On the Convexity of Latent Social Network Inference

NIPS 2010

Seth A. Myers[*] Jure Leskovec[†]

Stanford University

[*]Institute for Computational and Mathematical Engineering

[†]Computer Science

December 8, 2010 - W89

Motivating Problem

Many real world social networks are difficult to observe. For example:

- The sexual relationship network of a population.
- People are not forthcoming with their sexual history.
- Accurately identifying network edges is difficult.

But we can observe diffusive processes over the network

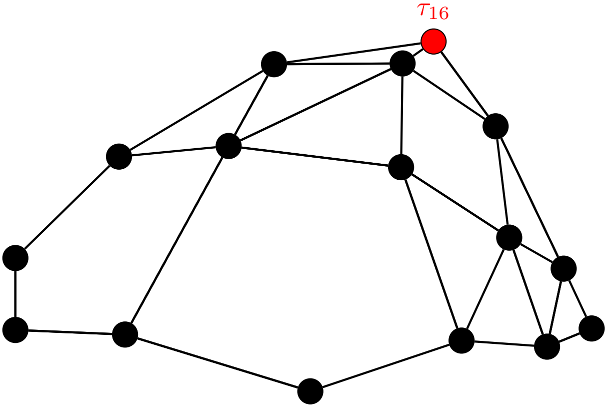
- STD's propagate through sexual relationships
- Observing when people become infected provides insight into the network.

Problem Definition

- An unobservable social network of influence interconnects nodes.
- Diffusive processes can be observed
    - Information cascades
    - Disease outbreaks.
- We observe infection times of nodes, and infer the social network.

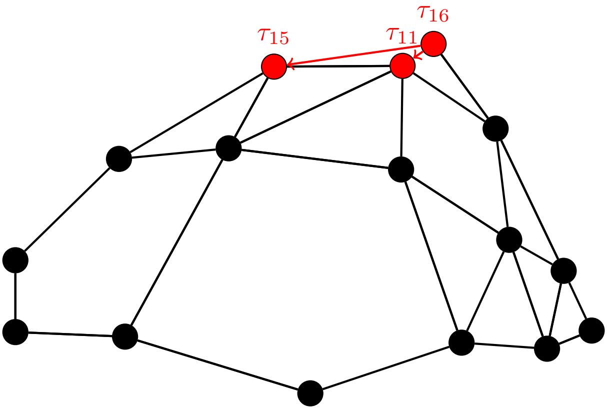
Problem Definition

Example:



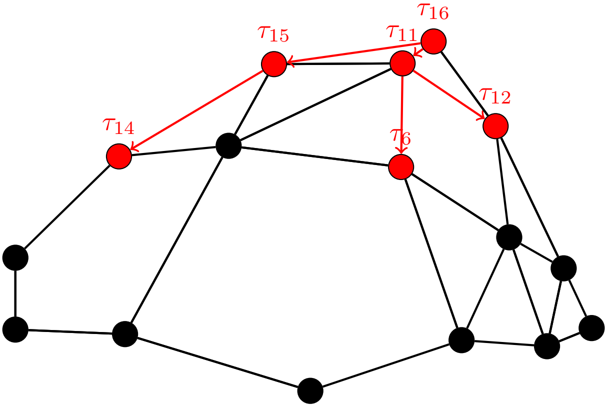
Problem Definition

Example:



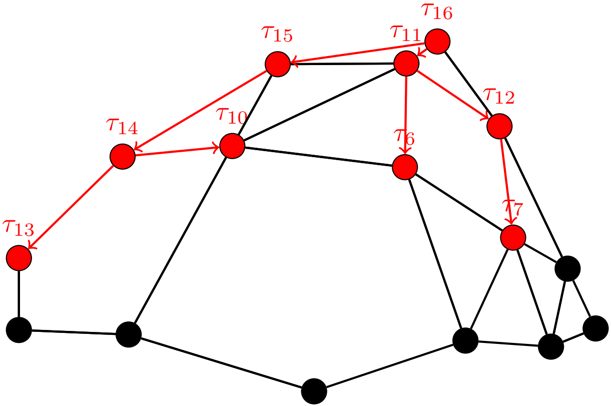
Problem Definition

Example:



