Signal Processing on Databases

Jeremy Kepner

Lecture 5: Perfect Power Law Graphs: Generation, Sampling, Construction, and Fitting

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

This work is sponsored by the Department of the Air Force under Air Force Contract #FA8721-05-C-0002. Opinions, interpretations, recommendations and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

Outline

- **Introduction**
- **Sampling**
- **Sub-sampling**
- **Joint Distribution**
- **Reuter's Data**
- **Summary**

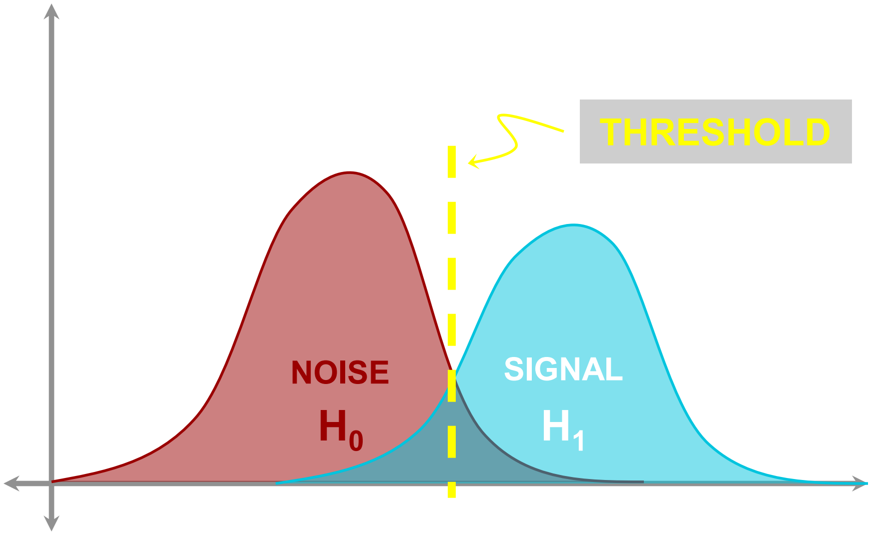
- ***Detection Theory***
- ***Power Law Definition***
- ***Degree Construction***
- ***Edge Construction***
- ***Fitting: α, N, M***
- ***Example***

Goals

- Develop a background model for graphs based on "perfect" power law
- Examine effects of sampling such a power law
- Develop techniques for comparing real data with a power law model
- Use power law model to measure deviations from background in real data

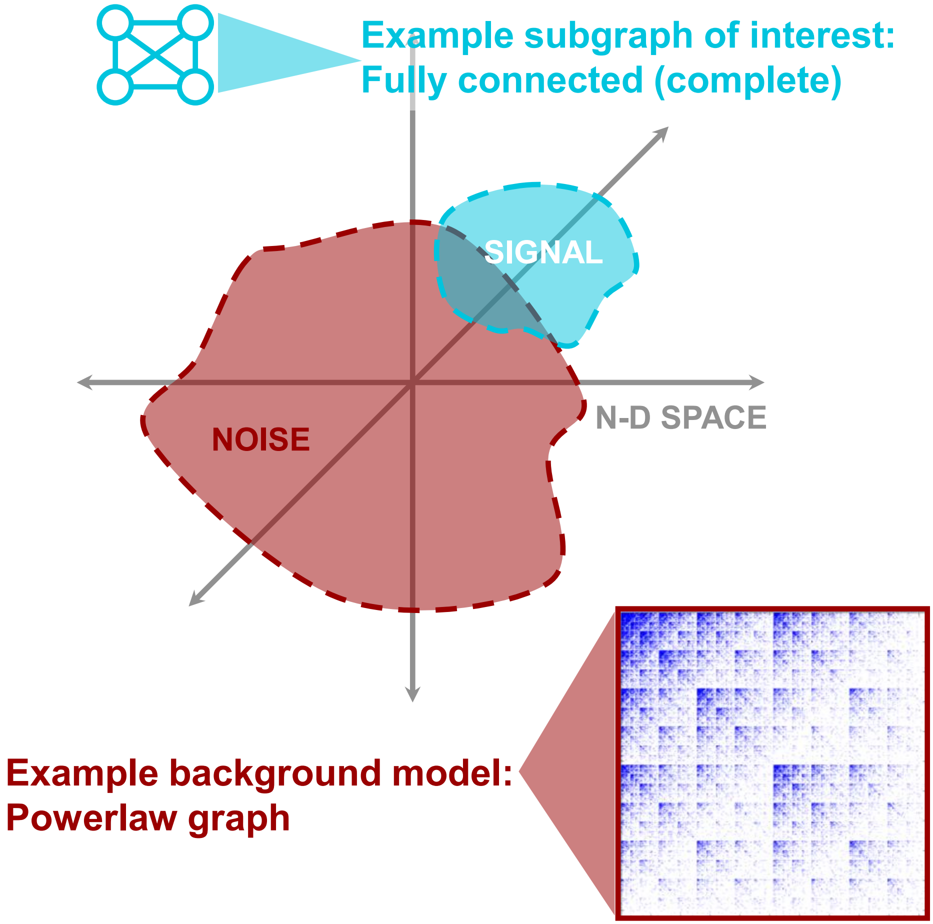
Detection Theory

DETECTION OF SIGNAL IN NOISE



ASSUMPTIONS
- Background (noise) statistics
- Foreground (signal) statistics
- Foreground/background separation
- Model ≈ reality

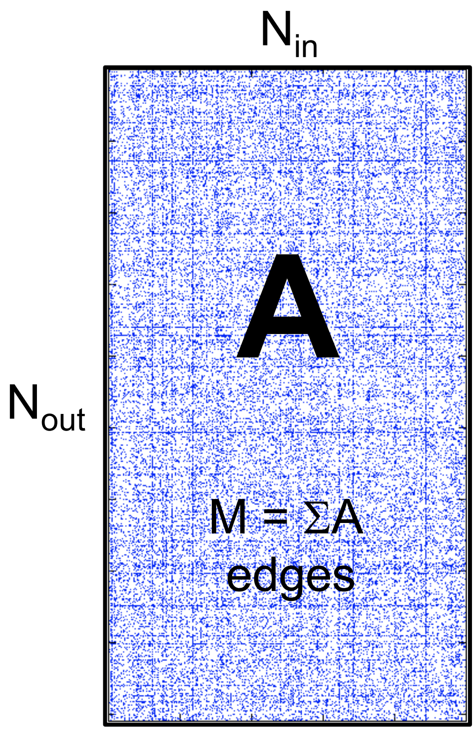
DETECTION OF SUBGRAPHS IN GRAPHS



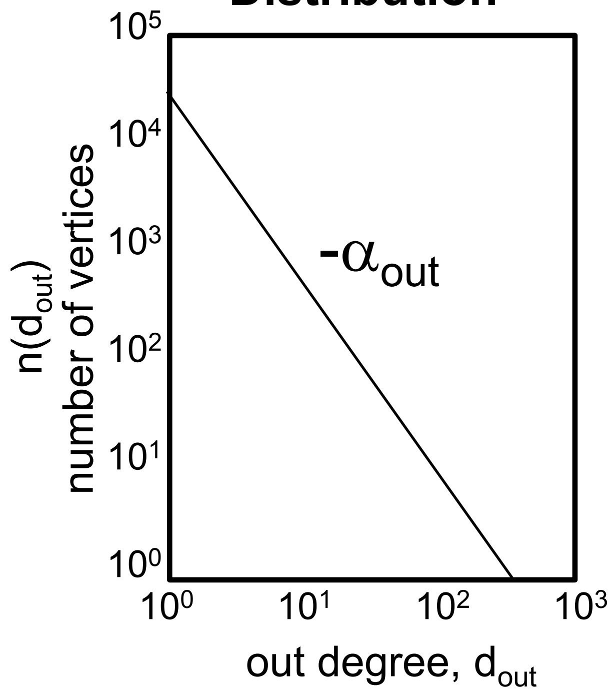
Can we construct a background model based on power law degree distribution?

"Perfect" Power Law Matrix Definition

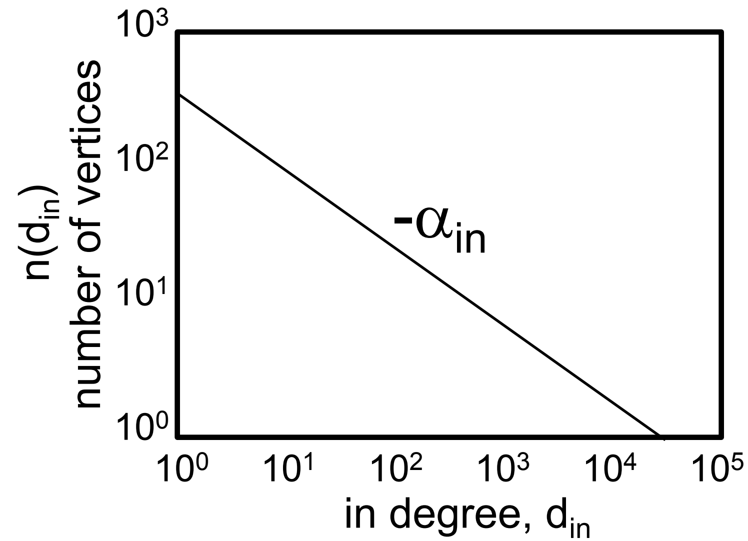
Adjacency/Incidence Matrix



Vertex Out Degree Distribution



Vertex In Degree Distribution



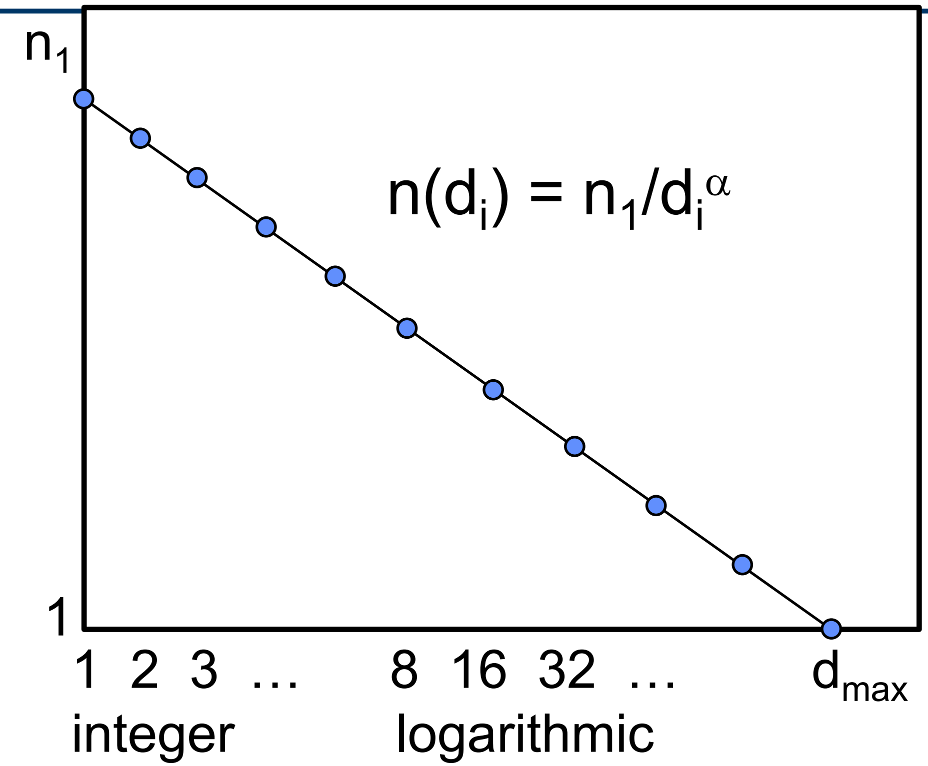
- **Graph represented as a rectangular sparse matrix**
  - **Can be undirected, multi-edged, self-loops, disconnected, hyper edges, …**
- **Out/in degree distributions are *independent* first order statistics**
  - **Only constraint:** $\Sigma\, n(d_{out})\, d_{out} = \Sigma\, n(d_{in})\, d_{in} = M$

Power Law Distribution Construction

- **Perfect power law matlab code**

```
function [di ni] = PPL(alpha,dmax,Nd)
logdi = (0:Nd) * log(dmax) / Nd;
di    = unique(round(exp(logdi)));
logni = alpha * (log(dmax) - log(di));
ni    = round(exp(logni));
```

- **Parameters**
  - alpha **= slope**
  - dmax **= largest degree vertex**
  - Nd **= number of bins (before unique)**



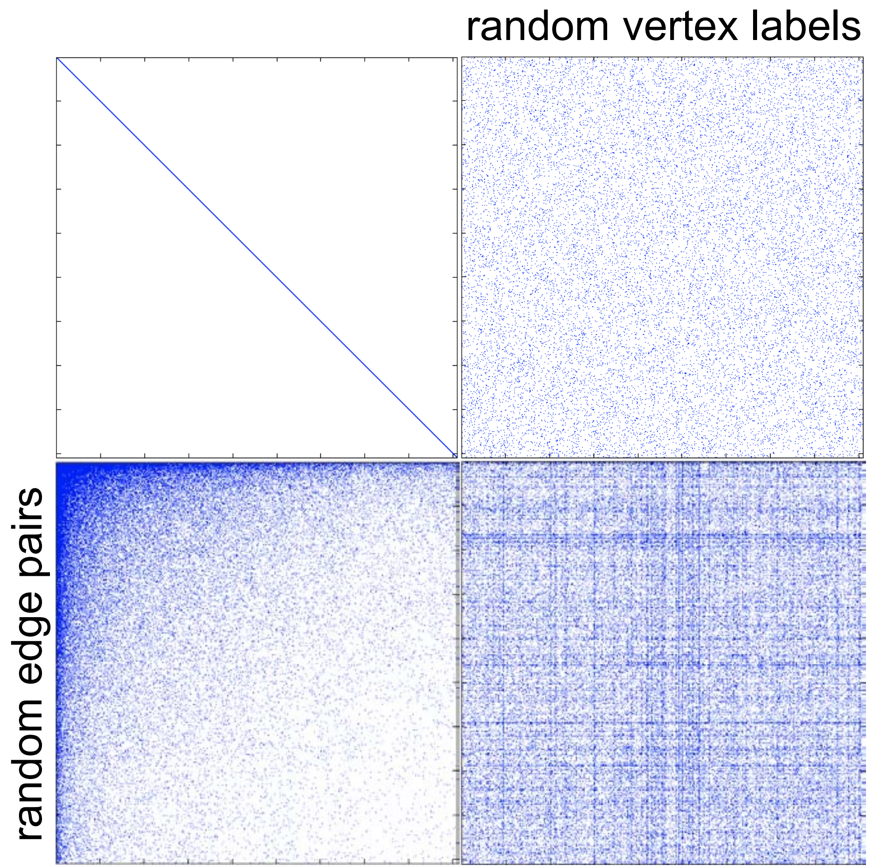
- **Simple algorithm naturally generates perfect power law**
- **Smooth transition from integer to logarithmic bins**
- **"Poor man's" slope estimator:** $\alpha = \log(n_1)/\log(d_{max})$

Power Law Edge Construction

- **Power law vertex list matlab code**

```
function v = PowerLawEdges(di,ni);
A1 = sparse(1:numel(di),ni,di);
A2 = fliplr(cumsum(fliplr(A1),2));
[tmp tmp d] = find(A2);
A3 = sparse(1:numel(d),d,1);
A4 = fliplr(cumsum(fliplr(A3),2));
[v tmp tmp] = find(A4);
```

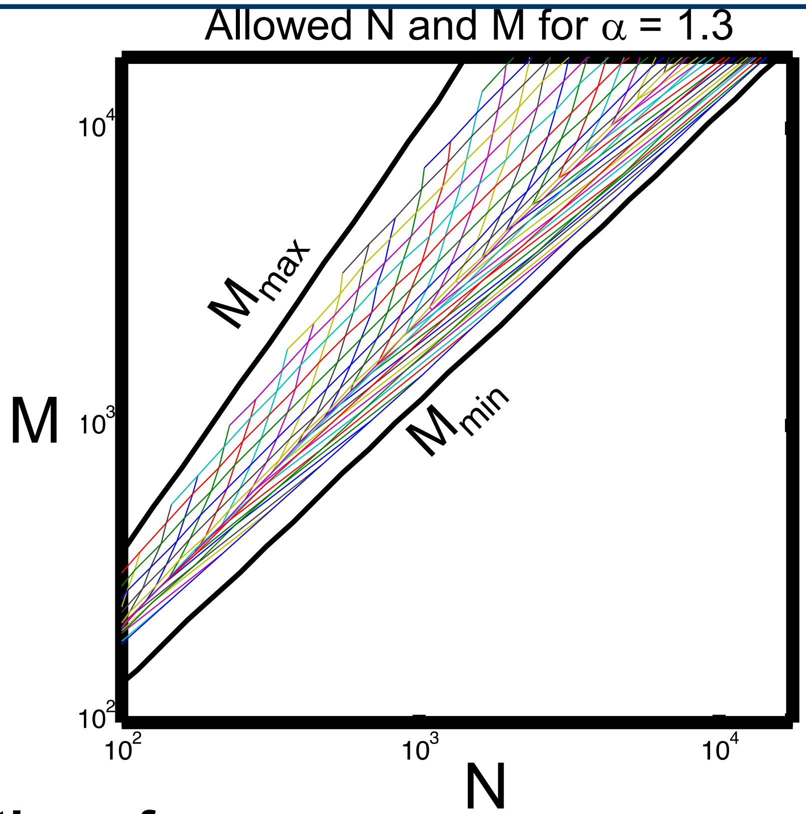
- **Degree distribution independent of**
  - **Vertex labels**
  - **Edge pairing**
  - **Edge order**



- **Algorithm generates list of vertices corresponding to any distribution**
- **All other aspects of graph can be set based on desired properties**

Fitting α, N, M

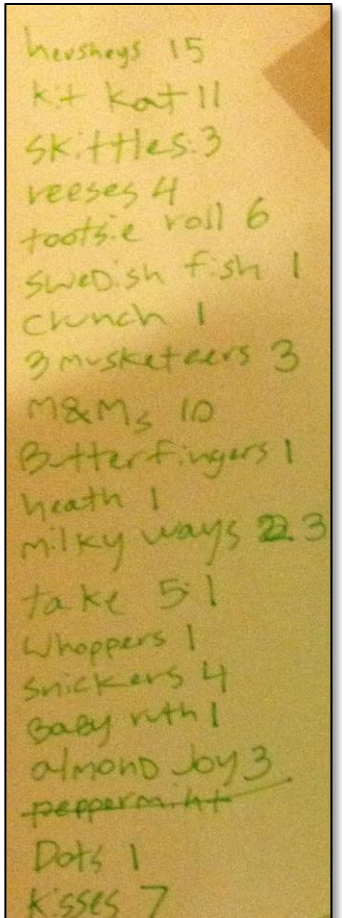
- **Power law model works for any**
  - $\alpha > 0$, $d_{max} > 1$, $N_d > 1$
- **Desire distribution that fits**
  - α, N, M
- **Can invert formulas**
  - $N = \Sigma_i n(d_i)$
  - $M = \Sigma_i n(d_i)\, d_i$



