

# Tutorial: Data-Driven Approaches towards Malicious Behavior Modeling



Meng Jiang  
University of Notre Dame



Srijan Kumar  
Stanford University



Christos Faloutsos  
Carnegie Mellon University



V.S. Subrahmanian  
University of Maryland, College Park

Tutorial link: <http://bit.ly/kdd2017>

# Outline

## Introduction

Feature-based algorithms

Bots

Sockpuppets

Vandals

Hoaxes

Spectral-based algorithms

Visualization: “spokes”, “blocks”, “staircases”

Camouflage

Theoretical guarantee

Density-based algorithms

Ill-gotten Likes

Synchronized Behaviors

Advertising campaigns

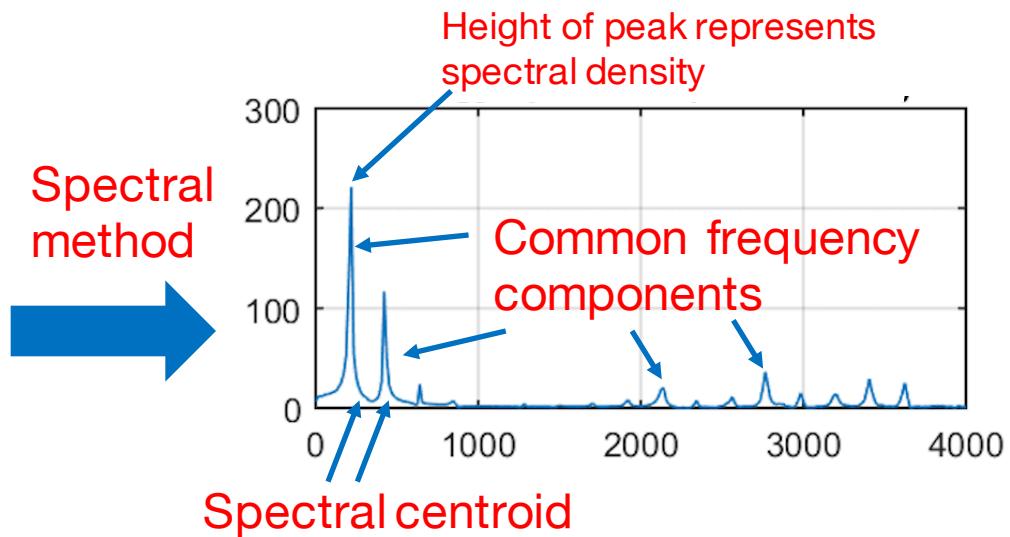
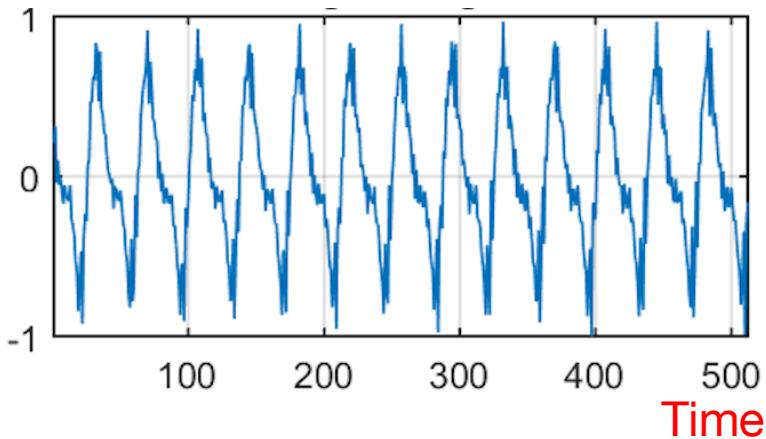
Social spam

Conclusions and future directions

Tutorial link: <http://bit.ly/kdd2017>

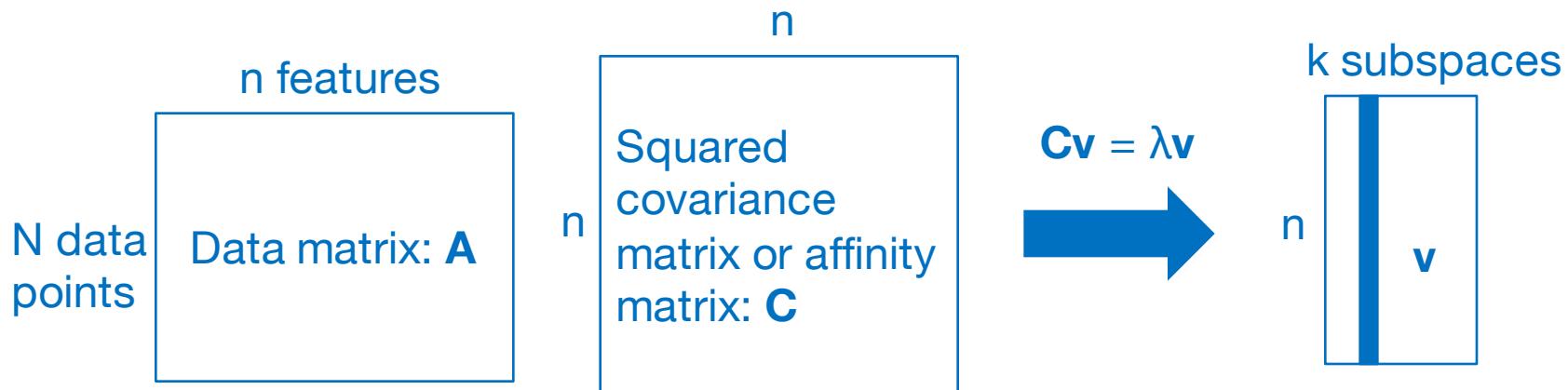
# Spectral features in speech signal processing

- **Temporal features** (*time* domain features): which are simple to extract and have easy physical interpretation
- **Spectral features** (*frequency* based features): which are obtained by converting the time based signal into the frequency domain using the Fourier Transform, like “frequency components”, “spectral centroid”, “spectral density”, etc.

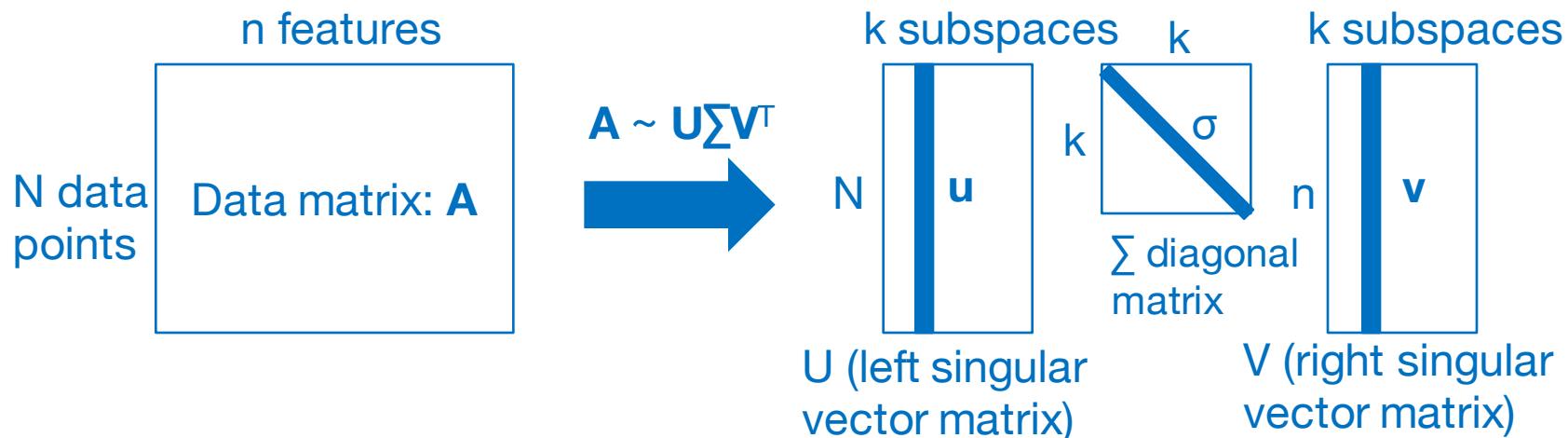


# How to get spectral subspaces?

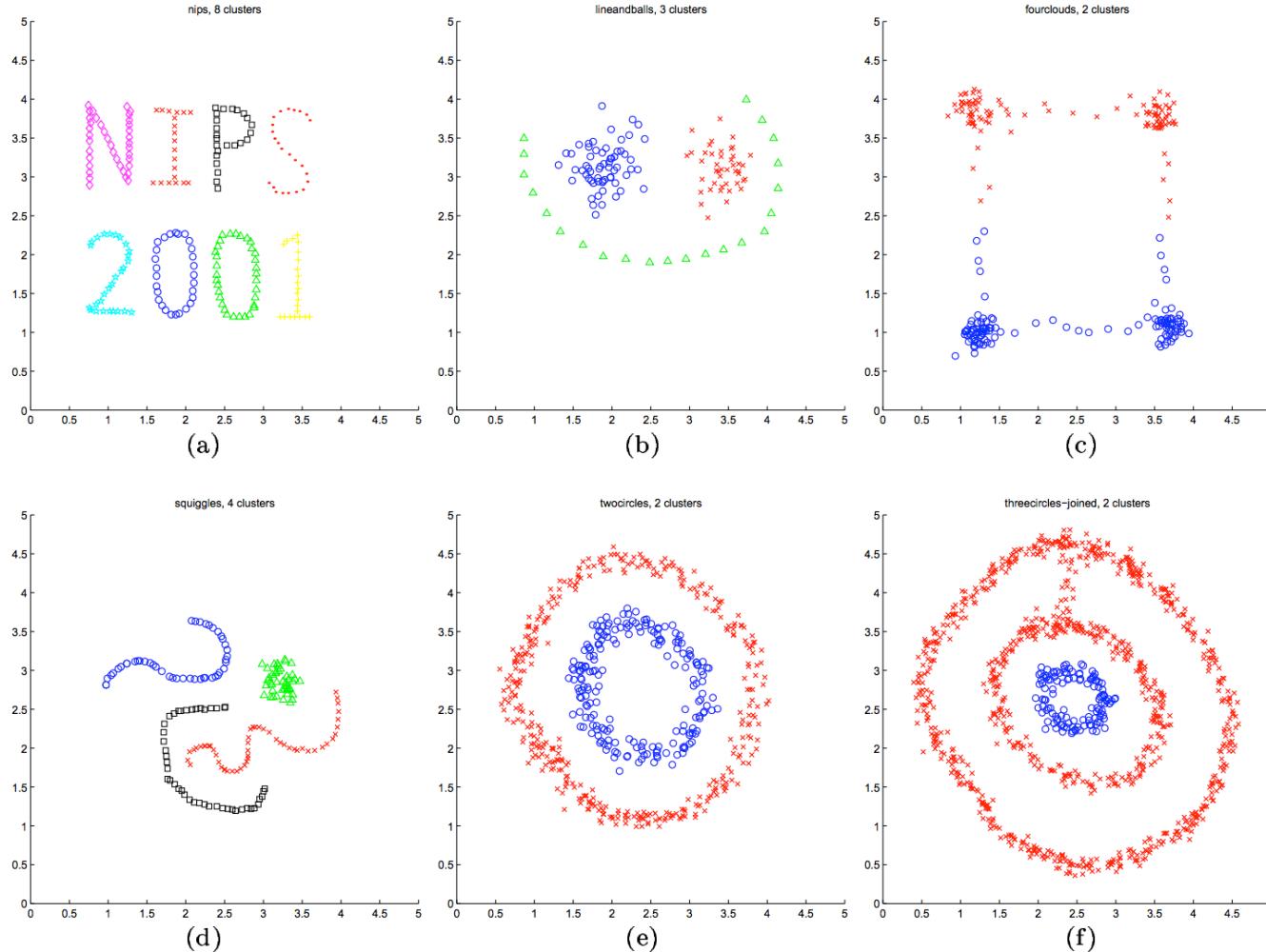
- Frequency components → Principal Component Analysis (PCA): Eigenvectors



- Other spectral decomposition methods. Singular Value Decomposition (SVD): Singular vectors



