Singular Value Decomposition

Panchatcharam Mariappan

Associate Professor

Department of Mathematics and Statistics,
IIT Tirupati

Course

- Visit this Course and understand how to read an image (imread command) and show it as a matrix
  - https://matlabacademy.mathworks.com/details/image-processing-onramp/imageprocessing
- Extract a grayscale image or RGB image and convert it to a matrix format
- Compute the SVD of this matrix (SVD is learnt from Introduction to Linear Algebra with MATLAB)

Image Reading

Reading Gray Scale Image

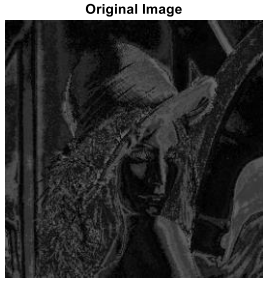
- Download the *GrayScale.png* file from here

https://facweb.iittp.ac.in/~pmariappan/Courses/MA517M/GrayScale.png

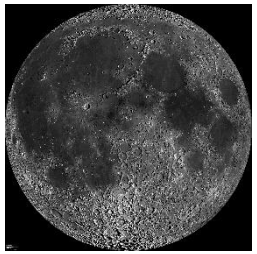
- Download the *Moon.jpg* file from here

https://facweb.iittp.ac.in/~pmariappan/Courses/MA517M/Moon.jpg

- Upload this file to your MATLAB Drive
- Do the following on MATLAB Online




```
I = imread('GrayScale.png');  % Read an image from file
imshow(I);                    % Display the image
title('Original Image');
```



```
I = imread('Moon.jpg');  % Read an image from file
imshow(I);               % Display the image
title('Original Image');
```

Reading Color Image

- Download the *Peppers.jpg* file from here

https://facweb.iittp.ac.in/~pmariappan/Courses/MA517M/Peppers.jpg

- Upload this file to your MATLAB Drive
- Do the following on MATLAB Online

```
I = imread('Peppers.jpg');  % Read an image from file

I_gray = rgb2gray(I);  % Convert color image to grayscale
subplot(1,2,1);
imshow(I); title('Original Image');

subplot(1,2,2);
imshow(I_gray); title('Gray Scale Image');
```

Rotating Image

```
I = imread('Peppers.jpg');  % Read an image from file

subplot(2,2,1);
imshow(I); title('Original Image');

subplot(2,2,2);
imshow(flipud(I)); title('Flip vertically');

subplot(2,2,3);
imshow(fliplr(I)); title('Flip horizontally');

subplot(2,2,4);
imshow(imrotate(I,45)); title('Rotated 45 Deg');
```

