Université de Genève

master's project

Comparative study on UMAP and t-SNE

Author: Deniz Geçer

Supervisor: Stéphane Marchant-Maillet

E-mail: Deniz.Sungurtekin@etu.unige.ch

September 2022

UNIVERSITÉ DE GENÈVE
FACULTÉ DES SCIENCES
Département d'informatique

Contents

1 Abstract **2**

2 Introduction **2**

3 Related work **3**

4 Definitions and Notations **4**

5 Methods **6**
5.1 t-SNE . . . 6
5.1.1 SNE . . . 6
5.1.2 t-SNE . . . 7
5.2 UMAP . . . 8
5.2.1 Manifold And Fuzzy Topological . . . 8
5.2.2 Low Dimensional Representation . . . 11

6 Implementation **12**
6.1 t-SNE . . . 12
6.2 UMAP . . . 13

7 Evaluation **15**
7.1 Datasets . . . 15
7.2 Measures of quality . . . 16
7.2.1 Shepard Diagram . . . 16
7.2.2 Configuration score . . . 17
7.2.3 Stress . . . 17
7.2.4 Spearman's rho . . . 17
7.2.5 Konig's Measure . . . 17
7.2.6 Mean relative rank error . . . 18
7.3 Methodology . . . 18
7.3.1 Experiment 0 . . . 18
7.3.2 Experiment 1 . . . 18
7.3.3 Experiment 2 . . . 18
7.3.4 Experiment 3 . . . 19
7.4 Results and analysis . . . 19
7.4.1 Experiment 0 . . . 19
7.4.2 Experiment 1 . . . 21
7.4.3 Experiment 2 . . . 28
7.4.4 Experiment 3 . . . 31

8 Conclusion **37**

1 Abstract

Nowadays, there is a strong need to process a huge quantity of data to help machine learning models to extract patterns. It is hard to have a global understanding of the data when it has many dimensions. Thus it is required to reduce the dimensionality using different techniques. This pre-processing step is called dimensionality reduction and generally allows the removal of superfluous and redundant features in order to improve the accuracy and the training time of a model. It is heavily used in data science for the analysis and visualization of data in all types of sectors. In the past few years, enthusiasm for non-linear dimensionality reduction has grown, and these techniques have demonstrated better efficiency than traditional linear techniques on many real-world data according to specific evaluation metrics. In this paper, an implementation, a discussion, and a comparison between two non-linear techniques for dimensionality reduction are presented. The investigation of the performance of UMAP and t-SNE for dimensionality reduction is made by using and discussing diverse evaluation metrics.

2 Introduction

Dimensionality reduction is mapping a high-dimensional dataset into a relevant representation of a dataset with fewer dimensions. Nowadays, a considerable quantity of data is processed daily, and this real-world data often has many dimensionalities. Thus working with high dimensional data generate several issues:

- When there are more features than observations, there is more chance of overfitting a model and obtaining poor performance. But, most importantly, it becomes harder to distinguish the clusters with too many features.
- The more dimensions there are, the more the data points tend to be equidistant from all the others. This is a serious issue because clustering uses distance measures to evaluate the similarity between the observations. Consequently, if the distances are more or less equal, the data points are interpreted as similar, and it is hard to highlight meaningful clusters.
- The exponential growth of the space's volume that data can occupy when there is an increase in dimensions. Thus, there is often a very sparse dataset. Ideally, high variational data without redundancy is processed because otherwise, data has no ambiguity and nothing to predict.
- Features that are highly correlated with each other are not essential as there is only a need to analyze one of them to understand their global behavior.

In order to deal with such issues and obtain processable data, dimensionality reduction techniques are used. However, this operation raises new challenges and problems:

- How is the reliability of the reduced data with respect to the original one measured?
- On which criteria the comparison between multiple reduction techniques is made?
- What is the relation between the strength of the reduction and the loss of information?
- How is the stability of reduction techniques analyzed?

Commonly dimensionality reduction is made using classical linear techniques such as Principal Components Analysis, factor analysis, or MDS. Unfortunately, these techniques cannot reasonably manage non-linear data. In contrast to conventional linear techniques, non-linear techniques have the advantage of handling complex non-linear data. Especially for real-world data, the non-linear dimensionality reduction techniques are often more efficient because real-world data is more likely to establish a non-linear manifold. Previous research has shown that non-linear techniques outperform linear techniques on complex tasks. For instance, the Swiss roll dataset includes a set of points on a two-dimensional spiral-like manifold embedded in a three-dimensional space. A lot of non-linear techniques can find this embedding, whereas linear techniques cannot do so.

Consequently, a great number of non-linear techniques have been suggested over the last few years. Theoretically, the reduced representation should have a dimensionality that matches the intrinsic dimensionality of the data. This intrinsic dimensionality can be interpreted as the needed number of variables in a minimal representation of the data.

Correspondingly in signal processing, the intrinsic dimension of the signal describes how many variables are needed to generate a good approximation of the signal. For instance, the swiss roll dataset (Dinoj Surendran, 2004) has an intrinsic dimension of 2 because it is a two-dimensional manifold folded as a roll in a 3D space.

Dimensionality reduction is crucial in various domains because it mitigates the negative aspects of dimensionality and unsought properties of high-dimensional spaces. This paper focuses on two non-linear techniques: UMAP (Leland McInnes John Healy and James Melville, 2020) and t-SNE (Laurens van der Maaten and Geoffrey Hinton 2008). These linear techniques are used for reducing high-dimensional data into a lower-dimensional subspace while keeping original data information. t-SNE is probably the most used non-linear dimensional reduction method. The main point of this technique is to consider the pairwise similarities of the points by minimizing the Kullback-Leibler divergence between the normalized pairwise similarities in low-dimensional embedding and the high-dimensional subspace. Very similarly, UMAP is a more recent dimension reduction technique based on graph theory that uses cross-entropy instead.

Motivated by the lack of a systematic comparison of dimensionality reduction techniques, this paper presents these two techniques followed by an implementation and a comparison of various properties. The paper aims to investigate the stability of this method, the preservation of the data structure after the reduction, and the link between the loss of information with respect to the reduction's strength by proposing a specific methodology. In this work, the following contributions are proposed:

1. A detailed overview of t-SNE and UMAP theory, mainly inspired by the official paper and documentation.
2. A non-optimized and simplified implementation of both methods.
3. Implementation of many known evaluation metrics
4. A proposition to evaluate the stability of an algorithm configuration.
5. Implementation of functions allowing to quickly test, plot, and save different configurations by having complete control of the parameters.
6. An analysis which compare t-SNE and UMAP reductions on different datasets by varying key parameters of the algorithms.

3 Related work

In order to understand the motivations behind reduction techniques, it is essential to mention some of the most used techniques over the past few years. With the rise of data science, large datasets are widespread in many disciplines. Many techniques have been developed to have a better interpretation of such datasets. However, these techniques can be widely classified as linear or non-linear. When the data lies in a linear subspace, it is transformed using a linear combination of the original variables. In contrast, a non-linear dimensional reduction is used when datasets show non-linear relationships like the swiss-roll dataset. In this case, the algorithms try to preserve local distances to establish a similarity ranking for each data point. Besides, there are two main approaches to construct dimensional reduction techniques. First, they can aim to preserve the high dimensional data's local or global structure. By capturing the local structure, a ranking of the data points by similarities is made among a limited neighborhood for each data point, when global methods seek to obtain better preservation of distances even with the points far away from each other.

Principal component analysis (PCA, Pearson, 1901) is one of the oldest and most widely used techniques. The idea is to reduce the dimensionality linearly while preserving the variance of the dataset. Metric Multidimensional scaling (MDS, Torgerson, 1952) is another important global method that aims to minimize the sum of the difference between the high dimensional and low dimensional distances. The main issue with these linear techniques is that they fail to handle real-world data with non-linear relationships efficiently. In order to deal with this issue, many non-linear methods have been developed, particularly Kernel PCA (Kernel PCA, Bernhard Scholkopf, 1997), which is an extension of PCA that takes advantage of the idea that linearly inseparable data can be linearly separable in a higher dimensional space.
There are also non-linear methods that focus on preserving the local structure as Isomap (Tenenbaum et al., 2000), an extension of MDS incorporating geodesic distances, the number of edges in the shortest path connecting them, instead of using straight-line Euclidean distance between data points. Similarly, LLE (Roweis Saul,

2000) is an unsupervised method that tries to preserve the geometric features of the original non-linear structure. It is a topology-preserving manifold learning method that assumes that the dataset lies on a non-linear manifold of low dimensions where the neighborhood structure is intact. Alternatively, Laplacian Eigenmaps (Belkin Niyogi, 2001) suggest a spectral approach where a graph based on a non-linear transformation of the data is constructed and represented by its corresponding Laplacian matrix. More recently, t-SNE, which also propose the preservation of local structure in a stochastic manner (van der Maaten Hinton, 2008), showed great result and had a real impact on data science. However, during these last years, UMAP (McInnes et al., 2018) seems to outperform t-SNE on some criteria, significantly scaling and preserving global data structure.

This evolution of reduction techniques motivated the desire to fully understand the functioning of UMAP and t-SNE, two of the most modern and widely used techniques for a non-linear data structure. Thus this paper will propose a "non-optimal" implementation of UMAP and t-SNE from scratch. Therefore this personal version will be easier to understand as it does not contain the hundreds of lines from the original implementation dedicated to optimization. Consequently, an evaluation and comparison of the scalability of these techniques will not be made because the computation time will not reflect reality. However, these methods' stability and ability to preserve the data structure can still be analyzed.
The evaluation methodology will be guided by multiple articles about the evaluation of dimensionality reduction techniques. In particular, an analysis of the stress value between a dataset before and after the reduction (Stephen France, 2007), a paper about the preservation of the topology by using spearman's rho, Konig's measure and mean relative rank errors (Rasa Karabauskaite,2008), a study of stability of dimensionality reduction techniques (Francisco J. García-Fernández, 2013), a methodology to compare algorithms in terms of loss of quality (Antonio Gracia, 2014), tools to evaluate the accuracy of different methods (Olga Kurasova, 2018) and an aggregation of metrics to evaluate overall stability proposed by Ruizhi Xiang in 2021.

4 Definitions and Notations

To fully understand this paper, it is necessary to define some mathematical concepts and their notation. Given a dataset X of size $N \times D$ where N is the number of data points, a dimension reduction operation can be defined as follow:

$$f : \mathbb{R}^D \to \mathbb{R}^d$$

$$X \to Y$$

where D is the initial dimension of the feature set and d is the target dimension with $d < D$.

Some linear techniques like PCA are efficient if there is data with a linear structure as they aim to represent the data in a linear subspace. More formally, if V is defined as a vector space over a field K, a subset W of V is a linear subspace of V if, under the operations of V, W is a vector space over K.

To put it simply W is a linear space if:

$$u, v \in W \Rightarrow \alpha u + \beta v \in W \qquad \alpha, \beta \in \mathbb{R}$$

W is a non linear space if:

$$u, v \in W \Rightarrow \alpha u + \beta v \notin W \qquad \alpha, \beta \in \mathbb{R}$$

However, if the data have many non-linear relationships, the use of techniques that can learn a non-linear manifold is preferred. Let's define a topological space rigorously to have a better understanding of what a manifold is. Let X be a dataset and $\mathcal{Z}$ a neighborhood function assigning for each x $\in$ X a non-empty subset Z of X. Then (X,$\mathcal{Z}$) is called a topological space:

- If Z is a neighborhood of x, then x $\in$ Z. Each point belongs to its neighborhood.
- If Z is a subset of X and includes a neighborhood of x, then Z is a neighborhood of x.
- The intersection of two neighborhoods of x is a neighborhood of x.
- Any neighborhood Z of x includes a neighborhood U of x such that Z is a neighborhood of each point of U.

A manifold is a generalization and abstraction of a curved surface, a topological space that is locally linear and similar to a Euclidean space. In other words, each point is locally homeomorphic to a Euclidean space. A homeomorphism is a function f: X $\rightarrow$ Y between two topological spaces with the following properties:

- f is continuous and bijective
- f^{-1} is continuous

There is also some needed mathematical knowledge to have in order to understand t-SNE. Thus some essential notions related to this method will be defined here. t-SNE exploits the entropy and perplexity to evaluate the dataset's structure and adjust some crucial parameters. The entropy measures the uncertainty or the "level of information", given a random variable X with values in the set $\mathcal{X}$ following a probability distribution p: x $\rightarrow$ [0,1], the entropy of X is defined as follow:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) log(p(x))$$

This notion is directly related to perplexity as it is defined with the entropy of a given probability distribution p:

$$Perplexity(p) = 2^{H(p)}$$

Which is a good indicator of how complex the prediction can be for a given distribution.

Finally, t-SNE also use the Kullback–Leibler divergence (KL), a statistical function measuring how much two probability distribution P and Q are dissimilar. The KL divergence of P from Q can also be interpreted as the error when using Q to model a distribution P. More formally, let's define P, and Q on the same probability space $\mathcal{X}$, the KL divergence between P and Q is given by:

$$D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) log(\frac{P(x)}{Q(x)})$$

This function is called relative entropy and is used as a cost function to optimize the embedding in the algorithm.

Concerning UMAP, there is a need to discuss and define a simplicial set since this notion is used to construct our low-dimensional manifold. These sets are a generalization of a directed graph and are composed of simplices in a particular way.
A simplex is the simplest possible polytope representation with line segments in any dimension. For instance, a 0-simplex is a point, a 1-simplex is a line segment, and a 2-simplex is a triangle. More generally a k-simplex is a k-dimensional polytope composed of k+1 affinely independent points u_0,...,$u_k \in \mathbb{R}^k$. Thus a simplex C is defined as:

$$C = \{\theta_0 u_0 + ... + \theta_k u_k | \sum_{i=0}^{k} \theta_i = 1 \ and \ \theta_i \geq 0 \ for \ i = 0, ..., k\}$$

Figure 1: simplex
source

In particular, a simplicial set is a mapping between categories. Let Δ be the simplex category where the objects are nonempty linearly ordered sets with the following form:

$$[k] = \{0, 1, ..., k\}$$

where $k \geq 0$.

A simplicial set X is determined by:

$$X : \Delta \rightarrow Set$$

With Set, the category of sets.

However, UMAP uses a fuzzy simplicial set which is slightly different. They are sets whose elements have degrees of membership following the principles of fuzzy logic. It is defined with a pair (A,v) where A is a non-empty set and v: A → [0,1] is a membership function. For x ∈ A, v(x) gives the membership level of x in (A,v). Thus v = μ_F is the membership function of the fuzzy set F = (A,v).

Lastly, the cost function used in UMAP is cross-entropy. Given two discrete probability distributions p and q, it can be interpreted as the entropy for Q using probability weights of P:

$$H(p,q) = -\sum_{x \in \mathcal{Z}} p(x) log(q(x))$$

Since the entropy and KL divergence are already defined, it can also be formulated in the following way:

$$H(p,q) = H(p) + D_{KL}(p||q)$$

Now that there is a good overview of the different concepts used in t-SNE and UMAP, a detailed description of these methods will be made.

5 Methods

5.1 t-SNE

5.1.1 SNE

As already mentioned, the purpose of the Stochastic Neighbor Embedding (SNE) is to find a meaningful and reduced representation of the data. The first idea behind SNE is to preserve the similarity between high-dimensional points in their low-dimensional representation. To do so, a matrix $(P_{j|i})_{i,j=1...N}$ containing the similarities between points (x_i) is constructed. The affinity between two points i and j is expressed with a conditional probability $p_{j|i}$ given by a Gaussian distribution:

$$p_{j|i} = \frac{\exp(-||x_i - x_j||^2/2\sigma^2)}{\sum_{k \neq i} \exp(-||x_i - x_k||^2/2\sigma_i^2)} \tag{1}$$

The probability depends on the squared L2 norm of Euclidean distance between data points and a given standard deviation σ that will be described later as it is an important aspect of SNE. The intention behind this is to represent the neighborhood of a point as a distribution with $\sum_j p_{j|i} = 1$ and $0 < p_{j|i} < 1$. In this case $p_{j|i}$ expresses the probability that a point x_i would pick x_j as its neighbor. In addition, normalization is made by summing each probability in the neighborhood distribution of a given point x_i.

These distributions need to be preserved in our reduced space, so the same reasoning is used. After initializing our embedding (y_i), a matrix $(Q_{j|i})_{i,j=1...N}$ containing the similarities between each reduced data point is defined:

$$q_{j|i} = \frac{\exp(-||y_i - y_j||^2/2\sigma^2)}{\sum_{k \neq i} \exp(-||y_i - y_k||^2/2\sigma_i^2)} \tag{2}$$

where $\sigma = \frac{1}{\sqrt{2}}$ is fixed (again this value will be discussed later)

$$q_{j|i} = \frac{\exp(-||y_i - y_j||^2)}{\sum_{k \neq i} \exp(-||y_i - y_k||^2)} \tag{3}$$

Ideally, if our embedding accurately models the affinity between data points, the probabilities $p_{j|i}$ and $q_{j|i}$ will be equal. In reality, there is a minimization of the difference between these two distributions. Consequently, the Kullback-Leibler divergence is used as a loss function to update our embedding points using a gradient descent method over all data points. So there is a cost function given by:

$$C(P,Q) = Kl(p,q) = \sum_i \sum_j p_{j|i} log(\frac{p_{j|i}}{q_{j|i}}) \tag{4}$$

where $p_{i|i} = q_{i|i} = 0$.

