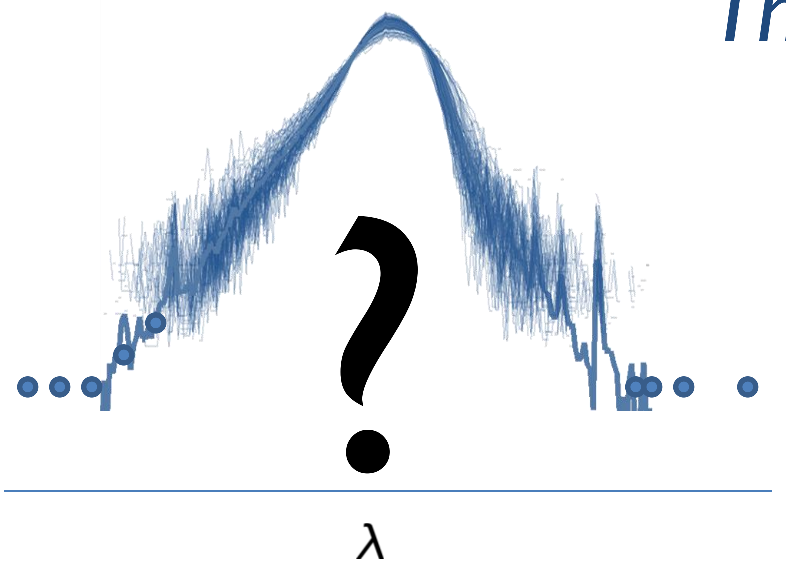
SAND2011-2505P



The Spectre of the Spectrum

An empirical study of the spectra of large networks

David F. Gleich

Sandia National Laboratories

UC Davis

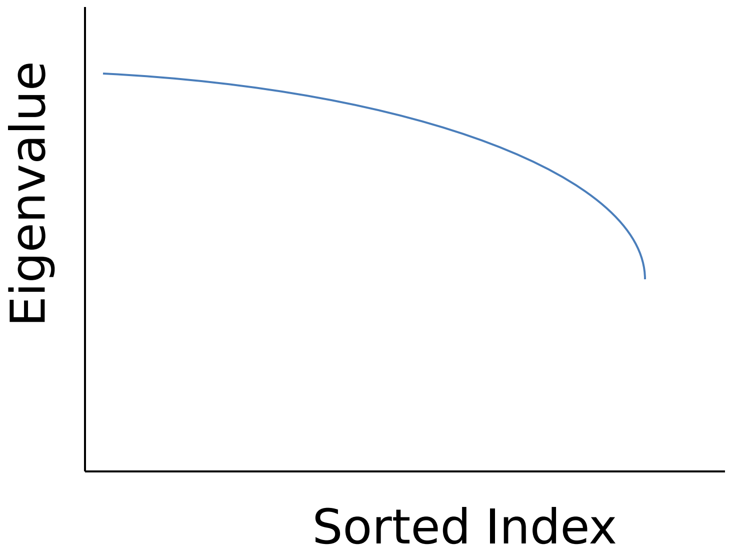
7 April 2011

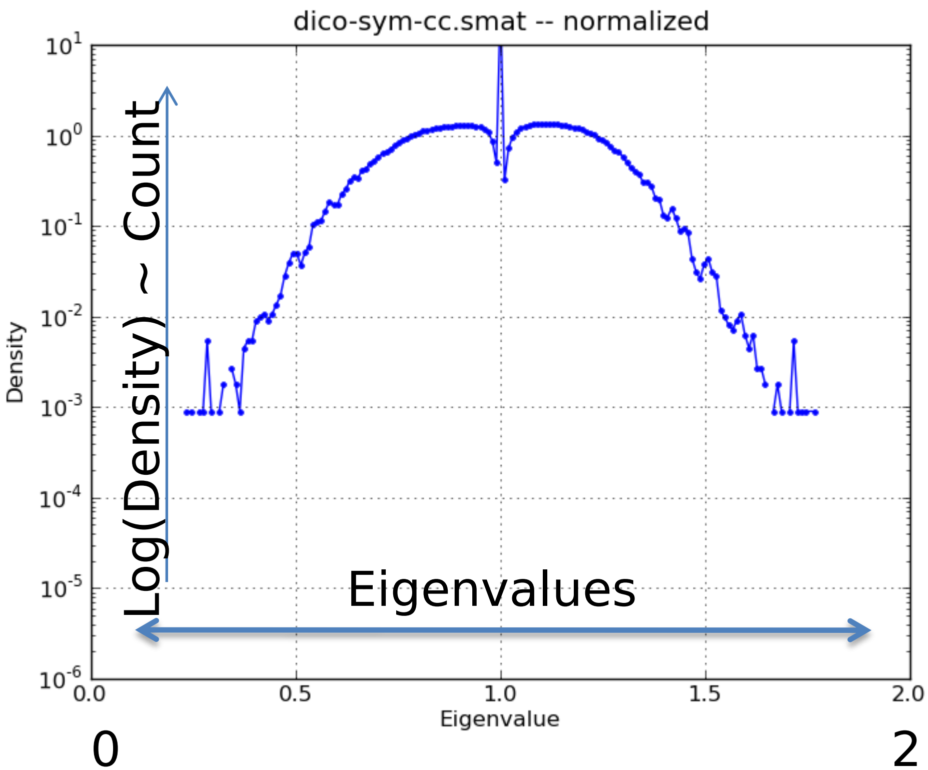
Thanks to Ali Pinar, Jaideep Ray, Tammy Kolda, C. Seshadhri, Rich Lehoucq @ Sandia
and
Jure Leskovec and Michael Mahoney @ Stanford
for helpful discussions.

Supported by Sandia's John von Neumann postdoctoral fellowship and the DOE Office of Science's Graphs project.

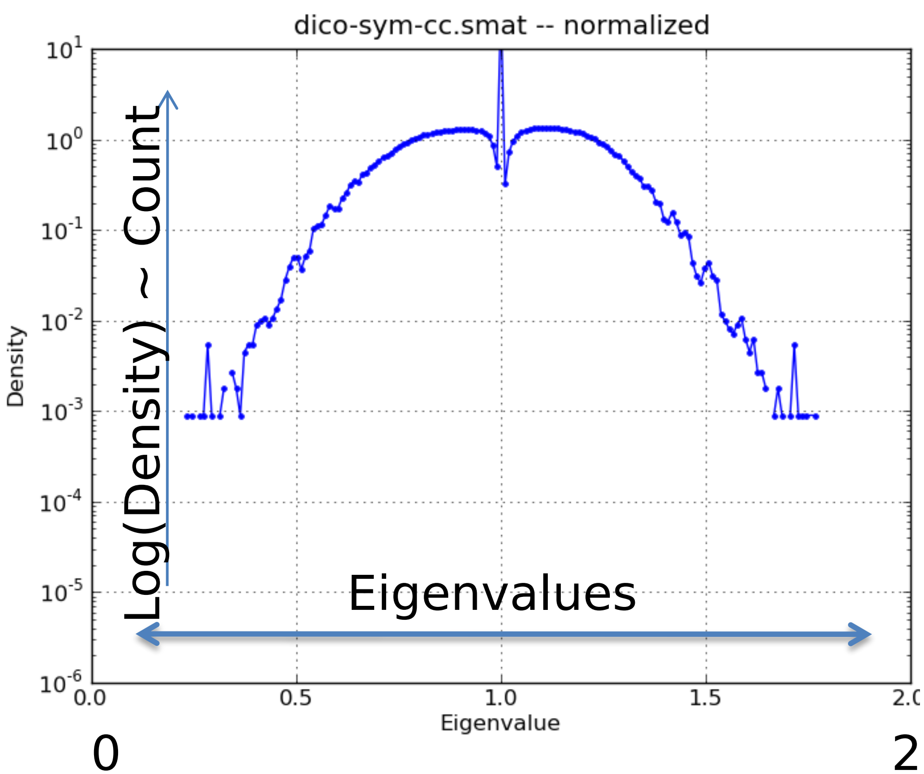
Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

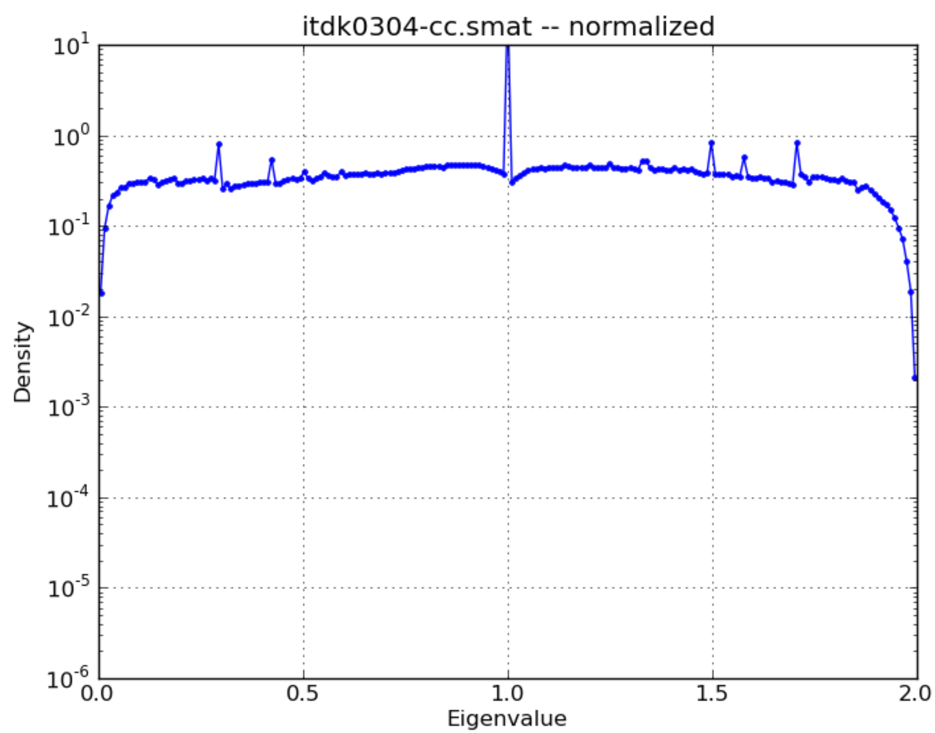
Specgral density plots





There's information inside the spectra



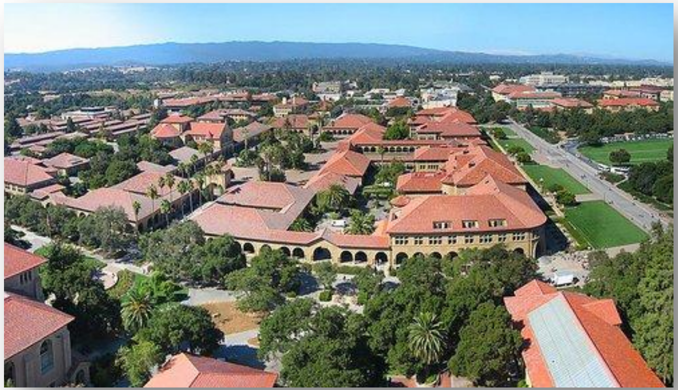


Words in dictionary definitions
111k vertices, 2.7M edges

Internet router network
192k vetices, 1.2M edges

These figures show the normalized Laplacian. Banerjee and Jost (2009) also noted such shapes in the spectra.

Overview



Graphs and their matrices

Data for our experiments

Issues with computing spectra

Many examples of graph spectra

Computing spectra for large networks

~~*Conclusion*~~ Future work

Images taken from Stanford, flickr, and Purdue, respectively

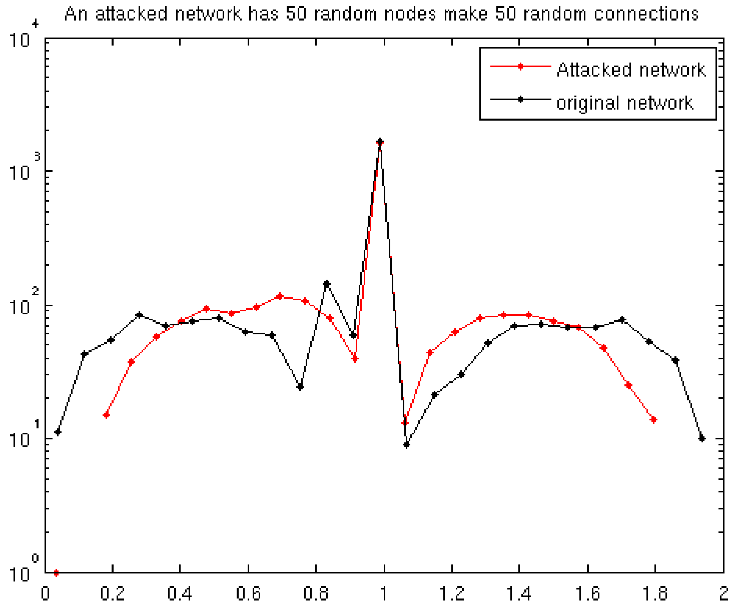
Why are we interested in the spectra?

