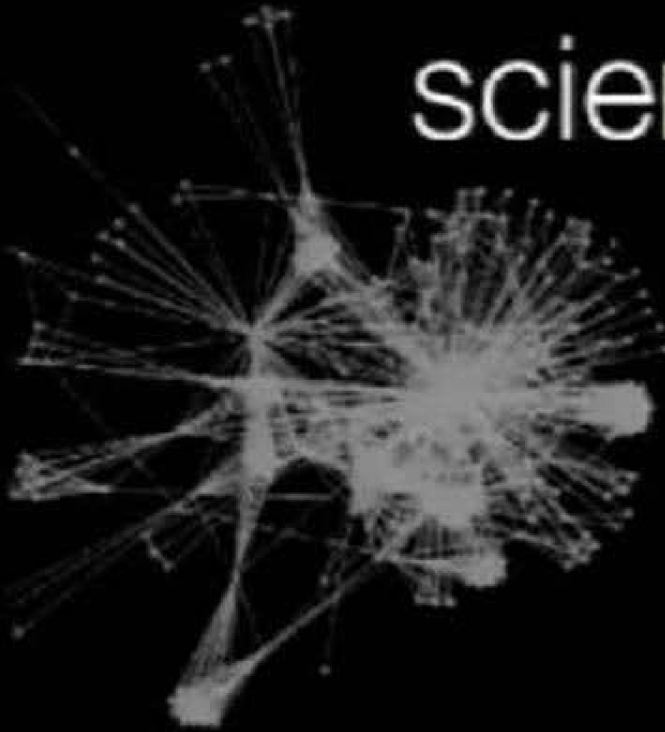
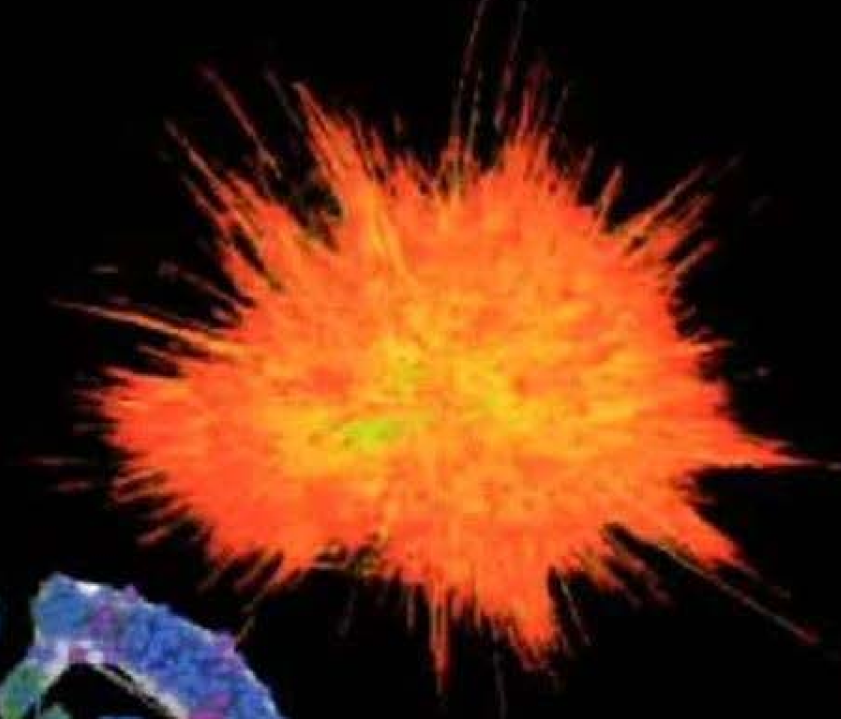
What is network science?





Network Science

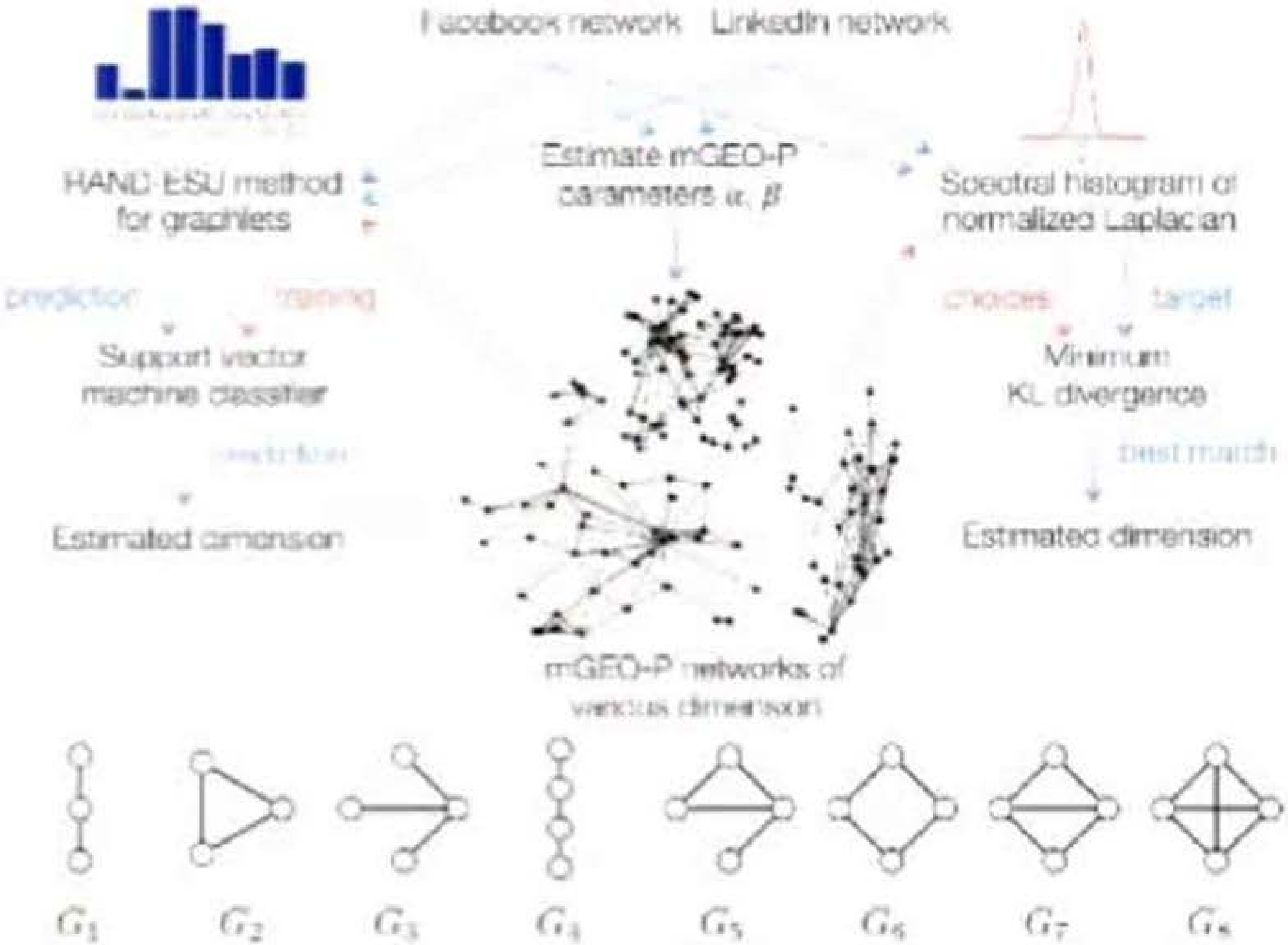
the study of ***network representations*** of physical, biological, and social phenomena leading to ***predictive models*** of these phenomena
National Research Council (via Wikipedia)

