Visualising How Non-linear Dimension Reduction Warps Your Data

Jayani P.G. Lakshika
Econometrics & Business Statistics, Monash University
and
Dianne Cook
Econometrics & Business Statistics, Monash University
and
Paul Harrison
MGBP, BDInstitute, Monash University
and
Michael Lydeamore
Econometrics & Business Statistics, Monash University
and
Thiyanga S. Talagala
Statistics, University of Sri Jayewardenepura

March 29, 2024

Abstract

Nonlinear dimension reduction (NLDR) techniques such as tSNE, and UMAP provide a low-dimensional representation of high-dimensional (high-D) data using non-linear transformation. The methods and parameter choices can create wildly different representations, making it difficult to decide which is best, or whether any or all are accurate or misleading. NLDR often exaggerates random patterns, sometimes due to the samples observed. But NLDR views have an important role in data analysis because, if done well, they provide a concise visual (and conceptual) summary of high-D distributions. To help evaluate the NLDR we have developed an algorithm to show the 2D NLDR model in the high-D space, viewed with a tour. One can see if the model fits everywhere or better in some subspaces, or completely mismatches the data. It is used to evaluate which 2D layout is the best representation of the high-D distribution and see how different methods may have similar summaries or quirks.

Keywords: high-dimensional data, dimension reduction, triangulation, hexagonal binning, low-dimensional manifold, manifold learning, tour, data vizualization

1 Introduction

High-dimensional (high-D) data is prevalent across various fields, such as ecology and bioinformatics (**?**), due to advancements in data collection technologies (**??**). However, visualization of high-D data introduces significant challenges, because the complexity of visualizing data beyond two dimensions (**?**). In recent years, interactive and dynamic graphics systems like **liminal** (**?**) —which employs interactive tools like brushing and linking (**?**)—and software tools such as **XGobi**, **GGobi** (**?**), **tourr** (**?**), **detourr** (**?**), and **langevitour** (**?**), involving dynamic methods like tours (**?**), have played a key role in visualizing high-D data (data-vis).

To create low-dimensional representations (typically in 2D) (m-vis) (**?**) of high-D data, it is common to apply dimension reduction (DR) techniques. Approaches for DR involve linear methods such as principal component analysis (PCA) (**?**), non-linear methods such as multi-dimensional scaling (MDS) (**?**). In the past decade, many new non-linear dimension reduction (NLDR) techniques have emerged, such as t-distributed stochastic neighbor embedding (tSNE) (**?**) and uniform manifold approximation and projection (UMAP) (**?**). NLDR techniques are the 2D models of high-D data in our context.

It is important to visualize various non-linear dimensionality reduction (NLDR) techniques for the same high-D data in order to understand and find the best representation. After doing so, the 2D models may differ considerably from each other and may also deviate from the original data structure in high-dimensional space. Therefore, visualizing the 2D model in high-D space (m-in-ds) is more useful to answer different types of questions:

- Is there a best 2D representation of high-D data or are they all providing equivalent information? Is there a best parameter choice to fit the 2D model? How does the model change when it's parameters change?
- How well the does the 2D models capture the data structure? Is the model fitting able to capture different data structure like non-linear, clustering?

If we cannot easily ask and answer these questions, our ability to understand the models is limited. To find the best 2D model and parameter choices, a better understanding of the underlying science is important.

Also, the importance of m-vis along with data-vis has been recognized and incorporated into interactive software, **liminal** (**?**). But the 2D model and high-D visualize side by side and interactive like brushing and linking connect the data in the two panels. To address this challenge, we propose a novel approach by combining the tour technique with a low-dimensional manifold. This manifold is created through the synergistic use of NLDR techniques, hexagonal binning, and triangulation. This integration facilitates a more understanding of the data structure, how well (or how poorly) NLDR techniques perform.

The outline of this paper is as follows. Section **??** provides an detailed overview of dimension reduction methods, and tour technique. Building upon this foundation, Section Section **??** describes the proposed algorithm, including its implementation details, model tuning, summaries, and a synthetic example to demonstrate its functionality. In Section

Section **??**, the algorithm's applications on various datasets, particularly in single-cell RNA-seq and handwritten digit data, are showcased. These applications demonstrate how some NLDR techniques effectively capture both local and global structures of the original data, while others may result in misleading interpretations. Finally, Section Section **??** summarizes the findings and describe the importance of the proposed approach in addressing the challenges associated with selecting the best NLDR method and parameter choices to achieve a more accurate 2D representation of high-D data.

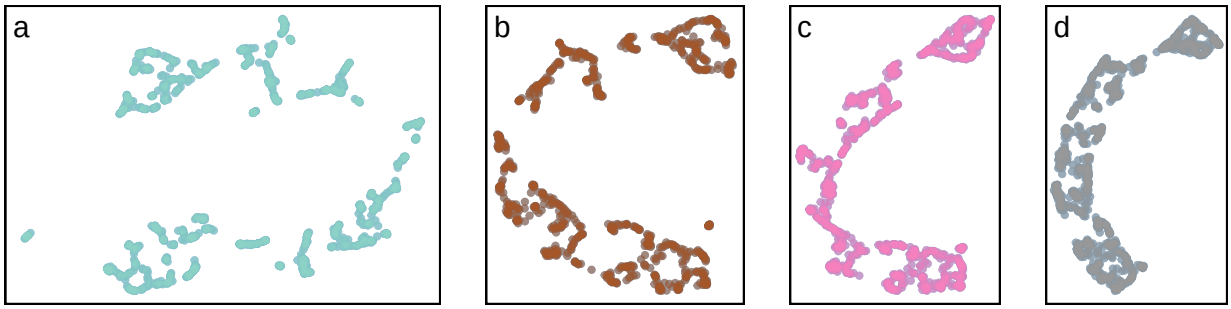


Figure 1: 2D layouts from UMAP applied for the S-curve data: (a) UMAP (n_neighbors = 7), (b) UMAP (n_neighbors = 15), (c) UMAP (n_neighbors = 32), (d) UMAP (n_neighbors = 50). Is there a best hyperparameter choice in representing UMAP or are they all providing equivalent information?

2 Background

2.1 Dimension Reduction

Consider the high-D data a rectangular matrix $X_{n\times p}$, where $X_{n\times p} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n\end{bmatrix}^\top$, with n observations in p dimensions. The objective is to discover a low-dimensional projection $Y_{n\times d} = \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_n\end{bmatrix}^\top$, represented as an $n \times d$ matrix, where $d \ll p$. The reduction process seeks to remove noise from the original data set while retaining essential information.

There are two main categories of dimension reduction techniques: linear and non-linear methods. Linear techniques involve a linear transformation of the data, with one popular example being PCA. PCA performs an eigen-decomposition of the sample covariance matrix to obtain orthogonal principal components that capture the variance of the data (**?**).

In contrast, NLDR techniques generate the low-dimensional representation Y from the high-dimensional data X, often using pre-processing techniques like k-nearest neighbors graph or kernel transformations. Multidimensional Scaling (MDS) is a class of NLDR methods that aims to construct an embedding Y in a low-dimensional space, approximating the pair-wise distances in X (**?**). Variants of MDS include non-metric scaling (**?**) and Isomap, which estimate geodesic distances to create the low-dimensional representation (**?**). Other approaches based on diffusion processes, like diffusion maps (**?**) and the PHATE (Potential of Heat-diffusion for Affinity-based Trajectory Embedding) algorithm (**?**), also fall under NLDR methods.

2.1.1 Non-linear dimension reduction techniques

NLDR techniques are crucial for analyzing and displaying high-dimensional data, where linear approaches may not adequately capture complexities in relationships between variables (**?**). One of the challenges with NLDR techniques is the selection and tuning of appropriate hyperparameters (**?**). This process involves finding the suitable combination of hyperparameters that enhances the performance of the NLDR technique, considering the characteristics of the dataset and the specific goals of the analysis.

Additionally, another challenge is lack of reverse mapping. Techniques like PCA and autoencoders (**?**) provide a way to map back from the low-dimensional space to the high-D space, facilitating data reconstruction. However, some NLDR methods, such as tSNE, don't have a specific way to reconstruct the original data from the low-dimensional space.

In this article, mainly focus on five NLDR techniques. They are tSNE, UMAP, PHATE, TriMAP (**?**), and Pairwise Controlled Manifold Approximation (PaCMAP) (**?**).

Among these, tSNE (**?**) stands out for its ability to preserve pairwise distances. By minimizing the divergence between probability distributions in both high and low-dimensional spaces, tSNE effectively uncovers intricate structures and patterns within the data. Its application is widespread, particularly in tasks requiring the visualization of clusters and local relationships. However, achieving effective results requires careful consideration of hyperparameters, such as perplexity.

UMAP (**?**) is a useful technique for simplifying data while maintaining both local and overall structures. It builds a fuzzy topological view by considering nearby data points and then optimizes a simplified version to match that view. UMAP is known for working well with different scales of relationships in data and is efficient in handling large datasets. However, it's important to choose parameters like neighbors and minimum distance carefully, as they can affect the results.

Furthermore, PHATE (**?**) is great for understanding how things develop, especially in single-cell genomics. It uses a heat diffusion process to capture relationships between data points, like points along a trajectory. While PHATE is excellent for revealing these developmental structures, it requires careful tuning of its parameters because of its specialized focus.

Additionally, TriMAP (**?**) takes a special approach by creating a triangulated graph representation of the data. This method is good at understanding both local and global structures by treating the data as a network of triangles. TriMAP is powerful in capturing complicated structures, but it's important to choose parameters carefully, like deciding how many neighbors to consider.

PaCMAP (**?**) is different because it adds supervised learning to make a 2D representation while keeping the relationships between pairs of points. It builds a graph using distances between pairs and then makes the 2D representation better using a customizable loss function. What's special about PaCMAP is that it can use class labels or extra information to guide how it makes the 2D representation. This gives users a way to change how PaCMAP works to fit their needs better.

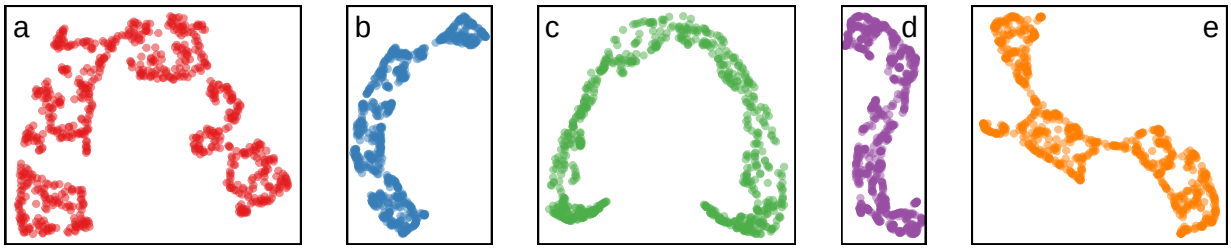


Figure 2: 2D layouts from different NLDR techniques applied the same data: (a) tSNE (perplexity = 27), (b) UMAP (n_neighbors = 50), (c) PHATE (knn = 5), (d) TriMAP (n_inliers = 5, n_outliers = 4, n_random = 3), and (e) PaCMAP (n_neighbors = 10, init = random, MN_ratio = 0.9, FP_ratio = 2). Is there a best representation of the original data or are they all providing equivalent information?

2.2 Linear overviews using tours

A tour is a powerful visualization technique used to explore the shape and global structure of high-dimensional data by generating a sequence of projections, typically into two dimensions. There are two main types of tours: the grand tour (**?**) and the guided tour (**?**). A grand tour involves randomly selecting new orthonormal bases, enabling users to understand the structure by exploring the subspace of d-dimensional projections (**?**). In contrast, a guided tour can be employed to generate a sequence of 'interesting' projections based on an index function (**?**).

The process begins with the data matrix X. It generates a sequence of $p \times d$ orthonormal projection matrices (bases) P_t, usually d is one or two dimensions. For each pair of orthonormal bases P_t and P_{t+1}, a geodesic path is interpolated to create smooth animation between projections. The resulting tour continuously visualizes the projected data $Y_t = XP_t$ as it interpolates between successive bases.