Modeling

Properties

Moments of the adjacency

Anomalies



Regularities

Network Comparison

Fay et al. 2010 – Weighted Spectral Density

The network is as19971108 from Jure's snap collect (a few thousand nodes) and we insert random connections from 50 nodes

Matrices from graphs

Adjacency matrix

$\boldsymbol{A} : n \times n, \boldsymbol{A} = \boldsymbol{A}^T$

$A_{i,j} = 1$ if $(i,j) \in E$

$-d_{\max} \le \lambda(\boldsymbol{A}) \le d_{\max}$

Laplacian matrix

$\boldsymbol{D} = \operatorname{diag}(\boldsymbol{A}\mathbf{e})$

$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$

$0 \le \lambda(\boldsymbol{L}) \le 2d_{\max}$

Normalized Laplacian matrix

$\tilde{\boldsymbol{L}} = \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$

$0 \le \lambda(\tilde{\boldsymbol{L}}) \le 2$

Random walk matrix
$\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$

Modularity matrix
$\mathbf{d} = \boldsymbol{A}\mathbf{e}$
$\boldsymbol{M} = \boldsymbol{A} - 1/(2|E|)\mathbf{d}\mathbf{d}^T$

Not covered
Signless Laplacian matrix

Incidence matrix
(It is incidentally discussed)

Seidel matrix

Heat Kernel

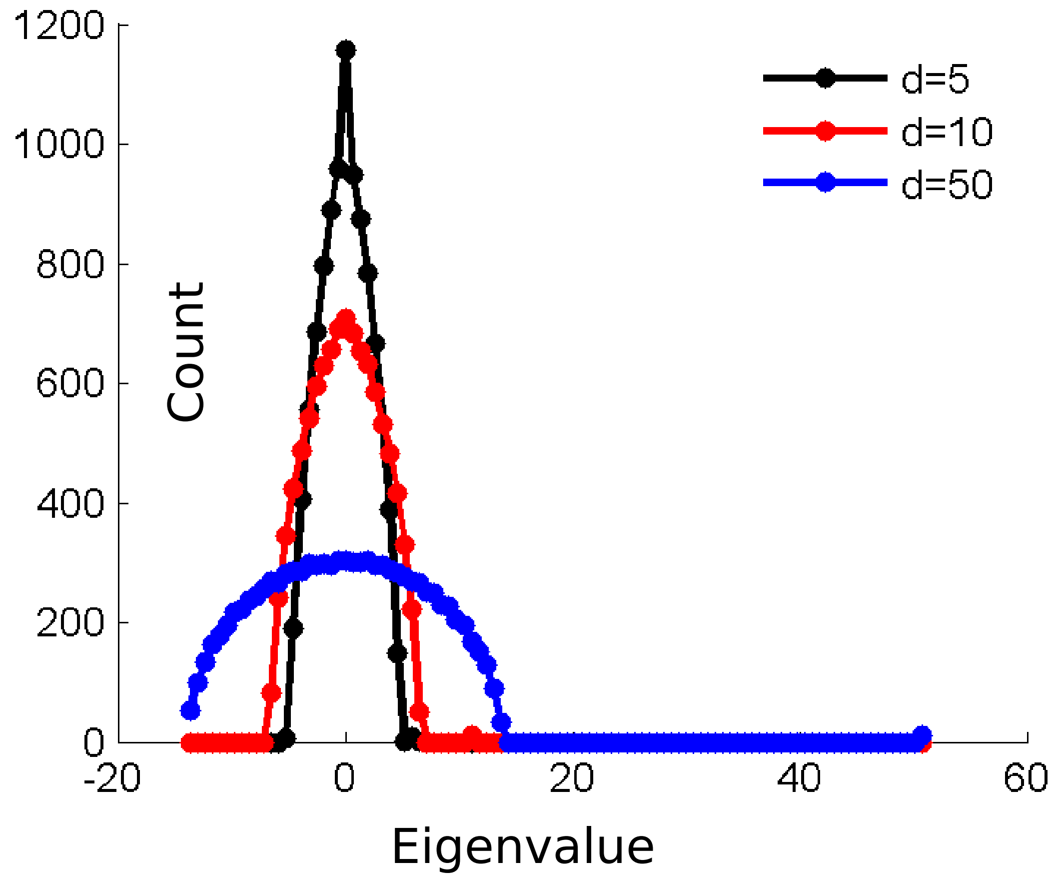
Everything is undirected. Mostly connected components only too.

Erdős–Rényi Semi-circles

Based on Wigner's semi-circle law.

The eigenvalues of the adjacency matrix for n=1000, averaged over 10 trials

Semi-circle with outlier if average degree is large enough.



Observed by Farkas and in the book "Network Alignment" edited by Brandes (Chapter 14)

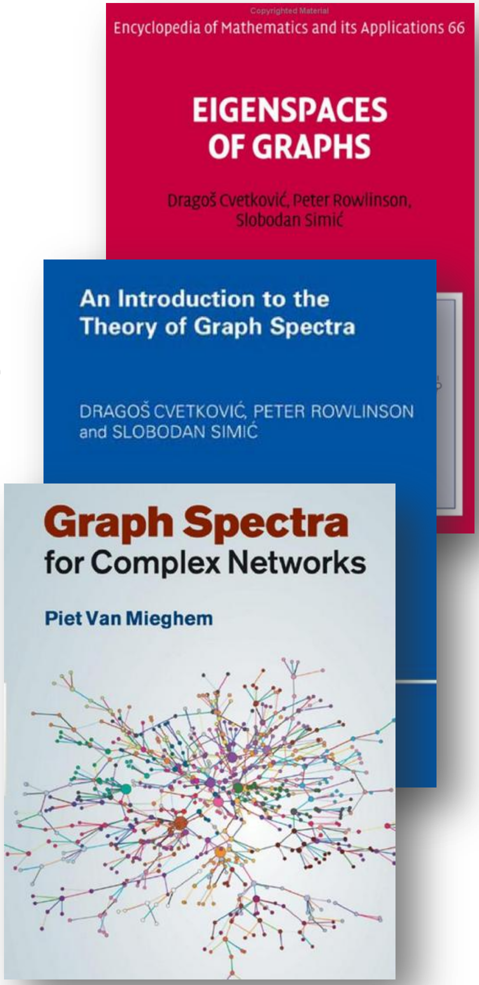
Previous results

Farkas et al. Significant deviation from the semi-circle law for the adjacency matrix

Mihail and Papadimitriou Leading eigenvalues of the adjacency matrix obey a power-law based on the degree-sequence

Chung et al. Normalized Laplacian still obeys a semi-circle law if min-degree large

Banerjee and Jost Study of types of patterns that emerge in evolving graph models – explain many features of the spectra



In comparison to other empiric studies

We use "exact" computation of spectra,
instead of approximation.

We study "all" of the standard matrices
over a range of large networks.

Our "large" is bigger.

We look at a few random graph models

- preferential attachment
- random powerlaw
- copying model
- forest fire model

ISSUES WITH COMPUTING SPECTRA

Why you should be very careful with eigenvalues.

Matlab!

Always a great starting point.
    My desktop has 24GB of RAM (less than $2500 now!)

24GB/8 bytes (per double) = 3 billion numbers
    ~ 50,000-by-50,000 matrix

Possibilities

    D = eig(A) – needs twice the memory for A,D
    [V,D] = eig(A) – needs three times the memory for A,D,V

These limit us to ~38000 and ~31000 respectively.

Bugs – Matlab

eig(A)

Returns incorrect eigenvectors

Seems to be the result of a bug in Intel's MKL library.

Bug – ScaLAPACK default

sudo apt-get install scalapack-openmpi

Allocate 36000x36000 local matrix

Run on 4 processors

Code crashes

Bug – LAPACK

Scalapack MRRR

Compare standard lapack/blas to atlas performance

Result: correct output from atlas

Result: incorrect output from lapack

Hypothesis: lapack contains a known bug that's apparently in the default ubuntu lapack

Moral

Always test your software.

Extensively.

COMPUTING SPECTRA OF LARGE NETWORKS

(SUPER)-COMPUTERS MORE LATER

EXAMPLES

Data sources

SNAP	Various	100s-100,000s
SNAP-p2p	Gnutella Network	5-60k, ~30 inst.
SNAP-as-733	Autonomous Sys.	~5,000, 733 inst.
SNAP-caida	Router networks	~20,000, ~125 inst.
Pajek	Various	100s-100,000s
Models	Copying Model	1k-100k 9 inst. 324 gs
	Pref. Attach	1k-100k 9 inst. 164 gs
	Forest Fire	1k-100k 9 inst. 324 gs
Mine	Various	2k-500k
Newman	Various	
Arenas	Various	
Porter	Facebook	100 schools, 5k-60k
IsoRank, Natalie	Protein-Protein	<10k , 4 graphs

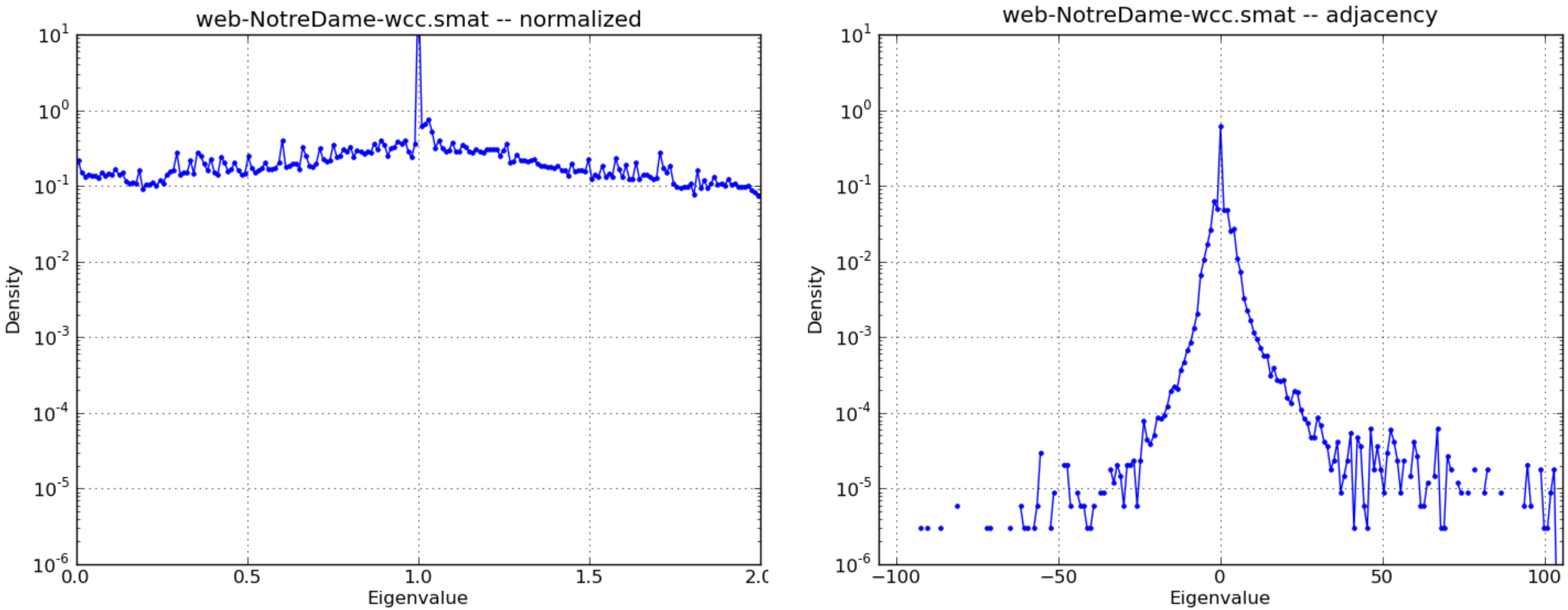
Thanks to all who make data available

Big graphs

Arxiv	86376	1035126	Co-author
Dblp	93156	356290	Co-author
Dictionary(*)	111982	2750576	Word defns.
Internet(*)	124651	414428	Routers
Itdk0304	190914	1215220	Routers
p2p-gnu(*)	62561	295756	Peer-to-peer
Patents(*)	230686	1109898	Citations
Roads	126146	323900	Roads
Wordnet(*)	75606	240036	Word relation
web-nb.edu	325729	2994268	Web

(*) denotes that this is a weakly connected component of a directed graph.

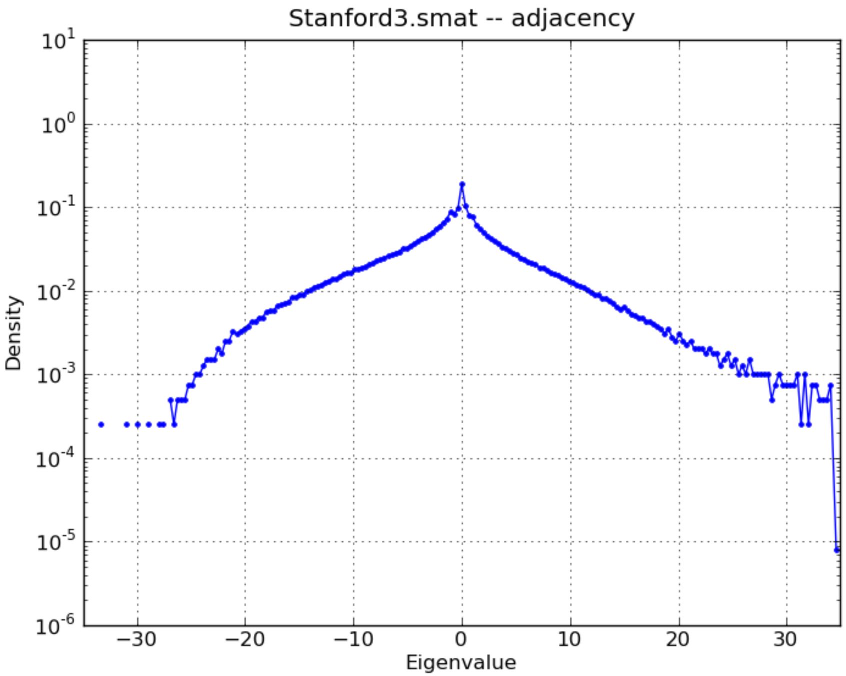
A $8,000 matrix computation

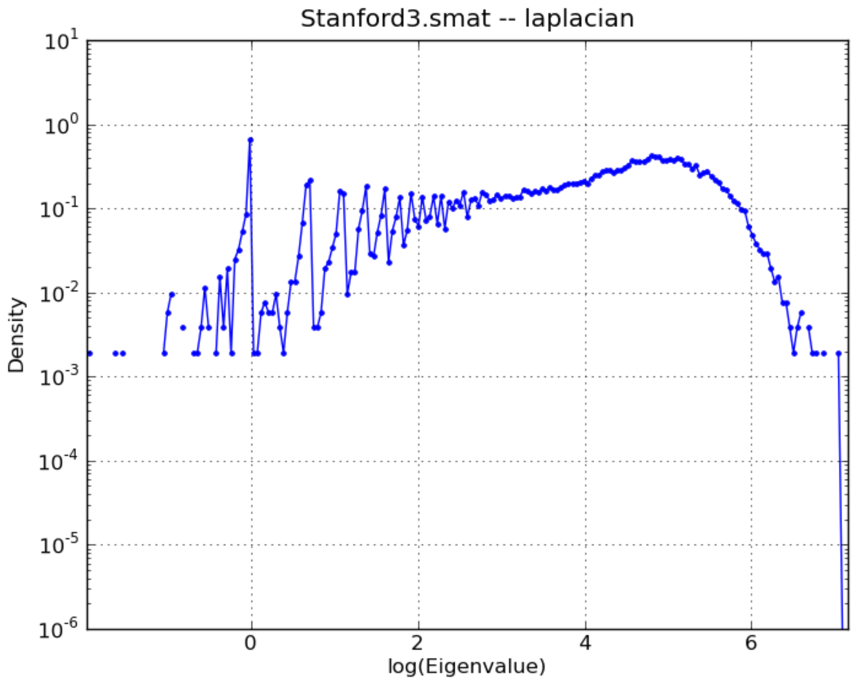


325729 nodes and 2994268 edges

500 nodes and 4000 processors on Redsky for 5 hours x 2 for normalized Laplacian/adjacency matrix

