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 11, 2024

Abstract

Non-Linear Dimension Reduction (NLDR) techniques have emerged as powerful tools to visualize high-dimensional data. However, their complexity and parameter choices may lead to distrustful or misleading results. To address this challenge, we propose a novel approach that combines the tour technique with a low-dimensional manifold generated using NLDR techniques, hexagonal binning, and triangulation. This integration enables a clear examination of the low-dimensional representation in the original high-dimensional space. Our approach not only preserves the advantages of both tours and NLDR but also offers a more intuitive perception of complex data structures and facilitates accurate data transformation assessments. The method and example data sets are available in the **quollr** R package.

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 (Guo et al. 2023), due to advancements in data collection technologies (Johnstone & Titterington 2009, Ayesha et al. 2020). However, visualization of high-D data introduces significant challenges, because the complexity of visualizing data beyond two dimensions (Jia et al. 2022). In recent years, interactive and dynamic graphics systems like **liminal** (Lee et al. 2020) —which employs interactive tools like brushing and linking (Wickham et al. 2015)—and software tools such as **XGobi**, **GGobi** (Swayne et al. 1998), **tourr** (Wickham et al. 2011), **detourr** (Hart & Wang 2022), and **langevitour** (Paul Harrison 2022), involving dynamic methods like tours (Asimov 1985), have played a key role in visualizing high-D data (data-vis).

To create low-dimensional representations (typically in 2D) (m-vis) (Buja et al. 1996) 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) (F.R.S. 1901), non-linear methods such as multi-dimensional scaling (MDS) (Torgerson 1967). In the past decade, many new non-linear dimension reduction (NLDR) techniques have emerged, such as t-distributed stochastic neighbor embedding (tSNE) (van der Maaten & Hinton 2008) and uniform manifold approximation and projection (UMAP) (McInnes & Healy 2018). 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** (Lee et al. 2020). 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. The Section 2 provides an detailed overview of dimension reduction methods, and tours. Building upon this foundation, the Section 3

delves into the proposed algorithm, its implementation details, how to tune the model, model summaries, and a synthetic example to illustrate the functionality of the algorithm. Subsequently, Section 4 showcases applications of the algorithm on different data sets, particularly in single-cell RNA-seq data. These applications reveal insights into the performance and trustworthiness of NLDR algorithms. We analyze the results to identify situations where NLDR techniques may lead to misleading interpretations. Finally, **?@sec-conclusions** concludes by summarizing the findings and emphasizing the significance of the proposed approach in tackling the challenges of high-dimensional data visualization.

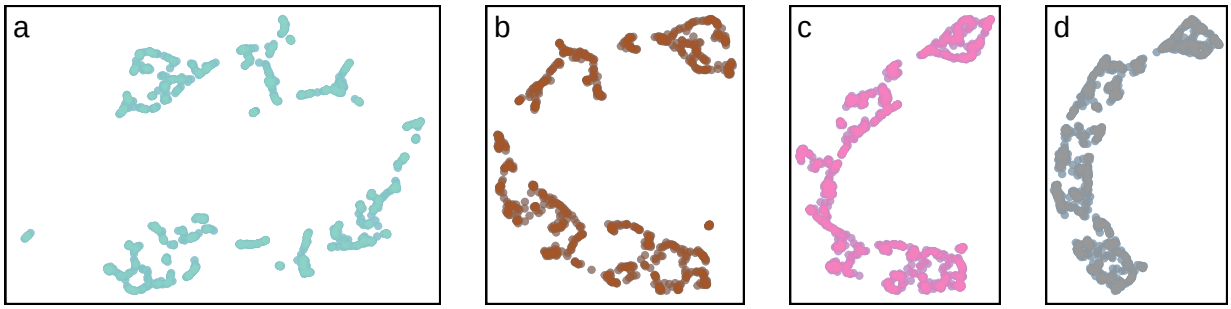


Figure 1: 2D layouts from UMAP applied for the S-curve data: (a) UMAP (n_neighbors = 7), (b) UMAP (n_neighbors = 15), (c) UMAP (n_neighbors = 32), (d) UMAP (n_neighbors = 50). Is there a best hyperparameter choice in representing UMAP or are they all providing equivalent information?

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 (F.R.S. 1901).

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 (Torgerson 1967). Variants of MDS include non-metric scaling (Kruskal 1964) and Isomap, which estimate geodesic distances to create the low-dimensional representation (Silva & Tenenbaum 2002). Other approaches based on diffusion processes, like diffusion maps (Coifman et al. 2005) and the PHATE (Potential of Heat-diffusion for

Affinity-based Trajectory Embedding) algorithm (Moon et al. 2019), 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 (Johnstone & Titterington 2009). One of the challenges with NLDR techniques is the selection and tuning of appropriate hyperparameters (Liao et al. 2023). 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 (Rumelhart et al. 1986) 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 (Amid & Warmuth 2022), and Pairwise Controlled Manifold Approximation (PaCMAP) (Wang et al. 2021).

Among these, tSNE (van der Maaten & Hinton 2008) 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 (McInnes & Healy 2018) 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 (Moon et al. 2019) 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 (Amid & Warmuth 2022) 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 (Wang et al. 2021) 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.

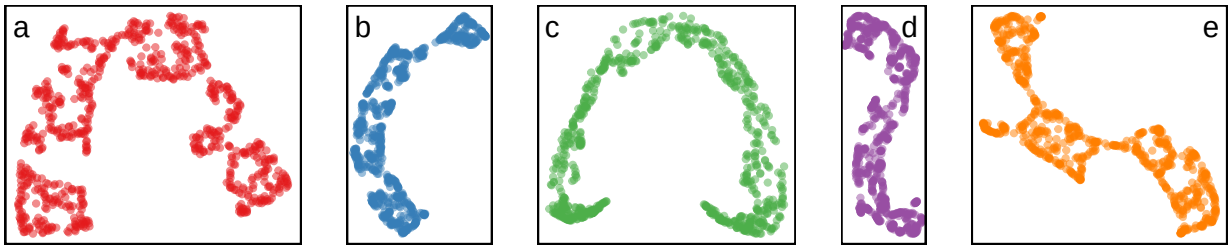


Figure 2: 2D layouts from different NLDR techniques applied the same data: (a) tSNE (perplexity = 27), (b) UMAP (n_neighbors = 50), (c) PHATE (knn = 5), (d) TriMAP (n_inliers = 5, n_outliers = 4, n_random = 3), and (e) PaCMAP (n_neighbors = 10, init = random, MN_ratio = 0.9, FP_ratio = 2). Is there a 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 (Asimov 1985) and the guided tour (Cook et al. 1995). A grand tour involves randomly selecting new orthonormal bases, enabling users to understand the structure by exploring the subspace of d-dimensional projections (Asimov 1985). In contrast, a guided tour can be employed to generate a sequence of 'interesting' projections based on an index function (Cook et al. 1995).

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 3. 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.

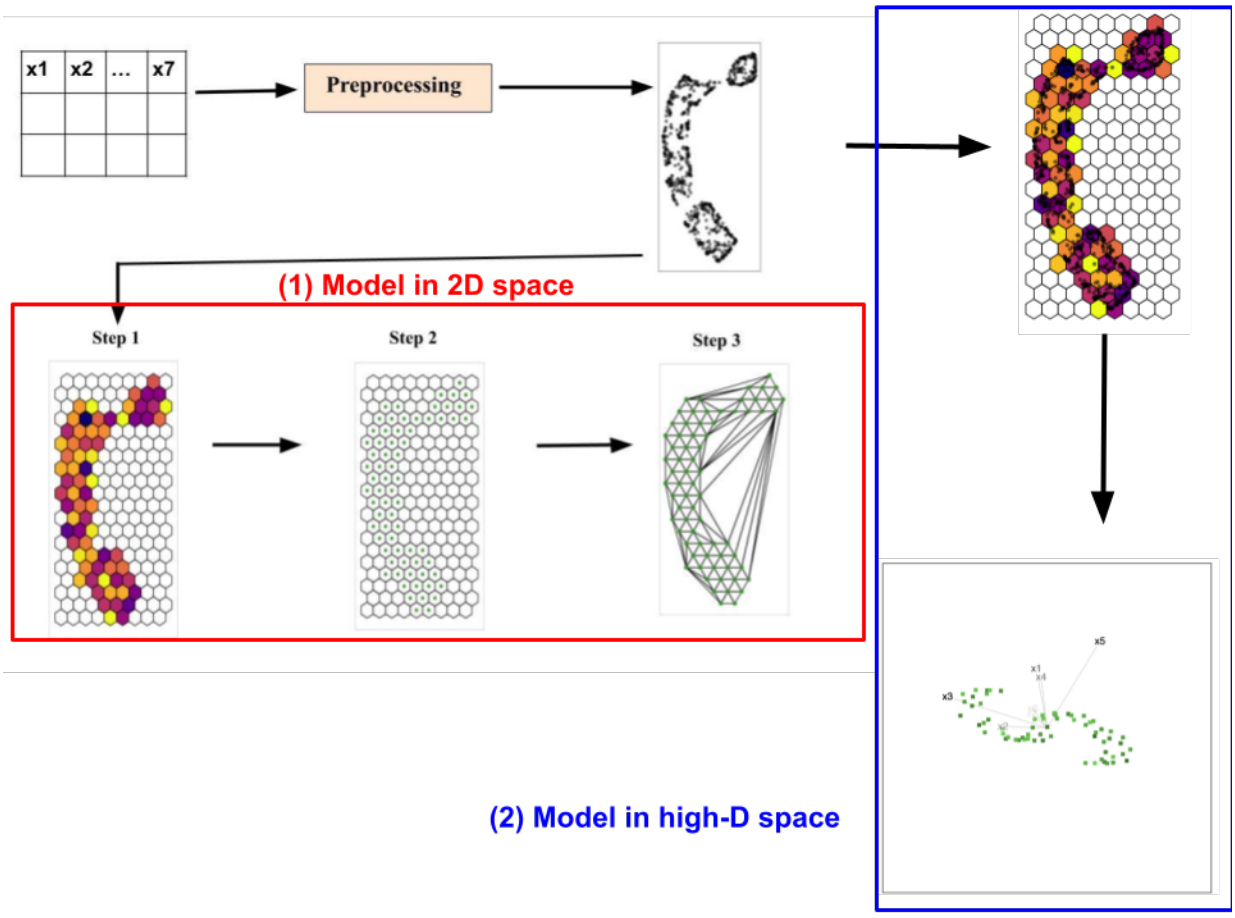


Figure 3: The flow diagram shows the main steps of our algorithm. There are two basic phases, one to generate the model in the 2D space, and other to map the model into the high-D space.

3.1 Preprocessing steps

To reduce computational complexity when applying NLDR techniques to high-D data and to reduce noise presence, PCA (Jolliffe & Cadima 2016, Howley et al. (2005), Indhumathi & Sathiyabama (2010)) 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 4).

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

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

$$y_{max} = \text{ceiling}(\frac{ar}{hr}) * hr \tag{3}$$

Step 2: Hexagonating NLDR data

(a) Determine the number of bins along the x axis (b_1)

(b) Determine the number of bins along the y axis (b_2)

(c) Determine the total number of bins (b)

(d) Assign NLDR data to hexagons

Step 3: Obtaining bin centroids or bin means

The mathematical expression for the hexagonal bin centers can be defined based on the hexagonal grid coordinates. Let's denote the hexagonal grid coordinates as ((q, r)), where (q) and (r) are integers representing the coordinates of the hexagon in the grid.

For a regular hexagonal grid, the coordinates of the hexagon centers can be calculated using the following formulas:

Where: - (s) is the side length of the hexagon. - (q) represents the column index of the hexagon in the grid. - (r) represents the row index of the hexagon in the grid. - (x) and (y) are the Cartesian coordinates of the center of the hexagon.

