Randomized SVD and its Applications

Katie Keegan, David Melendez, Jennifer Zheng

July 31, 2020

Introduction

The Singular Value Decomposition

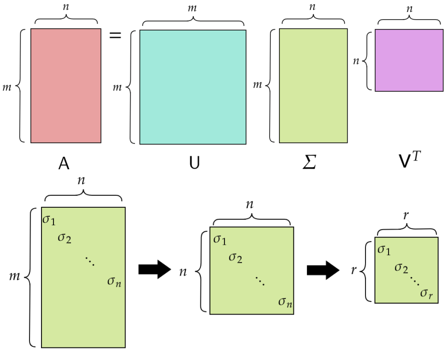
- An incredibly important matrix decomposition in linear algebra
- Has applications in many different domains

What we will cover

- Image, video, audio processing
- Data analysis
- Digital ownership protection

Singular Value Decomposition

- $A_{mxn} = U_{mxm}\Sigma_{mxn}V_{nxn}^\top$
- U and V are orthogonal matrices, Σ is a diagonal matrix with positive diagonal entries $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r$, and $r = rank(A)$.



Eckart-Young Theorem

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

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

Example

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

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

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

Singular Value Decomposition

Eckart-Young Theorem

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

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

Truncating...

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

Singular Value Decomposition

Eckart-Young Theorem

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

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

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

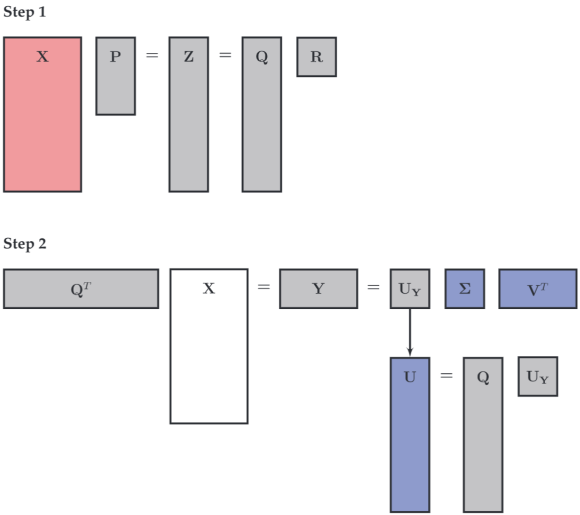


Figure from *Data-Driven Science and Engineering* by Steven Brunton and Nathan Kutz

Converting Color Images to Matrices

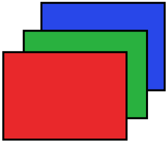
Color Stacking

- 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

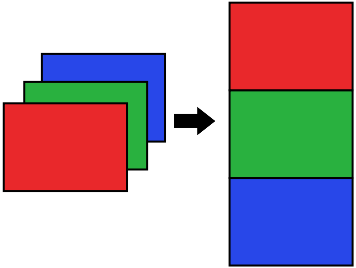
- Just like how a black-and-white image can be represented as a matrix of values between 0 and 1, color images can be represented as the combination of three matrices (color channels)
- We can take these channels apart, stack them, and then compute their SVD



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

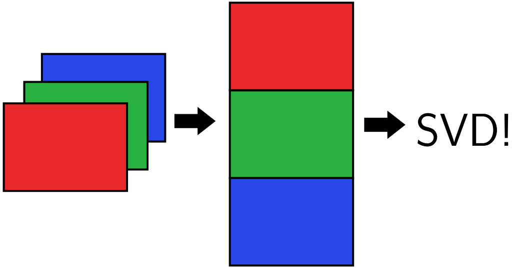
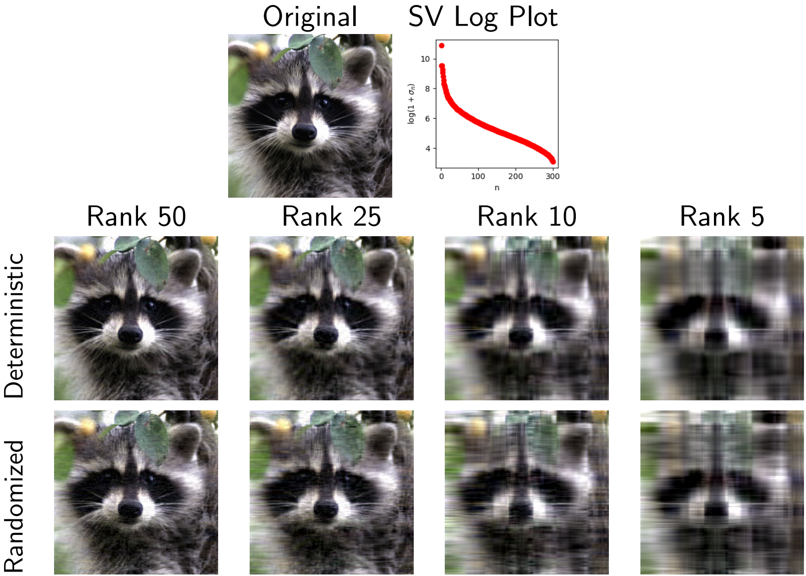


