IIT KHARAGPUR

IIT GANDHINAGAR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

# Scalable Data Science

## Lecture 15b:QB decomposition

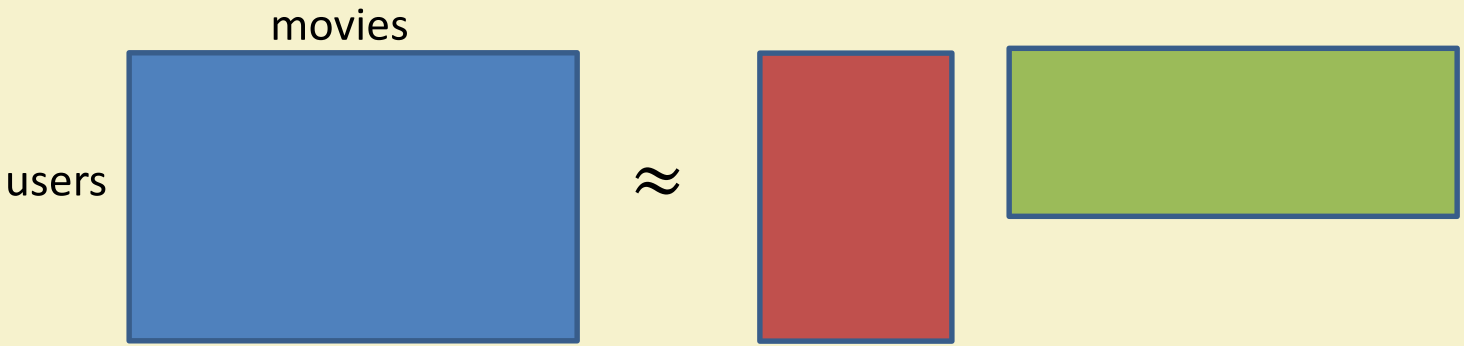
Anirban Dasgupta

Computer Science and Engineering

IIT GANDHINAGAR

# Application of dimension reduction

Recommendation system

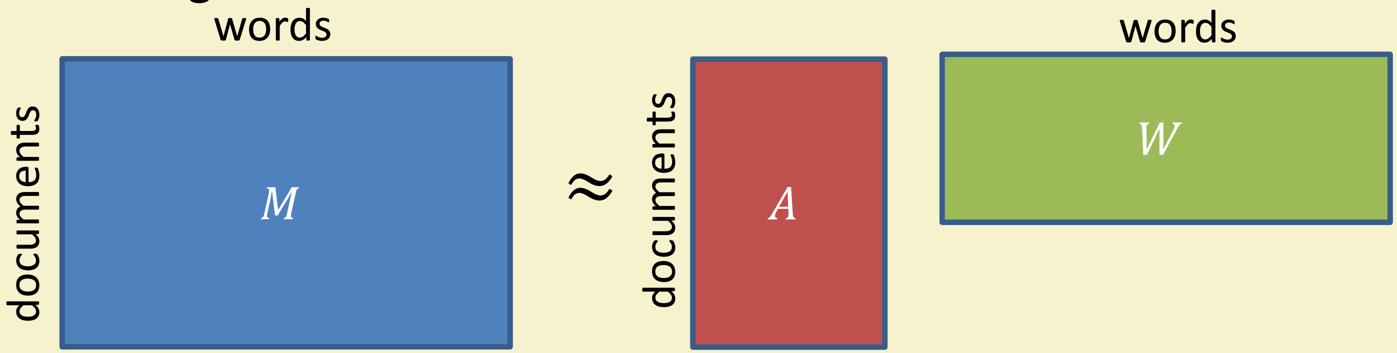



Minimize some loss function, e.g. $\sum \left(A_{ij} - X_{i*}Y_{*j}\right)^2$

Use $\sum_k X_{ik} \; Y_{kj}$ to predict missing entry $(i, j)$

# Application of dimension reduction

Topic modelling




Constraints e.g. $A$ and $W$ non-negative

# Singular Value Decomposition

- $A = U\Sigma V^t$
- Optimality properties e.g.

