CMSC5741 Big Data Tech. & Apps.

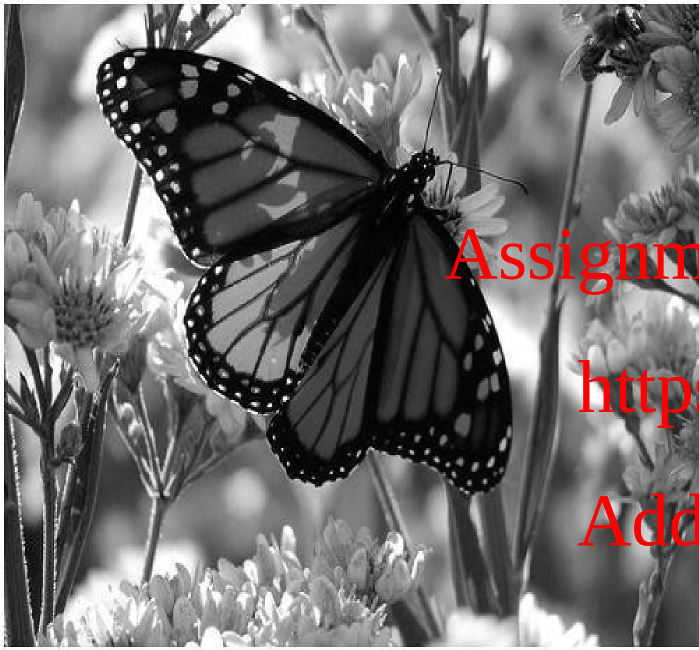
Lecture 6: Dimensionality Reduction

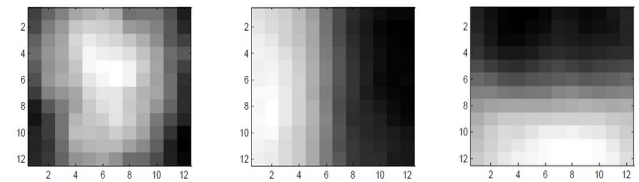
Prof. Michael R. Lyu

Computer Science & Engineering Dept.

The Chinese University of Hong Kong

A Compression Example

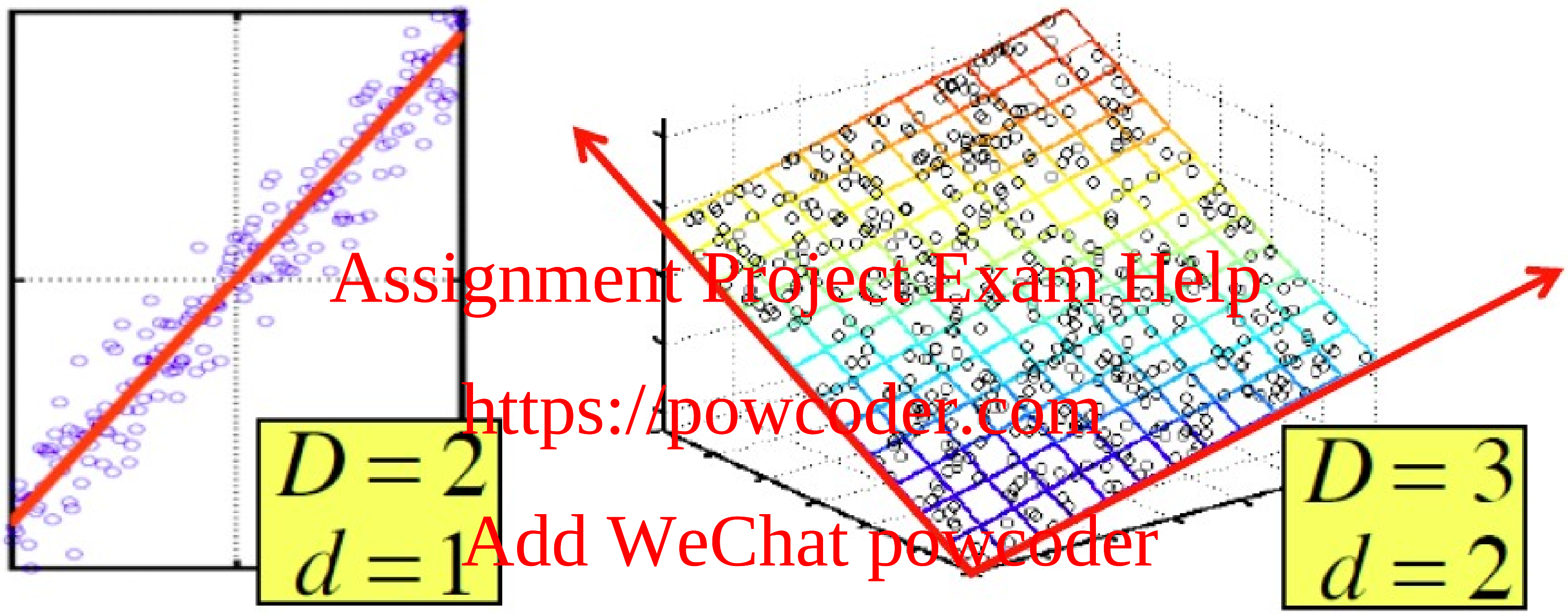




Outline

- Motivation
- SVD
- CUR
  - Application of SVD and CUR
- PCA
  - Extension to robust PCA

Dimensionality Reduction Motivation



- **Assumption:** Data lie on or near a low *d*-dimensional subspace
- **Axes of this subspace are effective representation of the data**

Dimensionality Reduction Motivation

- **Compress / reduce dimensionality:**
  - 10^6 rows; 10^3 columns; no updates
  - Random access to any cell(s); **small error: OK**

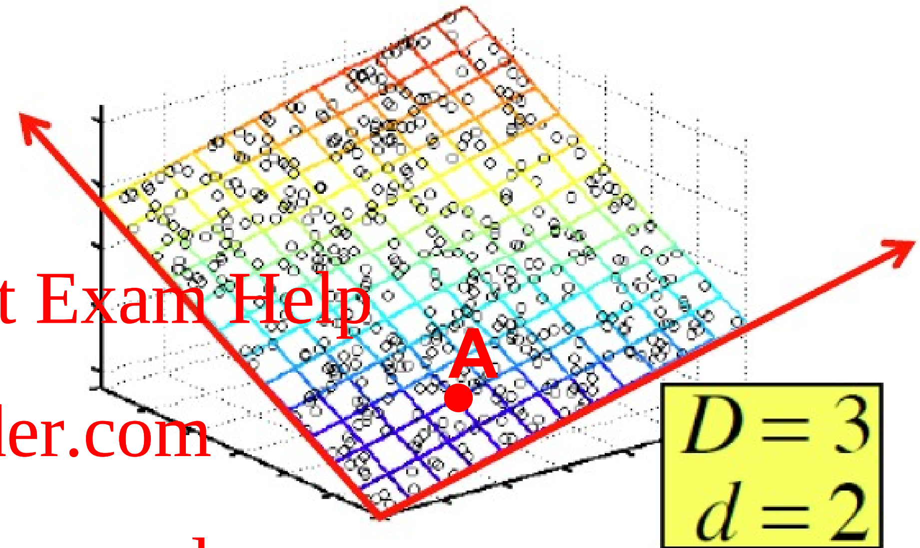
day / customer	We 7/10/16	Th 7/11/16	Fr 7/12/16	Sa 7/13/16	Su 7/14/16
ABC Inc.	1	1	1	0	0
DEF Ltd.	2	2	2	0	0
GHI Inc.	1	1	$\boxed{1}$	0	0
KLM Co.	5	5	5	0	0
Smith	0	0	0	2	2
Johnson	0	0	0	3	3
Thompson	0	0	0	1	1

The above matrix is really "2-dimensional." All rows can be reconstructed by scaling [1 1 1 0 0] or [0 0 0 1 1]

Rank of a Matrix

- **Q:** What is **rank** of a matrix **A**?
- **A:** No. of **linearly independent** rows/columns of **A**
- **For example:**
  - Matrix **A** = $\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$ has rank **r=2**
    - **Why?** The first two rows are linearly independent, so the rank is at least 2, but all three rows are linearly dependent (the first is equal to the sum of the second and third) so the rank must be less than 3.
- **Why do we care about low rank?**
  - We can write **A** as two "basis" vectors: [1 2 1] [-2 -3 1]
  - And new coordinates of : [1 0] [0 1] [1 -1]

Rank is “Dimensionality”



- **Cloud of points 3D space:**
  - Think of point positions as a matrix:

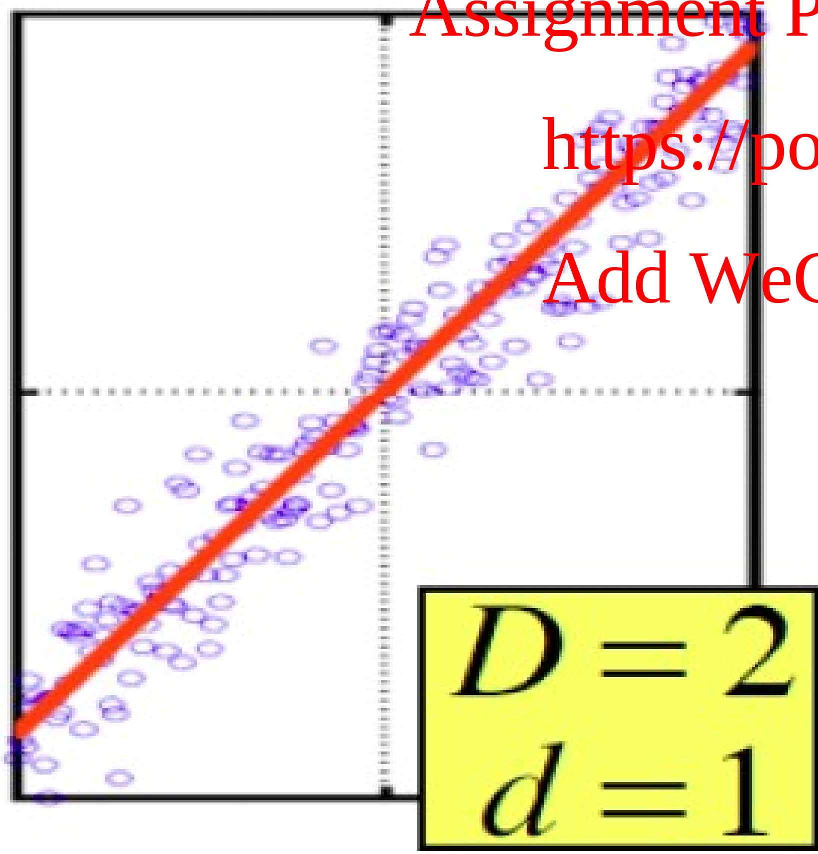
1 row per point:

$$\begin{array}{c c}\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} & \begin{array}{c} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{array}\end{array}$$

- **We can rewrite coordinates more efficiently!**
  - Old basis vectors: [1 0 0] [0 1 0] [0 0 1]
  - **New basis vectors: [1 2 1] [-2 -3 1]**
  - Then **A** has new coordinates: [1 0]. **B**: [0 1], **C**: [1 -1]
    - **Notice: We reduced the number of coordinates!**

Dimensionality Reduction

- **Goal of dimensionality reduction is to discover the axis of data!**



Rather than representing every point with 2 coordinates we represent each point with 1 coordinate (corresponding to the position of the point on the red line).

By doing this we incur a bit of **error** as the points do not exactly lie on the line

Why Reduce Dimensions?

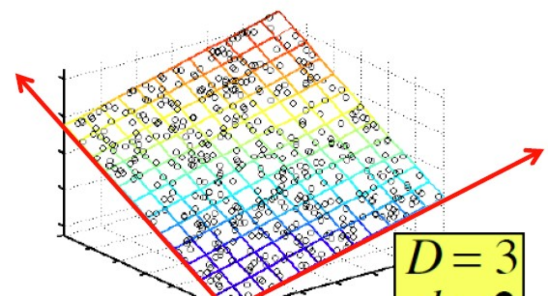
Why reduce dimensions?

- **Discover hidden correlations/topics**

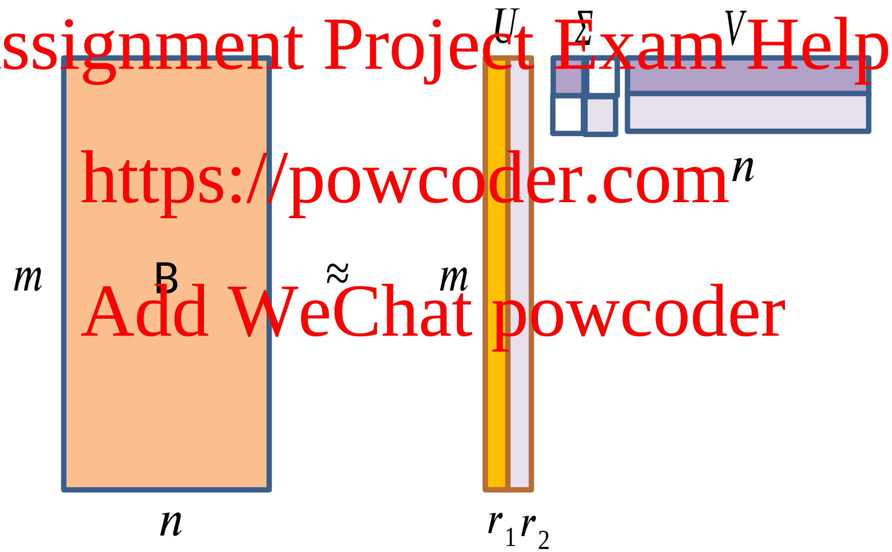
  - Words that occur commonly together

- **Remove redundant and noisy features**

  - Not all words are useful
- **Interpretation and visualization**
- **Easier storage and processing of the data**



SVD: Dimensionality Reduction

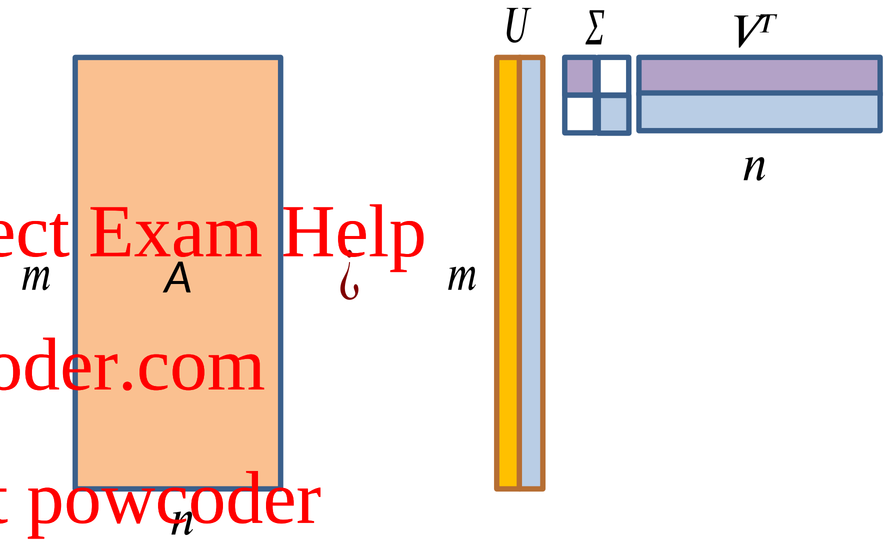


SVD: Singular Value Decomposition

- For an matrix A, we can decompose it as = , where
  - *U* is an real or complex orthonormal matrix (i.e., = = I)
  - *Σ* is an m × n rectangular diagonal matrix with non-negative real numbers on the diagonal, and
  - V^T (the conjugate transpose of V, or simply the transpose of *V* if *V* is real) is an real or complex orthonormal matrix.

