Homework 1: Extended Yale Faces Database - Eigenfaces
AMATH 482: Computational Methods for Data Analysis, Winter 2017

Sean Lam
seanlam@uw.edu

January 23, 2017

Abstract

This report investigates the use of singular value decomposition for analyzing the cropped images of the Extended Yale Faces Database B. The goal was to explore the power of the SVD and its use for image compression.

I. Introduction and Overview

Linear algebra plays a key role in almost every application of data analysis and computation. One of the most powerful tools of linear algebra is the singular value decomposition, which can decompose a matrix into three constitutive components. One useful application for this is for image compression, and more generally, data compression. Data compression allows computers and other electronic devices to store data in a format that requires less space than usual. This can lead to faster transmission times, faster run times for long computations, and the ability to store more data.

The cropped images set of the Extended Yale Faces Database B contains images of 38 people, with 64 images each under different poses and illumination conditions. All images were manually aligned, cropped, and re-sized to 192x168 pixel images. Unfortunately, 18 of the images were corrupted and were not included. The SVD was taken on a matrix containing all the images reshaped as column vectors. Different aspects of the SVD were explored, such as how it decomposed a matrix of faces into principal components, the rank of the face space, the singular value spectrum, and applications to image compression.

II. Theoretical Background

The SVD takes a matrix and decomposes it into three constitutive components. It takes the form:

$$\mathbf{A} = \mathbf{U\Sigma V^\star} \tag{1}$$

with the following three matrices:

$$\mathbf{U} \in \mathbb{C}^{m\times m} \text{ is unitary} \tag{2}$$

$$\mathbf{V} \in \mathbb{C}^{n\times n} \text{ is unitary} \tag{3}$$

$$\mathbf{\Sigma} \in \mathbb{R}^{m\times n} \text{ is diagonal} \tag{4}$$

The diagonal entries of Σ are nonnegative and are ordered from largest to smallest.

Computing the SVD is fairly straightforward if one considers these two matrix products:

$$\begin{aligned}\mathbf{AA^T} &= (\mathbf{U\Sigma V^*})(\mathbf{U\Sigma V^*})^\mathbf{T} \\ &= \mathbf{U\Sigma V^* V\Sigma U^*} \\ &= \mathbf{U\Sigma^2 U^*}\end{aligned} \tag{5}$$

and

$$\begin{aligned}\mathbf{A^TA} &= ((\mathbf{U\Sigma V^*})^\mathbf{T}(\mathbf{U\Sigma V^*}) \\ &= \mathbf{V\Sigma U^* U\Sigma V^*} \\ &= \mathbf{V\Sigma^2 V^*}\end{aligned} \tag{6}$$

Multiplying (5) and (6) by $\mathbf{U}$ and $\mathbf{V}$ respectively gives the two self-consistent eigenvalue problems:

$$\mathbf{AA^TU} = \mathbf{U\Sigma^2} \tag{7}$$

$$\mathbf{A^TAV} = \mathbf{V\Sigma^2} \tag{8}$$

This shows that the singular values are contained in $\mathbf{\Sigma}$ and the matrices $\mathbf{U}$ and $\mathbf{V}$ provide the orthonormal basis vectors.

The power of the SVD lies in its many mathematical properties and the fact that its existence can be guaranteed for every matrix regardless of dimension or whether or not it is real or complex.

In the context of image compression, the SVD breaks down images into principal orthogonal components, which directions are the columns of $\mathbf{U}$. These principal components can also be thought of as the components or characteristics of a face, like the eyes, mouth, and nose. $\mathbf{\Sigma}$ is a diagonal matrix of singular values from greatest to least. $\mathbf{\Sigma}$ shows "important" each of the principal components are. In other words, it shows how much of a contribution each principal component has to the image. Since σ_1 is the largest singular value , that means U_1 is the most influential component, and σ_2 is the second largest so U_2 is the second most influential component, and so on. $\mathbf{V}$ is the projection of an image on the orthogonal components $\mathbf{U}$, or the characteristics of a specific face. In

other words, it picks out how much weight each of the principal components has on a specific image. The SVD essentially reduces a rank R matrix to a rank r matrix, where R > r. This means that we can take a list of R unique vectors and find the low-rank approximation of that but with r unique vectors. We can apply this to images and find the best-rank approximation of an image originally composed of R components to r components, effectively compressing the image.

