Randomized SVD and its Applications

Katie Keegan, David Melendez, Jennifer Zheng

July 31, 2020
The Singular Value Decomposition

- An incredibly important matrix decomposition in linear algebra
- Has applications in many different domains

What we will cover

- Image, video, audio processing
- Data analysis
- Digital ownership protection
Singular Value Decomposition

\[ A_{m \times n} = U_{m \times m} \Sigma_{m \times n} V^T_{n \times n} \]

U and V are orthogonal matrices, \( \Sigma \) is a diagonal matrix with positive diagonal entries \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \), and \( r = \text{rank}(A) \).
Singular Value Decomposition

Eckart-Young Theorem

If $B$ has rank $k$ then $\|A - B\| \geq \|A - A_k\|

- $A = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \sigma_3 u_3 v_3^T + ... + \sigma_r u_r v_r^T$
- $A_k$ is the first $k$ matrices added together for $k < r$
- The closest rank $k$ matrix to $A$ is $A_k$

Example

$$A = \begin{bmatrix} | & | & | \\ u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \begin{bmatrix} \hdashline \vdash \vdash \\ - & v_1^T & - \\ - & v_2^T & - \\ - & v_3^T & - \end{bmatrix}$$
Singular Value Decomposition

Eckart-Young Theorem

If $B$ has rank $k$ then $\|A - B\| \geq \|A - A_k\|

$A = U\Sigma V^\top = \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \sigma_3 u_3 v_3^\top + \ldots + \sigma_r u_r v_r^\top$

$A_k$ is the first $k$ matrices added together for $k < r$

The closest rank $k$ matrix to $A$ is $A_k$

Outer Product Form

$A = \begin{bmatrix}
| & | & |
\end{bmatrix}
\begin{bmatrix}
\sigma_1 & 0 & 0 \\
0 & \sigma_2 & 0 \\
0 & 0 & \sigma_3 \\
\end{bmatrix}
\begin{bmatrix}
| & | & |
\end{bmatrix}
= \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \sigma_3 u_3 v_3^\top$

Katie Keegan, David Melendez, Jennifer Zheng
Singular Value Decomposition

Eckart-Young Theorem

If $B$ has rank $k$ then $\|A - B\| \geq \|A - A_k\|

- $A = U\Sigma V^\top = \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \sigma_3 u_3 v_3^\top + \ldots + \sigma_r u_r v_r^\top$
- $A_k$ is the first $k$ matrices added together for $k < r$
- The closest rank $k$ matrix to $A$ is $A_k$

Truncating...

$$A = \begin{bmatrix} | & | & | \\ u_1 & u_2 & u_3 \\ | & | & | \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \begin{bmatrix} | & | & | \\ v_1^\top & | & | \\ v_2^\top & | & | \\ v_3^\top & | & | \end{bmatrix}$$

$$= \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \sigma_3 u_3 v_3^\top$$
Singular Value Decomposition

Eckart-Young Theorem

If $B$ has rank $k$ then $\|A - B\| \geq \|A - A_k\|

- $A = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \sigma_3 u_3 v_3^T + ... + \sigma_r u_r v_r^T$
- $A_k$ is the first $k$ matrices added together for $k < r$
- The closest rank $k$ matrix to $A$ is $A_k$

Rank 2 Approximation

$$A_2 = \begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \end{bmatrix} = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T$$
Randomized SVD

Randomized SVD Algorithm

- Uses a random projection matrix to sample the column space of the original matrix
- Allows us to approximate the SVD of the original matrix by computing SVD on smaller matrix
Randomized SVD

Step 1

\[
X = P = Z = Q R
\]

Step 2

\[
Q^T X = Y = U_Y \Sigma V^T
\]

\[
U = Q U_Y
\]

Figure from *Data-Driven Science and Engineering* by Steven Brunton and Nathan Kutz

Katie Keegan, David Melendez, Jennifer Zheng
Converting Color Images to Matrices

Color Stacking

- Just like how a black-and-white image can be represented as a matrix of values between 0 and 1, color images can be represented as the combination of three matrices (color channels).
- We can take these channels apart, stack them, and then compute their SVD.