- **Highly non-linear; requires a combination of**
  - **Exhaustive search, simulated annealing, and Broyden's algorithm**

> - **Given** α, N, M **can solve for** N_d **and** d_{max}
> - **Not all combinations of** α, N, M **are consistent with power law**

Example: Halloween Candy

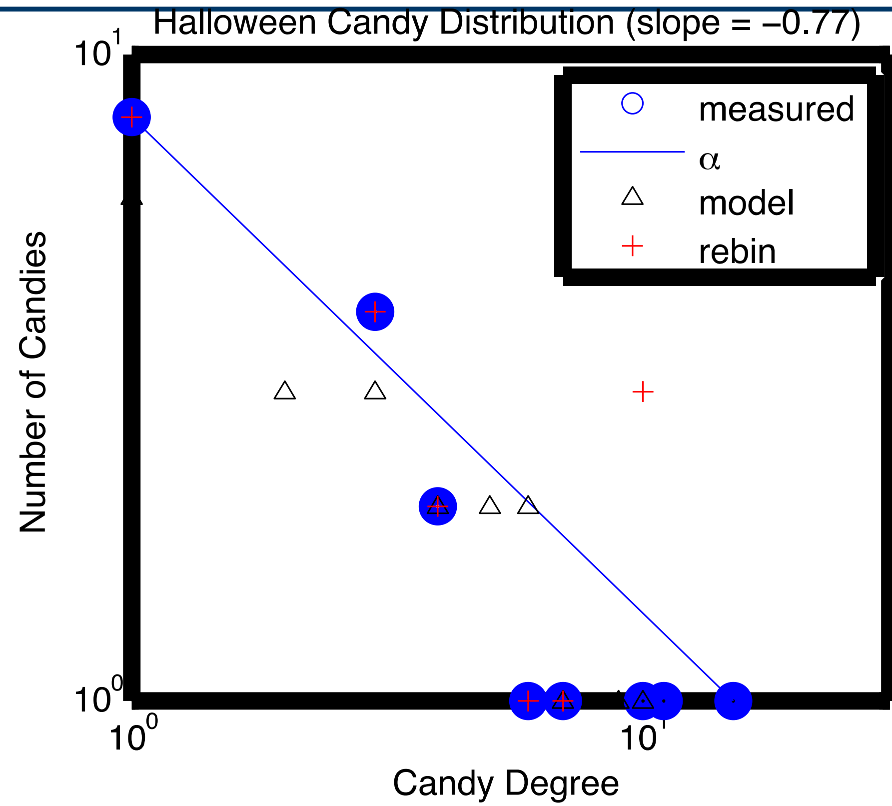


Distribution parameters

- M = 77
- N = 19
- M/N = 4.1
- n_1 = 8
- d_{max} = 15
- α = 0.77

Fit parameters

- M = 77
- N = 21
- M/N = 3.7



Procedure

- **Estimate parameters from data**
- **Determine if viable power law fit**
- **Rebin measured to power law and compare**

Outline

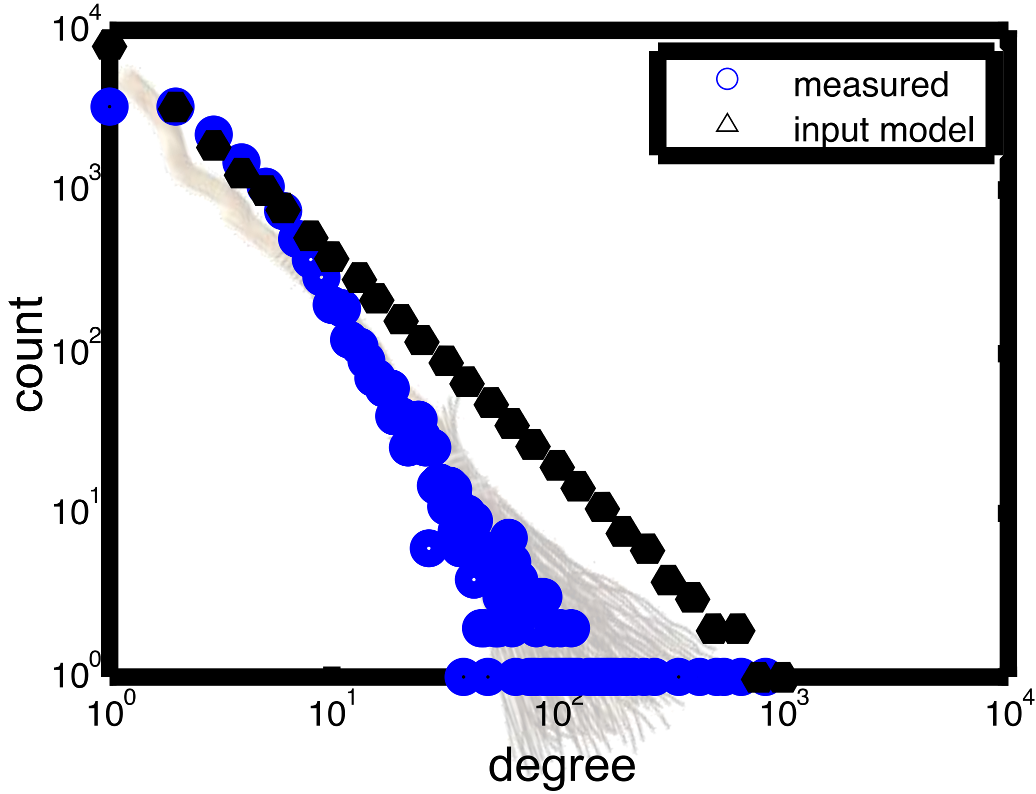
- Introduction
- **Sampling**
  - *Graph construction*
  - *Graphs from E' * E*
  - *Edge ordering and densification*
- Sub-sampling
- Joint Distribution
- Reuter's Data
- Summary

Graph Construction Effects

- **Generate a perfect power law NxN randomize adjacency matrix A**
  - α = 1.3, d_{max} = 1000, N_d = 50
  - N = 18K, M = 84K

- **Make undirected, unweighted, with no self-loops**

```
A = triu(A + A');
A = double(logical(A));
A = A - diag(diag(A));
```

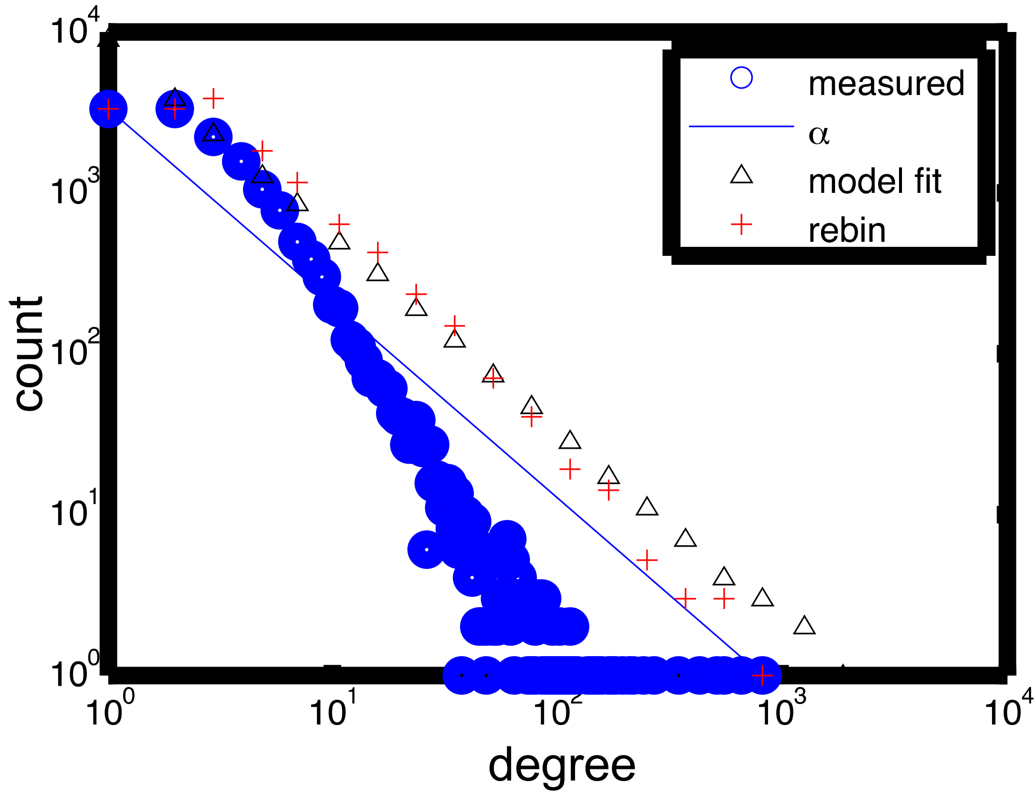


- **Graph theory best for undirected, unweighted graphs with no self-loops**
- **Often "clean up" real data to apply graph theory results**
- **Process mimics "bent broom" distribution seen in real data sets**

Power Law Recovery

Procedure

- **Compute** α, N, M **from measured**
- **Fit perfect power law to these parameters**
- **Rebin measured data using perfect power law degree bins**



- **Perfect power law fit to "cleaned up" graph can recover much of the shape of the original distribution**

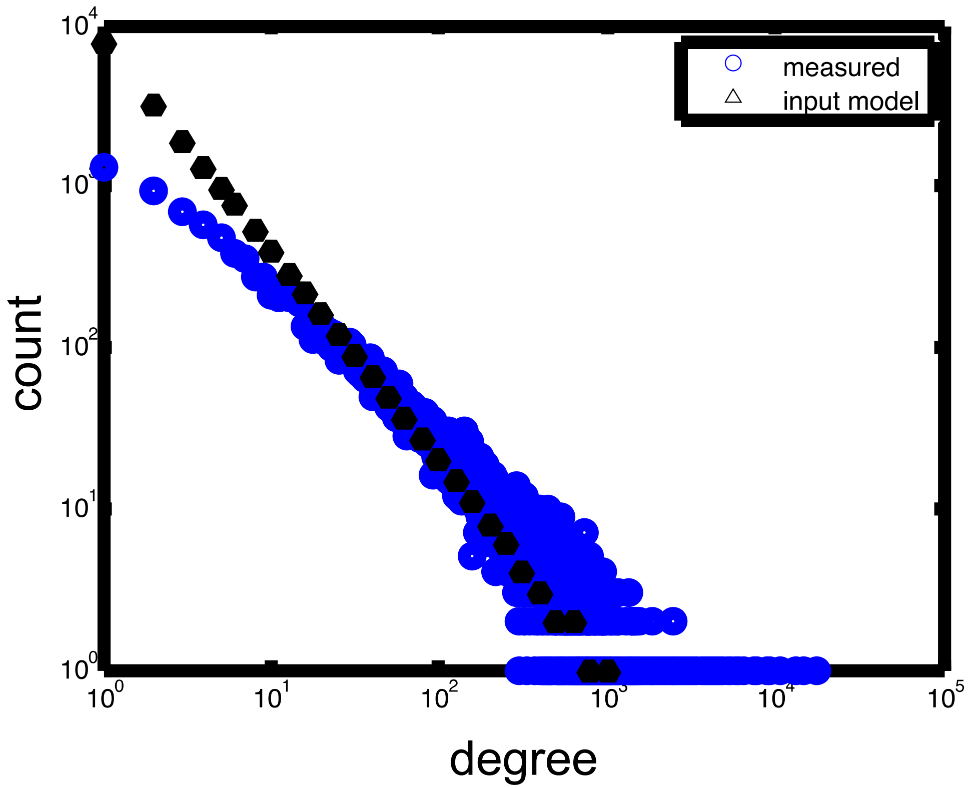
LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Correlation Construction Effects

- **Generate a perfect power law NxN randomize incidence matrix E**
  - $\alpha = 1.3$, $d_{max} = 1000$, $N_d = 50$
  - N = 18K, M = 84K

- **Make unweighted and use to form correlation matrix A with no self-loops**

```
E = double(logical(E));
A = triu(E' * E);
A = A - diag(diag(A));
```

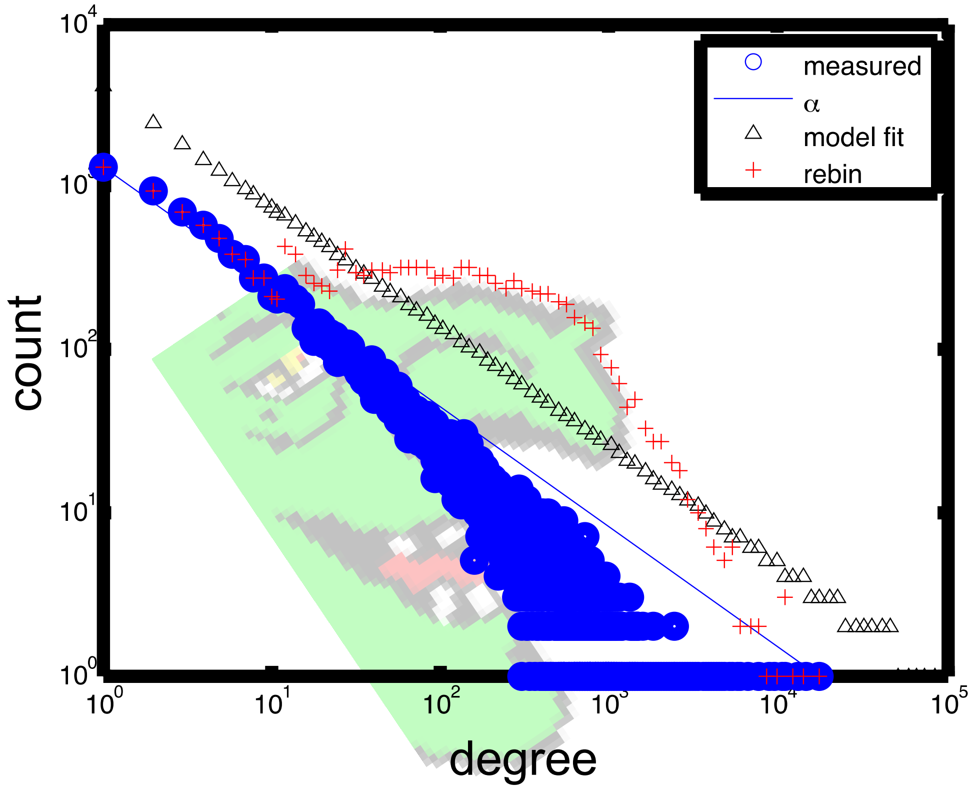


- **Correlation graph construction from incidence matrix results in a “bent broom” distribution that strongly resembles a power law**

Power Law Lost

Procedure

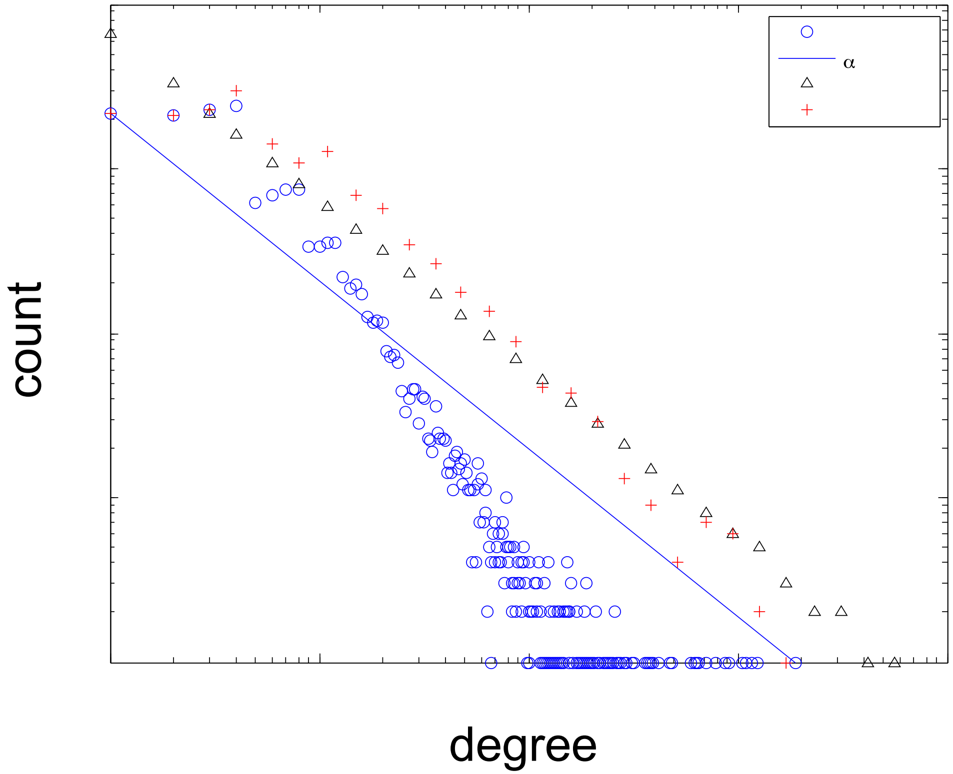
- **Compute α, N, M from measured**
- **Fit perfect power law to these parameters**
- **Rebin measured data using perfect power law degree bins**



- **Perfect power law fit to correlation shows non-power law shape**
- **Reveals "witches nose" distribution**

Power Law Preserved

- **In degree is power law**
  - α = 1.3, d_{max} = 1000, N_d = 50
  - N = 18K, M = 84K
- **Out degree is constant**
  - N = 16K, M = 84K
  - Edges/row = 5 (exactly)
- **Make unweighted and use to form correlation matrix A with no self-loops**



