ISYE 6740 – HW #2

Jing Ma

2025-01-29

Problem 1:

1.1

We are given the formula below that finds the principal component direction v:

$$v = \arg \max_{w:\|w\|\leq 1} \frac{1}{m} \sum_{i=1}^{m} \left(w^T x^i - w^T \mu\right)^2$$

We can rewrite the above equation as below:

$$\begin{aligned} v &= \arg \max_{w:\|w\|\leq 1} \frac{1}{m} \sum_{i=1}^{m} \left[w^T (x^i - \mu)\right]^2 \\ &= \arg \max_{w:\|w\|\leq 1} w^T \frac{1}{m} \sum_{i=1}^{m} (x^i - \mu)(x^i - \mu)^T w \end{aligned}$$

Let $C = \frac{1}{m} \sum_{i=1}^{m} (x^i - \mu)(x^i - \mu)^T$, then we have below:

$$v = w^T C w$$

Per the lecture slide ([1]), we can form Lagrangian function by introducing λ:

$$L(w, \lambda) = w^T C w - \lambda(1 - \|w\|^2)$$

Since this is an optimization problem of finding the weights to maximize variances, we can take partial derivative with respect to w ([1]):

$$\frac{\partial L}{\partial w} = 2Cw - 2\lambda w \Rightarrow Cw = \lambda w$$

We can observe that this is essentially eigendecomposition where λ are the eigenvalues and w represents the eigenvectors. We can further re-arrange the formula above ([1]):

$$w^T C w = \lambda w^T w \Rightarrow w^T C w = \lambda$$

This means that to solve this problem, we need to find the largest eigenvalue of C. Thus, we prove that v corresponds to the largest eigenvector of C.

1.2

As illustrated in part 1 above, the largest principal component directions correspond to the largest eigenvalues. This means, we need to find the second and third largest eigenvalues of the sample covariance matrix C.

1.3

Given that matrix A is a diagonal matrix, the values that fall on the diagonal are its eigenvalues $\lambda_1 = 3$ and $\lambda_2 = 2$. At the simplest form, we can find the corresponding unit eigenvectors $v_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $v_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Given the eigendecomposition equation $A = U \Lambda U^T$ and the eigenvalues and eigenvectors that we found above, we can construct the first eigendecomposition below:

First eigendecomposition

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix}$$

In this case, eigenvectors U is identity matrix.

Now if we scale v_1 by -1, we get the second eigendecomposition below:

Second eigendecomposition

$$A = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix}$$

From above, we prove that the eigendecomposition is not unique.

1.4

Per lecture 5 slides ([2]), we learned that the three key ideas in ISOMAP are:

1. The geodesic distances between two points can be approximated by finding the pairwise Euclidean distances in the neighborhood structure (locally treated as linear even in a nonlinear manifolds);
2. The distance matrix is constructed by computing the pairwise shortest path between all the points on a graph;
3. ISOMAP reduces nonlinear dimensionality by preserving the geodesic distances between the points in the lower dimension.

1.5

The following can be considered when choosing the number of principal components:

1. Data visualization needs. If the goal is being able to visualize data in 2D or 3D space, then we should choose k accordingly;
2. Set threshold. One common practice is setting 90% or 80% threshold and choose k based on the cumulative sum of explained variance that's close to the threshold ([3]);
3. Reduce collinearity. If multiple features are correlated, the effects of these variables can be combined, and in this case, only few principal components are needed as they likely explain most of the variance.

1.6

The impact of outlier(s) on PCA is demonstrated below using simple examples. Please see the eigenvalue computation in the accompanying code file for more details.

First we create a matrix without outliers.

$$M_1 = \begin{pmatrix} 7 & 7 \\ 4 & 8 \\ 8 & 5 \\ 5 & 4 \\ 7 & 8 \end{pmatrix}$$

We perform mean centering and compute the covariance matrix. Next, we obtain the eigenvalues of the covariance matrix, which are $\lambda_1 = 2.03121822$ and $\lambda_2 = 2.76878178$.

Similarly, we create another matrix with one outlier while all other values remain the same:

$$M_2 = \begin{pmatrix} 7 & 7 \\ 4 & 8 \\ 8 & 5 \\ 5 & 4 \\ 7 & 8 \\ 50 & 1 \end{pmatrix}$$

After performing the same steps as the first example, we compute the eigenvalues of the new covariance matrix. We found that now the eigenvalues become $\lambda_1 = 272.36294144$ and $\lambda_2 = 2.13705856$.