Furthermore, software like **langevitour** can visualize both types of tours, providing flexibility for exploring high-dimensional data with various objectives. In our context, use grand tour along with the model to observe how effectively the model captures the underlying structure of the data.

3 Methodology

In this paper, we introduce a novel method to determine the most effective NLDR technique and the best hyperparameter choice that provides the most useful representation of high-D data. Our approach involves dividing the high-D dataset into two parts: a training set for constructing the model and a test set for generating predictive values and residuals. Our algorithm takes a 2D embedding data as the input and generate a tour that displays the high-D wireframe to overlay the data. The flow chart of the proposed algorithm is shown in Figure **??**. The algorithm consists of two main phases: (1) generating the model in the 2D space and (2) lifting the model into high-D space. The main steps of the algorithm are described in detail in this section using UMAP 2D embedding of the S-curve dataset.

This dataset has seven dimensions, including four noise dimensions that were added to the original 3D data.

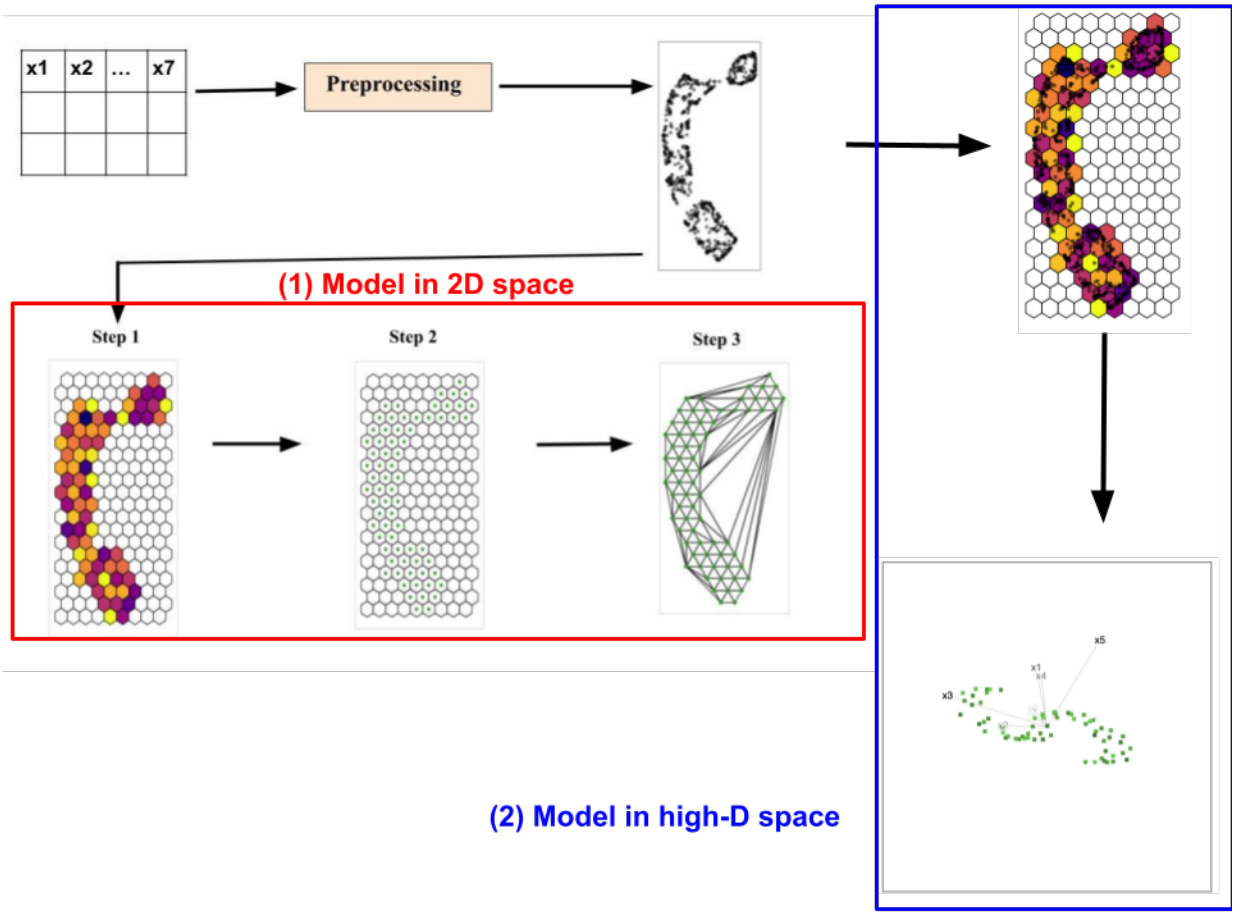


Figure 3: The flow diagram shows the main steps of our algorithm. There are two basic phases, one to generate the model in the 2D space, and other to map the model into the high-D space.

3.1 Preprocessing steps

To reduce computational complexity when applying NLDR techniques to high-D data and to reduce noise presence, PCA (**?**, **?**, **?**) is used as a preprocessing step. PCA involves identifying principal components that maximize variance. These components are then used as the high-D data for the algorithm.

3.2 Constructing the 2D model

Step 1: Scaling NLDR data

First, we prepare the 2D embedding data to fit within the bounds required for regular hexagonal binning. To achieve this, we implement two key scaling steps. Scale the first 2D

embedding component to range between 0 and 1, ensuring that all data points fall within this normalized interval. Secondly, we scale the second 2D embedding component to range between 0 and y_{max} (see Equation **??**).

The calculation of y_{max} involves several steps. First, the aspect ratio (ar) is computed by dividing the range of the second 2D embedding component (r_2) by the range of the first 2D embedding component (r_1) (see Equation **??**). Then, the hexagon ratio (hr) is determined by dividing the height of the hexagon (hb) by its width (wb) (see Equation **??**). Finally, y_{max} is derived by taking the ceiling of $\frac{ar}{hr}$ and multiplying it by hr. This process ensures that y_{max} is an integer multiple of hr, accommodating the grid layout of the hexagonal bins.

$$ar = \frac{r_2}{r_1} \tag{1}$$

$$hr = \frac{hb}{wb} \tag{2}$$

$$y_{max} = \left\lceil \frac{ar}{hr} \right\rceil * hr \tag{3}$$

Step 2: Hexagonating NLDR data

Hexagonating NLDR data (see Figure **??** Step 2) involves partitioning the NLDR data into hexagonal bins, a technique commonly referred to as hexagonal binning (**?**, **?**). This method use a hexagonal grid to create a bivariate histogram that effectively visualizes the structure of high-D data. Hexagons, one of only three regular polygons capable of tessellating a plane (**?**), offer unique advantages due to their symmetry of nearest neighbors and maximal number of sides for such tessellations (**?**). This geometric property makes hexagons more efficient in covering the plane compared to other regular tessellations and reduces visual bias when displaying data densities (**?**).

In our algorithm, we aim to conduct regular hexagonal binning, involving the computation of hexagonal grid configurations, the generation of the grid, and the assignment of the NLDR data points to hexagons.

(a) Computation of hexagonal grid configurations

In the computation of hexagonal bin configurations, there are mainly two configurations to consider: the number of bins along the x and y axes. The first step involves in determining the hexagonal size (s) and the buffer along the x and y axes ($buffer_x$ and $buffer_y$). Here, the hexagonal size (s) represents the radius of the outer circle surrounding the hexagon. The buffer is important in expanding the boundary to accommodate potential outliers or edge cases.

When computing the number of bins along the x and y axes (b_1 and b_2), the process begins by calculating the range of the respective 2D embedding component. After that, the range is adjusted to accommodate the buffer amount. Then, the spacing between hexagons is determined based on s. Finally, the number of bins along the axis is computed by dividing the adjusted range by the spacing.

For the computation of the number of bins along the x-axis (b_1) (see Equation **??**), the range of the first embedding component is defined as r_1, while the buffer amount along the x-axis is denoted as $buffer_x$, and the horizontal spacing is represented by h (see Equation **??**). On the other hand, the number of bins along the y-axis (b_2) (see Equation **??**) is computed based on the range of the second embedding component (r_2), the buffer amount along the y-axis ($buffer_y$), and the vertical spacing (v) (see Equation **??**).

$$h = \sqrt{3} * s \tag{4}$$

$$b_1 = \frac{r_1 + buffer_x}{h} \tag{5}$$

$$v = 1.5 * s \tag{6}$$

$$b_2 = \frac{r_2 + buffer_y}{v} \tag{7}$$

(b) Generation of the hexagonal grid

In this process, the first step involves generating centroids within the hexagonal grid. This begins by defining the starting coordinates for the hexagons. With the number of bins computed along the x and y axes, along with the horizontal and vertical spacing, the centroids are iteratively computed starting from the hexagonal starting coordinates.

Once the centroids for the hexagonal grid are obtained, the next step is to compute the hexagonal coordinates. For example, if the centroid of a hexagonal bin is defined as (C_x, C_y), the hexagonal coordinates can be computed using d_x (see Equation **??**) and d_y (see Equation **??**) (see Figure **??**).

$$d_x = \frac{h}{2} \tag{8}$$

$$d_y = \frac{2 * v * 1.15}{\sqrt{3}} \tag{9}$$

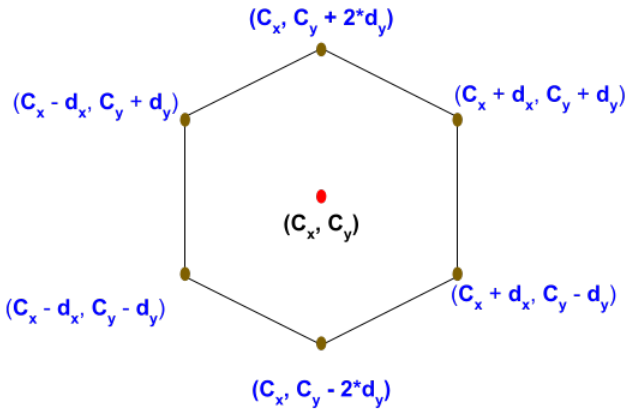


Figure 4: The hexagonal coordinates computed for the centroid (C_x, C_y).

(c) Assignment of the NLDR data points to hexagons

After obtaining the centroids of the hexagonal bins for the entire grid, the process of assigning NLDR points to hexagons involves determining the nearest hexagonal bin for each NLDR point using the 2D Euclidean distance. Then, the hexagonal ID of the nearest hexagon is assigned to the corresponding NLDR points.

Step 3: Obtaining bin centroids or bin means

In the previous step, the algorithm clusters the 2D embedding data into hexagons. Following this, in this step, the bin centroids or bin means (see Figure **??** Step 3) are obtained (**?**).

The bin centroid $(C_k^{(2)})$ for a k^{th} hexagon with hexagonal grid coordinates $(h^k x_i, h^k y_i)$, where $i = 1 \ldots 6$ can be defined as:

$$C_k^{(2)} = (C_{ky_1}, C_{ky_2}) = \left(\frac{\sum_{i=1}^{6} h^k x_i}{6}, \frac{\sum_{i=1}^{6} h^k y_i}{6} \right). \tag{10}$$

Also, the bin mean $(C_k^{(2)})$ is defined as the mean of the data points within the k^{th} hexagon (see Equation **??**).

$$C_k^{(2)} = (C_{ky_1}, C_{ky_2}) = \left(\frac{1}{n_k} \sum_{i=1}^{n_k} y_{1i}, \frac{1}{n_k} \sum_{i=1}^{n_k} y_{2i} \right), \tag{11}$$

where n_k is the number of data points within the hexagon, y_{1i} and y_{2i} are the x and y coordinates of the i^{th} data point within the hexagon.

Step 4: Triangulating bin centroids or bin means

In this step, the algorithm proceeds to triangulate the hexagonal bin centroids or bin means (see Figure **??** Step 4). Triangulation is a fundamental process in computational geometry and computer graphics that involves dividing a set of points in a given space into interconnected triangles (**?**). One common algorithm used for triangulation is Delaunay triangulation (**?**, **?**), where points are connected in a way that maximizes the minimum angles of the resulting triangles, leading to a more regular and well-conditioned triangulation.