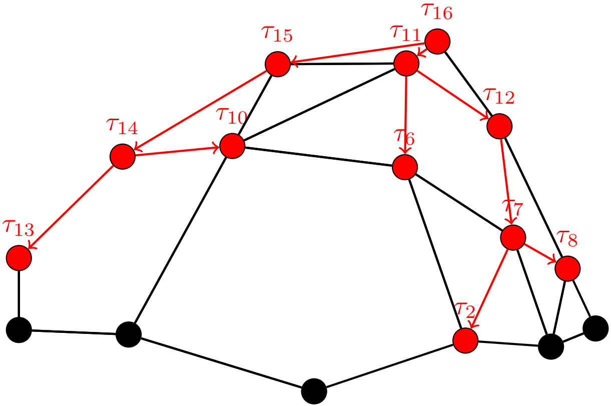
Problem Definition

Example:



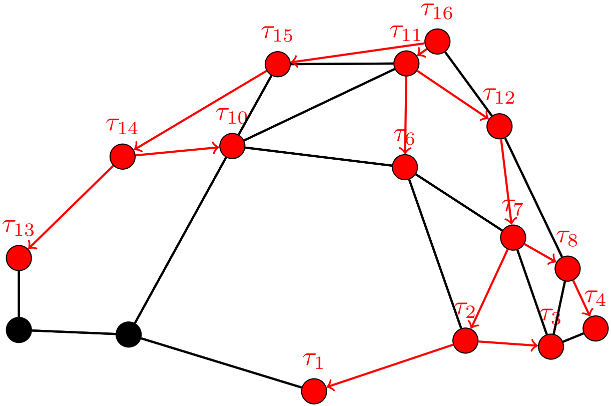
Problem Definition

Example:



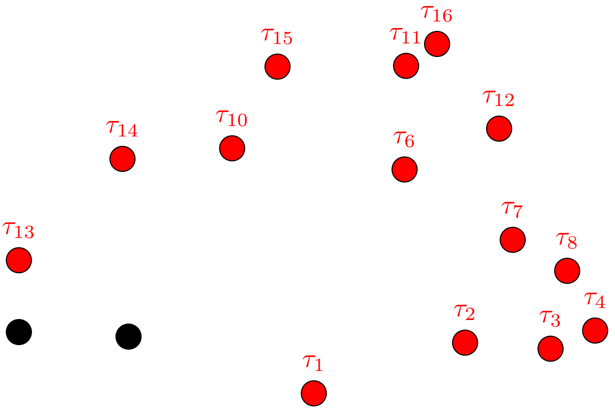
Problem Definition

Example:



Problem Definition

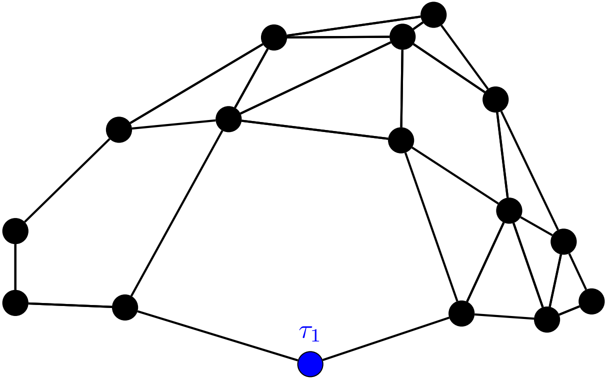
Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

