Streaming Coresets for Tensor Factorization

Rachit Chhaya, Jayesh Choudhari, Anirban Dasgupta and **Supratim Shit**

IIT Gandhinagar, India

Outline

- Motivation
- Problem Statement
- Main Algorithms
- Compare Guarantee
- Experiments

Motivation

Large data

Running time of any data analysis algorithm depends on datasize.

Courtesy: Google images

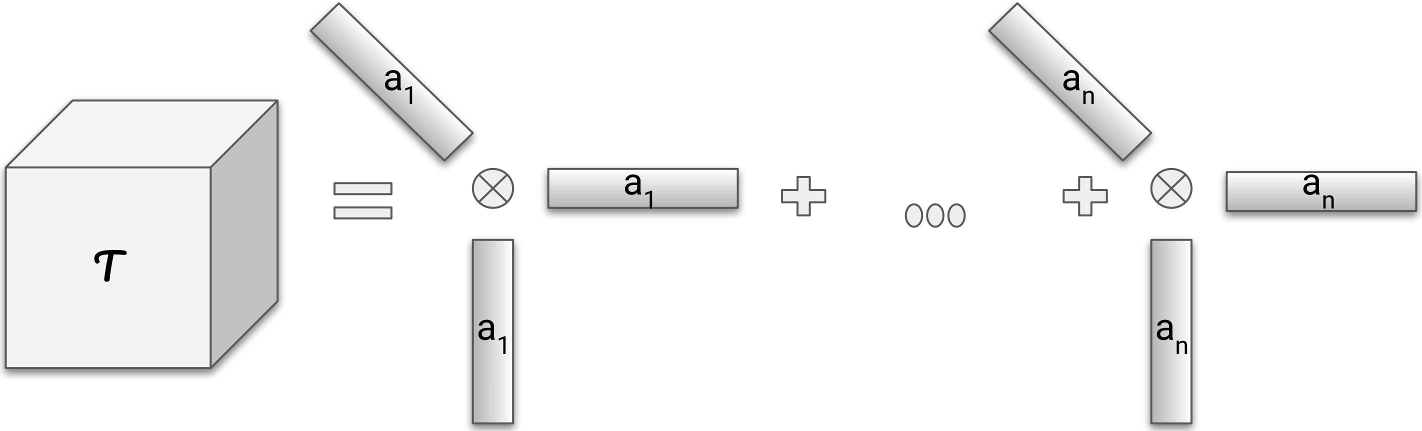
Coreset

Given a data set **A** and an algorithm M, a reduced set **C** is called a **coreset** if one can efficiently reduce from **A** to **C** such that M(**C**) $\simeq$ M(**A**)

Given an n x d matrix **A**, an m x d matrix **C** is an ε-coreset for ℓ_2 subspace embedding if $\forall$ **x**,

$$(1-\varepsilon)||\mathbf{Ax}||_2 \leq ||\mathbf{Cx}||_2 \leq (1+\varepsilon)||\mathbf{Ax}||_2$$

Tensors



Tensor factorization used to learn latent variables, neural networks parameters etc.

Tensor contraction is one of the most important operation in tensor factorization.

$$\mathcal{T}(\mathbf{x},\mathbf{x},\ldots,\mathbf{x}) = \sum_{\mathbf{a}_i^T \in \mathbf{A}} (\mathbf{a}_i^T \mathbf{x})^p$$

Anandkumar, Animashree, et al. "Tensor decompositions for learning latent variable models." *Journal of Machine Learning Research* 15 (2014): 2773-2832.

