Technion Coding Theory Seminar

June 2017

Gradient Coding from Cyclic MDS Codes and Expander Graphs

Netanel Raviv

Joint work with Itzhak Tamo, Rashish Tandon, and Alexandros G. Dimakis

Background – Machine Learning

- Goal – Learn an unknown function $f: \mathcal{X} \to \mathcal{Y}$.
  - $\mathcal{X}$ – "**feature space**" (usually $\mathcal{X} = \mathbb{R}^r$, ex. ultrasound scan, pixels in image).
  - $\mathcal{Y}$ – "**label space**" (usually $\mathcal{Y} = \mathbb{R}$, ex. weight of baby , man/woman).

$\mathcal{D}$ also on $\mathcal{Y}$ to allow errors/insufficient data

- Premise –
  - A set $S = \{z_i = (x_i, y_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathcal{Y}$ of examples ("**training set**"), drawn by an **unknown** distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$.
  - A set of functions $\mathcal{H}$ ("**hypothesis class**") from which the output is chosen.
  - Input – S. Output – $h \in \mathcal{H}$.

"**Bias-Complexity tradeoff**":
Larger $\mathcal{H}$ ⇒ higher complexity
Smaller $\mathcal{H}$ ⇒ higher error

- Definitions –
  - $\ell: \mathcal{H} \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}_{\geq 0}$ ("**loss function**") – Penalty for a given $h \in \mathcal{H}$ for erring on $(x, y) \in \mathcal{X} \times \mathcal{Y}$.
  - E.g., $\ell\big(h, (x, y)\big) = (h(x) - y)^2$ or $\ell\big(h, (x, y)\big) = 0$ if $h(x) = y$ and 1 otherwise.
  - $L_S(h) \triangleq \frac{1}{m} \sum_{z \in S} \ell(h, z)$ ("**empirical risk**") – The avg. error of a given hypothesis h on the training set.
  - $L_{\mathcal{D}}(h) \triangleq \mathbb{E}_{z \sim \mathcal{D}}\big(\ell(h, z)\big)$ ("**true risk**") – How "good" is a given hypothesis.

Solution:
Find h that minimizes $L_S(h)$, as long as we may guarantee that $|L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon$.

- Ultimate goal – Find $h \in \mathcal{H}$ that minimizes $L_{\mathcal{D}}(h)$.
- Problem – $\mathcal{D}$ unknown.

Background – Linear predictors

- This talk –
  - A hypothesis class $\mathcal{H}$ of **linear predictors** – A composition of a linear function with some $\phi: \mathbb{R} \to \mathcal{Y}$.
  - E.g., linear functions $\{h(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} \mid \boldsymbol{w} \in \mathbb{R}^r\}$ ($\phi = id$) or halfspaces $\{h(x) = sign(\boldsymbol{w} \cdot \boldsymbol{x}) \mid \boldsymbol{w} \in \mathbb{R}\}$ ($\phi = sign$).
  - Linear predictors are parametrized by $\boldsymbol{w} \in \mathbb{R}^r$,
    and hence $\ell\big(h, (x, y)\big), L_S(h), L_{\mathcal{D}}(h) \Rightarrow \ell\big(\boldsymbol{w}, (x, y)\big), L_S(\boldsymbol{w}), L_{\mathcal{D}}(\boldsymbol{w})$.
  - Find minimum using analytic techniques on $\boldsymbol{w} \in \mathbb{R}^r$.
- Stochastic Gradient Descent (**SGD**) –
  - The **gradient** ∇g of a (differentiable) r-variate function g is the vector of r-partial derivatives.
  - At any point $\boldsymbol{x_0} \in \mathbb{R}^r$, the vector $\nabla g(\boldsymbol{x_0})$ points at the direction of the largest increase of g.
  - **Gradient Descent** (alg. for finding the minimum of g) – Start with a guess $\boldsymbol{x_0}$. Compute gradient at this point, move away from it, and repeat.
  - Consider $\ell\big(\boldsymbol{w}, (x, y)\big), L_S(\boldsymbol{w}), L_{\mathcal{D}}(\boldsymbol{w})$ as differentiable functions in r variables $\boldsymbol{w} = (w_1, \ldots, w_r)$.
  - Apply the Gradient Descent algorithm, with a fresh training set in each iteration.