III. Algorithm Implementation and Development

The set of images was provided in 38 folders, each containing 64 images of a person in different postures/illuminations. 14 of the images were corrupted and were not included. Images were manually aligned, cropped, and re-sized to 192x168 pixels in portable graymap format (.pgm). Because the pictures were aligned and cropped, we did not have to worry about difference in angles between posture and positions of facial structures, the images were forced to have similar orientation. An algorithm was implemented to iterate through each folder and read all the .pgm files. .pgm files contain grayscale data for the 192x168 pixels in the image. Each pixel can be represented as a number between 0 to 255 where 0 is black and 1 is white, and everything else in between gray. Then for each image matrix, the algorithm reshapes them in a column vector of size 32256x1. This essentially cuts pictures into columns of pixels stacks those columns on top of each other. Each picture is a new column, with a total of 2414 columns (there were 2414 images), so the final matrix is size 32256x2414.

Then, the economy SVD was computed. The economic SVD computes faster than the full SVD while still retaining all relevant information needed for the purpose of image compression. The matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{\Sigma}$ were outputted.

IV. Computational Results

The singular value spectrum on Figure 1 covers a wide range, with the highest value being around 10^6 (or about 11% of total singular values). This means the first principal component contains information of about 11% of a face. The second singular value contains about 5% information, third and fourth contains about 2% information, fifth and sixth contain about 1%, and the rest decline pretty steadily from approximately 1% to almost 0.

Figure 2 shows the first six orthogonal basis vectors. The first vector that contains around 11% is the most basic image of a set of eyes, a nose, and a mouth. This can be thought of as the starting point of each image. Depending on what rank r approximation is desired, the image can be approximated as

$$\mathbf{A}_N = \sum_{j=1}^{r} \sigma_j \mathbf{u}_j \mathbf{v}_j^* \tag{9}$$

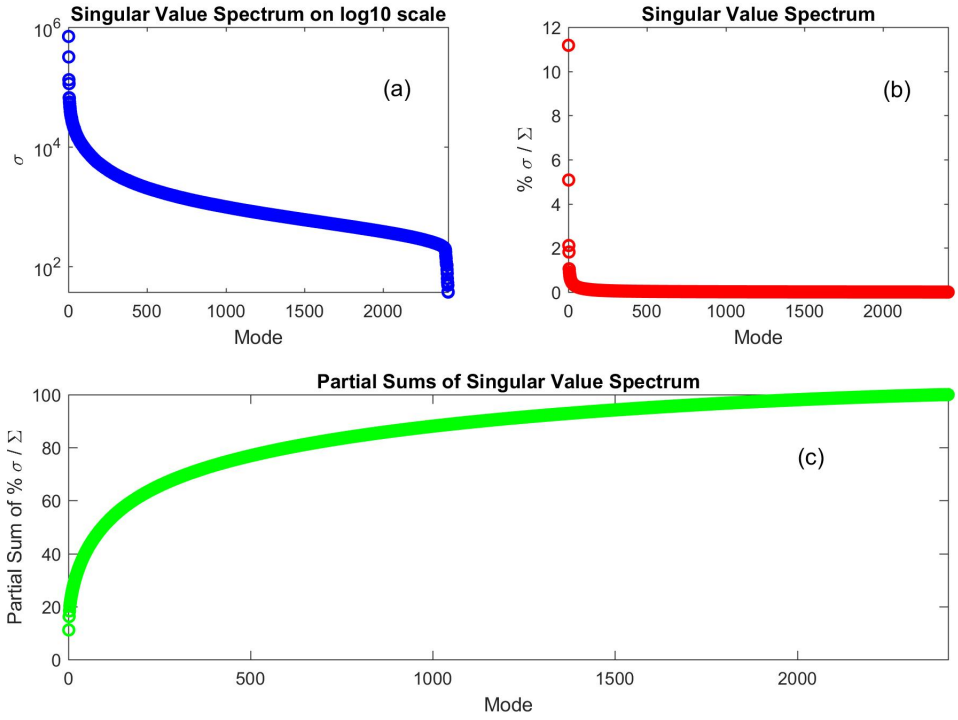

Figure 1: (a) Singular values on a log10 scaled y-axis plot. (b) Percentage of singular value over the total sum of singular values. (c) Cumulative partial sums of singular value over the total sum of singular values.

