Sparse Johnson-Lindenstrauss Transforms

Jelani Nelson
MIT

May 24, 2011

joint work with Daniel Kane (Harvard)

Metric Johnson-Lindenstrauss lemma

Metric JL (MJL) Lemma, 1984

Every set of n points in Euclidean space can be embedded into $O(\varepsilon^{-2}\log n)$-dimensional Euclidean space so that all pairwise distances are preserved up to a $1 \pm \varepsilon$ factor.

Metric Johnson-Lindenstrauss lemma

Metric JL (MJL) Lemma, 1984

Every set of n points in Euclidean space can be embedded into $O(\varepsilon^{-2} \log n)$-dimensional Euclidean space so that all pairwise distances are preserved up to a $1 \pm \varepsilon$ factor.

Uses:

- Speed up geometric algorithms by first reducing dimension of input [Indyk-Motwani, 1998], [Indyk, 2001]
- Low-memory streaming algorithms for linear algebra problems [Sarlós, 2006], [LWMRT, 2007], [Clarkson-Woodruff, 2009]
- Essentially equivalent to RIP matrices from compressive sensing [Baraniuk et al., 2008], [Krahmer-Ward, 2010] (used for sparse recovery of signals)

How to prove the JL lemma

Distributional JL (DJL) lemma

Lemma

For any $0 < \varepsilon, \delta < 1/2$ *there exists a distribution* $\mathcal{D}_{\varepsilon,\delta}$ *on* $\mathbb{R}^{k \times d}$ *for* $k = O(\varepsilon^{-2} \log(1/\delta))$ *so that for any* $x \in \mathcal{S}^{d-1}$,

$$\Pr_{S \sim \mathcal{D}_{\varepsilon,\delta}} \left[\left| \|Sx\|_2^2 - 1 \right| > \varepsilon \right] < \delta.$$

How to prove the JL lemma

Distributional JL (DJL) lemma

Lemma

For any $0 < \varepsilon, \delta < 1/2$ *there exists a distribution* $\mathcal{D}_{\varepsilon,\delta}$ *on* $\mathbb{R}^{k\times d}$ *for* $k = O(\varepsilon^{-2}\log(1/\delta))$ *so that for any* $x \in \mathcal{S}^{d-1}$,

$$\Pr_{S\sim\mathcal{D}_{\varepsilon,\delta}}\left[\left|\|Sx\|_2^2 - 1\right| > \varepsilon\right] < \delta.$$

Proof of MJL: Set $\delta = 1/n^2$ in DJL and x as the difference vector of some pair of points. Union bound over the $\binom{n}{2}$ pairs.

How to prove the JL lemma

Distributional JL (DJL) lemma

Lemma

For any $0 < \varepsilon, \delta < 1/2$ *there exists a distribution* $\mathcal{D}_{\varepsilon,\delta}$ *on* $\mathbb{R}^{k\times d}$ *for* $k = O(\varepsilon^{-2}\log(1/\delta))$ *so that for any* $x \in \mathcal{S}^{d-1}$,

$$\Pr_{S\sim\mathcal{D}_{\varepsilon,\delta}}\left[\left|\|Sx\|_2^2 - 1\right| > \varepsilon\right] < \delta.$$

Proof of MJL: Set $\delta = 1/n^2$ in DJL and x as the difference vector of some pair of points. Union bound over the $\binom{n}{2}$ pairs.

Theorem (Alon, 2003)

For every n*, there exists a set of* n *points requiring target dimension* $k = \Omega((\varepsilon^{-2}/\log(1/\varepsilon))\log n)$.

Theorem (Jayram-Woodruff, 2011; Kane-Meka-N., 2011)

For DJL, $k = \Theta(\varepsilon^{-2}\log(1/\delta))$ *is optimal.*

Proving the JL lemma

Older proofs

- [Johnson-Lindenstrauss, 1984], [Frankl-Maehara, 1988]: Random rotation, then projection onto first k coordinates.
- [Indyk-Motwani, 1998], [Dasgupta-Gupta, 2003]: Random matrix with independent Gaussian entries.
- [Achlioptas, 2001]: Independent Bernoulli entries.
- [Clarkson-Woodruff, 2009]: $O(\log(1/\delta))$-wise independent Bernoulli entries.
- [Arriaga-Vempala, 1999], [Matousek, 2008]: Independent entries having mean 0, variance $1/k$, and subGaussian tails (for a Gaussian with variance $1/k$).

