Rise and Fall Patterns of Information Diffusion: Model and Implications

Yasuko Matsubara (Kyoto University),
Yasushi Sakurai (NTT), B. Aditya Prakash (CMU),
Lei Li (UCB), Christos Faloutsos (CMU)

Motivation

Social media

facilitate faster diffusion of news and rumors

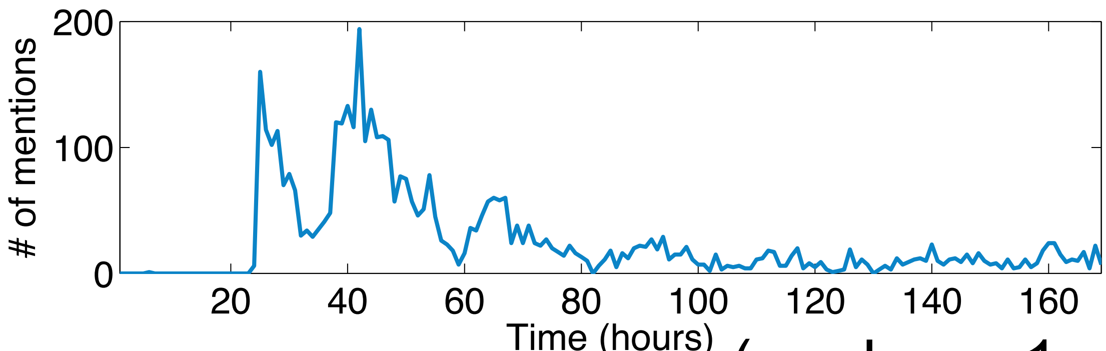
Q: How do news and rumors spread in social media?

News spread in social media

MemeTracker [Leskovec et al. KDD'09]

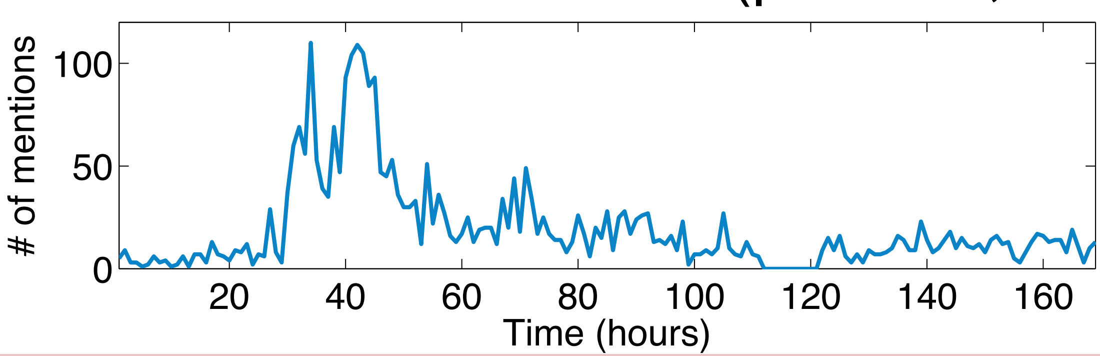
short phrases sourced from U.S. politics in 2008

"you can put lipstick on a pig" (# of mentions in blogs)



(per hour, 1 week)

"yes we can"

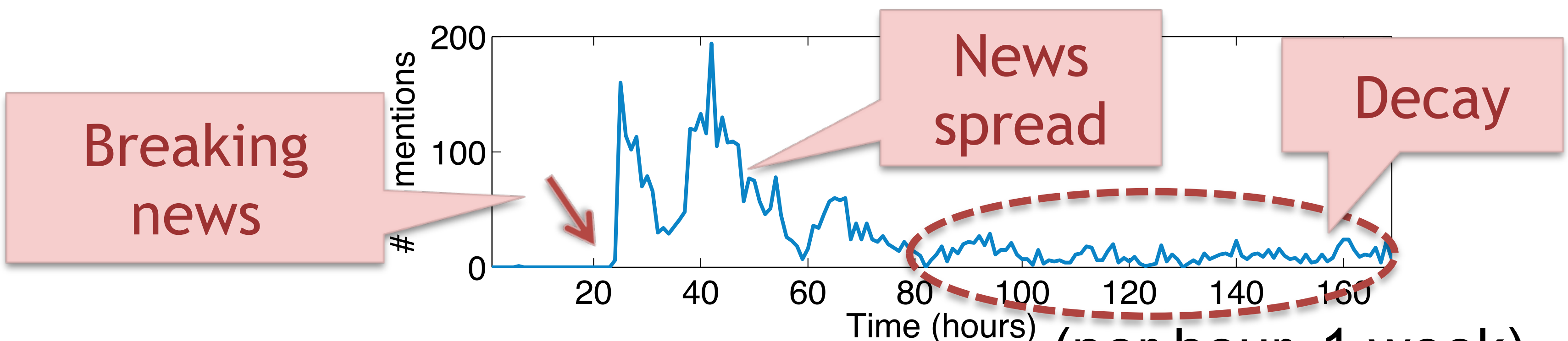


News spread in social media

MemeTracker [Leskovec et al. KDD'09]

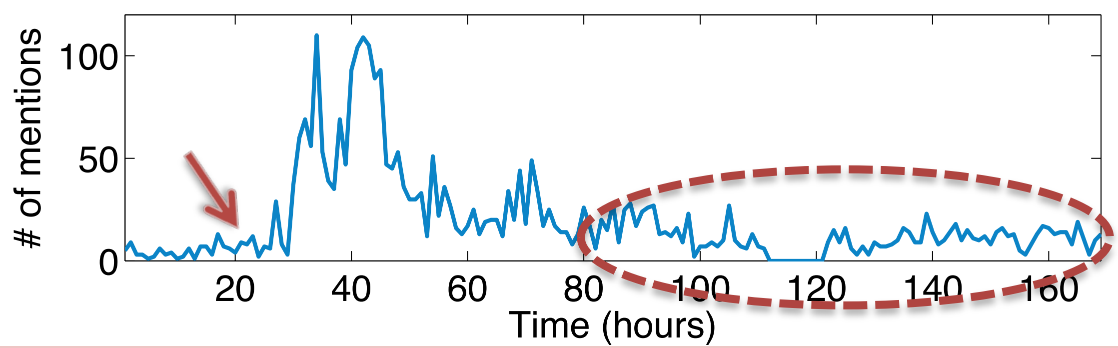
short phrases sourced from U.S. politics in 2008

"you can put lipstick on a pig" (# of mentions in blogs)



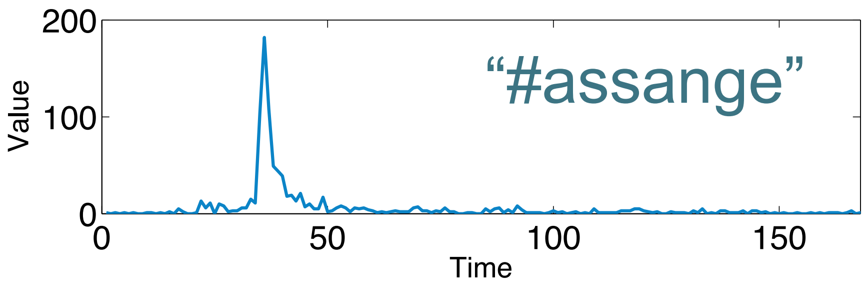
(per hour, 1 week)

"yes we can"

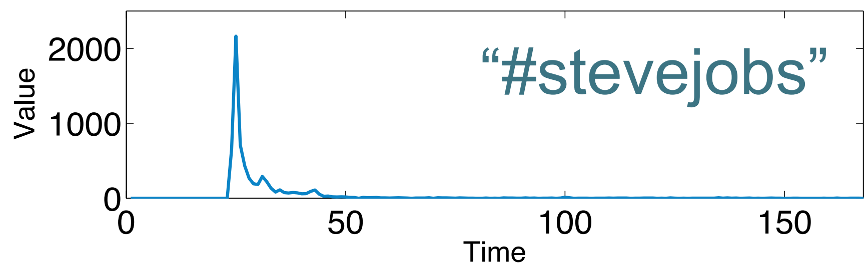


Rise and fall patterns in social media

Twitter (# of hashtags per hour)



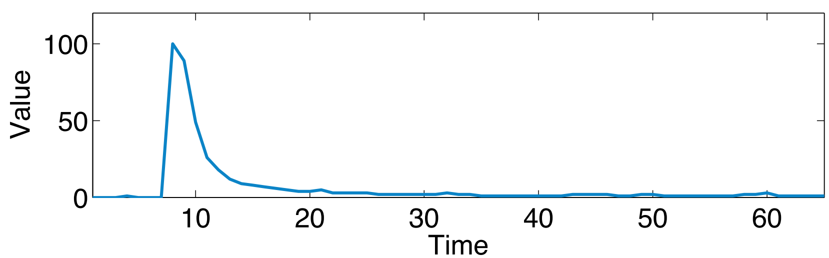
(per hour, 1week)



(per hour, 1 week)

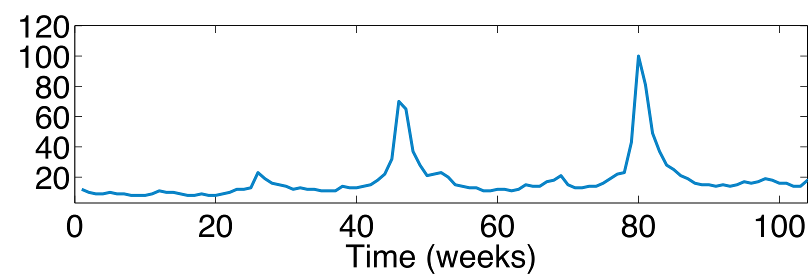
Google trend (# of queries per week)

“tsunami” (in 2005)



(per week, 1 year)

“harry potter” (2010 - 2011)



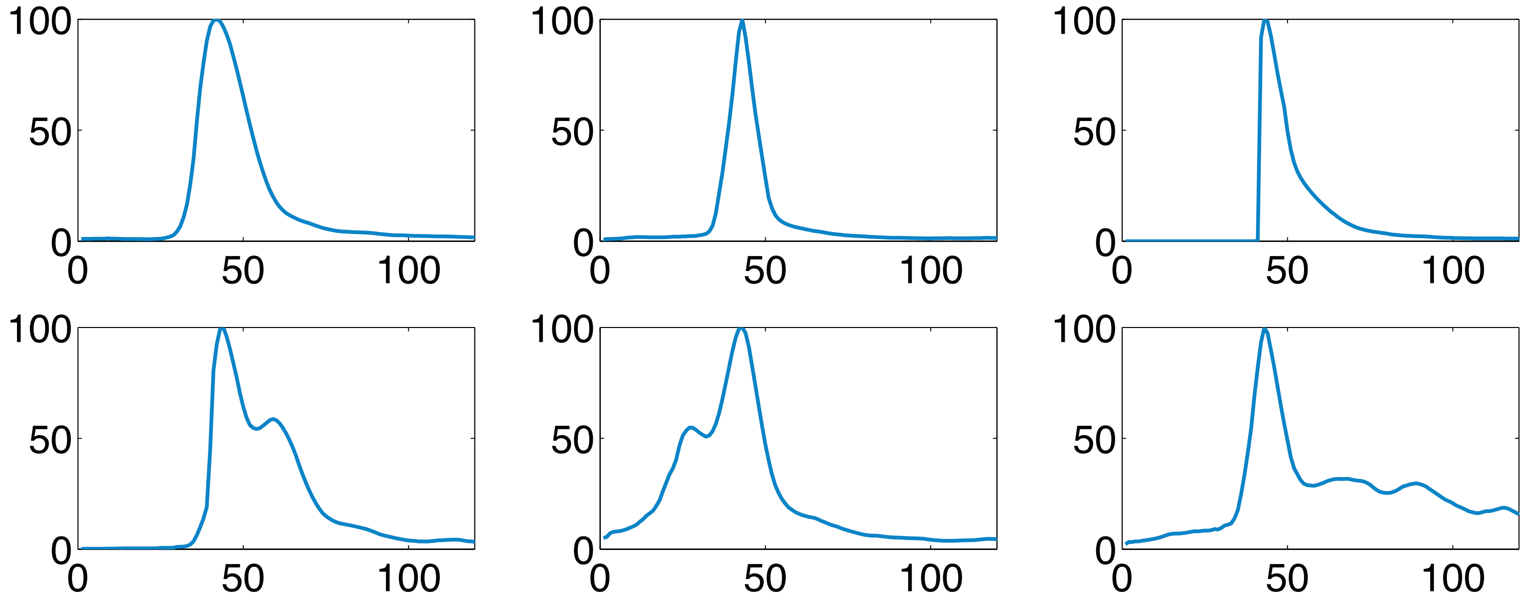
(per week, 2 years)

Rise and fall patterns in social media

How many patterns are there?

