AI Expert 프로그램 실습

9/11 Graphical Models, Gaussian Process, Hawkes Process

Part 3. Hawkes Process

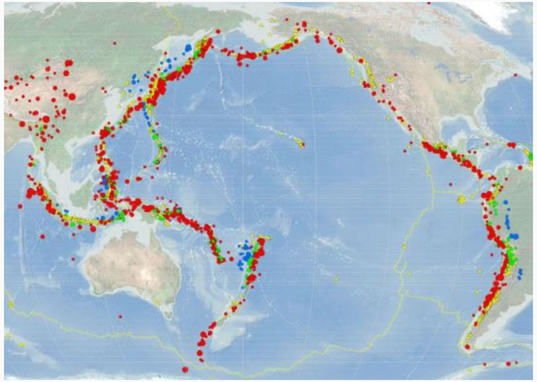
Overview

- Point Process
- Poisson Process
- Hawkes Process
- Predicting Retweet Dynamics

* This part refers to the ICML 2018 tutorial *Learning with Temporal Point Processes* (Rodriguez and Valera, 2018)

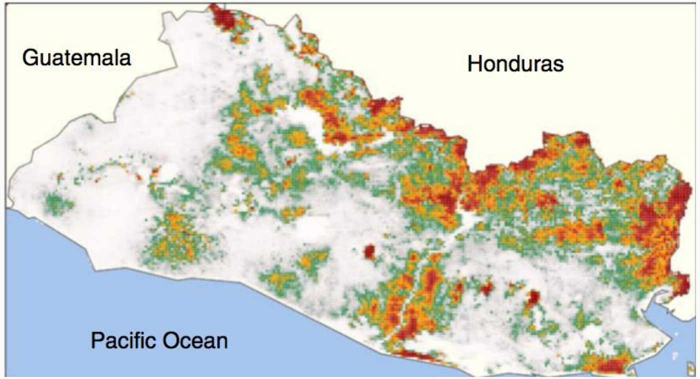
* Also, predicting Retweet dynamics refers to *TideH* (Kobayashi and Lambiotte, 2016).

Many discrete events in continuous time !