Delaunay triangulation can be defined as follows:

Let $C^{(2)} = \{C_1^{(2)}, C_2^{(2)}, ..., C_m^{(2)}\}$ be a set of m bin centroids or bin means in the plane. Delaunay triangulation of $C^{(2)}$, denoted as $DT(C^{(2)})$, is a triangulation of the convex hull of $C^{(2)}$ such that the circumcircle of every triangle in the triangulation contains no other points from $C^{(2)}$.

Given that the hexagons are regular, the resulting triangles will mostly be equilateral.

3.3 Lifting the model into high dimensions

3.3.1 Lifting the triangular mesh points into high dimensions

Consider $f : \mathbb{R}^p \to \mathbb{R}^2$ be a function that maps the high-D data ($X_{n\times p}$) to its NLDR equivalent ($Y_{n\times d}$). Then, let $g : \mathbb{R}^2 \to \mathbb{R}^2$ be a function that maps each 2D embedding point to its closest centroid ($C^{(2)}$). It follows that $f(g(x))$ maps the high-D points x to the centroid in 2D ($C_k^{(2)}$). Also, define a function $v : \mathbb{R}^2 \to \mathbb{R}^p$ maps the 2D centroid ($C^{(2)}$) to the high-D mean of the points ($C^{(p)}$) in the hexagon.

The high-D mean of all the points in k^{th} hexagon by

$$C_k^{(p)} = (C_{kx_1}, ..., C_{kx_p}) = \left(\frac{1}{n_k} \sum_{i=1}^{n_k} x_{1i}, \frac{1}{n_k} \sum_{i=1}^{n_k} x_{2i}, \dots, \frac{1}{n_k} \sum_{i=1}^{n_k} x_{pi} \right). \tag{12}$$

Therefore,

$$v(C_k^{(2)}) = C_k^{(p)}. \tag{13}$$

Therefore, $f(g(x))$ gives the 2D centroid associated with high-D points x, and $v(C_k^{(2)})$ gives the high-D centroid associated with 2D point $C_k^{(2)}$. Thus, $v(f(g(x)))$ gives the high-D centroid ($C_k^{(p)}$) associated with the 2D embedding of the points x.

3.3.2 Lifting the 2D triangular mesh into high dimensions

As described in Step 4 of Section **??**, during the triangulation process in 2D space, vertices are identified to form edges. With the knowledge of the high-D mappings for the 2D hexagonal bins, the vertices connected in 2D are also connected in high-D (see video linked in Figure **??**).

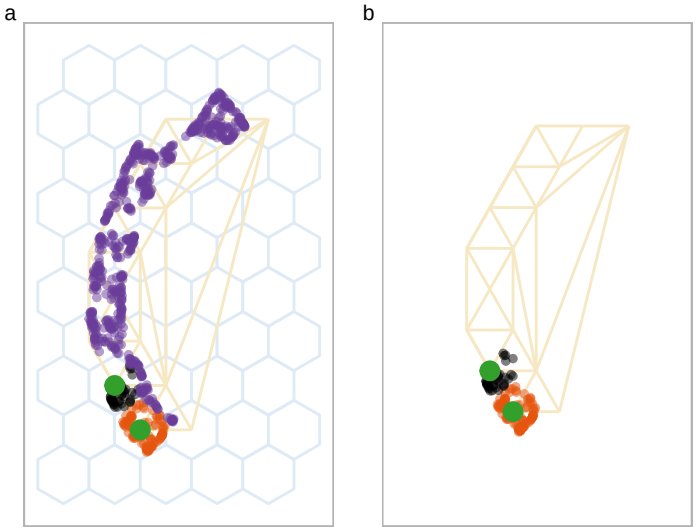


Figure 5: How the 2D model lift into high dimensions? (a) visualize the points and the hexagonal bin centroids related 7^{th} and 12^{th} hexagons, (b) visualization of the edge connected the 7^{th} and 12^{th} hexagons (colored in red) in the triangular mesh. A video of tour animation is available at https://youtu.be/hxU91xNTJL0.

3.4 Tuning the model

The performance and robustness of our model depend on four key parameters: (i) the total number of bins (b), (ii) a benchmark value used to remove low-density hexagons, (iii) a benchmark value used to remove long edges, and (iv) starting point of the hexagonal grid. However, there is no analytical formula to calculate an appropriate value for these parameters. The selection of these parameter values depends on the model performance computed by Mean Squared Error (MSE) (see Section **??**).

3.4.1 Total number of bins

The number of hexagonal bins in the hexagonal grid has a considerable impact on the construction of the 2D model, serving as the initial step in building the 2D model. The chosen total number of bins must effectively capture the structure of the NLDR data. If the number of bins is too low, the model may not be able to capture the structure of the NLDR data effectively (see Figure **??** (a)), while if there are too many bins, it may result in over-fitting the individual points of the NLDR data (see Figure **??** (c)). Therefore, it is important to determine an appropriate number of bins to construct an effective model.

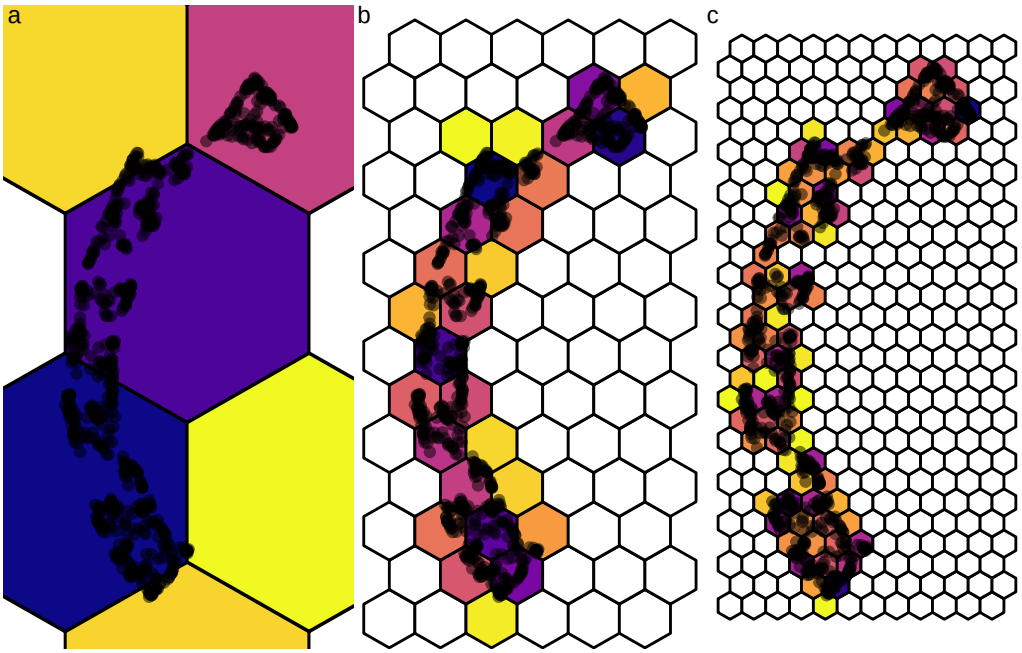


Figure 6: Hexbin plots from different number of bins for the **UMAP** embeddings of **S-curve** training data: (a) b = 12 (3, 4), s = 0.62, (b) b = 84 (6, 14), s = 0.13, and (c) b = 336 (12, 28), s = 0.06. The hexbins are colored based on the density of points, with darker colors indicating higher point density and yellow color representing lower point density within each bin. What is the number of bins that would be effective in representing low-dimensional data?

$$b = b_1 \times b_2 \tag{14}$$

Furthermore, the total number of bins is calculated based on the number of bins along the x-axis and y-axis, as shown in Equation **??**. To determine the effective total number of bins, candidate values are selected based on the range between the minimum and approximate maximum number of bins along the x and y axes. The minimum number of bins along each axis is set to 1, while the maximum number is estimated by taking the square root of the NLDR data points. The analysis evaluates the MSE across varying total number of bins within this range, covering the minimum to maximum values along both axes (see Figure **??**).

3.4.2 Benchmark value to remove low-density hexagons

After establishing the hexagonal grid with an appropriate number of bins, some hexagonal bins may have few or no data points within them (see Figure **??** (a)). To ensure comprehensive coverage of the NLDR data, it is necessary to select hexagonal bins with a considerable number of data points. This involves calculating the number of points within each hexagon. Then, the standard count is computed by dividing the number of points within each hexagon by the maximum number of points in the grid (see Equation **??**). Next, bins with a standard count less than a certain benchmark value are removed (see

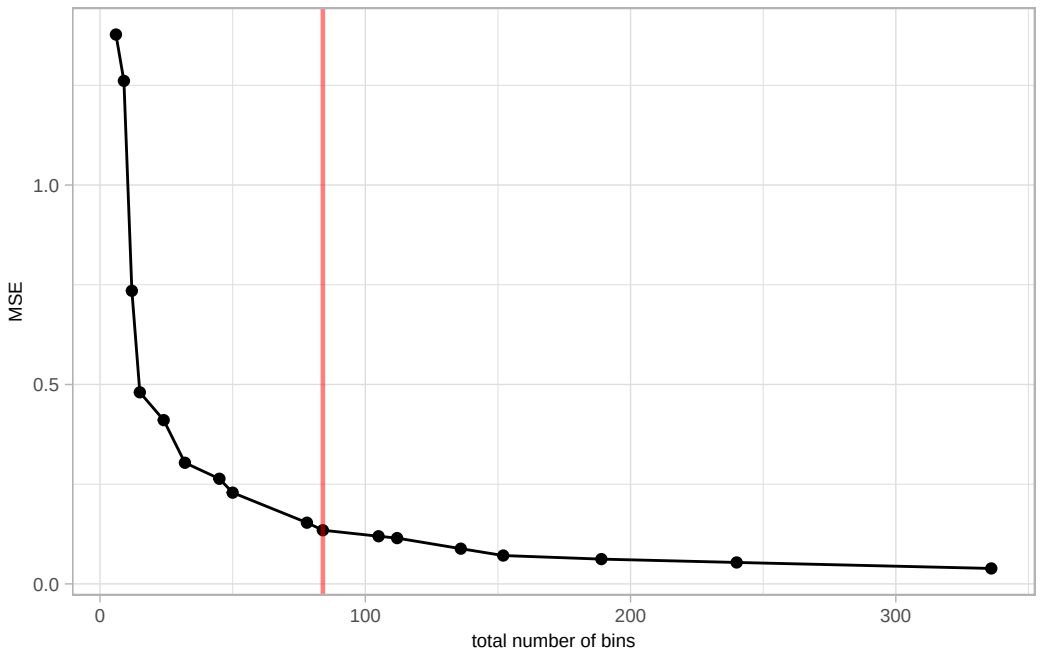


Figure 7: Goodness of fit statistics from UMAP applied to training S-curve dataset. What is the effective number of bins in each NLDR technique to create a 2D model? The MSE plot have a steep slope at the beginning, indicating that a smaller number of bins causes a larger amount of error. Then, the slope gradually declines or level off, indicating that a higher number of bins generates a smaller error. Using the elbow method, when the total number of bins is set to 84, the slope of the Mean Squared Error (MSE) plot experiences a sudden and noticeable change, resembling an elbow-like shape. This point indicates that adding less bins does not enough to capture the data structure.

Figure **??** (b)). The following steps will help in determining a suitable value for removing low-density hexagons:

1. Plot the distribution of the standardized counts (see Figure **??**).
2. Examine the distribution of counts.
3. Select the first quartile value if the distribution is skewed.

$$\text{standard count} = \frac{\text{count}}{\text{max count}} \tag{15}$$

Furthermore, selecting the benchmark value for removing low-density hexagons is important. Removing unnecessary bins may lead to the formation of long edges and an uneven 2D model. Hence, rather than solely relying on the benchmark value to identify hexagons for removal, it's essential to consider the standard number of points in the neighboring hexagons of the identified low-density ones (see Figure **??** (c)). If neighboring bins also show low counts, only those bins will be removed. The remaining bins are used to construct the 2D model.