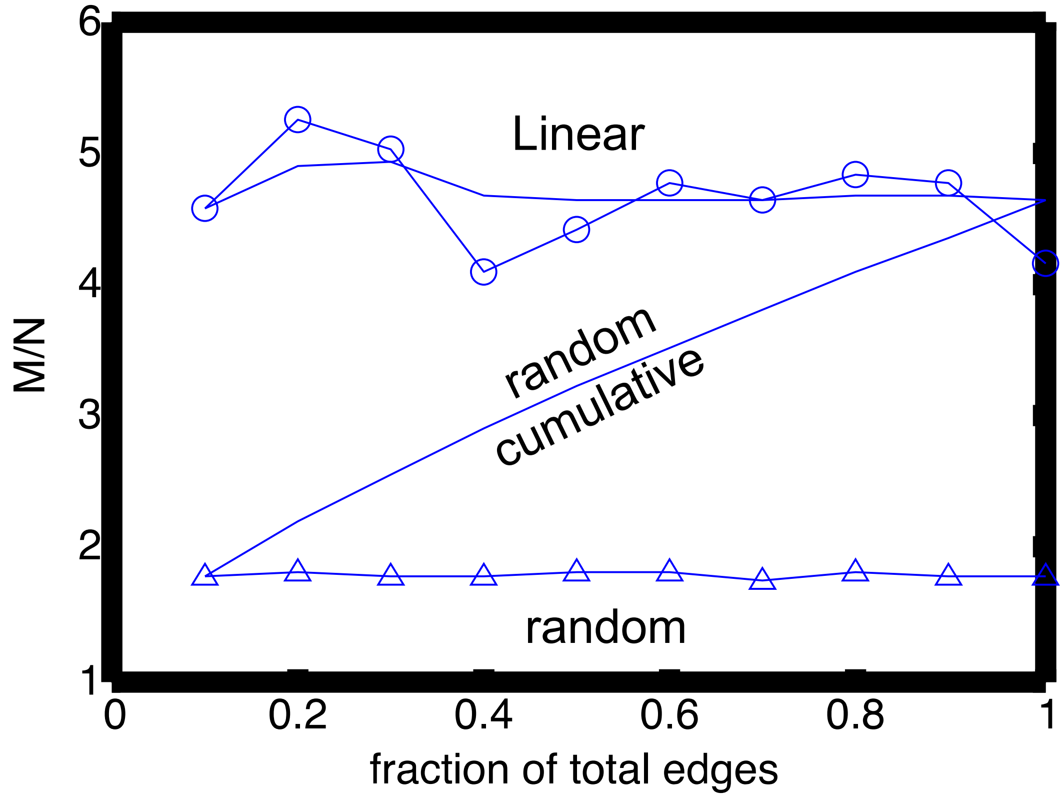
- **Uniform distribution on correlated dimension preserves power law shape**

Edge Ordering: Densification

- **Compute M/N cumulatively and piecewise for 2 orderings**
  - **Linear**
  - **Random**

- **By definition M/N goes from 1 to infinity for finite N**

- **Elimination of multi-edges reduces M and causes M/N to grow more slowly**

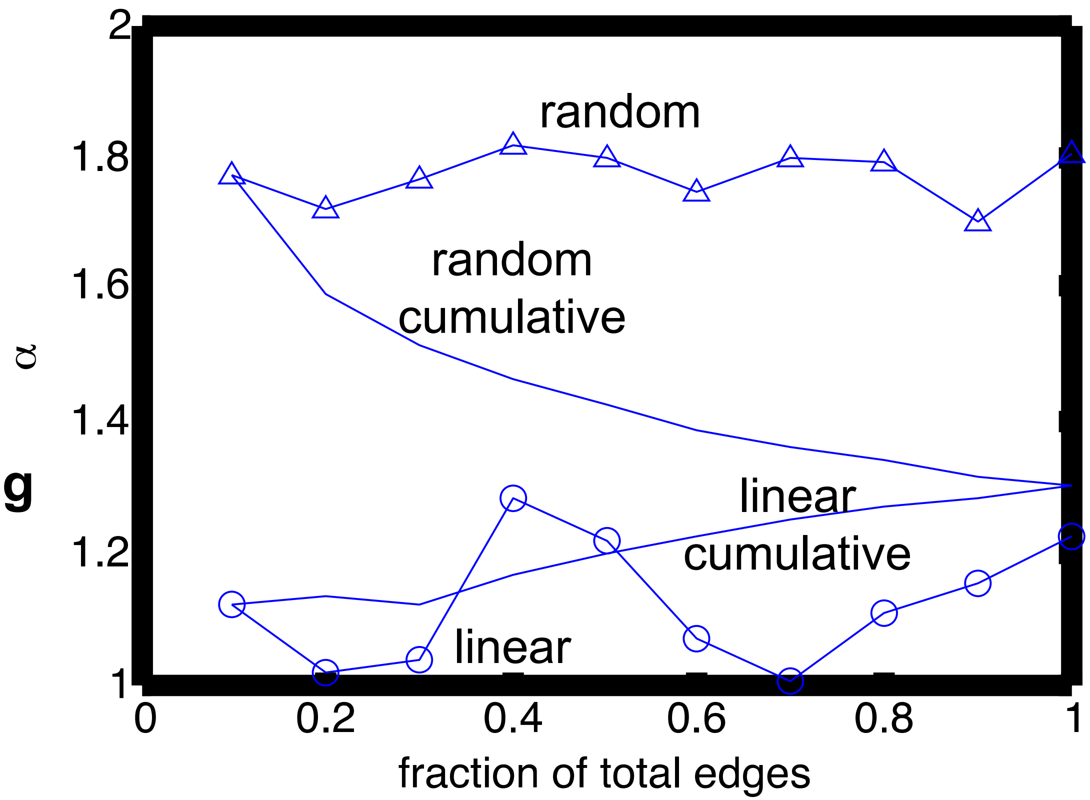


- **"Densification" is the observation that M/N increases with N**
- **Densification is a natural byproduct of randomly drawing edges from a power law distribution**
- **Linear ordering has constant M/N**

Edge Ordering: Power Law Exponent (α)

- **Compute α cumulatively and piecewise for 2 orderings**
  - **Linear**
  - **Random**

- **Edge ordering and sampling have large effect on the power law exponent**



- **Power law exponent is fundamental to distribution**
- **Strongly dependent on edge ordering and sample size**

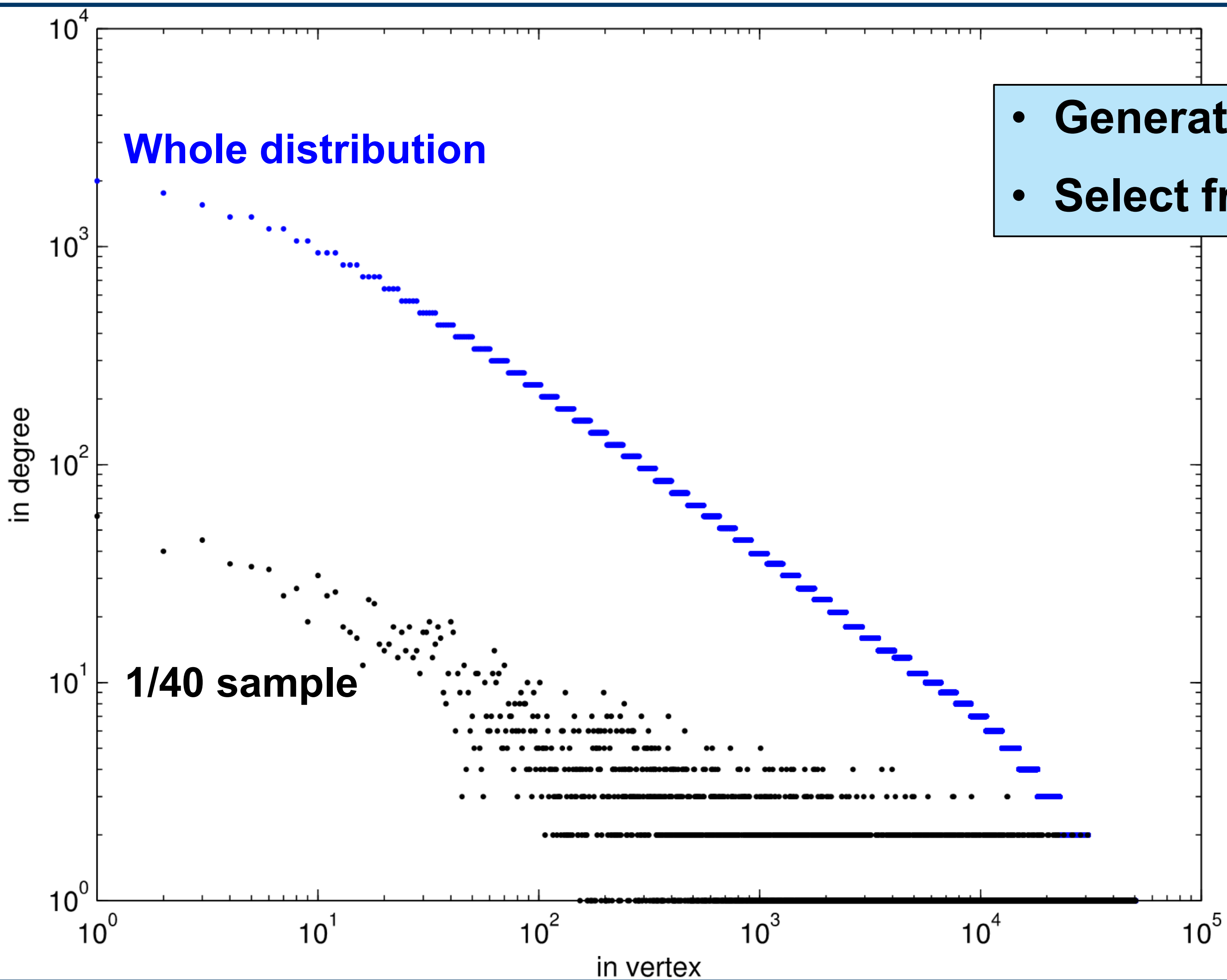
Outline

- Introduction
- Sampling
- **Sub-sampling**
- Joint Distribution
- Reuter’s Data
- Summary

Sub-Sampling Challenge

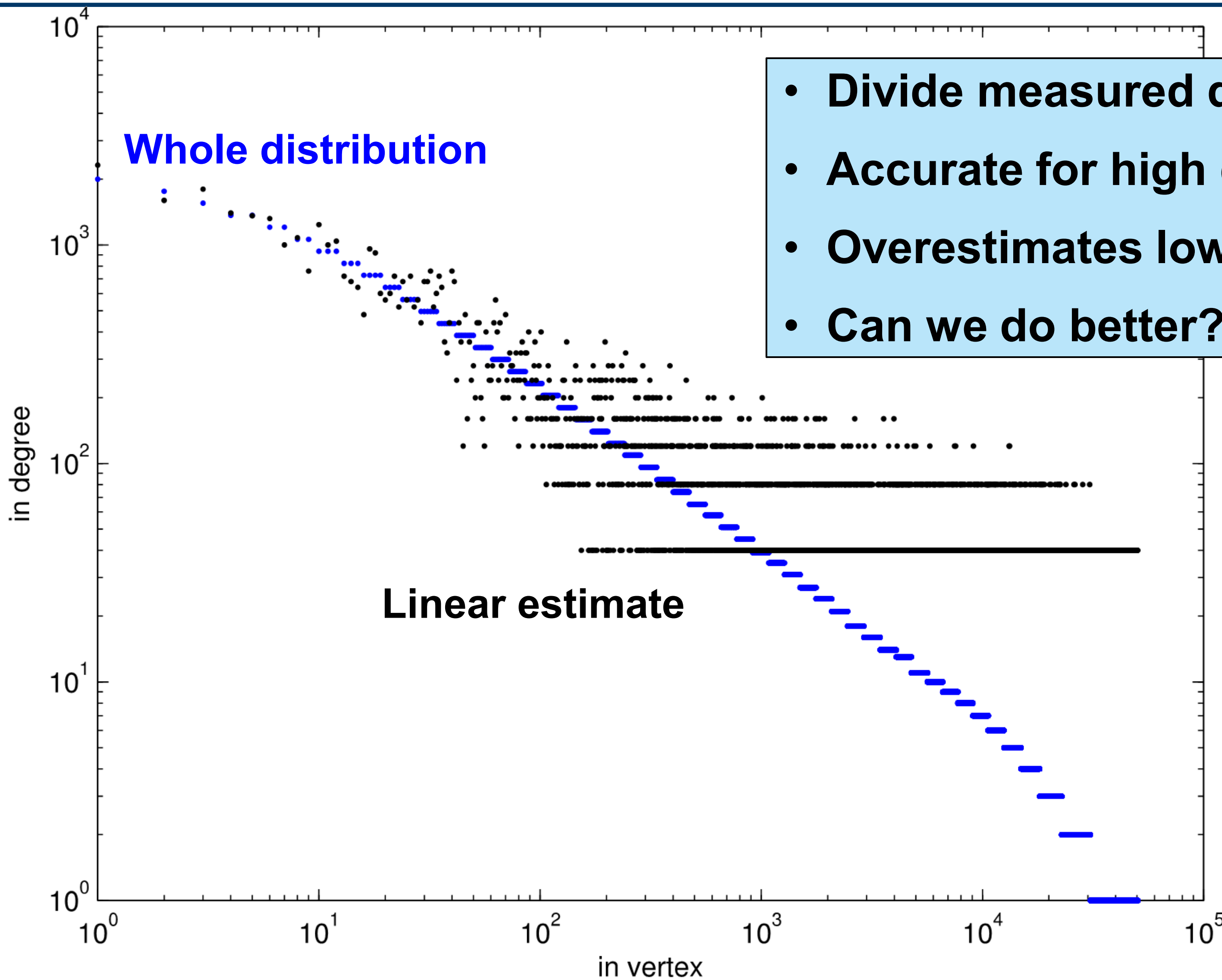
- **Anomaly detection requires good estimates of background**
- **Traversing entire data sets to compute background counts is increasingly prohibitive**
  - **Can be done at ingest, but often is not**
- **Can background be accurately estimated from a sub-sample of the entire data set?**

Sampling a Power Law



- **Generate power law**
- **Select fraction of edges**

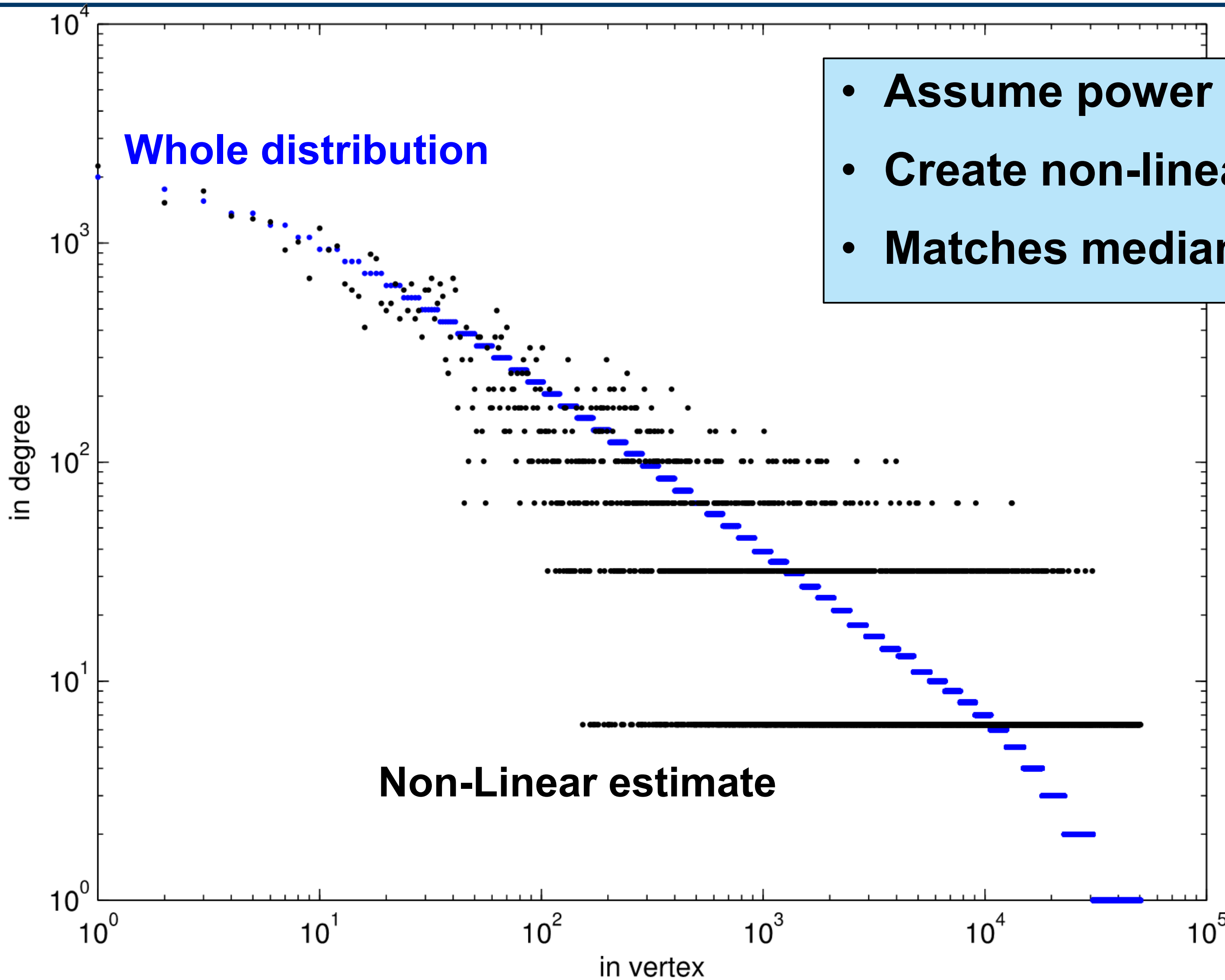
Linear Degree Estimate



- Divide measured degree by fraction
- Accurate for high degree
- Overestimates low degree
- Can we do better?

