

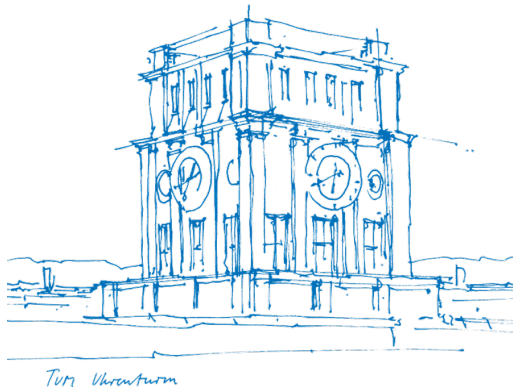
Manifold Learning

Comparison of Dimensionality Reduction Methods appropriate for non-linear data

Jazmin Inés Teng

Seminar Computational Aspects of Machine Learning
Chair of Scientific Computing in Computer Science (SCCS)
TUM School of Computation, Information and Technology
Technical University of Munich

February 1st, 2023



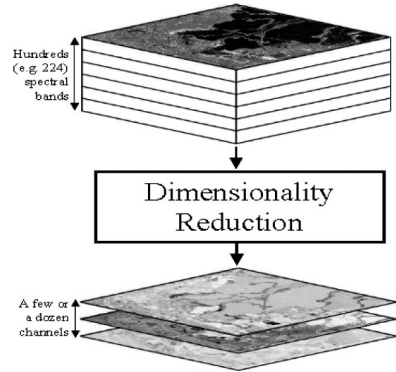
- 1 Motivation
- 2 Linear methods
- 3 Non-linear Methods - Manifold Learning
- 4 Comparison
- 5 Conclusion

Motivation

- Work with the full dimensionality of the data is computationally expensive and will slow down the training process
- Usually, some feature in complex dataset are correlated.
- High dimensional data is difficult to visualize.

Solution

Apply dimensionality reduction techniques



Source: Google.

Motivation

Dimensionality Reduction

The process of reducing the number of features to transform a dataset from a high-dimensional space into a low-dimensional space, conserving as much as possible the variation of the data.

Linear Methods

- PCA
- cMDS

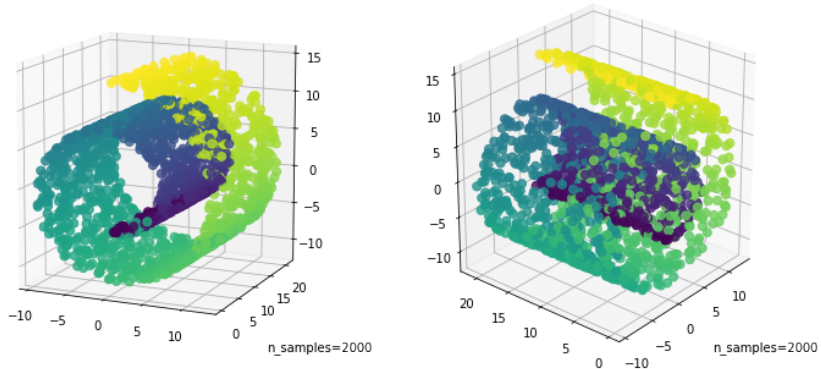
Manifold Learning (Non-linear)

- ISOMAP
- LLE
- Diffusion Map

Motivation

Swiss Roll Dataset

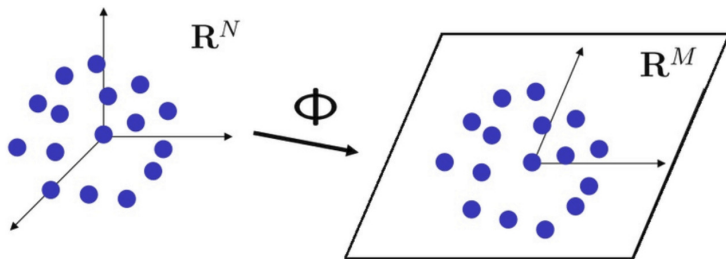
Swiss Roll data set in 3D space



- 1 Motivation
- 2 Linear methods
 - PCA
 - cMDS
- 3 Non-linear Methods - Manifold Learning
- 4 Comparison
- 5 Conclusion

Linear methods

The linear dimensionality reduction methods aim to project the data from a high to a low dimensional space through linear transformations.