Image Compression



Modifying Singular Values

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

Modifying Singular Values

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

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

Modifying Singular Values

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

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

What happens if we try to modify the singular values?

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

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

What happens if we try to modify the singular values?

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Multiplying by some scalar } c} \begin{bmatrix} c\sigma_1 & & \\ & \ddots & \\ & & c\sigma_k \end{bmatrix}$$

Modifying Singular Values

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

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

What happens if we try to modify the singular values?

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Multiplying by some scalar } c} \begin{bmatrix} c\sigma_1 & & \\ & \ddots & \\ & & c\sigma_k \end{bmatrix}$$

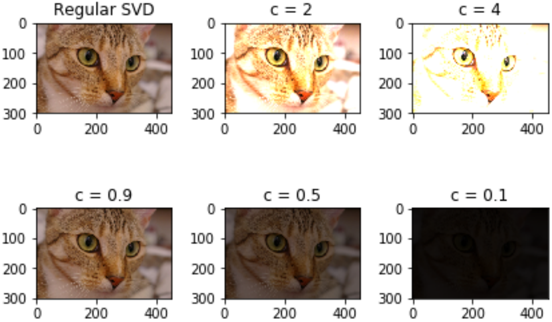
$$\begin{bmatrix} | & & | \\ u_1 & \dots & u_k \\ | & & | \end{bmatrix} \begin{bmatrix} c\sigma_1 & & \\ & \ddots & \\ & & c\sigma_k \end{bmatrix} \begin{bmatrix} - & v_1 & - \\ & \vdots & \\ - & v_k & - \end{bmatrix} = \ ?$$

Modifying Singular Values

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Multiplying by some scalar } c} \begin{bmatrix} c\sigma_1 & & \\ & \ddots & \\ & & c\sigma_k \end{bmatrix}$$

Modifying Singular Values

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Multiplying by some scalar } c} \begin{bmatrix} c\sigma_1 & & \\ & \ddots & \\ & & c\sigma_k \end{bmatrix}$$



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

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Adding some scalar } m} \begin{bmatrix} \sigma_1 + m & & \\ & \ddots & \\ & & \sigma_k + m \end{bmatrix}$$

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Raising to some exponent } n} \begin{bmatrix} \sigma_1^n & & \\ & \ddots & \\ & & \sigma_k^n \end{bmatrix}$$

Modifying Singular Values

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Adding some scalar } m} \begin{bmatrix} \sigma_1 + m & & \\ & \ddots & \\ & & \sigma_k + m \end{bmatrix}$$

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Raising to some exponent } n} \begin{bmatrix} \sigma_1^n & & \\ & \ddots & \\ & & \sigma_k^n \end{bmatrix}$$

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix} \xrightarrow{\text{Applying logarithmic mapping}} \begin{bmatrix} \log(\sigma_1 + 1)^p & & \\ & \ddots & \\ & & \log(\sigma_k + 1)^p \end{bmatrix}$$

Can also be applied to audio, video matrices

Video Compression

Video $\rightarrow$ 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

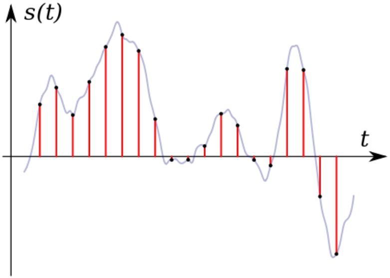
- For rank $r \leq 10$, only the most dominant features in every frame of image is captured
- The lower the rank, the less moving objects captured
- More blurry for shaky videos

Video Background Extraction

Audio Compresssion

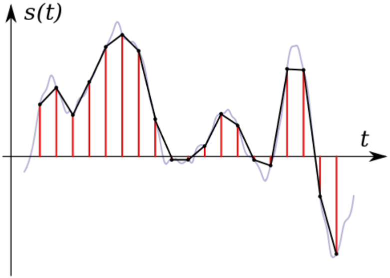
Representing Audio as a Signal

- 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.