Katie Keegan, David Melendez, Jennifer Zheng
Color Stacking

Just like how a black-and-white image can be represented as a matrix of values between 0 and 1, color images can be represented as the combination of three matrices (color channels).

We can take these channels apart, stack them, and then compute their SVD.
Converting Color Images to Matrices

Color Stacking

- Just like how a black-and-white image can be represented as a matrix of values between 0 and 1, color images can be represented as the combination of three matrices (color channels).
- We can take these channels apart, stack them, and then compute their SVD.
Converting Color Images to Matrices

Color Stacking

- Just like how a black-and-white image can be represented as a matrix of values between 0 and 1, color images can be represented as the combination of three matrices (color channels).
- We can take these channels apart, stack them, and then compute their SVD.

Katie Keegan, David Melendez, Jennifer Zheng
Image Compression

Original

SV Log Plot

Rank 50

Rank 25

Rank 10

Rank 5

Deterministic

Randomized

Katie Keegan, David Melendez, Jennifer Zheng
Modifying Singular Values

\[ A = U\Sigma V^T = \begin{bmatrix} | & | & | & | \ \\ u_1 & \ldots & u_k & | \\
| & | & | & | \ \\
\sigma_1 & & & \cdot \\
| & | & | & | \ \\
| & | & | & | \ \\
\sigma_k & & & v_k \\
| & | & | & | \ \\
| & | & | & | \ \\
- & v_1 & - & - \\
\cdot & - & - \\
\cdot & - & - \\
\cdot & - & - \\
\end{bmatrix} \]
Modifying Singular Values

\[ A = U\Sigma V^T = \begin{bmatrix} u_1 & \ldots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \begin{bmatrix} - & v_1 & - \\ & \vdots & \\ - & v_k & - \end{bmatrix} \]

\[ \Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \]
Modifying Singular Values

What happens if we try to modify the singular values?

\[ A = U\Sigma V^T = \begin{bmatrix} | & | & | \\ u_1 & \ldots & u_k \\ | & | & | \end{bmatrix} \begin{bmatrix} \sigma_1 & & \cdot \cdot \cdot \\ & \ddots & \vdots \\ & & \sigma_k \end{bmatrix} \begin{bmatrix} - & v_1 & - \\ & \ddots & \vdots \\ & & - v_k & - \end{bmatrix} \]

\[ \Sigma = \begin{bmatrix} \sigma_1 & \cdot \cdot \cdot \\ & \ddots \\ & & \sigma_k \end{bmatrix} \]
Modifying Singular Values

What happens if we try to modify the singular values?

\[
A = U\Sigma V^T = \begin{bmatrix}
    u_1 & \ldots & u_k \\
\end{bmatrix}
\begin{bmatrix}
    \sigma_1 & & \\
    & \ddots & \\
    & & \sigma_k \\
\end{bmatrix}
\begin{bmatrix}
    v_1 & \ldots \\
    & \ddots \\
    & & v_k \\
\end{bmatrix}
\]

\[
\Sigma = \begin{bmatrix}
    \sigma_1 & & \\
    & \ddots & \\
    & & \sigma_k \\
\end{bmatrix}
\]

Multiplying by some scalar \( c \):

\[
\begin{bmatrix}
    \sigma_1 \\
    \vdots \\
    \sigma_k \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
    c\sigma_1 \\
    \vdots \\
    c\sigma_k \\
\end{bmatrix}
\]

Katie Keegan, David Melendez, Jennifer Zheng
Modifying Singular Values

What happens if we try to modify the singular values?

\[ A = U \Sigma V^T = \begin{bmatrix} | & & | \\ u_1 & \ldots & u_k \end{bmatrix} \begin{bmatrix} \sigma_1 & & \end{bmatrix} \begin{bmatrix} \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_k \end{bmatrix} & \begin{bmatrix} - & v_1 & - \\ \vdots & \ddots & \vdots \\ - & v_k & - \end{bmatrix} \end{bmatrix} \]

\[ \Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \]