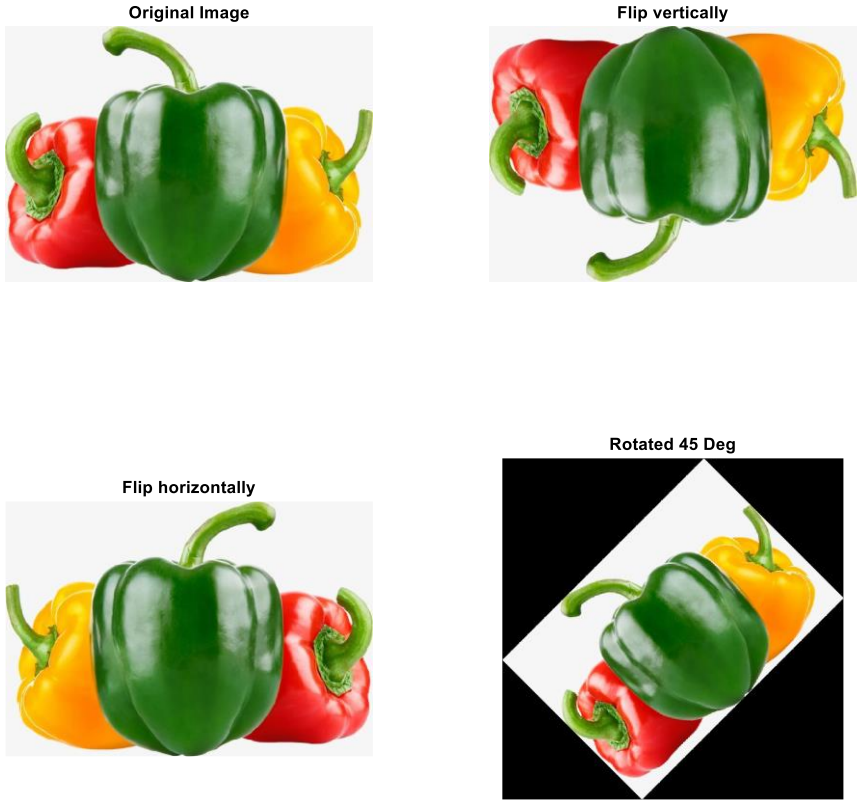


Image Cropping and Resizing

```
I = imread('Peppers.jpg');  % Read an image from file
I_crop = imcrop(I,[50 50 200 200]);
I_small = imresize(I, 0.5);

subplot(2,2,1);
imshow(I);
title('Original Image');

subplot(2,2,2);
imshow(I_crop);
title('Crop Image');

subplot(2,2,3);
imshow(I_small);
title('Resized Image');
```

Original Image

Crop Image

Resized Image

RGB Visualization

```
I = imread('Peppers.jpg');  % Read an image from file
imshow(I);                  % Display the image
R = I(:,:,1); G = I(:,:,2); B = I(:,:,3);
subplot(2,2,1);
imshow(I);
title('Color Image');

subplot(2,2,2);
imshow(R);
title('Red Channel');

subplot(2,2,3);
imshow(G);
title('Green Channel');

subplot(2,2,4);
imshow(B);
title('Blue Channel');
```

Color Image

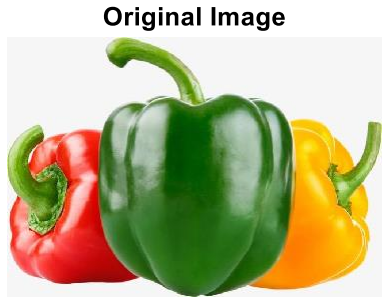
Red Channel

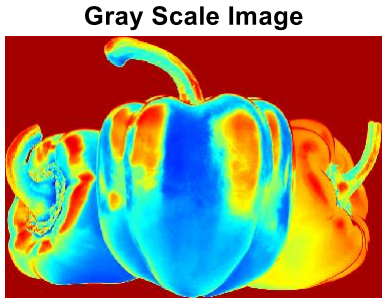
Green Channel

Blue Channel

Colormap Visualization

```
I = imread('Peppers.jpg');  % Read an image from file
I_gray = rgb2gray(I);  % Convert color image to grayscale
subplot(2,2,1)
imshow(I);                  % Display the image
title('Original Image');

subplot(2,2,2)
imshow(I_gray);
title('Gray Scale Image');
subplot(2,2,3)
imagesc(I_gray);
colormap('jet');
colorbar;
title('Color mapImage');
```





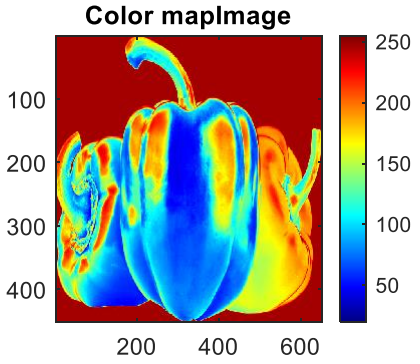


Image Arithmetic

```
I = imread('Peppers.jpg');  % Read an image from file
I_bright = I + 50;
I_highcontrast = imadjust(I, [0.2 0.8], []);
subplot(1,2,1);
imshow(I);
title('Color Image');

subplot(1,2,2);
imshow(rgb2gray(I_highcontrast));
title('Contrast Image');
```