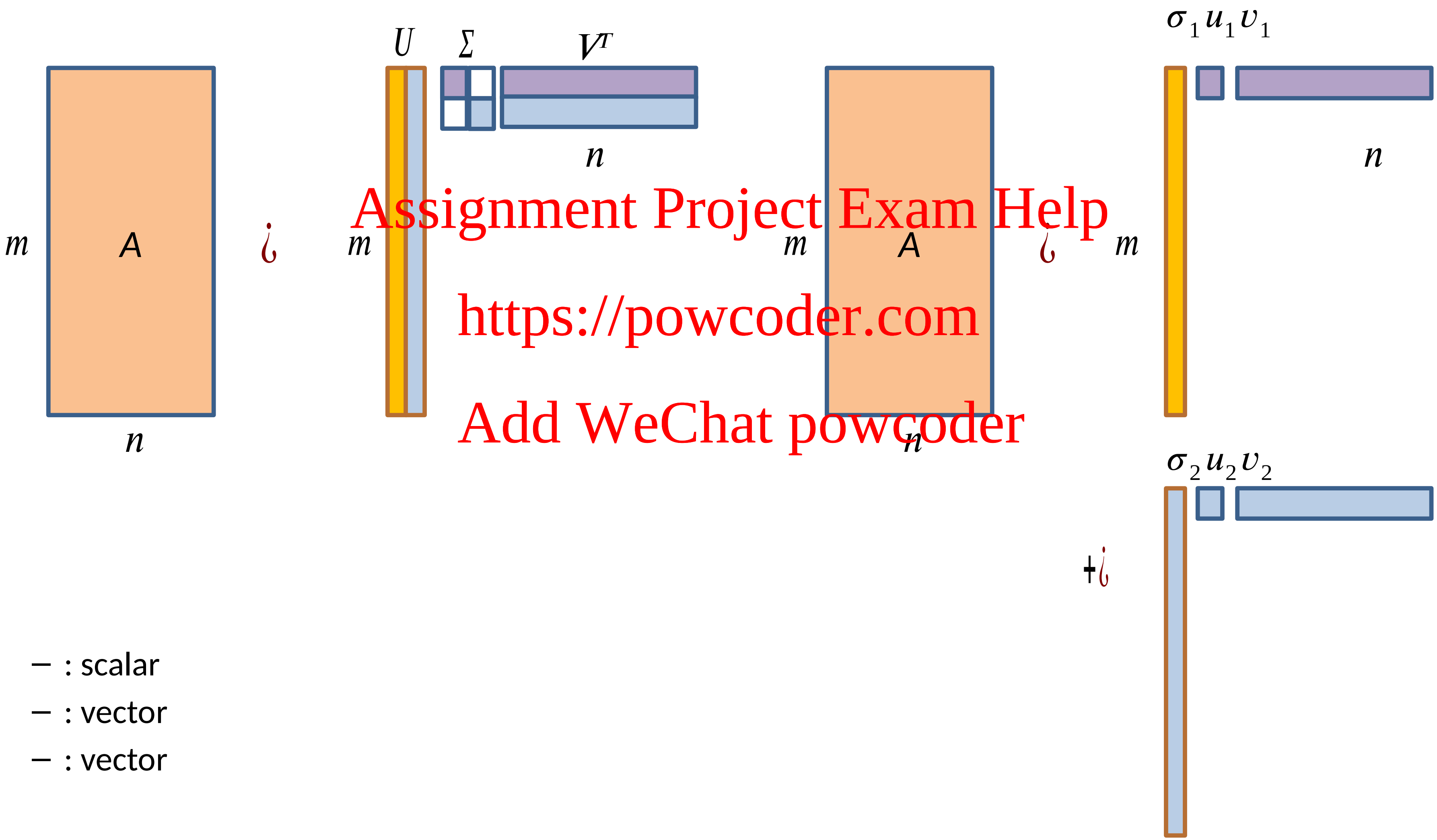
SVD: Singular Value Decomposition

- When rank(A) = r:
- : input data matrix
  - matrix (e.g., documents, terms)
- : left singular vectors
  - matrix (documents, topics)
- : singular values
  - diagonal matrix (strength of each "topic")
  - rank of matrix
- : right singular vectors
  - matrix (terms, topics)



- : scalar
- : vector
- : vector

SVD: Singular Value Decomposition



– : scalar
– : vector
– : vector

SVD Properties

- It is always possible to do SVD, i.e. decompose a matrix A into , where

- U, Σ, V: unique

- U,V: column orthonormal

  - , (I: identity matrix)
- Σ: diagonal
  - Entries (singular values) are non-negative,
  - Sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$).

SVD Example

- We give an example of a simple SVD decomposition

$$
\underset{A}{\begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \end{bmatrix}}
=
\underset{U}{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix}}
\underset{\Sigma}{\begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}
\underset{T}{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix}}
$$

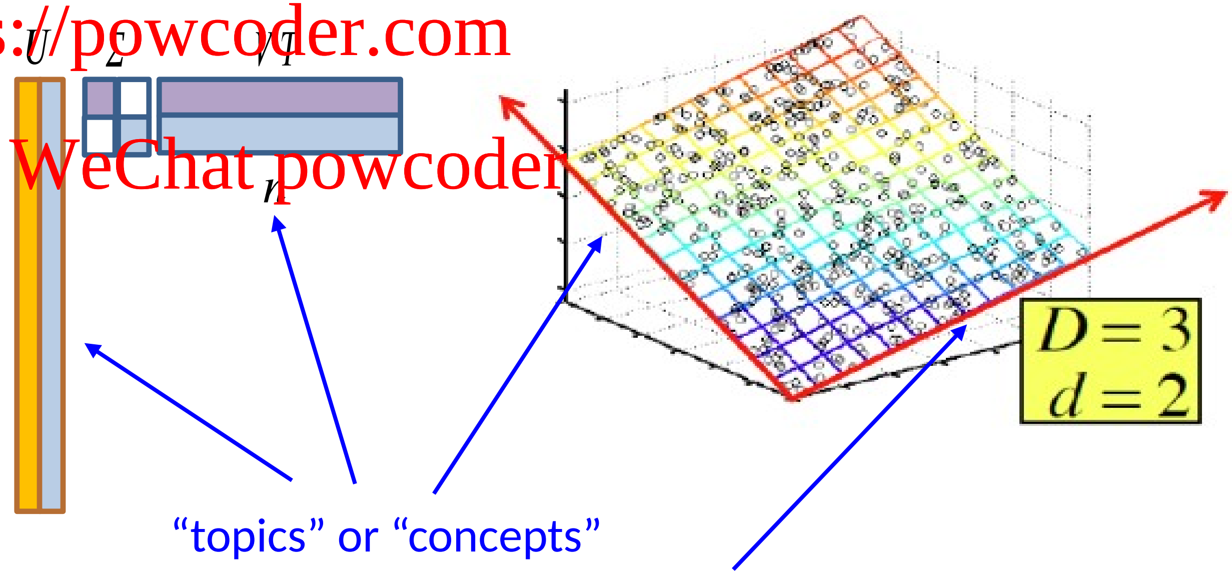
SVD: Example – Users-to-Movies

- - example: Users to Movies

$$
\begin{array}{ll|cccc}
 & & \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \\
\text{SciFi} & & 1 & 3 & 1 & 0 \\
 & & 3 & 4 & 3 & 1 \\
 & & 4 & 2 & 4 & 0 \\
 & m & 5 & 4 & 5 & 1 \\
\text{Romance} & & 0 & 1 & 0 & 4 \\
 & & 0 & 0 & 0 & 3 \\
 & & 0 & 1 & 0 & 5 \\
\end{array}
$$

n

¿ m



"topics" or "concepts"
aka. Latent dimensions
aka. Latent factors

SVD: Example – Users-to-Movies

- \- example

SciFi-concept

Romance-concept

$$
\begin{array}{ll}
 & \begin{array}{cccc} \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \end{array} \\
\begin{array}{l} \text{SciFi} \\ \\ \\ m \\ \text{Romance} \\ \\ \end{array} &
\begin{bmatrix}
1 & 3 & 1 & 0 \\
3 & 4 & 3 & 1 \\
4 & 2 & 4 & 0 \\
5 & 4 & 5 & 1 \\
0 & 1 & 0 & 4 \\
0 & 0 & 0 & 3 \\
0 & 1 & 0 & 5
\end{bmatrix} \\
 & \quad n
\end{array}
\;\; m \text{ ¿ } \;\;
\begin{bmatrix}
0.24 & 0.02 & 0.69 \\
0.48 & -0.02 & 0.43 \\
0.49 & -0.08 & -0.52 \\
0.68 & -0.07 & -0.16 \\
0.06 & 0.57 & 0.06 \\
0.01 & 0.41 & -0.20 \\
0.07 & 0.70 & -0.01
\end{bmatrix}
\times
\begin{bmatrix}
11.9 & 0 & 0 \\
0 & 7.1 & 0 \\
0 & 0 & 2.5
\end{bmatrix}
\times
\begin{bmatrix}
0.59 & 0.54 & 0.59 & 0.05 \\
-0.10 & 0.12 & -0.10 & 0.98 \\
-0.37 & 0.83 & -0.37 & -0.17
\end{bmatrix}
$$

n

SVD: Example – Users-to-Movies

- - example

***U* is "user-to-concept" similarity matrix**

SciFi-concept

Romance-concept

$$
\begin{array}{c c}
 & \begin{array}{cccc} \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \end{array} \\
\begin{array}{c} \text{SciFi} \\ \\ \\ m \\ \\ \text{Romance} \\ \\ \end{array} &
\begin{bmatrix}
1 & 3 & 1 & 0 \\
3 & 4 & 3 & 1 \\
4 & 2 & 4 & 0 \\
5 & 4 & 5 & 1 \\
0 & 1 & 0 & 4 \\
0 & 0 & 0 & 3 \\
0 & 1 & 0 & 5
\end{bmatrix} \\
 & n
\end{array}
\;\; m \text{¿} \;\;
\begin{bmatrix}
0.24 & 0.02 & 0.69 \\
0.48 & -0.02 & 0.43 \\
0.49 & -0.08 & -0.52 \\
0.68 & -0.07 & -0.16 \\
0.06 & 0.57 & 0.06 \\
0.01 & 0.41 & -0.20 \\
0.07 & 0.70 & -0.01
\end{bmatrix}
\times
\begin{bmatrix}
11.9 & 0 & 0 \\
0 & 7.1 & 0 \\
0 & 0 & 2.5
\end{bmatrix}
\times
$$

$$
\begin{array}{c}
\begin{bmatrix}
0.59 & 0.54 & 0.59 & 0.05 \\
-0.10 & 0.12 & -0.10 & 0.98 \\
-0.37 & 0.83 & -0.37 & -0.17
\end{bmatrix} \\
n
\end{array}
$$

SVD: Example – Users-to-Movies

- - example

$$
\begin{array}{ccc}
 & & \begin{array}{cccc} \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \end{array} \\
\begin{array}{c} \text{SciFi} \\ \\ \text{Romance} \end{array} & m & \begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} \\
 & & n
\end{array}
\quad m \;\text{¿}\;
\begin{bmatrix} 0.24 & 0.02 & 0.69 \\ 0.48 & -0.02 & 0.43 \\ 0.49 & -0.08 & -0.52 \\ 0.68 & -0.07 & -0.16 \\ 0.06 & 0.57 & 0.06 \\ 0.01 & 0.41 & -0.20 \\ 0.07 & 0.70 & -0.01 \end{bmatrix}
\times
\begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}
\times
\begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ -0.37 & 0.83 & -0.37 & -0.17 \end{bmatrix}_{n}
$$

SciFi-concept

"strength" of the SciFi-concept

SVD: Example – Users-to-Movies

- - example

V is "movie-to-concept" similarity matrix

SciFi-concept

$$
\begin{array}{c|c}
 & \text{The Avengers} \quad \text{Star Wars} \quad \text{Matrix} \quad \text{Twilight} \\
\begin{array}{c} \text{SciFi} \\ \\ \\ \\ \text{Romance} \\ \\ \end{array} \; m &
\begin{bmatrix}
1 & 3 & 1 & 0 \\
3 & 4 & 3 & 1 \\
4 & 2 & 4 & 0 \\
5 & 4 & 5 & 1 \\
0 & 1 & 0 & 4 \\
0 & 0 & 0 & 3 \\
0 & 1 & 0 & 5
\end{bmatrix}_{n}
\end{array}
= m
\begin{bmatrix}
0.24 & 0.02 & 0.69 \\
0.48 & -0.02 & 0.43 \\
0.49 & -0.08 & -0.52 \\
0.68 & -0.07 & -0.16 \\
0.06 & 0.57 & 0.06 \\
0.01 & 0.41 & -0.20 \\
0.07 & 0.70 & -0.01
\end{bmatrix}
\text{ x }
\begin{bmatrix}
11.9 & 0 & 0 \\
0 & 7.1 & 0 \\
0 & 0 & 2.5
\end{bmatrix}
\text{ x }
$$

SciFi-concept

$$
\begin{bmatrix}
0.59 & 0.54 & 0.59 & 0.05 \\
-0.10 & 0.12 & -0.10 & 0.98 \\
-0.37 & 0.83 & -0.37 & -0.17
\end{bmatrix}_{n}
$$

Q: Does the movie "Twilight" relate to concept "Romance"?

