CS 15-859: Algorithms for Big Data	Fall 2019
Lecture 4 (part 2) — September 26, 2019	
Prof. David Woodruff	*Scribe: Anthony Hsu*

1 Leverage Score Sampling (continued)

As a reminder, our leverage score sampling matrix is defined as follows:

Definition. (Leverage score sampling matrix) Define the $k \times n$ sampling matrix $S = D \cdot \Omega^T$ where D is $k \times k$ and Ω is $n \times k$, and

- Ω is a sampling matrix where for each column j, we independently, and with replacement, pick a row index i in $[n]$ with probability q_i and set $\Omega_{i,j} = 1$
- D is a rescaling matrix with $D_{j,j} = 1/\sqrt{q_i k}$, where q_i is the probability of the row index i picked by Ω in column j

Claim 1. Leverage score sampling gives a subspace embedding (i.e. $\|SUy\|_2^2 = (1 \pm \varepsilon)\|y\|_2^2$).

Proof. We prove this by showing the equivalent statement $\left\|U^T S^T S U - I\right\|_2 \leq \varepsilon$ with high probability. We will do this by applying the matrix Chernoff bound. As a reminder, the matrix Chernoff bound states:

Theorem 1. *(Matrix Chernoff Bound) Let $X_1, ..., X_k$ be independent copies of a symmetric random matrix $X \in \mathbb{R}^{d \times d}$ with $\mathbb{E}[X] = 0$, $\|X\|_2 \leq \gamma$, and $\left\|\mathbb{E}[X^T X]\right\|_2 \leq \sigma^2$. Let $W = \frac{1}{k}\sum_{j \in [k]} X_j$. For any $\varepsilon > 0$,*

$$\mathbf{Pr}[\|W\|_2 > \varepsilon] \leq 2d \cdot e^{-k\varepsilon^2/(\sigma^2 + \frac{\gamma\varepsilon}{3})}$$

where $\|W\|_2 = \sup \frac{\|Wx\|_2}{\|x\|_2}$, which is equal to $\sup_{\|x\|_2=1} x^T W x$ since W is symmetric.

For our proof that leverage score sampling gives a subspace embedding, we defined the following:

- $i(j)$ denotes the index of the row of U sampled in the j-th trial
- $X_j = I_d - \frac{U_{i(j)}^T U_{i(j)}}{q_{i(j)}}$, where $U_{i(j)}$ is the j-th sampled row of U

We showed that

- The X_j's are independent copies of a symmetric matrix random variable
- $\mathbb{E}[X_j] = 0^{d \times d}$
- $\|X_j\|_2 \leq 1 + \frac{d}{\beta}$ (where β was defined so that $q_i \geq \frac{\beta \ell(i)}{d}$ for all i)

- $\mathbb{E}[X^T X] \leq \left(\frac{d}{\beta} - 1\right) I_d$ (where $A \leq B$ means $x^T A x \leq x^T B x$ for all x)

Now we can apply the matrix Chernoff bound with $\gamma = 1 + \frac{d}{\beta}$ and $\sigma^2 = \frac{d}{\beta} - 1$.

$$\begin{aligned}
W &= \frac{1}{k}\sum_{j=1}^{k} X_j \\
&= \frac{1}{k}\sum_{j=1}^{k}\left(I_d - \frac{U_{i(j)}^T U_{i(j)}}{q_{i(j)}}\right) \\
&= I_d - \frac{1}{k}\sum_{j=1}^{k}\frac{U_{i(j)}^T U_{i(j)}}{q_{i(j)}}
\end{aligned}$$

To see what the summation evaluates to, note that

$$U^T S^T S U = \underset{d \times n}{\begin{bmatrix} \quad U^T \quad \end{bmatrix}} \underset{n \times k}{\begin{bmatrix} \\ \Omega \\ \\ \end{bmatrix}} \underset{k \times k}{\begin{bmatrix} D^T \end{bmatrix}} \underset{k \times k}{\begin{bmatrix} D \end{bmatrix}} \underset{k \times n}{\begin{bmatrix} \quad \Omega^T \quad \end{bmatrix}} \underset{n \times d}{\begin{bmatrix} \\ U \\ \\ \end{bmatrix}}$$

Our sampling matrix $S = D \cdot \Omega^T$ chooses some rows of U and scales each $U_{i(j)}$ by $1/\sqrt{kq_{i(j)}}$. So the j-th row of SU is just

$$\begin{aligned}
(SU)_j &= \frac{1}{\sqrt{kq_{i(j)}}} U_{i(j)} \\
\Leftrightarrow U_{i(j)} &= \sqrt{kq_{i(j)}}(SU)_j
\end{aligned}$$