-Earlier work claims there're several classes

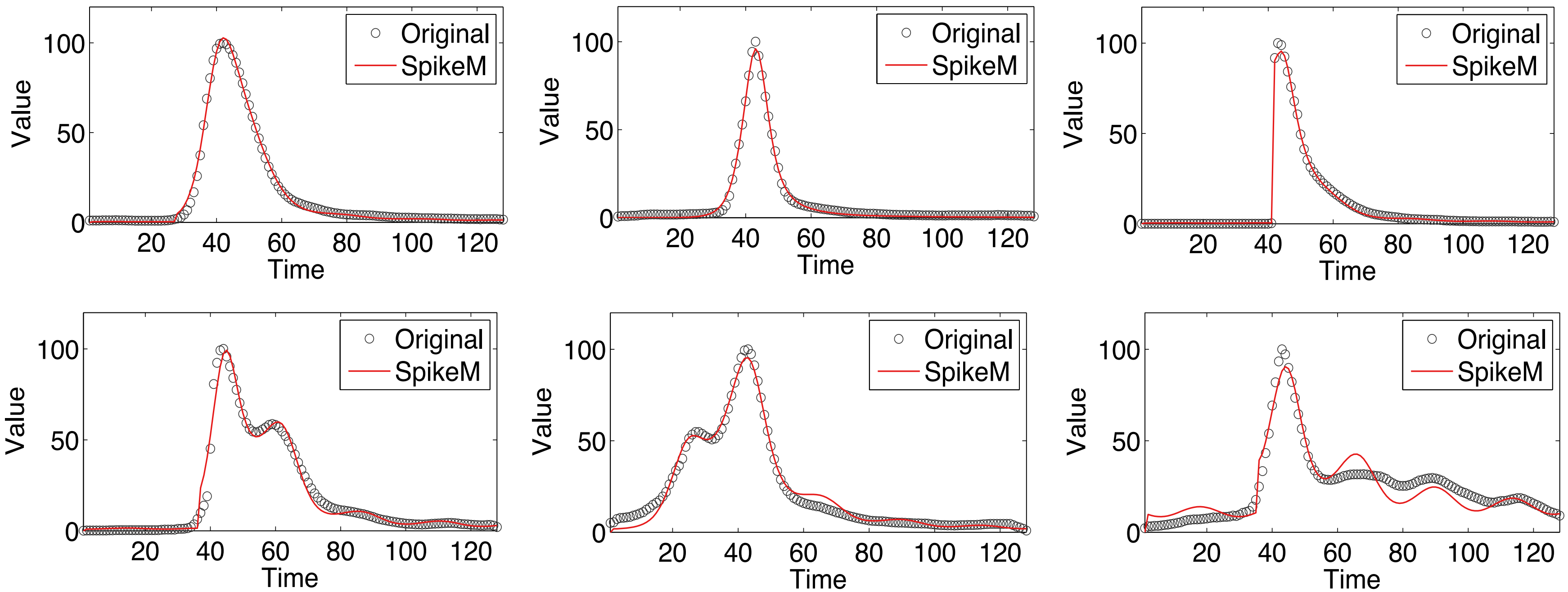
- **four** classes on YouTube [Crane et al. PNAS'08]
- **six** classes on Media [Yang et al. WSDM'11]



Rise and fall patterns in social media

Q. How many classes are there after all?

A. Our answer is “ONE”!



We can represent all patterns by single model

Outline

- Motivation
- Problem definition
- Proposed method
- Experiments
- Discussion – SpikeM at work
- Conclusions

Problem definition

Goal: predict/model social activity

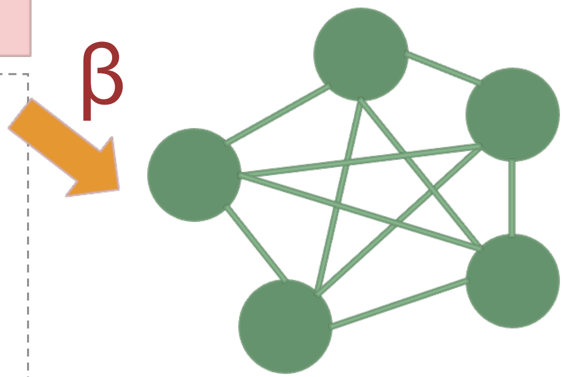
Problem 1 (What-if?)

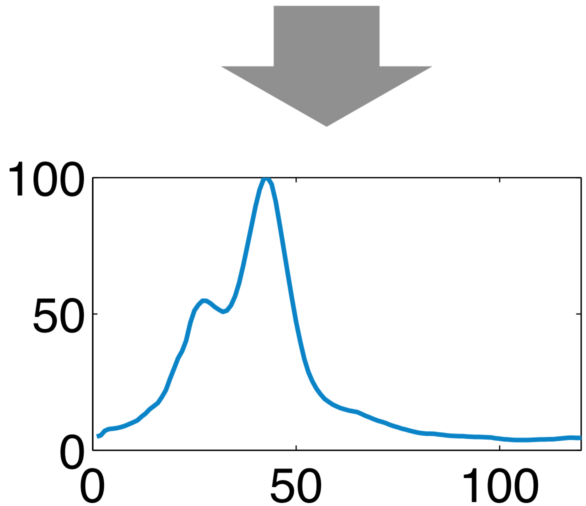
Given:

- Network of bloggers/users
- External shock/event
- Quality of the event β

Find:

- How blogging activity will evolve over time





Problem definition

Goal: predict/model social activity

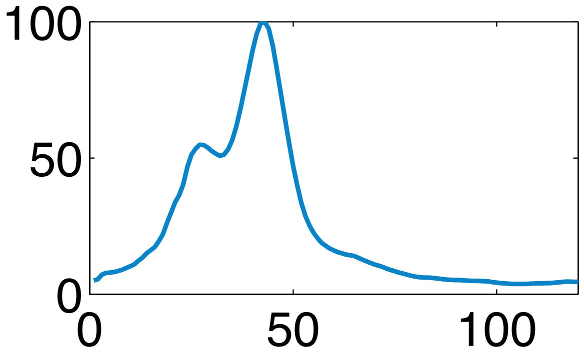
Problem 2 (Model design)

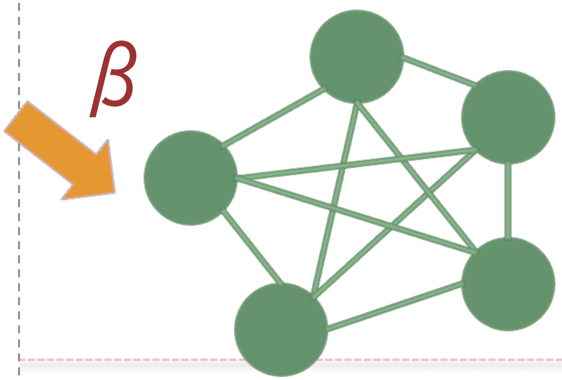
Given:

- Behavior of spikes

Find:

- Equation/model that can explain them, e.g.,
  - # of potential bloggers
  - Strength of external shock
  - Quality of the event β





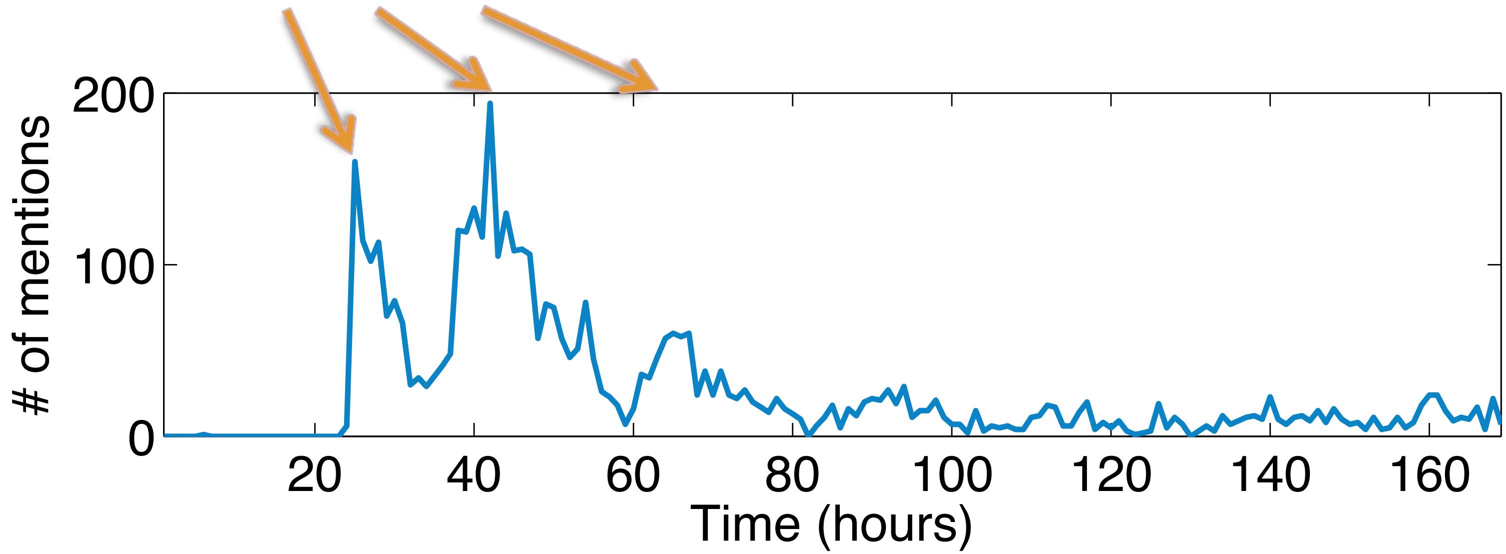
Outline

- Motivation
- Problem definition
- Proposed method
- Experiments
- Discussion – SpikeM at work
- Conclusions