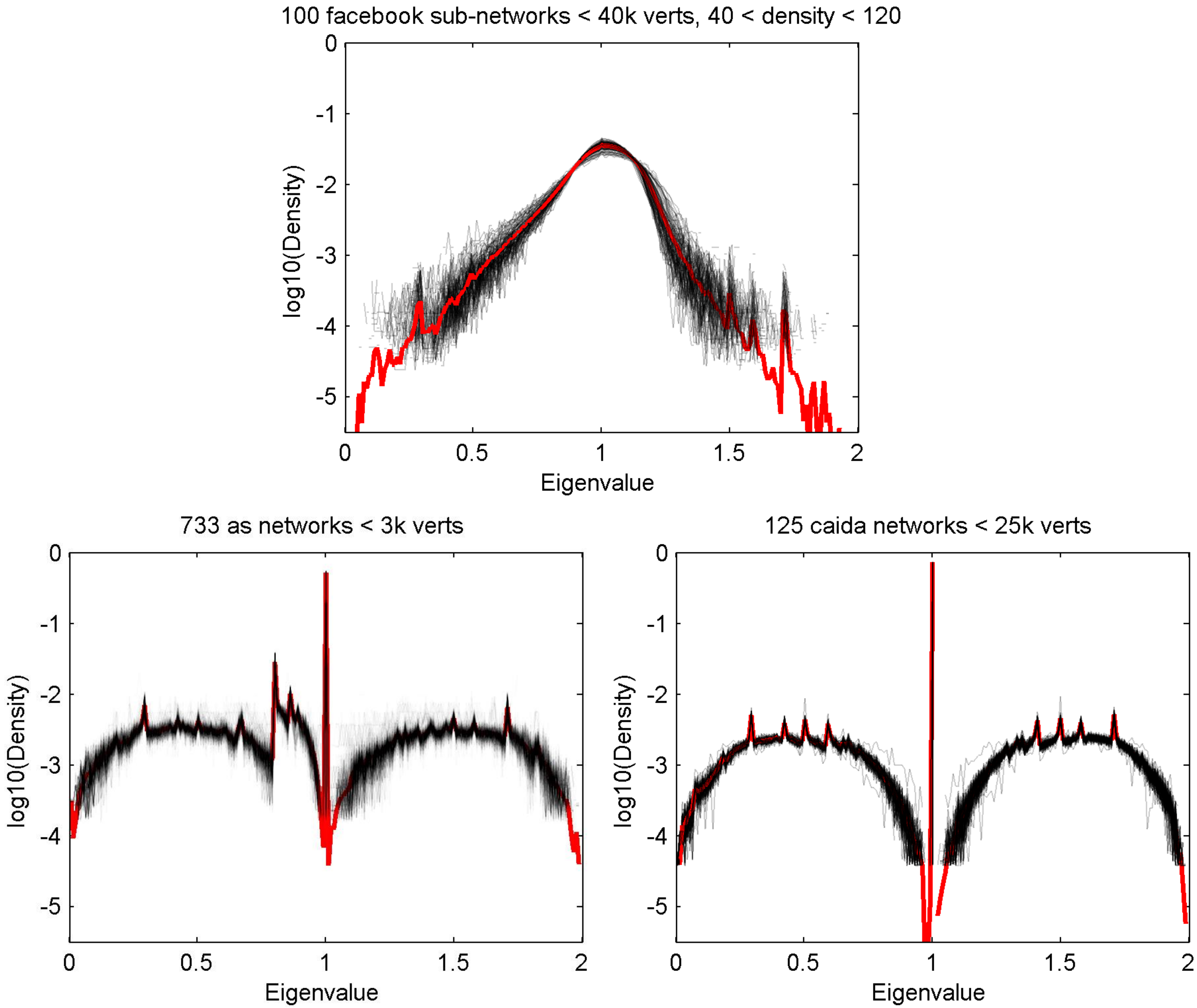
Stanford's Facebook Network





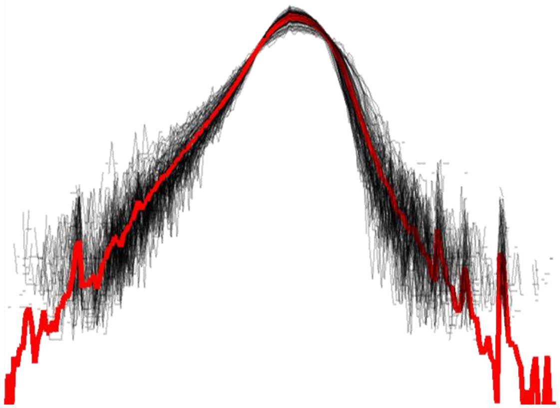
Data from Mason Porter. Aka, the start of a $50,000,000,000 graph.

Stability? Yes!



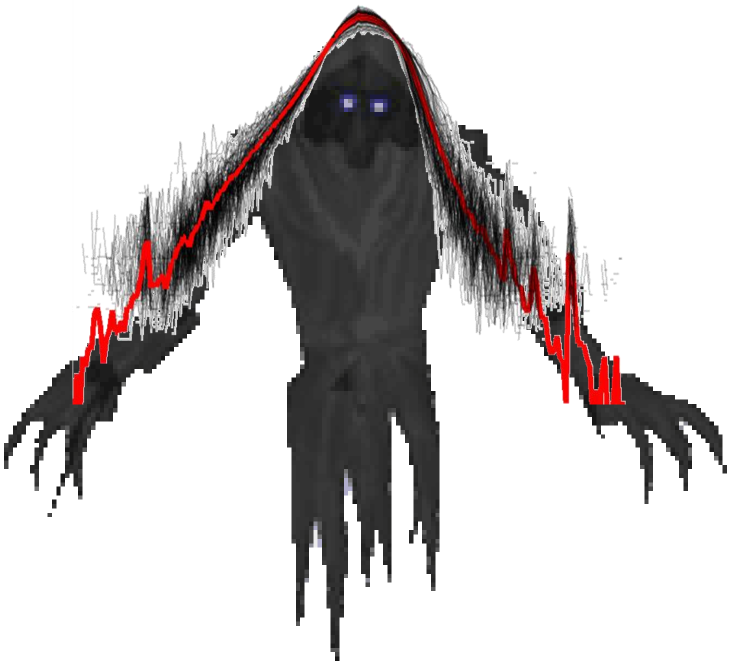
These are cases where we have multiple instances of the same graph.

Already known?



Just the facebook spectra.

Already known?



I soon realized I was searching for "spectre" instead of spectrum, oops.

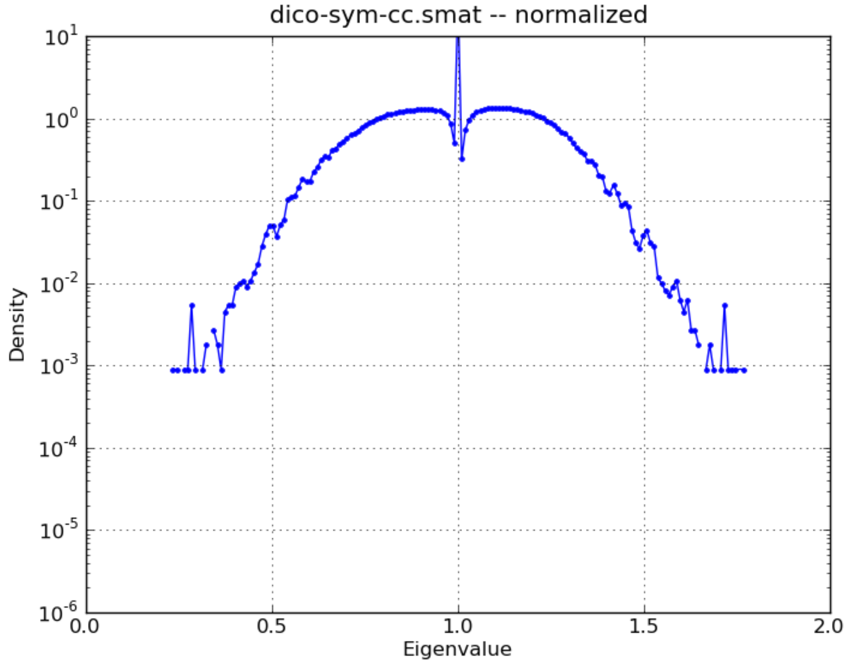
Spikes?

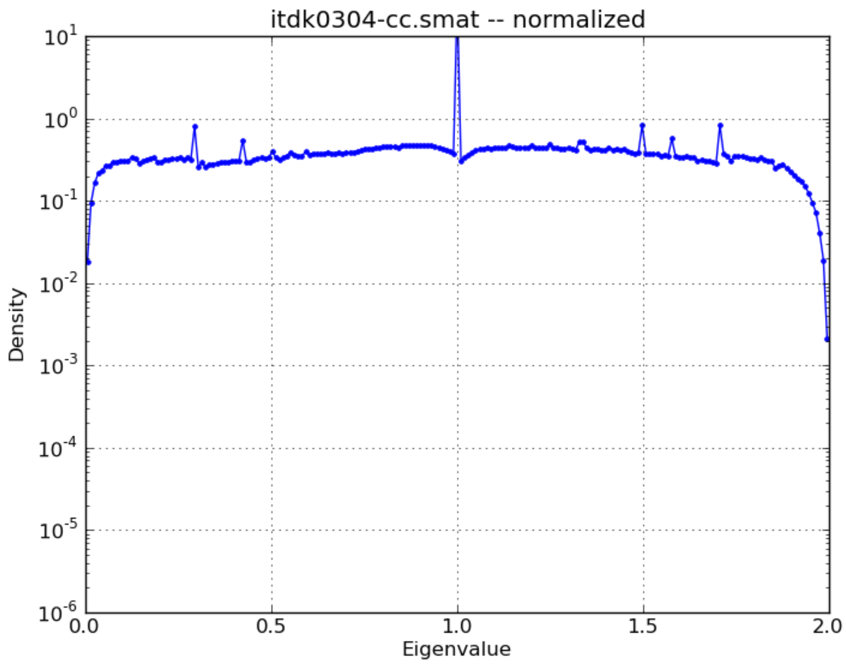
Unit eigenvalue

$(\boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{A})\mathbf{x} = \mathbf{x} \Rightarrow \boldsymbol{A}\mathbf{x} = 0$

Repeated rows
Identical rows grow the null-space.

Banerjee and Jost
Motif doubling and joining small graphs will tend to cause repeated eigenvalues and null vectors.



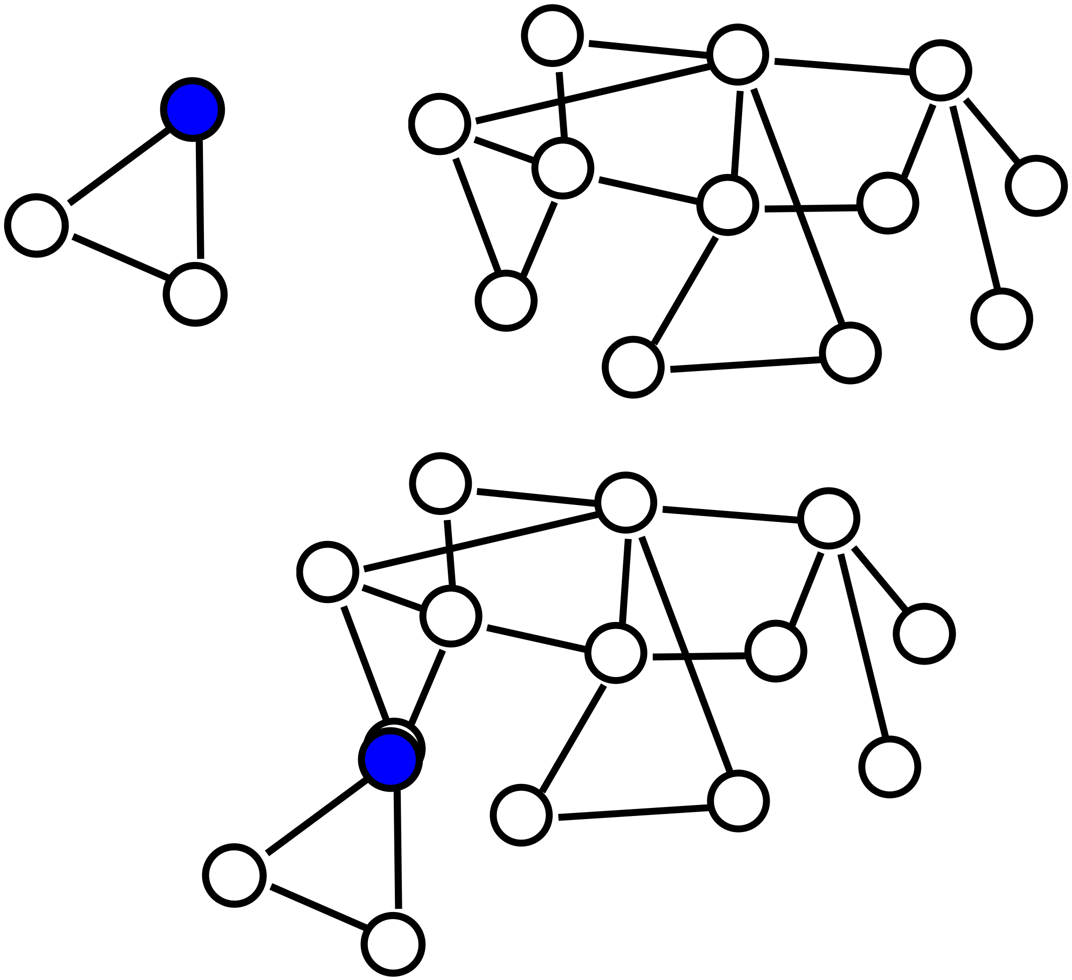


Banerjee and Jost explained how evolving graphs should produce repeated eigenvalues

