Digital Image Manipulation: Modern Applications of Linear Algebra

Oscar Alvarez and Hannah Grant

Department of Mathematics and Computer Science
Texas Woman's University
Advisor: Dr. Alicia Machuca

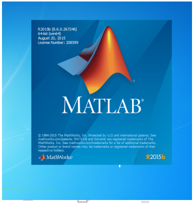
April 18, 2017

Overview

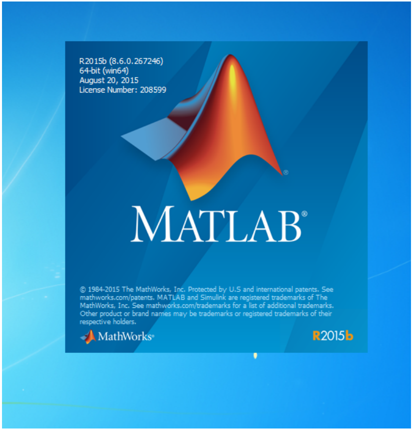
1. Project 1
   - Image Blending Using Convex Combinations
2. Project 2
   - Image Compression Using the SVD (Singular Value Decomposition)
3. Tutorials

Using Linear Algebra “in real life”

- It is important to introduce students to real life applications of linear algebra
  - Motivates learning
  - Develops interest
- Google, Netflix, NFL, data mining, etc.
  - Websites that you may like
  - New movie suggestions from ones you have already seen
- MATLAB
  - Software used

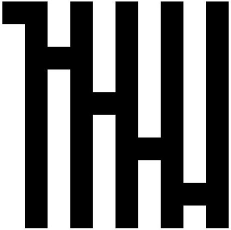


MATLAB



- Mathematics Engine Software
- Used by scientists and engineers around the world
- Uses a matrix-based programming language for mathematical computation

Image as a matrix using MATLAB



$$= \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Where 0 is black and 1 is white.
For a gray scale image, the pixel value is between 0 and 1.

Convex Combinations

- Convex combinations

$$I = \alpha J + (1 - \alpha)B$$

  - J and B are matrices
  - $0 \leq \alpha \leq 1$
  - $\alpha + (1 - \alpha) = 1$

- Applying convex combinations to image blending, where J and B are images.

Convex Combinations and Image Blending

Image J

Image B

When α=.40

(a) αJ (b) $(1-\alpha)B$ (c) $\alpha J + (1-\alpha)B$

Convex Combinations and Image Blending

MATLAB Program

```
clear

%read first image and save as a matrix
sqrl = imread('squirrel.jpeg');
%convert color image to black and white
BWsqrl = rgb2gray(sqrl);
%save image with double precision
J = double(BWsqrl);

%read second image and save as a matrix
bgbooks = imread('operahouse.jpeg');
%convert color image to black and white
BWbgbooks = rgb2gray(bgbooks);
%save image with double precision
B = double(BWbgbooks);

%for loop
for alpha=[.40]

    %convex combination using image J and image B
    %we use (:,:) to define J and B as matrices
  Blended = alpha*J(:,:)+(1-alpha)*B(:,:);

  %show the blended image
  figure;
  imshow(uint8(Blended))


end
```

The SVD (Singular Value Decomposition)

$$A = U\Sigma V^T$$

$$A = \begin{bmatrix} u_{11} & u_{12} & . & . & . & u_{1k} \\ u_{21} & . & & & & \\ . & & . & & & \\ . & & & . & & \\ . & & & & & \\ u_{i1} & & & & & u_{ik} \end{bmatrix} \begin{bmatrix} s_{11} & 0 & . & . & . & 0 \\ 0 & s_{22} & & & & \\ . & & . & & & \\ . & & & . & & \\ . & & & & & \\ 0 & & & & & 0 \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} & . & . & . & v_{1k} \\ v_{21} & . & & & & \\ . & & . & & & \\ . & & & . & & \\ . & & & & & \\ v_{i1} & & & & & v_{ik} \end{bmatrix}^T$$

- The matrix U is formed by the eigenvectors of AA^T.
- Σ is the matrix with the singular values of A on the diagonal.
- Matrix V is formed by the normalized eigenvectors of A^TA.