Proposed method

SpikeM capture 3 properties of real spike

1. periodicities

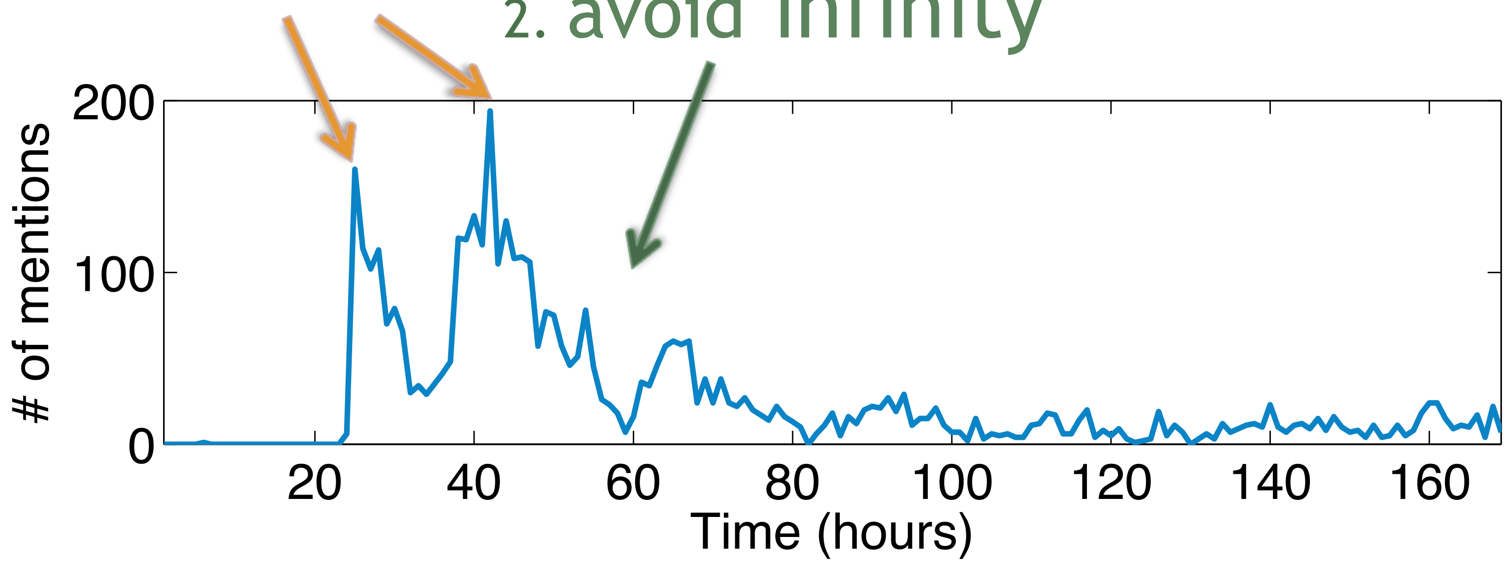


Proposed method

SpikeM capture 3 properties of real spike

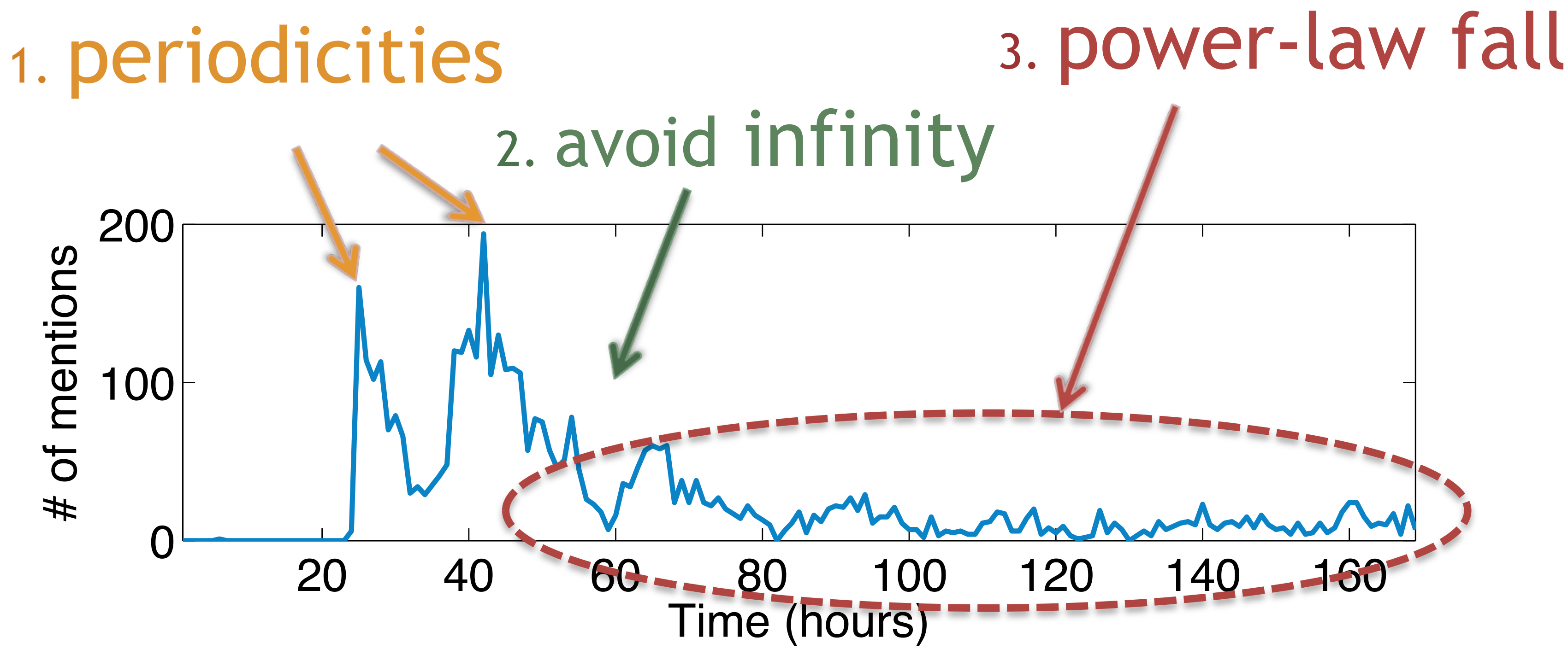
1. periodicities

2. avoid infinity



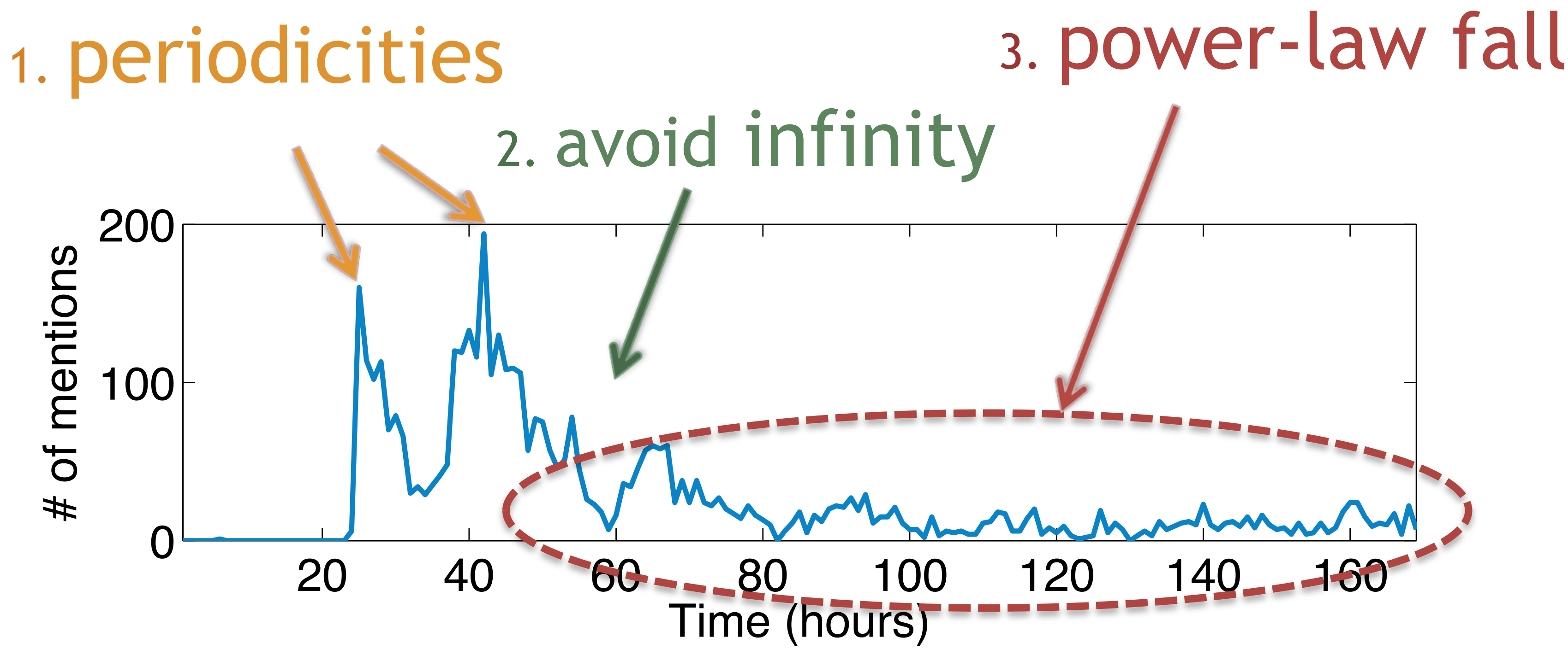
Proposed method

SpikeM capture 3 properties of real spike



Proposed method

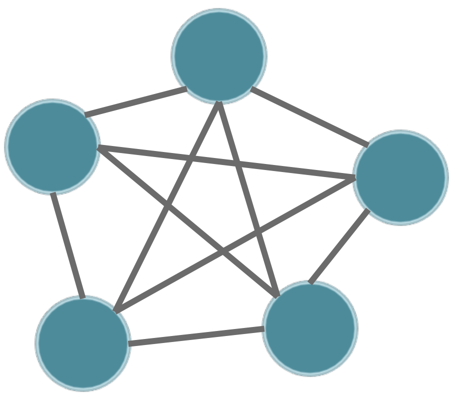
SpikeM capture 3 properties of real spike



SpikeM capture behavior of real spikes using few parameters

Main idea (details)

- 1. Un-informed bloggers (clique of N bloggers/nodes)



Time n=0

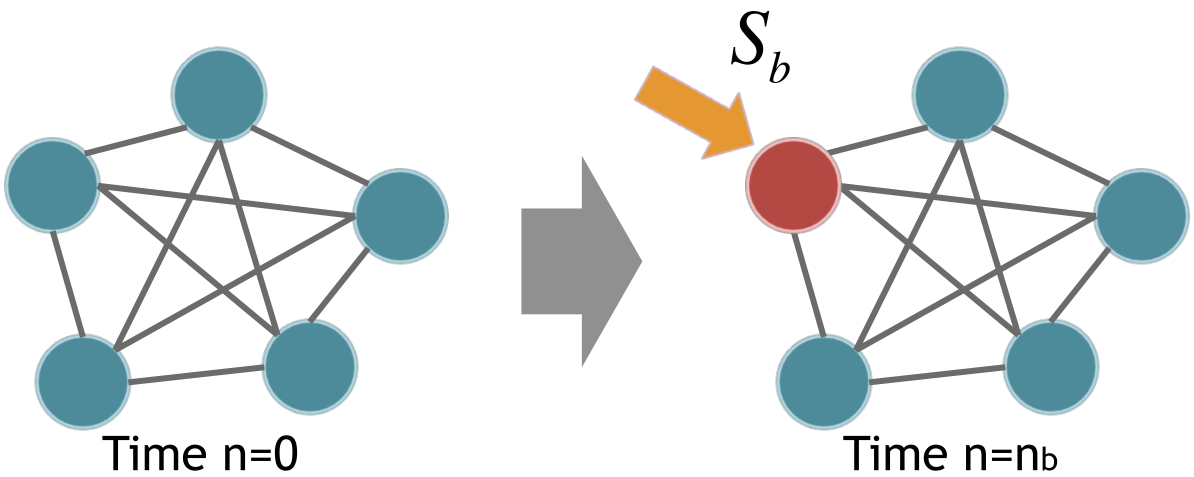
Nodes (bloggers) consist of two states

U – Un-informed of rumor

B – informed, and Blogged about rumor

Main idea (details)

- 1. Un-informed bloggers (clique of N bloggers/nodes)
- 2. External shock at time n_b (e.g, breaking news)



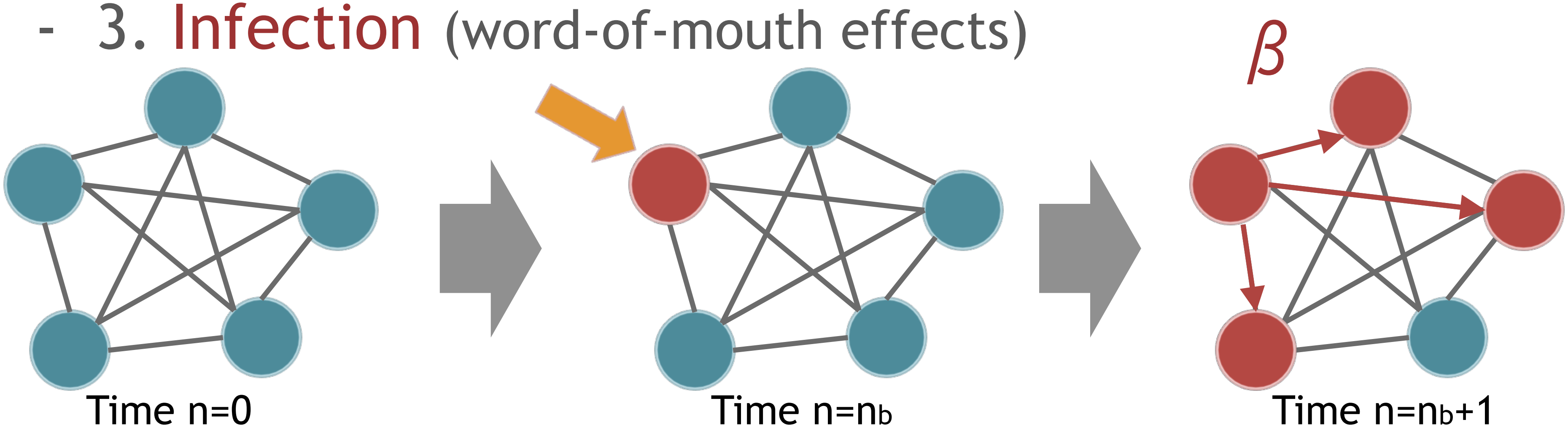
External shock

– Event happened at time n_b

– S_b bloggers are informed, blog about news

Main idea (details)

- 1. Un-informed bloggers (clique of N bloggers/nodes)
- 2. External shock at time n_b (e.g, breaking news)
- 3. Infection (word-of-mouth effects)



Infectiveness of a blog-post

β – Strength of infection (quality of news)