Non-Linear Degree Estimate

- Assume power law input
- Create non-linear estimate
- Matches median degree



Sub-Sampling Formula

- f = fraction of total edges sampled
- $\underline{n}_1$ = # of vertices of degree 1
- $\underline{d}_{max}$ = maximum degree
- Allowed slope: $\ln(\underline{n}_1)/\ln(\underline{d}_{max}/f) < \alpha < \ln(\underline{n}_1)/\ln(\underline{d}_{max})$

- Cumulative distribution

$$P(\alpha,d) = (f^{1-\alpha}\,\underline{d}_{max}{}^{\alpha} / \underline{n}_1)\,\Sigma_{i<d}\, i^{1-\alpha}\, e^{-fi}$$

- Find α^* such that $P(\alpha^*,\infty) = 1$
- Find $d_{50\%}$ such that $P(\alpha^*,d_{50\%}) = ½$
- Compute $K = 1/(1 + \ln(d_{50\%})/\ln(f))$

- Non-linear estimate of true degree of vertex v from sample $\underline{d}(v)$

$$d(v) = \underline{d}(v) / f^{1-1/(K\,\underline{d}(v))}$$

Outline

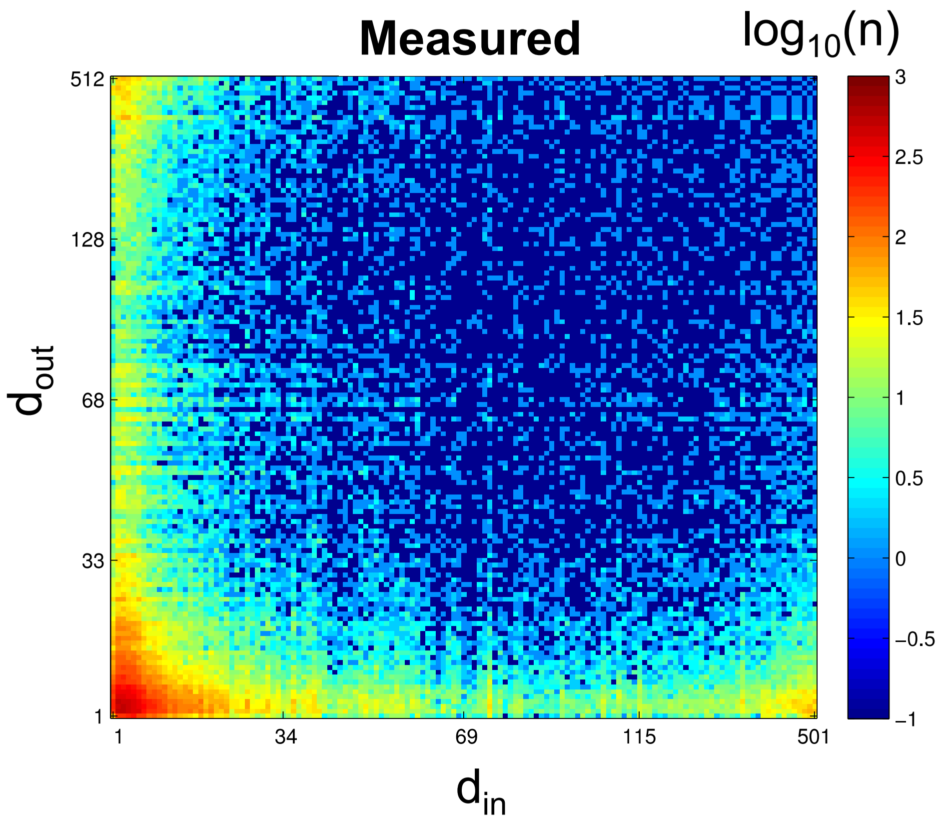
- Introduction
- Sampling
- Sub-sampling
- **Joint Distribution**
  - *Measured*
  - *Expected*
  - *Time Evolution*
- Reuter's Data
- Summary

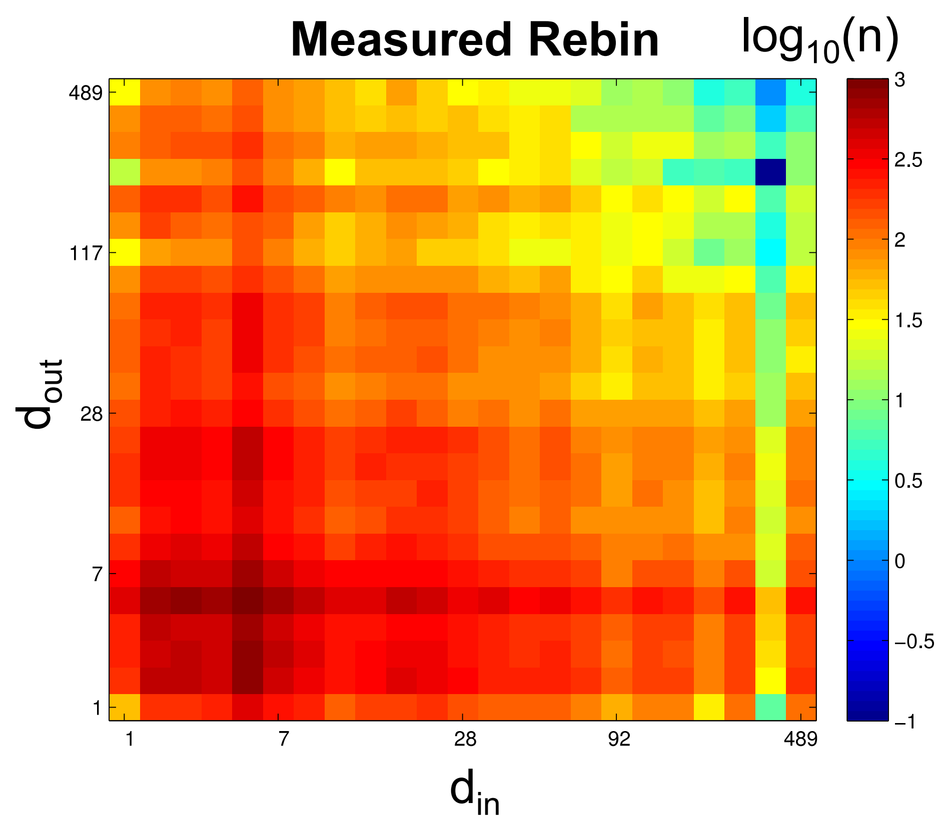
Joint Distribution Definitions

- **Label each vertex by degree**
- **Count number of edges from** d_{out} **to** d_{in}**:** $n(d_{out},d_{in})$
- **Rebin based on perfect power law model**
- **Can compare measured vs. expected**

- **Power law model allows precise quantitative comparison of observed data with a model**

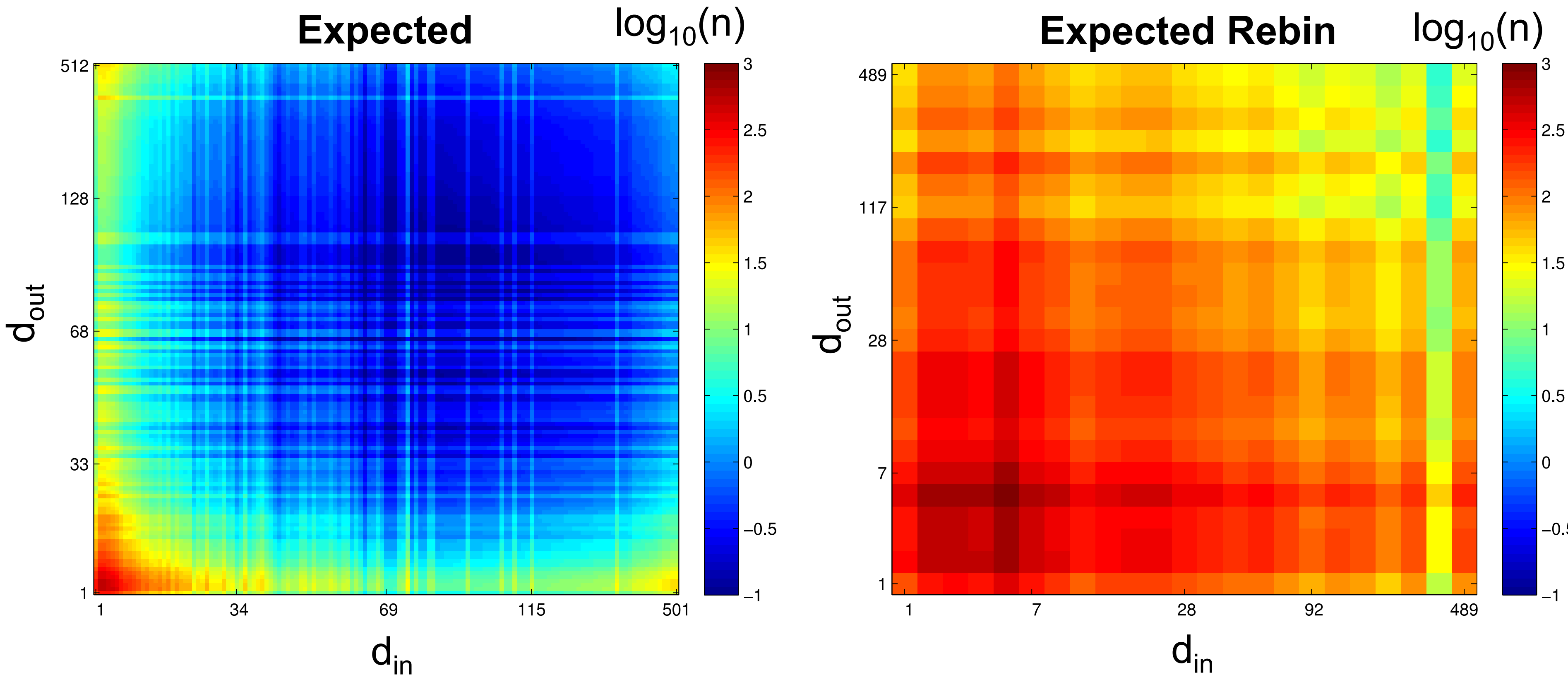
Measured Joint Distribution





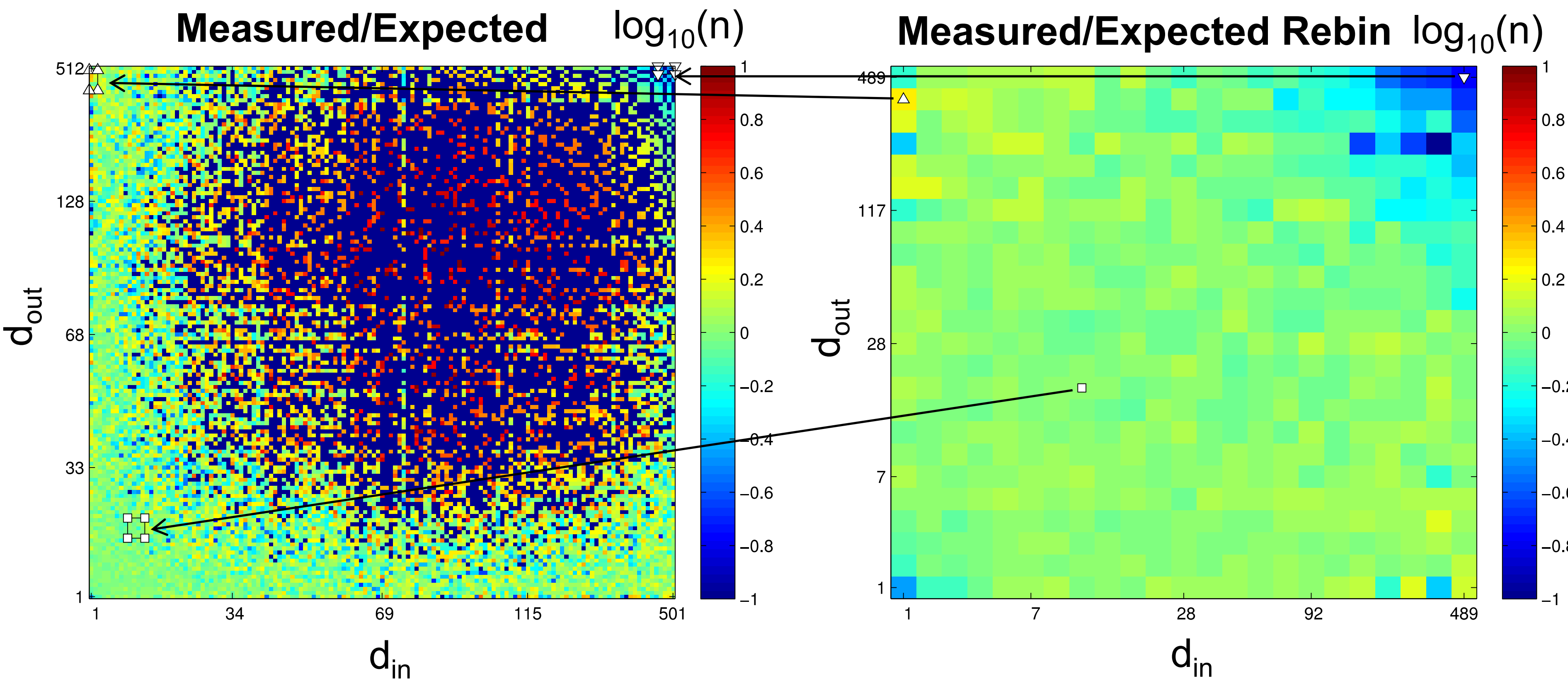
- **Measured distribution is highly sparse**
- **Rebinning based on power law fit degree bins makes most bins not empty**

Expected Joint Distribution



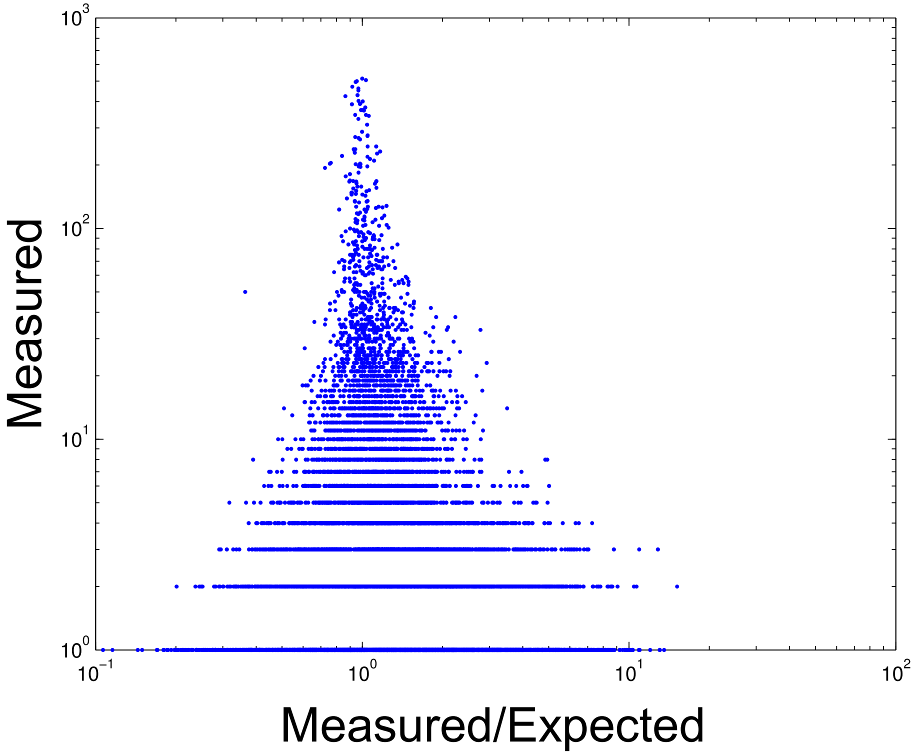
- **Using** $n(d_{out})$ **and** $n(d_{in})$ **can compute expected** $n(d_{out},d_{in}) = n(d_{out}) \times n(d_{in})/M$

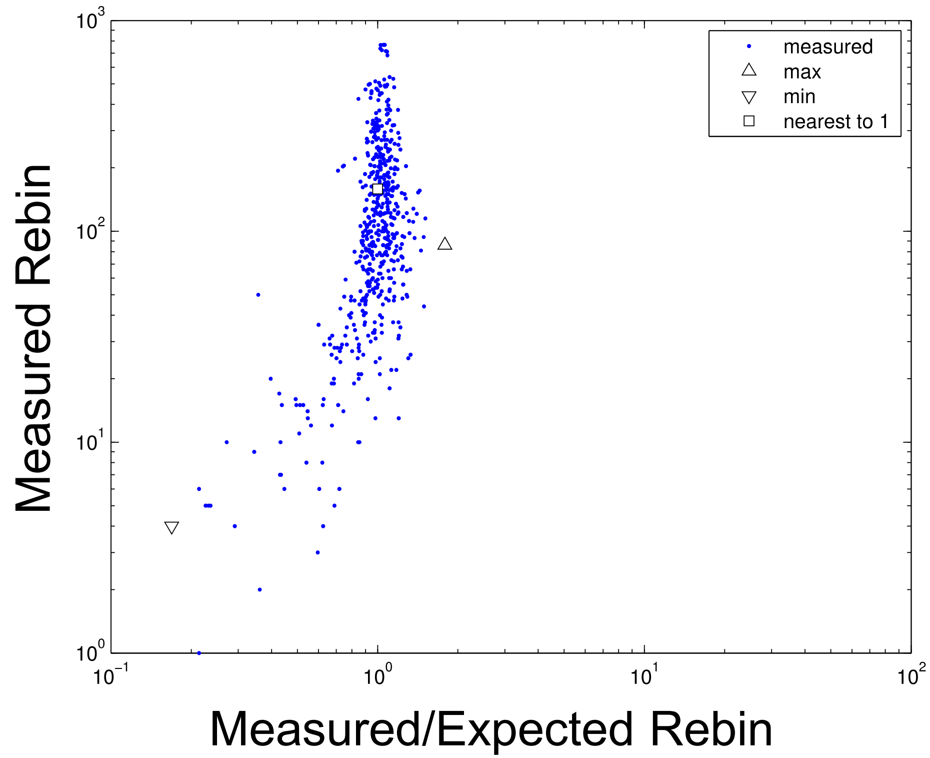
Measured/Expected Joint Distribution



- **Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □**
- **Binning reduces Poisson fluctuations and allows for more meaningful selection**

