

A Story of Principal Component Analysis in the Distributed Model

David Woodruff
IBM Almaden

Based on works with Christos Boutsidis, Ken Clarkson,
Ravi Kannan, Santosh Vempala, and Peilin Zhong

Talk Outline

1. What is low rank approximation?
2. How do we solve it offline?
3. How do we solve it in a distributed setting?

Low rank approximation

- A is an $n \times d$ matrix
 - Think of n points in \mathbb{R}^d
- E.g., A is a customer-product matrix
 - $A_{i,j}$ = how many times customer i purchased item j
- A is typically well-approximated by low rank matrix
 - E.g., high rank because of noise
- **Goal:** find a low rank matrix approximating A
 - Easy to store, data more interpretable

What is a good low rank approximation?

Singular Value Decomposition

Any matrix A can be approximated by a rank k matrix $A_k = \operatorname{argmin}_{\text{rank } k \text{ matrices } B} \|A - B\|_F$

- U

The rows of V_k are the top k **principal components**

- R

$$\begin{pmatrix} \mathbf{A} \end{pmatrix} = \begin{pmatrix} \mathbf{U}_k \end{pmatrix} \begin{pmatrix} \Sigma_k \end{pmatrix} \begin{pmatrix} \mathbf{V}_k \end{pmatrix} + \begin{pmatrix} \mathbf{E} \end{pmatrix}$$

Low rank approximation

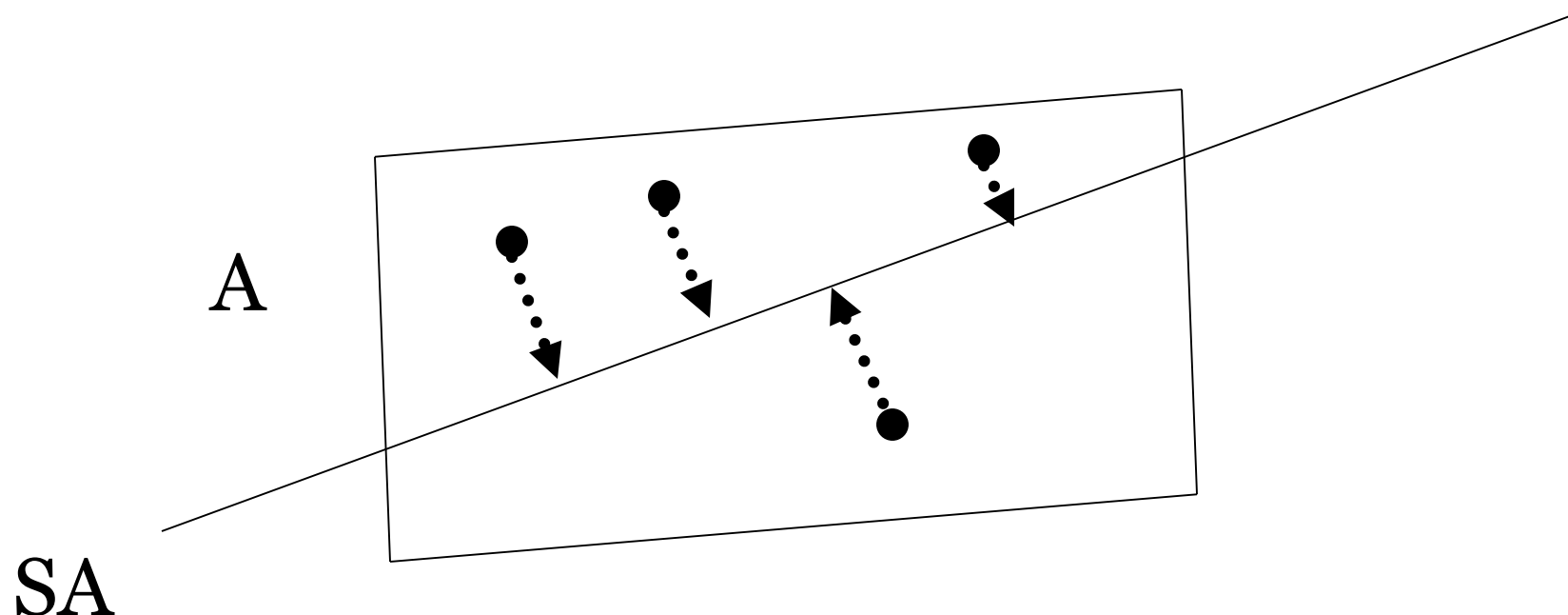
- **Goal:** output a rank k matrix A' , so that