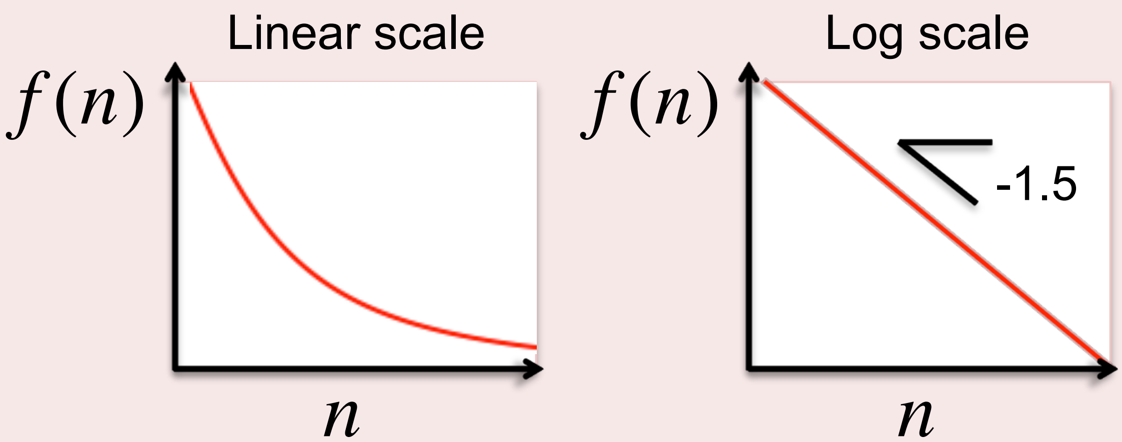
$f(n)$ – Decay function (how infective a blog posting is)

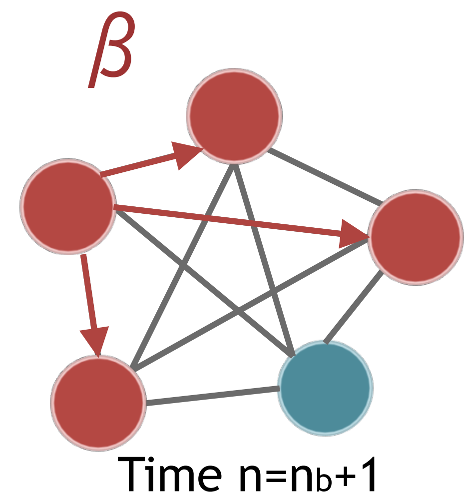
Main idea (details)

- 1. Un-informed bloggers (clique of N bloggers/nodes)

aking news)

Decay function: $f(n) = \beta * n^{-1.5}$





Infectiveness of a blog-post

β – Strength of infection (quality of news)

$f(n)$ – Decay function (how infective a blog posting is)

SpikeM-base (details)

Equations of SpikeM (base)

$$\underbrace{\Delta B(n+1)}_{\text{Blogged}} = U(n) \cdot \sum_{t=n_b}^{n} (\Delta B(t) + S(t)) \cdot f(n+1-t) + \varepsilon$$

$$\underbrace{U(n+1)}_{\text{Un-informed}} = U(n) - \Delta B(n+1)$$

N – Total population of available bloggers

β – Strength of infection/news

n_b, S_b – External shock S_b at birth (time n_b)

ε – Background noise

SpikeM - with periodicity (details)

Full equation of SpikeM

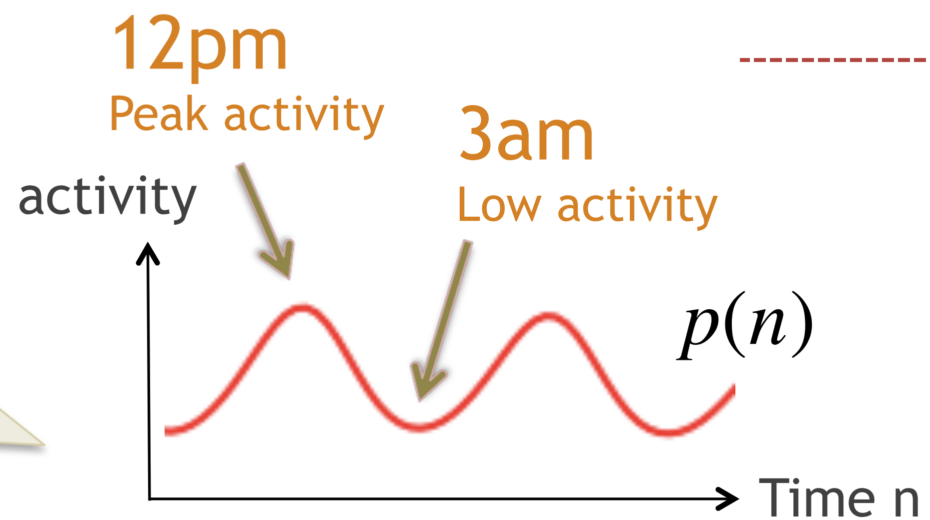
$$\Delta B(n+1) = \boxed{p(n+1)} \cdot \left[U(n) \cdot \sum_{t=n_b}^{n} (\Delta B(t) + S(t)) \cdot f(n+1-t) + \varepsilon \right]$$

Blogged Periodicity

$$U(n+1) = U(n) - \Delta B(n+1)$$

Un-informed

Bloggers change their activity over time
(e.g., daily, weekly, yearly)



Model fitting (Details)

SpikeM consists of 7 parameters

$$\theta = \{N, \beta, n_b, S_b, \varepsilon, P_a, P_s\}$$

Learning parameters

- Given a real time sequence

$$X = \{X(1), \ldots X(n), \ldots, X(n_d)\}$$

- Minimize the error

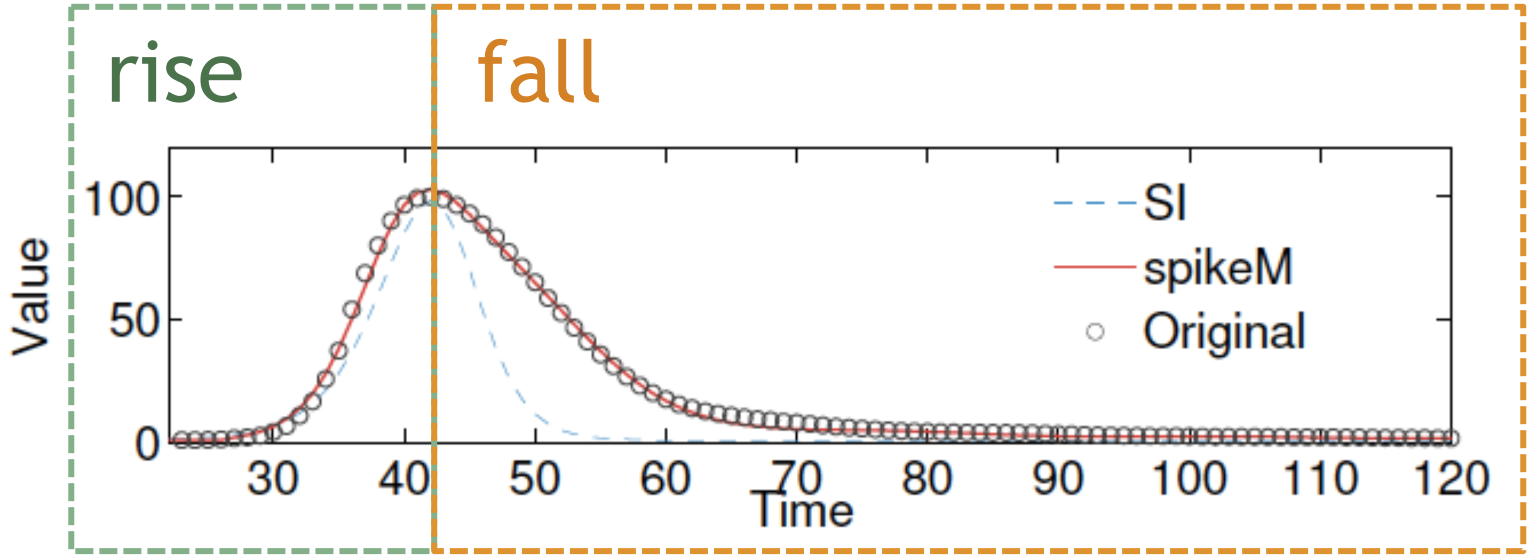
(Levenberg-Marquardt (LM) fitting)

$$D(X, \theta) = \sum_{n=1}^{n_d} (X(n) - \Delta B(n))^2$$

Analysis

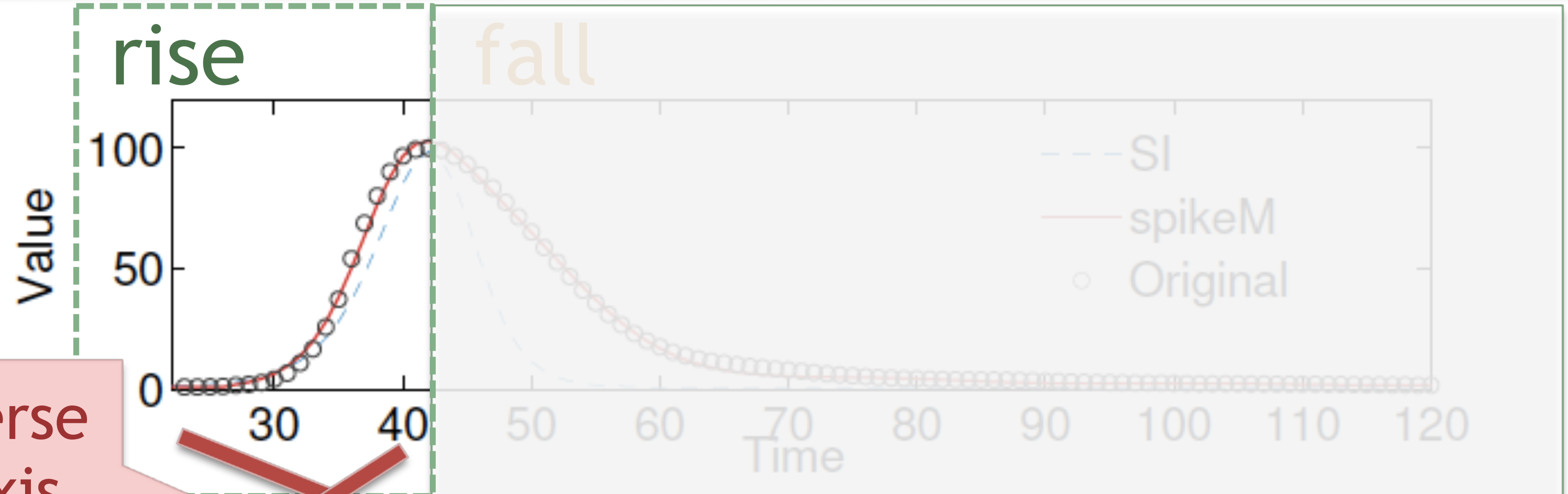
SpikeM matches reality

exponential rise and power-raw fall



SpikeM vs. **SI model** (susceptible infected model)