Plugging in, we get

$$\begin{aligned}
W &= I_d - \frac{1}{k}\sum_{j=1}^{k}\frac{\left(\sqrt{kq_{i(j)}}(SU)_j\right)^T\left(\sqrt{kq_{i(j)}}(SU)_j\right)}{q_{i(j)}} \\
&= I_d - \sum_{j=1}^{k}((SU)_j)^T(SU)_j \\
&= I_d - U^T S^T S U
\end{aligned}$$

Substituting into the matrix Chernoff bound, we get

$$\mathbf{Pr}\left[\left\|I_d - U^T S^T S U\right\|_2 > \varepsilon\right] \leq 2d \cdot e^{-k\varepsilon^2 \Theta(\beta/d)}$$

Set $k = \Theta(\frac{d \log d}{\beta \varepsilon^2})$ and we can get an arbitrarily small bound, implying that SU is a subspace embedding with high probability. ■

However, we still have a problem: how do we calculate the leverage scores $\ell(i)$?

2 Fast Computation of Leverage Scores

As a reminder, the leverage score of a matrix is defined as follows:

Definition. (Leverage score) Given an $n \times d$ matrix A with rank d and its SVD $U\Sigma V^T$, the i-th leverage score $\ell(i)$ of A is defined to be $\|U_{i,*}\|_2^2$.

Naively, we could calculate the leverage scores by computing the SVD of A, but this requires $O(nd^2)$ time. Instead, we will compute a subspace embedding SA and use it to compute the leverage scores.

Definition. (Approximate leverage score) Let $SA = QR^{-1}$ such that Q is an $s \times d$ matrix with orthonormal columns and R^{-1} is a $d \times d$ matrix. We define an approximate leverage score to be $\ell'_i = \left\|e_i^T AR\right\|_2^2$

Claim 2. ℓ'_i is a $1 \pm O(\varepsilon)$ approximation of ℓ_i.

Proof. Since AR has the same column span as A, we can write $AR = UT^{-1}$, where U is from A's SVD and T^{-1} is some matrix. We know

$$(1 - \varepsilon) \|ARx\|_2 \leq \|SARx\|_2 = \|Qx\|_2 = \|x\|_2$$

and also

$$(1 + \varepsilon) \|ARx\|_2 \geq \|SARx\|_2 = \|Qx\|_2 = \|x\|_2$$

Thus,

$$(1 \pm O(\varepsilon)) \|x\|_2 = \|ARx\|_2 = \left\|UT^{-1}\right\|_2 = \left\|T^{-1}x\right\|_2$$

$\|T^{-1}x\|_2 = (1 \pm O(\varepsilon)) \|x\|_2$ implies T^{-1} is well-conditioned, i.e. all its singular values must be about 1. Therefore,

$$\begin{aligned}
\ell_i &= \left\|e_i^T U\right\|_2^2 \\
&= \left\|e_i^T ART\right\|_2^2 \\
&= (1 \pm O(\varepsilon)) \left\|e_i^T AR\right\|_2^2 && \text{since } T \text{ is well-conditioned} \\
&= (1 \pm O(\varepsilon))\ell'_i
\end{aligned}$$

We have now shown that ℓ'_i is a $(1 \pm O(\varepsilon))$ approximation of the actual leverage score ℓ_i. ■

So we can compute a single leverage score in poly(d) time. But how do we calculate *all* the leverage scores quickly? We'd like something about nnz(A) time.

Naively, we could compute AR, but this takes too long. Instead, we'd like to sketch R while preserving row norms. We take advantage of the following lemma (used to prove the Johnson-Lindenstrauss lemma), which we state without proof:

Lemma 1. *Let G be a $d \times O(\log n)$ matrix of i.i.d. normal random variables. Then, for all vectors z,*

$$\mathbf{Pr}\Big[\left\|z^T G\right\|_2^2 = (1 \pm \varepsilon)\left\|z\right\|_2^2\Big] \geq 1 - \delta$$

Substituting in $e_i^T AR$ for z, we get

$$\mathbf{Pr}\Big[\left\|e_i^T ARG\right\|_2^2 = (1 \pm \varepsilon)\left\|e_i^T AR\right\|_2^2\Big] \geq 1 - \delta$$

Claim 3. We can now compute the approximate leverage scores ℓ_i' in $(\mathrm{nnz}(A) + d^2)\log n$ time.

