# MATH4995 Final Report
## Topic 3: Eigenface

MAK, Chi Yuen Kevin

Fall 2022

# Abstract

This report will summarize the work we have done thus far for the capstone project. We first demonstrate our implementation of facial recognition and compare 2 different kinds of low-rank approximation, Non-negative matrix factorization (NMF) and Principal Component Analysis (PCA). We will then show that NMF given specific constraints can also be used for clustering and the milestones of the research by others on this topic.


# Introduction

Non-negative matrix factorization (NMF) and Principal Component Analysis (PCA) are both ways to conduct lower-rank matrix approximation. In NMF, given an $n \times m$ matrix $\mathbf{X}$ and a lower rank r, we wish to decompose it into

$$\mathbf{X} \approx \mathbf{WH}$$

where $\mathbf{W}$ is an $n \times r$ matrix and $\mathbf{H}$ is $r \times m$, with the constraint of all three matrices not having any negative entries. This is helpful, for the fact that it wouldn't make sense for data such as images and audio to have negative values. This property makes the result of the decomposition easier to interpret.

For PCA, we find a similar decomposition with a different constraint. The non-negativity constraint is relaxed, but the columns of $\mathbf{W}$ and the rows of $\mathbf{H}$ are constrained to be orthonormal to each other. In the first 2 sections of this report, we will look into the application of both NMF and PCA in facial-recognition.

On top of that, Ding et al.[1] made an observation that NMF in the form of

$$\mathbf{X} \approx \mathbf{HH}^T$$

is inherently equivalent to kernel K-means clustering. In the last section of the report, we will take a look at the details of this observation.

# 1 Facial Recognition with the PCA approach

The intuition behind this approach is that we look for the set of features that encode most of the variation between our face images [2]. To do that, We find the principal components (PC) of our image data, which are vectors for our data to project onto such that the variance is maximized. The PCs are simply the eigenvectors of the covariance matrix of our data.

To show this, suppose we have a vector $\mathbf{v}$ and dataset $\mathbf{M}$, $\mathbf{v}^T\mathbf{M}$ would then be the projection of our data onto the vector. The variance of the projection is

Figure 1: Mean Face of the subjects

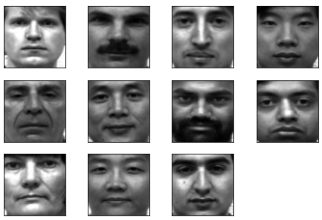


Figure 2: The 11 training faces

now simply

$$\mathbf{v}^T\mathbf{M}(\mathbf{v^T M})^T = \mathbf{v}^T\mathbf{M}\mathbf{M}^T\mathbf{v}$$

where $\mathbf{M}\mathbf{M}^T$ is just our covariance matrix. Note that $\mathbf{M}\mathbf{M}^T$ is symmetric, meaning that it is diagonalizable. Suppose $\mathbf{M}\mathbf{M}^T$ is diagonalized into $\mathbf{P}\mathbf{D}\mathbf{P}^T$, where the columns of $\mathbf{P}$ are the orthogonal eigenvectors of the covariance matrix and the trace of $\mathbf{D}$ are the eigenvalues, then the $\mathbf{v}$ that maximizes $\mathbf{v}^T\mathbf{P}\mathbf{D}\mathbf{P}^T\mathbf{v}$ is obviously the eigenvector with the biggest eigenvalue.

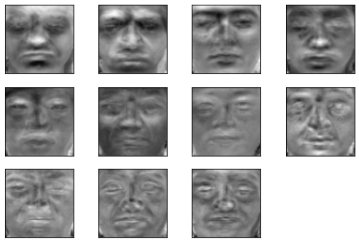


Figure 3: Eigenfaces ranked by eigenvalues in descending order

The matrix $\mathbf{M}$ is constructed in a way such that its columns are our flattened images. The dimension of each image is $250 \times 250$ with 11 training images, making $\mathbf{M}$ $62500 \times 11$. The covariance matrix is too expensive to be calculated directly. However, notice that for the equation $\mathbf{M}^T\mathbf{M}\mathbf{x} = \lambda\mathbf{x}$, where $\lambda$ denotes an eigenvalue and $\mathbf{x}$ denotes and eigenvector, if we left-multiply by $\mathbf{M}$, then we have $\mathbf{M}\mathbf{M}^T\mathbf{M}\mathbf{x} = \lambda\mathbf{M}\mathbf{x}$, meaning $\mathbf{M}\mathbf{M}^T$ and $\mathbf{M}^T\mathbf{M}$ have the same eigenvalues and the eigenvector of $\mathbf{M}\mathbf{M}^T$ is simply $\mathbf{M}\mathbf{x}$.