This formula calculates the (x) and (y) coordinates of the center of each hexagon based on its grid coordinates (q) and (r), allowing you to determine the position of each hexagon in the plane.

Step 4: Triangulating bin centroids or bin means

Delaunay triangulation is a geometric algorithm used to create a triangular mesh from a set of points in a plane. The key principle behind Delaunay triangulation is to connect points in such a way that no point falls within the circumcircle of any triangle formed by the points.

Mathematically, given a set of points (P) in a plane, the Delaunay triangulation constructs a set of triangles such that:

1. Each triangle is formed by connecting three points from (P).
2. No point in (P) is inside the circumcircle of any triangle in the triangulation.

Formally, the Delaunay triangulation can be defined as follows:

Let (P = {p_1, p_2, ..., p_n}) be a set of (n) points in the plane. The Delaunay triangulation of (P), denoted as (DT(P)), is a triangulation of the convex hull of (P) such that the circumcircle of every triangle in the triangulation contains no other points from (P).

Step 1: Computing the hexagonal grid configuration

The hexagonal grid, formed through hexagonal binning (Carr et al. 1987, Carr (1992)), serves as a type of bivariate histogram employed to visualize the structure of high-D data. Hexagons, being one of only three regular polygons capable of tessellating a plane (Carr et al. 2013), possess both symmetry of nearest neighbors and the maximum number of sides for a regular tessellation of the plane. This unique combination makes hexagons more efficient in covering the plane compared to other regular tessellations. Additionally, hexagons exhibit lower visual bias when displaying densities, setting them apart from other regular tessellations (Carr et al. 2023). In our algorithm, hexagonal binning is used as the initial step of constructing the 2D model and the total number of bins (b) is the crucial parameter that defines the granularity of the hexagonal grid.

(a) Determine the number of bins along the x axis (b_1)

First, the number of bins along the x-axis (b_1) is computed using the relationship between the diameter (h) and the area (A) of regular hexagons (see Equation 4).

In the process of computing hexagonal bin configurations, the first step involves determining the number of bins along the x-axis. This begins with the initialization of parameters such as the hexagonal size (s) and the buffer along the x-axis. Subsequently, the range of the first 2D embedding component (r_1) is calculated to ensure comprehensive coverage of the data. This range is then adjusted by incorporating the buffer amount, effectively expanding the boundary to accommodate potential outliers or edge cases. The default buffer amount along the x-axis is $\sqrt{3} * s * 1.5$. Next, the horizontal spacing (h) between hexagons is computed based on the hexagon size. Finally, the number of bins along the x-axis is computed based on the adjusted range ($r_1 + buffer_x$) and the horizontal spacing (h).

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

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

(b) Determine the number of bins along the y-axis (b_2)

Next, the number of bins along the y-axis is computed based on the number of bins along the x-axis (b_1) and the shape parameter (s) (see Equation 5) (Carr et al. 2013).

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

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

(c) Determine the total number of bins (b)

The total number of bins is determined by multiplying the number of bins along the x-axis (b_1) with the number of bins along the y-axis (b_2) (see Equation 8).

$$b = b_1 \times b_2 \tag{8}$$

Step 2: Obtain bin centroids or bin means

As a result of hexagonal binning for high-D data, all the high-D data are clustered into hexagons. In this step, the bin centroids ($C_k^{(2)} \equiv (C_{ky_1}, C_{ky_2})$) (see Figure 3 Step 2) are obtained (Carr et al. 2013).

Step 3: Triangulate bin centroids

In this step, the algorithm proceeds to triangulate the hexagonal bin centroids (see Figure 3 Step 3). 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 (Lloyd 1977). One common algorithm used for triangulation is Delaunay triangulation (Lee & Schachter 1980, Renka (1996)), 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.

Since we are working with the centroids of regular hexagonal bins, the resulting mesh will predominantly comprise equal-sized regular triangles. However, the triangulation also helps span any gaps that may exist between clusters of points, allowing for a more complete and interconnected representation of the data.

3.3 Lifting the model into high dimensions

3.3.1 Lifting the triangular mesh points into high dimensions

Step1: Cluster 2D points to hexagons

Expanding upon the information regarding hexagonal binning discussed in Step 1 of Section 3.2, the primary objective in this step is to determine the 2D embedding points associated with each hexagon. As the initial process of hexagonal binning, the 2D points are clustered into their respective hexagonal bins. By mapping this information with the hexagonal bin centroids ($C_k^{(2)} \equiv (C_{ky_1}, C_{ky_2})$) that obtained in Step 2 of the 2D model building (see Section 3.2), we can find which 2D points are assigned to which data set (see **?@fig-wkhighD** (a)).

Step2: Cluster high dimensional points to hexagons

Following the step 1, the main focus in this step to determine the corresponding high dimensional points for each hexagon. Every 2D embedding point serves as a projection of a data point belonging to high dimensional space. By using this mapping between the high dimensional data and their corresponding projections, the high dimensional points allocated to each hexagons are determined (see video linked in **?@fig-wkhighD**).

Step3: Compute the mean within hexagon

Having identified the high-dimensional points associated with hexagons, the final step involves computing the mean within each hexagonal bin. This implies calculating the average of the high-dimensional data points located within each hex bin. These averaged high-dimensional data points, denoted as $C_k^{(p)} \equiv (C_{kx_1}, ..., C_{kx_p})$, serve as the representative coordinates for the hex bin centroids within the expansive high-dimensional space (see video linked in **?@fig-wkhighD**).

3.3.2 Lifting the 2D triangular mesh into high dimensions

Based on the insights gained in Step 3 of Section 3.2, where connected edges in the 2D triangular mesh were identified, this step involves lifting these connections into the high dimensions. Having the mappings of all 2D triangular mesh points into high dimensions, the points connected in 2D are also connected in high dimensional space (see video linked in **?@fig-wkhighD**).

3.4 Tuning the model

The performance and robustness of our model depend on three key parameters: (i) the total number of bins (b), (ii) a benchmark value used to remove low-density hexagons, and (iii) a benchmark value used to remove long edges. However, there is no analytical formula to calculate an appropriate value for these parameters (see Appendix for details on how default values are calculated). The selection of these parameter values depends on the model performance computed by Mean Squared Error (MSE) (see Section 3.5.2).

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. This is because it is the initial step in building the 2D model. The hexagonal grid with the chosen total number of bins must be able to 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 **?@fig-binsize** (a)), while if there are too many bins, it may result in over-fitting the individual points of the NLDR data (see **?@fig-binsize** (c)). Therefore, it is important to determine an appropriate number of bins to build an effective model.

Furthermore, the total number of bins is determined by the number of bins along the x-axis and y-axis (see **?@eq-equation13**). To calculate the effective total number of bins, the minimum and approximate maximum number of bins along the x-axis are used as candidate values. The minimum number of bins along the x-axis is 1. The maximum number of bins along the x-axis can be estimated by taking the square root of the number of NLDR data

points. To investigate the impact of the number of bins on MSE, the analysis is focused on the MSE for different total numbers of bins. This range encompasses the total number of bins calculated for the minimum and maximum values of the number of bins along the x-axis.

Typically, the MSE plot will have a steep slope at the beginning, indicating that a smaller number of bins causes a larger amount of error. Then, the slope will gradually decline or level off, indicating that a higher number of bins generates a smaller error. This MSE plot is useful in determining the effective number of bins required to construct the 2D model. **?@fig-diagnosticpltScurve** (b) has created a graph that shows how the Mean Squared Error (MSE) changes depending on the number of bins assigned in the hexagonal grid for various NLDR techniques applied to the S-curve data set. As per the rule of thumb mentioned earlier, the effective number of bins for each NLDR method is shown in the **?@tbl-msebinsnldr**. Furthermore, **?@fig-modelScurve** shows the 2D models constructed for each NLDR technique with the selected total number of bins and how the models look in high-D with original data.

3.4.2 Benchmark value to remove low-density hexagons

After setting up the hexagonal grid with an appropriate number of bins, it is possible that some hexagonal bins may have very few or no data points within them. To ensure comprehensive coverage of the NLDR data, it is necessary to select hexagonal bins that have a considerable number of data points within their hexagons. To achieve this, the standard number of points within each hexagon is first calculated. This standard count is obtained by dividing the number of points within each hexagon by the maximum number of points in the grid (see Equation 9). Next, the bins that have a standard number of points less than a certain benchmark value are removed. After removing the hexagons that have insufficient data density, the hexagons with more substantial data representation are used to construct the 2D model.

$$\text{standard count} = \frac{\text{count}}{\text{max count}} \tag{9}$$

Furthermore, it is crucial to choose the benchmark value to remove low-density hexagons carefully. If we remove unnecessary bins, then it may result in long edges and an uneven 2D model. Therefore, instead of removing hexagons only identified by the benchmark value, we examine the standard number of points in the neighboring hexagons of the identified low-density hexagons. If the neighboring bins also have low counts, then only those bins will be removed. The other identified bins will be kept and used for constructing the 2D model along with the high-density bins (see Appendix for more details).

The benchmark value for removing low-density hexagons has a range of 0 and 1. When analyzing how these benchmark values affect model performance, it's important to observe the change in Mean Squared Error (MSE) as the benchmark value increases. The MSE exhibits a gradual increase as the benchmark value progresses from 0 to 1. Assessing this rate of increase is crucial. If the increment is not substantial, the decision might lean towards retaining low-density hexagons. According to **?@fig-diagnosticpltScurvelwd**,

the change in MSE for NLDR techniques is relatively small across different benchmark values, except for the PHATE method.

Consequently, it is not necessary to remove low-density hexagons when constructing the 2D model for the S-curve dataset in NLDR techniques, except for PHATE. While there may be some fluctuations in MSE between benchmark values of 0 to 1, the absence of significant changes suggests that maintaining low-density hexagons is preferable. However, to choose the best selection, it's recommended to examine the benchmark values around the first local minimum in the MSE curve. However, the MSE for PHATE fluctuates when benchmark values range between 0 and 1. Therefore, it is necessary to explore the benchmark values surrounding the first local minimum to determine an effective benchmark value. There is no universal rule for selecting a process. As shown in **?@fig-diagnosticpltScurvelwd**, the MSE values for PHATE fluctuate, and the first local minimum is at 0. Therefore, nearby benchmark values such as 0.01, 0.04, and 0.06 were explored to identify the best value for preserving the data structure. In this case, a benchmark value of 0.06 was chosen. The resulting 2D model is shown in **?@fig-modelScurve** (c), while the video link provides a visual representation of how the model appears in high-D.

The following steps will help to find a suitable value to remove low density hexagons:

1. Plot the distribution of the standardized counts
2. See the distribution of counts
3. Take the first quartile

3.4.3 Benchmark value to remove long edges

Achieving a smooth representation in 2D space is crucial, and one factor influencing this smoothness is the presence of long edges (see **?@fig-modelScurvermlgimp**). These long edges occur when a line connects points that are distant in the triangular mesh, impacting the overall smoothness of the mesh.

To investigate the effect of removing long edges on the MSE, we analyzed the MSE for various benchmark values. Surprisingly, the results indicated that the MSE remained consistent across different benchmark values, as shown in **?@fig-diagnosticpltScurvelgrm**. It's important to note that edges are defined only in the 2D model and do not extend to higher dimensions. Consequently, removing edges does not affect the model in higher dimensions.

There is no definitive rule for determining the benchmark value to remove long edges. However, we have a method to find a default value (see Appendix). Adjusting values around this default can help to find benchmark value to remove long edges, contributing to the construction of a smoother 2D representation (see **?@fig-diagnosticpltScurvelgrm**).