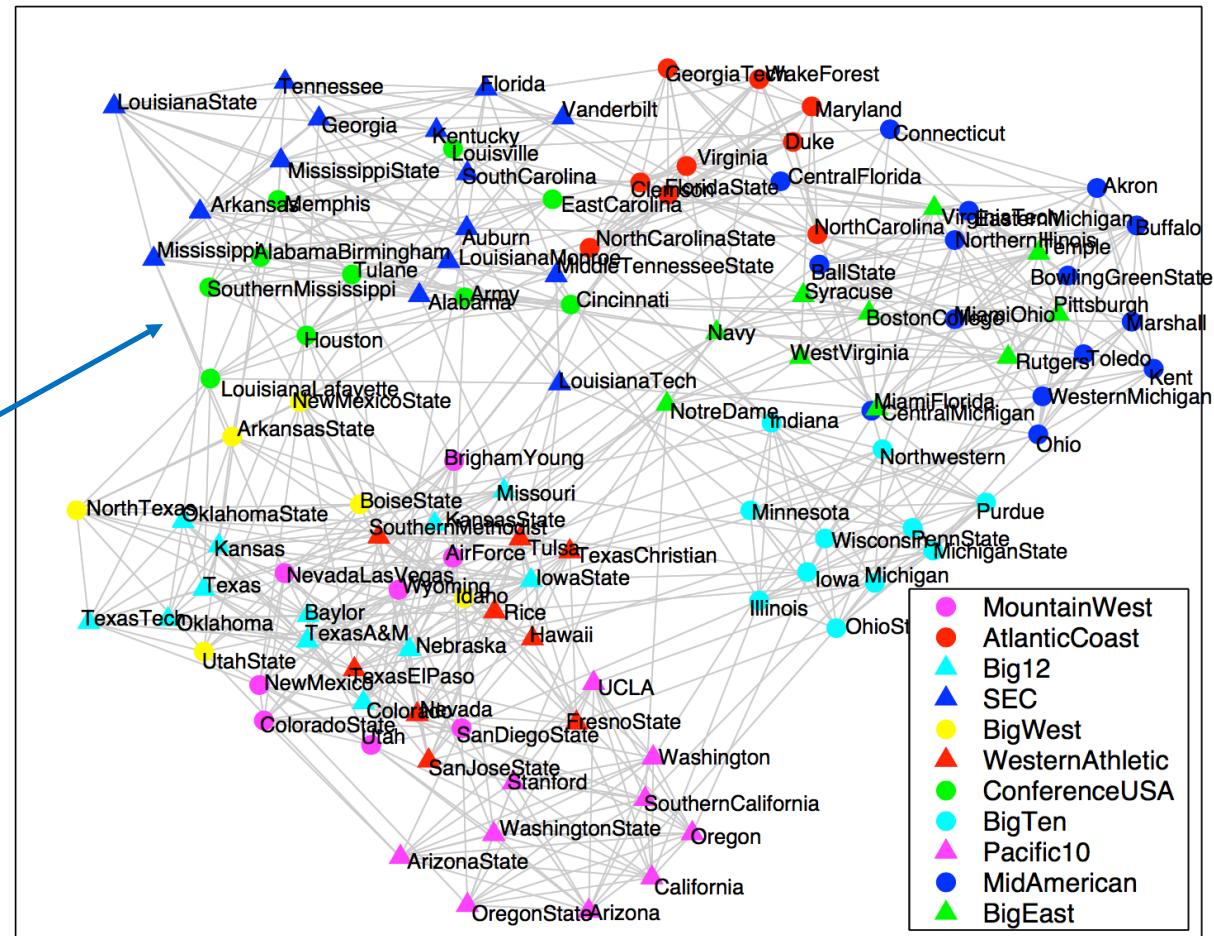
# Spectral clustering: Visualization



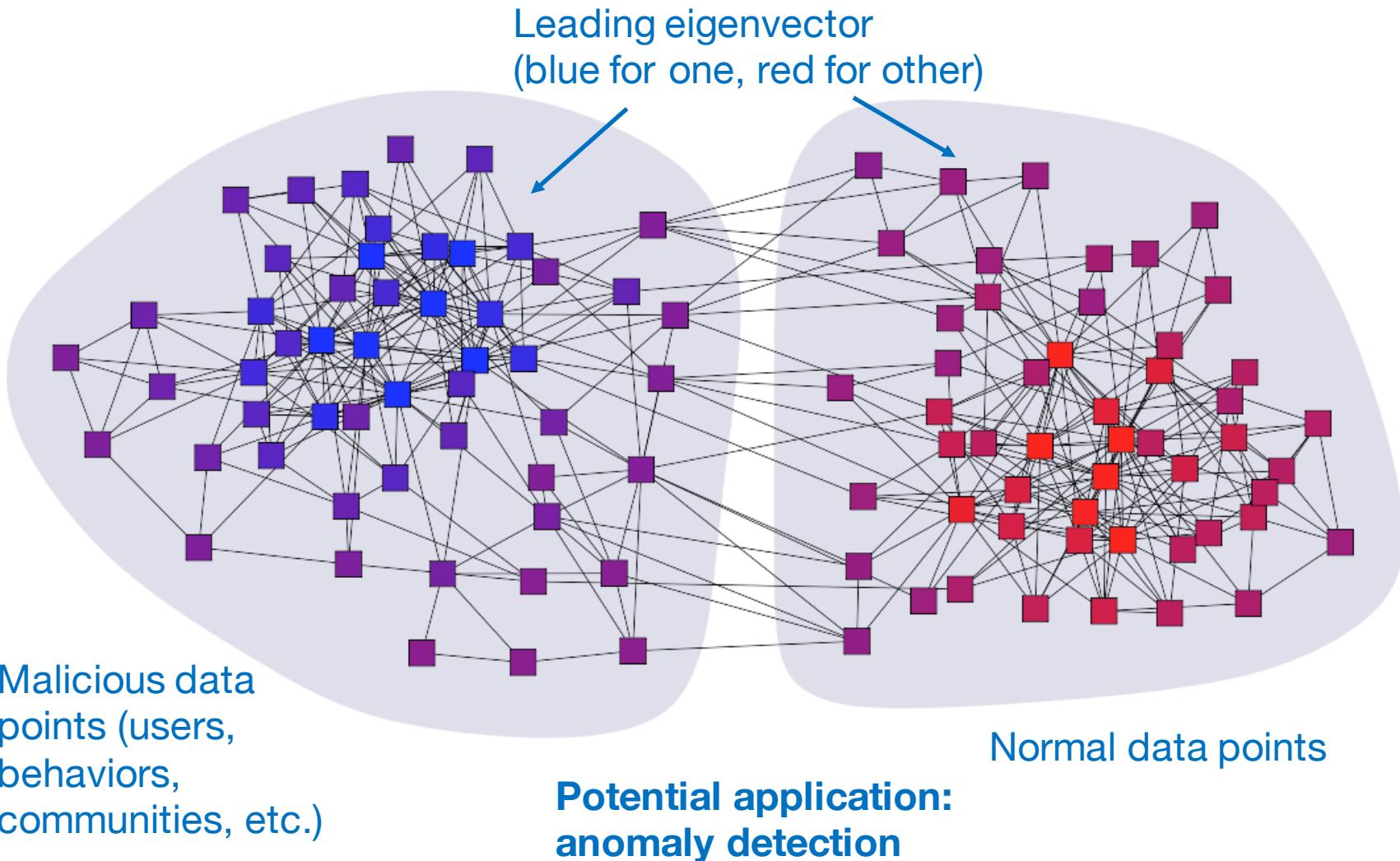
# Spectral methods for communities: Identification

Nodes are USA college football teams and edges represent which team played with which other team.

Communities represent groups of frequently co-playing teams.



# Spectral methods for communities: Applications



# Spectral-based methods

- Advantages
  - Visualization: tunable value of  $k$  = number of subspaces
  - Feature extraction by data distribution rather than manual or automatic selection
  - “Principal” components represent “leading” vectors
  - Data: Can easily work with  $N$ -by- $N$  graphs,  $N$ -by- $N$ -by- $N$  tensors
  - Applications: Finding communities and anomalies
- Disadvantages
  - Lack of interpretability of the subspaces/features

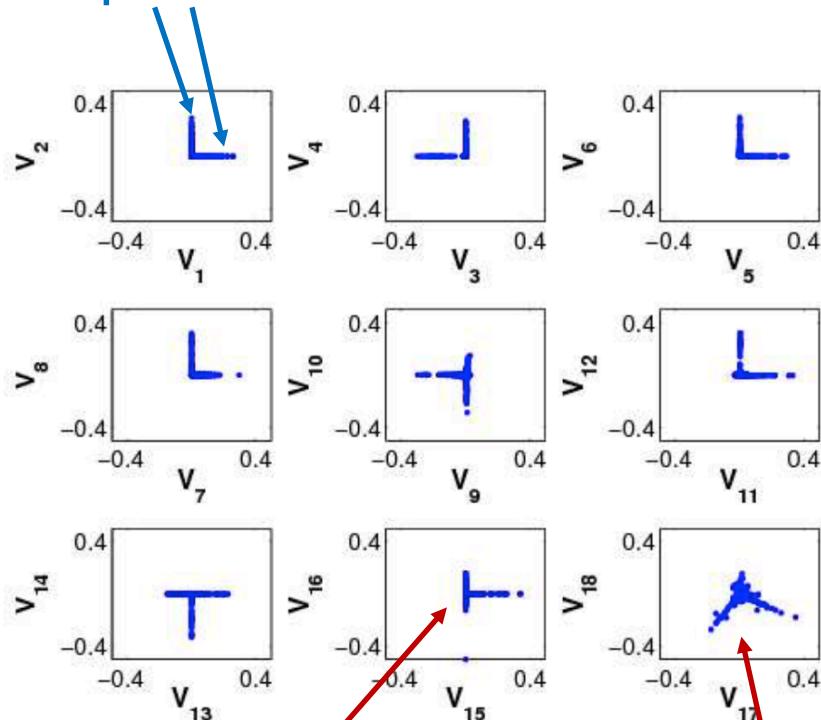
# Finding Surprising Patterns in Large Graphs

- **Problem definition:** Given a social graph based on mobile calls made from/to callers, find caller communities.
- Dataset: Activity over the duration of a month, 186,000 nodes and 464,000 edges.
- Key contribution: Discovery of the “spokes” phenomenon
  - **The singular vectors of the graph, when plotted against each other, often have clear separate lines, typically aligned with axes.**
  - Use EigenEigen (EE) plots to identify communities in the form of **cliques or near-cliques, perfect or near-perfect bipartite-cores**. (\*Malicious?)

# Spokes and Dense Cliques

V-V plots: Right spectral subspaces

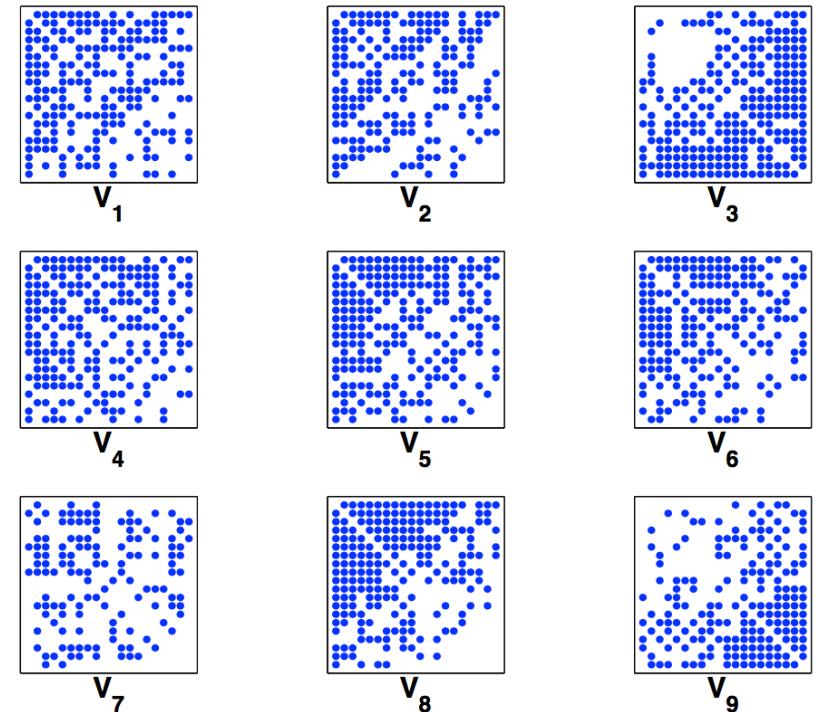
Spokes



(a) *EE-plot*

Long vs. Short?

Subgraphs of nodes identified as important by different V vectors: all are tightly knit



(b) Spy Plots of Sub-graph of Top 20 Nodes

Tilted?

- What is the meaning of spokes, elongated spokes, tilted spokes?
- Are there other patterns?
- Can these patterns be used to identify malicious behavior?

# Inferring Lockstep Behavior from Connectivity Patterns

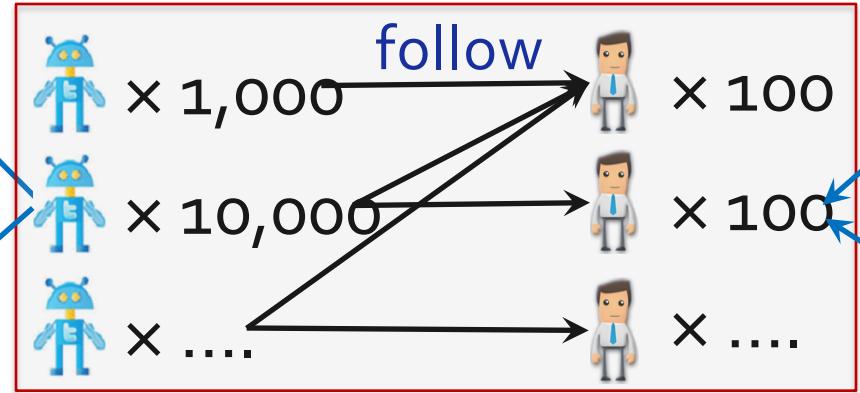
- Problem definition: Given a large graph, from **spectral subspace plots**, can we infer **lockstep behavior** patterns?



# Inferring Lockstep Behavior from Connectivity Patterns

- Problem definition: Given a large graph, from **spectral subspace plots**, can we infer **lockstep behavior patterns**?

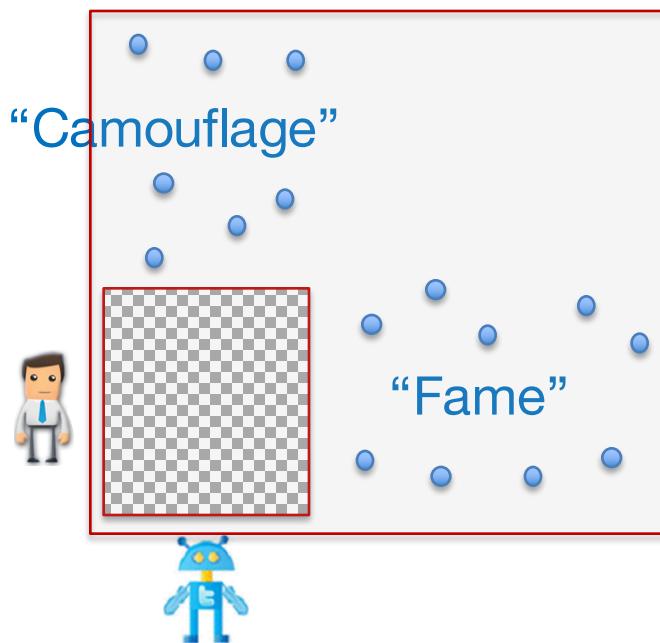
“Camouflage”: fraudsters follow other users to hide their behavior



“Fame”: popular users are followed by several users

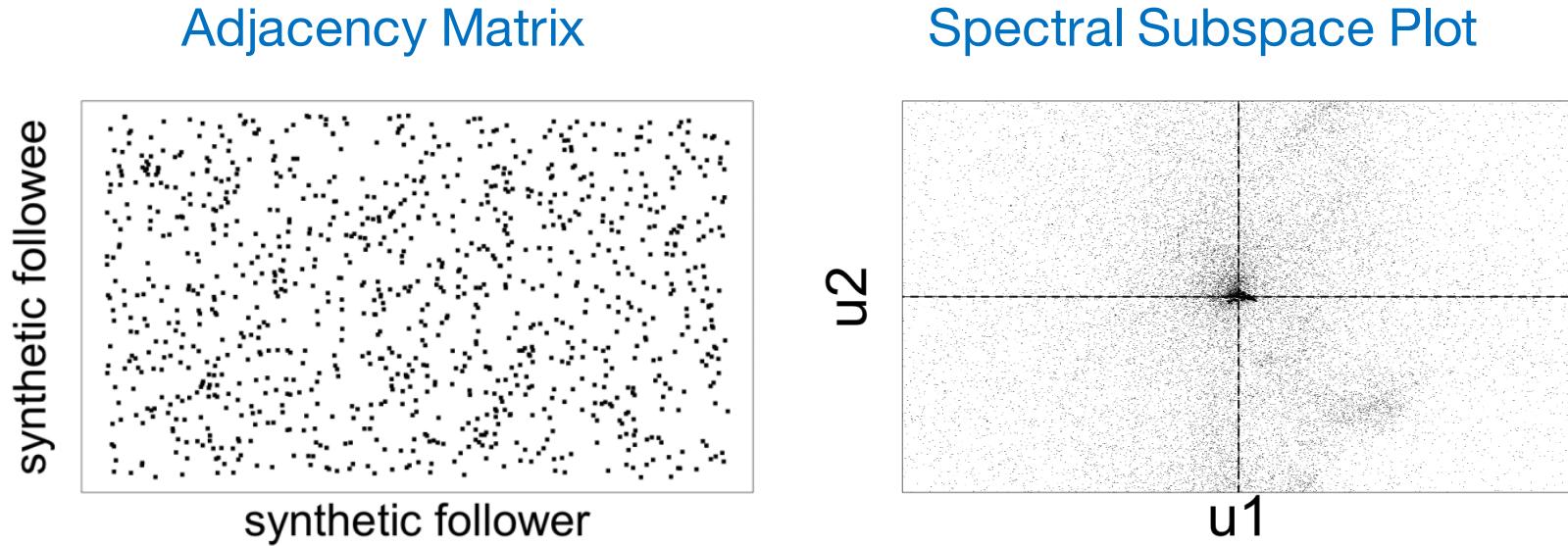
# Inferring Lockstep Behavior from Connectivity Patterns

- Problem definition: Given a large graph, from **spectral subspace plots**, can we infer **lockstep behavior** patterns?



