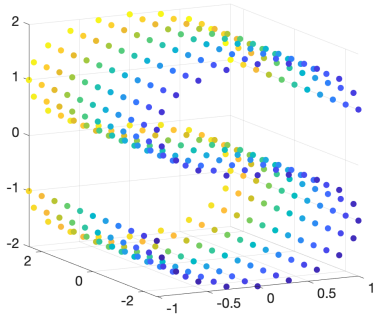
Homework 5

Yuancheng Xu

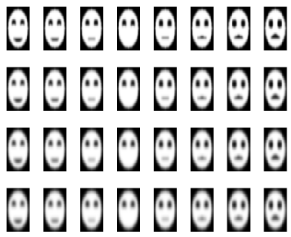
Nov 15, 2020

Exercise 2

In this problem I will try several dimension reduction (visualization) methods on two datasets: the curve data and the emoji data, visualized in Figure 1. Note that the intrinsic dimension of both datasets is two. For the curve data, we will add noise and see the robustness of each methods against noise. For the face data, there are two outliers and a good dimension reduction method should identify these outliers in visualization, and are also robust to them.



(a) curve data with no noise



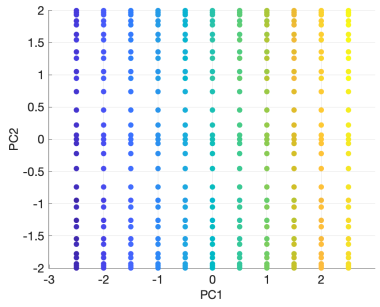
(b) Emoji (face) data

Figure 1: Original data

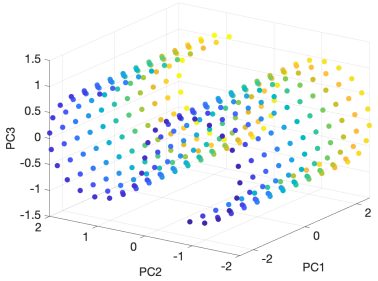
0.1 PCA

PCA finds linear transformations, one after another, to maximize the variations in the transformed data. I use the standard PCA tool in matlab to do experiments. PCA on the curve data is shown in Figure 2, where the noise level (standard deviation of Gaussion noise) is 0 and 0.2. We see that 3-dimensional PCA essentially recover the original data and 2-dimensional PCA does a good job seperating points.

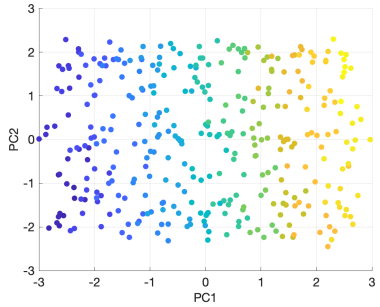
The performance of PCA on the face data is shown in Figure 3, which shows that PCA is robust to outliers and can identify them, since the visualizations are the same (except the outliers) with or without outliers. To sum up, PCA