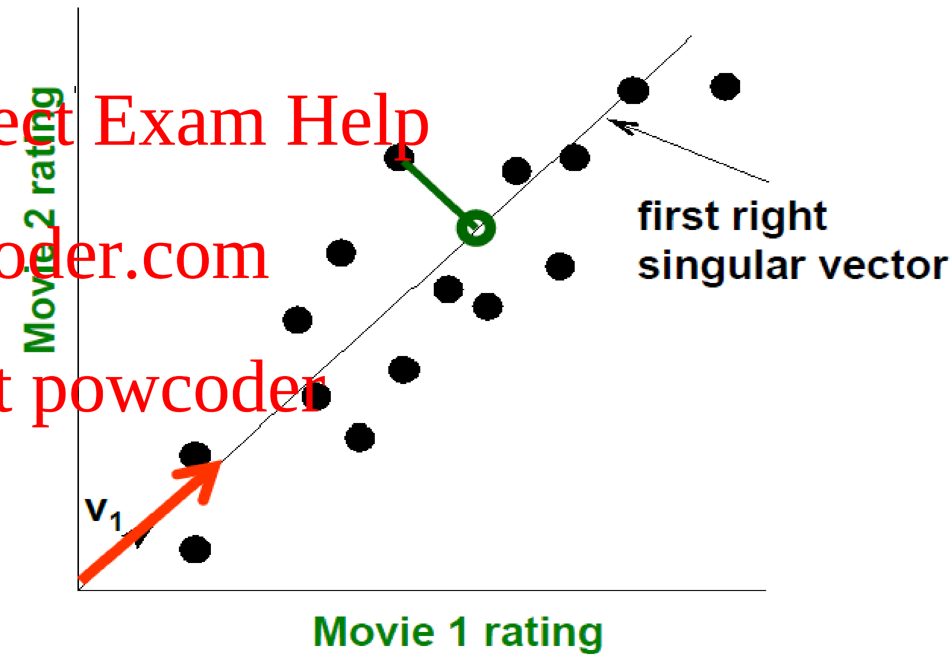
SVD: Interpretation #1

- "movies", "users" and "concepts"
  - : user-to-concept similarity matrix
  - : movie-to-concept similarity matrix
  - : its diagonal elements
    - 'strength' of each concept

SVD: Interpretations #2

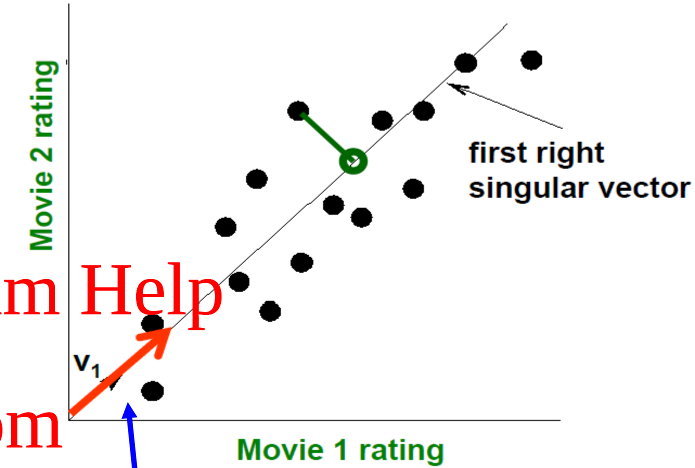
- SVD gives 'best' axis to project on
  - 'best' = minimal sum of squares of projection errors
- In other words,

  minimum reconstruction error



SVD: Interpretation #2

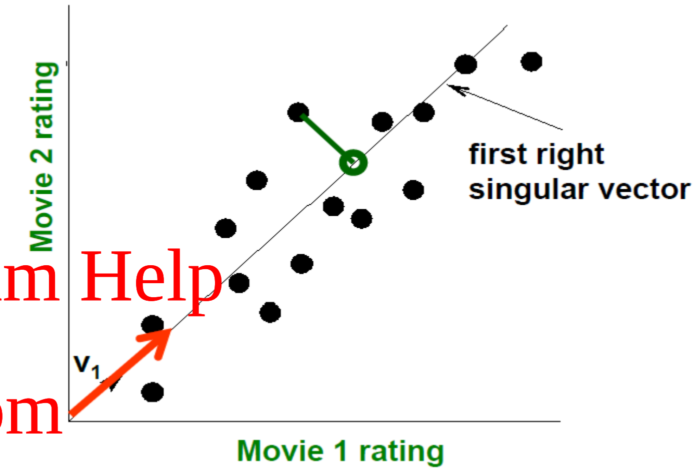
- - example
  - U: user-to-concept matrix
  - V: movie-to-concept matrix



$$\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} ¿ \begin{bmatrix} 0.24 & 0.02 & 0.69 \\ 0.48 & -0.02 & 0.43 \\ 0.49 & -0.08 & -0.52 \\ 0.68 & -0.07 & -0.16 \\ 0.06 & 0.57 & 0.06 \\ 0.01 & 0.41 & -0.20 \\ 0.07 & 0.70 & -0.01 \end{bmatrix} \mathrm{x} \begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & 2.5 \end{bmatrix} \mathrm{x} \begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ -0.37 & 0.83 & -0.37 & -0.17 \end{bmatrix}$$

SVD: Interpretation #2

- - example



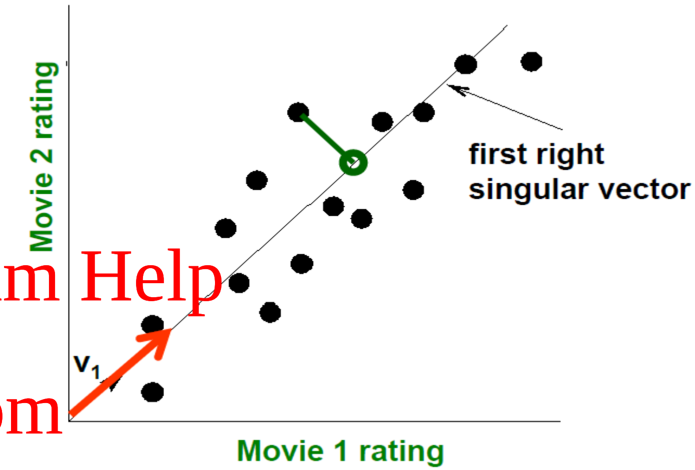
variance ("spread")
on the v_1 axis

$$\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} \text{¿} \begin{bmatrix} 0.24 & 0.02 & 0.69 \\ 0.48 & -0.02 & 0.43 \\ 0.49 & -0.08 & -0.52 \\ 0.68 & -0.07 & -0.16 \\ 0.06 & 0.57 & 0.06 \\ 0.01 & 0.41 & -0.20 \\ 0.07 & 0.70 & -0.01 \end{bmatrix} \text{X} \begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & 2.5 \end{bmatrix} \text{X}$$

$$\begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ -0.37 & 0.83 & -0.37 & -0.17 \end{bmatrix}$$

SVD: Interpretation #2

- - example
  - : the coordinates of the points in the projection axis



Projection of users on the “Sci-Fi” axis

$$\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} \qquad \begin{bmatrix} 2.86 & 0.24 & 8.21 \\ 5.71 & -0.24 & 5.12 \\ 5.83 & -0.95 & -6.19 \\ 8.09 & -0.83 & -1.90 \\ 0.71 & 6.78 & 0.71 \\ 0.12 & 4.88 & -2.38 \\ 0.83 & 8.33 & -0.12 \end{bmatrix}$$

SVD: Interpretation #2

- Q: how exactly is dimension reduction done?

$$\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} ¿ \begin{bmatrix} 0.24 & 0.02 & 0.69 \\ 0.48 & -0.02 & 0.43 \\ 0.49 & -0.08 & -0.52 \\ 0.68 & -0.07 & -0.16 \\ 0.06 & 0.57 & 0.06 \\ 0.01 & 0.41 & -0.20 \\ 0.07 & 0.70 & -0.01 \end{bmatrix} \text{x} \begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & 2.5 \end{bmatrix} \text{x}$$

$$\begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ -0.37 & 0.83 & -0.37 & -0.17 \end{bmatrix}$$

SVD: Interpretation #2

- Q: how exactly is dimension reduction done?
- A: Set smallest singular values to zero

$$\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} \text{¿} \begin{bmatrix} 0.24 & 0.02 & 0.69 \\ 0.48 & -0.02 & 0.43 \\ 0.49 & -0.08 & -0.52 \\ 0.68 & -0.07 & -0.16 \\ 0.06 & 0.57 & 0.06 \\ 0.01 & 0.41 & -0.20 \\ 0.07 & 0.70 & -0.01 \end{bmatrix} \text{x} \begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & 2.5 \end{bmatrix} \text{x} \begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ -0.37 & 0.83 & -0.37 & -0.17 \end{bmatrix}$$

SVD: Interpretation #2

- Q: how exactly is dimension reduction done?
- A: Set smallest singular values to zero
  - Approximate original matrix by low-rank matrices

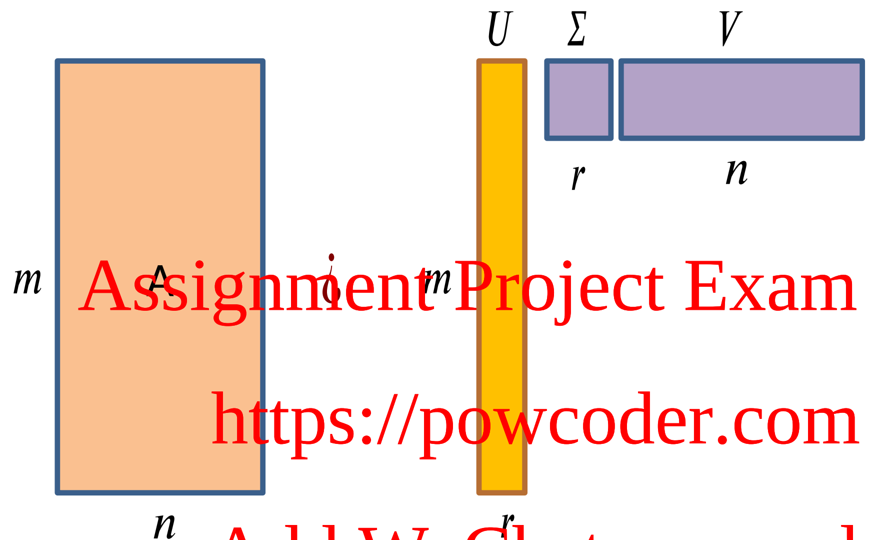
$$
\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix}
\approx
\begin{bmatrix} 0.24 & 0.02 & \cancel{0.69} \\ 0.48 & -0.02 & \cancel{0.43} \\ 0.49 & -0.08 & \cancel{-0.52} \\ 0.68 & -0.07 & \cancel{-0.16} \\ 0.06 & 0.57 & \cancel{0.06} \\ 0.01 & 0.41 & \cancel{-0.20} \\ 0.07 & 0.70 & \cancel{-0.01} \end{bmatrix}
\text{X}
\begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & \cancel{2.5} \end{bmatrix}
\text{X}
\begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ \cancel{-0.37} & \cancel{0.83} & \cancel{-0.37} & \cancel{-0.17} \end{bmatrix}
$$

SVD: Interpretation #2

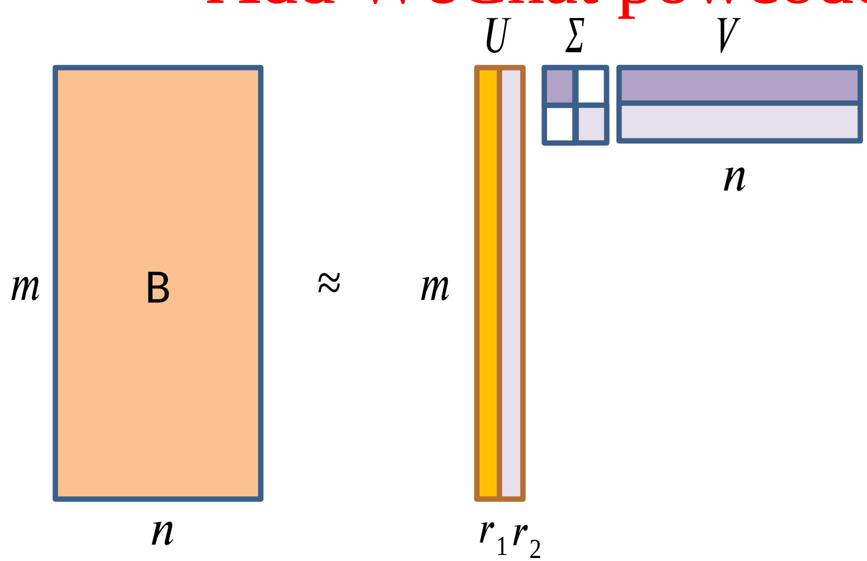
- Q: how exactly is dimension reduction done?
- A: Set smallest singular values to zero
  - Approximate original matrix by low-rank matrices

$$\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix} \approx \quad \times \begin{bmatrix} 11.9 & 0 \\ 0 & 7.1 \end{bmatrix} \times$$

SVD: Best Low Rank Approximation



is a rank matrix



is the **best** rank approximation of matrix

SVD: Best Low Rank Approximation

- **Theorem**: Let (), and
    - = diagonal matrix where (and ()
    - or equivalently, is the best rank- approximation to :
    - or equivalently,

- Intuition (spectral decomposition)

  – Why setting small to 0 is the right thing to do?
    - Vectors and are unit length, so scales them.
    - Therefore, zeroing small introduces less error.

SVD: Interpretation #2

- Q: How many σ_i to keep?
- A: Rule-of-a thumb

Keep 80~90% “energy” ()

$$m \begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix}_{n} \text{¿} \quad \sigma_1 u_1 \circ v_1^T + \sigma_2 u_2 \circ v_2^T + \cdots$$

Assume: $\sigma_1 \geq \sigma_2 \geq \cdots$

SVD: Complexity

- SVD for full matrix

- But

  - Less work, if we only want to compute singular values
  - or if we only want first k singular vectors (thin-svd).

  - or if the matrix is sparse (sparse svd).
- Stable implementations
  - LINPACK, Matlab, Splus, Mathematica...
  - Available in most common languages

SVD: Conclusions so far

- SVD: : unique
  - user-to-concept similarities
  - movie-to-concept similarities
  - : strength to each concept
- Dimensionality reduction
  - Keep the few largest singular values (80-90% of "energy")
  - SVD: picks up linear correlations

SVD: Relationship to Eigen-decomposition

- SVD gives us

- Eigen-decomposition

  - is symmetric
  - are orthonormal (
  - are diagonal

SVD: Relationship to Eigen-decomposition

- Eigen-decomposition of and

  - So, , and
  - That is, is the matrix of eigenvectors of , and
 is the matrix of eigenvectors of
  - This shows how to use eigen-decomposition to compute SVD
  - Ahe singular values of are the square roots of the corresponding eigenvalues of
- Note: and are the dataset covariance matrices

A Brief Review of Eigen-Decomposition

- Eigenvalues and eigenvectors
  - matrix.
  - eigenvalue of , : eigenvector of , eigenpair.
- Simple computational method of eigenvalues
  - Solve the equation
  - Example
    - Then
    - Then
    - Solve , we get

A Brief Review of Eigen-Decomposition

- Example (continued)

  - solve , we get eigenvalues
  - now we compute eigenvectors.

    - for eigenvalue we need to find
    - solve
    - We get Since needs to be a unit vector, therefore
    - Similarly, we can compute

Computing Eigenvalues: Power Method

- Power method
  - choose an arbitrary
  - .
  - Theorem: sequence converges to the principal eigenvector (i.e., the eigenvector corresponds to the largest eigenvalue)
- Normalized power method
  - choose an arbitrary
  - Theorem: sequence converges to the principal eigenvector.

In-class Practice

- Go to practice

SVD Case Study: How to Query?

