Prelim Exam

Xi Tan

April 15, 2016

Problem Description

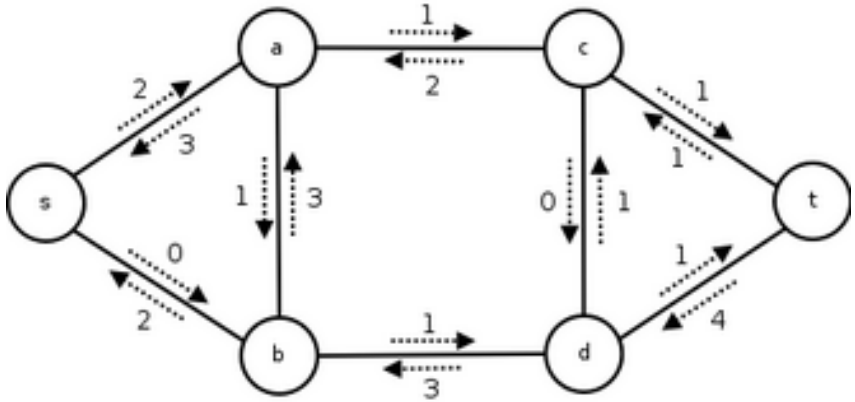


Figure: An network consists of three elements: *nodes*, *links*, and *messages*. (Source: Wikipedia)

1. Learn cluster structure of nodes;
2. Model link intensities;
3. Model message contents.

Contributions

1. INMF extended previous parametric models to a non-parametric one, esp. learn # clusters from data and can impose cluster structure constraints when learning;
2. HP+GP+IRM extended the HP+IRM by introducing GPs to learn clusters and links from message contents;
3. Both use CRP, MCMC to model and learn network structures;
4. Propose to combine and generalize INMF and HP+GP+IRM and design a new (and efficient) algorithm to learn the model.

Table of Contents

TABLE OF CONTENTS

Page

LIST OF TABLES . . . iv

LIST OF FIGURES . . . v

ABSTRACT . . . vii

1 Background . . . 1
- 1.1 Modeling Entity Link Structure: the Infinite Relational Model (IRM) . . . 1
  - 1.1.1 Introduction to the Infinite Relational Model . . . 1
  - 1.1.2 Inference of the Infinite Relational Model . . . 1
- 1.2 Modeling Temporal Dynamics of Link Intensities: the Poisson Processes (PP) and Hawkes Processes (HP) . . . 4
  - 1.2.1 Introduction to Temporal Point Processes . . . 4
  - 1.2.2 Poisson Processes . . . 7
  - 1.2.3 Hawkes Processes . . . 8
- 1.3 Two Example Frameworks: PP + IRM and HP + IRM . . . 11
  - 1.3.1 The Chinese Restaurant Process (CRP) . . . 11
  - 1.3.2 Poisson Processes with Infinite Relational Model (PP+IRM) . . . 11
  - 1.3.3 Hawkes Processes with Infinite Relational Model (HP+IRM) . . . 15
- 1.4 Modeling Link Intensities based on Marks: Gaussian Processes (GP) . . . 17
- 1.5 Approximate Inference Algorithms: Markov Chain Monte Carlo (MCMC) Methods . . . 18

2 An Example of Learning Structures: Learning Network Legos by Infinite Non-Negative Matrix Factorization . . . 23
- 2.1 Introduction . . . 23
- 2.2 Model . . . 27
- 2.3 Inference . . . 30
  - 2.3.1 Joint posterior distribution of z_{ik} and a_{kj} . . . 30
  - 2.3.2 Sample z_{ik} and a_{kj} . . . 31
  - 2.3.3 Speed Up the Gibbs Sampler . . . 32
- 2.4 Experimental Results . . . 34
  - 2.4.1 Results on Synthetic Dataset . . . 34
  - 2.4.2 Results on Real Datasets . . . 39

3 An Example of Learning Link Intensities: Content-based Modeling of Reciprocal Relationships using Hawkes Processes and Gaussian Processes (HP + GP + IRM) . . . 43

Page

- 3.1 Introduction . . . 43
- 3.2 Model . . . 45
- 3.3 Inference . . . 47
  - 3.3.1 Notations . . . 47
  - 3.3.2 Inference using the Gibbs Sampling Algorithm . . . 48
- 3.4 Experimental Results . . . 51
  - 3.4.1 Results on Synthetic Datasets . . . 51
  - 3.4.2 Results on Real Datasets . . . 56

4 Proposal for the Thesis: Sparse Approximate Learning of Multivariate Generalized Hawkes Processes . . . 61
- 4.1 Proposed Extensions . . . 61
- 4.2 Preliminary Results: An Example of Efficient Learning Algorithms (EP + SGD) . . . 63
  - 4.2.1 Introduction to Expectation Propagation (EP) . . . 63
  - 4.2.2 Algorithm . . . 67
  - 4.2.3 Experimental Results . . . 71

5 Summary . . . 83

REFERENCES . . . 84

A Derivations Related to the Inference of Hawkes Processes . . . 87
- A.1 Rate Functions of Hawkes Processes . . . 87
- A.2 Stationary Conditions of Hawkes Processes . . . 88
- A.3 Recursive Definition of Rate Functions . . . 89
- A.4 Cumulative Rate Functions . . . 90
- A.5 Inference of Hawkes Processes . . . 91

B Simulation of Hawkes Processes . . . 94
- B.1 Simulation of Univariate Hawkes Processes . . . 94
- B.2 Simulation of Multivariate Hawkes Processes . . . 96