KL divergence is not symmetric, so the pairwise affinity of our embedding is not equally weighted. Thus widely separated points in our reduced space (small $q_{j|i}$) induce a high cost to represent nearby points in our initial space (large $p_{j|i}$). In contrast, there is a small cost to represent separated data points by nearby embedding points, which means that this cost function favors the local structure of the data in the reduced space.

Now that the objective of this method is described, it is crucial to figure out how the standard deviation of the Gaussian distribution affects the results. A good value of σ will depend on the sparsity of the dataset. A smaller value is generally better when our points are close to each other because a better distinction between the nearest points is made. The larger σ is, the higher the entropy of the distribution is:

$$H(P_i) = -\sum_j p_{j|i} log_2(p_{j|i}) \tag{5}$$

In our case, SNE performs a binary search to find a good standard deviation value for each distribution. Thus a standard deviation that produces a perplexity fixed by the user is desired:

$$Perplexity(P_i) = 2^{H(P_i)} \tag{6}$$

Since entropy is the number of bits needed to encode the information in P, perplexity is the number of configurations in the sample space related to the number of samples. Perplexity can also be interpreted as a measure of the number of neighbors of the data points. Generally, it takes a value between 5 and 50.

5.1.2 t-SNE

Conditional probabilities are used to define the neighborhood distribution. However, it is also possible to use joint probabilities to obtain a simpler gradient to speed up the training. So a minimization of Kullback-Leibler divergence between a joint probability distribution in the high and low dimension space is made:

$$C(P,Q) = Kl(p,q) = \sum_i \sum_j p_{i,j} log(\frac{p_{i,j}}{q_{i,j}}) \tag{7}$$

where again $p_{i,i} = q_{i,i} = 0$.

This variation is called *Symmetric SNE* because $p_{i,j} = p_{j,i}$, $q_{i,j} = q_{j,i}$ and it defines:

$$q_{j,i} = \frac{\exp(-||y_i - y_j||^2/2\sigma^2)}{\sum_{k \neq l} \exp(-||y_k - y_l||^2/2\sigma_i^2)} \tag{8}$$

There is a slight difference with equation (2). Here there is a normalization using the entire matrix, not row by row, with the same reasoning for the joint probability in high-dimension space:

$$p_{i,j} = \frac{\exp(-||x_i - x_j||^2/2\sigma^2)}{\sum_{k \neq l} \exp(-||x_k - x_l||^2/2\sigma_i^2)} \tag{9}$$

However this induces very small ($p_{i,j}$) when there is an outlier x_i because all pairwise distance $||x_i - x_j||^2$ will be too large for this x_i. Consequently, y_i will have a negligible impact on the cost function, and his position will have an inaccurate update by other embedded points. In order to avoid this, the previously conditional probability is used and applied:

$$p_{i,j} = \frac{p_{j|i} + p_{i|j}}{2N} \tag{10}$$

where N is the number of sample in our dataset.

With this, all points have enough contribution to the cost function:

$$\sum_j p_{ij} > \frac{1}{2N} \ \forall x_i \tag{11}$$

Unfortunately, sometimes it is impossible to preserve the distance of the whole neighborhood when the dimension space is reduced. Such a problem is called *The Crowding problem*. It is a classical problem related to local

techniques. The idea is that many points are at a moderate distance in high dimension from a point i that is far away in the reduced space and induce a small attractive force on the mapped point. Even if these forces are small, they are numerous. Thus, the mapped points tend to be compressed in the middle of the reduced space, which prevents the formation of separate clusters. In order to avoid this phenomenon, the gaussian distribution used in the low-dimensional (equation 8) is transformed into a Student t-distribution with one degree of freedom (Cauchy distribution). This distribution has heavier tails than a Gaussian distribution. Therefore it enables points in a moderate distance to be mapped to a much larger distance in the reduced space. Subsequently, the unnecessary attractive forces are removed.

As a result, the following t-SNE model is obtained where:

$$q_{j,i} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^{-1}} \tag{12}$$

Now that the transition from SNE to t-SNE is understood, the gradient between the two distributions is the following:

$$(t - SNE): \qquad \frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i + y_j||^2)^{-1} \tag{13}$$

t-SNE gradient shows advantages compared to traditional SNE:

$$(SNE): \qquad \frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j) \tag{14}$$

Different data points with small pairwise affinity in the reduced space are much more separated at each step because there is a more substantial repulsive effect, and the strength of the repulsion cannot go to infinity. In addition, the optimization of this new cost function is easier because t-SNE establishes longer-range forces which can bring together similar points in the reduced space far away at the beginning of the optimization. In other words, t-SNE does not need to use a mechanism like simulated annealing to boost the exploration of solutions and find good local optima.

To speed up the optimization, it is possible to multiply our initial probabilities $p_{i,j}$ by a given factor in the first iteration. Obviously, by doing this, the $q_{i,j}$ cannot model their corresponding $p_{i,j}$, so it will produce larger $q_{i,j}$, which form a separated cluster in the reduced space. Later different initialization methods of the y_i will also be analyzed.

5.2 UMAP

5.2.1 Manifold And Fuzzy Topological

Now that there is a good understanding of t-SNE, the theory underlying the UMAP algorithm will be discussed. This new algorithm shows many advantages compared to t-SNE. Many studies demonstrate that UMAP has better scaling and better global structure preservation. Here the mathematical motivation behind UMAP is presented to discuss the crucial points and differences between UMAP and t-SNE.

A simplicial set is composed of simplices in a particular way and can be interpreted as a generalization of a directed multigraph.

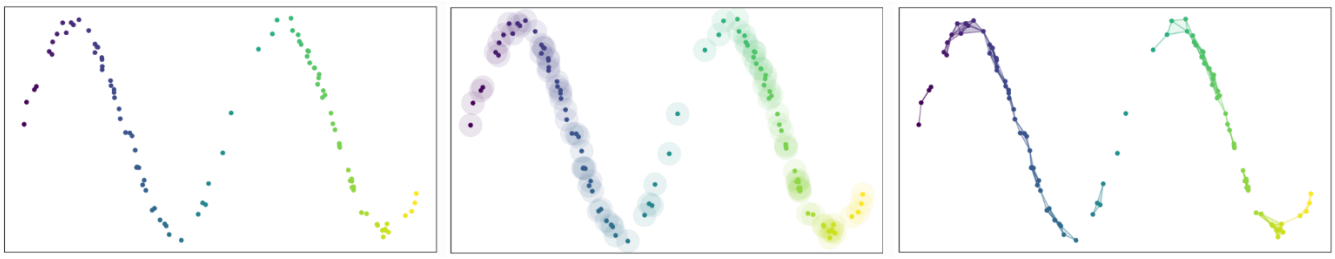


Figure 2: Dataset of a noisy sine wave → open sets of the cover as circle → A simplicial complex built from the test data
source

Figure 2 presents a dataset sampled along a sine wave (left). The aim is to fix a radius, so the data is covered using circles. Then by using 0-,1- and 2-simplices, a simplicial complex composed of simplices formed out of points whose circles overlap can be built. Therefore UMAP algorithm aims to find a reduced space with a similar topological representation. In short, a neighborhood graph on the data is constructed and laid it out in lower dimensions.

So UMAP uses local manifold approximations and puts together their local fuzzy simplicial set to obtain a new representation of the initial space. Capturing manifold structure with real-world data is not that simple because picking the correct radius of the circles that cover our data is complicated when there is no uniform distribution. If this radius is too small, the simplicial complex will be split, and on the other side, accurate capture of the manifold with too large a value will fail.

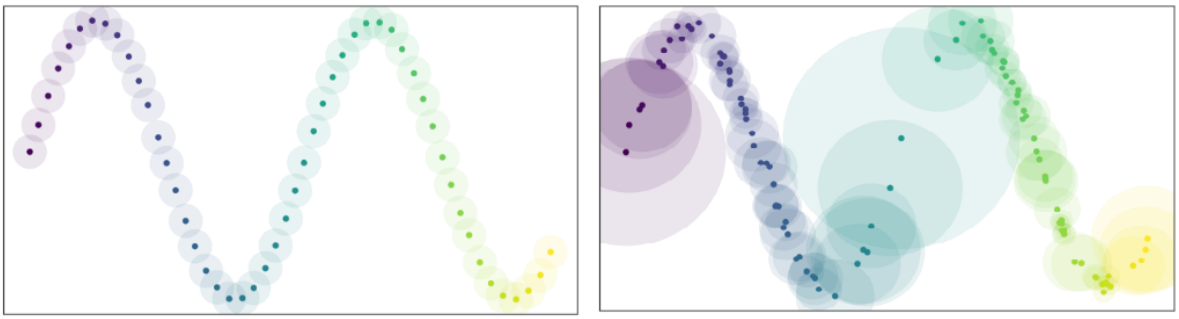


Figure 3: uniform vs real-world data
source

When the data is uniformly distributed (figure3, left), finding one unique radius to cover the manifold is no problem. However, instead of using a ball with a given radius, a k-neighbor graph with real-world data can be used to define the connectivity. So for each data point, a good value of k nearest neighbors need to be found so that the result of having circles of radius 1 with different locally metric is simulated. Furthermore, this method is less dependent on the dataset's structure as a reasonable value of k can be chosen for most datasets. The value of k defines how the structure of the manifold is captured locally. A small k means an accurate manifold estimation, and choosing a large k will capture larger regions. Now let us visualize the neighborhood of our points as a fuzzy set instead of a regular one. The idea is that being in a set is no longer a binary response but a fuzzy value in [0,1]:

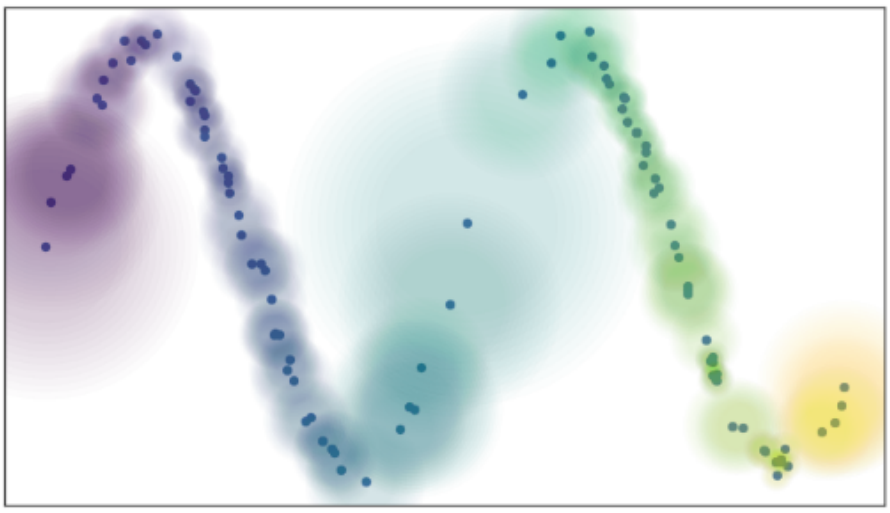


Figure 4: Fuzzy set representation
source

Each data point has its neighborhood distribution, where the probability of being a neighbor decays as it

gets far from this point.

Besides, having different local metrics raises a new problem. Each point has its metric, so how can one know which distance is right? For example, if there is a distance of 2 from point x to y but a distance of 1 from point y to x, what should be considered? This conflict between the two perspectives can be illustrated by a directed graph where there are two edges with different weights between two same points. Since there are fuzzy simplicial sets, it is possible to take their union to obtain a graph with combined edge weights. In order to merge an edge having weights a and b with probabilistic semantics, a combination by computing a new weight given by: **(a + b) - (a·b)** is possible. The weights can be interpreted as the probability for an edge to exist. On the other hand, the combined weight is the probability that at least one of the edges exists.

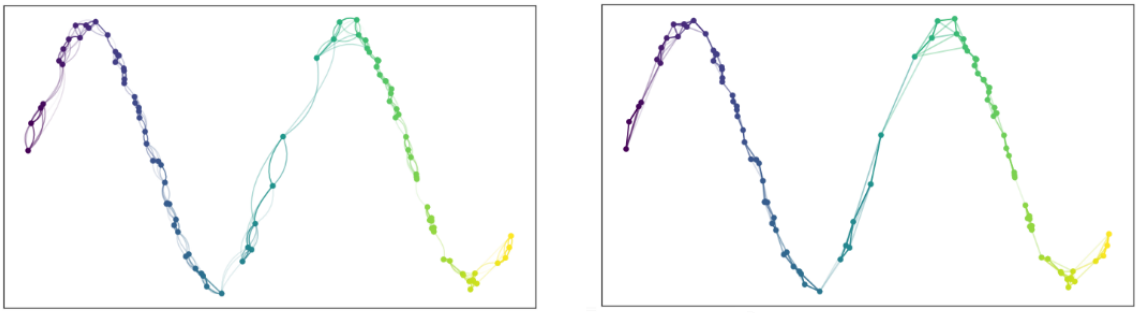


Figure 5: Incompatible weights vs combined weights
source

This explanation justifies using a k-neighbor graph to represent a manifold. In particular, it clarifies the essential motivation in terms of topological theory.

From a more practical perspective, UMAP can be described in terms of construction and operations on a weighted graph. Ultimately UMAP is a k-neighbor-based graph technique where a k-neighbor graph is constructed (the high dimensional manifold). Let's define a dataset $X = \{x_1, ..., x_N\}$ and a chosen distance metric $d : X \times X \to \mathbb{R}_{\geq 0}$. Given an hyperparameter k, the k nearest neighbour is computed for each x_i: $\{x_{i1}, ..., x_{ik}\}$ and define ρ_i as follow:

$$\rho_i = min\{d(x_i, x_{i_j}) | 1 \leq j \leq k, d(x_i, x_{i_j}) > 0\} \tag{15}$$

In simple terms ρ_i is the distance to the nearest neighbor for the point i. In addition let fix σ_i such that:

$$\sum_{j=1}^{k} \exp(\frac{-max(0, d(x_i, x_j) - \rho_i}{\sigma_i}) = log_2(k) \tag{16}$$

The x_i need to be at least connected to one other data point to obtain a connected graph covering the high dimensional manifold. This local-connectivity constraint is expressed by selecting ρ_i. On the other hand, σ_i define a normalization factor defining the Riemannian metric local to the point x_i. These definitions allow us to describe a directed weighted graph G = (V,E,w) where the vertices V are the dataset X and the edges E are the pairs $\{(x_i,x_{i_j})\ |1 \leq j \leq k, 1 \leq i \leq N\}$. Finally, the weights are defined as:

$$w((x_i, x_{i_j})) = \exp(\frac{-max(0, d(x_i, x_j) - \rho_i}{\sigma_i}) \tag{17}$$

A graph is obtained, which is the 1-skeleton of the fuzzy simplicial set where the weight of each edge is the membership strength to the fuzzy set. The weights need to be combined to obtain a coherent graph despite the different local metrics. Consequently, the following operation must be done:

$$B = A + A^T - A \circ A^T \tag{18}$$

where A is the adjacency matrix of G and $\circ$ the component-wise product.

A transformation into reduced space is needed now that there is a better understanding.

5.2.2 Low Dimensional Representation

The next step of the UMAP algorithm is to find a corresponding fuzzy topological structure in the low dimensional space. The target manifold can be chosen for Y (the mapped data points) as $\mathbb{R}^d$ where d is the chosen reduced dimension. The fuzzy structure representation is computed without tricks to make the data uniformly distributed. With this purpose, the same approach is used without varying the notion of distance over the manifold since embedding the data into a low euclidean space is desired. However, the same distance to the nearest neighbor is wanted. Therefore, to globally satisfy this property over the manifold during the optimization, a parameter **min_dist** specifying the expected distance between nearest neighbors in the reduced space needs to be defined. The smaller this parameter, the more packed the clusters will be after the reduction.

A measure of closeness between two fuzzy sets is needed to find the closest low-dimensional representation by resolving an optimization problem. A fuzzy set is defined by a reference set A and a membership function μ : A $\rightarrow$ [0,1]. In addition, two fuzzy sets must have the same reference set to be compared. Thus by using the cross entropy, the following result is obtained.

$$C((A,\mu),(A,v)) = \sum_{a\in A}(\mu(a)log(\frac{\mu(a)}{v(a)}) + (1-\mu(a))log(\frac{1-\mu(a)}{1-v(a)})) \tag{19}$$

To put it another way, the weights of the graph are the probability of the simplex existing. Since both topological shares the same 0-simplices, it is possible to use two vectors of probabilities indexed by the 1-simplices to make the comparison between our two fuzzy sets. This can be seen as Bernoulli variables as the simplex can exist or not. Let's define E as the set of all possible 1-simples, $w_h(e)$ the weight of the 1-simplex e in the original space and $w_l(e)$ the weight of e in the reduced space. Then the cross entropy can be defined as:

$$C(w_h,w_l) = \sum_{e\in E}(w_h(e)log(\frac{w_h(e)}{w_l(e)}) + (1-w_h(e))log(\frac{1-w_h(e)}{1-w_l(e)})) \tag{20}$$

$w_h(e)log(\frac{w_h(e)}{w_l(e)})$ can be interpreted as the attractive force between the points e when there is a large weight in the original space because when the distance between the points are small, large $w_l(e)$ is obtained which minimize this term. In opposition $(1-w_h(e))log(\frac{1-w_h(e)}{1-w_l(e)})$ is a repulsive force between the points e when $w_h(e)$ is small, as it is minimized by having a small $w_l(e)$.