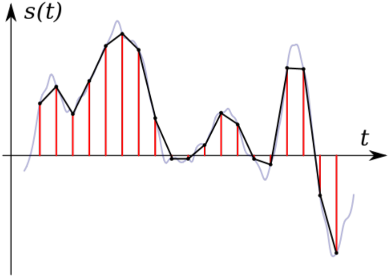
Representing Audio as a Signal

- 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.



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.



$$[10, 20, 8, 22, 40, 50, 38, \ldots]$$

Audio Compression

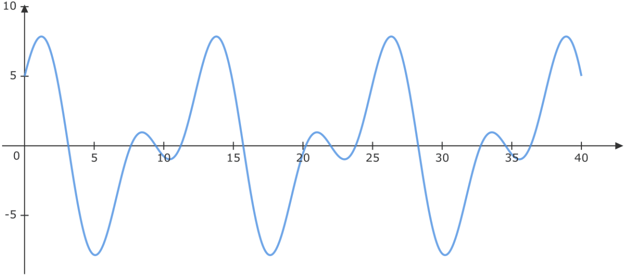
Fourier Transform

- 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

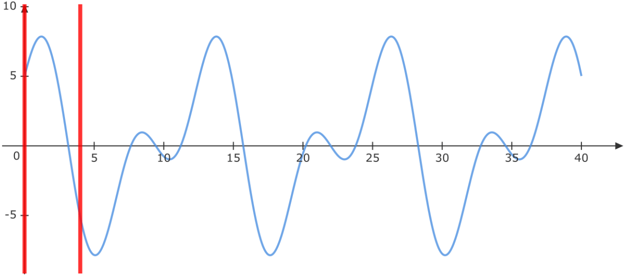
Short-Time Fourier Transform

- Takes the Fourier transform of small "windows" of the signal
- Puts the resulting spectra in columns of a matrix

Short-Time Fourier Transform

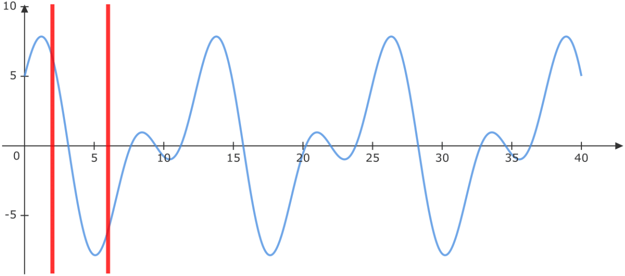


Short-Time Fourier Transform



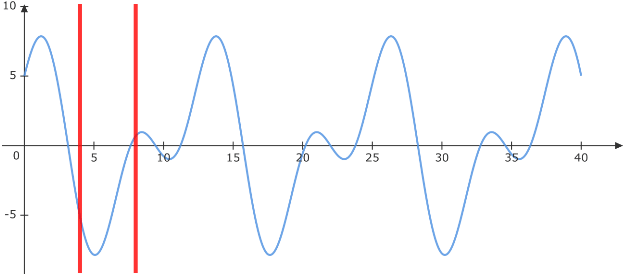
$$\begin{bmatrix} * & & & \\ * & & & \\ * & & & \\ * & & & \\ * & & & \\ * & & & \end{bmatrix}$$

Short-Time Fourier Transform



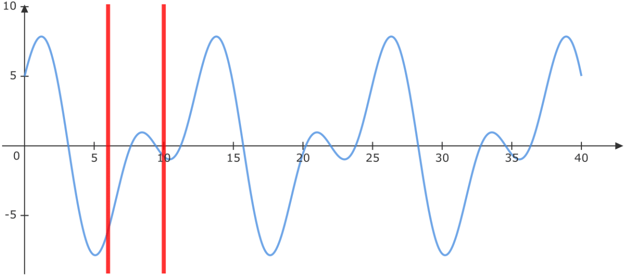
$$\begin{bmatrix} * & * & & & & \\ * & * & & & & \\ * & * & & & & \\ * & * & & & & \\ * & * & & & & \\ * & * & & & & \end{bmatrix}$$

Short-Time Fourier Transform



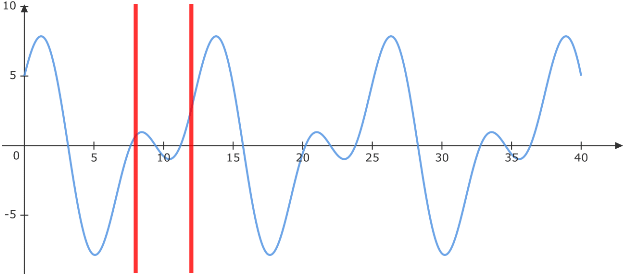
$$\begin{bmatrix} * & * & {\color{red}*} & & & & \\ * & * & {\color{red}*} & & & & \\ * & * & {\color{red}*} & & & & \\ * & * & {\color{red}*} & & & & \\ * & * & {\color{red}*} & & & & \\ * & * & {\color{red}*} & & & & \end{bmatrix}$$