- Models
- Algorithms
- Data

Network Science is
CS&E applied to graphs
Me

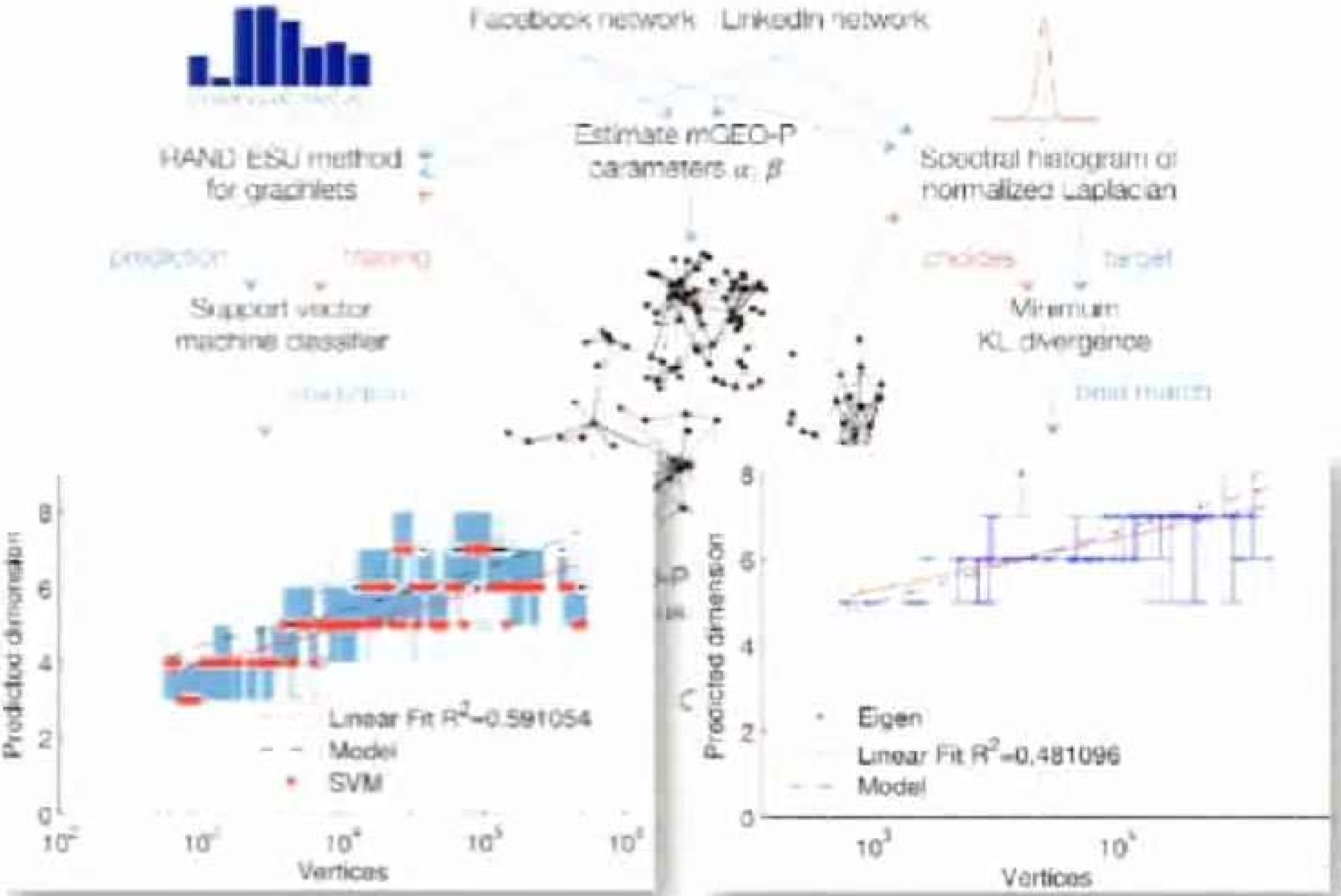
Dimensionality of social networks using motifs and eigenvalues