The algorithm iteratively applies the attractive and repulsive forces at each edge or vertex. The convergence to a local minimum is guaranteed by decreasing these forces slowly. It is possible to take the previously defined graph (equation 17) for a more practical view. The attractive force between two vertices i and j at position y_i and y_j is given by:

$$\frac{-2ab||y_i-y_j||_2^{2(b-1)}}{1+||y_i-y_j||_2^2}w((x_i,x_j))(y_i-y_j) \tag{21}$$

where a and b are hyper parameters described later. Whenever there is an attraction on edge, one of his vertices is repulsed by a sampling of other vertices. More formally, the repulsive force is defined by:

$$\frac{2b}{(\epsilon+||y_i-y_j||_2^2)(1+a||y_i-y_j||_2^{2b})}(1-w((x_i,x_j)))(y_i-y_j) \tag{22}$$

where ϵ prevent division by zero when $y_i = y_j$

Following t-SNE, the embedding Y is optimized regarding the cross entropy obtained using stochastic gradient descent.

Additionally, it is possible to use more than the 1-simplices set to compute our cost function. l-simplices set can be incorporated by defining a new cost function C_l as:

$$C_l(X,Y) = \sum_{i=1}^{l}\lambda_i C(X_i,Yi) \tag{23}$$

where λ_i are the chosen weights and X_i the fuzzy set of i-simplices. This technique captures the topological structure more precisely, even if it increases the computational cost.

6 Implementation

6.1 t-SNE

First, initialization of the mapped points Y is needed to update it iteratively at each step. In this implementation, four different initialization methods are used:

- Gaussian: Use of a Gaussian distribution to sample our points Y.
- PCA: Principal component analysis to perform a linear dimension reduction technique on our data to initialize our embedding.
- Laplacian Eigenmaps: Spectral clustering to reduce dimensionality by exploiting the eigenvalues of the similarity matrix of the data.
- Random: Random uniform sampling between a specific range to initialize Y.

As discussed earlier, a matrix containing the similarities between all points needs to be constructed. This similarity is expressed as a conditional probability $p_{j|i}$ (equation 1), which depends on the distance between point i and j. Therefore the first step is to compute the negative squared euclidean distance $-||x_i - x_j||^2$ between all pairs of points in a matrix. Some linear algebra is used to turn this operation into a matrix operation in order to reduce the computation time:

$$D = R - 2AA' + R' \tag{24}$$

where D is our squared euclidean distance matrix, A our data matrix, and $R(i)$ is the squared norm of the original matrix's i^{th} row. So R is of dimension $N \times 1$ and becomes $N \times N$ after the NumPy automatic broadcasting rule, which broadcasts N times the row's value on the N new columns.

-D gives the negative squared euclidean distance used in equation 1 to compute our conditional probability by using the softmax function for each row $x_i = (v_1 ... v_N)$:

$$Softmax(x_i) = \frac{\exp(v_i)}{\sum_j^N \exp(v_j)}$$

$$\Leftrightarrow v_j = -||x_i - x_j||^2 / 2\sigma_i^2$$

In addition, the diagonal is set to $\epsilon^2 =$ 1e-6 instead of zero to add stability when the entropy is computed during the binary search as $log_2(0)$ is undefined.

Finally, these conditional probabilities also depend on a given standard deviation that produces a perplexity value fixed by the user. Thus for each row of our data, a binary search is performed where multiple standard deviations are tested, and their corresponding perplexity is computed. At each iteration the obtained perplexity and the given target perplexity are compared, and if the difference is smaller than a chosen tolerance ($10e - 5$ in this paper), the current standard deviation is returned.

Here is the binary search algorithm:

Algorithm 1 Binary Search

```
Binary Search
    low ← 1
    upper ← 100
    for i ← 1 to max_iteration do
        σ ← (low + upper)/2
        perplexity_current ← compute_perplexity(σ)
        if |perplexity_current − target_perplexity| < tol then
            break
        else
            if perplexity_current < target_perplexity then
                low ← σ
            else
                upper ← σ
    return σ
```

where low is the lower bound of the search and upper the upper bound.

After all, a vector containing the optimal standard deviation for each row is returned. Consequently, the correct conditional probability can be computed and transformed into joint probability (equation 10).

The next step is to compute our probability q in our reduced space. Similarly, the negative squared distance D_q needs to be computed the same way it is for $p_{i,j}$ on the mapped points Y. Then a matrix operation is applied to obtain our probability: $(1 - D_q)^{-1}$. Additionally, the diagonal need to be fixed to 0 in order to obtain $q_{i,j} = 0$. Then a normalization by the sum of the entire matrix is made.

The last step is to get the gradient to update the mapped points at each iteration. As a means to have the simplest way to compute the gradient described in equation 13, it is possible to use four matrix operations and a basic trick. Since $(1 + ||y_i + y_j||^2)^{-1}$ need to be computed to obtain our q distribution, the matrix J can be kept in memory and passed as a parameter when the gradient is computed in order to avoid the useless cost. Then the dimension of these matrix P,Q,Y and J can be adjusted to obtain a final matrix containing $(p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i + y_j||^2)^{-1}$ by making subtractions and component-wise multiplication. Finally, the matrix is multiplied by 4, and the sum of its columns is taken.

Then straightforward gradient descent is used to update our mapped points:

$$Y_k \leftarrow Y_{k-1} - learning_rate * gradient_k$$

Moreover, momentum can be added to help accelerate gradients vectors in the right directions to obtain faster convergence:

$$Y_k \leftarrow Y_k + (Y_k - Y_{k-1})$$

6.2 UMAP

Henceforth the implementation of UMAP will be discussed. As mentioned earlier, the graph is characterized as a vector of probabilities indexed by the 1-simplices. Similarly to t-SNE, it is necessary to construct a matrix D containing the squared pairwise Euclidean distances of our initial data X. In contrast to t-SNE, any other distance metric can be used, but in this example, only the Euclidean distance is used. Next, ρ_i the local connectivity parameter is computed for each data point. The weight defined in equation (17) can be interpreted as a conditional probability to represent the neighborhood distribution of a given point:

$$p_{i|j} = \exp(\frac{-D(x_i, x_j) - \rho_i}{\sigma_i})$$

Now the optimal standard deviation which satisfies equation 16 needs to be found. In the same manner, as t-SNE, a binary search is performed:

Algorithm 2 Binary Search

```
Binary Search
    low ← 1
    upper ← 100
    for i ← 1 to max_iteration do
        σ ← (low + upper)/2
        k_current ← compute_k(σ)
        if |k_current − k_target| < tol then
            break
        else
            if k_current < k_target then
                low ← σ
            else
                upper ← σ
    return σ
```

With these standard deviations, the probabilities which satisfy our initial condition can be computed. However, a symmetry condition is needed to implement a coherent directed graph with combined weights (equation 18). Thus the following operation is performed:

$$p_{i,j} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i}$$

In modeling distance probabilities in low dimensions, a better understanding of the cost function defined in equations 19 and 20 is needed.

$$\begin{aligned} C((A,\mu),(A,v)) &= \sum_{a\in A}(\mu(a)log(\frac{\mu(a)}{v(a)}) + (1-\mu(a))log(\frac{1-\mu(a)}{1-v(a)})) \\ &= \sum_{a\in A}(\mu(a)log(\mu(a)) + (1-\mu(a))log(1-\mu(a))) \\ &\quad - \sum_{a\in A}(\mu(a)log(v(a)) + (1-\mu(a))log(1-v(a))) \end{aligned}$$

The first term only depends on μ, so the cross entropy optimization only depends on the second term because μ takes fixed values. Thus the following is minimized:

$$-\sum_{a\in A}(\mu(a)log(v(a)) + (1-\mu(a))log(1-v(a))) \tag{25}$$

where $\mu(a)$ is our high dimensional probability and v(a) our low dimensional probability.

Consequently, there is also a need to find a differentiable approximation to v(a) for a given 1-simplex in order to do the optimization. UMAP defines a smooth approximation of v(a) in the following way:

$$\Phi(x,y) = (1 + a(||x-y||_2^{2b}))^{-1} \tag{26}$$

Φ: $\mathbb{R}^d \times \mathrm{R}^d \rightarrow$ [0,1] the membership strength of a 1-simplex between two points in $\mathbb{R}^d$. a and b are given by non-linear least squares fitting against the curve Ψ: $\mathbb{R}^d \times \mathrm{R}^d \rightarrow$ [0,1] where:

$$\Psi(x,y) = \begin{cases} 1, & \text{if} ||x-y||_2 \leq \text{min_dist} \\ \exp(-(||x-y||_2 - \text{min_dist})), & \text{otherwise} \end{cases} \tag{27}$$

In short, min_dist impacts how the clusters are packed in the reduced space, this parameter is given by the user and used to find optimal a and b parameters for the family of curves $1/(1 + a * y^{2b})$. So the following function is used to model distance probabilities in low dimensions:

$$q_{i,j} = (1 + a(y_i - yj)^{2b})^{-1} \tag{28}$$

Now that our two distributions function P and Q exist, the initial embedding Y needs to be optimized. In this implementation, the initialization method can be chosen. However, UMAP recommends Laplacian Eigenmaps or random initialization. Obviously, to update our mapped points, a derivation of our cost function is needed:

$$CE(X, d_{ij}) = \sum_j \left[-P(X)\log Q(d_{ij}) + (1 - P(X))\log(1 - Q(d_{ij}))\right], \qquad \text{where } d_{ij} = y_i - y_j$$

$$Q(d_{ij}) = \frac{1}{1 + ad_{ij}^{2b}}; \qquad 1 - Q(d_{ij}) = \frac{ad_{ij}^{2b}}{1 + ad_{ij}^{2b}}; \qquad \frac{\delta Q}{\delta d_{ij}} = -\frac{2abd_{ij}^{2b-1}}{\left(1 + ad_{ij}^{2b}\right)^2}$$

$$\frac{\delta CE}{\delta y_i} = \sum_j \left[-\frac{P(X)}{Q(d_{ij})}\frac{\delta Q}{\delta d_{ij}} + \frac{1 - P(X)}{1 - Q(d_{ij})}\frac{\delta Q}{\delta d_{ij}}\right] =$$

$$\sum_j \left[\left(-P(X)\left(1 + ad_{ij}^{2b}\right) + \frac{(1 - P(X))\left(1 + ad_{ij}^{2b}\right)}{\left(ad_{ij}^{2b}\right)}\right)\frac{\delta Q}{\delta d_{ij}}\right]$$

$$\frac{\delta CE}{\delta y_i} = \sum_j \left[\frac{2abd_{ij}^{2(b-1)}P(X)}{1 + ad_{ij}^{2b}} - \frac{2b(1 - P(X))}{d_{ij}^2(1 + ad_{ij}^{2b})}\right](y_i - y_j)$$

source

Figure 6: Rewrite and differentiate the cross-entropy

With this derivation, our cost function can be minimized (equation 20). Let us rewrite it in a more probabilistic way:

$$CE(X, Y) = \sum_i \sum_j [p_{ij}(X)log(\frac{p_{ij}(X)}{q_{ij}(Y)}) + (1 - p_{ij}(X))log(\frac{1 - p_{ij}(X)}{1 - q_{ij}(Y)})] \tag{29}$$

where the first term $p_{ij}(X)log(\frac{p_{ij}(X)}{q_{ij}(Y)})$ simulate the attractive force between the points of our graph whenever there is a high probability in the original space. Once the distance between the points is small, $q_{ij}(Y)$ will be large, which minimizes this term. In addition the second term $(1 - p_{ij}(X))log(\frac{1-p_{ij}(X)}{1-q_{ij}(Y)})$ is minimized when q_{ij} is small so a repulsive force between the ends of our 1-simples is obtained when p_{ij} is small.

Finally, our gradient is computed at each step and updates our embedding similarly to t-SNE:

$$Y_k = Y_{k-1} - learning_rate * gradient_k$$

The original implementation of UMAP uses Stochastic Gradient Descent instead of regular Gradient Descent, which speeds up the computation time. Moreover, the official version uses a sampling-based approach during the optimization where a 1-simplices is sampled with probability $\mu(a)$ and updated according to $v(a)$, which deals with the term $\mu(a)log(v(a))$ (equation 25). On the other side a negative sampling is used to handle the term $(1 - \mu(a))log(1 - v(a))$. Rather than computing these values over all simplices, 1-simplicies are randomly sampled and considered as a negative example with membership strength of 0 and updated according to the value of $1 - v(a)$.

7 Evaluation

7.1 Datasets

In this section, an analysis between t-SNE and UMAP is made using a specific methodology on different kinds of datasets containing 200 points for the first experiment and 500 points for experiments 2 and 3:

- **Swiss roll dataset:**
  Generated by mapping the 2 dimensional plot obtained from a gaussian distribution to a 3 dimensional space using (x,y) → (xcos(x),y,xsin(x)). This dataset is widely exploited in various scientific documentation to demonstrate diverse results on dimensionality reduction.

- **Randomly generated clustered data**
  The make_blobs function proposed in sklearn.datasets libraries is used to generate clustered data in a specified dimension with 3 clusters in a bounding box [-10,10] for each center.
- **Randomly generated nonclustered data**
  Uniform data with specific dimensions is generated within the interval [-5,5].

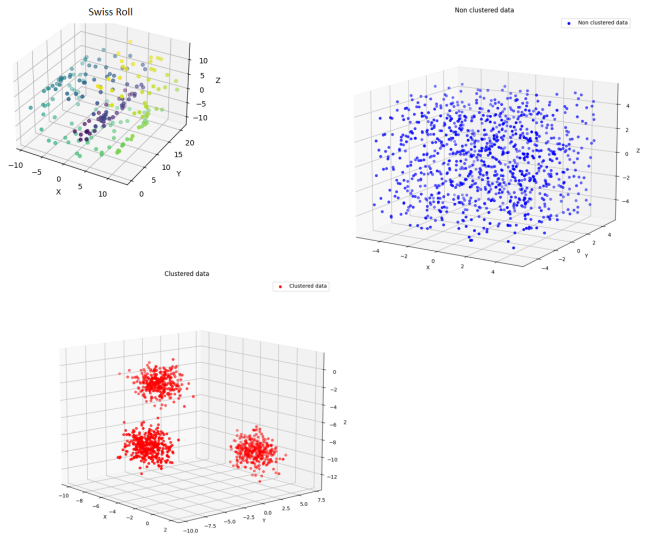


Figure 7: Datasets

7.2 Measures of quality

7.2.1 Shepard Diagram

The first measure is the Shepard Diagram which allows visualizing the mapping between the distances in the high dimensional and the distance in the reduced space.

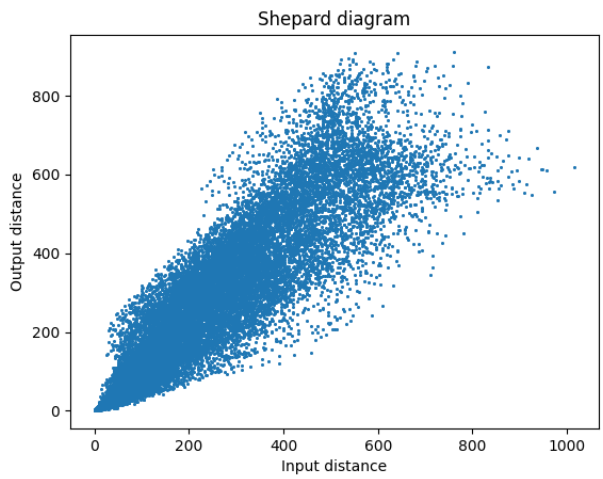


Figure 8: Shepard Diagrams

Ideally, all the points lie in the diagonal line with all the distances in the low dimensional space matching the original distance, but obviously, it is almost impossible.

7.2.2 Configuration score

In order to compare multiple configurations of the same reduction technique, a stability score is defined for each configuration to evaluate which run has better conservation of the distances. The reduction is performed using different parameters on the three datasets defined earlier. The idea is to take s1 pivot points and their distances to s2 other points (s2>s1). By doing this, a matrix S of dimension s1× s2 is obtained where S[i][j] indicates the distance between the pivot point i and the point j. Then the mean distance on each row is computed to obtain an s1× 1 vector. This method is performed on the original dataset and the reduced dataset in order to compute the absolute value of the subtraction between the two obtained vectors. This allows us to obtain a scalar **d** by computing the mean value of this final vector. However, as this scalar can be hard to interpret because the conservation of distance highly varies depending on the used technique and its parameters, this value is mapped to the interval [0,1] by using the following formula:

$$score = 1 - \frac{x_i - min(x)}{max(x) - min(x)}$$

where x contain all the computed scalar **d** for each configuration.

