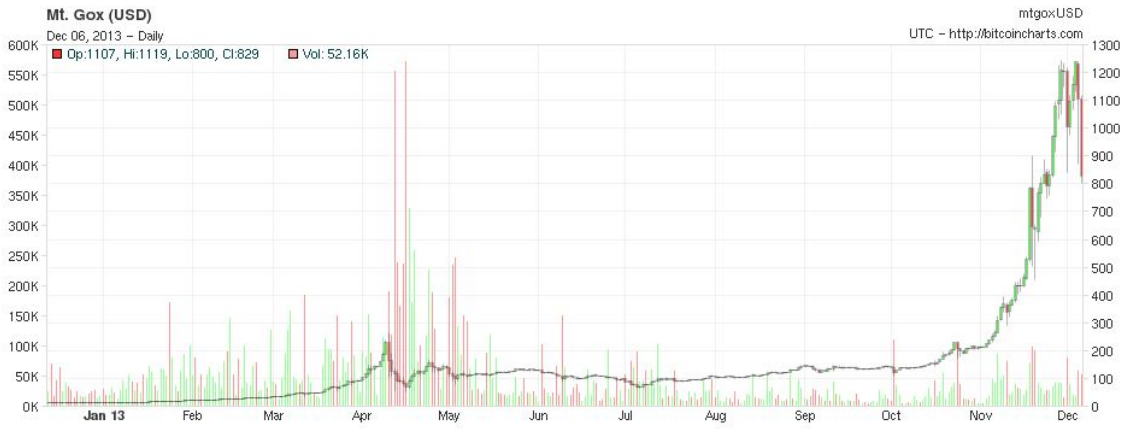
Estimating Latent Network and Identifying Influential Nodes for BitCoin Transactions

Group 11: Sidd Jagadish, Heerad Farkhoor, Christopher Wildman
Stanford University

Dec 10, 2013

1 Introduction

Since its inception in 2009, Bitcoin has been a first-of-its-kind, peer-to-peer means of transacting money between anonymous parties. The Bitcoin electronic currency is built on top of a distributed peer to peer system. To prevent double spending the record of all transactions is maintained on many nodes and is publicly available. The system uses proof of work as a means to prevent malicious users from manipulating the record. Anonymity is the most popular feature of Bitcoin, it is achieved through users generating public-private keys and digitally signing transactions. Unfortunately pure anonymity is rarely achieved as has been shown in An Analysis of Anonymity in the Bitcoin System. One interesting means of leaking user information is transactions that have multiple inputs (when a user does not have enough Bitcoins from a single previous transaction to serve as input to the next), public keys used in the two transactions to produce these inputs are assumed to be under the control of the same user. In this project we study users by using the Bitcoin Transaction Network Dataset from UIC which has already joined many public keys that appear to be under the control of a single user. Because of this novel transaction structure, little is known about the structure of the interaction network between Bitcoin users. Recently, the BitCoin has garnered much media attention because of volatility in the value of the digital currency. The value of the BitCoin against the US Dollar on December 6 has risen more than 1000% since August, and BitCoin suffered a drop of 50% in value in mid-April 2013. This is illustrated in the graph below:



Considering the exceptional volatility of this currency, as well as the increasing value of investment users are placing in BitCoin, a greater understanding of the underlying network behind BitCoin transactions would no doubt be useful to BitCoin investors. In this paper, we investigate whether we can model the actions of

buying and selling BitCoins (from BitCoin exchanges) as a cascading action and whether we can identify individual nodes who possess the power to drive deflation in the value in the BitCoin.

2 Prior Work

We use the Bitcoin transaction data set provided to us on the course web page. Unfortunately, this data set only describes, for each transaction, the time of the transaction, the value of the transaction, and an identification number for each party involved in the transaction. Intuitively, it is unlikely that people are (much) more likely to convert Bitcoins to dollars if their neighbors in the transaction network (people with whom theyve exchanged Bitcoins) convert their Bitcoins to dollars. As such, we need to convert the network given to us into a network that better represents the influence each node has on other nodes with respect to converting Bitcoins to dollars.

In order to do so, we leverage the techniques discussed in On the Convexity of Latent Social Network Inference [Myers et al., 2010]. In this paper, the authors discuss the construction of an "influence network." Given data about when nodes in a network are "infected" (say by a disease), but not by which other node each node is infected, the authors discuss a technique to construct an "influence network." Here, instead of "infection", we deal with trading BitCoins for dollars. A node is "infected" if that node trades BitCoins for dollars. We make major modifications to the model proposed by Myers et al., allowing for a node to be "infected" by multiple other nodes, and considering various "levels" of infection.

