SEEM5020 Algorithms for Big Data
Dimension Reduction: PCA and SVD

Sibo WANG

Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Recap: Dimension Reduction

Definition 1 (Dimension reduction)

Given a set of points $X = \{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^d$ (where $d$ is the original high dimension), the goal of dimension reduction is to find a function $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ such that $k < d$, and each point $x_i$ is mapped to a point in $\mathbb{R}^k$ with preservation of certain properties (e.g., pairwise distances) in the lower-dimensional space.

Why do we need dimension reduction?

- Efficiency: Reducing the dimensionality can speed up algorithms without significantly compromising the quality of results.
- Curse of dimensionality: High-dimensional spaces can be counterintuitive; algorithms can behave poorly as dimension increases.
- Storage: Reducing the storage space.
- ...
Warm-up: Basic Concepts

Given a sequence of $S$ of real numbers $(r_1, r_2, \cdots, r_n)$, the mean and variance are as follows:

$$
\mu(S) = \frac{\sum_{i=1}^{n} r_i}{n}, \quad \text{Var}(S) = \frac{1}{n} \sum_{i=1}^{n} (r_i - \mu(S))^2
$$

Given a sequence $S$ of real numbers $r_1, r_2, \cdots, r_n$ and a sequence $S'$ of real numbers $r'_1, r'_2, \cdots, r'_n$, the covariance between sequence $S$ and $S'$ is:

$$
cov(S, S') = \frac{1}{n} \sum_{i=1}^{n} (r_i - \mu(S))(r'_i - \mu(S'))
$$

Example 1

Given two sequences of real numbers $S = (3, 3, 2, 8)$ and $S' = (4, 7, 11, 6)$. Then, $\mu(S) = 4, \mu(S') = 7$, $cov(S, S') = \frac{1}{4} ((3 - 4)(4 - 7) + (3 - 4)(7 - 7) + (2 - 4)(11 - 7) + (8 - 4)(6 - 7)) = -\frac{9}{4}$. 
Warm-up: Covariance Matrix

Covariance Matrix

For a set of $n$ $d$-dimensional points $X = \{x_1, x_2, \ldots, x_n\}$, the covariance matrix $A$ is a $d \times d$ matrix with $A_{i,j}$ representing the covariance between the $i$-th and $j$-th coordinate sequences of all $n$ points.

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension 1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Dimension 2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Dimension 3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table: A set of 3-dimensional data points

Example 2

Given the above table, we can compute that $A_{1,1} = \frac{2}{4}, A_{1,2} = -\frac{2}{4}, A_{1,3} = 0, A_{2,1} = -\frac{2}{4}, A_{2,2} = \frac{2}{4}, A_{2,3} = 0, A_{3,1} = 0, A_{3,2} = 0, A_{3,3} = 0$. 

Sibo WANG (CUHK)
Let $A$ be a $d \times d$ matrix. If for a $d$-dimensional unit vector $v$, it holds that:

$$A \cdot v = \lambda \cdot v$$

Then $v$ is called a *unit eigenvector* of $A$, and $\lambda$ is called the *eigenvalue* of $A$.

Several popular solutions for calculating eigenvectors and eigenvalues.

- **Power-method**
  - Derive the largest eigenvalue and the corresponding eigenvector
  - After deriving the largest, we can further derive the second largest, third largest, etc.