Background – Stochastic Gradient Descent

Algorithm: Stochastic Gradient Descent (SGD) for minimizing $L_{\mathcal{D}}(\boldsymbol{w})$

Input: Num. of iterations T.
Output: An estimate $\overline{\boldsymbol{w}} \in \mathbb{R}$ that minimizes $L_{\mathcal{D}}(\boldsymbol{w})$.
Initialize $\boldsymbol{w}^{(1)} = (0, \ldots, 0)$.
For $t = 1,2, \ldots, T$ **do**
Sample $S = \{z_i\}_{i=1}^{m} \sim \mathcal{D}^m$.
Compute $\{\nabla \ell(\boldsymbol{w}, z)\}_{z \in S}$
Define $\boldsymbol{v}_t = \nabla L_S\left(\boldsymbol{w}^{(t)}\right) = \frac{1}{m} \cdot \sum_{i \in [m]} \nabla \ell\left(\boldsymbol{w}^{(t)}, z_i\right)$.
Update $\boldsymbol{w}^{(t+1)} \triangleq \boldsymbol{w}^{(t)} - \boldsymbol{v}_t$
End
Return $\overline{\boldsymbol{w}} = \boldsymbol{w}^{(T)}$.

Reminder:
- Loss function - $\ell\left(\boldsymbol{w}, (x, y)\right)$.
- Empirical risk - $L_S(\boldsymbol{w}) \triangleq \frac{1}{m} \sum_{z \in S} \ell(\boldsymbol{w}, z)$.
- True risk - $L_{\mathcal{D}}(\boldsymbol{w}) \triangleq \mathbb{E}_{z \sim \mathcal{D}}\left(\ell(\boldsymbol{w}, z)\right)$.

Theorem – $\mathbb{E}_{S \sim \mathcal{D}^m}\left(\boldsymbol{v}_t | \boldsymbol{w}^{(t)}\right) = \nabla L_{\mathcal{D}}\left(\boldsymbol{w}^{(t)}\right)$

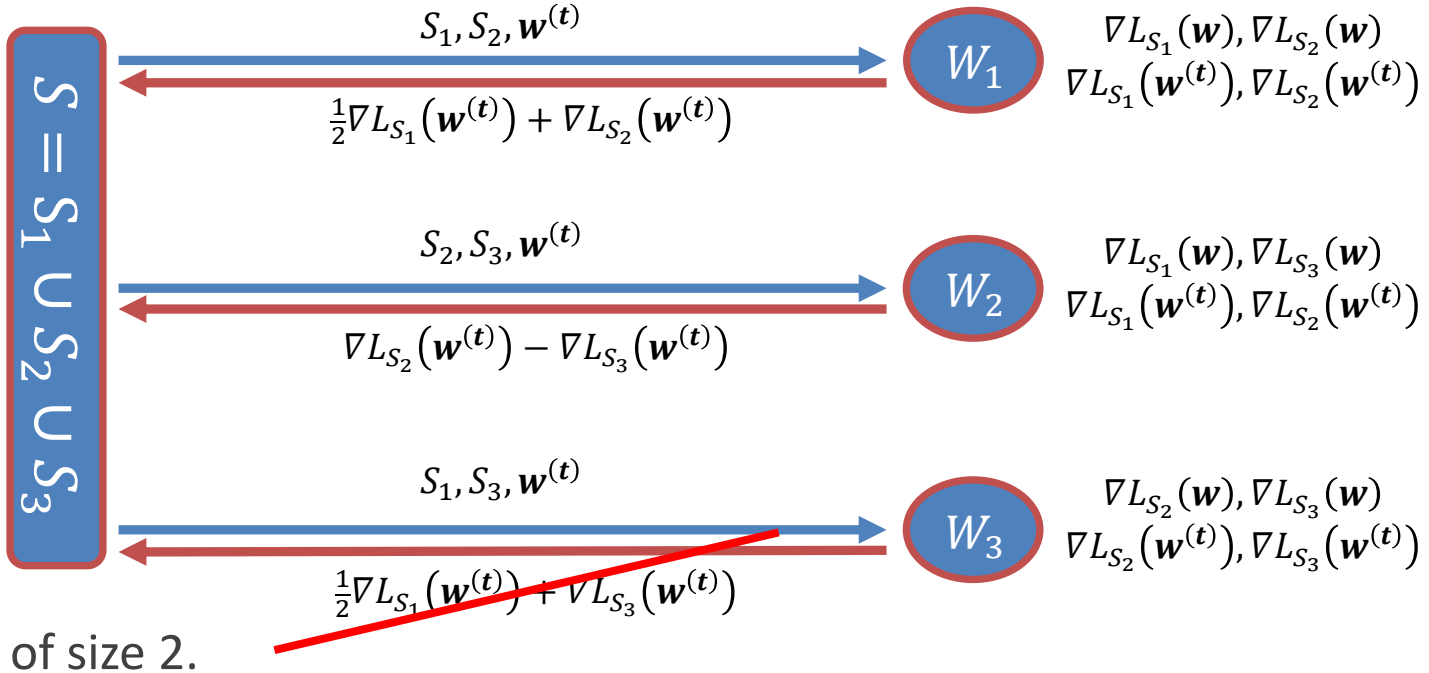
⇒ Although SGD operates on the **empirical risk**, it actually minimizes the **true risk**.

