Dimensionality Reduction of Massive Sparse Datasets Using Coresets

Poster by Abzal Yessengazy and Jo Quinn   Work by Dan Feldman, Mikhail Volkov, Daniela Rus [1]

Introduction

Given a (large) dataset $\mathcal{D}$, and an algorithm $\mathcal{A}$ with $\mathcal{A}[\mathcal{D}]$ intractable, can we efficiently reduce $\mathcal{D}$ to $\mathcal{C}$ such that $\mathcal{A}[\mathcal{C}]$ is fast and $\mathcal{A}[\mathcal{C}] \approx \mathcal{A}[\mathcal{D}]$?

Often $\mathcal{A}$ is a **dimensionality reduction** algorithm. It is often desirable to reduce the dimension of the data, to improve on storage overheads, assist with visualisation, and help avoid the "curse of dimensionality". Some dimensionality reduction techniques include:

- Principal Components Analysis (PCA)
- Random Projection
- Low Rank Approximation

Background

Coresets come from *computational geometry*. A strict definition depends on the problem at hand; in short we want to approximate a large set of points with a **smaller weighted subset**.

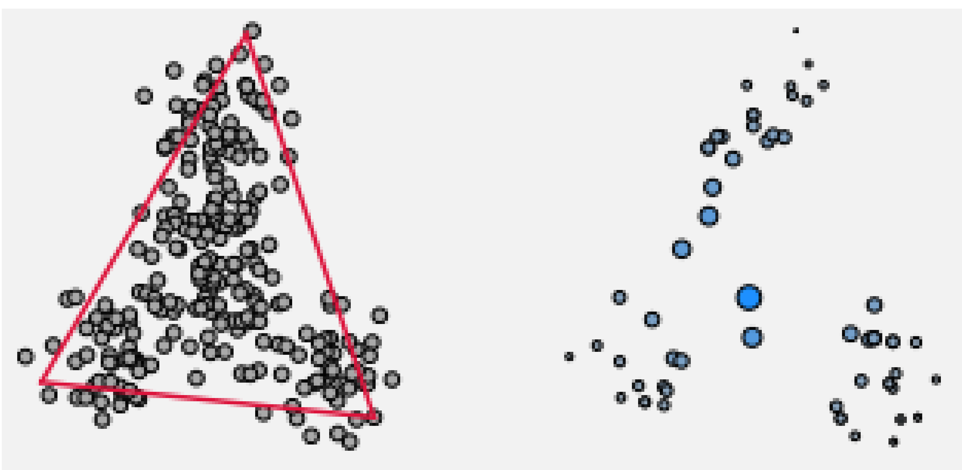
The paper [1] focuses on dimensionality reduction, and defines a coreset as follows:

$$C = \{w_i a_i | w_i > 0, i = 1, ..., n\}$$

such that for every k-subspace S of $\mathbb{R}^d$,

$$|\sum_i dist^2(a_i, S) - \sum_i dist^2(w_i a_i, S)|$$

$$\leq \epsilon \sum_i dist^2(a_i, S)$$



Left: full dataset, right: coreset [3].

We want to minimise the **cardinality**

$$|C| = \{w_i | w_i \neq 0, i = 1, ..., n\}.$$

Key Findings

The key findings are as follows:

1. **Proof** that one can always find an (ϵ, k)-coreset of size $O(k/\epsilon^2)$; not dependent on either n or d!.
2. **An algorithm** for computing such a coreset.
3. **Application** of these methods to two open problems: *low rank approximation* and *1-mean queries*.

Previous best method could find a coreset of size $O(dk^3/\epsilon^2)$ [2].