The following steps will help to find a suitable value to remove long edges:

1. Plot the distribution of the 2D distances
2. Find a value which is greater than smallest value

3.4.4 Starting point of the hexagonal grid

According to Carr et al. (2023), the hexagonal binning is done by tessellating the xy plane over the set (range(x), range(y)) (see **?@fig-scurveshifthexgridsexp** (b)). In that case, bin centroids are defined as shown in **?@fig-scurveshifthexgridsexp** (a) with gray colour. Rather than sticking to the typical hexagonal grid, introducing a meaningful shift in both the x and y directions presents an opportunity for an improved 2D model. Therefore, investigating this shift in the hexagonal grid is an important parameter to consider.

As shown in **?@fig-scurveshifthexgrids**, the shifting influences the distribution of points and number of non-empty bins, impacting the resulting 2D model. According to **?@fig-diagnosticpltScurvehexbins**, the 2D model with a total of 144 bins applied to S-curve UMAP data does not require any shifting because the lowest MSE occurs when no shift is introduced.

3.5 Model summaries

3.5.1 Predicted values and residuals

The prediction approach involves employing the K-nearest neighbors (KNN) algorithm. In this method, which operates as an unsupervised classification problem, the nearest high-D model point (averaged high-D point) is identified for the new high-D data point. Once the nearest high-D model point is identified, its coordinates are used as the predicted 2D embedding for the new high-D data point. This step is particularly valuable due to the limitations of certain NLDR techniques, like tSNE, which don't provide a straightforward method for predicting 2D embedding. As a result, our approach presents a solution that can generate predicted 2D embedding, irrespective of the NLDR technique employed, effectively addressing this gap in functionality.

The concept of "residuals" is pivotal in evaluating the accuracy of representing bin centroids in high dimensions. To quantify this accuracy, we introduce an error metric, which measures the sum of squared differences between the high-dimensional data (x_{ij}) and the predicted bin centroid data in high-dimensional space (C_{x_ij}) across all bins and dimensions. Mathematically, this error is expressed as:

$$\text{Error} = \sum_{j=1}^{n} \sum_{i=1}^{p} (x_{ij} - C_{x_ij})^2 \tag{10}$$

Here, n represents the number of observations, p represents the dimensions, x_{ij} is the actual high-dimensional data, and C_{x_ij} is the predicted bin centroid data in high dimensions.

The error metric outlined above provides valuable insights into the overall accuracy of our predictive model. By quantifying the squared deviations between the actual and predicted values across all bins and dimensions, we gain a comprehensive understanding of how well our method captures and represents the underlying structure of the data in the reduced 2D space. This assessment is crucial for evaluating the efficacy of our NLDR technique in preserving the essential information present in the original high-dimensional data.

3.5.2 Goodness of fit statistics

Moving on to the assessment of prediction accuracy, we calculate the Mean Squared Error (MSE). The MSE helps measure the average squared differences between the actual high-dimensional data (x_{ij}) and the predicted bin centroid data in high-D (C_{x_ij}) values across all bins. Mathematically, this is expressed as:

$$\text{MSE} = \sum_{j=1}^{n} \frac{\sum_{i=1}^{p} (x_{ij} - C_{x_ij})^2}{n} \tag{11}$$

Here, b' signifies the total number of bins without empty bins, p denotes the number of dimensions in the high-dimensional data, and n represents the number of observations.

Additionally, we gauge the model's performance using the Akaike Information Criterion (AIC), calculated by the formula:

$$\text{AIC} = 2b'p + np * log(\text{MSE}) \tag{12}$$

These metrics, MSE and AIC, collectively offer valuable insights into the model's predictive performance, considering both accuracy and complexity in the predictions.

3.6 Simulated data example

In this section, we showcase the effectiveness of our methodology using simulated data. The dataset comprises five spherical Gaussian clusters in 4-d, with each cluster containing an equal number of points and consistent within variation.

In the 2D representations created by all NLDR techniques, as shown in Figure 4, except for PHATE, there are five distinct clusters. In tSNE, these five clusters are closely located to each other (see Figure 4 (a)). However, in PHATE, two clusters are closely positioned, while the other three are more distant (see Figure 4 (c)). In PaCMAP, one cluster is at the center, and the remaining four are positioned in four different directions (see Figure 4 (e)). In TriMAP, two clusters are close, although not as close as in PHATE, and the other three are well-separated (see Figure 4 (d)). In UMAP, all clusters are arranged in a parallel manner, with three in one line and the other two in a separate line (see Figure 4 (b)).

Visualizing the models alongside the original high-D data provides insights into how different techniques capture the underlying clustering structure. The tSNE model exhibits five well-separated clusters, effectively preserving both local and global structures (see video link of **?@fig-modelfiveGau** (a)). UMAP, on the other hand, also presents five distinct clusters but with a more flattened surface appearance (see video link of **?@fig-modelfiveGau** (b)). The PHATE model shows five separated clusters resembling triangles or partial triangles but struggles to capture local structures (see video link of **?@fig-modelfiveGau** (c)). TriMAP and PaCMAP both display five well-separated clusters with flat surfaces, each capturing within-cluster variation to varying extents. Despite the various 2D representations, all NLDR techniques preserve the global structure (see video link of **?@fig-modelfiveGau**

(d), (e) respectively). However, tSNE, effectively capture both local and global structures, as indicated by lower AIC values in Figure 5 (a).

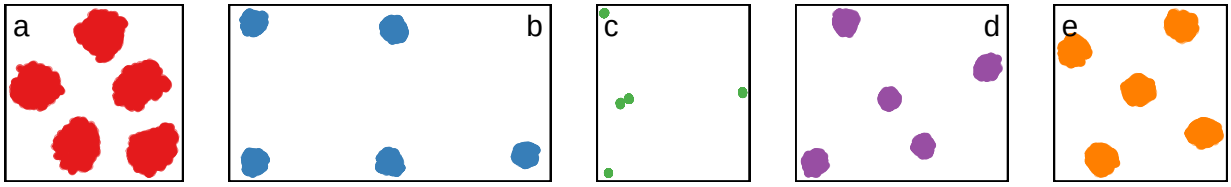


Figure 4: 2D layouts from different NLDR techniques applied the same data: (a) tSNE (perplexity = 61), (b) UMAP (n_neighbors = 15), (c) PHATE (knn = 5), (d) TriMAP (n_inliers = 5, n_outliers = 4, n_random = 3), and (e) PaCMAP (n_neighbors = 10, init = random, MN_ratio = 0.9, FP_ratio = 2). Is there a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

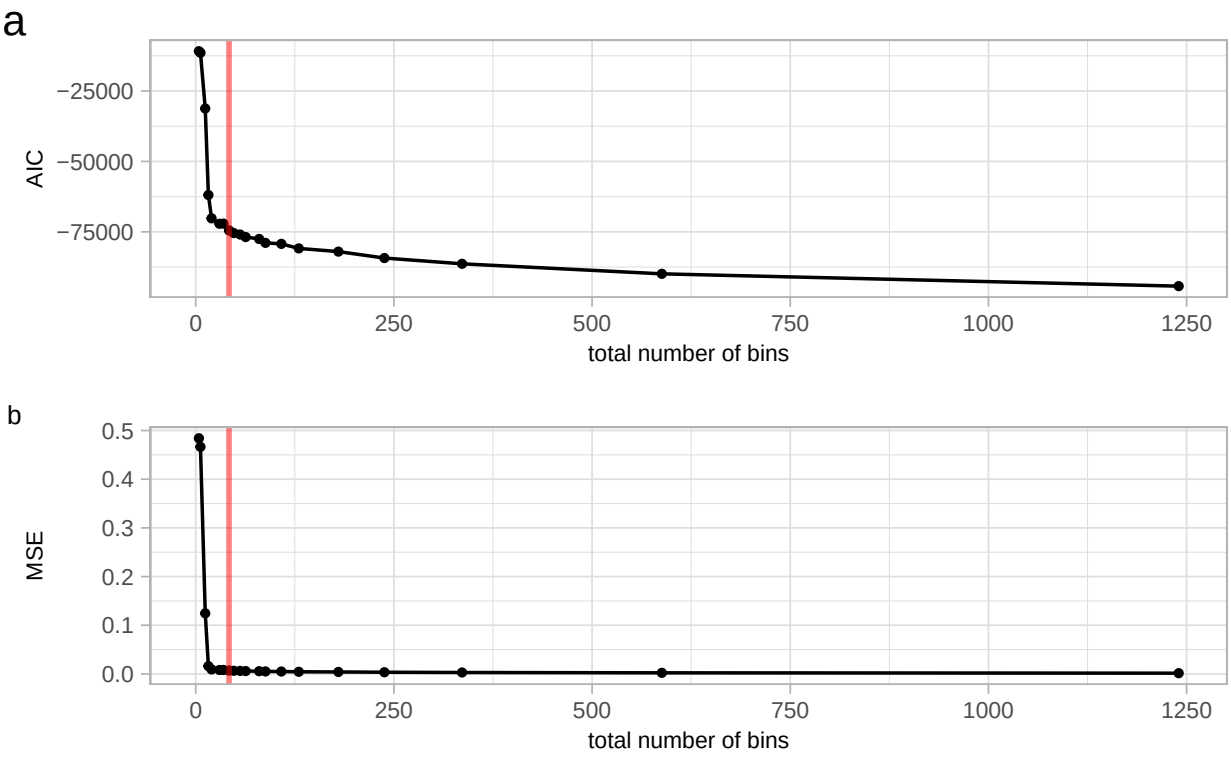
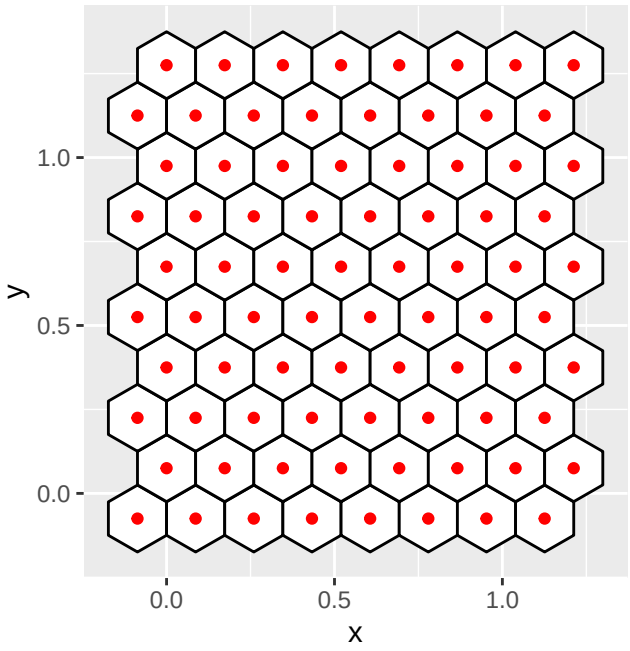
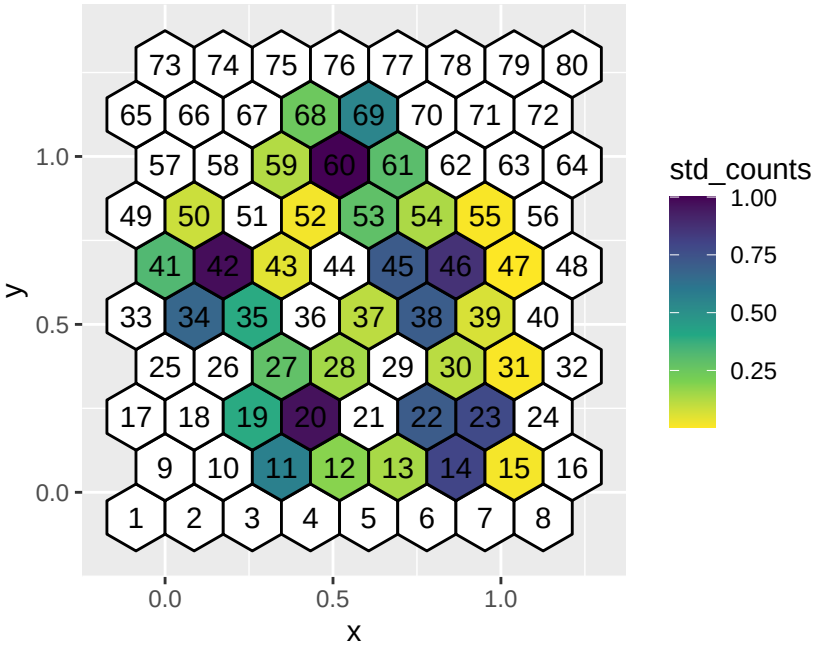
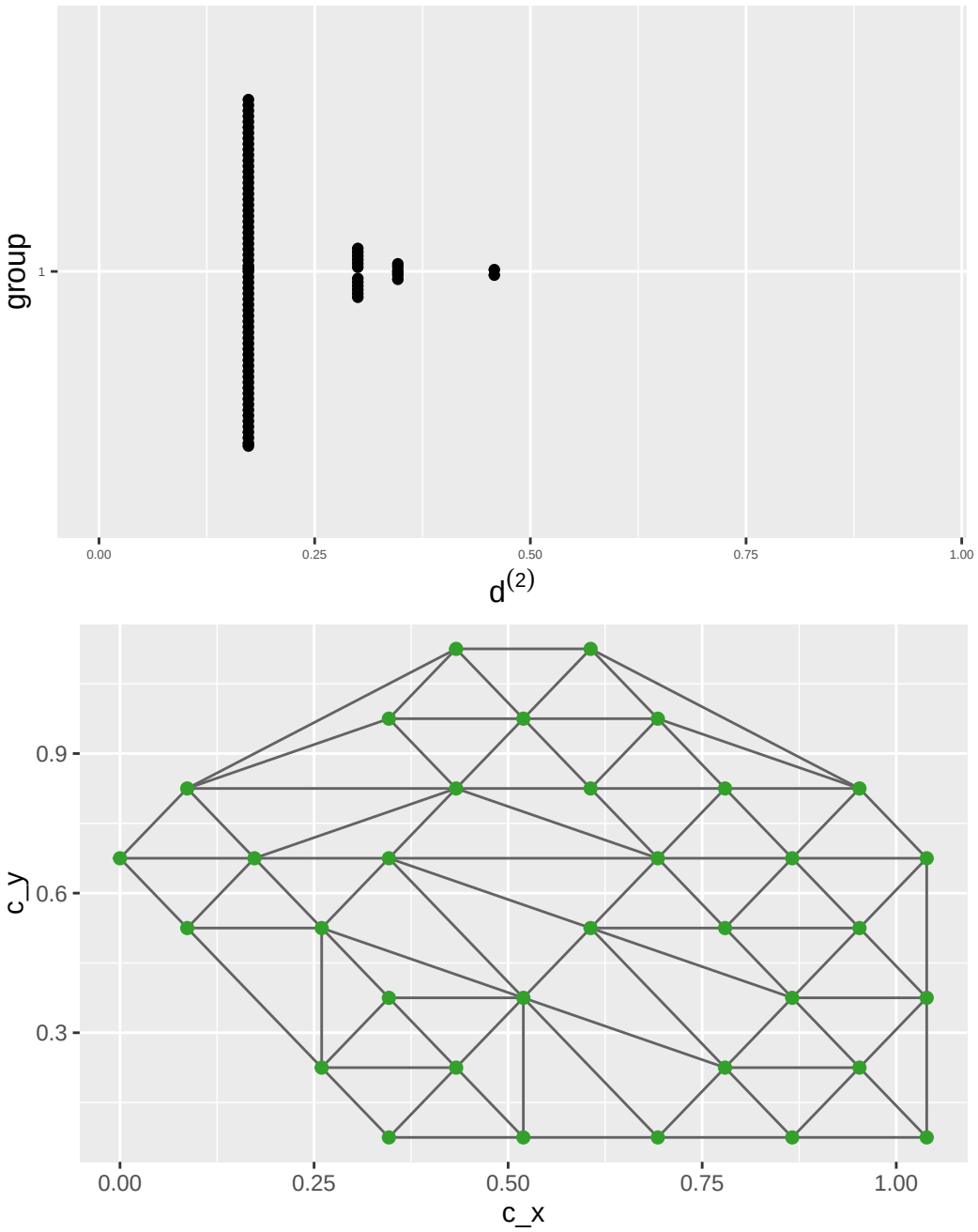