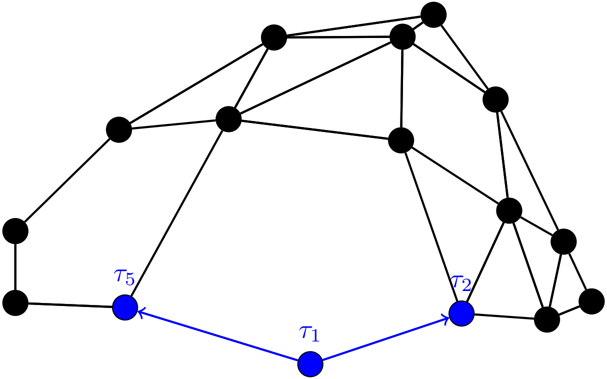
Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

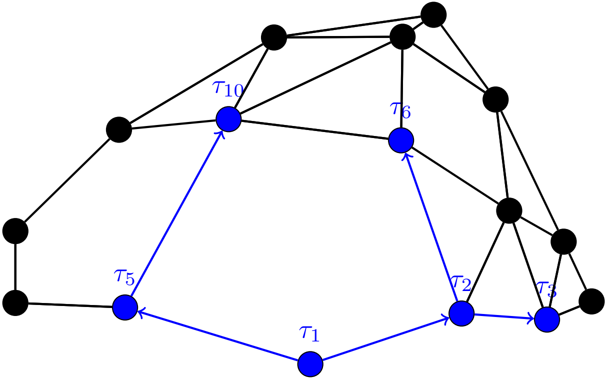
Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

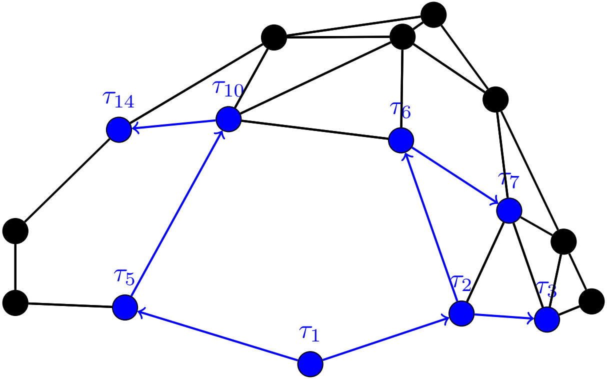
Example:



1st Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

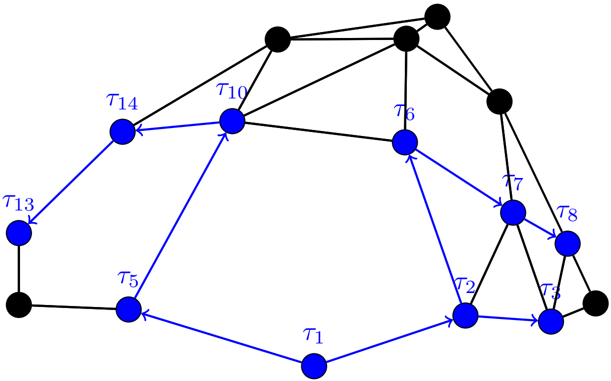
Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

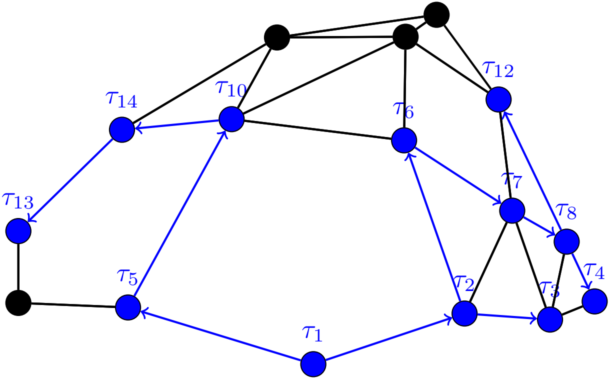
Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

