Iterative Non-linear Dimensionality Reduction by Manifold Sculpting

+

Unsupervised Learning Exam 17/09/2024

Candidate: Roberta Lamberti, SM360007

The problem: The curse of dimensionality

- In today's world, thanks to all the technology, we have access to lots of information we can gather and capture from data, that not always are biddable. What we mean is that it's not easy to interpret them when their dimensionality is so high to be impossible to work with them.
- One of the main problem Unsupervised Learning field tries to achieve is to reduce the dimensionality of these data, aiming to preserve (almost) the same charateristics of the original ones. Dimensionality reduction is, we can say, a pre-treatment to the data.

The problem

- There is a plethora of methods developed to solve this problem, but none of them is perfect. Each algorithm is able to solve a specific kind of problem is its own way.
- In this paper, a new algorithm, called Manifold Sculpting, is proposed to face the problem of non linear dimensionality reduction (it's suited for non linear surfaces, such as sphere or rolls)
- Its performance is compared to other 3 algorithms: Isomap, Local Linear Embedding and Hessian Local Linear Embedding.

Object of the tests

+For testing the perfomances of the test, we have used:

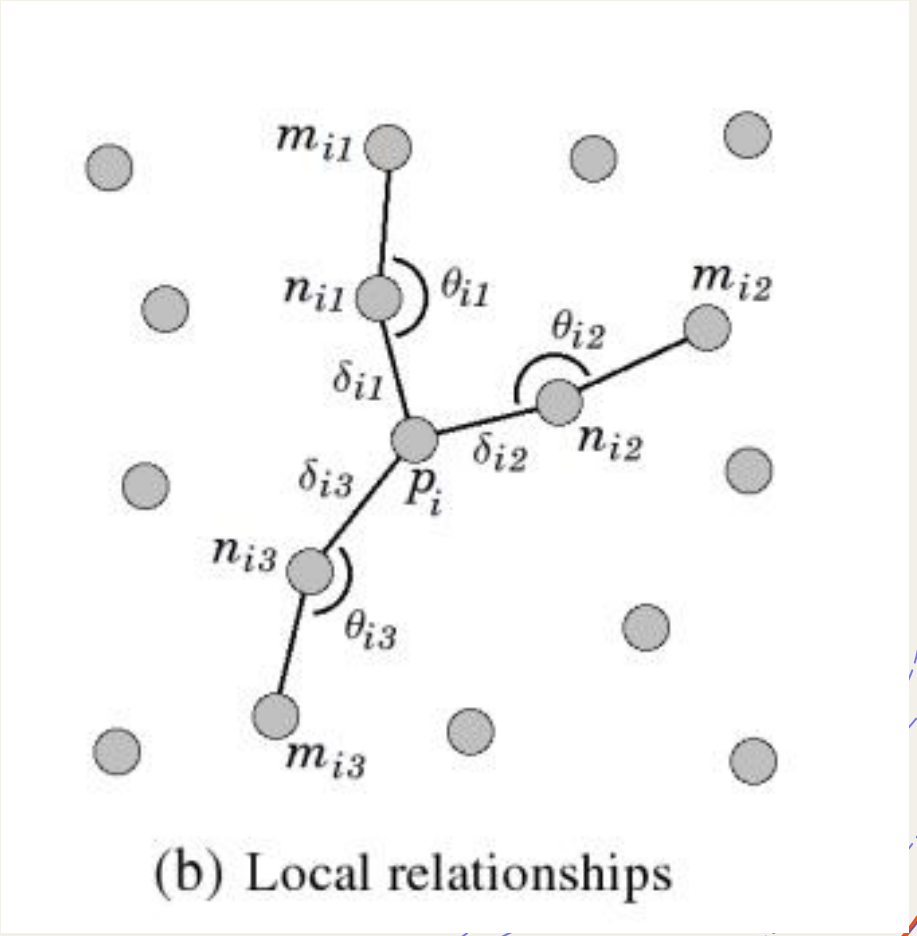
A Swiss Roll, with 2000 datapoints.

