The SVD
(Singular Value Decomposition)
Singular Value Decomposition

- A bit like eigendecomposition, but different
- Any matrix $A$ (no exceptions) can be decomposed as

\[
A = U \Sigma V^T
\]

- $U$ and $V$ are unitary matrices
- $\Sigma$ is a diagonal matrix with singular values $\sigma_i$
- $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \cdots \geq \sigma_m \geq 0$
- $U^TU = I$
- $V^TV = I$
Unitary Matrices: Orthonormality

\[ \mathbf{U}^\top = \begin{bmatrix} \vec{u}_1^T & \vec{u}_2^T & \vec{u}_3^T & \cdots & \vec{u}_n^T \end{bmatrix} \]

\[ \mathbf{U} = \begin{bmatrix} \vec{u}_1 & \vec{u}_2 & \vec{u}_3 & \cdots & \vec{u}_n \end{bmatrix} \]

\[ \mathbf{I} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} \]

\[ \vec{u}_1^T \vec{u}_1 = 1 \quad \vec{u}_1^T \vec{u}_2 = 0 \quad \vec{u}_1^T \vec{u}_3 = 0 \quad \cdots \quad \vec{u}_1^T \vec{u}_n = 0 \]

\[ \vec{u}_2^T \vec{u}_1 = 0 \quad \vec{u}_2^T \vec{u}_2 = 1 \quad \vec{u}_2^T \vec{u}_3 = 0 \quad \cdots \quad \vec{u}_2^T \vec{u}_n = 0 \]

\[ \vdots \]

\[ \vec{u}_n^T \vec{u}_1 = 0 \quad \vec{u}_n^T \vec{u}_2 = 0 \quad \vec{u}_n^T \vec{u}_3 = 0 \quad \cdots \quad \vec{u}_n^T \vec{u}_n = 1 \]

\[ \vec{u}_1, \vec{u}_2, \cdots, \vec{u}_n \]

called ORTHONORMAL

\[ \vec{u}_i^T \vec{u}_j = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \]

\[ \| \vec{u}_i \| = 1 \]

Similarly, \[ \vec{v}_1, \vec{v}_2, \cdots, \vec{v}_m \]

are ORTHONORMAL

\[ \| \vec{v}_j \| = 1 \]
Rank 1 Matrices and Outer Products

- Consider

$$A = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
3 & 3 & 3 & 3 & 3 \\
2 & 2 & 2 & 2 & 2
\end{bmatrix} = \begin{bmatrix} 1 \\
3 \\
2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

- rank-1 matrix can be written as $\overline{xy^T}$: an outer product

- outer product: product of col and row vectors

- rank-1: a very “simple” type of matrix
  - its “data” can be “compressed” very easily
    - can be written as outer product: $A = \overline{xy^T}$

$n+m << nm$: data compression
Matrix Multiplication using Outer Products

Example:

\[
\begin{bmatrix}
    a & b \\
    c & d \\
\end{bmatrix}
\begin{bmatrix}
    x & y & z \\
    p & q & r \\
\end{bmatrix} =
\begin{bmatrix}
    ax + bp & ay + bq & az + br \\
    cx + dp & cy + dq & cz + dr \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
    a \\
    c \\
\end{bmatrix}
\begin{bmatrix}
    x & y & z \\
\end{bmatrix} =
\begin{bmatrix}
    ax & ay & az \\
    cx & cy & cz \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
    b \\
    d \\
\end{bmatrix}
\begin{bmatrix}
    p & q & r \\
\end{bmatrix} =
\begin{bmatrix}
    bp & bq & br \\
    dp & dq & dr \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
    a & b \\
    c & d \\
\end{bmatrix}
\begin{bmatrix}
    x & y & z \\
\end{bmatrix} +
\begin{bmatrix}
    b \\
    d \\
\end{bmatrix}
\begin{bmatrix}
    p & q & r \\
\end{bmatrix}
\]

Each of these is a rank-1 outer product.

\[
\begin{bmatrix}
    y_1^T \\
    y_2^T \\
    y_3^T \\
    \vdots \\
    y_n^T \\
\end{bmatrix}