Multiplying by some scalar \( c \)

\[ \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \begin{bmatrix} | & & | \\ u_1 & \ldots & u_k \end{bmatrix} \begin{bmatrix} c\sigma_1 & & \end{bmatrix} \begin{bmatrix} \begin{bmatrix} c\sigma_1 \\ \vdots \\ c\sigma_k \end{bmatrix} & \begin{bmatrix} - & v_1 & - \\ \vdots & \ddots & \vdots \\ - & v_k & - \end{bmatrix} \end{bmatrix} = ? \]
Modifying Singular Values

\[
\begin{bmatrix}
\sigma_1 \\
\vdots \\
\sigma_k
\end{bmatrix}
\text{Multiplying by some scalar } c
\begin{bmatrix}
c\sigma_1 \\
\vdots \\
c\sigma_k
\end{bmatrix}
\]
Modifying Singular Values

\[
\begin{bmatrix}
\sigma_1 \\
\vdots \\
\sigma_k
\end{bmatrix}
\] Multiplying by some scalar \( c \)

\[
\begin{bmatrix}
\sigma_1 \\
\vdots \\
\sigma_k
\end{bmatrix}
\]
Modifying Singular Values

\[
\begin{bmatrix}
\sigma_1 \\
\ddots \\
\sigma_k
\end{bmatrix}
\xrightarrow{\text{Adding some scalar } m}
\begin{bmatrix}
\sigma_1 + m \\
\ddots \\
\sigma_k + m
\end{bmatrix}
\]
Modifying Singular Values

\[ \begin{bmatrix} \sigma_1 & \cdots & \sigma_k \\ \sigma_1 & \cdots & \sigma_k \end{bmatrix} \xrightarrow{\text{Adding some scalar } m} \begin{bmatrix} \sigma_1 + m & \cdots \\ \cdots & \sigma_k + m \end{bmatrix} \]

\[ \begin{bmatrix} \sigma_1 & \cdots & \sigma_k \\ \sigma_1 & \cdots & \sigma_k \end{bmatrix} \xrightarrow{\text{Raising to some exponent } n} \begin{bmatrix} \sigma_1^n & \cdots \\ \cdots & \sigma_k^n \end{bmatrix} \]
Modifying Singular Values

\[
\begin{bmatrix}
\sigma_1 \\
\vdots \\
\sigma_k
\end{bmatrix}
\xrightarrow{\text{Adding some scalar } m}
\begin{bmatrix}
\sigma_1 + m \\
\vdots \\
\sigma_k + m
\end{bmatrix}
\]

\[
\begin{bmatrix}
\sigma_1 \\
\vdots \\
\sigma_k
\end{bmatrix}
\xrightarrow{\text{Raising to some exponent } n}
\begin{bmatrix}
\sigma_1^n \\
\vdots \\
\sigma_k^n
\end{bmatrix}
\]

\[
\begin{bmatrix}
\sigma_1 \\
\vdots \\
\sigma_k
\end{bmatrix}
\xrightarrow{\text{Applying logarithmic mapping}}
\begin{bmatrix}
\log(\sigma_1 + 1)^p \\
\vdots \\
\log(\sigma_k + 1)^p
\end{bmatrix}
\]

Can also be applied to audio, video matrices
### Video → Matrix
- Video is a sequence of pictures
- reshape each frame of picture as a long matrix
- reshape a video into a skinny tall matrix

### Low rank approximations of surveillance video
- For rank $r \leq 10$, only the most dominant features in every frame of image is captured
- The lower the rank, the less moving objects captured
- More blurry for shaky videos
Video Background Extraction

Katie Keegan, David Melendez, Jennifer Zheng
Audio Compression

Representing Audio as a Signal

- Audio is represented as a waveform - a function of wave height over time.
- On the computer, we need to represent them discretely by **sampling** the waveform in fixed time steps.
Audio is represented as a waveform - a function of wave height over time.