Proving the JL lemma

Older proofs

- [Johnson-Lindenstrauss, 1984], [Frankl-Maehara, 1988]: Random rotation, then projection onto first k coordinates.
- [Indyk-Motwani, 1998], [Dasgupta-Gupta, 2003]: Random matrix with independent Gaussian entries.
- [Achlioptas, 2001]: Independent Bernoulli entries.
- [Clarkson-Woodruff, 2009]: $O(\log(1/\delta))$-wise independent Bernoulli entries.
- [Arriaga-Vempala, 1999], [Matousek, 2008]: Independent entries having mean 0, variance $1/k$, and subGaussian tails (for a Gaussian with variance $1/k$).

Downside: Performing embedding is dense matrix-vector multiplication, $O(k \cdot \|x\|_0)$ time

Fast JL Transforms

- [Ailon-Chazelle, 2006]: $x \mapsto PHDx$, $O(d \log d + k^3)$ time
  P is a random sparse matrix, H is Hadamard, D has random ± 1 on diagonal
- [Ailon-Liberty, 2008]: $O(d \log k + k^2)$ time, also based on fast Hadamard transform
- [Ailon-Liberty, 2011], [Krahmer-Ward]: $O(d \log d)$ for MJL, but with suboptimal $k = O(\varepsilon^{-2} \log n \log^4 d)$.

Fast JL Transforms

- [Ailon-Chazelle, 2006]: $x \mapsto PHDx$, $O(d \log d + k^3)$ time
  P is a random sparse matrix, H is Hadamard, D has random ± 1 on diagonal
- [Ailon-Liberty, 2008]: $O(d \log k + k^2)$ time, also based on fast Hadamard transform
- [Ailon-Liberty, 2011], [Krahmer-Ward]: $O(d \log d)$ for MJL, but with suboptimal $k = O(\varepsilon^{-2} \log n \log^4 d)$.

Downside: Slow to embed sparse vectors: running time is $\Omega(\min\{k \cdot \|x\|_0, d\})$ even if $\|x\|_0 = 1$

Where Do Sparse Vectors Show Up?

- **Documents as bags of words:** x_i = number of occurrences of word i. Compare documents using cosine similarity.

  d = lexicon size; most documents aren't dictionaries
- **Network traffic:** $x_{i,j} = \#$bytes sent from i to j

  $d = 2^{64}$ (2^{256} in IPv6); most servers don't talk to each other
- **User ratings:** x_i is user's score for movie i on Netflix

  $d = \#\mathrm{movies}$; most people haven't watched all movies
- **Streaming:** x receives updates $x \leftarrow x + v \cdot e_i$ in a stream. Maintaining Sx requires calculating Se_i.
- $\ldots$

Sparse JL transforms

One way to embed sparse vectors faster: use sparse matrices.

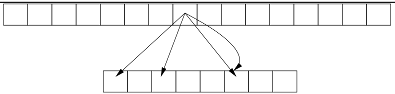
Sparse JL transforms

One way to embed sparse vectors faster: use sparse matrices.

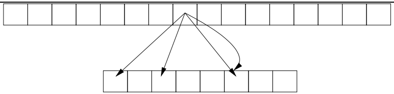
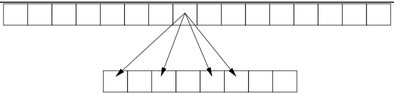
$s = \#$non-zero entries per column
(so embedding time is $s \cdot \|x\|_0$)

reference	value of s	type
[JL84], [FM88], [IM98], ...	$k \approx 4\varepsilon^{-2} \log(1/\delta)$	dense
[Achlioptas01]	$k/3$	sparse Bernoulli
[WDALS09]	no proof	hashing
[DKS10]	$\tilde{O}(\varepsilon^{-1} \log^3(1/\delta))$	hashing
[KN10a], [BOR10]	$\tilde{O}(\varepsilon^{-1} \log^2(1/\delta))$	"
[KN10b]	$O(\varepsilon^{-1} \log(1/\delta))$	hashing (random codes)

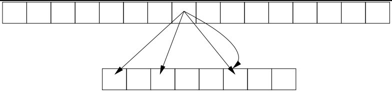
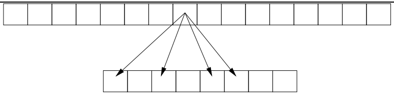
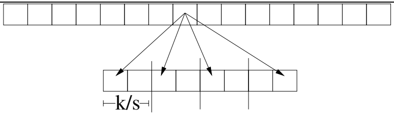
Sparse JL Constructions