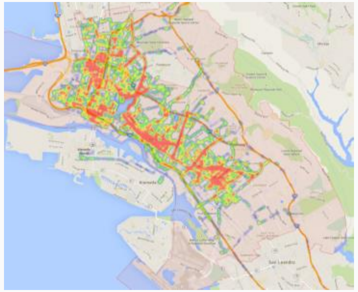
Earthquake

Online actions



Disease dynamics

Financial trading



Mobility dynamics

Variety of processes behind these events

Stock trading

Flu spreading

Article creation in Wikipedia

News spread in Twitter

Reviews and sales in Amazon

Ride-sharing requests

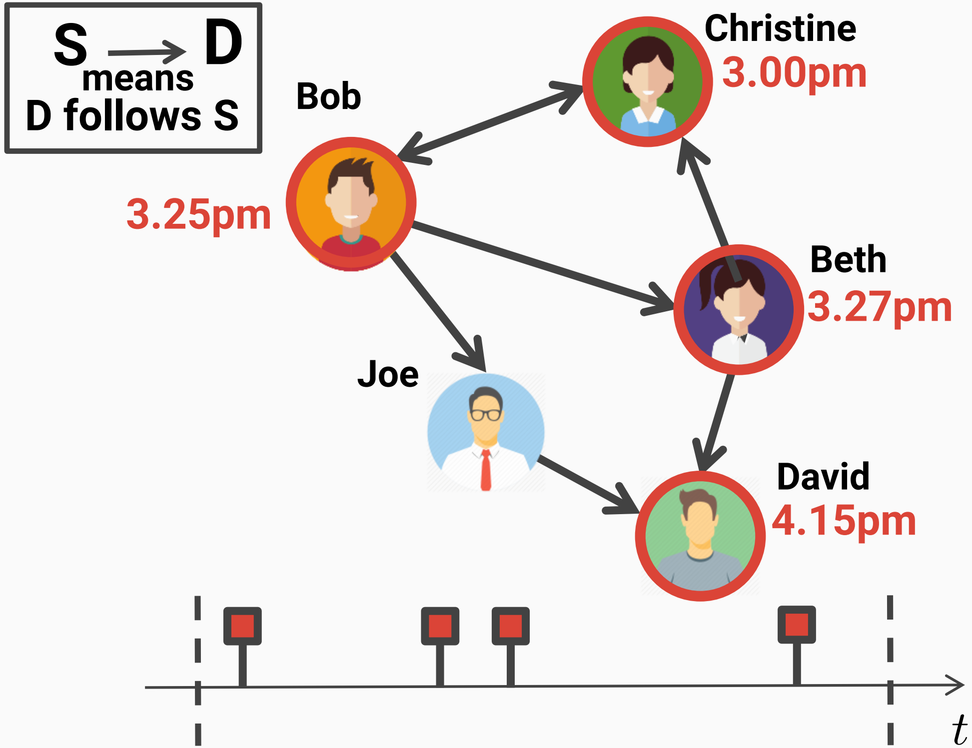
A user's reputation in Quora

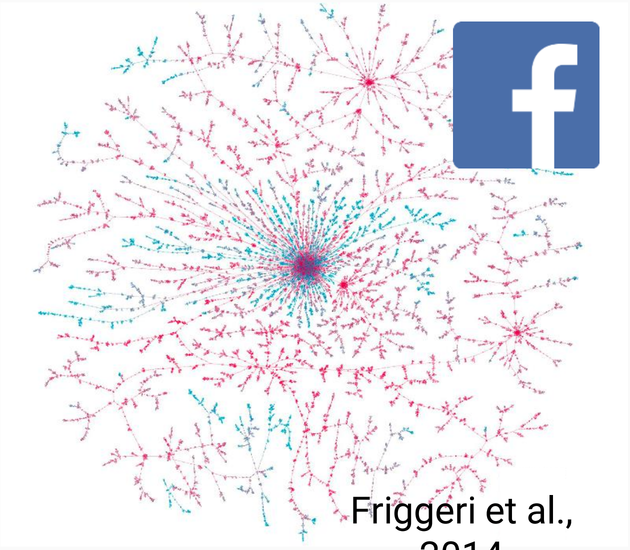
FAST → SLOW

…in a wide range of temporal scales.