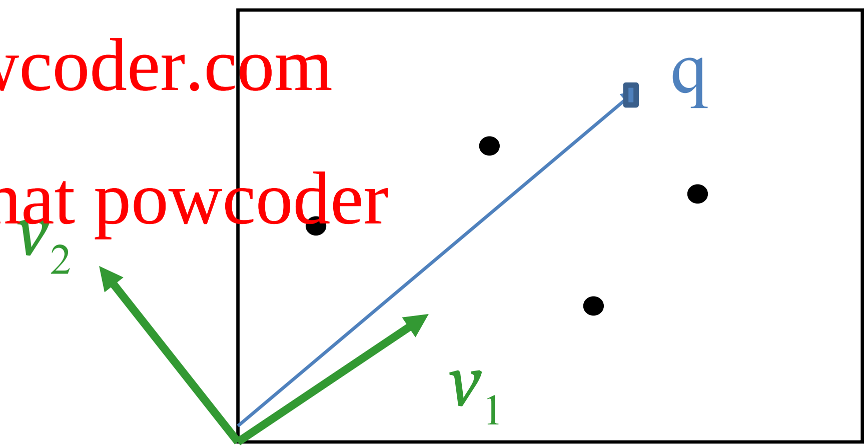
- Q: Find users that like "The Avengers"
- A: Map query into a "concept space" – how?

$$
\begin{array}{l c}
 & \begin{array}{cccc} \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \end{array} \\
\begin{array}{l} \text{SciFi} \left\{ \begin{array}{l} \\ \\ \\ \\ \end{array} \right. \\ \text{Romance} \left\{ \begin{array}{l} \\ \\ \\ \end{array} \right. \end{array} &
\begin{bmatrix} 1 & 3 & 1 & 0 \\ 3 & 4 & 3 & 1 \\ 4 & 2 & 4 & 0 \\ 5 & 4 & 5 & 1 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \end{bmatrix}
\end{array}
\; m\,¿ \;
\begin{bmatrix} 0.24 & 0.02 & 0.69 \\ 0.48 & -0.02 & 0.43 \\ 0.49 & -0.08 & -0.52 \\ 0.68 & -0.07 & -0.16 \\ 0.06 & 0.57 & 0.06 \\ 0.01 & 0.41 & -0.20 \\ 0.07 & 0.70 & -0.01 \end{bmatrix}
\;\text{x}\;
\begin{bmatrix} 11.9 & 0 & 0 \\ 0 & 7.1 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}
\;\text{x}\;
\begin{bmatrix} 0.59 & 0.54 & 0.59 & 0.05 \\ -0.10 & 0.12 & -0.10 & 0.98 \\ -0.37 & 0.83 & -0.37 & -0.17 \end{bmatrix}
$$

Case Study: How to Query?

- Q: Find users that like "The Avengers"
- A: Map query into a "concept space" – how?

$$\begin{array}{ccccc} & \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \\ q & [5 & 0 & 0 & 0] \end{array}$$

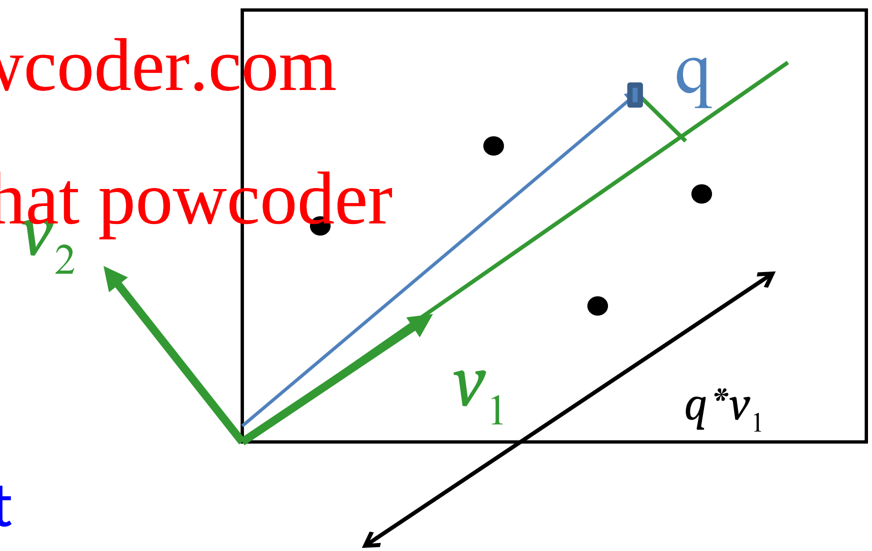


Project into concept space:
Inner product with each concept vector v_i

Case Study: How to Query?

- Q: Find users that like "The Avengers"
- A: Map query into a "concept space" – how?

$$
\begin{array}{cccccc}
 & \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \\
q & [5 & 0 & 0 & 0]
\end{array}
$$



Project into concept space:
Inner product with each concept vector v_i

Case Study: How to Query?

- Compactly, we have
  - $q_c = qV$

SciFi-concept

$$
\begin{array}{cccccc}
 & \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \\
q & 5 & 0 & 0 & 0
\end{array}
\quad \text{x} \quad
\begin{bmatrix} 0.59 & -0.10 \\ 0.54 & 0.12 \\ 0.59 & -0.10 \\ 0.05 & 0.98 \end{bmatrix}
\quad = \quad
\begin{bmatrix} 2.95 & -0.50 \end{bmatrix}
$$

movie-to-concept
similarities ()

Case Study: How to Query?

- How would the user *d* that rated ('Star Wars', 'Matrix') be handled?
  - $d_c = d\,V$

$$d \quad \begin{array}{cccc} \text{The Avengers} & \text{Star Wars} & \text{Matrix} & \text{Twilight} \\ \hline 0 & 4 & 5 & 0 \end{array} \quad \times \quad \begin{bmatrix} 0.59 & -0.10 \\ 0.54 & 0.12 \\ 0.59 & -0.10 \\ 0.05 & 0.98 \end{bmatrix} \quad = \quad \begin{array}{cc} \text{SciFi-concept} & \\ 5.11 & -0.02 \end{array}$$

movie-to-concept similarities ()

Case Study: How to Query?

- Observation
  - User d that rated ('Star Wars') will be similar to user q that rate ('The Avengers'), although d and q have zero ratings in common!

SciFi-concept

$$\begin{array}{ccc} & \text{The Avengers} \quad \text{Star Wars} \quad \text{Matrix} \quad \text{Twilight} & \\ d & \begin{bmatrix} 0 & 4 & 5 & 0 \end{bmatrix} \dashrightarrow & \begin{bmatrix} 5.11 & -0.02 \end{bmatrix} \\ q & \begin{bmatrix} 5 & 0 & 0 & 0 \end{bmatrix} \dashrightarrow & \begin{bmatrix} 2.95 & -0.50 \end{bmatrix} \end{array}$$

Zero ratings in common

Similarity ≠ 0

Cosine similarity: 0.99

SVD: Drawbacks

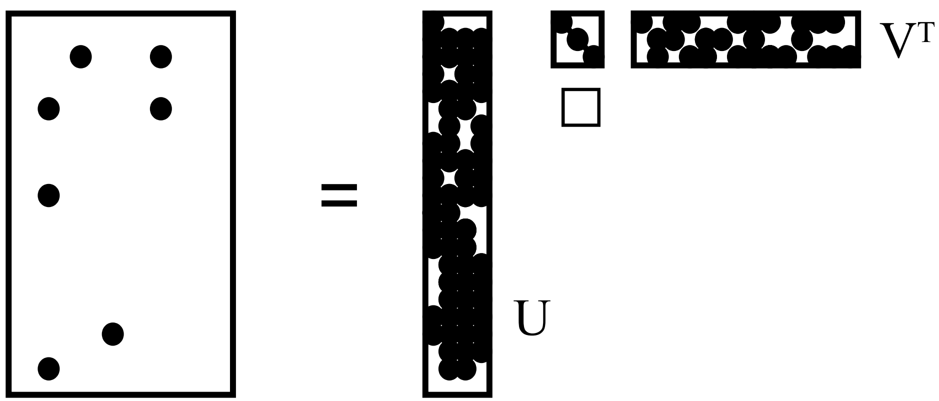
\+ **Optimal low-rank approximation** in terms of Euclidean norm

– **Interpretability problem:**

  – A singular vector specifies a linear combination of all input columns or rows

– **Lack of sparsity:**

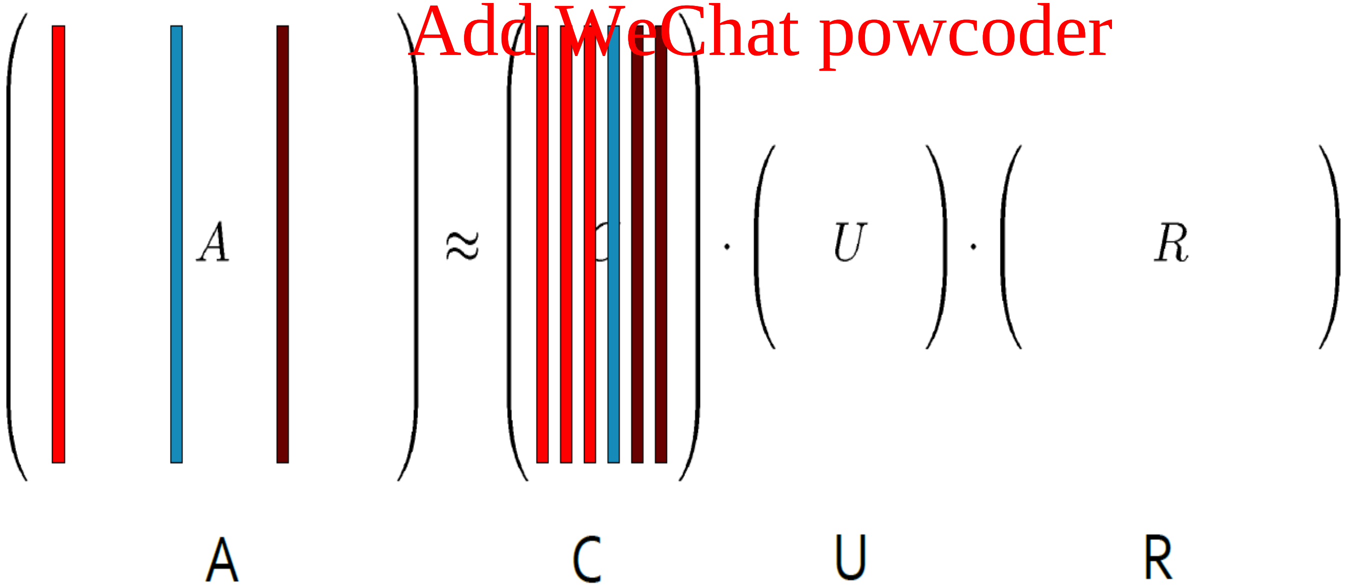
  – Singular vectors are **dense!**



CUR Decomposition

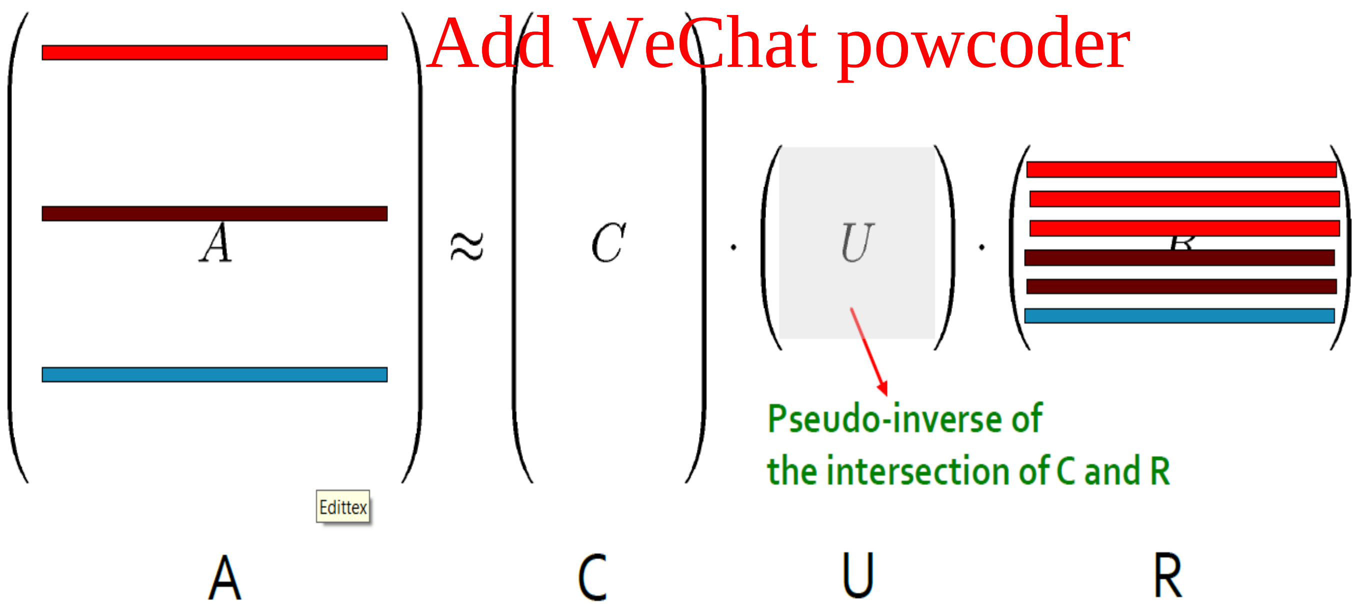
- Goal: express A as a product of matrices
  - Minimize

- Constraints on



CUR Decomposition

- Goal: express A as a product of matrices
  - Minimize
- Constraints on



CUR: Good Approximation to SVD

- Let be the best rank approximation of (obtain by SVD)
- Theorem
  - CUR algorithm in time achieves
  - with probability at least , by picking
    - columns and
    - rows
    - (in practice, choose columns/rows)

CUR: How it Works

- Sampling columns (similarly for rows):

Input: matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, sample size c

Output: $\mathbf{C}_d \in \mathbb{R}^{m \times c}$

for $x = 1 : n$ [column distribution]

$P(x) = \sum_i \mathbf{A}(i, x)^2 / \sum_{i,j} \mathbf{A}(i, j)^2$

for $i = 1 : c$ [sample columns]

Pick $j \in 1 : n$ based on distribution $P(x)$

Compute $\mathbf{C}_d(:, i) = \mathbf{A}(:, j) / \sqrt{cP(j)}$

CUR: Computing U

- Let be the “intersection” of sampled columns C and rows R
  - Let SVD of

- Then: , where
  - is the “Moore-Penrose pseudo-inverse”.

  - , if .