Problem Statement

Given **A**, set of n vectors each in R^d, coming in streaming fashion, is there an *efficient* way of choosing a set of m vector in **C** such that it is an ε-coreset for p-order tensor contraction, for integer p ≥ 2 and ∀**x** ∈ **Q**, where **Q** is a k-dimensional

$$\left| \sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} (\tilde{\mathbf{a}}_j^T \mathbf{x})^p - \sum_{i \in [n]} (\mathbf{a}_i^T \mathbf{x})^p \right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p$$

Note: With following cost function which is *similar but not the same*, the set **C** is also an ε-coreset for ℓ_p subspace embedding.

$$\left| \sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} |\tilde{\mathbf{a}}_j^T \mathbf{x}|^p - \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p \right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p$$

Our Contribution

- We show two main modules **LineFilter** and **KernelFilter**, based on which we propose four streaming algorithms. which matches or beats the current state of the art in terms of sampling complexity, update time and working space.

- *LineFilter*: Online algorithm, for every incoming vectors it decides which one to sample. The expected sample size is $\tilde{o}(n^{1-2/p}dk)$.

- *KernelFilter*: Online algorithm for very incoming vector first it Kernelizes to a higher dimension vector. Then it decides whether to sample the vector or not. It returns expected sample size of $\tilde{o}(d^{p/2}k)$ for even p and $\tilde{o}(n^{1/(p+1)}d^{p/2}k)$ for odd p.

- At $p = 2$, our online row sampling method ensures a relative error approximation.

Importance Sampling

Sensitivity score for i^th vector,

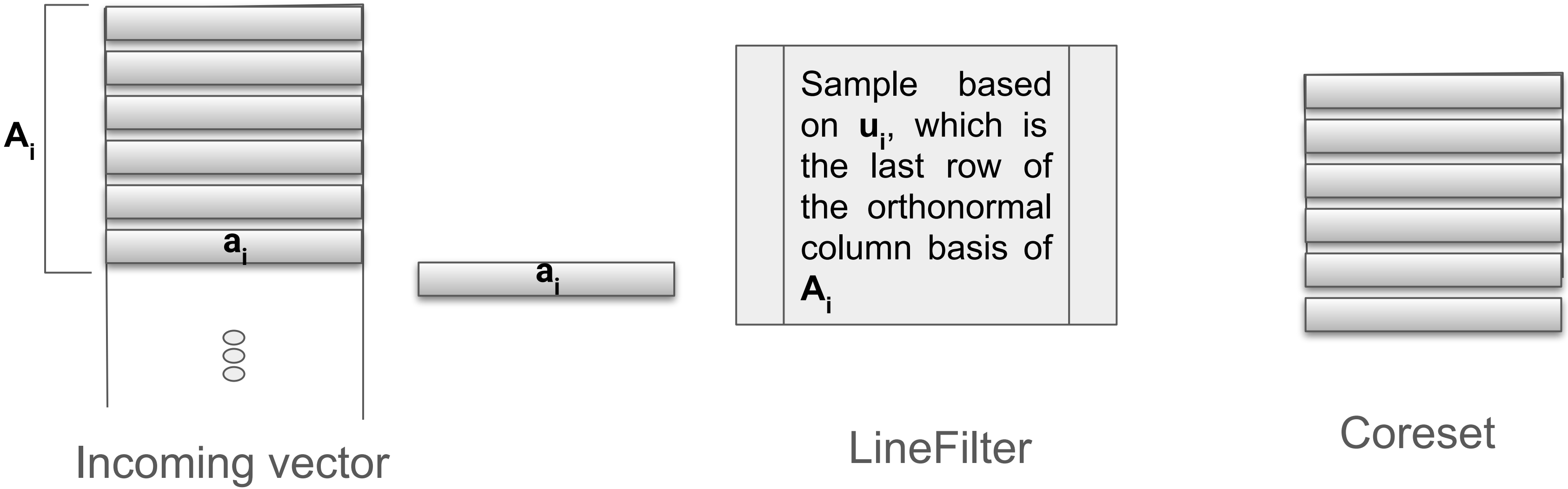
$$\tilde{s}_i = \sup_{\mathbf{x}} \frac{|\mathbf{a}_i^T \mathbf{x}|^p}{\sum_{j \le i} |\mathbf{a}_j^T \mathbf{x}|^p} \quad \tilde{e}_i$$

A set of sub-sampled vectors form **A** based on $\tilde{e}_i$ solves our problem.