Distributed Coded SGD

- Bottleneck in SGD – Computation of $\{\nabla\ell(\boldsymbol{w},z)\}_{z\in S}$ (used for $\nabla L_S\left(\boldsymbol{w}^{(t)}\right)=\frac{1}{m}\cdot\sum_{i\in[m]}\nabla\ell\left(\boldsymbol{w}^{(t)},z_i\right)$).
- Observation – Computation is different for each $z\in S$.
  - I.e., if $S=\cup_{i\in[n]}S_i$ then $\nabla L_S(\boldsymbol{w})=\frac{1}{m}\cdot\sum_{i\in[n]}\nabla L_{S_i}(\boldsymbol{w})$.
- Solution for bottleneck – Distribute the training set S to n **worker nodes**. Problem – **Straggler** nodes.
- E.g., for $n=3$,
- From non-stragglers $K=\{1,2\}$, the master receives –

$$\begin{pmatrix}\frac{1}{2} & 1 & \\ & 1 & -1\end{pmatrix}\cdot\begin{pmatrix}\nabla L_{S_1}\left(\boldsymbol{w}^{(t)}\right)\\ \nabla L_{S_2}\left(\boldsymbol{w}^{(t)}\right)\\ \nabla L_{S_3}\left(\boldsymbol{w}^{(t)}\right)\end{pmatrix}\triangleq B_K\cdot M$$

- Want $\nabla L_S(\boldsymbol{w})=\frac{1}{m}\cdot\bar{1}\cdot M$.
- Need B with $\bar{1}\in rowspan(B_K)$ for **any** K of size 2.



Distributed Coded SGD – Definitions

- A scheme (A, B) for (distributed coded) SGD with n nodes and **storage overhead** d –
  - $B \in \mathbb{F}^{n \times n}$ and $A: \mathcal{P}([n]) \to \mathbb{F}^n$ (where $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$).
  - $|supp(B_i)| \leq d$ for every row B_i of B.
  - $supp(A(K)\ ) \subseteq K$ for all $K \in \mathcal{P}([n])$.
  - $A(K) \cdot B = \overline{1}$ for all "large enough" K.
- Two variants –
  - Exact Computation (**EC**), fixed number of stragglers s –
    - $A(K) \cdot B = \overline{1}$ For any $|K| \geq n - s$.
  - ϵ-Approximate Computation (**ϵ-AC**) varying number of stragglers –
    - $\epsilon: [n] \to \mathbb{R}$, a decreasing function.
    - $d_2\left(A(K) \cdot B, \overline{1}\right) \leq \epsilon(|K^c|)$.
- <u>Goal</u> – Construct such A and B.