Color Image

Contrast Image

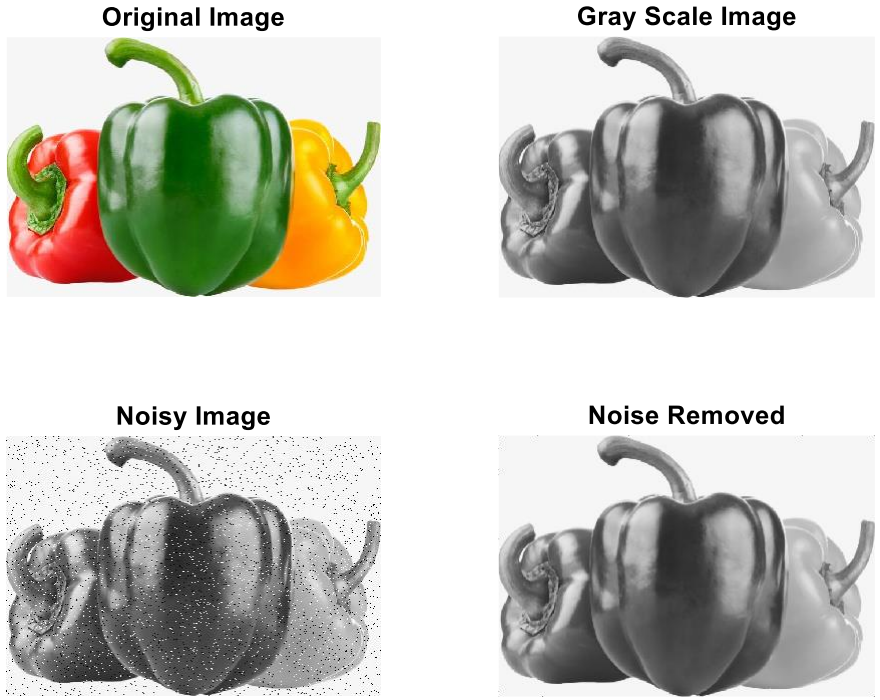
Image Blending

```
I1 = imread('Peppers.jpg');  % Read an image from file
I2 = imread('Onion.jpg');
I2 = imresize(I2, size(I1(:,:,1)));
blend = 0.6*I1 + 0.4*I2;
subplot(2,2,1);
imshow(I1);

subplot(2,2,2);
imshow(I2)

subplot(2,2,3);
imshow(uint8(blend));
```

Download the *Onion.jpg* file from here
https://facweb.iittp.ac.in/~pmariappan/Courses/MA517M/Onion.jpg

Noisy Images

```
I = imread('Peppers.jpg');  % Read an image from file
I_gray = rgb2gray(I);  % Convert color image to
grayscale
subplot(2,2,1)
imshow(I); title('Original Image');

subplot(2,2,2)
imshow(I_gray); title('Gray Scale Image');

I_noisy=imnoise(I_gray,'salt & Pepper',0.05);
subplot(2,2,3);
imshow(I_noisy); title('Noisy Image');

I_filtered = medfilt2(I_noisy);
subplot(2,2,4);
imshow(I_filtered); title('Noise Removed');
```



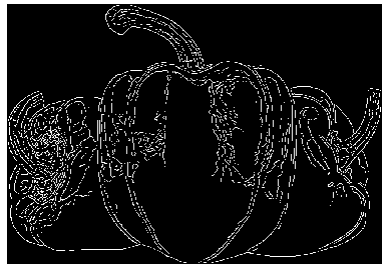
Edge Detection

```
I = imread('Peppers.jpg'); % Read an image from file
I_gray = rgb2gray(I); % Convert color image to grayscale
subplot(2,2,1)
imshow(I); title('Original Image');


subplot(2,2,2)
imshow(I_gray); title('Gray Scale Image');


subplot(2,2,3);
BW = edge(I_gray,'canny');
imshow(BW); title('Edge Detected');


subplot(2,2,4);
imshowpair(I,BW,'montage');
title('Edge Pair');
```