Short-Time Fourier Transform



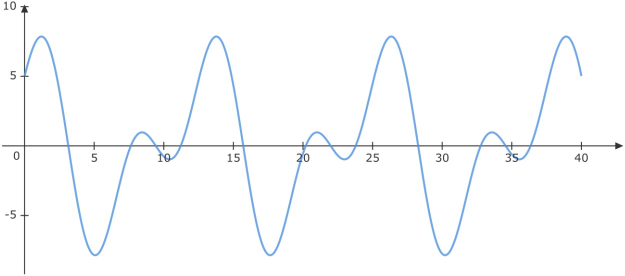
$$\begin{bmatrix} * & * & * & & & \\ * & * & * & & & \\ * & * & * & \cdots & & \\ * & * & * & & & \\ * & * & * & & & \\ * & * & * & & & \end{bmatrix}$$

Short-Time Fourier Transform



$$\begin{bmatrix} * & * & * & & \\ * & * & * & & \\ * & * & * & \cdots & \\ * & * & * & & \\ * & * & * & & \\ * & * & * & & \end{bmatrix}$$

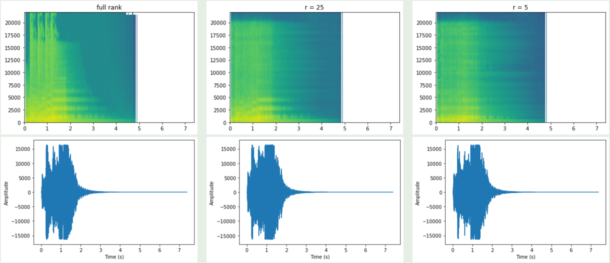
Short-Time Fourier Transform



$$\begin{bmatrix} * & * & * & & * & * \\ * & * & * & & * & * \\ * & * & * & \cdots & * & * \\ * & * & * & & * & * \\ * & * & * & & * & * \\ * & * & * & & * & * \end{bmatrix}$$

Audio Demo

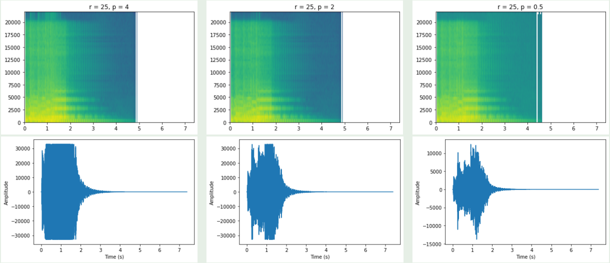
Low-rank Approximation of Audios



Rank 118 Rank 25 Rank 5

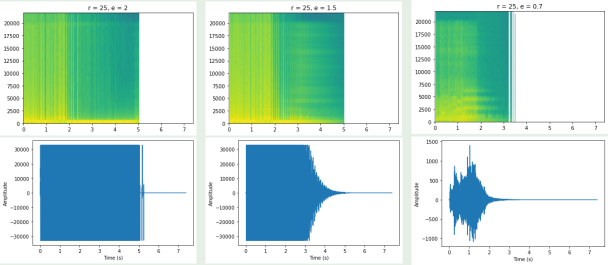
Singular Value Modification on Audios

Multiply σ by some scalar p



p = 4 p = 2 p = 0.5

Raise σ to some exponent n



$\mathrm{n} = 2$ $\mathrm{n} = 1.5$ $\mathrm{n} = 0.7$

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)

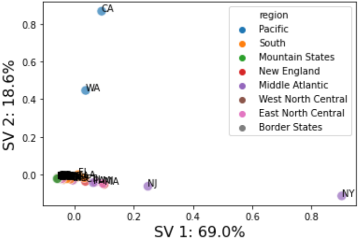
Snippet of Data

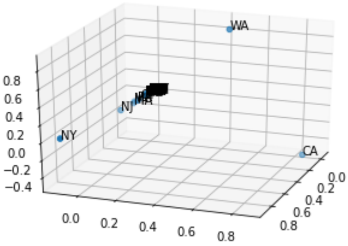
State	County	7/1/20	7/2/20	7/3/20	7/4/20	7/5/20
FL	Alachua	1245	1332	1423	1506	1578
FL	Baker	72	80	84	98	105
FL	Bay	408	581	625	684	713
FL	Bradford	84	89	92	94	95
FL	Brevard	1962	2180	2366	2453	2521