We see that after we introduced one outlier in the second matrix, λ_1 becomes significantly larger than the matrix without outlier. This is because PCA prioritizes the capture of the largest variance. When there is outlier, it increases the range of values along certain direction, and therefore inflates the variance. As a result, the total variance is dominated by the extreme values introduced by the outlier, and this increases the eigenvalue associated with the first principal component.

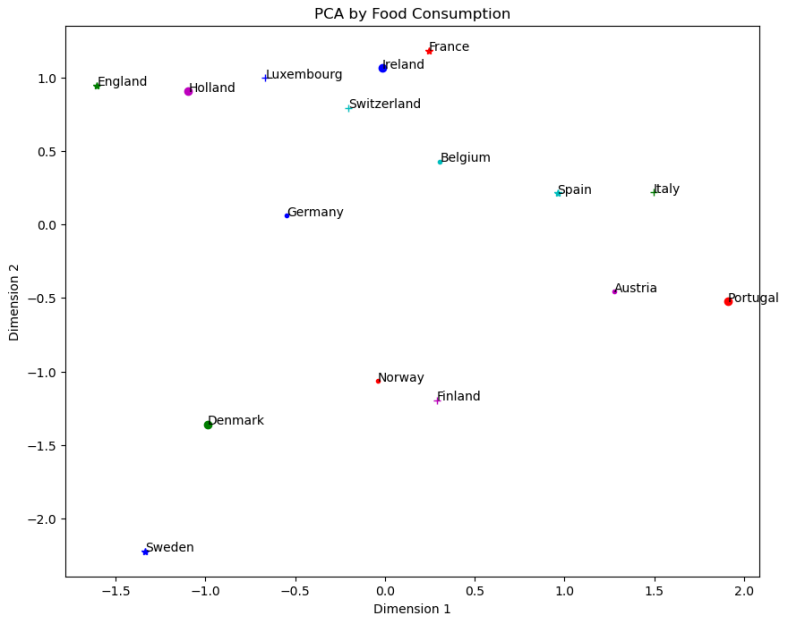
Problem 2:

2.1

We are given food-consumptions dataset, which contains 16 European countries and their consumption for 20 food items. In the first part of the problem, we are treating food consumptions as feature variables and each country is a sample or observation. The goal is to extract the top 2 principal components for all 16 samples. I adopted the provided PCA demo code and adjusted it accordingly for this problem. Below is the setup of the PCA algorithm implementation:

1. Normalize the data to ensure that values for all features are in the similar range;
2. Mean center the data;
3. Compute the covariance matrix;
4. Get the top 2 eigenvalues and associated eigenvectors;
5. Get the dimensions by computing the dot product between the data and eigenvectors and divide by the square root of the eigenvalues.

Below is the scatter plot of European countries based on the food consumption.



Based on the plot above, we observe roughly three clusters:

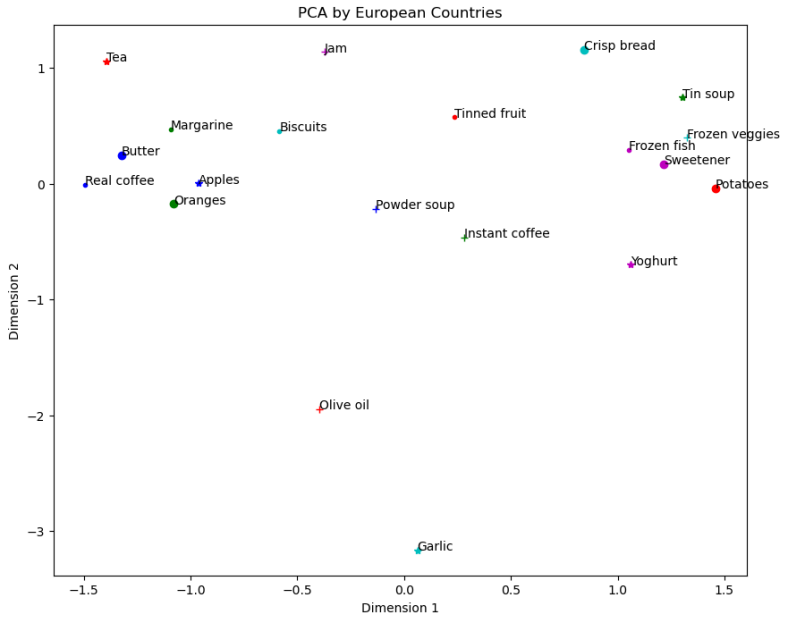
1. Closely positioned countries or situate around the center of Europe: England, Holland, Luxembourg, Ireland, France, Switzerland, Belgium, and Germany;
2. Surrounded around oceans, closely positioned or have similar diet: Italy, Spain, Austria, and Portugal;
3. Nordic countries: Sweden, Denmark, Norway, and Finland.