Proof.

Definition. Set $\ell_i' = \left\|e_i^T ARG\right\|_2^2$.

We can calculate RG in $O(d^2 \log n)$ time, which results in a $d \times O(\log n)$ matrix. We can multiply that matrix by A in $\mathrm{nnz}(A)\log n$ time. Thus, the total time to compute the approximate leverage scores ℓ_i' is $(\mathrm{nnz}(A) + d^2)\log n$. ■

We can thus solve regression in $(\mathrm{nnz}(A) + \mathrm{poly}(d/\varepsilon))\log n$ time.

3 Distributed low rank approximation

We have shown some fast algorithms for doing low-rank approximation. A natural follow-up question is: are there such algorithms for a distributed setting? A matrix A might be distributed among s servers because it either can't fit on a single machine or because there are multiple machines collecting data.

Suppose we have s servers. If each is collecting customer-product information, then each server t has its own customer-product matrix A^t. The full customer-product matrix is then $A = A^1 + A^2 + ... + A^s$. This is known as the **arbitrary partition model**. Another model is the row partition model, where each server just gets a subset of rows of A. The arbitrary partition model is more general than the row partition model.

3.1 Communication Model

Before discussing low-rank approximation algorithms in a distributed setting, we first need to define our communication model. We will consider a setting where each of the s servers only communicates with a special coordinator machine. Servers cannot talk to one another directly. All communication must pass through the coordinator. Communication is two-way, meaning each server can talk to the coordinator, and the coordinator can talk to each server.

We can simulate point-to-point (server-to-server) communication up to a factor of 2 (since for each message we need to do an additional hop through the coordinator) and an additive $O(\log s)$ bits per message (since we also need to append the destination server to each message sent to the coordinator).

3.2 Communication cost of low rank approximation

Now consider the following problem:

Input: An $n \times d$ matrix A is split across s servers, each with its $n \times d$ matrix A^t. $A = A^1 + A^2 + ... + A^2$. Assume the entries of A^t are $O(\log(nd))$-bit integers.

Output: Each server outputs its part of the matrix projected onto the same k-dimensional subspace W of $\mathbb{R}^d$. Server t will output $A^t P_W$, where P_W is a projection matrix onto W. Note that $P_W = VV^T$ for some basis V with k columns. V is $d \times k$. The final output is

$$C = A^1 P_W + A^2 P_W + ... + A^s P_W = AP_W$$

and should satisfy

$$\|A - C\|_F \leq (1 + \varepsilon) \|A - A_k\|_F$$

where A_k is the optimal k-dimensional approximation of A.

Resource Goals: We want to minimize the amount of communication and computation. Ideally, we want $O(1)$ rounds of communication and input sparsity time.

Remark 1. One such application of a distributed low rank approximation algorithm is for doing k-means clustering.

3.3 Prior work on distributed low rank approximation

- [FSS13]: This introduced the first protocol for the row-partition model. The protocol uses $O(sdk/\varepsilon)$ real numbers (bit complexity is not analyzed, though) and depends linearly on the number of servers s and the matrix dimension d but does not depend on n. For info on SVD running time, see [BKLW14].
- [KVW13]: Introduced the arbitrary partition model. Provides a communication protocol that uses $O(skd/\varepsilon)$ words of size $\log(nd)$ bits.
- [BWZ16]: Presents a communication protocol for the arbitrary partition model that requires $O(skd) + \text{poly}(sk/\varepsilon)$ words. Notice the first term does not depend on ε. It also proves that $\Omega(skd)$ is an optimal lower bound.
- Other variants include kernel low rank approximation [BLS$^+$15], low rank approximation of an implicit matrix [WZ16], and low rank approximation of sparse matrices [BWZ16].

We will now go through three of these protocols.

3.4 Constructing a coreset [FSS13]

Let $A = U\Sigma V^T$, $m = k + k/\varepsilon$ (where k is the target rank and k/ε is small compared to n or d), and Σ_m be the singular value matrix in A's SVD that agrees with Σ for the m largest singular values and is 0 elsewhere.