The benchmark value for removing low-density hexagons ranges between 0 and 1. When analyzing how these benchmark values influence model performance, it's essential to observe the change in Mean Squared Error (MSE) as the benchmark value increases (see Figure **??**). The MSE shows a gradual increase as the benchmark value progresses from 0 to 1. Evaluating this rate of increase is important. If the increment is not considerable, the decision might lean towards retaining low-density hexagons.

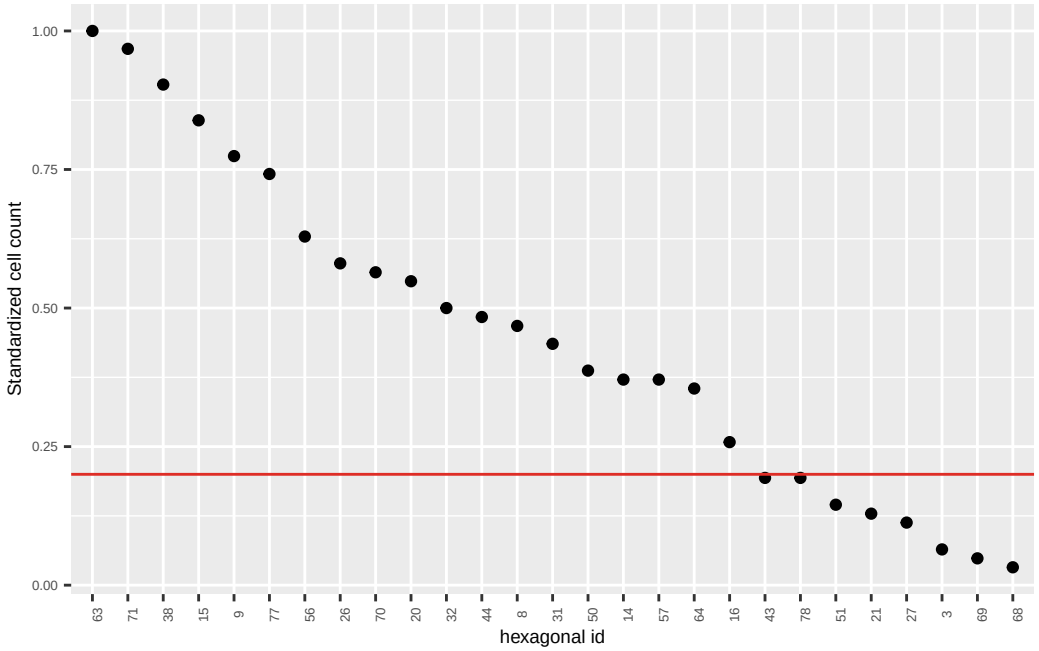


Figure 8: Distribution of standardize counts by hexagons.

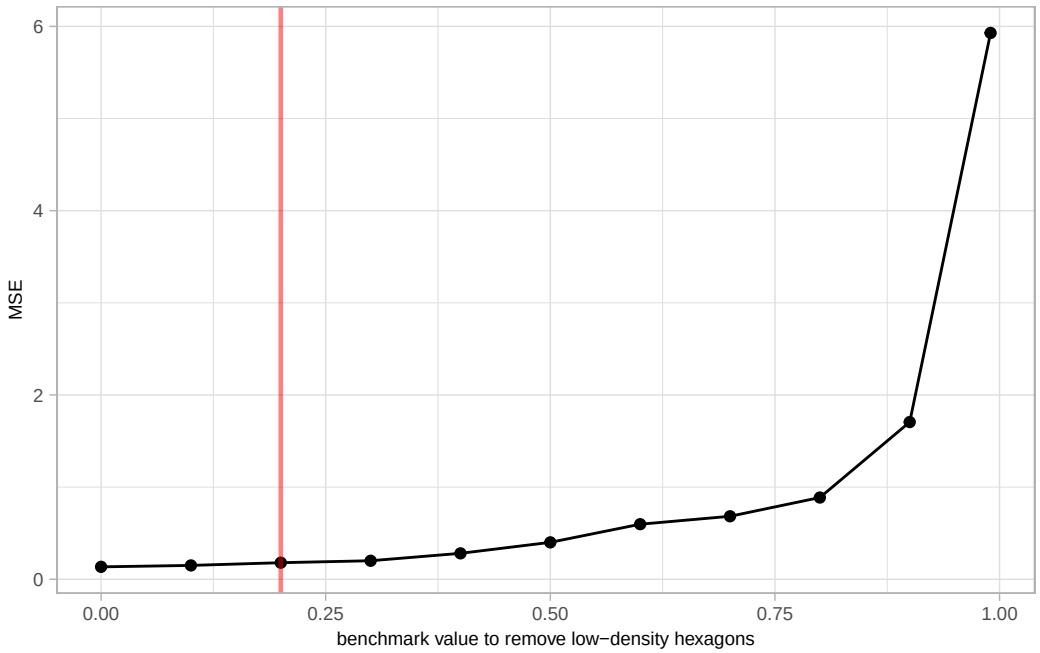


Figure 9: Goodness of fit statistics from UMAP applied to training S-curve dataset with different benchmark values to remove the low-density heaxgons. What is the effective benchmark value to remove the low-density heaxgons? The MSE plot have a steep slope at the beginning, indicating that a smaller benchmark causes a small amount of error. Then, the slope gradually increases or level up, indicating that a higher number of benchmark values generates a higher error. Using the reverse elbow method, when the benchmark value is set to 0.2, the slope of the Mean Squared Error (MSE) plot experiences a sudden and noticeable change, resembling an elbow-like shape. This point indicates that higher benchmark values remove the necessary bins as well which lead to distruct the 2D structure.

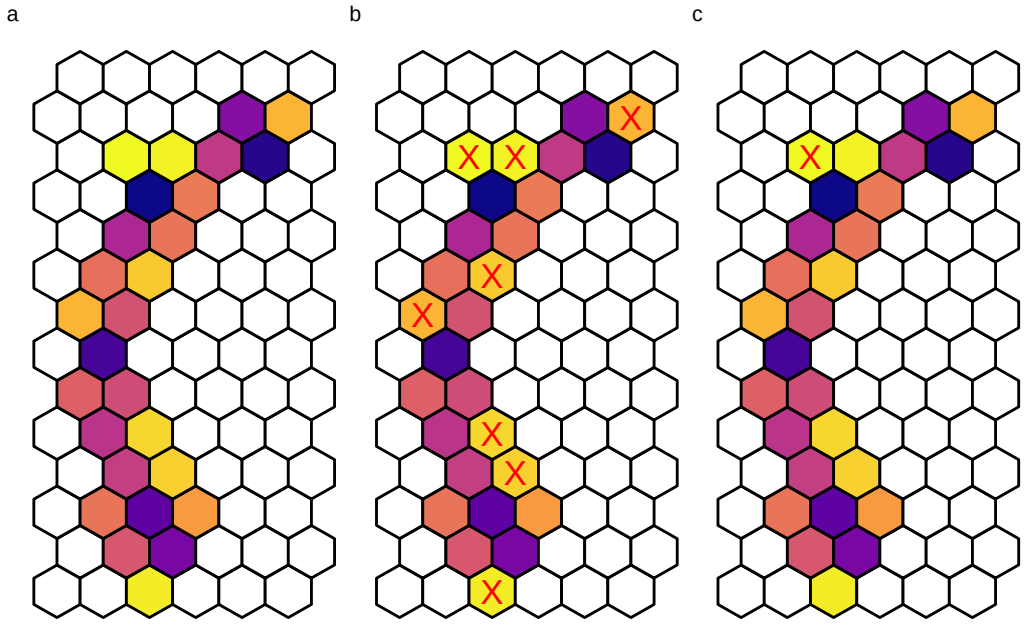


Figure 10: (a) Hexbin plot with colored hexbins based on point density, (b) Identification of low-density hexagons using a benchmark value of 0.2, and (c) Identification of low-density hexagons considering neighboring bins.

3.4.3 Benchmark value for removing long edges

Creating a smooth 2D representation (see Figure **??** (c)) requires removing edges that connect distant bin centroids within the triangular mesh (see Figure **??** (b)). These edges only exist in the 2D model and do not extend into higher dimensions, ensuring that their removal does not impact the model in higher dimensions. While specific criteria for determining the benchmark value to remove long edges do not exist, the following steps provide an approach to identifying a suitable threshold:

1. Plot the distribution of the 2D Euclidean distances (see Figure **??**).
2. Identify the largest difference between consecutive distance values.
3. Take the largest distance value corresponding to this difference as the benchmark value.

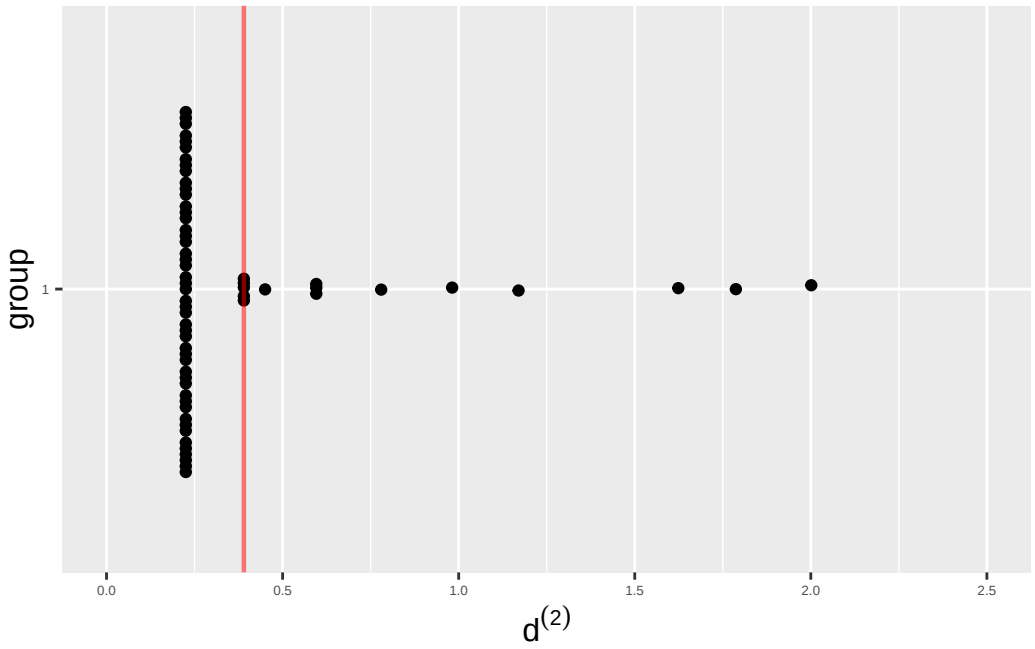


Figure 11: Distribution of 2D Euclidean distances between bin centroids of the triangular mesh generated with S-curve UMAP data.

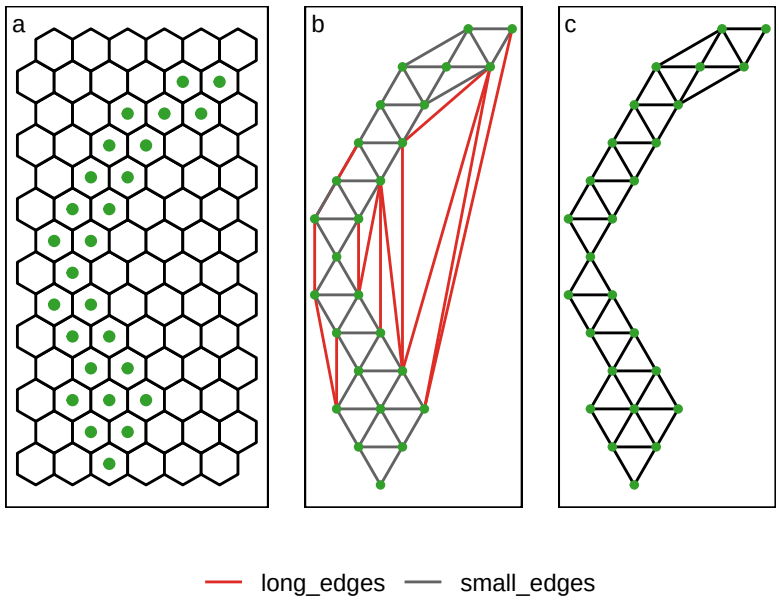


Figure 12: (a) Full hexagonal grid with bin centroids, (b) Model constructed in 2D with long edges, and (c) Model constructed in 2D after removing the long edges.