Part 3. Hawkes Process

Real World Examples





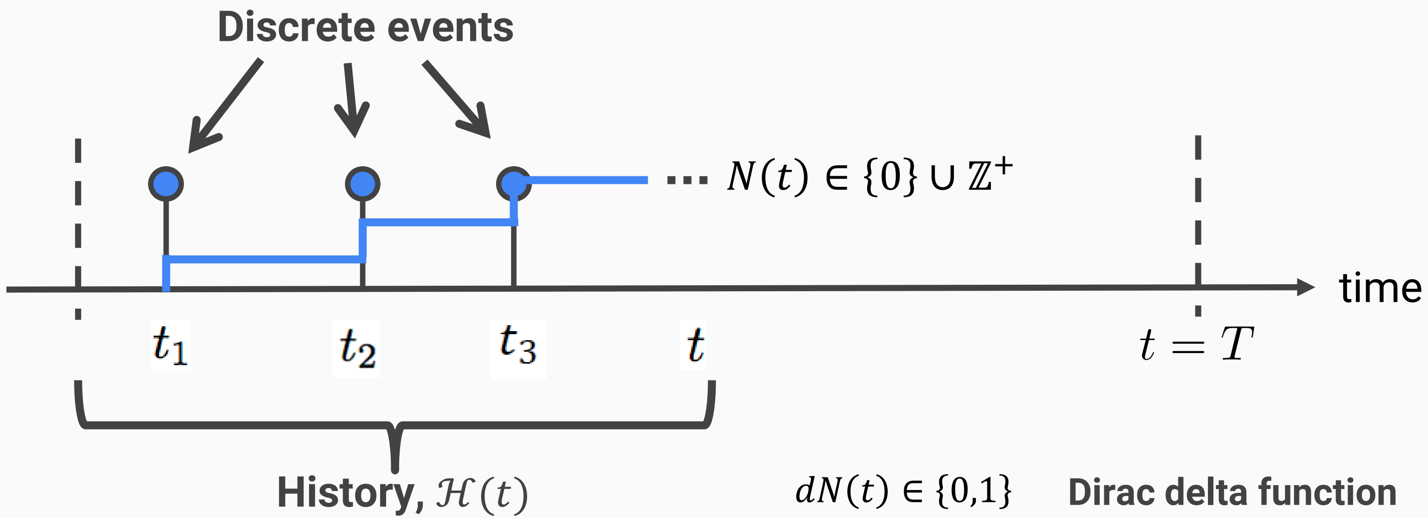
Friggeri et al., 2014

theguardian

Click and elect: how fake news helped Donald Trump win a real election

Part 3. Hawkes Process

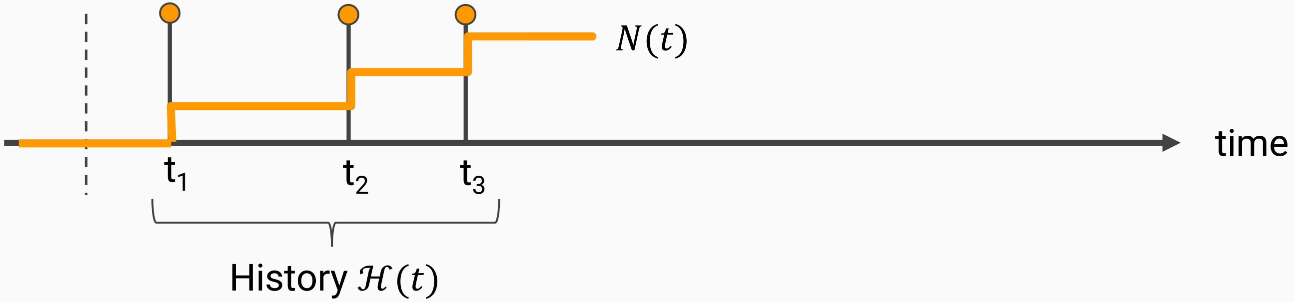
Temporal point process is a random process whose realization consists of discrete events localized in time



$dN(t) \in \{0,1\}$ **Dirac delta function**

Formally: $N(t) = \int_0^t dN(s) \Rightarrow dN(t) = \sum_{t_i \in \mathcal{H}(t)} \delta(t - t_i)dt$

Temporal Point process can be represented as Intensity function

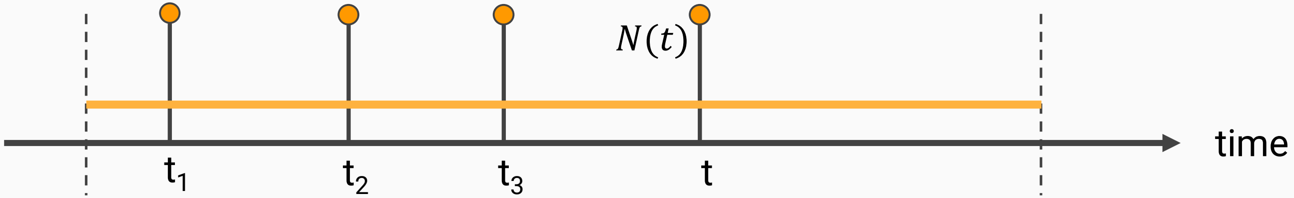


Since it is cumbersome to model event counts over time,
we model event **intensity** over time.

$$\underline{\lambda^*(t)dt} = \mathbb{E}[dN(t)|\mathcal{H}(t)]$$

It is a rate = # of events / unit of time

Poisson process



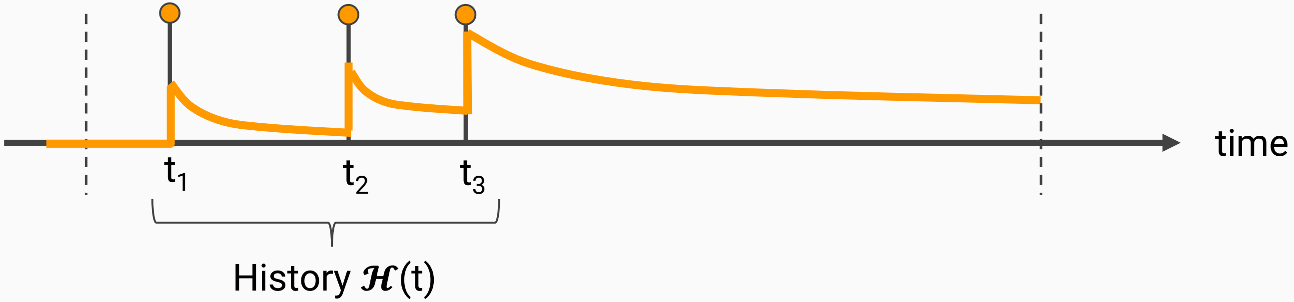
Intensity of a Poisson process

$$\lambda^*(t) = \mu$$

Observations:

1. Intensity independent of history
2. Uniformly random occurrence
3. Time interval follows exponential distribution

Hawkes process: Self-exciting intensity function



Self-exciting: each arrival increase the rate of future arrivals for some period

Intensity of Hawkes process:

$$\lambda^*(t) = \overset{\text{Baseline}}{\overline{\mu}} + \alpha \sum_{t_i<t} \overset{\text{Triggering kernel}}{\overline{\phi(t-t_i)}}$$

Observations:

1. Clustered (or bursty) occurrence of events
2. Intensity is stochastic and history dependent

Tick: Python Library for statistical learning about point processes

How to Install

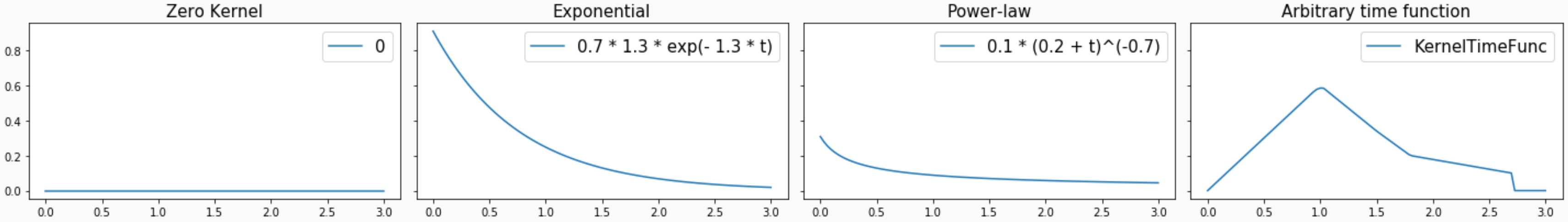
```
cogito@digits-1:~$ pip install tick
cogito@digits-1:~$ pip install --upgrade scipy==1.0.0 # optional
cogito@digits-1:~$ pip install dill # optional
```

Tick: Python Library for statistical learning about point processes

How to Use - Simulation

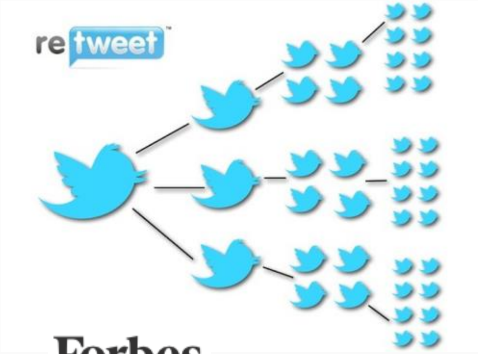
```
from tick.hawkes import HawkesKernel0, HawkesKernelExp, HawkesKernelPowerLaw, HawkesKernelTimeFunc
...
kernel_exp = HawkesKernelExp(.7, 1.3) # HawkesKernelExp(intensity, decay)
t_values = np.linspace(0, 3, 100)
kernel_exp.get_values(t_values)) # Simulation using get_values([times])
```

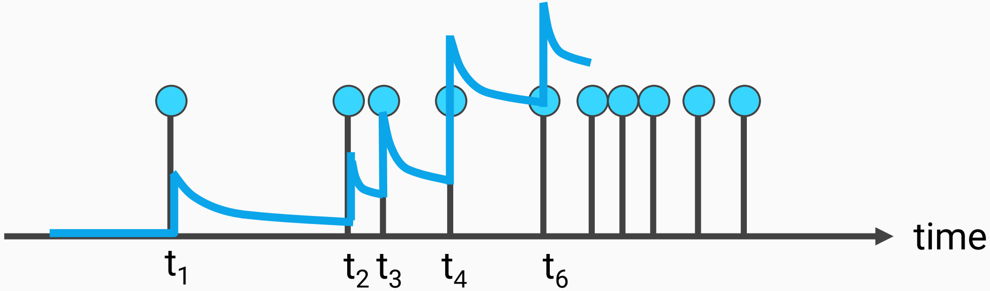
Let's run the sample code !