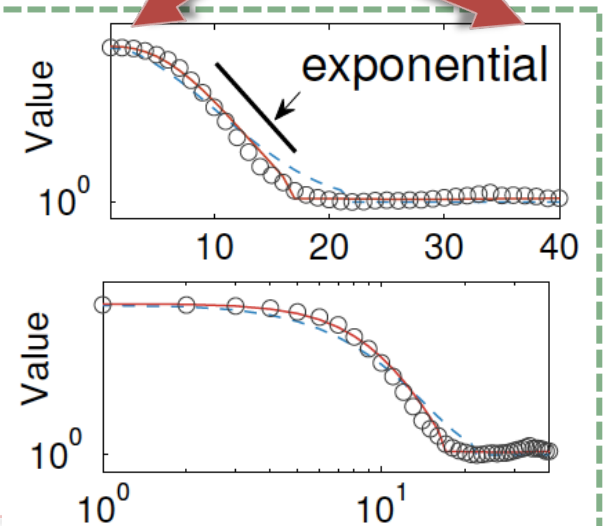
Analysis



Reverse x-axis

Linear-log

Log-log

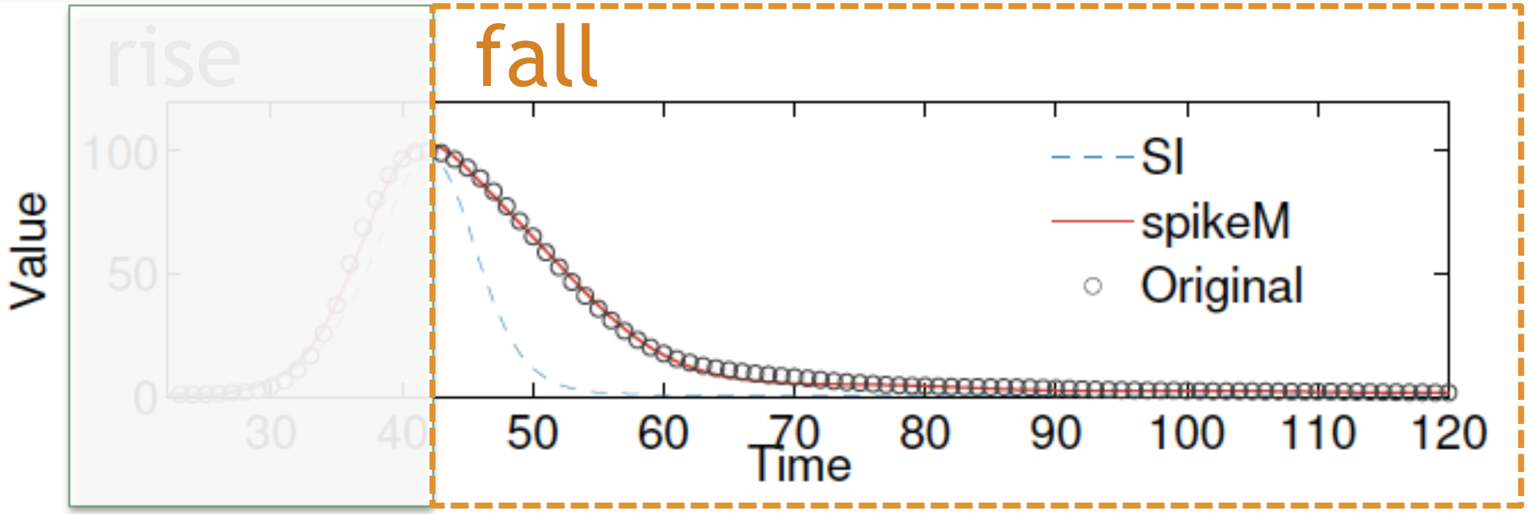


Rise-part

SpikeM: exponential

SI model: exponential

Analysis

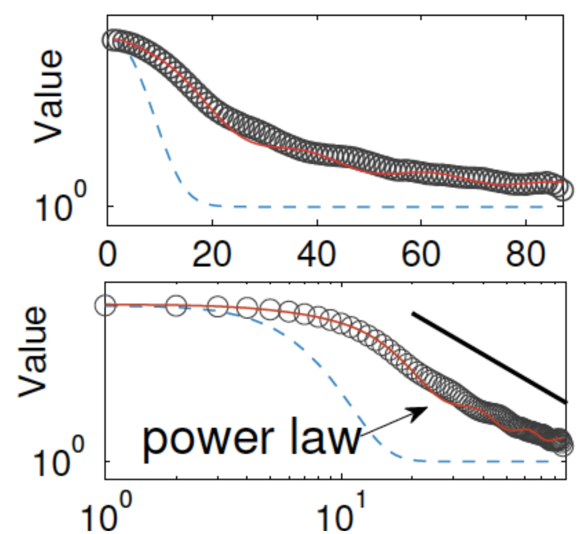


Fall-part

SpikeM: power law

SI model: exponential

SpikeM matches reality



Linear-log

Log-log

Outline

- Motivation
- Problem definition
- Proposed method
- Experiments
- Discussion – SpikeM at work
- Conclusions

Experiments

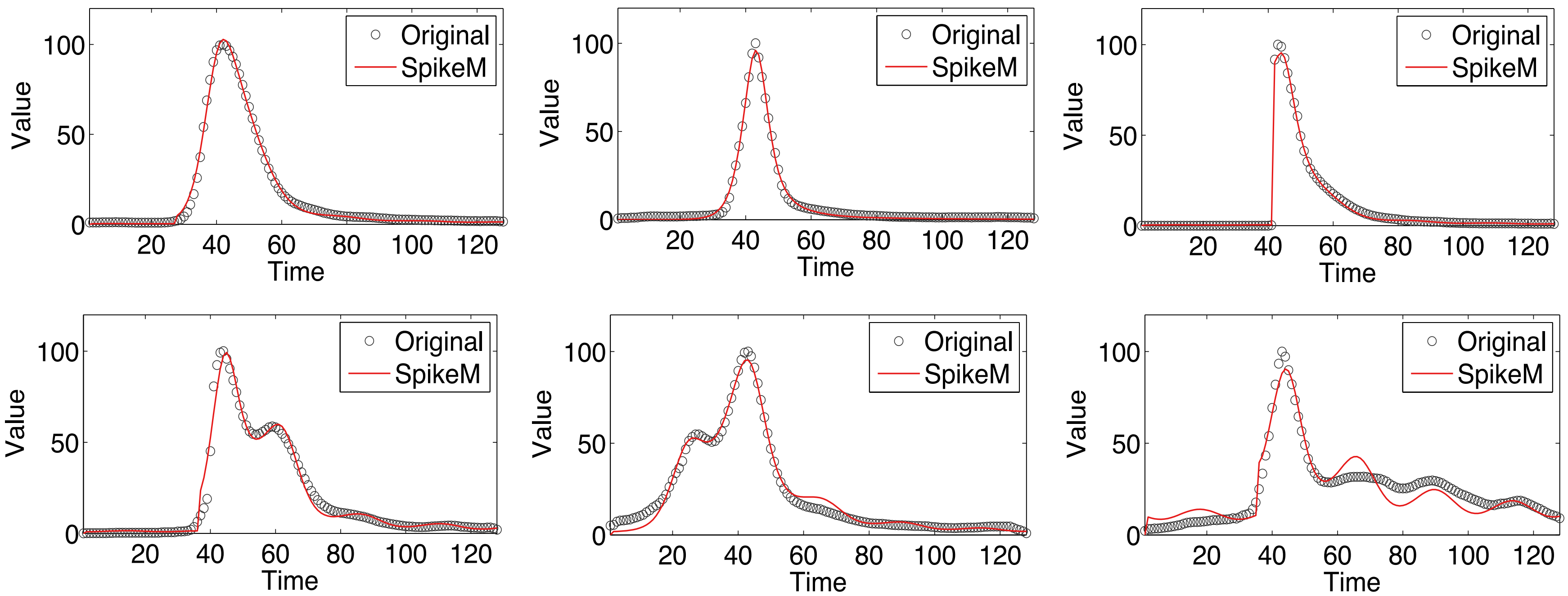
We answer the following questions…

Q1. Match real spikes
- Q1-1: K-SC clusters
- Q1-2: MemeTracker
- Q1-3: Twitter
- Q1-4: Google trend

Q2. Forecast future patterns

Q1-1 Explaining K-SC clusters

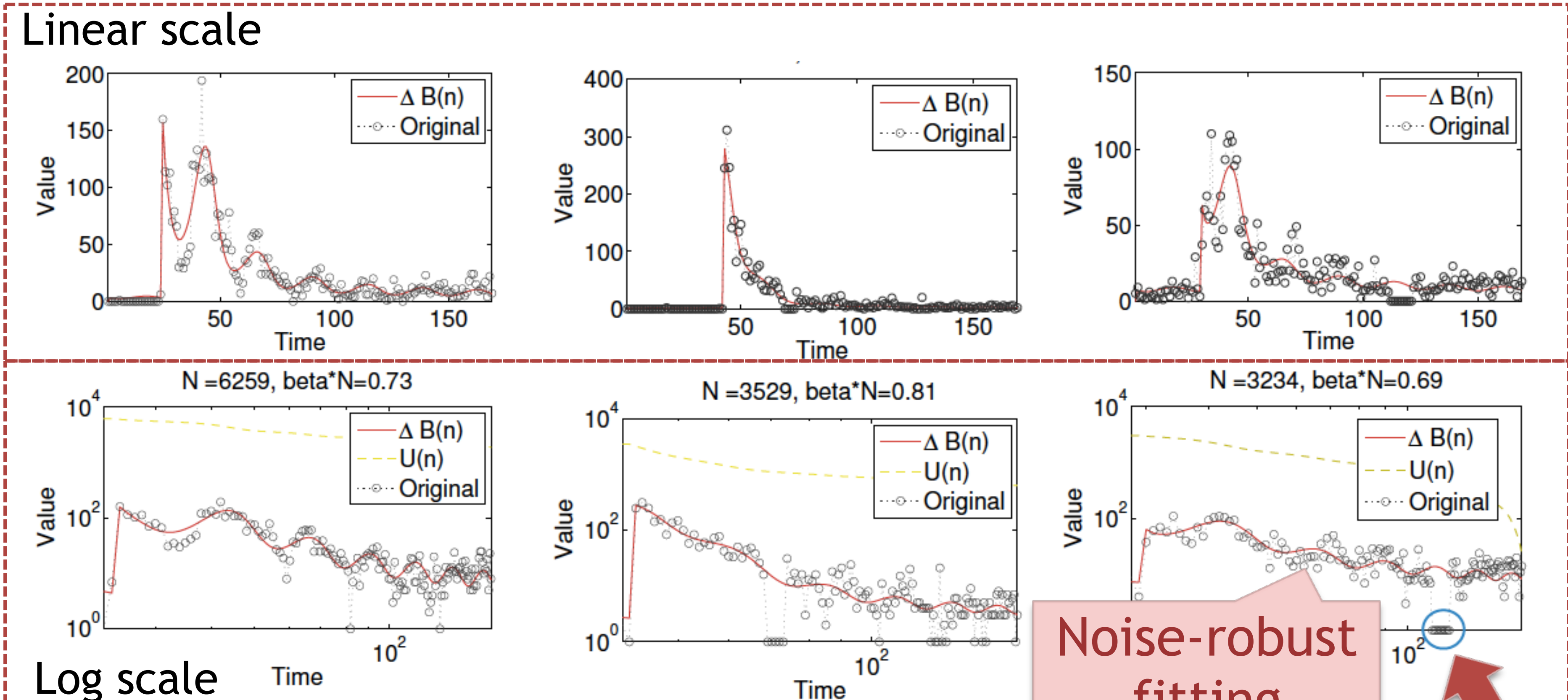
Six patterns of K-SC [Yang et al. WSDM'11]



SpikeM can generate all patterns in K-SC

Q1-2 Matching MemeTracker patterns

MemeTracker (memes in blogs) [Leskovec et al. KDD'09]



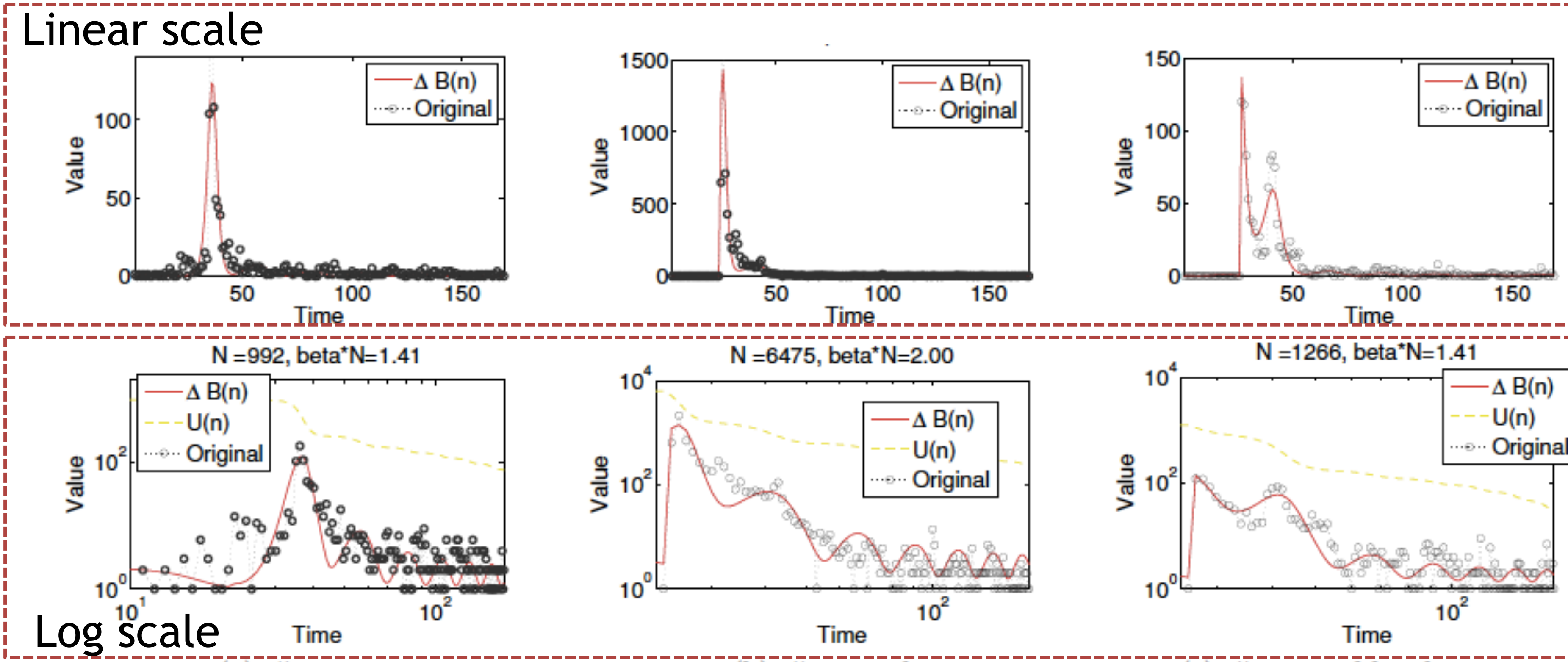
Noise-robust fitting

Outliers

SpikeM can fit various patterns in blog

Q1-3 Matching Twitter data

Twitter data (hashtags)