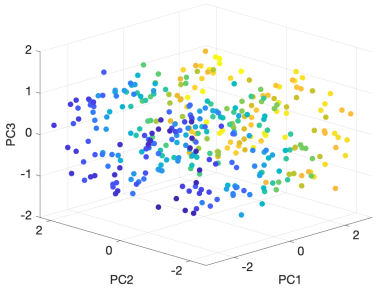
(a) noise = 0, dim = 2



(b) noise = 0, dim = 3



(c) noise = 0.2, dim = 2



(d) noise = 0.2, dim = 3

Figure 2: PCA on the curve data

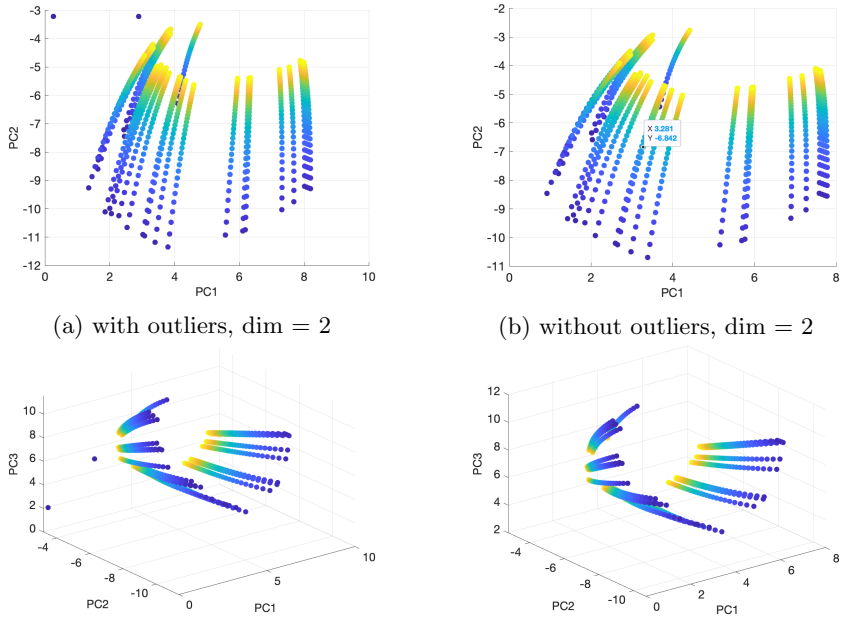
does a really good job, which is kind of surprising since it is really a simple method!

0.2 Isomap

Isomap approximately find the geodesic distance among points and use those distance to do multidimensional scaling (MDS). Since geodesic distance is really the "most accurate" distance, Isomap is expected to be good when there is no noise (will guaranteed to recover the intrinsic dimension under some conditions). However, since the shortest distance or the geodesic distance is sensitive to noise, Isomap should fail against noise. I use the code in the lecture notes. In the face data experiments, I use larger k to make sure that points are mutually reachable (if k is too small, there may be paris of points that are not reachable by neighbor points of both, and thus the matlab function "findshortestpath" will return inf).

Denote the number of neighbors as k. When k is large, the 'shortest path' is computed more accurately but a lot more expensive. Isomap is the slowest method in this report.

The visualization of Isomap is shown in Figure 4. When there is no noise, it perfectly unrolls the curve data. When noise increases, you need to increase k so that the computation of shortest distance is less sensitive to noise. On the face data, we see that Isomap is robust to the outliers and can identify them.



(a) with outliers, dim = 2

(b) without outliers, dim = 2

(c) with outliers, dim = 3

(d) without outliers, dim = 3

Figure 3: PCA on the face data

0.3 LLE

Locally Linear Embedding finds embedding such that the local embedding landscape (characterized by the weights that are used to represent a point using its neighbor points) is similar to the original space. I use Rowie's code and modify parts of it: I encounter singular matrix issues when using face data, so I regularize the matrix so that it is nonsigular.

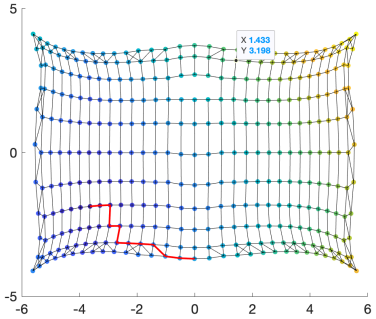
The results are visualized in Figure 5. I experiment with k. Interestingly, on the noiseless curve data, LLE fails only when k=10 (not 9, not 11), which I don't know how to explain. In general, larger k gives visualization with more regular geometry on both datasets. Also, we need larger k for the face dataset, partially because face dataset has more samples and they are in higher dimension.

0.4 t-SNE

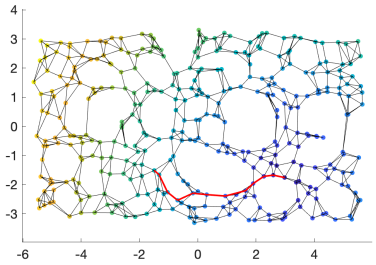
T-distributed stochastic neighbor embedding finds embedding that preserve the pair-wise distance in the original space, but with heavier tails (so that the embedding in lower dimension will not be too crowded). The results are given in Figure 6. On curve data, it seems that t-SNE is more robust to noise than previous methods on the curve data, where t-SNE unrolls the curve perfectly in 2-D. On the face data, t-SNE results in clusters whose relative placing is a little strange. However it is robust to outliers.