The sample size depends on sum of these scores.

Feldman, Dan, and Michael Langberg. "A unified framework for approximating and clustering data." *Proceedings of the forty-third annual ACM symposium on Theory of computing*. 2011.

LineFilter



Here $\tilde{e}_i = \min\{1, r \cdot n^{p/2-1} \|\mathbf{u}_i\|^p\}$ and $p_i = \frac{\tilde{e}_i}{\sum_{j \le i} \tilde{e}_j}$

LineFilter Result

To ensure, $\forall \mathbf{x} \in \mathbf{Q}$, which is a k-dimensional subspace

$$\left| \sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} (\tilde{\mathbf{a}}_j^T \mathbf{x})^p - \sum_{i \in [n]} (\mathbf{a}_i^T \mathbf{x})^p \right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p$$ and $$\left| \sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} |\tilde{\mathbf{a}}_j^T \mathbf{x}|^p - \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p \right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p$$

Update time: $O(d^2)$

Working Space: $O(d^2)$

Coreset Size: $\tilde{O}(n^{1-2/p} d k \varepsilon^{-2})$

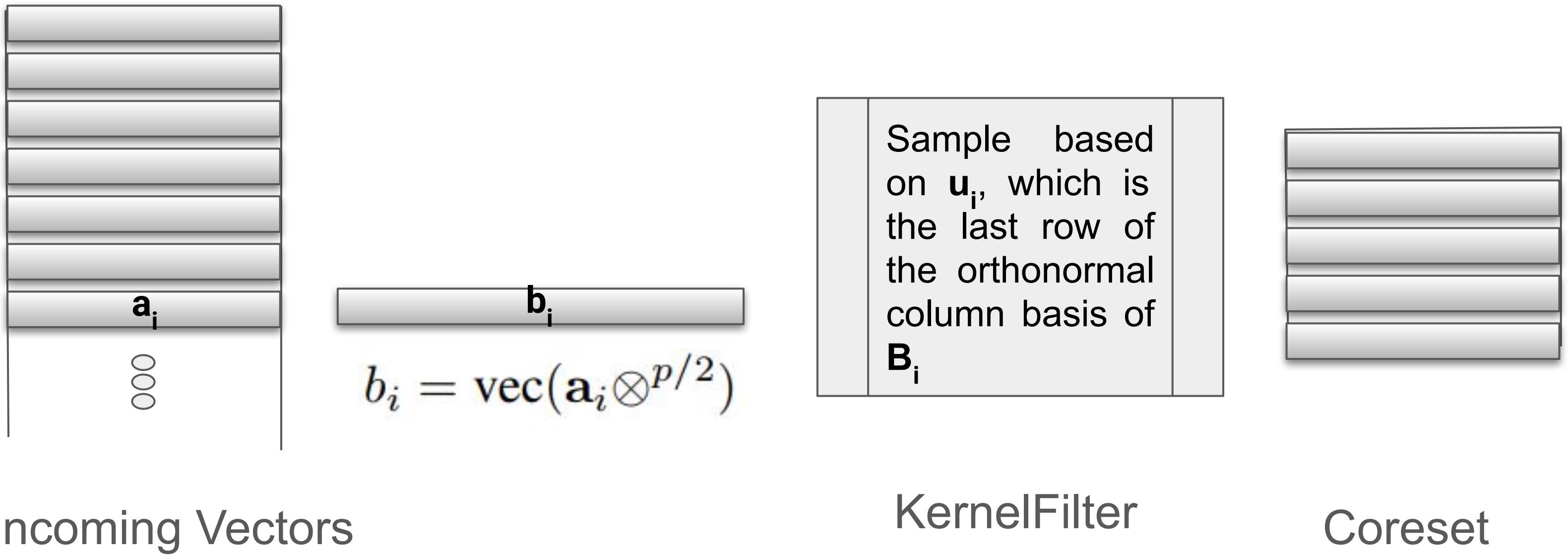
KernelFilter (even *p*)