3 Finding the Exchanges within Bitcoin

We first sought out to identify the exchanges within the Bitcoin Transaction Network. We used the class dataset that has all transactions through April 7, 2013. Before looking directly at the transaction network we can analyze the users and their associated public keys. Assuming exchanges are well behaved and generate a new public key for each of their transactions they should have the largest number of public keys within the Bitcoin network. Also because exchanges are the main entrance and exit points to BitCoin they should have a very large number of transactions. Below in Figure 1 is a log log frequency distribution showing the number of users from the Bitcoin Transaction Network with a given number of public keys. Clearly the distribution follows a power law; many users only have one public key associated with them whereas there are a few users who have over 100,000 public keys, these are the users we are interested in.

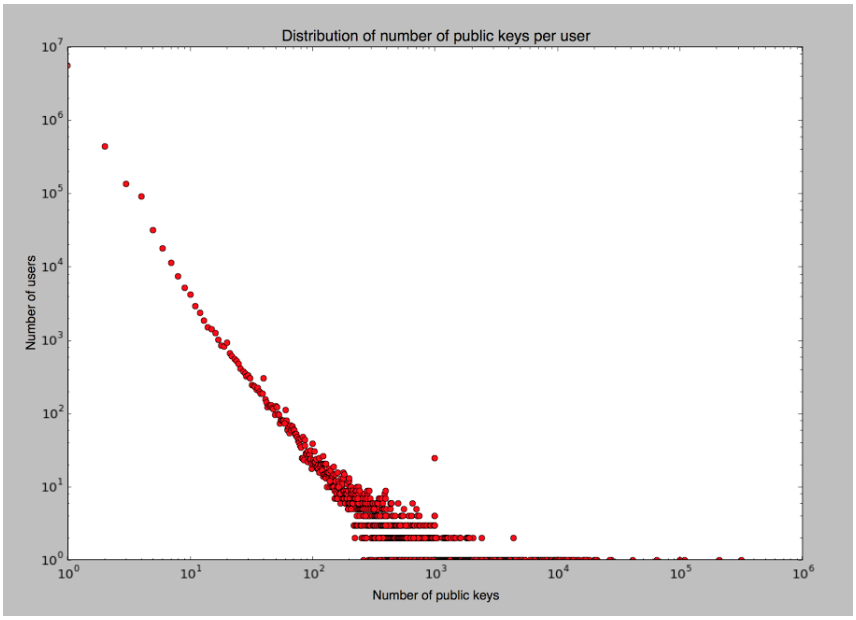


Figure 1: Distribution of public keys per user

Now moving to the Bitcoin Transaction Network transactions we can plot the degree distribution that represents the number of transactions of a user. Transactions are directed, there is the user bitcoins are coming from and the user the coins are going to. Below in Figure 2 we show the total degree distributions of the users as nodes within a transaction network. As expected the distribution follows a power law.

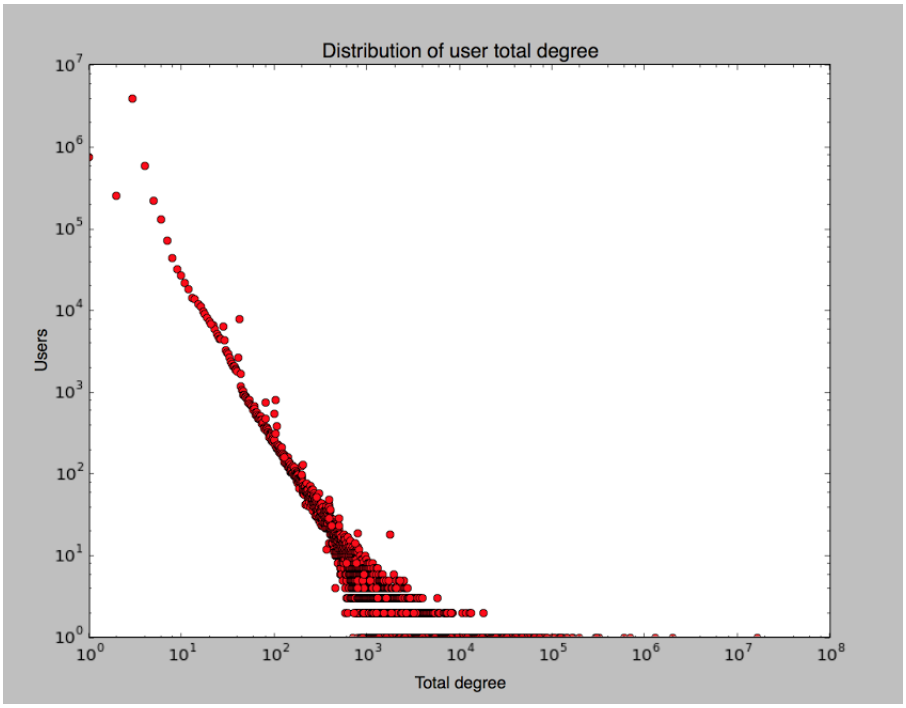


Figure 2: Distribution of user total degree

We are interested in the users on the far right of this distribution with a huge number of transactions, from the plot we can see there is a user with over 10 million transactions.