Using this mapping, a score of 1 is obtained for the configuration with the best conservation of the distance and a score of 0 for the worst. It allows us to interpret the efficiency of the reduction technique better. However, it also restricts the comparison to the tested configurations and does not give us a global measure of evaluation of the technique.

7.2.3 Stress

Stress is used in MDS as a cost function that consists of the square root of the sum of squared differences between the high dimensional distances and low dimensional distances normalized by the sum of the initial squared distances:

$$stress = \sqrt{\frac{\sum_i \sum_j (d_{ij} - \hat{d}_{ij})^2)}{\sum_i \sum_j d_{ij}^2}}$$

It varies from 0 to 1 where 0 indicates perfect conservation of the distance after the reduction.

7.2.4 Spearman's rho

Spearman's rho is used to estimate the preservation of the topology, and it evaluates the correlation of rank order after the reduction. In other words, it measures the preservation of the order of the pairwise distances. The computation of a rank matrix R given a distance matrix A before and after the reduction is needed. For a dataset M of dimension N× D the distance matrix A ∈ N× N is defined where A(i,j) contains the Euclidean distance between the points i and j. Now for a given distance matrix, R ∈ N× N can be computed where R(i,j) is the rank of j for the point i. To put it simply, point j is the $R(i,j)^{th}$ neighbor of the point i. After computing the rank Matrix of the high dimensional space and the corresponding low dimensional space for each row, the following correlation is computed:

$$\rho_S = 1 - \frac{6\sum_{i=1}^{T}(r_X(i) - r_Y(i))^2}{T^3 - T}$$

T is the total number of possible ranks (1 to N = N) and r_X the row from the high dimensional rank matrix, and r_Y the row from the low dimensional matrix. After this, a N× 1 vector is obtained, and Spearman's rho rank correlation is computed by taking the mean value.

7.2.5 Konig's Measure

Similarly, Konig's measure is also a tool to measure the preservation of the topology based on the rank order of the high and low dimensional space. However, this method has two parameters, k1 and k2, with k2 > k1. These variables define what is considered a good match between two given ranks of a point. Let's define $\overline{K_1}$ as the k1 nearest neighbors of a point and $\overline{K_2}$ the k2 nearest neighbors of a point not being in $\overline{K_1}$. Moreover, it is necessary to compute the two same rank matrices R defined above where R_X is the high dimensional rank matrix and R_Y is the low dimensional rank matrix. Then a matrix KM is constructed where KM(i,j) measure

for the ith point and the jth neighbor the following value:

$$KM(i,j) = \begin{cases} 3, & \text{if } \mathrm{R}_X(i,j) = R_Y(i,j), \\ 2, & \text{if } \mathrm{R}_X(i,j) = R_Y(i,l), l \in \overline{K_1}, j \neq l, \\ 1, & \text{if } \mathrm{R}_X(i,j) = R_Y(i,t), t \in \overline{K_2}, j \neq t, \\ 0, & \text{otherwise} \end{cases} \tag{30}$$

Finally the Konig measure is computed as follow:

$$km = \frac{1}{3k_{k1} \times m} \sum_{i=1}^{m} \sum_{j=1}^{k_1} KM(i,j) \tag{31}$$

where km $\in$ [0,1], 1 indicates perfect neighborhood preservation and 0 a poor one.

7.2.6 Mean relative rank error

Mean relative error (MRE) can be used on the ranks matrix (MRRE) to evaluate how the ranks are preserved after the dimensionality reduction. These methods use K as a parameter where $N_K(x_i)$ denotes the set of order numbers of K neighbors of x_i. Given the two same rank matrix R_X and R_Y, MRRE is defined as:

$$MRRE_X = \frac{1}{C} \sum_{i=1}^{N} \sum_{j \in N_K(x_i)} \frac{|R_X(i,j) - R_Y(i,j)|}{R_X(i,j)}$$

$$MRRE_Y = \frac{1}{C} \sum_{i=1}^{N} \sum_{j \in N_K(y_i)} \frac{|R_X(i,j) - R_Y(i,j)|}{R_Y(i,j)}$$

where the major difference is the normalization factor,

finally, C is given by:

$$C = N \sum_{i=1}^{K} \frac{|2i - N - 1|}{i}$$

Both MRRE values are between 0 and 1 and tend to be 0 if the K nearest neighbors of each data appear in the same order in both dimensional spaces. Thus there is good preservation of the topology with small values of MRRE.

7.3 Methodology

7.3.1 Experiment 0

In order to validate the implementation, a quick comparison between the official and implemented algorithms will be made. The reduction will be performed on the 3 datasets previously defined from 3D to 2D, and the parameters will take the same default value fixed by the official algorithm.

7.3.2 Experiment 1

First, the stability of UMAP and t-SNE is analyzed by adding white noise as a 4th dimension with different standard deviations to the Swiss Roll dataset. Then a reduction to 3D and 2D is performed to observe the impact of different parameters on the loss function, the Shepard diagram, and the plot. These parameters will be the perplexity (t-SNE) or k (UMAP), the strength of the added noise, and the initialization method used to embed the first Y.

7.3.3 Experiment 2

The second analysis will be around the stability between multiple configurations of a reduction technique to observe which parameter values give the best distance conservation. Thus for each technique, multiple configurations are run on the 3 datasets where the perplexity (t-SNE) or k (UMAP) and the initializations methods are varied. Then the embedding is evaluated using the configuration score defined above.

7.3.4 Experiment 3

The last experiment consists in performing 2D reduction with a specific configuration. This test can be divided into two parts. The first one will be about performing the reduction to the swiss roll dataset and varying the perplexity value. Then the reduction is performed on clustered data and non-clustered data. In this second part, a perplexity/k value is fixed, but the number of features varies to observe dimensionality's impact. The following evaluation measure is used to make our analysis:

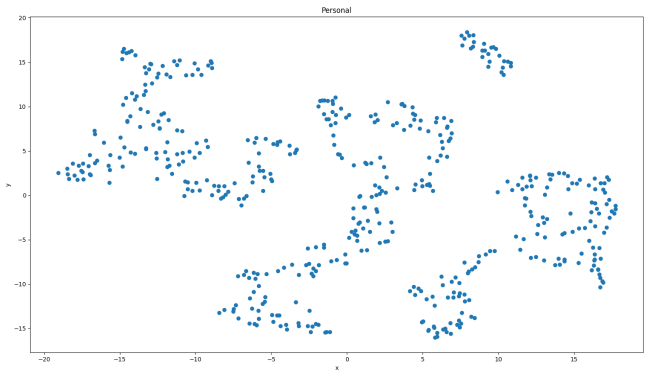
- Stress
- Spearman rho rank
- Konig's Measure
- Mean relative rank error

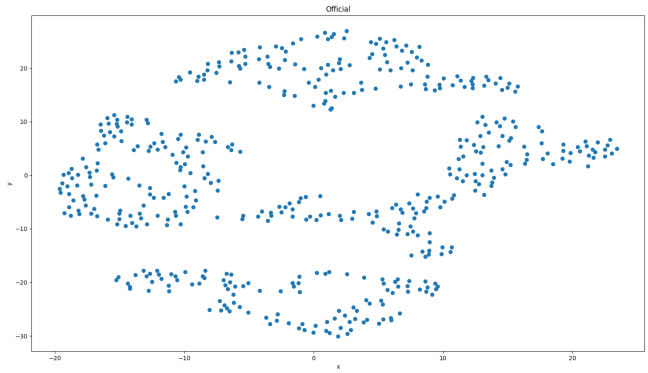
7.4 Results and analysis

7.4.1 Experiment 0

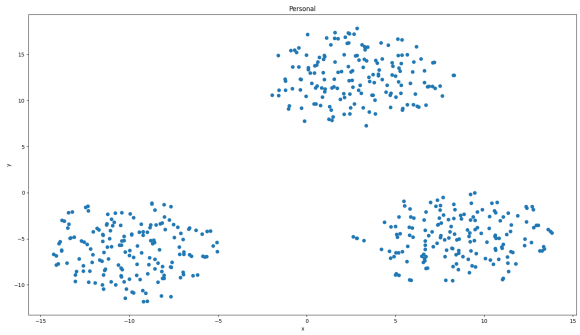
t-SNE:

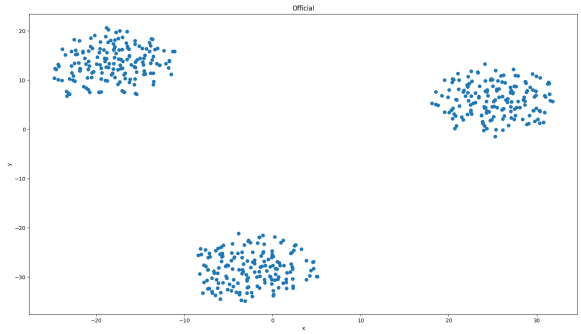
Swiss-roll



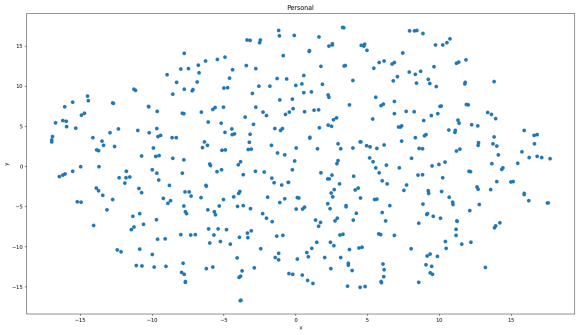


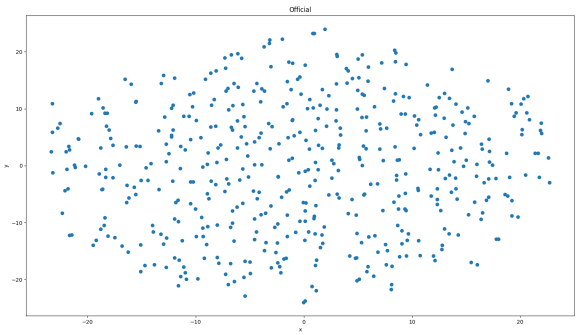
Clustered





Non clustered

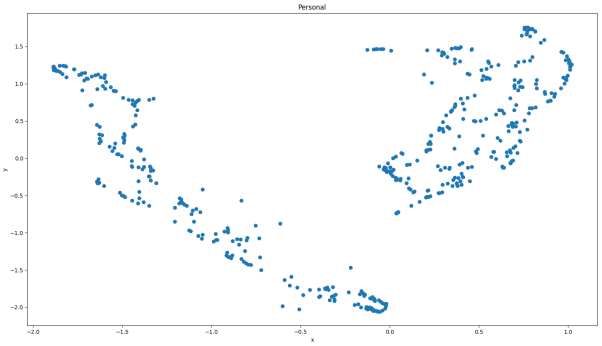


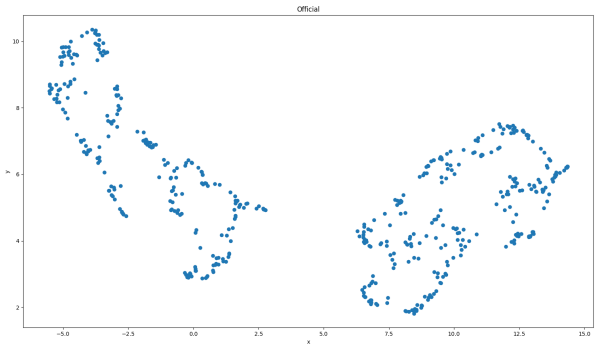


Concerning the implementation of t-SNE, the result on the clustered and non-clustered data seems similar. There is more space between the clusters because the official t-SNE use early exaggeration which controls how much space will be between the cluster after the reduction by multiplying $p_{i,j}$ in early stages. The reduction on the swiss roll dataset is less similar, the cluster seems to be more packed in the implemented versions. It can be explained by many factors, the official algorithm can stop the optimization when it computes a gradient norm below a default value of 1e-7. Moreover, the random initialization is different for both algorithms, so the result will not be the same. Globally the results seem acceptable.

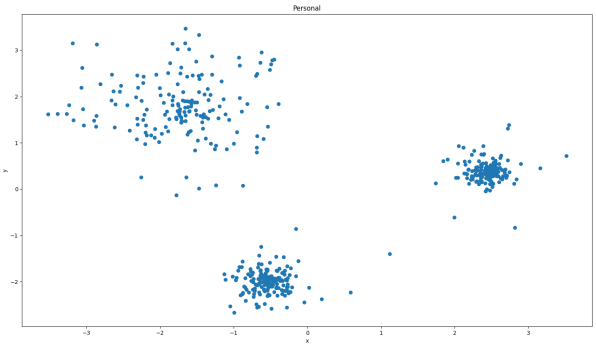
UMAP

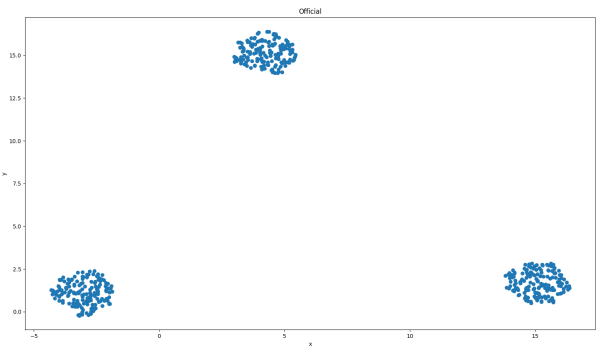
Swiss-roll



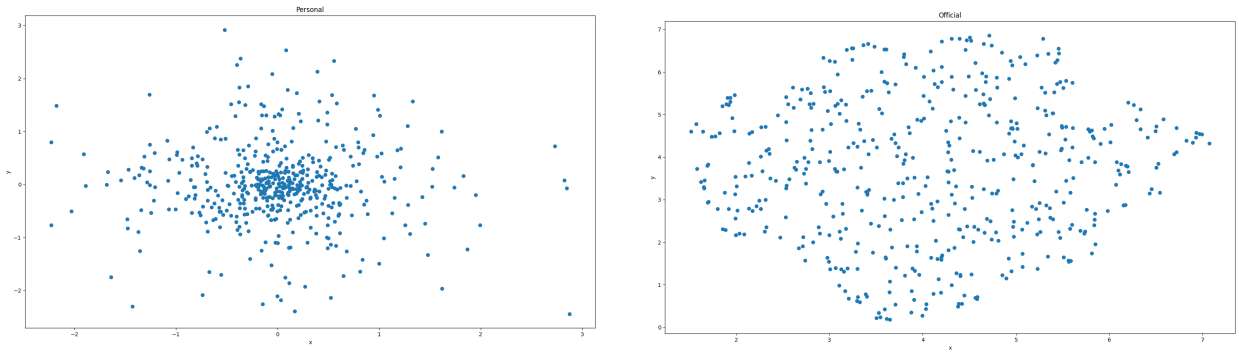


Clustered





Non clustered



Concerning UMAP, there is a less random factor as the Laplacian Eigenmaps initialization is used. However, the optimization of the embedding is simplified in the implemented algorithm. As discussed, the original UMAP algorithm uses a sampling approach to optimize the embedding. Thus a different balance between attraction and repulsive force is observed. These differences show a lack of repulsive forces (or too many attractions) for the non-clustered dataset and a lack of attraction force (or too many repulsions) for the clustered dataset. However, these forces are not entirely different and still allow the analysis of the impact of the parameters on the algorithm. Lastly, UMAP uses stochastic gradient descent, which impacts the behavior of the optimization.

7.4.2 Experiment 1

This experiment generated more than a thousand images available on **github**, in order to not overload this paper with images, only some of the 2D reductions where two of these following three parameters are fixed will be showed:

1. The strenght of the added noise
2. The initialization method
3. Perplexity for t-SNE and k for UMAP

For each configuration, the evolution of the loss function, the Shepard diagram, the obtained plot, and a comparison with the corresponding 3D reduction is observed.

t-SNE:

Strength of the noise:

Let's begin with t-SNE by varying the standard deviation of our added noise is 1,5,10,20:

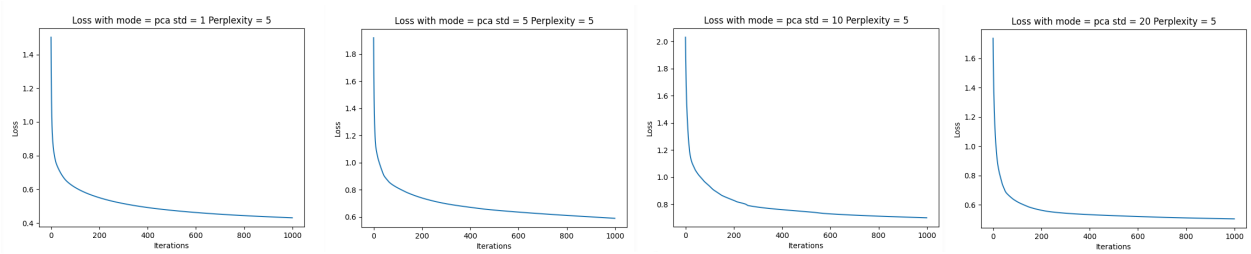


Figure 9: Evolution of loss function

The loss function seems to have the same behavior regarding the noise, except that the initial loss value seems to have a better chance of being high.