An S-Curve, with 20 neighbors.

The algorithm

In the algorithm there are 5 steps:

1. Find the k nearest neighbors of each point.
2. Compute a set of relationships between the neighbors.
3. Optionally align axes with PCA (this improves the performance).
4. While stopping criteria has not been met: a) scle the data in the non preserved dimensions; b) Adjust the points to restore the relationships
5. Project the data



(b) Local relationships

The algorithm: diving

- **STEP 1:** Our set of points is P, where all the points are represented as real vectors. We need to find the k-nearest neighbors N such that n_{ij} is the j^{th} neighbor of the point p_i
- **STEP 2:** Computing relationships; for each j we compute the euclidean distance between the point p_i and each neighbors n_{ij}. We also compute the angle β_{ij} (in the pic is theta) formed by two line segments (p_i to n_{ij} and n_{ij} to m_{ij}), where m_{ij} is the most colinear neighbors of n_{ij} with p_i. The most colinear is the neighbor that forms the angle closest to π. The values of α (it's delta) and β (it's theta) are the relationships the algorithm attempt to preserve during the transformation. β_{ave} is the global average distance between all of the neighbors of all points.

The algorithm: diving

+**STEP 3:** The data may be optionally preprocessed with the PCA. Preprocessing cn result in significantly faster convergence. In this way, PCA will move the information in the data into as few dimensions as possible, leaving less work to step 4. This step is performed by computing the first preserved distance, and rotating the dimensional axes to align with the principal components.

The algorithm: diving

- **STEP 4:** The data is iteratively transformed until some stopping criterion has been met. One effective technique is to stop when the sum change of all points during the current iteration falls below a threshold.
- a) SCALE VALUES. All the values in D_{scal} (the set of dimensions what will be eliminated by the projection) are scaled by a constant factor sigma, a value between 0 and 1. Overtime, D_{scal} will converge to 0.
- b) RESTORE ORIGINAL VALUES. For each point pi, the values in D_{pres} are adjusted to recover the relastionships distorted by the scaling. Intuitevely this step simulates tension on the manifold surface. An heuristic error value is used to evaluate the current relationships among data points relative to the original ones:

$$\epsilon_{p_i} = \sum_{j=0}^{k} w_{ij} \left(\left(\frac{\delta_{ij} - \delta_{ij_0}}{2\delta_{ave}} \right)^2 + \left(\frac{\theta_{ij} - \theta_{ij_0}}{\pi} \right)^2 \right)$$

- $Delta_{ij}$ is the current distance to n_{ij}
- $Delta_{ij0}$ is the original distance to n_{ij} measured at step 2
- $Theta_{ij}$ is the current angle
- $Theta_{ij0}$ is the original angle measured at step 2
- The denominators are chosen as normalizing factors, because the value of angle term can range between 0 and π, and the value of the distance will have a mean of about the $delta_{ave}$.