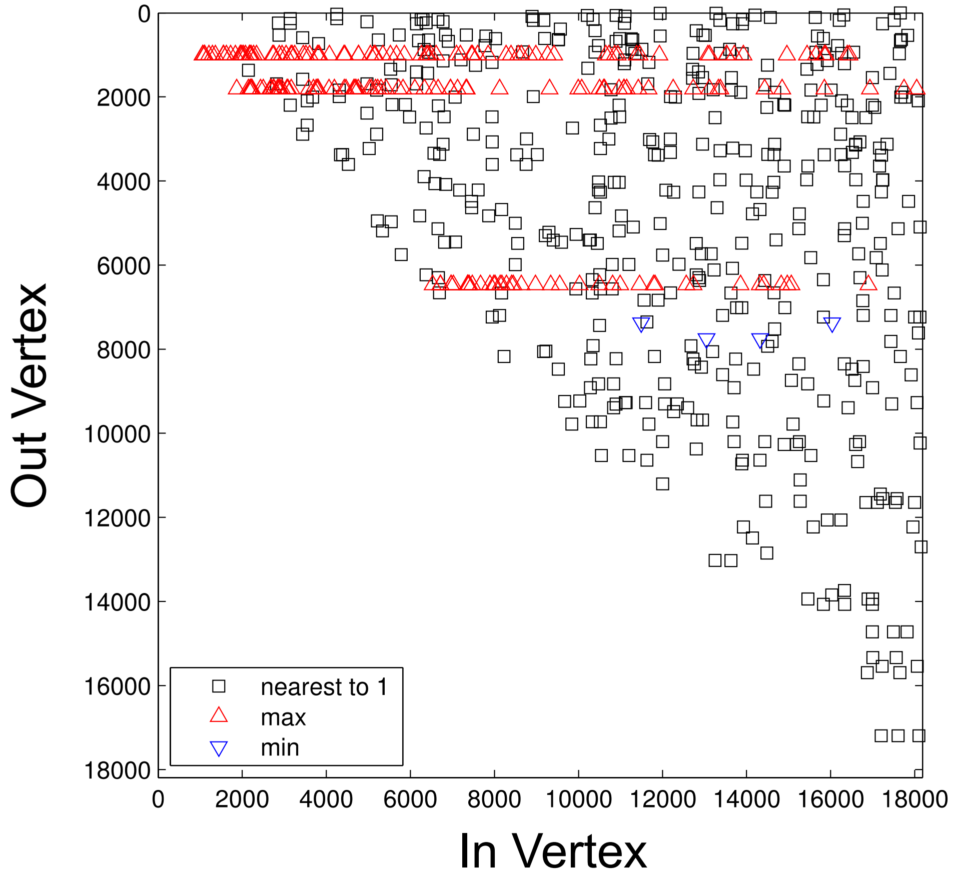
Measured/Expected Joint Distribution





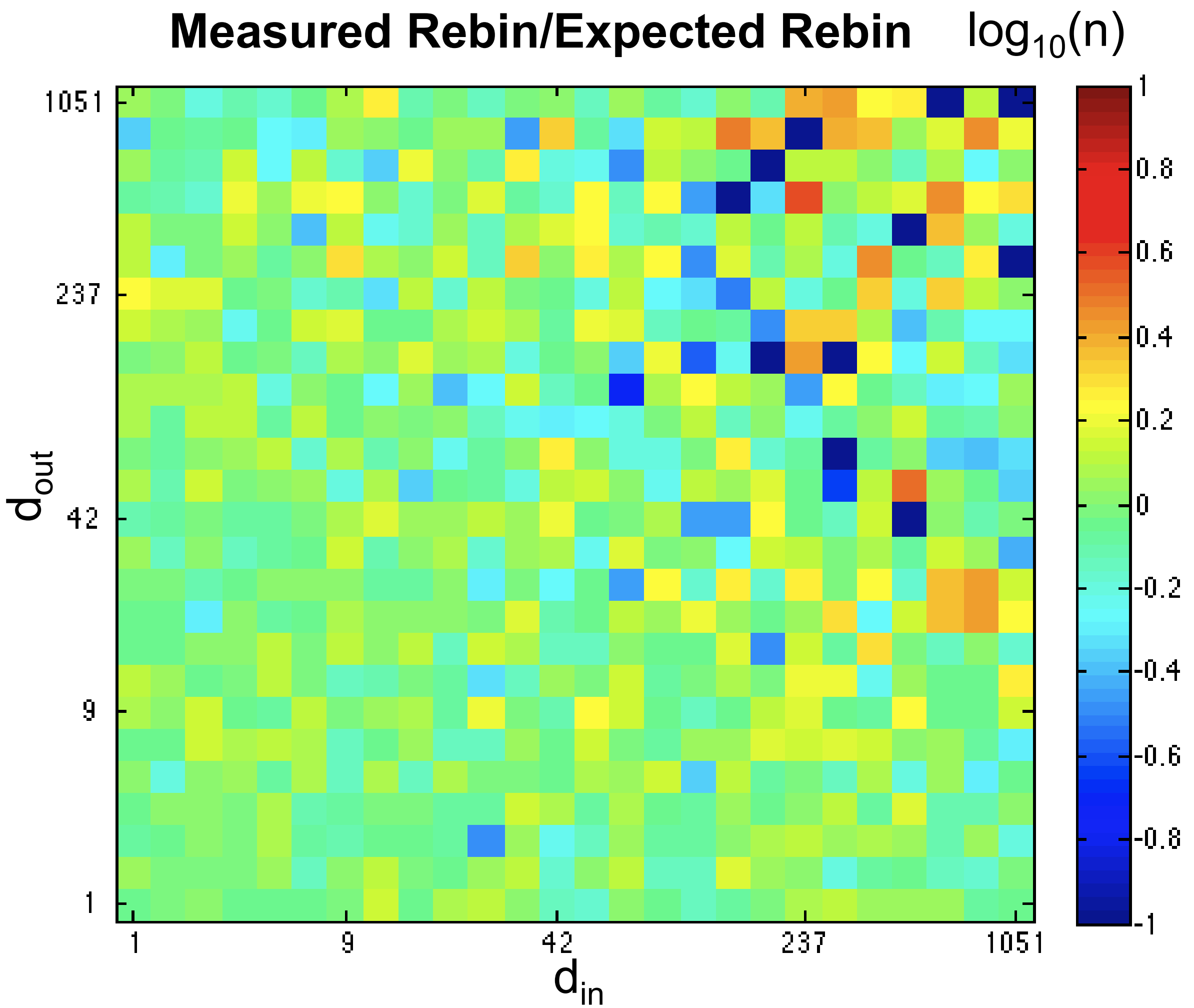
- Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □
- Binning reduces Poisson fluctuations and allows for more meaningful selection

Selected Edges



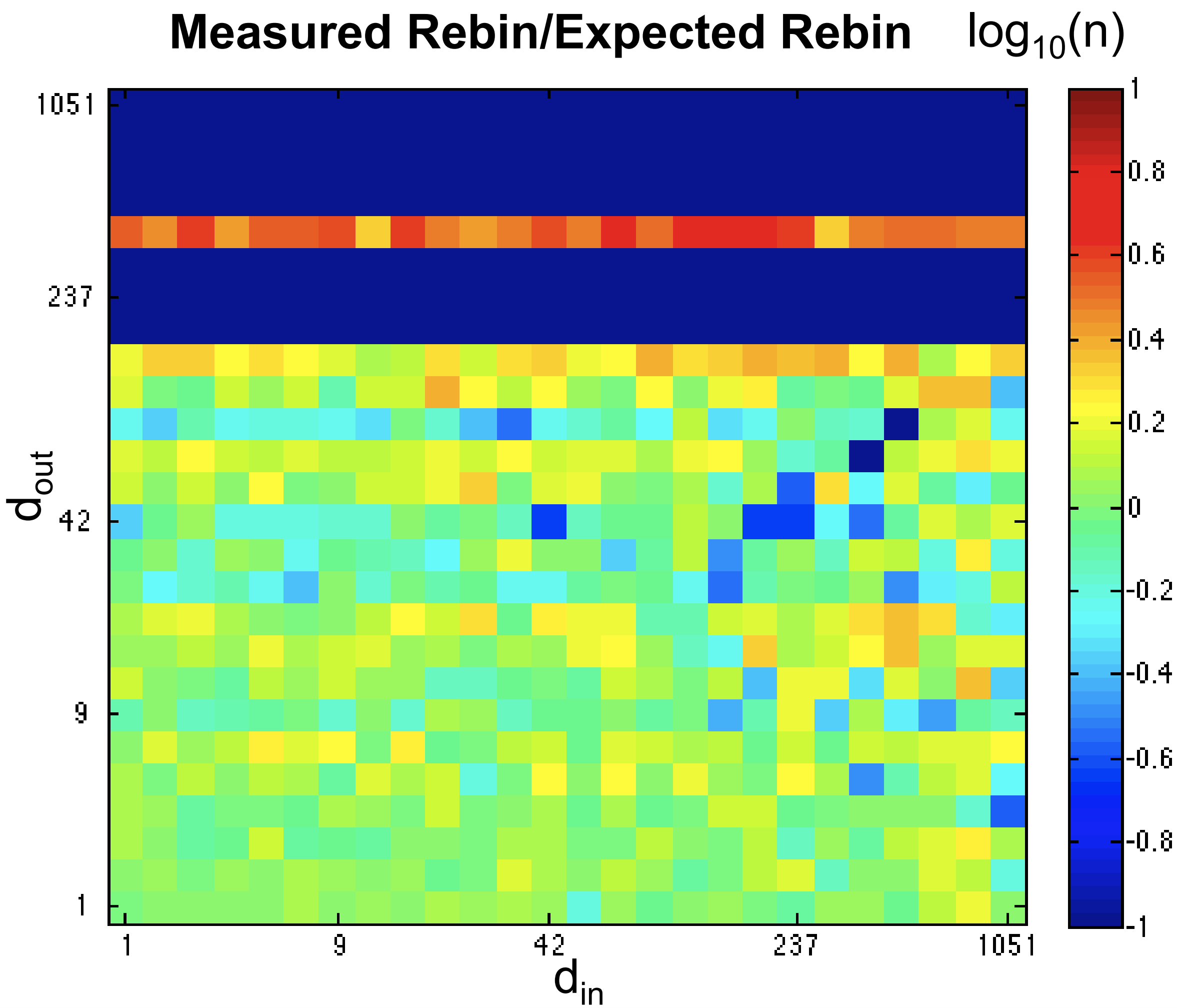
- **Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □**
- **Can use to select actual edges that correspond to fluctuations**

Measured/Expected Random Edge Order



- **Ratio of measured to expected highlights unusual correlations**

Measured/Expected Linear Edge Order



- **Ratio of measured to expected highlights unusual correlations**

Outline

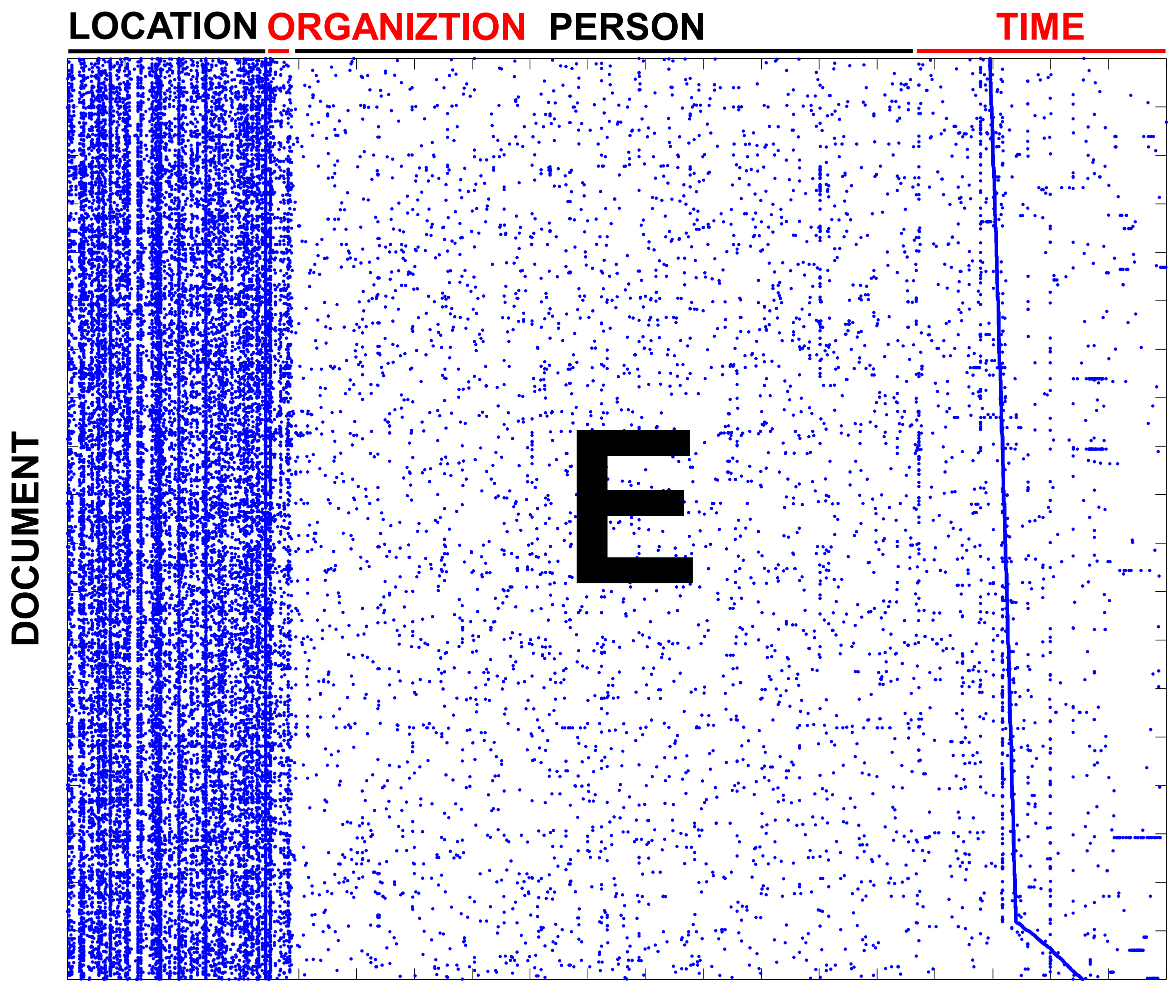
- Introduction
- Sampling
- Sub-sampling
- Joint Distribution
- **Reuter's Data**
  - *Degree distributions*
  - *Correlation Graph*
  - *Densification*
  - *Joint distributions*
- Summary

Reuter's Incidence Matrix

- **Entities extracted from Reuter's Corpus**
- **E(i,j) = # times entity appeared in document**

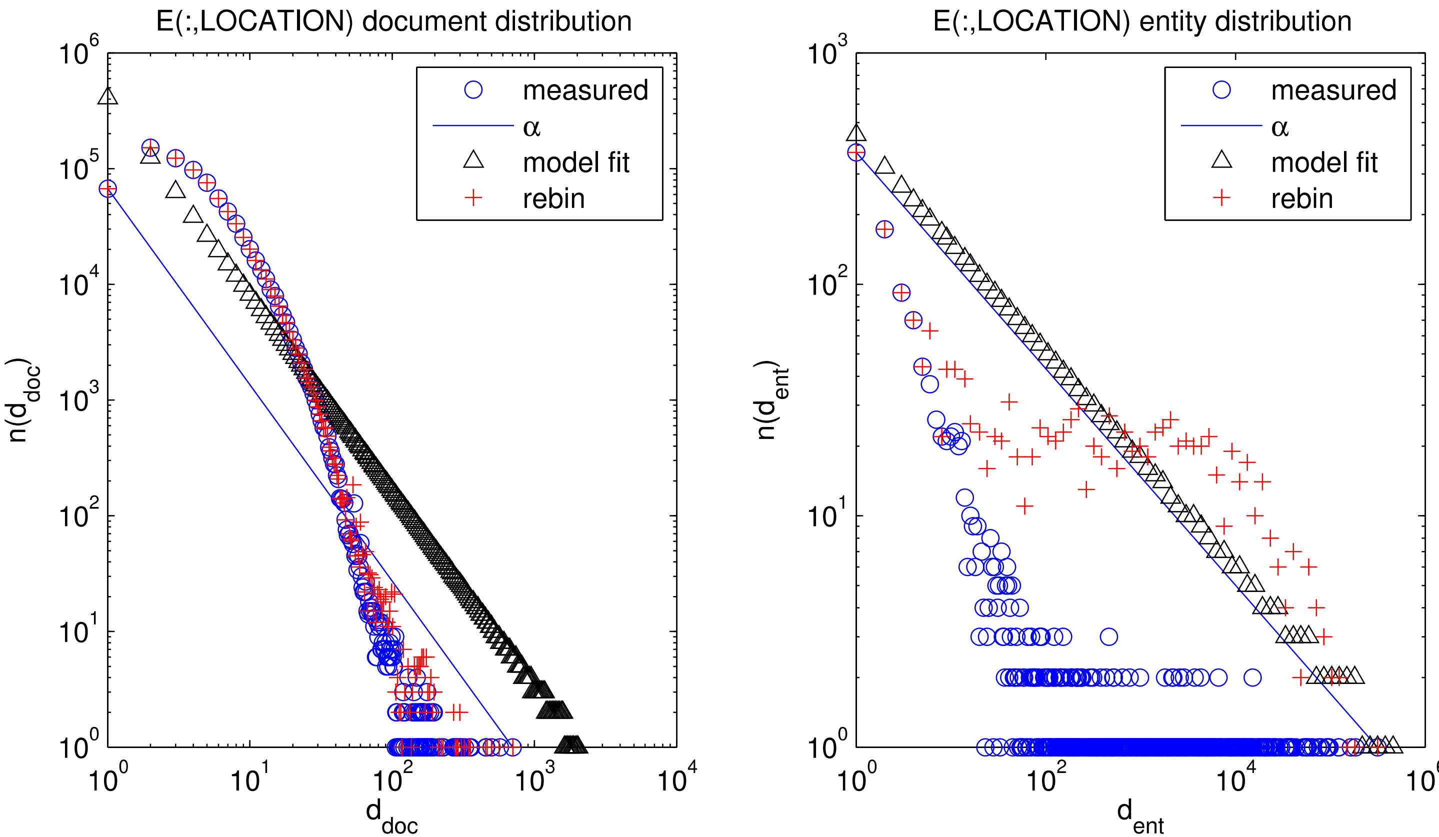
- N_{doc} = 797677
- N_{ent} = 47576
- M = 6132286

- **Four entity classes with different statistics**
  - **LOCATION**
  - **ORGANZATION**
  - **PERSON**
  - **TIME**



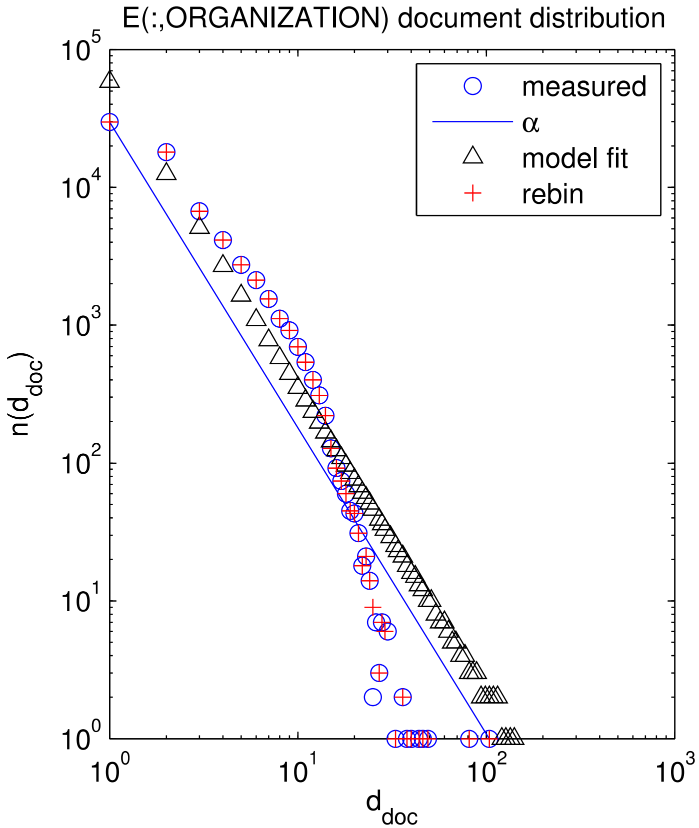
- **Fit power law model to each entity class**