Here are some statistics we can use to help identify the exchanges within the Bitcoin network:

Total users: 6,336,769
Total edges/transactions: 37,450,461
Total value of transactions: 1,415,132,586.756584
Total public keys: 11,885,361

Looking at the degree distribution of transactions from the total degree plot above we decided to focus in on the nodes with roughly 300,000 transactions or more as these are the most likely to be exchanges or other very influential bitcoin users. There are eight users in the Bitcoin network of 6.3 million that have over 300K transactions. Below are statistics on each user, the number of transactions, the total value of their transactions and the number of public keys associated with them. We can see for most users there is a direct correlation between number of transactions and public keys, five of these nodes have the largest number of public keys. The other nodes have very few keys, this may be because it is not important for an exchange to remain anonymous.

User ID	# of Transactions	Public Keys	Value of Transactions (BitCoins)	Approximate Value of Transactions (BitCoins)
25	16,411,589	4	26,364,748.515	26.3M
1374	2,028,456	2	46,788,960.402367	46.7M
11	1,309,914	318,221	104,917,800.625803	104.9M
29	705,309	209,249	10,661,388.593724	10.6M
74	595,416	109,128	4,245,593.970460	4.2M
645014	324,094	3	761,122.514800	0.7M
12564	300,115	99,939	7,339,826.021520	7.3M
27	298,406	64,993	4,122,283.443577	4.1M

We can now look at what percent of the Bitcoin Transaction Network these eight users represent and realize their significance:

The total number of transactions in the largest eight nodes equals 58.67% of the total transactions.
The total value of transactions of the largest eight nodes is 14.5% of the total value of transactions.
The total public keys owned by the largest eight nodes is 6.7% of the total public keys in use.

Now that we have isolated what we think are the exchanges in the network we can search for transactions representing sell-ofs which will be our "infection times" used to infer the latent network.

4 Detailed Summary of Myers et al., 2010: Cascade Model, Maximum Likelihood Formulation and Convex Optimization

To infer a latent network from the infection times of other nodes [Myers et al. 2010] develop a cascade model for the spread of the infection in the inferred network. [Myers et al. 2010] detail their cascade model as follows. In a network with k unique nodes, for all nodes $i, j, (i, j = 1, ..., k), i \neq j$, the authors define the following quantities:

A : A [kxk] adjacency matrix
$A_{ij} = P(i \text{ infects } j | i \text{ is infected})$: Probability of of infecting neighbor j once a node i is infected
$w(t)$: Distribution of time it takes for node j to become infected when its neighbor is infected
c: A cascade initiated by randomly selecting a node to become infected at t=0
D: Set of all cascades c
τ_i^c: Time node i was infected during cascade c
$X_c(t)$: The set of all nodes that are in infected state at time t in cascade c

To clarify the meaning of $w(t)$, we write that if node i is infected and infects node j, the time of infection of j is given by $\tau_j = \tau_i + t$, where t comes from the distribution w. [Myers et al. 2010] suggests using a Weibull distribution for $w(t)$ ($w(t) = (k/\alpha)(t/\alpha)^{(k-1)} e^{-(t/\alpha)^k}$), however, they also mention the use of various other distributions, including a Power-Law distribution. The Weibull distribution has parameters k and α. [Myers et al. 2010] allow distinct parameters for each pair of nodes i and j, so $w_{ij} = (k_{ij}/\alpha_{ij})^{(k_{ij}-1)} e^{-(t/\alpha_{ij})^k_{ij}}$. Then, to solve for A_{ij}, k_{ij} and a_{ij} given the set of cascades D we write the likelihood of our data under this model, as follows:

$$L(A, k, \alpha | D) = \prod_{c \in D} \left[\left(\prod_{i; \tau_i^c < \infty} P(i \text{ infected at } \tau_i^c | X_c(\tau_i^c)) \right) \cdot \left(\prod_{i; \tau_i^c < \infty} P(i \text{ never infected } | X_c(t) \forall t) \right) \right] \quad (1)$$

From our discussion above, we know an equivalent expression is:

$$L(A, k, \alpha | D) = \prod_{c \in D} \left[\left(\prod_{i; \tau_i^c < \infty} (1 - \prod_{j; \tau_j < \tau_i} (1 - w(\tau_i^c - \tau_j^c) A_{ji})) \right) \cdot \left(\prod_{i; \tau_i^c = \infty} \prod_{\tau_j^c < \infty} (1 - A_{ji}) \right) \right] \quad (2)$$