0.5 Diffusion Map

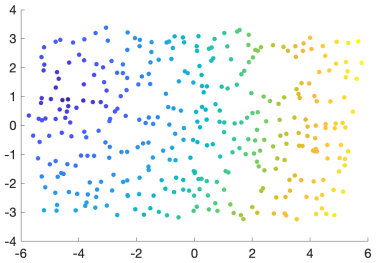
I implement Diffusion map (general form) in matlab and I experiment with different "fac" to find a good ϵ, where $\epsilon = \text{fac} * \text{mean of row min}$. Since Face data lies in higher dimension, "fac" is set to 1000. The results are given in Figure 7. On the noiseless curve data, almost all "fac" (from 2 to 1000) works. However, smaller "fac" results in more regular geometry in 3-D, while larger "fac" essentially recovers the original 3-D curve data. Also, Diffusion Map turns out to be most robust to noise on the curve data. It can visualize the data with noise = 0.5 in 2-D even if the original data looks very messy in 3-D. Moreover, when noise = 0.5 (observe Figure 7e and 7g), diffusion map makes the geometry of the original curve data more regular in the 3-D embedding (no dimension reduction). In this case, diffusion map has the effect of **denoising** and **regularization of geometry**, which is very interesting! Lastly, diffusion map works perfectly on the face data.



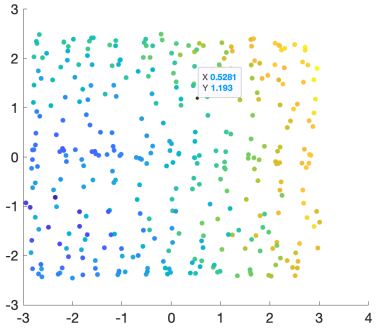
(a) Curve data, noise = 0, with edges plotted



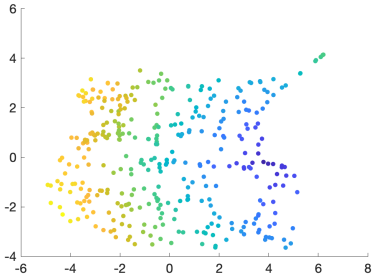
(b) Curve data, noise = 0.1, with edges plotted



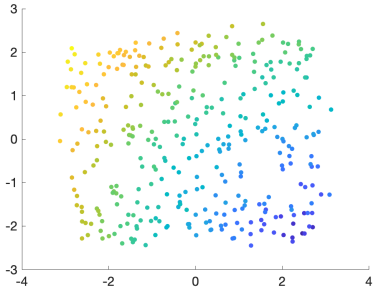
(c) Curve data, noise = 0.1, k=5



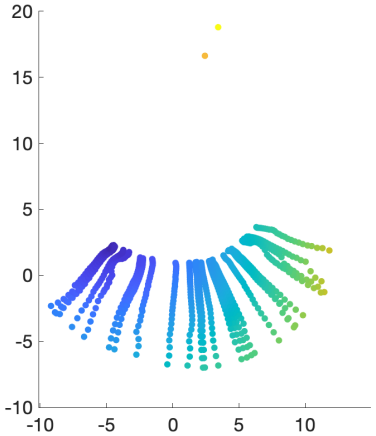
(d) Curve data, noise = 0.1, k=50



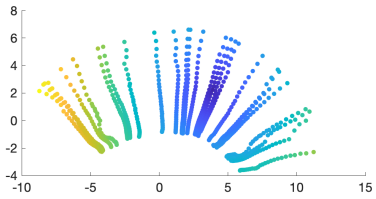
(e) Curve data, noise = 0.2, k=5



(f) Curve data, noise = 0.2, k=50

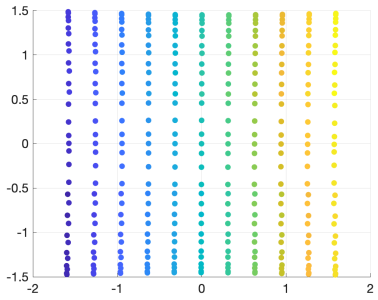


(g) Face data

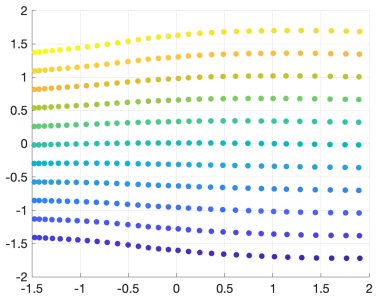


(h) Face data without outliers

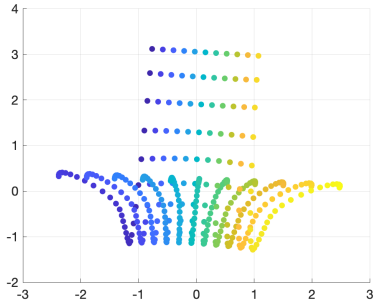
Figure 4: Isomap visualization on the curve and face data