The SVD (Singular Value Decomposition)

- How can we apply the SVD?
  - Singular Values
    - Values that define the distance in relationship between related matrices.
  - Rank
    - The number of linearly independent rows a matrix contains.

By reducing the rank of the matrix that contains the singular values (Σ), we can eliminate values that are not as crucial.

Image Compression using the SVD

Break up an image matrix using the SVD

```
mtwu =

   117   117   117   117   117   117   117   117   117   117   117   117   117   117   117
   117   201   201   201   117   166   117   117   117   166   117   218   117   218   117
   117   117   201   117   117   166   117   117   117   166   117   218   117   218   117
   117   117   201   117   117   166   117   166   117   166   117   218   117   218   117
   117   117   201   117   117   117   166   117   166   117   117   218   218   218   117
   117   117   117   117   117   117   117   117   117   117   117   117   117   117   117
```

$=$

```
U =

   -0.3344    0.3913   -0.4605   -0.1428   -0.0508   -0.7071
   -0.4571   -0.4912   -0.3049    0.6653    0.1195    0.0000
   -0.4266   -0.3175    0.2149   -0.2915   -0.7655   -0.0000
   -0.4350   -0.2969    0.1665   -0.5537    0.6231    0.0000
   -0.4429    0.5133    0.6396    0.3537    0.0787    0.0000
   -0.3344    0.3913   -0.4605   -0.1428   -0.0508    0.7071

S =

   1.0e+03 *

    1.3337         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0    0.1355         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0    0.1266         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0    0.0867         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0    0.0298         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0    0.0000         0         0         0         0         0         0         0         0         0

V =

   -0.2132    0.1643   -0.1894   -0.1509   -0.1798    0.9146    0.0000    0.0000    0.0000   -0.0000    0.0000   -0.0000   -0.0000   -0.0000         0
   -0.2420   -0.1401   -0.3917    0.4937    0.1570    0.0000    0.0161    0.6922    0.0161   -0.0412    0.0137   -0.0946    0.0186   -0.0946    0.0137
   -0.3242   -0.2028    0.2858    0.0174   -0.0225    0.0185    0.1521   -0.1768    0.1521   -0.1907    0.1483   -0.5425    0.1562   -0.5425    0.1483
   -0.2420   -0.1401   -0.3917    0.4937    0.1570    0.0000   -0.0161   -0.6922   -0.0161    0.0412   -0.0137    0.0946   -0.0186    0.0946   -0.0137
   -0.2132    0.1643   -0.1894   -0.1509   -0.1798   -0.1787    0.2502    0.0015    0.2502   -0.3753   -0.1385    0.1748    0.6627    0.1748   -0.1385
   -0.2617   -0.2354   -0.1598   -0.2526   -0.2175   -0.1362   -0.3310   -0.0090   -0.3310   -0.3161   -0.3941   -0.1033   -0.2641   -0.1033   -0.3941
   -0.2295    0.3499    0.0582    0.0490   -0.0504   -0.1061    0.7507    0.0039   -0.2493    0.0541   -0.1255   -0.0105   -0.3807   -0.0105   -0.1255
   -0.2292    0.0570   -0.1249   -0.4639    0.8447    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000
   -0.2295    0.3499    0.0582    0.0490   -0.0504   -0.1061   -0.2493    0.0039    0.7507    0.0541   -0.1255   -0.0105   -0.3807   -0.0105   -0.1255
   -0.2617   -0.2354   -0.1598   -0.2526   -0.2175   -0.1362    0.0554    0.0309    0.0554    0.8146   -0.0709   -0.0465    0.1895   -0.0465   -0.0709
   -0.2132    0.1643   -0.1894   -0.1509   -0.1798   -0.1787   -0.1291    0.0012   -0.1291   -0.0734    0.8539    0.0386   -0.1112    0.0386   -0.1461
   -0.3466   -0.2771    0.3819    0.0515    0.0094    0.0584    0.0036    0.0682    0.0036   -0.0416    0.0511    0.7620   -0.0469   -0.2380    0.0511
   -0.2468    0.5469    0.3209    0.2611    0.0870   -0.0292   -0.3769    0.0068   -0.3769    0.1893   -0.1038   -0.0626    0.3333   -0.0626   -0.1038
   -0.3466   -0.2771    0.3819    0.0515    0.0094    0.0584    0.0036    0.0682    0.0036   -0.0416    0.0511   -0.2380   -0.0469    0.7620    0.0511
   -0.2132    0.1643   -0.1894   -0.1509   -0.1798   -0.1787   -0.1291    0.0012   -0.1291   -0.0734   -0.1461    0.0386   -0.1112    0.0386    0.8539
```