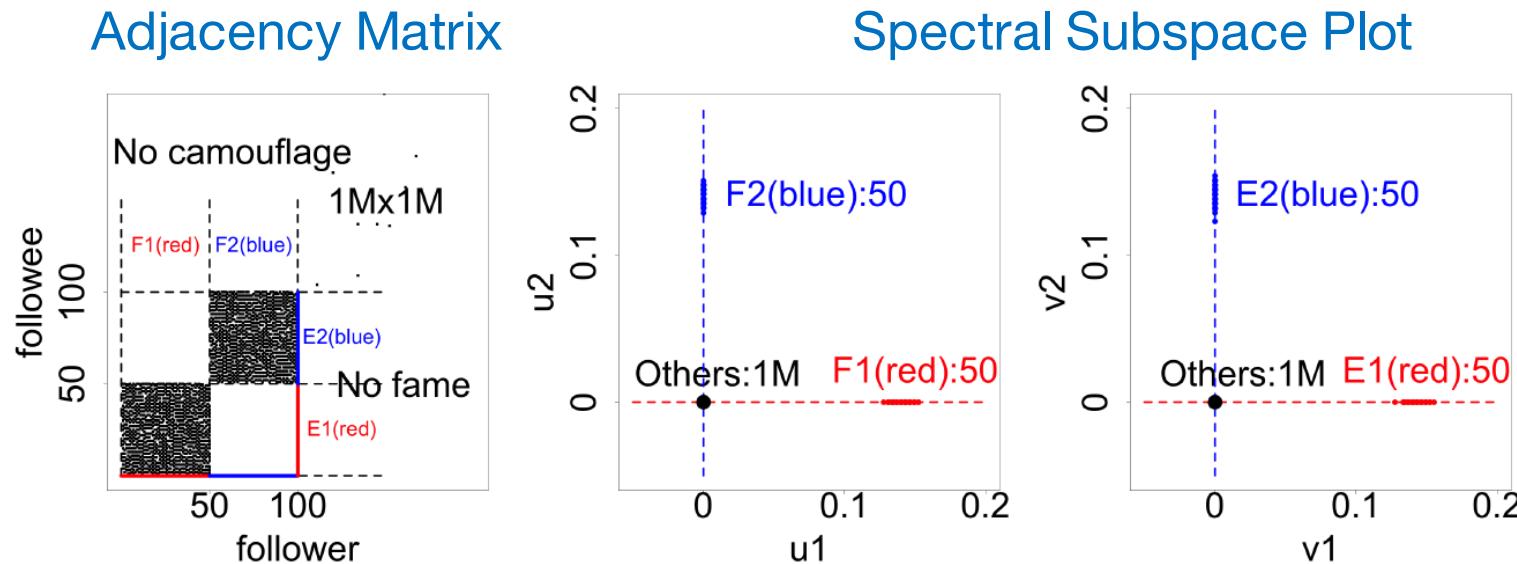
# Case 0: No lockstep behavior

- No blocks in adjacency matrix lead to scattering and no patterns in spectral subspace



# Case 1: Non-overlapping dense lockstep

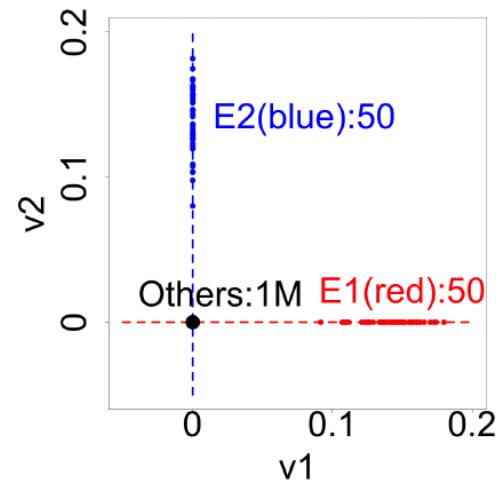
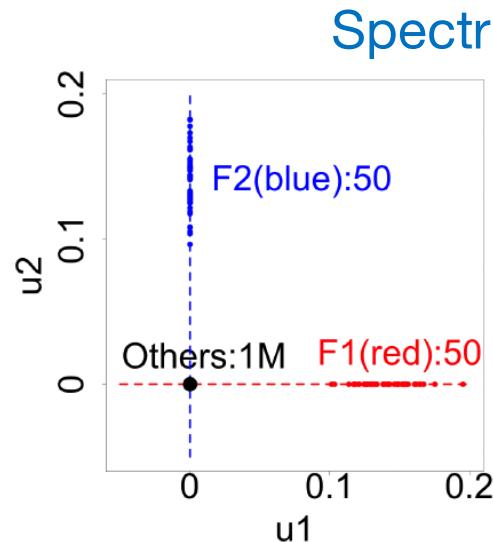
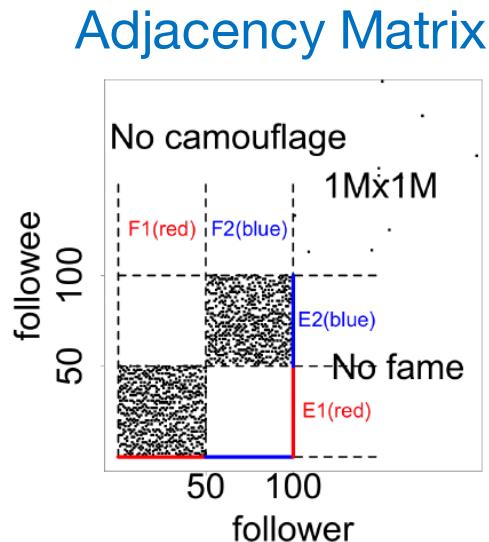
- Dense blocks in adjacency matrix generates “rays” in spectral subspace



Rule 1 (short “rays”): two blocks, high density (90%), no “camouflage”, no “fame”

## Case 2: Non-overlapping sparse lockstep

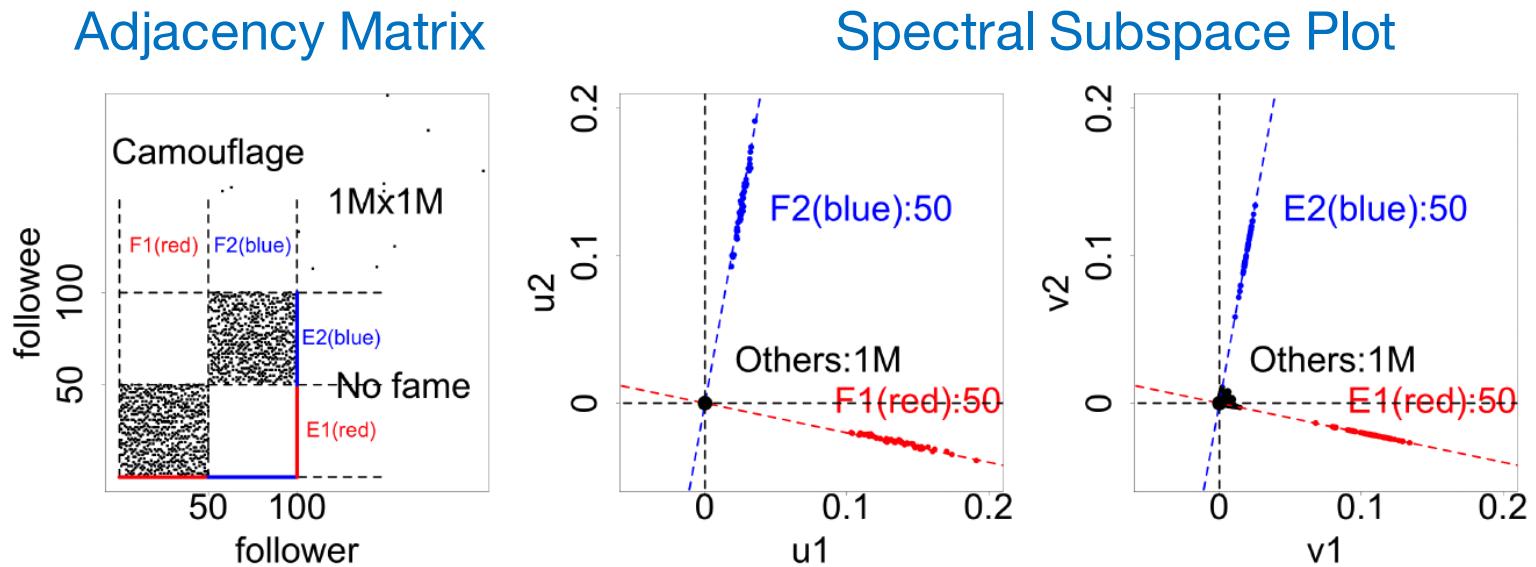
- Low density blocks in adjacency matrix leads to elongation of rays, indicating more varied behavior



Rule 2 (long “rays”): two blocks, low density (50%), no “camouflage”, no “fame”

# Case 3: Non-overlapping lockstep with outside edges

- Edges to or from blocks in adjacency matrix leads to tilting of rays in spectral subspace
- Edges going out of block: “camouflage” by fraudsters
- Edges into the block: “fame” edges to popular users

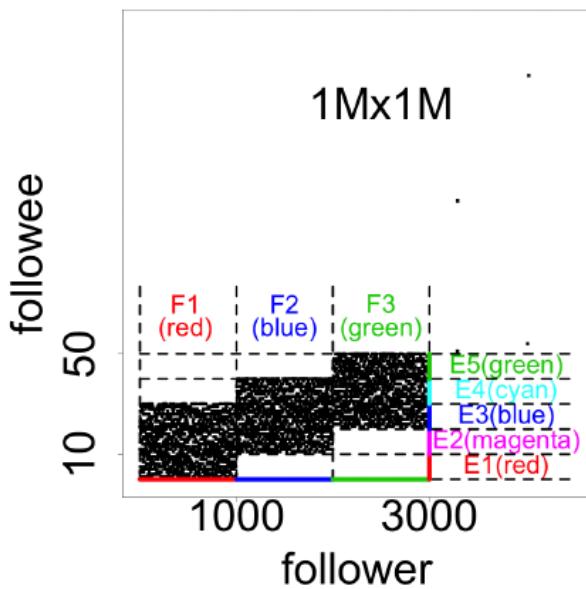


Rule 3 (tilting “rays”): two blocks, with “camouflage”, no “fame”

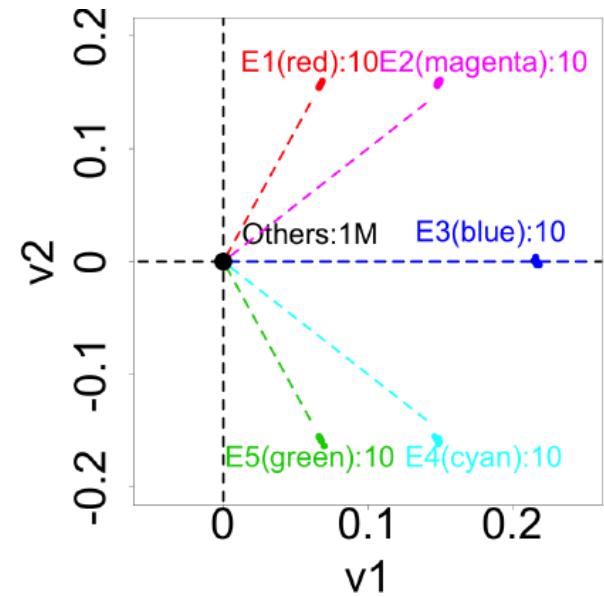
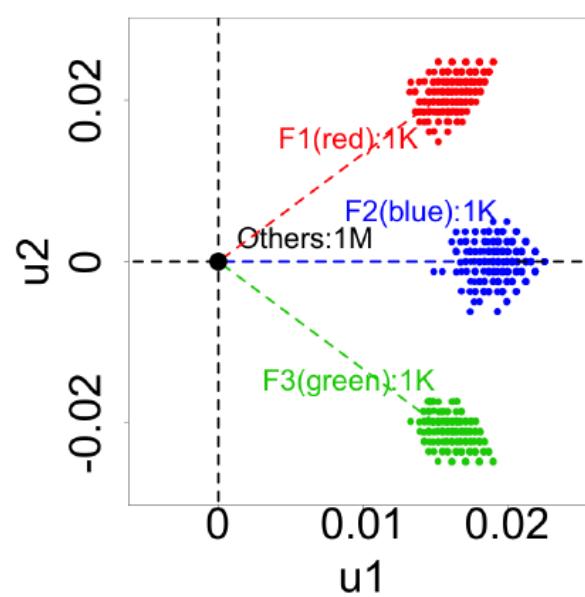
# Case 4: Overlapping lockstep

- “Staircase”, i.e. sequentially overlapping blocks, in adjacency matrix generates “Pearls” in spectral subspace

Adjacency Matrix  
with staircase



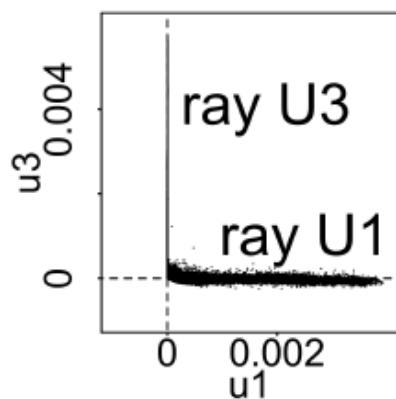
Spectral subspace plot  
showing pearls



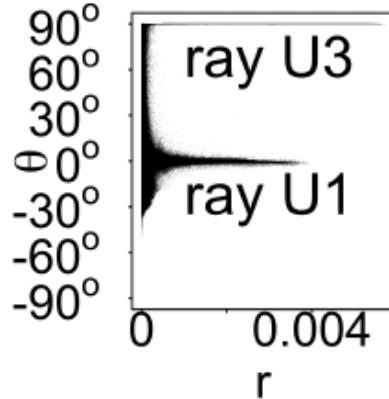
Rule 4 (“pearls”): a “staircase” of three partially overlapping blocks.

# LockInfer Algorithm: Reading Spectral Subspace Plots

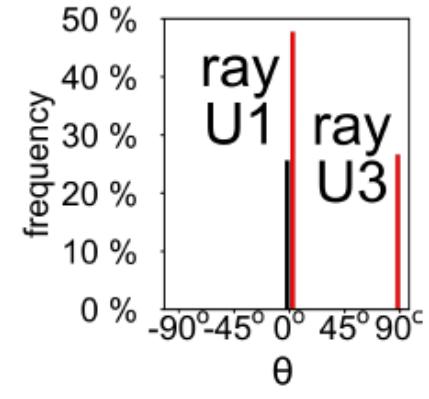
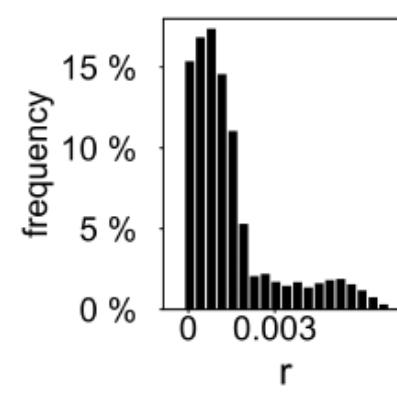
Spectral  
Subspace Plot



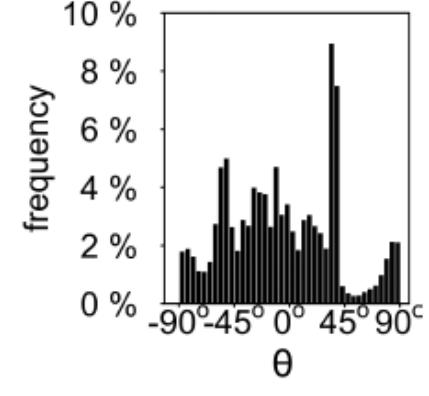
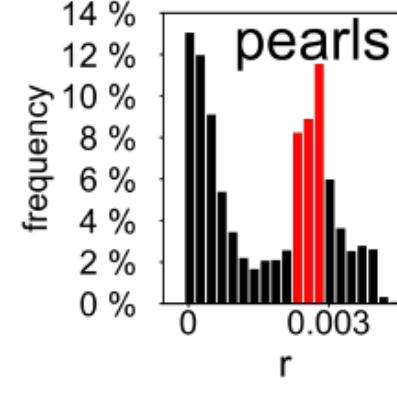
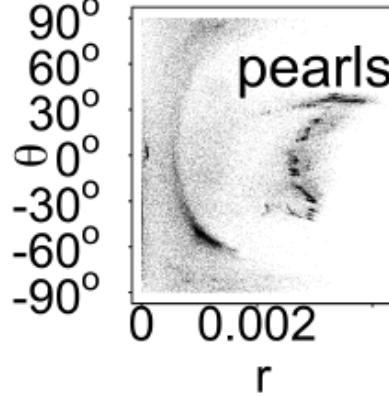
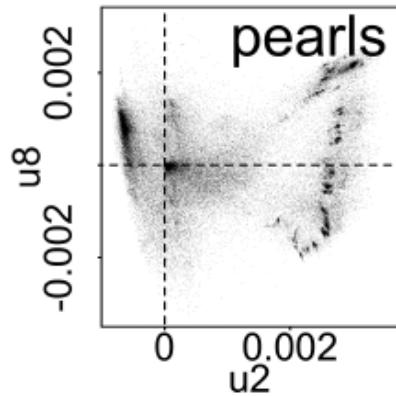
Polar Coordinate  
Transform