Combining Eigenvalues

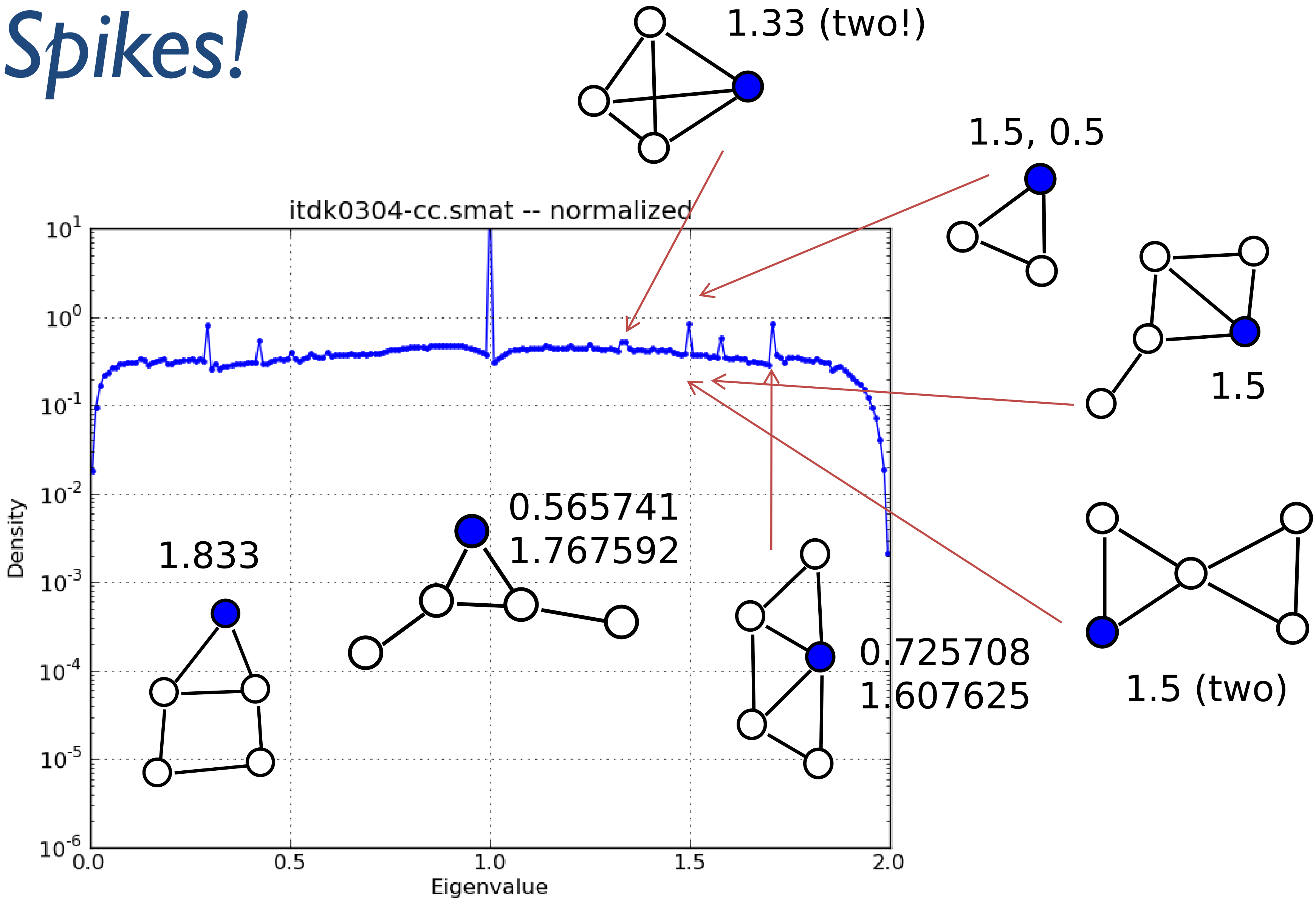
If A has an eigenvector with a zero component, then

“A + B” (as in the figure) has the same eigenvalue with eigenvector extended with zeros on B.



Bannerjee and Jost observed this for the normalized Laplacian.

Spikes!

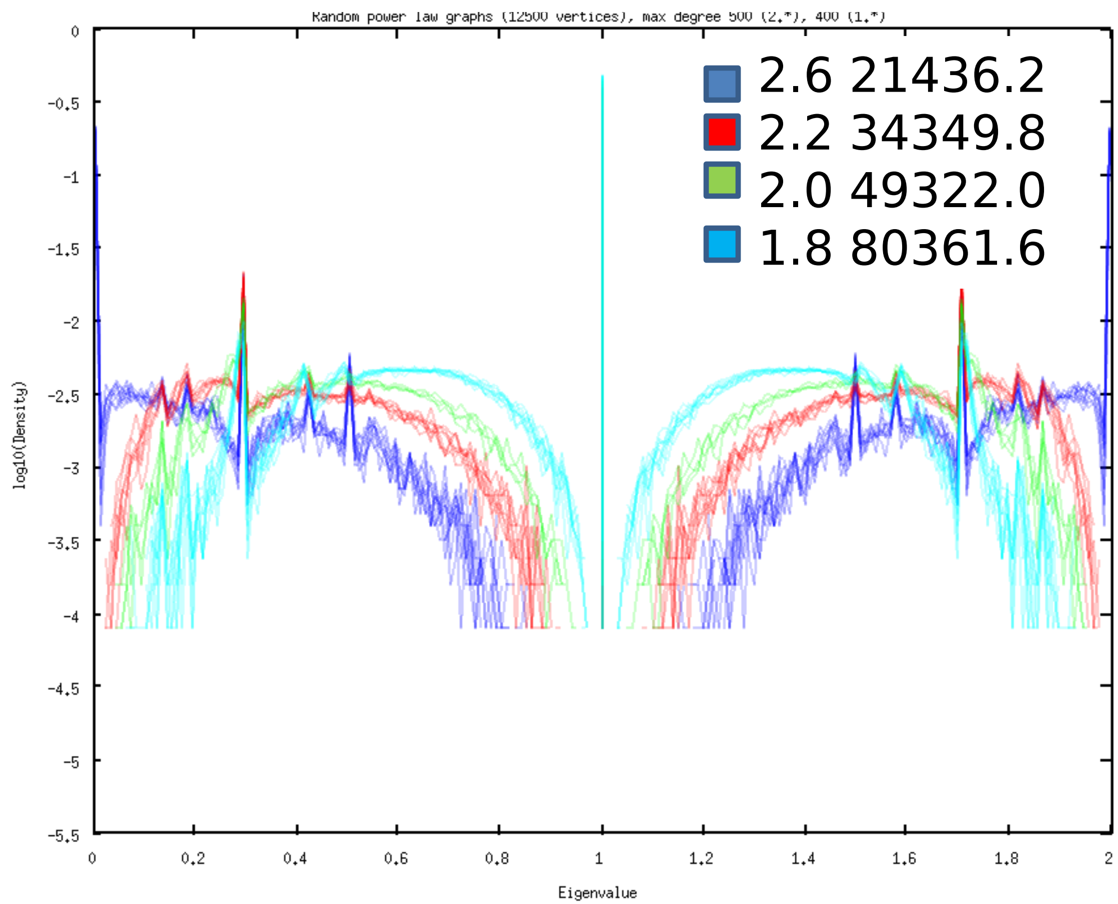

1.33 (two!)
1.5, 0.5
itdk0304-cc.smat -- normalized
1.5
1.5 (two)
0.565741
1.767592
1.833
0.725708
1.607625
Density
Eigenvalue
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
0.0
0.5
1.0
1.5
2.0

Random power law

Generate a power law degree distribution.

Produce a random graph with a prescribed degree distribution using the Bayati-Kim-Saberi procedure.

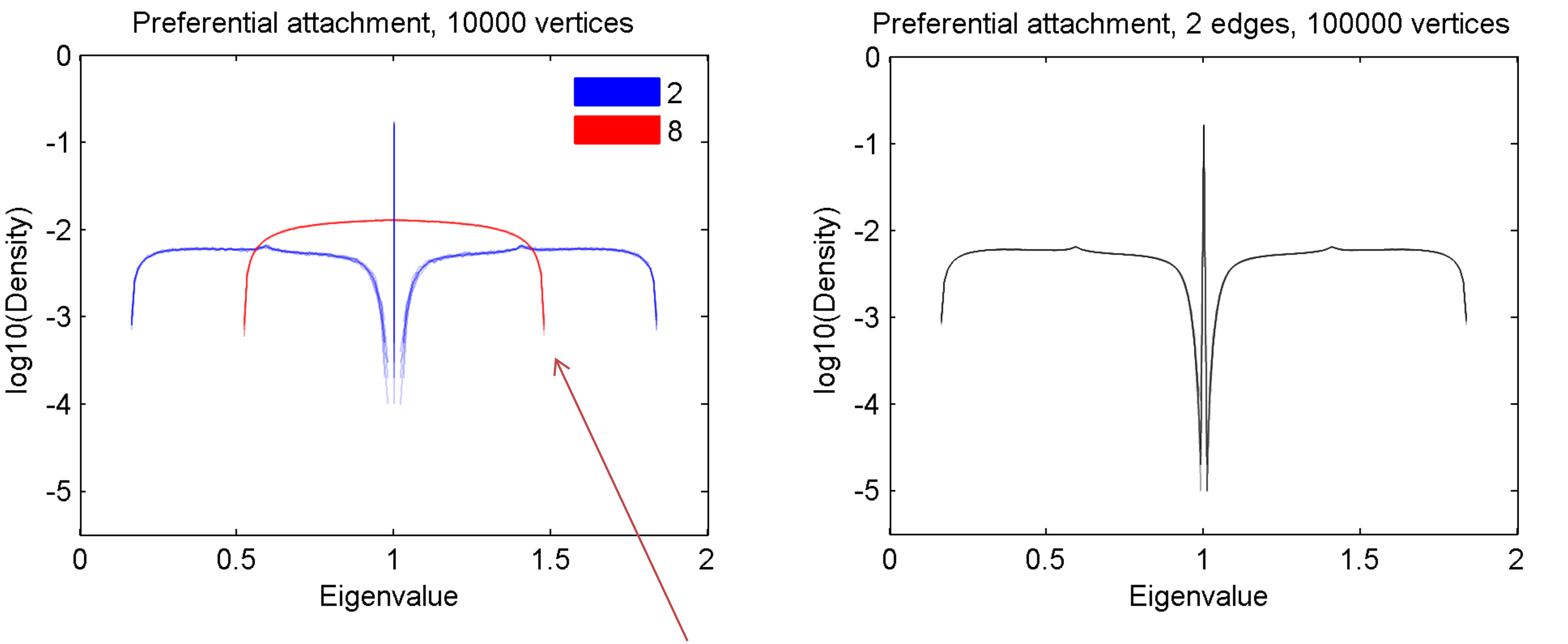
Random power law
12500 vertices, 500 (2.*) /400 (1.8) min degree



Bad figure. Matlab was only producing nasty output this morning! My apologies

Preferential Attachment

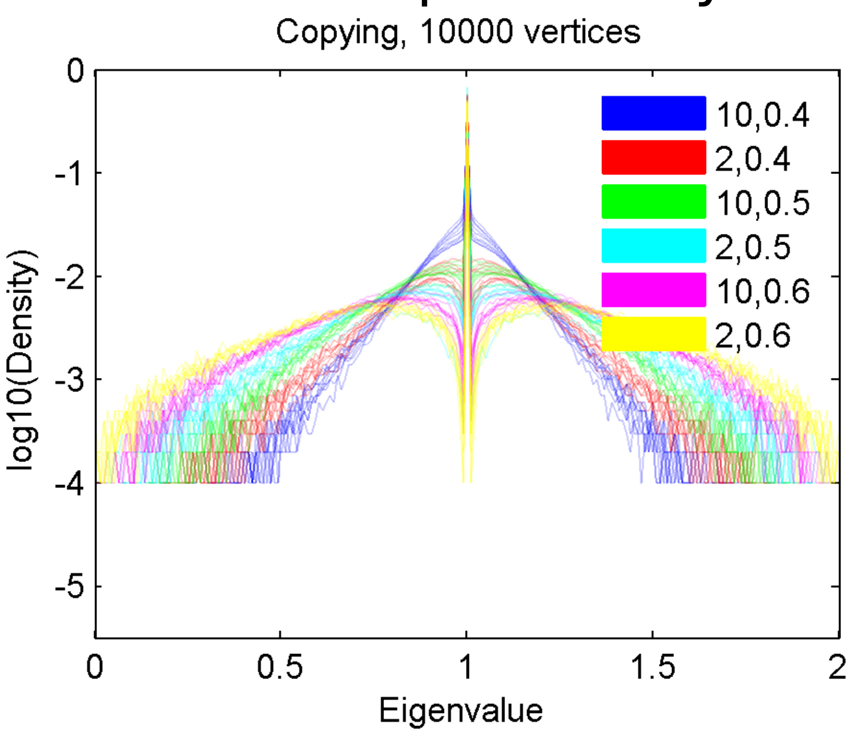
Start graph with a k-node clique. Add a new node and connect to k random nodes, chosen proportional to degree.

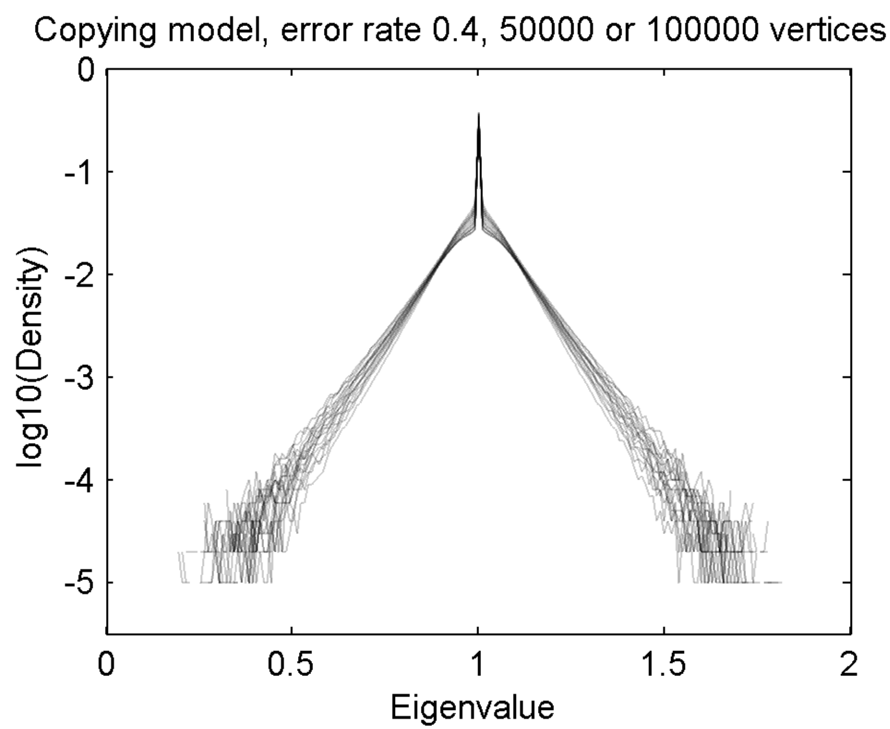


Semi-circle in log-space!