Cumulative Known COVID-19 Deaths Nationwide

The Data

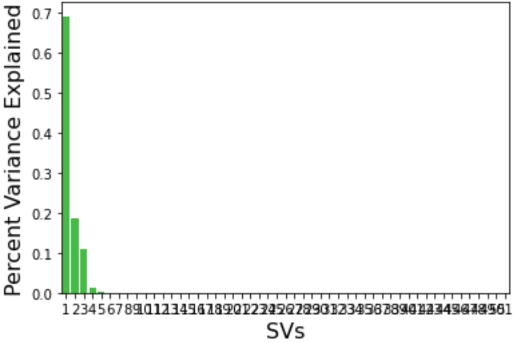
- Cumulative known COVID-19 deaths each day in each of the 50 states plus Washington DC
- February 6, 2020 to July 5, 2020
- Data Matrix: 51 states $\times$ 166 days





Scree Plot

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



New COVID-19 Cases Nationwide

New COVID-19 Cases Nationwide

The Data

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

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

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

New COVID-19 Cases Nationwide

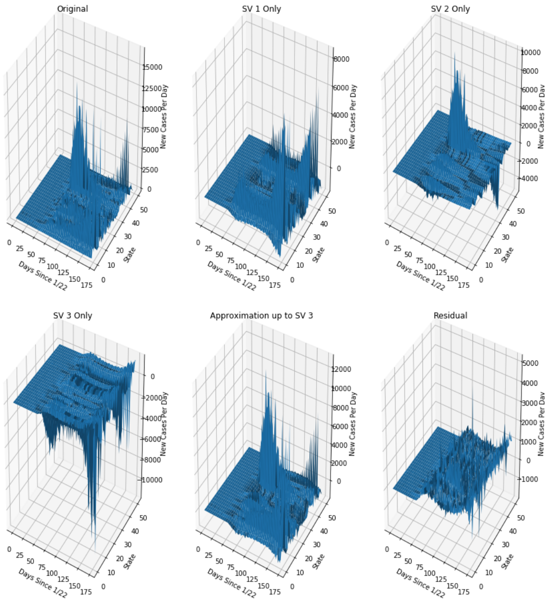
The Data

- 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

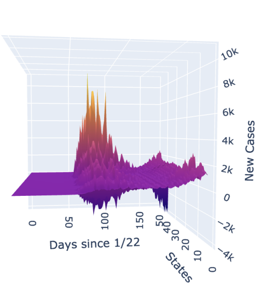
The Method

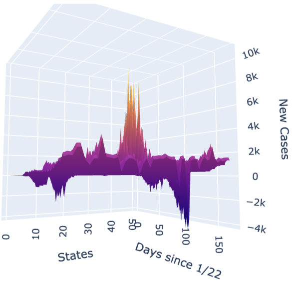
- Take SVD: $X = U\Sigma V^T = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$
- Look at SV spectrum
- Plot most dominant rank 1 components $\sigma_1 u_1 v_1^T$, $\sigma_2 u_2 v_2^T$, etc