**Lemma 1**

*The Power-Method, when executed sequentially, yields the eigenvalues of a matrix in descending order of magnitude, along with their respective eigenvectors.*
The Power-method works as follows:

- Make an initial (non-zero) guess for the vector, say \( u_0 \)
- Repeat \( u_k = \frac{A \cdot u_{k-1}}{\|A \cdot u_{k-1}\|_2} \) until \( \|u_k - u_{k-1}\|_2 \) is smaller than some pre-defined small constant, say 0.0000001
- Denote this vector as \( v_1 \)
- Derive the eigenvalue: \( \lambda_i = v_1^T A v_1 \)
- Update the matrix as \( A = A - \lambda_1 v_1 v_1^T \) and the repeat the above process to computer \( \langle v_2, \lambda_2 \rangle, \langle v_3, \lambda_3 \rangle, \cdots \)

\[
A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}, \quad u_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad A \cdot u_0 = \begin{bmatrix} 5 \\ 8 \end{bmatrix} \Rightarrow u_1 = \frac{1}{\sqrt{5^2+8^2}} \begin{bmatrix} 5 \\ 8 \end{bmatrix} = \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix}
\]

\[
\cdots \Rightarrow v_1 = \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix} \Rightarrow \lambda_1 = v_1^T A v_1 = 6.993
\]

\[
A = A - \lambda_1 v_1 v_1^T = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} - 6.993 \cdot \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix} \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix}
\]
Eigen-Decomposition

Given a symmetric square matrix $A \in \mathbb{R}^{n \times n}$, let $v_1, v_2, \cdots, v_n$ be the $n$ eigenvectors, associated with the eigen values $\lambda_1, \lambda_2, \cdots, \lambda_n$, the matrix $A$ can be decomposed as:

$$A = V \Lambda V^T,$$

where $V = [v_1, v_2, \cdots, v_n]$ and $\Lambda$ is the diagonal matrix where $\Lambda_{i,i} = \lambda_i$.

- Important property of $A^k$: the eigenvectors of $A^k$ is the same as $A$ and eigenvalues of $A^k$ are $\lambda_1^k, \lambda_2^k, \cdots, \lambda_n^k$.

$$A^k = (V \Lambda V^T)^k = V \Lambda^k V^T$$
Table of Contents

1 Principal Component Analysis

2 Singular Value Decomposition and Low-Rank Approximation
Preserving the Variation of the Data

**Goal of PCA (Principal Component Analysis):** Transform the dataset into a new set of orthogonal axes, spanning $k$ dimensions, in a manner that maximizes the retained variance or spread of the data points.

![Diagram showing PCA transformation](image)

**Figure:** Convert 2d points into 1d points by projecting into a line.
Principal Component Analysis

- Step 1: Make each dimension have zero mean (important). Subtract the mean of each dimension. Let $\mu_i$ be the mean of the values in the $i$-th dimension for all $n$ data points. Update each data point $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,d})$ as $(x_{i,1} - \mu_1, x_{i,2} - \mu_2, \cdots, x_{i,d} - \mu_d)$.

- Step 2: Compute the covariance matrix $\mathbf{A}$ for the points after adjustment.

- Step 3: Find the eigenvectors and eigenvalues of $\mathbf{A}$ and sort the eigenvectors in descending order of their eigenvalues.

- Step 4: Keep the $k$ eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k$ whose corresponding eigenvalues are the top-$k$ highest.

- Step 5: For each data point $\mathbf{x}_i$, map it into a $k$-dimensional point using these $k$ eigenvectors: $(\mathbf{v}_1 \cdot \mathbf{x}_i, \mathbf{v}_2 \cdot \mathbf{x}_i, \cdots, \mathbf{v}_k \cdot \mathbf{x}_i)$. 
Property of PCA

The dot-product $\mathbf{a} \cdot \mathbf{b}$ is essentially the projection of a vector $\mathbf{a}$ to direction of unit vector $\mathbf{b}$.

Projecting the $n$ points according to the direction of the $i$-th eigenvector, it keeps the $i$-th largest variance.

- $\mathbf{v}_1$ is the direction along which the projections of $\mathbf{X}$ have the largest variance. $\mathbf{v}_2$ is the direction with the second largest variance, etc.
- We skip the last $d - k$ dimensions that has the smallest variances.
PCA Quality: The Variation of the Projection

Intuition: The larger the variation of the projection it is, the more variation information among different data points is kept, and the better the projection it is. Goal:

$$\max_{||u||_2=1} \text{Var}(S) \quad \text{where} \quad S = (x_1 \cdot u, x_2 \cdot u, \cdots, x_n \cdot u)$$
Theorem 1

The eigenvector associated with the largest eigenvalue gains the highest variance.