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

- Can do this in $\text{nnz}(A) + (n+d) \cdot \text{poly}(k/\varepsilon)$ time [S,CW]
 - $\text{nnz}(A)$ is number of non-zero entries of A

Solution to low-rank approximation [S]

- Given $n \times d$ input matrix A
- Compute S^*A using a random matrix S with $k/\epsilon \ll n$ rows. S^*A takes random linear combinations of rows of A



- Project rows of A onto SA , then find best rank- k approximation to points inside of SA .

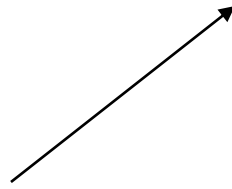
What is the matrix S ?

- S can be a $k/\epsilon \times n$ matrix of i.i.d. normal random variables
- [S] S can be a $k/\epsilon \times n$ Fast Johnson Lindenstrauss Matrix
 - Uses Fast Fourier Transform
- [CW] S can be a $\text{poly}(k/\epsilon) \times n$ CountSketch matrix

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$S \cdot A$ can be computed in $\text{nnz}(A)$ time

Caveat: projecting the points onto SA is slow

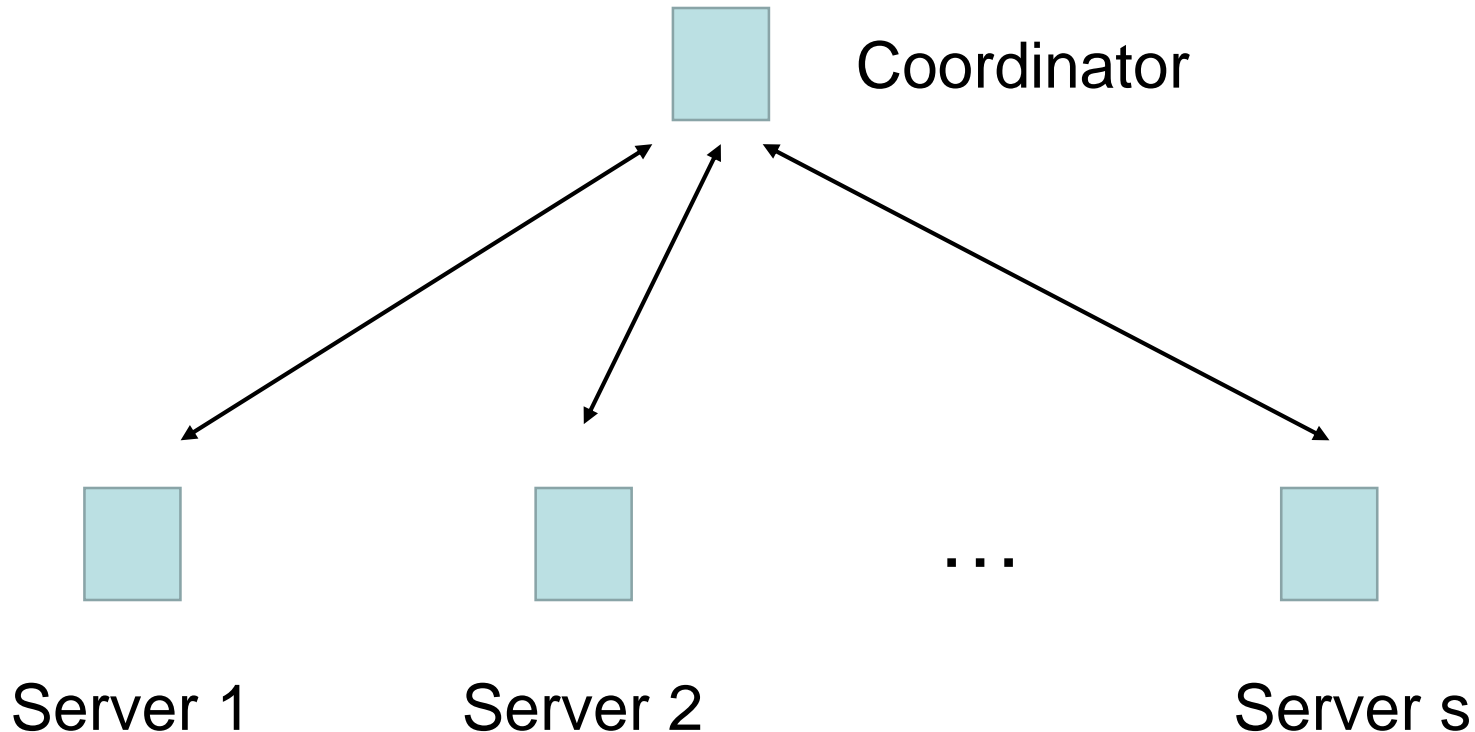
- Current algorithm:
 1. Compute S^*A
 2. Project each of the rows onto S^*A
 3. Find best rank-k approximation of projected points inside of rowspace of S^*A
- Bottleneck is step 2
- [CW] Approximate the projection
 - Fast algorithm for approximate regression
$$\min_{\text{rank-}k \ X} |X(SA)-A|_F^2$$

