# SVD and Eigenfaces

Eigenface is a term first introduced by Sirovich and Kirby in 1987, which is a set of feature basis obtained by principle component analysis (PCA) building on singular value decomposition (SVD), to project the higher-dimensional face-image space to a lower dimension. So that a set of **r** basic features, which are called eigenpictures, and in this case, eigenfaces, can be used to perform linear combinations to achieve the total number of **n** face images, where **r < n**.

Eigenface is the fundamental concept used for facial recognition. Later on, various preprocessing algorithms, such as gamma correction for illumination preprocessing, were built upon it to improve accuracy.

## SVD

Singular value decomposition is to find 3 factors U, Σ, and of a real matrix M, where U and are orthogonal matrices, and Σ is a diagonal matrix and the entries on the diagonal are called singular values.

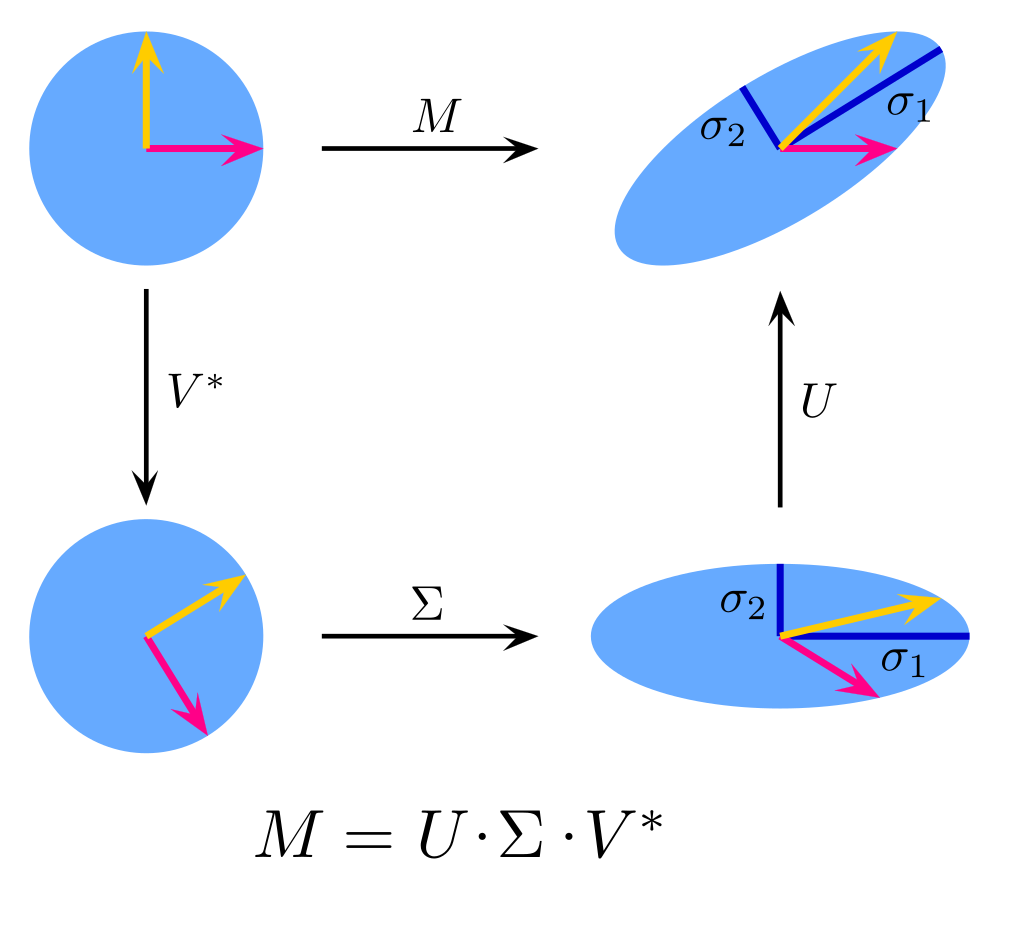


Figure 1: SVD by [Georg-Johann](https://commons.wikimedia.org/wiki/User:Georg-Johann)

The singular values in the Σ matrix are in descending order, i.e., if a matrix Σ has diagonal entries , , … from left to right,

then,

≥ ≥ ≥ … ≥ .

For an m×n matrix **M**, with n linearly independent columns (i.e., rank n), the dimensions of its full singular value decomposition matrices are:

U Σ

m×m m×n n×n

However, even if **m > n**, there will only be **n** non-trivial entries of σs in Σ, so the SVD matrices of **M** will be equivalent to consider only the first **n** rows of **Σ** and **n** columns of **U** since other entries of the product matrix of these 3 matrices will be 0s. Such SVD is called economy-sized decomposition.