w/ Bonato, et al. PLOS One (2014) 10.1371/journal.pone.0106052 arxiv:1406.0167



Dimensionality of social networks using motifs and eigenvalues

w/ Bonato, et al. PLOS One (2014) 10.1371/journal.pone.0106052 arXiv:1405.0167

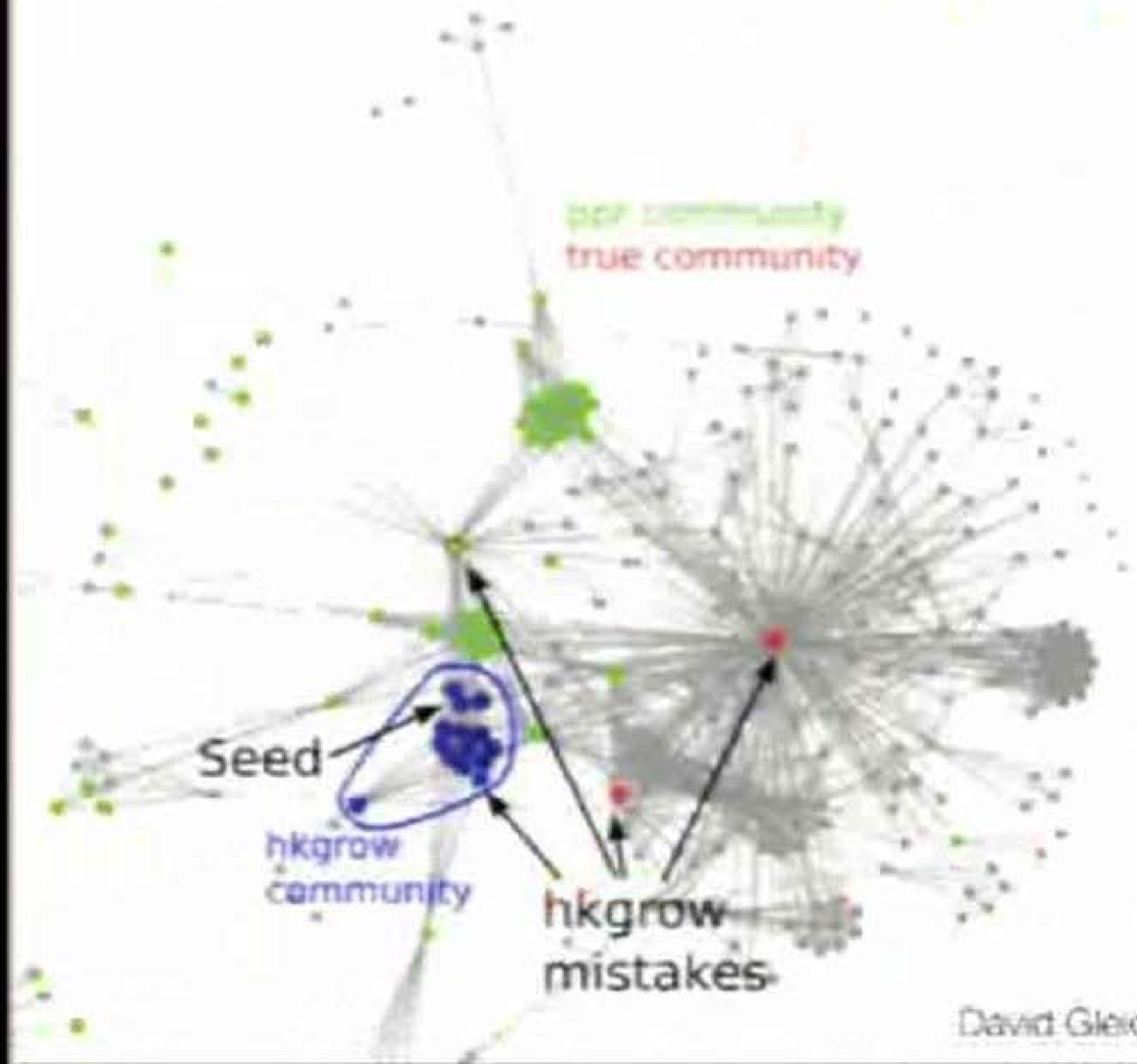


Local methods in network science

Joint work with Joyce Whang, Kyle Kloster, Michael Mahoney, and Inderjit Dhillon. Supported by NSF CAREER Award

David F. Gleich
Purdue University

Local methods identify small, meaningful regions in massive networks



Co-purchasing in Amazon
330k Vertices, 1M edges

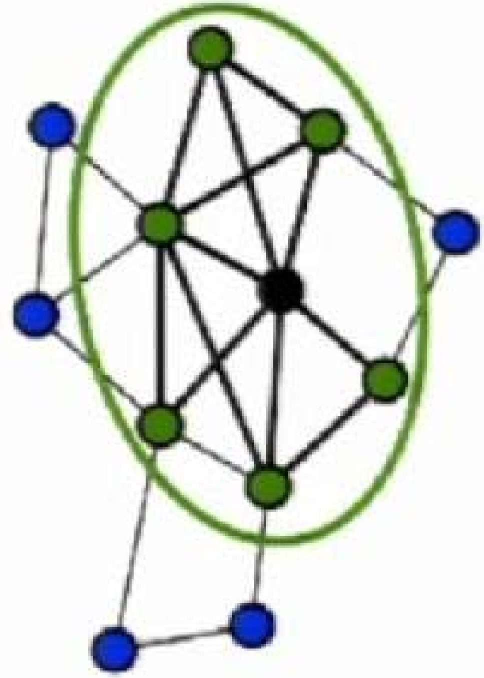
Ground-truth set has
~10 vertices

Two local methods

- pprgrow
- hkgrow

find most of the true set starting from a seed inside the set.

Local methods help characterize the graph!