[DKS, 2010]		$s = \tilde{\Theta}(\varepsilon^{-1} \log^2(1/\delta))$

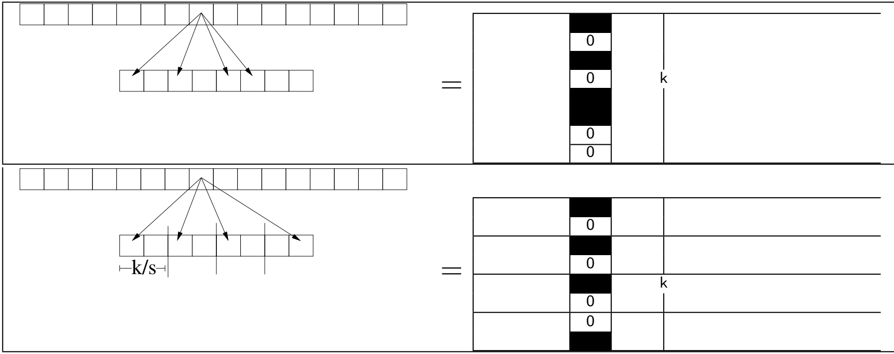
Sparse JL Constructions

[DKS, 2010]		$s = \tilde{\Theta}(\varepsilon^{-1} \log^2(1/\delta))$
[this work]		$s = \Theta(\varepsilon^{-1} \log(1/\delta))$

Sparse JL Constructions

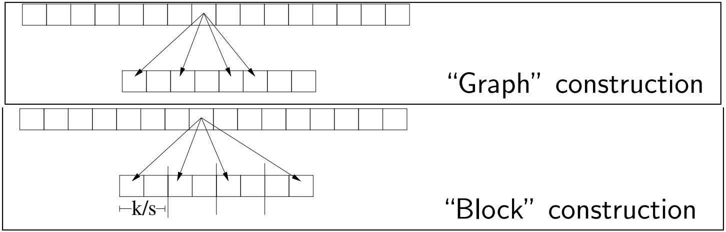
[DKS, 2010]		$s = \tilde{\Theta}(\varepsilon^{-1} \log^2(1/\delta))$
[this work]		$s = \Theta(\varepsilon^{-1} \log(1/\delta))$
[this work]	 	$s = \Theta(\varepsilon^{-1} \log(1/\delta))$

Sparse JL Constructions (in matrix form)



Each black cell is $\pm 1/\sqrt{s}$ at random

Sparse JL Constructions (nicknames)



Sparse JL intuition

- Let $h(j, r), \sigma(j, r)$ be random hash location and random sign for rth copy of x_j.

Sparse JL intuition

- Let $h(j, r), \sigma(j, r)$ be random hash location and random sign for rth copy of x_j.
- $(Sx)_i = (1/\sqrt{s}) \cdot \sum_{h(j,r)=i} x_j \cdot \sigma(j, r)$

Sparse JL intuition

- Let $h(j,r), \sigma(j,r)$ be random hash location and random sign for rth copy of x_j.
- $(Sx)_i = (1/\sqrt{s}) \cdot \sum_{h(j,r)=i} x_j \cdot \sigma(j,r)$

$$\|Sx\|_2^2 = \|x\|_2^2 + (1/s) \cdot \sum_{\substack{(j,r)' \\ \neq (j',r')}} x_j x_{j'} \sigma(j,r)\sigma(j',r') \cdot \mathbf{1}_{h(j,r)=h(j',r')}$$

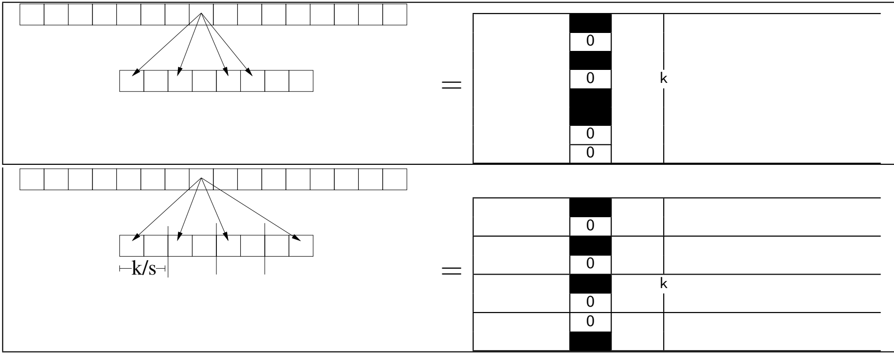
Sparse JL intuition

- Let $h(j,r), \sigma(j,r)$ be random hash location and random sign for rth copy of x_j.
- $(Sx)_i = (1/\sqrt{s}) \cdot \sum_{h(j,r)=i} x_j \cdot \sigma(j,r)$

$$\|Sx\|_2^2 = \|x\|_2^2 + (1/s) \cdot \sum_{\substack{(j,r)' \\ \neq (j',r')}} x_j x_{j'} \sigma(j,r) \sigma(j',r') \cdot \mathbf{1}_{h(j,r)=h(j',r')}$$

