Differentially Private Mechanisms for Cut-Queries of a Graph

A Survey

Or Sheffet

Harvard university

osheffet@seas.harvard.edu

Charles River Workshop on Private Analysis of Social Networks

May 19^{th} 2014

Privacy in Graphs

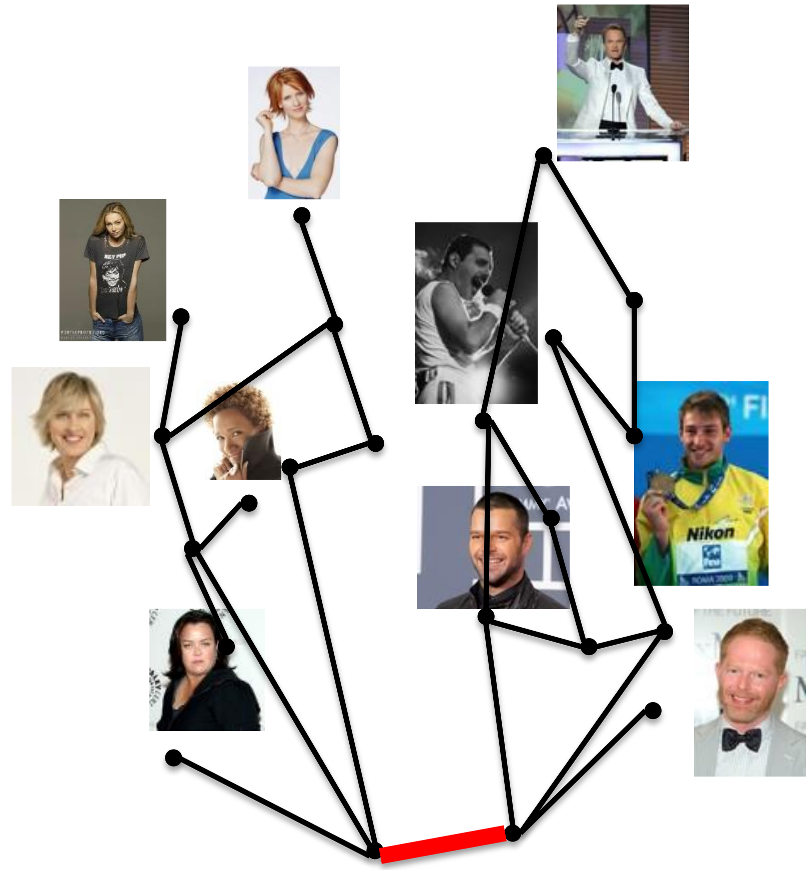
Neighboring graphs:

Differ on a single edge

- Goal: satisfy (ϵ,δ)-privacy w.r.t edge changes, while approximating cut queries

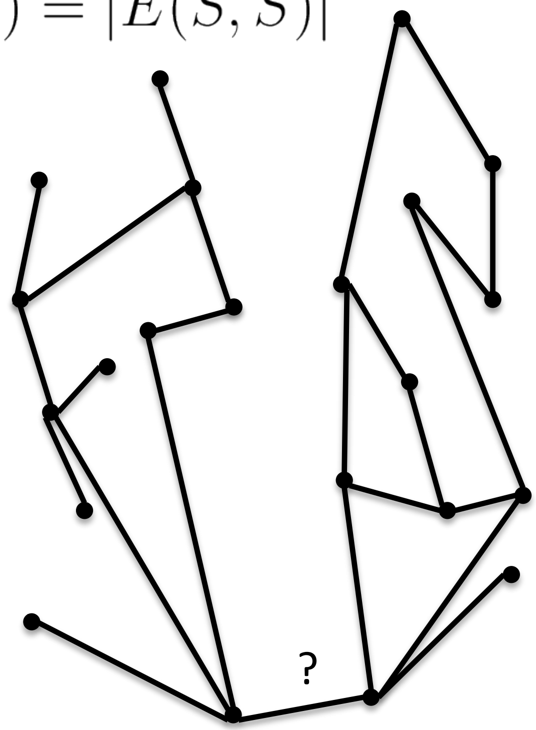
$$\Phi_G(S) = |E(S,\bar{S})|$$

$$\Phi_G(S) = \sum_{a \in S, b \in \bar{S}} w_{a,b}$$



The Right goal?