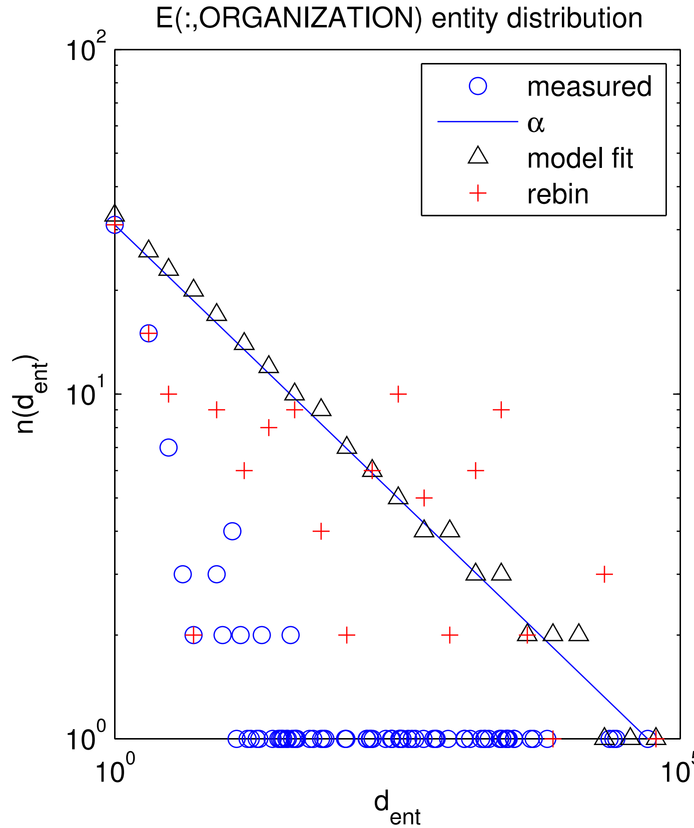
E(:,LOCATION) Degree Distribution



	M	N	M/N	α	M_{fit}	N_{fit}	M_{fit}/N_{fit}
Document	4694260	796414	5.89	1.70	4699280	811364	5.79
Entity	4694260	1786	2628	0.47	4696734	3680	1276

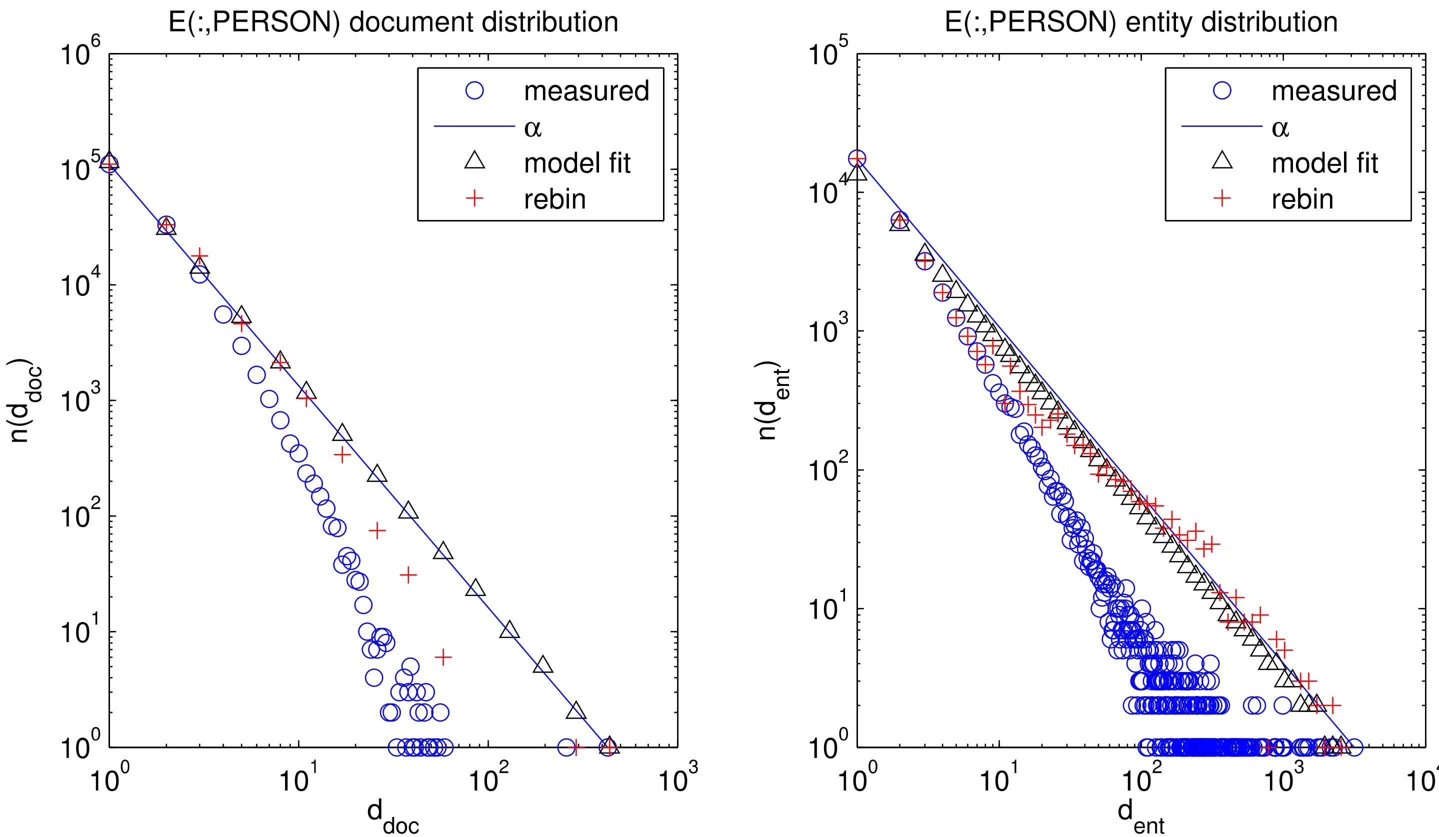
E(:,ORGANIZATION) Degree Distribution





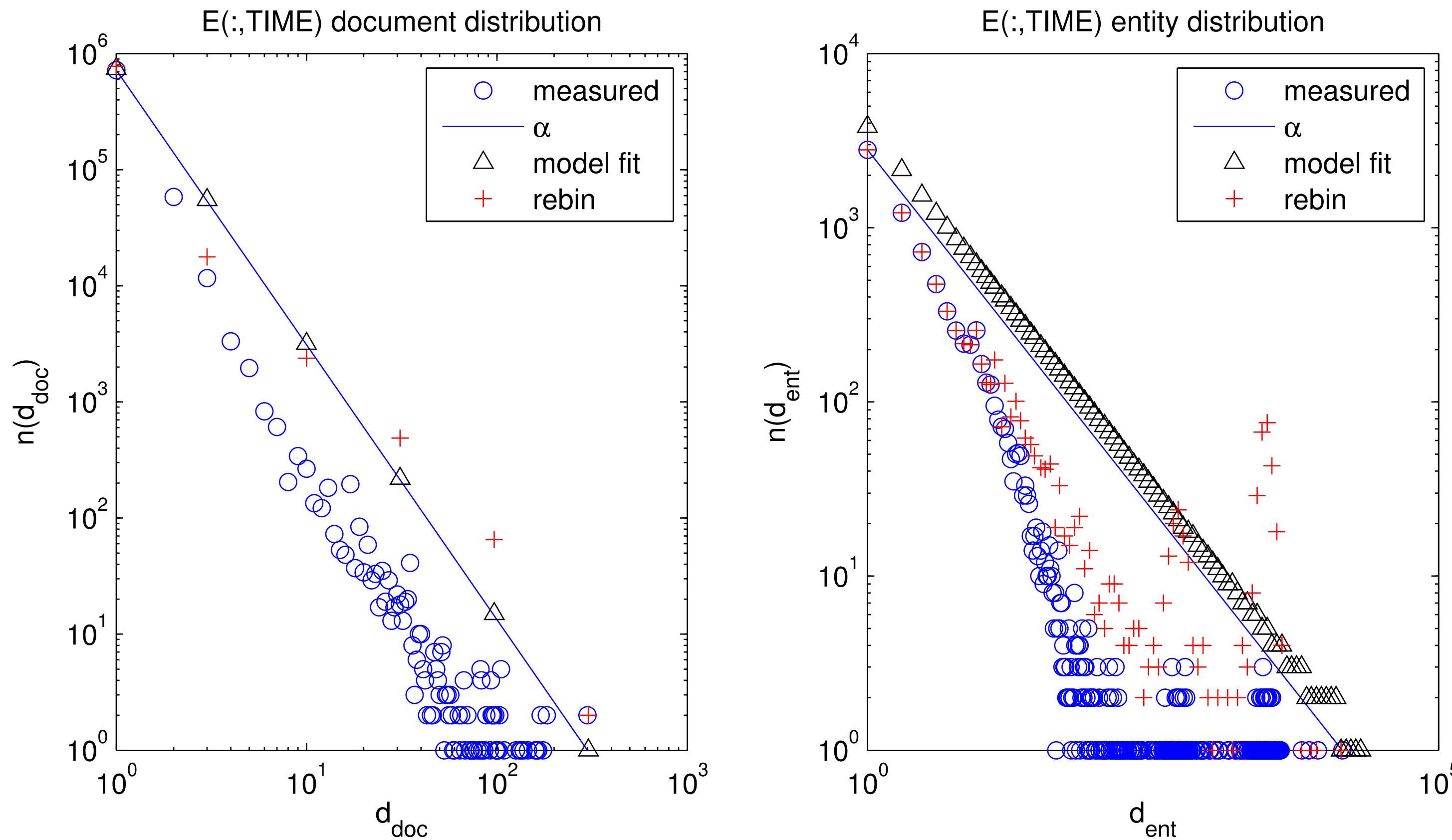
	M	N	M/N	α	M_{fit}	N_{fit}	M_{fit}/N_{fit}
Document	192390	69919	2.75	2.22	185800	85835	2.16
Entity	192390	141	1364	0.32	191943	205	936

E(:,PERSON) Degree Distribution



	M	N	M/N	α	M_{fit}	N_{fit}	M_{fit}/N_{fit}
Document	299333	170069	1.76	1.92	302478	170066	1.78
Entity	299333	37191	8.05	1.21	299748	37449	8.00

E(:,TIME) Degree Distribution



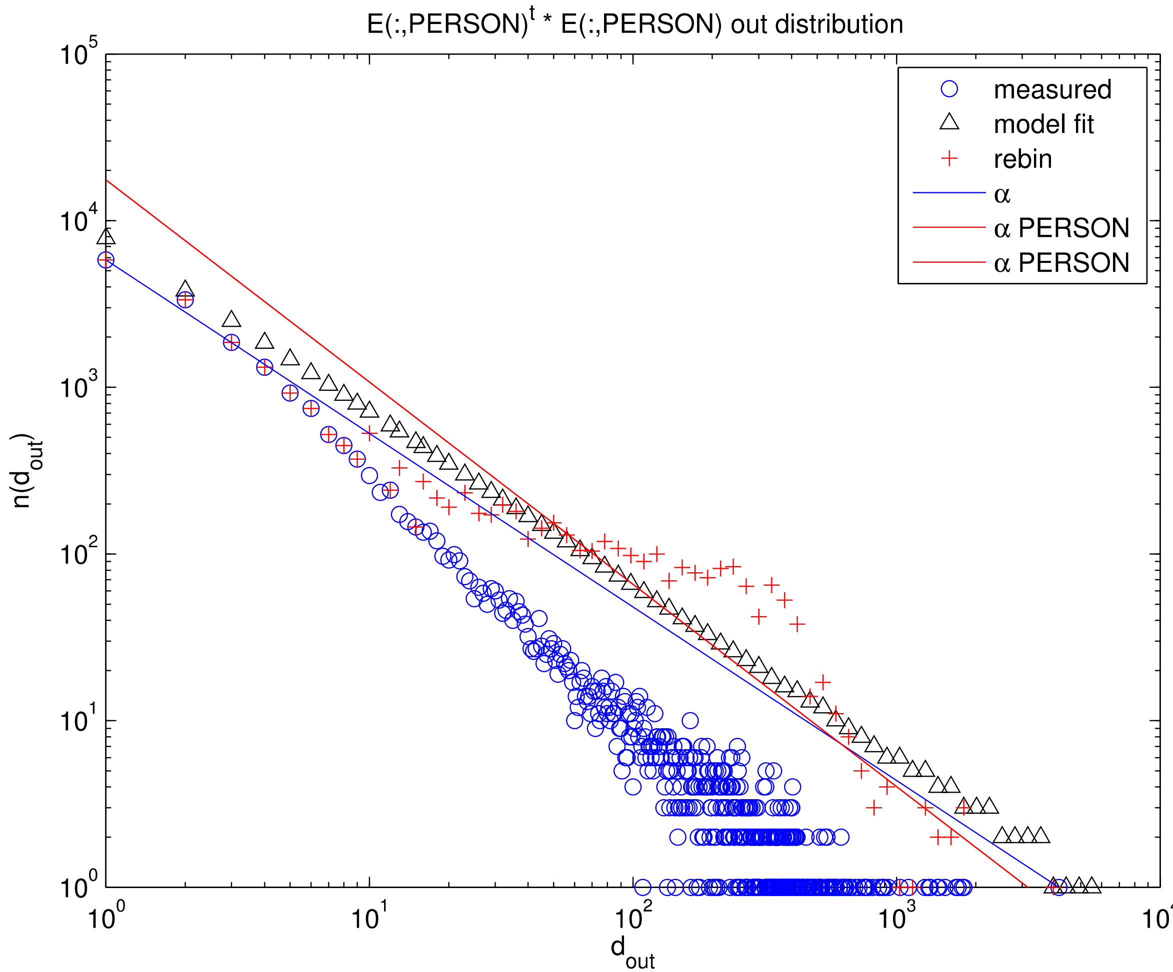
	M	N	M/N	α	M_{fit}	N_{fit}	M_{fit}/N_{fit}
Document	946299	797677	1.19	2.37	944653	797734	1.18
Entity	946299	8444	112	0.83	947711	19848	47.7

E(:,PERSON)^t x E(:,PERSON)

Procedure

- **Make unweighted and use to form correlation matrix A with no self-loops**

```
E = double(logical(E));

A = triu(E' * E);

A = A - diag(diag(A));
```



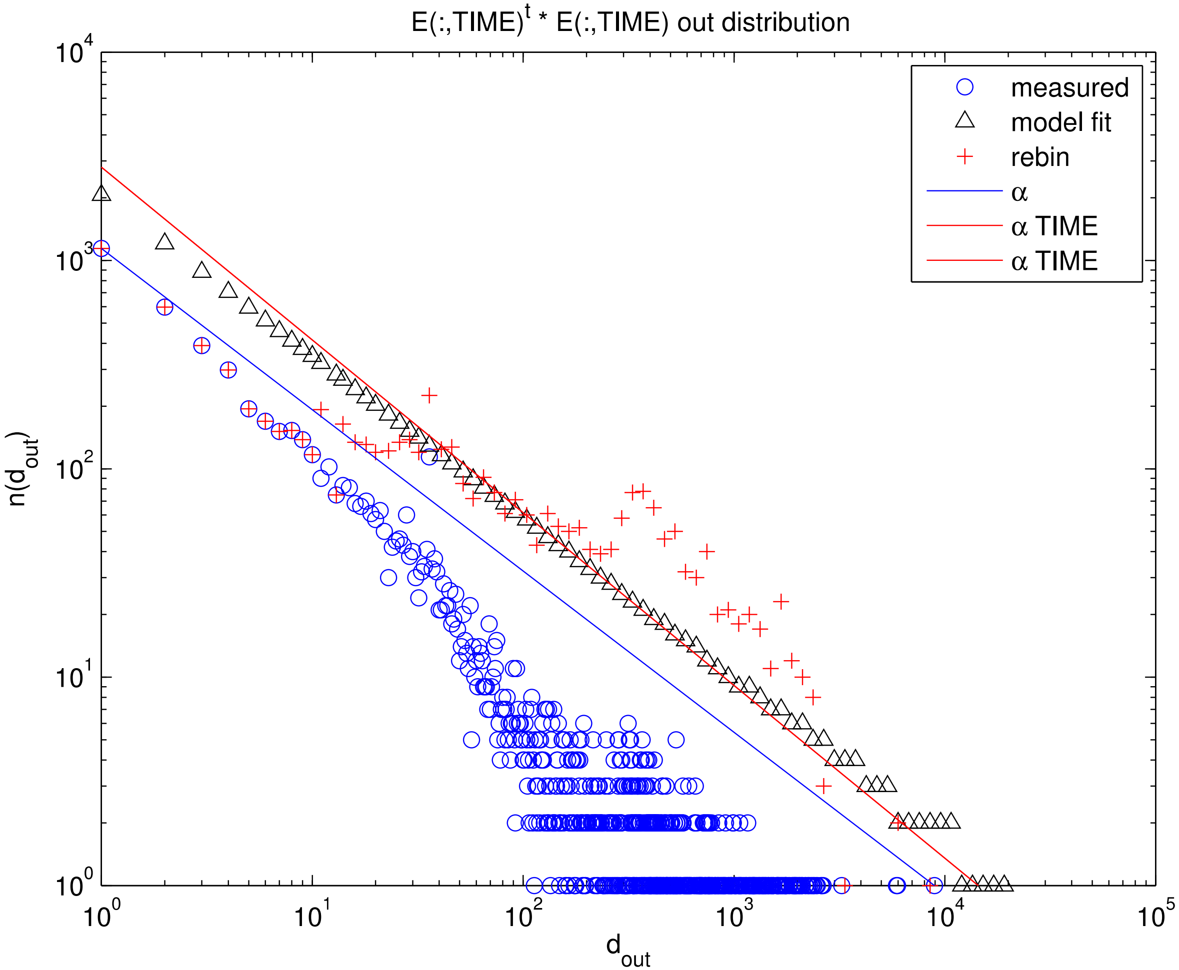
- **Perfect power law fit to correlation shows non-power law shape**
- **Reveals “witches nose” distribution**

E(:,TIME)^t x E(:,TIME)