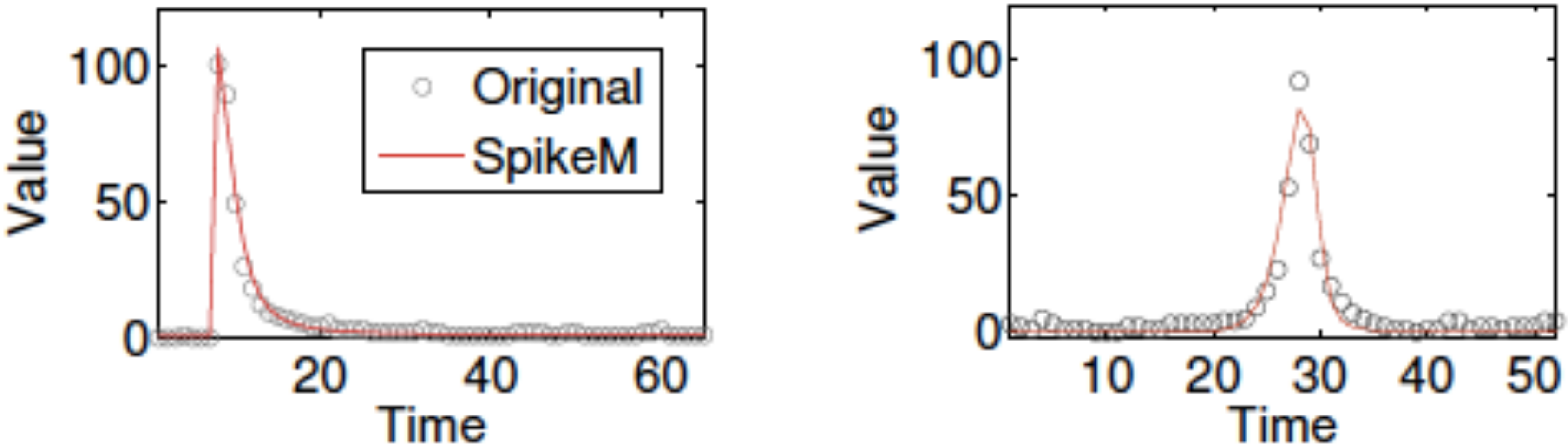


(a) #assange (b) #stevejobs (c) #arresteddevelopment

SpikeM can generate various patterns in social media

Q1-4 Matching Google trend data

Volume of searches for queries on Google

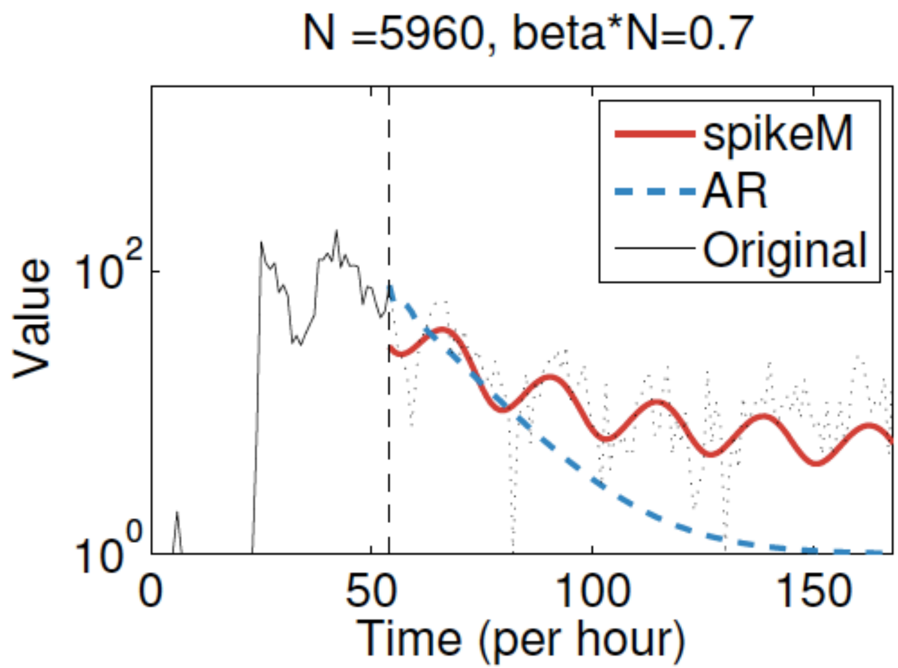


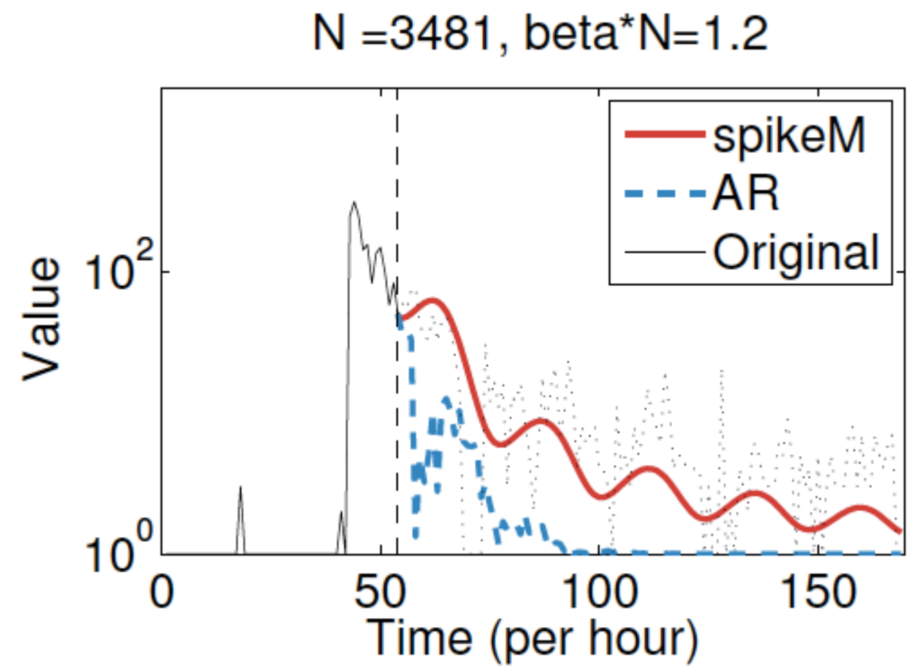
(a) "tsunami" (2005)   (b) "Harry Potter" (2007)

SpikeM can capture various patterns

Q2 Tail-part forecasts

- Given a first part of the spike
    - forecast the tail part





SpikeM can capture tail part (AR: fail)

Outline

- Motivation
- Problem definition
- Proposed method
- Experiments
- Discussion – SpikeM at work
- Conclusions