- Goal: satisfy (ϵ,δ)-privacy w.r.t edge changes, while approximating cut queries $\Phi_G(S) = |E(S, \bar{S})|$

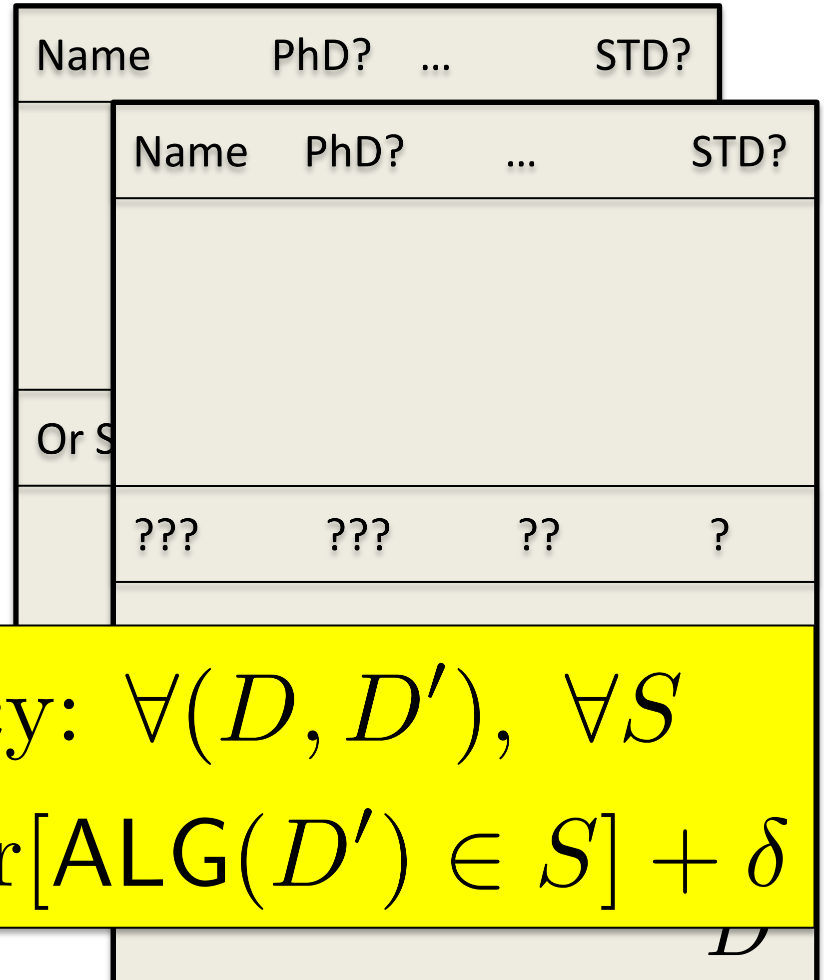


- Cuts help in divide-and-conquer
- Cuts – communities and clustering
  - Need extra info: avg degree in subgraph
- Error: max-error over all cuts

  $\max_{S \in Q} |\Phi_G(S)$ - $\mathrm{answer}(S)|$
- Helps in making qualitative observations???
  – You tell me / us!

Differential Privacy

- n people
- Neighboring datasets:
  - x changes



[DMNS06, DKMMN06]