These clusters may not be completely accurate and are influenced by the diets from neighboring countries, where the country sits at, and their respective diet preferences.

2.2

We follow the same setup for PCA implementation except that for part 2, we treat food items consumptions as samples and European countries as features.

Below is a similar 2D scatter plot:



From the plot, we can observe the following clusters:

1. Fresh or moderately processed food: Tea, jam, margarine, biscuits, apples, oranges, real coffee, and butter;
2. Processed or instant food items: Crisp bread, tin soup, tinned fruit, powder soup, instant coffee, and yogurt;
3. Frozen or starchy food: Frozen veggies, frozen fish, sweetener, and potatoes;
4. Pantry: Olive oil and garlic.

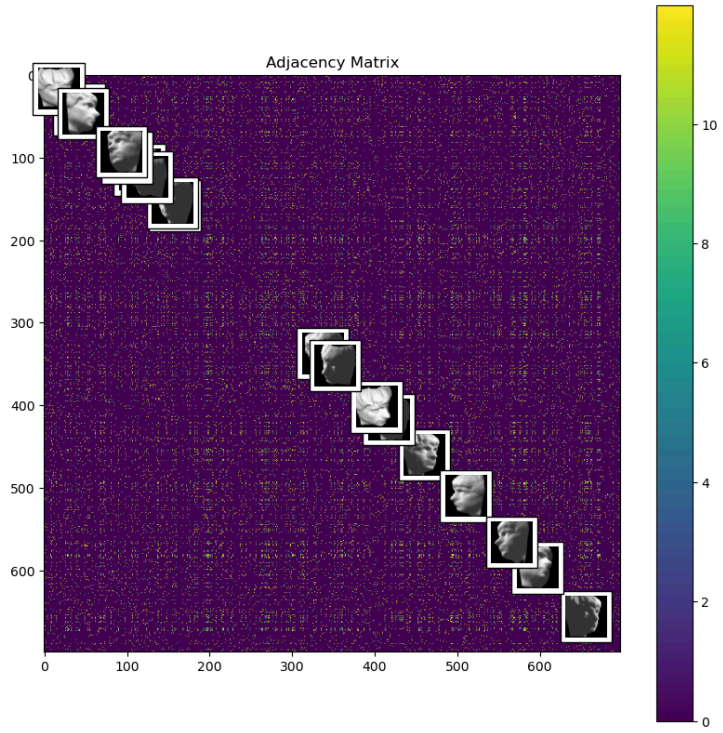
Problem 3:

3.1

We are given 698 images with 64 x 64 pixels. In this question, we need to construct the ε-ISOMAP. The first step is constructing the adjacency matrix based on the following:

$$A^{ij} = \begin{cases} \|x^i - x^j\|, & \text{if } \|x^i - x^j\| \leq \varepsilon \\ 0\ , & \text{otherwise} \end{cases}$$

Below is the heat map created using the adjacency matrix, including some randomly selected images. Note that we used $\varepsilon = 12$, which is considered to be the optimal kernel based on the analysis conducted in part 2.