Copying model

Start graph with a k-node clique. Add a new node and pick a parent uniformly at random. Copy edges of parent and make an error with probability α

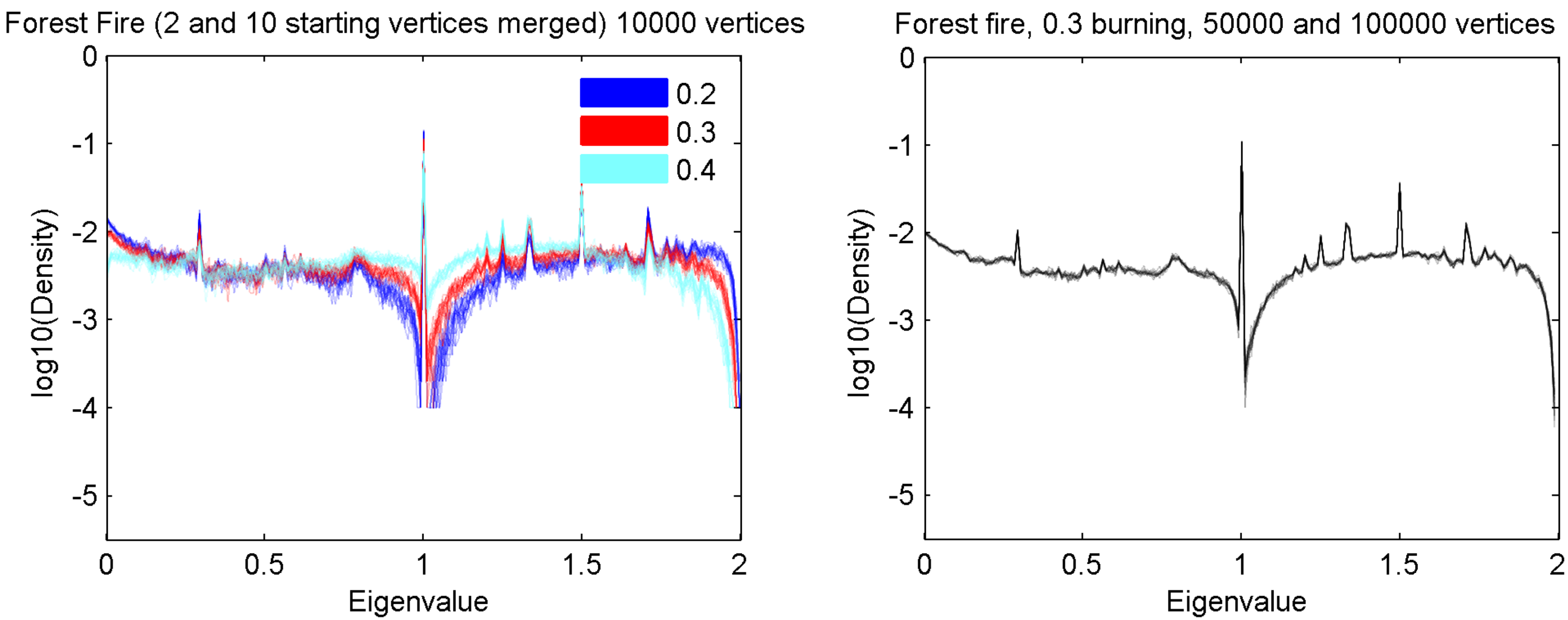




Obvious follow up here: does a random sample with the same degree distribution show the same thing?

Forest Fire models

Start graph with a k-node clique. Add a new node and pick a parent uniformly at random. Do a random “bfs’/”forest fire” and link to all nodes “burned”



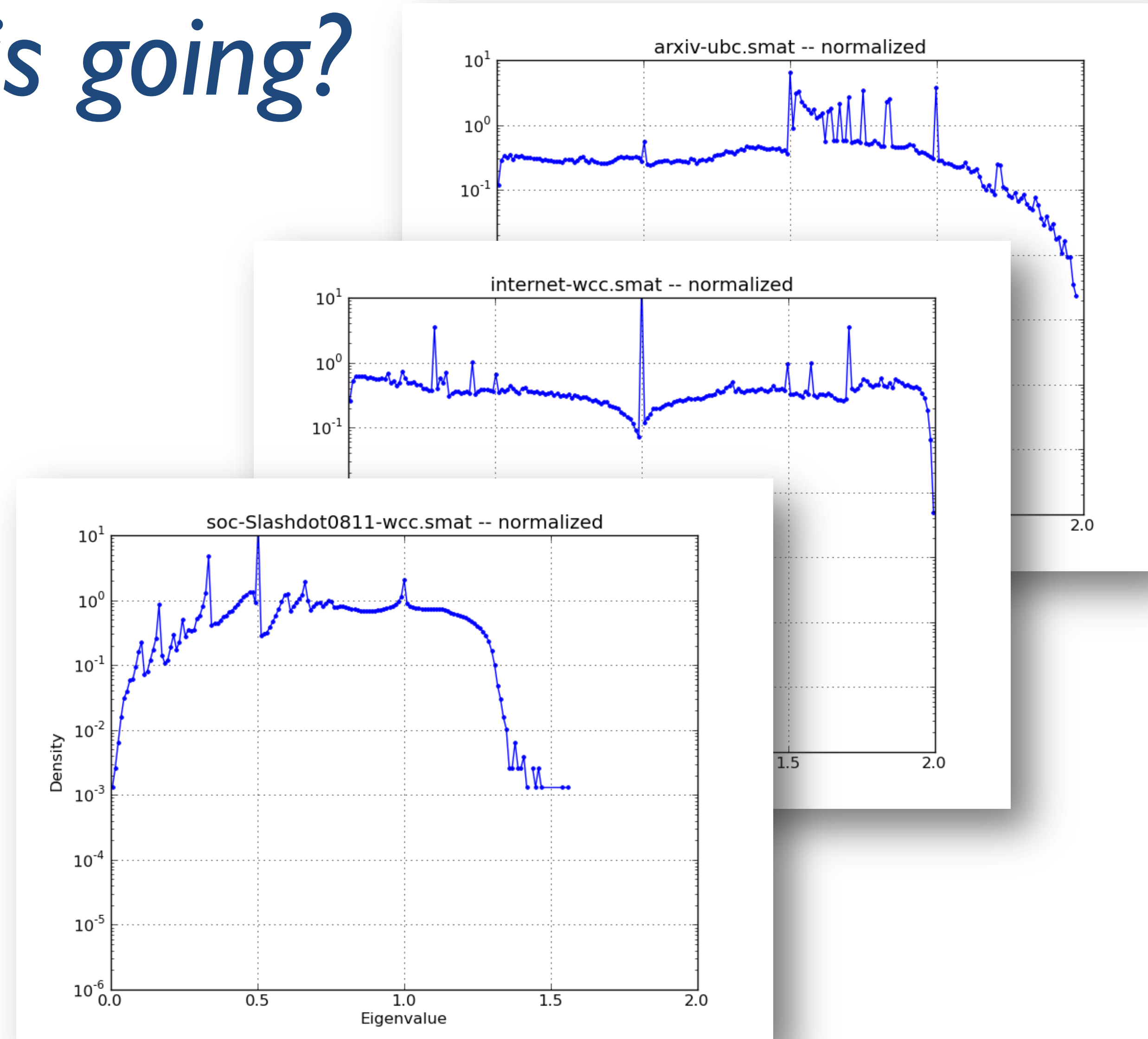
Where is this going?

We can compute spectra for large networks if needed.

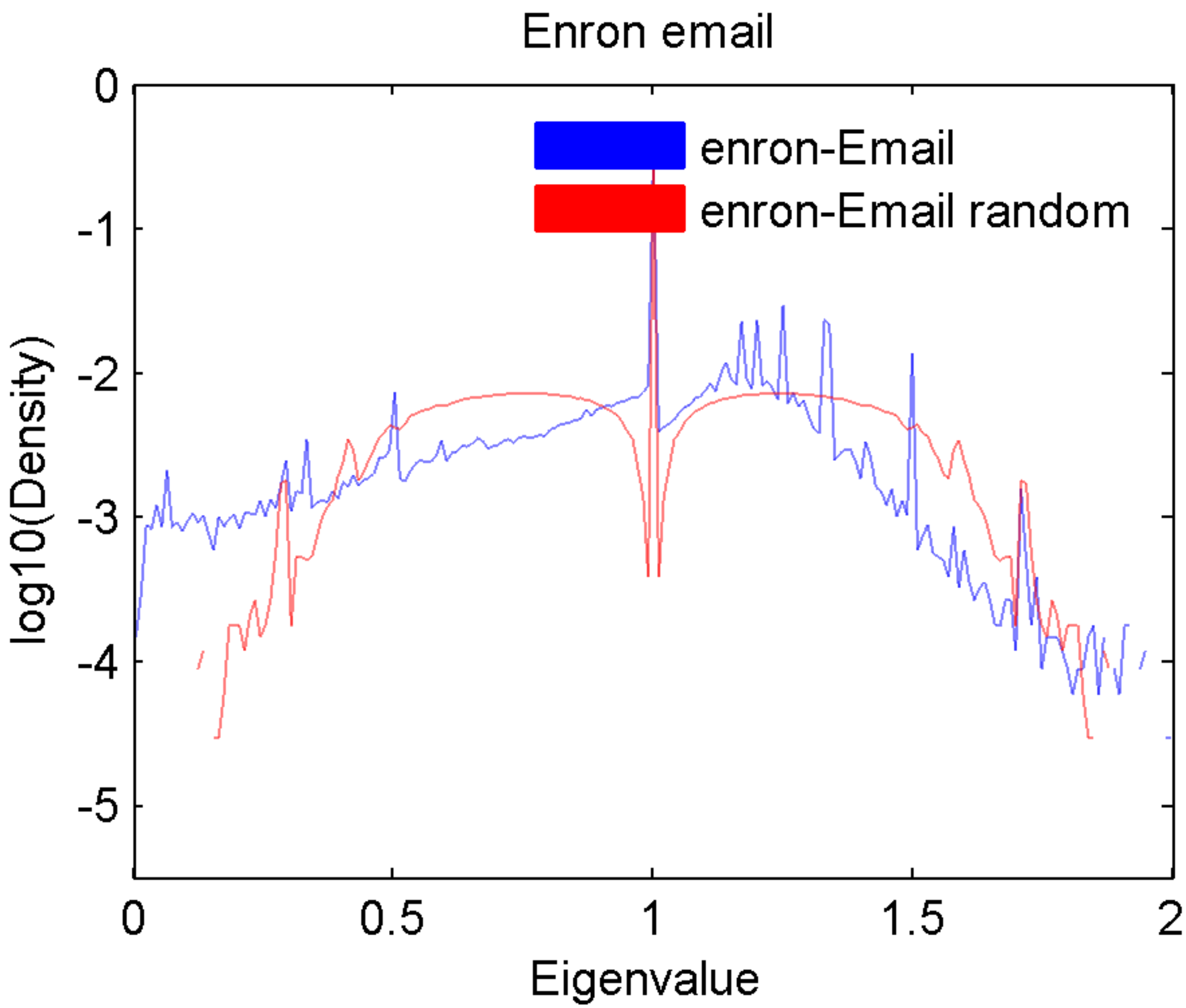
Study relationship with known power-laws in spectra