Previous work

- Tandon et al. –
  - **Theorem** – For **EC**, $d \geq s + 1$.
  - For EC, a construction for $s + 1 | n$ and $d = s + 1$ using fractional repetition.
  - For EC, a **randomized** construction for $d = s + 1$ and any n.
- This work –
  - For **EC**, a **deterministic** construction for $d = s + 1$ and …
    - … any n and s over the complex numbers, using **Reed-Solomon Codes**.
    - … $n \neq s_{mod\ 2}$ over the real numbers, using **BCH Codes**.
  - For **ϵ-AC**, a construction with $d = O(1)$ using **Expander Graphs**.
    - The error function ϵ determined by d, n, s, and by the spectrum of the underlying graph.

Exact Computation – Construction

- Let C be an $[n, n-s]$ cyclic MDS code over $\mathbb{F} \in \{\mathbb{C}, \mathbb{R}\}$ such that $\overline{1} \in C$.
- Note – Hamming distance (*not* the ℓ_2 distance).
- Recall –
  - $C^{\perp}$ is an $[n, s]$ MDS code.
  - For any set $P \subseteq [n]$ of size $s+1$ there exists a word $c_P \in C$ such that $supp(c_P) = P$.
  - $C^R = \{(c_n, \ldots, c_1) \mid (c_1, \ldots, c_n) \in C\}$ is an $[n, n-s]$ cyclic MDS code.
- Let $c_1 = (\beta_1, \ldots, \beta_{s+1}, 0, \ldots, 0) \in C$, and let $c_2, \ldots, c_n$ be all its cyclic shifts.
- Define $B \triangleq \left(c_1^{\top}, c_2^{\top}, \ldots, c_n^{\top}\right)$.
- Observe:
  - For each row B_i, $|supp(B_i)| = s+1$.
  - Each B_i is a codeword of C^R.
  - $colspan(B) = C$.

$$B \triangleq \begin{pmatrix} \beta_1 & 0 & \cdots & 0 & \beta_{s+1} & \beta_s & \cdots & \beta_2 \\ \beta_2 & \beta_1 & 0 & \cdots & 0 & \beta_{s+1} & \cdots & \beta_3 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \ddots & \ddots & \cdots \\ \beta_s & \beta_{s-1} & \cdots & \beta_1 & 0 & \cdots & 0 & \beta_{s+1} \\ \beta_{s+1} & \beta_s & \cdots & \beta_2 & \beta_1 & 0 & \cdots & 0 \\ 0 & \beta_{s+1} & \cdots & \beta_3 & \beta_2 & \beta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \beta_{s+1} & \beta_s & \beta_{s-1} & \cdots & \beta_1 \end{pmatrix}$$

Exact Computation – Construction

- Claim – Any $n-s$ rows of B are linearly independent.
- Proof –
  - Assume there exists a vector $v \in \mathbb{F}^n$ with $weight(v) \leq n-s$ such that $vB = 0$.
  - Since $colspan(B) = C$, it follows that $v \in C^{\perp}$.
  - $C^{\perp}$ is an $[n, s]$ MDS code, and hence $weight(v) \geq n-s+1$, a contradiction.
- Corollary –
  - Any $n-s$ rows of B span C^R.
  - Since $\overline{1} \in C$, we have $\overline{1} \in C^R$.
  - $\overline{1}$ in the span of any $n-s$ rows of B. ■

Cyclic $[n, n-s]$ MDS code with $\overline{1}$
$\Rightarrow$
An (A, B) scheme for **EC**

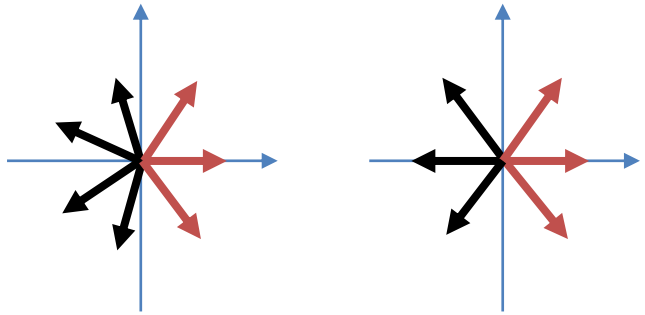
$$B \triangleq \begin{pmatrix} \beta_1 & 0 & \cdots & 0 & \beta_{s+1} & \beta_s & \cdots & \beta_2 \\ \beta_2 & \beta_1 & 0 & \cdots & 0 & \beta_{s+1} & \cdots & \beta_3 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \ddots & \ddots & \cdots \\ \beta_s & \beta_{s-1} & \cdots & \beta_1 & 0 & \cdots & 0 & \beta_{s+1} \\ \beta_{s+1} & \beta_s & \cdots & \beta_2 & \beta_1 & 0 & \cdots & 0 \\ 0 & \beta_{s+1} & \cdots & \beta_3 & \beta_2 & \beta_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \beta_{s+1} & \beta_s & \beta_{s-1} & \cdots & \beta_1 \end{pmatrix}$$