CUR: Pros & Cons

+ easy interpretation
  – the basis vectors are actual columns and rows

- duplicate columns and rows

  – columns of large norms will be sampled many times

CUR: Duplicate Columns

- If we want to get rid of the duplicates
  - Throw them away
  - Scale the columns/rows by the square root of the number of duplicates

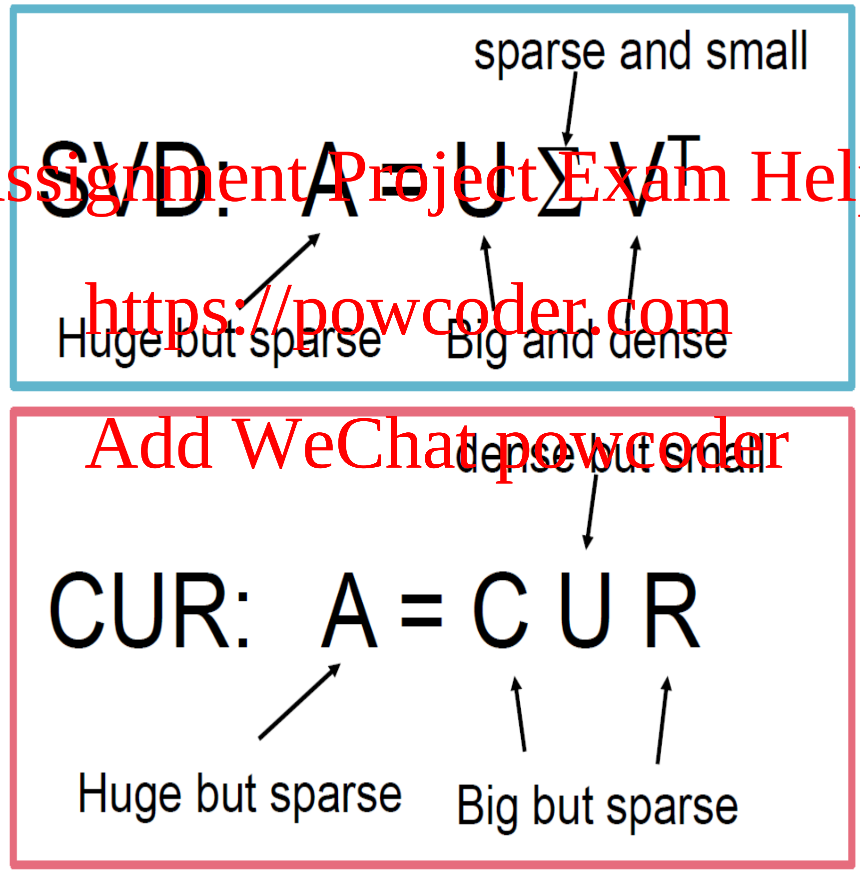
SVD vs CUR

Question: Large or small? Dense or sparse?

SVD: $A = U\,\Sigma\,V^T$

CUR: $A = C\,U\,R$

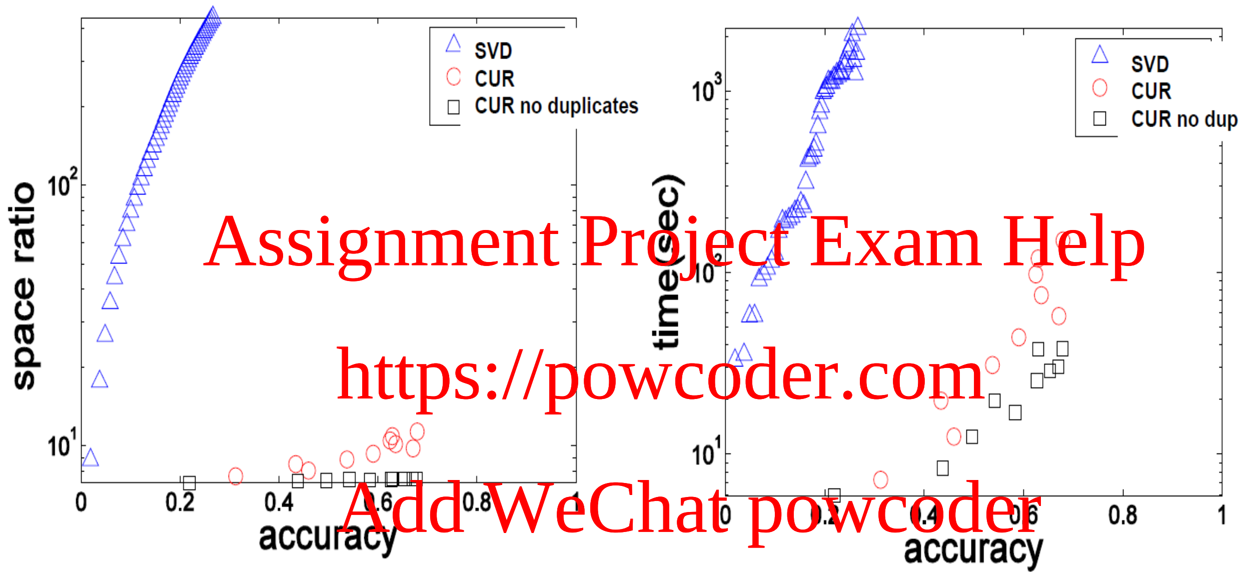
SVD vs CUR


sparse and small
SVD: A = U Σ V^T
Huge but sparse
Big and dense
dense but small
CUR: A = C U R
Huge but sparse
Big but sparse

SVD & CUR: Simple Experiments

- DBLP data
  - author-to-conference matrix
    - very sparse
  - : number of papers published by author at conference .
  - 428k authors (rows)
  - 3659 conferences (column)
- Dimensionality reduction
  - Running time?
  - Space?
  - Reconstruction error?

Results: DBLP



courtesy: Sun, Faloutsos: *Less is more, Compact Matrix Decomposition for Large Sparse Graph*, SDM'07

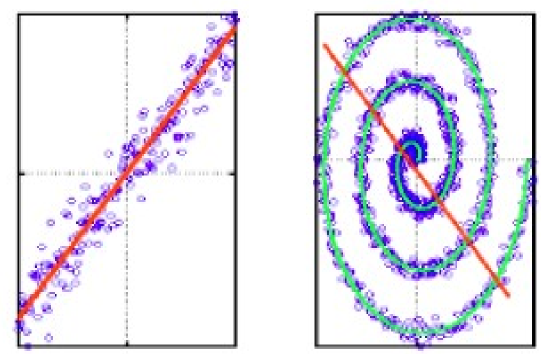
- accuracy: 1-relative sum squared errors
- space ratio: # output non-zero matrix entries / # input non-zero matrix entries

The Linearity Assumption

- SVD is limited to linear projections
  - Data lies on a low-dimensional linear space

- Non-linear methods: Isomap

  - Data lies on a low-dimensional manifold
    - Non-linear



  - How?
    - Build adjacency graph
    - SVD the graph adjacency matrix
    - Further reading: wikipage of Isomap

PCA: An Application of SVD

- PCA = Principle Component Analysis

- Motivation
  - Visualization

PCA: Data Visualization

- Example:
  - Given 53 blood samples (features) from 65 people (data item or instance)

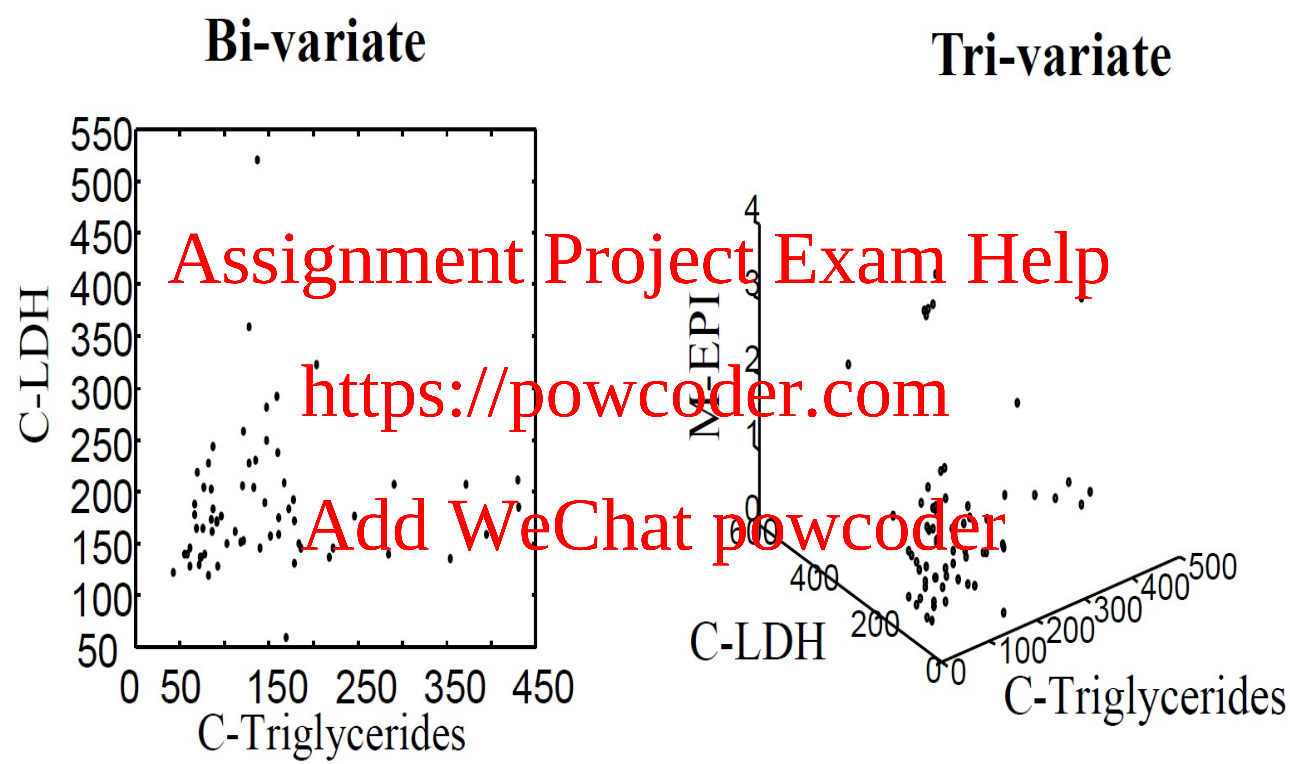
Instances

	H-WBC	H-RBC	H-Hgb	H-Hct	H-MCV	H-MCH	H-MCHC
A1	8.0000	4.8200	14.1000	41.0000	85.0000	29.0000	34.0000
A2	7.3000	5.0200	14.7000	43.0000	86.0000	29.0000	34.0000
A3	4.3000	4.4800	14.1000	41.0000	91.0000	32.0000	35.0000
A4	7.5000	4.4700	14.9000	45.0000	101.0000	33.0000	33.0000
A5	7.3000	5.5200	15.4000	46.0000	84.0000	28.0000	33.0000
A6	6.9000	4.8600	16.0000	47.0000	97.0000	33.0000	34.0000
A7	7.8000	4.6800	14.7000	43.0000	92.0000	31.0000	34.0000
A8	8.6000	4.8200	15.8000	42.0000	88.0000	33.0000	37.0000
A9	5.1000	4.7100	14.0000	43.0000	92.0000	30.0000	32.0000

Features

  - How can we visualize the samples

PCA: Data Visualization



How can we visualize the other variables???

… difficult to see in 4 or higher dimensional spaces …

PCA: Data Visualization

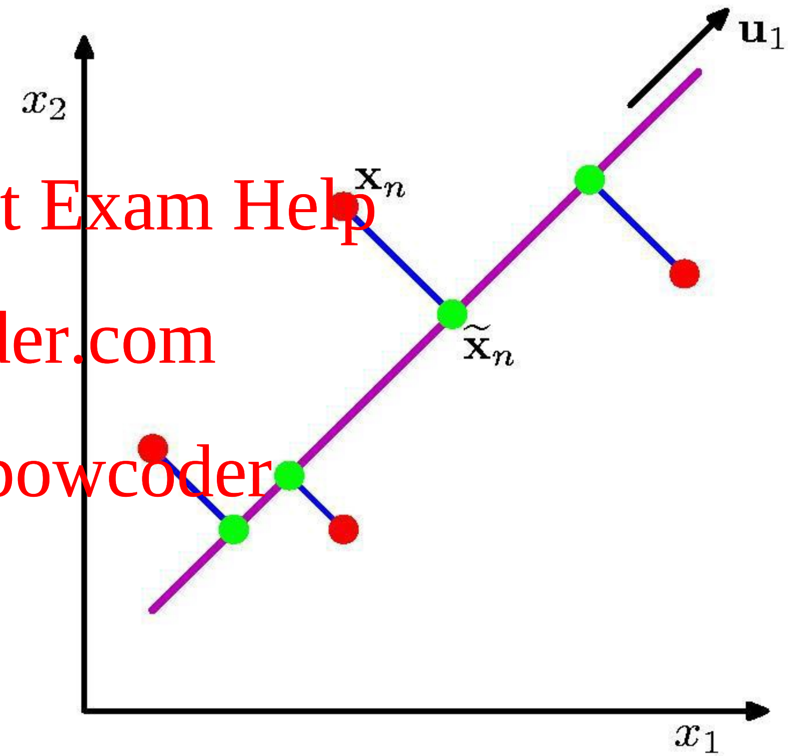
- Is there a representation better than the coordinate axes?
- Is it really necessary to show all the 53 dimensions?

  – What if there are strong correlations between the features?
  – How could we find the smallest subspace of the 53-D space that keeps the most information about the original data?

- A solution: Principal Component Analysis
  – An application of SVD.

PCA: Definition and Algorithms

- PCA
  - Orthogonal projection of the data onto a lower-dimensional linear space such that
    - Maximize variance of projected data (purple line)
    - Minimize mean squared distance between
      - Data point
      - Projection (sum of blue lines)
  - Look data from a literally different angle.



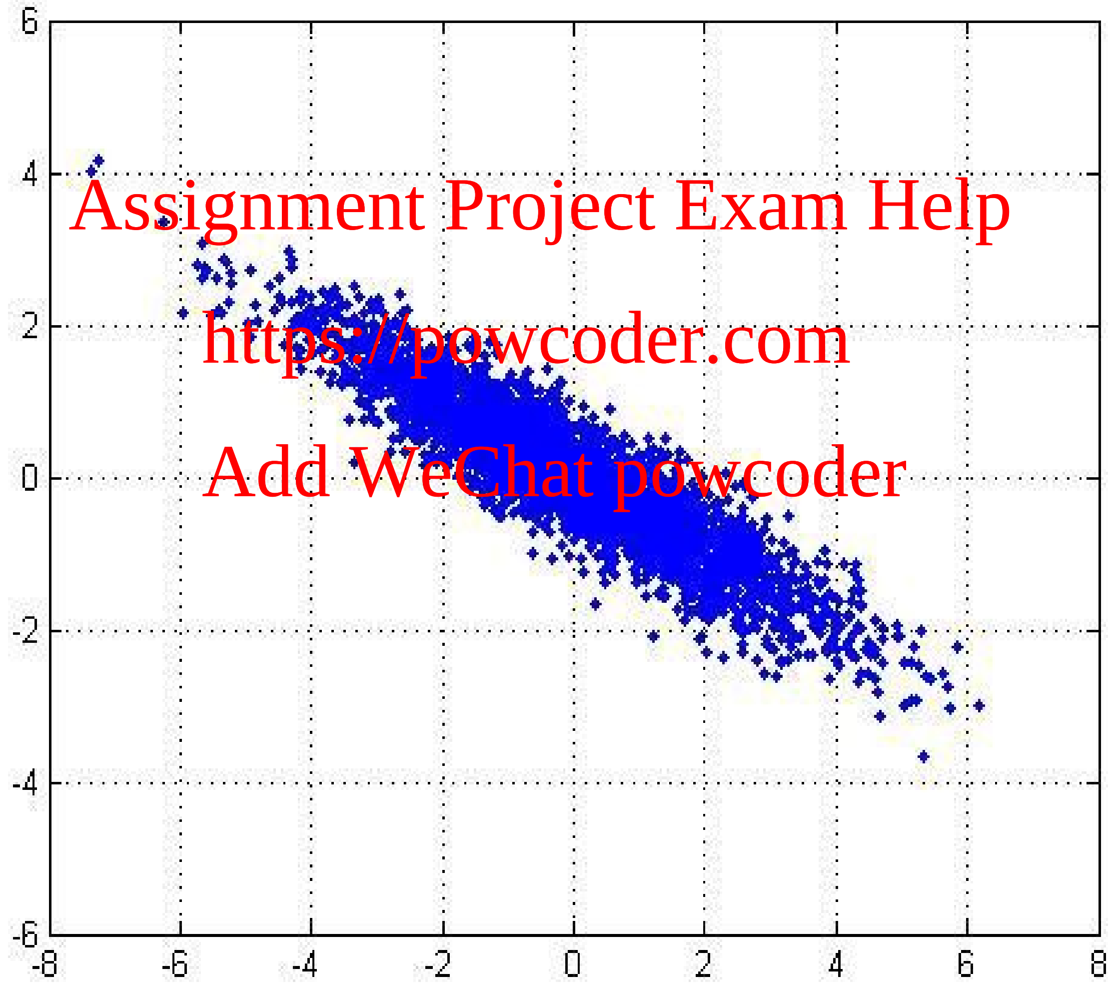
PCA: Idea

- Given data points in a d-dimensional space, project them into a lower dimensional space while preserving as much information as possible.
  - Find best planar approximation to 3D data
  - Find best 12-D approximation to 104-D data
- In particular, choose projection that minimizes squared error in reconstructing the original data.
  - Implement through SVD