$\mathbf{a}_i$ $\qquad$ $\mathbf{b}_i$

$$b_i = \text{vec}(\mathbf{a}_i \otimes^{p/2})$$

$$\tilde{s}_i = \sup_{\mathbf{x}} \frac{|\mathbf{a}_i^T \mathbf{x}|^p}{\sum_{j \le i} |\mathbf{a}_j^T \mathbf{x}|^p} \le \sup_{\mathbf{y}} \frac{|\mathbf{b}_i^T \mathbf{y}|^2}{\sum_{j \le i} |\mathbf{b}_j^T \mathbf{y}|^2}$$

KernelFilter (even p)



Here $\tilde{e}_i = \min\{1, r \cdot ||\mathbf{u}_i||^2\}$ and $p_i = \frac{\tilde{e}_i}{\sum_{j \leq i} \tilde{e}_j}$

KernelFilter (odd p)

$$\mathbf{b}_i = \text{vec}(\mathbf{a}_i \otimes^{\lceil p/2 \rceil})$$

$$\tilde{s}_i = \sup_{\mathbf{x}} \frac{|\mathbf{a}_i^T \mathbf{x}|^p}{\sum_{j \le i} |\mathbf{a}_j^T \mathbf{x}|^p} \le \sup_{\mathbf{y}} \frac{|\mathbf{b}_i^T \mathbf{y}|^{2p/(p+1)}}{\sum_{j \le i} |\mathbf{b}_j^T \mathbf{y}|^{2p/(p+1)}}$$

Here $\tilde{e}_i = \min\{1, r \cdot ||\mathbf{u}_i||^{2p/(p+1)}\}$ and $p_i = \frac{\tilde{e}_i}{\sum_{j \le i} \tilde{e}_j}$