Figure 5: Goodness of fit statistics from different NLDR techniques applied to training five spherical Gaussian cluster dataset. What is the best NLDR technique to represent the original data in 2D?


1.0
0.5
0.0
y
0.0
0.5
1.0
x


73 74 75 76 77 78 79 80
65 66 67 68 69 70 71 72
57 58 59 60 61 62 63 64
49 50 51 52 53 54 55 56
41 42 43 44 45 46 47 48
33 34 35 36 37 38 39 40
25 26 27 28 29 30 31 32
17 18 19 20 21 22 23 24
9 10 11 12 13 14 15 16
1 2 3 4 5 6 7 8
1.0
0.5
0.0
y
0.0
0.5
1.0
x
std_counts
1.00
0.75
0.50
0.25



4 Applications

4.1 Single-cell RNA-seq data of human

In the field of single-cell studies, a common analysis 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. In clustering workflows, the main objective is to verify the existence of clusters and subsequently identify them as specific cell types by examining the expression of "known" marker genes. Marker genes are specific genes or transcripts that are uniquely or highly expressed in particular cell types. Marker genes are identified through differential gene expression (DEG) analysis,

comparing the expression levels of genes across different cell types. In this context, a "faithful" embedding should ideally preserve the topology of the data, ensuring that cells corresponding to the same cell type are situated close to the high-dimensional space.

4.2 Processing steps

To begin our analysis, we installed the Peripheral Blood Mononuclear Cells (pbmc) data set obtained from 10x Genomics using the `SeuratData` R package (Satija et al. 2019), which facilitates the distribution of data sets in the form of Seurat objects (Hao et al. 2021). This data set contains 13,714 features (genes) across 2,700 samples (cells) within a single assay. The active assay is RNA, with 13,714 features representing different gene expressions. Then, the cells that have unique feature counts over 2,500 or less than 200 and cells that have $> 5\%$ mitochondrial counts are filtered. After removing unwanted cells from the dataset, the next step is to normalize the data. By default, we employ a global-scaling normalization method "LogNormalize" that normalizes the feature expression measurements for each cell by the total expression, multiplies this by a scale factor (10,000 by default), and log-transforms the result.

Next, top 1000 genes are selected by Festem (Chen et al. 2023) DEG method. Then, scale the data. Then, perform PCA on the scaled data. Louvain algorithm is used to cluster the single cells based on the genes selected by Festem. Using the Festem-selected genes and 15 PCs, identified 10 clusters in the PBMC3k data and annotated them based on the expression of canonical markers. The 10 clusters included immune cells such as naive CD4 T cells, memory CD4 T cells, CD8 T cells, CD14 monocytes, FCGR3A monocytes, Natural Killer (NK) cells, B cells and Dendritic Cells (DC). In addition to these common cell types, Festem identified a fine cell type, CD27− CD4+ memory T cells, which were often missed by other methods.

- Human peripheral blood mononclear cells (PBMCs) data
- Intro to the data set

Used Louvain algorithm to cluster the single cells based on the genes selected by Festem method

- Explain what "Evaluate Festem on DEG detection" means, in terms of this data
- How did they "identified 10 clusters"? What were the variables used, PCs or full data or 2D representation, spell this out.
- Used top 1000 genes selected by Festem to run PCA, top 15 PCs (original paper mentioned)
- The "10 clusters included immune cells such as naive CD4 T …" does this mean that every cluster had all of these, or one cluster was mostly "CD4 T cells", another mostly "memory CD4 T", …?
- How is your use of this data different from the original application.
- Original paper used this data set to compare different DEG methods, but we need to look how the performance of UMAP with the author's selected parameter choice.

- Why 15 PCs? Is that from the original paper? YES

Chen et al. (2023)

- Intro to the data set (This data set contains 13,714 features across 2,700 samples within a single assay. The active assay is RNA, with 13,714 features representing different gene expressions.)
- 2622 cells
- Evaluate Festem on DEG detection (Festem enabled identification of often-missed fine cell types)
- Using the Festem-selected genes, identified 10 clusters in the PBMCsk data and annotated them based on the expression of canonical markers
- The 10 clusters included immune cells such as naive CD4 T cells, memory CD4 T cells, CD8 T cells, CD14 monocytes, FCGR3A monocytes, Natural Killer (NK) cells, B cells and Dendritic Cells (DC).
- In addition to these common cell types, Festem identified a fine cell type, CD27− CD4+ memory T cells, which were often missed by other methods.
- The CD27. CD4+ memory T cells identified by Festem expressed common marker genes (IL7R and S100A4) of memory T cells, but did not express CD27. These cells also had downregulated expression of SELL, CCR7, MAL and LEF1, and upregulated expression of CCL5 (Fig. 4B) and thus were the CD27- CD4+ memory T cells in the literature (36). The CD27- CD4+ memory T cells were known to be at a more differentiated state and have stronger antigen-recall responses than their CD27+ counterparts.
- 15 PCs
- Num bins along the x-axis = 23, shape parameter = 0.8772751
- learned from the model: There are 3 well-separated clusters in 2D, but if look at the model in high-D space, the three clusters are much closer. Also, there is some continuity within the clustering structure and it didn't capture well. Therefore UMAP with n:neighbour 30 is not a good representation for this data set.
- The most important parameter is n_neighbors - the number of approximate nearest neighbors used to construct the initial high-dimensional graph. It effectively controls how UMAP balances local versus global structure - low values will push UMAP to focus more on local structure by constraining the number of neighboring points considered when analyzing the data in high dimensions, while high values will push UMAP towards representing the big-picture structure while losing fine detail.
- The second parameter we'll investigate is min_dist, or the minimum distance between points in low-dimensional space. This parameter controls how tightly UMAP clumps points together, with low values leading to more tightly packed embeddings. Larger values of min_dist will make UMAP pack points together more loosely, focusing instead on the preservation of the broad topological structure.

- The third parameter is metric. This parameter plays a crucial role in determining how distances are computed within the ambient space of the input data (The term "ambient space" refers to the original or initial space in which the data exists before any transformation or dimensionality reduction).
- If it is completely a mess, then that is also further evidence that the fit is poor in high-dimensions.