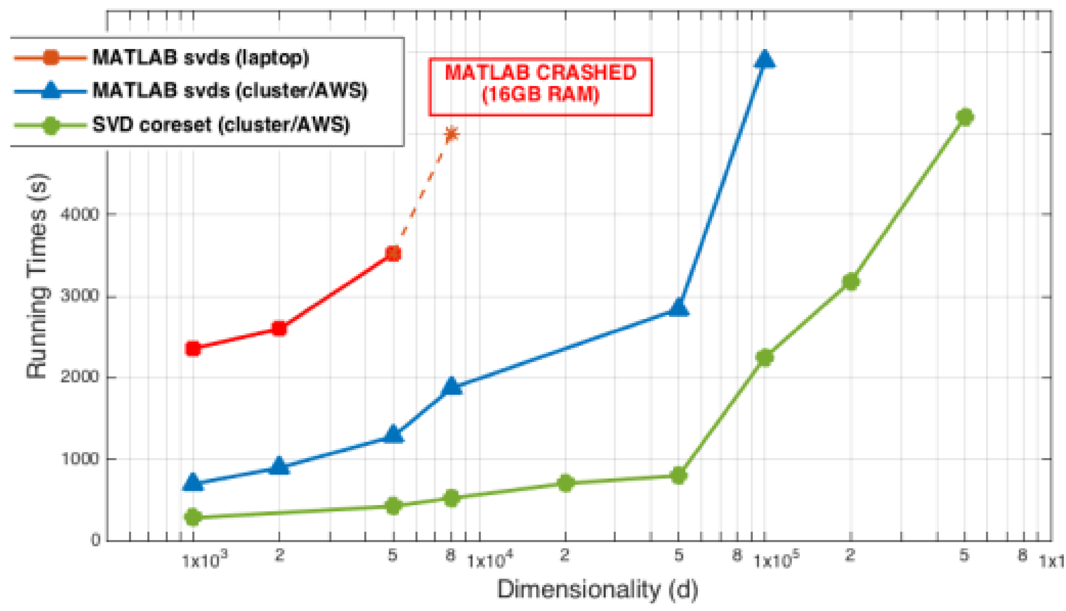
Example: Wikipedia

We can associate an **occurence matrix** with the **English Wikipedia**, documenting the number of occurrences of a word in each document:

	word1	word2	word3	word4...
doc1	0	0	1	0
doc2	0	0	0	0
doc3	1	0	0	0

This matrix is **fat** ($O(d) = O(n)$) and **sparse**.

- Want to do *Latent Semantic Analysis* (ie *SVD* with occurence matrices).
- Modern SVD implementations crash or take several days.
- Coreset algorithm doesn't depend on number of rows or columns
- Since coreset consists of **few weighted rows**, it tells us which records in are most important, and how much so!



Running time of SVD Coreset vs MATLAB SVD

Methodology

(ϵ, k)-coreset C is computed as follows:

1. Restructure input points using k-rank SVD decomposition UDV^T into new input matrix P.
2. Normalise P to get X
3. Arbitrarily select starting point $X_j = X_1$, and weight $w_j = 1$ (with all other w_i set to 0)
4. Compute farthest point from X_j by projecting the weighted points onto the current point.
5. α = ratio of distances from current point, current center, and new center, and weights are updated based on this new α.