C EP and SGD Related Derivations . . . 98
- C.1 Product of Two Gaussian Density Functions . . . 98
- C.2 Update Parameters of $q(\theta)$ using SGD . . . 98
- C.3 Proof of the Integral of the Product of Gaussian CDF and PDF . . . 99
- C.4 Proof of the Moments of the Product of Gaussian CDF and PDF . . . 100
  - C.4.1 First Moment . . . 100
  - C.4.2 Second Moment . . . 102
  - C.4.3 The Joint Density of $p(\theta, \mathbf{x})$ in the Clutter Example . . . 103

The Chinese Restaurant Process (CRP)

Suppose there are infinite number of round tables; the first customer chooses a new table with probability 1; the n^{th} customer with probability $\frac{1}{N-1+\alpha}$ chooses to sit with either any one of the previous $n-1$ customers, or a new table with probability $\frac{\alpha}{N-1+\alpha}$. The joint probability is

$$p(\pi|\alpha) = \frac{\alpha^{|B|}\prod_{j=1}^{|B|}(|B_j|-1)!}{\prod_{j=1}^{N}(j-1+\alpha)} = \alpha^{|B|}\frac{\Gamma(\alpha)}{\Gamma(N+\alpha)}\prod_{j=1}^{|B|}\Gamma(|B_j|) \quad (1)$$

where $|B|$ is the total number of tables, and $(|B_j|-1)!$ is the factorial of $|B_j|-1$, the number of individuals in the j^{th} table minus one.

Infinite Relational Model (IRM)

$$\pi|\alpha \sim CRP(\alpha) \tag{2}$$

$$\lambda_{pq}|\gamma \sim Beta(\gamma,\gamma) \quad \forall p,q \in range(\pi) \tag{3}$$

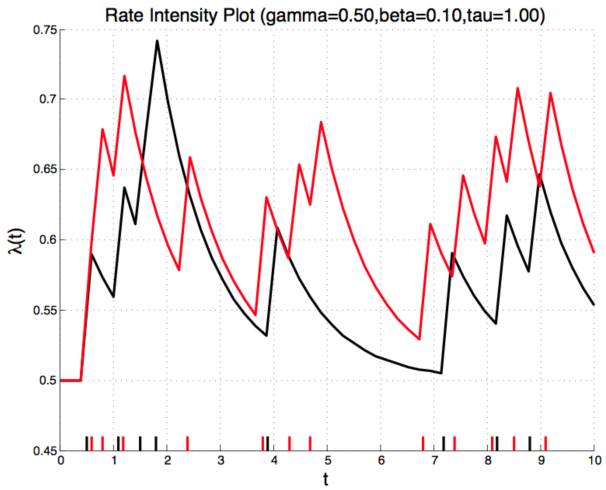
$$e_{uv}|\pi,\lambda_{\pi(u)\pi(v)} \sim Bernoulli(\lambda_{\pi(u)\pi(v)}) \quad \forall u,v \in V \tag{4}$$

Point Processes

	$\lambda(s)$: rate function at a specific time s
Homogeneous Poisson Processes	constant, independent of time
Non-homogeneous Poisson Processes	constant, dependent on time
Doubly Stochastic (Cox) Processes	independent random variable
Self-exciting Hawkes Processes	random variable dependent on history of oneself
Mutually-exciting Hawkes Processes	random variable dependent on history of others

Table: Comparison of different point processes in terms of rate function types.

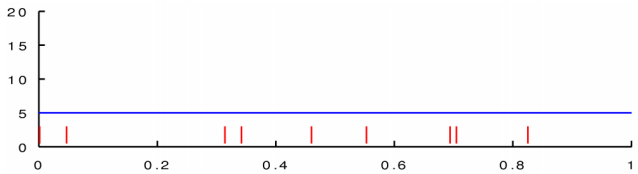
Hawkes Processes (1/2)



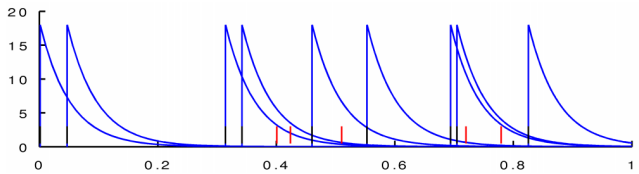
$$\lambda_{uv}(t) = \gamma + \beta \int_0^t e^{-\frac{t-s}{\tau}} dN_{vu}(s) \tag{5}$$

If $\beta = 0$, Hawkes Processes reduce to (Homogeneous) Poisson Processes.

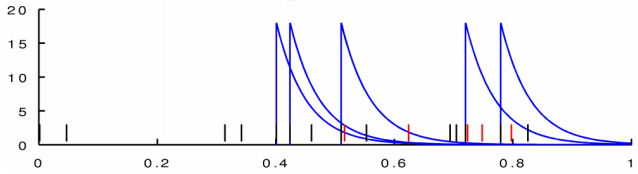
Hawkes Processes (2/2)