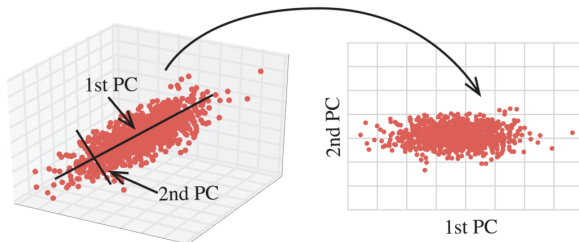


Source: Lotfi Fejri.

PCA - Principal Component Analysis

PCA finds a linear mapping between a high dimensional space D and a subspace d that captures most of the variability in the data. This subspace is specified by d orthogonal vectors called the principal components and it indicates the directions of maximum variance of the data.

- We can compute using eigenvalue decomposition (EVD) where the eigenvectors are the principal components.



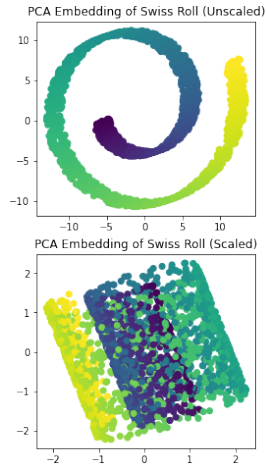
Source: Google.

Algorithm 1 PCA Algorithm

- 1: Standardize the D -dimensional dataset X .
 - 2: Construct the covariance matrix.
 - 3: Decompose the covariance matrix into its eigenvectors and eigenvalues.
 - 4: Select d eigenvectors which correspond to the d largest eigenvalues. (d is the dimension of the new feature subspace).
 - 5: Construct a projection matrix W from the top d eigenvectors.
 - 6: Transform the D -dimensional input dataset X using the projection matrix W to obtain the new d -dimensional feature subspace.
-

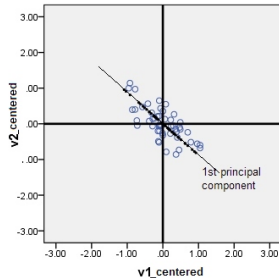
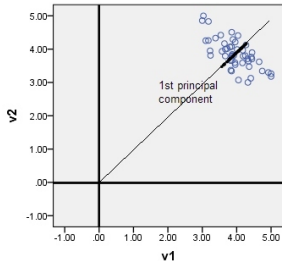
PCA embedding of Swiss Roll

- Fail to unroll!

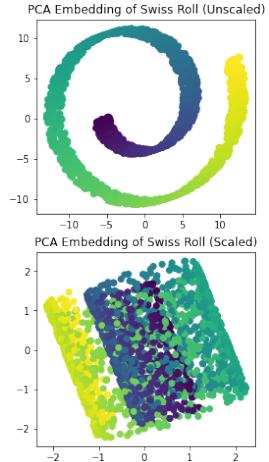


PCA embedding of Swiss Roll

- Fail to unroll!
- Standardization matters



Source: [1]



Torgerson MDS (classical MDS) is based on:

- Pairwise distance/dissimilarity matrix D .
- Euclidean distance between data points.
- Useful when instead of data points you have the distances or dissimilarities.

$$D^2 = [d_{ij}^2]$$

$$d_{ij}^2 = \|x_i - x_j\|^2$$

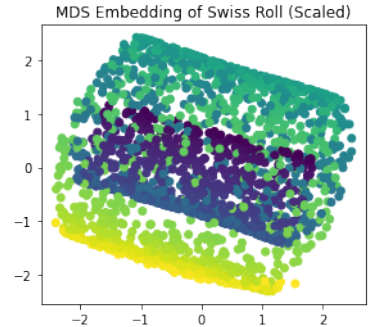
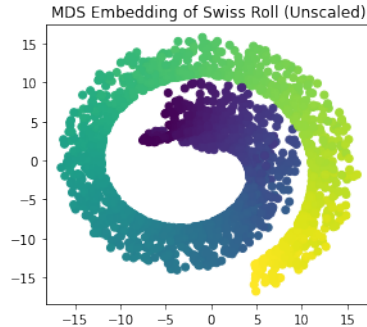
$$\begin{bmatrix} 0 & d_{12}^2 & d_{13}^2 & \dots & d_{1n}^2 \\ d_{21}^2 & 0 & d_{23}^2 & \dots & d_{2n}^2 \\ d_{31}^2 & d_{32}^2 & 0 & \dots & d_{3n}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n1}^2 & d_{n2}^2 & d_{n3}^2 & \dots & 0 \end{bmatrix}$$