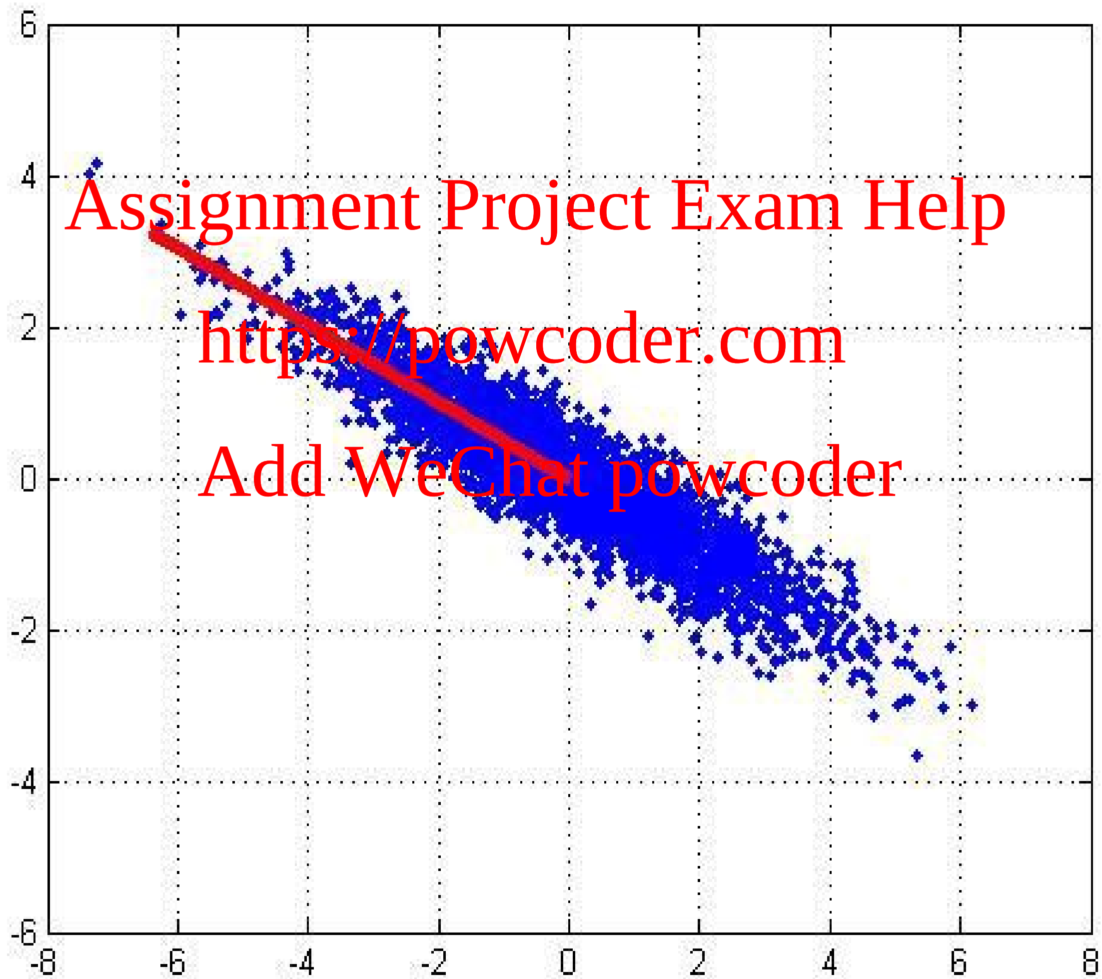
PCA

- **PCA Vectors** originate from the center of mass.
- Principal component #1: points in the direction of the **largest variance**.
- Each subsequent principal component
  - is **orthogonal** to the previous ones, and
  - points in the directions of the **largest variance of the residual subspace**

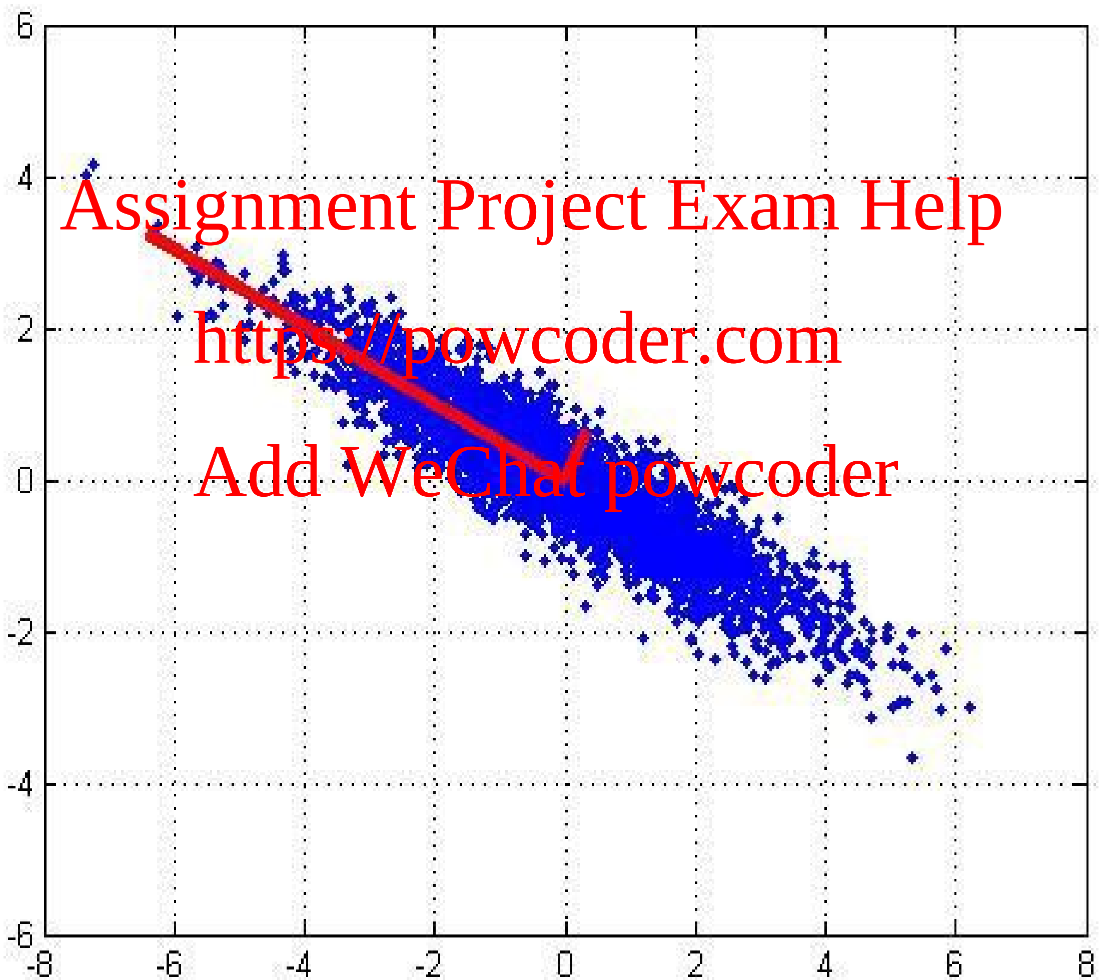
PCA: 2D Gaussian dataset



1^{st} PCA axis



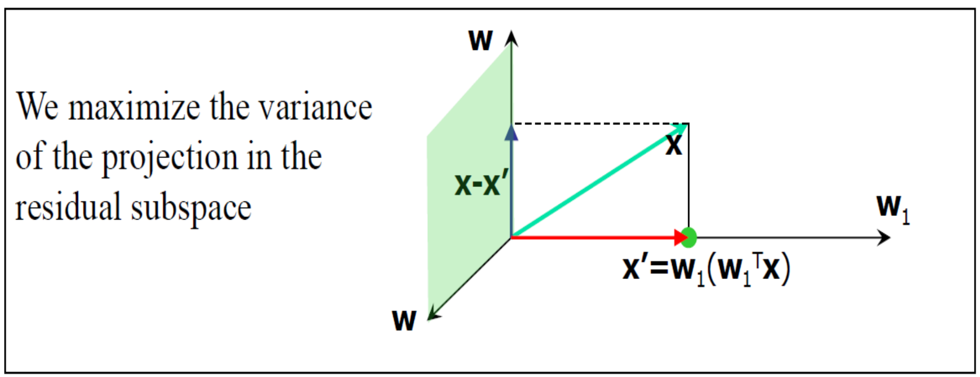
2^{nd} PCA axis



PCA: Algorithm

- Given centered data , compute principle vectors
  - 1^{st} principle vector

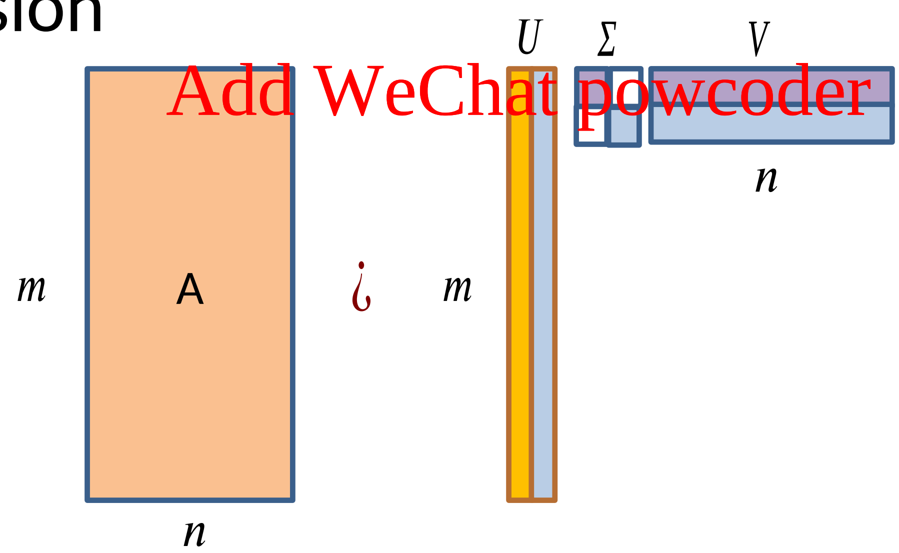
  - maximize the variance of projection of



PCA: Algorithm by SVD

- SVD of the centered data matrix

  - : number of instances
  - : dimension



PCA: Algorithm by SVD

- Columns of
  - is exactly the principal vectors,
  - orthogonal and has unit norm
- Matrix
  - Diagonal
  - Strength of each eigenvector
- Columns of
  - Coefficients for reconstructing the samples.

Application: Face Recognition

- Want to identify specific person, based on facial image
- Can’t just use the given 256 x 256 pixels

Applying PCA

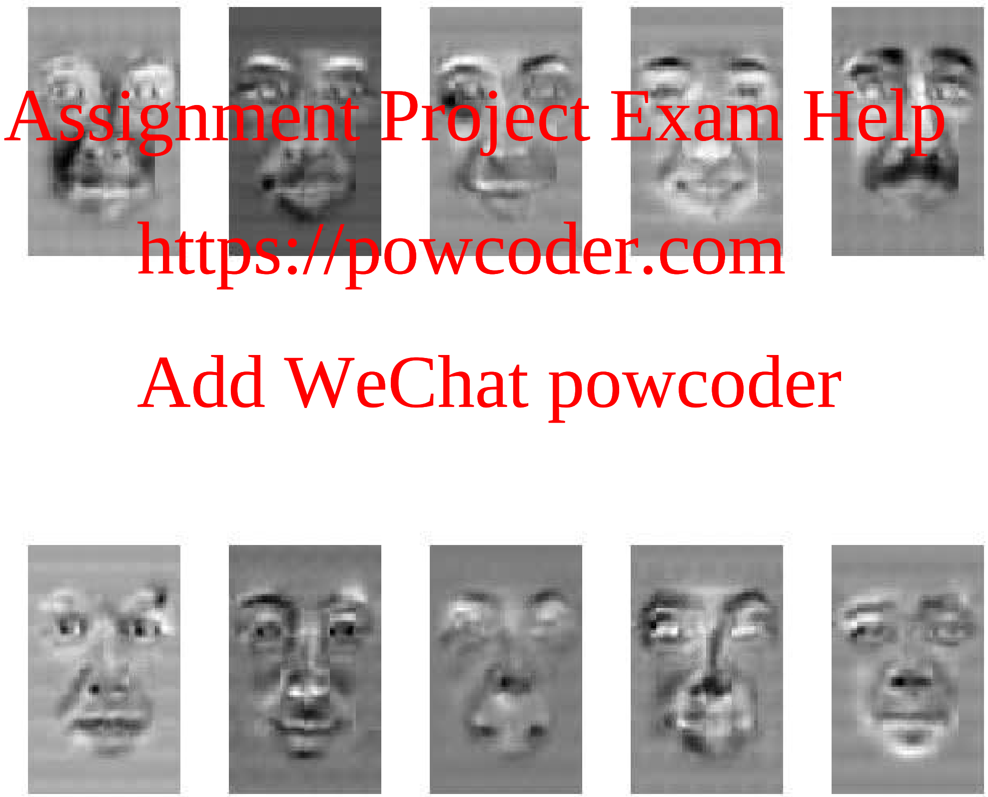
- **Method A:** Build a PCA subspace for each person and check which subspace can reconstruct the test image the best
- **Method B:** Build one PCA database for the whole dataset and then classify based on the weights.