In the first term above for every node i infected at time τ_i^c they calculate the probability that at least one other previously infected node could have infected this node. The second term calculates for every non infected node ($\tau_i^c = \infty$) the probability that no other node could have infected it. Using this equation the maximum likelihood estimate of A is a solution to $min_A - log(L(A, k, \alpha|D))$ where $0 \leq A_{ij} \leq 1$ for each i, j. Myers et al. go on to show that solving the $k(k-1)$ variables in this matrix can be broken down into subproblems and that each nodes incoming edges can be solved independently, this leaves k subproblems each with $(k-1)$ variables. Solving individually for the ith column of A can then be written as:

$$L(A_{:,i}, k_{:,i}, \alpha_{:,i}|D) = \prod_{c \in D; \tau_i^c < \infty} \left[1 - \prod_{j; \tau_j < \tau_i} (1 - w(\tau_i^c - \tau_j^c) A_{ji})\right] \cdot \prod_{c \in D; \tau_i^c = \infty} \left[\prod_{j \in c; \tau_j^c < \infty} (1 - A_{ji})\right] \quad (3)$$

Lastly Myers et. al observe that if j is never infected in the same cascade as node i, then the MLE of $A_{ji} = 0$ and therefore it can be excluded from the set of variables. For small cascades this eliminates a large number of variables as the infections are sparse.

Using a change of variables $B_{ji} = 1 - A_{ji}$, $\gamma_c = (1 - \prod_{j \in X_c(\tau_i^c)}(1 - w(\tau_i^c - \tau_j^c)A_{ji}))$, $\hat{B}_{ji} = log(B_{ji})$, and $c = (c)$, and take the negative logarithm of the expression to generate the convex optimization problem:

$$\min_{\hat{\gamma}_c, \hat{B}_{:,i}} \sum_{c \in D; \tau_i^c < \infty} -\hat{\gamma}_c - \sum_{c \in D; \tau_i^c = \infty} \sum_{j \in c; \tau_j^c < \infty} \hat{B}_{ji} \quad (4)$$

$$\begin{gathered} \text{subject to} \\ \hat{B}_{ji} \leq 0 \forall j \\ \hat{\gamma}_c \leq 0 \forall c \\ \log\left[\hat{\gamma}_c + \prod_{j; \tau_j^c < \tau_i^c} \left(1 - w_{ij}(\tau_i^c - \tau_j^c) + w_{ij}(\tau_i^c - \tau_j^c)\hat{B}_{ji}\right)\right] \leq 0 \end{gathered}$$

Finally, an L1-like regularization term $\varrho \sum_{j=1}^{N} \frac{1}{1 - A_{ji}}$ is added to preserve network sparsity. Plugging this convex program into a standard solver gives infection probabilities and infection time distributions for each pair of nodes in the Bitcoin network. For example, A_{ij} represents the probability that user js cashing out of Bitcoins influences user i to cash out as well.

5 Modifications to the Data-Generating Process

We now make a few vital observations about differences between our BitCoin problem and that considered by [Myers et al., 2010]. In their problem, each node could only be infected by one other node, whereas in our problem, each node can be infected by multiple nodes. Another even more important difference is that in [Myers et al., 2010], infection was binary, taking (0,1) values. In the BitCoin problem, "infection" is real-valued, representing the action of selling BitCoins to an exchange in return for dollars (or other traditional currency).

In order to accommodate these differences, we modify the data-generating process outlined by [Myers et al., 2010]. We maintain the existence of a matrix A_{ij}, but now we include as well a model of the level of infection of node j. For any node i, let v_i denote the level of infection of node i (a level of 0 means that i has not been infected yet). We model that $\hat{v_i} \sim N(\mu_i, \sigma^2 \mu_i^2)$, and $v_i = \max(\gamma \mu_i, \hat{v_i})$, where $\gamma \mu_i$ is the minimum amount of Bitcoin that i would sell in a sell-off. In addition, we hold a belief that the nodes i with higher values of μ_i will have higher values of A_{ij} for all j i.e. nodes that trade larger values will be more influential.

In addition to the above, we believe that in the case of BitCoin, the probability that node j is infected because node i has been infected always decreases in time, so we choose to use an exponential distribution $w(t) = \alpha e^{-\alpha t}$.

5.1 Some Observations about this Modified Data-Generating Process

Note that it may seem implicit in the above statements that whether or not node j is infected is unrelated to the level of infection of other nodes. However, this is not the case. We would think that if node i has a higher average level of infection, we would expect A_{ij} to be higher for all j. In addition, we note that the level of j's infection is unrelated to the level of node i's infection. The intuition behind this is that we believe that when people sell off their Bitcoins in cascades, they sell nearly all of their BitCoins, so the level of infection of node j is related only to node j's wealth. Note that the standard deviation of infection levels increases with average level of infection (linearly).

6 Heuristic Network Identification

Below, we describe a heuristic method of influence detection modeled on our modifications of the data-generating process described by [Myers et. al 2010].