3.4.4 Starting point of the hexagonal grid

Hexagonal binning involves tessellating the xy plane over the set defined by the range of x and y (**?**). Therefore, the starting point of the hexagonal grid is important parameter to consider. Having different starting point results in shifting the hexagonal grid and bin

centroids (see Figure **??**). With a specific total number of bins (which corresponds to a specific range along the x and y axes), it's essential to explore how different starting points affect the model's performance (see Figure **??**). This is because different starting points result in varying distributions of non-empty bins (see Figure **??**), impacting the model.

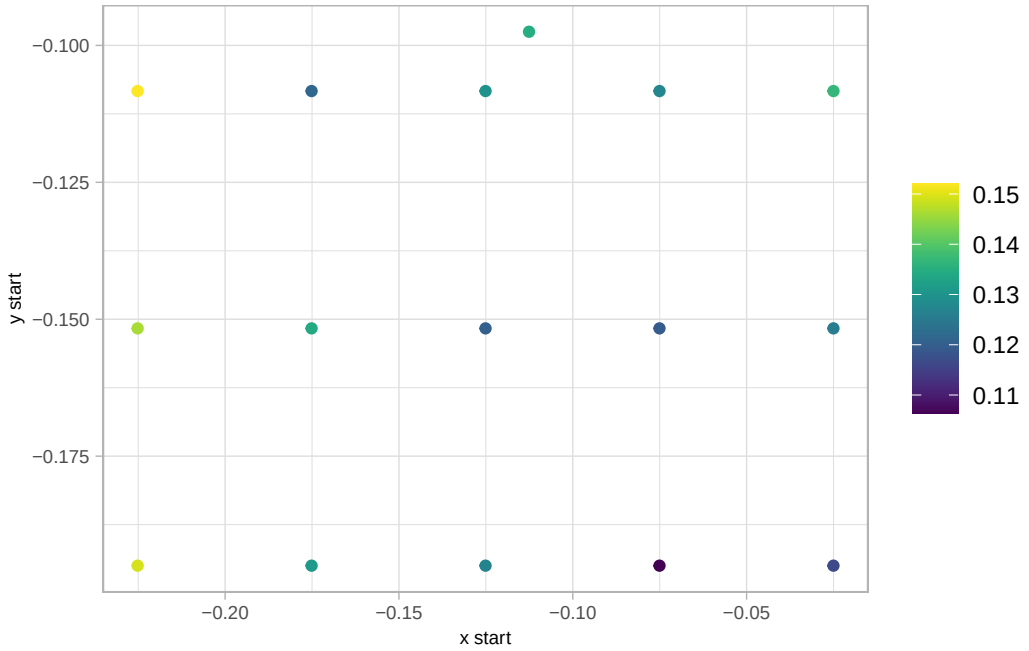


Figure 13: Goodness of fit statistics from UMAP applied to training S-curve dataset with different benchmark values to remove the low-density heaxgons. What is the effective benchmark value to remove the low-density heaxgons? The MSE plot have a steep slope at the beginning, indicating that a smaller benchmark causes a small amount of error. Then, the slope gradually increases or level up, indicating that a higher number of benchmark values generates a higher error. Using the reverse elbow method, when the benchmark value is set to 0.2, the slope of the Mean Squared Error (MSE) plot experiences a sudden and noticeable change, resembling an elbow-like shape. This point indicates that higher benchmark values remove the necessary bins as well which lead to distruct the 2D structure.

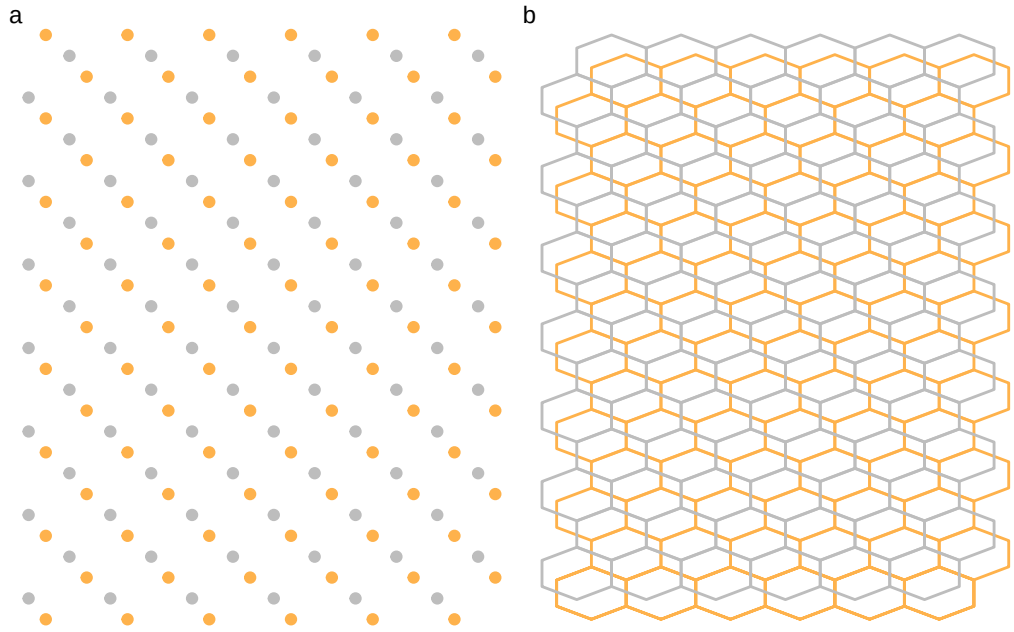


Figure 14: (a) Visualization of bin centroids and (b) hexagonal grid, showcasing the configuration when the starting point defined at $(-0.113, -0.098)$ (colored in gray) and the starting point defined at $(-0.025, -0.108)$ (colored in light orange).

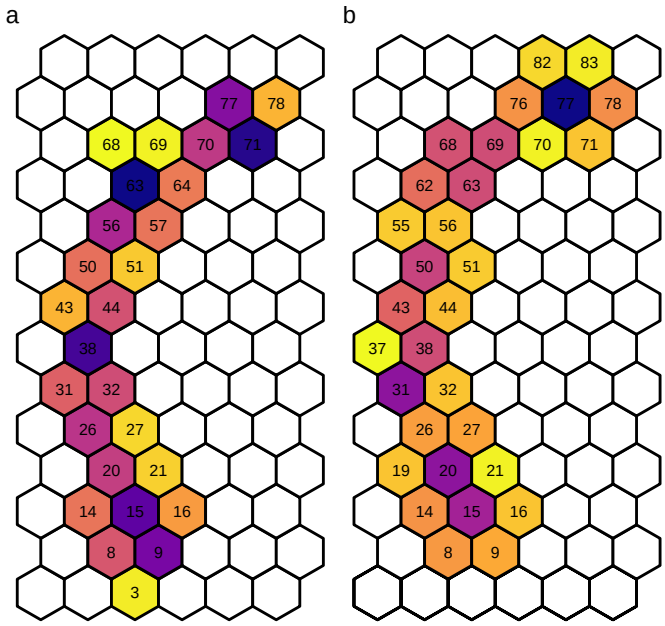


Figure 15: Hexbin plots shows the distribution of points before and after changing the starting point of the hexagonal grid generated for UMAP applied to the training S-curve dataset. (a) Hexbin plot when the starting point defined at $(-0.113, -0.098)$, and (b) Hexbin plot after changing the starting point to $(-0.025, -0.108)$.

3.5 Model summaries

3.5.1 Predicted values and residuals

The prediction approach involves performing the K-nearest neighbors (KNN) algorithm for an unsupervised classification problem. First, the nearest high-D model point is identified for a given new high-D point. Then, the corresponding 2D centroid mapping for the identified high-D model point is determined. Finally, the coordinates of this 2D centroid are used as the predicted 2D embedding for the new high-D data point. This step is particularly valuable due to the limitations of some NLDR techniques, like tSNE, which don't provide a straightforward method for prediction. As a result, our approach offers a solution that capable of generating predicted 2D embedding regardless of the NLDR technique employed, effectively addressing this functional gap.

Residuals are essential for evaluating the accuracy of representing high-D points by the high-D mapping of 2D bin centroids. To measure this accuracy, an error metric is introduced, quantifying the sum of squared differences between the high-D data (x_{ij}) and the high-D mapping of the 2D bin centroid data (C_{x_ij}) across all observations and dimensions (see Equation **??**).

$$\text{Error} = \sum_{j=1}^{n}\sum_{i=1}^{p}(x_{ij} - C_{x_ij})^2 \tag{16}$$

Here, n represents the number of observations, p represents the dimensions of high-D data, x_{ij} is the high-D data, and C_{x_ij} is the high-D mapping of the 2D bin centroid.

3.5.2 Goodness of fit statistics

To assess how well our method captures and represents the underlying structure of the high-D data, Mean Squared Error (MSE) is used. When computing MSE, total model error (see Section **??**) is divided by the number of observations to make it as a mean value (see Equation **??**).

$$\text{MSE} = \sum_{j=1}^{n}\frac{\sum_{i=1}^{p}(x_{ij} - C_{x_ij})^2}{n} \tag{17}$$

3.6 Simulated data example

In this section, the effectiveness of the algorithm is described using a simulated dataset. The dataset consists of five spherical Gaussian clusters in 4-d, with each cluster containing an equal number of points and the same within-cluster variation.

In the 2D layouts generated by various NLDR techniques, as shown in Figure **??**, five distinct clusters are observable, except for PHATE. In tSNE (see Figure **??** (a)), these clusters appear closely. UMAP arranges all clusters in a parallel manner, with three aligned

in one line and the other two in a separate line (see Figure **??** (b)). In contrast, PHATE shows two closely positioned clusters and three more distant ones (see Figure **??** (c)). PaCMAP shows one central cluster and the remaining four spread out in different directions (see Figure **??** (e)). Finally, in TriMAP, two clusters are close, though not as tightly as PHATE, while the other three are well-separated (see Figure **??** (d)).