Cyclic $[n, n-s]$ MDS codes over $\mathbb{C}, \mathbb{R}$ containing $\overline{1}$

- Over $\mathbb{C}$, "Conventional" **Reed-Solomon** codes –
  - Let $\omega = e^{\frac{2\pi i}{n}}$ and $\alpha_j = \omega^j$ for all $j \in [n]$.
  - $C = \{(f(\alpha_1), f(\alpha_2), \dots, f(\alpha_n)) \mid \deg f \leq n-s-1\}$.
  - Clearly $\overline{1} \in C$.
  - For any $c \in C$ w/ polynomial $f_c(x)$, the polynomial $f_{c'}(x) \triangleq f_c(\omega x)$ is of the same degree, and c' is a cyclic rotation of c.

- Over $\mathbb{R}$, **BCH** codes for $n \neq s_{mod\ 2}$ (codewords are polynomials $(c_0, \dots, c_{n-1}) \leftrightarrow c(x) = \sum_{i=0}^{n-1} c_i x^i$) –
  - BCH codes – **real** polynomials with mutual **complex** roots.
  - Take a set J of s roots of unity that are **symmetric around** (-1).
  - E.g., $(n,s) = (6,3) \Rightarrow \{\omega^2, \omega^3, \omega^4\}$ and $(n,s) = (7,4) \Rightarrow \{\omega^2, \omega^3, \omega^4, \omega^5\}$
  - Reminder: C cyclic since the cyclic shift of $c(x)$ is $c'(x) = x \cdot c(x) - c_{n-1}(x^n - 1)$.
  - By BCH bound, $\mathrm{dist}(C) \geq s+1$.
  - Roots closed under conjugacy $\Rightarrow g(x) = \Pi_{j \in J}(x - \omega^j)$ is **real**
    $\Rightarrow C = \{g(x)p(x) \mid \deg p \leq n-s-1\} \Rightarrow \dim C \geq n-s \Rightarrow C$ is MDS.
  - $\overline{1}(\omega^j) \triangleq \sum_{t \in \{0,1,\dots,n-1\}} (\omega^j)^t = 0 \Leftrightarrow j \neq 0$. Hence, $\overline{1} \in C$.

$(n,s) = (7,4)$ $(n,s) = (6,3)$



ϵ-Approximate Computation

- Cannot have $s > d - 1$ for EC.
- Want: $d_2\left(A(K) \cdot B, \overline{1}\right) \leq \epsilon(|K^c|)$ for some decreasing function ϵ.
- Trivial scheme: $B = I_n, A(K) = \overline{1}_K \Rightarrow$ storage overhead $d = 1$ and $\epsilon(s) = d_2\left(A(K) \cdot B, \overline{1}\right) = d_2\left(\overline{1}_K, \overline{1}\right) = \sqrt{s}$.
- Trivial scheme is used in practice!
- Can we do better?
- Tool: Adjacency matrices of **expander graphs**.

Background – Expander Graphs.

- The **spectrum of a (connected d-regular) graph** $G = (V, E)$ is the spectrum of its adjacency matrix $((A_G)_{i,j} = 1 \Leftrightarrow \{i, j\} \in E)$.
  - $\lambda_1 = d \leq \lambda_2 \leq \cdots \leq \lambda_n$, with $\lambda_n = -d \Leftrightarrow G$ is bipartite.
  - A_G is real and symmetric, and hence has n orthogonal eigenvectors $v_1 = \overline{1}, v_2, \ldots, v_n$ (w.l.o.g. $\|v_i\|_2 = 1$ for all $i = 2, \ldots, n$).
  - $\lambda_G \triangleq \max\{|\lambda_2|, |\lambda_n|\}$.
  - Alon-Boppana – $\lambda_G \geq 2\sqrt{d-1} - o_n(1)$.
  - G is a **Ramanujan graph** if $\lambda_G \leq 2\sqrt{d-1}$.
- Motivation – Construct **expanders**, i.e., graphs with **good connectivity** $c(G) \triangleq \min_{U \subseteq V, |U| \leq \frac{n}{2}} \frac{E(U, U^c)}{|U|}$.
  - Usually, $d = O(1)$.
- Cheeger's inequality – $c(G) \geq \frac{d - \lambda_G}{2}$.
- Hence, graphs with **small λ_G** are **good expanders**.