Suppose the ith node, in the cth cascade (of a total of C cascades) is infected at time τ_i^c and has infection level v_i^c. Further, suppose that at some timestep, node j is the most recently infected node. Let I denote the set of all nodes currently infected, sans node j. Then we define our empirically found influence matrix as follows:

$$\forall i \notin I, \hat{A_{ij}} = 0 \tag{5}$$

$$\forall i \in I, c, \hat{A_{ij}^c} = \frac{\log(1+v_i^c)\exp(-\beta(\tau_j^c - \tau_i^c))}{W_j^c} \tag{6}$$

$$W_j^c = \sum_{i \in I} \log(1+v_i^c)\exp(-\beta(\tau_j^c - \tau_i^c)) \tag{7}$$

$$\forall i, j, \hat{A_{ij}} = \sum_c \hat{A_{ij}^c}/C \tag{8}$$

In the above equations 4 and 5, β is a parameter that determines the relative importance of recency of infection of node i to the size of the infection of node i. We can think of this as the rate at which the infection decays (a value of 0 would mean no decay at all, where as a high value would mean that recency is very important). W_j^c is a normalization constant so that for $\forall j, c$ such that $\tau_j^c > 0, \sum_i A_{ij}^c = 1$.

We also want to identify each user's effect on the BitCoin market as a whole. As such, we define node i's influence ϕ_i as follows:

$$\hat{\phi_i} = \sum_j \hat{A_{ij}}\hat{\mu_j} = \sum_j \hat{A_{ij}} \frac{\sum_c v_j^c}{\sum_c I[v_j^c > 0]} \tag{9}$$

This is a heuristic estimate for:

$$\phi_i = \sum_j A_{ij}\mu_j \tag{10}$$

6.1 Choice of β

As mentioned in the previous section, the choice of β will be extremely important to determining which nodes we mark as influential. Choosing $\beta = 0$ would lead to nodes influence being entirely dependent on the amount of currency they sell, and not at all on whether it seems that other nodes are responding, whereas a high choice of β would render the level of infection negligible.

To determine a suitable value of β, we first estimate α, the rate of the exponential distribution $w(t)$, then choose β based on that estimate.

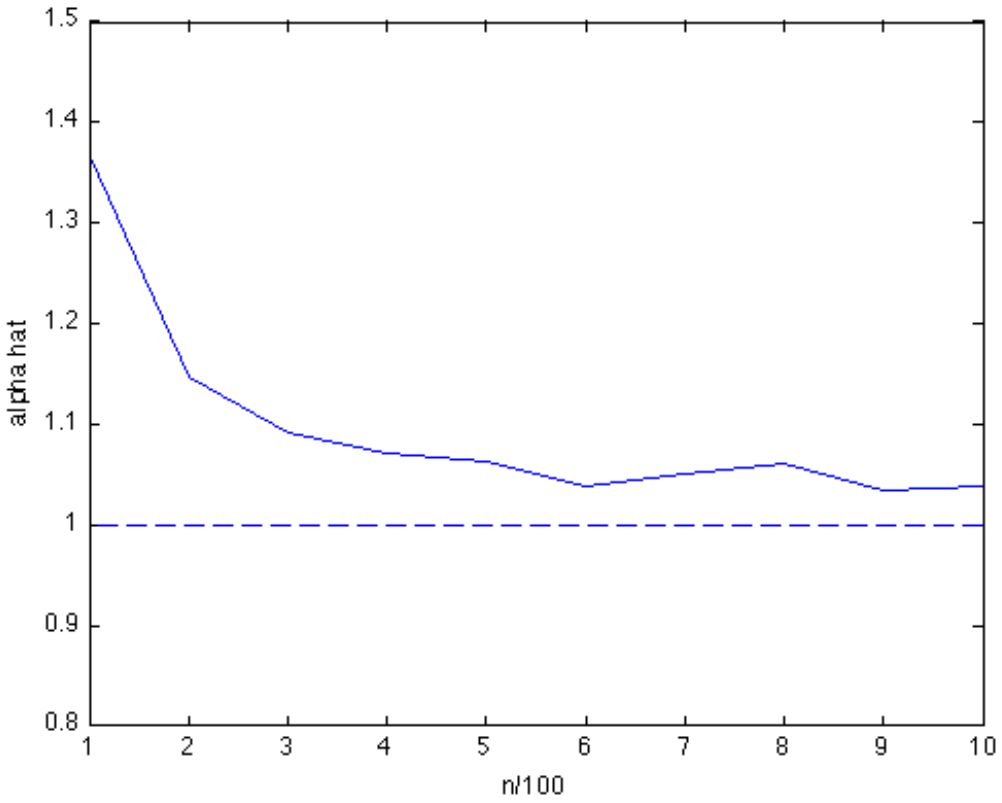
Given a sequence of infection times across several independent cascades, we note that the difference between two consecutive infection times does not necessarily correspond to one of the realized values of $w(t)$. If there are n nodes in the system, there are $O(n)$ nodes waiting for their infection timers to go off at a time.