- Example data set: Images of faces
- Each face is …
  - values

Principal Components (Method B)

Reconstructing … (Method B)

Happiness Subspace (Method A)

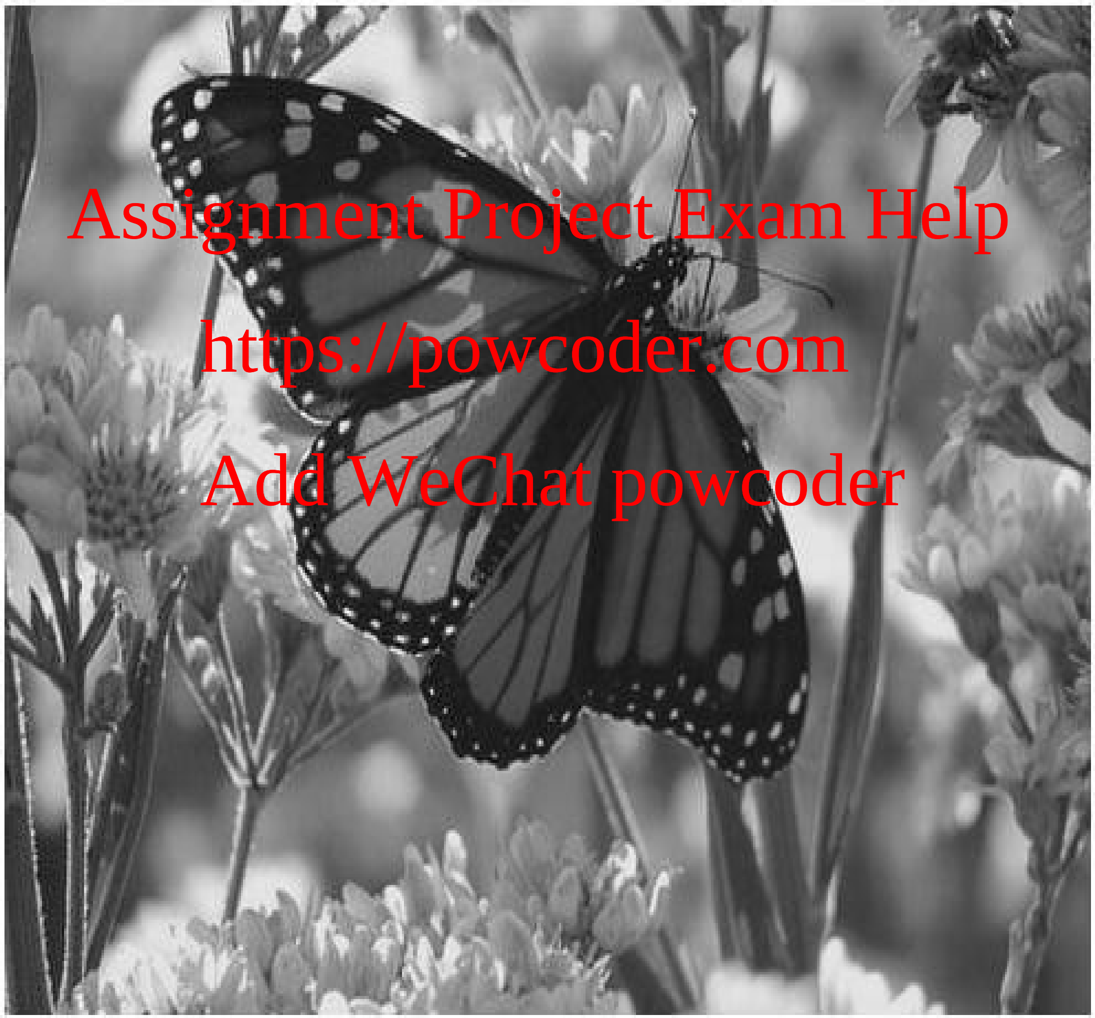


Disgust Subspace (Method A)

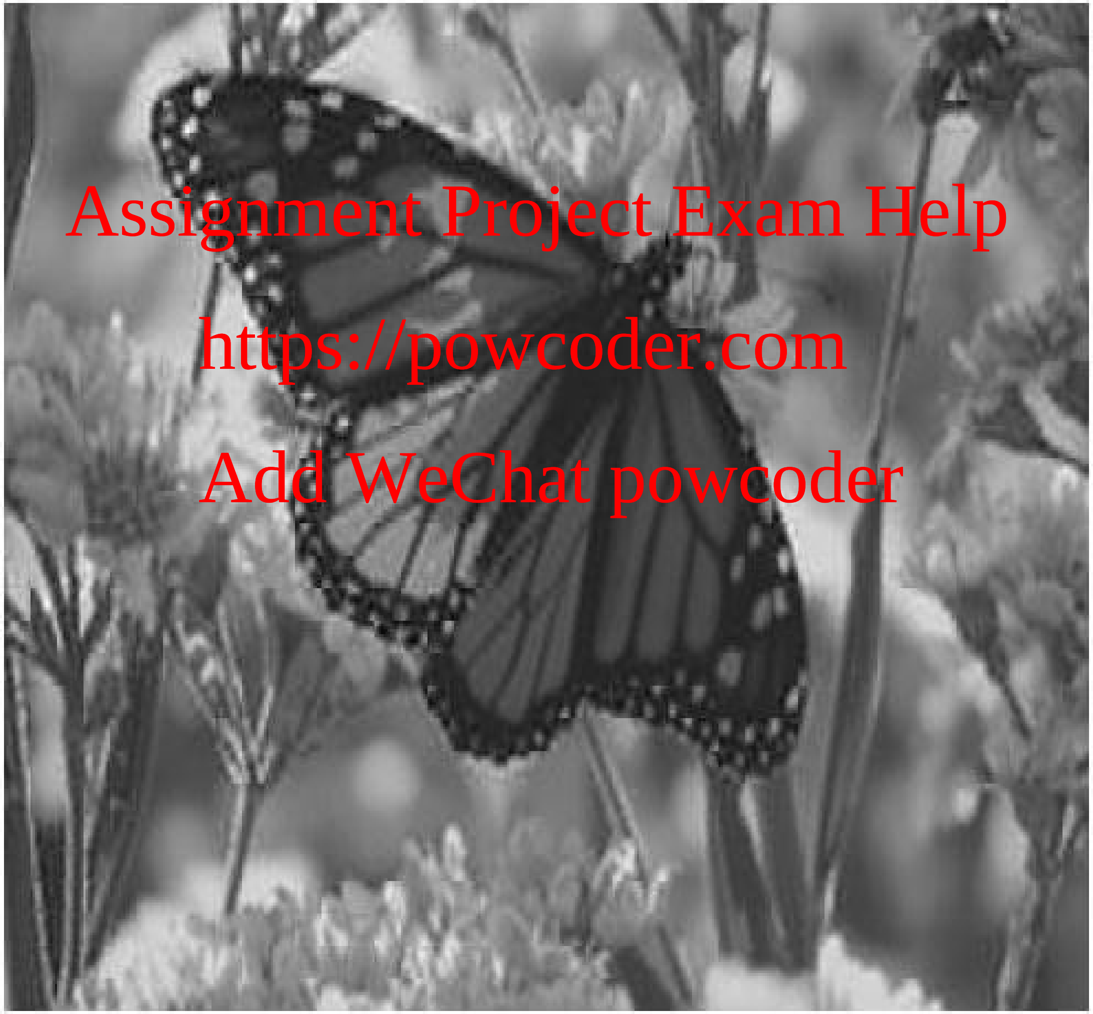
Image Compression

- Divide the original 372x492 image into patches:
  - Each patch is an instance that contains 12x12 pixels on a grid
  - View each as a 144-D vector