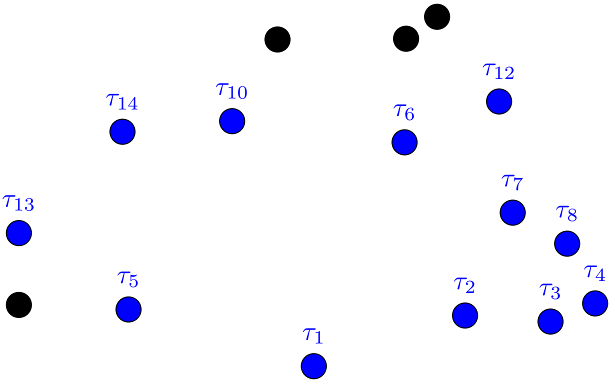
Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

Problem Definition

Example:



1^{st} Cascade: $c_1 = \{\tau_1, \tau_2, ...\}$

2^{nd} Cascade: $c_2 = \{\tau_1, \tau_2, ...\}$

Examples

	Disease Spread	**Viral Marketing**
Process	Infection spreads between people	People recommend products to others
We observe	When people become infected	When people buy products
We do **not** observe	Who infected them	Who influenced them

Examples

	Disease Spread	**Viral Marketing**
Process	Infection spreads between people	People recommend products to others
We observe	When people become infected	When people buy products
We do **not** observe	Who infected them	Who influenced them

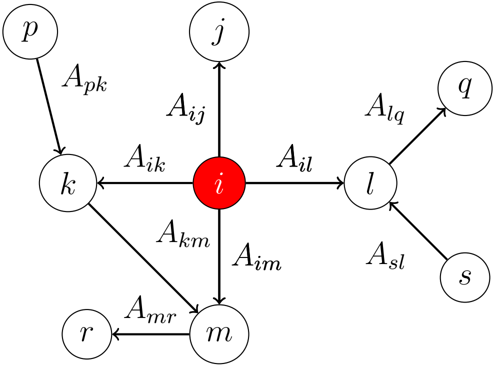
Can we infer who infected who?

Our Approach

- Given:
  - A set of cascades.
- Goal:
  - Infer the network over which the cascades spread i.e it's adjacency matrix A.
    - A_{ij} is the probability of i infecting j.
- Our approach:
  1. Define a probabilistic model for cascade propagation.
  2. Find the likelihood function of observed cascades
  3. Turn likelihood maximization into a series of convex subproblems.
  4. Generalize method to handle sparse networks.

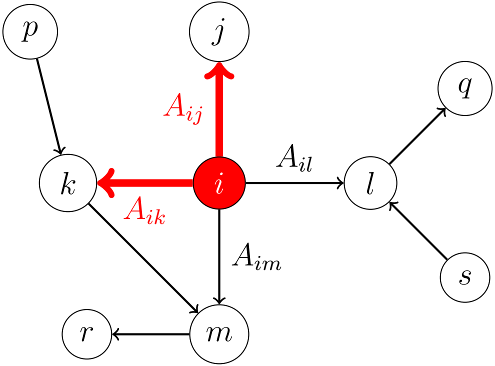
Note: we learn both the structure of the network and the edge weights that model infection probabilities

The Cascade Model



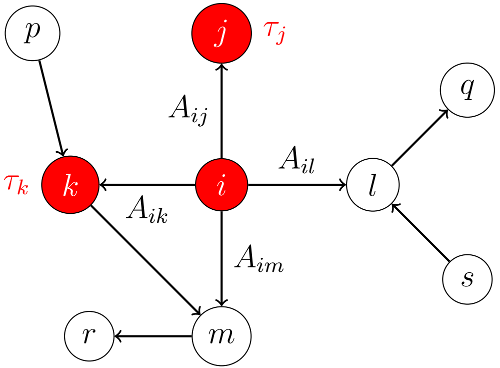
1. Adjacency matrix A defines the influence network.
    - Node i is initially infected.