Algorithm 2 classical MDS Algorithm

- 1: Set up the squared proximity matrix $D^2 = [d_{ij}^2]$
 - 2: Apply the double centering: $B = -\frac{1}{2}CD^2C$ using the centering matrix $C = I - \frac{1}{n}J_n$, where n is the number of objects, I is the $n \times n$ identity matrix and J_n is an $n \times n$ matrix of all ones.
 - 3: Extract the m largest positive eigenvalues $\lambda_1 \dots \lambda_m$ of B and the corresponding m eigenvectors $e_1 \dots e_m$.
 - 4: A m -dimensional spacial configuration of the n objects is derived from the coordinate matrix $X = E_m \Lambda_m^{1/2}$, where E_m is the matrix of m eigenvectors and Λ_m is the diagonal matrix of m eigenvalues of B , respectively.
-

cMDS embedding of Swiss Roll

- Fail to unroll!
- Preserve more variability than PCA



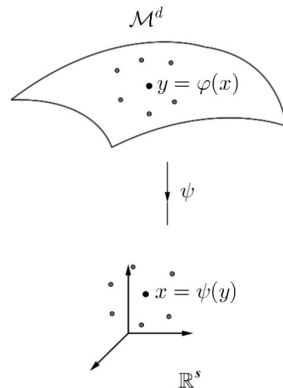
Outline

- 1 Motivation
- 2 Linear methods
- 3 Non-linear Methods - Manifold Learning**
 - ISOMAP
 - LLE
 - Diffusion Maps
- 4 Comparison
- 5 Conclusion

Assumption

All nonlinear data resides in a high-dimensional space but can actually be represented by a low-dimensional manifold.

- Manifold learning: find an embedding that describes non-linear datasets as low-dimensional manifolds.

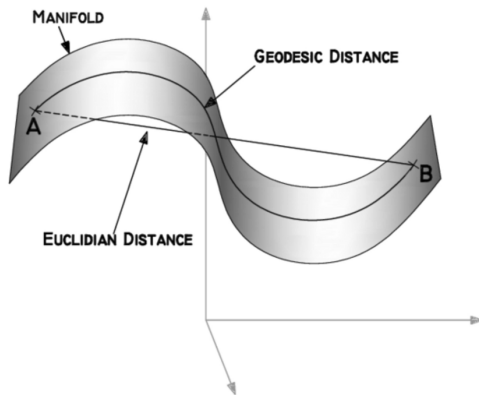


Source: K. Miranda, G. F.[2]

Geodesic distance

The geodesic between two points is defined as the shortest curve on the manifold connecting the two points. The length of the geodesic is the geodesic distance.

- Non linear data are reflected in geodesic distance.



Source: Karam, Zahi N and Campbell, William M.[3]

ISOMAP - Isometric Mapping

Isomap tries to preserve the geodesic distances in the lower dimension by creating a neighborhood network and uses its graph distance to approximate the geodesic distance between data points.

- The neighborhood graph $G(X)$ is constructed by connecting each data point with their neighbors, the weight of the edges are the Euclidean distance between them.
Using KNN or ϵ -neighborhood.
- Use the graph distance between two points to approximate the geodesic distances:

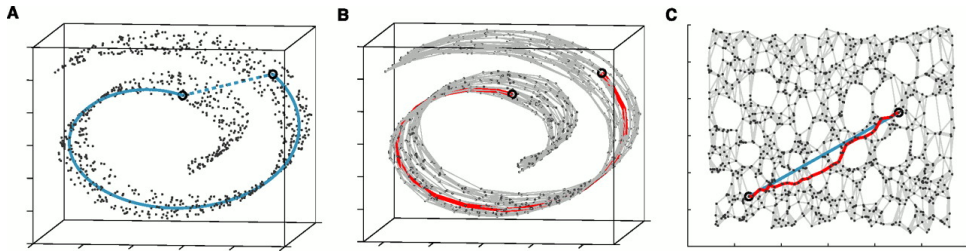
$$D_{ij}^{Geo} \sim \text{shortestPath}(x_i, x_j)$$