Procedure

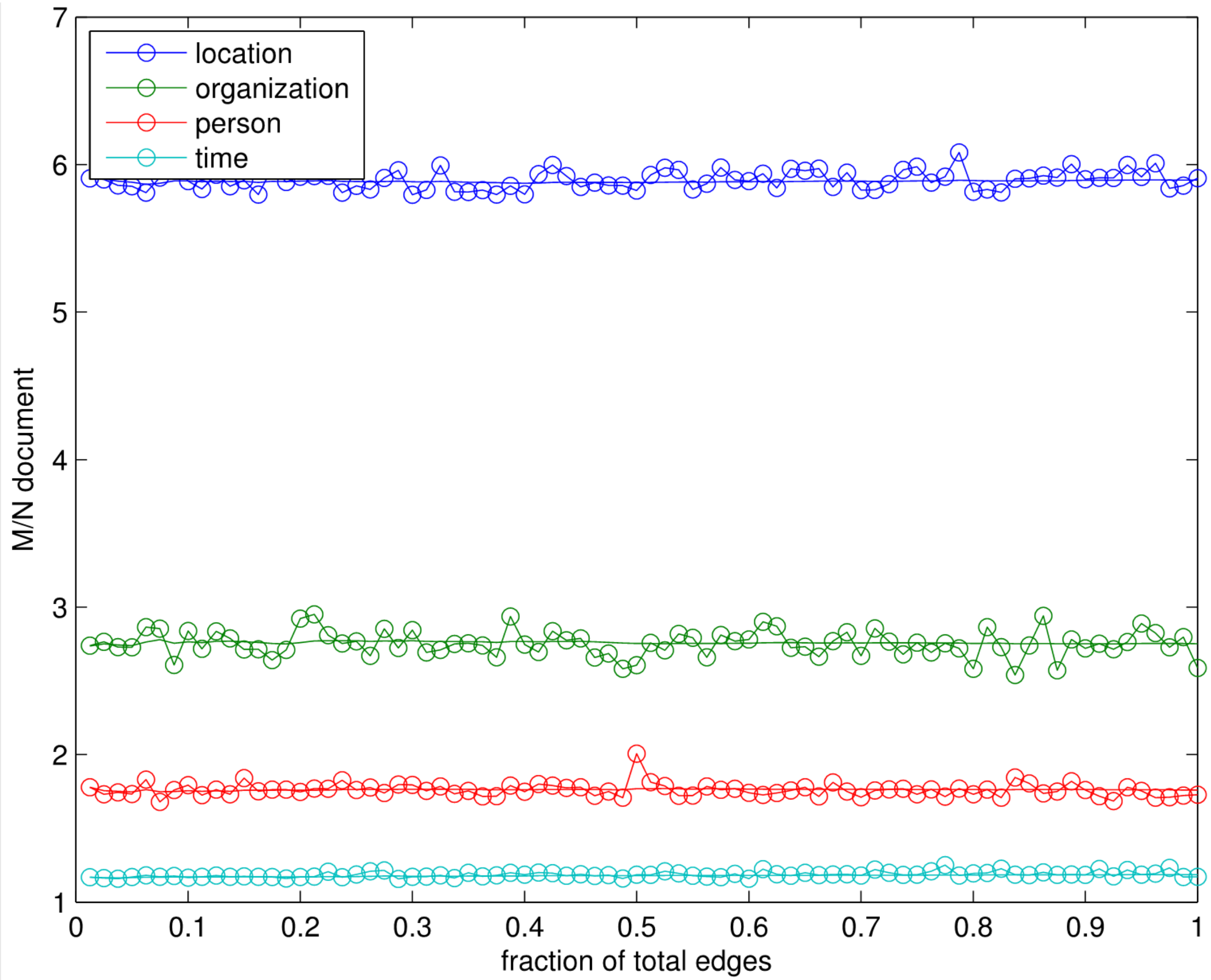
- **Make unweighted and use to form correlation matrix A with no self-loops**

```
E = double(logical(E));

A = triu(E' * E);

A = A - diag(diag(A));
```



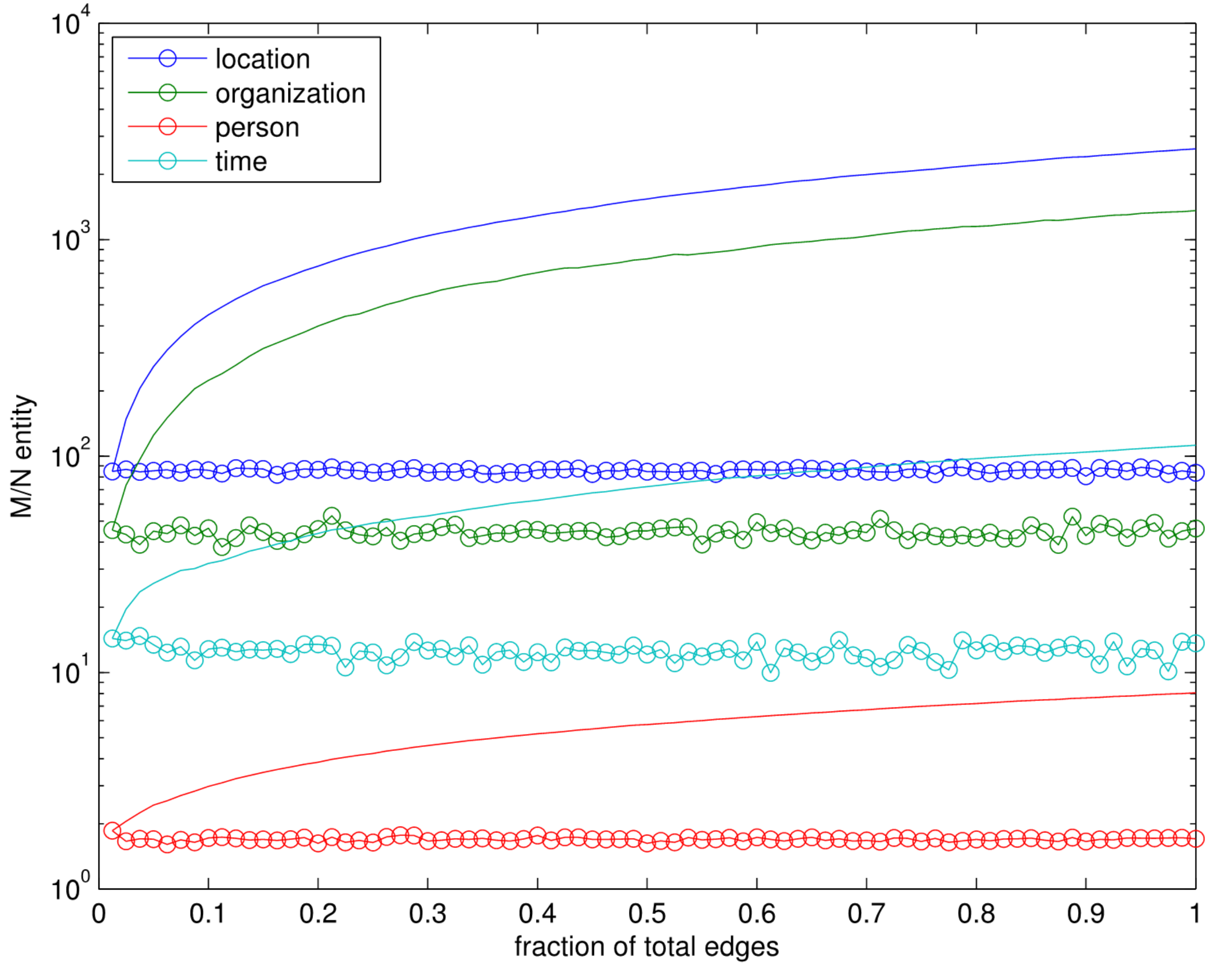
- **Perfect power law fit to correlation shows non-power law shape**
- **Reveals "witches nose" distribution**

Document Densification



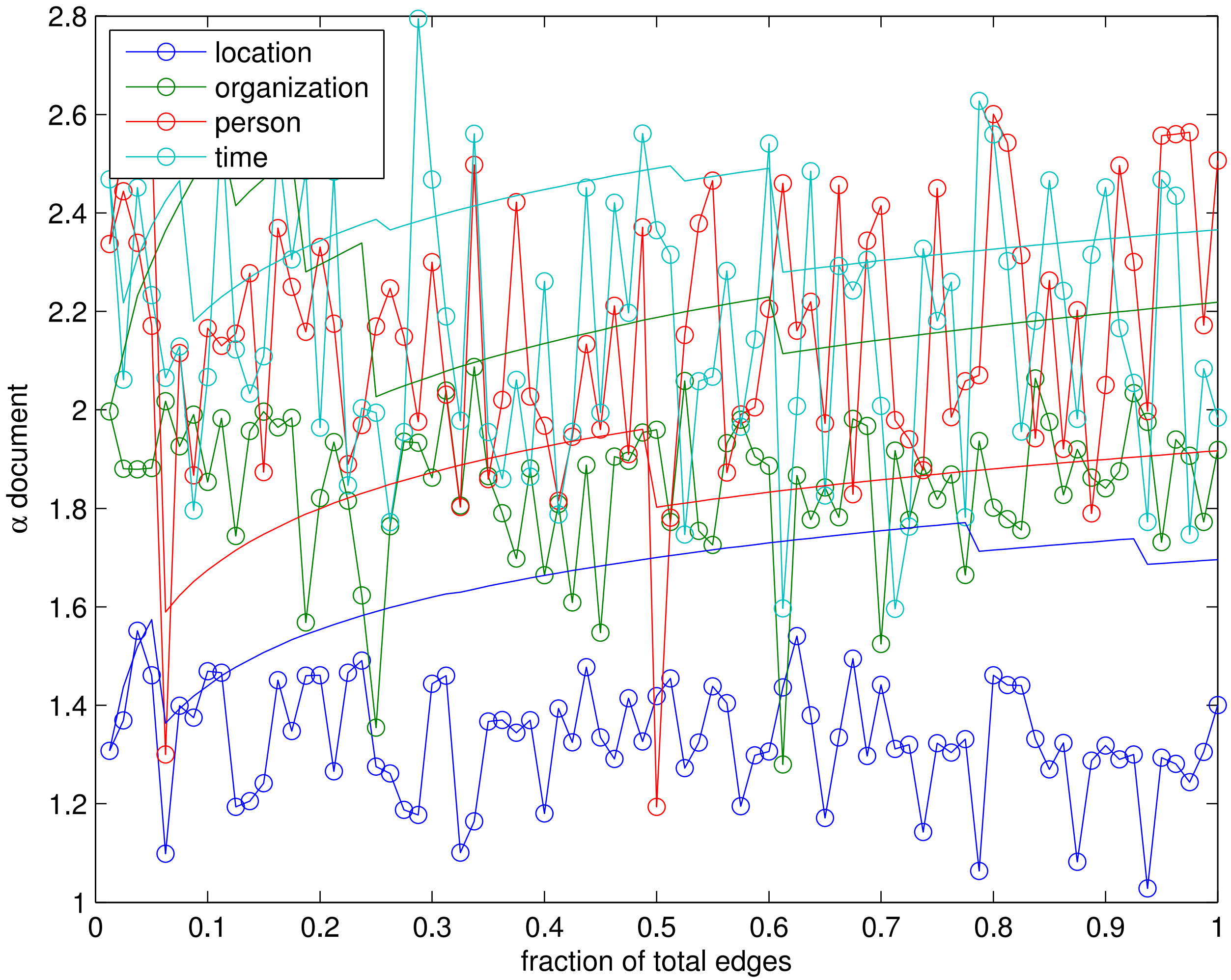
- **Constant M/N consistent with sequential ordering of documents**

Entity Densification



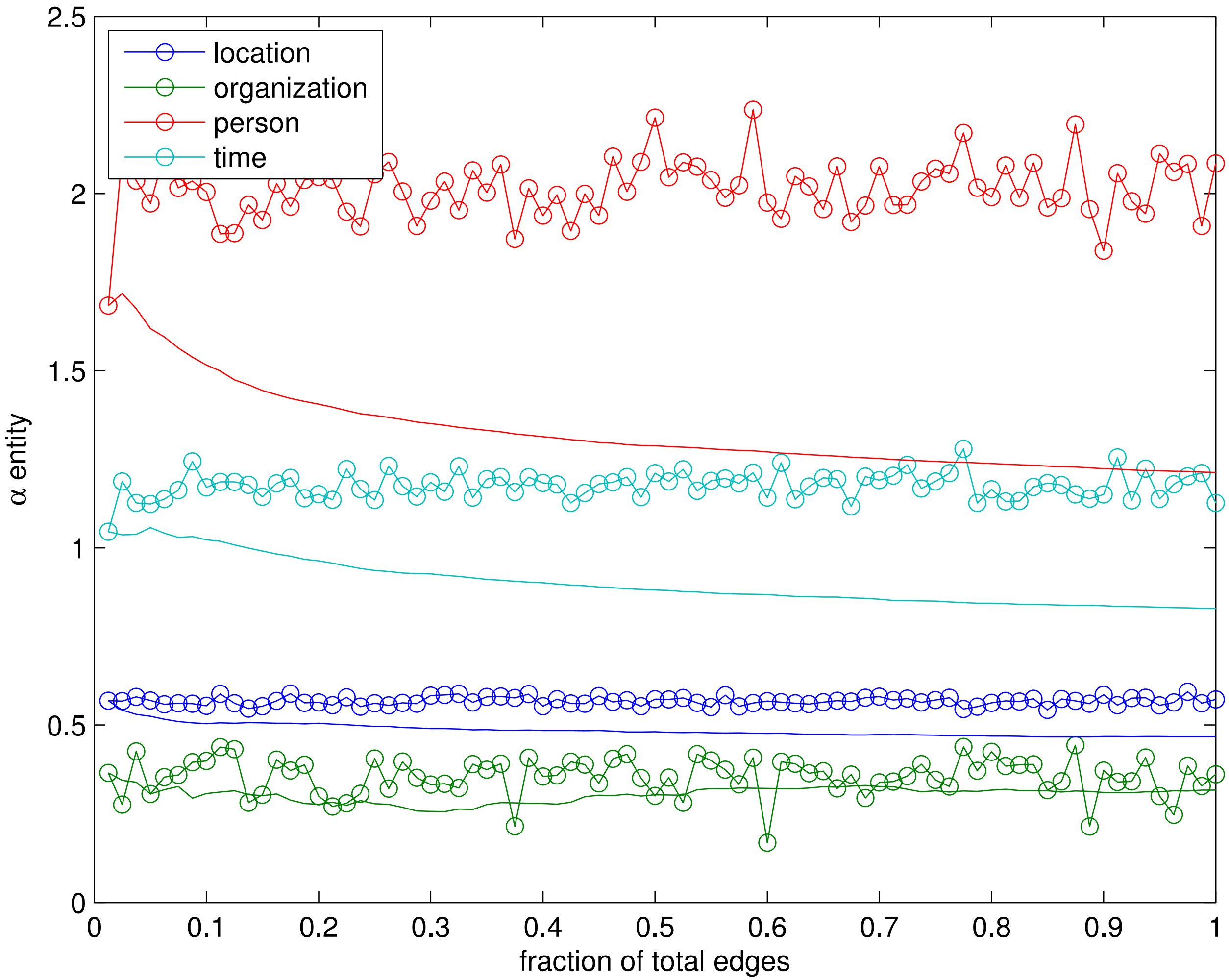
- **Increasing M/N consistent with random ordering of entities**

Document Power Law Exponent (α)



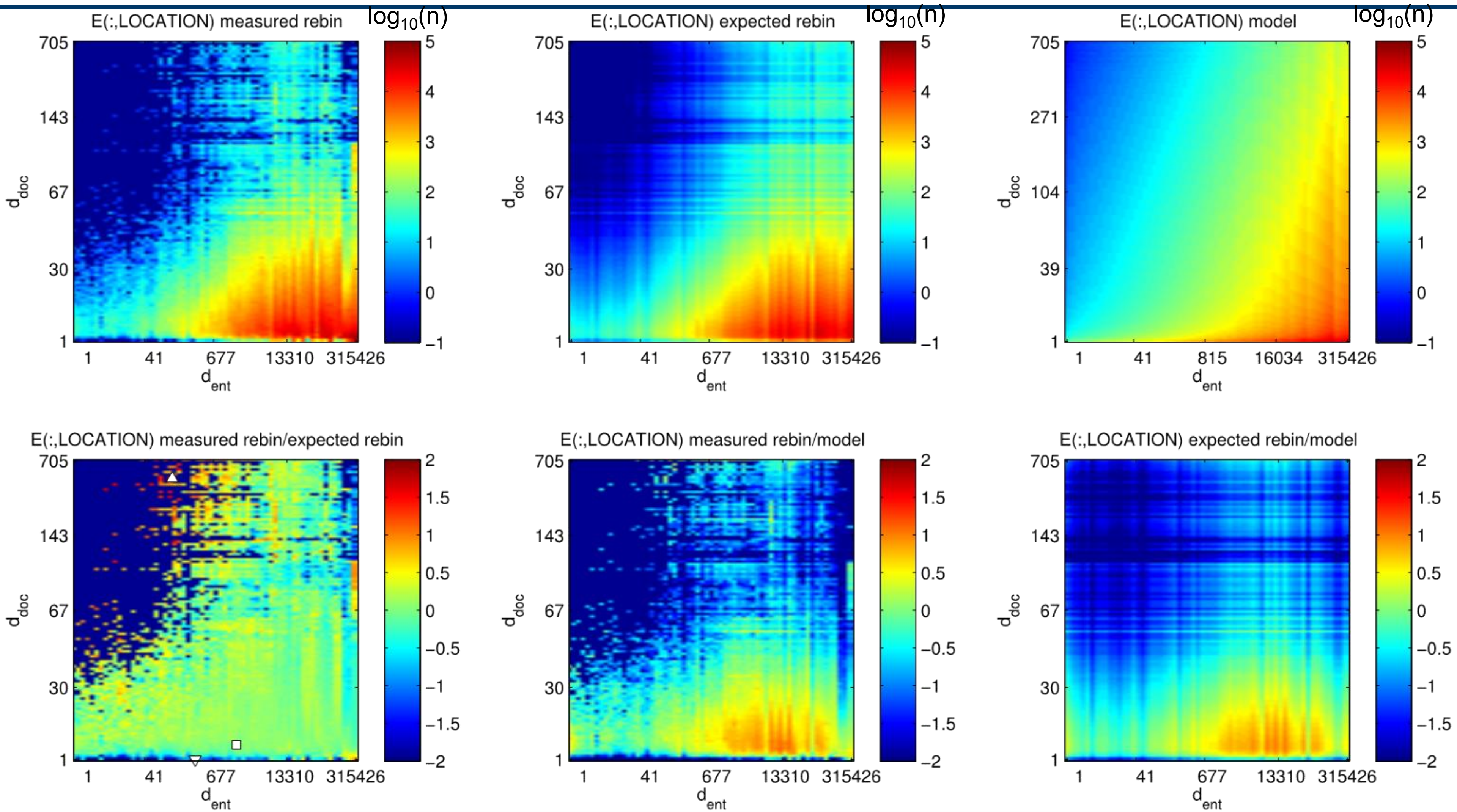
- **Increasing α consistent with sequential ordering of documents**

Entity Power Law Exponent (α)