= \begin{bmatrix}
    x_1 y_1^T + x_2 y_2^T + x_3 y_3^T + \cdots + x_n y_n^T \\
\end{bmatrix}
\]
SVD: Sum of Outer Products Form

A = \sum_{i=1}^{m} \sigma_i \vec{u}_i \vec{v}_i^T

SVD splits a matrix into a weighted sum of rank-1 matrices of norm 1
Using the SVD for Image Analysis and Compression
Example: B&W Polish Flag as a Matrix

- size: 281x450

\[
A = \begin{bmatrix}
6.75 \times 10^4 \\
1 \\
1 \\
\vdots \\
\vdots \\
0.2 \\
1
\end{bmatrix}
\]

original: 3.2MB

rank=1: 58kB

\[\sigma_1 \approx 6.75 \times 10^4, \quad \sigma_2 \approx 10^{-10}\]

This is a RANK-1 FLAG

(Actual values are normalized)
Example: Polish Flag as a Matrix

- size: 281x450 (x 3 colours: R, G, B)

\[ A = \begin{bmatrix} R \ G \ B \end{bmatrix} \]

original: 10MB  
rank 1: 17.5kB

This is a \textbf{RANK-1 FLAG}

\( \sigma_1 \approx 10^{-10} \) for R, G, B

\((8.5, 6.4, 6.6) \times 10^4\)

RGB components (actual ones are normalized)

---

EE16B, Spring 2018, Lectures on SVD and PCA (Roychowdhury)
Example: SVD of the Austrian Flag

- size: 281x450 (x 3 colours: R, G, B)

\[ A = \]

original: 73MB

rank 1: 48.5kB

This is ALSO a RANK-1 FLAG

RGB components (actual ones are normalized)
Example: SVD of the Greek Flag

- size: 295x450 (x 3 colours: R, G, B)

original: 10.1MB

rank 1: 18kb
\[ \sigma_1 \tilde{u}_1 \tilde{v}_1^T \]

rank 2: 36kB
\[ \sigma_1 \tilde{u}_1 \tilde{v}_1^T + \sigma_2 \tilde{u}_2 \tilde{v}_2^T \]

rank 3: 54kB
\[ \sigma_1 \tilde{u}_1 \tilde{v}_1^T + \sigma_2 \tilde{u}_2 \tilde{v}_2^T + \sigma_3 \tilde{u}_3 \tilde{v}_3^T \]

This is a RANK-3 FLAG

\[ \sum_{i=1}^{3} \sigma_i \tilde{u}_i \tilde{v}_i^T \]
Example: SVD of the US Flag

- size: 450x237 (x 3 colours: R, G, B)

- original: 8.8MB

- rank 5: 83kB
- rank 10: 167kB
- rank 15: 253kB

Strongest "feature"
Example: SVD of Michel Maharbiz

- size: 1100x757 (x 3 colours: R, G, B)

Features not always intuitive
Michel’s Singular Values

- How Michel’s singular values drop off

Singular Values drop off rapidly in typical real-life applications

Typical resolution of eye (rule of thumb): 2 orders of magnitude below max SV

Keeping only the top few singular values (and associated cols of U, rows of $V^T$) is usually a good approximation (and requires much less data to store)
Geometric View of Orthogonality

Projection onto Orthonormal Bases

Geometric View of Unitary Operations
Geometric View of Orthogonality

- recall:
  \[ \overrightarrow{u}_i^T \overrightarrow{u}_j = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \]

- In 2D:
  - \( \overrightarrow{u}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \overrightarrow{u}_2 = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \)
  - \( \overrightarrow{u}_1^T \overrightarrow{u}_2 = 0 \) (ORTHOGONAL TO EACH OTHER)
  - \( \overrightarrow{u}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \overrightarrow{u}_2 = \frac{1}{2\sqrt{2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix} \)
  - \( \overrightarrow{u}_1^T \overrightarrow{u}_2 = -1/(2\sqrt{2}) \) (NOT ORTHOGONAL TO EACH OTHER)

- 3D: orthogonality also means at right angles
  - \( \overrightarrow{u}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \overrightarrow{u}_2 = \frac{1}{2} \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} \)
  - \( \overrightarrow{u}_1^T \overrightarrow{u}_2 = 0 \) (ORTHOGONAL TO EACH OTHER)

- 4D and higher: “right angles” means orthogonality!
Projection onto Orthonormal Bases

How can we calculate the projections?