Because of this, any estimate for the rate α of $w(t)$ based on the infection time data must be scaled up by a factor proportional to n.

As another heuristic we simply use

$$\hat{\alpha} = n\bar{\Delta}t \tag{11}$$

where $\bar{\Delta}t$ is the average difference between consecutive infection times across all cascades. Empirically this works well, and we see that the estimate $\hat{\alpha}$ approaches α as n approaches ∞. See the figure below, which shows the estimation procedure being applied when $\alpha = 1$.



Now that we have an estimate for α, and consequently an estimate of $w(t)$, we choose β so that if $\tau_j^c - \tau_i^c$ is large enough such that it is unlikely under our estimated $w(t)$, then we heavily down-weight $\hat{A_{ij}^c}$.

An exponential random variable with rate α is highly unlikely to take on a value larger than 3α, so we specify:

$$\exp() - \beta(3\alpha)) = 0.01 \tag{12}$$

$$\beta \approx 1.535/\hat{\alpha} \tag{13}$$

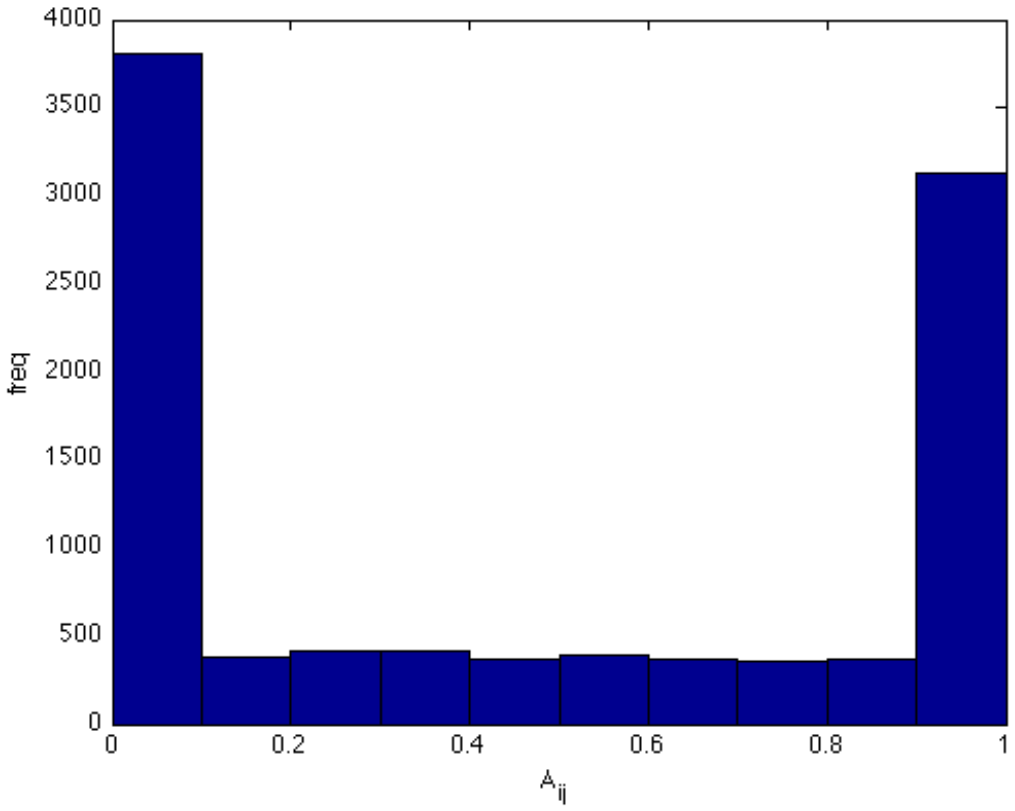
6.2 Testing the model

We test the model on a number of networks according to the modified data-generation process described previously. Parameters used:

- Number of nodes: $n = 100$
- Expected infection amount: $\mu_i \sim 10^{N(0,1)}$