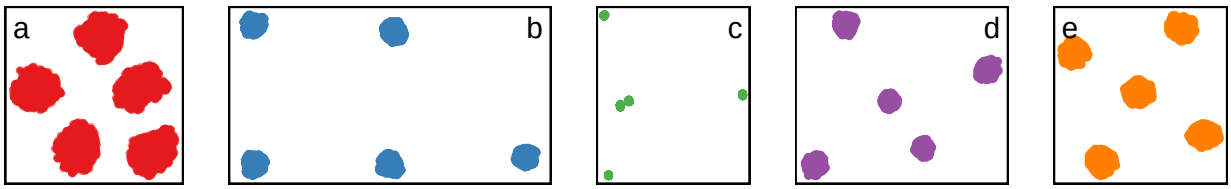
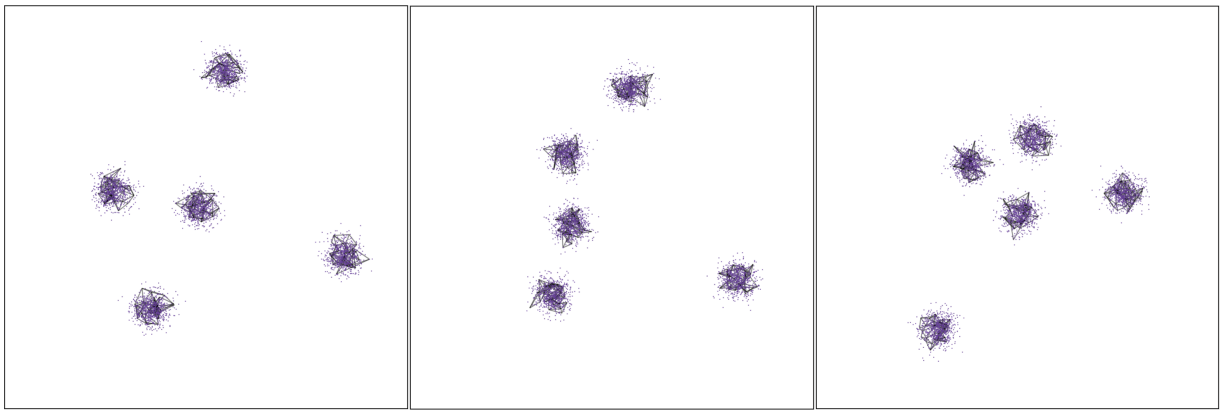


Figure 16: 2D layouts from different NLDR techniques applied the same data: (a) tSNE (perplexity = 61), (b) UMAP (n_neighbors = 15), (c) PHATE (knn = 5), (d) TriMAP (n_inliers = 5, n_outliers = 4, n_random = 3), and (e) PaCMAP (n_neighbors = 10, init = random, MN_ratio = 0.9, FP_ratio = 2). Is there a best representation of the original data or are they all providing equivalent information?

To investigate which is the best representation to visualize the original data or all NLDR methods provide equivalent information, we visualize the model-in-data space. Models from all NLDR methods show five well-separated clusters (see Figure **??**, Figure **??**, Figure **??**, Figure **??**, and Figure **??**). This suggests that for the five Gaussian cluster dataset, all NLDR methods effectively preserve the global structure. tSNE, UMAP, and PHATE display clusters with varying densities, indicating their ability to capture within-cluster variation (see Figure **??**, Figure **??**, and Figure **??**). On the other hand, both TriMAP and PaCMAP show clusters with flat surfaces, suggesting a failure to capture within-cluster variation (see Figure **??** and Figure **??**). Therefore, TriMAP and PaCMAP do not capture the local structure as effectively as other methods.



(a) (b) (c)

Figure 18: Screen shots of the **langevitour** of the five Gaussian clusters dataset, shows the model with tSNE in high-D, a video of the tour animation is available at (https://youtu.be/oQxEb4wRdHI).

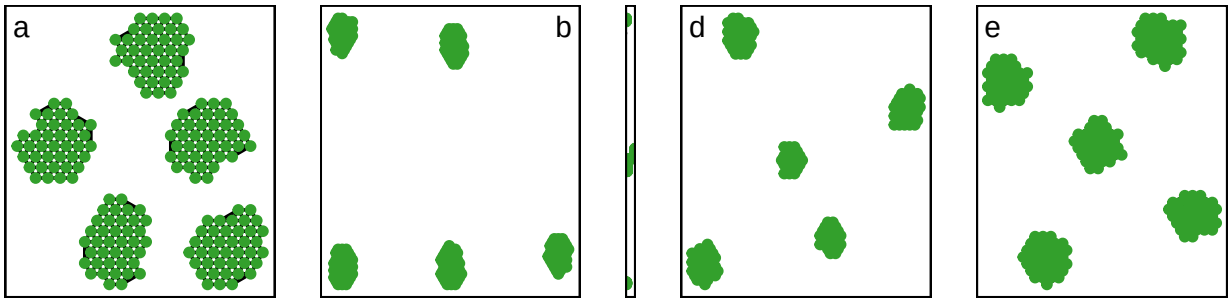
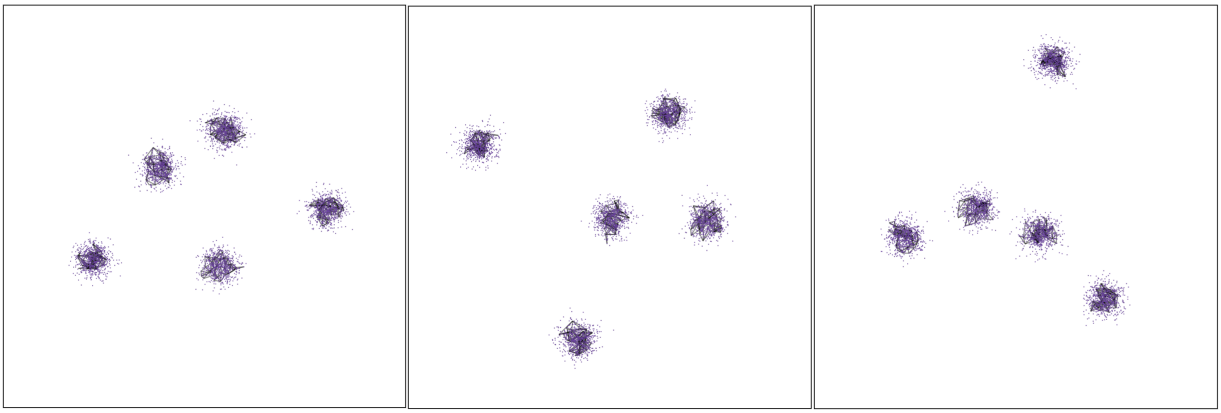
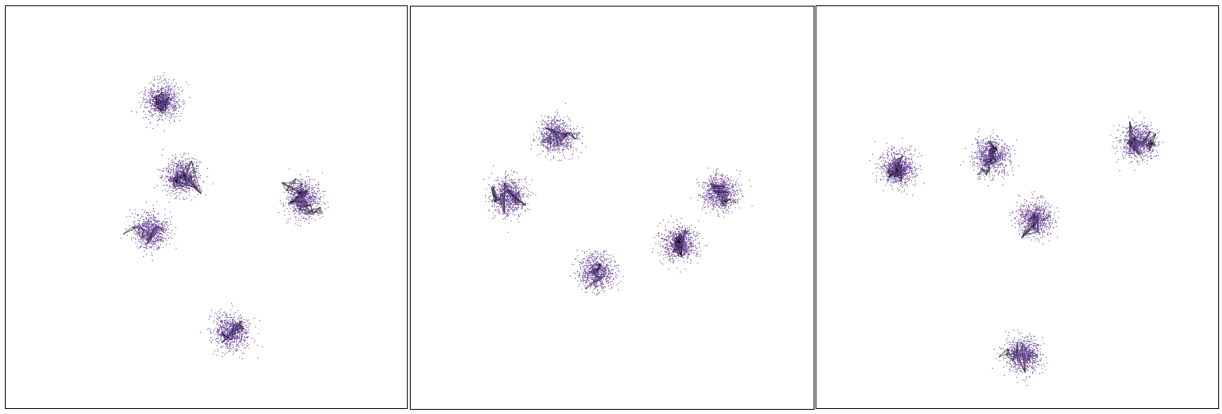


Figure 17: Is there a best model to represent the original data in 2D space or are they all providing equivalent information?, (a) Model generated with 21 and 28 bins along the x and y axes, respectively, using a hexagonal size of 0.03 in the 2D space with tSNE, (b) Model generated with 60 and 79 bins along the x and y axes, respectively, using a hexagonal size of 0.01 in the 2D space with UMAP, (c) Model generated with 60 and 1927 bins along the x and y axes, respectively, using a hexagonal size of 0.01 in the 2D space with PHATE, (d) Model generated with 46 and 61 bins along the x and y axes, respectively, using a hexagonal size of 0.013 in the 2D space with TriMAP, and (e) Model generated with 31 and 40 bins along the x and y axes, respectively, using a hexagonal size of 0.02 in the 2D space with PaCMAP.



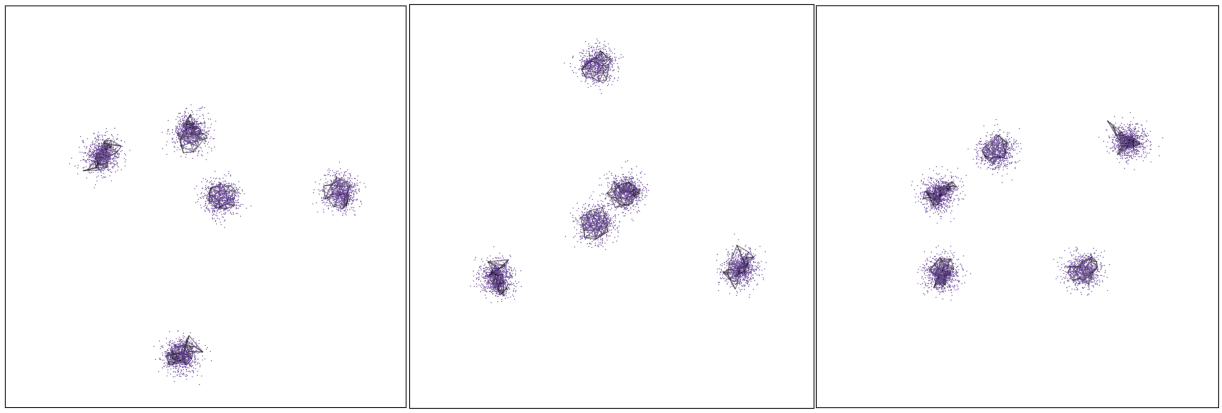
(a) (b) (c)

Figure 19: Screen shots of the **langevitour** of the five Gaussian clusters dataset, shows the model with UMAP in high-D, a video of the tour animation is available at (https://youtu.be/JW49csPpDx4).



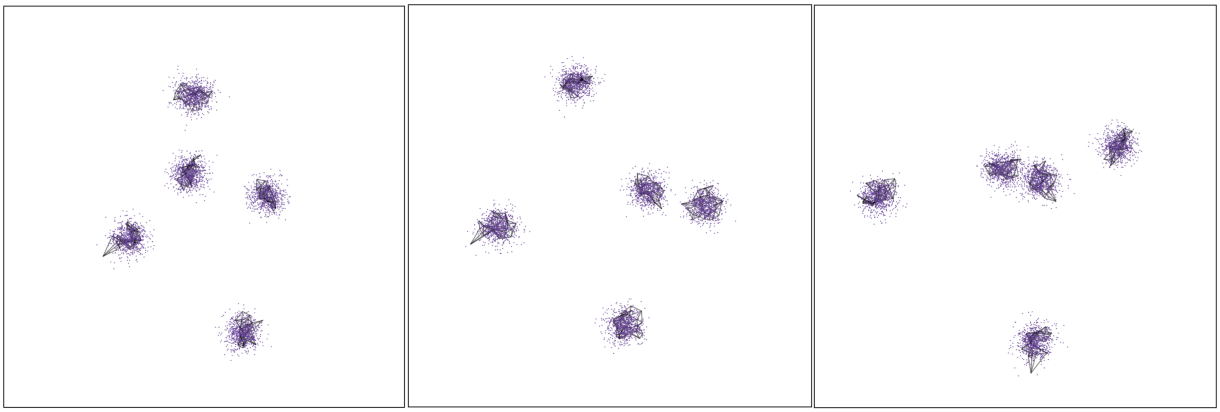
(a) (b) (c)

Figure 20: Screen shots of the **langevitour** of the five Gaussian clusters dataset, shows the model with PHATE in high-D, a video of the tour animation is available at (https://youtu.be/xqhRODt8Z-o).



(a) (b) (c)

Figure 21: Screen shots of the **langevitour** of the five Gaussian clusters dataset, shows the model with TriMAP in high-D, a video of the tour animation is available at (https://youtu.be/X-uHSB4VoZ0).



(a) (b) (c)

Figure 22: Screen shots of the **langevitour** of the five Gaussian clusters dataset, shows the model with PaCMAP in high-D, a video of the tour animation is available at (https://youtu.be/UxvU8HQd7JM).