KernelFilter Result

To ensure, $\forall$ **x** ϵ **Q**, which is a k-dimensional subspace

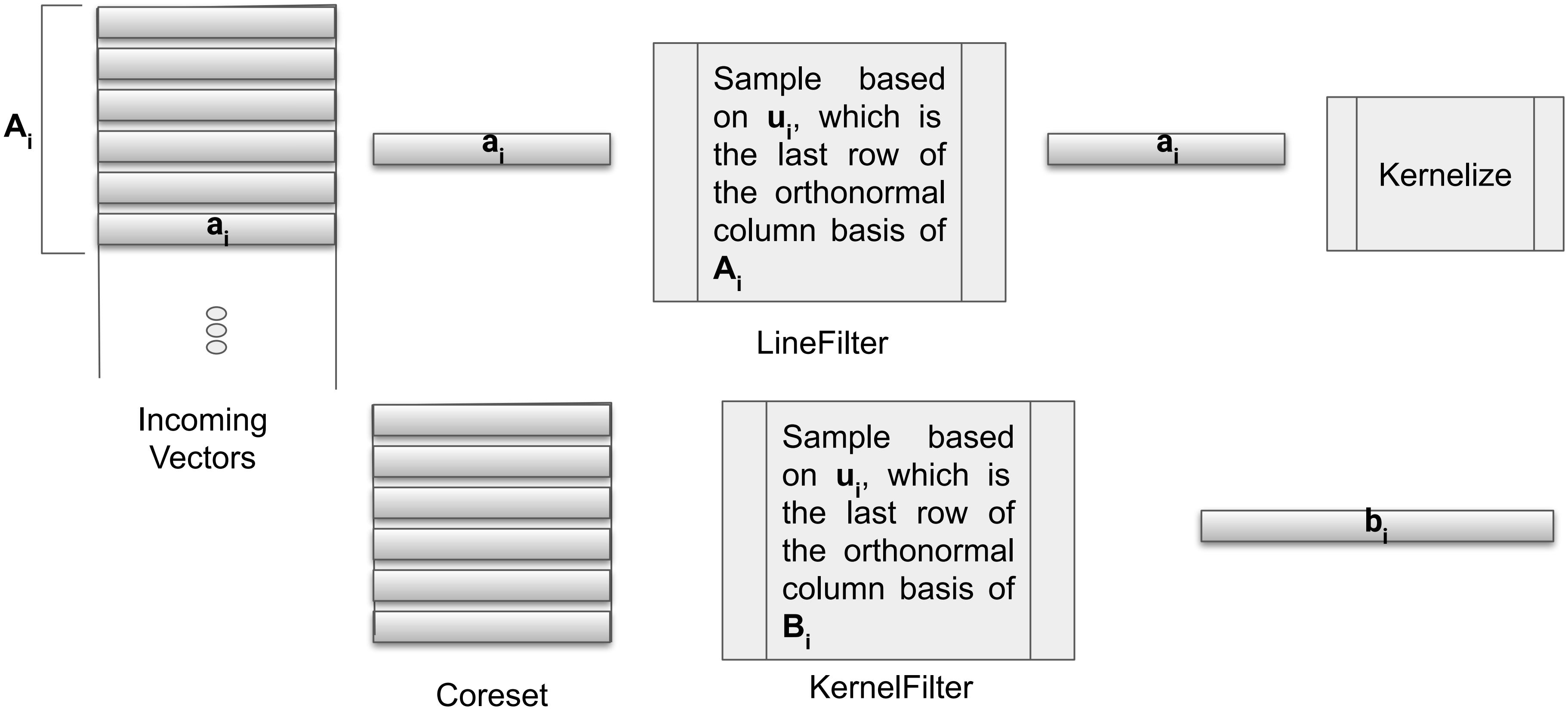
$$\left|\sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} (\tilde{\mathbf{a}}_j^T \mathbf{x})^p - \sum_{i \in [n]} (\mathbf{a}_i^T \mathbf{x})^p\right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p \quad \text{and} \quad \left|\sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} |\tilde{\mathbf{a}}_j^T \mathbf{x}|^p - \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p\right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p$$

Update time: $O(d^{p+1})$

Working Space: $O(d^{p+1})$

Coreset Size: even p - $\tilde{O}(d^{p/2} k \varepsilon^{-2})$ and odd p - $\tilde{O}(n^{1/(p+1)} d^{p/2} k \varepsilon^{-2})$

LineFilter + KernelFilter



LineFilter+KernelFilter Result

To ensure, $\forall \mathbf{x} \in \mathbf{Q}$, which is a k-dimensional subspace

$$\left|\sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} (\tilde{\mathbf{a}}_j^T \mathbf{x})^p - \sum_{i \in [n]} (\mathbf{a}_i^T \mathbf{x})^p\right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p \quad \text{and} \quad \left|\sum_{\tilde{\mathbf{a}}_j \in \mathbf{C}} |\tilde{\mathbf{a}}_j^T \mathbf{x}|^p - \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p\right| \leq \epsilon \cdot \sum_{i \in [n]} |\mathbf{a}_i^T \mathbf{x}|^p$$

Amortized Update time: $O(d^2)$

Working Space: $O(d^{p+1})$

Coreset Size: even p - $\tilde{O}(d^{p/2}k\varepsilon^{-2})$ and odd p - $\tilde{O}(n^{(p-2)/p(p+1)}d^{p/2+1/4}k^{5/4}\varepsilon^{-2})$