Eigenvector localization

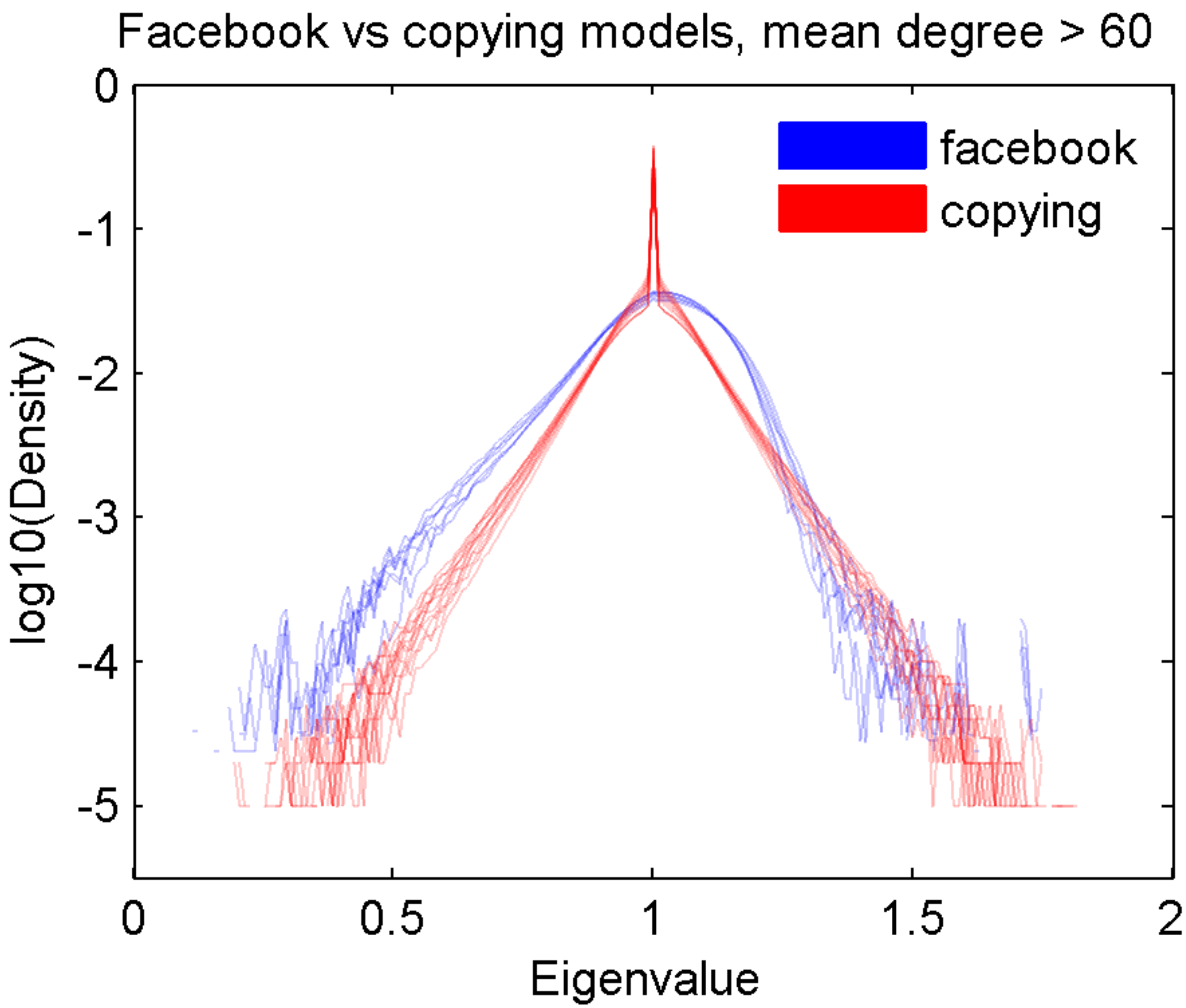
Directed Laplacians



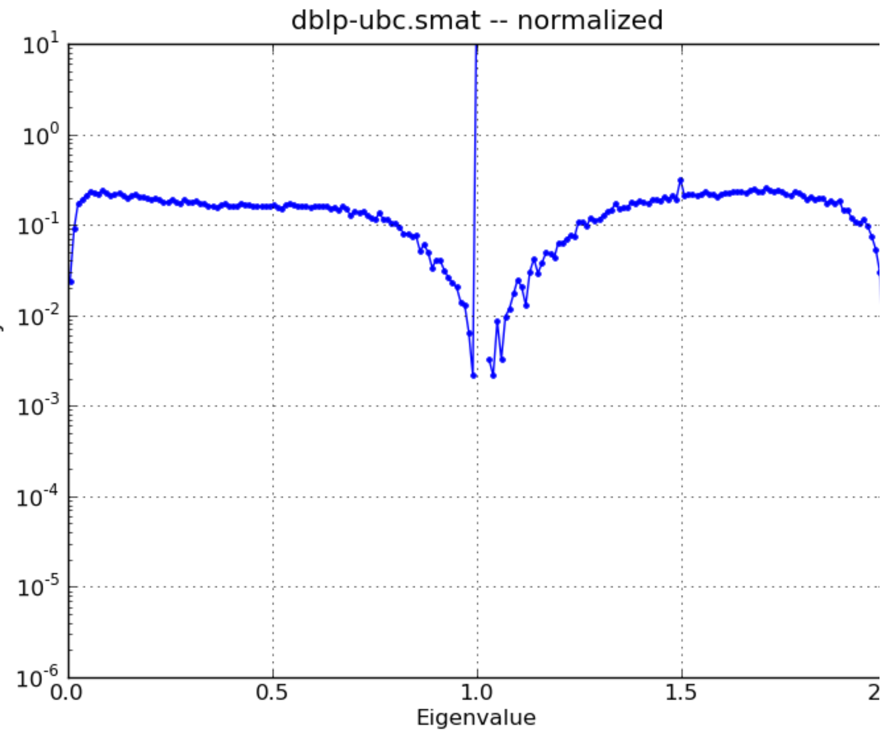
Just the degree distribution? No

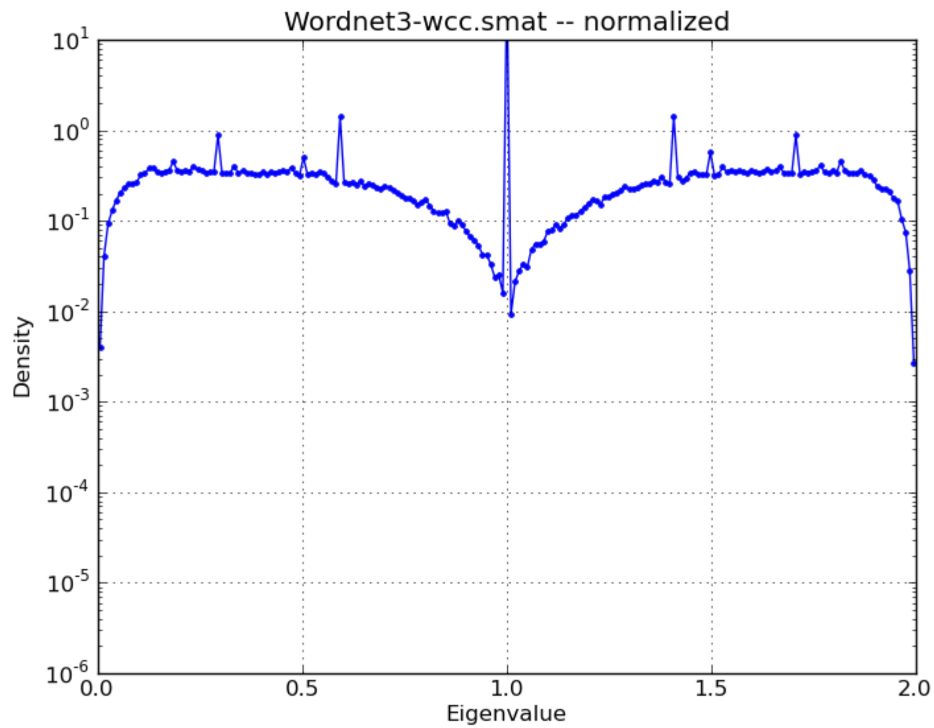


Facebook is not a copying model



Same density

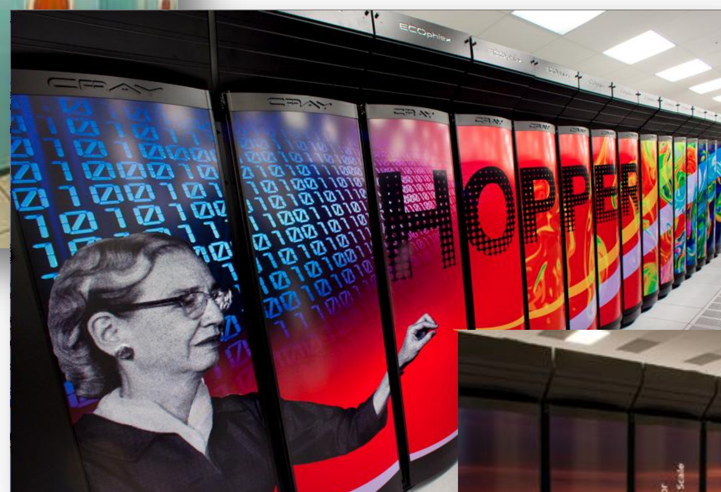




Both have a mean degree of 3.8

COMPUTING SPECTRA OF LARGE NETWORKS

(Super)-Computers

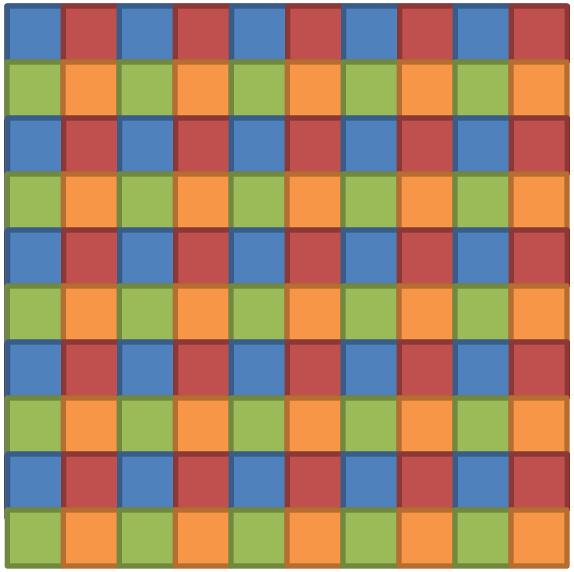


Redsky, Hopper I, Hopper II, and a Cielo testbed. Details if time.

Eigenvalues with ScaLAPACK

Mostly the same approach as in LAPACK

1. Reduce to tridiagonal form (most time consuming part)
2. Distribute tridiagonals to all processors
3. Each processor finds all eigenvalues
4. Each processor computes a subset of eigenvectors

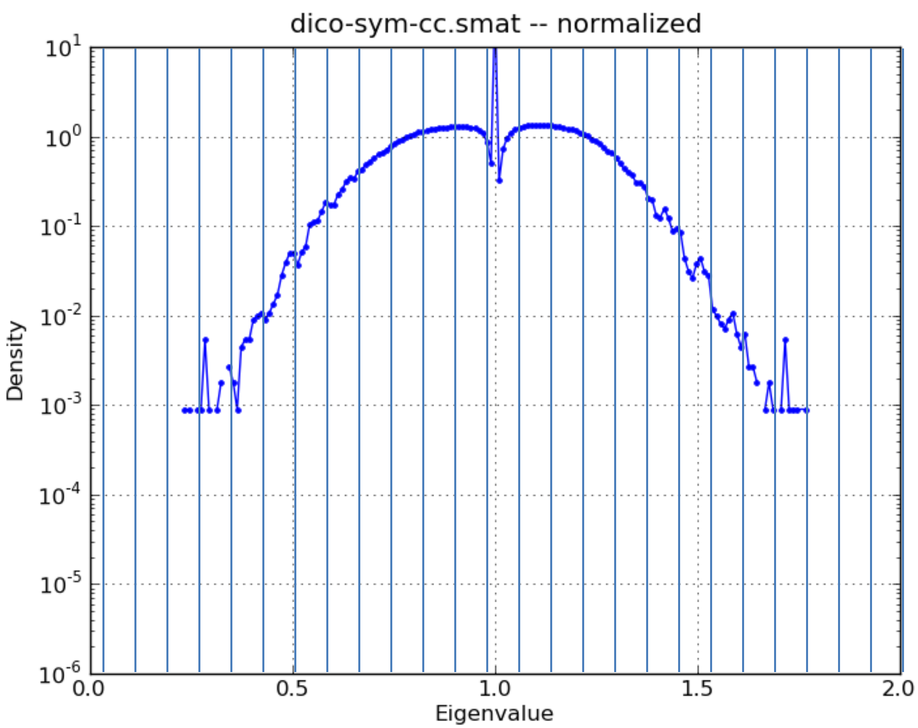


ScaLAPACK's 2d block cyclic storage

I'm actually using the **MRRR algorithm**, where steps 3 and 4 are better and faster

MRRR due to Parlett and Dhillon; implemented in ScaLAPACK by Christof Vomel.

Estimating the density directly



$\boldsymbol{A}$ and $\boldsymbol{F}^T\boldsymbol{A}\boldsymbol{F}$ have the same eigenvalue inertia if $\boldsymbol{F}$ is non-singular.

Eigenvalue inertia = (p,n,z)
Positive eigenvalues
Negative eigenvalues
Zero eigenvalues
If $\boldsymbol{F}^T\boldsymbol{A}\boldsymbol{F}$ is diagonal, inertia is easy to compute

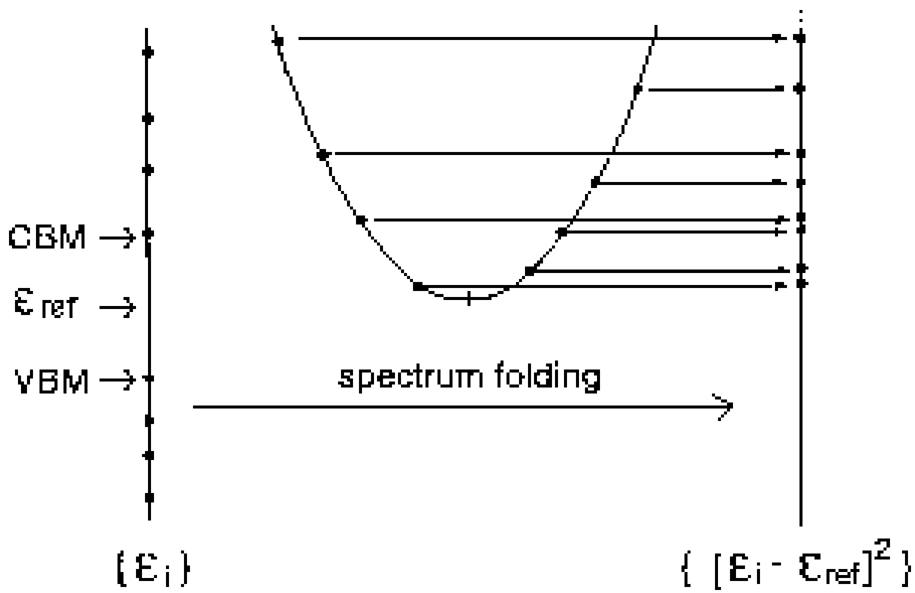
$\tilde{\boldsymbol{L}}$ has inertia (n-1,0,1)
$\tilde{\boldsymbol{L}} - \lambda_e \boldsymbol{I}$ has inertia (sum($\lambda > \lambda_e$), sum($\lambda < \lambda_e$), …)

This is an old trick in linear algebra. I know that Fay et al. used it in their weighted spectral density. You could use this to check me!

Alternatives

Use ARPACK to get extrema

Use ARPACK to get interior around λ_0 via the folded spectrum $((\boldsymbol{A} - \lambda_0)^2)^k$



Large nearly repeated sets of eigenvalues will make this tricky.

Farkas et al. used this approach. Figure from somewhere on the web… sorry!

Adding MPI tasks vs. using threads

Most math libraries have threaded versions
(Intel MKL, AMD ACML)

Is it better to use threads or MPI tasks?

It depends.