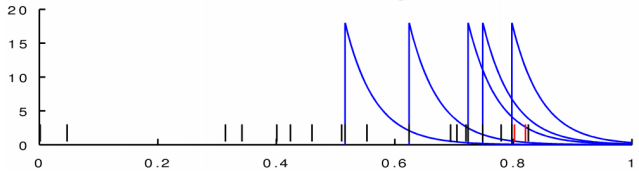
(a) Mother process realisation



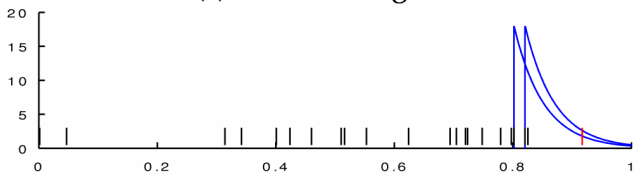
(b) Level 1 daughters



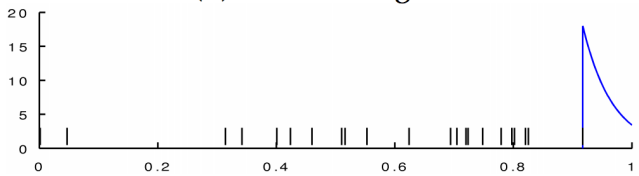
(c) Level 2 daughters



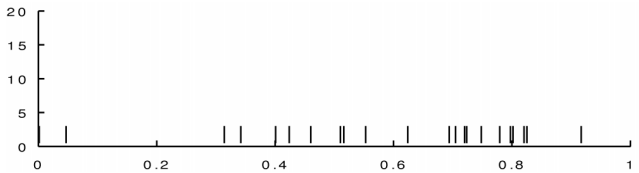
(d) Level 3 daughters



(e) Level 4 daughters



(f) Level 5 daughters



(g) Final observations

Poisson IRM

$$\pi|\alpha \sim CRP(\alpha) \tag{6}$$

$$\lambda_{pq}|\delta,\beta \sim Gamma(\delta,\beta) \quad \forall p,q \in range(\pi) \tag{7}$$

$$N_{uv}(\cdot)|\pi,\lambda_{\pi(u)\pi(v)} \sim PoissonProcess(\lambda_{\pi(u)\pi(v)}) \quad \forall u,v \in V \tag{8}$$

where π is the assignment vector of length $|V|$, the total number of individuals; λ_{pq} the rate of an individual in cluster p sends a message to an individual in cluster q; N_{uv} the number of messages sending from individuals u to v.

Hawkes IRM

$$\pi|\alpha \sim CRP(\alpha) \tag{9}$$

$$\lambda_{pq}(t)|\gamma_{pq}, \beta_{pq}, \tau_{pq} = \gamma_{pq} n_p n_q \quad + \int_{-\infty}^{t} g_{pq}(t-s) dN_{qp}(s) \;\; \forall p, q \in range(\pi) \tag{10}$$

$$N_{pq}(\cdot)|\lambda_{pq} \sim HawkesProcess(\lambda_{pq}) \tag{11}$$

$$N_{uv}(\cdot)|N_{\pi(u)\pi(v)}, \pi \sim Thin(N_{\pi(u)\pi(v)}) \;\; \forall u, v \in V \tag{12}$$

The operator *Thin* refers to thinning; this means distributing the atoms of $N_{pq}(\cdot)$ among each $N_{uv}(\cdot)$, such that $N_{pq} = \sum_{u,v} N_{u,v}(\cdot)$.

Gaussian Processes (GP)

The observed dataset $\mathcal{D} = \{(x_1, y_1), \cdots, (x_n, y_n)\}$ and the point of interest (x_*, y_*) are jointly Gaussian:

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{k}_* \\ \mathbf{k}_*^T & k_{**} \end{bmatrix}\right) \tag{13}$$

where $\mathbf{k}_* = k(x_*, x_i)$ and $k_{**} = k(x_*, x_*)$. The posterior of $y_*|\mathbf{y}$ is also Gaussian, with mean and variance

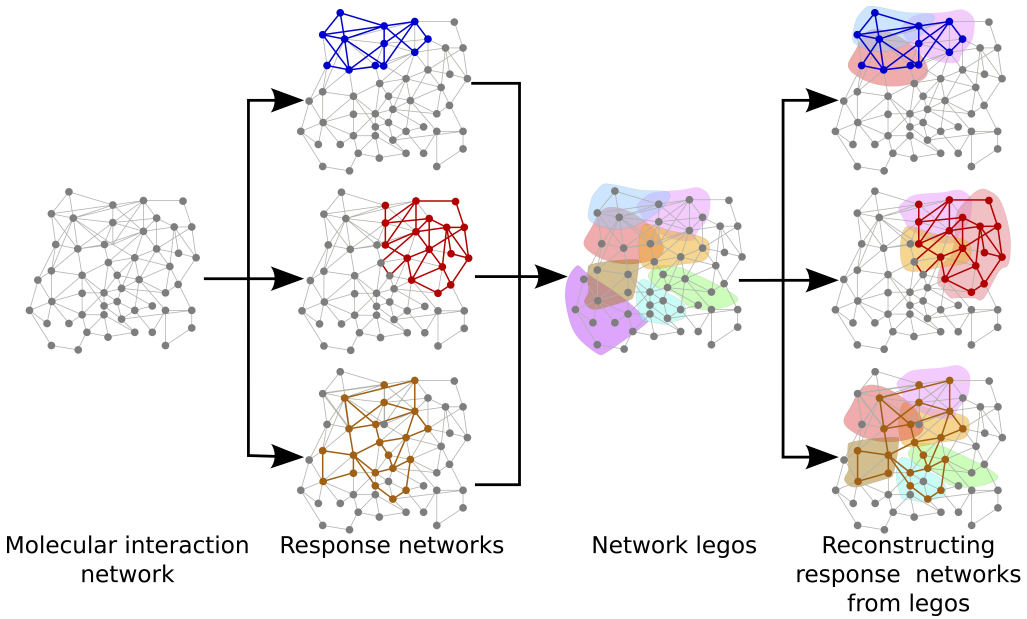
$$\mu = \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{y} \tag{14}$$

$$\sigma^2 = k_{**} - \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{k}_* \tag{15}$$

The inference of Gaussian Processes, largely depends on the covariance matrix and its inverse (the precision matrix), costs $O(n^2)$ space and $O(n^3)$ computations.