The Cascade Model



2 Infected node i infects each neighbor j with probability A_{ij}.

The Cascade Model



3. The incubation time of each new infection is sampled from known density $w(t)$.

$$\tau_j = \tau_i + \Delta t_j$$
$$\Delta t_j \sim w(t)$$

The Likelihood Function

- For a given cascade c, we observe the infection time τ_i^c of each node i.
- Then the Likelihood is:

$$L(j \text{ infected } i) = A_{ji} \cdot w(\tau_i^c - \tau_j^c).$$

$$L(i \text{ infected in } c) = 1 - \prod_{j;\tau_j^c<\tau_i^c} \left[1 - A_{ji} \cdot w(\tau_i^c - \tau_j^c)\right].$$

If i is not infected ($\tau_i^c = \infty$):

$$L(i \text{ never infected in } c) = \prod_{j;\tau_j^c<\infty} \left(1 - A_{ji}\right).$$

The Likelihood Function

For all cascades C, the likelihood function is

$$L(A;C) = \prod_{c \in C} \left[\underset{\substack{\uparrow \\ \text{All nodes infected by } c}}{\prod_{i;\tau_i^c < \infty}} L(i \text{ infected in } c) \times \underset{\substack{\uparrow \\ \text{All nodes not infected by } c}}{\prod_{i;\tau_i^c = \infty}} L(i \text{ never infected in } c) \right].$$

To find A, we maximize the likelihood:

$$\boxed{\begin{gathered} \min_{A} \; -\log\left(L(A;C)\right) \\ \text{subject to} \\ 0 \le A_{ij} \le 1 \; \forall \; i,j. \end{gathered}}$$

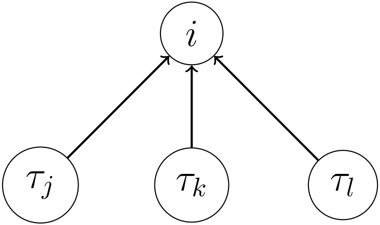
Convexity

- Maximizing the likelihood is non-convex.
- We derive an equivalent convex problem.
  1. Break the problem down into N independent sub problems.
  2. Add parameters to create a geometric program.
  3. Convert geometric program into convex program.

Convexity: Subproblems

- All infections occur independently.
- The likelihood of infection depends only on node's inbound edges.
- We can maximize the likelihood of each node independently
  - N subproblems with $N-1$ parameters.

It does not matter how other nodes become infected.



Convexity

- We treat $L(i \text{ infected in } c)$ as an independent parameter $\gamma_c^{(i)}$:

$$L_i(A;C) = \underset{\substack{\uparrow \\ \text{Cascades that infected } i}}{\prod_{c;\tau_i^c<\infty}} L(i \text{ infected in } c) \times \underset{\substack{\uparrow \\ \text{Cascades that did not infect } i}}{\prod_{c;\tau_i^c=\infty}} L(i \text{ never infected in } c).$$

- We constrain $\gamma_c^{(i)}$:

$$\gamma_c^{(i)} \leq L(i \text{ infected in } c).$$

Convexity

- We treat $L(i \text{ infected in } c)$ as an independent parameter $\gamma_c^{(i)}$:

$$L_i(\gamma^{(i)}, A; C) = \underset{\substack{\uparrow \\ \text{Cascades that infected } i}}{\prod_{c;\tau_i^c<\infty}} \gamma_c^{(i)} \times \underset{\substack{\uparrow \\ \text{Cascades that did not infect } i}}{\prod_{c;\tau_i^c=\infty}} L(i \text{ never infected in } c).$$

- We constrain $\gamma_c^{(i)}$:

$$\gamma_c^{(i)} \leq L(i \text{ infected in } c).$$

Convexity

- Change of variables:

$$\hat{\gamma}_c^{(i)} = \log \gamma_c^{(i)} \qquad \text{and} \qquad \hat{B}_{ji} = \log(1 - A_{ji})$$