Proof.

- Fact 1: Let \( \mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \) after making zero mean for each dimension. Then, it satisfies that \( \frac{1}{n} \mathbf{X} \cdot \mathbf{X}^T = \mathbf{A} \), where \( \mathbf{A} \) is the covariance matrix.

- Fact 2: After projecting to the direction of a unit vector \( \mathbf{u} \), the mean is still zero. Let \( \mathbf{S} = (\mathbf{x}_1 \cdot \mathbf{u}, \mathbf{x}_2 \cdot \mathbf{u}, \cdots, \mathbf{x}_n \cdot \mathbf{u}) \). Then,

\[
\text{Var}(\mathbf{S}) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i \cdot \mathbf{u} - 0)^2
\]
Proof (Cont.)

We rewrite $\text{Var}(S)$ with the following transformation:

$$
\text{Var}(S) = \frac{1}{n} \sum_{i=1}^{n} (x_i \cdot u)^2 = (u^T X)(u^T X)^T
$$

$$
= \frac{1}{n} u^T X X^T u = u^T Au
$$

The goal becomes:

$$
\max_u u^T Au
$$

subjective to:

$$
u^T u = 1
$$
Theorem 2 (Lagrange Multiplier Method)

Let \( f : \mathbb{R}^n \to \mathbb{R} \) be a function to be optimized. Let \( g : \mathbb{R}^n \to \mathbb{R} \) be a constraint such that \( g(x) = 0 \). Define the Lagrangian function \( L \) as
\[
L(x, \lambda) = f(x) - \lambda g(x).
\]
Then, the points that satisfy the following system of equations correspond to the extrema of \( f \) subject to the constraint \( g(x) = 0 \):
\[
\frac{\partial L}{\partial x} = 0, \quad \frac{\partial L}{\partial \lambda} = g(x) = 0.
\]
Here, \( \lambda \) is known as the Lagrange multiplier.

Remark: (i) We can extend to multiple input variables. Simply take the partial derivative for all variables and make them zero.

(ii) The Lagrange multiplier method can be further extended to support multiple constraints \( g_1(x), g_2(x), \ldots, g_k(x) \). The Lagrangian function \( L \) is then defined as
\[
L(x, \lambda_1, \lambda_2, \ldots, \lambda_k) = f(x) - \lambda_1 g_1(x) - \lambda_2 g_2(x) - \ldots - \lambda_k g_k(x)
\]
Example of Lagrange Multiplier Method

Example 3

Maximize $f(x, y) = x + y$ subject to constraint $g(x, y) = x^2 + y^2 - 1 = 0$.

Solution using the Lagrange Multiplier Method:

1. Define the function and the constraint:
   
   $$f(x, y) = x + y, \quad g(x, y) = x^2 + y^2 - 1$$

2. Define the Lagrangian:
   
   $$L(x, y, \lambda) = f(x, y) - \lambda g(x, y) = x + y - \lambda (x^2 + y^2 - 1)$$

3. Compute partial derivatives of $L$ with respect to $x$, $y$, and $\lambda$, and set them to zero:
   
   $$\frac{\partial L}{\partial x} = 1 - 2\lambda x = 0, \quad \frac{\partial L}{\partial y} = 1 - 2\lambda y = 0, \quad \frac{\partial L}{\partial \lambda} = -(x^2 + y^2 - 1) = 0$$