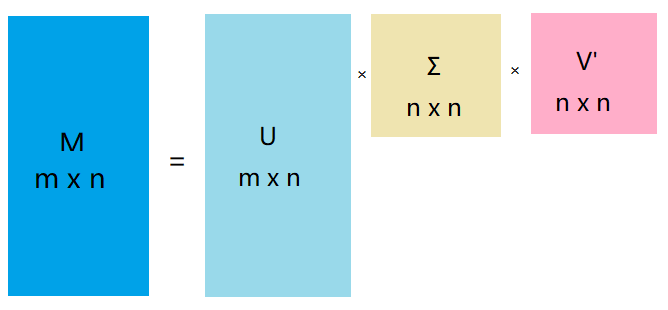


Figure 2: Economy-sized SVD

In the real-life scenario, an image of a person’s face can have millions of pixels with the development of photographic equipment. However, in that two-million-dimension vector space, there are directions containing much less information than others, so we would not want to analyze or compute or train algorithms with every bit of info in the two-million-dimension vector space; otherwise, it will increase the cost of time and space due to the amount of computation.

Singular value decomposition can be used to compress images by truncating SVD matrices to lower dimensions. Since the components with the most important information are ordered to be in the front, we can use the first **r** rows and columns to compress an image to reduce not-as-important information in the picture, by truncating the SVD matrices to rank **r**.

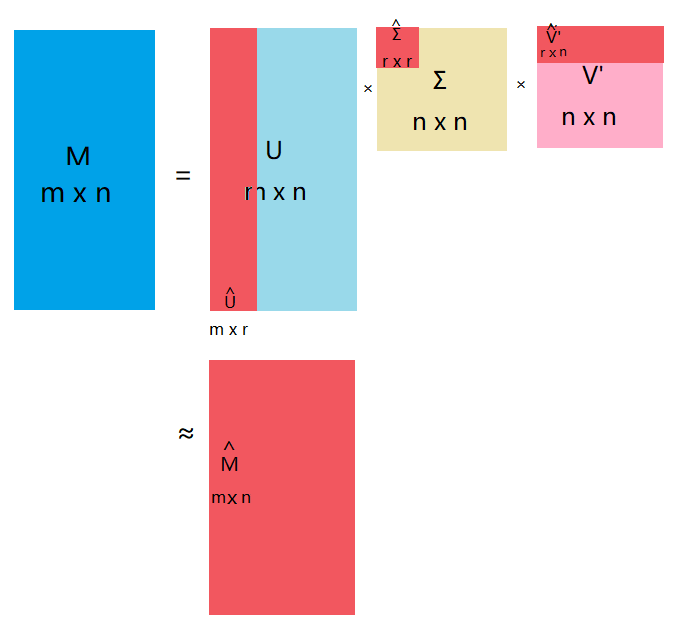


Figure 3: Approximated matrix M-hat by truncating SVD

I will use a picture of my cat with a resolution of this picture is 1920×1080 as an example of such approximation.

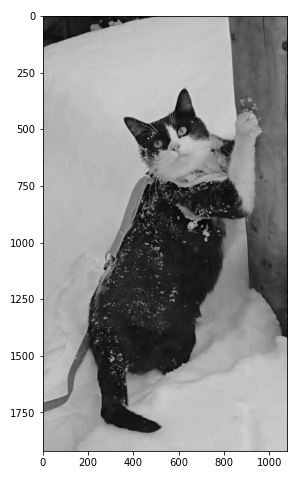


Figure 4: Original cat image

The components in this image include the cat (“mooncake”), its leash, a part of a trunk, and snow. Now we can use SVD to compress this picture to see if those pieces of information are reserved. SVD can be computed with linear algebra (linalg) package under numpy in python. We can first plot the singular values to see how much information each value has. And as shown in figure 5 and 6, it is obvious that the first couple of dozens of columns in SVD matrices will give us the majority of information from the original picture.

