, \boldsymbol{f}_d, \boldsymbol{f}_{d_i})\kappa_m(t-t_i)\right).\end{aligned} \tag{22}$$

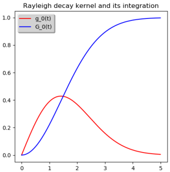
Exogenous Intensity and Endogenous Impact:

$$\mu(d, \boldsymbol{f}_d, \boldsymbol{f}_s) = \begin{cases} g_\mu(\mu_d), \\ g_\mu(\boldsymbol{w}_d^\top \boldsymbol{f}_s), \\ g_\mu(\boldsymbol{f}_d^\top \boldsymbol{W}\boldsymbol{f}_s), \\ NN(d, \boldsymbol{f}_d, \boldsymbol{f}_s). \end{cases} \qquad a_m(d, d_i, \boldsymbol{f}_d, \boldsymbol{f}_{d_i}) = \begin{cases} g_a(a_{dd_i m}), \\ g_a(\boldsymbol{u}_{d,m}^\top \boldsymbol{v}_{d_i,m}), \\ g_a(\boldsymbol{w}_{d,m}^\top \boldsymbol{f}_{d_i}), \\ g_a(\boldsymbol{f}_d^\top \boldsymbol{W}_m \boldsymbol{f}_{d_i}), \\ NN(d, d_i, \boldsymbol{f}_d, \boldsymbol{f}_{d_i}). \end{cases}$$

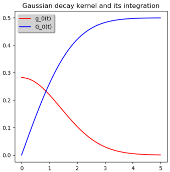
# PoPPy: Flexible model design



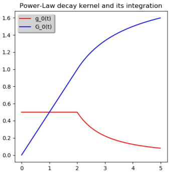

(a) Exponential



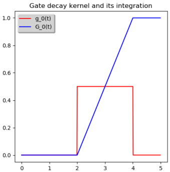

(b) Rayleigh kernel



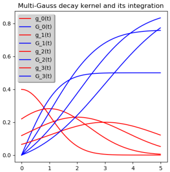

(c) Gaussian kernel




(d) Powerlaw kernel




(e) Gate kernel




(f) Multi-Gaussian

Figure 16: Examples of decay kernels and their integration values.

# PoPPy: Flexible data operations

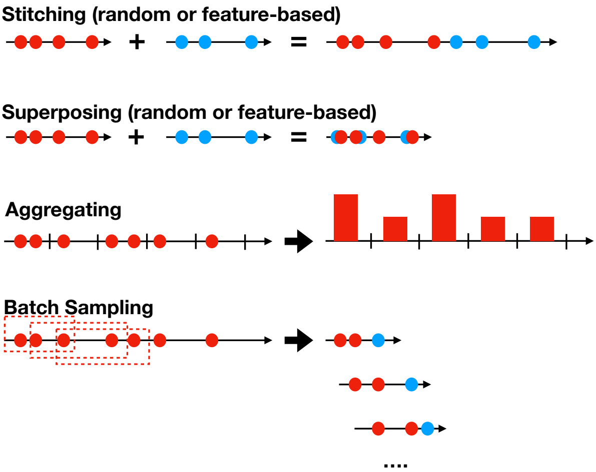



Figure 17: Typical data operations.

# Summary

- Temporal point processes have been widely used to describe the dynamic mechanisms hidden in real-world event sequences.
- The key of TPPs is modeling their intensity functions.
- The learning and the simulation of TPPs are flexible and theoretically-supportive.
- Hawkes processes are powerful to model the self- and mutually-triggering patterns among different event types, which have many useful properties for practical applications.

# References


Emmanuel Bacry, Martin Bompaire, Stéphane Gaïffas, and Soren Poulsen.
tick: a python library for statistical learning, with an emphasis on time-dependent modeling. *arXiv:1707.03003*, 2017.

Bacry, Emmanuel, and Jean-Francois Muzy.
First-and second-order statistics characterization of Hawkes processes and non-parametric estimation. *IEEE TIT*, 2016.

Michael Eichler, Rainer Dahlhaus, and Johannes Dueck.
Graphical modeling for multivariate hawkes processes with nonparametric link functions. *Time Series Analysis*, 2017.

Alan Hawkes.
Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society. Series B*, 1971.

Liniger, Thomas Josef.
Multivariate Hawkes processes, 2009.

Ogata, Yosihiko.
Statistical models for earthquake occurrences and residual analysis for point processes. In *JASA*, 1988.

Yosihiko Ogata.
On lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 1981.

Hongteng Xu and Hongyuan Zha.
THAP: a Matlab toolkit for learning with Hawkes processes. *arXiv:1708.09252*, 2017.

Hongteng Xu.
PoPPy: A Point Process Toolbox Based on PyTorch. *arXiv:1810.10122*, 2018.

Hongteng Xu, Dixin Luo, and Hongyuan Zha.
Learning hawkes processes from short doubly-censored event sequences. *ICML*, 2017.

Hongteng Xu, Farajtabar, Mehrdad, and Hongyuan Zha.
Learning Granger causality for Hawkes processes. *ICML*, 2016.

Farajtabar, Mehrdad, et al.
Shaping social activity by incentivizing users. *NIPS*, 2014.

Farajtabar, Mehrdad, et al.
Back to the past: Source identification in diffusion networks from partially observed cascades. *AISTATS*, 2015.

Ke Zhou, Hongyuan Zha, and Le Song.
Learning Social Infectivity in Sparse Low-rank Networks Using Multi-dimensional Hawkes Processes. In *AISTATS*, 2013.

Zhao, Qingyuan, et al.
Seismic: A self-exciting point process model for predicting tweet popularity. *KDD*, 2015.

Xu, Hongteng and Wu, Weichang and Nemati, Shamim and Zha, Hongyuan.
Patient flow prediction via discriminative learning of mutually-correcting processes *TKDE*, 2016.

Hongteng Xu, Dixin Luo, Xu Chen, and Lawrence Carin.
Benefits from superposed Hawkes processes *AISTATS*, 2018.

Wang, Yichen, et al.
Isotonic hawkes processes. *ICML*, 2016.

Møller, Jesper and Rasmussen, Jakob G
Approximate simulation of Hawkes processes. *Methodology and Computing in Applied Probability*, 2006.

Li, Shuang, et al.
Learning temporal point processes via reinforcement learning. *NIPS*, 2018.

Zammit-Mangion, Andrew, et al.
Point process modelling of the Afghan War Diary. *PNAS*, 2012.