On the computer, we need to represent them discretely by sampling the waveform in fixed time steps.
Audio is represented as a waveform - a function of wave height over time.

On the computer, we need to represent them discretely by **sampling** the waveform in fixed time steps.
Audio Compression

Fourier Transform
- Represents the average frequency content of a signal over its duration
- The Fourier transform gives us another 1D array representing the weight of each frequency in the overall signal

Short-Time Fourier Transform
- Takes the Fourier transform of small “windows” of the signal
- Puts the resulting spectra in columns of a matrix
Short-Time Fourier Transform

Katie Keegan, David Melendez, Jennifer Zheng
Short-Time Fourier Transform
Short-Time Fourier Transform

Katie Keegan, David Melendez, Jennifer Zheng
Short-Time Fourier Transform
Low-rank Approximation of Audios

Rank 118  
Rank 25  
Rank 5  

Katie Keegan, David Melendez, Jennifer Zheng
Multiply $\sigma$ by some scalar $p$.
Singular Value Modification on Audios

Raise $\sigma$ to some exponent $n$

- $n = 2$
- $n = 1.5$
- $n = 0.7$
## Data Analysis

### USA Facts Data Set

- Provides data on cumulative new deaths and cases reported for each county in the United States since January 22.
- Can be reformatted to provide information about new deaths/cases reported per day, per state, etc.
- Relies on daily government-reported data, so cumulative numbers fluctuate (some days have negative numbers).

### Snippet of Data

<table>
<thead>
<tr>
<th>State</th>
<th>County</th>
<th>7/1/20</th>
<th>7/2/20</th>
<th>7/3/20</th>
<th>7/4/20</th>
<th>7/5/20</th>
</tr>
</thead>
<tbody>
<tr>
<td>FL</td>
<td>Alachua</td>
<td>1245</td>
<td>1332</td>
<td>1423</td>
<td>1506</td>
<td>1578</td>
</tr>
<tr>
<td>FL</td>
<td>Baker</td>
<td>72</td>
<td>80</td>
<td>84</td>
<td>98</td>
<td>105</td>
</tr>
<tr>
<td>FL</td>
<td>Bay</td>
<td>408</td>
<td>581</td>
<td>625</td>
<td>684</td>
<td>713</td>
</tr>
<tr>
<td>FL</td>
<td>Bradford</td>
<td>84</td>
<td>89</td>
<td>92</td>
<td>94</td>
<td>95</td>
</tr>
<tr>
<td>FL</td>
<td>Brevard</td>
<td>1962</td>
<td>2180</td>
<td>2366</td>
<td>2453</td>
<td>2521</td>
</tr>
</tbody>
</table>

Katie Keegan, David Melendez, Jennifer Zheng
Cumulative Known COVID-19 Deaths Nationwide

The Data

- Cumulative known COVID-19 deaths each day in each of the 50 states plus Washington DC
- February 6, 2020 to July 5, 2020
- Data Matrix: 51 states $\times$ 166 days

Katie Keegan, David Melendez, Jennifer Zheng
Scree Plot

- a line plot of the eigenvalues of factors or principal components
- used to determine an “appropriate” number of PC components

Katie Keegan, David Melendez, Jennifer Zheng
New COVID-19 Cases Nationwide

The Data

New known COVID-19 Cases each day in each of the 50 states plus Washington DC

January 22, 2020 to July 12, 2020

Data Matrix: 51 states × 173 days

The Method

Take SVD:

\[ X = U \Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T \]

Look at SV spectrum

Plot most dominant rank 1 components \( \sigma_1 u_1 v_1^T, \sigma_2 u_2 v_2^T, \) etc

Katie Keegan, David Melendez, Jennifer Zheng
## The Data

<table>
<thead>
<tr>
<th>State</th>
<th>January 22, 2020 to July 12, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>State 1</td>
<td>123</td>
</tr>
<tr>
<td>State 2</td>
<td>456</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**The Method**

Take SVD: $X = U \Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$.