 - $\text{nnz}(A) + (n+d)*\text{poly}(k/\epsilon)$ time

$$\min_{\text{rank-}k \ X} |X(SA)R-AR|_F^2$$

Distributed low rank approximation

- *We have fast algorithms, but can they be made to work in a distributed setting?*
- Matrix A distributed among s servers
- For $t = 1, \dots, s$, we get a customer-product matrix from the t -th shop stored in server t . Server t 's matrix = A^t
- Customer-product matrix $A = A^1 + A^2 + \dots + A^s$
 - Model is called the **arbitrary partition model**
- More general than the **row-partition model** in which each customer shops in only one shop

The Communication Model



- Each player talks only to a Coordinator via 2-way communication
- Can simulate arbitrary point-to-point communication up to factor of 2 (and an additive $O(\log s)$ factor per message)

Communication cost of low rank approximation

- **Input:** $n \times d$ matrix A stored on s servers
 - Server t has $n \times d$ matrix A^t
 - $A = A^1 + A^2 + \dots + A^s$
 - Assume entries of A^t are $O(\log(nd))$ -bit integers
- **Output:** Each server outputs the same k -dimensional space W
 - $C = A^1 P_W + A^2 P_W + \dots + A^s P_W$, where P_W is the projector onto W
 - $|A-C|_F \leq (1+\epsilon)|A-A_k|_F$
 - Application: k -means clustering
- **Resources:** Minimize total communication and computation. Also want $O(1)$ rounds and input sparsity time

Work on Distributed Low Rank Approximation

- [FSS]: First protocol for the row-partition model.
 - $O(sdk/\epsilon)$ real numbers of communication
 - Don't analyze bit complexity (can be large)
 - SVD Running time, see also [BKLW]
- [KVW]: $O(skd/\epsilon)$ communication in arbitrary partition model
- [BWZ]: $O(skd) + \text{poly}(sk/\epsilon)$ words of communication in arbitrary partition model. Input sparsity time
 - Matching $\Omega(skd)$ words of communication lower bound
- Variants: kernel low rank approximation [BLSWX], low rank approximation of an implicit matrix [WZ], sparsity [BWZ]

Outline of Remainder of the Talk

- [FSS] protocol
- [KVW] protocol
- [BWZ] protocol

[FSS] Row-Partition Protocol

[KVW] protocol will handle 2, 3, and 4

Problems:

1. sdk/ϵ real numbers of communication
2. bit complexity can be large
3. running time for SVDs [BLKW]
4. doesn't work in arbitrary partition model

P^1

This is an SVD-based protocol. Maybe our random matrix techniques can improve communication just like they improved computation?