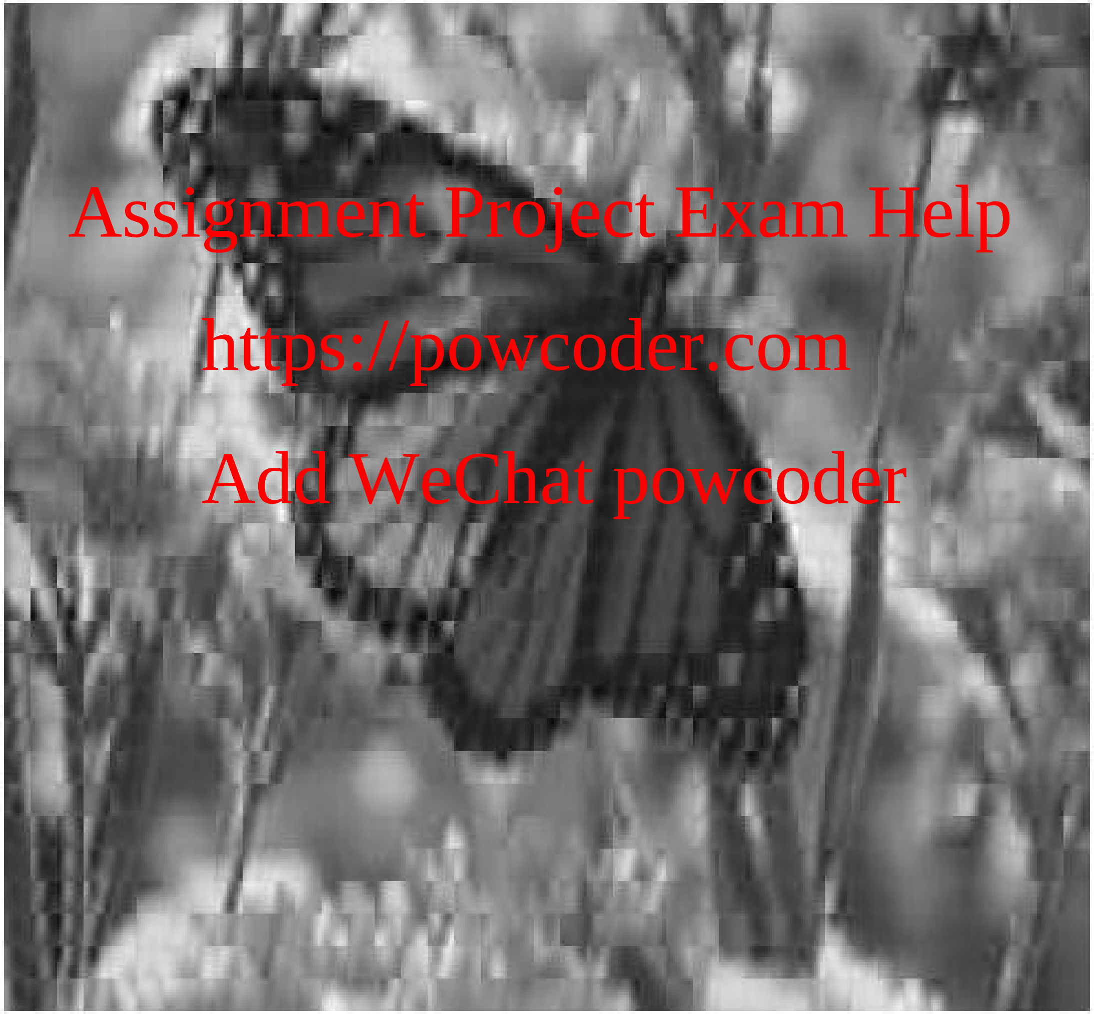
PCA Compression: 144D => 60D



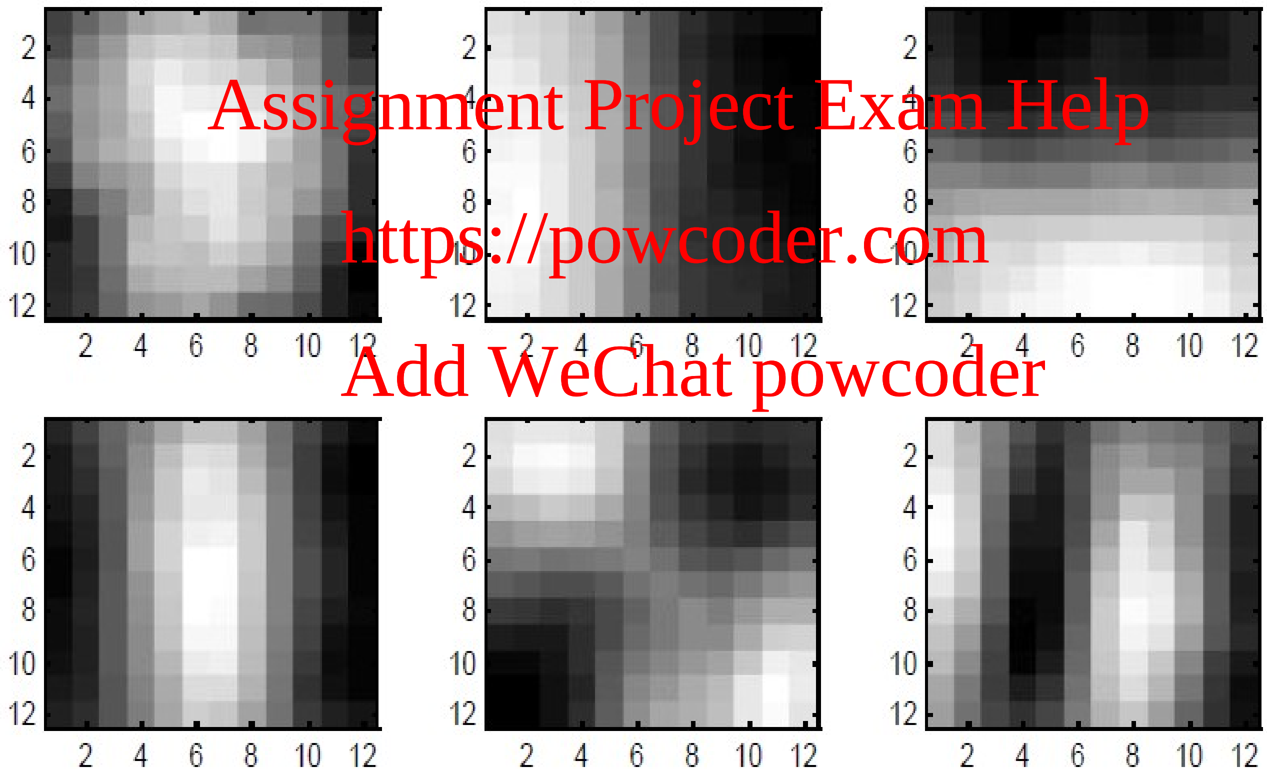
PCA Compression: 144D => 16D



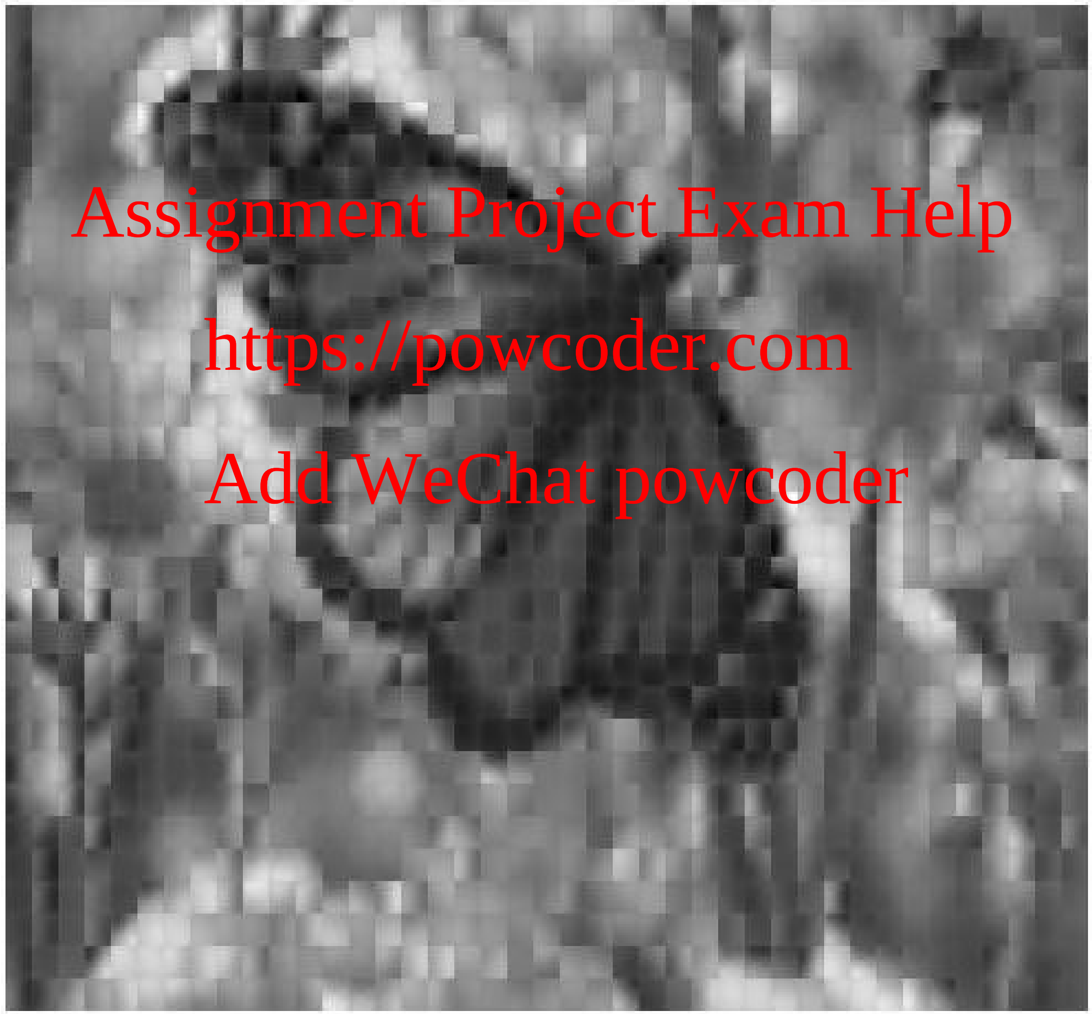
PCA Compression: 144D => 6D



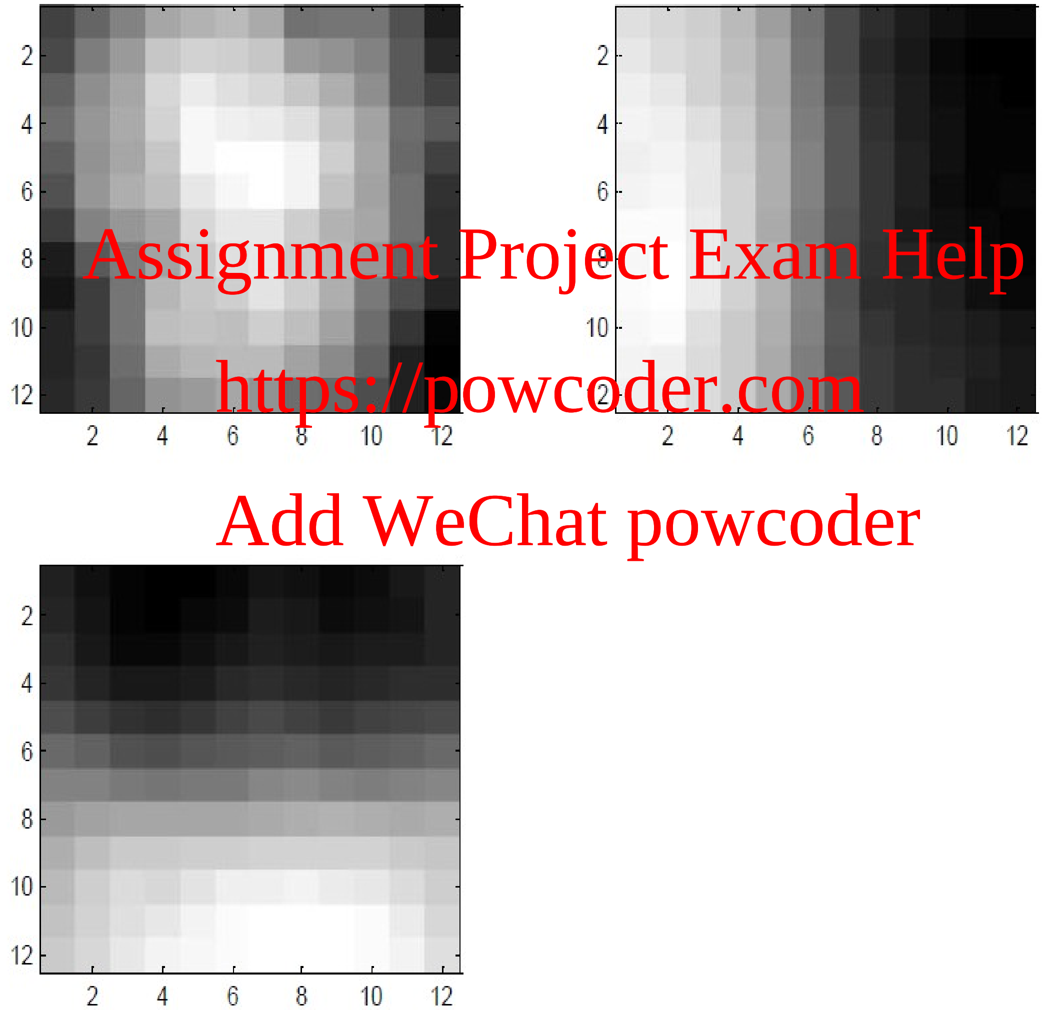
6 Most Important Eigenvectors



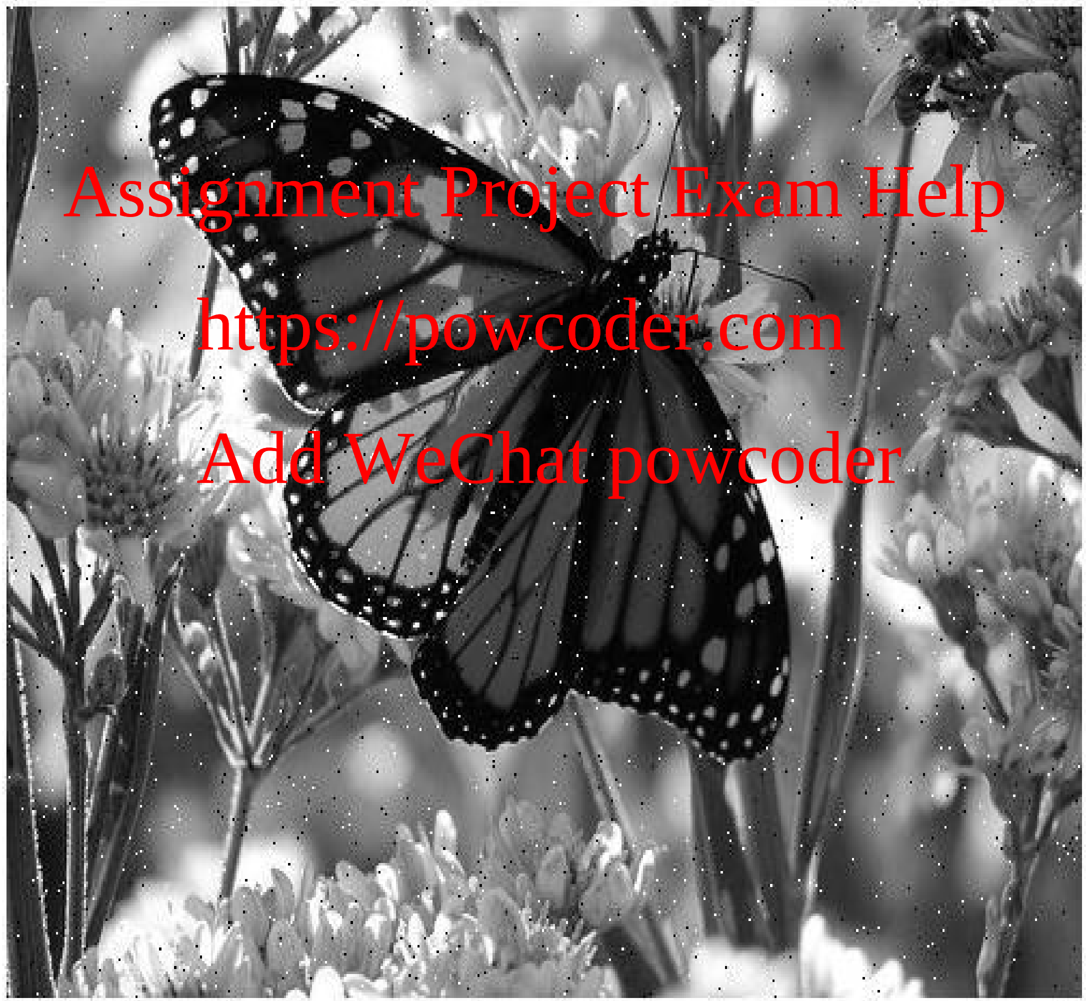
PCA Compression: 144D => 3D



3 Most Important Eigenvectors



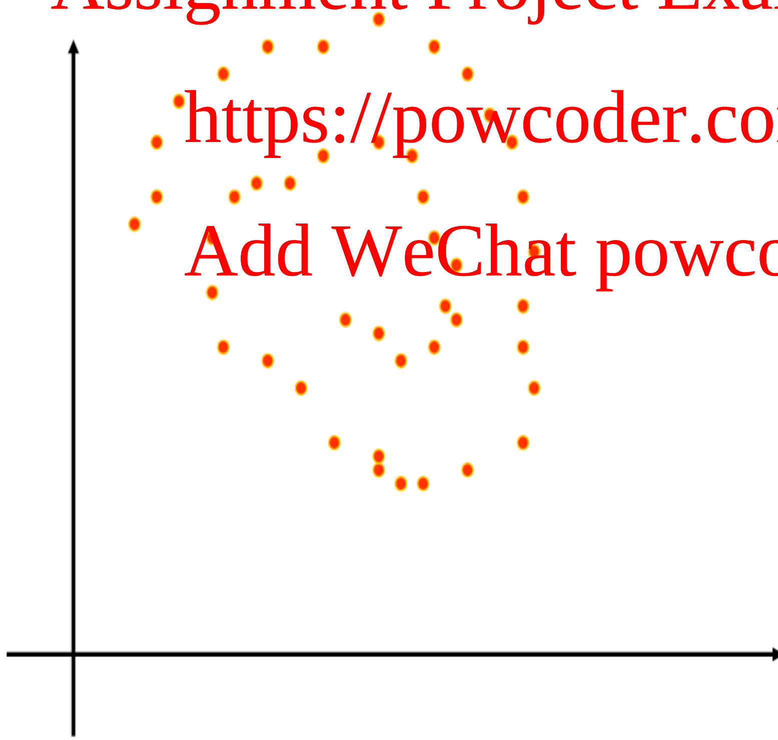
Noisy Image



Denoised Image using 15 PCA Components

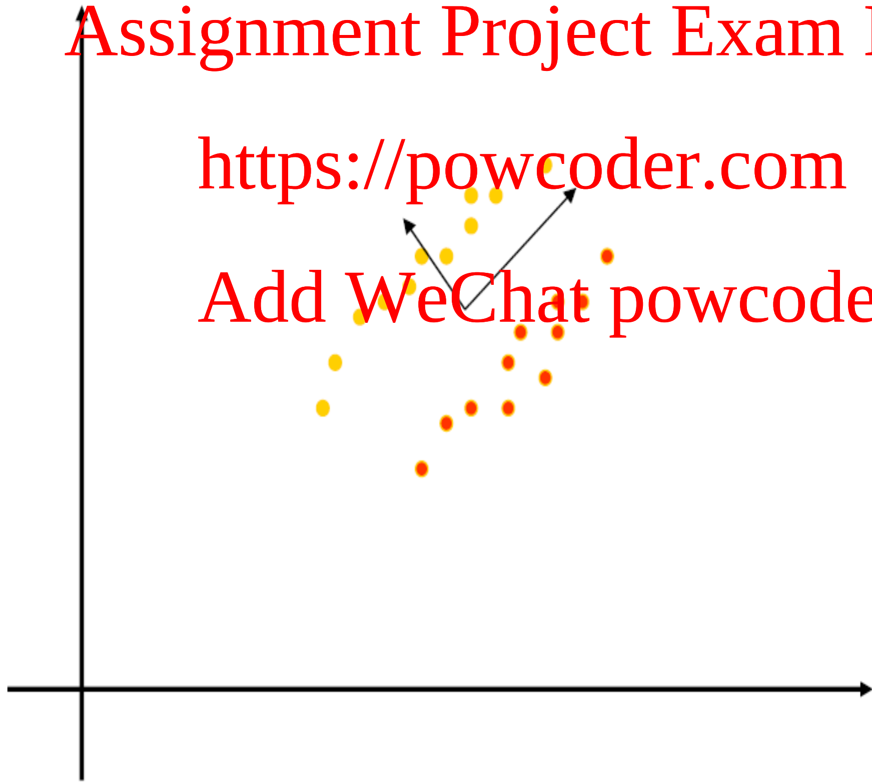
PCA: Shortcomings

- PCA cannot capture non-linear structure
  - Similar with SVD



PCA: Shortcomings

- PCA does not know labels



PCA: Conclusions

- PCA
  - find orthonormal basis for data
  - sort dimensions in order of "strength"
  - discard low significance dimensions
- Uses
  - Get compact description
  - Ignore noise
  - Improve classification (hopefully)
- Not magic:
  - Doesn't know class labels
  - Can only capture linear variations
  - One of many tricks to reduce dimensionality!

Extra: Compute PCA Using Eigen-Decomposition

- Given centered data compute covariance matrix

- Top PCA components = Top eigenvectors of .

  – Equivalence between eigen-decomposition and SVD

    - SVD decomposition of ,
      – SVD-based algorithm for PCA
    - Eigen-decomposition of .
      – Eigen-based algorithm for PCA
    - The equivalence gives .

One-slide Takeaway

- Dimensionality reduction
  - compress/reduce dimension
  - reconstruct the original matrix by two or more smaller matrices
- Singular value decomposition (SVD)
  - decompose a matrix into
    - : column-orthonormal. diagonal matrix.
- CUR decomposition
    - set of **c**olumns of . set of **r**ows of .
- Principle component analysis (PCA)
  - reconstruct data matrix by a smaller number of eigenvectors
  - view the data from a *literally* different angle.

In-class Practice

- **1.** Describe briefly (informally or formally) the relationship between singular value decomposition and eigenvalue decomposition.

- **2.1** Compute the eigenvalues and eigenvectors of matrix

- **2.2** Let It is easy to check that . What are the singular values of ?
- **2.3** Obtain SVD for A where $A = U \Sigma V^T$