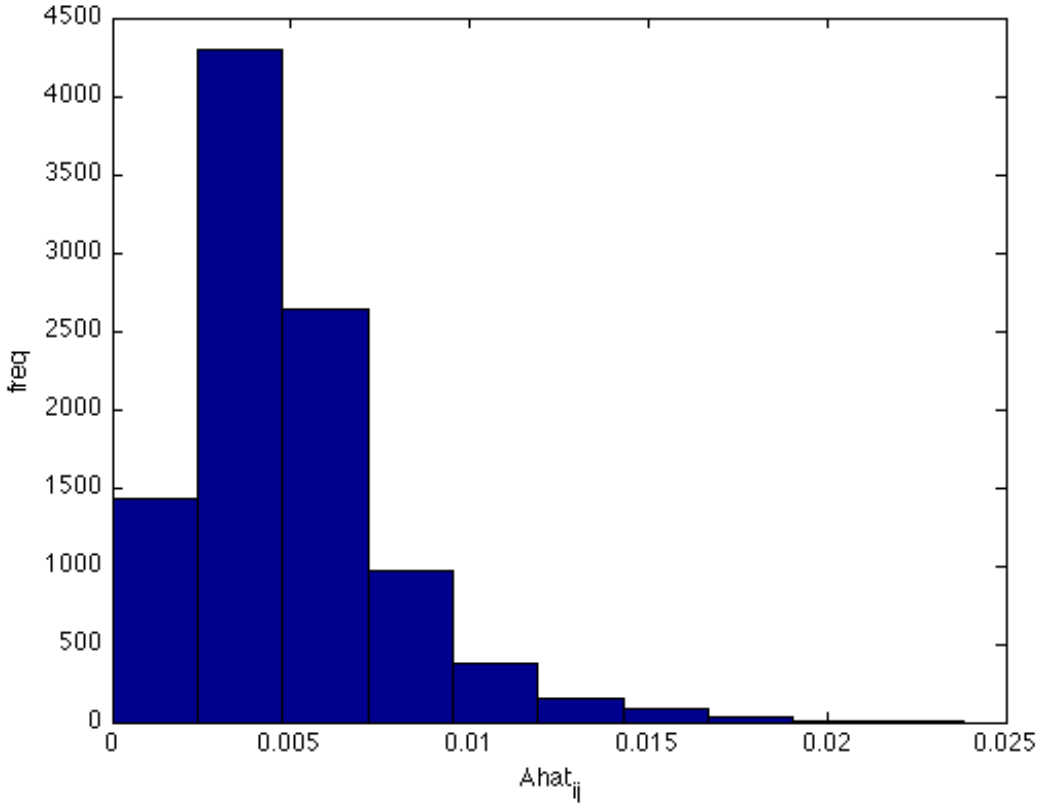
- Infection probabilities: $A_{ij} \sim N(\mu_i / \max(\mu), 1)$ truncated to $[0, 1]$
- Rate of $w(t)$: $\alpha = 1$
- Minimum fractional infection amount: $\gamma = 0.2$
- Scale of infection amount variance: $\sigma = 5$
- Number of infections per cascade: $l = 10, 20, \ldots, 90$
- Number of cascades: $C = 1, 10, 100, 1000$

We scale the mean infection probability for a node i by μ_i so that nodes with greater infection amounts are biased towards having more influence on other nodes, which is usually true in a real network. The figure below is an empirical distribution of A_{ij}.



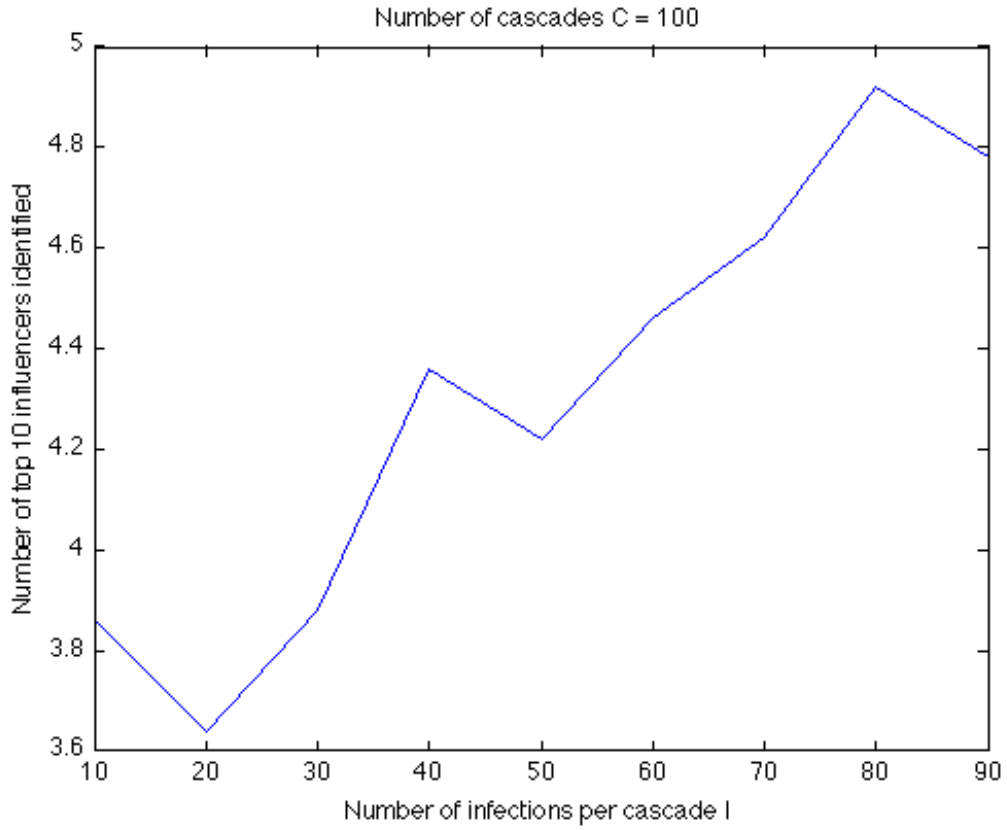
We see that the truncation creates a significant number of values at 0 (no influence from node i to j) and 1 (causal influence).