Original Image

Gray Scale Image

Edge Detected



Edge Pair

Segmentation

```
I = imread('Peppers.jpg');  % Read an image from file
I_gray = rgb2gray(I);  % Convert color image to
grayscale
subplot(2,2,1)
imshow(I); title('Original Image');

subplot(2,2,2)
imshow(I); title('Gray Scale Image');

BW = imbinarize(I_gray, graythresh(I_gray));
subplot(2,2,3);
imshow(BW); title('Edget Detected');
```

Original Image

Gray Scale Image

Segmentation

Singular Value Decomposition

Panchatcharam M

Data Compression

- Image, Audio, Video Compression
- Linear Transformations like Fourier Transformation
- In the Fourier Space, information can be cut away without disturbing the main information
- Cut a picture, audio and video into smaller junks
- If U is the Fourier transformation and P is a cut off function, then $y=PUx$ is transferred or stored on a CD or DVD
- The receiver like the DVD player recovers $U^T y$ which is close to x, but with small noise or error which human eye or ear does not recognize

Image Compression

- Hundreds of ways to compress images
- *Singluar Value Decomposition*
- 9MP gray scale image, which is 3000 x 3000 pixel.
- For each pixel, we have some level of black and white, given by values between 0 and 255. For each pixel, we require 1 bit of store, that is approximately 8.6Mb
- In case of RGB image, it is 25.8Mb
- Let us see how to compress them using SVD

Singular Value Decomposition

- Any non-zero real *m x n* matrix can be factored as $A = U\Sigma V^*$, where *U* and *V* are an *m x m* and *n x n* unitary matrices respectively, Σ is a diagonal *m x n* matrix with non-negative real numbers on the diagonal
- Σ entries are known as singular values
- Spectral Theorem: If *A* is a symmetric matrix, then the entries of Σ are eigenvalues of *A*, U=V=Orthogonal matrix of eigenvectors
- For

$$\mathrm{A^T A} = V\Sigma U^T \mathrm{U\Sigma V^T} = \mathrm{V\Sigma^2 V^T}, \mathrm{AA^T} = \mathrm{U\Sigma V^T V\Sigma U^T} = \mathrm{U\Sigma U^T}$$

Both are spectral decompositions, Σ : positive square roots of the eigenvalues of $\mathrm{A^T A}$.

Usually, matrices are rearrange so that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$

Reducing the SVD

- An invertible matrix A can be written as

$$A = U\Sigma V^T = [u_1, u_2, .., u_n]\begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_n \end{bmatrix}\begin{bmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_n^T \end{bmatrix} = \sum p_i \sigma_i q_i^T$$

Since $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$, $p_i \sigma_i q_i^T$ with small *i* contribute most to the sum. Keeping a few terms results poor image quality but lower storage space. Principal Component Analysis