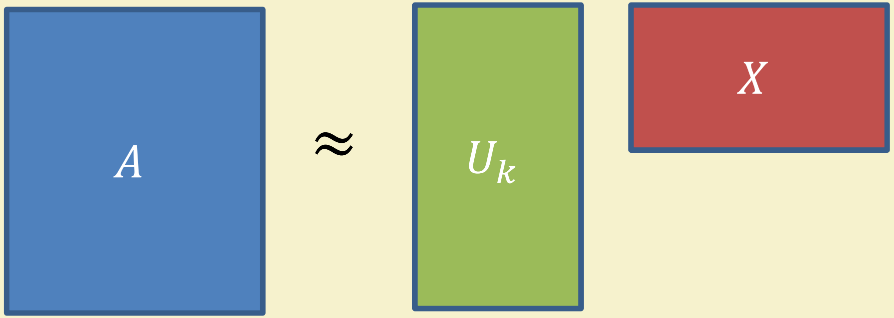
$$A_k = U_k \Sigma_k V_k^t = argmin_{X:rank(X)\leq k} \mid A - X \mid_F$$

$$A_k = U_k \Sigma_k V_k^t = argmin_{X:rank(X)\leq k} \mid A - X \mid_2$$

Issues: high complexity, negative $U, V$ etc.

Yet often forms the initialization for other matrix factorizations

# Alternate look at SVD



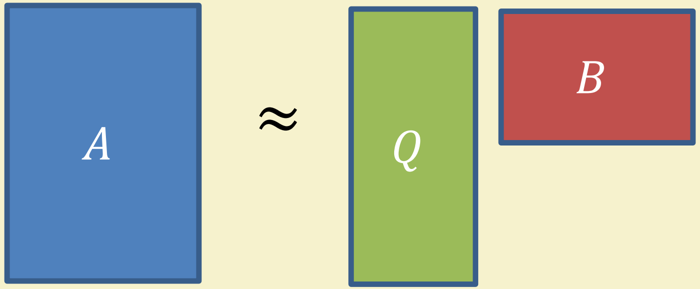

$$X = \Sigma_k V_k^t = U_k^t\, A$$

Can find a similar decomposition, but more efficiently?

Should retain (approximately) the optimality properties

Optimal Frob error = $\left|A - U_k\left(U_k^t A\right)\right|_F^2 = \sum_{j>k} \sigma_j^2$

# QB decomposition

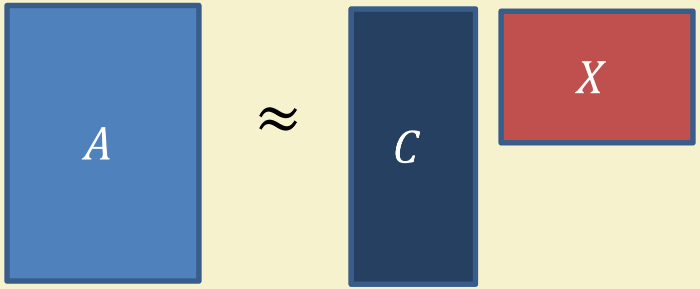



Can we find orthogonal $Q \in \Re^{n\times(k+p)}$ and $B$ such that

$|A \ - QB|_F \approx |A \ - A_k|_F$ and/or $|A \ - QB|_F \approx |A \ - A_k|_2$

Also want $k + p = O(k)$

# CX decomposition

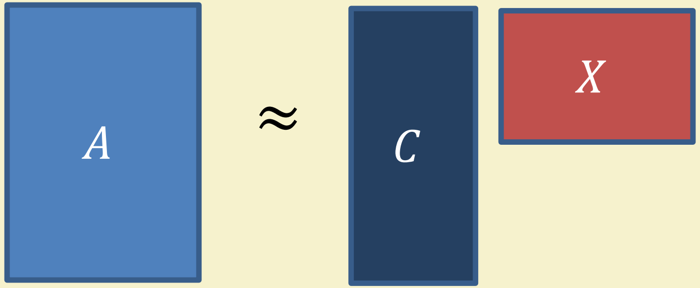



Can we find $C$, a subset of columns of $A$, such that

$$|A - CX|_F \approx |A - A_k|_F$$

and $|C| = O(k)$ ?

# CX decomposition




Can we find $C$, a subset of columns of $A$, such that

$$|A - CX|_F \approx |A - A_k|_F$$

and $|C| = O(k)$ ? It is not clear that this even exists!! Useful in ML applications when actual columns, and not their linear combinations, are needed.