Learning Network Legos by Infinite Non-negative Matrix Factorization (INMF)

Problem Description



1. Columns 1 and 2: A collection of response networks;
2. Column 3: Want to learn network legos as building blocks;
3. Column 4: Express response networks in terms of legos.

Matrix Representation

1. Response Network $\mathbf{Y}_{N\times D}$

$$\mathbf{Y} = \begin{array}{c} \\ RN_1 \\ \cdots \\ RN_N \end{array} \begin{array}{c} \begin{array}{cccc} Edge_1 & Edge_2 & \cdots & Edge_D \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 1 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 1 & \cdots & 1 \end{pmatrix}_{N\times D} \end{array}$$

2. Loading Matrix $\mathbf{Z}_{N\times K}$

$$\mathbf{Z} = \begin{array}{c} \\ RN_1 \\ \cdots \\ RN_N \end{array} \begin{array}{c} \begin{array}{cccc} Lego_1 & Lego_2 & \cdots & Lego_K \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 1 & 1 & \cdots & 0 \end{pmatrix}_{N\times K} \end{array}$$

3. Lego Matrix $\mathbf{A}_{K\times D}$

$$\mathbf{A} = \begin{array}{c} \\ Lego_1 \\ \cdots \\ Lego_K \end{array} \begin{array}{c} \begin{array}{cccc} Edge_1 & Edge_2 & \cdots & Edge_D \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 1 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 1 & \cdots & 1 \end{pmatrix}_{K\times D} \end{array}$$

Earlier Work

This problem is essentially a matrix factorization problem, where a number of different methods have been developed, such as PCA, SVD, NMF, and etc. There are several limitations for the earlier work.

1. Need to specify the number of legos K, which is actually unknown.
2. When dealing with graphs, structure (connectivity, overlapness, and etc.) not take into account.
3. We propose a new method called Infinite-NMF (INMF), which can learn K automatically from the data, and find the best factorization under various structure constraints.

Model Description

1. The likelihood can be modeled as:

$$P(\mathbf{Y}|\mathbf{Z},\mathbf{A}) = p^{\|\mathbf{Y}\|_0 - \|\mathbf{Y}\oplus\hat{\mathbf{Y}}\|_0}(1-p)^{\|\mathbf{Y}\oplus\hat{\mathbf{Y}}\|_0}$$

where $\hat{\mathbf{Y}} = (\mathbf{ZA} > 0)$

2. The prior for $\mathbf{Z}$ is an Indian Buffet Process (IBP)

$$P(\mathbf{Z}) = \frac{\alpha^{K_+}}{\prod_{h=1}^{2^N-1} K_h!}\exp\{-\alpha H_N\}\prod_{k=1}^{K_+}\frac{(N-m_k)!(m_k-1)!}{N!}$$

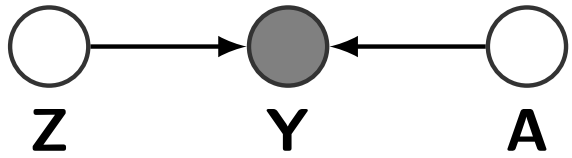
3. The conditional for an edge a_{kj} is described explicitly

$$p(a_{kj}=1|\mathbf{a}_{k,-j}) = \begin{cases} 0 & \text{if } a_{kj} \text{ is NOT attached to } \mathbf{a}_{k,-j} \\ 1 & \text{if } a_{kj} \text{ is attached to } \mathbf{a}_{k,-j} \text{ and a bridge} \\ \frac{J1}{J1+J0} & \text{if } a_{kj} \text{ is attached to } \mathbf{a}_{k,-j} \text{ and NOT a bridge} \end{cases}$$

where $J1$ and $J0$ can be any structure measure when $a_{kj}=1$ and $a_{kj}=0$, respectively.

The MCMC Solution

First of all, notice that $\mathbf{Z}$ and $\mathbf{A}$ are conditionally independent given the observation $\mathbf{Y}$



So we can write

$$p(z_{ik}, a_{kj}|\mathbf{A}_{-kj}, \mathbf{Z}_{-ik}, \mathbf{Y}) = p(z_{ik}|\mathbf{A}_{-kj}, \mathbf{Z}_{-ik}, \mathbf{Y})p(a_{kj}|\mathbf{A}_{-kj}, \mathbf{Z}, \mathbf{Y}) \quad (16)$$

$$p(z_{ik}|\mathbf{A}_{-kj}, \mathbf{Z}_{-ik}, \mathbf{Y}) \propto p(z_{ik}|\mathbf{Z}_{-ik}) \sum_{a_{kj}=0,1} p(\mathbf{Y}|\mathbf{Z}, \mathbf{A})p(a_{kj}|\mathbf{A}_{-kj}) \quad (17)$$

$$p(a_{kj}|\mathbf{A}_{-kj}, \mathbf{Z}, \mathbf{Y}) \propto p(\mathbf{Y}|\mathbf{Z}, \mathbf{A})p(a_{kj}|\mathbf{A}_{-kj}) \quad (18)$$

Two Tricks that Save Everything

Without going into too much details, there is one simple yet extremely time consuming matrix operation that takes most of the computer time.