Figure 3 shows the low rank-approximation images of selected ranks, and the original image. Rank 1 contains about 11% of the information from the original image, which looks like a basic face containing a set of eyes, a nose, and a mouth. At rank 10, about 25% information, it is still impossible to compare faces with the respected person, but more light contrast and face shape starts to take form. By rank 50, about 41% information, the image becomes a comparable face. The face has more definition, and distinct characteristics such as freckles, scars, blemishes, eyebrows, and facial hair become apparent. By rank 100, at 50% information, the image becomes very similar to the original, only differing by sharpness and illumination. Rank 500, with about 77% information, looks pretty much identical to the original image, except it is slightly lighter.

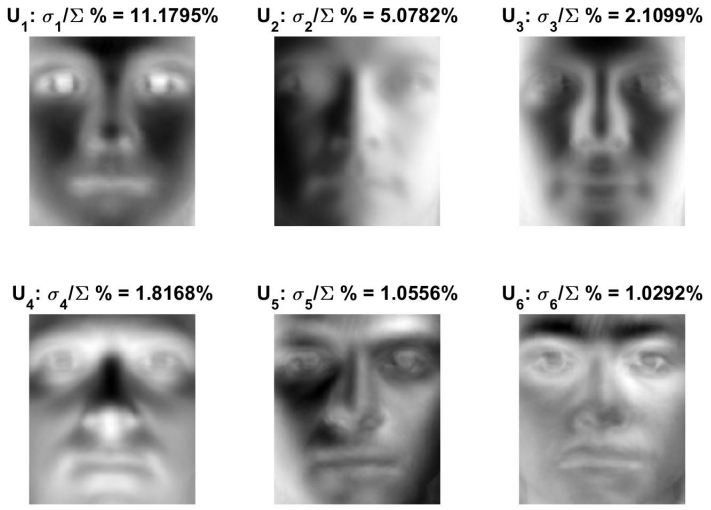

Figure 2: These images are the first six principal components with their corresponding σ/Σ percentages.

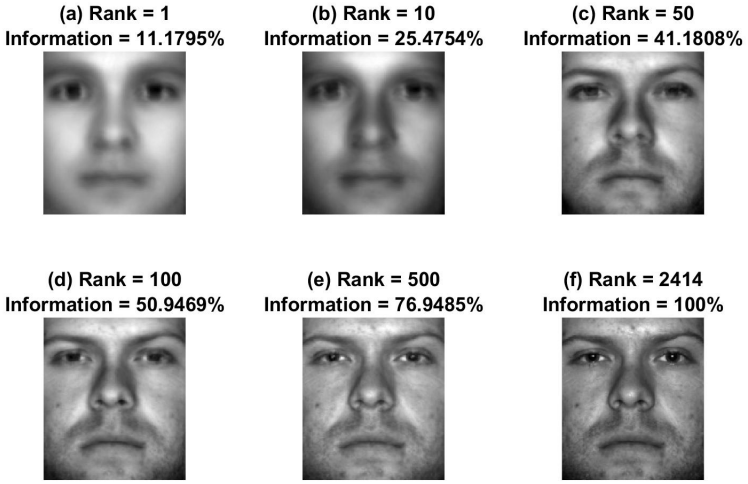

Figure 3: These are the low-rank approximations for the first image with rank (a) 1, (b) 10, (c) 50, (d) 100, (e) 500, (f) 2414.

V. Summary and Conclusions

This report used the SVD to analyze a set of faces and perform image compression. From taking the SVD of a matrix containing all the given cropped faces, the image was split up into three parts: $\mathbf{U}$, the orthogonal basis vectors that represented the image, $\mathbf{\Sigma}$, which shows how much each principal component contributes to a facial reconstruction, and $\mathbf{V}$, the projection of each individual image onto the principal components (how much information to grab from each principal component). From this, the singular value spectrum was found. Facial reconstruction started looking similar to the original image at rank 50. By rank 500, the approximated image looked almost identical to the original, with only differences in contrast and sharpness.