Table 1: errors for different parameter combinations

n_neighbors	min_dist	metric	error (x 100)
5	0.99	cosine	262.18
15	0.99	cosine	279.49
84	0.99	cosine	295.64
5	0.30	cosine	303.90
15	0.30	cosine	308.97
84	0.30	cosine	319.59
30	**0.30**	**cosine**	**386.35**

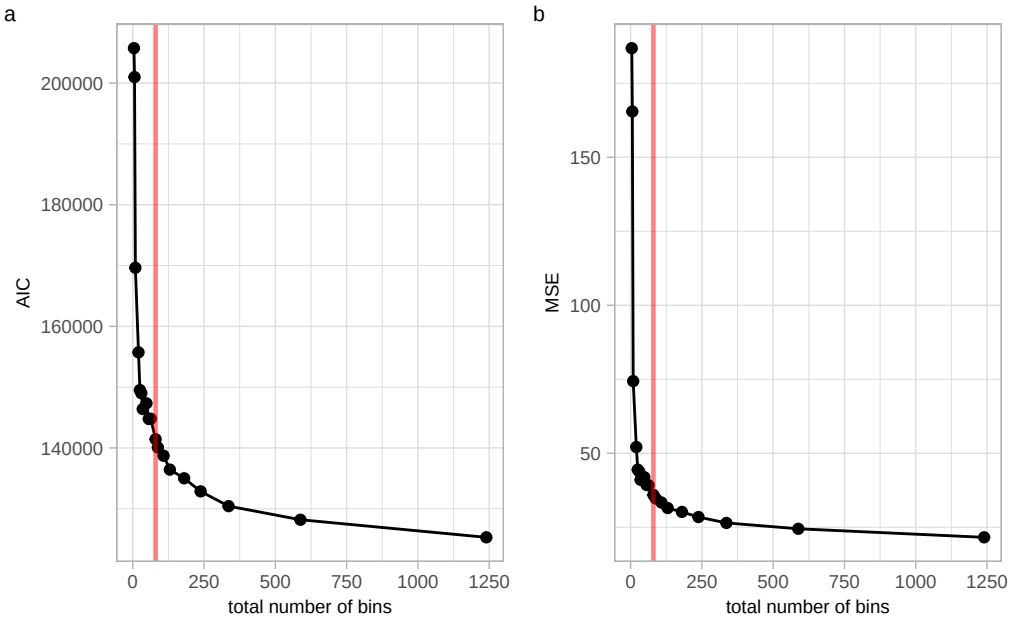
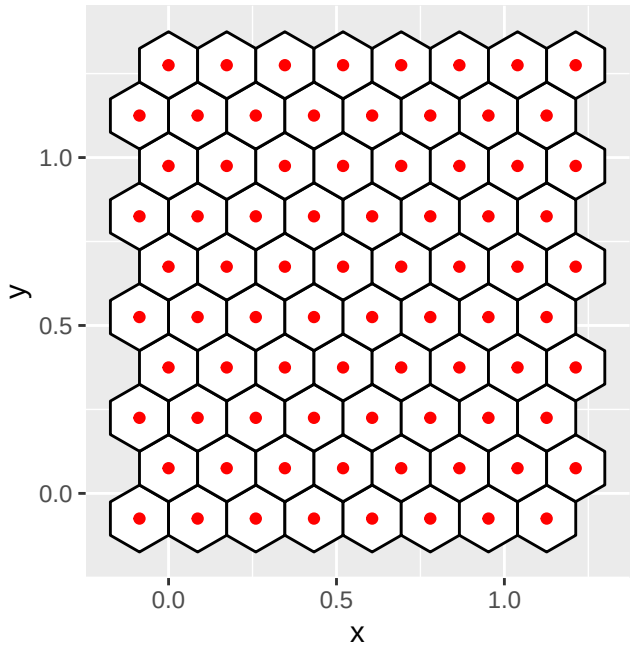
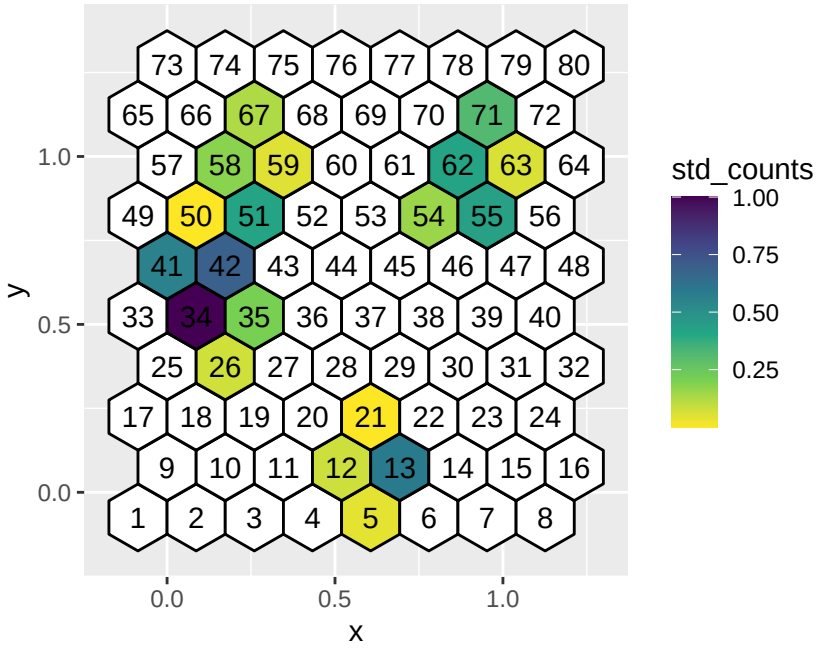
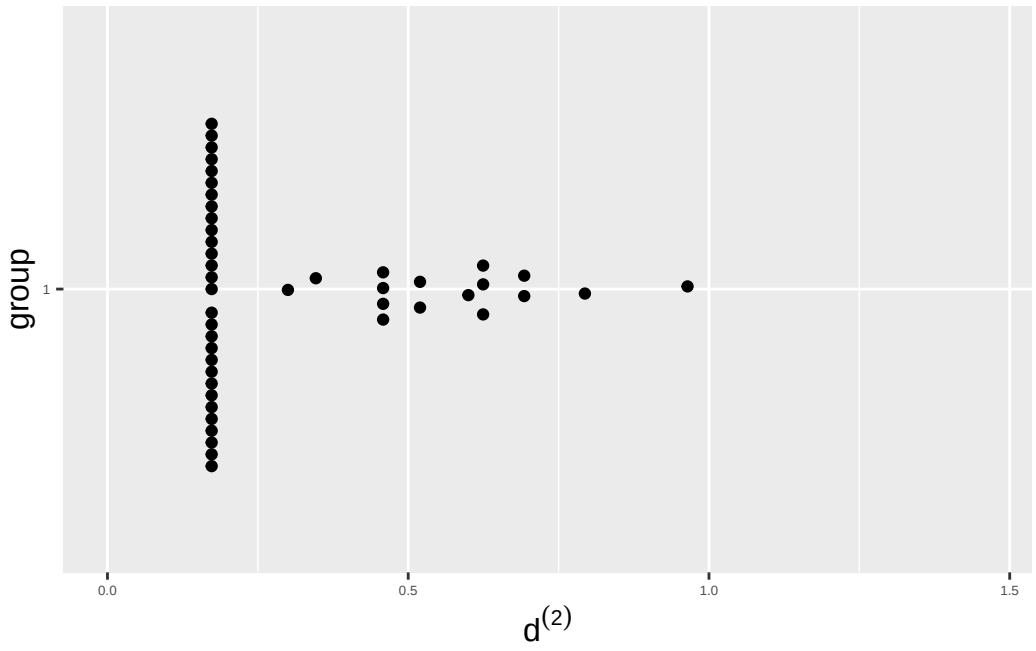
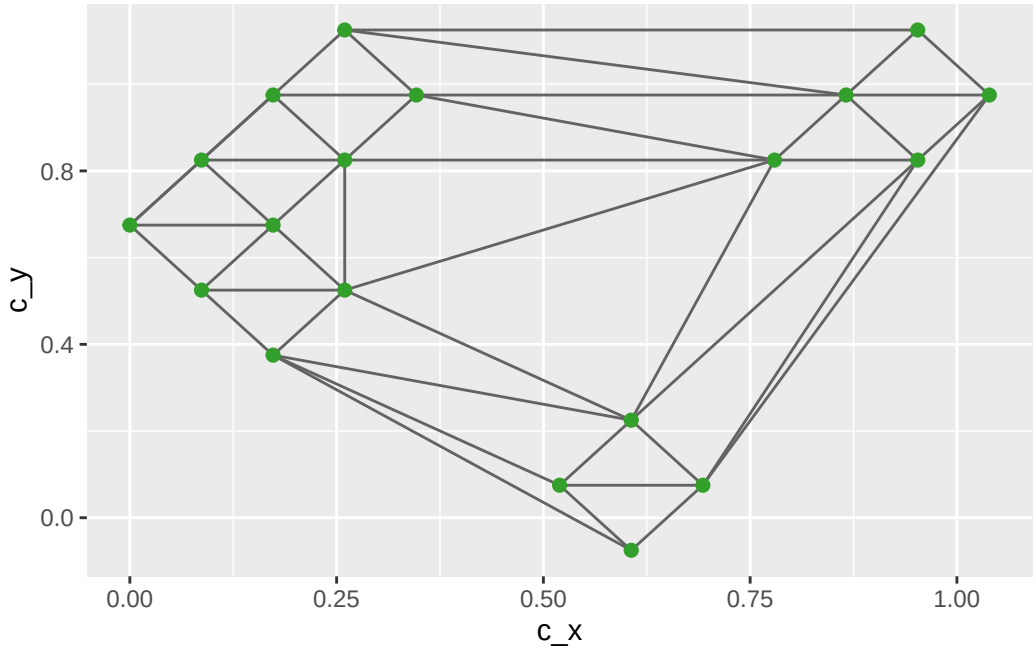


Figure 6: Goodness of fit statistics from UMAP applied to training S-curve dataset. What is the effective number of bins in each NLDR technique to create a 2D model? The MSE plot have a steep slope at the beginning, indicating that a smaller number of bins causes a larger amount of error. Then, the slope 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 144, 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.


1.0
0.5
0.0
y
0.0
0.5
1.0
x


73 74 75 76 77 78 79 80
65 66 67 68 69 70 71 72
57 58 59 60 61 62 63 64
49 50 51 52 53 54 55 56
41 42 43 44 45 46 47 48
33 34 35 36 37 38 39 40
25 26 27 28 29 30 31 32
17 18 19 20 21 22 23 24
9 10 11 12 13 14 15 16
1 2 3 4 5 6 7 8
1.0
0.5
0.0
y
0.0
0.5
1.0
x
std_counts
1.00
0.75
0.50
0.25


group
1
0.0
0.5
1.0
1.5
d(2)


c_y
0.8
0.4
0.0
0.00
0.25
0.50
0.75
1.00
c_x

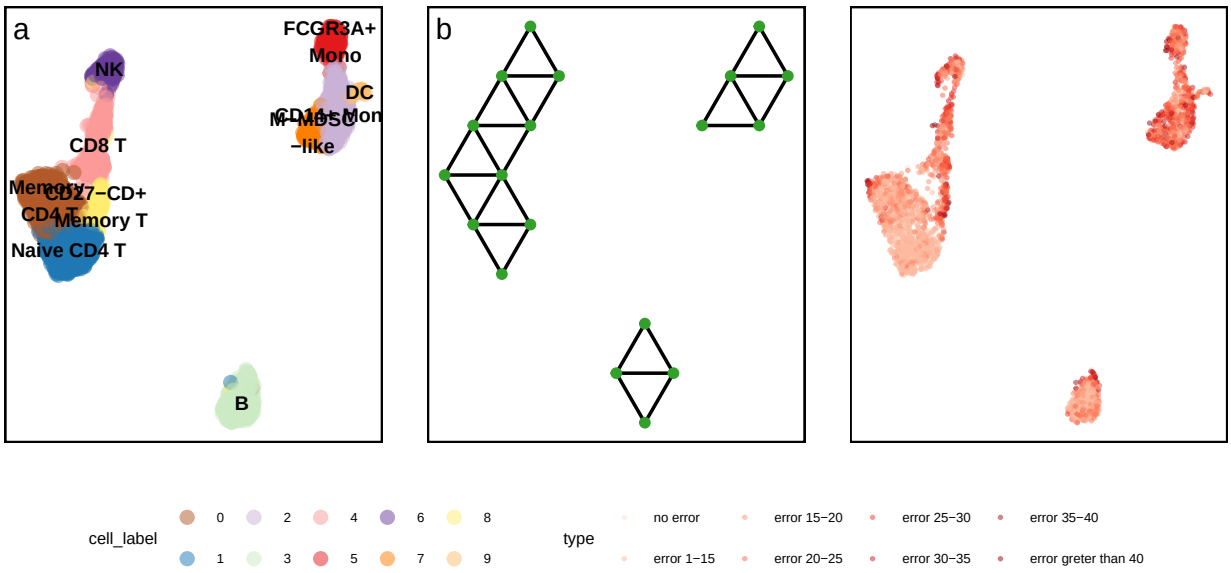
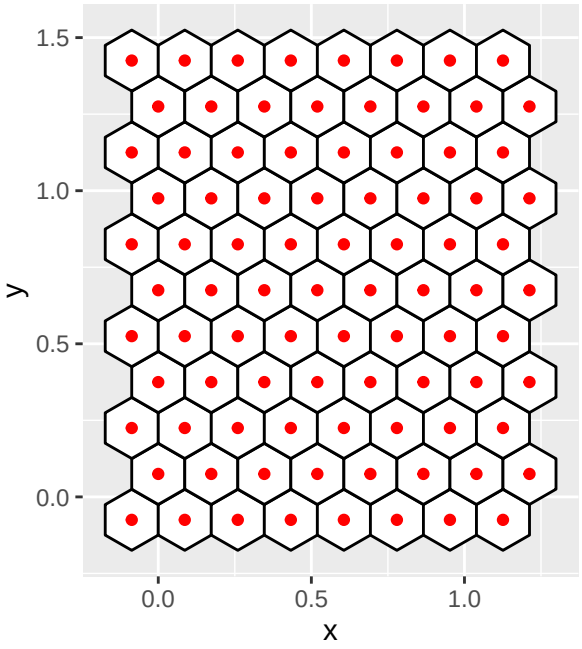