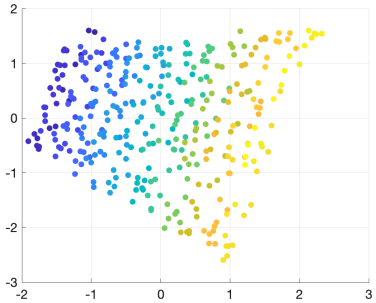
(a) Curve data, noise = 0, k=30



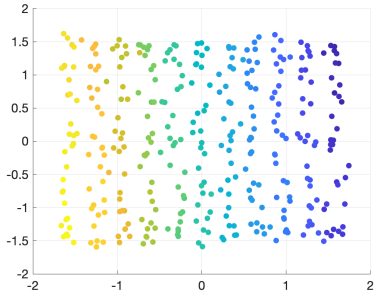
(b) Curve data, noise = 0, k=9



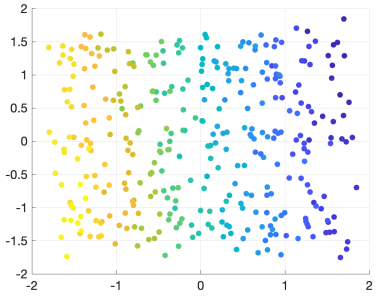
(c) Curve data, noise = 0, k=10



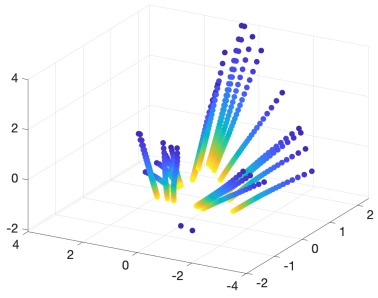
(d) Curve data, noise = 0.1, k=10



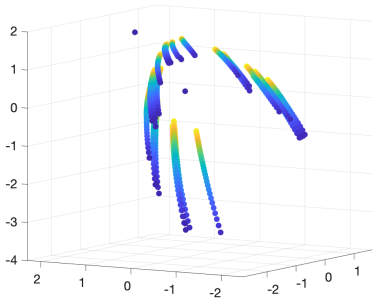
(e) Curve data, noise = 0.1, k=30



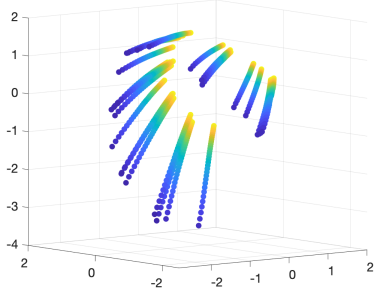
(f) Curve data, noise = 0.2, k=30



(g) Face data, k=40

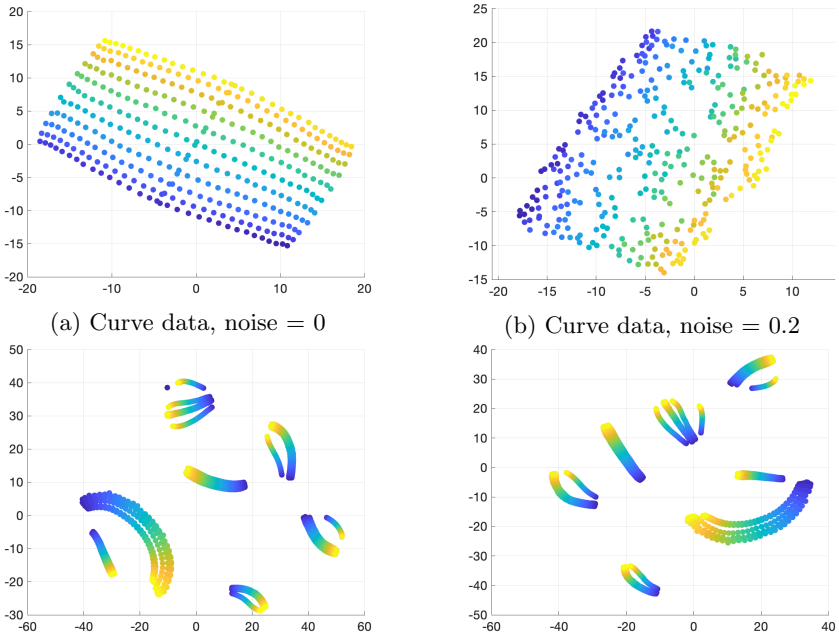


(h) Face data, k=500



(i) Face data without outliers, k=500

Figure 5: LLE visualization on the curve and face data



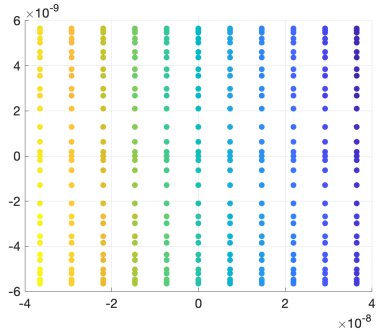
(a) Curve data, noise = 0

(b) Curve data, noise = 0.2

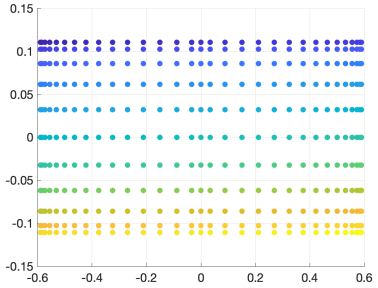