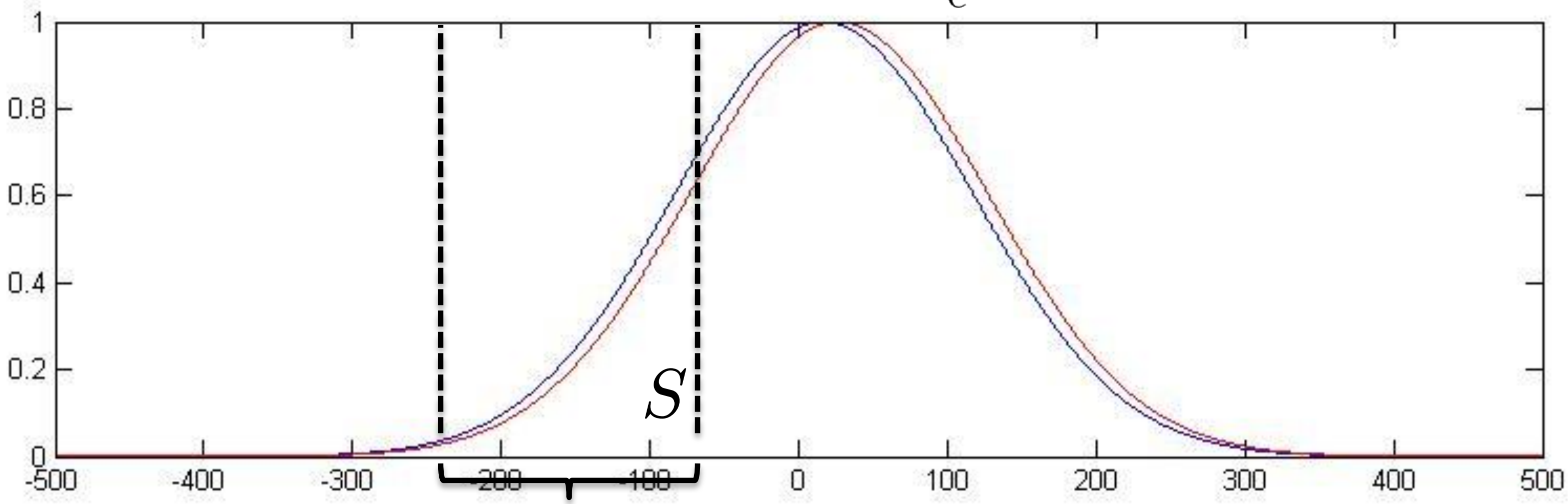
(ϵ, δ)-differential privacy: $\forall (D, D'),\ \forall S$

$$\Pr[\mathsf{ALG}(D) \in S] \le e^{\epsilon} \Pr[\mathsf{ALG}(D') \in S] + \delta$$

Differential Privacy

- Def: $GS(f) = \max_{(D,D')} |f(D) - f(D')|$
- The basic mechanism: Given f, answer:

$$f(D) + GS(f) \cdot \mathcal{N}(0, \frac{\log(1/\delta)}{\epsilon^2})$$



Differential Privacy

- Def: $GS(f) = \max_{(D,D')} |f(D) - f(D')|$
- The basic mechanism: Given f, answer:

$$f(D) + GS(f) \cdot \mathcal{N}(0, \frac{\log(1/\delta)}{\epsilon^2})$$

What can we do when $GS(f)$ is big?

How to answer $f_1, f_2, \ldots, f_t$?

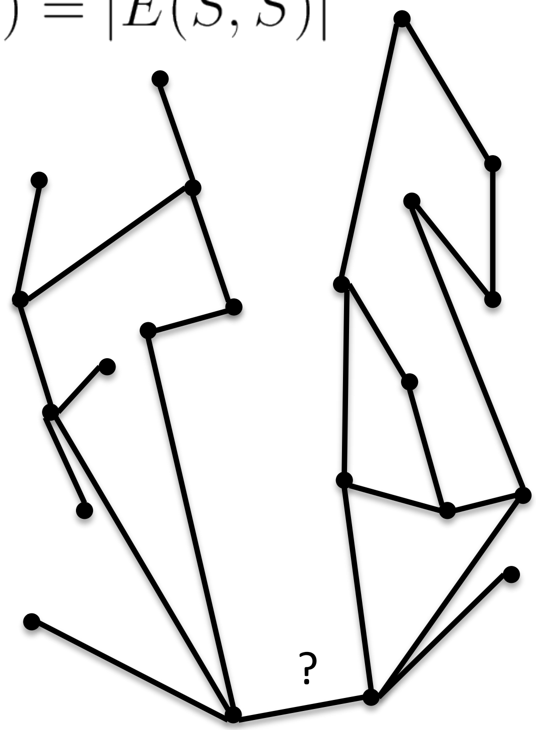
[DVR10] –noise proportional to √t

[BLR08, HR10] – inefficient/efficient technique to answer general/linear queries with noise proportional to $\log(t)$

Graphs = Matrices

- <u>Goal:</u> satisfy (ϵ,δ)-privacy w.r.t edge changes, while approximating cut queries $\Phi_G(S) = |E(S,\bar{S})|$
- A_G adjacency-matrix

$$\begin{array}{c|ccc|}
 & a & b & b' \\
\hline
a & & 0 & 1 \\
b & 0 & & \\
b' & 1 & & \\
\hline
\end{array}$$