Image Compression using the SVD

Σ (S) matrix contains all of the singular values.

```
S =

   1.0e+03 *

    1.3337         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0    0.1355         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0    0.1266         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0    0.0867         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0    0.0298         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0    0.0000         0         0         0         0         0         0         0         0         0
```

Reduce Σ matrix size retaining only the first-most, largest values.

```
C =

   1.0e+03 *

    1.3337         0         0
         0    0.1355         0
         0         0    0.1266
```

Reduce the size of other two matrices to multiply.

```
A =

   -0.3344    0.3913   -0.4605
   -0.4571   -0.4912   -0.3049
   -0.4266   -0.3175    0.2149
   -0.4350   -0.2969    0.1665
   -0.4429    0.5133    0.6396
   -0.3344    0.3913   -0.4605
```

```
E =

   -0.2132    0.1643   -0.1894
   -0.2420   -0.1401   -0.3917
   -0.3242   -0.2028    0.2858
   -0.2420   -0.1401   -0.3917
   -0.2132    0.1643   -0.1894
   -0.2617   -0.2354   -0.1598
   -0.2295    0.3499    0.0582
   -0.2292    0.0570   -0.1249
   -0.2295    0.3499    0.0582
   -0.2617   -0.2354   -0.1598
   -0.2132    0.1643   -0.1894
   -0.3466   -0.2771    0.3819
   -0.2468    0.5469    0.3209
   -0.3466   -0.2771    0.3819
   -0.2132    0.1643   -0.1894
```

Image Compression using the SVD

Multiply new matrices together to form new matrix with 3 singular values.

A =

-0.3344	0.3913	-0.4605
-0.4571	-0.4912	-0.3049
-0.4266	-0.3175	0.2149
-0.4350	-0.2969	0.1665
-0.4429	0.5133	0.6396
-0.3344	0.3913	-0.4605

$\times$

C =

1.0e+03 *

1.3337	0	0
0	0.1355	0
0	0	0.1266

$\times$

E = ^{T}

-0.2132	0.1643	-0.1894
-0.2420	-0.1401	-0.3917
-0.3242	-0.2028	0.2858
-0.2420	-0.1401	-0.3917
-0.2132	0.1643	-0.1894
-0.2617	-0.2354	-0.1598
-0.2295	0.3499	0.0582
-0.2292	0.0570	-0.1249
-0.2295	0.3499	0.0582
-0.2617	-0.2354	-0.1598
-0.2132	0.1643	-0.1894
-0.3466	-0.2771	0.3819
-0.2468	0.5469	0.3209
-0.3466	-0.2771	0.3819
-0.2132	0.1643	-0.1894

This creates our new, compressed, image.