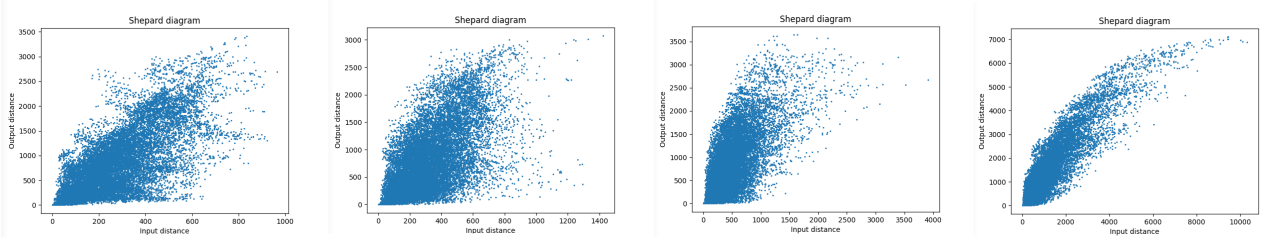


Figure 10: Evolution of shepard diagram

Adding more substantial noise increases the values of our input distances and helps the algorithm to map the distances as the values are more on the diagonal line. It can be explained by the fact that there are fewer overlapping distances.

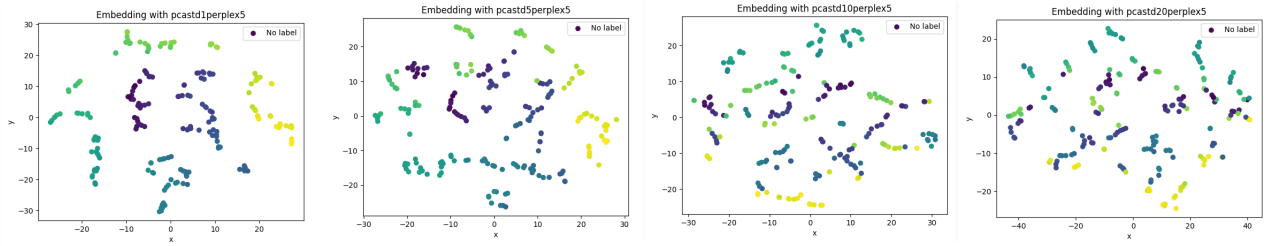


Figure 11: Visualization

The more the noise strengthens, the more there are "mini-clusters" with fewer points. However, resistance to reasonable noise can be observed as the two first plots show good categorization.

Initialization methods:

Now let us observe the impact of the initialization methods on the reduction.

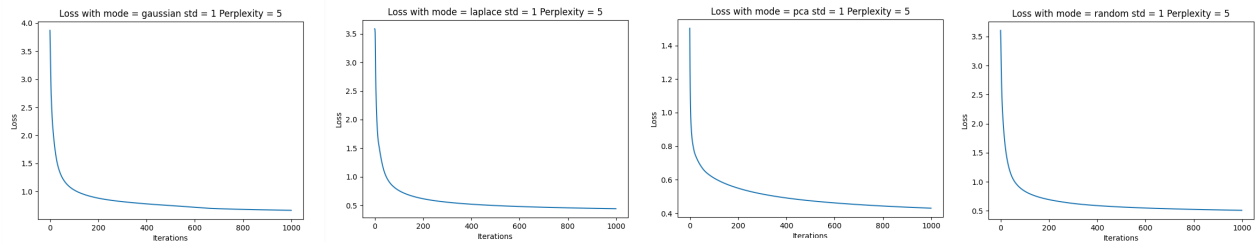


Figure 12: Evolution of loss function

Similarly to the change in the noise, the loss function does not have a difference in the behavior but starts with a lower value under PCA.

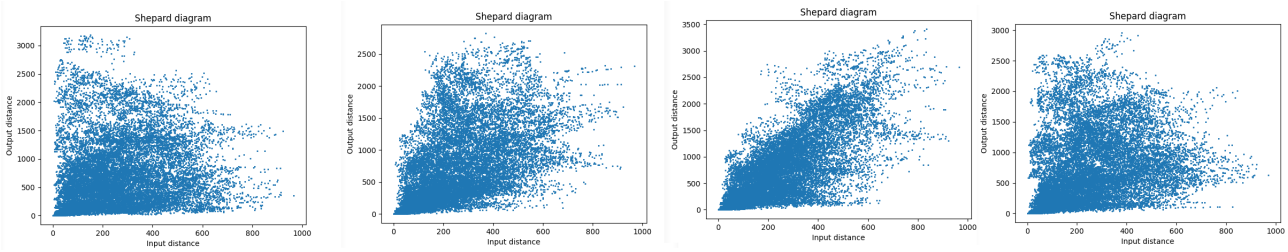


Figure 13: Evolution of shepard diagram

There is a better diagonalization of the Shepard diagram under PCA.

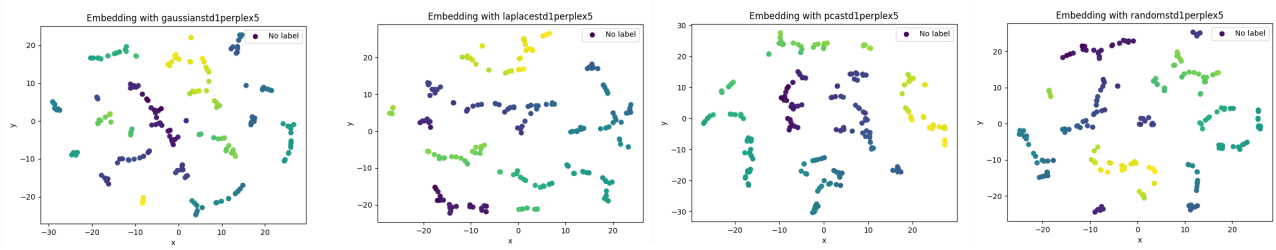


Figure 14: Visualization

However, the classification seems good for all configurations and shows slight differences. Some clusters are similar among the plots where only small rotations differ. (Example: Yellow cluster between random and PCA)

Perplexity:

Regarding the perplexity,

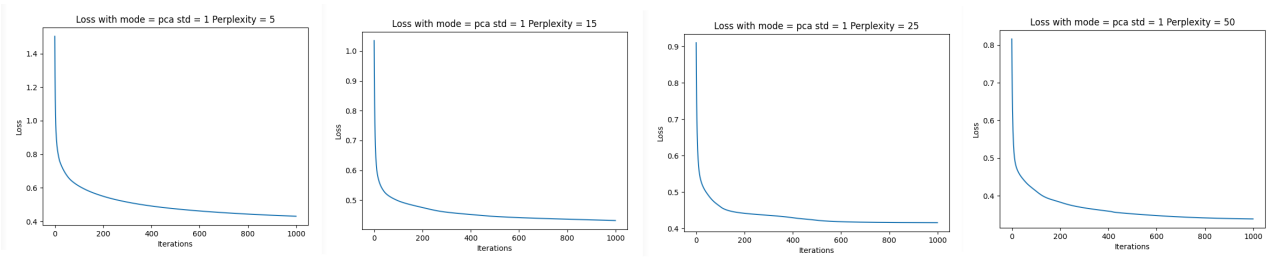


Figure 15: Evolution of loss function

once again, it does not impact global behavior and the convergence's speed. However, the more the perplexity is high, the more the initial value of the loss function is small, which probably depends on the structure of the high dimensional data. Therefore, having a high perplexity should improve the global preservation of the method so it can produce a smaller loss value at the first iteration.

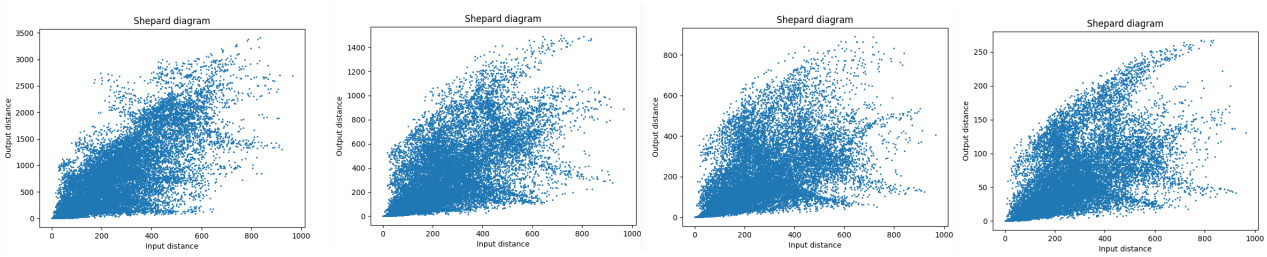


Figure 16: Evolution of shepard diagram

Clearly, there is a very similar mapping in each plot with a diminution of the output distance as the perplexity increase

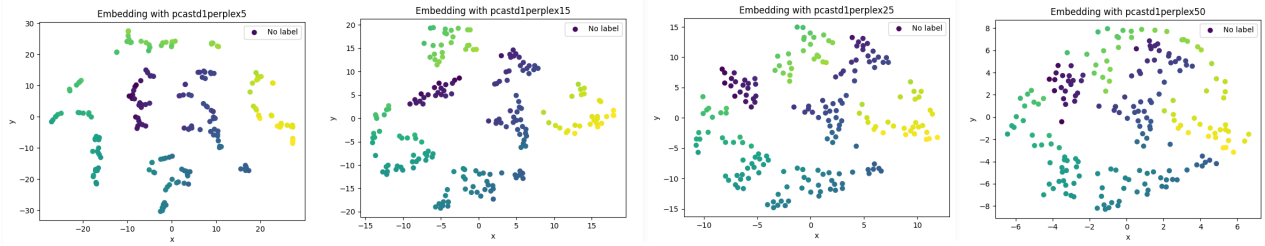


Figure 17: Visualization

The plot of the reduction clarifies this phenomenon. The data are contained in a smaller range and seem to construct a less compressed cluster. In other words, it constructs a more distant cluster with small perplexity, so the data are contained in a bigger range.

Now let us check the corresponding 3D reduction with a perplexity of 25.

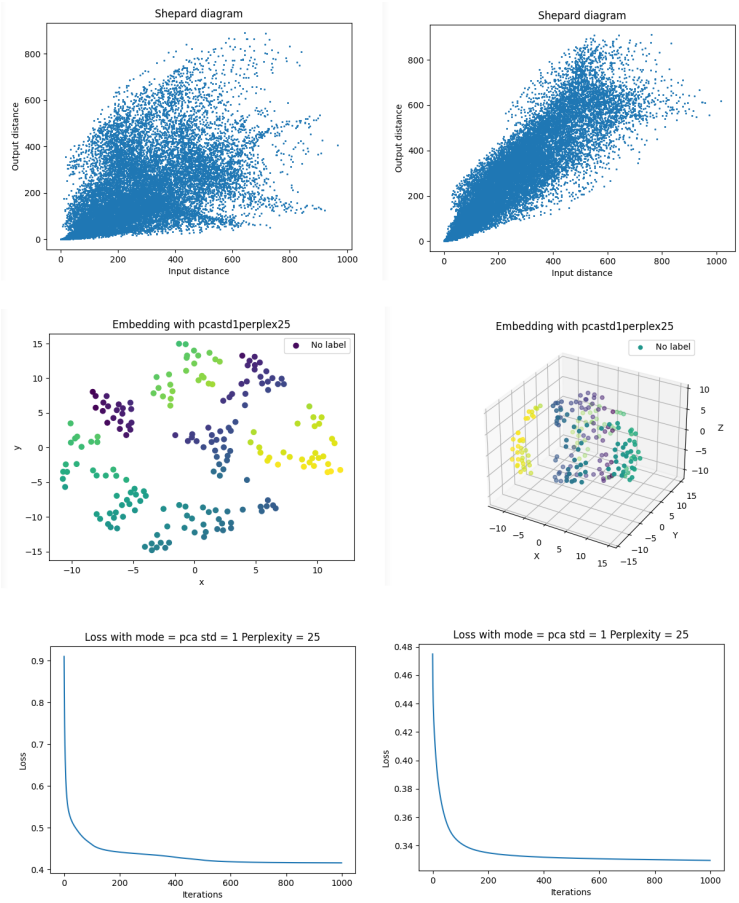


Figure 18: 2D vs 3D

As regards the mapping of the distance, there is a better repartition of the 3D case, which can be explained by the fact that it is easier to construct a "one-to-one" mapping" in a bigger space. Moreover, the data lie in a smaller range on axis z and x as it can also use the y-axis to place the data points. In other words, there is more possibility for points to have the same distance as they can be placed in one more direction. In addition, a smaller loss value is observed, which makes sense because the 2D space is more restrictive than the 3D space. Thus the dissimilarities with the 4D distribution are bigger.

UMAP:

Strength of the noise:

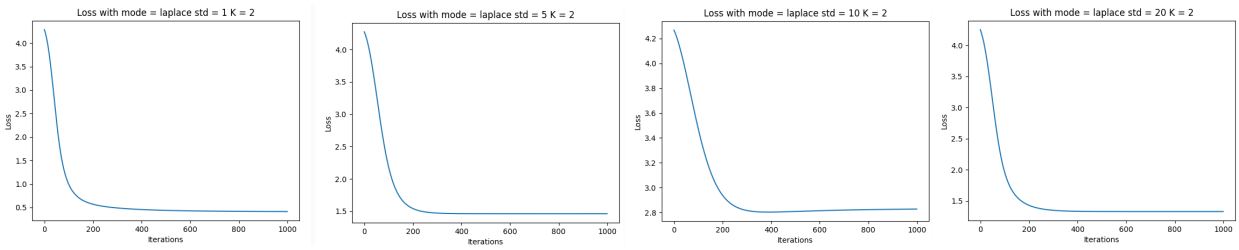


Figure 19: Evolution of loss function

Except with a standard deviation of 20, the loss seems to increase with the noise's strength, which can be explained by the fact that more volatile high dimensional data points can be harder to place in a 2D space.

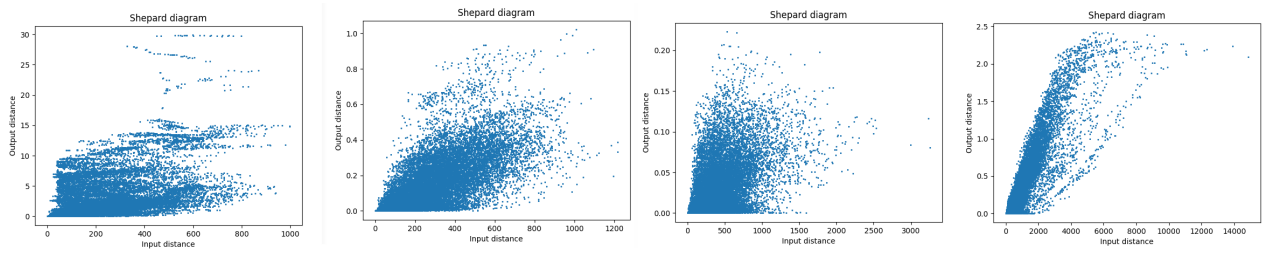


Figure 20: Evolution of shepard diagram

Similarly to t-SNE, as the noise increase, the input distances increase, and more distance lie in a diagonal. However, there is a pretty bad mapping for the first case, which can be explained by the k=2 parameters that will be explored later.

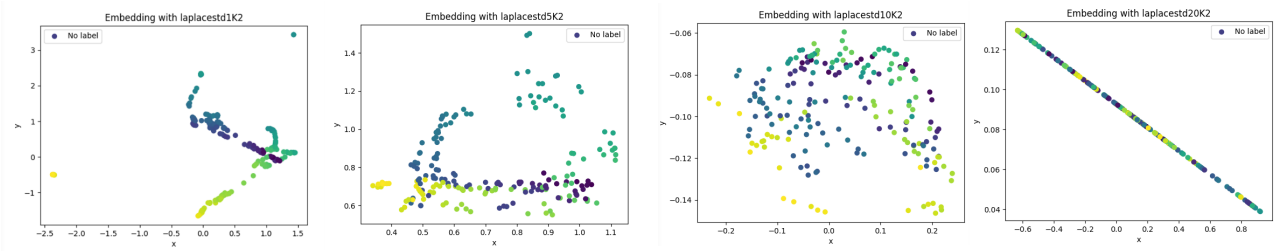


Figure 21: Visualization

UMAP seems to be more vulnerable to noise, and the plot completely changes with the increase of noise. Except for the two outliers, the first graphic made an acceptable mapping with some overlapping clusters. In the same way, the second and third ones are in a smaller range and still have overlaps. However, the last plot shows a 1D mapping. The algorithm depends on other parameters as the min_dist mentioned in the previous section. This parameter specifies the expected distance between the nearest neighbors in the reduced space. Adding noise can modify the high dimensional data such that the current min_dist value (0.1) becomes too small. Moreover, the chosen k value can impact the computed σ used in the high dimensional as a normalization factor defining the Riemannian metric local to a point. Depending on the structure of our initial data, it is possible that a bad choice of k can produce the wrong high dimensional probability and impact the reduction in a bad way.