Adjacency Matrix and Cuts

$\mathbf{1}_S$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

$$\begin{array}{c} \\ a \\ b \\ b' \end{array} \begin{array}{c} \begin{array}{ccc} a & b & b' \end{array} \\ \begin{bmatrix} & 0 & 1 \\ 0 & & \\ 1 & & \end{bmatrix} \end{array}$$

$\mathbf{1}_T$

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \sum_{\substack{i\in S \\ j\in T}} A_{i,j}$$

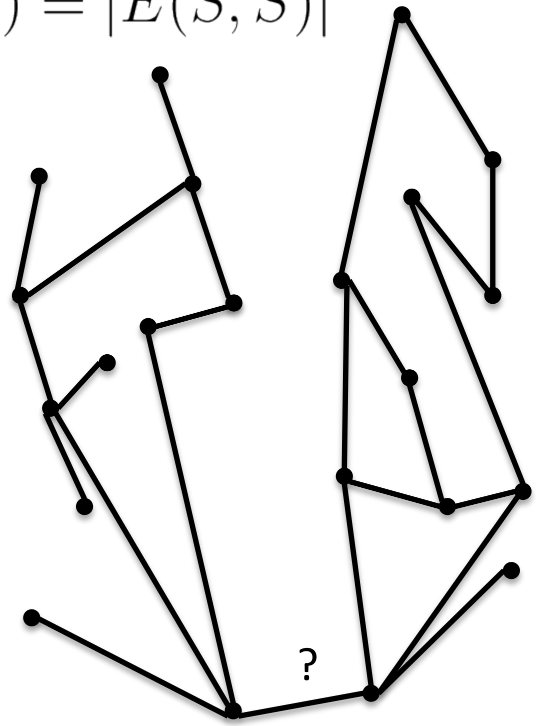
Release A' that approximates A well w.r.t cut-norm:

$$\|A - A'\|_{\square} = \max_{S,T} \Big| \sum_{\substack{i\in S \\ j\in T}} (A - A')_{i,j} \Big|$$

Graphs = Matrices

- <u>Goal:</u> satisfy (ϵ,δ)-privacy w.r.t edge changes, while approximating cut queries $\Phi_G(S) = |E(S, \bar{S})|$
- E_G Edge-matrix

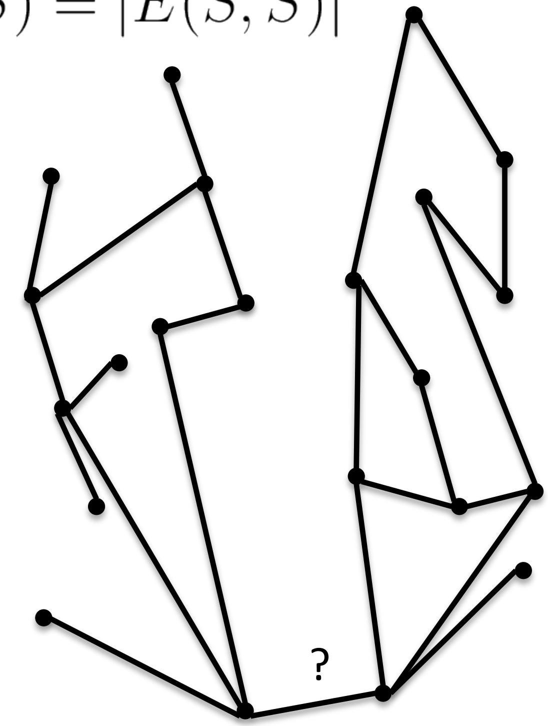
$$\begin{array}{ccc} & n & \\ \binom{n}{2} & \left[\begin{matrix} 0 \ldots 0 \;\; \ldots \;\; \ldots \ldots \ldots \ldots 0 \\ 0 \ldots 0 \; 1 \ldots 0 \ldots -1 \ldots 0 \end{matrix}\right] & \begin{matrix} \{a,b\} \\ \end{matrix} \end{array}$$



Graphs = Matrices

- <u>Goal:</u> satisfy (ϵ,δ)-privacy w.r.t edge changes, while approximating cut queries $\Phi_G(S) = |E(S, \bar{S})|$
- E_G Edge-matrix
- $L_G = E_G{}^{\mathsf{T}} E_G$ (PSD)

$$\begin{array}{c|cc} n & a & b \\ \hline a & \deg_a & -1 \\ b & -1 & \deg_b \end{array}$$