Figure 7: (a) 2D layout from UMAP (n_neighbors = 30, min_dist = 0.3) applied for the PBMC3k dataset. Is this a best representation of the original data?, (b) Model in the 2D space with UMAP (https://youtu.be/CNXjWZMyh_M)



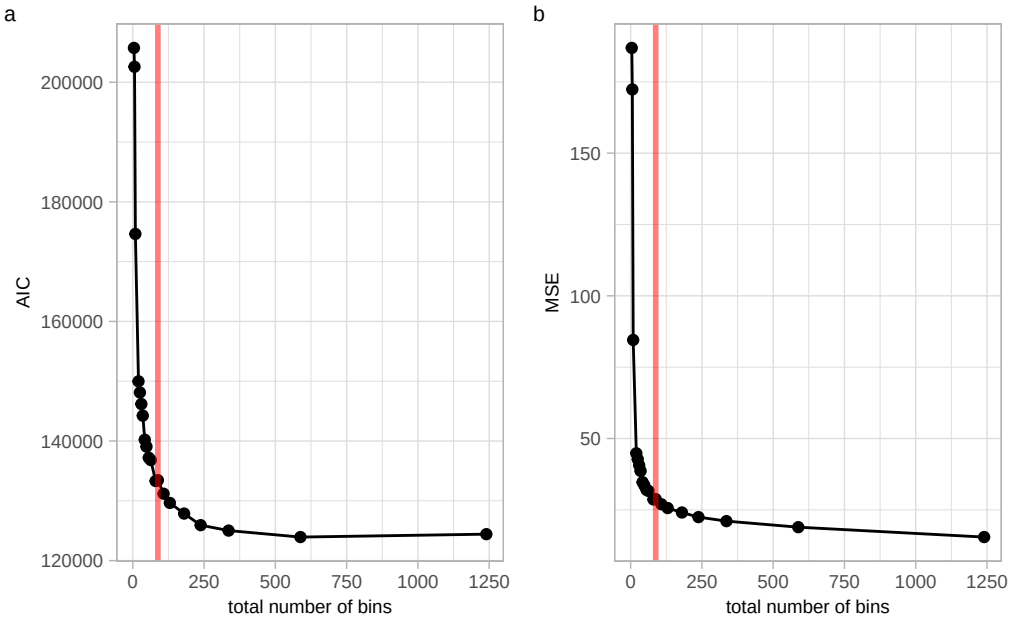
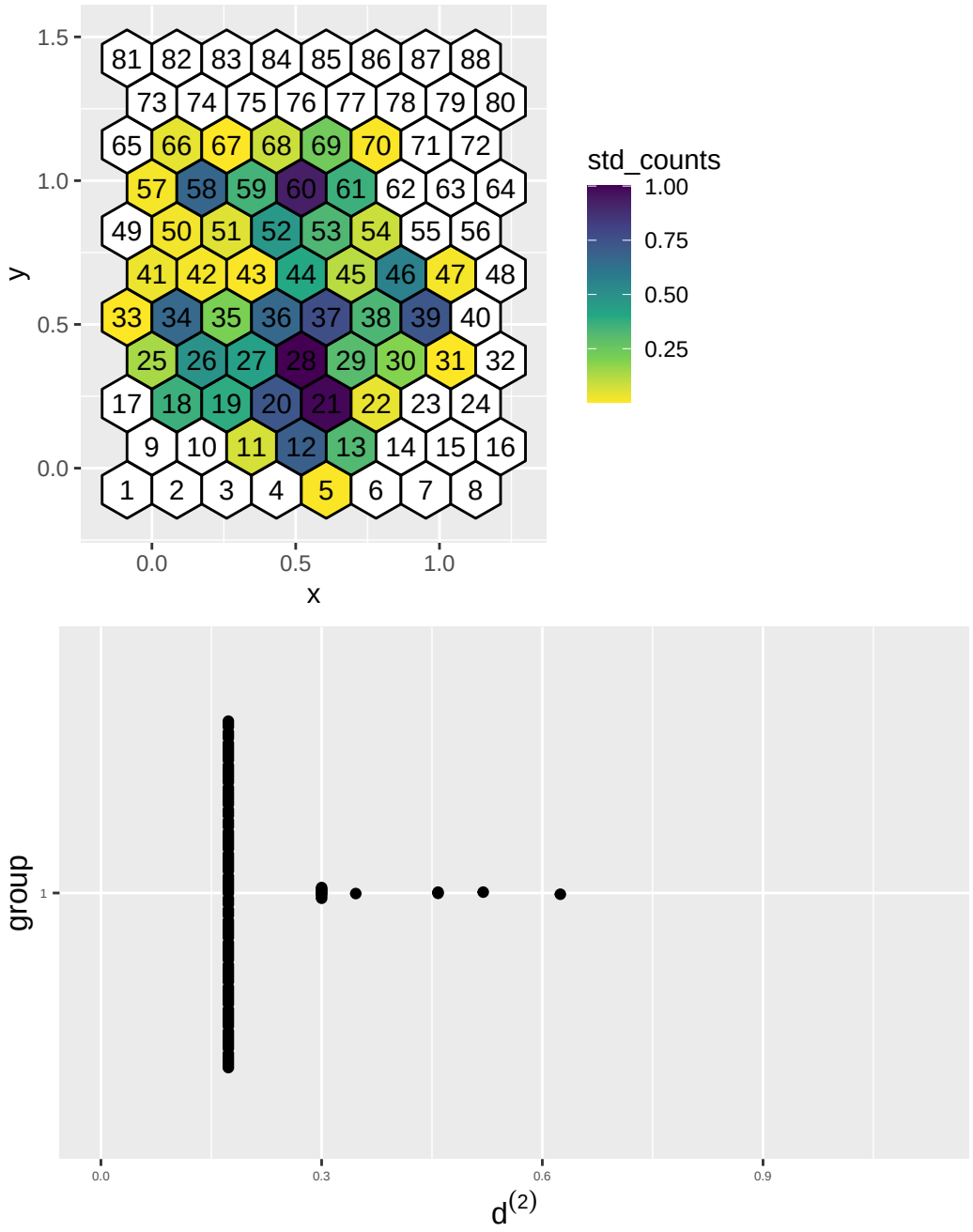


Figure 8: Goodness of fit statistics from UMAP applied to training S-curve dataset. What is the effective number of bins in each NLDR technique to create a 2D model? The MSE plot have a steep slope at the beginning, indicating that a smaller number of bins causes a larger amount of error. Then, the slope 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 144, 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.


1.5
1.0
0.5
0.0
y
0.0
0.5
1.0
x
81 82 83 84 85 86 87 88
73 74 75 76 77 78 79 80
65 66 67 68 69 70 71 72
57 58 59 60 61 62 63 64
49 50 51 52 53 54 55 56
41 42 43 44 45 46 47 48
33 34 35 36 37 38 39 40
25 26 27 28 29 30 31 32
17 18 19 20 21 22 23 24
9 10 11 12 13 14 15 16
1 2 3 4 5 6 7 8
std_counts
1.00
0.75
0.50
0.25
group
1
0.0
0.3
0.6
0.9
d(2)

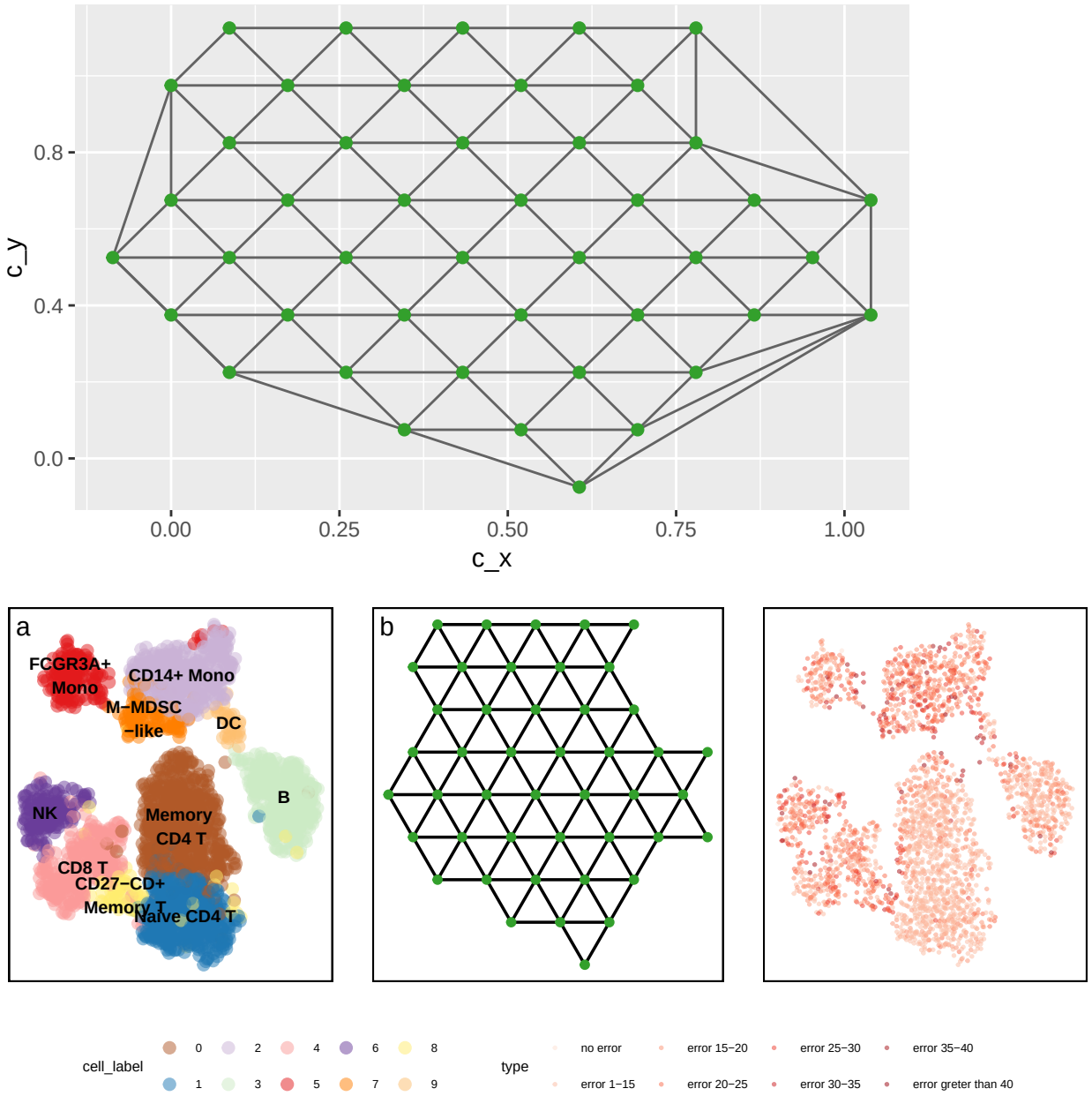
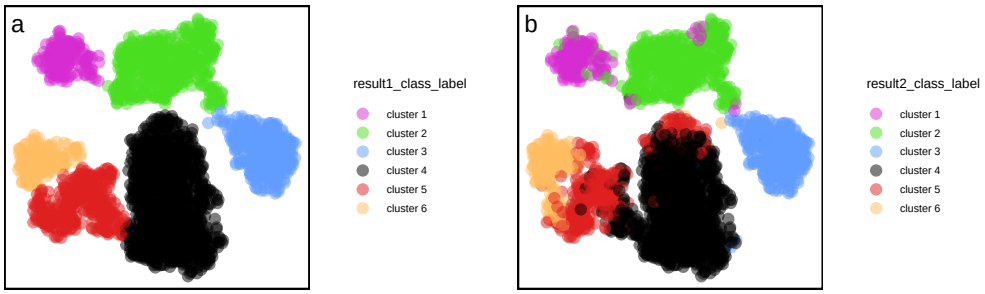