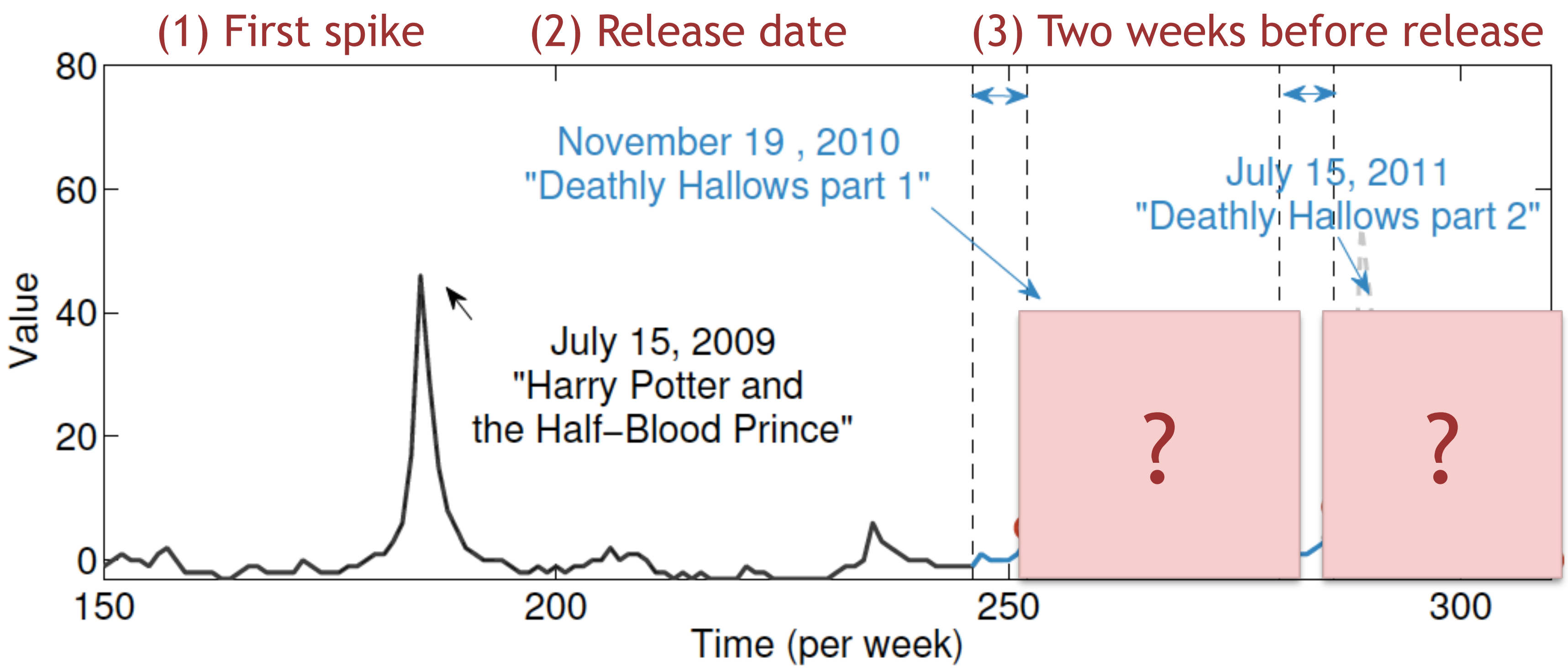
SpikeM at work

SpikeM is capable of various applications

- A1. What-if forecasting
- A2. Outlier detection
- A3. Reverse engineering

A1. "What-if" forecasting

Forecast not only tail-part, but also **rise-part!**



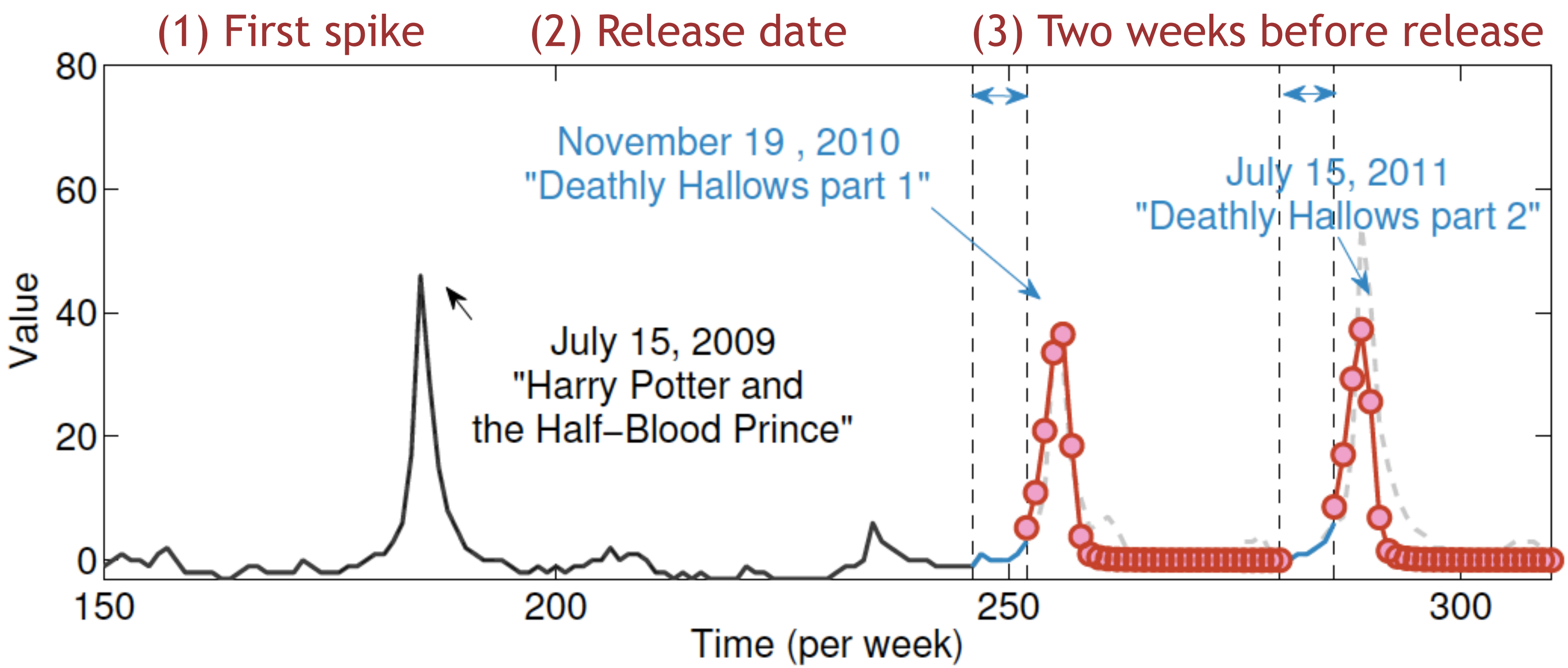
e.g., given (1) first spike,

(2) release date of two sequel movies

(3) access volume before the release date

A1. "What-if" forecasting

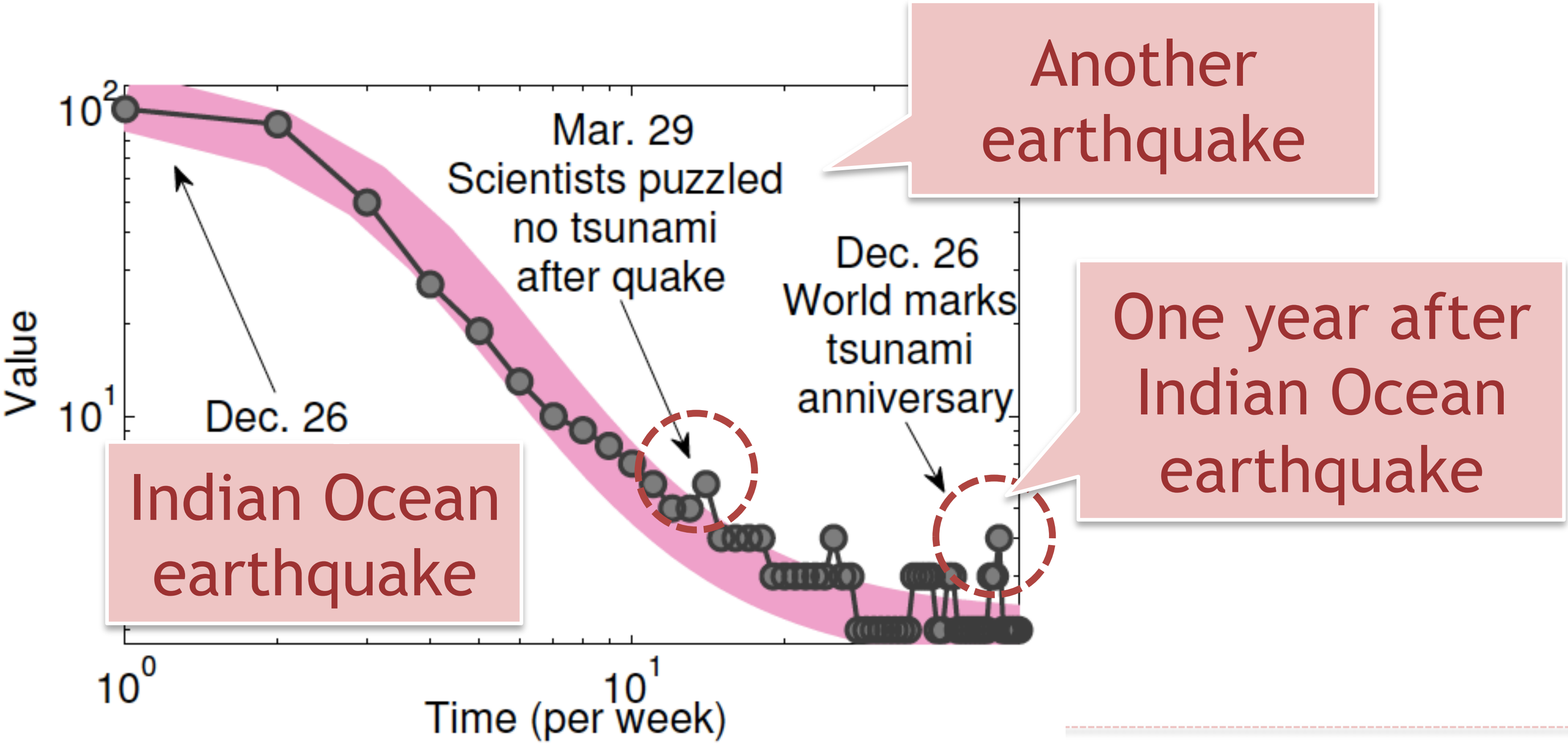
Forecast not only tail-part, but also **rise-part!**



SpikeM can forecast upcoming spikes

A2. Outlier detection

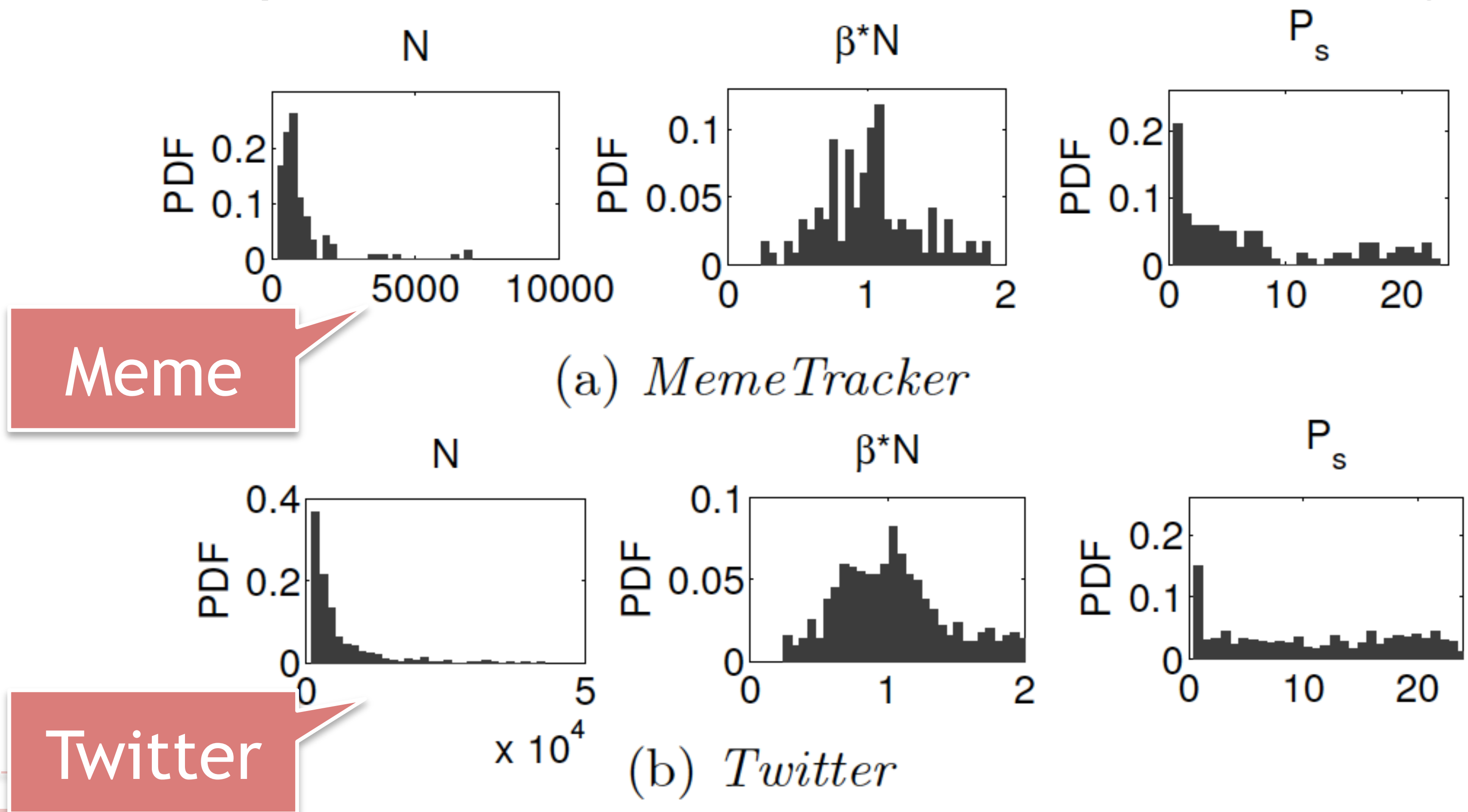
Fitting result of "tsunami (Google trend)" **in log-log** scale



A3. Reverse engineering

SpikeM provide an intuitive explanation

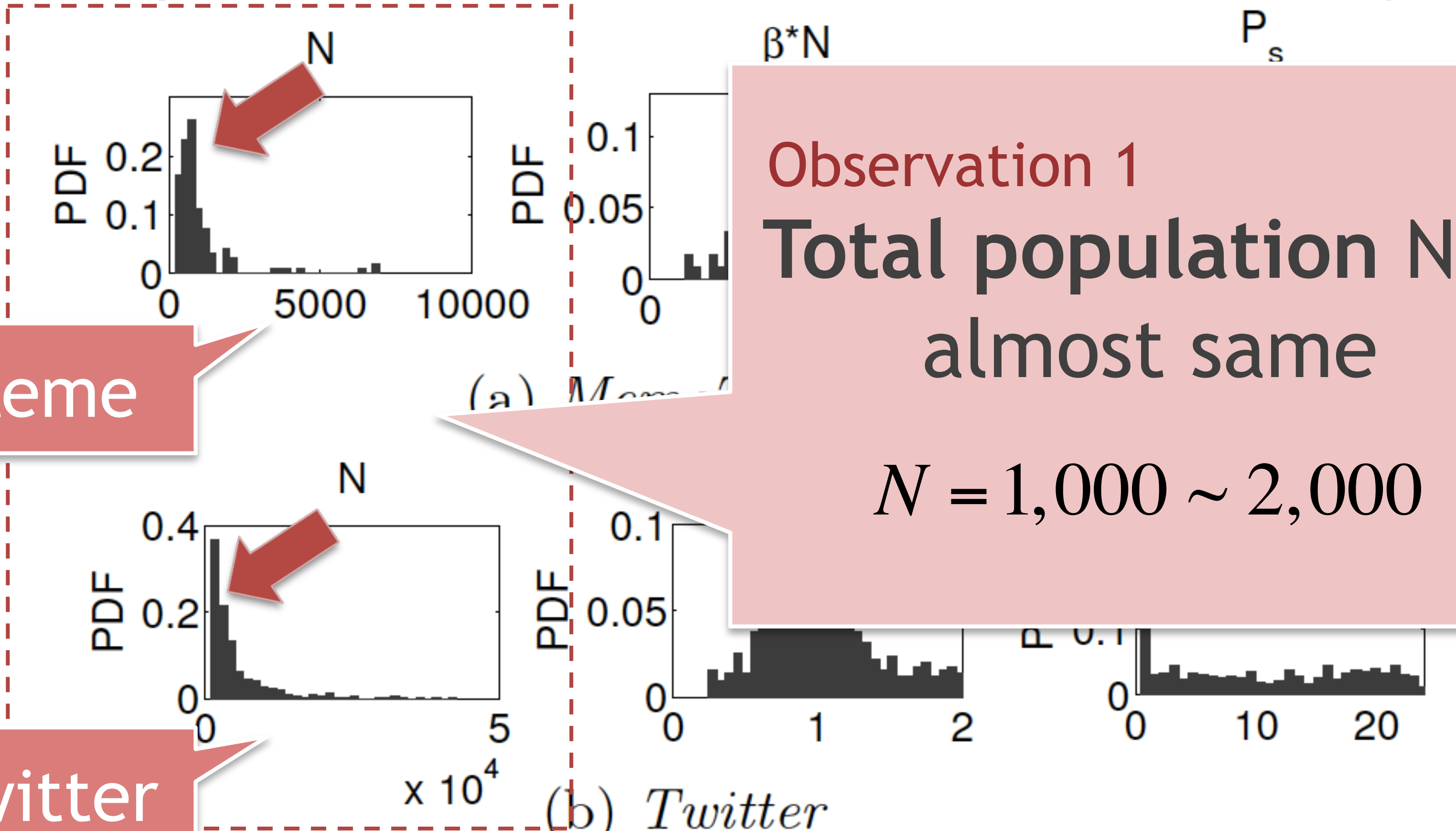
PDF of parameters over 1,000 memes/hashtags