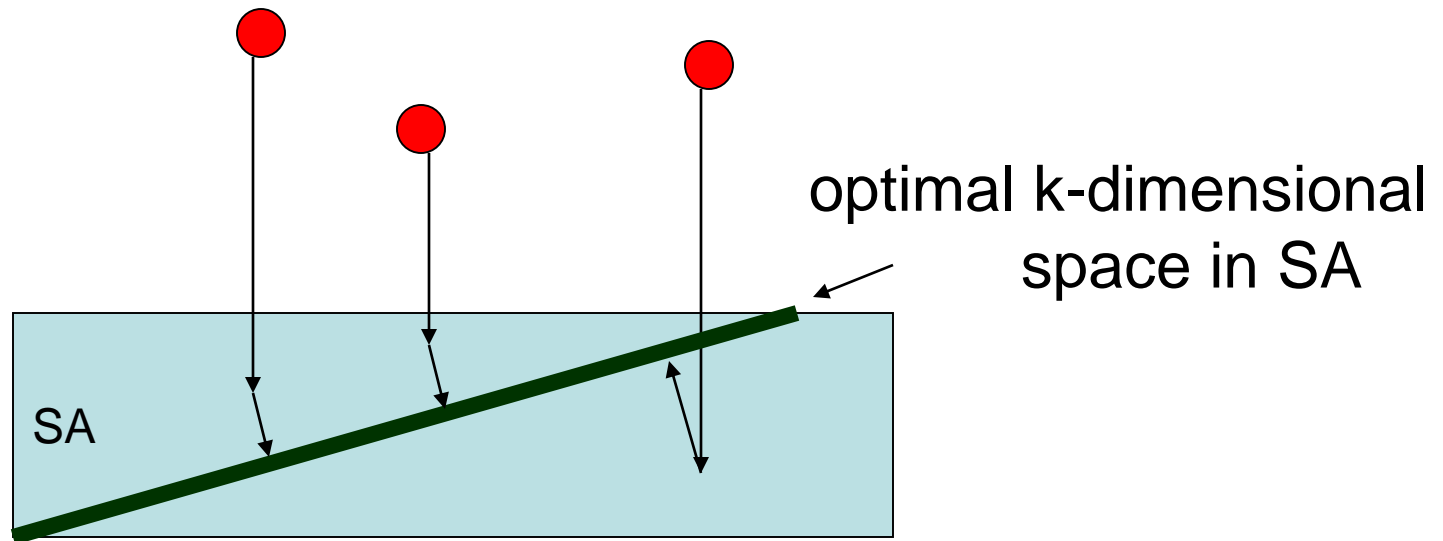
- Coordinates of P^1 are $(\sum^1 V^1; \sum^2 V^2; \dots; \sum^s V^s)$

[KVV] Arbitrary Partition Mod

- In
- **Fix:**
 - Instead of projecting A onto SA , project small number of random linear combinations of A onto SA
 - Find best k -dimensional space of random linear combinations inside of SA
 - Communication of this part is independent of n and d
- There are k vectors in SA . If we knew the projection of A^t onto it