- data point: \[ \begin{bmatrix} x \\ y \end{bmatrix} = \alpha \vec{p}_1 + \beta \vec{p}_2 \], or \[ \begin{bmatrix} x & y \end{bmatrix} = \alpha \vec{p}_1^T + \beta \vec{p}_2^T \]

- post-multiply by basis vectors: \[ \begin{bmatrix} x & y \end{bmatrix} \vec{p}_1 = \alpha \], \[ \begin{bmatrix} x & y \end{bmatrix} \vec{p}_2 = \beta \]

\[ \begin{bmatrix} x \beta \\ \alpha \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} B_2 \]; or, for all the data \[ D_2 = \begin{bmatrix} \alpha_1 & \beta_1 \\ \alpha_2 & \beta_2 \\ \alpha_3 & \beta_3 \end{bmatrix} = DB_2 \]

Samples

\[ D = \begin{bmatrix} 1 & 1.5 \\ -2 & 1 \\ -1 & -0.5 \end{bmatrix} \]

Orthonormal basis

\[ B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \]

Another orthonormal basis

\[ B_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \]

\[ \| \vec{p}_i \| = 1 \]

\[ \vec{p}_1^T \vec{p}_2 = 0 \]
Using the SVD for Data Analysis, Feature Extraction and Clustering
## Matrices Representing Ratings

- **Movies rated by Users** (eg, Netflix, Amazon Video)

<table>
<thead>
<tr>
<th>Movie → User Name</th>
<th>Full Metal Jacket</th>
<th>Die Hard</th>
<th>Yojimbo</th>
<th>2001: A Space Odyssey</th>
<th>The Quiet Earth</th>
<th>On The Beach</th>
<th>Would I Lie to You</th>
<th>Dr. Strangelove</th>
<th>Hokkabaz</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Lighter colours = stronger ratings

\[
= U \Sigma V^T
\]
### Features of Rating Matrices

<table>
<thead>
<tr>
<th>Movie → User Name</th>
<th>Full Metal Jacket</th>
<th>Die Hard</th>
<th>Yojimbo</th>
<th>2001: A Space Odyssey</th>
<th>The Quiet Earth</th>
<th>On The Beach</th>
<th>Would I Lie to You</th>
<th>Dr. Strangelove</th>
<th>Hokkabaz</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**“most typical” user feature:**
- 65% are like D, 50% are like A,
- 40% are like B, 35% are like C,
- 20% like E

**“most typical” movie feature:**
- more SF;
- rather less action;
- even less comedy

\[
\sigma_1 \vec{u}_1 \vec{v}_1^T
\]
### Features of Rating Matrices (contd.)

<table>
<thead>
<tr>
<th>Movie →</th>
<th>Full Metal Jacket</th>
<th>Die Hard</th>
<th>Yojimbo</th>
<th>2001: A Space Odyssey</th>
<th>The Quiet Earth</th>
<th>On The Beach</th>
<th>Would I Lie to You</th>
<th>Dr. Strangelove</th>
<th>Hokkabaz</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

2\textsuperscript{nd} most typical user feature: 55\% are like A, 70\% unlike B, 35\% are like C, 15\% unlike C negligibly like E

2\textsuperscript{nd} most typical movie feature: more action; less SF; strongly anti-comedy
### Projection in the Feature Basis

**Express each col of A (movie column) as a linear combination of user features**

**e.g., Full Metal Jacket column:**

\[
\alpha_{i1} = \hat{u}_i^T \begin{bmatrix}
5 \\
1 \\
2 \\
5 \\
1
\end{bmatrix}, \quad i = 1, \ldots, 5
\]