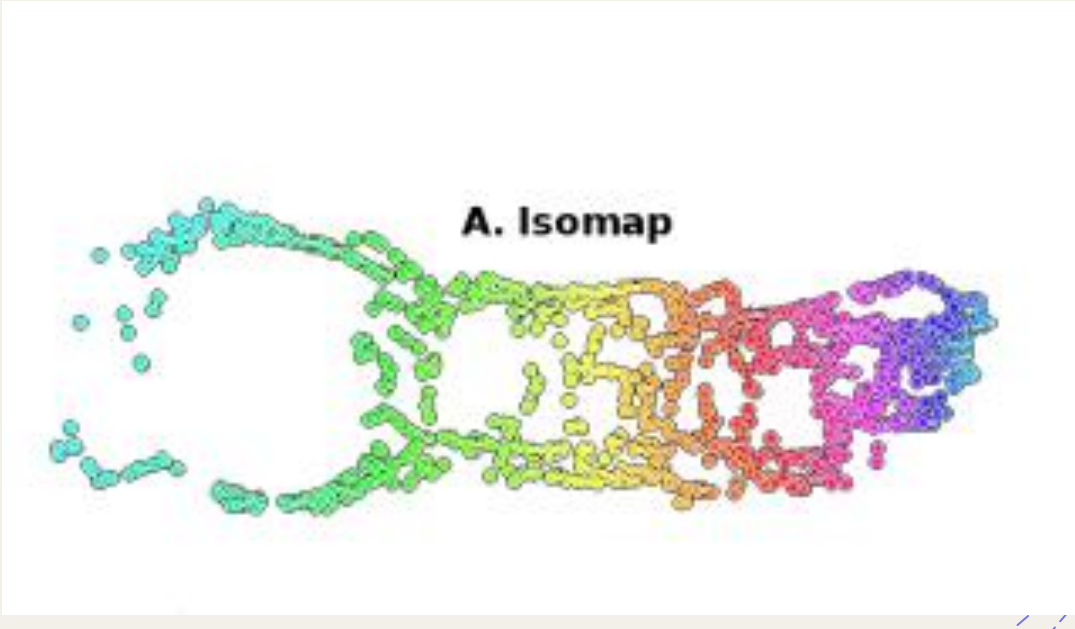
The algorithm: diving

- STEP 5: We project the data. D_{scal} containes now only values that are close to zero. The data is projected by simply dropping the redundant dimensions. This reduces the overall dimensionality of the data, letting us focus on the most important features that are able to retain most of the information.

- The algorithm gradually minimizes the values in D_{scal} until they are close to zero, at which point the dimension can be discarded without significant loss of information.

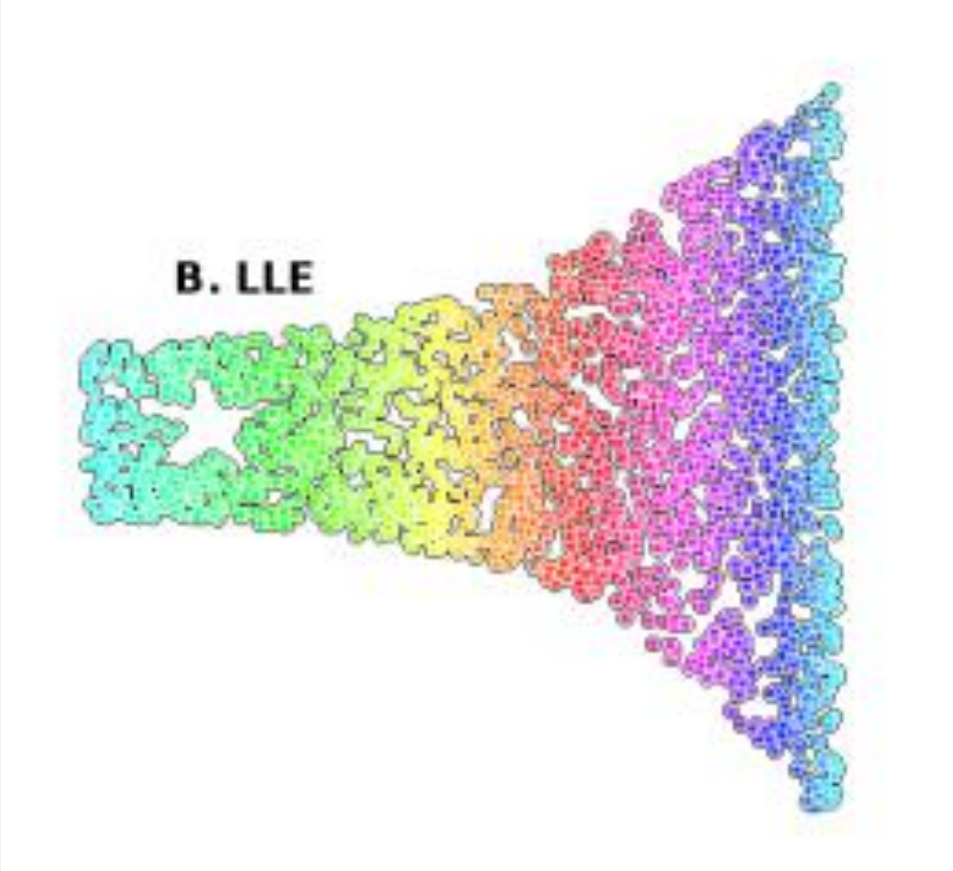
What makes Manifold Sculpting special?