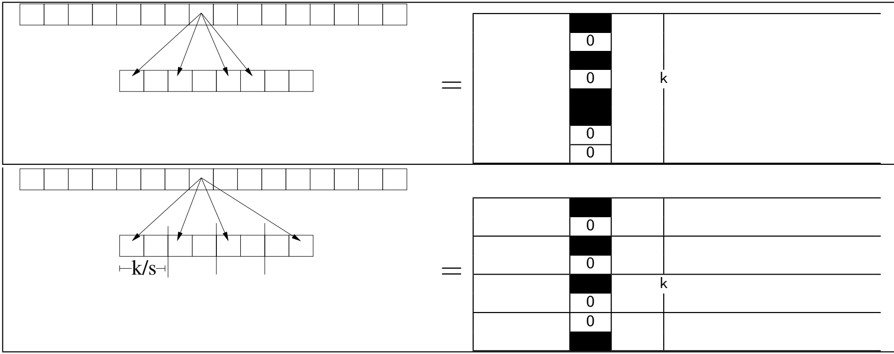
- $x = (1/\sqrt{2}, 1/\sqrt{2}, 0, \ldots, 0)$ with $t < (1/2)\log(1/\delta)$ collisions. All signs agree with probability $2^{-t} > \sqrt{\delta} \gg \delta$, giving error t/s. So, need $s = \Omega(t/\varepsilon)$. (Collisions are bad)

Sparse JL via Codes



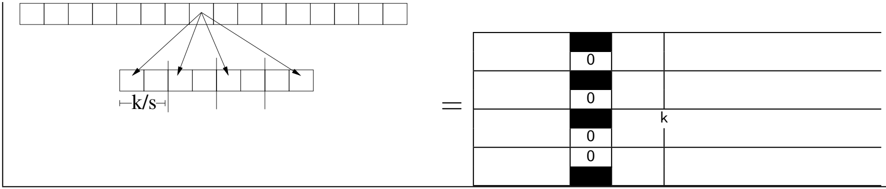
- Graph construction: Constant weight binary code of weight s.
- Block construction: Code over q-ary alphabet, $q = k/s$.

Sparse JL via Codes



- Graph construction: Constant weight binary code of weight s.
- Block construction: Code over q-ary alphabet, $q = k/s$.
- Will show: Suffices to have minimum distance $s - O(s^2/k)$.

Analysis (block construction)

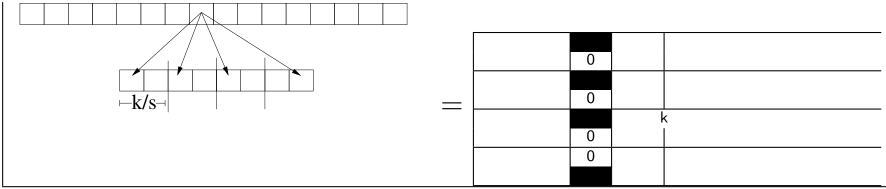


- $\eta_{i,j,r}$ indicates whether i, j collide in ith chunk.
- $\|Sx\|_2^2 = \|x\|_2^2 + Z$

  $Z = (1/s)\sum_r Z_r$

  $Z_r = \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$

Analysis (block construction)

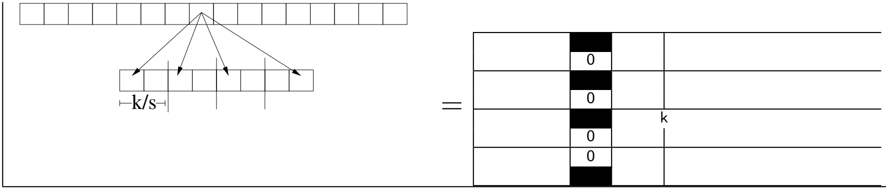


- $\eta_{i,j,r}$ indicates whether i, j collide in ith chunk.
- $\|Sx\|_2^2 = \|x\|_2^2 + Z$

  $Z = (1/s)\sum_r Z_r$

  $Z_r = \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$
- Plan: $\Pr[|Z| > \varepsilon] < \varepsilon^{\ell} \cdot \mathbf{E}[Z^{\ell}]$

Analysis (block construction)