   Solving the above equations, we derive: $x = \frac{1}{\sqrt{2}}$, $y = \frac{1}{\sqrt{2}}$ to gain the maximum value $f(x, y) = \sqrt{2}$. 
Apply Lagrange multipliers to find the maximum. Introduce a real value $\lambda$, and consider the Lagrangian function:

$$L(u, \lambda) = u^T A u - \lambda (u^T u - 1)$$

Take the partial derivative of $L(u, \lambda)$ with respect to $u$ and set it to zero:

$$\frac{\partial L}{\partial u} = 2Au - 2\lambda u = 0$$

It means when $u$ is an eigenvector of $A$, we have the maximum value, and $\lambda$ is its corresponding eigenvalue. Further multiple with $u^T$, and note that $u^T u = 1$:

$$2u^T Au - 2\lambda u^T u = 0 \Rightarrow u^T Au = \lambda.$$

That means, our optimization goal is: $u^T Au = \lambda$. To make it maximum, $\lambda$ should be the largest eigenvalue. This finishes the proof.
We can further extend to \( k \) dimensions.

**Theorem 3**

Let \( \{ \lambda_i \}_{i=1}^k \) be the top-\( k \) highest eigenvalues and let \( \{ v_i \}_{i=1}^k \) be their associated eigenvectors. If \( n \) data points are projected onto the subspace spanned by these \( k \) eigenvectors, the total variance retained on these \( k \) dimensions is maximized and is given by:

\[
\sum_{i=1}^{k} \lambda_i
\]

The proof is left as a self-exercise.
Time Complexity of PCA

Given a matrix of size $d \times d$, the eigendecomposition has a cost of $O(d^3)$ [1]. If the input vector $\mathbf{x}_i$ has a large dimension, e.g., $O(n)$, the running cost is $O(n^3)$, which is too expensive. Here are some such examples:

- The Laplacian matrix of undirected graphs.
- The proximity matrix of each vertex or each data point.
Table of Contents

1 Principal Component Analysis

2 Singular Value Decomposition and Low-Rank Approximation
Theorem 4

For any matrix $A_{m \times n}$ with rank $r$, it can be decomposed as:

$$A_{m \times n} = U_{m \times r} \Sigma_{r \times r} (V_{n \times r})^T,$$

where $U$, $\Sigma$, $V$ are unique and satisfies:

1. $U = [u_1, u_2, \cdots, u_r]$ is an $m \times r$ orthonormal matrix ($UU^T = I$), and $u_1, u_2, \cdots, u_r$ are called left singular vectors.

2. $\Sigma$ is a $r \times r$ diagonal matrix where the $(i, i)$-th entry is $\sigma_i$, and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are called the singular values.

3. $V = [v_1, v_2, \cdots, v_r]$ is a $n \times r$ orthonormal matrix ($VV^T = I$), and $v_1, v_2, \cdots, v_r$ are called the right singular vectors.

```
  m
A  =  m
  n
```

$U$, $\Sigma$, $V$ are unique and satisfies:

1. $U = [u_1, u_2, \cdots, u_r]$ is an $m \times r$ orthonormal matrix ($UU^T = I$), and $u_1, u_2, \cdots, u_r$ are called left singular vectors.

2. $\Sigma$ is a $r \times r$ diagonal matrix where the $(i, i)$-th entry is $\sigma_i$, and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are called the singular values.

3. $V = [v_1, v_2, \cdots, v_r]$ is a $n \times r$ orthonormal matrix ($VV^T = I$), and $v_1, v_2, \cdots, v_r$ are called the right singular vectors.
Alternatively, we can rewrite \( A_{m \times n} = U_{m \times r} \Sigma_{r \times r} (V_{n \times r})^T \) as:

\[
A_{m \times n} = \sum_{i=1}^{r} \sigma_i u_i v_i^T.
\]

Intuition of low-rank approximation with SVD: Since \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \), even if we drop the terms \( \sigma_i u_i v_i \) for \( i \) from \( k+1 \) to \( r \), when \( \sigma_{k+1}^2 + \sigma_{k+2}^2 \cdots \sigma_r^2 \) is sufficiently small. It will not significantly affects the original matrix \( A \).
Left Singular Vectors and Singular Values