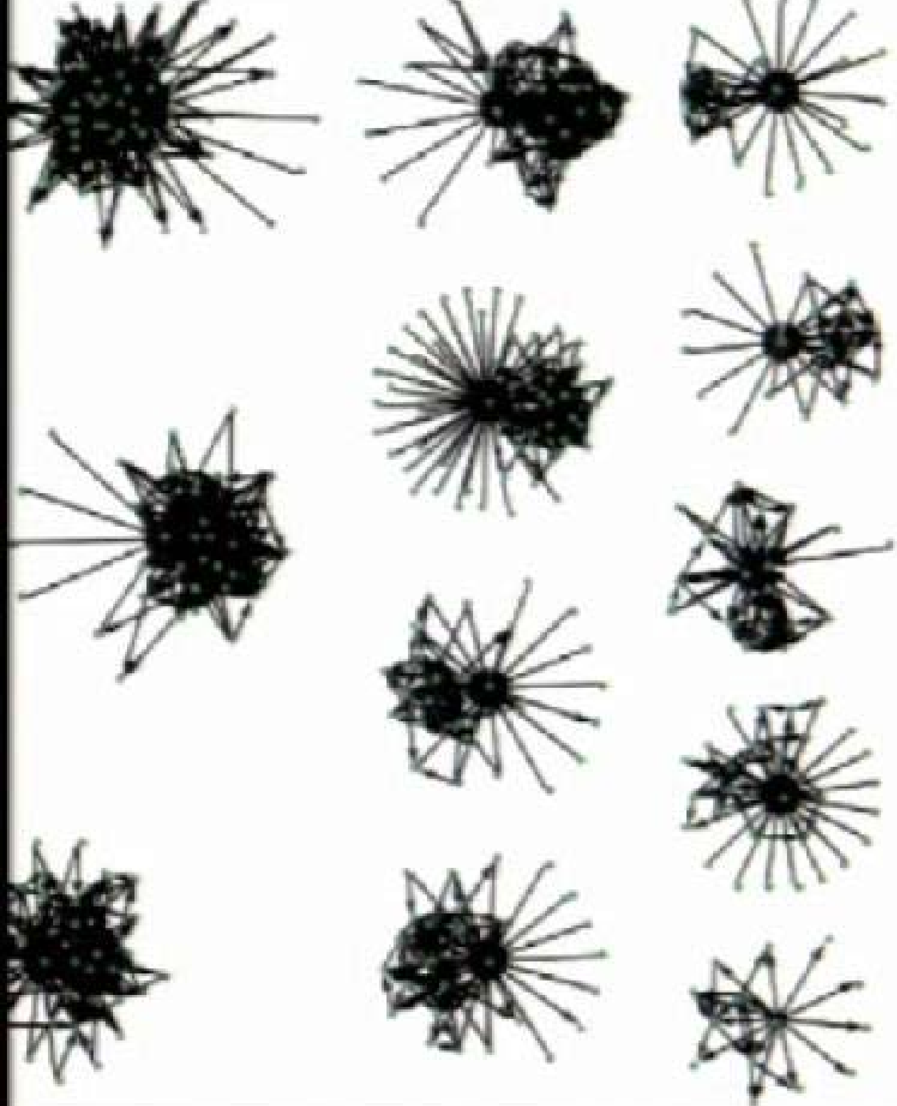
Vertex neighborhood or egonet

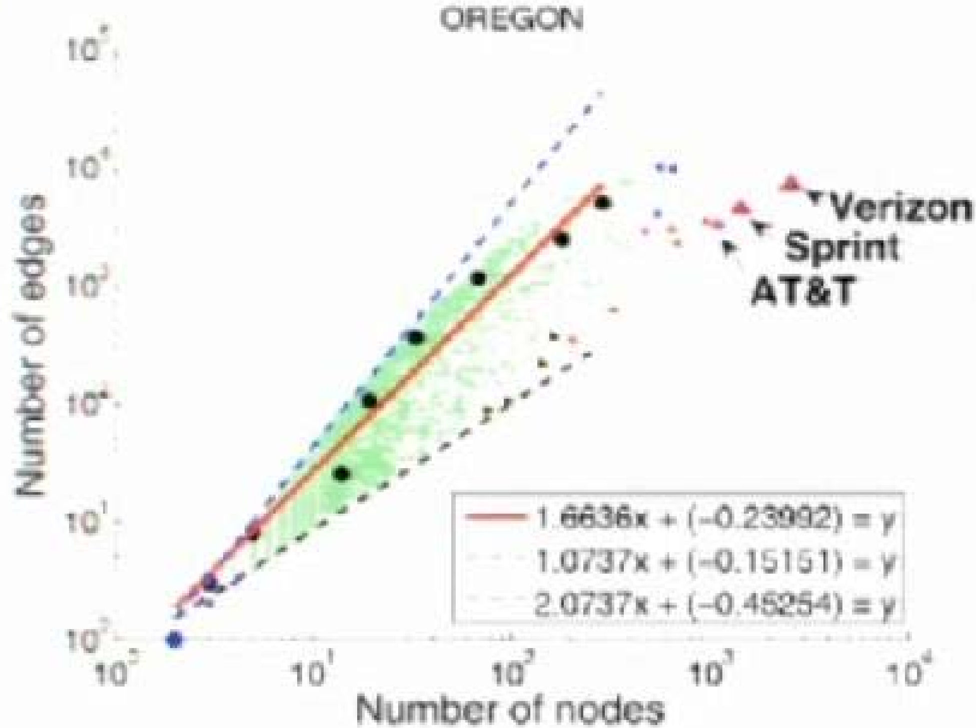
The induced subgraph of a vertex its neighbors

Egonets of social networks should show "structural holes" *[Burt95,Kleinberg08]*.

Used for anomaly detection *[Akoglu10]*,
community seeds *[Huang11,Schaeffer11]*,
overlapping communities *[Schaeffer07,Rees10]*.

Characterizing local anomalies in graphs using egonets





From OddBall: Spotting Anomalies in Weighted Graphs
Akoglu et al. PAKDD2010

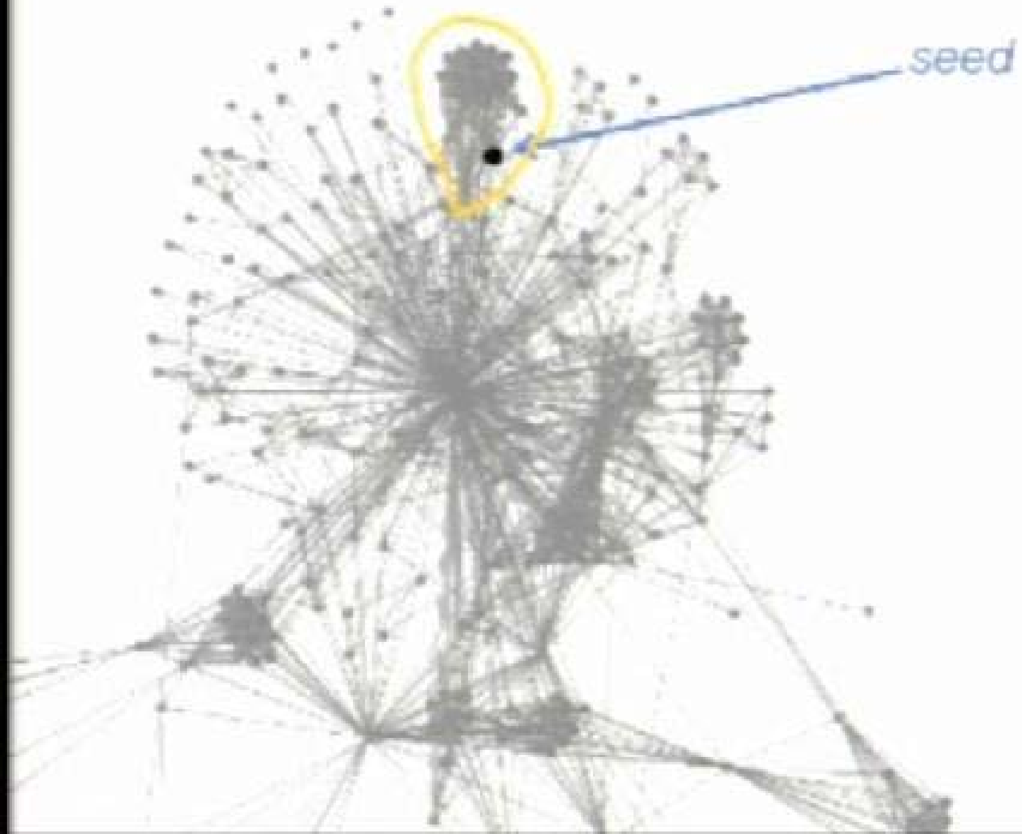
From Knowledge Sharing and Yahoo Answers
Adamic et al. WWW2005

Our perspective

Local diffusions are some of the best community detection algorithms available!

Local Community Detection

Given seed(s) S in G, find a community that contains S.

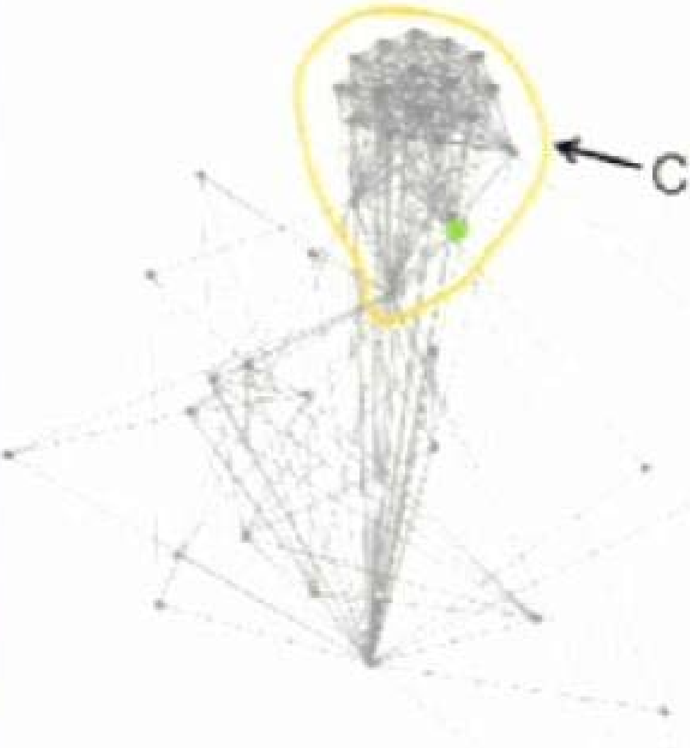


Community
A set of vertices with high internal and low external connectivity

Low-conductance sets are communities

$$\text{conductance}(T) = \frac{\#\text{ edges leaving } T}{\#\text{ edge endpoints in } T}$$

= " chance a random step exits T "