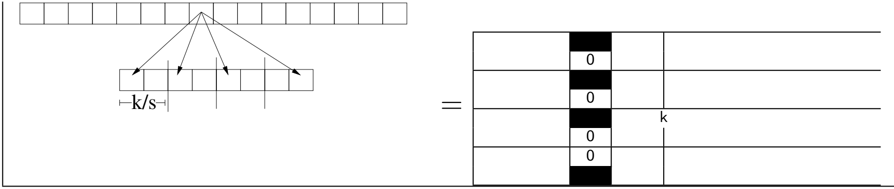


- $\eta_{i,j,r}$ indicates whether i, j collide in ith chunk.
- $\|Sx\|_2^2 = \|x\|_2^2 + Z$

  $Z = (1/s)\sum_r Z_r$

  $Z_r = \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$
- Plan: $\Pr[|Z| > \varepsilon] < \varepsilon^{\ell} \cdot \mathbf{E}[Z^{\ell}]$
- Z is a quadratic form in σ, so apply known moment bounds for quadratic forms

Analysis



Theorem (Hanson-Wright, 1971)

$z_1, \ldots, z_n$ independent Bernoulli, $B \in \mathbb{R}^{n\times n}$ symmetric. For $\ell \geq 2$,

$$\mathbf{E}\left[\left|z^T B z - \mathrm{trace}(B)\right|^\ell\right] < C^\ell \cdot \max\left\{\sqrt{\ell}\|B\|_F, \ell\|B\|_2\right\}^\ell$$

Reminder:

- $\|B\|_F = \sqrt{\sum_{i,j} B_{i,j}^2}$
- $\|B\|_2$ is largest magnitude of eigenvalue of B

Analysis

$$Z = \frac{1}{s} \cdot \sum_{r=1}^{s} \sum_{i \neq j} x_i x_j \sigma(i,r) \sigma(j,r) \eta_{i,j,r}$$

Analysis

$$Z = \frac{1}{s} \cdot \sum_{r=1}^{s} \sum_{i \neq j} x_i x_j \sigma(i,r) \sigma(j,r) \eta_{i,j,r}$$

$$= \sigma^T T \sigma$$

$$T = \frac{1}{s} \cdot \begin{array}{|c|c|c|c|} \hline T_1 & 0 & \dots & 0 \\ \hline 0 & T_2 & \dots & 0 \\ \hline 0 & 0 & \ddots & 0 \\ \hline 0 & \dots & 0 & T_s \\ \hline \end{array}$$

- $(T_r)_{i,j} = x_i x_j \eta_{i,j,r}$

Analysis (cont'd)

$$T = \frac{1}{s} \cdot \begin{array}{|c|c|c|c|}
\hline
T_1 & 0 & \dots & 0 \\
\hline
0 & T_2 & \dots & 0 \\
\hline
0 & 0 & \ddots & 0 \\
\hline
0 & \dots & 0 & T_s \\
\hline
\end{array}$$

- $(T_r)_{i,j} = x_i x_j \eta_{i,j,r}$

Analysis (cont'd)

$$T = \frac{1}{s} \cdot \begin{array}{|c|c|c|c|}
\hline
T_1 & 0 & \dots & 0 \\
\hline
0 & T_2 & \dots & 0 \\
\hline
0 & 0 & \ddots & 0 \\
\hline
0 & \dots & 0 & T_s \\
\hline
\end{array}$$

- $(T_r)_{i,j} = x_i x_j \eta_{i,j,r}$
- $\|T\|_F^2 = \frac{1}{s^2} \sum_{i \neq j} x_i^2 x_j^2 \cdot (\#\text{times } i, j \text{ collide})$

Analysis (cont'd)

$$T = \frac{1}{s} \cdot \begin{array}{|cccc|} \hline \boxed{T_1} & 0 & \ldots & 0 \\ 0 & \boxed{T_2} & \ldots & 0 \\ 0 & 0 & \boxed{\ddots} & 0 \\ 0 & \ldots & 0 & \boxed{T_s} \\ \hline \end{array}$$

- $(T_r)_{i,j} = x_i x_j \eta_{i,j,r}$
- $\|T\|_F^2 = \frac{1}{s^2} \sum_{i \neq j} x_i^2 x_j^2 \cdot (\#\text{times } i, j \text{ collide})$

$$< O(1/k) \cdot \|x\|_2^4 = O(1/k) \text{ (good code!)}$$

Analysis (cont'd)

$$T = \frac{1}{s} \cdot \begin{array}{|c|c|c|c|}
\hline
T_1 & 0 & \ldots & 0 \\
\hline
0 & T_2 & \ldots & 0 \\
\hline
0 & 0 & \ddots & 0 \\
\hline
0 & \ldots & 0 & T_s \\
\hline
\end{array}$$

- $(T_r)_{i,j} = x_i x_j \eta_{i,j,r}$
- $\|T\|_F^2 = \frac{1}{s^2} \sum_{i \neq j} x_i^2 x_j^2 \cdot (\#\text{times } i, j \text{ collide})$