4 Applications

4.1 Single-cell RNA-seq data of human

In the field of single-cell studies, a common analytical task involves clustering to identify groups of cells with similar expression profiles. Analysts often turn to NLDR techniques to verify and identify these clusters and explore developmental trajectories. To illustrate the importance of NLDR techniques and parameter selection in identifying clusters, Human Peripheral Blood Mononclear Cells (PBMC3k) dataset (**?**) is used. In a study by **?**, this dataset was used to demonstrate how UMAP represents clusters (see Figure **??**). As shown in Figure **??**, there are three distinct and well-separated clusters.

Figure 23: 2D layout from UMAP (n_neighbors = 30, min_dist = 0.3) applied for the PBMC3k dataset. Is this a best representation of the original data?

To determine whether the UMAP representation with the parameter choice suggested by **?** preserves the original data structure, we visualize the model constructed with UMAP overlaid on the high-D data (model-in-data space). The figures in Figure **??** show three well-separated clusters, indicating that the suggested UMAP representation preserves the global structure (see Figure **??**). However, as shown in Figure **??**, these clusters are close to each other in high-D space. Also, nonlinear continuity patterns and high-density patches within the clusters are observed (see Figure **??**). Therefore, the suggested UMAP representation (see Figure **??**) does not accurately preserve the local structure of the PBMC3k dataset.

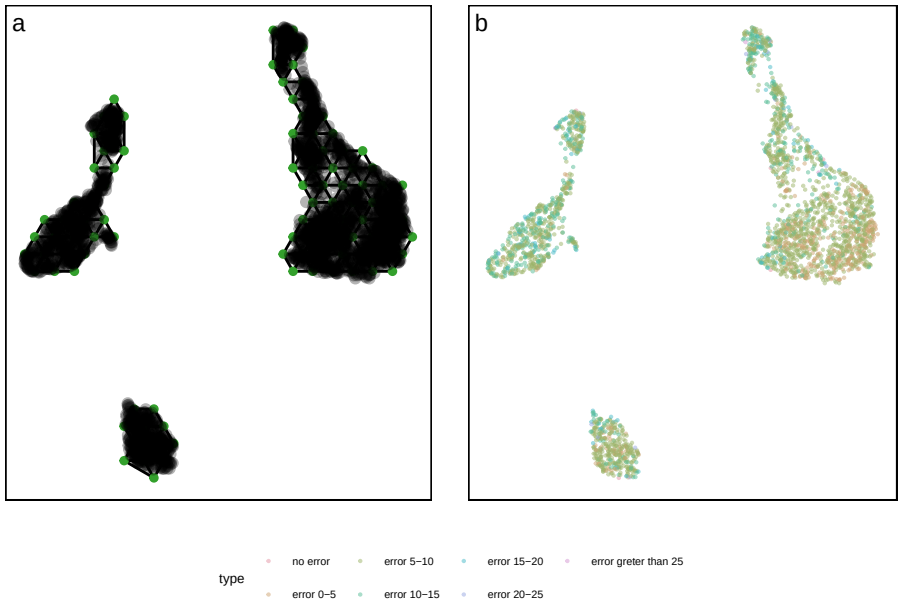
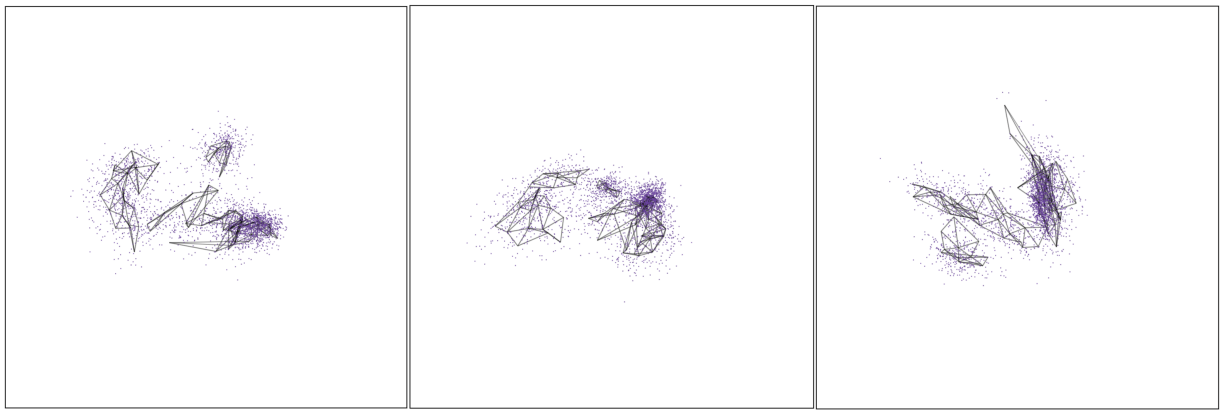


Figure 24: (a) Model generated with 21 and 28 bins along x and y axes with hexagonal size 0.03 in the 2D space overlaid on UMAP data, and (b) high-D model error in model space.



(a) (b) (c)

Figure 25: Screen shots of the **langevitour** of the PBMC3k data set, shows the model in high-D, a video of the tour animation is available at (https://youtu.be/Gnhh4hfsbbg).

To find the best NLDR representation for PBMC3k dataset, we observe the absolute error for different number of non-empty bins and select the one with the lowest error. As shown in Figure **??**, tSNE with a perplexity set to 51, has the lowest error. Therefore, tSNE with a perplexity of 51 is used as the best representation for PBMC3k dataset.

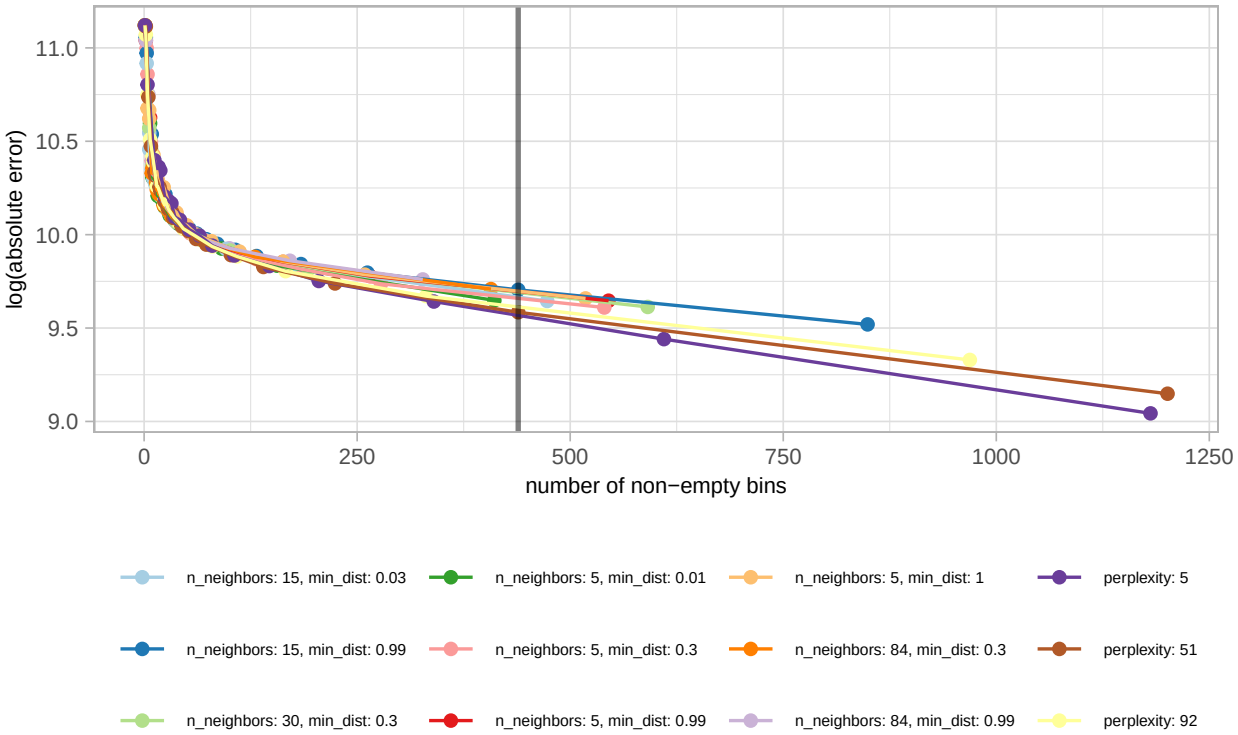


Figure 26: Absolute error from UMAP and tSNE applied to training PBMC3k dataset with diffferent parameter choices. What is the best parameter choice to create the model? The residual plot have a steep slope at the beginning, indicating that a smaller number of non-empty bins causes a larger amount of error. Then, the slope gradually declines or level off, indicating that a higher number of non-empty bins generates a smaller error. Using the elbow method, it was observed that when the number of non-empty bins is set to 439, the lowest error occurred with the parameters perplexity: 51.

As shown in Figure **??**, there are three well-separated clusters, although they are located close to each other. Additionally, nonlinear structures can be observed within the clusters (see Figure **??**). Hence, the data structure that UMAP failed to capture for the PBMC3k dataset is successfully captured by tSNE.

Figure 27: 2D layout from tSNE (perplexity = 51) applied for the PBMC3k dataset. Is this a best representation of the original data?

Next, the model is fitted for tSNE, and the resulting model is visualized in the data space. The model shows some quirks, as shown in Figure **??**. The large cluster is divided into three subclusters, which are in close to each other.

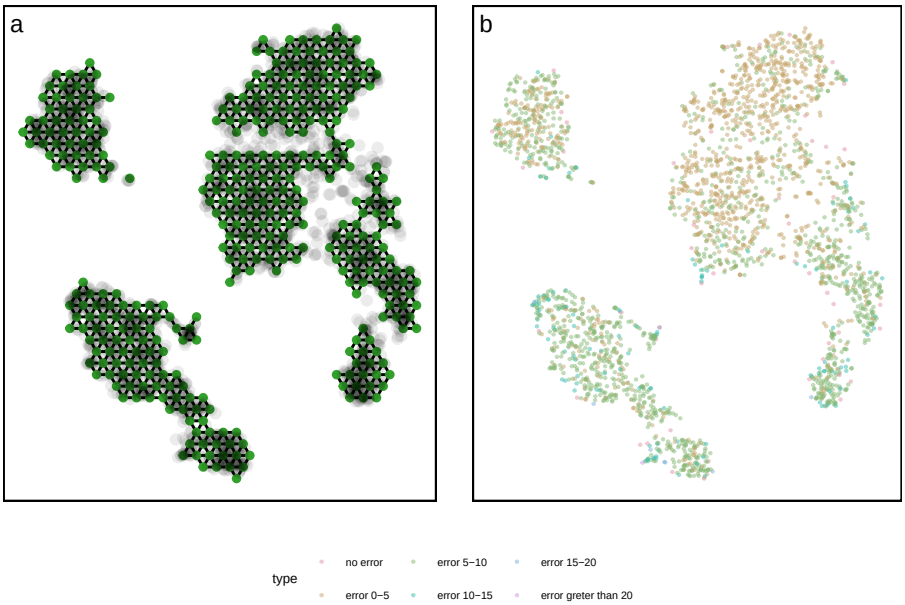
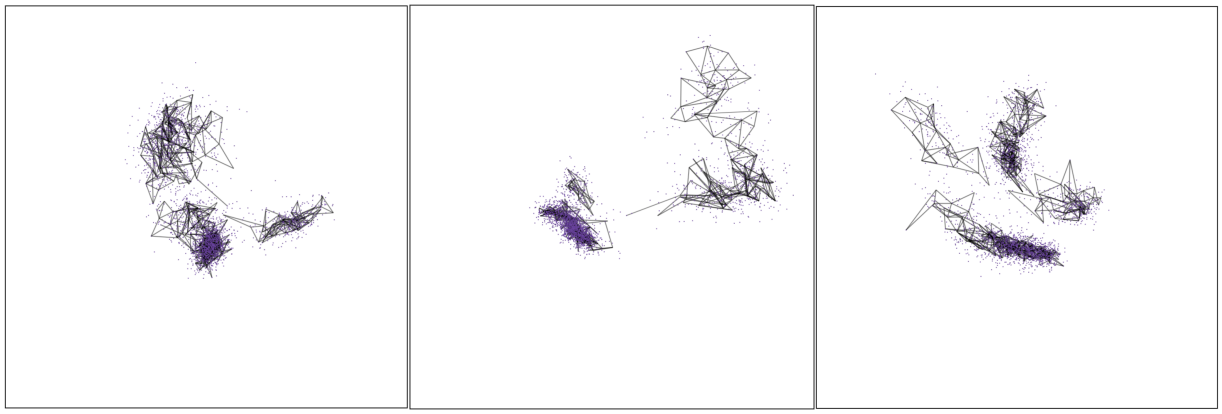


Figure 28: (a) Model generated with 31 and 40 bins along x and y axes with hexagonal size 0.02 in the 2D space overlaid on tSNE data, and (b) high-D model error in model space.