Histograms



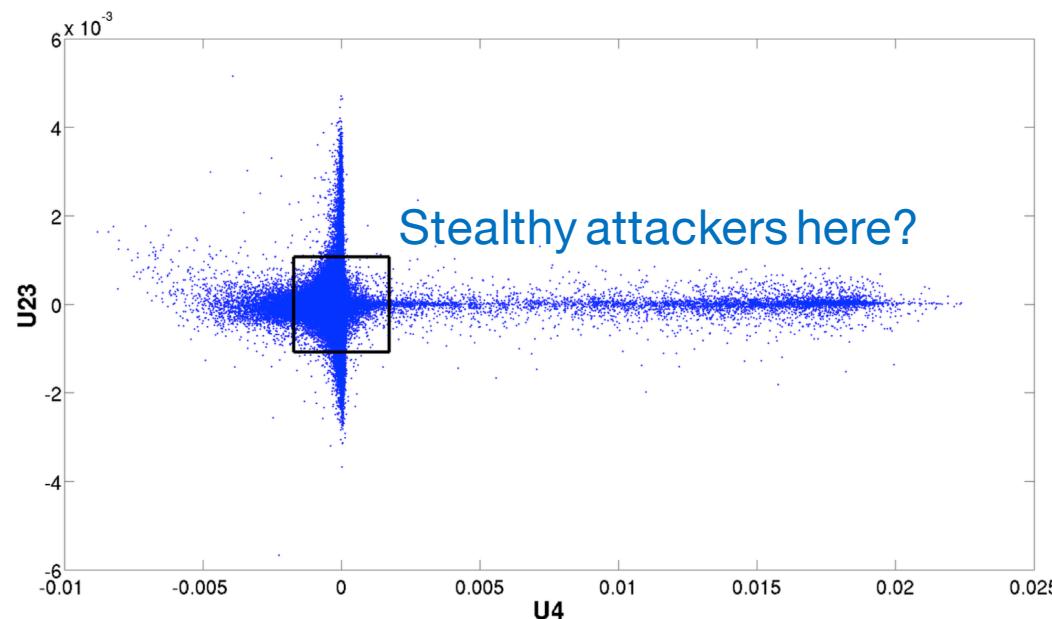
“rays” show two apparent spikes on  $\theta$  frequency at  $0^\circ$  and  $90^\circ$



“pearls” show a spike on  $r$  frequency at a much-greater-than-zero value 20

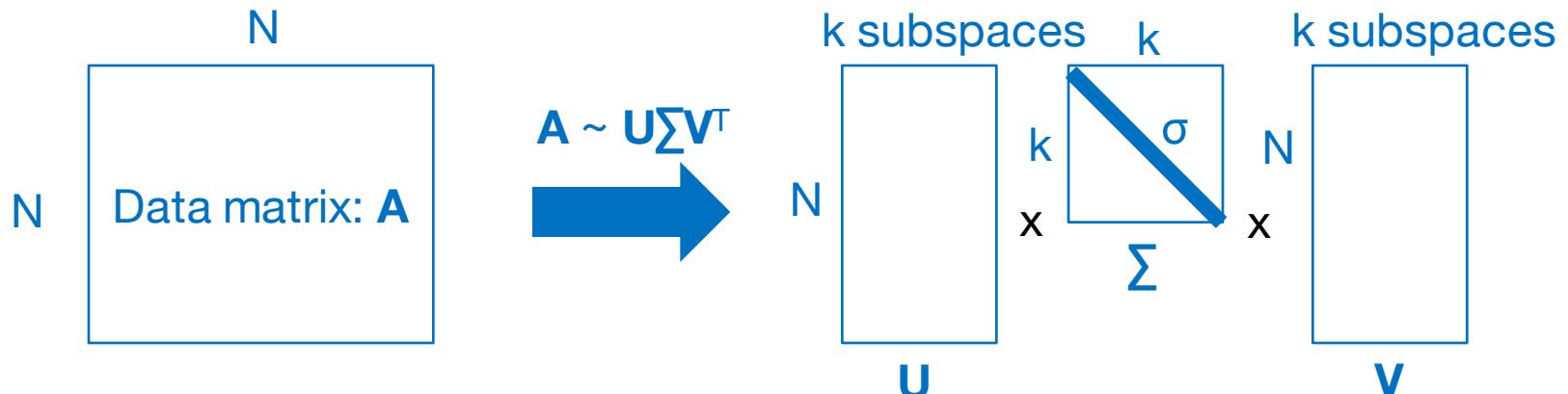
# Spotting Small-Scale, Stealthy Attacks

- **Problem definition:** Can we catch stealthy attacks that are missed by traditional spectral methods?
- Dataset: Twitter “who-follows-whom” social graph, 41.7 million nodes, 1.5 billion edges



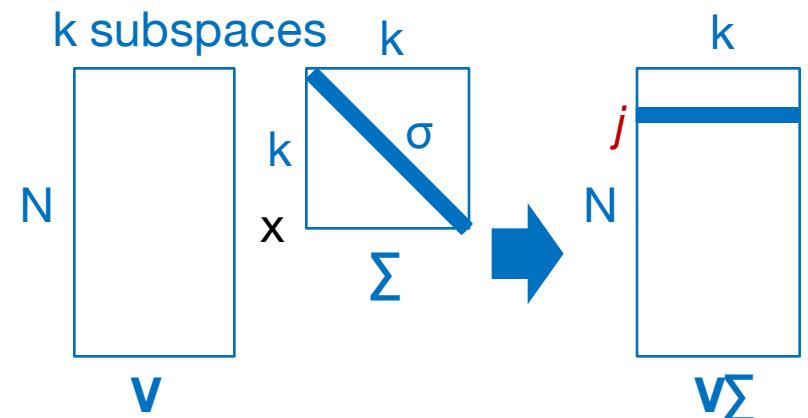
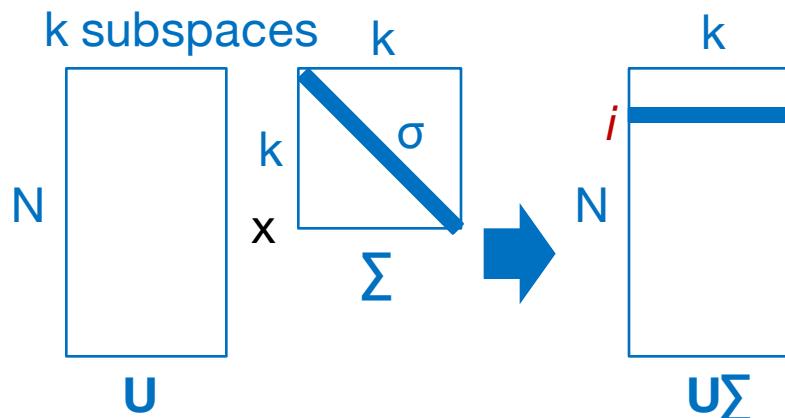
# fBox: Reconstructed Degrees

SVD



$$\text{Reconstructed out-degree}(i) = \|(\mathbf{U}\Sigma)_i\|_2$$

$$\text{Reconstructed in-degree}(j) = \|(\mathbf{V}\Sigma)_j\|_2$$



# Norm-Preserving Property of SVD

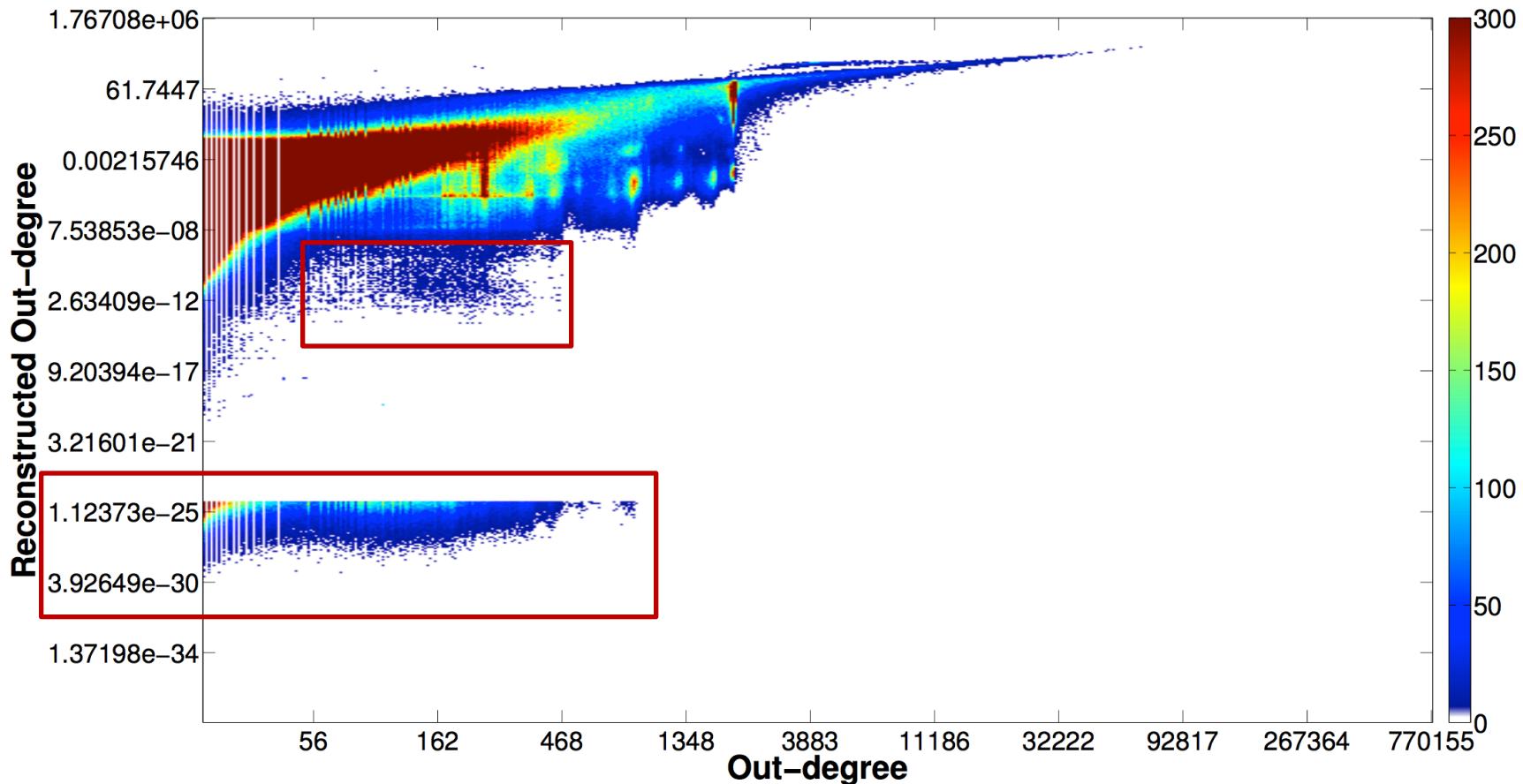
- The row vectors of a full rank decomposition and associated projection will retain the same  $L_2$  norm or vector length as in the original space:
  - For  $k = \text{rank}(\mathbf{A})$ ,  $\|\mathbf{A}_i\|_2 = \|(\mathbf{U}\Sigma)_i\|_2$  and  $\|\mathbf{A}^T j\|_2 = \|(\mathbf{V}\Sigma)_j\|_2$
- So, compare:
  - Reconstructed out-degree vs. real out-degree
  - Reconstructed in-degree vs. real in-degree

# Why does fBox work?

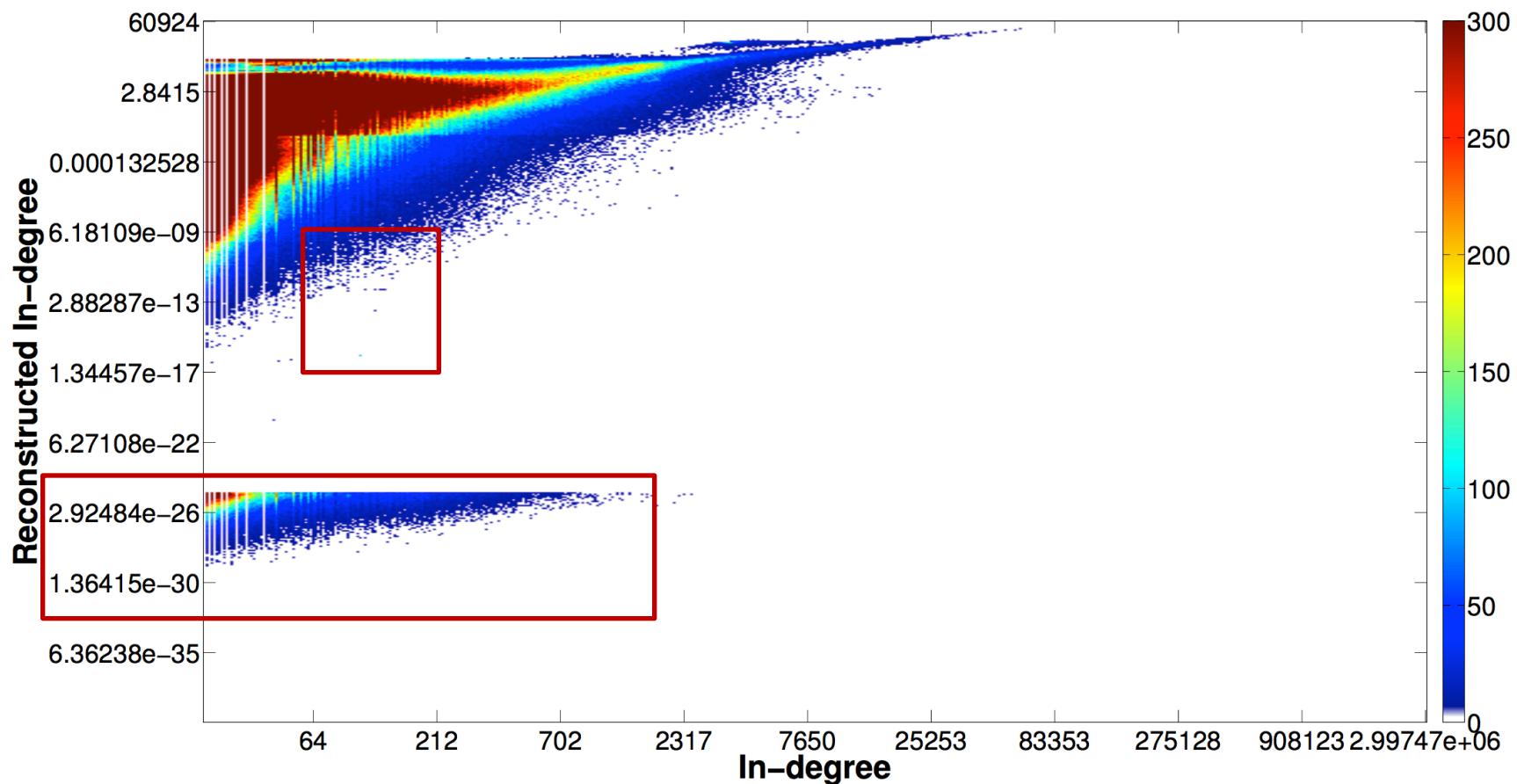
For  $k < \text{rank}(A)$ , dishonest users' reconstruction is poor compared to that of honest users.