Initialization methods:

In contrast to t-SNE, the behavior of the loss function changes according to the initialization method:

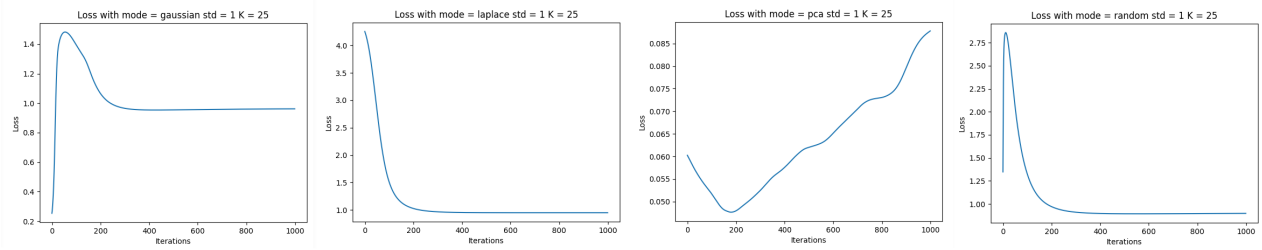


Figure 22: Evolution of loss function

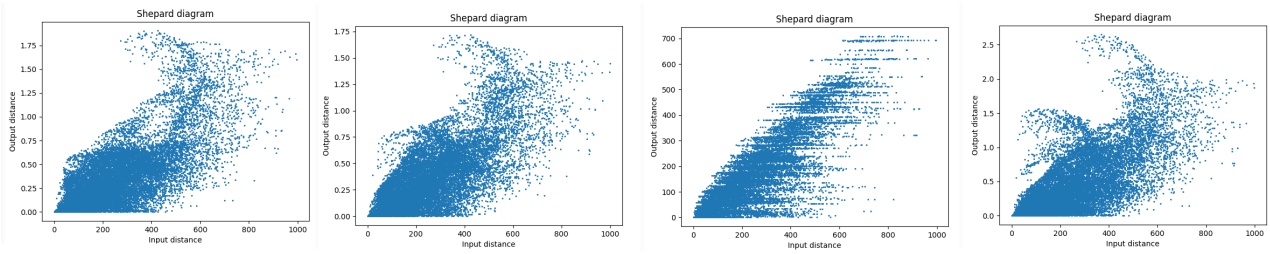


Figure 23: Evolution of shepard diagram

Oppositely the Shepard diagram appears to be less sensitive to the initial embedding. However, almost the same graphics for all methods except PCA have been obtained.

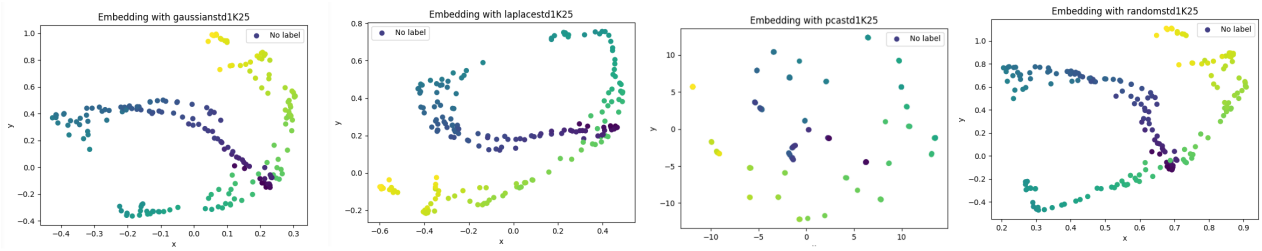


Figure 24: Visualization

Gaussian and random initialization are almost identical and slightly differ from Laplace, a simple rotation and flip of the data points. However, the loss function shows a problem with PCA as the reduction is unsuccessful and shows many points at the same position and sometimes too far away from the points with the same label.

Parameter k:

Now let us take a small noise and the random initialization to analyze the parameter k.

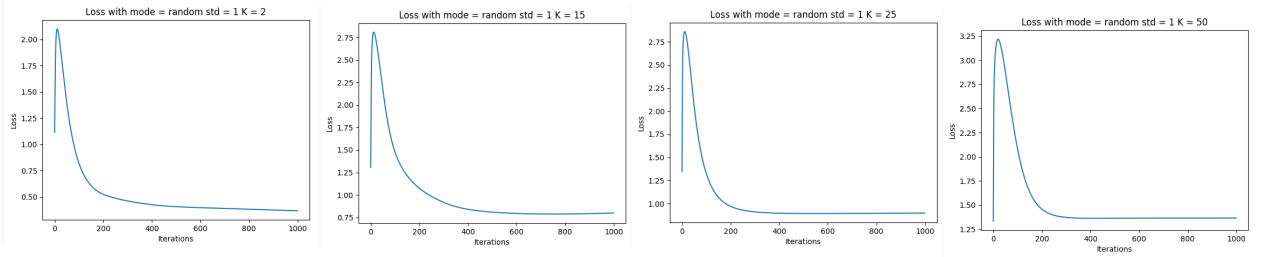


Figure 25: Evolution of loss function

There is almost no difference between the loss function except the initial value of the loss function and the peak where the loss function starts to converge to a minimum.

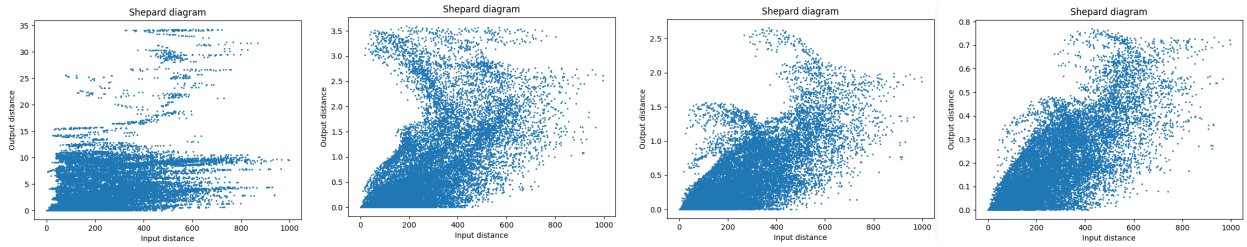


Figure 26: Evolution of shepard diagram

Regarding the Shepard diagram, there are similar plots for the last two values of k and a pretty bad mapping for the first diagram, even though the corresponding loss function is smaller. In addition, the output distances are smaller as k increases.

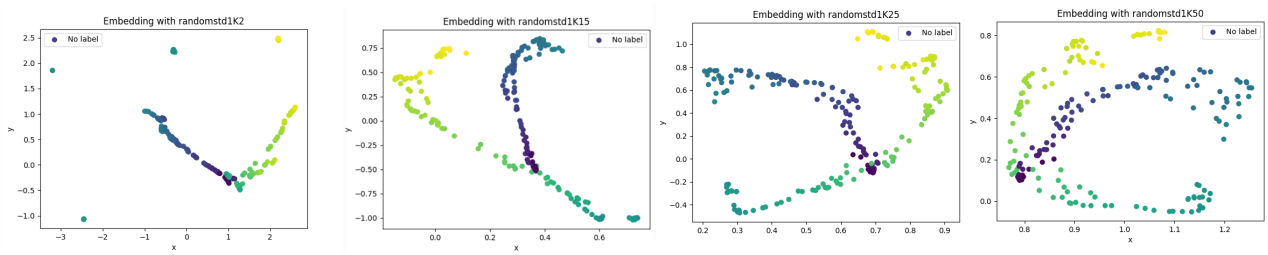


Figure 27: Visualization

Again except for the first mapping, which looks much more compressed than the others because of some outliers, the three others are very similar. While the two last graphics differs from a 180° flip, the second one shows more differences with a very similar global structure (almost a 90° rotation with the fourth graphic). Moreover, very similarly to t-SNE, the more k is small, the more the points are compact.

Lastly, let us visualize and compare with a 3D reduction.

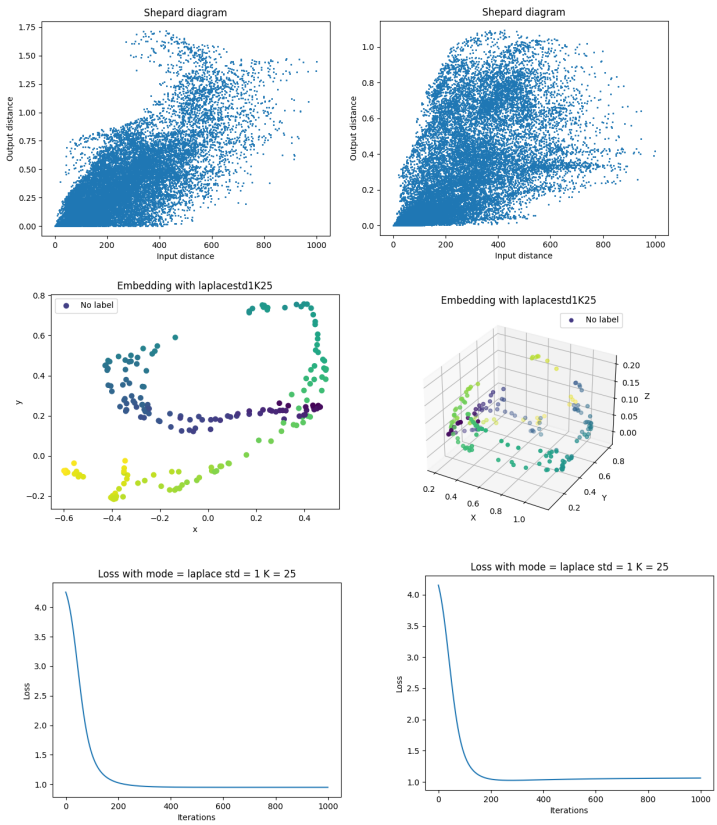


Figure 28: 2D vs 3D

The Shepard diagram is different, and the output distance is smaller in the 3D reduction, which makes sense as there is one more dimension to place our points closer. However, the distances do not lie more on the diagonal because the output distance is smaller. Moreover, the 3D graphic does not show a perfect reduction, but clearly, the different clusters are distinguishable. In addition, it does not appear that quality has been lost on the 2D reduction as there is almost the same loss function.

7.4.3 Experiment 2

As discussed in the methodology section, the idea here is to compare multiple configurations of a reduction technique on the conservation of the distance. The objective is to observe which parameters favor good preservation on each dataset.

t-SNE

Each point represents one of the 16 tested configurations. Four different value of perplexity has been tested with each initialization method.

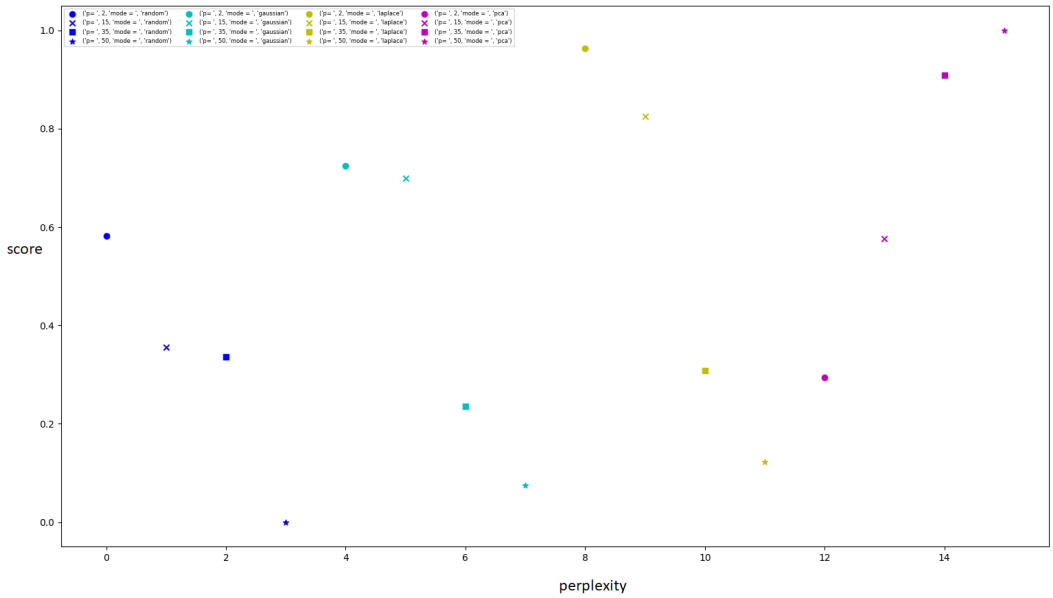


Figure 29: Configuration score - Swiss Roll - t-SNE

The swiss roll dataset obtains better conservation with PCA and Laplace. However, there is better preservation with a smaller perplexity value for Laplace than PCA. The larger the perplexity, the more it will retain non-local information in the reduction, so it should have better global preservation of distance with bigger perplexity. However, this phenomenon is not always observed as only some pivot points are taken to make this analysis, so a "global" score is not measured. Furthermore, instead of thinking that perplexity is how much global structure is preserved, it is better to say that perplexity should be seen as a parameter that balances global and local preservation of the data.

Clustered

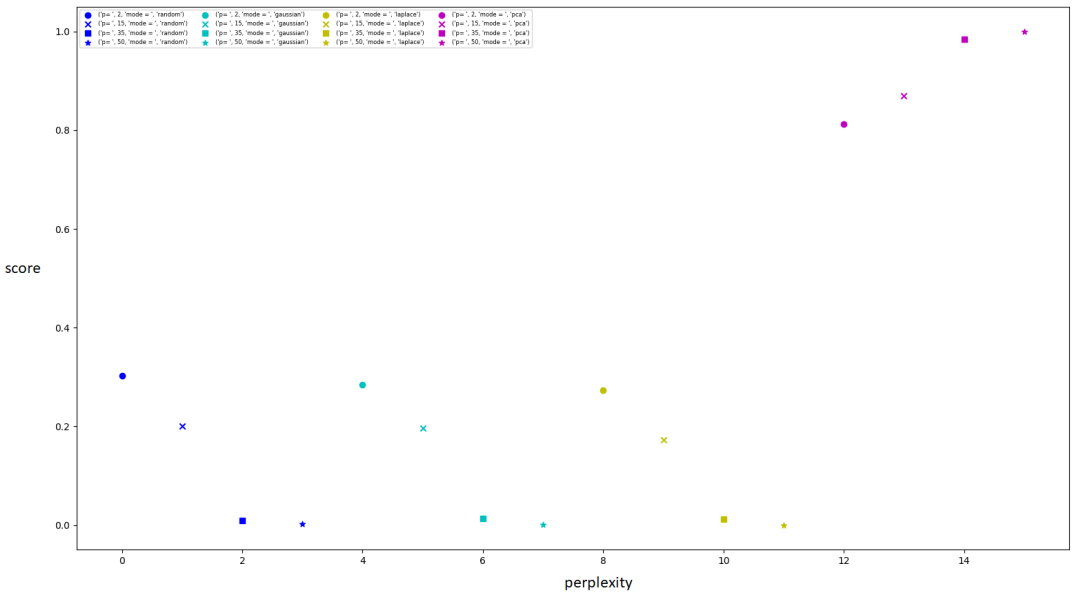


Figure 30: Configuration score - Cluster - t-SNE

Here PCA completely outperforms the other methods, which seem to converge exactly on the same result. Again PCA shows better preservation with a higher perplexity value in contrast to other methods.

Non clustered

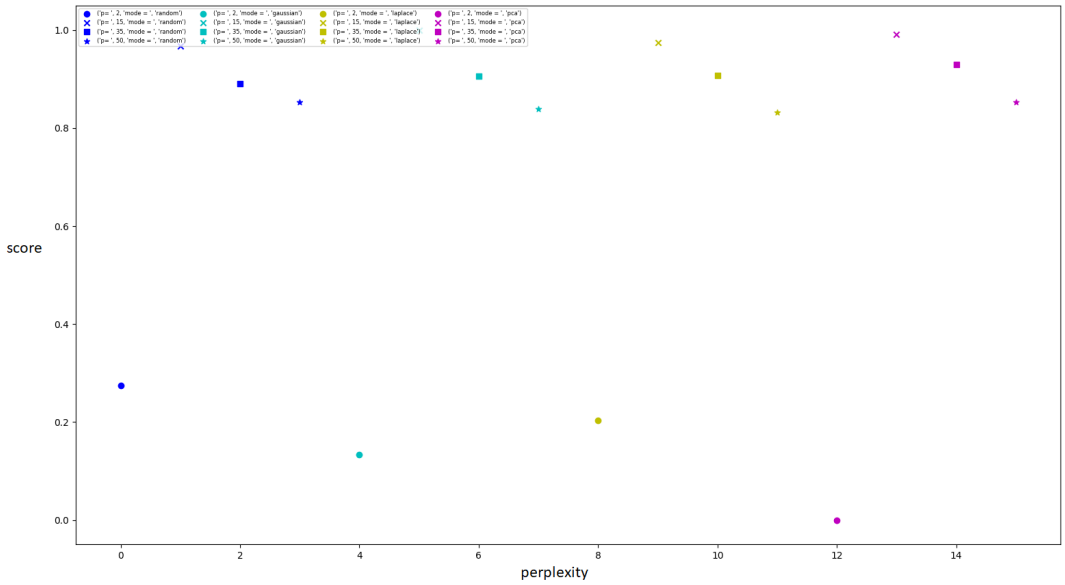


Figure 31: Configuration score - Non Cluster - t-SNE

This time, the configuration score almost completely depends on the perplexity value disregarding the initialization.