The algorithm runs for k/ϵ^2 iterations, or until α converges to 1

Output: sparse vector of weights

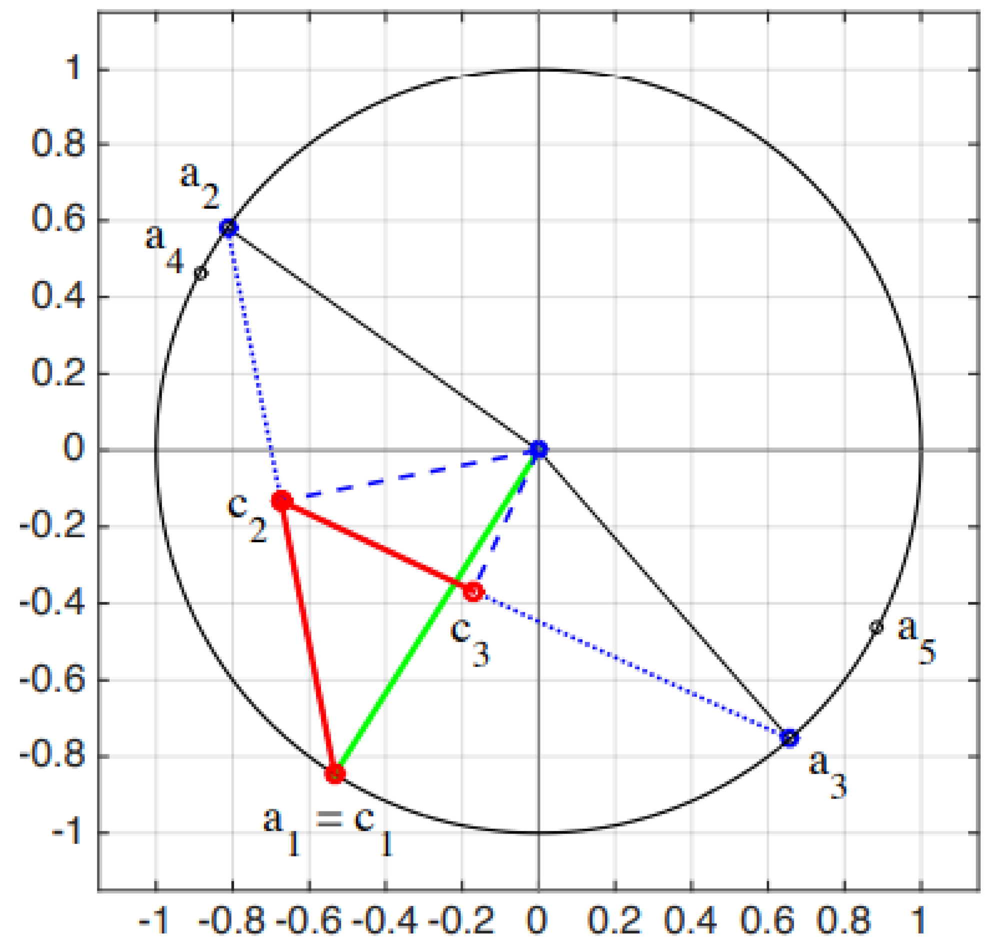
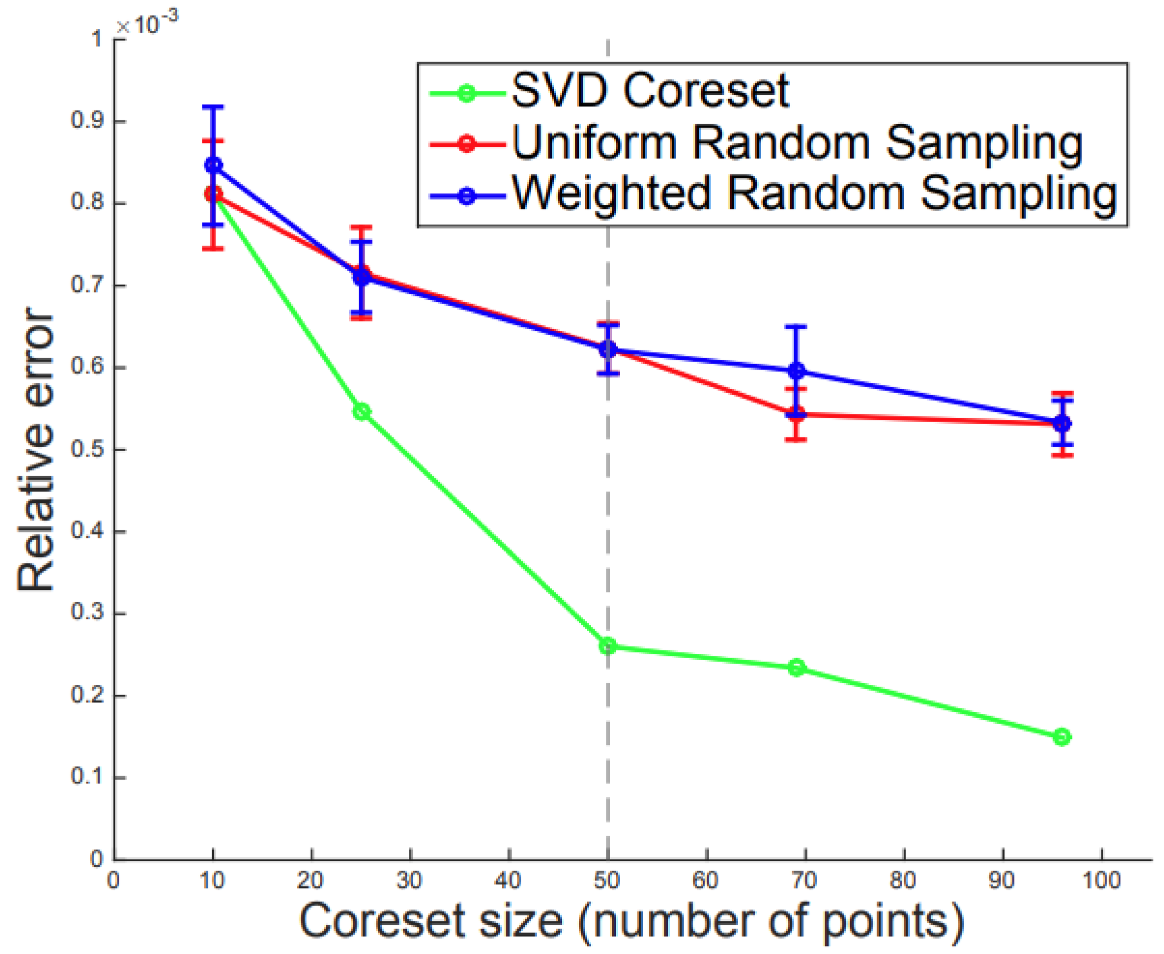


Illustration of steps 3 and 4

Conclusion



Comparison of sampling methods with k=50 coresets

Now have a way of finding coresets independent of size of the data. Applications include

- **Fast approximations**: efficiently compute low-rank approximations
- **Sparse data**: handle massive datasets with limited memory
- **Streaming parallel computation**: can compute C in *embarrassingly parallel* fashion
- **Interpretation**: weights w_i can tell us the most important data points

Coresets are ideal for modern dimensionality reduction challenges!

References

[1] Dan Feldman, Mikhail Volkov, and Daniela Rus. Dimensionality reduction of massive sparse datasets using coresets. In *Advances in Neural Information Processing Systems*, pages 2766–2774, 2016.

[2] Kasturi Varadarajan and Xin Xiao. On the sensitivity of shape fitting problems. *arXiv preprint arXiv:1209.4893*, 2012.

[3] Sebastian Mair and Ulf Brefeld. Coresets for archetypal analysis. In *Advances in Neural Information Processing Systems*, pages 7245–7253, 2019.