- Dishonest users who either form isolated components or link to dishonest objects will project poorly and have characteristically low reconstruction degrees
- Honest users who are well-connected to real products and brands should project more strongly and have characteristically higher reconstruction degrees

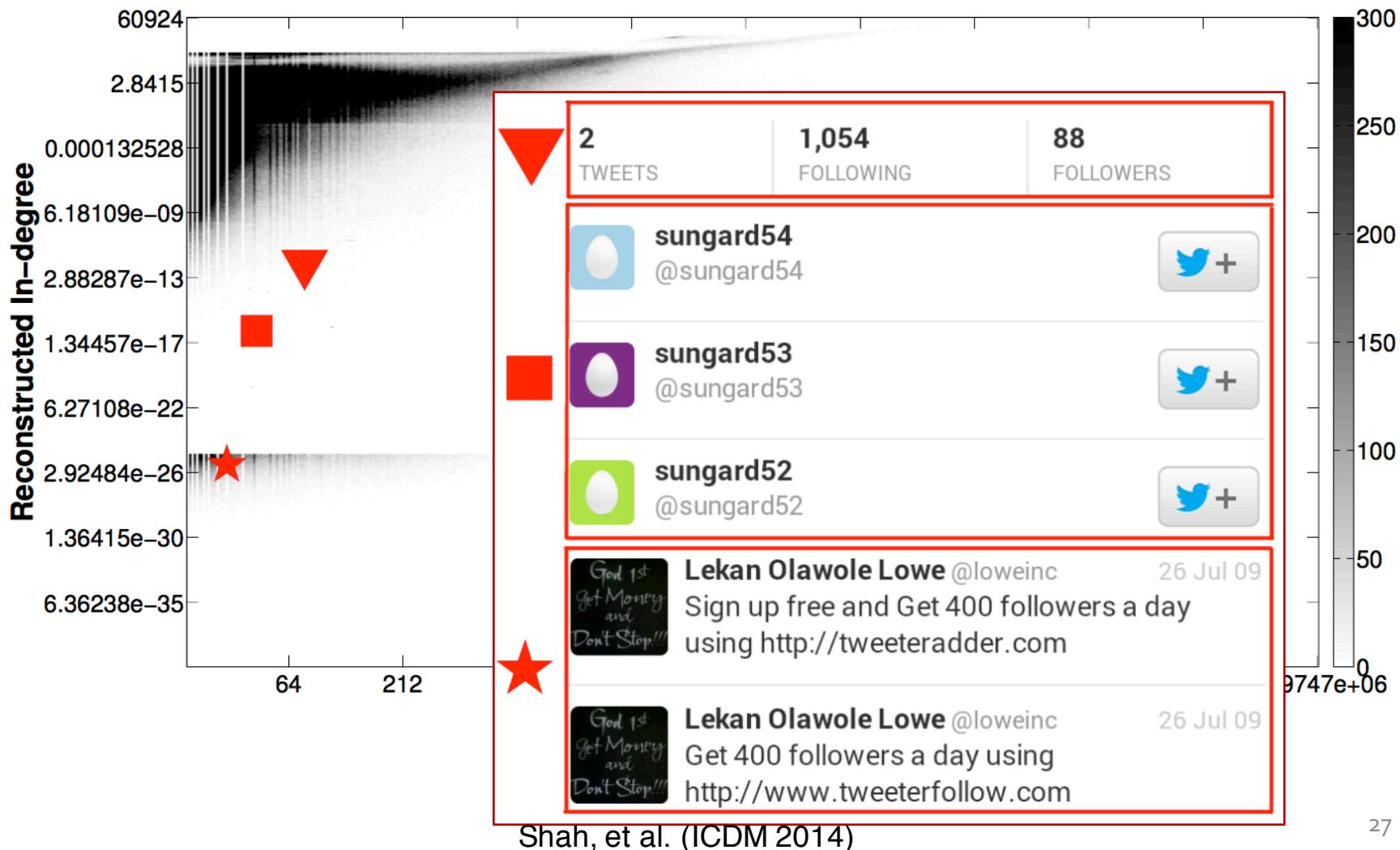
# Reconstructed out-degree vs. Real out-degree



# Reconstructed in-degree vs. Real in-degree



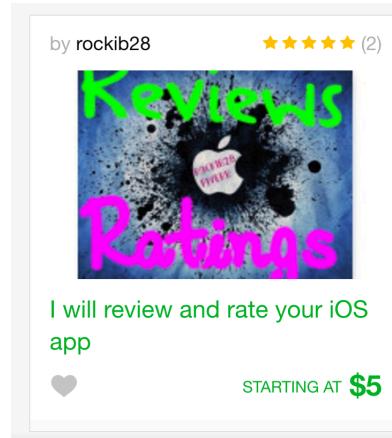
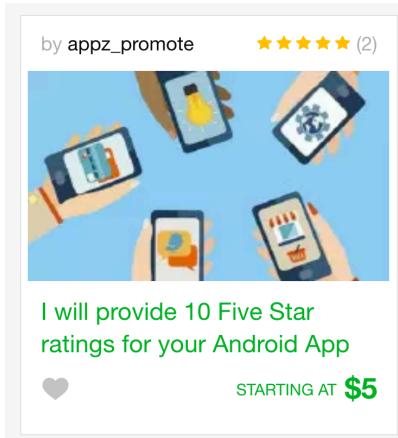
# Reconstructed in-degree vs. Real in-degree (cont.)



# Bounding Graph Fraud in Camouflage

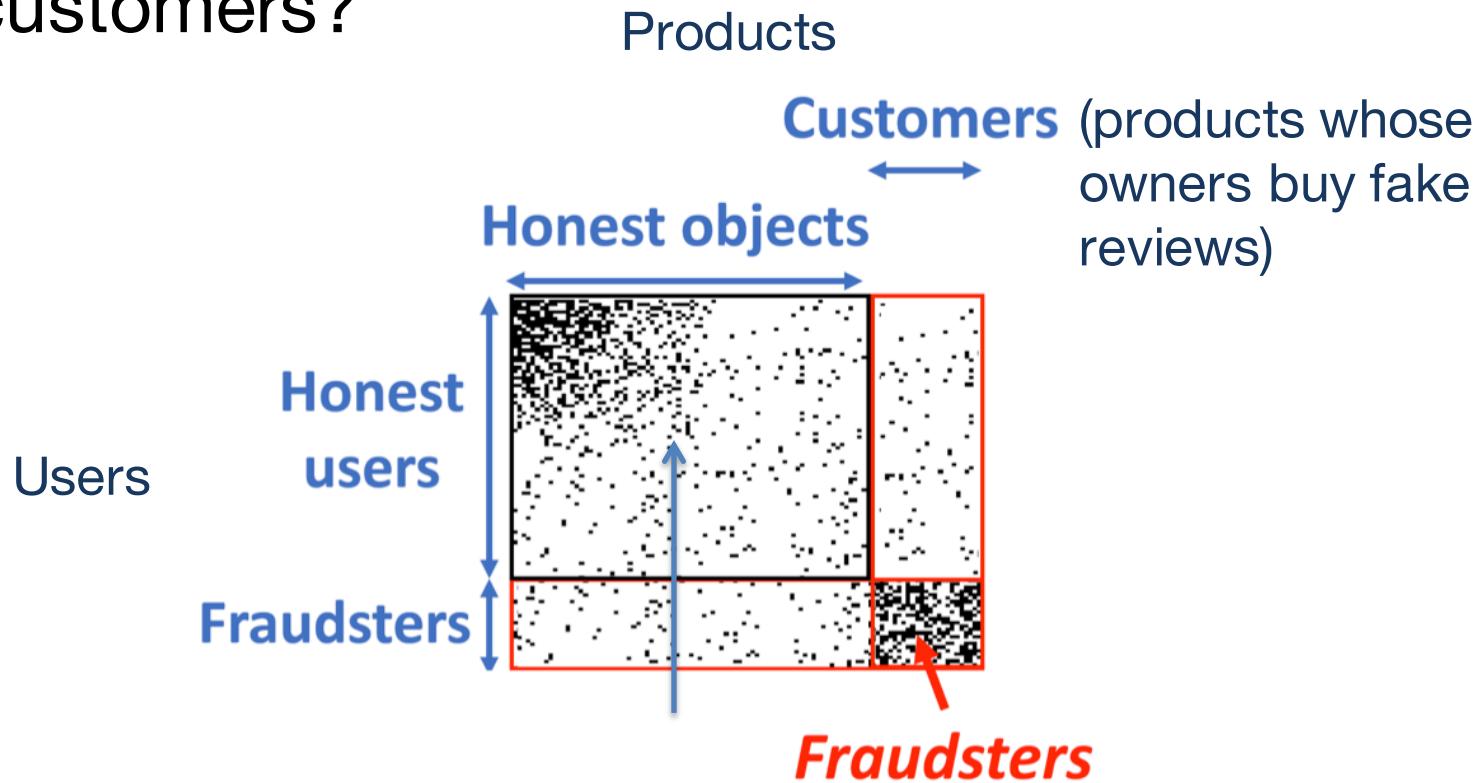
- An application: Fake reviews

**I will do 5 five star reviews, all from real profiles**



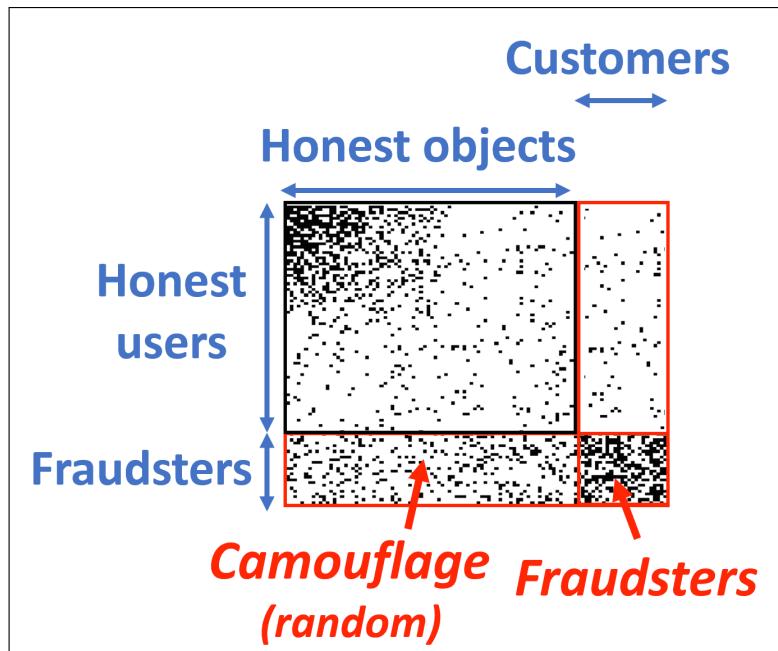
# “User-Product” Review Graph

- Problem definition: Given a “user-product” review graph, can we spot fraudsters and customers?

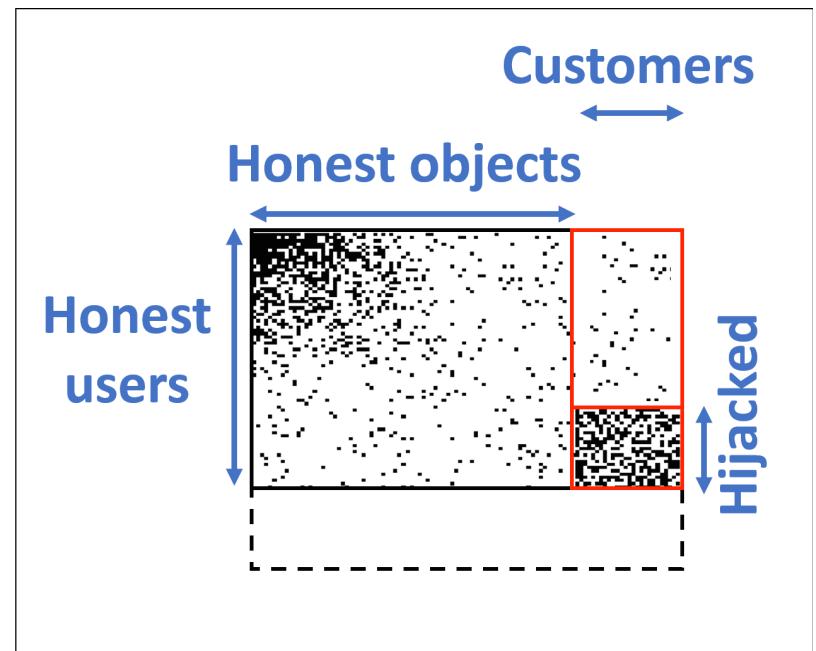


# Camouflage: Evading Detection

Random camouflage

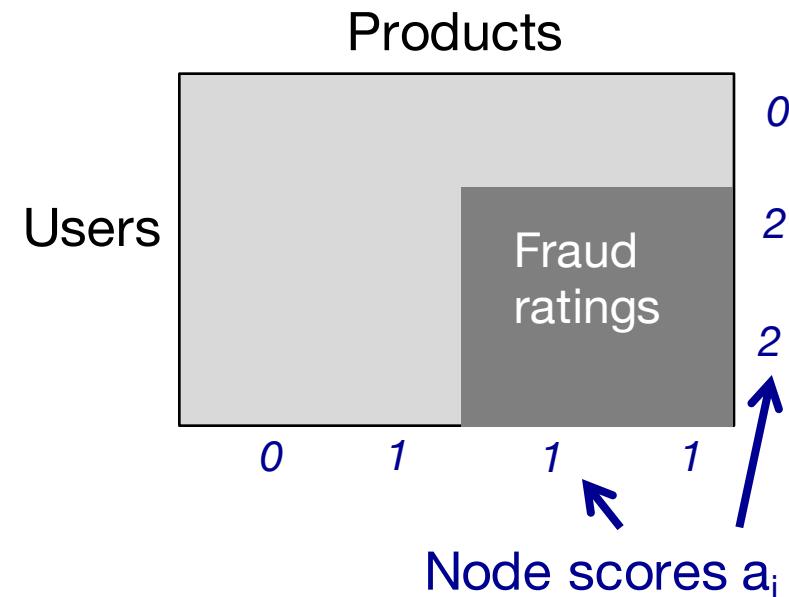


Hijacked user accounts



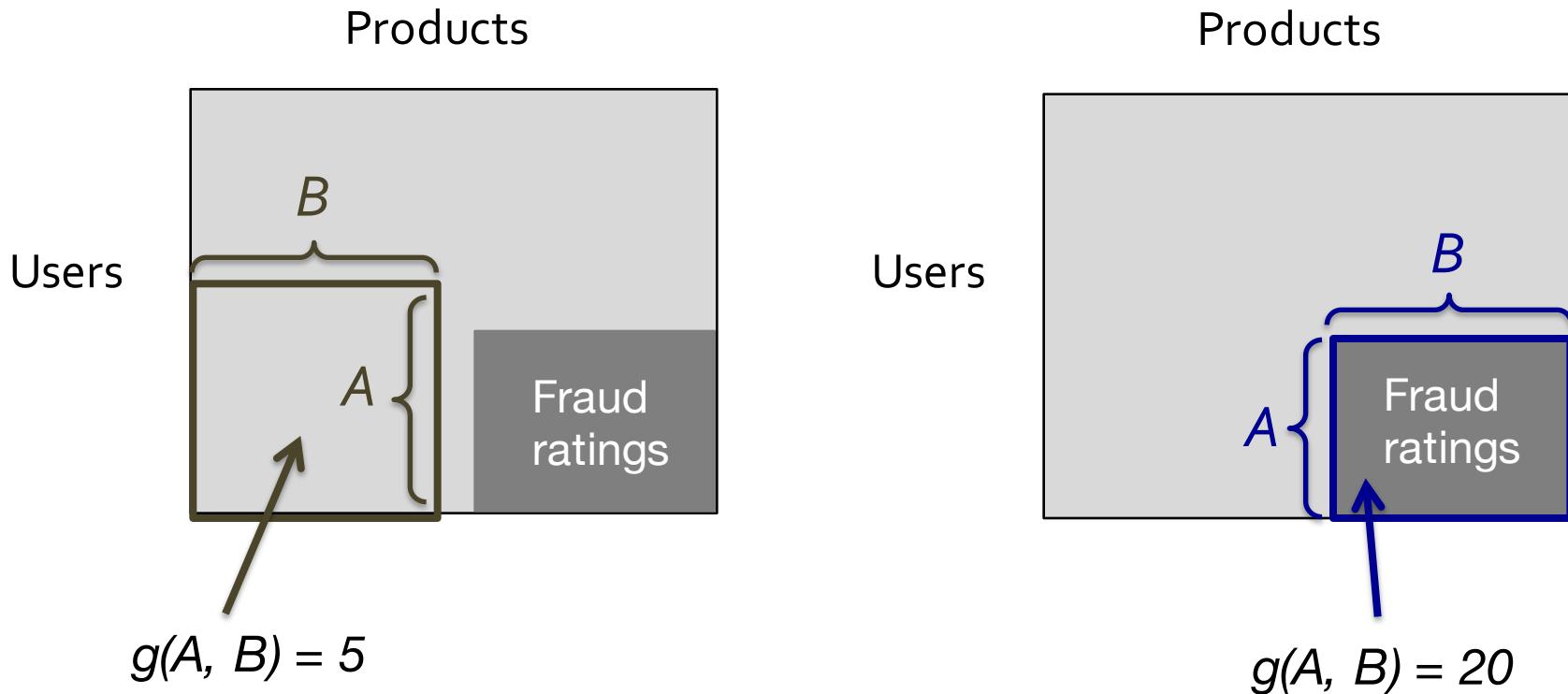
# Formal Problem Definition

- Given:
  - Bipartite graph between users and products
  - May have prior node suspiciousness scores
- Develop detection metric that is:
  - **Camouflage-resistant**
  - **Near-linear time**
  - **Offers provable bounds**
  - **Works well in practice**