  $< O(1/k) \cdot \|x\|_2^4 = O(1/k)$ (good code!)
- $\|T\|_2 = \max_r \|T_r\|_2$, can bound by $1/s$

Analysis (cont'd)

$$T = \frac{1}{s} \cdot \begin{array}{|c|c|c|c|}\hline T_1 & 0 & \dots & 0 \\ \hline 0 & T_2 & \dots & 0 \\ \hline 0 & 0 & \ddots & 0 \\ \hline 0 & \dots & 0 & T_s \\ \hline \end{array}$$

- $(T_r)_{i,j} = x_i x_j \eta_{i,j,r}$
- $\|T\|_F^2 = \frac{1}{s^2} \sum_{i \neq j} x_i^2 x_j^2 \cdot (\#\text{times } i,j \text{ collide})$
  $< O(1/k) \cdot \|x\|_2^4 = O(1/k)$ (good code!)
- $\|T\|_2 = \max_r \|T_r\|_2$, can bound by $1/s$

$$\Pr[|Z| > \varepsilon] < C^\ell \cdot \max\left\{\frac{1}{\varepsilon} \cdot \sqrt{\frac{\ell}{k}}, \frac{1}{\varepsilon} \cdot \frac{\ell}{s}\right\}^\ell$$

$\ell = \log(1/\delta)$, $k = \Omega(\ell/\varepsilon^2)$, $s = \Omega(\ell/\varepsilon)$, **QED**

Code-based Construction: **Caveat**

Need a sufficiently good code.

Code-based Construction: **Caveat**

Need a sufficiently good code.

- Each pair of codewords should agree on $O(s^2/k)$ coordinates.
- Can get this with random code by Chernoff + union bound over pairs, but then need: $s^2/k \geq \log(d/\delta) \Rightarrow$ $s \geq \sqrt{k \log(d/\delta)} = \Omega(\varepsilon^{-1}\sqrt{\log(d/\delta)\log(1/\delta)})$.

Code-based Construction: **Caveat**

Need a sufficiently good code.

- Each pair of codewords should agree on $O(s^2/k)$ coordinates.
- Can get this with random code by Chernoff + union bound over pairs, but then need: $s^2/k \geq \log(d/\delta) \Rightarrow$ $s \geq \sqrt{k \log(d/\delta)} = \Omega(\varepsilon^{-1}\sqrt{\log(d/\delta)\log(1/\delta)})$.
- Can assume $d = O(\varepsilon^{-2}/\delta)$ by first embedding into this dimension with $s = 1$ and 4-wise independent σ, h (Analysis: Chebyshev's inequality)

  $\Rightarrow$ Can get away with $s = O(\varepsilon^{-1}\sqrt{\log(1/(\varepsilon\delta))\log(1/\delta)})$.

Code-based Construction: **Caveat**

Need a sufficiently good code.

- Each pair of codewords should agree on $O(s^2/k)$ coordinates.
- Can get this with random code by Chernoff + union bound over pairs, but then need: $s^2/k \geq \log(d/\delta) \Rightarrow$ $s \geq \sqrt{k \log(d/\delta)} = \Omega(\varepsilon^{-1}\sqrt{\log(d/\delta)\log(1/\delta)})$.
- Can assume $d = O(\varepsilon^{-2}/\delta)$ by first embedding into this dimension with $s = 1$ and 4-wise independent σ, h (Analysis: Chebyshev's inequality)

  $\Rightarrow$ Can get away with $s = O(\varepsilon^{-1}\sqrt{\log(1/(\varepsilon\delta))\log(1/\delta)})$.

  Can we avoid the loss incurred by this union bound?

Improving the Construction

- Idea: Random hashing gives a good code, but it gives much more! (it's <u>random</u>).

Improving the Construction

- Idea: Random hashing gives a good code, but it gives much more! (it's random).
- Pick h at random
- Analysis: Directly bound the $\ell = \log(1/\delta)$th moment of the error term Z, then apply Markov to Z^ℓ

Improving the Construction

- Idea: Random hashing gives a good code, but it gives much more! (it's <u>random</u>).
- Pick h at random
- Analysis: Directly bound the $\ell = \log(1/\delta)$th moment of the error term Z, then apply Markov to Z^ℓ
- $Z = (1/s) \cdot \sum_{r=1}^{s} \sum_{i \neq j} x_i x_j \sigma(i,r) \sigma(j,r) \eta_{i,j,r}$