To reconstruct a face, we simply project the mean adjusted image onto each of our PCs to get a vector $\omega$ [2]. We can consider this as how much each PC contributes to the variation of our face from the mean face. We then multiply this $\omega$ onto the eigenface matrix $\mathbf{W}$ and then add the mean face. Namely $\mathbf{W}\omega + \mathbf{m}$, where $\mathbf{m}$ is the mean face. Recognizing a face is done similarly. We find the $\omega_s$ of our new image in the same manner, then find the closest $\omega_i$ that belongs to one of the face classes within our dataset. If the Euclidean distance between the 2 is smaller than a certain threshold $\epsilon_0$, we will say that this new image belongs to our face class $i$. There are many ways for this $\epsilon_0$ to be chosen. In this project, we pick:

$$\max_{i \neq j} \frac{1}{2}||\omega_i - \omega_j||_2^2$$

(a) Existing Face

(b) Reconstructed Face

Figure 4: Reconstruction by projecting existing face onto the eigenfaces

as our $\epsilon_0$, which is the max distance between any 2 of the weight vectors of our face classes. If the euclidean distance between the original image and the reconstructed is beyond a threshold $\epsilon_1$, we will also say that the new image is not a face [2]. The choice of this $\epsilon_1$ is commonly empirically chosen. With this knowledge, we began testing the PCA approach with the 21 images in figure 6. The result showed that PCA identified the tests with 85.7% accuracy with all non-face images identified as such correctly.

# 2 Facial Recognition with the NMF approach

Figure 5: The 21 test images

In this section we introduce the use of NMF for facial recognition. NMF is primarily used as a low-rank approximation method. In this application, we decompose our matrix of image data $\mathbf{X_{n\times m}}$ into $\mathbf{W_{n\times r}}$ and $\mathbf{H_{r\times m}}$. In other words, we wish to minimize the following objective function:

$$\min_{W,H>0} ||X - WH||_F^2$$

Note that the Hessian for the function is not always positive semi-definite, so the function is convex *w.r.t* $\mathbf{W}$ or $\mathbf{H}$, but not both. Hence, we can only approximate the two matrices numerically instead of finding them analytically.

Lee et al.[3] proposed a multiplicative update rule as follows:

$$\mathbf{H} \leftarrow \mathbf{H} \cdot \frac{\mathbf{W}^T\mathbf{X}}{\mathbf{W}^T\mathbf{W}\mathbf{H}}$$

$$\mathbf{W} \leftarrow \mathbf{W} \cdot \frac{\mathbf{X}\mathbf{H}^T}{\mathbf{W}\mathbf{H}\mathbf{H}^T}$$

To derive this algorithm, one can first compute the gradient of the objective function *w.r.t* $\mathbf{H}$ and $\mathbf{W}$ as

$$-2\mathbf{W}^T\mathbf{X} + 2\mathbf{W}^T\mathbf{W}\mathbf{H}$$

$$-2\mathbf{X}\mathbf{H}^T + 2\mathbf{W}\mathbf{H}\mathbf{H}^T$$

Setting the step-size for gradient descent as $\frac{\mathbf{H}}{\mathbf{W}^T\mathbf{W}\mathbf{H}}$ and $\frac{\mathbf{W}}{\mathbf{W}\mathbf{H}\mathbf{H}^T}$, which will eliminate subtraction to prevent negative values will complete the derivation. We will be using multiplicative update for another topic, so for this section we will use the non-negative alternating least-squares algorithm instead. The algorithm is given as follows[4]:

---

*1: initialize* $\mathbf{W}$ *and* $\mathbf{H}$ *randomly*
*2: While* $||\mathbf{X} - \mathbf{W}\mathbf{H}||_F^2 > \epsilon$ *do:*
*3:* $\quad \mathbf{H} \leftarrow \arg\min_{\mathbf{H}} ||\mathbf{X} - \mathbf{W}\mathbf{H}||$
*4:* $\quad$ *convert all negative entries of* $\mathbf{H}$ *to 0*
*5:* $\quad \mathbf{W} \leftarrow \arg\min_{\mathbf{W}} ||\mathbf{V}^T - (\mathbf{W}\mathbf{H})^T||$
*6:* $\quad$ *convert all negative entries of* $\mathbf{W}$ *to 0*

---

Since NMF does not allow subtractions, we can think of the matrix $\mathbf{W}$ as features to add together and the matrix $\mathbf{H}$ as the weight for each of these features to reconstruct a face.