(a) (b) (c)

Figure 29: Screen shots of the **langevitour** of the PBMC3k data set, shows the model in high-D, a video of the tour animation is available at (https://youtu.be/R0EroJcGbas).

4.2 Handwritten digits

The MNIST dataset consists of grayscale images of handwritten digits (**?**). **?** used this dataset to demonstrate how the PaCMAP preserves local structure. We selected the 2D embedding of PaCMAP for the handwritten digit 1. As shown in Figure **??**, we observe a nonlinear structure present in 2D space. Furthermore, the angle of digit 1 images varies along the nonlinear structure (see Figure **??**). To answer the following questions,

- Is the original data structure of the handwritten digit 1 dataset nonlinear?
- Where does PaCMAP preserve the original data structure?
- Where does PaCMAP show quirks?

we visualize the model constructed with PaCMAP overlaid on data in high-D space (model-in-data space).

According to Figure **??**, the nonlinear continuity structure observed in the 2D representation of PaCMAP is also visible when visualizing the model overlaid on the data space. This provides evidence that the model accurately captures the structure of the high-D data. Also, the model has a twisted pattern within the nonlinear structure, which is an addition pattern of data that cannot be seen in the 2D representation (see Figure **??**).

In order to visually assess the preservation of local structure, it's important to consider the edges in the model. In a model that effectively preserves local structure, there shouldn't be any long edges in high-D space. However, as shown in Figure **??**, some long edges exist in high-D space that are not identified as long edges in the 2D representation.

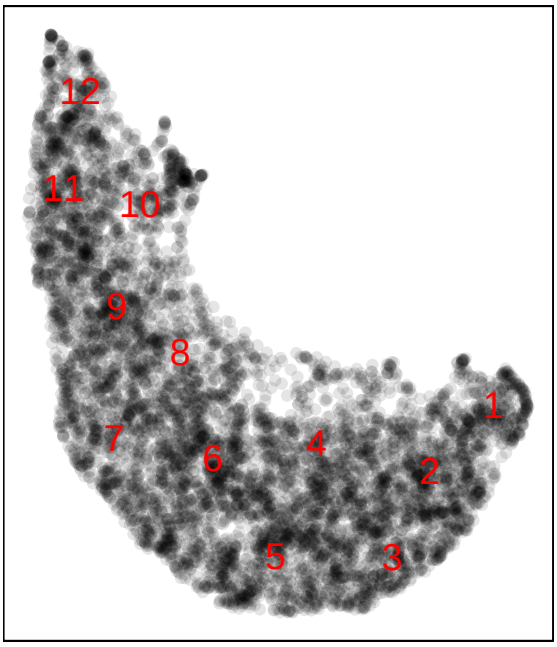


Figure 30: (a) 2D layout from PaCMAP (n_components=2, n_neighbors=10, init=random, MN_ratio=0.9, FP_ratio=2.0) applied for the digit 1 of the MNIST dataset. Is this a best representation of the original data?

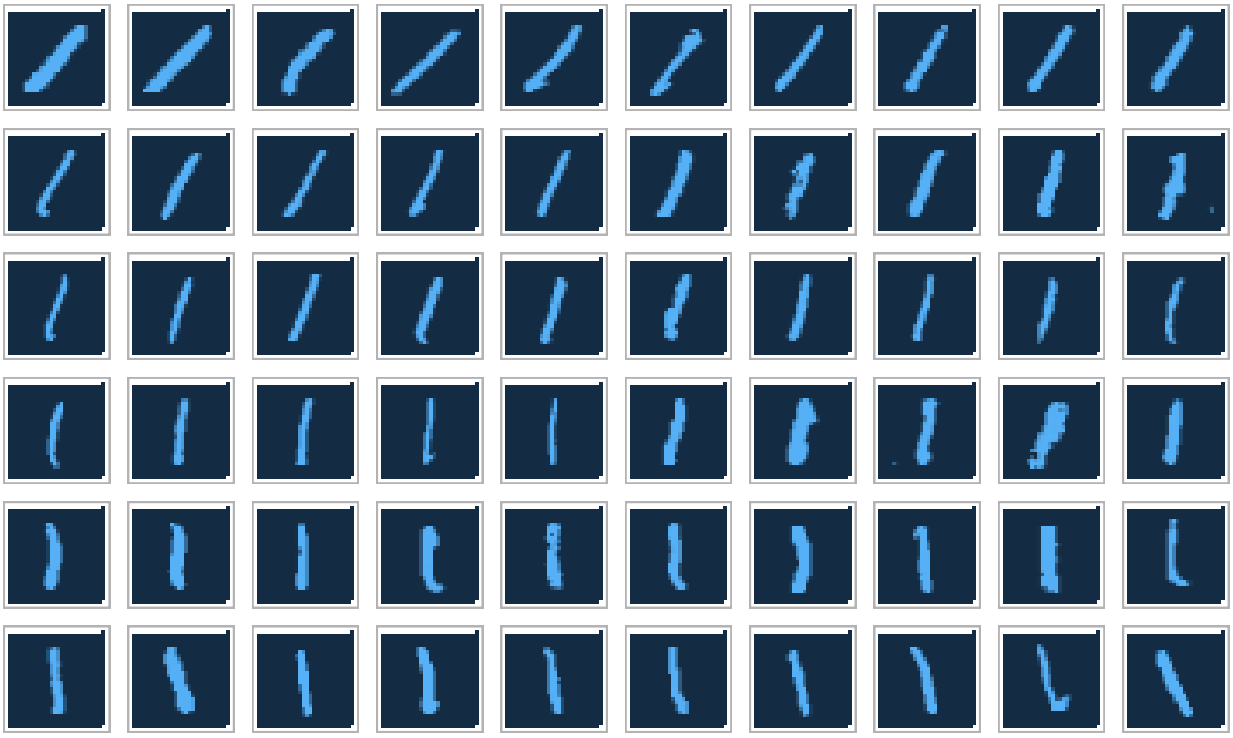


Figure 31: Images of the handwritten digit 1 are distributed along the right-to-left of first embedding axis of the PaCMAP nonlinear structure.

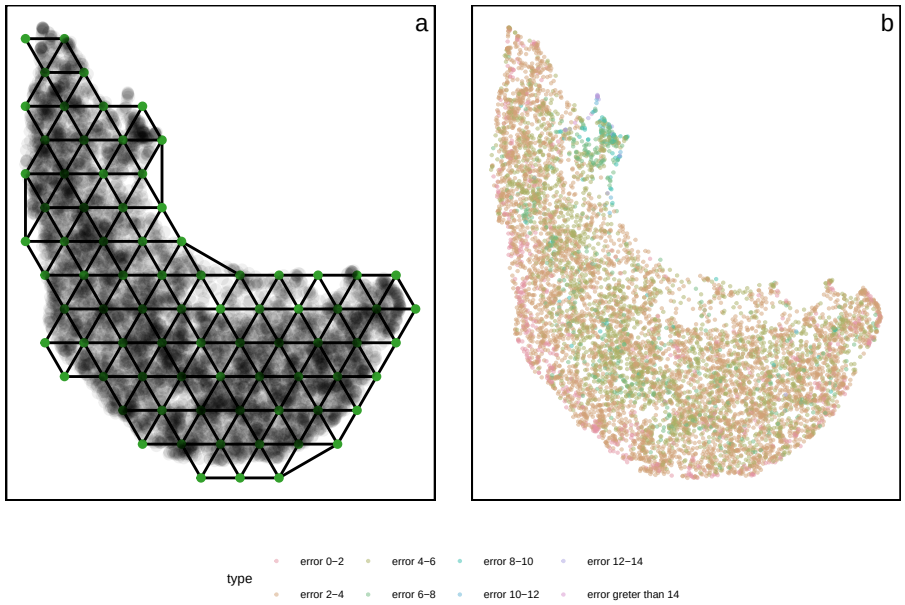
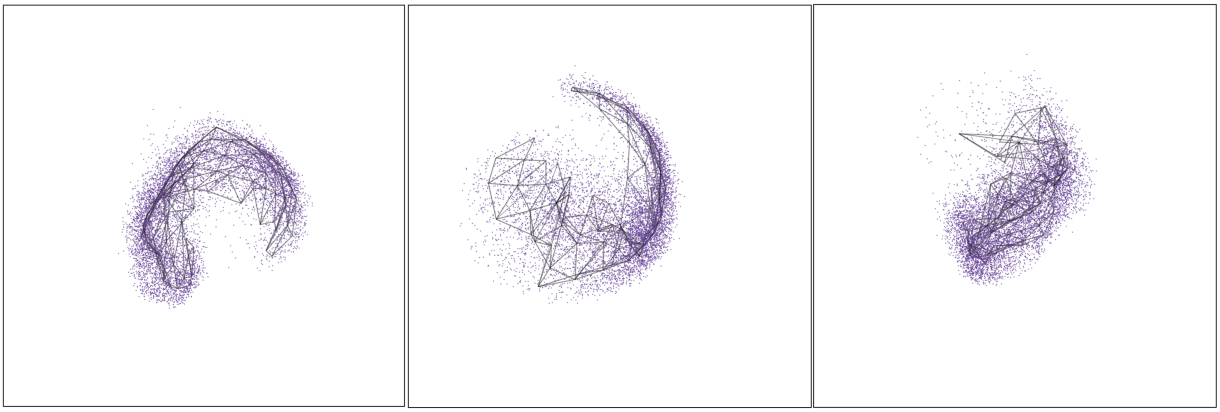


Figure 32: (a) Model generated with 12 and 15 bins along x and y axes with hexagonal size 0.06 in the 2D space overlaid on PaCMAP data, and (b) high-D model error in model space.



(a) (b) (c)

Figure 33: Screen shots of the **langevitour** of the MNIST digit 1 data set, shows the model-in-data space, a video of the tour animation is available at (https://youtu.be/zq2GM9qvUNA).

In terms of model error, some data points show high error (see Figure **??**) due to their deviation from the typical high-D data structure, acting as outliers. This is evident when comparing the images associated with these high model error points (see Figure **??**) to others (see Figure **??**).

Figure 34: Images of handwritten digit 1 which occur large model error.

5 Conclusion

In conclusion, our proposed model offers a novel approach for visualizing high-dimensional (high-D) data by leveraging non-linear dimension reduction (NLDR) techniques. Through a series of steps including hexagonating NLDR data, triangulating bin centroids, and lifting the 2D triangular mesh into high dimensions, our model effectively transforms complex high-D data into interpretable 2D representations. The model's performance is evaluated using goodness of fit statistics such as Mean Squared Error (MSE) and Akaike Information Criterion (AIC), providing insights into its accuracy and reliability. Overall, our model presents a valuable tool for researchers and practitioners in various fields to gain deeper insights from high-dimensional datasets.

Our algorithm mainly consists…

These Goodness of fit statistics useful in encountering the prediction accuracy when we predict the 2D embedding for the new high-D data. Because, we try to find the nearest high-D mappings as the first step of prediction which will participate the prediction process. When we need to find which different NLDR technique and which parameter choices is giving the best representation of high-D data, MSE and AIC are also useful.

We introduce a new tool to help to determine which method, which parameter choice provide the most useful representation of high-D data.

References