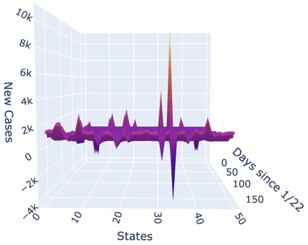
New COVID-19 Cases Nationwide



New COVID-19 Cases Nationwide - SV2





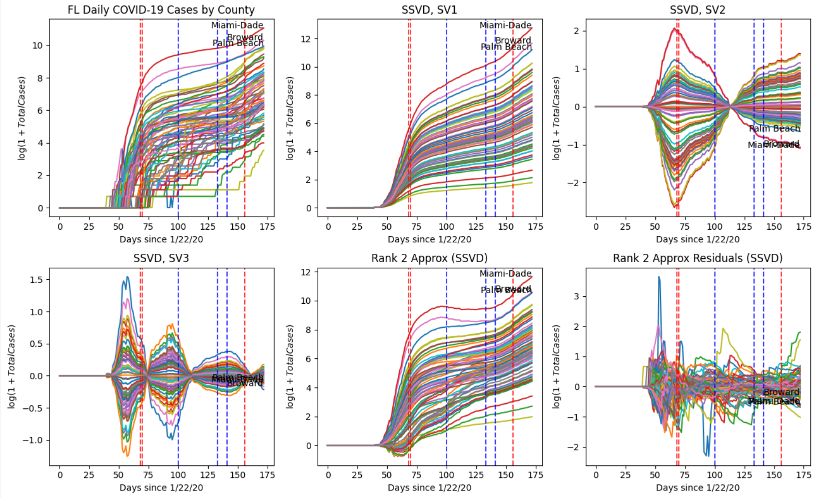


Florida COVID-19 Data

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 $\times$ 173 days

SVD Plots: Log Cumulative Known FL COVID-19 Cases



Watermarking

Digital Ownership Protection

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

Watermarking

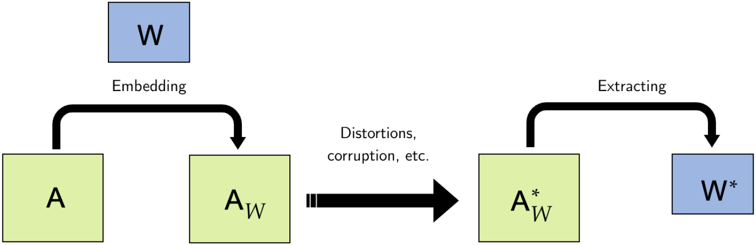
Digital Ownership Protection

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

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



Embedding

- $A \to USV^\top$

Liu & Tan Watermarking Scheme

Embedding

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

Embedding

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

Liu & Tan Watermarking Scheme

Embedding

- $A \to USV^\top$
- $W \to U_W S_W V_W^\top$
- $S + \alpha W \to U_S S' V_S^\top$
- $A_W \leftarrow US'V^\top$

Liu & Tan Watermarking Scheme

Embedding

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

Liu & Tan Watermarking Scheme

Embedding

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

Extraction

- Given $\widetilde{A}_W$, U_S, V_S, S, α

Liu & Tan Watermarking Scheme

Embedding

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

Extraction

- Given $\widetilde{A}_W$, U_S, V_S, S, α
- $\widetilde{A}_W \to US'V^\top$

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 $\widetilde{A}_W$, U_S, V_S, S, α
- $\widetilde{A}_W \rightarrow US'V^\top$
- Note $U_S S' V_S^\top = S + \alpha W$

Liu & Tan Watermarking Scheme

Embedding

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

Extraction

- Given $\widetilde{A}_W$, U_S, V_S, S, α
- $\widetilde{A}_W \to US'V^\top$
- Note $U_S S' V_S^\top = S + \alpha W$
- Then $\widetilde{W} = \alpha^{-1}(U_S S' V_S^\top - S)$

Watermarking Examples

Image

Watermark

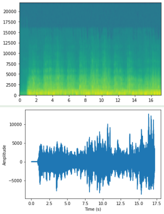
$\alpha = 1$

$\alpha = 0.5$

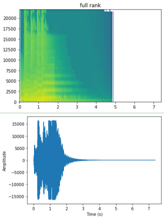
$\alpha = 0.25$

$\alpha = 0.1$

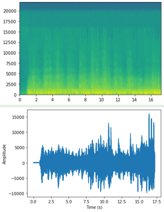
Audio Watermarking Examples



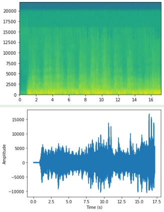
Original



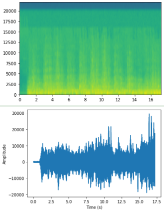
Watermark



$\alpha = 0.1$

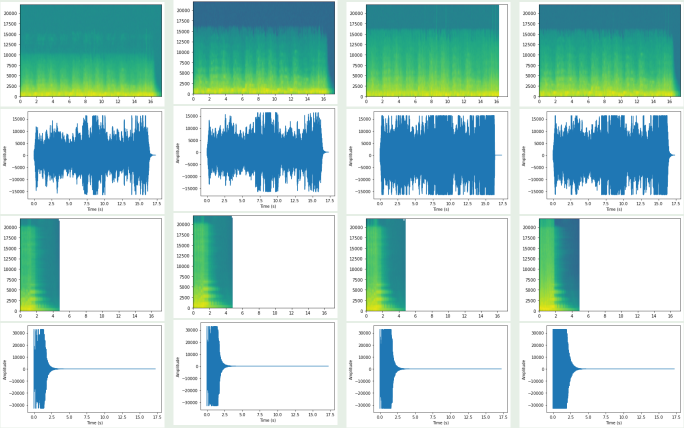


$\alpha = 0.4$



$\alpha = 1.6$

Audio Watermarking With Distortions



Lower Pitch | Frequency | remove noise | Add reverb

Security Test

Liu & Tan Watermarking Scheme

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

Security Test

- Embed watermark W into A to obtain A_W
- Attempt to extract phony watermark X from A_W