$d - \lambda_G$ is called the **spectral gap of G**

ϵ-AC from Expander Graphs.

- $u_K \cdot \overline{1} = 0 \Rightarrow u_K \in span(v_2, \dots, v_n) \Rightarrow u_K = \alpha_2 v_2 + \alpha_3 v_3 + \cdots + \alpha_n v_n.$
- $\|u_K\|_2 = \sqrt{\left\langle \left(\sum_{i\in\{2,\dots,n\}} \alpha_i\, v_i\right) \cdot \left(\sum_{i\in\{2,\dots,n\}} \alpha_i\, v_i\right) \right\rangle} = \sqrt{\sum_{i\in\{2,\dots,n\}} \alpha_i^2} = \sqrt{\frac{ns}{n-s}}$
- Let $B = \frac{1}{d} \cdot A_G$ and $A(K) = \overline{1} + u_k$.
  - storage overhead d.
  - $supp\big(A(K)\big) = K$ for all K.

For $K \in \mathcal{P}([n])$ of size $n - s$ let –

$$(u_K)_i = \begin{cases} -1, i \notin K \\ \frac{s}{n-s}, i \in K \end{cases}$$

- Claim – $d_2\left(A(K) \cdot B, \overline{1}\right) \leq \frac{\lambda_G}{d}\sqrt{\frac{ns}{n-s}}.$
- Proof – $d_2\left(A(K) \cdot B, \overline{1}\right) = d_2\left(\left(\overline{1} + u_K\right) \cdot B, \overline{1}\right) = d_2\left(\left(\overline{1} + \alpha_2 v_2 + \alpha_3 v_3 + \cdots + \alpha_n v_n\right) \cdot B, \overline{1}\right)$
  $= \left\|\left(\overline{1} + \alpha_2 v_2 + \alpha_3 v_3 + \cdots + \alpha_n v_n\right) \cdot B - \overline{1}\right\|_2 = \left\|\overline{1} \cdot \frac{\lambda_1}{d} + \alpha_2 \frac{\lambda_2}{d} v_2 + \alpha_3 \frac{\lambda_3}{d} v_3 + \cdots + \alpha_n \frac{\lambda_n}{d} v_n - \overline{1}\right\|_2$
  $= \sqrt{\sum_{j=2}^{n} \frac{\lambda_j \alpha_j^2}{d}} \leq \frac{\lambda_G}{d}\sqrt{\sum_{j=2}^{n} \alpha_j^2} = \frac{\lambda_G}{d}\|u_K\|_2 = \frac{\lambda_G}{d}\sqrt{\frac{ns}{n-s}}.$

ϵ-AC from Expander Graphs.

- Claim – $d_2\left(A(K)\cdot B, \overline{1}\right) \leq \frac{\lambda_G}{d}\sqrt{\frac{ns}{n-s}}$.
- The trivial scheme attains $\epsilon(s) = \sqrt{s}$ and storage overhead 1.
- Asymptotic –

  Lower error for any d-regular graph and any s such that $\frac{\lambda_G}{d}\sqrt{\frac{n}{n-s}} < 1$ (True for any $s = o(n)$).
- Finite length examples (% improvement over trivial) –
  - Margulis graphs (8-regular graphs for any n with $\lambda_G = 5\sqrt{2}$) – For $n = 500$, $s = 50$, improvement of 6.8%.
  - Ramanujan graphs [Lubotzky, Phillip, Saranac] –
    - For $n = 1092, s = 20, d = 6$, improvement of 19.7%.
    - For $n = 2448, s = 100, d = 14$, improvement of 47%.

Thank you!

Questions?