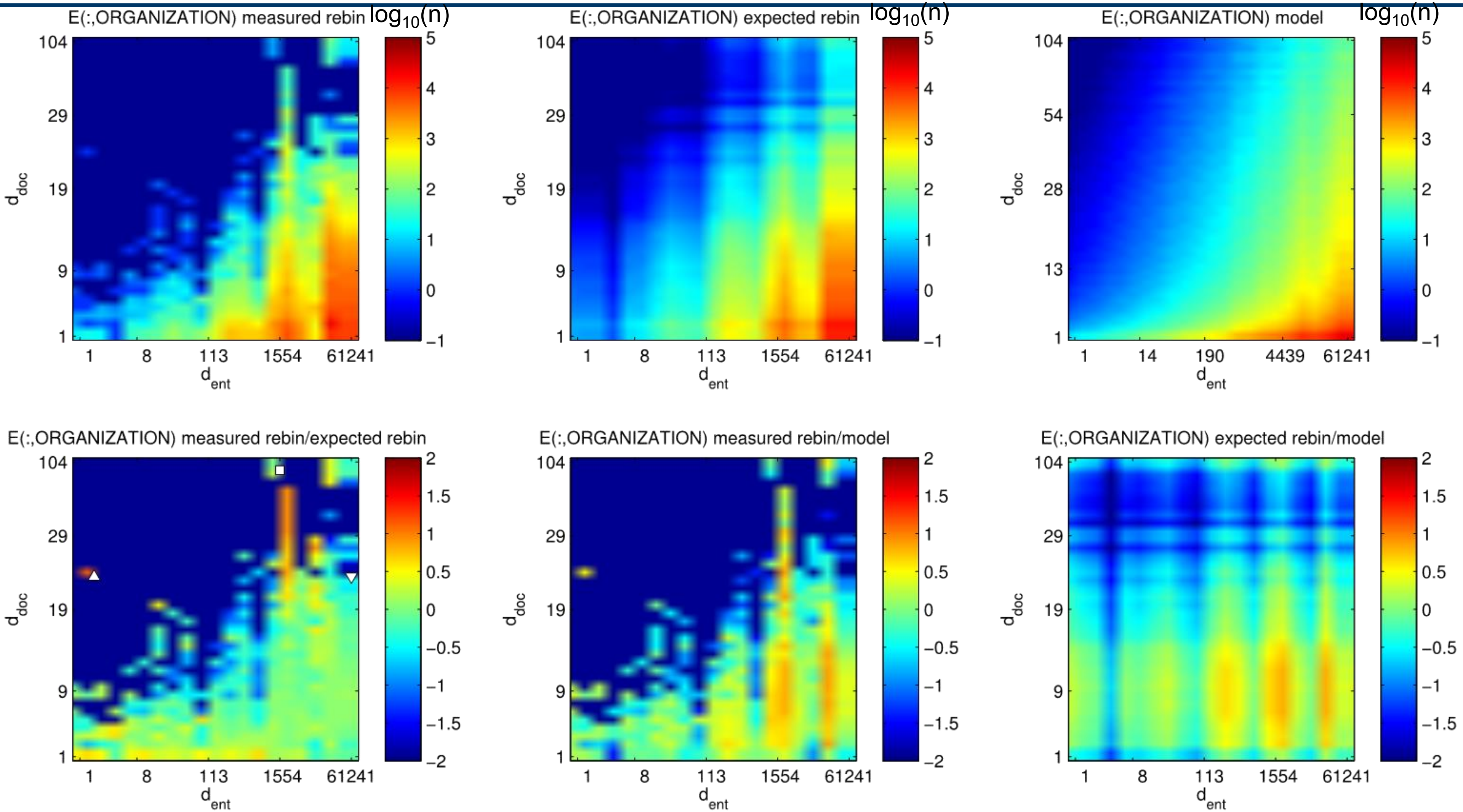
- **Decreasing α consistent with random ordering of entities**

E(:,LOCATION) Joint Distribution



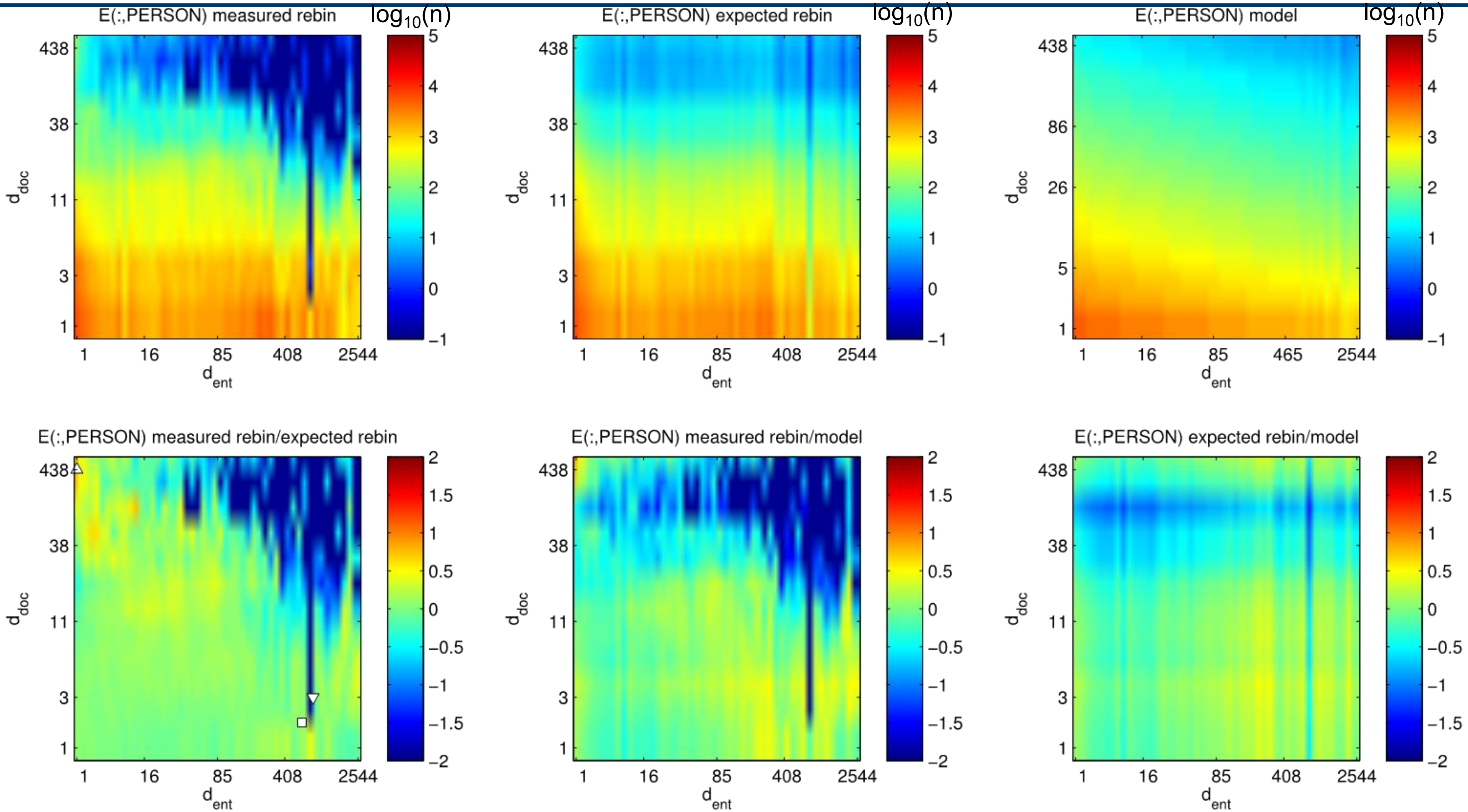
- **Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □**

E(:,ORGANIZATION) Joint Distribution



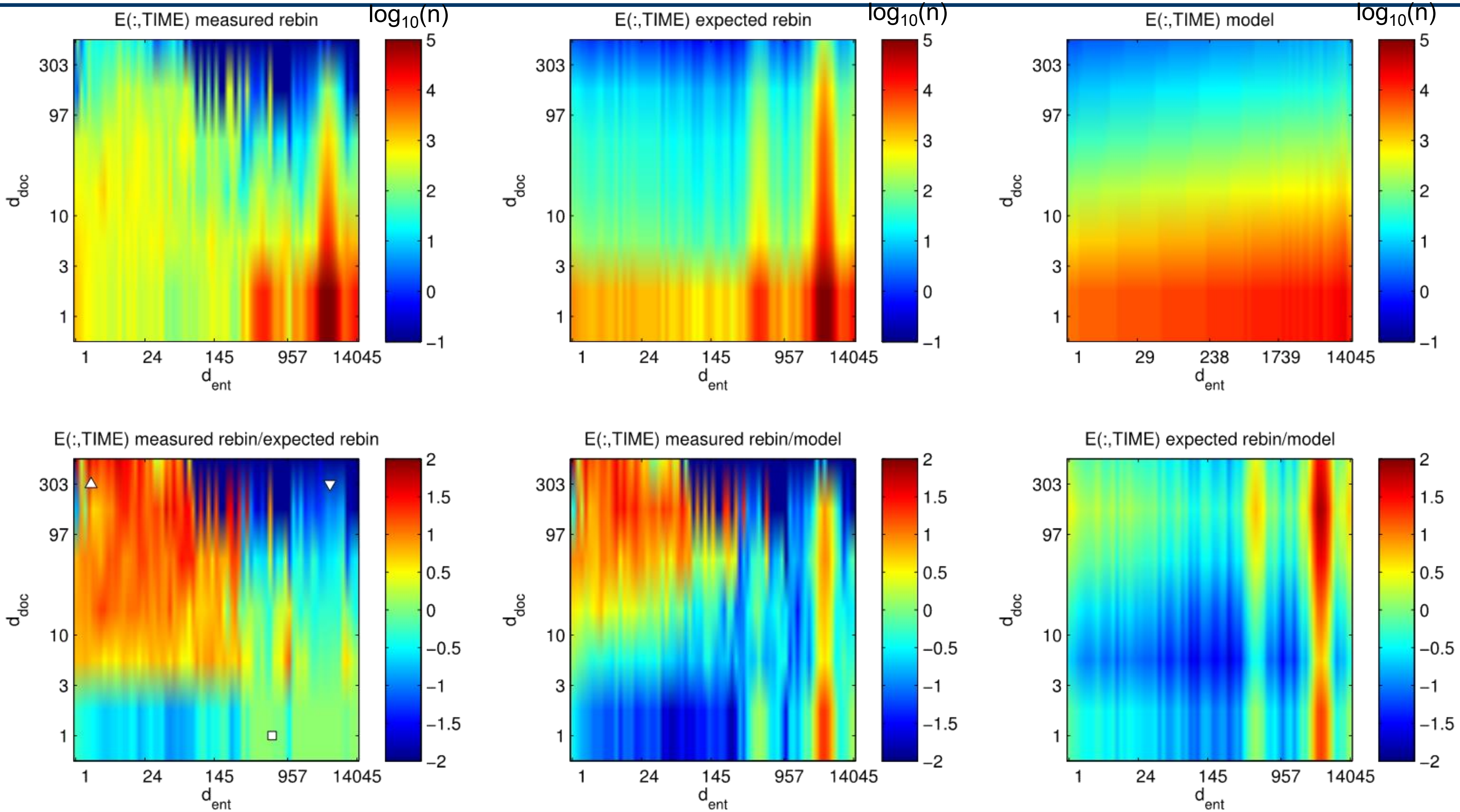
- **Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □**

E(:,PERSON) Joint Distribution



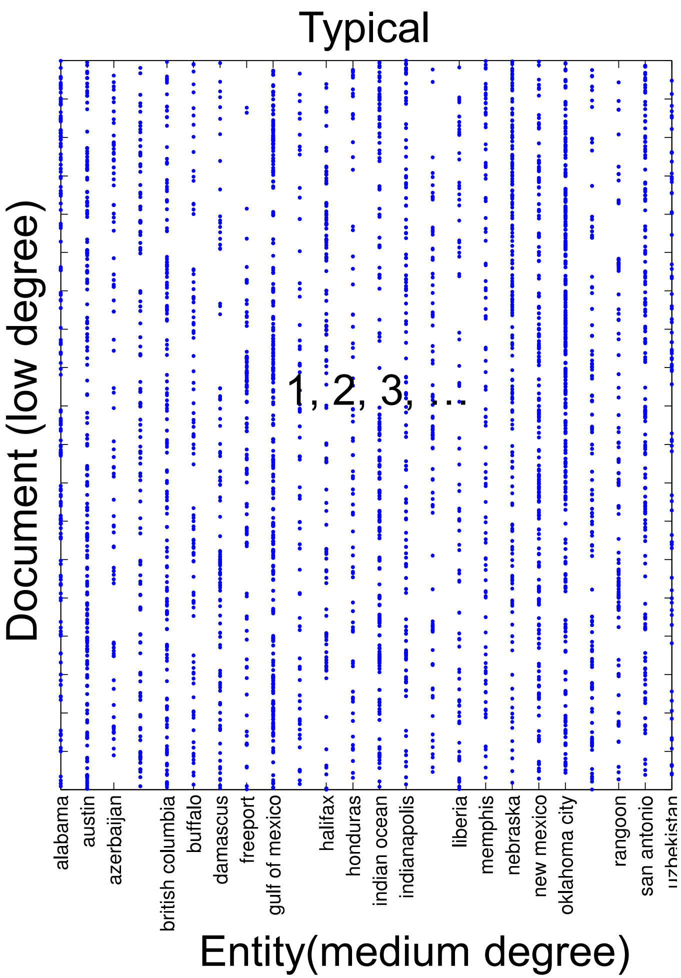
- **Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □**

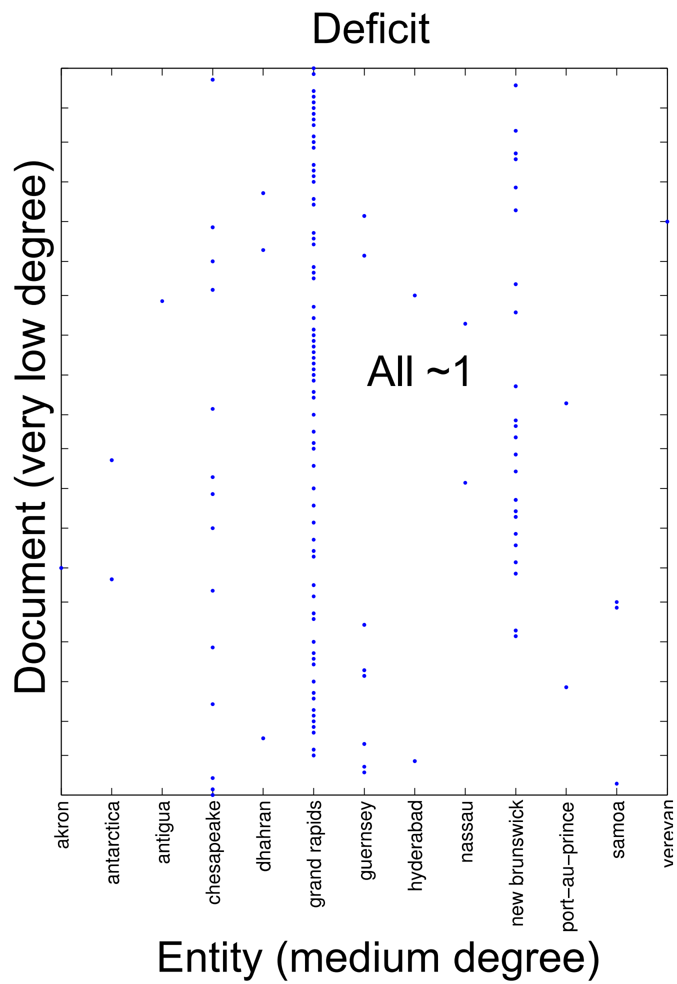
E(:,TIME) Joint Distribution



- **Ratio of measured to expected highlights surpluses △, deficits ▽, typical edges □**

Selected Edges E(:,LOCATION)





Surplus

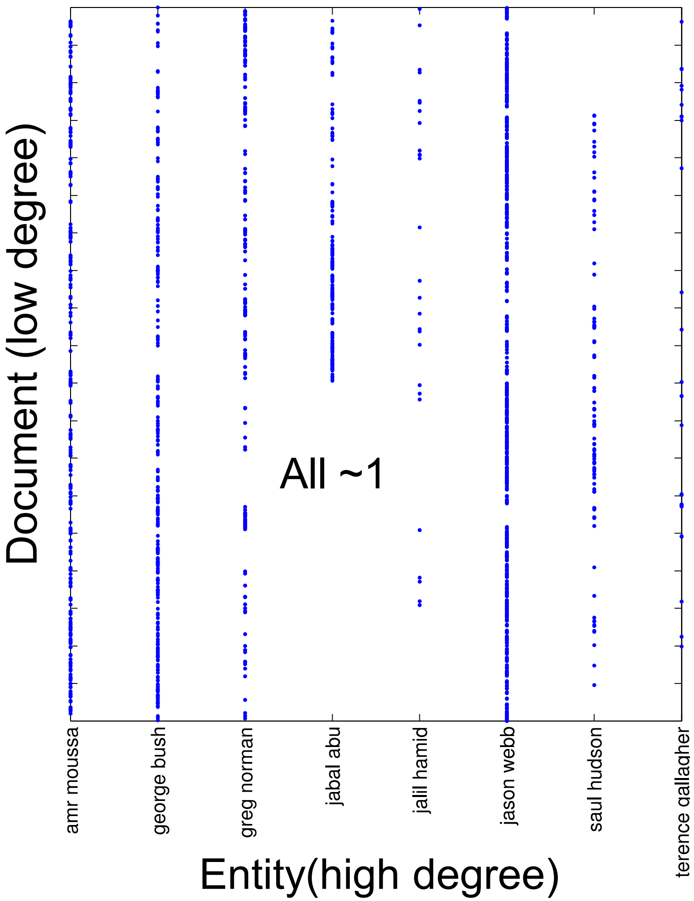
Document (very high degree)	aruba	isle of man	tahiti
19970425_538281.txt	3	6	2

Entity (medium degree)

- **Highlights anomalous edges**

Selected Edges E(:,PERSON)

Typical



Deficit

Document (low degree)	jeremy smith	samir shah
19970106_289115.txt	1	
19970313_439431.txt		1

Entity (high degree)

Surplus

Document (high degree)	adam bruce	…	bernard gentry	…	carol buchanan	…
19970502_555295.txt	1	1	1	1	1	1

Entity (low degree)

- **Highlights anomalous edges**

Summary

- Develop a background model for graphs based on "perfect" power law
  - Can be done via simple heuristic
  - Reproduces much of observed phenomena
- Examine effects of sampling such a power law
  - Lossy, non-linear transformation of graph construction mirrors many observed phenomena
- Traditional sampling approaches significantly overestimate the probability of low degree vertices
  - Assuming a power law distribution it is possible to construct a simple non-linear estimate that is more accurate
- Develop techniques for comparing real data with a power law model
  - Can fit perfect power-law to observed data
  - Provided binning for statistical tests
- Use power law model to measure deviations from background in real data
  - Can find typical, surplus and deficit edges

Example Code & Assignment

- **Example Code**
  - d4m_api/examples/2Apps/3PerfectPowerLaw
- **Assignment 4**
  - Compute the degree distributions of cross-correlations you found in Assignment 2
  - Explain the meaning of each degree distribution