Recall that we need to compute $\|\mathbf{Y} \oplus \hat{\mathbf{Y}}\|_0$, where $\hat{\mathbf{Y}} = (\mathbf{ZA} > 0)$ involves

$$\mathbf{Z}_{N\times K} \times \mathbf{A}_{K\times D}$$

where $N \approx 20$, $K \approx 100$, and $D \approx 10K$.

We need to flip the value of just one element in $\mathbf{A}$, and re-evaluate this matrix product. This is repeated $N \times K + K \times D \approx 1M$ times in every iteration. If there are 200 iterations (reasonable number to converge), we are talking about $200 \times 20M \times 1M = 4,000,000,000,000,000$ multiplication operations here. That's why it took 3 weeks.

Trick One: Local Update

Trick One: If we flip the value of A_{kj}, only the *jth* column of **ZA** is affected. No need to compute the whole product, only local update.

Trick Two: The XOR Trick

$\hat{y}_{ij}^{(k)}$	$\hat{y}_{ij}^{(-k)}$	y_{ij}	$\hat{y}_{ij} \oplus y_{ij}$	$\hat{y}_{ij}^{(-k)} \oplus y_{ij}$
0	0	0	$0 \oplus 0 = 0$	$0 \oplus 0 = 0$
0	0	1	$0 \oplus 1 = 1$	$0 \oplus 1 = 1$
0	1	0	$1 \oplus 0 = 1$	$1 \oplus 0 = 1$
0	1	1	$1 \oplus 1 = 0$	$1 \oplus 1 = 0$
1	0	0	$1 \oplus 0 = 1$	$0 \oplus 0 = 0$
1	0	1	$1 \oplus 1 = 0$	$0 \oplus 1 = 1$
1	1	0	$1 \oplus 0 = 1$	$1 \oplus 0 = 1$
1	1	1	$1 \oplus 1 = 0$	$1 \oplus 1 = 0$

$\hat{\mathbf{Y}}^{(k)} = (\mathbf{Z}_k \mathbf{A}_k > 0)$ and $\hat{\mathbf{Y}}^{(-k)} = (\mathbf{Z}_{-k} \mathbf{A}_{-k} > 0)$, so $\hat{\mathbf{Y}} = \hat{\mathbf{Y}}^{(k)} \vee \hat{\mathbf{Y}}^{(-k)}$

Trick Two: We only need to update two cases when $\hat{y}_{ij}^{(k)} = 1$, otherwise, $\|\mathbf{Y} \oplus \hat{\mathbf{Y}}\|_0$ stays the same. This is 4 times faster!

Results on Synthetic Datasets

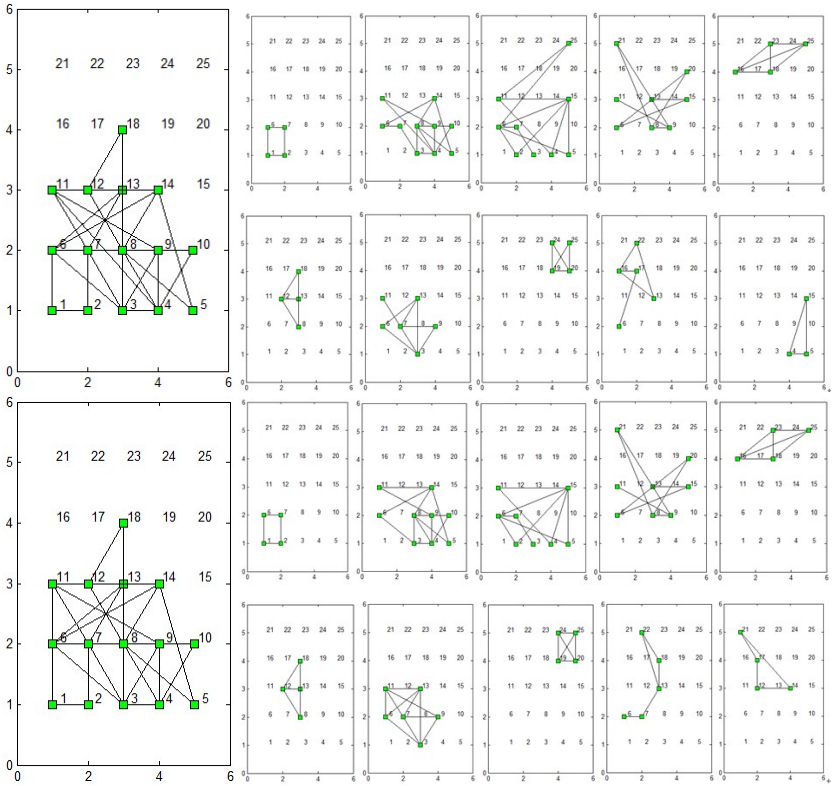


Figure: Response network consists of bases 1, 2, 6, and 7; closewise from top left: true RN, true bases, predicted bases, predicted RN.

Results on Real Datasets

	Node Recovery	Edge Recovery
response network #1	81.25%	80.54%
response network #2	85.48%	87.45%
response network #3	91.25%	90.48%
response network #4	74.25%	68.78%
response network #5	81.47%	80.48%
response network #6	84.91%	85.01%
response network #7	98.55%	97.15%
response network #8	94.15%	90.71%
response network #9	87.54%	84.15%
response network #10	90.54%	89.45%
Average	89.87%	88.23%

Table: Summary statistics of the network legos for the ten response network dataset.