Image

Watermark

Phony

Watermarked Image

Watermark Extracted

Phony Extracted

Jain et el. Watermarking Scheme

- Modification to Liu & Tan scheme
- Improved security

Jain et el. Watermarking Scheme

- Modification to Liu & Tan scheme
- Improved security

Embedding

- $A \to USV^\top$
- $W \to U_W S_W V_W^\top$
- $S_1 \leftarrow S + \alpha U_W S_W$
- $A_W \leftarrow U S_1 V^\top$
- $(A_W = A + \alpha U U_W S_W V^\top)$

Extraction

- Given $\widetilde{A}_W$, A, V_W
- $A_W = A + \alpha U U_W S_W V^\top$
- Solve for W!
- $\widetilde{W} \leftarrow \alpha^{-1} U^\top (\widetilde{A}_W - A) V V_W^\top$

Jain et al. Watermarking Scheme

Watermarking Examples

Image

Watermark

$\alpha = 0.5$

$\alpha = 0.25$

$\alpha = 0.1$

$\alpha = 0.01$

Security Test

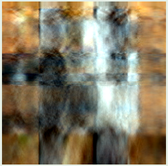
Image

Watermark

Phony

Watermarked Image

Watermark Extracted



Phony Extracted

Jain et al. Watermarking Scheme

Extraction after Compression

Image | Rank 100 | Rank 10 | Rank 1

Watermark | Extracted | Extracted | Extracted

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

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

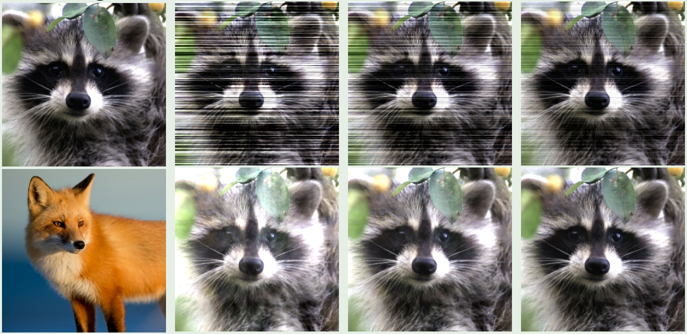
Modified Jain et al. Scheme:
$A_W = A + \alpha U_W S_W V^\top$

Extraction

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

Watermarking Examples

Image | Jain, $\alpha = 0.5$ | Jain, $\alpha = 0.25$ | Jain, $\alpha = 0.1$



Watermark | Mod, $\alpha = 0.5$ | Mod, $\alpha = 0.25$ | Mod, $\alpha = 0.1$

Evaluating Watermarked Image vs Original Image

Evaluating Watermarked Image vs Original Image

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

Evaluating Watermarked Image vs Original Image

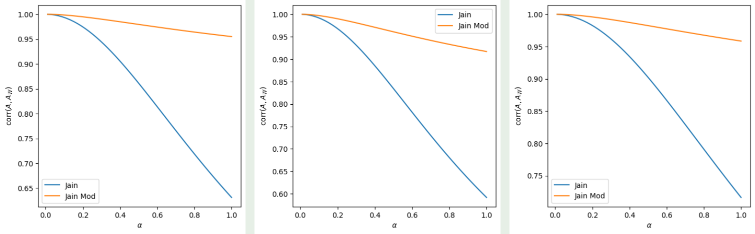
$$\|A - A_{\text{Jain}}\|_F = \|A - A_{\text{Mod}}\|_F = \alpha \|W\|_F$$

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

Evaluating Watermarked Image vs Original Image

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

$$\operatorname{corr}(A, A_W) = \frac{\langle A, A_W\rangle_F}{\|A\|_F \, \|A_W\|_F} = \cos(\angle(A, A_W))$$