Figure 6: Basis images of NMF

To reconstruct a face, we solve the least square problem:

$$\min ||\mathbf{W}\mathbf{x} - \mathbf{v}||_F^2$$

for $\mathbf{x}$ where $\mathbf{v}$ is our input image. The reconstruction is then simply $\mathbf{x}$ multiplied onto $\mathbf{W}$. Recognizing a face is done similarly to PCA, but here we find the face class with the closest weight vector using cosine similarity $sim(x, y) = \frac{x \cdot y}{||x|| \cdot ||y||}$. If the result between the input and its closest vector is bigger than a certain $\epsilon$ where $0 \leq \epsilon \leq 1$, we say this image belongs to that face class.

We again tested the NMF approach with the 21 images in figure 6. with an accuracy of 80.9% and all non-face images being identified as "not recognized". Overall, there is no significant difference between the accuracy of PCA and NMF. However, one can see that the basis images resemble selected features from individuals whereas those from PCA lacks any interpretability. Due to the non-negative constraint, NMF fits the intuition of fitting features together to form an image and can be interpreted better. For this reason, NMF is sometimes chosen over PCA when the original data is non-negative as demonstrated from this project.

# 3 Clustering with Symmetric NMF

In this section we look into a way to relate NMF to clustering. Suppose there is a way to decompose a data matrix $\mathbf{V}$ into $\mathbf{W}$ and $\mathbf{H}$ such that $\mathbf{V} \approx \mathbf{WH}$ with the columns of $\mathbf{W}$ being the centroids and the columns of $\mathbf{H}$ being $\mathbf{e_i}$ if the j-th column of $\mathbf{H}$ belongs to class i, the NMF would in theory be equivalent to clustering. In practice, although NMF does find success in clustering in some ways such as topic modeling by Wang et al.[5] recently, the objective function of NMF is often different from that of clustering. Low-rank approximation aims to use basis vectors to approximate the data matrix whereas clustering aims to find a way to partition data such that similarity within clusters is high and low across clusters.

Ding et al.[1] made the observation that if we can perform NMF in a way such that $\mathbf{W} \approx \mathbf{HH^T}$ where $\mathbf{W}$ is the kernel matrix, then the objectives would coincide. To demonstrate this, recall the objective function of K-means clustering:

$$\sum_{k=1}^{K} \sum_{i \in C_k} ||\mathbf{x_i} - \mathbf{m_k}||^2 = \sum_{i=1}^{N} ||\mathbf{x_i}||^2 - \sum_{k=1}^{K} \frac{1}{\mathbf{n_k}} \sum_{i,j \in C_k} \mathbf{x_i^T x_j}$$

where $\mathbf{m_k}$ is the centroid of cluster $\mathbf{C_k}$ and $\mathbf{n_k}$ is the number of data points assigned to $\mathbf{C_k}$. We can then write the objective function of Kernel K-means as

$$\sum_{i=1}^{N} ||\phi(\mathbf{x_i})||^2 - \sum_{k=1}^{K} \frac{1}{\mathbf{n_k}} \sum_{i,j \in C_k} \phi(\mathbf{x_i})^\mathbf{T} \phi(\mathbf{x_j})$$

Ignoring the constant first term, note that the second term is simply the kernel function $K(x, y) = < \phi(\mathbf{x}), \phi(\mathbf{y}) >$. We can rewrite it as $Tr(\mathbf{H^T WH})$ where $\mathbf{W}$ is the kernel matrix and the k-th column of $\mathbf{H}$ is orthogonal and indicates the membership of each data point belonging to $\mathbf{C_k}$. Rewriting the objective function

$$\begin{aligned} H &= \arg \max_{H^T H = I, H \geq 0} Tr(H^T W H) \\ &= \arg \min_{H^T H = I, H \geq 0} -2Tr(H^T W H) \\ &= \arg \min_{H^T H = I, H \geq 0} ||W||^2 - 2Tr(H^T W H) + ||H^T H||^2 \\ &= \arg \min_{H^T H = I, H \geq 0} ||W - HH^T||^2 \end{aligned}$$

completes the proof that the two objective functions are equivalent. Ding et al.[1] then proposed the following multiplicative update rule without providing the derivation:

$$\mathbf{H_{ik}} \leftarrow \mathbf{H_{ik}}(1 - \beta + \beta \frac{(\mathbf{WH})_\mathbf{ik}}{(\mathbf{HH^T H})_\mathbf{ik}})$$

where $0 < \beta \leq 1$ is recommended to be chosen as $\frac{1}{2}$. We derive this algorithm by first computing the gradient of the objective function $||\mathbf{W} - \mathbf{HH^T}||^2$ to be $-\mathbf{4WH} + \mathbf{4HH^TH}$. Setting the step size for gradient descent as