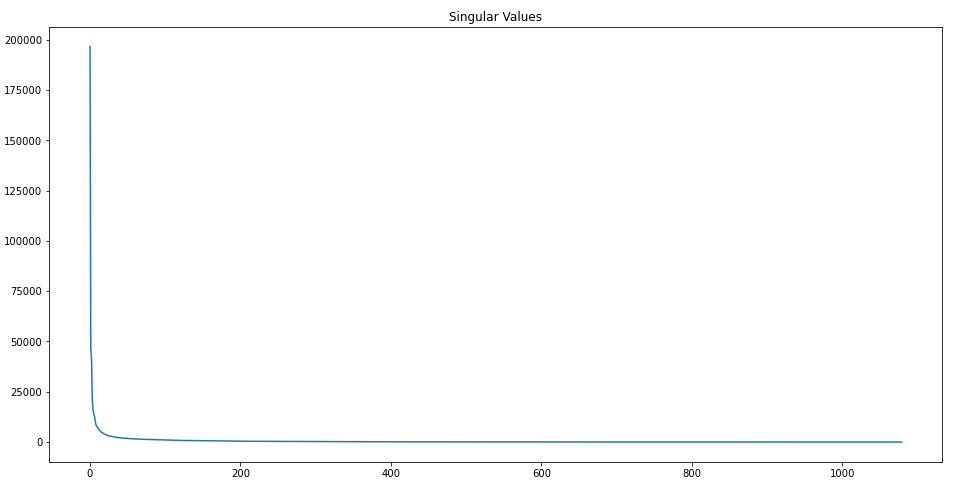


Figure 5: Plotted singular values

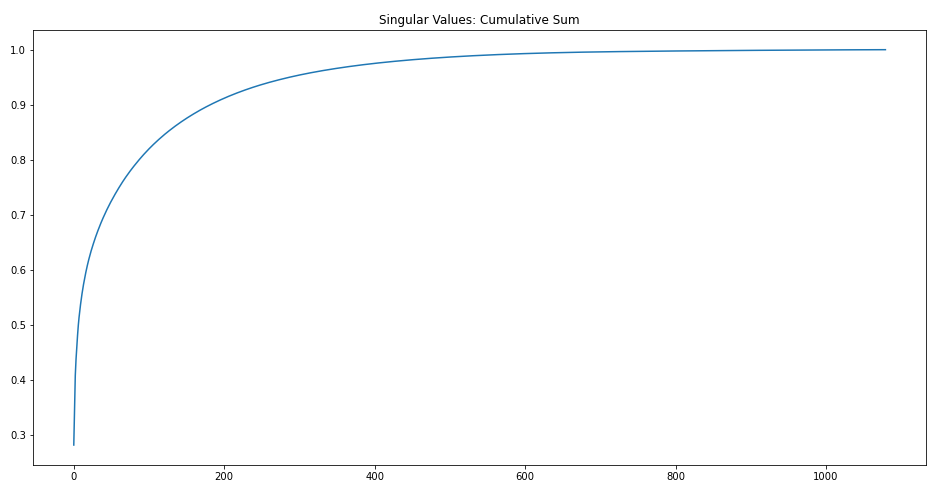


Figure 6: Cumulative proportions of singular values

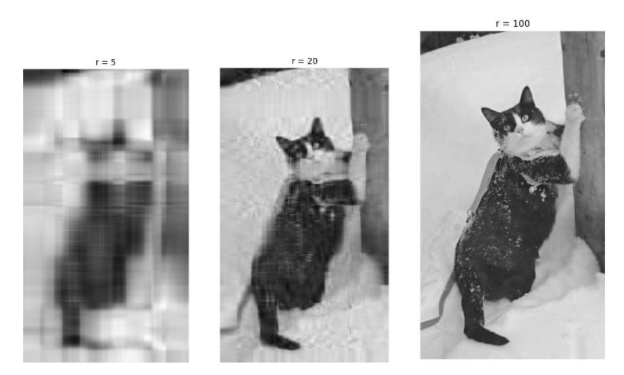


Figure 7: Compressed cat image to rank **r**

As shown in figure 6, when we truncate the SVD matrices by keeping **r** singular values, when **r** = 5, an image of color blocks is reproduced, where we can identify an animal, may be a rabbit, and an object on the right.

Moving on, the information we identified from the original picture is already able to be found with first 20 components. And by keeping 100 singular values, the compressed image is quite clear.

## Using Yale Face Database B

There are 38 human objects in total from [Yale Face Database B](http://vision.ucsd.edu/~leekc/ExtYaleDatabase/Yale%20Face%20Database.htm) and [Extended Yale Face Database B](http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html). The first 36 objects can be used as a training set and the last two faces can be used as a test set.