#### Table

<table>
<thead>
<tr>
<th>Movie → User Name ↓</th>
<th>Full Metal Jacket</th>
<th>Die Hard</th>
<th>Yojimbo</th>
<th>2001: A Space Odyssey</th>
<th>The Quiet Earth</th>
<th>On The Beach</th>
<th>Would I Lie to You</th>
<th>Dr. Strangelove</th>
<th>Hokkabaz</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**Projection of FMJ col. onto user feature basis**

\[
\begin{bmatrix}
5 \\
1 \\
2 \\
5 \\
1
\end{bmatrix} = \alpha_{11} \hat{u}_1 + \alpha_{21} \hat{u}_2 + \alpha_{31} \hat{u}_3 + \cdots + \alpha_{51} \hat{u}_5
\]

**"how much the most typical user likes FMJ"**

**"how much the 3rd most typical user likes FMJ"**
Clustering in Feature Bases

- Scatter plot of $\alpha_{11}$, $\alpha_{21}$, and $\alpha_{31}$ for all movies

Movies classified by user features

Users classified by movie features

```
user E row = [1 2 1 2 1 2 2 2 1]
= $\beta_{51}v_1^T + \beta_{52}v_2^T + \beta_{53}v_3^T + \cdots + \beta_{59}v_9^T$
```
Principal Component Analysis (PCA)
Covariance Matrices

- Each col has average 0
- Each col is a type of data (eg: position, velocity)

\[ \tilde{A} \]
\[ \text{data matrix} \]
\[ \text{A} \quad \text{(assumed real)} \]
\[ \frac{1}{n} \]

\[ S \]
\[ \text{(mxm matrix)} \]
\[ \tilde{A}^T \]
\[ \tilde{A} \]

\[ \text{covariance matrix of A} \]

\[ \text{each row is a measurement (sample)} \]
\[ \text{mean of col 1} \]
\[ \text{row of col means of A} \]

\[ \text{subtract row-of-means from EACH row of A} \]
Covariance Matrices: Properties

- $S \triangleq \frac{1}{n} \bar{A}^T \bar{A}$
- $S$ is square and symmetric: $S = S^T$ or $s_{ij} = s_{ji}$
- The diagonal entries of $S$ are real and $\geq 0$
- $s_{ii}^2 \triangleq s_{ii} = \frac{1}{n} \sum_{j=1}^{n} \bar{a}_{ij}^2 \geq 0$: variance of $i^{th}$ row of $A$

$S = \begin{bmatrix}
    s_1^2 & s_{12} & s_{13} & \cdots & s_{1m} \\
    s_{21} & s_2^2 & s_{23} & \cdots & s_{2m} \\
    s_{31} & s_{32} & s_3^2 & \cdots & s_{3m} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    s_{m1} & s_{m2} & s_{m3} & \cdots & s_m^2
\end{bmatrix}$

- can also show: $|s_{ij}| \leq s_i s_j$
  - using the Cauchy-Schwartz inequality
The Correlation Matrix

- \( r_{ij} \triangleq \frac{S_{ij}}{s_i s_j} \); \( r_{ij} = r_{ji} \) (symmetry);

\[
R = \begin{bmatrix}
1 & r_{12} & r_{13} & \cdots & r_{1m} \\
r_{21} & 1 & r_{23} & \cdots & r_{2m} \\
r_{31} & r_{32} & 1 & \cdots & r_{3m} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
r_{m1} & r_{m2} & r_{m3} & \cdots & 1
\end{bmatrix}
\]

\( \Rightarrow r_{ii} = 1 \)
\( \Rightarrow |r_{ij}| \leq 1 \)

why?
Correlation: Geometric Intuition

$r_{12} = 0$

$r_{12} = 0.29$

$r_{12} = 0.51$

$r_{12} = 0.71$

$r_{12} = 0.87$

$r_{12} = 0.96$

$r_{12} = 0.56$

$r_{12} = 0.8$

$r_{12} = 0.17$

Correlation provides some insight ... but it leaves a lot out

5000 x 2 matrices (each point is a row)
The Intuition behind PCA

- PCA: finds (orthogonal) “main axes along which the data lie”: the principal components
- provides weights indicating “strength” of each axis