Consider the matrix $A A^T$:

1. Note that $A$ is $m \times m$, $A A^T$ is hence $m \times m$ and it is a square symmetric matrix. Let the rank of $A$ be $r$. The rank of $A A^T$ is still $r$.
2. Via the SVD property, we can write $A$ as $U \Sigma V^T$. Then:

$$A A^T = U \Sigma V^T (U \Sigma V^T)^T = U \Sigma V^T V \Sigma U^T = U \Sigma^2 U^T$$

3. Recap that with eigen-decomposition, we can decompose $A A^T$ as

$$A A^T = W_{[m \times r]} \Lambda_{[r \times r]} (W_{[m \times r]})^T,$$

where $W = [w_1, w_2, \cdots, w_r]$ is the set of eigenvectors in descending order of the eigenvalues and $\Lambda$ is a diagonal matrix where $\Lambda_{i,i}$ is the $i$-th largest eigenvalue of $A A^T$. Then, the singular value $\sigma_i = \sqrt{\lambda_i}$ and we can set:

$$U = W, \Sigma = \sqrt{\Lambda}.$$
Right Singular Vectors

Similarly, by considering matrix $A^T A$, we can prove that:

- By taking $V = W' = [w'_1, w'_2, \cdots, w'_r]$, where $w'_i$ ($1 \leq i \leq r$) are the $r$ eigenvectors of $A^T A$ in descending order of the eigenvalues, we have the matrix for the right singular vectors. Alternatively, we can directly compute $V$ by:

$$A = U \Sigma V^T \Rightarrow V = A^T U \Sigma^{-1}$$

Lemma 2

*If $\lambda \neq 0$ is an eigenvalue of $AA^T$, then, it is also an eigenvalue of $A^T A$."

Proof.

Let $v$ be the eigenvector of $AA^T$ corresponding to the eigenvalue $\lambda$. Then, we have:

$$AA^T v = \lambda v \Rightarrow A^T AA^T v = \lambda A^T v$$

Define $A^T v$ as $u$. We get $A^T Au = \lambda u$. \qed

Sibo WANG (CUHK)
SEEM5020 Algorithms for Big Data
Fall 2023
Problem 1

Given a matrix $A_{m \times n}$, and an integer $k$, find a matrix $B$ of rank at most $k$ such that $||A - B||$ is minimized.

When $|| \cdot ||$ is the Frobenius norm, SVD can easily achieve the optimal solution. By keeping $U_k$ (the first-$k$ columns of $U$), $\Sigma_k$ (the top $k$ row and columns for $\Sigma$), and $V_k$ (the top-$k$ columns of $V$).

Theorem 5 (Truncated SVD)

Define $A_k = U_k \Sigma_k V_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$. Then, we have that $A_k$ is the best $k$-rank approximation to $A$:

$$||A - A_k||_F = \min_{B: \text{rank}(B \leq k)} ||A - B||_F = \sum_{i=k+1}^{r} \sigma_i^2.$$ 

$||A - A_k||_F$ is also called the reconstruction error.
Analysis of Truncated SVD

We first prove for $k = 1$. For matrix $A_{m \times n}$, let $a_i \in \mathbb{R}^{1 \times n}$ be the $i$-th row of $A$. Since the rank is 1, the matrix $B$ can be represented as $B = x_{[m \times 1]}(y_{[n \times 1]})^T$. Without loss of generality, we assume $y$ is a unit vector. Otherwise, we can scale $x$ to make $y$ become a unit vector.

\[
\|A - B\|_F^2 = \|A - xy^T\|_F^2 = \sum_{i=1}^{n} \|a_i - x_i y\|_2^2
\]

Given the $i$-th row, if $y$ is fixed, how to set $x_i$ to minimize $\|a_i - x_i y\|_2^2$?