Modified Jain et al. Watermarking Scheme

Security Test

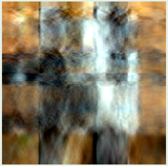
Image

Watermark

Phony

Watermarked Image

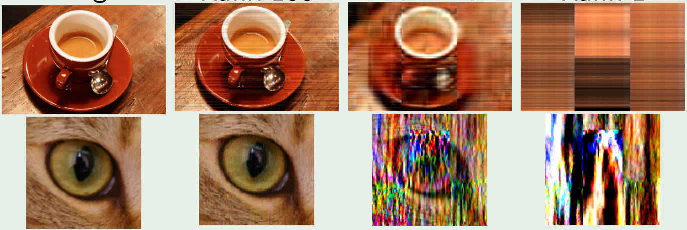
Watermark Extracted



Phony Extracted

Extraction after Compression

Image | Rank 100 | Rank 10 | Rank 1



Watermark | Extracted | Extracted | Extracted

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

- Coronavirus locations: Covid-19 map by county and state, Jul 2020.
- Clifford Bergman and Jennifer Davidson.
  Unitary embedding for data hiding with the svd.
  volume 5681, pages 619–630, 03 2005.
- Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup.
  Matrices, vector spaces, and information retrieval.
  SIAM Review, 41:335–362, 1999.
- Steven Brunton and J. Kutz.
  Singular Value Decomposition (SVD), pages 3–46.
  02 2019.
- Carl Eckart and Gale Young.
  The approximation of one matrix by another of lower rank.
  Psychometrika, 1(3):211–218, Sep 1936.

N. Erichson, S. L. Brunton, and J. Kutz.
Compressed singular value decomposition for image and video processing.
In *2017 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 1880–1888, Los Alamitos, CA, USA, oct 2017. IEEE Computer Society.

Chirag Jain, Siddharth Arora, and Prasanta K. Panigrahi.
A reliable SVD based watermarking schem.
CoRR, abs/0808.0309, 2008.

Dan Kalman.
A singularly valuable decomposition: The svd of a matrix.
The College Mathematics Journal, 27(1):2–23, 1996.

Nasser Kehtarnavaz.
Chapter 7 - frequency domain processing.
In Nasser Kehtarnavaz, editor, *Digital Signal Processing System Design (Second Edition)*, pages 175 – 196. Academic Press, Burlington, second edition edition, 2008.

Ruizhen Liu and Tieniu Tan.
An svd-based watermarking scheme for protecting rightful ownership.
IEEE Transactions on Multimedia, 4:121–128, 08 2002.

Carla D. Martin and Mason A. Porter.
The extraordinary svd.
The American Mathematical Monthly, 119:838 – 851, 2012.

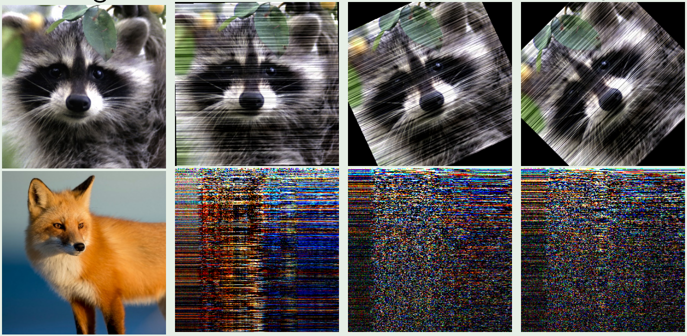
G. Strang.
Linear Algebra and Learning from Data.
Wellesley-Cambridge Press, 2019.

L. Zhang and A. Li.
Robust watermarking scheme based on singular value of decomposition in dwt domain.
In *2009 Asia-Pacific Conference on Information Processing*, volume 2, pages 19–22, 2009.

Lingsong Zhang, J. S. Marron, Haipeng Shen, and Zhengyuan Zhu.
Singular value decomposition and its visualization.
Journal of Computational and Graphical Statistics, 16:1–22, 2007.

Extraction after Rotation

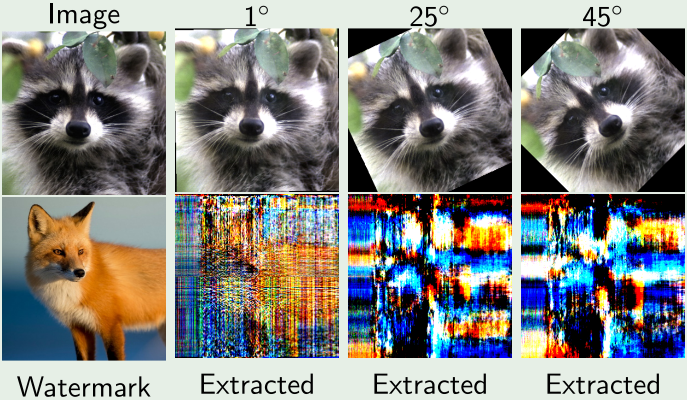
Image | 1° | 25° | 45°



Watermark | Extracted | Extracted | Extracted

Modified Jain et al. Watermarking Scheme

Extraction after Rotation



Low Rank Distortion Plots - Various Scaling Factors