- Manifold Sculpting results to be robust compared to Isomap, LLE and HLLE. But the real question is why?
- ISOMAP: it's computationally expensive, as it requires to solve eigenvectors for a large dense matrix. It's not easy to perform it with poorly sampled areas of the manifold;



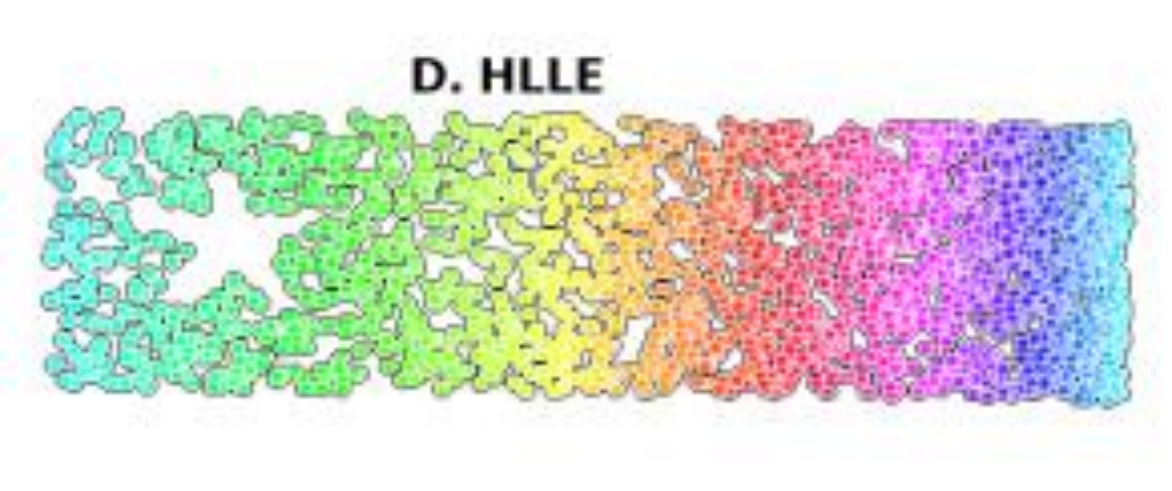
What makes Manifold Sculpting special?

- LLE is able to compute a similar computation but with a sparse matrix by using a metric that measures only relationships between vectors in local neighboord. It produces distored results when the sample density is non uniform.



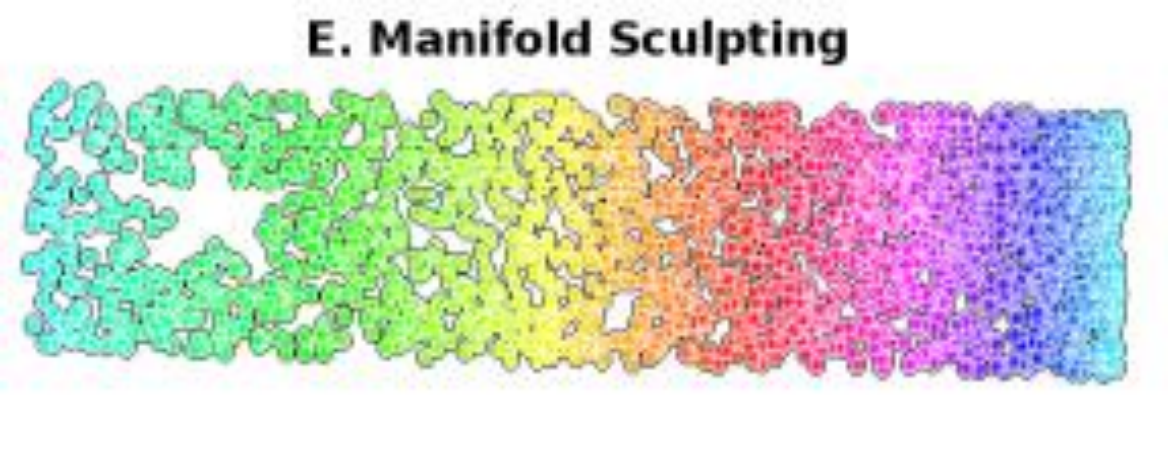
What makes Manifold Sculpting special?