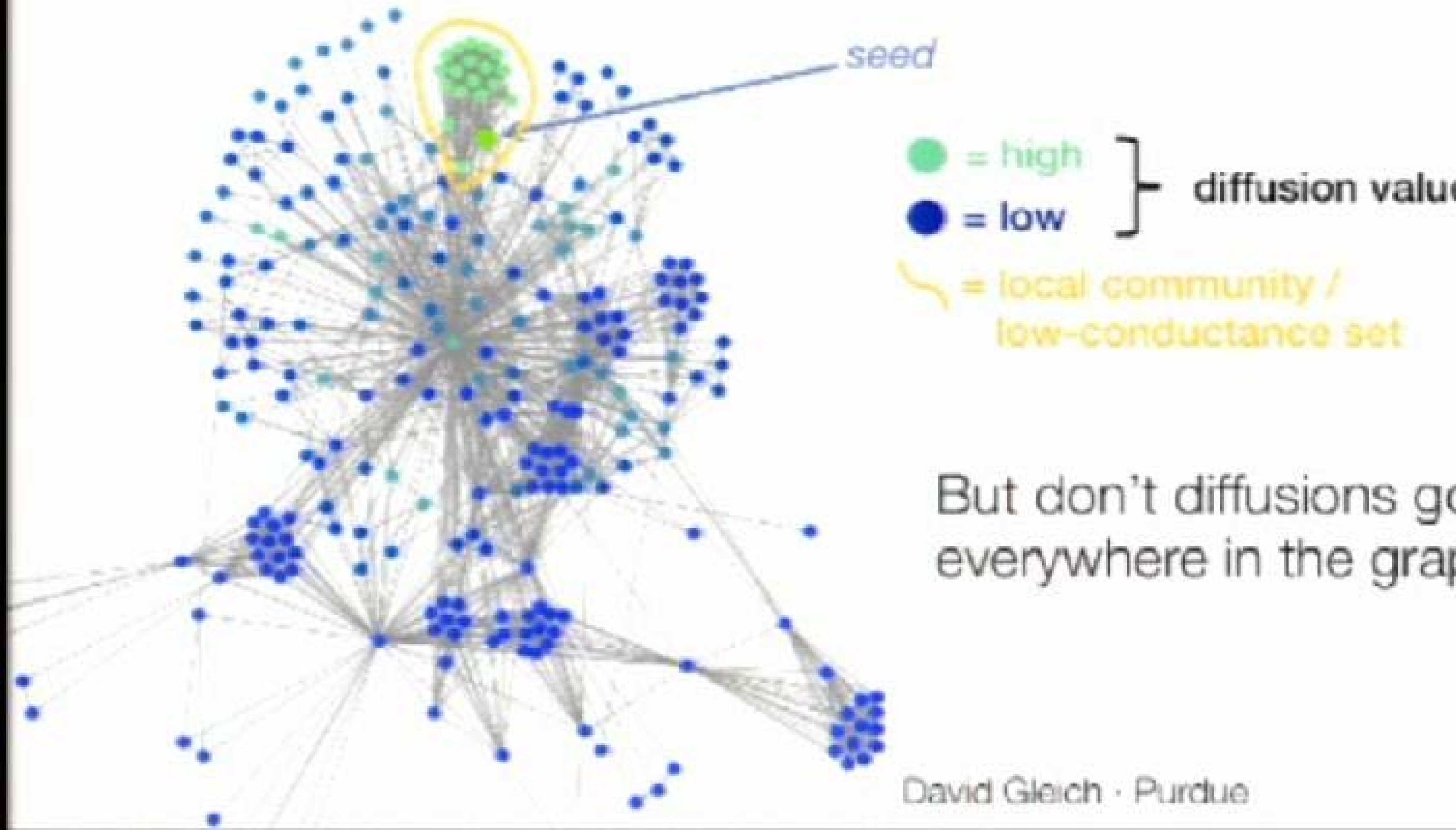


← conductance(comm) =
39/381 = .102

How to find these ?

Graph diffusions find low-conductance sets

A diffusion propagates "rank" from a seed across a graph.



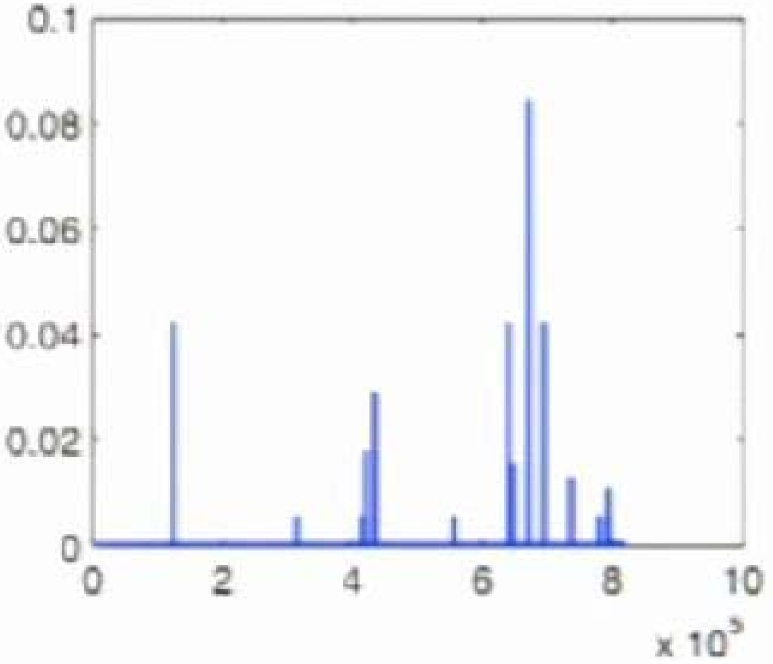
But don't diffusions go everywhere in the graph?

The most used diffusions stay localized even in massive graphs

$$(\mathbf{I} - \beta \mathbf{P})\mathbf{x} = (1 - \beta)\mathbf{s}$$

plot($\mathbf{x}$)