A3. Reverse engineering

SpikeM provide an intuitive explanation

PDF of parameters over 1,000 memes/hashtags



Meme

Twitter

Observation 1

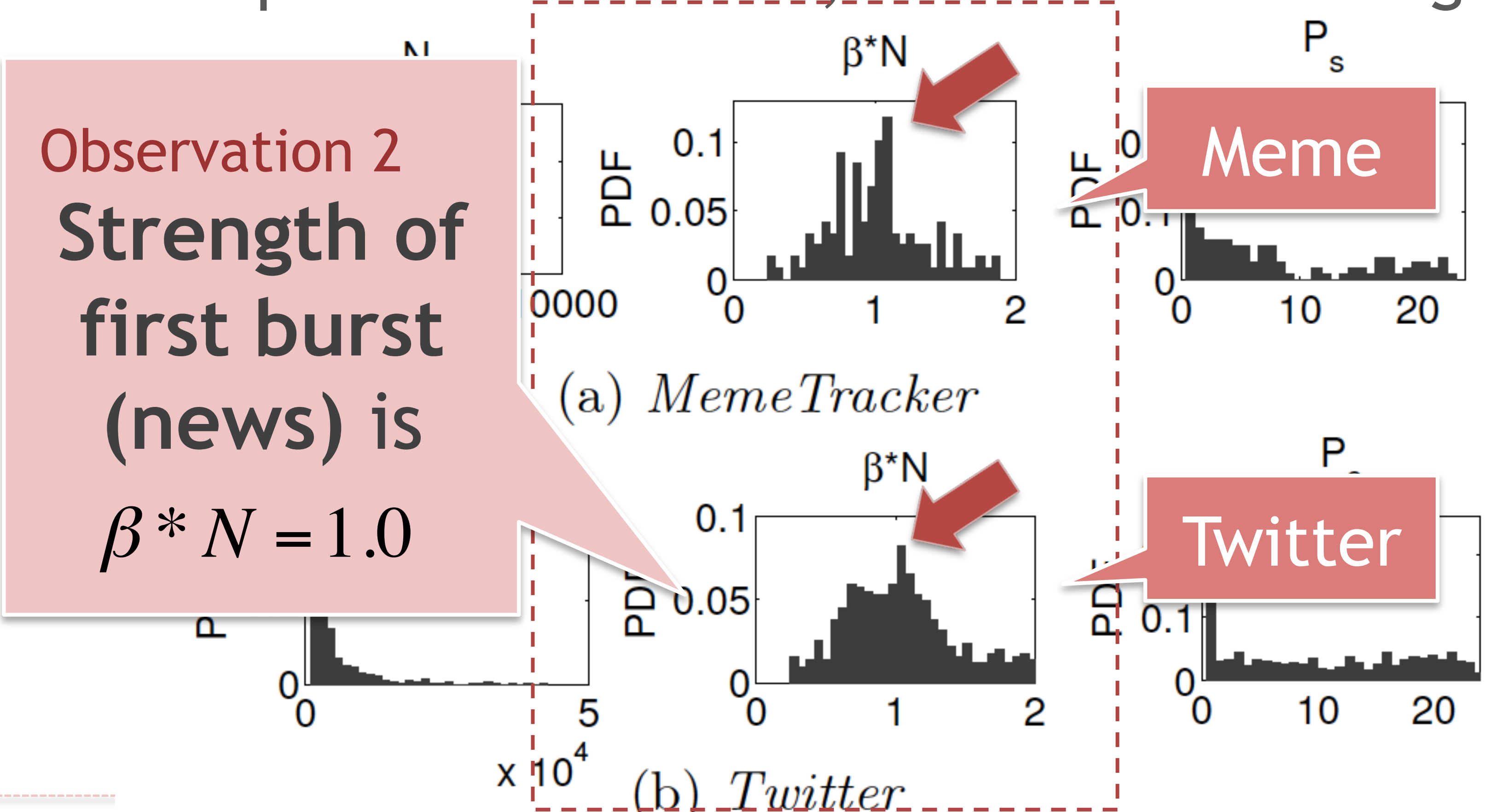
Total population N is almost same

$$N = 1,000 \sim 2,000$$

A3. Reverse engineering

SpikeM provide an intuitive explanation

PDF of parameters over 1,000 memes/hashtags



A3. Reverse engineering

SpikeM provide an intuitive explanation

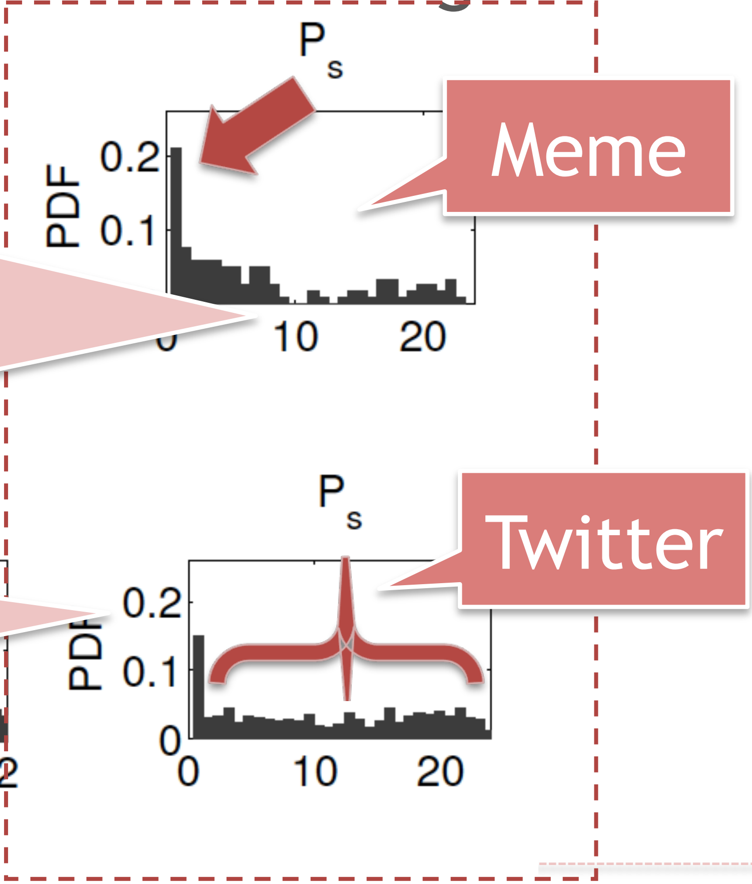
P memes/hashtags

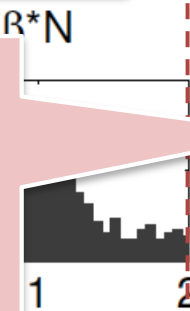
Observation 3

Daily periodicity

with phase shift $P_s = 0$

Every meme has the same periodicity without lag





(Twitter)

Daily periodicity with

more spread in P_s

(i.e., Multiple time zone)

Outline

- Motivation
- Background
- Proposed method
- Experiments
- Discussion – SpikeM at work
- Conclusions

Conclusions

SpikeM has following advantages:

- **Unification power**

  It includes earlier patterns/models

- **Practicality:**

  It Matches real datasets

- **Parsimony**

  It requires only 7 parameters

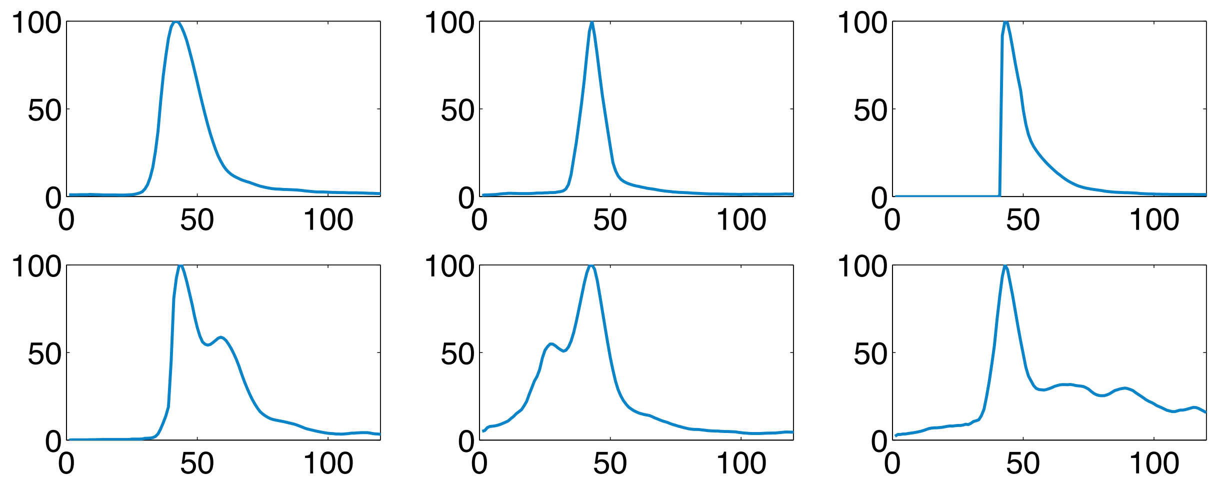
- **Usefulness:**

  What-if scenarios, outliers, etc.

Acknowledgements

Thanks Jaewon Yang & Jure Leskovec

for the six clusters [WSDM'11]



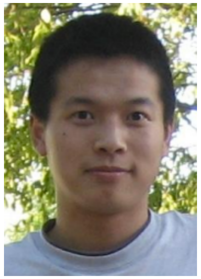
Funding

Thank you

Yasuko Matsubara

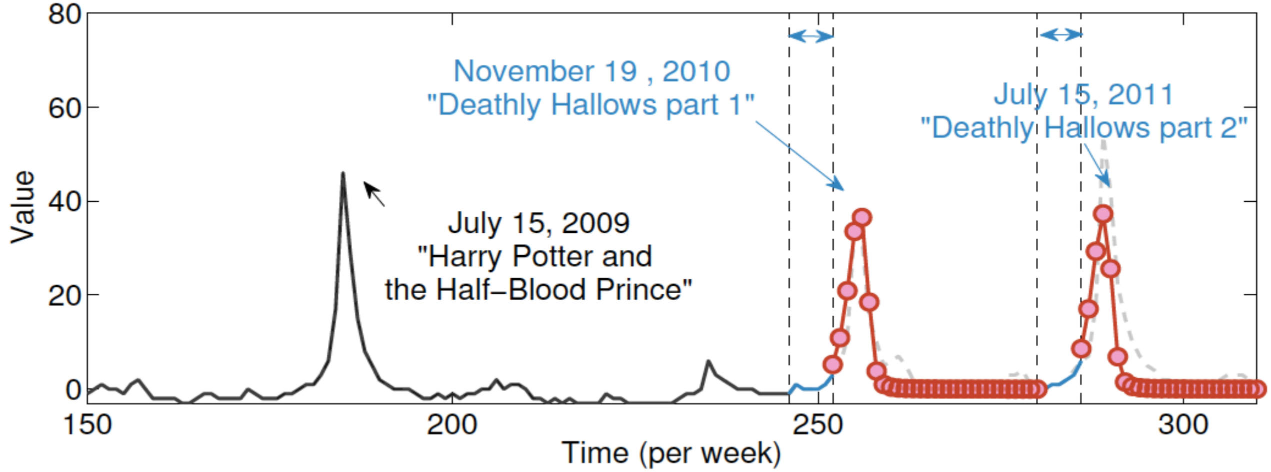
Yasushi Sakurai

B. Aditya Prakash



Lei Li

Christos Faloutsos



Code: http://www.kecl.ntt.co.jp/csl/sirg/people/yasuko/software.html

Email: matsubara.yasuko @ lab.ntt.co.jp