- starting point for PCA: the covariance matrix S
PCA: The Procedure

- Eigendecompose the covariance matrix

\[ S = \begin{bmatrix}
  s_{11}^2 & s_{12} & s_{13} & \cdots & s_{1m} \\
  s_{21} & s_{22}^2 & s_{23} & \cdots & s_{2m} \\
  s_{31} & s_{32} & s_{33}^2 & \cdots & s_{3m} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  s_{m1} & s_{m2} & s_{m3} & \cdots & s_{mm}^2
\end{bmatrix} = P\Lambda P^{-1} \]

- \[ \sqrt{\lambda_i} \] = the weights (i.e., the variances)
- with \[ \lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_m \geq 0 \]
- eigenvectors \( \vec{p}_i \) = the principal components

Cols will be orthonormal

These are the PCs
Principal Components of the Data

First PC always captures direction of maximum data spread
(2nd PC: max spread in orthogonal direction)
PCA: Why it Works: The Flow

- First: establish some properties of $P$ and $\Lambda$
  - properties of real symmetric matrices
    - real eigenvalues
    - real set of orthonormal eigenvectors
  - properties of real $A^T A$
    - eigenvalues $\geq 0$
- Express data in eigenvector basis
  - project each data point onto eigenvectors
- Show that the covariance matrix of the projected data is diagonal
  - the variances of the projections along each axis/PC
- First PC maximizes variance along any 1D projection
  - 2nd PC maximizes remaining variance; and so on
Properties of Covariance Matrices

- If $S$ is a real $m \times m$ symmetric matrix ($s_{ij} = s_{ji}$)
  - 1. Its eigenvalues are all real
    - $S \vec{p} = \lambda \vec{p}$. $S$ symmetric $\rightarrow \vec{p}^T S = \lambda \vec{p}^T \vec{p} \rightarrow \vec{p}^T S \vec{p} = \lambda \vec{p}^T \vec{p} = \lambda \| \vec{p} \|^2$
    - $S$ real $\rightarrow S \vec{p} = \overline{\lambda} \vec{p} \rightarrow \vec{p}^T S \vec{p} = \overline{\lambda} \vec{p}^T \vec{p} = \overline{\lambda} \| \vec{p} \|^2$.
    - Hence $\lambda \| \vec{p} \|^2 = \overline{\lambda} \| \vec{p} \|^2 \rightarrow \lambda = \overline{\lambda} \rightarrow \lambda$ is real.
  - 2. A set of real eigenvectors can be found (see the notes)
  - 3. The eigenvectors form an orthonormal set (basis).
    - (see the notes)

- If $S$ is in the form $A^T A$ (A real)
  - 4. Its eigenvalues are all $\geq 0$.
    - $A^T A \vec{p} = \lambda \vec{p} \rightarrow \vec{p}^T A^T A \vec{p} = \lambda \vec{p}^T \vec{p} \rightarrow (A \vec{p})^T A \vec{p} = \lambda \vec{p}^T \vec{p}$
    - $\rightarrow \| A \vec{p} \|^2 = \lambda \| \vec{p} \|^2 \rightarrow \lambda = \frac{\| A \vec{p} \|^2}{\| \vec{p} \|^2} \geq 0$. 

EE16B, Spring 2018, Lectures on SVD and PCA (Roychowdhury)
PCA Basis Diagonalises the Data

- eigenvectors orthonormal $\rightarrow PP^T = I \rightarrow P^T = P^{-1}$
- eigendecomposition of $S$: $S = P\Lambda P^T$
- project rows of (zero-mean) $A$ in basis $P$: $F = \tilde{A}P$
  - columns of $F$ are the projections along $\vec{p}_i$
- Let $G$ be the co-variance matrix of $F$: $G \triangleq (F^TF)/n$
  - $nG = F^TF = P^T\tilde{A}^T\tilde{A}P = nP^TS\tilde{P} = nP^T\Lambda P^TP = n\Lambda$
  - $= \Lambda$ (diagonal)
- the diagonal entries are the variances of the data projected along $\vec{p}_i$ (recall: from defn. of covariance matrix)

Data projected on PC basis becomes UNCORRELATED
Why do PCs Align with Visual Axes?