UMAP

Swiss roll and clustered

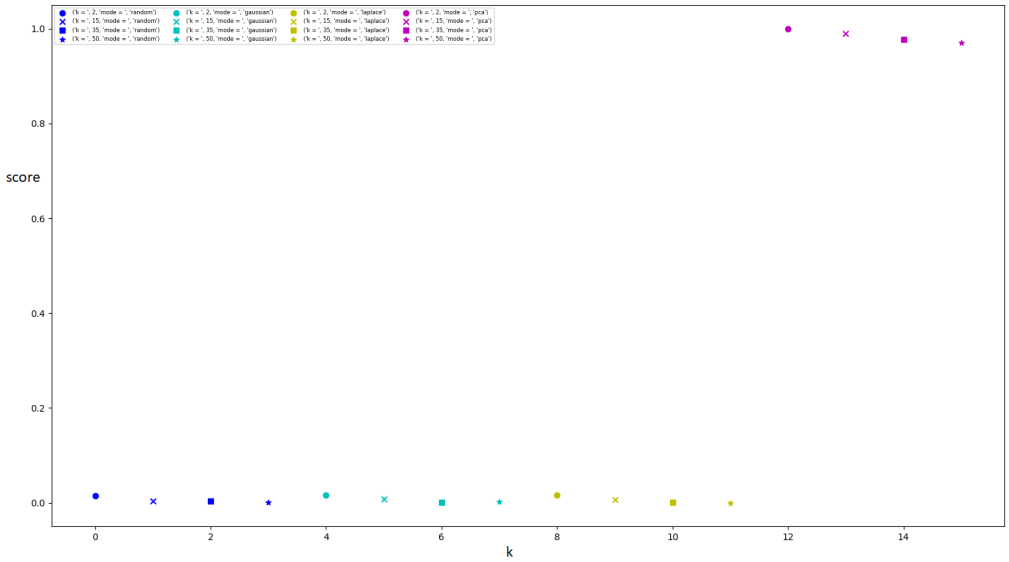


Figure 32: Configuration score - Swiss Roll - UMAP

UMAP on Swiss-roll and clustered dataset showed the same result, and against all odds, PCA had the best score by far. As discussed earlier, the algorithm had difficulties converging with PCA, and the results are not convincing. It can be explained by observing figure 23. All the plots except PCA are on an entirely new scale. The UMAP algorithm map the high dimensional data according to the parameters min_dist, which by default is set to 0.1 and generates packed cluster after the reduction. Thus during the optimization of the method using initial PCA embedding, the points are barely updated as smaller learning rate values are tested to make the

loss function converge. In other words, the final mapping did not have "time" to converge, so it is still on the same scale as the original data. Although this example shows us a weakness in our way to evaluate the quality of reduction, if all the configurations are not at the same scale, there is a solid chance of obtaining the wrong result because the comparison of distance strongly depends on the intensity of our data points. Thus using a relative distance could be more adapted. For instance, it is possible to normalize the two distance vectors defined in section 7.2.2 by the sum of their elements to obtain a value between 0 and 1 for both vectors.

Non clustered

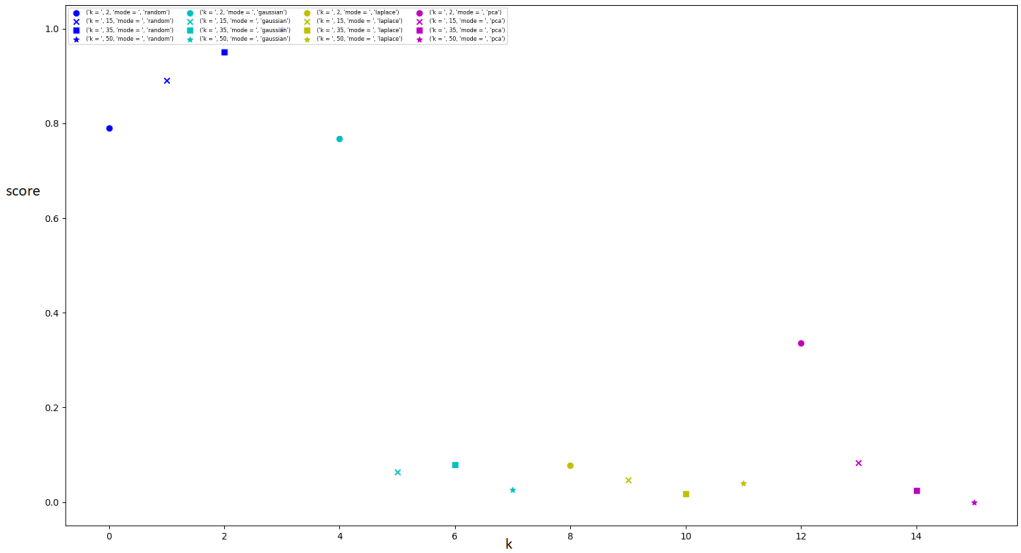


Figure 33: Configuration score - Non Cluster - UMAP

Here the random Gaussian initialization showed the best result. Again the optimization is less dependent on the initialization when the data is uniformly distributed.

7.4.4 Experiment 3

In this last experiment the focus is on diverse evaluation metric defined in section 7.3.3.

First, the global evaluation measure will be analyzed. Let us visualize the different values of stress and Spearman's rho rank for different perplexity and k:

Swiss-roll

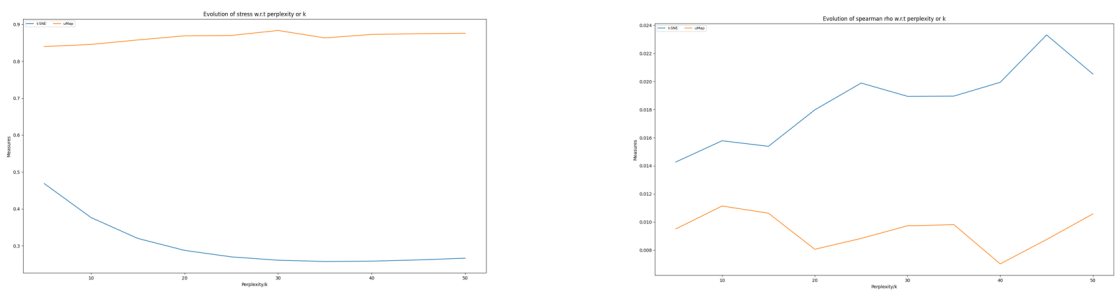


Figure 34: Stress and Spearman's Rho - Swiss Roll

For the Swiss-roll dataset, the stress is better with t-SNE. Moreover, the perplexity and k values do not seem to impact the stress as only a slight decrease with the first perplexity values is observed. Regarding spearman rho, both methods show the mitigated result as the score is defined between -1 and 1. t-SNE and UMAP do not ideally conserve the data points' neighborhood ranking. On the other hand, t-SNE fails in both preservation

and shows poor results.

Clustered

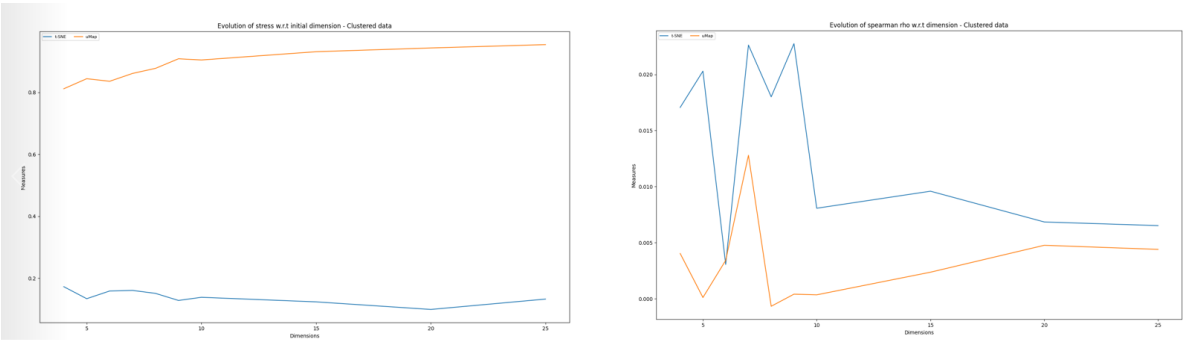


Figure 35: Stress and Spearman's Rho - Cluster

With clustered data, there are no important differences regarding the different dimensions. However, the stress value for UMAP slightly increases, and spearman's rho converges almost at the same value for both methods. Parameters k and perplexity should favor the conservation of global structure. However, the impact of stress and spearman's rho is minor.

Non clustered

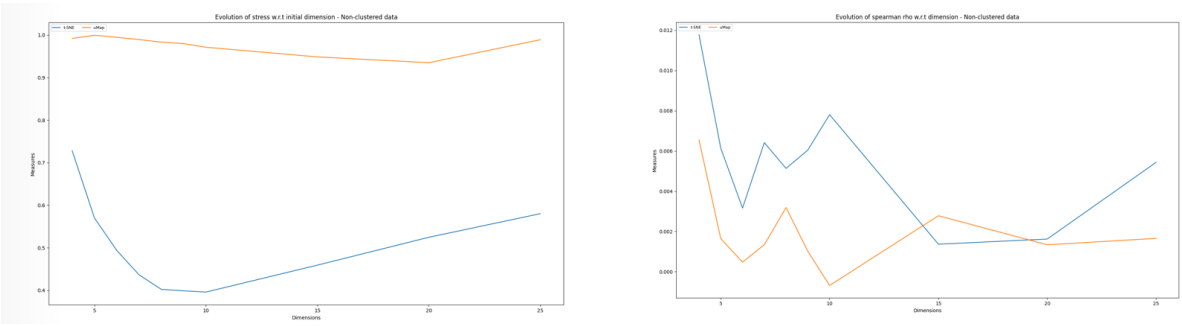


Figure 36: Stress and Spearman's Rho - Non Cluster

Unfortunately, significant differences have not been seen with non-clustered data. However, a similar variation of spearman's rho is observed for both datasets.

Concerning the first plot, the effect of varying k1 is observed. This parameter defines what a very close neighborhood is. Obviously, if k2 is fixed and k1 is increased, more points are attributed to our reduction, but the normalization factor defined in equation 30 has a more significant growth. Thus increasing k1 without modifying k2 gives a lower value of könig measure. Concerning the second plot, the same reasoning is followed, if only k2 increases, more points can be attributed, and since equation 30 does not depend on k2, the score is higher. Regarding MRRE X or Y, the more K is low, and the best is our score. It is explained by the fact that even if the normalization factor C is more significant as K grows, there is less element to sum because our neighborhood N_K is much smaller.

t-SNE

Swiss-roll

Now let us evaluate the topology preservation of t-SNE with Konig measure and mean relative rank error.

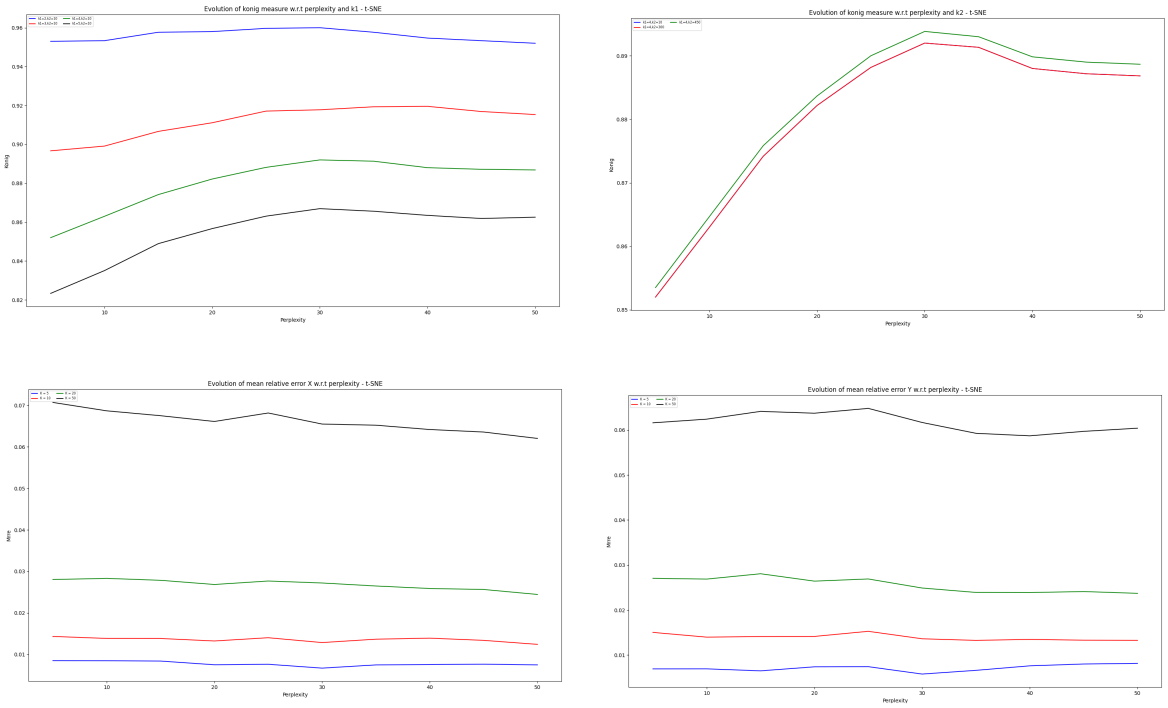


Figure 37: Konig, Mrre - Swiss Roll - t-SNE

Even the worst scores are acceptable, and globally, there is a better score with greater perplexity values. According to the previous analysis, there is a better result with higher perplexity values (25 to 30), indicating better learning of the topology with these perplexities. Moreover, the score is much more sensitive to perplexity when k2 is big.

Concerning MRRE there is no significant difference between the two measures, but excellent scores are observed, which means that the rank has good preservation within a defined neighborhood. However, there is no direct correlation with the evolution of perplexity. Theoretically, it should have a more accurate conservation of neighborhoods with small perplexity.

Clustered

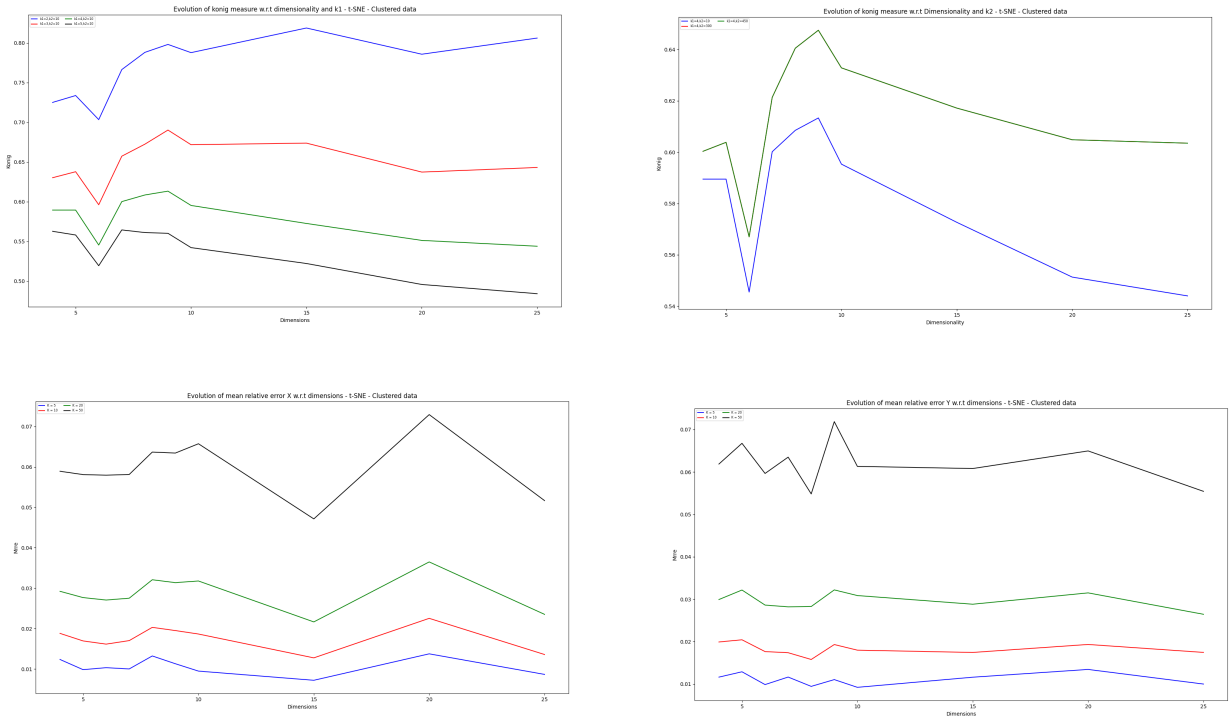


Figure 38: Konig and Mrre - Cluster - t-SNE

In comparison to the Swiss-Roll dataset, lower values of könig measure are observed, which can be hard to understand as clustered data should be "easier" to reduce. However, it can be explained by the fact that the reduction is more powerful because there is a bigger initial feature size. In addition, there is a correlation with dimension, but it is hard to explain why there is a better score with a size of 10 dimensions. It can be related to the structure of the high dimensional data, which has a better reduction with the chosen perplexity in this experience. Concerning the MRRE good values are obtained, but an absolute correlation with the different dimensions is not identified.