Part 3. Predicting Retweet Dynamics using Hawkes Process

Predicting Retweet Dynamics using Hawkes Process





Forbes

For Brands And PR: When Is The Best Time To Post On Social Media?

THE HUFFINGTON POST

THE BLOG

The Best Times to Post on Social Media

Kobayashi, R., & Lambiotte, R. (2016, March). Tideh: Time-dependent hawkes process for predicting retweet dynamics. In *Tenth International AAAI Conference on Web and Social Media*.

Part 3. Predicting Retweet Dynamics using Hawkes Process

Get ready for the codes!

https://github.com/NII-Kobayashi/TiDeH

```
cogito@digits-1:~$ git clone https://github.com/NII-Kobayashi/TiDeH
Cloning into 'TiDeH'...
remote: Enumerating objects: 132, done.
remote: Total 132 (delta 0), reused 0 (delta 0), pack-reused 132
Receiving objects: 100% (132/132), 1.61 MiB | 310.00 KiB/s, done.
Resolving deltas: 100% (9/9), done.
Checking connectivity... done.
cogito@digits-1:~$ cd TiDeH/
```

Kobayashi, R., & Lambiotte, R. (2016, March). Tideh: Time-dependent hawkes process for predicting retweet dynamics. In *Tenth International AAAI Conference on Web and Social Media*.

Part 3. Predicting Retweet Dynamics using Hawkes Process

What you will see is …

```
TiDeH/
| data/
| | example/
| | | sample_file.txt*
| | training/
| tideh/
| | __init__.py*
| | estimate.py*
| | fit.py*
| | functions.py*
| | main.py*
| | prediction.py*
| | simulate.py*
| | training.py*
| example_native.py*
| example_optimized.py*
| example_simulation.py*
| example_training.py*
```

```
2150 160.627488
0.000000 503173
0.000021 133
0.000324 40
0.000910 73
0.003047 83
0.003141 30
0.003451 15
0.003552 305
0.003970 208
0.006932 82
0.006984 287
0.007528 124
0.007718 100
0.008874 17
0.008999 77
0.009056 101
0.009210 28
```

* one file per tweet
* space separated
* only tweets with more than 2000 retweets were used

First row

- <number of total retweets> <start time of tweet in days>

Part 3. Predicting Retweet Dynamics using Hawkes Process

What you will see is …

```
TiDeH/
| data/
| | example/
| | | sample_file.txt*
| | training/
| tideh/
| | __init__.py*
| | estimate.py*
| | fit.py*
| | functions.py*
| | main.py*
| | prediction.py*
| | simulate.py*
| | training.py*
| example_native.py*
| example_optimized.py*
| example_simulation.py*
| example_training.py*
```

```
2150 160.627488
0.000000 503173
0.000021 133
0.000324 40
0.000910 73
0.003047 83
0.003141 30
0.003451 15
0.003552 305
0.003970 208
0.006932 82
0.006984 287
0.007528 124
0.007718 100
0.008874 17
0.008999 77
0.009056 101
0.009210 28
```

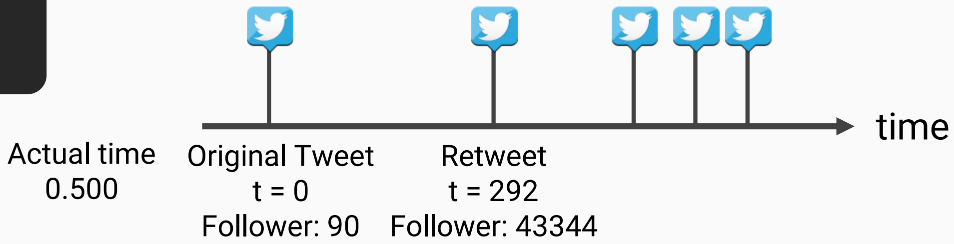
* one file per tweet
* space separated
* only tweets with more than 2000 retweets were used

First row

- <number of total retweets> <start time of tweet in days>

Every other row

- <relative tweeted/retweeted time in seconds> <number of followers of retweeting person>



Part 3. Predicting Retweet Dynamics using Hawkes Process

How to run the example code

```
cogito@digits-1:~/TiDeH$ python3 example_native.py
```

Example of predicting future retweet activity. Given an observation time, the parameters of the infectious rate are estimated and then, the number of retweets before a given time is predicted.