$$\beta \frac{(\mathbf{WH})_{\mathbf{ik}}}{4(\mathbf{HH^TH})_{\mathbf{ik}}}$$

completes the derivation. Note that the updating rule does not constrain $\mathbf{H}$ to be orthogonal. However, the matrix will preserve a near-orthogonal property[1]. To show this, first note $\min_{\mathbf{H^TH=I}, \mathbf{H} \geq 0} ||\mathbf{W}||^2 - 2Tr(\mathbf{H^TWH}) + ||\mathbf{H^TH}||^2$ can be separated as

$$\max_{\mathbf{H}>0} Tr(\mathbf{H^TWH}) \tag{1}$$

$$\min_{\mathbf{H}>0} ||\mathbf{H^TH}||^2 \tag{2}$$

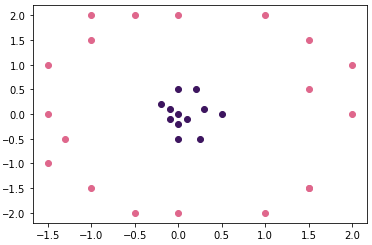
For (2), we can expand it into the form

$$||\mathbf{H^TH}||^2 = \sum_{lk} (\mathbf{H^TH})^2_{lk} = \sum_{l \neq k} (\mathbf{h}_l^T \mathbf{h}_k)^2 + \sum_k (\mathbf{h}_k^T \mathbf{h}_k)$$

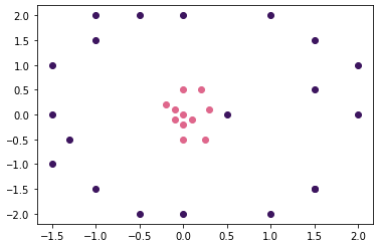
Minimizing the first term will give us near-orthogonality $\mathbf{h}_l^T \mathbf{h}_k \approx 0$. We can also see that for (1), $\sum_{ij} \mathbf{w}_{ij} \approx \sum_{ij} (\mathbf{HH^T})_{ij} = \sum_{kij} \mathbf{h}_{ik} \mathbf{h}_{jk} = \sum_k |\mathbf{h}_k|^2$, which is just the $L_1$-norm squared. Maximizing it will give us non-zero columns for $\mathbf{H}$. Combining (1) and (2), we arrive at the conclusion

$$\mathbf{h_l}^T \mathbf{h_k} \approx \begin{cases} 0 & \text{if } l \neq k \\ ||\mathbf{h_k}||^{\mathbf{2}} > \mathbf{0} & \text{if } l = k \end{cases}$$

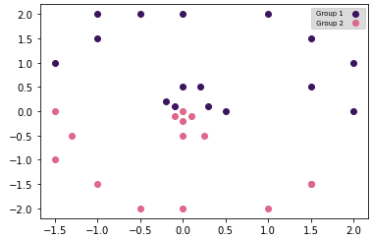
Which makes the columns of $\mathbf{H}$ nearly orthogonal. The cluster the i-th data point belongs to is simply the column with the biggest value in the i-th row of $\mathbf{H}$.



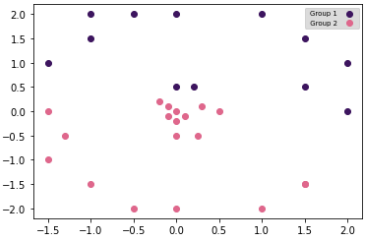
(a) Kernel K-means



(b) Symmetric NMF




(c) Symmetric NMF (Linear Kernel)




(d) K-means (Linear Kernel)

Figure 7: Clustering results from different methods

In this project, we applied both Kernel K-means and Symmetric NMF with different kernel matrices on random data, which showed no significant difference between the clustering quality of the 2 algorithms. It is noted that although Symmetric NMF often does converge to a perfect result, sometimes it would misclassify a small subset of data points, possibly due to the "soft clustering" quality of Symmetric NMF where strict orthogonality is not enforced.

# References

[1] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering"
[2] M Turk, A Pentland Eigenfaces for recognition. Journal of cognitive neuroscience, 1991.
[3] Daniel D. Lee and H. Sebastian Seung Learning the parts of objects by nonnegative matrix factorization. Nature volume 401, pages 788–791 (1999)
[4] Chen, D.,& Plemmons, R. J. Nonnegativity constraints in numerical analysis. The Birth of Numerical Analysis, 109–139. (2009)
[5] Wang, J., Zhang, X.-L. Deep NMF topic modeling. Neurocomputing, 515, 157–173. (2023)