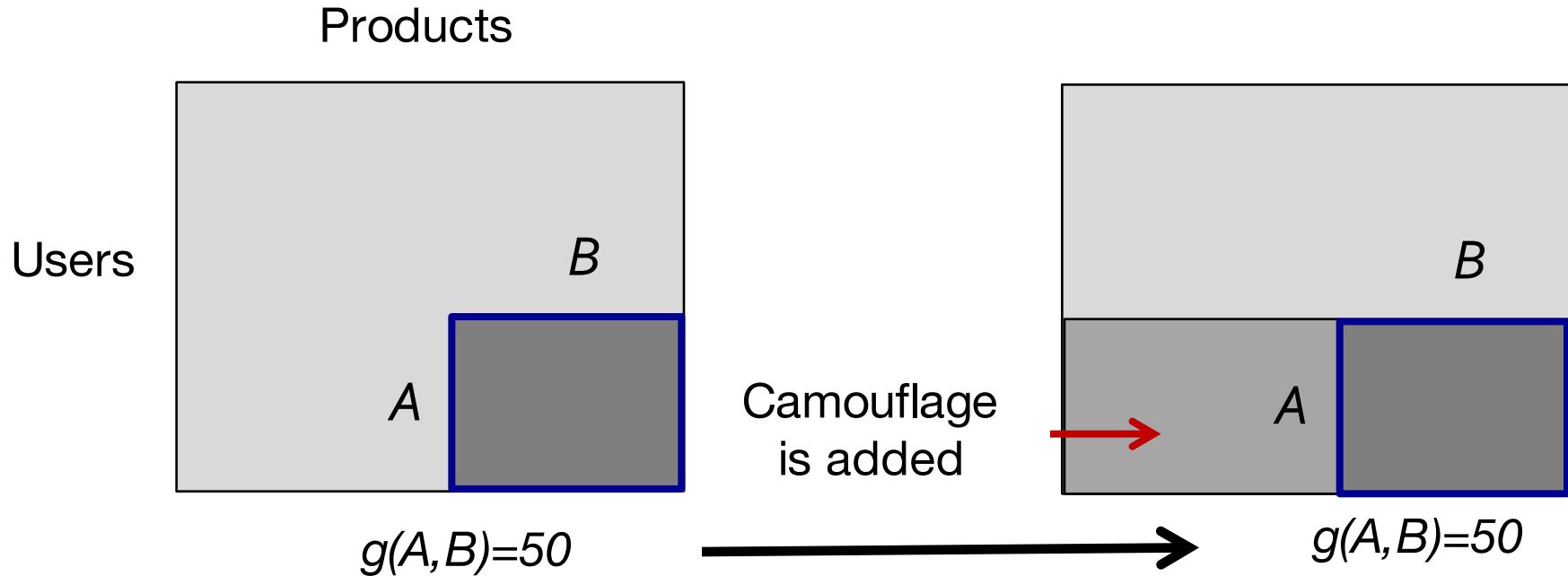
# Suspiciousness Metric

$g(A, B)$  is a density metric for edges from set of users A to set of products B.



# Camouflage-Resistance

Metric  $g$  is **camouflage-resistant** if  $g(A, B)$  does not decrease when camouflage is added to  $A$ .

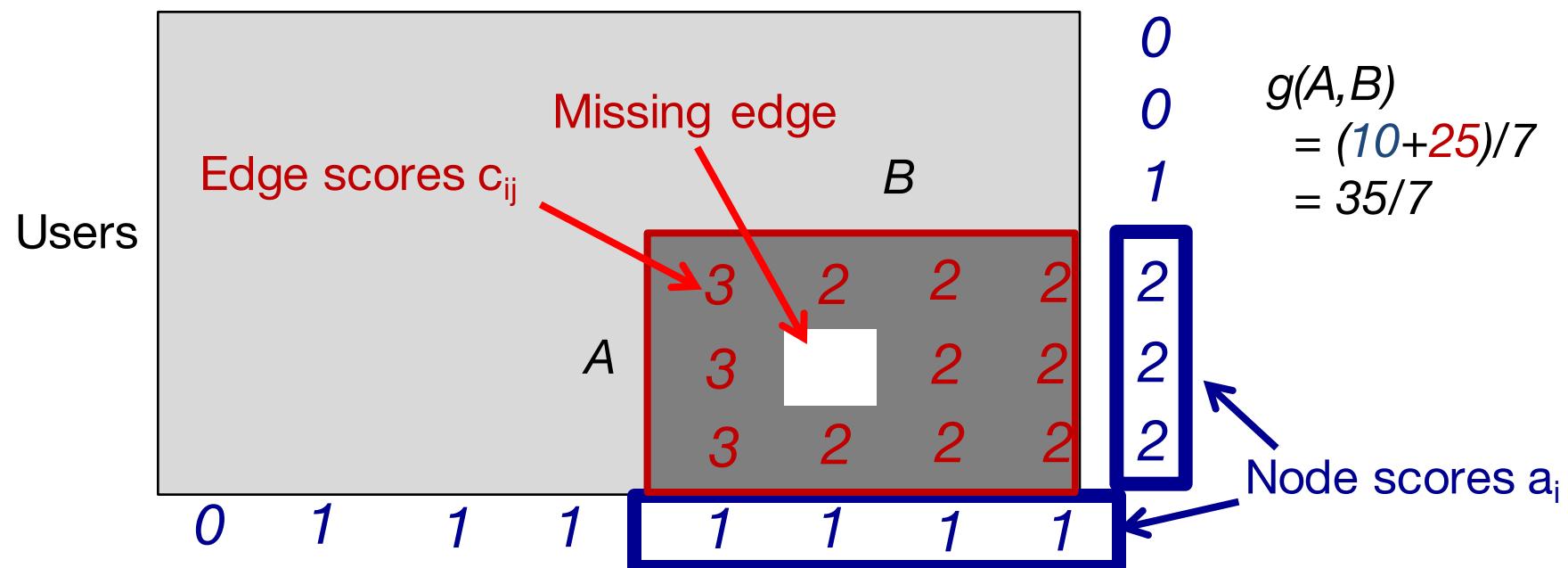


# Proposed Suspiciousness Metric

“Average suspiciousness”  $g(A, B)$

$$= \frac{(\text{sum of node susp.}) + (\text{sum of edge susp.})}{|A| + |B|}$$

Products

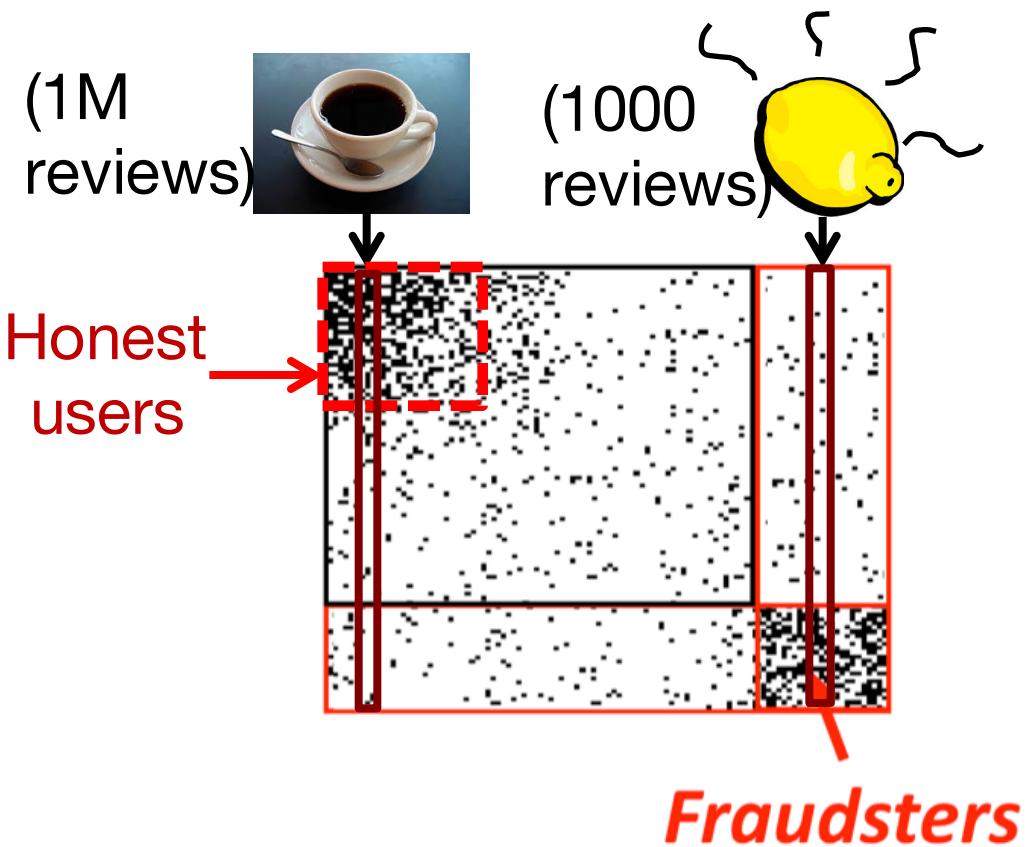


# Edge Scores $c_{ij}$

## Proposed weighting scheme:

$$c_{ij} = 1 / \log(\text{unweighted sum of } j\text{-th column})$$

tf-idf



Why?

- Popular products are not necessarily suspicious
- Fraudulent products have a high fraction of edges from fraudsters

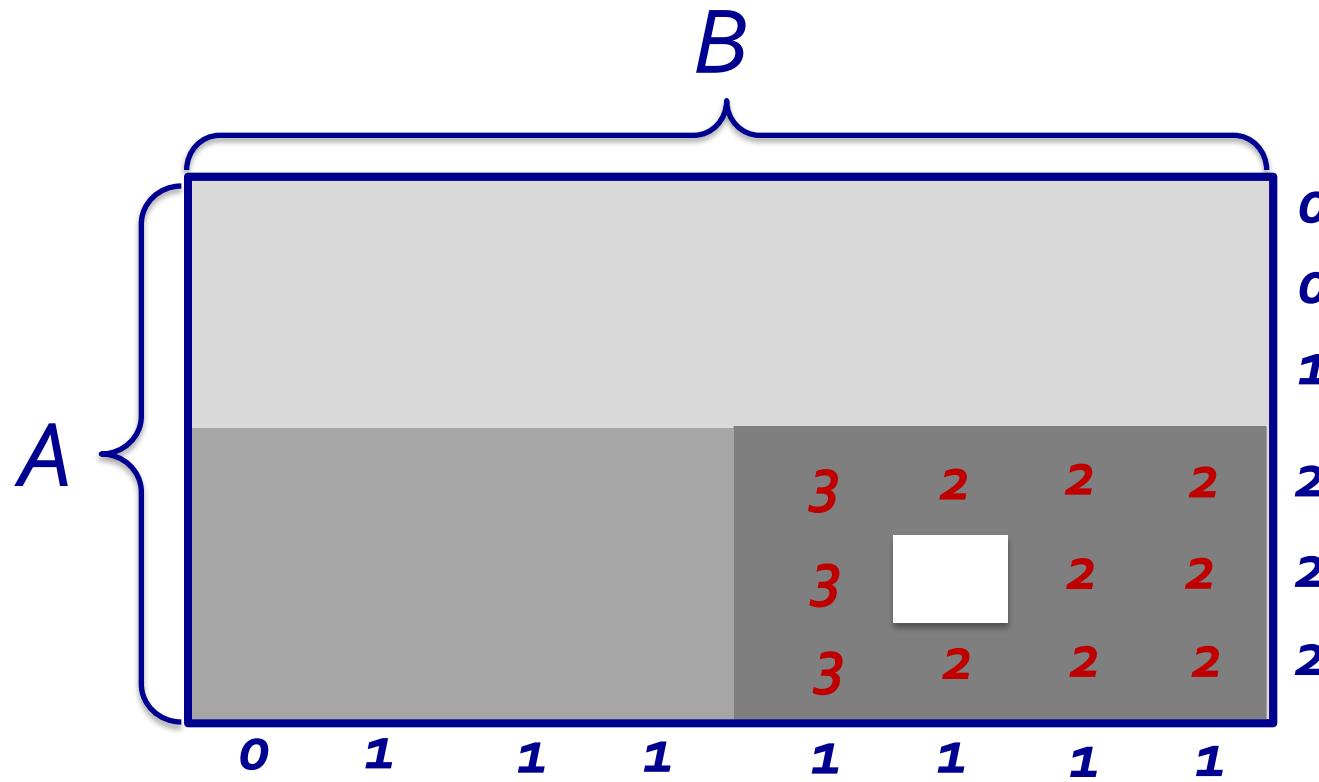
# Metric Properties

*Average suspiciousness  $g(A, B)$ :*

- Can be optimized in near-linear time**
- Provable bounds
- Camouflage-resistant
- Works in practice