=

D =

114.8603	123.3479	117.1814	123.3479	114.8602	113.5444	117.5300	112.5388	117.5300	113.5444	114.8602	117.6512	120.3633	117.6512	114.8602
126.3435	171.9656	200.0760	171.9656	126.3434	181.3437	114.3538	140.7445	114.3538	181.3437	126.3434	214.9981	101.6302	214.9981	126.3434
109.0838	133.0601	200.9275	133.0601	109.0837	154.6541	117.0889	124.5490	117.0889	154.6541	109.0837	219.5150	125.5843	219.5150	109.0837
113.0963	137.7838	202.2526	137.7838	113.0962	157.9130	120.2873	128.0467	120.2873	157.9130	113.0962	220.2962	127.9186	220.2962	113.0962
122.0483	101.4947	200.5190	101.4947	122.0483	125.2550	164.6161	129.2394	164.6161	125.2550	122.0483	216.4000	209.7902	216.4000	122.0483
114.8603	123.3479	117.1814	123.3479	114.8602	113.5444	117.5300	112.5388	117.5300	113.5444	114.8602	117.6512	120.3633	117.6512	114.8602

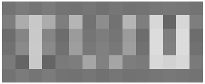
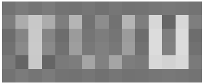


Image Compression using the SVD

$\rightarrow$



Reducing the Rank of the Σ matrix reduces image quality but saves memory.

This could prove useful when using high quality images on small phones with lower screen resolutions.

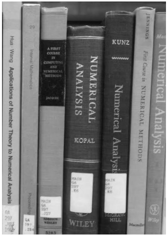
The less memory your wallpaper takes up, the faster your phone can be, all-the-while retaining and displaying a good quality image.

MATLAB Program

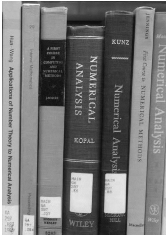
```
clear

bork = imread('twu.png');   % import the image file on this line
bwtwu = rgb2gray(bork);     % convert the image file from color to black and white on this line
mtwu = single(bwtwu);       % convert the black and white image into a matrix with values of single presision

[U,S,V]=svd(mtwu,0);        % take the SVD of the image matrix using this function command



for k=[400 200 150 20]      % select the amount of singular values desired. It will create a new image for each value specified.

    for i=1:k;              % this 'for' loop reduces the size of each matrix in the SVD to the size that is input. Do not touch.

        A(:,i) = U(:,i);
        C(i,i) = S(i,i);
        E(:,i) = V(:,i);

    end

    D = A*C*E';             % D will be our new matrix that makes up our compressed image. E is transposed here.

    figure;
    imshow(uint8(D))        % Displays new image. "uint8" is required in order to create the image.

end
```

Image Compression using the SVD

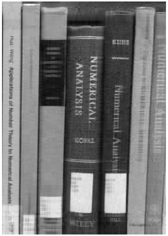
SVD Rank 421



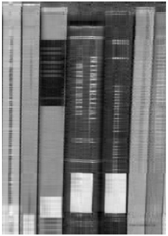
SVD Rank 150



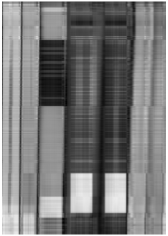
SVD Rank 30



SVD Rank 15



SVD Rank 5



* 421 being the matrix rank of the original image

Tutorials

- Structure
  - Brief review of matrix manipulation and terms
  - Introduction to mathematical concepts within the projects
    - Convex Combinations
    - The SVD
  - Introduction to MATLAB exercises
  - Step by step process of the projects
    - Image blending
    - Image compression
- Student feedback
  - Google Forms

Recap

Recap:

- Image Blending using convex combinations
- Image Compression using the SVD (Singular Value Decomposition)
- Tutorials

Acknowledgements

- Dr. Alicia Machuca, Dr. Jian Zhang, Dr. Marie-Anne Demuynck
- TWU's Quality Enhancement Plan's Experiential Student Scholar Program
- Lamar University and TUMC

Bibliography

Austin, David *We Recommend a Singular Value Decomposition* http://www.ams.org/samplings/feature-column/fcarc-svd. Retrieved: June 2016.

Axler, Sheldon *Linear Algebra Done Right* (2010).

Chartier, Tim *When Life is Linear: From Computer Graphics to Bracketology* (2015).

Downing, Douglas *Dictionary of Mathematics Terms* (1995).

MathWorks *MATLAB: Major Update* http://www.mathworks.com/products/matlab/. Retrieved: July 2016