Edge Matrix and Cuts

$$\begin{array}{cc} & E_G \\ \{a,b\} & \begin{bmatrix} \\ 0 \ldots 0 \ \ldots \ \ldots \ldots \ldots \ldots \ 0 \\ 0 \ldots 0 \ 1 \ldots 0 \ldots -1 \ldots 0 \\ \\ \end{bmatrix} \end{array} \begin{array}{c} \mathbf{1}_S \\ \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array} = \begin{array}{c} E_G\mathbf{1}_S \\ \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array}$$

$$\Phi_G(S) = \|E_G 1_S\|^2 = 1_S^\top L_G 1_S = (1_S 1_S^\top) \cdot L_G$$

- Indeed, $1_S^\top L_G 1_T$ is the value of the (S,T)-cut – but caution:
- Approximating $(S,\bar{S})$ cuts = approximate vector lengths.
- Approximating (S,T) cuts = approximating dot-products (for large vectors!)

Straw-Man Algorithm #1

- Given t queries, S_1, S_2, …, S_t, answer each one with small additive noise.
  - Good: efficient; adaptive; non-restricted queries
  - Bad: $t < \epsilon n^4$
    - If $\mathrm{error}{=}O(n/\epsilon)$, then $t \approx (n^2/\epsilon)$

Straw-Man Algorithm #2

- Use exponential mechanism [MT07, BLR08]
- Scoring function: $\mathrm{sc}(M,G)=\max_S |\Phi_G(S)\text{-}\Phi_M(S)|$
  - Over adjacency matrices:
    - 2^{n^2} matrices $\Rightarrow \mathrm{error} \approx n^2/\epsilon$
  - Over Laplacians:
    - 2^{n^2} edge-matrices $\Rightarrow \mathrm{error} \approx n^2/\epsilon$
    - $2^{n\log(n)/\eta^2}$ sparsifiers $\Rightarrow \mathrm{error}(S) \approx n\log(n)/\epsilon + \eta\,\Phi(S)$
  - Good: low error, non-restricted queries.
  - Bad: Non efficient.

Here's where we restrict outselves to $(S,\bar{S})$ cuts

Straw-Man Algorithm #3

- Use Private Multiplicative Weights [HR10,GRU12] (or other iterative) mechanism
  - Over edges = coordinates of the adjacency matrices
    - Universe size of $O(n^2)$ (each update step take poly-time)
  - Start with M = uniform adjacency matrix
  - Per query in Q:
    - Either tell user "$\text{answer according to } M$"
    - Or tell user "$\text{update using } \textit{the query} \text{ \& private answer } a$"
  - Error of $O(|E|\,|V|/\epsilon)^{1/2}$ or $O(|E|\log(|Q|)/\epsilon)^{1/2}$
  - Good: low error (for sparse graphs)
  - Bad: Non efficient

  - Q: Possible to find update-queries efficiently and privately?

Crux: #"update" is bounded;
Non-updates hardly leak privacy

THE Open Question

- Efficient algorithm that answers all cut-queries with $\mathrm{error} = \mathrm{O}(n \log(n))$
  - Best known to date: $n^{3/2}$
    - For the general case (*any* graph, *any* cut-query)

- Or just for sparse cuts? (with $|S| \leq s$)
  - Best known to date: $s^{3/2}$

- Or just for sparse graphs? (with $|E| = O(|V|)$)
  - Best known to date: with Q of poly-size. (PMW-mechanism)

- PLEASE(!) break the $n^{3/2}$ barrier

Rest of this Talk

1. Getting $n^{3/2}$ via Randomized Response [GRU12]
2. Getting $s^{3/2}$ for cuts of size s [DTTZ14]
3. Better bounds for better graphs [BBDS12]
   – When all eigenvalues of the Laplacian are large
4. Approximating the PCA [DDTZ14]+[HR13, H14]
   – Spectral gap ($\sigma_{k+1} \gg \sigma_k$) and $\{v_1 \ldots v_k\}$ capture many cut-queries
   – Better guarantees for incoherent matrices

1. Randomized Response ~[GRU12]

Algorithm:

Fill adjacency matrix with 1,-1 iid,

$$\Pr[A_{u,v} = 1 \mid (u,v) \in E(G)] = \tfrac{1+\epsilon}{2}$$

$$\Pr[A_{u,v} = 1 \mid (u,v) \notin E(G)] = \tfrac{1}{2}$$

$$\begin{array}{c|ccc|} & a & b & b' \\ \hline a & & 1 & -1 \\ b & 1 & & \\ & -1 & & \\ \hline \end{array}$$