Intel MKL

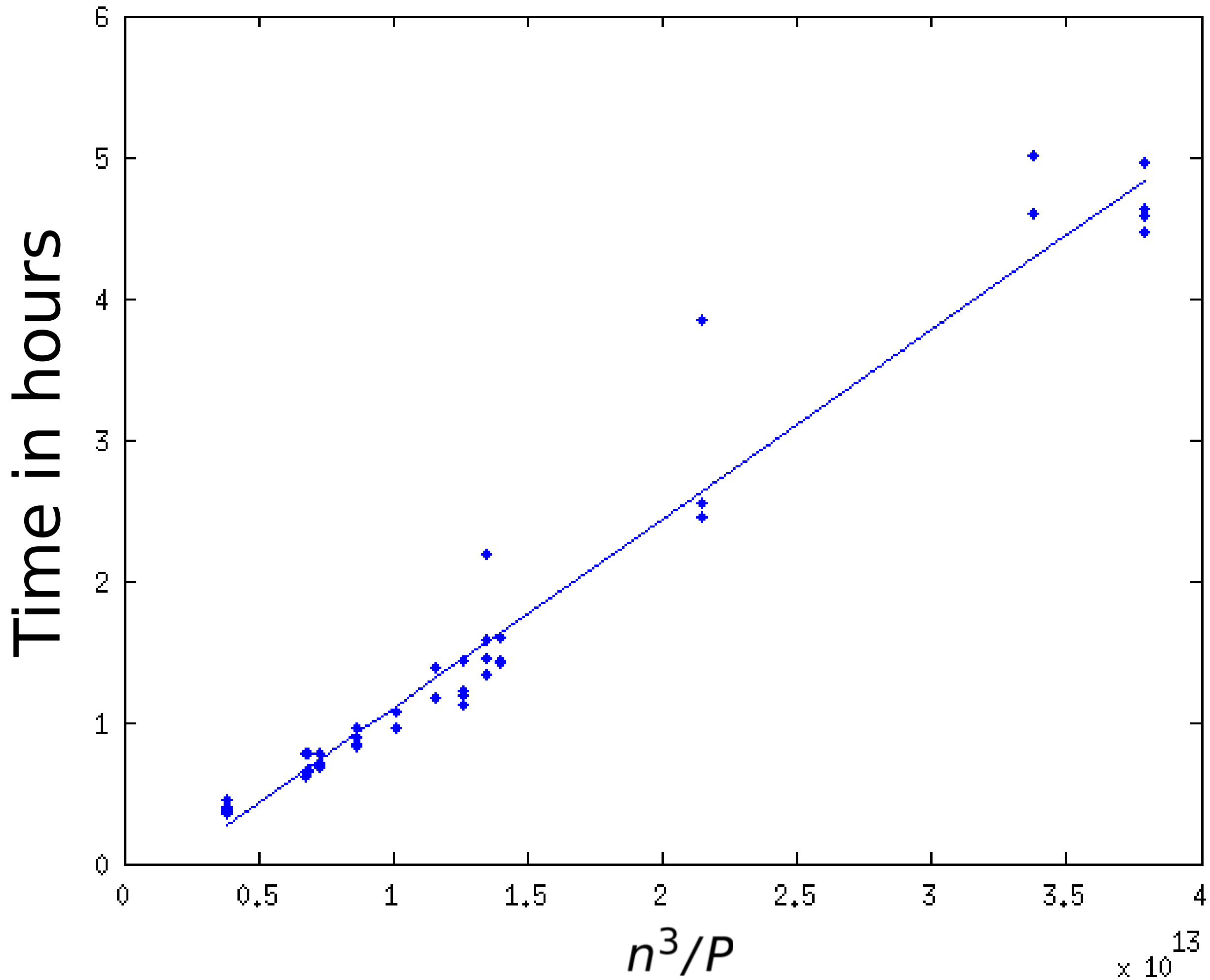
Threads	Ranks	Time-T	Time-E
1	36	1271.4	339.0
4	9	1058.1	456.6

Cray libsci

Threads	Ranks	Time
1	64	1412.5
4	16	1881.4
16	4	Omitted.

Normalized Laplacian for 36k-by-36k co-author graph of CondMat

Weak Parallel Scaling



Time $\propto (1.3) n^3/P$

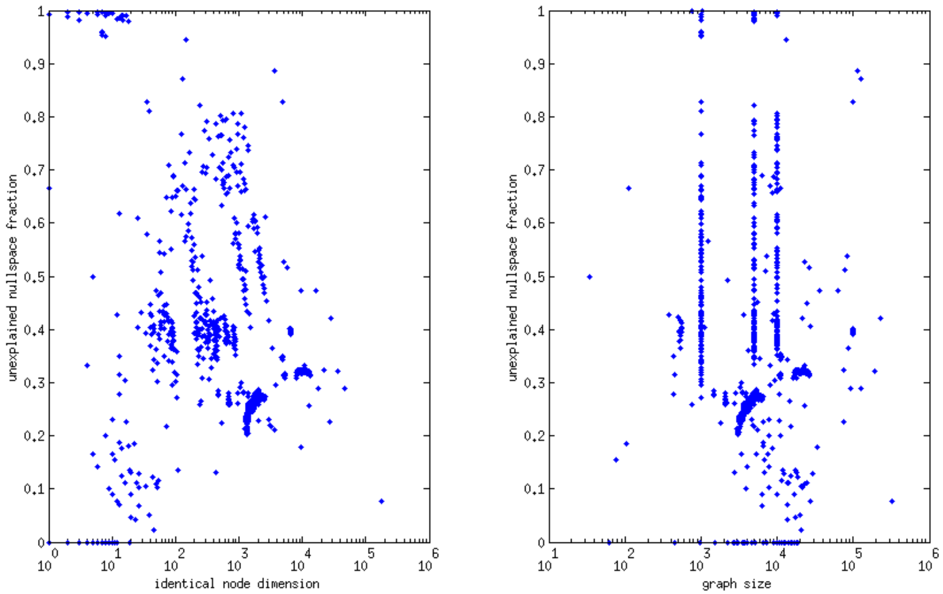
Good strong scaling up to 325,000 vertices

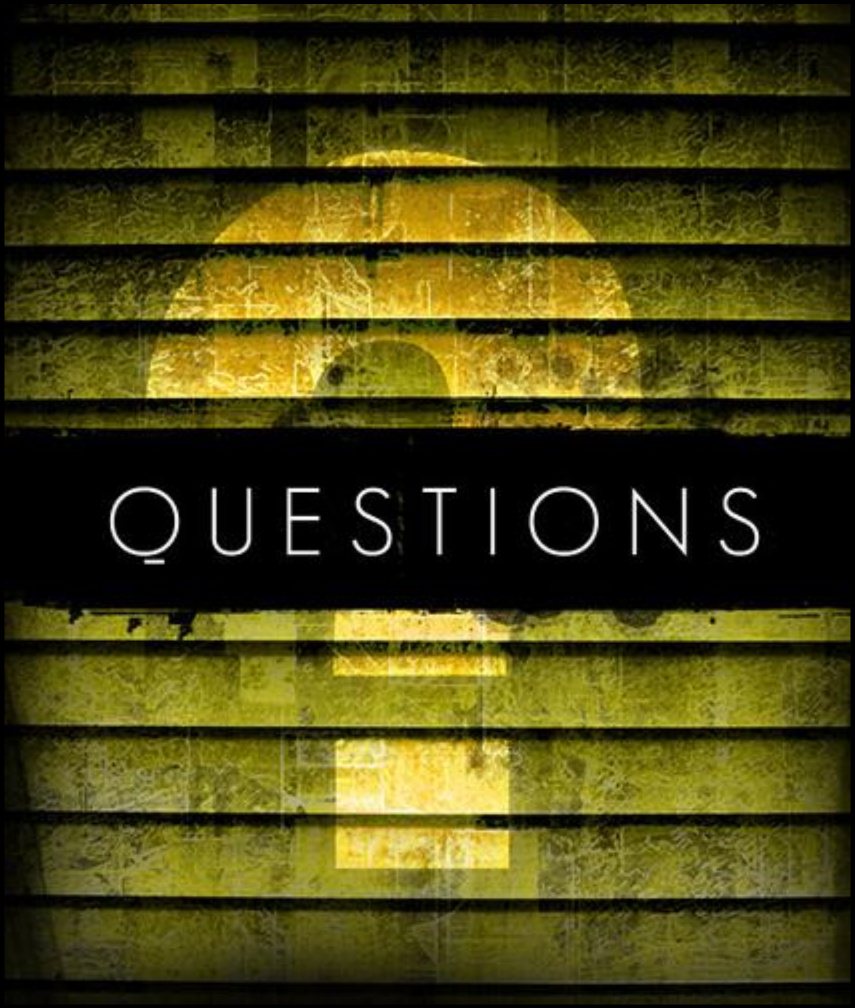
Estimated time for 500,000 nodes **9 hours with 925 nodes (7400 procs)**

Nullspaces of the adjacency matrix

$$(\boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{A})\mathbf{x} = \mathbf{x} \Rightarrow \boldsymbol{A}\mathbf{x} = 0$$

So unit eigenvalues of the normalized Laplacian are null-vectors of the adjacency matrix.





Code will be available eventually. Image from good financial cents.

GRAPHS AND THEIR MATRICES

As well as things we already know about graph spectra.

Spectral bounds from Gerschgorin

$$-d_{\max} \le \lambda(\mA) \le d_{\max}$$

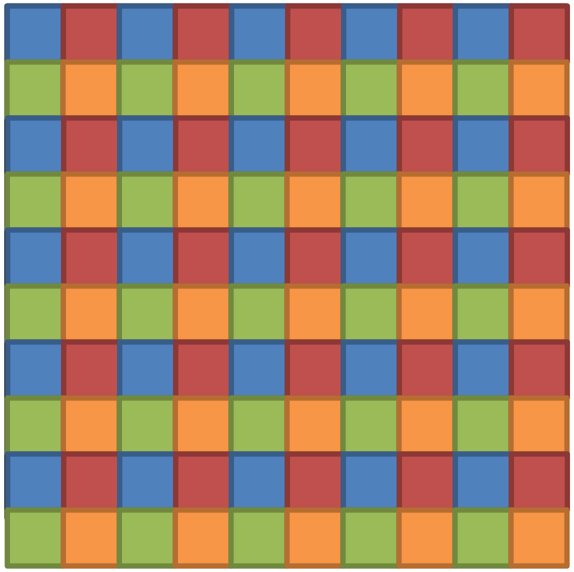
$$0 \le \lambda(\mL) \le 2 d_{\max}$$

$$0 \le \lambda(\mat{\tilde{L}}) \le 2$$
(from a slightly different approach)