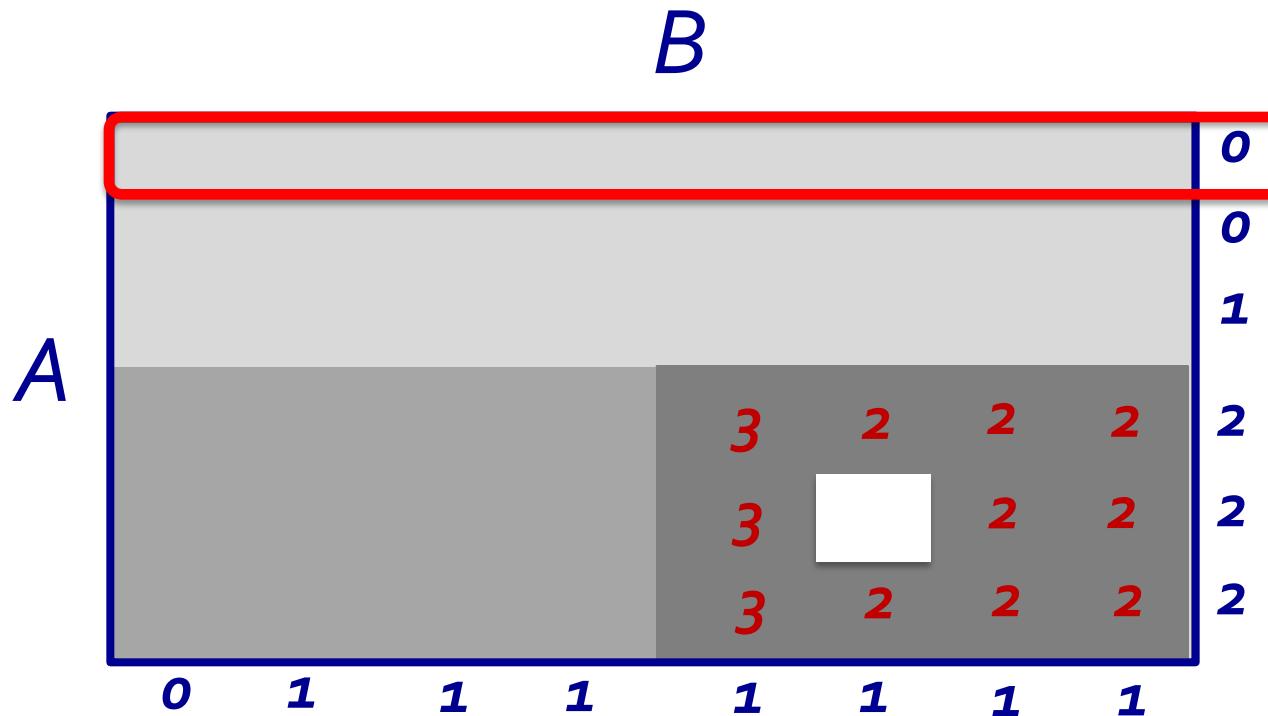
# FRAUDAR: Greedy Algorithm

- Start with sets A, B as all users / products



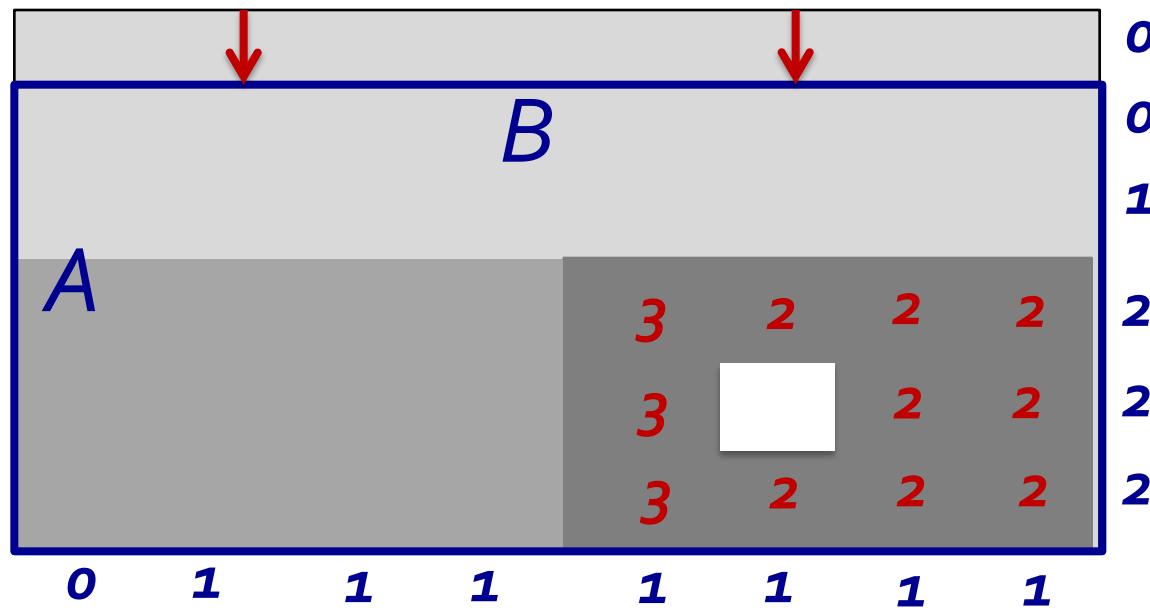
# FRAUDAR: Greedy Algorithm (cont.)

- Delete rows / columns greedily to maximize  $g$  (average suspiciousness)



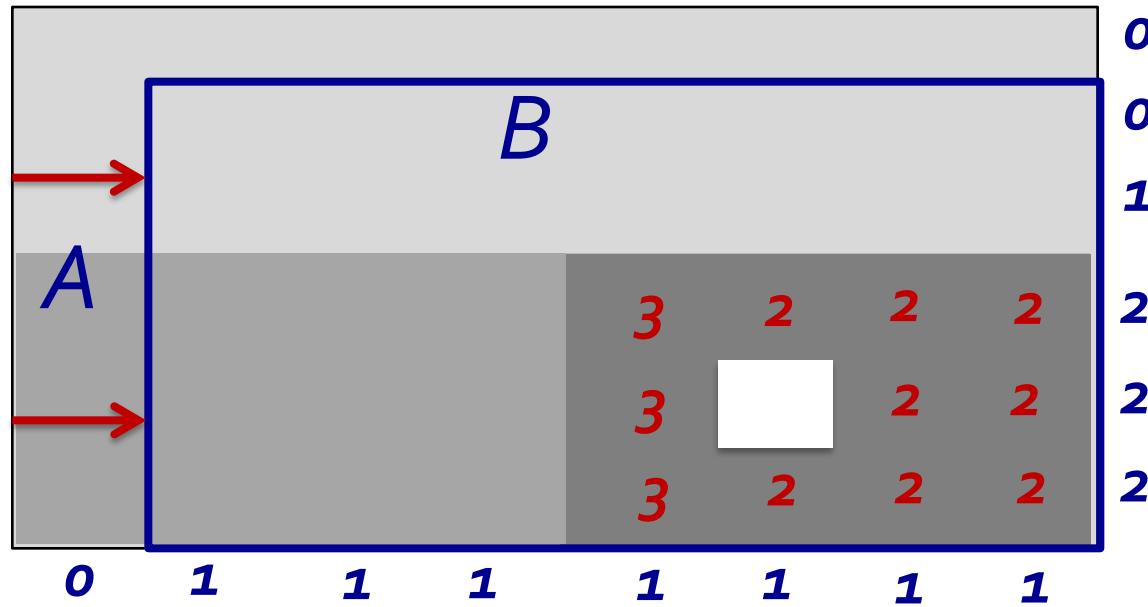
# FRAUDAR: Greedy Algorithm (cont.)

- Delete rows / columns greedily to maximize  $g$  (average suspiciousness)



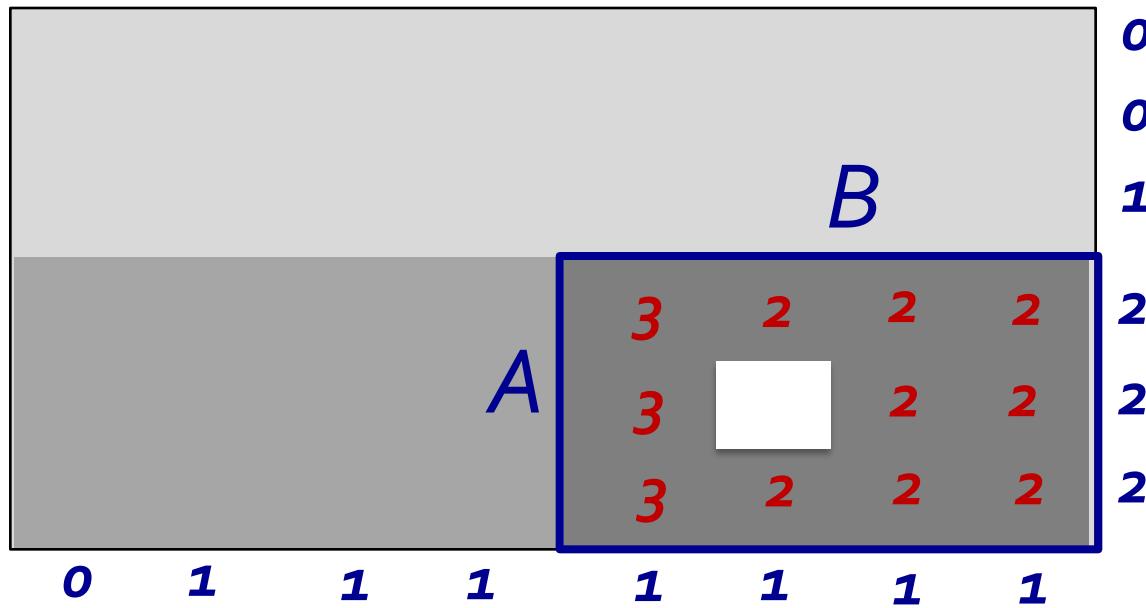
# FRAUDAR: Greedy Algorithm (cont.)

- Delete rows / columns greedily to maximize  $g$  (average suspiciousness)



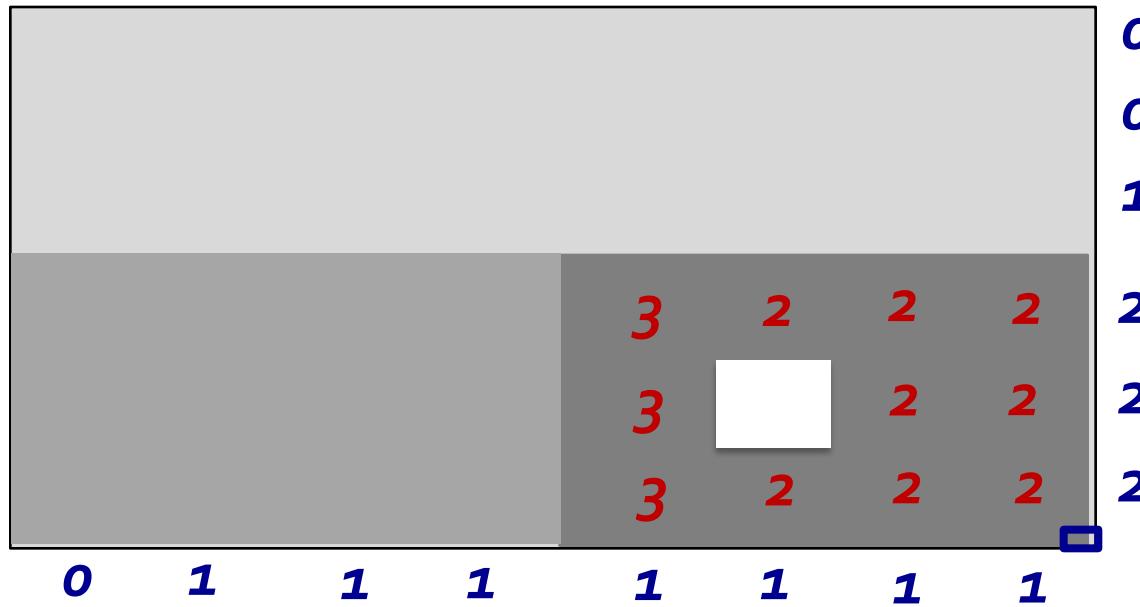
# FRAUDAR: Greedy Algorithm (cont.)

- Delete rows / columns greedily to maximize  $g$  (average suspiciousness)



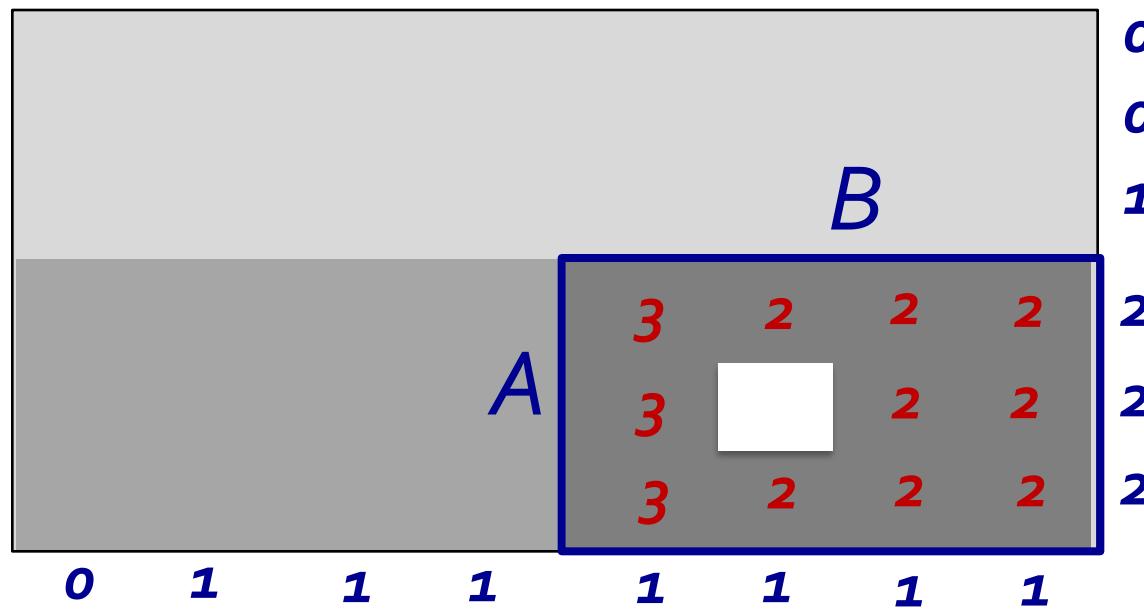
# FRAUDAR: Greedy Algorithm (cont.)

- Continue until A and B are empty



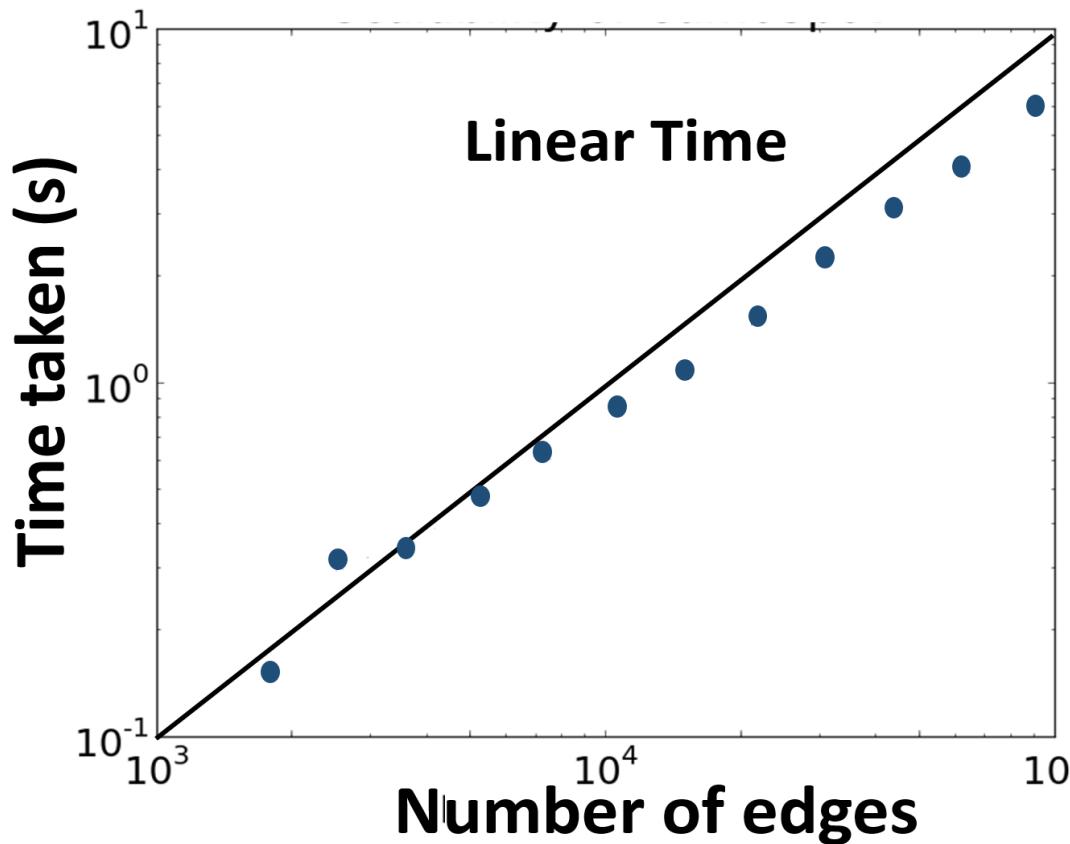
# FRAUDAR: Greedy Algorithm (cont.)