- HLLE preserves the manifold structure better then other algorithms, but is computationally expensive. So it would become proibitive to compute.



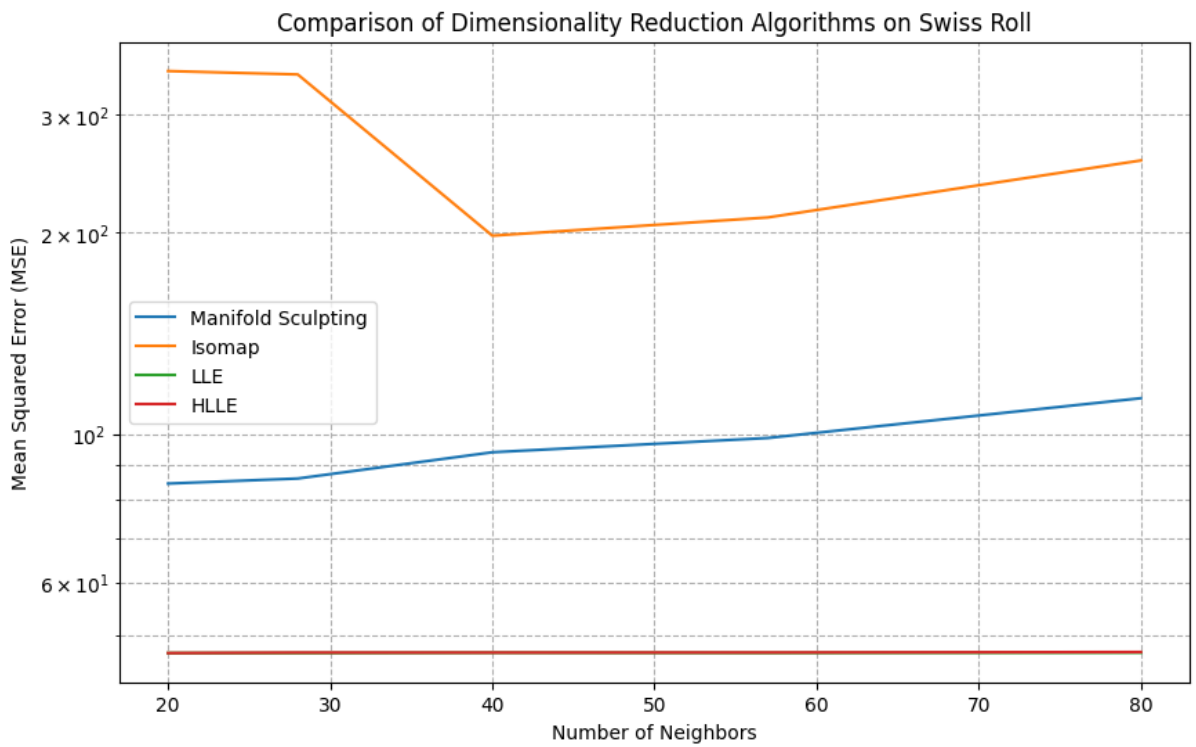
What makes Manifold Sculpting special?

- So finally, we come to the perks of Manifold Sculpting. Similarly to the HLLE, is able to cover almost the entire space, translating from the original space to the projected one.