Look at SV spectrum and plot most dominant rank 1 components: $\sigma_1 u_1 v_1^T$, $\sigma_2 u_2 v_2^T$, etc.

Katie Keegan, David Melendez, Jennifer Zheng
New COVID-19 Cases Nationwide

The Data

- New known COVID-19 Cases each day in each of the 50 states plus Washington DC
- January 22, 2020 to July 12, 2020
- Data Matrix: 51 states $\times$ 173 days

The Method

- Take SVD: $X = U \Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$
- Look at SV spectrum
- Plot most dominant rank 1 components $\sigma_1 u_1 v_1^T$, $\sigma_2 u_2 v_2^T$, etc.

Katie Keegan, David Melendez, Jennifer Zheng
New COVID-19 Cases Nationwide

The Data

- New known COVID-19 Cases each day in each of the 50 states plus Washington DC
- January 22, 2020 to July 12, 2020
- Data Matrix: 51 states × 173 days

The Method

- Take SVD: \( X = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T \)
New COVID-19 Cases Nationwide

The Data
- New known COVID-19 Cases each day in each of the 50 states plus Washington DC
- January 22, 2020 to July 12, 2020
- Data Matrix: 51 states × 173 days

The Method
- Take SVD: $X = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$
- Look at SV spectrum

Katie Keegan, David Melendez, Jennifer Zheng
## The Data
- New known COVID-19 Cases each day in each of the 50 states plus Washington DC
- January 22, 2020 to July 12, 2020
- Data Matrix: 51 states × 173 days

## The Method
- Take SVD: \( X = U \Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T \)
- Look at SV spectrum
- Plot most dominant rank 1 components \( \sigma_1 u_1 v_1^T, \sigma_2 u_2 v_2^T, \) etc
New COVID-19 Cases Nationwide

Katie Keegan, David Melendez, Jennifer Zheng
The Data

- Log cumulative known COVID-19 cases each day in each Florida county
- January 22, 2020 to July 12, 2020
- Data Matrix: 68 counties × 173 days
Florida COVID-19 Data

SVD Plots: Log Cumulative Known FL COVID-19 Cases

Katie Keegan, David Melendez, Jennifer Zheng
Digital Ownership Protection

- How can we verify the owner of a piece of media?
### Digital Ownership Protection

- How can we verify the owner of a piece of media?
- Hide a watermark inside the media

Katie Keegan, David Melendez, Jennifer Zheng
Watermarking

Digital Ownership Protection

- How can we verify the owner of a piece of media?
- Hide a watermark inside the media

Concerns:
- Perceptibility
- Security
- Robustness against distortions

Katie Keegan, David Melendez, Jennifer Zheng
Liu & Tan Watermarking Scheme

Embedding

\[ \tilde{A} \rightarrow USV^T \]

\[ W \rightarrow US^W \]

\[ W \rightarrow S \]

\[ W + \alpha \rightarrow US^W \]

\[ S \leftarrow US^W \]

\[ VS^T \]

Save \( US, VS, S \) for extraction

Extraction

Given \( \tilde{A}, US, VS, S, \alpha \)

\[ \tilde{W} = \alpha^{-1}(US^SV^T - S) \]

Note \( US^SV^T = S + \alpha W \)

Then

Katie Keegan, David Melendez, Jennifer Zheng
Embedding

\[ A \rightarrow USV^T \]
 Embedding

- $A \rightarrow USV^T$
- $W \rightarrow U_W S_W V_W^T$

Liu & Tan Watermarking Scheme

Katie Keegan, David Melendez, Jennifer Zheng
Embedding

- $A \rightarrow USV^\top$
- $W \rightarrow U_W S_W V_W^\top$
- $S + \alpha W \rightarrow U_S S' V_S^\top$

Note: $U_S S' V_S^\top = S + \alpha W$
Liu & Tan Watermarking Scheme

Embedding

- $A \rightarrow USV^T$
- $W \rightarrow U_W S_W V_W^T$
- $S + \alpha W \rightarrow U_S S' V_S^T$
- $A_W \leftarrow US' V^T$
Liu & Tan Watermarking Scheme