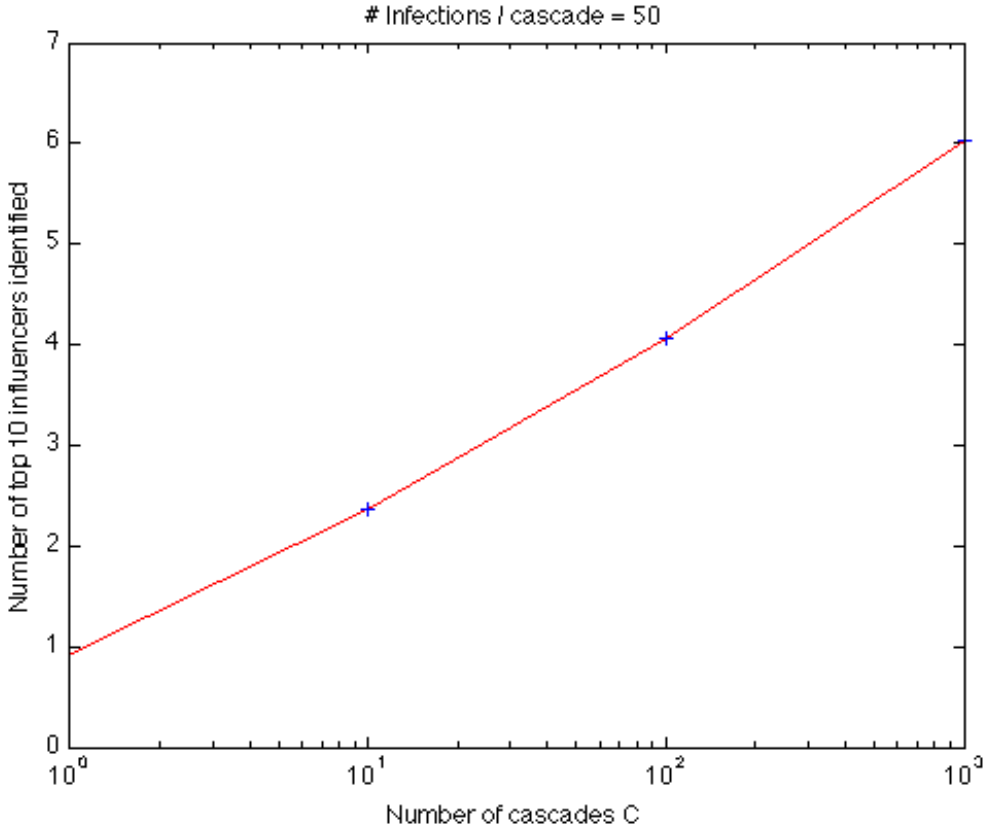
Unfortunately, the heuristic model clearly generates conservative estimates of $\hat{A_{ij}}$, with all values close to 0, and never any values near 1 (as shown in the figure below for $l = 50$ and $C = 100$).



Because of this, instead of using a matrix norm of $A - \hat{A}$ to judge the model's performance, we compared the overall influence values $\phi = A\mu$ to their estimates $\hat{\phi} = \hat{A}\hat{\mu}$. We then see how many of the top k (say $k = 10$) influencers ϕ_i appear in the top k estimated influencers with the model $\hat{\phi}_i$. I.e. compare the top k elements of $A\mu$ and $\hat{A}\hat{\mu}$.

The figures below show the average number of top-10 influencers correctly identified by the algorithm as a function of l and C.


Number of cascades C = 100
Number of top 10 influencers identified
Number of infections per cascade I
5
4.8
4.6
4.4
4.2
4
3.8
3.6
10
20
30
40
50
60
70
80
90


Infections / cascade = 50
Number of top 10 influencers identified
Number of cascades C
7
6
5
4
3
2
1
0
10^0
10^1
10^2
10^3

Clearly, in order to correctly identify half of the most influential nodes in a network, the number of infections per cascade and especially the number of cascades must be on the same order as the number of nodes. Otherwise, the signal is too weak to properly estimate influences.

7 Applying the model to the Bitcoin network

To be added in follow-up email to Bell (guanw@stanford.edu)

8 Bibliography

1. Myers et. al. "On the Convexity of Latent Social Network Inference." *CoRR* abs/1010.5504, 2010.
2. http://compbio.cs.uic.edu/data/bitcoin/, accessed December 2013.