HP+GP+IRM: Content-based Modeling of Reciprocal Relationships using Hawkes and Gaussian Processes

Model Description

We define the Hawkes process conditional rate function as:

$$\lambda_{uv}(t) = \gamma_{pq} + \int_0^t \beta_{uv} e^{-\frac{t-s}{\tau_{uv}}} dN_{vu}(s) \tag{19}$$

where $p = \pi^{-1}(u), q = \pi^{-1}(v)$ are the clusters individuals u and v belong to; and the triggering function $g_{uv}(\cdot)$ is defined as:

$$g_{uv}(\delta) = \beta_{uv} e^{-\frac{\delta}{\tau_{uv}}} \tag{20}$$

Model Description

We propose to use two sets of Gaussian Process (GP) priors to address sources of inhomogeneity of the excitation functions $\beta_{uv}(\cdot)$, one for the significance of the message and one for the receptivity of the message:

$$\beta_{uv}(s) = e^{r_u(x_{vu}(s)) + s_v(x_{vu}(s))} \tag{21}$$

where

$$r_u(\cdot) \sim \mathcal{GP}(0, k_r) \tag{22}$$
$$s_v(\cdot) \sim \mathcal{GP}(0, k_s) \tag{23}$$

k_r and k_s are radial basis function (RBF) kernels of the GPs. The exponential transformation is used to make sure that $\beta_{uv}(\cdot)$ is non-negative.

Model Description

$$\pi|\alpha \sim CRP(\alpha) \tag{24}$$

$$\lambda_{uv}(t)|\gamma_{pq}, \beta_{uv}(\cdot), \tau_{uv} = \gamma_{pq} + \int_{-\infty}^{t} \beta_{uv}(\mathcal{X}_{vu}) e^{-\frac{t-s}{\tau_{uv}}} dN_{vu}(s) \tag{25}$$

$$N_{uv}(\cdot)|\lambda_{uv} \sim HawkesProcess(\lambda_{uv}) \tag{26}$$

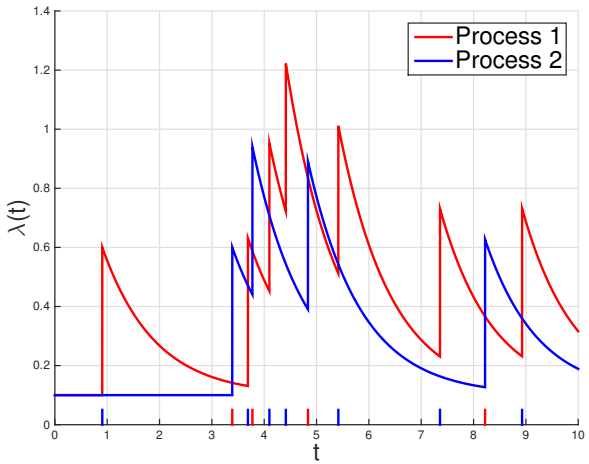
where $\mathcal{X}_{vu} = \{x_{vu}(s)\}$ is the set of all messages sent from v to u, and the cluster level excitation function β_{pq} can be seen as an additive effect of β_{uv}:

$$\beta_{pq}(\mathcal{X}_{qp}) = \sum_{\pi(u)=p, \pi(v)=q} \beta_{uv}(x_{vu}(s)) \tag{27}$$

Inference

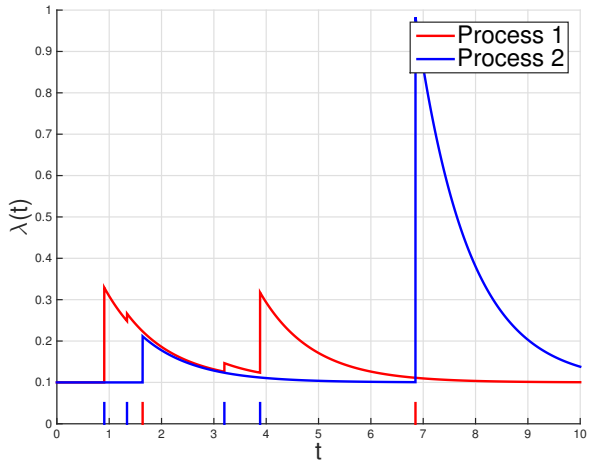
We use elliptical slice sampling to learn GP related quantities, e.g., r_u, s_v, and use slice sampling to learn other parameters.

Synthetic Datasets: Two Individuals (1/3)



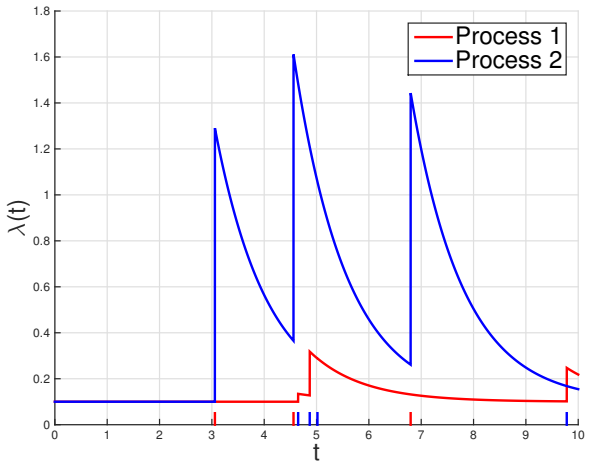
In part (a), case 1 used a constant message content $x_{12}(t_i) = x_{21}(t_i') = 1$ for all event times t_i and t_i', and a constant excitation function $\beta_{12}(x) = \beta_{21}(x) = x = 1$ for all messages.