**Embedding**

- \( A \rightarrow USV^\top \)
- \( W \rightarrow U_W S_W V_W^\top \)
- \( S + \alpha W \rightarrow US'S'V_S^\top \)
- \( A_W \leftarrow US'V^\top \)
- Save \( US, VS, S \) for extraction
Liu & Tan Watermarking Scheme

**Embedding**
- \( A \rightarrow USV^T \)
- \( W \rightarrow U_W S_W V_W^T \)
- \( S + \alpha W \rightarrow U_S S' V_S^T \)
- \( A_W \leftarrow US' V^T \)
- Save \( U_S, V_S, S \) for extraction

**Extraction**
- Given \( \tilde{A}_W, U_S, V_S, S, \alpha \)
Liu & Tan Watermarking Scheme

Embedding
- $A \rightarrow USV^\top$
- $W \rightarrow U_W S_W V_W^\top$
- $S + \alpha W \rightarrow U_S S' V_S^\top$
- $A_W \leftarrow US' V^\top$
- Save $U_S, V_S, S$ for extraction

Extraction
- Given $\tilde{A}_W, U_S, V_S, S, \alpha$
- $\tilde{A}_W \rightarrow US' V^\top$
Liu & Tan Watermarking Scheme

**Embedding**
- $A \rightarrow USV^\top$
- $W \rightarrow U_W S_W V_W^\top$
- $S + \alpha W \rightarrow U_S S' V_S^\top$
- $A_W \leftarrow US' V^\top$
- Save $U_S$, $V_S$, $S$ for extraction

**Extraction**
- Given $\tilde{A}_W$, $U_S$, $V_S$, $S$, $\alpha$
- $\tilde{A}_W \rightarrow US' V^\top$
- Note $US' V_S^\top = S + \alpha W$
Liu & Tan Watermarking Scheme

**Embedding**
- $A \rightarrow USV^T$
- $W \rightarrow U_W S_W V_W^T$
- $S + \alpha W \rightarrow U_S S' V_S^T$
- $A_W \leftarrow US'V^T$
- Save $U_S, V_S, S$ for extraction

**Extraction**
- Given $\tilde{A}_W, U_S, V_S, S, \alpha$
- $\tilde{A}_W \rightarrow US'V^T$
- Note $U_S S' V_S^T = S + \alpha W$
- Then $\tilde{W} = \alpha^{-1}(U_S S' V_S^T - S)$
Liu & Tan Watermarking Scheme

Audio Watermarking Examples

\[ \alpha = 0.1 \]
\[ \alpha = 0.4 \]
\[ \alpha = 1.6 \]
Liu & Tan Watermarking Scheme

Audio Watermarking With Distortions

Lower Pitch
Frequency
remove noise
Add reverb

Katie Keegan, David Melendez, Jennifer Zheng
### Security Test

<table>
<thead>
<tr>
<th>Watermark</th>
<th>Phony Watermark</th>
<th>Watermarked Image</th>
<th>Extracted Phony Watermark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embed watermark $W$ into $A$ to obtain $A_W$</td>
<td>Attempt to extract phony watermark $X$ from $A_W$</td>
<td>Image</td>
<td>Extracted</td>
</tr>
</tbody>
</table>
Security Test

- Embed watermark $W$ into $A$ to obtain $A_W$
Security Test

- Embed watermark \( W \) into \( A \) to obtain \( A_W \)
- Attempt to extract phony watermark \( X \) from \( A_W \)
Liu & Tan Watermarking Scheme

Security Test

- Embed watermark $W$ into $A$ to obtain $A_W$
- Attempt to extract phony watermark $X$ from $A_W$
Jain et al. Watermarking Scheme

- Modification to Liu & Tan scheme
- Improved security
Jain et al. Watermarking Scheme