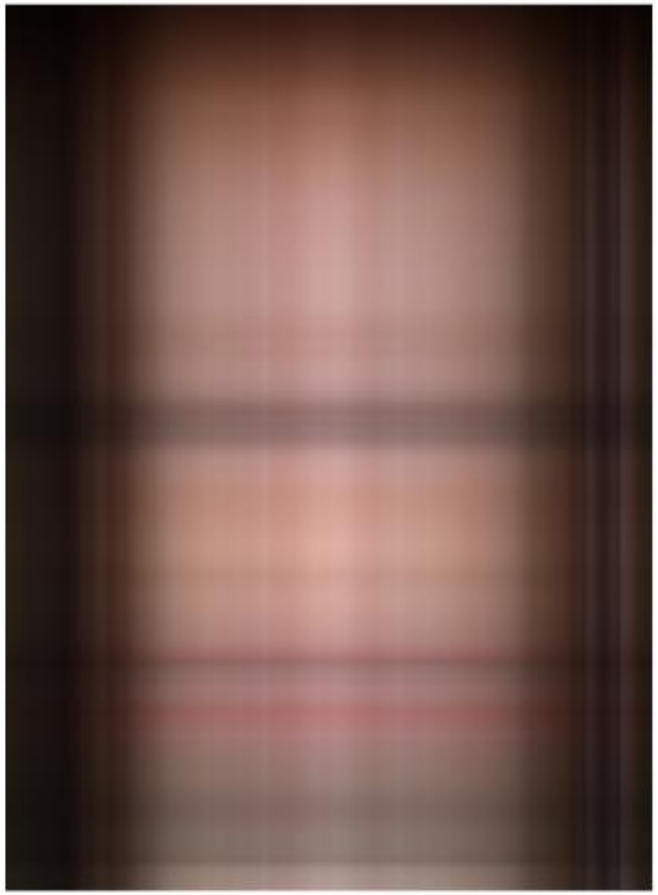
Reducing the SVD

- *U, V, A* requires n^2 elements. Σ requires *n* elements. Total, $2n^2 + n$ elements
- Keeping 1 term in SVD requires *2n+1* elements
- If we keep $k = \frac{n}{2}$ terms, then reduced SVD and original matrix are approximately same
- Error $E = 1 - \frac{\sum_1^k \sigma_i}{\sum_1^n \sigma_i}$
- Color images are stored using three matrices. Hence three reduced SVD