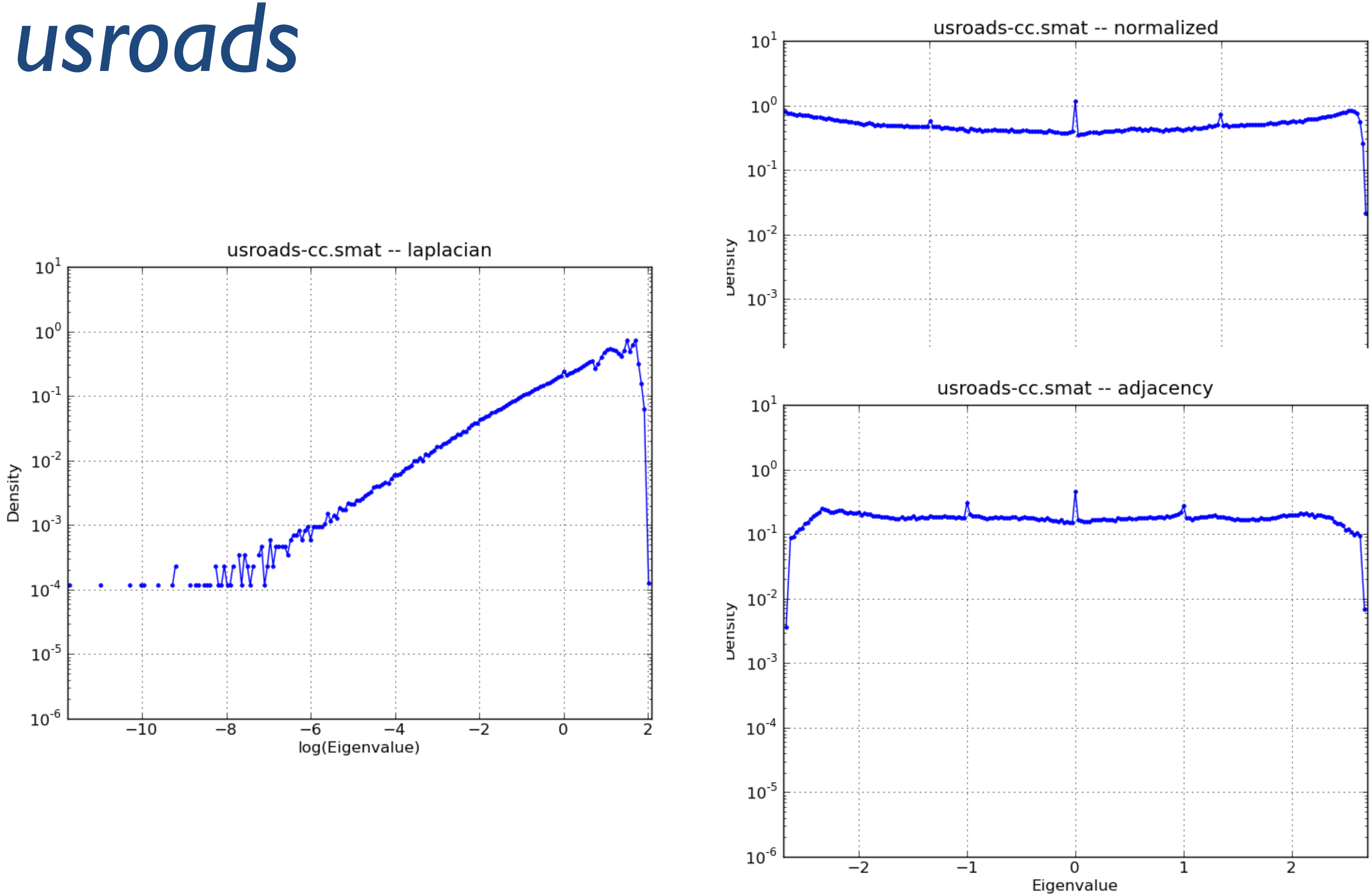
ScaLAPACK

LAPACK with distributed memory dense matrices

Scalapack uses a 2d block-cyclic dense matrix distribution



usroads

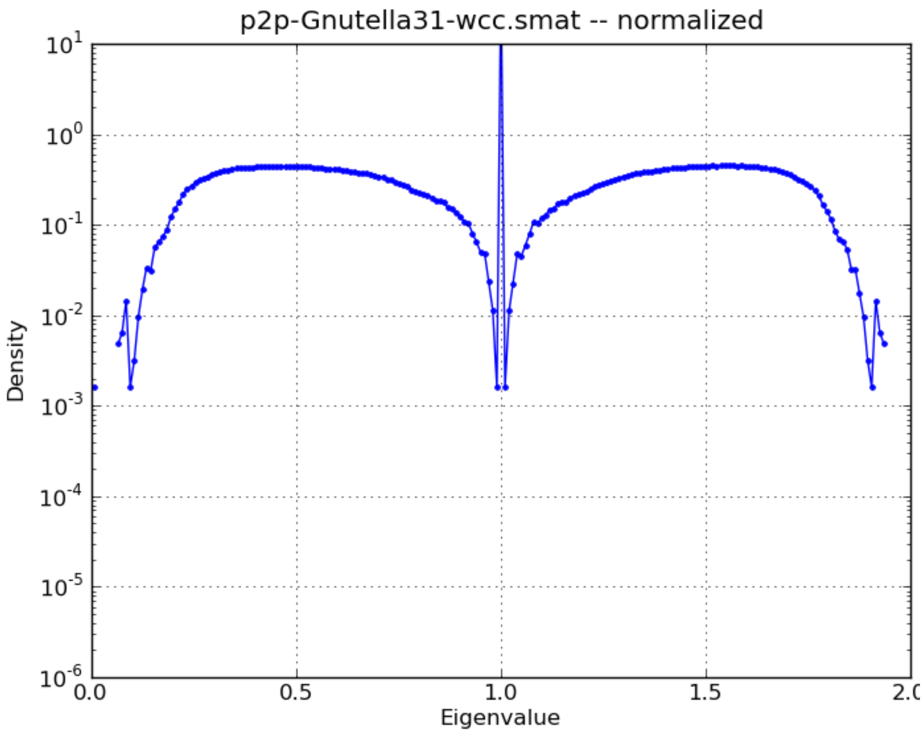


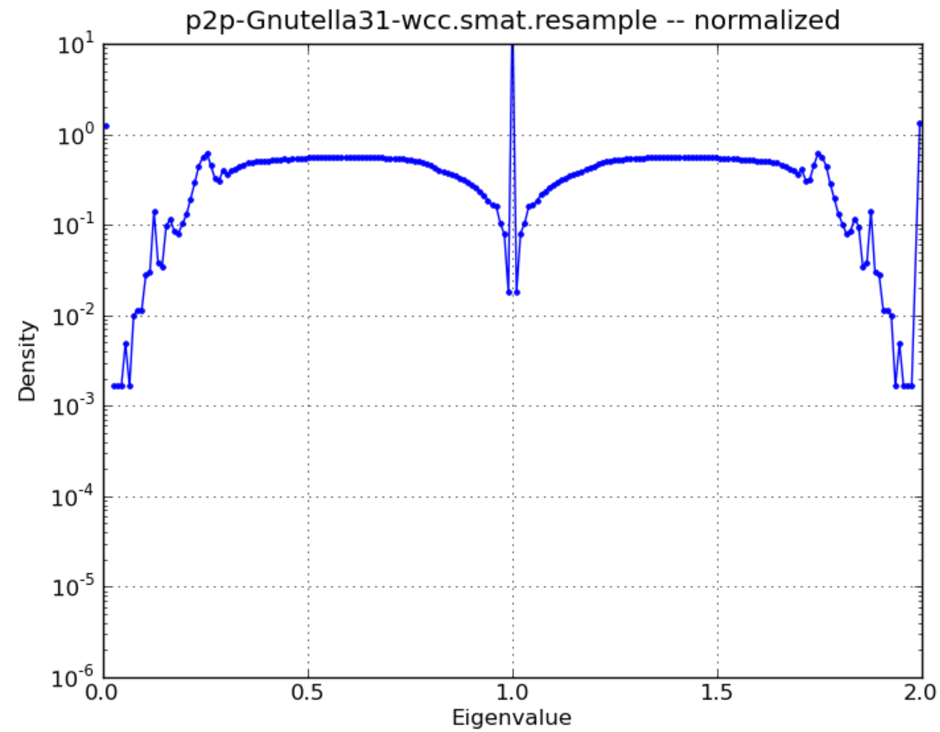
Connected component from the us road network

Movies of spectra...

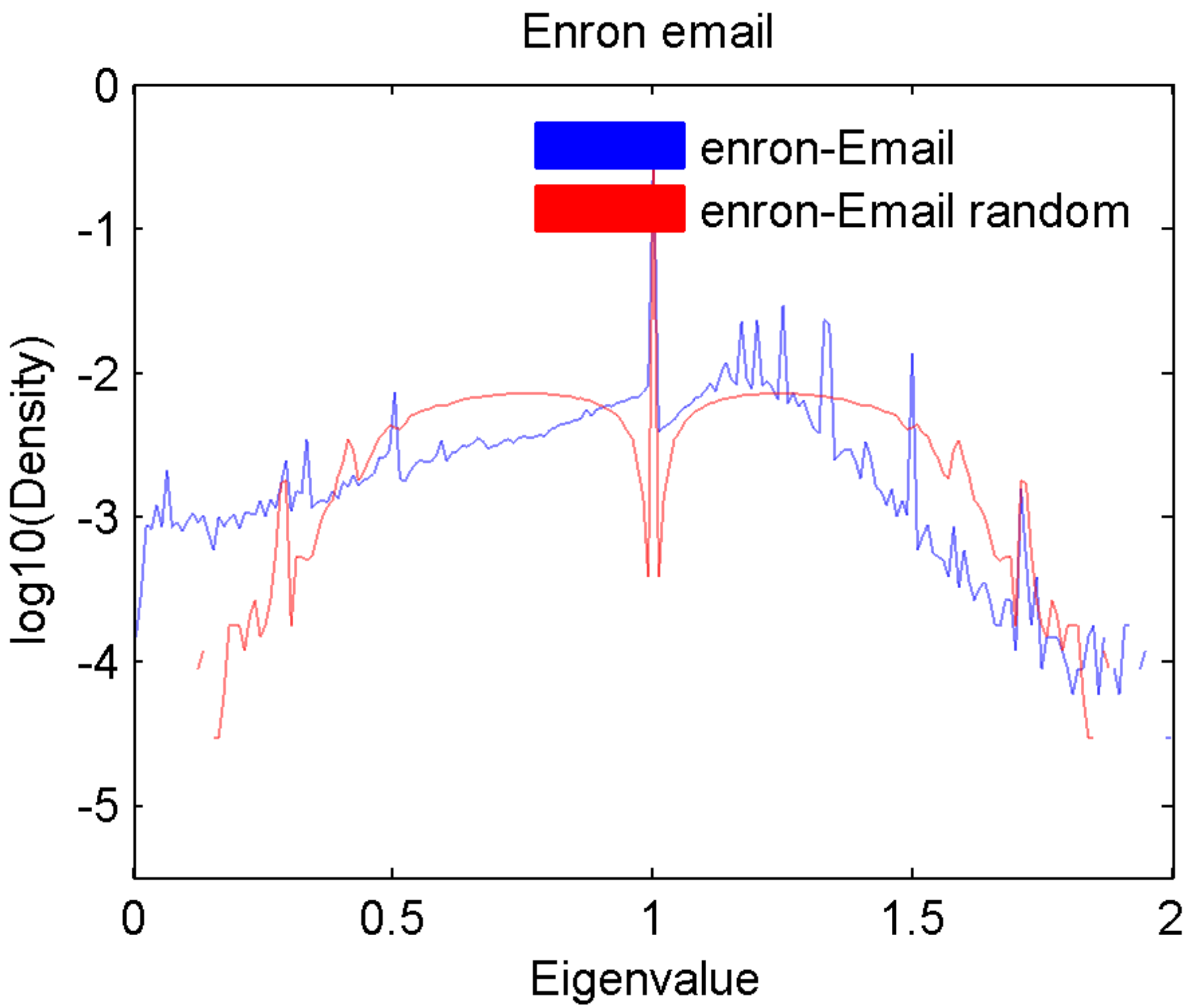
As these models evolve, what do the spectra look like?

Gnutella large vs. resampled

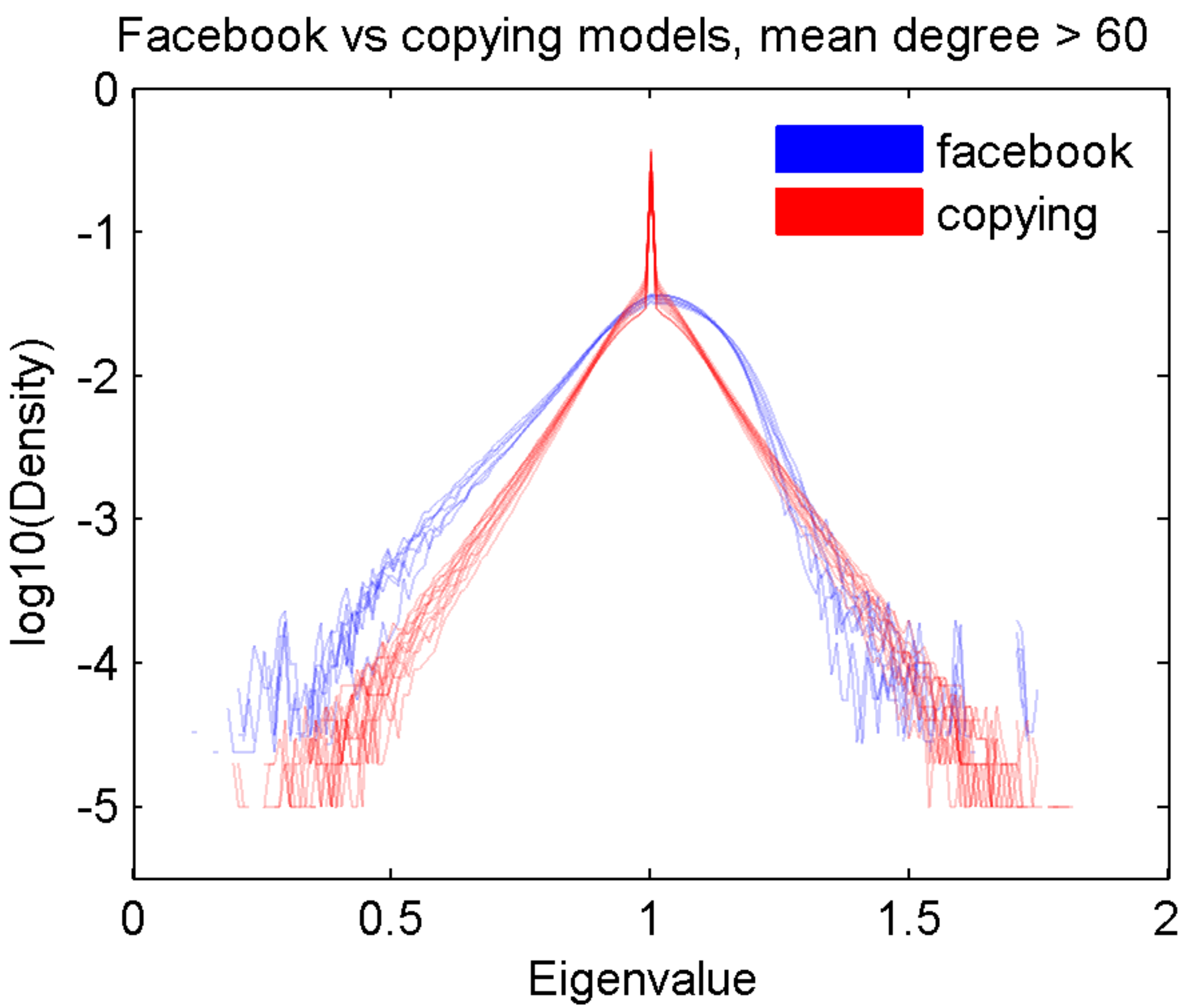




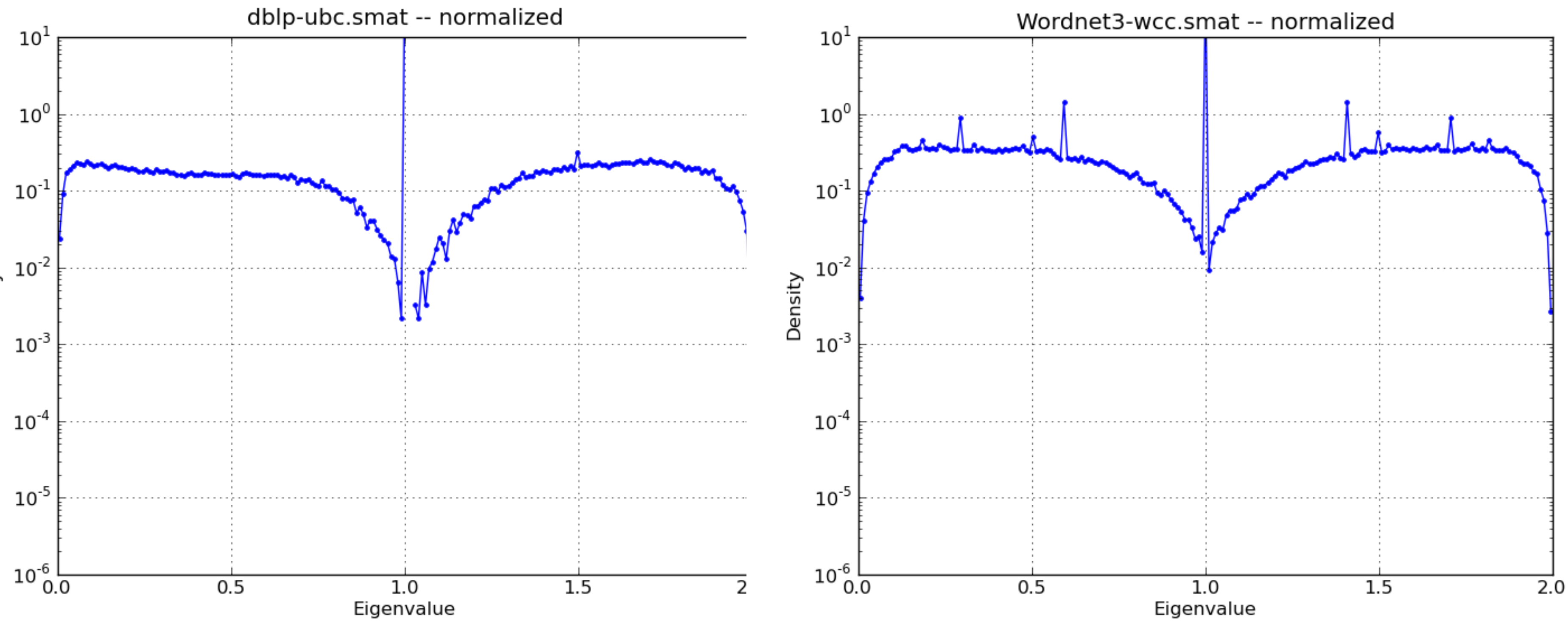
Just the degree distribution? No



Facebook vs. Copying model



Same density



Both have a mean degree of 3.8

Online Note

This talk is preliminary work. Make sure to check for updated versions!

Models

Preferential Attachment

Start graph with a k-node clique. Add a new node and connect to k random nodes, chosen proportional to degree.

Copying model

Start graph with a k-node clique. Add a new node and pick a parent uniformly at random. Copy edges of parent and make an error with probability $$\alpha$$

Forest Fire

Start graph with a k-node clique. Add a new node and pick a parent uniformly at random. Do a random “bfs’/”forest fire” and link to all nodes “burned”