# QB: prototype algorithm

Find $\Omega \in \Re^{d\times(k+p)}$ a random matrix

$Y = A\,\Omega$

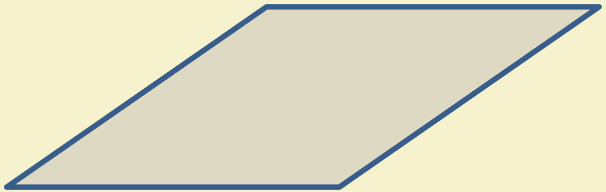
Find $Q$ = orthogonal basis for col$(Y)$

Return $(Q, Q^t A)$

# Intuition

Assume $rank(A) = k$

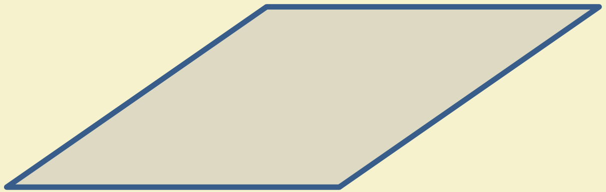
$\omega_1 \ldots \omega_k \in \Re^d$ be random vectors



colspace(A) = span($\{A_{*i}\}$)

$A\omega_1, A\omega_2, \ldots, A\omega_k$ are in general position in column space of $A$

# Intuition



colspace(A) = span($\{A_{*i}\}$)

Assume $rank(A) = k$

$\omega_1 \ldots \omega_k \in \Re^d$ be random vectors

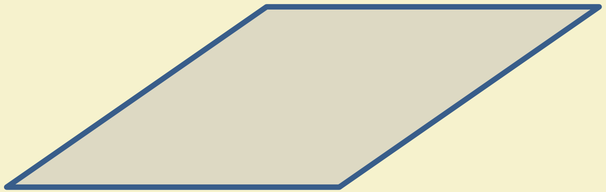
$A\omega_1, A\omega_2, \ldots, A\omega_k$ are in general position in column space of $A$

$Q$ = orthogonal basis of $[A\omega_1, .. A\omega_k] = A\Omega$ is also basis for colspace(A)

# Intuition

$A = A_k + E$ , $|E|$ small

$\omega_1 \ldots \omega_k \in \Re^d$ be random vectors



colspace(A) = span($\{A_{*i}\}$)

$A\omega_i = A_k\omega_i + E\omega_i$

The perturbation $E$ can take a vector out of the column-space $A$

But, if we take $k + p$ vectors, the chance that $\{A\omega_i\}$ does not span $\{A_{*i}\}$ is small

# Random matrix

$\Omega =$ any JL matrix e.g. $N(0,1)$ or $\pm 1$ or FJLT or Sparse JL

Theoretical bounds are the same, however these differ in numerical stability issues

Most practical implementations use $N(0,1)$

# Theoretical Guarantee

Claim: $E[|A - QB|_F] \leq \left(1 + \frac{k}{p-1}\right) |A - A_k|_F$

Bounds can also be obtained on the L2 norm error, few different variants of the bound

Qualitative take-away: Need to choose $p$ to be a small constant, in practice $\approx 5$

# Modified Algorithm

In reality, the bound $|A - QB|_F$ depends on the singular values $\{\sigma_{k+1}, \dots, \}$

Bound improves if $\sum_{i \le k} \sigma_i^2 \gg \sum_{i > k} \sigma_i^2$

# Power scheme

In reality, the bound $|A - QB|_F$ depends on the singular values $\{\sigma_{k+1}, \dots, \}$

Bound improves if $\sum_{i \le k} \sigma_i^2 \gg \sum_{i > k} \sigma_i^2$

To achieve this, we use $(AA^t)^P A = U\, \Sigma^{2P+1} V^t$

# Modified Algorithm [HMK10]

$$\text{function } [\mathbf{Q},\mathbf{B}] = \texttt{randQB\_p}(\mathbf{A},\ell,P)$$

$$\mathbf{\Omega} = \texttt{randn}(n,\ell).$$

$$\mathbf{Q} = \texttt{orth}(\mathbf{A\Omega}).$$

$$\text{for } j = 1:P$$

$$\quad \mathbf{Q} = \texttt{orth}(\mathbf{A}^*\mathbf{Q}).$$

$$\quad \mathbf{Q} = \texttt{orth}(\mathbf{AQ}).$$

$$\text{end for}$$

$$\mathbf{B} = \mathbf{Q}^*\mathbf{A}$$

Total runtime = $O(nd\ell + n\ell^2)$

$\ell = k + p \approx k + 5$

$P = 1\ or\ 2$ is sufficient

# References:

- Primary reference
  - Lecture notes on Randomized Numerical Linear Algebra by Petros Drineas and Michael Mahoney, https://arxiv.org/abs/1712.08880
  - Finding Structure with Randomness, Halko, Martinsson and Tropp, https://arxiv.org/pdf/0909.4061.pdf
  - A Practical Guide to Randomized Matrix Computations with MATLAB Implementations, Shushen Wang, https://arxiv.org/pdf/1505.07570.pdf

# Thank You!!