Privacy:

$$\frac{\Pr[A_{u,v} = 1 \mid (u,v) \in E(G)]}{\Pr[A_{u,v} = 1 \mid (u,v) \notin E(G)]} \le (1+\epsilon) \le e^{\epsilon}$$

For all queries:

- $s = n/2$
- $\nu = 1/2^s$

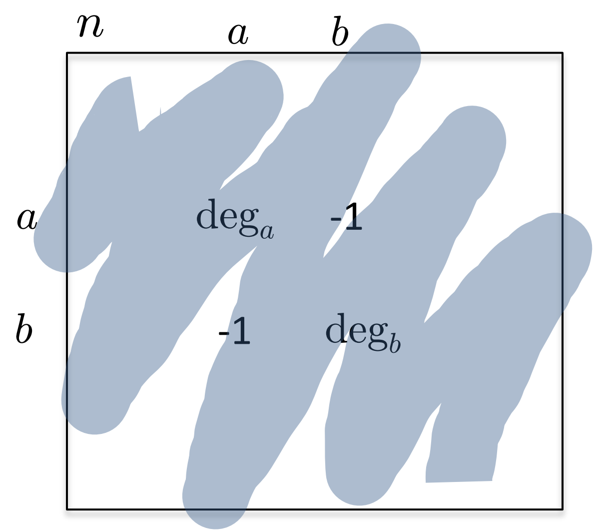
Hence the $n^{3/2}$ bound

Utility:

$$\Pr\left[\left|\sum_{u\in S, v\notin S} A_{u,v} - \epsilon \sum_{u\in S, v\notin S} w_{u,v}\right| \ge O(\sqrt{\log(1/\nu)s(n-s)})\right] \le \nu$$

2. Laplacian Perturbation [DTTZ14]

Algorithm:
Additive random (Gaussian) noise of $O(1/\epsilon)$ for each entry in the Laplacian