- Result is a convex program:

Optimal network guaranteed!

$$\min_{\hat{\gamma}_c, \hat{B}(:,i)} \sum_{c \in C; \tau_i^c < \infty} -\hat{\gamma}_c - \sum_{c \in C; \tau_i^c = \infty} \sum_{j \in C; \tau_j^c < \infty} \hat{B}_{ji}$$

subject to

$$\hat{B}_{ji} \leq 0 \, \forall \, j$$

$$\hat{\gamma}_c \leq 0 \, \forall \, c$$

$$\log \left[\exp \hat{\gamma}_c + \prod_{j; \tau_j \leq \tau_i} \left(1 - w_j^c + w_j^c \exp \hat{B}_{ji} \right) \right] \leq 0 \, \forall \, c.$$

Network Sparsity

- Social networks are almost always sparse.
  - Most pairs of people are not friends/connected.
- The maximum likelihood estimation is almost never sparse.
- The l_1 penalty function ruins convexity.
- We propose a new penalty function:

$$\sum_j \frac{1}{1 - A_{ji}}$$

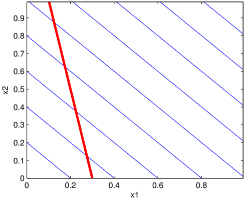
- Convexity is preserved.
- Sparsity is induced.

Network Sparsity

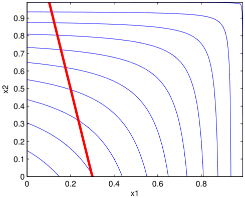
- Why does this sparsity penalty function work?
  - The l_1 penalty comes from the relaxation of

$$\min_x \; ||x||_0$$
$$\text{s.t. } Ax = b.$$

Often, the l_1 and l_0 norms intersect the constraints at the same place.



$||x||_1$



$\sum_i \frac{1}{1-x_i}$

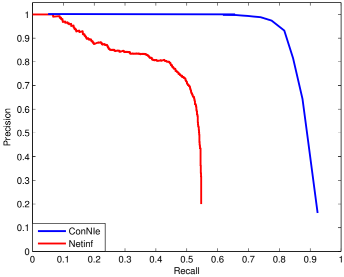
Experimental Setup

- Evaluation Metric:
  - The precision and recall of inferred edges.
  - The mean square error (MSE) of edge weights (infection probabilities).
- Baseline - Netinf [1]
  - An approximation algorithm based on submodular optimization.
  - Assumes all infection probabilities are the same.

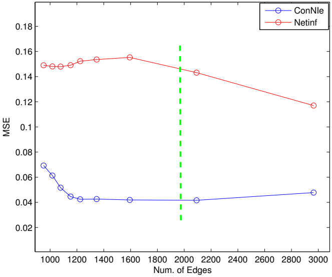
[1] Gomez-Rodriguez, et al; KDD '10

Experiments: Synthetic Network, Synthetic Cascades

1. *Network*: Scale-free Network of $N = 500$ nodes with $M = 2000$ edges
2. *Infection probabilities*: uniform random
3. *Incubation time model*: Power-law: $w(t) \sim t^{-2}$
4. Generated cascades until 99% of all edges propagated an infection



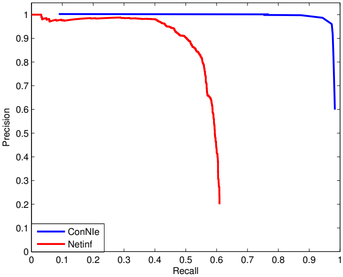
Precision-Recall



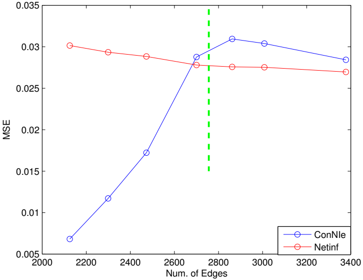
MSE

Experiments: Real Network, Synthetic Cascades

1. *Network*: Real email network, $N = 593$ nodes and $M = 2824$ edges
2. *Infection probabilities*: based on volume of emails
3. *Incubation time model*: Power-law: $w(t) \sim t^{-2}$
4. Generated cascades until 99% of all edges propagated an infection