Improving the Construction

- Idea: Random hashing gives a good code, but it gives much more! (it's random).
- Pick h at random
- Analysis: Directly bound the $\ell = \log(1/\delta)$th moment of the error term Z, then apply Markov to Z^ℓ
- $Z = (1/s) \cdot \sum_{r=1}^{s} \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$
  $(Z = (1/s) \sum_{r=1}^{s} Z_r)$

Improving the Construction

- Idea: Random hashing gives a good code, but it gives much more! (it's <u>random</u>).
- Pick h at random
- Analysis: Directly bound the $\ell = \log(1/\delta)$th moment of the error term Z, then apply Markov to Z^ℓ
- $Z = (1/s) \cdot \sum_{r=1}^{s} \sum_{i \neq j} x_i x_j \sigma(i,r) \sigma(j,r) \eta_{i,j,r}$
  $(Z = (1/s) \sum_{r=1}^{s} Z_r)$

$$\mathbf{E}_{h,\sigma}[Z^\ell] = \frac{1}{s^\ell} \cdot \sum_{\substack{r_1 < \ldots < r_n \\ t_1, \ldots, t_n > 1 \\ \sum_i t_i = \ell}} \binom{\ell}{t_1, \ldots, t_n} \cdot \prod_{i=1}^{n} \mathbf{E}_{h,\sigma}[Z_{r_i}^{t_i}]$$

Bound the tth moment of any Z_r, then get the ℓth moment bound for Z by plugging into the above

Bounding $\mathbf{E}[Z_r^t]$

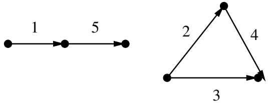
- $Z_r = \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$

Bounding $\mathbf{E}[Z_r^t]$

- $Z_r = \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$
- Monomials appearing in expansion of Z_r^t are in correspondence with directed multigraphs.

$$(x_1 x_2) \cdot (x_3 x_4) \cdot (x_3 x_8) \cdot (x_4 x_8) \cdot (x_2 x_{10})$$

$$\mapsto$$

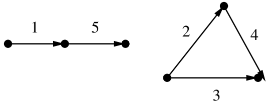


Bounding $\mathbf{E}[Z_r^t]$

- $Z_r = \sum_{i \neq j} x_i x_j \sigma(i,r)\sigma(j,r)\eta_{i,j,r}$
- Monomials appearing in expansion of Z_r^t are in correspondence with directed multigraphs.

$$(x_1 x_2) \cdot (x_3 x_4) \cdot (x_3 x_8) \cdot (x_4 x_8) \cdot (x_2 x_{10})$$

$$\mapsto$$



- Monomial contributes to expectation iff all degrees even
- Analysis: Group monomials appearing in Z_r^t according to isomorphism class then do some combinatorics.

Bounding $\mathbf{E}[Z_r^t]$

$m = \#\text{connected components}$, $v = \#\text{vertices}$, $d_u =$ degree of u

$$\mathbf{E}_{h,\sigma}[Z_r^t] \quad = \quad \sum_{G \in \mathcal{G}_t} \sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \mathbf{E}\left[\prod_{u=1}^{t} \eta_{i_u, j_u, r}\right] \cdot \left(\prod_{u=1}^{t} x_{i_u} x_{j_u}\right)$$

Bounding $\mathbf{E}[Z_r^t]$

$m = $ #connected components, $v = $ #vertices, $d_u = $ degree of u

$$\begin{aligned}
\mathbf{E}_{h,\sigma}[Z_r^t] &= \sum_{G \in \mathcal{G}_t} \sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \mathbf{E}\left[\prod_{u=1}^t \eta_{i_u, j_u, r}\right] \cdot \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right) \\
&= \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot \left(\sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right)\right)
\end{aligned}$$

Bounding $\mathbf{E}[Z_r^t]$

$m = \#\text{connected components}$, $v = \#\text{vertices}$, d_u = degree of u

$$\begin{aligned}
\mathbf{E}_{h,\sigma}[Z_r^t] &= \sum_{G \in \mathcal{G}_t} \sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \mathbf{E}\left[\prod_{u=1}^t \eta_{i_u, j_u, r}\right] \cdot \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right) \\
&= \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot \left(\sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right)\right) \\
&\leq \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot v! \cdot \frac{1}{\binom{t}{d_1/2, \ldots, d_v/2}}
\end{aligned}$$

Bounding $\mathbf{E}[Z_r^t]$

$m = \#\text{connected components}$, $v = \#\text{vertices}$, $d_u =$ degree of u

$$
\begin{aligned}
\mathbf{E}_{h,\sigma}[Z_r^t] &= \sum_{G \in \mathcal{G}_t} \sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \mathbf{E}\left[\prod_{u=1}^t \eta_{i_u, j_u, r}\right] \cdot \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right) \\
&= \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot \left(\sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right)\right) \\
&\leq \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot v! \cdot \frac{1}{\binom{t}{d_1/2, \ldots, d_v/2}} \\
&\leq 2^{O(t)} \sum_{v,m} t^{-t} v^v \left(\frac{s}{k}\right)^{v-m} \cdot \left(\sum_G \prod_u \sqrt{d_u}^{d_u}\right)
\end{aligned}
$$