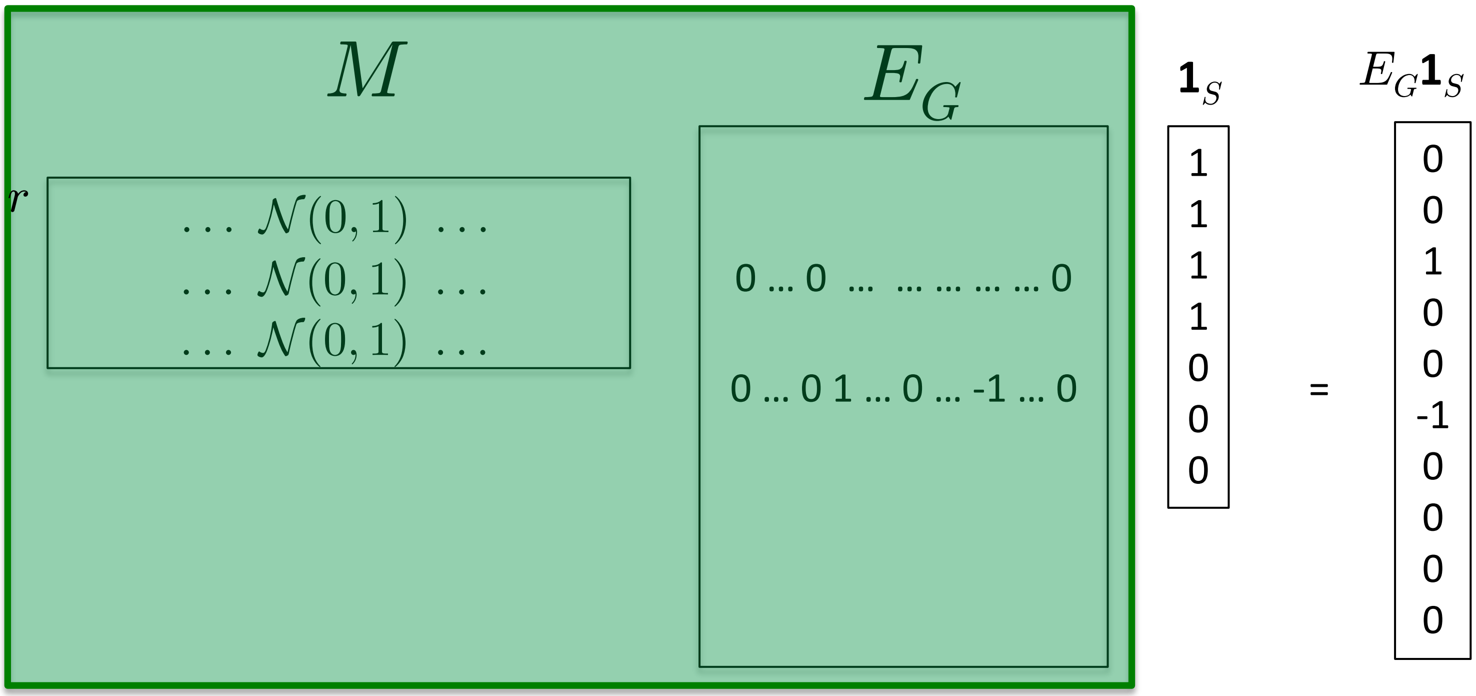
Privacy:
Each entry changes by ≤ 1

Utility: $(1_S 1_S^{\mathsf{T}}) \cdot (L_G + \text{noise}), \quad \text{noise} \sim \mathcal{N}(0, s^2/\epsilon^2)$

$\Rightarrow$ expected error of a single query $= O(s/\epsilon)$

$\Rightarrow$ w.h.p all n^s cut have error $O(s^{3/2}\log(n)/\epsilon)$

3. Johnson-Lindenstrauss [BBDS12]



$$\|(M \cdot E_G)\mathbf{1}_S\|^2 \approx \|E_G \mathbf{1}_S\|^2 = |E(S, \bar{S})|$$

JL Mechanism - Main Theorem

E_G, $E_{G'}$ – two neighboring edge-matrices guaranteed to $\boxed{\text{have singular values } \Omega(\log(1/\delta)/\epsilon)}$

R – a row in the JL matrix

(iid coordinates $\sim$ $(0{,}1)$-Gaussian)

$$\Pr_{x \sim RL_G}\left[\mathsf{PDF}_{RL_G}(x) \in e^{\pm\epsilon}\mathsf{PDF}_{RL_{G'}}(x)\right] > 1 - \delta$$

$$\#\text{rows} \approx \log(\#\text{Queries})/\eta^2$$

$\Rightarrow$ singular values $\geq$

$$\Omega((\sqrt{\#\text{rows}})\cdot\log(1/\delta)/\epsilon) = \Omega((\sqrt{\log(\#\text{Queries})})\cdot\log(1/\delta)/\epsilon\eta)$$

4. PCA of a Matrix

- Cut-queries – a private case of a matrix (L_G) operating on a vector ($\mathbf{1}_S$)
- PCA: Given L_G output M of rank k s.t. we minimize

$$\frac{x^{\mathsf{T}}(L_G - M)x}{x^{\mathsf{T}}x}$$

- Non-privately – the top-k eigenvectors of L_G
- We would like to be as close to these k vectors as possible
- All works assume:
  - General matrices
  - Neighboring matrices – a single entry differs by at most 1

- Doesn't necessarily imply we give good answers to all / many cuts…
  - Query vector should have large weight on subspace spanned by top-k eigenvectors
    - A user can find it out
  - And we should have a large gap $\sigma_k > \sigma_{k+1}$
    - We can release this information privately

PCA of a Matrix [DTTZ14]

- Algorithm:
  - $M = L_G + \mathrm{noise}(1/\epsilon)$ per entry
  - Output u_1, u_2, …, u_k – top-k eigenvectors of M
- Analysis:
  - Notation: v_1, v_2, … v_k – top-k eigenvectors of L_G
  - $u_1{}^\intercal M u_1 \geq v_1{}^\intercal M v_1 = v_1{}^\intercal(L_G + \mathrm{noise}(1/\epsilon))v_1 = \sigma_1 + \mathrm{E}[\mathrm{noise}]$
  - $u_1{}^\intercal M u_1 \leq u_1{}^\intercal L_G u_1 + \mathrm{E}[\text{max-noise}(1/\epsilon)]$
  - $\Rightarrow u_1{}^\intercal L_G u_1 \geq \sigma_1 - O(\sqrt{n}/\epsilon)$