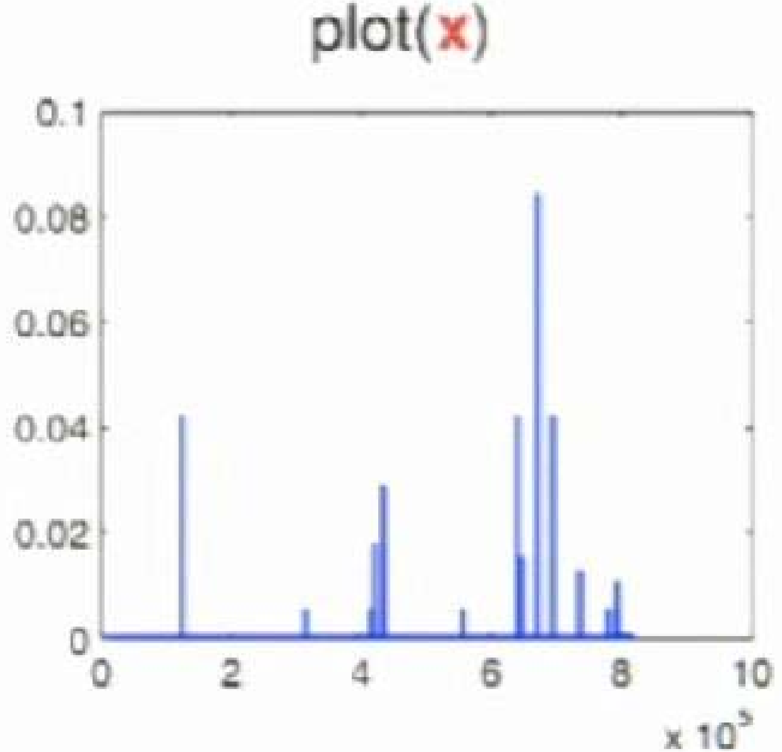
$\text{nnz}(\mathbf{x}) \approx 800k$

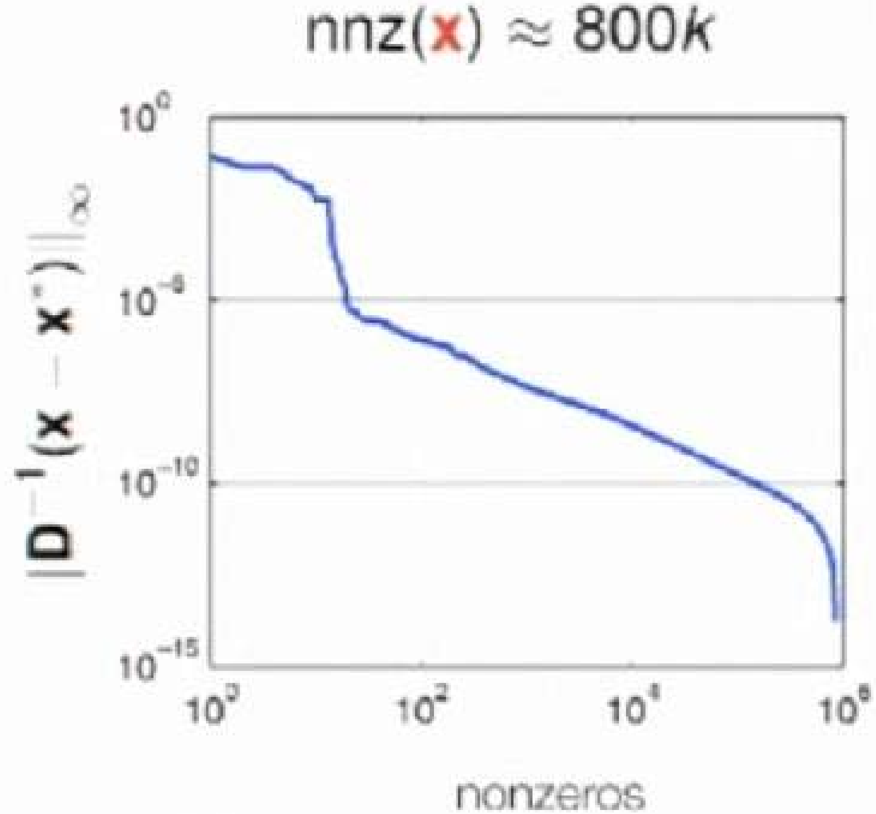


Crawl of flickr from 2006 ~800k nodes, 6M edges, beta=1/2

The most used diffusions stay localized even in massive graphs

$$(\mathbf{I} - \beta \mathbf{P})\mathbf{x} = (1 - \beta)\mathbf{s}$$





Crawl of flickr from 2006 ~800k nodes, 6M edges, beta=1/2

Our mission

Find the solution with work roughly proportional to the *localization*, not the matrix.

Our Point

Coordinate relaxation methods yield localized algorithms for diffusions

in a pleasingly wide variety of settings.

Our Results

New empirical and theoretical insights into *why* and *how* a specific form is so effective.

Localized methods for diffusions use the push coordinate relaxation strategy

The push method

Coordinate relaxation
for $\boldsymbol{A}\mathbf{x} = \mathbf{b}$

Update $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \rho_j \mathbf{e}_j$

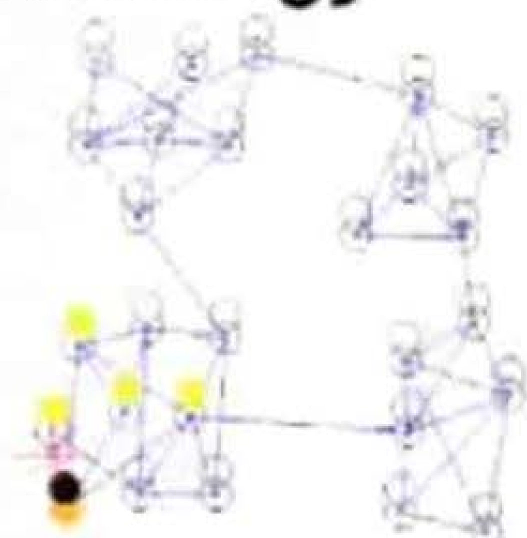
such that $[\boldsymbol{A}\mathbf{x}^{(k+1)}]_j = [\mathbf{b}]_j$

or $[\boldsymbol{A}\mathbf{x}^{(k+1)}]_j = [\mathbf{b}]_j + \varepsilon_j$

Used in coordinate descent, Gauss-Seidel, Gauss-Southwell, and many other methods.

Localized methods for diffusions use the push coordinate relaxation strategy

Push on a graph-based linear system
has a super-duper awesome property



The Push Method for PPR on a graph ε, ρ

1. $\mathbf{x}^{(1)} = 0, \mathbf{r}^{(1)} = (1-\beta)\mathbf{e}_i, k = 1$
2. *while any* $r_j > \varepsilon d_j$ (d_j *is the degree of node* j)
3. $\quad \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + (r_j - \varepsilon d_j \rho)\mathbf{e}_j$
4. $\quad r_i^{(k+1)} = \begin{cases} \varepsilon d_j \rho & i = j \\ r_i^{(k)} + \beta(r_j - \varepsilon d_j \rho)/d_j & i \sim j \\ r_i^{(k)} & \text{otherwise} \end{cases}$
5. $\quad k \leftarrow k + 1$

Push is fast!

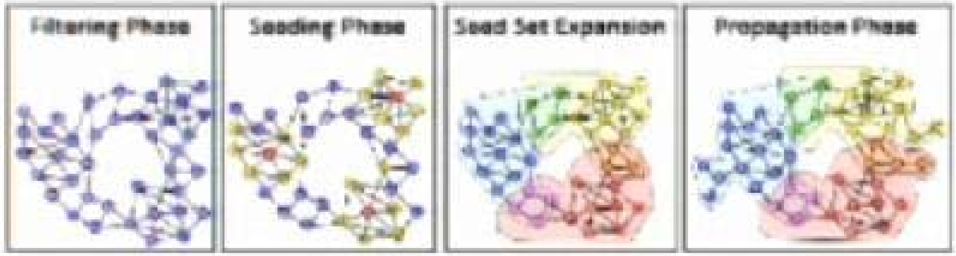
PageRank $(\mathbf{I} - \alpha \mathbf{P})\mathbf{x} = (1-\alpha)\mathbf{e}_i$