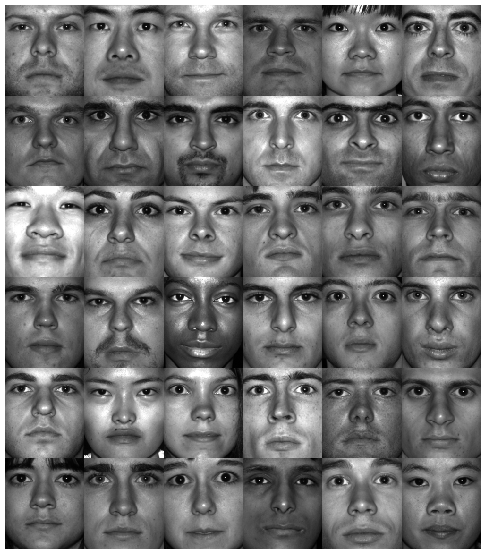


Figure 8: The first 36 gray-scaled objects from Yale Face Database B

Every face image has 168×192 pixels and images are aligned so that eyebrows, eyes, noses, and mouths are at a similar position in the pictures of the dataset. Face images are then being stretched to column vectors and all face-vectors are horizontally concatenated to construct a face matrix X.

To perform PCA, an average face needs to be calculated by finding the mean of all dimensions. The average face can be reshaped as an image after computing.



Figure 9: The average face

Then subtract the mean from face matrix **X** to recenter data around the origin. We can name the resulting matrix A.

After finding singular value decomposition of matrix A, column vectors in U are eigenvectors of the covariance matrix of A, which is also called eigenfaces.

The first 8 eigenfaces can be found by reshaping the first 8 columns of U (U[:, 0:7]) to size 168×192 matrices:

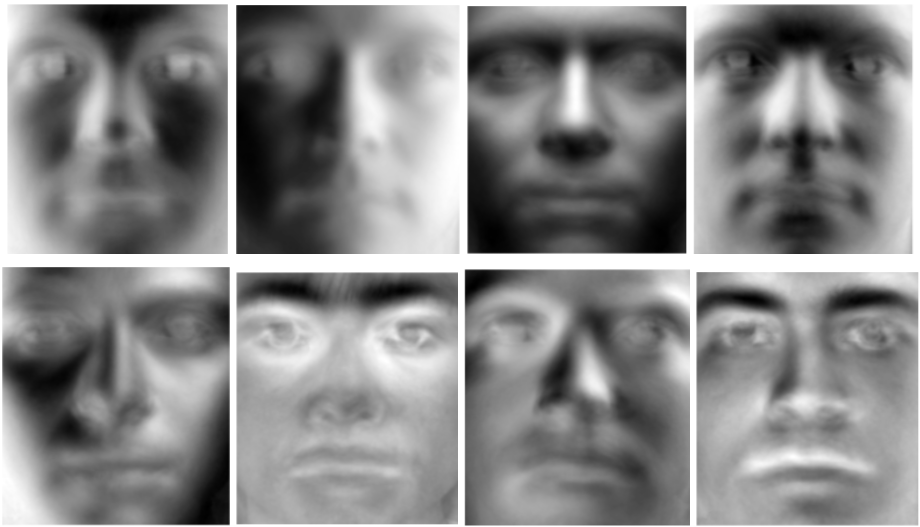


Figure 10: The first 8 eigenfaces

As shown in figure 10, the first principal component gives a general shape of a human face. As defined in PCA, the principal components are ordered descendingly by how much variances each principal component captures, which is the reason that eigenface 1 looks like all human faces with only the elliptical shapes of face and eyes and showing the existence of nose and mouth. The columns after that capture more features such as lips, eyebrows, shadows, and muscles on the face.



Figure 11: The 51th eigenface

We can project new faces that were not used in the training dataset to the first **r** eigenfaces to store data with a lower volume of memory by × x, and use **r** eigenfaces as the basis for linear combinations to achieve an approximation of these new faces.

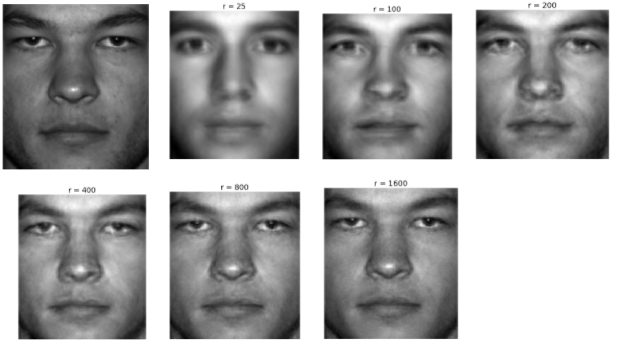


Figure 12: The 37th object original image (top left) and linear composition of first **r** eigenfaces

As we can see, when r=800, the approximated picture is already good enough to reproduce the face of subject 37, so that the computation cost is reduced by a large amount comparing to the 32,256-dimension vector space of the original images.