[KVW] protocol

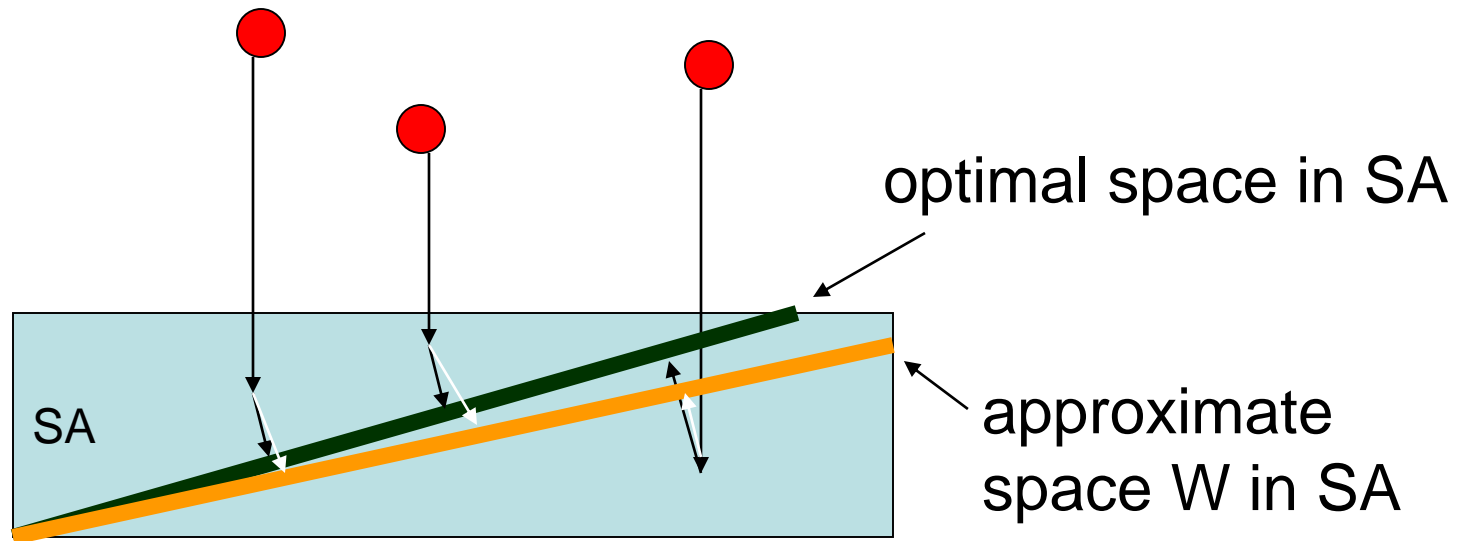
- Phase 1:
- Learn the row space of SA



$$\text{cost} \leq (1+\varepsilon)|A-A_k|_F$$

[KVW] protocol

- Phase 2:
- Communicate random linear combinations of points inside SA
- Find an approximately optimal space W inside of SA



$$\text{cost} \leq (1+\varepsilon)^2 |A - A_k|_F$$

[BWZ] Protocol

- Main Problem: communication is $O(\text{skd}/\epsilon) + \text{poly}(\text{sk}/\epsilon)$
- We want $O(\text{skd}) + \text{poly}(\text{sk}/\epsilon)$ communication!
- Idea: use **projection-cost preserving sketches** [CEMMP]
- Let A be an $n \times d$ matrix
- If S is a random $k/\epsilon^2 \times n$ matrix, then there is a constant $c \geq 0$ so that for all k -dimensional projection matrices P :
$$|SA(I - P)|_F + c = (1 \pm \epsilon)|A(I - P)|_F$$

[BWZ] Protocol

- Let S be a $k/\epsilon^2 \times n$ projection-cost preserving sketch
- Let T be a $d \times k/\epsilon^2$ projection
- Server t sends SA^t
 - Intuitively, U looks like top k left singular vectors of SA
- Coordinator sends back $SA^T = \sum_t SA^t$ to servers
- Thus, $U^T SA^T$ looks like top k right singular vectors of SA^T
 - Top k left singular
- Server t sends $U^T SA^t$
 - Top k right singular vectors of SA work because S is a projection-cost preserving sketch!
- Coordinator returns the space $U^T SA^T = \sum_t U^T SA^t$ to output

[BWZ] Analysis

- Let W be the row span of $U^T SA$, and P be the projection onto W
- Want to show $|A - AP|_F \leq (1 + \epsilon)|A - A_k|_F$
- Since T is a projection-cost preserving sketch,

$$(*) \quad |SA - SAP|_F \leq |SA - UU^T SA|_F \leq (1 + \epsilon)|SA - [SA]_k|_F$$

- Since S is a projection-cost preserving sketch, there is a scalar $c > 0$, so that for all k -dimensional projection matrices P ,

$$|SA - SAP|_F + c = (1 \pm \epsilon)|A - AP|_F$$

- Add c to both sides of $(*)$ to conclude $|A - AP|_F \leq (1 + \epsilon)|A - A_k|_F$

Conclusions

- [BWZ] Optimal $O(\text{sdk}) + \text{poly}(\text{sk}/\varepsilon)$ communication protocol for low rank approximation in arbitrary partition model
 - Handle bit complexity by adding Tao/Vu noise
 - Input sparsity time
 - 2 rounds, which is optimal [W]
 - Optimal data stream algorithms improves [CW, L, GP]
- Communication of other optimization problems?
 - Computing the rank of an $n \times n$ matrix over the reals
 - Linear Programming
 - Graph problems: Matching
 - etc.