It is not obvious what the rank r for the face space are, since the singular value spectrum covers such a wide range of values at decline slowly once values get under 1% as shown in figure 1a, there is no sudden drop in values besides the last part at the end (about mode 2400). The number of modes necessary for good facial reconstruction depends on the application for which it is used for. At rank 10, with about 25% information, faces can be made distinct from one another. They all have the general features every face has: eyes, nose, cheeks, mouth, eyebrows, etc although each image's features will look slightly different. For simple facial recognition, where the goal is to identify the person, a rank of 50 will do a pretty consistent job of that with 41% information contained. Also, it is around this point where the small distinct features that are unique to this face, like freckles or blemishes, start showing however they are very light. A higher rank approximation is necessary to really bring out the detail in a unique face.

This exploratory project provides an example of how powerful of a tool the SVD is for working with very large data sets and high dimensional data. This linear algebra tool plays an integral role in almost every application of mathematics, including biology, economics, image processing, communications, medicine, engineering, and more.

Appendix A. MATLAB functions used and brief implementation explanation

dir - names the files in a directory
fullfile(foldername) - build full file name from parts
imread(filename) - reads an image from filename, turns it into a data matrix
reshape(A,[m,n]) - reshapes matrix A into m x n dimensions, keeping the same number of elements
svd(A) - performs the singular value decomposition on matrix A and outputs U, Σ, and V
imshow(A) - takes in a data matrix A and constructs the corresponding image
semilogy - plots points with a log10 based y-axis

Appendix B. MATLAB codes

Creating the data matrix and taking the SVD

```
people = dir(fullfile('CroppedYale')); % list of folders
people = {people.name}';
cd CroppedYale
numImages = 0;
% iterate through each folder
for ii = 3:numel(people)
    dirName = char(people(ii))
    cd (dirName)
    % find all .pgm files in folder
    files = dir(fullfile('*.pgm'));
    files = {files.name}';
    size = numel(files);
    for j = 1:size
        % read each image and reshape into column vector
        image = imread(fullfile(dirName,files{j}));
        image = double(image);
        image = reshape(image,192*168,1);
        A(:,j+numImages) = image; %add onto main matrix
    end
    numImages = numImages + size;
    cd ..
end
[U,S,V] = svd(A,'econ');
```

Plotting the singular value spectrum and related graphs

```
figure(1)
subplot(2,2,1),
semilogy(diag(S),'bo','Linewidth',[1.5])
title('Singular Value Spectrum on log10 scale')
xlabel('Mode')
ylabel('\sigma')
axis([0 2414 0 10^6])
text(2000,10^5,'(a)','Fontsize',[13])

subplot(2,2,2),
plot(100*diag(S)/sum(diag(S)),'ro','Linewidth',[1.5])
title('Singular Value Spectrum')
xlabel('Mode')
ylabel('% \sigma / \Sigma')
axis([0 2414 0 12])
text(2000,9.25, '(b)','Fontsize',[13])
```

```
subplot(2,2,[3,4]),
percentSum = 0;
Sdiag = diag(S);
for j = 1:length(S)
   current = 100*Sdiag(j)/sum(Sdiag);
   percentSum = current + percentSum;
   percentSumList(j, 1) = percentSum;
end
plot(percentSumList,'go','Linewidth',[1.5])
title('Partial Sums of Singular Value Spectrum')
xlabel('Mode')
ylabel('Partial Sum of % \sigma / \Sigma')
axis([0 2414 0 100])
text(2000,77,'(c)','Fontsize',[13])
```

Images of principal components

```
figure(2)
for j = 1:6
    subplot(2,3,j)
    imshow(reshape(U(:,j),[192,168]),[])
    title(['U_' num2str(j) ': \sigma_' num2str(j) '/\Sigma % = ' ...
        num2str(100*S(j,j)/sum(diag(S))) '%'])
end
```

Images of rank approximations

```
%Rank approximation
figure(3)
rank = [1 10 50 100 500];
S_rank = S;
captions = ['a'; 'b'; 'c'; 'd'; 'e'; 'f'];
for j = 1:5
   subplot(2,3,j)
   A_rank = U(:,1:rank(j)) * S_rank(1:rank(j), 1:rank(j)) * V(:,1:rank(j))';
   imshow(reshape(A_rank(:,1),[192,168]),[])
   str = {['(' captions(j,1) ') Rank = ' num2str(rank(j))];
       ['Information = ' num2str(percentSumList(rank(j),1)) '%']};
   title(str)
end
subplot(2,3,6)
imshow(reshape(A(:,1),[192,168]),[])
str = {['(f) Rank = 2414']; 'Information = 100%'};
title(str)
```