Precision-Recall

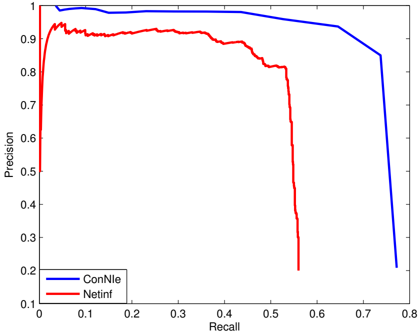


MSE

Experiments: Real Network, Real Cascades

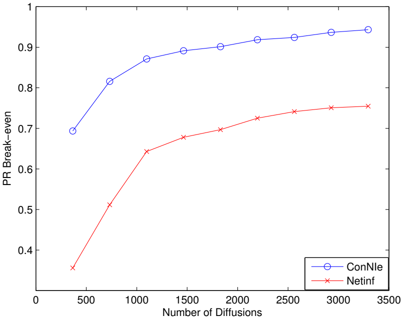
1. *Network*: Recommendation network, $N = 275$ and $M = 1522$
2. *Infection probabilities*: Real
3. *Incubation time model*: Observed to be Power-law
4. Inferring from 625 recommendation cascades.

- Each product is a different cascade
- It is known when one user buys product on recommendation of another user
- Using product purchase times, we infer recommendations



Accuracy Vs. Number of Cascades

1. *Network*: Scale-free Network of $N = 500$ nodes with $M = 2000$ edges
2. *Infection probabilities*: uniform random
3. *Incubation time model*: Power-law: $w(t) \sim t^{-2}$
4. Generated cascade sets of size 400-3500



Summary - More At Poster W89!

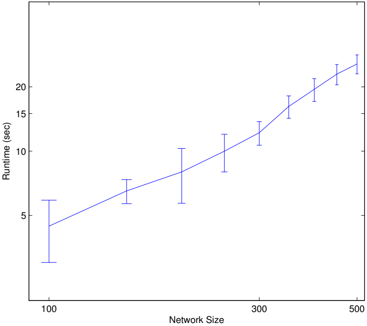
- We presented a scalable and robust algorithm for inferring social networks
  - 1000 node networks inferred inside of 10 minutes.
- Applications can include
  - Epidemiology - back tracing infection outbreaks
  - Viral marketing - identifying the biggest influencers
- Further study
  - Inferring missing nodes
  - More specialized cascade models
  - Methods to handle an unknown incubation model $w(t)$
  - Explore connections to inferring more general graphical models.

Implementation

- Likelihood was maximized using SNOPT7
- Nonlinear constraints slow it down
  - Faster to solve nonconvex problem
  - Results were plugged into KKT conditions of convex problem to confirm global optimality

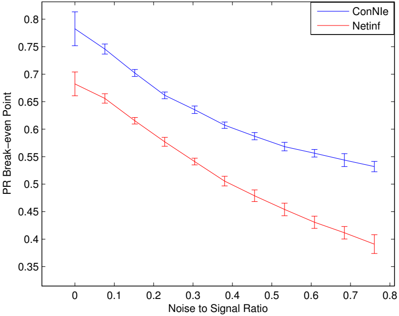
- We measured the runtime empirically.

- We can infer 1000 node networks inside of 10 minutes



Robustness to Error

- Incubation times were perturbed by i.i.d gaussian random variables
- The noise to signal ratio is the average perturbation over the average incubation time


ConNIe
Netinf
PR Break−even Point
Noise to Signal Ratio