- Modification to Liu & Tan scheme
- Improved security

**Embedding**

- \( A \rightarrow USV^T \)
- \( W \rightarrow U_W S_W V_W^T \)
- \( S_1 \leftarrow S + \alpha U_W S_W \)
- \( A_W \leftarrow US_1 V^T \)
- \( (A_W = A + \alpha UU_W S_W V^T) \)

**Extraction**

- Given \( \tilde{A}_W, A, V_W \)
- \( A_W = A + \alpha UU_W S_W V^T \)
- Solve for \( W \! \)
- \( \tilde{W} \leftarrow \alpha^{-1} U^T (\tilde{A}_W - A) V V_W^T \)
Jain et al. Watermarking Scheme

Watermarking Examples

<table>
<thead>
<tr>
<th>Image</th>
<th>Watermark</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="watermark1.png" alt="Watermark" /></td>
</tr>
<tr>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="watermark2.png" alt="Watermark" /></td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="watermark3.png" alt="Watermark" /></td>
</tr>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="watermark4.png" alt="Watermark" /></td>
</tr>
</tbody>
</table>

\[ \alpha = 0.5 \] \[ \alpha = 0.25 \] \[ \alpha = 0.1 \] \[ \alpha = 0.01 \]
Jain et al. Watermarking Scheme

Security Test

Image  Watermark  Phony

Watermarked Image  Watermark Extracted  Phony Extracted
### Jain et al. Watermarking Scheme

#### Extraction after Compression

<table>
<thead>
<tr>
<th>Image</th>
<th>Rank 100</th>
<th>Rank 10</th>
<th>Rank 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Watermark" /></td>
<td><img src="image2.png" alt="Extracted" /></td>
<td><img src="image3.png" alt="Extracted" /></td>
<td><img src="image4.png" alt="Extracted" /></td>
</tr>
</tbody>
</table>

Katie Keegan, David Melendez, Jennifer Zheng
Modified Jain et al. Watermarking Scheme

\[ A = USV^\top \]
\[ W = U_W S_W V_W^\top \]
Modified Jain et al. Watermarking Scheme

\[ A = USV^\top \]
\[ W = U_W S_W V_W^\top \]

**Embedding**

Jain et al. Scheme:
\[ A_W = A + \alpha U U_W S_W V^\top \]

**Extraction**

Jain et al. Scheme:
\[ W = \alpha^{-1} U^\top (A_W - A) V V_W^\top \]
Modified Jain et al. Watermarking Scheme

\[ A = USV^\top \]
\[ W = U_W S_W V_{W}^\top \]

Embedding

Jain et al. Scheme:
\[ A_W = A + \alpha UU_W S_W V^\top \]

Extraction

Jain et al. Scheme:
\[ W = \alpha^{-1} U^\top (A_W - A) VV_W^\top \]
$A = USV^\top$

$W = U_W S_W V_W^\top$

**Embedding**

Modified Jain et al. Scheme:

$A_W = A + \alpha U_W S_W V^\top$

**Extraction**

Modified Jain et al. Scheme:

$W = \alpha^{-1} (A_W - A) V V_W^\top$
### Modified Jain et al. Watermarking Scheme

#### Watermarking Examples

<table>
<thead>
<tr>
<th>Image</th>
<th>Jain, $\alpha = 0.5$</th>
<th>Jain, $\alpha = 0.25$</th>
<th>Jain, $\alpha = 0.1$</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td><img src="watermark1" alt="Watermark" /></td>
<td><img src="watermark2" alt="Watermark" /></td>
<td><img src="watermark3" alt="Watermark" /></td>
</tr>
<tr>
<td><img src="image2" alt="Image" /></td>
<td><img src="watermark4" alt="Watermark" /></td>
<td><img src="watermark5" alt="Watermark" /></td>
<td><img src="watermark6" alt="Watermark" /></td>
</tr>
</tbody>
</table>

Katie Keegan, David Melendez, Jennifer Zheng
Modified Jain et al. Watermarking Scheme