Non clustered

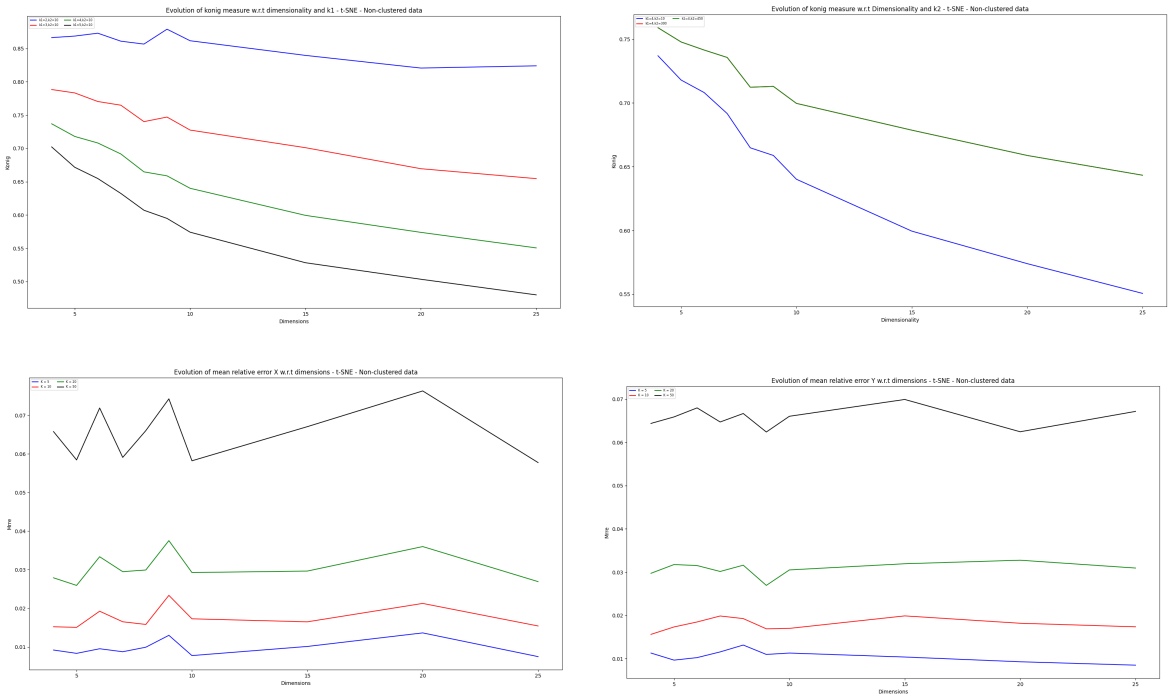


Figure 39: Konig and Mrre - Non Cluster - t-SNE

Here the könig measure shows a lower score as the dimension increase and is more sensitive when there are significant values of k2. The reduction loses in quality with more dimensions. On the other hand, MRRE does not show any difference, which means that the global rank conservation of local neighborhoods keeps good preservation.

UMAP

Swiss Roll

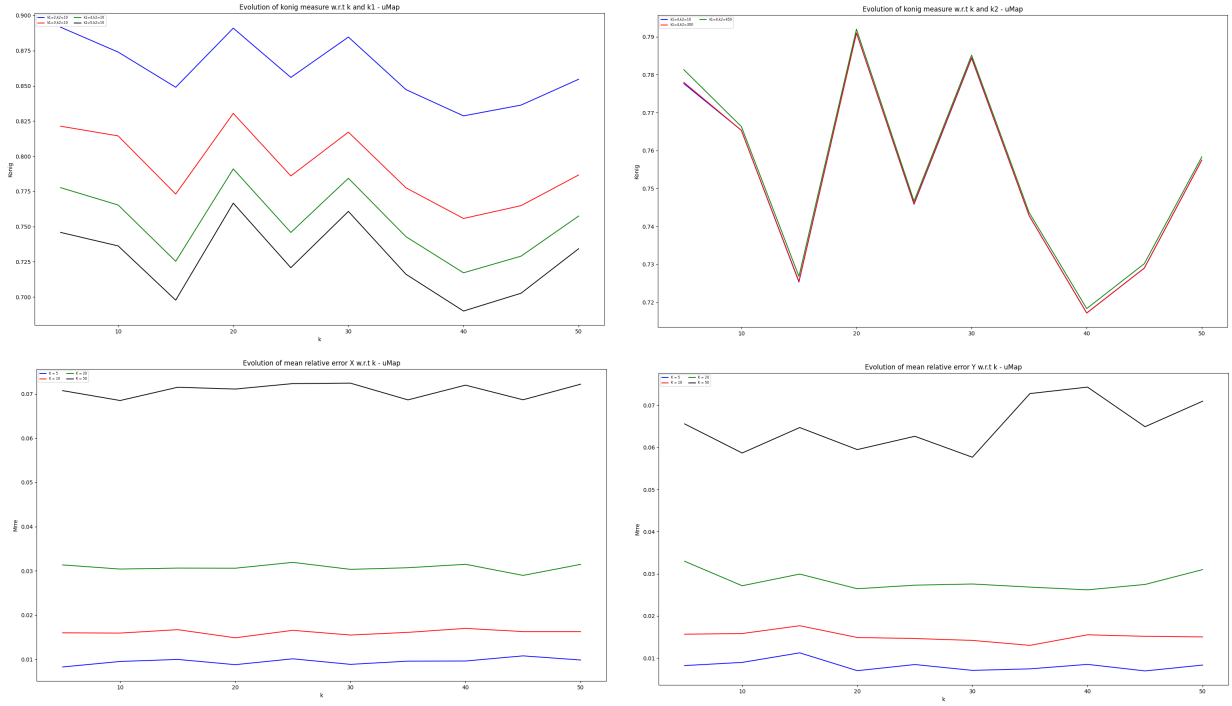


Figure 40: Konig, Mrre - Swiss Roll - UMAP

Similarly to t-SNE, k does not have a tangible impact on könig measure and the mean relative rank error. A peak at k = 20 and k = 30 is observed, which corresponds to the best local conservation.

Clustered

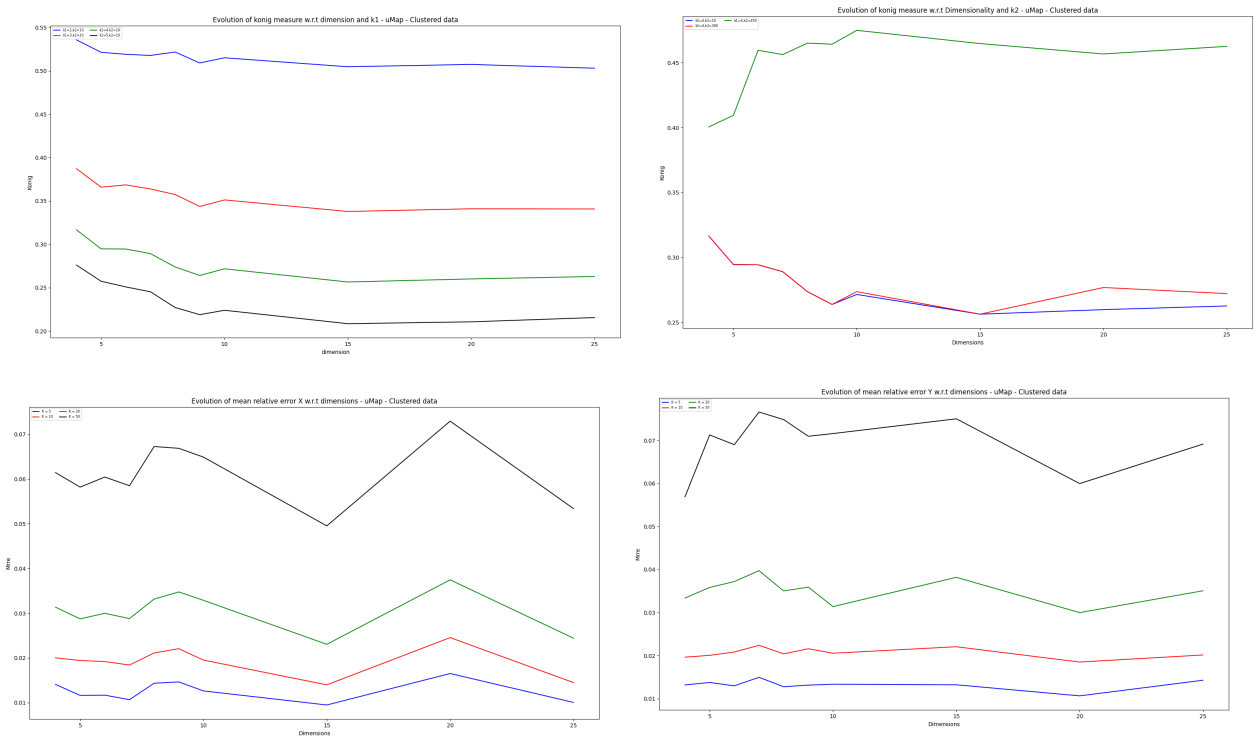


Figure 41: Konig and Mrre - Cluster - UMAP

Regarding the variation of dimensionality, clustered data appear to be very robust to the growth of dimensions. The major variations are in the first growth of sizes, then stabilization of the score is observed. However, they are much lower than the swiss-roll dataset or t-SNE, which shows poorer conservation of the local structure.

Non clustered

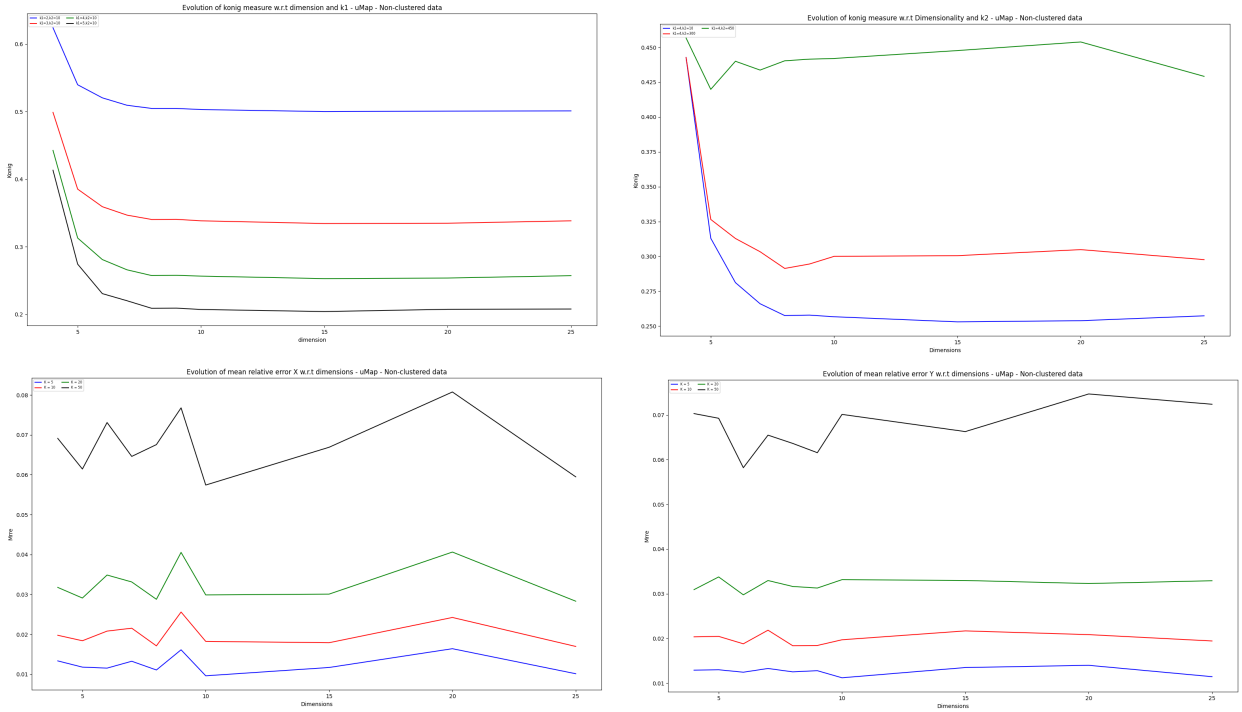


Figure 42: Konig and Mrre - Non Cluster - UMAP

Lastly, non-clustered data shows much more loss of quality in the first growth of the high-dimensional feature size. Similarly, the preservation of the neighborhood does not seem to be affected by our data structure as there are equivalent results to clustered data.

Regarding MRRE it appears not to be sensitive to perplexity,k, dimensionality, or the initial structure of the data. Therefore, good values are observed for UMAP or t-SNE regardless of the different tested parameters, which means good conservation of the rank matrix for a given neighborhood for both methods.

8 Conclusion

In data science and data visualization, it is essential to find a low-dimensional representation of the data from a high-dimensional space while preserving its initial structure. Consequently, various reduction techniques have been proposed in the last few years, each having its rational motivation. Regarding non-linear techniques, they have shown better efficiency than traditional linear techniques on real-world data, and thus they are massively used. However, not enough paper comparing multiple non-linear techniques has been published yet. Moreover, there is still no ground truth for evaluation metrics to precisely measure the quality of a reduction technique. Obviously, without a clear definition of these quality criteria, it is impossible to affirm with certitude that one method is better than another, as it also depends on the technique's chosen parameters or the structure from the original data.
In this paper, an exhaustive explanation of UMAP and t-SNE has been made to establish a comparison between these two non-linear reduction techniques. Undoubtedly it is necessary to have a theoretical comprehension to understand the motivation behind each statement. In addition, a more straightforward implementation has been proposed to have practical apprehension and total control over the various parameters. These implementations allowed us to analyze these techniques by testing and visualizing different configurations. Then, an evaluation and a comparison of these results have been made using various evaluation metrics and a specific methodology.

t-SNE showed better results for conserving the local and global structure. Also, the effect of k and perplexity on the reduction has been visualized, but their impact on evaluation metrics is much lighter as it did not have a significant impact on global and local evaluation metrics. Moreover, in our experiment, UMAP seemed less robust to the addition of noise and to the different initialization methods. Regarding the loss of information, the main loss is made during the very first dimensionality. Lastly, both methods showed good rank preservation in a close neighborhood in various configurations.
In order to deepen the research, it can be interesting to investigate different configuration scores by normalizing our distance vectors to compare relative distances instead of distances. As discussed in this work, the parameters *min_dist* and the chosen distance metric (Euclidean in this paper) also strongly impact the UMAP reduction. Thus, analyzing these parameters' impact with more detail could be another research direction.
Lastly, the code behind this project can be found on **github** where there is a complete description and documentation of the project.

References

[1] James Melville Leland McInnes, John Healy. Umap: Uniform manifold approximation and projection for dimension reduction. 2020.

[2] Geoffrey Hinton Laurens van der Maaten. Visualizing data using t-sne. 2008.

[3] Jaap van den Herik Laurens van der Maaten, Eric Postma. Dimensionality reduction: A comparative review. 2009.

[4] Víctor Robles Ernestina Menasalvas Antonio Gracia, Santiago González. A methodology to compare dimensionality reduction algorithms in terms of loss of quality. 2014.

[5] Ramzan Talib Shaeela Ayesha, Muhammad Kashif Hanif. Overview and comparative study of dimensionality reduction techniques for high dimensional data. 2020.

[6] Marius Liutvinavičius Jelena Zubova, Olga Kurasova. Dimensionality reduction methods: The comparison of speed and accuracy. 2018.

[7] Bagavathi Sivakumar Praveen Vijai. Performance comparison of feature reduction techniques in-terms of compactness, computation time and accuracy. 2018.

[8] Kriti Saroha Nitika Sharma. Study of dimension reduction methodologies in data mining. 2015.

[9] Stefan Steinerberger George C. Linderman. Clustering with t-sne, provably. 2017.

[10] Oscar Pedreira Stephane Marchand-Maillet and Edgar Chavez. Structural intrinsic dimensionality. 2021.

[11] Fakhri Karray Mark Crowley Benyamin Ghojogh, Ali Ghodsi. Uniform manifold approximation and projection (umap) and its variants: Tutorial and survey. 2021.

[12] Cynthia Rudin Yaron Shaposhnik Yingfan Wang, Haiyang Huang. Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. 2021.

[13] Stephen France. Development of an agreement metric based upon the rand index for the evaluation of dimensionality reduction techniques, with applications to mapping customer data. 2007.

[14] Gintautas Dzenyda Rasa Karbauskaite. Topology preservation measures in the visualization of manifold-type multidimensional data. 2009.

[15] Zongben Xu Deyu Meng, Yee Leung. A new quality assessment criterion for nonlinear dimensionality reduction. 2010.

[16] Lei Yang Shiyuan Wang Chaohan Xu Xiaowen Chen Ruizhi Xiang, Wencan Wang. A comparison for dimensionality reduction methods of single-cell rna-seq data. 2021.

[17] John A. Lee Francisco J. García-Fernández, Michel Verleysen and Ignacio Díaz. Stability comparison of dimensionality reduction techniques attending to data and parameter variations. 2013.