Synthetic Datasets: Two Individuals (2/3)



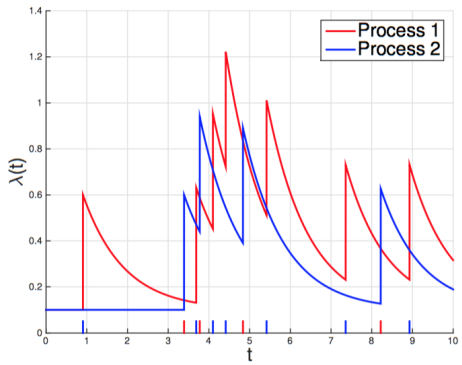
In part (b), case 2 used the same settings as part (a), except for the introduction of variable message content, where both $x_{12}(t_i)$ and $x_{21}(t_i')$ follow an exponential distribution exp(0.5), which can be thought of as different message entropy values at different event times t_i and t_i'. We see that the jump sizes of both processes are no longer constant.

Synthetic Datasets: Two Individuals (3/3)

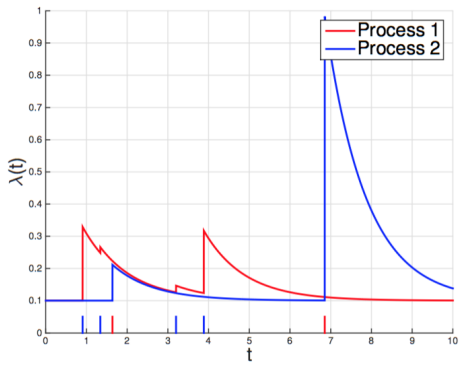


In part (c), case 3 further introduced non-constant $\beta_{uv}(\cdot)$, with all other settings being the same as in case 2, but
$\beta_{12}(t_i) = e^{2sin(x_{21}(t_i))+1.5log(x_{21}(t_i))}$ and
$\beta_{21}(t_i') = e^{0.1cos(x_{12}(t_i'))+0.2\sqrt{x_{12}(t_i')}}$.

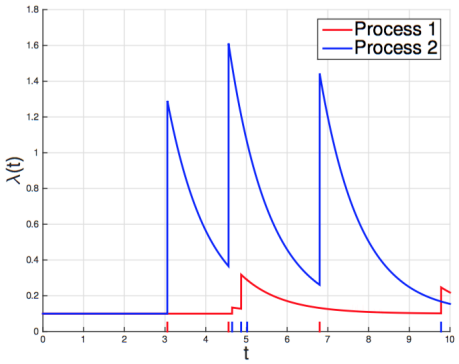
Synthetic Datasets: Two Individuals



(a) Case 1: x constant, β simple function $\beta = x$. The "jump sizes" are constant.



(b) Case 2: x random, β simple function $\beta = x$. The "jump sizes" are not constant.



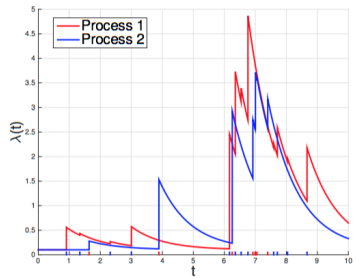
(c) Case 3: x random, β non-trivial function. The "jump sizes" are not constant.

Synthetic Datasets: Two Individuals

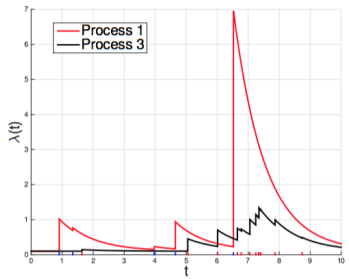
Table: Log likelihood comparison for the three-case synthetic data set

	CASE 1	CASE 2	CASE 3
HP+GP+IRM	-21.88	-13.41	-10.86
HP+IRM	-22.97	-35.53	-82.78
POISSON + IRM	-72.31	-89.73	-126.33
HP	-129.37	-238.94	-192.78
POISSON	-127.83	-182.76	-187.23

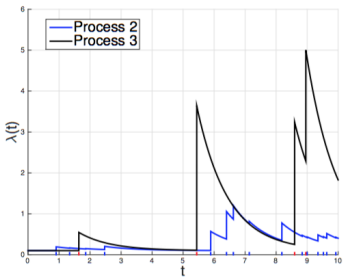
Synthetic Datasets: Three Individuals



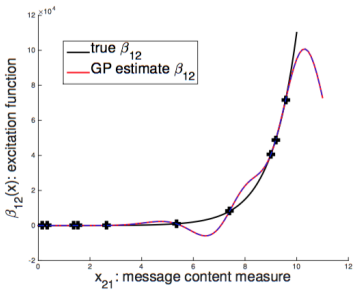
(a) $\beta_{12} = 5\exp(1/x), \beta_{21} = 5\exp(1/x)$



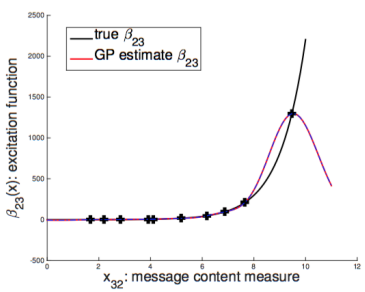
(b) $\beta_{13} = 10\exp(1/x), \beta_{31} = 0.1\exp(1/x)$