Evaluating Watermarked Image vs Original Image

∥\(A - A_{\text{Mod}}\)∥_F = ∥\(W\)∥_F 

corr(\(A\), \(A_{\text{W}}\)) = \langle A, A_{\text{W}} \rangle_F 

∥\(A\)∥_F ∥\(A_{\text{W}}\)∥_F = \cos(\angle(\(A\), \(A_{\text{W}}\)))
Modified Jain et al. Watermarking Scheme

Evaluating Watermarked Image vs Original Image

\[ \| A - A_{Jain} \|_F = \| A - A_{Mod} \|_F = \alpha \| W \|_F \]
Modified Jain et al. Watermarking Scheme

Evaluating Watermarked Image vs Original Image

\[ \| A - A_{\text{Jain}} \|_F = \| A - A_{\text{Mod}} \|_F = \alpha \| W \|_F \]

\[ \text{corr}(A, A_W) = \frac{\langle A, A_W \rangle_F}{\| A \|_F \| A_W \|_F} = \cos(\angle(A, A_W)) \]
Modified Jain et al. Watermarking Scheme

Evaluating Watermarked Image vs Original Image

\[ \| A - A_{Jain} \|_F = \| A - A_{Mod} \|_F = \alpha \| W \|_F \]

\[ \text{corr}(A, A_W) = \frac{\langle A, A_W \rangle_F}{\| A \|_F \| A_W \|_F} = \cos(\angle(A, A_W)) \]

Katie Keegan, David Melendez, Jennifer Zheng
Modified Jain et al. Watermarking Scheme

Security Test

Image | Watermark | Phony
---|---|---
Watermarked Image | Watermark Extracted | Phony Extracted
Modified Jain et al. Watermarking Scheme

Extraction after Compression

Image | Rank 100 | Rank 10 | Rank 1
--- | --- | --- | ---
Watermark | Extracted | Extracted | Extracted

Katie Keegan, David Melendez, Jennifer Zheng
Conclusion

Summary
- Media Compression and Processing
- Data Analysis
- Digital Ownership Protection
- Modified Watermarking Scheme

Future Directions
- Video background removal
- Modern watermarking algorithms
- Audio watermarking
- Randomized watermark extraction
THANK YOU!
References I


N. Erichson, S. L. Brunton, and J. Kutz. Compressed singular value decomposition for image and video processing.


Nasser Kehtarnavaz.  
Chapter 7 - frequency domain processing.  

Ruizhen Liu and Tieniu Tan.  
An svd-based watermarking scheme for protecting rightful ownership.  

Carla D. Martin and Mason A. Porter.  
The extraordinary svd.  
G. Strang.  
Linear Algebra and Learning from Data.  

L. Zhang and A. Li.  
Robust watermarking scheme based on singular value of decomposition in dwt domain.  

Lingsong Zhang, J. S. Marron, Haipeng Shen, and Zhengyuan Zhu.  
Singular value decomposition and its visualization.  
Jain et al. Watermarking Scheme

<table>
<thead>
<tr>
<th>Extraction after Rotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
</tr>
<tr>
<td><img src="image1.jpg" alt="Image" /></td>
</tr>
<tr>
<td><img src="watermark1.jpg" alt="Watermark" /></td>
</tr>
</tbody>
</table>

Katie Keegan, David Melendez, Jennifer Zheng
Modified Jain et al. Watermarking Scheme

Extraction after Rotation

<table>
<thead>
<tr>
<th>Image</th>
<th>1°</th>
<th>25°</th>
<th>45°</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watermark</td>
<td>Extracted</td>
<td>Extracted</td>
<td>Extracted</td>
</tr>
</tbody>
</table>

Katie Keegan, David Melendez, Jennifer Zheng
Low Rank Distortion Plots - Various Scaling Factors

Watermark Extraction Error for Low Rank Approximations

Watermark Extraction Error for Low Rank Approximations (Jain)

Watermark Extraction Error for Low Rank Approximations (Jain Mod)

Random Matrix

Katie Keegan, David Melendez, Jennifer Zheng