*500 * 500 image*

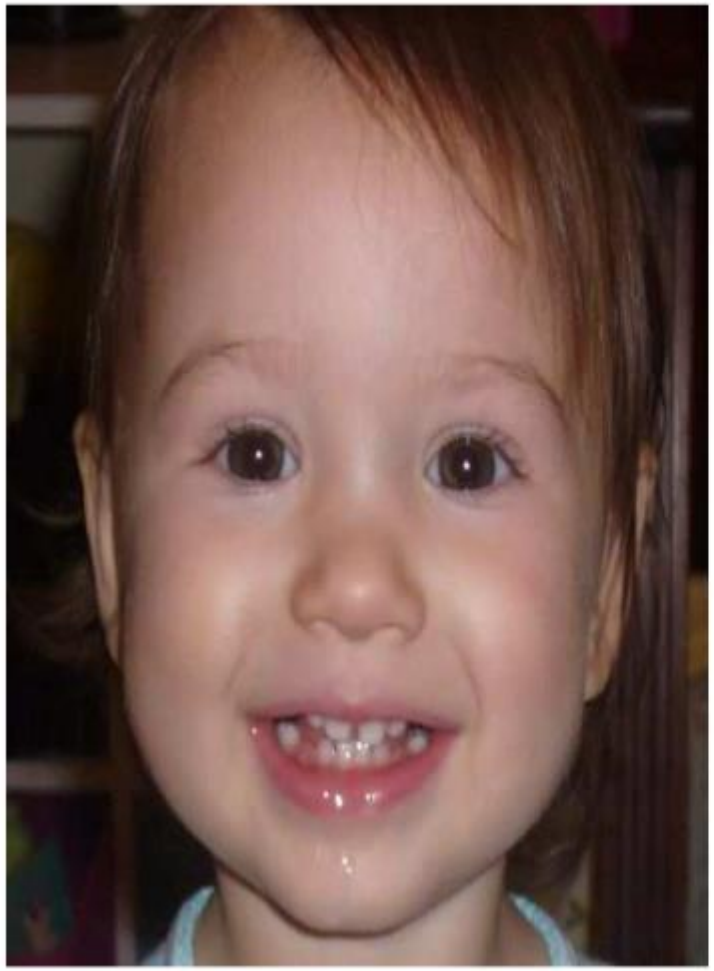
Original

Using 1 terms



*500 * 500 image*

Original

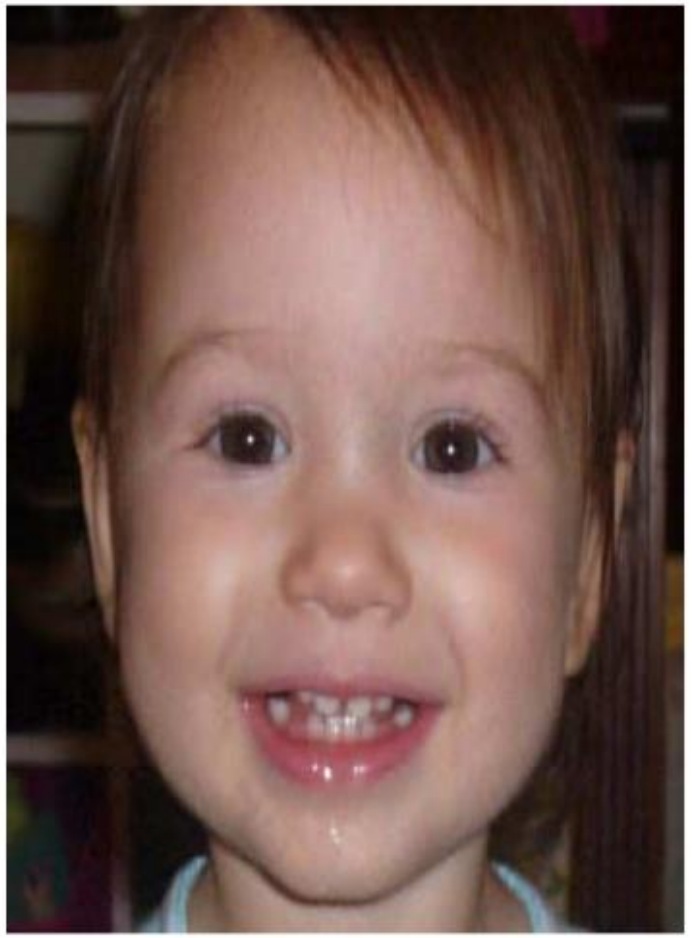


Using 5 terms

*500 * 500 image*

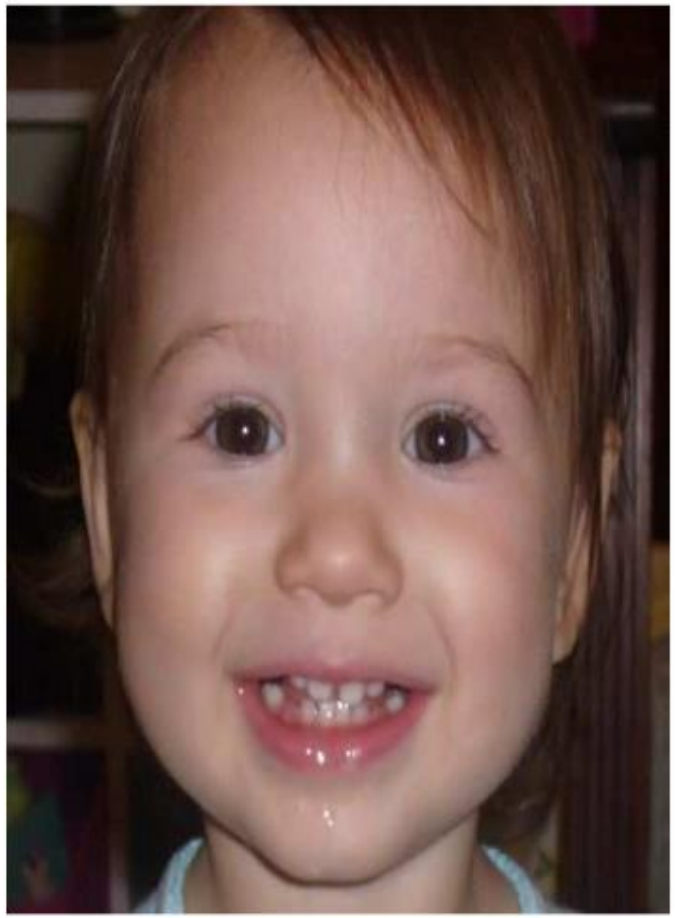
Original

Using 50 terms

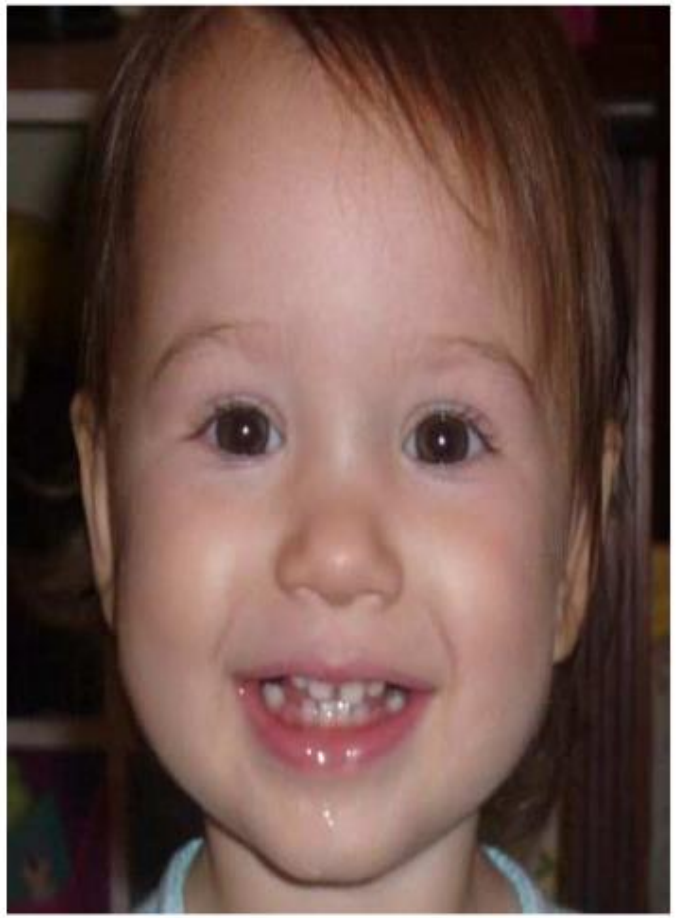


*500 * 500 image*

Original



Using 100 terms



Reducing the SVD

- *k=100,* accurate reproduction with 7.53% error