- As shown in the figure below we should set $x_i = a_i \cdot y$ so that the distance from $a_i$ to the line following the direction of $y$ is minimized.
The goal is to find the unit vector $\mathbf{y}$ such that the following is minimized:

$$\sum_{i=1}^{m} \|\mathbf{a}_i - (\mathbf{a}_i \cdot \mathbf{y})\mathbf{y}\|_2^2$$

Further consider the $i$-th row.

$$\|\mathbf{a}_i - (\mathbf{a}_i \cdot \mathbf{y})\mathbf{y}\|_2^2 = \sum_{i=1}^{m} (\mathbf{a}_i - (\mathbf{a}_i \cdot \mathbf{y})\mathbf{y}) \cdot (\mathbf{a}_i - (\mathbf{a}_i \cdot \mathbf{y})\mathbf{y})$$

$$= \|\mathbf{a}_i\|_2^2 - 2(\mathbf{a}_i \cdot \mathbf{y})\mathbf{a}_i \cdot \mathbf{y} + (\mathbf{a}_i \cdot \mathbf{y})^2 = \|\mathbf{a}_i\|_2^2 - (\mathbf{a}_i \cdot \mathbf{y})^2$$

Then, summing all $m$ rows, we have:

$$\sum_{i=1}^{m} \|\mathbf{a}_i\|_2^2 - (\mathbf{a}_i \cdot \mathbf{y})^2 = \|\mathbf{A}\|_F^2 - \sum_{i=1}^{m} (\mathbf{a}_i \cdot \mathbf{y})^2 = \|\mathbf{A}\|_F^2 - \|\mathbf{A}\mathbf{y}\|_2^2$$

Note that $\|\mathbf{A}\|_F^2 = \sum_{i=1}^{r} \sigma_i^2$. The proof is left as self-exercise.
Goal: Maximize $||Ay||^2_2$ subject to the constraint that $y$ is a unit vector.

Define $L(u, \lambda) = ||Ay||^2_2 - \lambda(y^Ty - 1)$. Equivalently, we have:

$$L(u, \lambda) = (Ay)^T(Ay) - \lambda(y^Ty - 1) = y^TA^TAy - \lambda(y^Ty - 1).$$

Take the partial derivative with respect to $y$ and set it to zero.

$$\frac{\partial L}{\partial y} = 2A^TAy - 2\lambda y = 0$$

Then, $y$ must be a eigenvector of $A^TA$. Also

$$y^TA^TAy = y^T\lambda y = \lambda = \sigma_1^2.$$ 

By choosing the maximum eigen vector of $A^TA$, we achieve the minimum reconstruction error for $k = 1$. The error is $\sum_{i=2}^r \sigma_i^2$. 

We can generalize to $k \leq r$. Define the $k$-rank matrix $B$ as

$$B = \sum_{i=1}^{k} x_j y_j^T,$$

assuming that $y_1, y_2, \cdots, y_k$ are unit vectors and are orthogonal to each other. Please verify by yourself that the final optimization goal will become:

$$\min \|A\|_F^2 - \sum_{j=1}^{k} \|Ay_j\|.$$

When $y_i$ is the eigenvector of $A^T A$ with the $i$-th largest eigenvalue, we achieve the minimum reconstruction error, which is $\sum_{i=k+1}^{r} \sigma_i^2$. 
Time complexity of SVD

- Compute the matrix $\mathbf{A}^T \mathbf{A}$.
  - Time complexity: $O\left(\min(m^2 n, mn^2)\right)$.
- Derive the eigenvectors and eigenvalues of $\mathbf{A}^T \mathbf{A}$:
  - Time complexity: $O(\min\{n^3, m^3\})$.
- Total cost: $O\left(\min(m^2 n, mn^2)\right)$.
Victor Y. Pan and Zhao Q. Chen.
The complexity of the matrix eigenproblem.