(c) Face data

(d) Face data without outliers

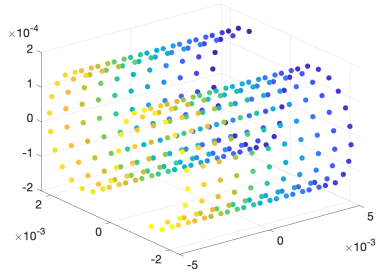
Figure 6: t-SNE visualization on the curve and face data



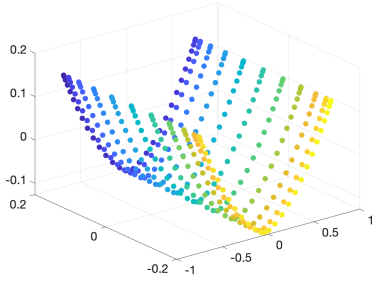
(a) Curve data, noise = 0, fac=1000



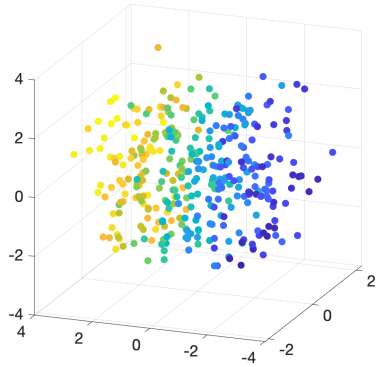
(b) Curve data, noise = 0, fac=2



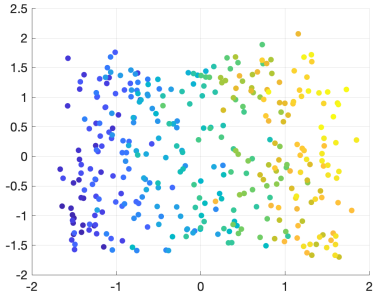
(c) Curve data, noise = 0, fac=1000



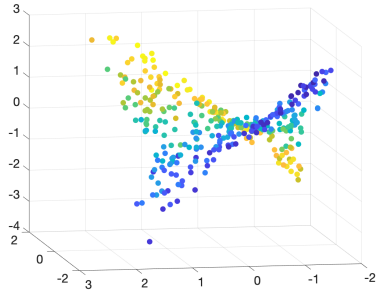
(d) Curve data, noise = 0, fac=2



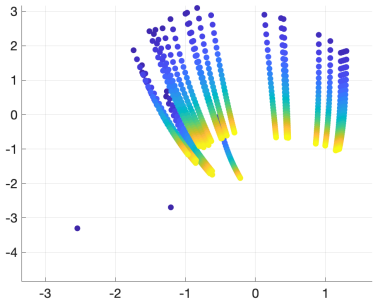
(e) Curve data itself with noise = 0.5



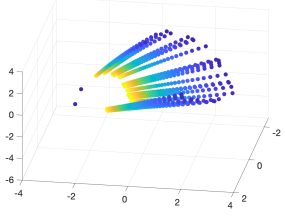
(f) Curve data, noise = 0.5, dim=2



(g) Curve data, noise = 0.5, dim=3



(h) Face data, dim=2



(i) Face data, dim=3

Figure 7: Diffusion Map visualization on the curve and face data