Claim 4. For all projection matrices $Y = I_d - X$ (where X is a projection matrix WW^T (where W is $d \times k$) onto a k-dimensional subspace) onto a $(d-k)$-dimensional subspace,

$$\left\|\Sigma_m V^T Y\right\|_F^2 + c = (1 \pm \varepsilon) \|AY\|_F^2$$

where $c = \|A - A_m\|_F^2$ (where A_m is the m-rank approximation of A)

Remark 2. You can think of $\Sigma_m V^T Y$ and $A - A_m$ as our **coreset**.

Remark 3. The claim says we can get a good k-dimensional approximation to AY (the distance of A from X) while storing just an m-rank approximation $\Sigma_m V^T$ plus a scalar. You can think of $\Sigma_m V^T$ as applying a sketching matrix $S = U_m^T$ to A: $SA = U_m^T U \Sigma V^T = \Sigma_m V^T$.

Proof.

$$\begin{aligned}
\|AY\|_F^2 &= \left\|U\Sigma_m V^T Y\right\|_F^2 + \left\|U(\Sigma - \Sigma_m)V^T Y\right\|_F^2 && \text{We break } AY \text{ up into two orthogonal components} \\
&= \left\|U\Sigma_m V^T Y\right\|_F^2 + \|(A - A_m)Y\|_F^2 \\
&\le \left\|\Sigma_m V^T Y\right\|_F^2 + \|A - A_m\|_F^2 && \text{Projection can only reduce norms} \\
&= \left\|\Sigma_m V^T Y\right\|_F^2 + c
\end{aligned}$$

Now we want to bound $\left\|\Sigma_m V^T Y\right\|_F^2 + c - \|AY\|_F^2$:

$$\begin{aligned}
&\left\|\Sigma_m V^T Y\right\|_F^2 + \|A - A_m\|_F^2 - \|AY\|_F^2 \\
&= \left\|\Sigma_m V^T\right\|_F^2 - \left\|\Sigma_m V^T X\right\|_F^2 + \|A - A_m\|_F^2 - \|A\|_F^2 + \|AX\|_F^2 && \left[\|A\|_F^2 = \|AX\|_F^2 + \|AY\|_F^2\right] \\
&= \|AX\|_F^2 - \left\|\Sigma_m V^T X\right\|_F^2 && \left[\|A+B\|_F^2 = \|A\|_F^2 + \|B\|_F^2 + 2\operatorname{Tr}(A^T B)\right] \\
&= \left\|(\Sigma - \Sigma_m)V^T X\right\|_F^2 \\
&\le \left\|(\Sigma - \Sigma_m)V^T\right\|_F^2 \|X\|_F^2 \\
&\le \sigma_{m+1}^2 k && \left[\|X\|_F^2 = \left\|WW^T\right\|_F^2 = \|W\|_F^2 = k\right] \\
&= \varepsilon \sigma_{m+1}^2 (m - k) && [m = k + k/\varepsilon] \\
&\le \varepsilon \sum_{i=k+1}^{m+1} \sigma_i^2 \\
&\le \varepsilon \|A - A_k\|_F^2
\end{aligned}$$

This implies

$$\begin{aligned}
\left\|\Sigma_m V^T Y\right\|_F^2 + \|A - A_m\|_F^2 - \|AY\|_F^2 &\le \varepsilon \|A - A_k\|_F^2 \\
\Leftrightarrow \left\|\Sigma_m V^T Y\right\|_F^2 + c &\le \|AY\|_F^2 + \varepsilon \|A - A_k\|_F^2 \\
&\le \|AY\|_F^2 + \varepsilon \|AY\|_F^2 \\
&= (1 + \varepsilon) \|AY\|_F^2
\end{aligned}$$

■

Next lecture we will see how to use coresets to build a an efficient distribution communication protocol for low-rank approximation.

References

[BKLW14] Maria-Florina Balcan, Vandana Kanchanapally, Yingyu Liang, and David P. Woodruff. Improved distributed principal component analysis. *CoRR*, abs/1408.5823, 2014.

[BLS+15] Maria-Florina Balcan, Yingyu Liang, Le Song, David P. Woodruff, and Bo Xie. Distributed kernel principal component analysis. *CoRR*, abs/1503.06858, 2015.

[BWZ16] Christos Boutsidis, David P. Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 236–249, New York, NY, USA, 2016. ACM.

[FSS13] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 1434–1453, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics.

[KVW13] Ravi Kannan, Santosh S. Vempala, and David P. Woodruff. Principal component analysis and higher correlations for distributed data. In *COLT*, 2013.

[WZ16] D. P. Woodruff and P. Zhong. Distributed low rank approximation of implicit functions of a matrix. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 847–858, May 2016.