Comparison of Our Results

Existing Results

Algorithm	Sample Size $\tilde{O}(\cdot)$	Update time	Working space $\tilde{O}(\cdot)$
StreamingWCB (Dasgupta et al., 2009)	$d^p k \epsilon^{-2}$	$d^5 p \log d$	$d^p k \epsilon^{-2}$
StreamingLW (Cohen & Peng, 2015)	$d^{p/2} k \epsilon^{-5}$	$d^C p \log d$	$d^{p/2} k \epsilon^{-5}$
StreamingFC (Clarkson et al., 2016)	$d^{7p/2} \epsilon^{-2}$	d	$d^{7p/2} \epsilon^{-2}$
Streaming (Dickens et al., 2018)	$n^{\gamma} d \epsilon^{-2}$	$n^{\gamma} d^5$	$n^{\gamma} d$

Our Results

Algorithm	Sample Size $\tilde{O}(\cdot)$	Update time	Working space $\tilde{O}(\cdot)$
LineFilter	$n^{1-2/p} d k \epsilon^{-2}$	d^2	d^2
LineFilter+StreamingLW	$d^{p/2} k \epsilon^{-5}$	d^2 amortized	$d^{p/2} k \epsilon^{-5}$
KernelFilter (even p)	$d^{p/2} k \epsilon^{-2}$	d^p	d^p
KernelFilter (odd p)	$n^{1/(p+1)} d^{p/2} k \epsilon^{-2}$	d^{p+1}	d^{p+1}
LineFilter+KernelFilter (even p)	$d^{p/2} k \epsilon^{-2}$	d^2 amortized	d^p
LineFilter+KernelFilter (odd p)	$n^{(p-2)/(p^2+p)} d^{p/2+1/4} k^{5/4} \epsilon^{-2}$	d^2 amortized	d^{p+1}

Har-Peled, Sariel, and Soham Mazumdar. "On coresets for k-means and k-median clustering." *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. 2004.

Experimental Results on Topic Modeling

Data: 10K data points from 20Newsgroups dataset.

Sampling method: Uniform, LinerFilter(2), LineFilter+KernelFilter

Output: Taking the best matching between empirical and estimated topics based on ℓ_1 distance and report the average ℓ_1 difference between them.

SAMPLE	UNIFORM	LINEFILTER(2)	LINEFILTER +KERNELFILTER
50	0.5725	0.6903	**0.5299**
100	0.5093	0.6385	**0.4379**
200	0.4687	0.5548	**0.3231**
500	0.3777	0.3992	**0.2173**
1000	0.2548	0.2318	**0.1292**

Future Work

- Improve or remove the factor n for odd value p of our online algorithm.

- Improve running time of LineFilter to input sparsity time.

- Improve the update time for KernelFilter to make it practical for any value of d.

References

Anandkumar, Animashree, et al. "Tensor decompositions for learning latent variable models." *Journal of Machine Learning Research* 15 (2014): 2773-2832.

Clarkson, Kenneth L., et al. "The fast cauchy transform and faster robust linear regression." *SIAM Journal on Computing* 45.3 (2016): 763-810.

Cohen, Michael B., Cameron Musco, and Jakub Pachocki. "Online Row Sampling." *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques* (2016).

Cohen, Michael B., and Richard Peng. "Lp row sampling by lewis weights." *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 2015.

Cormode, Graham, Charlie Dickens, and David P. Woodruff. "Leveraging Well-Conditioned Bases: Streaming\& Distributed Summaries in Minkowski $ p $-Norms." *arXiv preprint arXiv:1807.02571* (2018).

References

Dasgupta, Anirban, et al. "Sampling algorithms and coresets for \ell_p regression." *SIAM Journal on Computing* 38.5 (2009): 2060-2078.

Har-Peled, Sariel, and Soham Mazumdar. "On coresets for k-means and k-median clustering." *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. 2004.

Feldman, Dan, and Michael Langberg. "A unified framework for approximating and clustering data." *Proceedings of the forty-third annual ACM symposium on Theory of computing*. 2011.

Woodruff, David P. "Sketching as a Tool for Numerical Linear Algebra." *Foundations and Trends® in Theoretical Computer Science* 10.1–2 (2014): 1-157.

Thank You