- Based on MDS but using geodesic distance matrix D

$$Y = cMDS(D)$$

Algorithm 3 ISOMAP Algorithm

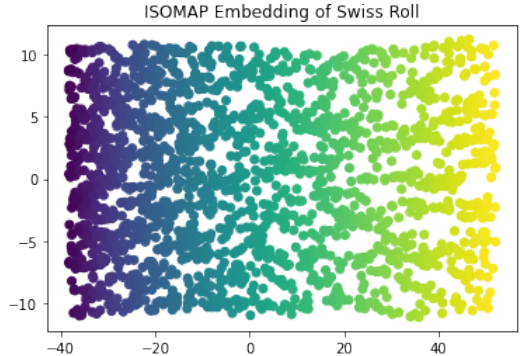
- 1: A neighborhood graph is constructed.
 - 2: ISOMAP proceeds with the estimation of geodesic distances.
 - 3: ISOMAP recovers y_i on R^d using the classical MDS on the geodesic distances.
-



Source: Tenenbaum, J. B., Silva, V. D., Langford, J. C. (2000).[4]

ISOMAP embedding of Swiss Roll

- Uniform embedding
- Preserves geodesic distance
- Can recover the global and local structure

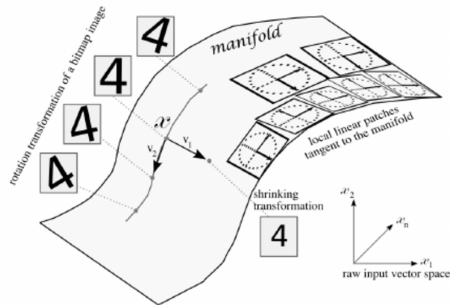


LLE - Locally Linear Embedding

Locally Linear Embedding finds an embedding that preserves the local geometry in the neighborhood of each data point.

Assumption: Locally Linear

Provided sufficient data, we expect each data point and its neighbors to lie on or close to a locally linear patch of the manifold and we can characterize the local geometry of these patches by linear coefficients that reconstruct each data point from its neighbors.



Source: Bengio, Yoshua.[5]

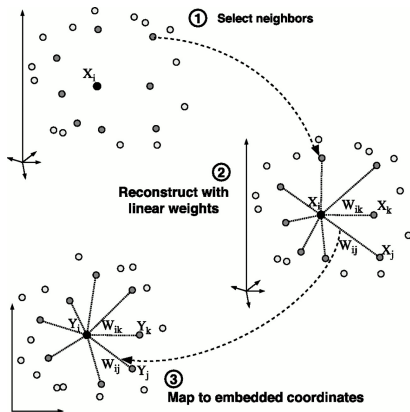
LLE - Locally Linear Embedding

- K nearest neighbors of each point are used to construct the weight matrix W that minimize the reconstruction cost function $\xi(W)$:

$$\xi(W) = \sum_i \left| \vec{X}_i - \sum_j W_{ij} \vec{X}_j \right|^2$$

- Then the data point is reconstructed in lower dimension using W and minimizing the embedding cost function $\Phi(Y)$.

$$\Phi(Y) = \sum_i \left| \vec{Y}_i - \sum_j W_{ij} \vec{Y}_j \right|^2$$



Source: Roweis, S.T., Saul, L.K.(2000).[6]

Algorithm 4 Locally Linear Embedding Algorithm

- 1: Compute the neighbors of each data point, \vec{X}_i
 - 2: Compute the weights W_{ij} that best reconstruct each data point \vec{X}_i from its neighbors, minimizing the reconstruction cost function $\xi(W)$ by constrained linear fits.
 - 3: Compute the vectors \vec{Y}_i best reconstructed by the weights W_{ij} , minimizing the embedding cost function $\Phi(Y)$
-

$$\text{minimize } \xi(W) = \sum_i \left| \vec{X}_i - \sum_j W_{ij} \vec{X}_j \right|^2$$