Figure 9: (a) 2D layout from UMAP (n_neighbors = 5, min_dist = 0.99, metric = cosine) applied for the PBMC3k dataset. Is this a best representation of the original data?, (b) Model in the 2D space with UMAP (<>)

	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5	cluster 6
cluster 1	138	13	0	0	0	0
cluster 2	11	509	0	1	0	0
cluster 3	1	0	348	1	0	1
cluster 4	0	0	3	1029	64	2
cluster 5	0	0	0	85	226	32
cluster 6	0	0	0	0	19	139



4.3 Handwritten digits

- Digit 1 (7877 images only)

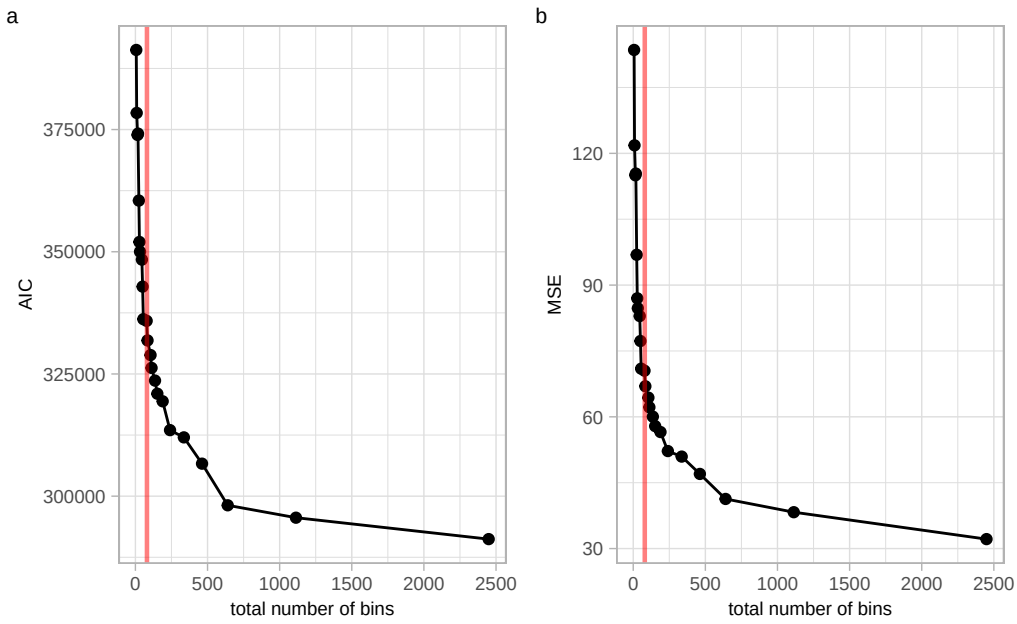
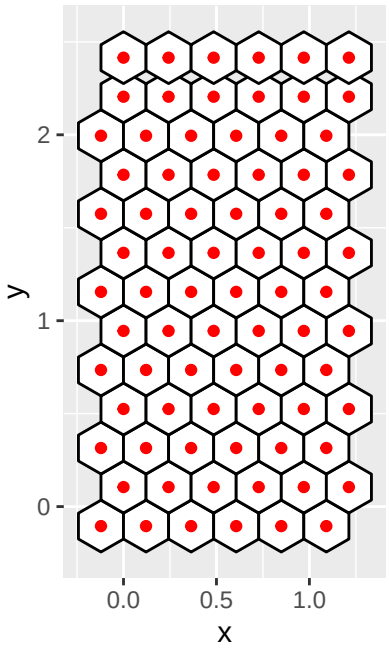
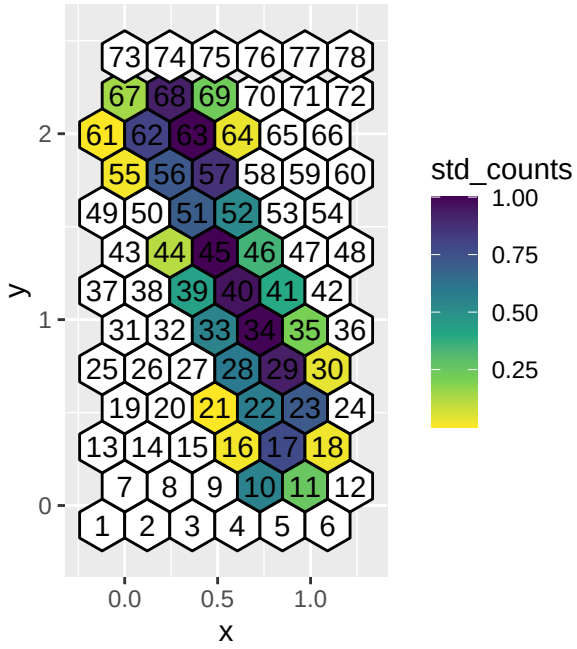
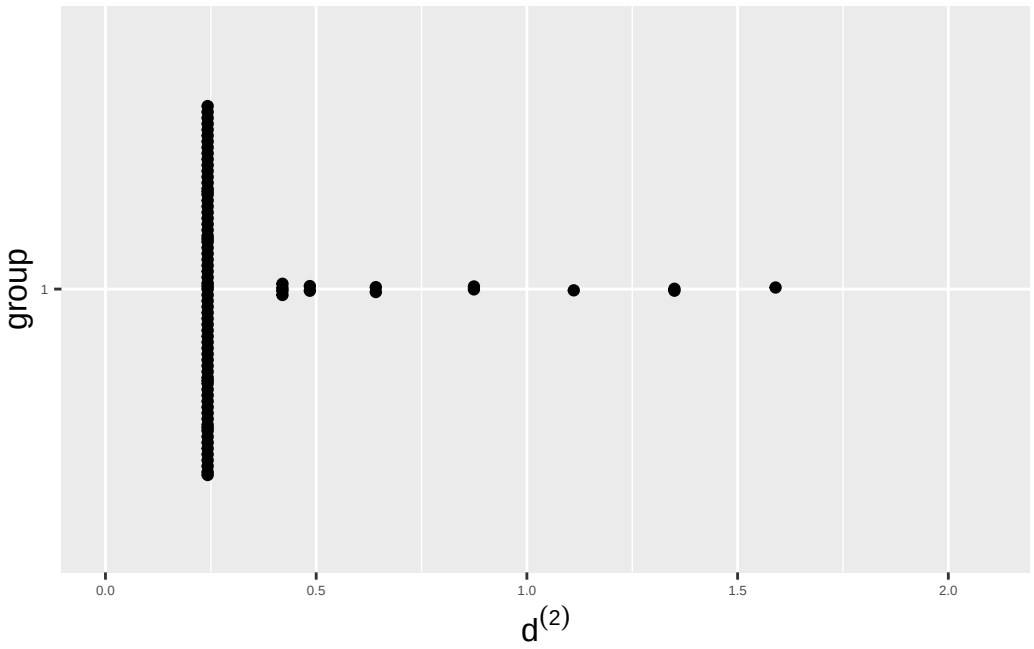
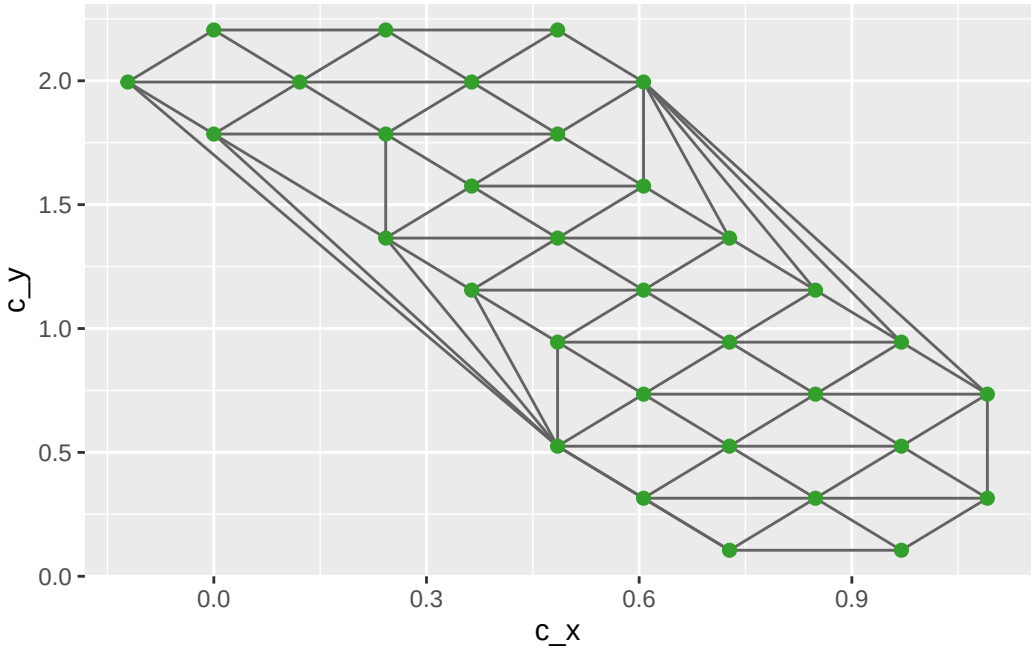


Figure 10: Goodness of fit statistics from UMAP applied to training S-curve dataset. What is the effective number of bins in each NLDR technique to create a 2D model? The MSE plot have a steep slope at the beginning, indicating that a smaller number of bins causes a larger amount of error. Then, the 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 144, 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.


2
1
0
y
0.0
0.5
1.0
x


73 74 75 76 77 78
67 68 69 70 71 72
61 62 63 64 65 66
55 56 57 58 59 60
49 50 51 52 53 54
43 44 45 46 47 48
37 38 39 40 41 42
31 32 33 34 35 36
25 26 27 28 29 30
19 20 21 22 23 24
13 14 15 16 17 18
7 8 9 10 11 12
1 2 3 4 5 6
2
1
0
y
0.0
0.5
1.0
x
std_counts
1.00
0.75
0.50
0.25





5 Conclusion

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

References

Amid, E. & Warmuth, M. K. (2022), 'Trimap: Large-scale dimensionality reduction using triplets'.