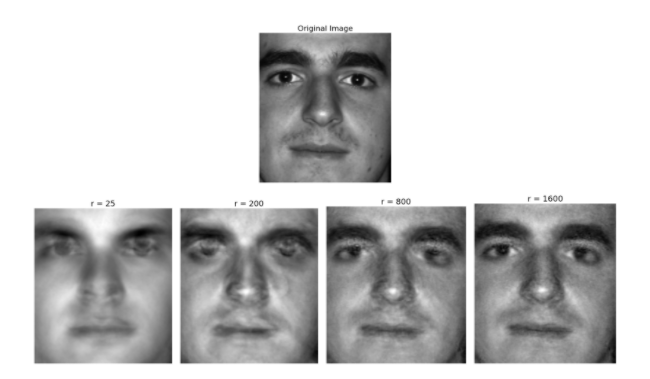


Figure 13: The 38th object original image (top) and linear composition of first **r** eigenfaces.

To differentiate people, for example, person 37 and 38 who were not in the training set, we can project them onto a plane given by two principal components because there are only two people to choose from.

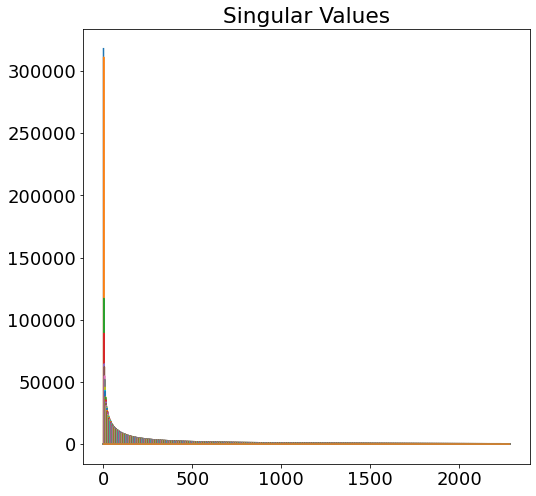


Figure 14: Singular values of original face matrix

After plotting singular values, figure 14 tells us by the color, that the first four singular values are much larger than the rest of them, thus, the first four columns of matrix U would capture a lot more variations than the rest of the columns, resulting in the worse split between points of different classes (faces).

Hence, we can project the mean-subtracted 37th and 38th faces after stretching them to column vectors, to the plane constructed by the eigenfaces 5 and 6, i.e., column numbers 4 and 5 in U since numbers start from 0 in python.

The coordinates of points is calculated by:

[:, [4,5]] × Face\_37

Here is the resulting plot.

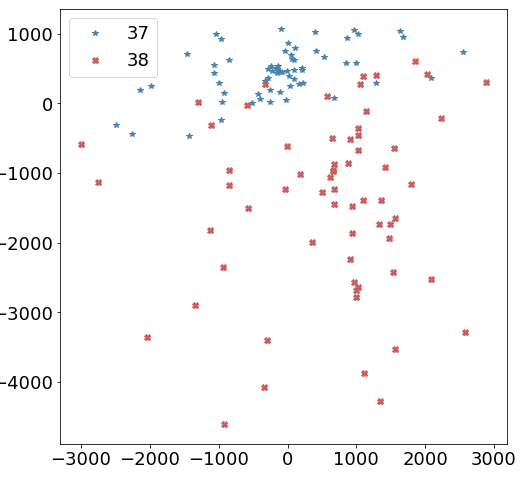


Figure 15: Projection of object 37 and 38 onto PC5-PC6 plane

After projection, Figure 15 tells us that most of the data of object 37 lie in the positive region of PC6, and object 38 has data that are more negative in PC6, while having a larger standard deviation in principal component 6. From here, we can train algorithms to recognize these faces with more principal components included.

### References:

Hu, C., Lu, X., Ye, M., & Zeng, W. (2017). Singular value decomposition and local near neighbors for face recognition under varying illumination. Pattern Recognition, 64, 60–83. https://doi.org/10.1016/j.patcog.2016.10.029

Steve Brunton. (2020, January 19). Singular Value Decomposition [Video]. YouTube. <https://www.youtube.com/playlist?list=PLMrJAkhIeNNSVjnsviglFoY2nXildDCcv>

Georghiades, A.S. and Belhumeur, P.N. and Kriegman, D.J. From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose. IEEE Trans. Pattern Anal. Mach. Intelligence 23(6):643-660 (2001).