Then, you will see the results

```
Estimated parameters are:
p0: 0.001
r0: 0.414
phi0: 0.140
tm: 1.822
Average % error (estimated to fitted): 12.33
Predicted number of retweets from 48 to 168 hours: 37
Predicted number of retweets at hour 168: 2170
Prediction error (absolute): 20
```

How to run the example code

Let's look at the code *example_native.py*

```
# Module import
from tideh import estimate_parameters
from tideh import load_events
from tideh import predict
# Read dataset
filename = 'data/example/sample_file.txt'
(_, start_time), events = load_events(filename)
```

How to run the example code

Let's look at the code *example_native.py*

```
# additional parameters passed to infectious rate function
add_params = {'t0': start_time, 'bounds': [(-1, 0.5), (1, 20.)]}
obs_time = 48 # observation time of 2 days
pred_time = 168 # predict for one week
params, err, _ = estimate_parameters(events=events, obs_time=obs_time, **add_params)
```

The probability for getting a retweet in a small-time interval $[t, t + \Delta t]$ is $\lambda(t)\Delta t$.

$$\lambda(t) = \underbrace{p(t)}_{\text{Infectious rate}} \sum_{i:t_i<t} \underbrace{d_i}_{\text{\# Followers}} \overbrace{\phi(t - t_i)}^{\text{Exciting kernel}}$$

Part 3. Predicting Retweet Dynamics using Hawkes Process

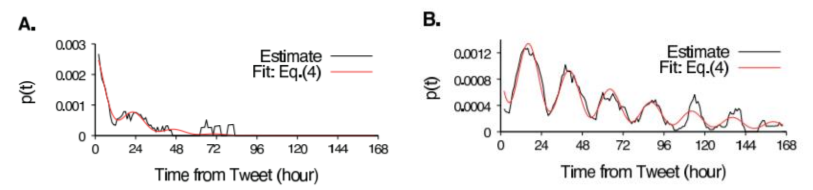
How to run the example code

Let's look at the code *example_native.py*

```
# additional parameters passed to infectious rate function, bounds for r0 and taum
add_params = {'t0': start_time, 'bounds': [(-1, 0.5), (1, 20.)]}
obs_time = 48 # observation time of 2 days
pred_time = 168 # predict for one week
params, err, _ = estimate_parameters(events=events, obs_time=obs_time, **add_params)
```

Infectious rate is

$$p(t) = p_0 \left\{ 1 - r_0 \sin\left(\frac{2\pi}{T_m}(t + \phi_0) \right) \right\} e^{-(t-t_0)/\tau_m}$$



- t_0: time of the original tweet
- $T_m = 1$ day: period of oscillation
- p_0: intensity
- r_0 : the relative amplitude of the oscillation
- ϕ_0: its phase
- τ_m: characteristic time of popularity decay

Part 3. Predicting Retweet Dynamics using Hawkes Process

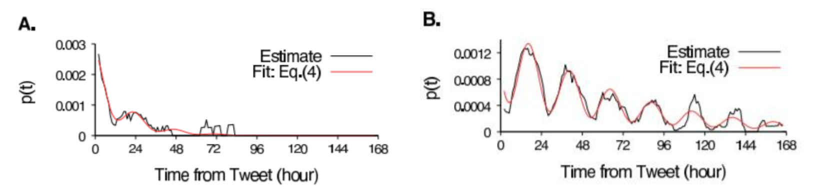
How to run the example code

Let's look at the code *example_native.py*

```
# additional parameters passed to infectious rate function, bounds for r0 and taum
add_params = {'t0': start_time, 'bounds': [(-1, 0.5), (1, 20.)]}
obs_time = 48 # observation time of 2 days
pred_time = 168 # predict for one week
params, err, _ = estimate_parameters(events=events, obs_time=obs_time, **add_params)
```

Infectious rate is

$$p(t) = p_0 \left\{ 1 - r_0 \sin\left(\frac{2\pi}{T_m} (t + \phi_0) \right) \right\} e^{-(t-t_0)/\tau_m}$$



- t_0: time of the original tweet
- $T_m = 1$ day: period of oscillation
- p_0: intensity
- r_0 : the relative amplitude of the oscillation
- ϕ_0: its phase
- τ_m: characteristic time of popularity decay