- Return the best subsets A and B seen so far (based on g)



# Computation Time

- $O(|E| \log(|V|))$ : using appropriate data structures



# Metric Properties

*Average suspiciousness  $g(A, B)$ :*

- Can be optimized in near-linear time
- Provable bounds**
- Camouflage-resistant
- Works in practice

# Theoretical Guarantee

- **Theorem 1:** The subgraph  $(A, B)$  returned by FRAUDAR satisfies

$$g(\mathcal{A} \cup \mathcal{B}) \geq \frac{1}{2}g_{OPT}$$

FRAUDAR subgraph

Optimum value of g

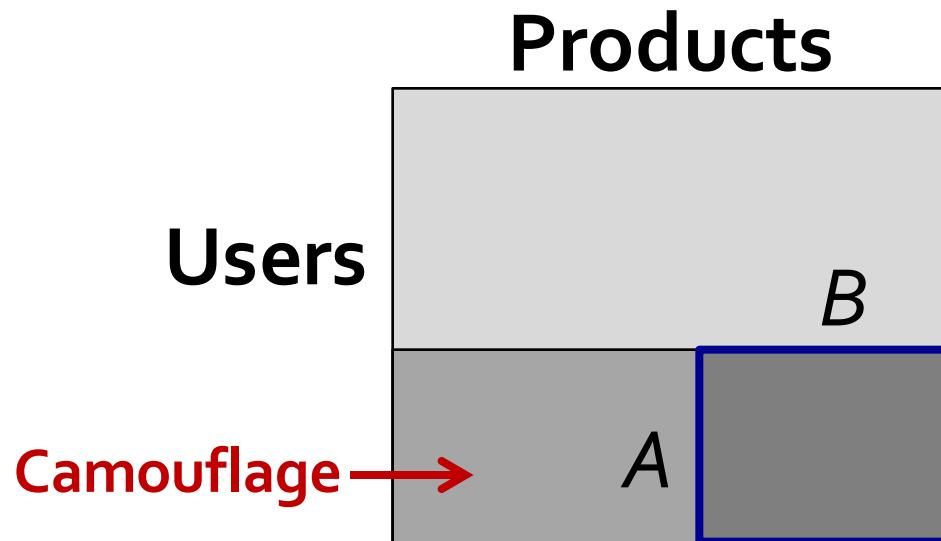
# Metric Properties

*Average suspiciousness  $g(A, B)$ :*

- Can be optimized in near-linear time
- Provable bounds
- Camouflage-resistant**
- Works in practice

# Camouflage Resistance

- **Theorem 2:** If  $c_{ij}$  is a *column weighting* (i.e.  $c_{ij}$  is any function of the  $j$ -th column), then **g is camouflage-resistant.**



$(c_{ij} = 1 / \log(\text{sum of } j\text{-th column})$  satisfies this)

# Metric Properties

*Average suspiciousness  $g(A, B)$ :*

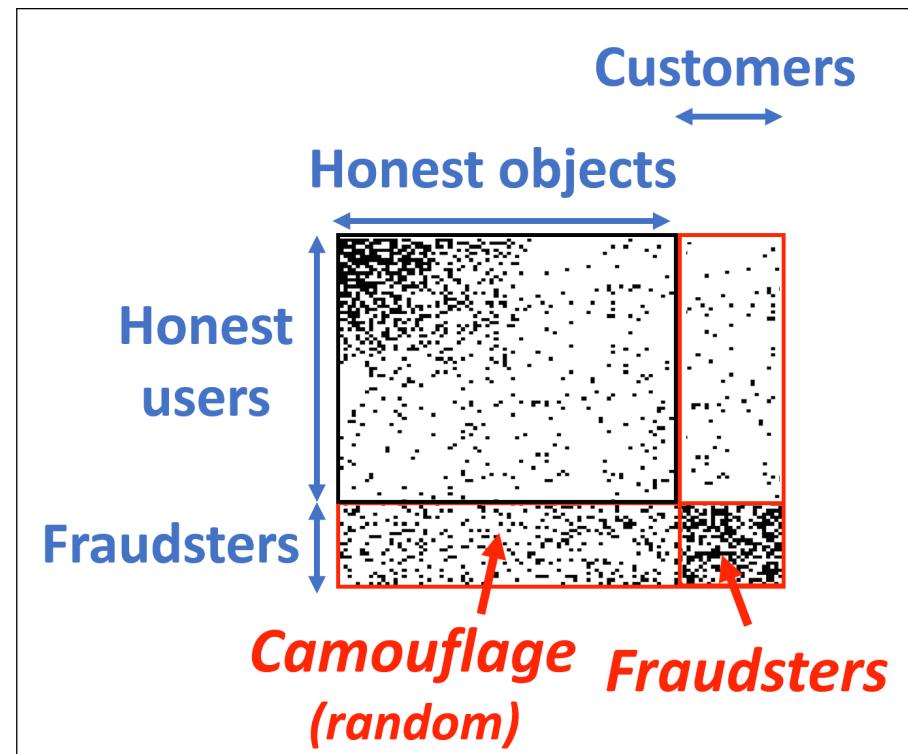
- Can be optimized in near-linear time
- Provable bounds
- Camouflage-resistant
- Works in practice**

# Experiments: Detecting Injection of Various Types of “Camouflage”

- Amazon Review Graph:  
24K users, 4K products
- Injected  $200 \times 200$  blocks with various types of camouflage
  - None
  - Random camouflage
  - Biased camouflage
  - Hijacked accounts

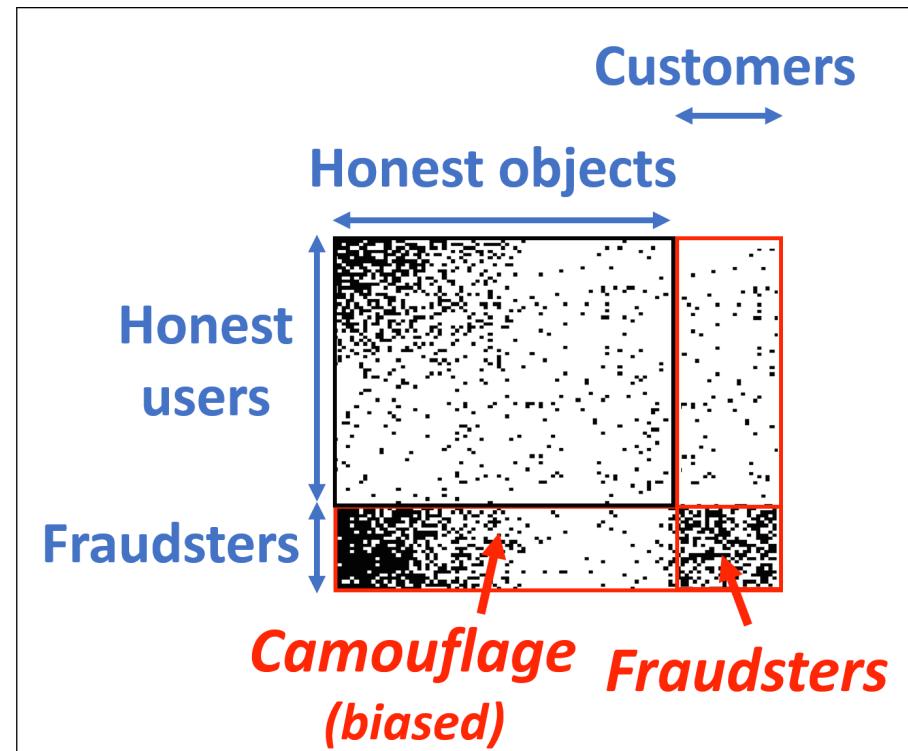
# Experiments: Detecting Injection of Various Types of “Camouflage”

- Amazon Review Graph:  
24K users, 4K products
- Injected  $200 \times 200$  blocks with various types of camouflage
  - None
  - **Random camouflage**
  - Biased camouflage
  - Hijacked accounts



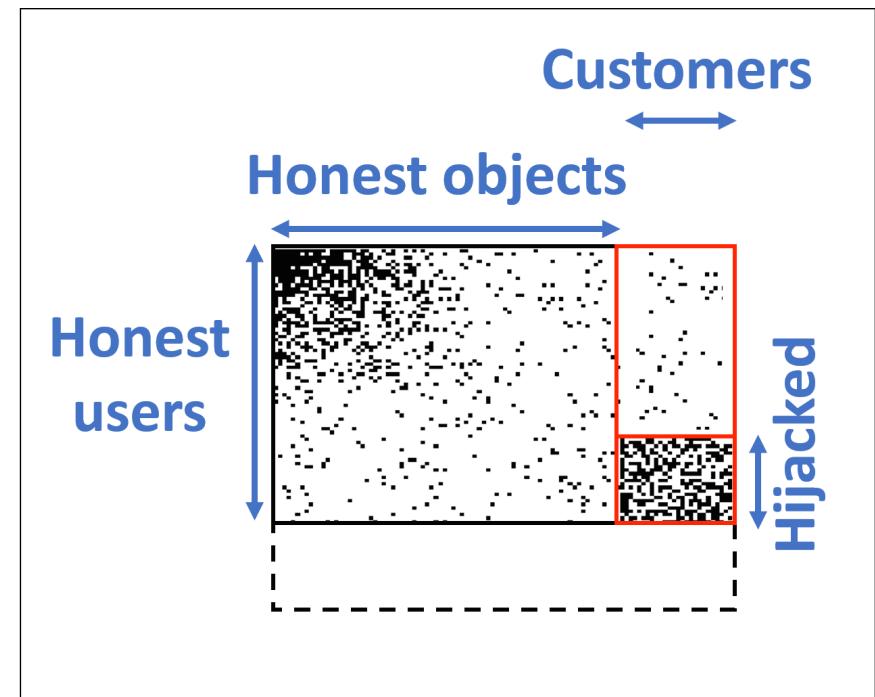
# Experiments: Detecting Injection of Various Types of “Camouflage”

- Amazon Review Graph:  
24K users, 4K products
- Injected  $200 \times 200$  blocks with various types of camouflage
  - None
  - Random camouflage
  - **Biased camouflage**
  - Hijacked accounts

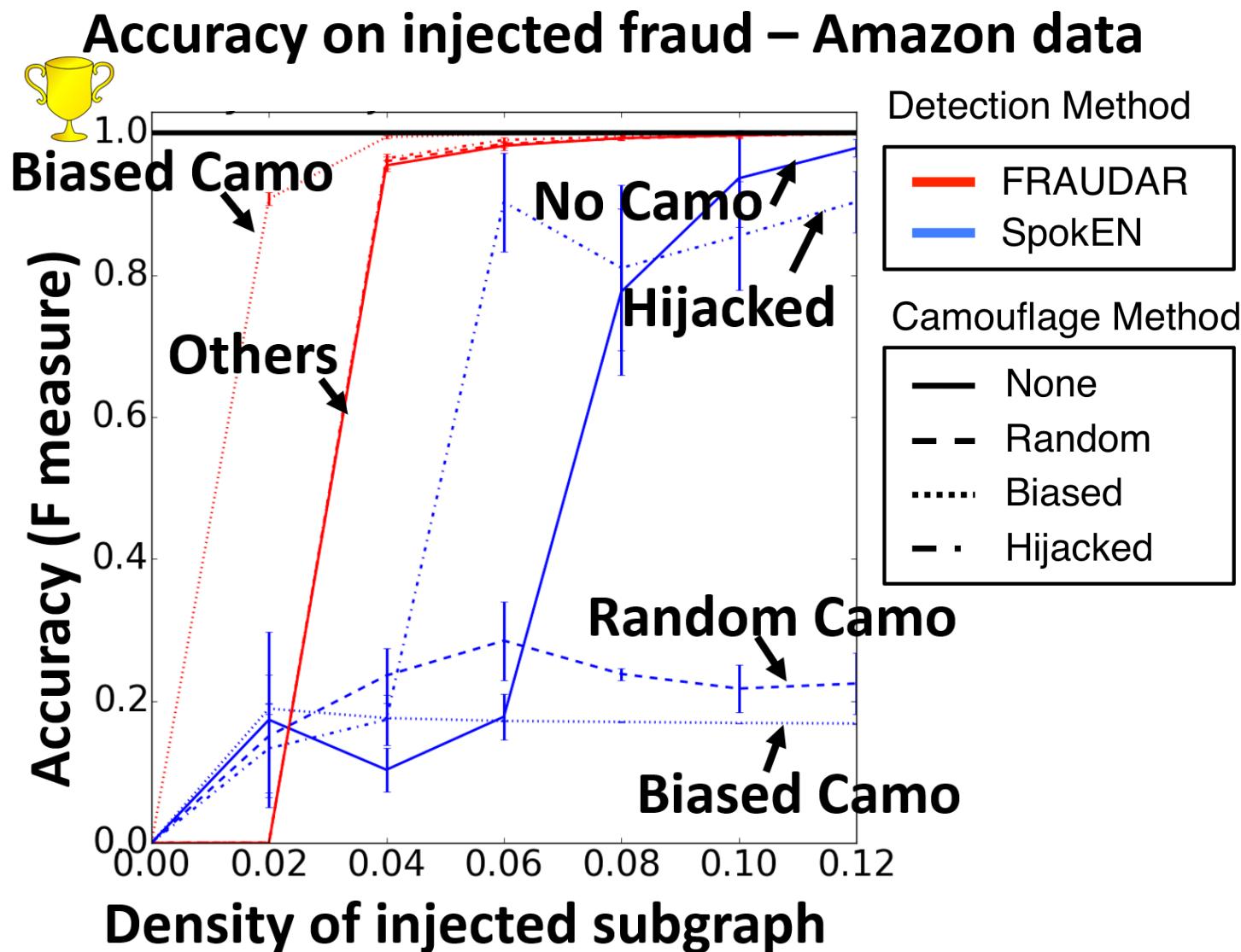


# Experiments: Detecting Injection of Various Types of “Camouflage”

- Amazon Review Graph:  
24K users, 4K products
- Injected  $200 \times 200$  blocks with various types of camouflage
  - None
  - Random camouflage
  - Biased camouflage
  - **Hijacked accounts**



# Accuracy on Detecting Injected Fraud



# Summary

- Spectral methods
  - Spectral clustering and community detection
- Spectral subspaces and spectral subspace plots
- **EigenSpokes** (singular vectors and “spokes”)
- **LockInfer** (“**camouflage**”, “fame”, “pearls”, “staircase”, etc.)
- **fBox** (small-scale, stealthy attacks; reconstructed degrees)
- **FRAUDAR** (theoretical guarantees for bounding graph fraud in the face of **camouflage**)
- Applications: Mobile calls, Twitter social network, “user-product” reviews

# References

- Andrew Y. Ng, Michael I. Jordan, Yair Weiss. “On Spectral Clustering: Analysis and an algorithm”, NIPS 2001.
- Scott White and Padhraic Smyth. “A spectral clustering approach to finding communities in graphs”, SDM 2005.
- M. E. J. Newman. “Finding community structure in networks using the eigenvectors of matrices”, Physical Review E 2006.
- Aditya Prakash, Mukund Seshadri, Ashwin Sridharan, Sridhar Machiraju, Christos Faloutsos. “EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs”, PAKDD 2010.
- Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, Shiqiang Yang. “Inferring lockstep behavior from connectivity pattern in large graphs”, KAIS 2016.
- Neil Shah, Alex Beutel, Brian Gallagher, Christos Faloutsos. “Spotting suspicious link behavior with fBox: An adversarial perspective”, ICDM 2014.
- Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, Christos Faloutsos. “FRAUDAR: Bounding Graph Fraud in the Face of Camouflage”, KDD 2016 Best Research Paper Award.



KDD 2017

Halifax, Nova Scotia - Canada  
August 13 - 17, 2017

# Tutorial: Data-Driven Approaches towards Malicious Behavior Modeling



Meng Jiang  
University of Notre Dame



Srijan Kumar  
Stanford University



Christos Faloutsos  
Carnegie Mellon University



V.S. Subrahmanian  
University of Maryland, College Park

Tutorial link: <http://bit.ly/kdd2017>