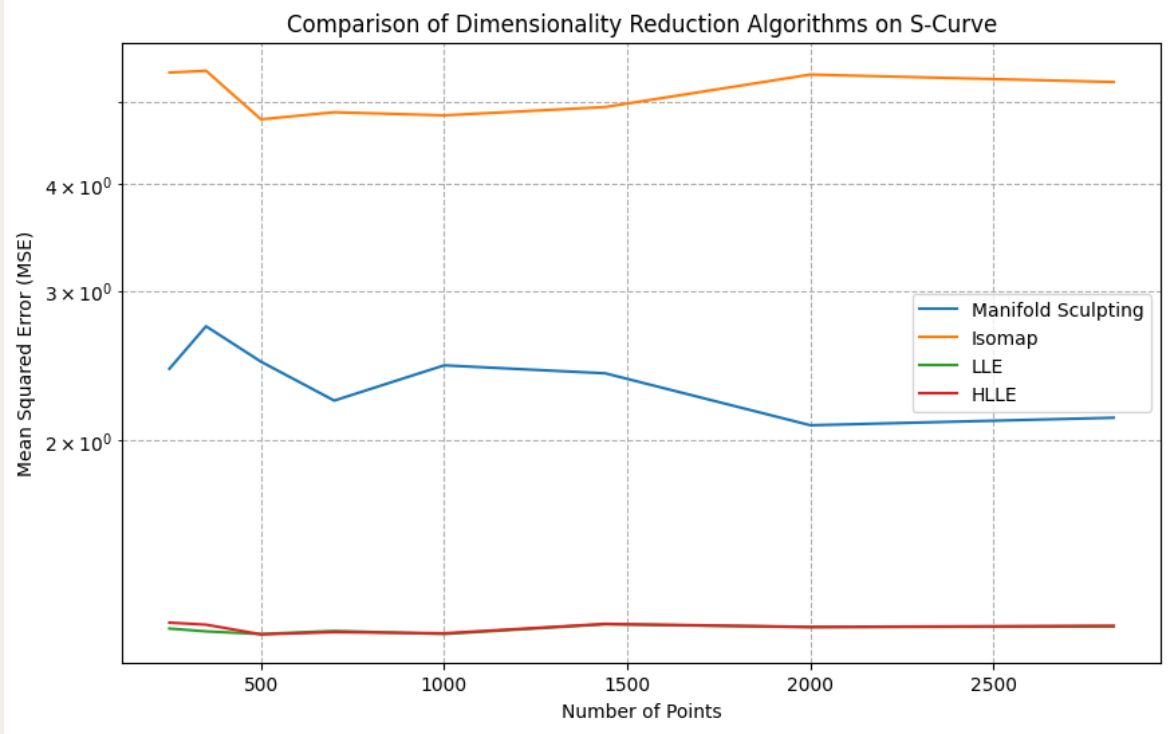


- The tests show that Manifold Sculpting yealds better results. Unfortunately

The Results



Number of Neighbors	Manifold Sculpting	Isomap	LLE	HLLE
20	84.358970	348.961997	47.071218	47.076838
28	85.840110	344.896641	47.074060	47.152109
40	93.956424	197.997064	47.126058	47.148122
57	98.689740	210.846812	47.096171	47.163498
80	113.167623	256.527247	47.136508	47.240216



Number of Neighbors	Manifold Sculpting	Isomap	LLE	HLLE
250	2.429537	5.405859	1.204229	1.223667
350	2.724696	5.433200	1.195098	1.216965
500	2.476168	4.764960	1.186569	1.185633
700	2.229242	4.856955	1.196554	1.192734
1000	2.451796	4.816208	1.186654	1.188986
1440	2.399478	4.925771	1.218377	1.219538
2000	2.085442	5.377373	1.208901	1.209481
2828	2.128202	5.270438	1.211112	1.213349

Conclusions

- Although not the same results of the paper have been reached overall, it's still clear that Manifold Sculpting perfoms better then Isomap.
  At the increasing number of neighbors, for 2500 samples performed in both the tests, Isomap MSE dramatically increases when the neighbors increase.
- HLLE seems to perform better then Manifold Sculpting, but it's computationally expensive.
- Overall, Manifold Sculpting performs well at the increasing number of neighbors, keeping the MSE steadily stable.