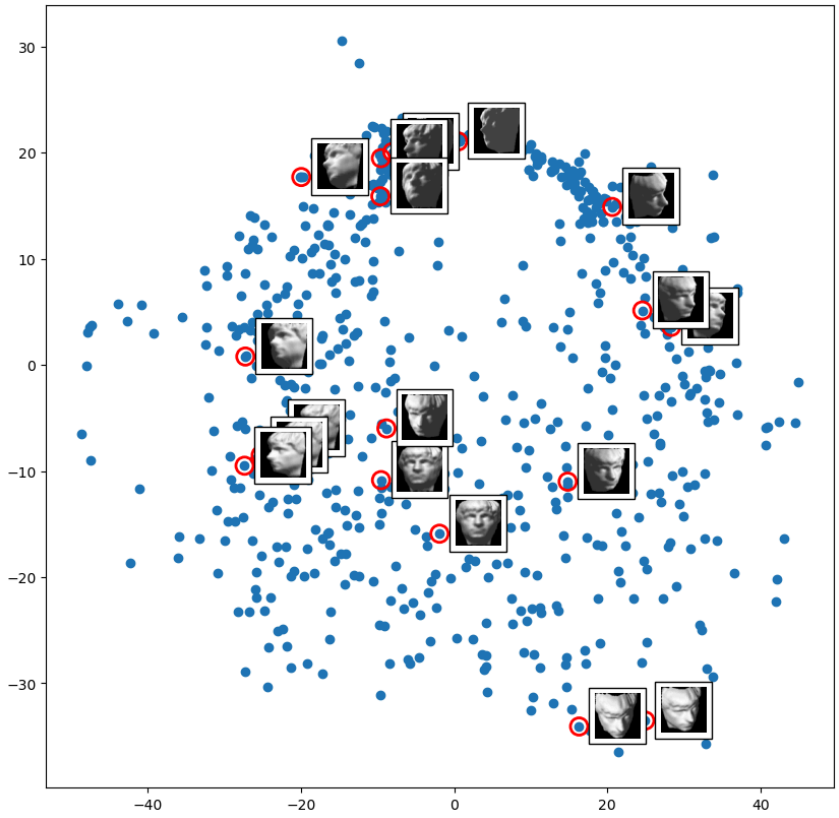
3.2

In part 2, we need to implement the ISOMAP algorithm and tune ε.

First, to implement ISOMAP, we carried out the following steps:

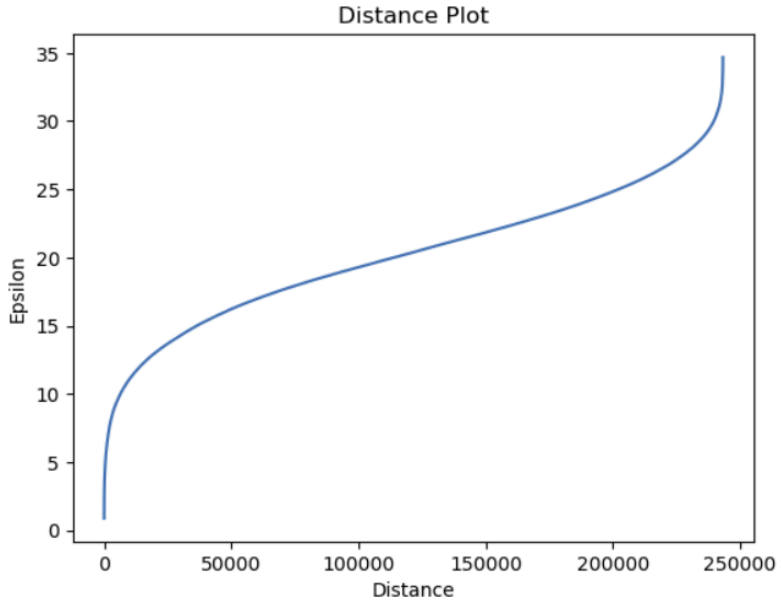
1. Create the undirected shorted paths matrix, D, using scipy shortest_path() function. Compute D^2;
2. Compute the centering matrix $H = I - \frac{1}{m}11^T$, and use it to get $C = -\frac{1}{2}HD^2H$;
3. Dot product the top two eigenvectors with square root of eigenvalues to get the dimensions.

Below is the plot of images in the 2D space using the dimensions.



As we can see, sculptures with faces oriented to the left or right are correctly placed on the respective sides of the scatter plot. The sculptures with faces oriented downward are placed at the bottom of the graph, whereas those facing upward situate at the top of the graph. In the middle are the faces looking straight ahead. So we can see a clear transition in the sculpture face orientation from the left to straight ahead and then to the right. Similarly, we can see the same trend as we trace the images from the top to the bottom, where the sculpture's face orientation changes accordingly.