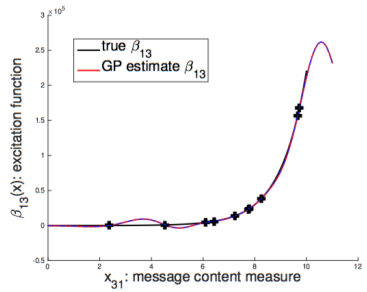
(c) $\beta_{23} = 0.1\exp(1/x), \beta_{32} = 10\exp(1/x)$



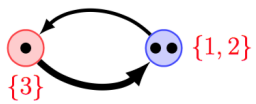
(d) GP plot of β_{12}



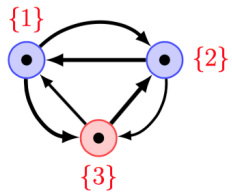
(e) GP plot of β_{23}



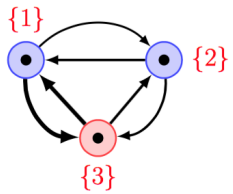
(f) GP plot of β_{13}



(g) HPGP+IRM

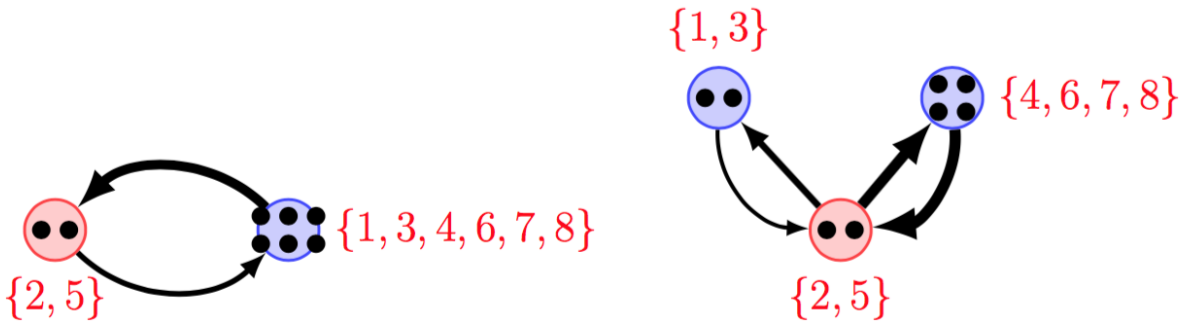


(h) HP + IRM



(i) Poisson + IRM

Real Datasets



(a) HPGP+IRM (b) HP+IRM

Figure: The data set used (#33) is a lively family argument/discussion recorded at a vacation home in Falmouth, Massachusetts. There are eight participants, all relatives or close friends. Discussion centers around a disagreement Jennifer (#2) is having with her mother Lisbeth (#5).

Sparse Multivariate Hawkes Processes

In the HP+GP+IRM framework, we can also impose desired constraints such as sparsity with multivariate Hawkes Processes:

$$\lambda_u(t) = \gamma_{\pi^{-1}(u)} + \sum_{v=1}^{|V|} z_{uv} a_{uv} \int_{-\infty}^{t} \beta_{uv}(\mathcal{X}_{vu}) e^{-\frac{t-s}{\tau_{uv}}} dN_{vu}(s) \quad (28)$$

where the binary adjacency matrix $\mathbf{Z} = \{z_{uv}\}$ and the non-negative weight matrix $\mathbf{A} = \{a_{uv}\}$ specify the structure and strength of the interaction network.

Generalized Hawkes Processes

If we take a closer look at a realized Hawkes Process, especially a high-frequency one, we see that the rate function can be composed into a global "trend" and local "fluctuations". This observation is common in Finance, where "trend" and "fluctuations" are usually referred to as "drift" and "volatility", respectively. To model this two level structure, we may use the Generalized Hawkes Process, where the dynamics of the rate function can be written as:

$$d\lambda(t) = k(\lambda_\infty - \lambda(t))dt + \delta dN_t + \sigma dW_t \tag{29}$$

where the added randomness $W = \{W_t : 0 \leq t\}$ is a standard Brownian motion independent of N_t. The first two terms in the above equation capture the "drift" of the rate function and the last term captures the "volatility" of the rate function.

Efficient Learning Algorithms for Hawkes Processes

Recall the likelihood function of both Poisson Processes and Hawkes Processes (and many other Poisson related processes):

$$\mathcal{L}(\lambda(t)|\mathcal{H}) = \exp\{-\Lambda(0,T)\}\prod_{i=1}^{n}\lambda(t_i) \tag{30}$$

This product form of likelihood fits right into the framework of EP.

Other Possible Extensions

Spectral analysis on Hawkes Processes may provide us with further insight from the frequency domain.

Another perspective of learning Hawkes Processes is inspired by the "drift" and "volatility" observation of the rate function, and their "immigrants" and "offspring" representation. At the first level, we can learn the "drift" part (or the "global" picture, the "trend") of the rate function from the "immigrants" data points, and then recursively learn its "volatility" part (or the "local" picture, the "details").

Summary

1. An Example of Learning Structures: INMF;
2. An Example of Learning from Contents: HP+GP+IRM;
3. Proposal for the Thesis: Sparse Approximate Learning of Multivariate Generalized Hawkes Processes.

Thank You!