Katz $(\mathbf{I} - \beta \mathbf{A})\mathbf{x} = (1-\alpha)\mathbf{e}_i$

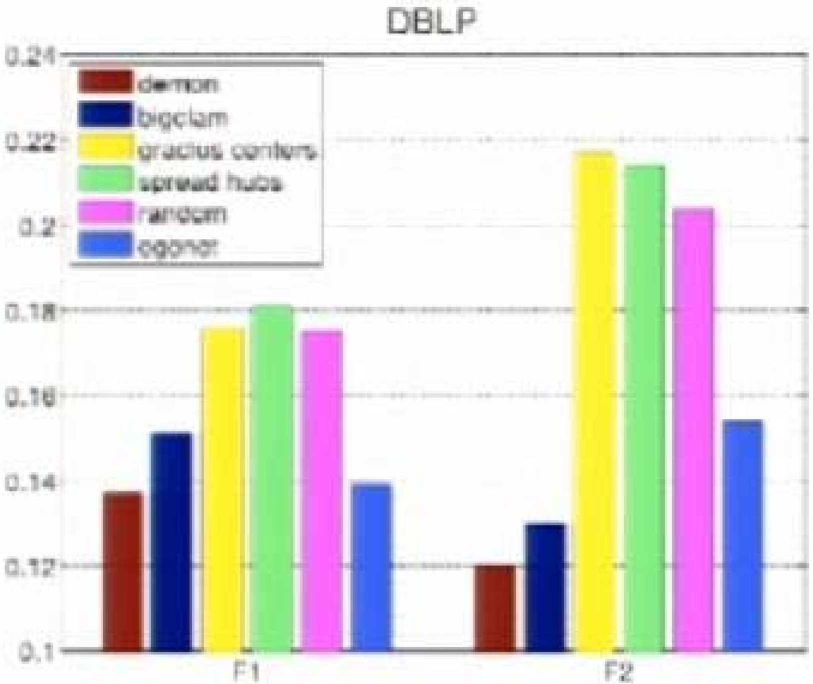
For the PageRank diffusion, Push gives constant work (entry-wise)

1. For the Katz diffusion
   Push works empirically fast
2. For the exponential $\mathbf{x} = \exp(\mathbf{P})\mathbf{e}_i$
   Push gives uniform localization on power-law graphs and fast runtimes
3. For the heat kernel diffusion
   Push gives constant work (entry-wise) $\mathbf{x} = \exp(t\mathbf{P})\mathbf{e}_i$
4. For the PageRank diffusion
   Push yields sparsity regularization
5. For a general class of diffusions
   There is a Cheeger inequality like before
6. For the PageRank diffusion
   Push gives the solution path in constant work (entry-wise)

Push is useful!



1. Push implicitly regularizes semi-supervised learning
   Gleich and Mahoney, submitted
2. Push gives state of the art results for overlapping community detection
   Whang, Gleich, Dhillon, CIKM 2013
   Whang, Gleich, Dhillon, in prep.
3. Push for overlapping clusters decrease communication in parallel solutions
   Andersen, Gleich, Mirrokni, WSDM 2012



Heat kernel localization

General recipe

1. Take problem X, convert into a linear system
2. Apply "push" to that linear system
3. Analyze and bound total work

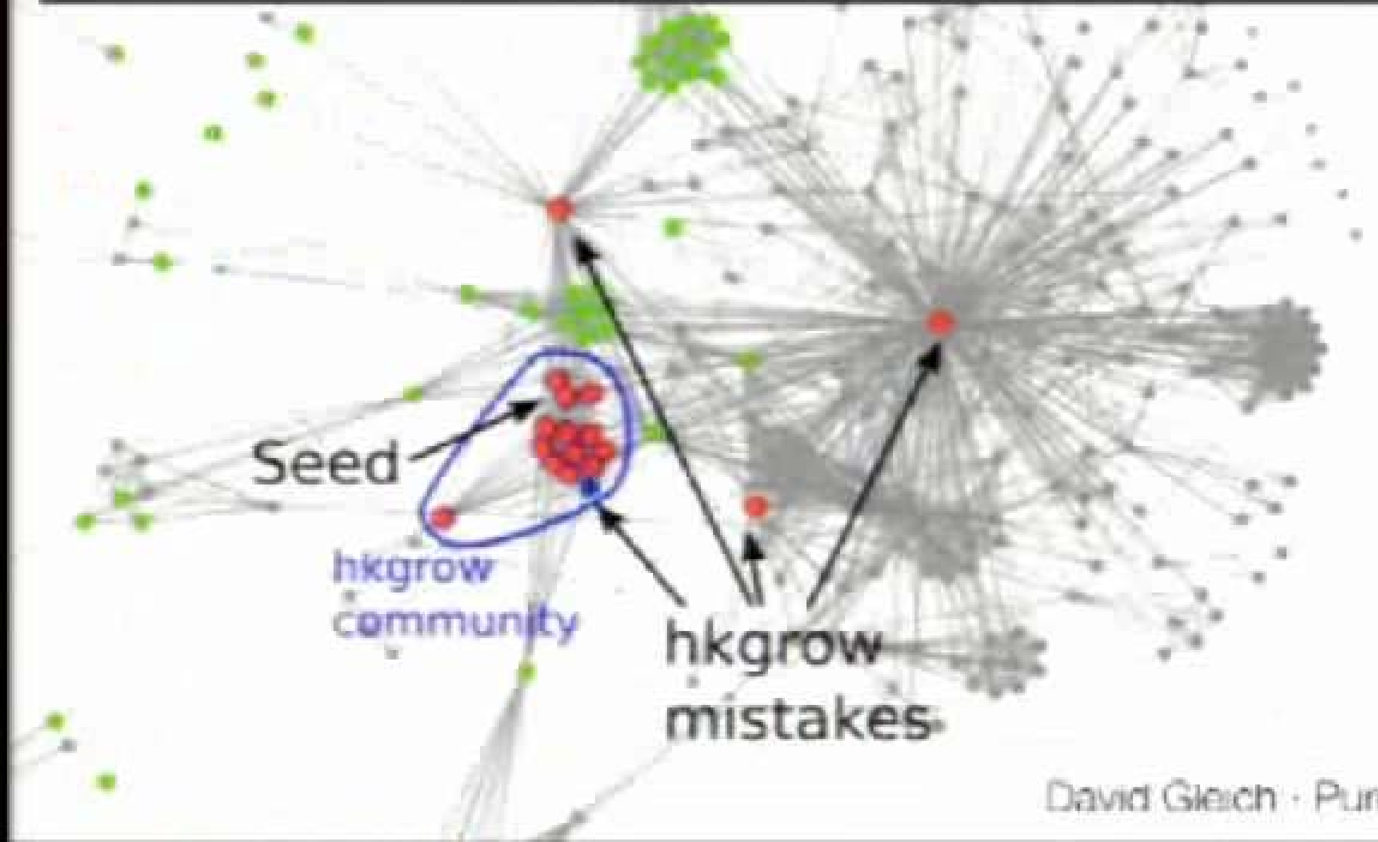
Heat kernel recipe

1. Convert $\mathbf{x} = \exp(t\mathbf{P})\mathbf{e}_i$ into

$$\begin{bmatrix} I & & & & \\ -t\mathbf{P}/1 & I & & & \\ & -t\mathbf{P}/2 & \ddots & & \\ & & \ddots & I & \\ & & & -t\mathbf{P}/N & I \end{bmatrix} \begin{bmatrix} \mathbf{v}_0 \\ \mathbf{v}_1 \\ \vdots \\ \vdots \\ \mathbf{v}_N \end{bmatrix} = \begin{bmatrix} \mathbf{e}_i \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