  - For $k>1$

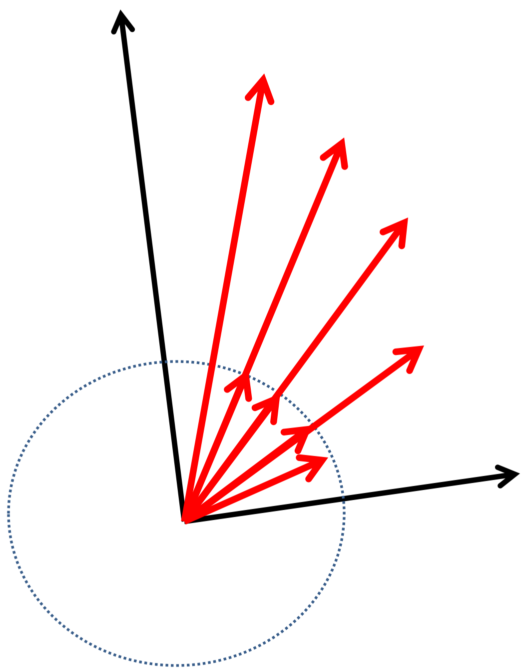
$$u_k{}^\intercal M u_k = \max_{S:\dim=k} \min_{x\in S} x^\intercal M x$$
$$\geq \min_{x\in\mathrm{span}\{v_1, \dots v_k\}} x^\intercal M x$$
$$= \min_{x\in\mathrm{span}\{v_1, \dots v_k\}} x^\intercal L_G x - \min_{x\in\mathrm{span}\{v_1, \dots v_k\}} x^\intercal[\mathrm{noise}]x$$

  - $\Rightarrow u_k{}^\intercal L_G u_k \geq \sigma_k - O(\sqrt{n}+\sqrt{k})/\epsilon)$

PCA of a Matrix [HR13, H14]

- Release leading eigenvector via power iterations:

$$x^{t+1} = \mathrm{normalize}(L_G x^t)$$



PCA of a Matrix [HR13, H14]

- Private power iterations

$$x^{t+1} = \mathrm{normalize}(L_G x^t + \mathcal{N}(0, 1/\epsilon^2)^n)$$

- In general – roughly same guarantees
  - $x^{*\top} L_G\, x^* \geq \sigma_1$ - $O(\sqrt{n}/\epsilon)$

- Denote SVD $L_G = U^\top \Sigma\, U$
- HR main observation: suffices to use noise $\propto \|U\|_\infty{}^2$
  - $\|x^t\|_\infty \leq \|U\|_\infty$ w.h.p
  - So adding Noise of $(\sqrt{\#\mathrm{Iterations}})\cdot\mathrm{coherence}/\epsilon$ per coordinate should maintain privacy in all power-iterations
  - Error $= O(\sqrt{n}\ \|U\|_\infty/\epsilon)$
  - Optimal noise: matching lower bound.

PCA of a Matrix [HR13, H14]

- To get a rank-k approximation:
  - Run power-iteration k times
    - At time i, approx first eigenvector of L_G-$\sum_{j=1}^{i\text{-}1} \sigma_j \, v_j v_j^{\mathsf{T}}$
  - Gives error of $\mathrm{O}(k^2(n \cdot \mathrm{coherence} + \sqrt{k})/\epsilon)$

  - Run power iteration for a $n \times k$ matrix
  - Gives error of $\mathrm{O}(\sqrt{(nk)} \|U\|_\infty \sigma_1/\sigma_k \, \epsilon)$

Summary

- We know:
  - Efficiently answer all cut-queries with $\mathrm{error} = n^{3/2}$
  - Inefficiently answer all cut-queries with $\mathrm{error} = n \log(n)$
  - Efficiently answer all s-sparse cut-queries with $\mathrm{error} = s^{3/2}$
  - Inefficiently answer all s-sparse cut-queries with $\mathrm{error} = (|E|s)^{1/2}$
  - Efficiently answer many-cut queries for nice graphs
  - Efficiently compute PCA of any matrix with error of $\mathrm{error} = \sqrt{n}$
  - Efficiently compute PCA of incoherent matrices with $\mathrm{error} = \sqrt{n}\|U\|_\infty$
- We don't know:
  - Efficiently answer all cut-queries with error $\mathrm{error} = n \log(n)$
  - Efficiently answer all s-sparse cut-queries with $\mathrm{error} = (ns)^{1/2}$
  - Other notions of "niceness"

Thank you!