- *k=100, 40%* of the original image size

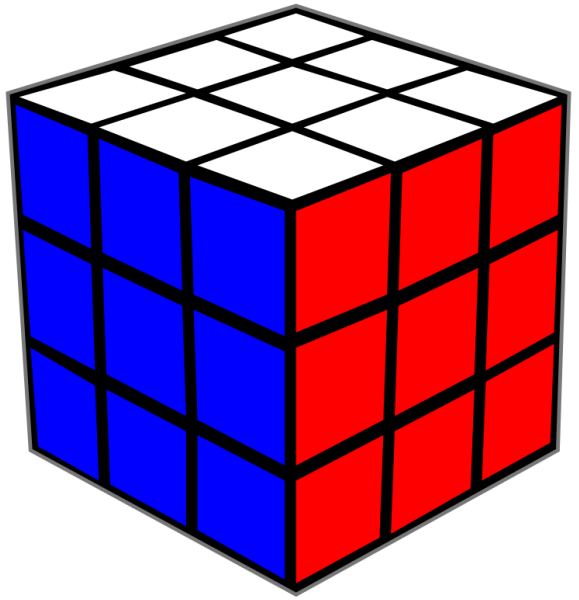
Tensors and Tensor Toolbox

Panchatcharam Mariappan

Associate Professor

Department of Mathematics,
IIT Tirupati

Tensors



What is Tensor?

- An algebraic object that describes multilinear relationship between sets of algebraic objects related to a vector space
- It is a generalization of scalars and vectors.