So far: have shown that PCs are orthonormal
- data projected onto them becomes uncorrelated
- but why is the first PC aligned with the direction of maximum spread?

Key property of PCA
- consider any norm-1 vector (“direction”) \( \hat{p} \)
- project the data along it: \( \hat{A}\hat{p} \)
- find the variance of the projected data: \( \frac{1}{n} (\hat{A}\hat{p})^T (\hat{A}\hat{p}) \)
- the first PC \( \hat{p}_1 \) maximizes this variance (the max is \( \hat{\lambda}_1 \))

\( \rightarrow \) 2\( \text{nd} \) PC: maximizes variance along directions orthogonal to \( \hat{p}_1 \)
\( \rightarrow \) 3\( \text{rd} \) PC: maximizes var. along dirs. orthogonal to \( \hat{p}_1 \) and \( \hat{p}_2 \); and so on
PCA: the Connection with the SVD

• Suppose you run an SVD on the data: 
\[ \tilde{A} = U \Sigma V^T \]

• the covariance matrix is:
\[ S \triangleq \frac{1}{n} \tilde{A}^T \tilde{A} = \frac{1}{n} V \Sigma^T U^T U \Sigma V^T = V \frac{\Sigma^T \Sigma}{n} V^T \]

• recall PCA:
\[ S = P \Lambda P^T \]

• i.e., can use the SVD of \( \tilde{A} \) for PCA:

\[ \text{just set } \lambda_i \triangleq \frac{\sigma_i^2}{n} \text{ and } P \triangleq V \] (no need to even form \( S \))!
Computing SVDs via Eigendecomposition

- Prev. slide: SVD: \( S = V \frac{\sum T \sum}{n} V^T \); PCA: \( S = P \Lambda P^T \)

- Q: how to calculate an SVD of a matrix \( A \)?
  - using eigendecomposition

- A: just use the above insight (PCA/eigendecomposition)!

- form \( S \triangleq A^T A \), eigendecompose \( S = P \Lambda P^T \)

- set \( \sigma_i \triangleq \sqrt{\lambda_i} \), \( V \triangleq P \)

- what about \( U \)?
  - just eigendecompose \( \hat{S} \triangleq A A^T = Q \hat{\Lambda} Q^T \); then \( U \triangleq Q \)
  - can also get \( V \) from the same eigendecomposition

- \( A = U \Sigma V^T \rightarrow U^T A = \Sigma V^T \rightarrow A^T U = V \Sigma^T \rightarrow \)

- set \( \sigma_i \triangleq \sqrt{\lambda_i} \)

* why didn’t we subtract means from \( A \) and normalize by \( n \)?

\* if \( \sigma_i = 0 \), choose \( v_i \) arbitrarily to complete orthonormal basis for \( V \)
Who Invented the SVD?

- SVD: "Swiss Army Knife" of numerical analysis

- Eugenio Beltrami (1835-1900) proposed the SVD via eigendecomposition of $A^T A$ or $A A^T$
- Camille Jordan (1838-1922)
- Erhardt Schmidt (1878-1959)
- James Joseph Sylvester (1814-97)
- Hermann Weyl (1885-1955)
- Gene Golub (1932-2007)
- Bill Kahan (UCB EECS)
- Jim Demmel (UCB EECS)
Summary: SVD and PCA

• **Singular Value Decomposition (SVD)**
  • useful for “low-rank approximations” of matrices
    - image analysis and compression
    - general data analysis, finding important features, clustering

• **Covariance, Correlation and PCA**
  • visualizing data as scatter plots
  • covariance and correlation matrices of data
  • Principal Component Analysis
    - eigenvvecs of covariance matrix: principal components
      - directions along which data varies maximally
        - dropping later PCs can, eg, clean out (small) noise
    - eigenvalues correspond to variances along PCs
    - SVD can be used instead of eigendecomposition
      - eigendecomposition of covariance matrix: performs SVD