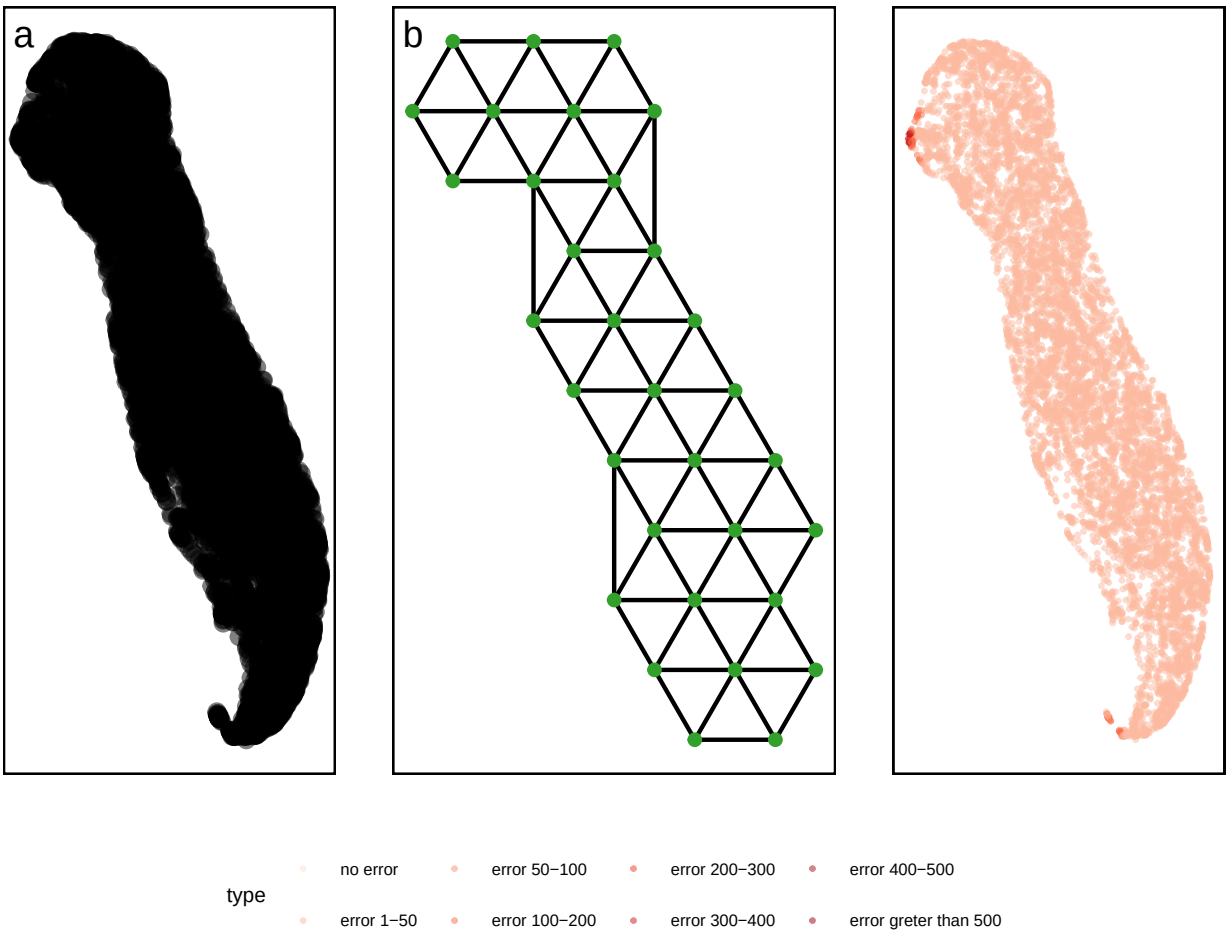


Figure 11: (a) 2D layout from UMAP (n_neighbors = 30, min_dist = 0.3) applied for the PBMC3k dataset. Is this a best representation of the original data?, (b) Model in the 2D space with UMAP (https://youtu.be/CNXjWZMyh_M)

Asimov, D. (1985), 'The grand tour: A tool for viewing multidimensional data', *SIAM Journal on Scientific and Statistical Computing* **6**(1), 128–143.
URL: *https://doi.org/10.1137/0906011*

Ayesha, S., Hanif, M. K. & Talib, R. (2020), 'Overview and comparative study of dimensionality reduction techniques for high dimensional data', *Information Fusion* **59**, 44–58.

Buja, A., Cook, D. & Swayne, D. F. (1996), 'Interactive high-dimensional data visualization', *Journal of Computational and Graphical Statistics* **5**(1), 78–99.
URL: *http://www.jstor.org/stable/1390754*

Carr, D. B. (1992), Looking at large data sets using binned data plots.

Carr, D. B., Littlefield, R. J., Nicholson, W. L. & Littlefield, J. S. (1987), 'Scatterplot matrix techniques for large n', *Journal of the American Statistical Association* **82**(398), 424–436.
URL: *http://www.jstor.org/stable/2289444*

Carr, D., Olsen, A. & White, D. (2013), 'Hexagon mosaic maps for display of univariate and

bivariate geographical data', *Cartography and Geographic Information Systems* **19**, 228–236.

Carr, D., ported by Nicholas Lewin-Koh, Maechler, M. & contains copies of lattice functions written by Deepayan Sarkar (2023), *hexbin: Hexagonal Binning Routines*. R package version 1.28.3.
URL: *https://CRAN.R-project.org/package=hexbin*

Chen, Z., Wang, C., Huang, S., Shi, Y. & Xi, R. (2023), 'Directly selecting differentially expressed genes for single-cell clustering analyses', *bioRxiv* .
URL: *https://www.biorxiv.org/content/early/2023/07/29/2023.07.26.550670*

Coifman, R., Lafon, S., Lee, A., Maggioni, M., Nadler, B., Warner, F. & Zucker, S. (2005), 'Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps', *Proceedings of the National Academy of Sciences of the United States of America* **102**, 7426–31.

Cook, D., Buja, A., Cabrera, J. & Hurley, C. (1995), 'Grand tour and projection pursuit', *Journal of Computational and Graphical Statistics* **4**(3), 155–172.
URL: *https://www.tandfonline.com/doi/abs/10.1080/10618600.1995.10474674*

F.R.S., K. P. (1901), 'Liii. on lines and planes of closest fit to systems of points in space', *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2**(11), 559–572.
URL: *https://doi.org/10.1080/14786440109462720*

Guo, B., Huuki-Myers, L. A., Grant-Peters, M., Collado-Torres, L. & Hicks, S. C. (2023), 'escheR: Unified multi-dimensional visualizations with Gestalt principles', *bioRxiv* . Publisher: Cold Spring Harbor Laboratory _eprint: https://www.biorxiv.org/content/early/2023/06/08/2023.03.18.533302.full.pdf.
URL: *https://www.biorxiv.org/content/early/2023/06/08/2023.03.18.533302*

Hao, Y., Hao, S., Andersen-Nissen, E., III, W. M. M., Zheng, S., Butler, A., Lee, M. J., Wilk, A. J., Darby, C., Zagar, M., Hoffman, P., Stoeckius, M., Papalexi, E., Mimitou, E. P., Jain, J., Srivastava, A., Stuart, T., Fleming, L. B., Yeung, B., Rogers, A. J., McElrath, J. M., Blish, C. A., Gottardo, R., Smibert, P. & Satija, R. (2021), 'Integrated analysis of multimodal single-cell data', *Cell* .
URL: *https://doi.org/10.1016/j.cell.2021.04.048*

Hart, C. & Wang, E. (2022), *detourr: Portable and Performant Tour Animations*. R package version 0.1.0.
URL: *https://casperhart.github.io/detourr/*

Howley, T., Madden, M., O'Connell, M.-L. & Ryder, A. (2005), The effect of principal component analysis on machine learning accuracy with high dimensional spectral data, pp. 209–222.

Indhumathi, R. & Sathiyabama, S. (2010), 'Reducing and clustering high dimensional data through principal component analysis', *International Journal of Computer Applications* **11**(8), 1–4.

Jia, W., Sun, M., Lian, J. & Hou, S. (2022), 'Feature dimensionality reduction: a review', *Complex & Intelligent Systems* **8**(3), 2663–2693.
URL: *https://doi.org/10.1007/s40747-021-00637-x*

Johnstone, I. M. & Titterington, D. M. (2009), 'Statistical challenges of high-dimensional data', *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **367**(1906), 4237–4253.
URL: *https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2009.0159*

Jolliffe, I. T. & Cadima, J. (2016), 'Principal component analysis: a review and recent developments', *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **374**(2065), 20150202.
URL: *https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2015.0202*

Kruskal, J. B. (1964), 'Nonmetric multidimensional scaling: a numerical method', *Psychometrika* **29**(2), 115–129.

Lee, D. T. & Schachter, B. J. (1980), 'Two algorithms for constructing a Delaunay triangulation', *International Journal of Computer & Information Sciences* **9**(3), 219–242.
URL: *https://doi.org/10.1007/BF00977785*

Lee, S., Laa, U. & Cook, D. (2020), 'Casting multiple shadows: High-dimensional interactive data visualisation with tours and embeddings'.
URL: *https://arxiv.org/abs/2012.06077*

Liao, Y.-T., Luo, H. & Ma, A. (2023), 'Efficient and robust bayesian selection of hyperparameters in dimension reduction for visualization'.

Lloyd, E. L. (1977), On triangulations of a set of points in the plane, *in* '18th Annual Symposium on Foundations of Computer Science (sfcs 1977)', pp. 228–240.

McInnes, L. & Healy, J. (2018), 'Umap: Uniform manifold approximation and projection for dimension reduction', *ArXiv* **abs/1802.03426**.

Moon, K. R., van Dijk, D., Wang, Z., Gigante, S. A., Burkhardt, D. B., Chen, W. S., Yim, K., van den Elzen, A., Hirn, M. J., Coifman, R. R., Ivanova, N. B., Wolf, G. & Krishnaswamy, S. (2019), 'Visualizing structure and transitions in high-dimensional biological data', *Nature Biotechnology* **37**, 1482 – 1492.

Paul Harrison (2022), 'langevitour: smooth interactive touring of high dimensions, demonstrated with scRNA-Seq data', *bioRxiv* p. 2022.08.24.505207.
URL: *http://biorxiv.org/content/early/2022/08/26/2022.08.24.505207.abstract*

Renka, R. J. (1996), 'Algorithm 751: Tripack: A constrained two-dimensional delaunay triangulation package', *ACM Trans. Math. Softw.* **22**(1), 1–8.
URL: *https://doi.org/10.1145/225545.225546*

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986), 'Learning representations by back-propagating errors', *Nature* **323**, 533–536.

Satija, R., Hoffman, P. & Butler, A. (2019), *SeuratData: Install and Manage Seurat Datasets*. http://www.satijalab.org/seurat, https://github.com/satijalab/seurat-data.

Silva, V. & Tenenbaum, J. (2002), 'Global versus local methods in nonlinear dimensionality reduction', *Advances in neural information processing systems* **15**.

Swayne, D. F., Cook, D. & Buja, A. (1998), 'Xgobi: Interactive dynamic data visualization in the x window system', *Journal of Computational and Graphical Statistics* **7**(1), 113–130.
URL: *http://www.jstor.org/stable/1390772*

Torgerson, W. S. (1967), *Theory and methods of scaling*, Wiley New York.

van der Maaten, L. & Hinton, G. E. (2008), 'Visualizing data using t-sne', *Journal of Machine Learning Research* **9**, 2579–2605.

Wang, Y., Huang, H., Rudin, C. & Shaposhnik, Y. (2021), 'Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization', *Journal of Machine Learning Research* **22**(201), 1–73.
URL: *http://jmlr.org/papers/v22/20-1061.html*

Wickham, H., Cook, D. & Hofmann, H. (2015), 'Visualizing statistical models: Removing the blindfold', *Statistical Analysis and Data Mining: The ASA Data Science Journal* **8**(4), 203–225.
URL: *https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.11271*

Wickham, H., Cook, D., Hofmann, H. & Buja, A. (2011), 'tourr: An r package for exploring multivariate data with projections', *Journal of Statistical Software* **40**(2), 1—18.
URL: *http://www.jstatsoft.org/v40/i02/*