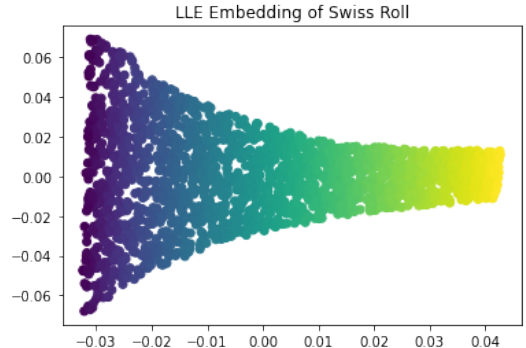
$$\text{subject to } \sum_j W_{ij} = 1$$

$$\text{minimize } \Phi(Y) = \sum_i \left| \vec{Y}_i - \sum_j W_{ij} \vec{Y}_j \right|^2$$

$$W = \begin{bmatrix} W_{11} & \dots & W_{1n} \\ \vdots & W_{ij} & \vdots \\ W_{n1} & \dots & W_{nn} \end{bmatrix}$$

LLE embedding of Swiss Roll

- Preserves local structure
- Some distortion in the global structure



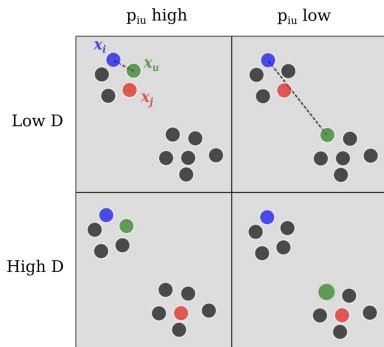
Diffusion Maps

The diffusion map embeds data in a lower dimensional space, so that the Euclidean distance between points approximates the diffusion distance in the original feature space.

- $D_t(x_i, x_j)$ is called diffusion distance after t time steps between points x_i and x_j . It is used to measure the similarity between those points.

$$D_t(x_i, x_j)^2 = \sum_u |p_t(x_i, x_u) - p_t(x_j, x_u)|^2$$

- If x_i and x_j are close, then $p_t(x_i, x_u) \sim p_t(x_j, x_u)$ and therefore $D_t(x_i, x_j) \rightarrow 0$.
- If they are far away, then $p_t(x_i, x_u) \neq p_t(x_j, x_u)$ and therefore the diffusion distance will be higher.



Source: Sebastian D. [7]

Diffusion Maps

- Calculation of diffusion distance is expensive.
- Efficient alternative: Use the proximity matrix P obtained after applying a kernel function $k(x_i, x_j)$ that measures connectivity between two points.

$$\text{connectivity}(x_i, x_j) = p(x_i, x_j) = \frac{1}{d_X} k(x_i, x_j)$$

- Given a value of t , we compute the decomposition of the Matrix P^t to obtain the eigenvectors and eigenvalues that form the matrix Y so that

$$D_t(x_i, x_j)^2 = \sum_u |P_{iu}^t - P_{ju}^t|^2 = \|Y_i - Y_j\|^2$$

$$K = \begin{bmatrix} K_{11} & \dots & K_{1n} \\ \vdots & \ddots & \vdots \\ K_{n1} & \dots & K_{nn} \end{bmatrix}$$

$$P = \begin{bmatrix} P_{11} & \dots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \dots & P_{nn} \end{bmatrix}$$

$$Y_i = \begin{bmatrix} P_{i1}^t \\ \vdots \\ P_{in}^t \end{bmatrix} = \begin{bmatrix} \lambda_1^t \psi_1(i) \\ \vdots \\ \lambda_n^t \psi_n(i) \end{bmatrix}$$

Algorithm 5 Diffusion Map Algorithm

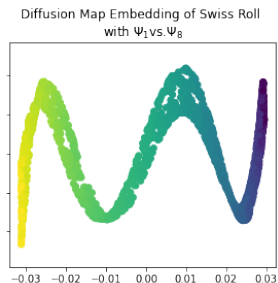
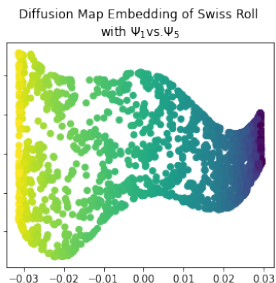
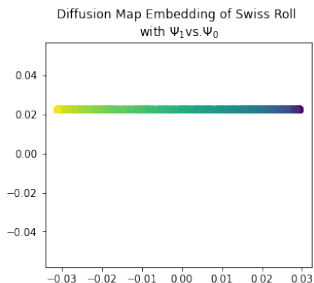
Input High dimensional data set $x_i, i = 0 \dots N - 1$

- 1: Define a kernel, $k(x, y)$ and create a kernel matrix, K , so that $K_{ij} = k(x_i, x_j)$.
- 2: Create the diffusion matrix by normalising the rows of the kernel matrix.
- 3: Calculate the eigenvectors of the diffusion matrix.
- 4: Map to the d -dimensional diffusion space at time t , using the d dominant eigenvectors and eigenvalues.

