Sublinear Time Low-Rank Approximation of Distance Matrices

Ainesh Bakshi and David P. Woodruff

Carnegie Mellon University

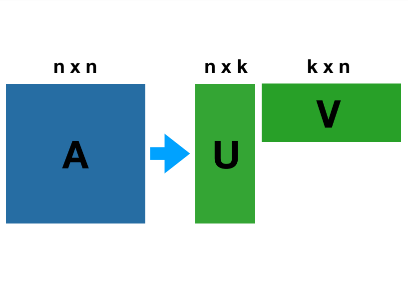
Motivation

- Data and Matrix compression
- De-noising and Dimensionality Reduction
- Applications to Clustering, Topic Modelling, Recommendation Systems and Distribution Learning.

Low-Rank Approximation

Given a $n \times n$ matrix A and an integer k, compute

$$A_k = \min_{\mathrm{rank}(X) \leq k} \|A - X\|_F$$



Distance Matrix

1. Let $\mathcal{P} = \{p_1, p_2, \ldots p_n\}$ be a set of n points in $\mathbb{R}^d$

Distance Matrix

1. Let $\mathcal{P} = \{p_1, p_2, \ldots p_n\}$ be a set of n points in $\mathbb{R}^d$
2. Let A be the resulting $n \times n$ pair-wise Distance Matrix, i.e.

$$\mathsf{A} = \begin{bmatrix} \|p_1 - p_1\| & \cdot & \cdot & \|p_1 - p_n\| \\ \cdot & \cdot & & \\ \cdot & & \cdot & \\ \|p_n - p_1\| & & & \|p_n - p_n\| \end{bmatrix}$$

Distance Matrix

1. Let $\mathcal{P} = \{p_1, p_2, \ldots p_n\}$ be a set of n points in $\mathbb{R}^d$
2. Let A be the resulting $n \times n$ pair-wise Distance Matrix, i.e.

$$\mathsf{A} = \begin{bmatrix} \|p_1 - p_1\| & \cdot & \cdot & \|p_1 - p_n\| \\ \cdot & \cdot & & \\ \cdot & & \cdot & \\ \|p_n - p_1\| & & & \|p_n - p_n\| \end{bmatrix}$$

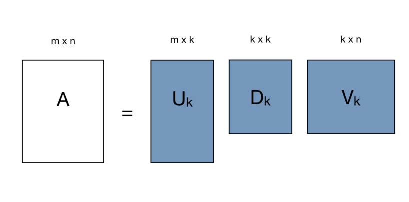
3. A is a dense matrix and has $O(n^2)$ non-zero entries

Singular Value Decomposition

1. Decompose $\mathbf{A}$ into $\mathbf{UDV}^T$ such that $\mathbf{U}$ has orthonormal columns, V has orthonormal rows and $\mathbf{D}$ is a diagonal matrix

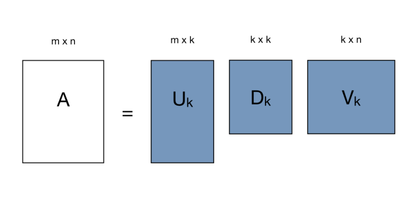
Singular Value Decomposition

1. Decompose **A** into **UDV**^{T} such that **U** has orthonormal columns, V has orthonormal rows and **D** is a diagonal matrix
2. Truncate D to it's top k entries



Singular Value Decomposition

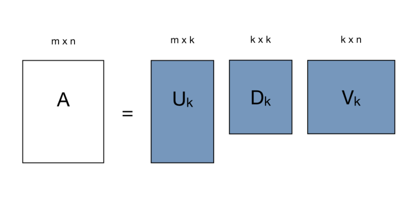
1. Decompose **A** into $\mathbf{UDV}^T$ such that **U** has orthonormal columns, V has orthonormal rows and **D** is a diagonal matrix
2. Truncate D to it's top k entries



3. Optimal!

Singular Value Decomposition

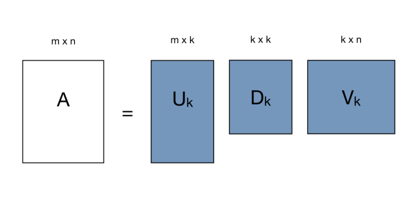
1. Decompose **A** into $\mathbf{UDV}^T$ such that **U** has orthonormal columns, V has orthonormal rows and **D** is a diagonal matrix
2. Truncate D to it's top k entries