Part 3. Predicting Retweet Dynamics using Hawkes Process

How to run the example code

Let’s look at the code *example_native.py*

```
# tideh/main.py - Estimate of model parameters of TiDeH (p_0, r_0, phi_0, t_m).
def estimate_parameters(events, obs_time=None, window_size=4, window_stride=1,
                        kernel_int=functions.integral_zhao,
                        p=functions.infectious_rate_tweets, values=None, **p_params):
    obs_time = int(min(obs_time, events[len(events) - 1][0]))
    events = [e for e in events if e[0] < obs_time] # e[0] holds time of the event
    estimations, window_event_count, window_middle = \
        estimate.estimate_infectious_rate(events, kernel_int, obs_time, window_size, window_stride)
    fitted = fit.fit_parameter(estimates=estimations, fun=lambda x, *args: p(x, *args, **p_params),
    start_values=values, xval=window_middle)
```

```
# tideh/estimate.py - Esitmate base rate p0
def estimate_infectious_rate(events, kernel_integral=functions.integral_zhao, obs_time=24, window_size=4,
                             window_stride=1):
for start in range(0, obs_time - window_size + window_stride, window_stride):
    ...
    est = estimate_infectious_rate_constant(events=events_tmp, t_start=start, t_end=end,
                                            kernel_integral=kernel_integral, count_events=count_current)
    ...
```

Part 3. Predicting Retweet Dynamics using Hawkes Process

How to run the example code

Let’s look at the code *example_native.py*

```python
# tideh/main.py
def estimate_parameters(events, obs_time=None, window_size=4, window_stride=1,
                        kernel_int=functions.integral_zhao,
                        p=functions.infectious_rate_tweets, values=None, **p_params):
    obs_time = int(min(obs_time, events[len(events) - 1][0]))
    events = [e for e in events if e[0] < obs_time] # e[0] holds time of the event
    estimations, window_event_count, window_middle = \
        estimate.estimate_infectious_rate(events, kernel_int, obs_time, window_size, window_stride)
    fitted = fit.fit_parameter(estimates=estimations, fun=lambda x, *args: p(x, *args, **p_params),
    start_values=values, xval=window_middle)
```

```python
# tideh/fit.py - Estimate of model parameters r_0, phi_0, t_m
from scipy.optimize import leastsq
def fit_parameter(estimates, fun, start_values, xval):
    if start_values is None:
        start_values = np.array([0, 0, 0, 1.])
    return leastsq(func=loss_function, x0=start_values, args=(estimates, fun, xval))[0]
```

Part 3. Predicting Retweet Dynamics using Hawkes Process

How to run the example code

Then, where is the exciting kernel?

```
# additional parameters passed to infectious rate function
add_params = {'t0': start_time, 'bounds': [(-1, 0.5), (1, 20.)]}
obs_time = 48 # observation time of 2 days
pred_time = 168 # predict for one week
params, err, _ = estimate_parameters(events=events, obs_time=obs_time, **add_params)
```

The probability for getting a retweet in a small-time interval $[t, t + \Delta t]$ is $\lambda(t)\Delta t$.

$$\lambda(t) = p(t) \sum_{i:t_i<t} d_i \overline{\phi(t - t_i)} \longleftarrow \textbf{Exciting kernel}$$

Exciting kernel is fitted to the empirical data by the function where is the heavily tailed in a variety of social networks (Crane, et al. 2008, Masuda, et al. 2013)

$$\phi(s) = \begin{cases} 0 & (s < 0) \\ c_0 & (0 \leq s \leq s_0) \\ c_0 (s/s_0)^{-(1+\theta)} & (\text{Otherwise}) \end{cases},$$

- $c_0 = 6.49 \times 10^{-4}$ seconds
- $s_0 = 300$ seconds
- $\theta = 0.242$

Part 3. Predicting Retweet Dynamics using Hawkes Process

We are going to estimate the exciting kernel from the dataset.

Then, where is the exciting kernel?

$$\phi(s) = \begin{cases} 0 & (s < 0) \\ c_0 & (0 \leq s \leq s_0) \\ c_0 (s/s_0)^{-(1+\theta)} & (\text{Otherwise}) \end{cases},$$

- $c_0 = 6.49 \times 10^{-4}$ seconds
- $s_0 = 300$ seconds
- $\theta = 0.242$

Now, let's estimate the kernel !