Bounding $\mathbf{E}[Z_r^t]$

$m = \#\text{connected components}$, $v = \#\text{vertices}$, $d_u = \text{degree of } u$

$$
\begin{aligned}
\mathbf{E}_{h,\sigma}[Z_r^t] &= \sum_{G \in \mathcal{G}_t} \sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \mathbf{E}\left[\prod_{u=1}^t \eta_{i_u, j_u, r}\right] \cdot \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right) \\
&= \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot \left(\sum_{\substack{i_1 \neq j_1, \ldots, i_t \neq j_t \\ f((i_u, j_u)_{u=1}^t) = G}} \left(\prod_{u=1}^t x_{i_u} x_{j_u}\right)\right) \\
&\leq \sum_{G \in \mathcal{G}_t} \left(\frac{s}{k}\right)^{v-m} \cdot v! \cdot \frac{1}{\binom{t}{d_1/2, \ldots, d_v/2}} \\
&\leq 2^{O(t)} \sum_{v,m} t^{-t} v^v \left(\frac{s}{k}\right)^{v-m} \cdot \left(\sum_G \prod_u \sqrt{d_u}^{d_u}\right)
\end{aligned}
$$

Bounding $\mathbf{E}[Z_r^t]$

- Can bound the sum by forming G one edge at a time, in increasing order of label

  For example, if we didn't worry about connected components:

$$S_{i+1}/S_i \leq \sum_{u \neq v} \sqrt{d_u d_v} \leq \left(\sum_u \sqrt{d_u} \right)^2 \overset{\text{C-S}}{\leq} 2tv$$

Bounding $\mathbf{E}[Z_r^t]$

- Can bound the sum by forming G one edge at a time, in increasing order of label

  For example, if we didn't worry about connected components:

$$S_{i+1}/S_i \leq \sum_{u \neq v} \sqrt{d_u d_v} \leq \left(\sum_u \sqrt{d_u} \right)^2 \overset{\mathrm{C-S}}{\leq} 2tv$$

- In the end, can show

$$\mathbf{E}[Z_r^t] \leq 2^{O(t)} \cdot \begin{cases} s/k & t < \log(k/s) \\ (t/\log(k/s))^t & \text{otherwise} \end{cases}$$

- Plug this into formula for $\mathbf{E}[Z^\ell]$, **QED**

Tightness of Analysis

Analysis of required s is tight:

- $s \leq 1/(2\varepsilon)$: Look at a vector with $t = \lfloor 1/(s\varepsilon) \rfloor$ non-zero coordinates each of value $1/\sqrt{t}$, and show probability of exactly one collision is $\gg \delta$, and $> \varepsilon$ error when this happens and signs agree.

Tightness of Analysis

Analysis of required s is tight:

- $s \leq 1/(2\varepsilon)$: Look at a vector with $t = \lfloor 1/(s\varepsilon) \rfloor$ non-zero coordinates each of value $1/\sqrt{t}$, and show probability of exactly one collision is $\gg \delta$, and $> \varepsilon$ error when this happens and signs agree.
- $1/(2\varepsilon) < s < c\varepsilon^{-1}\log(1/\delta)$: Look at vector $(1/\sqrt{2}, 1/\sqrt{2}, 0, \ldots, 0)$ and show that probability of exactly $\lceil 2s\varepsilon \rceil$ collisions is $\gg \sqrt{\delta}$, all signs agree with probability $\gg \sqrt{\delta}$, and $> \varepsilon$ error when this happens.

Open Problems

Open Problems

- **OPEN:** Devise distribution which can be sampled using few random bits

  Current record:
  $O(\log d + \log(1/\varepsilon)\log(1/\delta) + \log(1/\delta)\log\log(1/\delta))$
  [Kane-Meka-N.]

  Existential: $O(\log d + \log(1/\delta))$
- **OPEN:** Can we embed a k-sparse vector into $\mathbb{R}^k$ in $k \cdot \mathrm{polylog}(d)$ time with the optimal k? This would give a fast amortized streaming algorithm without blowing up space (batch k updates at a time, since we're already spending k space storing the embedding). **Note:** Embedding should be correct for *any* vector, but time should depend on sparsity.
- **OPEN:** Embed any vector in $\tilde{O}(d)$ time into optimal k