3. Optimal!
4. Running time is $O(n^3)$

Singular Value Decomposition

1. Decompose **A** into $\mathbf{UDV}^T$ such that **U** has orthonormal columns, V has orthonormal rows and **D** is a diagonal matrix
2. Truncate D to it's top k entries



3. Optimal!
4. Running time is $O(n^3)$
5. Extremely slow for a large dataset

Input Sparsity Low Rank Approximation

Clarkson-Woodruff showed how to output a rank k matrix B such that

$$\|\mathsf{A} - \mathsf{B}\|_F^2 \leq (1+\epsilon) \min_{\mathrm{rank}(\mathsf{X}) \leq k} \|\mathsf{A} - \mathsf{X}\|_F^2$$

1. Running time is $O\left(n^2 + n \ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$

Input Sparsity Low Rank Approximation

Clarkson-Woodruff showed how to output a rank k matrix B such that

$$\|\mathsf{A}-\mathsf{B}\|_F^2 \leq (1+\epsilon) \min_{\mathrm{rank}(\mathsf{X})\leq k} \|\mathsf{A}-\mathsf{X}\|_F^2$$

1. Running time is $O\left(n^2 + n \ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$
2. Might still be too slow

Can we leverage the structure of a Distance Matrix to get faster algorithms?

Sublinear Low Rank Approximation

Theorem : Compute $\mathsf{U} \in \mathbb{R}^{n \times k}$, $\mathsf{V} \in \mathbb{R}^{k \times n}$ such that

$$\|\mathsf{A} - \mathsf{U}\mathsf{V}\|_F^2 \leq \min_{\mathrm{rank}(\mathsf{X}) \leq k} \|\mathsf{A} - \mathsf{X}\|_F^2 + \epsilon \|\mathsf{A}\|_F^2$$

in time $O\left(n^{1.001}\ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$

1. Does not read most of the input!

Sublinear Low Rank Approximation

Theorem : Compute $\mathsf{U} \in \mathbb{R}^{n \times k}$, $\mathsf{V} \in \mathbb{R}^{k \times n}$ such that

$$\|\mathsf{A} - \mathsf{UV}\|_F^2 \leq \min_{\mathrm{rank}(\mathsf{X}) \leq k} \|\mathsf{A} - \mathsf{X}\|_F^2 + \epsilon \|\mathsf{A}\|_F^2$$

in time $O\left(n^{1.001}\ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$

1. Does not read most of the input!
2. Only accesses $O\left(n^{1.001}\ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$ entries in A

Running Time Comparison

Algorithm	Running Time
Singular Value Decomposition	$O(n^3)$
Input Sparsity Low-Rank Approximation	$O\left(n^2 + n \ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$
Sublinear Low-Rank Approximation	$O\left(n^{1.001} \ \mathrm{poly}\left(\frac{k}{\epsilon}\right)\right)$

Experiments: Running Time

Algorithm	Clustering	MNIST
Singular Value Decomposition	398.76	398.50
Input Sparsity Low-Rank Approximation	8.94	34.32
Sublinear Low-Rank Approximation	1.69	4.16

Experiments: Absolute Error

Synthetic Clustering Dataset

MNIST Dataset

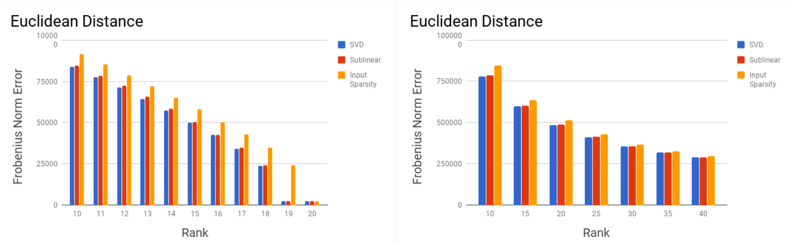


Figure 1: We plot $\|A - B\|_F$ on a synthetic dataset with 20 clusters and the MNIST dataset using ℓ_2 as the metric. We compare the error achieved by SVD (optimal), our Sublinear Algorithm and the Input Sparsity Algorithm.

Thank You!