2. Apply "push"
3. Analyze work bound

data	F_1		precision		set size		comm
	HK	PR	HK	PR	HK	PR	size
amazon	0.325	0.140	0.244	0.107	193	15293	495
dblp	0.257	0.115	0.208	0.081	44	16026	1429
youtube	0.177	0.136	0.135	0.098	1010	6079	1615
lj	0.131	0.107	0.102	0.086	283	738	662
orkut	0.055	0.044	0.036	0.031	537	1989	4526
friendster	0.078	0.090	0.066	0.075	229	333	724

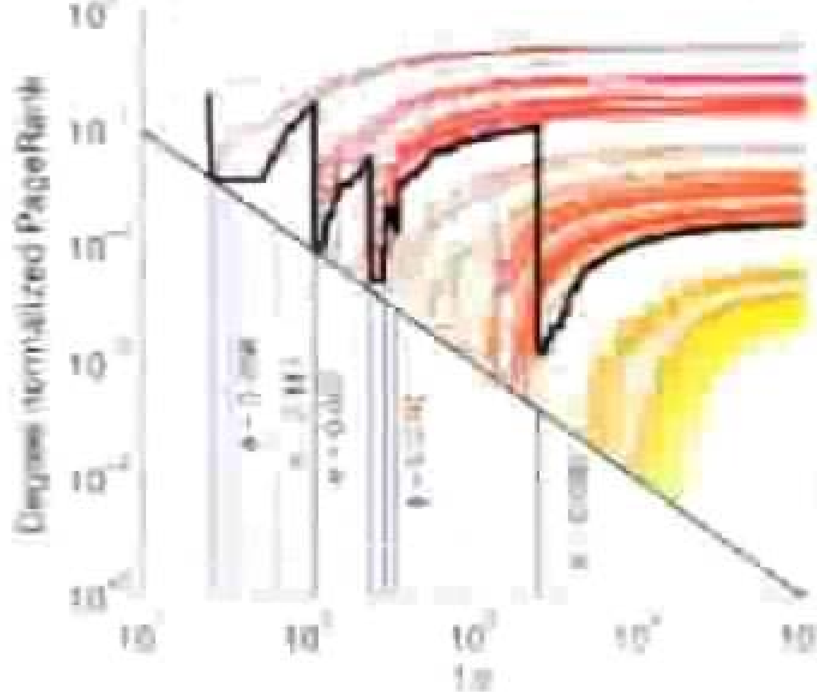


PR achieves high recall by "guessing" a huge set

HK identifies a tighter cluster, so attains better precision

PageRank solution paths

Kloster & Gleich, arXiv:1503.00322



Compute one diffusion, and all sweep-cuts, for all values of epsilon

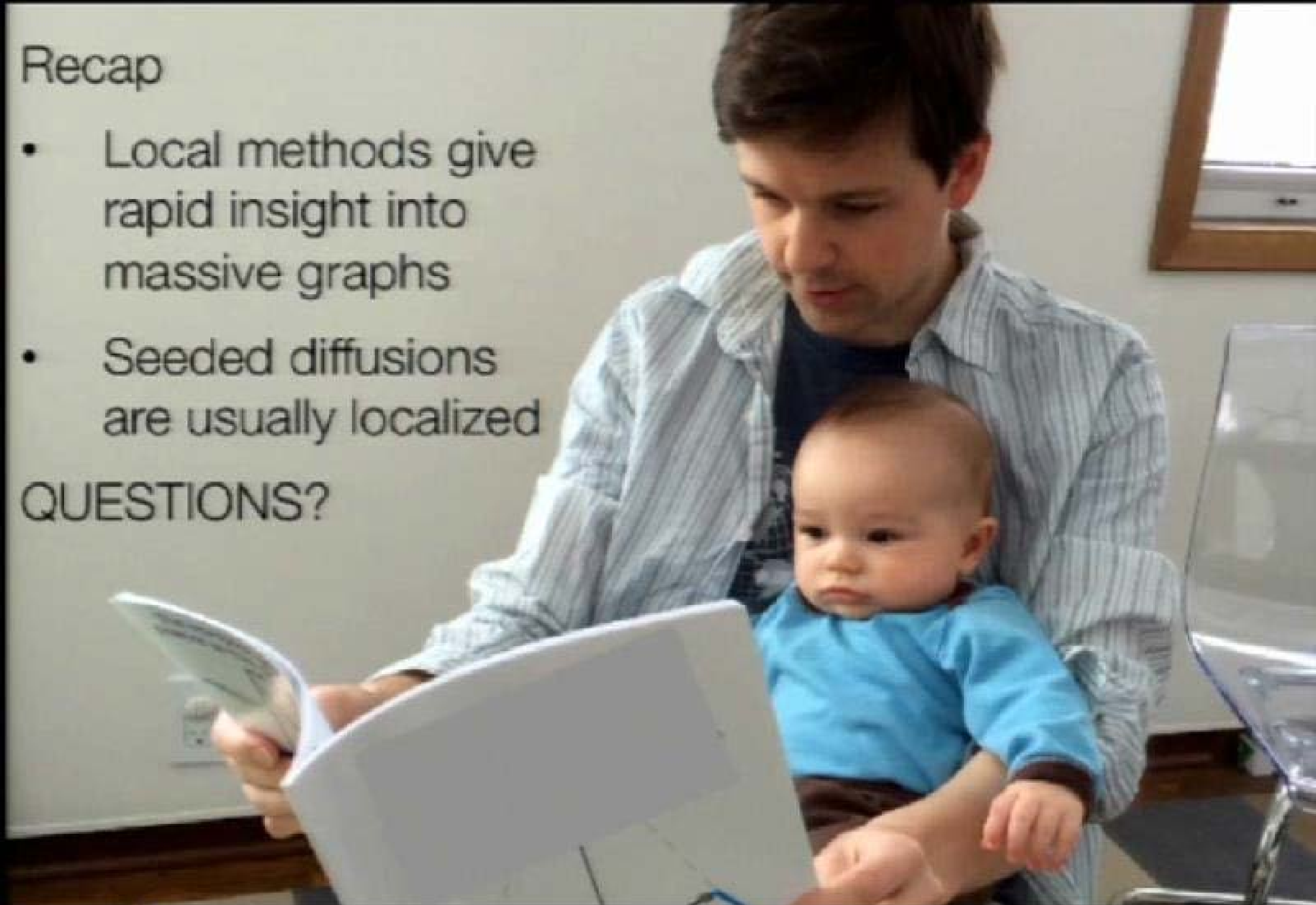
An algorithm to find overlapping communities using local diffusions

1. Extract part of the graph that might have overlapping communities.
2. Compute a partitioning of the network into many pieces (think sqrt(n)) using Graclus.
3. Find the center of these partitions.
4. Use "push" seeded with the egonets of these partitions
5. Add back any missing pieces.

Recap

- Local methods give rapid insight into massive graphs
- Seeded diffusions are usually localized

QUESTIONS?



Supported by NSF CAREER 1149756-CCF