- Rank (or order) of a Tensor: Number of directions required to describe it.
- The dimensionality of the array

- Zero rank tensor → Scalar
- First rank tensor → Vector
- Second rank tensor → Matrix

Note on Tensor

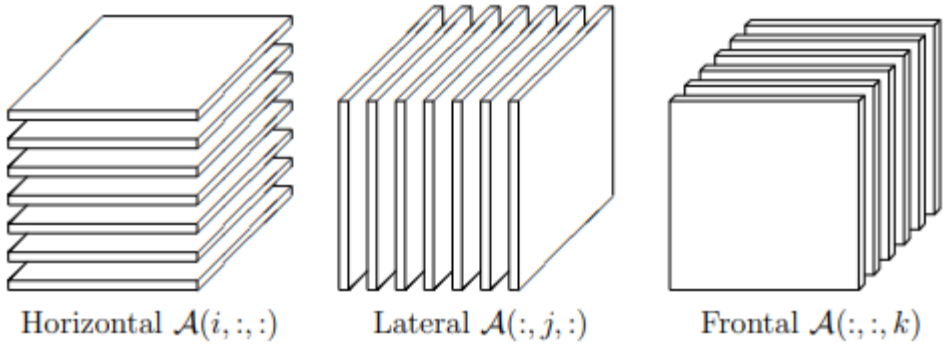
- Scalars and vectors are special cases of a more general object called tensor of order n
- Tensor specification requires 3^n numbers for a tensor of order n
- Scalar requires $3^0 = 1$ component
- Vectors requires $3^1 = 3$ components

Tensor in Matlab

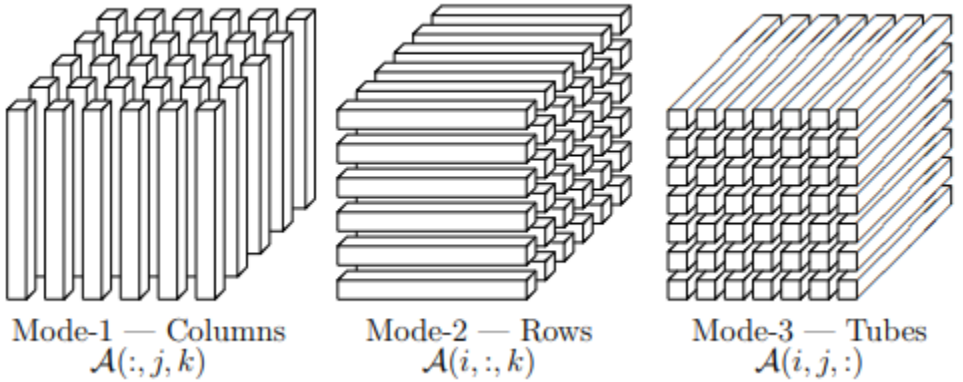
```
>> A=rand(3,4,2)

A(:,:,1) =

    0.1386    0.8407    0.2435    0.1966
    0.1493    0.2543    0.9293    0.2511
    0.2575    0.8143    0.3500    0.6160


A(:,:,2) =

    0.4733    0.5853    0.2858    0.3804
    0.3517    0.5497    0.7572    0.5678
    0.8308    0.9172    0.7537    0.0759
```

```
>> T=tensor(A)
T is a tensor of size 3 x 4 x 2
        T(:,:,1) =
            0.1386    0.8407    0.2435    0.1966
            0.1493    0.2543    0.9293    0.2511
            0.2575    0.8143    0.3500    0.6160
        T(:,:,2) =
            0.4733    0.5853    0.2858    0.3804
            0.3517    0.5497    0.7572    0.5678
            0.8308    0.9172    0.7537    0.0759
```

Slices and Fibers of Third order Tensor



Slices



Fibers

Applications of Tensor

- Image Processing (RGB images)
- Machine Learning (Neural Network)
- Continuum Mechanics (Cauchy Stress tensor, strain tensor)
- Quantum Mechanics (Pauli matrices)
- Differential Geometry (Riemann Curvature Tensor)
- Fluid Dynamics (Stress Tensor)
- Economics and Finance (Multivariate time series data)
- Weather prediction (pressure, velocity, temperature, point is space and time)

Exercise

- Download the following Tensor Toolbox:
- https://gitlab.com/tensors/tensor_toolbox/-/releases/v3.6
- Compute SVD for a gray scale image using matrices
- Compute SVD for an RGB image using tensors
- You can read this document for reference: https://www.osti.gov/biblio/974890