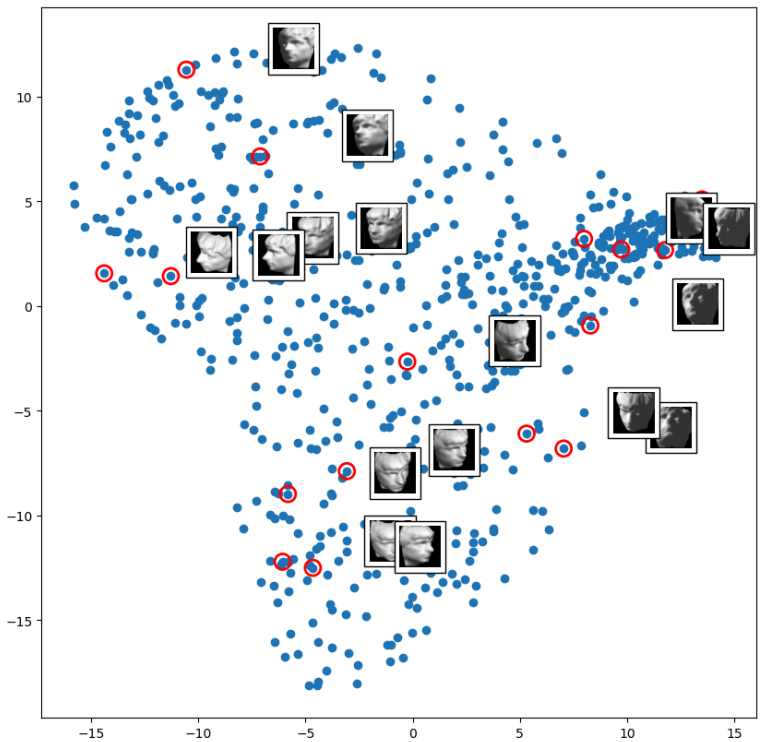
To tune ε, we focus on the connectivity of the graph. If $\varepsilon = 0$, the graph is fully disconnected. However, if ε is too large, the graph becomes fully connected. Therefore, we need to find a value that lies somewhere in between. We find all the pairwise distances and sort from smallest to largest. We then plot the sorted distances and find the point where the line transitions from steep to gradual as it indicates that as ε increases, the distance goes up very slowly, so data points are mostly connected and form small clusters. The plot below shows the general shape of the curve:



Based on this plot, we can see that the slope transitions from steep to more flat between the 10 - 15 range. Therefore, I judgmentally picked ε of 12 as the desired value.

3.3

In this question, we implement PCA on the same set of images and the scatter plot is shown below:

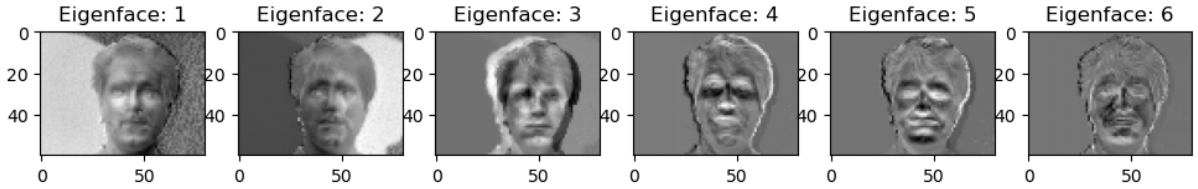


As we can see, PCA does not perform very well compared to ε-ISOMAP as some images on the right contain images facing toward the left. Also toward the right bottom corner, there's a sculpture facing upward but was not placed properly on the top portion of the graph.

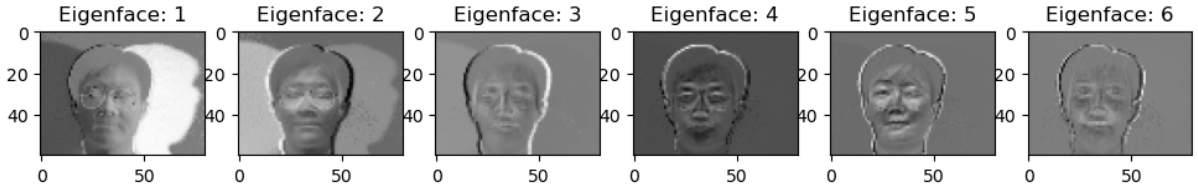
Problem 4:

4.1

In this eigenfaces problem, we have images pertaining to two subjects. We first vectorize each image and downsample by a factor of 4. Hence, each vector has 60 x 80 (4800) features. We then create a matrix for each subject by combining the respective images as rows. Lastly, we use PCA to extract the top six eigenvectors and reshape them into images. Below are the top six eigenfaces for subject01:



Similarly, the following are the top six eigenfaces for subject02:



As we can observe, the first eigenfaces for both subjects have high resemblance to the subjects' overall facial structure and lighting condition as the actual images. However, as we move from left to right, we start to see certain facial features such as eyes, nose, mouth etc. being highlighted in more details. Each eigenface seems to focus on different aspects of the same subject because eigenvectors extract unique uncorrelated features that best describe the subject.

4.2

Next we aim to compute the projection residual of the test image and eigenface vectors using the given formula:

$$s_{ij} = \left\| (\text{test image})_j - (\text{eigenface})_i (\text{eigenface})_i^T (\text{test image})_j \right\|_2^2$$

We perform the same pre-processing steps as part 1 for both test images. Further, we take the top eigenface and perform the comparison. Below are the results:

subject01 eigenface vs subject01 test image: s_{11} = 7,100,741.52196747

subject01 eigenface vs subject02 test image: s_{12} = 4,789,550.33819623

subject02 eigenface vs subject01 test image: s_{21} = 6,182,373.7198808

subject02 eigenface vs subject02 test image: s_{22} = 4,169,232.92382878

Interestingly, we note that even though for the same subject, there is a high residual between subject01′s eigenface and test image. However, for subject02, the residual is much lower between its eigenface versus test image. Also, we can observe that somehow the residual is much lower between subject01′s eigenface and subject02 test image. This could be a result of multiple causes, such as the lighting or shadow conditions, or the poor representation of subject01′s eigenface.

However, it's not entirely impossibly to distinguish the faces of the test images using the projection residual scores. This is because we can see that the residual between subject02′s eigenface and test

image is much lower compared to the score between subject01 eigenface and subject02 test image. So knowing that, we can easily tell the test image pertains to subject02 due to the better alignment with its eigenface.

4.3

Overall, it appears that the algorithm is not performing very well. The PCA algorithm aims to capture the largest variations, which can be influenced by lighting or shadow conditions and be mistaken as part of facial features. Further, it's possible that the accessories such as glasses could also be treated as feature variance, though may make very small contributions to the overall variations. Therefore, one main improvement would be designing PCA in a way so that it's more robust to external elements. Another way to reduce noises would be pre-process images to unify the local contrast, such as reducing shadow or lighting so these external elements would have much less influence on the algorithm results.

Problem 5:

In this question, we set out to explore whether mean centering a dataset has any effect on the eigenvectors of its covariance matrix. Let's first expand the covariance matrix:

$$\begin{aligned} C &= \frac{1}{m}\sum_{i=1}^{m}(x^i-\mu)(x^i-\mu)^T \\ &= \frac{1}{m}\sum_{i=1}^{m}\left[x^i(x^i)^T - x^i\mu^T - \mu(x^i)^T + \mu\mu^T\right] \end{aligned}$$

Let

$$\tilde{C} = \frac{1}{m}\sum_{i=1}^{m} x^i(x^i)^T$$

Further, we know that

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^i$$

Hence, we can simply C as following

$$C = \tilde{C} - \mu\mu^T - \mu\mu^T + \mu\mu^T = \tilde{C} - \mu\mu^T$$

If $\mu = 0$, then C is equivalent to $\tilde{C}$, which means that they both have the same eigenvectors. However, in most cases, $\mu \neq 0$, and hence the expressions below are not the same

$$Cw^1 = \lambda_1 w^1 <\neq> \tilde{C}\tilde{w}^1 = \tilde{\lambda}_1\tilde{w}^1$$

Therefore, we can deduct that $w^1 \neq \tilde{w}^1$.

References:

1. Topic 4 Dimensionality Reduction with PCA lecture slides.
2. Topic 5 Nonlinear Dimensionality Reduction with ISOMAP lecture slides.
3. https://towardsdatascience.com/pca-102-should-you-use-pca-how-many-components-to-use-how-to-interpret-them-da0c8e3b11f0