Output Lower dimensional data set $Y_i, i = 0 \dots N - 1$

Diffusion Maps embedding of Swiss Roll

- Using the eigenvector Ψ_1 and Ψ_0 we get an one-dimensional embedding
- Using the eigenvector Ψ_1 and Ψ_5 the resulting embedding shows the unrolled Swiss Roll
- Using the eigenvector Ψ_1 and Ψ_8 the resulting embedding recover the height of the data in the original dimension.



- 1 Motivation
- 2 Linear methods
- 3 Non-linear Methods - Manifold Learning
- 4 Comparison**
- 5 Conclusion

Comparison

■ Speed and Computational complexity

$$PCA > LLE, DiffusionMaps > cMDS > ISOMAP$$

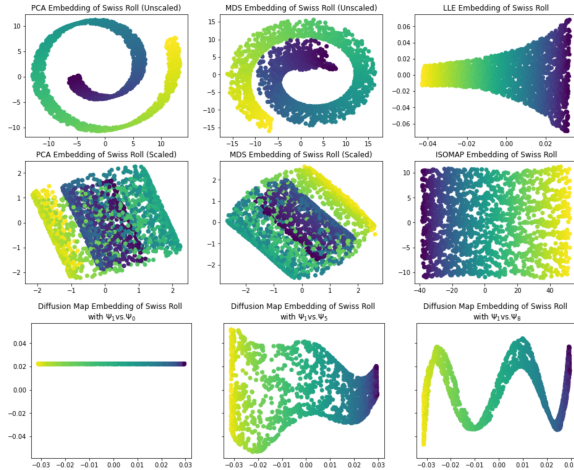
■ Sensible to outliers, holes and noises

$$PCA, cMDS > DiffusionMaps > ISOMAP > LLE$$

■ Sensible to parameters

$$PCA, cMDS > ISOMAP > LLE > DiffusionMaps$$

Comparison



- 1 Motivation
- 2 Linear methods
- 3 Non-linear Methods - Manifold Learning
- 4 Comparison
- 5 Conclusion**

Conclusion





1. Linear methods aren't suitable for non-linear data.
2. PCA is faster than cMDS but cMDS preserves better the variability of the data.
3. ISOMAP is the best preserving the global and local structure, but it is the slowest.
4. LLE is the fastest method but it can distort the global structure.
5. Both Isomap and LLE perform poorly on data with noises, outliers or holes.
6. Diffusion maps is the most robust method but is very sensible to the parameter.

Final Thoughts




Given a new dimensionality reduction problem, try the different methods on the dataset to find out which result it delivers, keeping in mind the properties of each method.

Thank you for the attention! Questions?

References I

-  ttnphns (<https://stats.stackexchange.com/users/3277/ttnphns>), “How does centering the data get rid of the intercept in regression and pca?” Cross Validated, uRL:<https://stats.stackexchange.com/q/22331> (version: 2016-09-16). [Online]. Available: <https://stats.stackexchange.com/q/22331>
-  G. F. Miranda, C. E. Thomaz, and G. A. Giraldi, “Geometric data analysis based on manifold learning with applications for image understanding,” in *2017 30th SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)*. IEEE, 2017, pp. 42–62.
-  Z. N. Karam and W. M. Campbell, “Graph embedding for speaker recognition,” *Graph embedding for pattern analysis*, pp. 229–260, 2013.
-  J. B. Tenenbaum, V. d. Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

References II

-  Y. Bengio, “Evolving culture versus local minima,” *Growing adaptive machines: Combining development and learning in artificial neural networks*, pp. 109–138, 2014.
-  S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
-  S. D., “Unwrapping the swiss roll with diffusion maps,”
<https://towardsdatascience.com/unwrapping-the-swiss-roll-9249301bd6b7>.