MONASH University

New Interactive Visual Tools and Statistical Methodology for Selecting and Evaluating Nonlinear Dimension Reduction Layouts of High-Dimensional Data

Piyadi Gamage Jayani Lakshika
B.Sc. (Hons) in Statistics, University of Sri Jayewardenepura (USJ), Sri Lanka

A thesis submitted for the degree of
Doctor of Philosophy
at Monash University in 2026
Department of Econometrics & Business Statistics

Table of contents

Copyright notice

Abstract

High-dimensional data has many variables recorded for each observation, and is commonly visually summarized using dimension reduction. While linear methods such as principal component analysis are broadly applied, they cannot capture many types of nonlinear structures with a few dimensions. Nonlinear dimension reduction (NLDR) techniques address this limitation by applying nonlinear transformations to produce low-dimensional layout that make a summary that can display a variety of structures with a few dimensions. However, NLDR methods can also distort the structure, leading to a misunderstanding of the high-dimensional data. The main objective of this research is to develop new methods and software tools to diagnose, evaluate, and interpret NLDR results in relation to the structures present in high-dimensional data.

This research presents five original contributions. The first contribution (Chapter 2) introduces a new method for visualizing how NLDR warps data. This method improves the diagnostics of NLDR techniques. The second contribution (Chapter 3) involves implementing the method introduced in Chapter 2 as an R package, `quollr`. The third contribution (Chapter 4) introduces an R package, `cardinalR`, designed to generate high-dimensional clustering data structures, with features such as adding noise dimensions and background noise. The fourth contribution (Chapter 5) provides evidence in the identification of clusters at various distances when observing NLDR representation and the tour view of high-dimensional data. This finding is based on a human subject experiment that explores both the perception and misperception of NLDR representations. Finally, the fifth contribution (Chapter 6) provides a Shiny app that offers a user-friendly interface for analysts to obtain the most accurate NLDR representation. Overall, this work advances the field of diagnosing NLDR by improving the visualization of high-dimensional data.

Declaration

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

This thesis includes one paper that has been revised and resubmitted, two papers that have been submitted to a peer-reviewed journal, and two papers that are planned for future submission. The core theme of the thesis is to "develop methods and software to evaluate and understand nonlinear dimension reduction methods". The ideas, development, and writing up of all the papers in the thesis were the principal responsibility of me, the student, working within the Department of Econometrics and Business Statistics under the supervision of Professor Dianne Cook, Dr Paul Harrison (MGBP, BDInstitute), Dr Michael Lydeamore, and Dr Thiyanga S. Talagala (University of Sri Jayewardenepura).

Chapter 2 has been revised and resubmitted to the *Journal of Computational and Graphical Statistics*. Chapter 3 and chapter 4 have been submitted to *The R Journal*. Chapter 5 and chapter 6 are planned for submission to peer-reviewed journals.

To ensure the clarity and coherence of the written content, artificial intelligence tools were employed to assist in smoothing and refining the language throughout the thesis.

This thesis uses American spelling, as that's the style followed by the journals where the work will be submitted or published.

I have renumbered sections of submitted papers in order to generate a consistent presentation within the thesis.

Student name: Piyadi Gamage Jayani Lakshika

Student signature:

Date: 14th January 2026

I hereby certify that the above declaration correctly reflects the nature and extent of the student's and co-authors' contributions to this work. In instances where I am not the responsible author, I have consulted with the responsible author to agree on the respective contributions of the authors.

Main Supervisor name: Dianne Cook

Main Supervisor signature:

Date: 14th January 2026

Acknowledgements

This thesis would have never been a success without the guidance, encouragement, and support of my supervisors, family, and friends. This note is to show my gratitude to them.

First, I would like to sincerely thank my supervisors, Professor Dianne Cook, Dr Paul Harrison, Dr Michael Lydeamore, and Dr Thiyanga S. Talagala, for being incredible supervisors and mentors. Their endless enthusiasm for research, productivity, and wealth of knowledge has motivated me throughout my PhD at Monash University. No matter how busy they were, they always kept the doors open to welcome me whenever I needed their help. I believe that they are the best supervisors I could ever find in my life long research journey. Thank you for believing in my ability to navigate difficulties, even when I doubted myself. I am immensely grateful and honoured to have worked under the guidance of you all. Your expertise, way of thinking, and leadership will continue to inspire me now and always.

I gratefully acknowledge the constructive feedback, suggestions, and encouragement I received from my PhD milestone evaluation panel members: Professor Catherine Forbes, Professor Xibin Zhang, Associate Professor Ruben Loaiza Maya, Dr Jessica Leung, Dr Shanika Wickramasuriya, and Dr Kate Saunders. Special thanks go to Professor Catherine Forbes and Professor Xibin Zhang for being outstanding PhD directors and for their immense support throughout my candidature. I am also thankful to Professor Mervyn Silvapulle and Emeritus Professor Donald Poskitt, who taught our PhD coursework units during the first year of the programme.

I am thankful to Monash University, Australia, for this invaluable opportunity to pursue my PhD at such a prestigious institution. The exposure and experiences I gained over the years at Monash have been immense. I am particularly grateful for the financial support provided through the Co-funded Graduate Research Scholarship and the Monash Business School Co-Funded Graduate Research Scholarship, which enabled me to concentrate fully on my research. I am also thankful for the financial assistance provided to attend international conferences and visit some invaluable mentors in the USA (PhD external fund by the Department) and Austria. Also, the Prestigious International Conference Visit Scheme provided funding to attend the useR! 2024 conference. I take this opportunity to express

my sincere gratitude to all other academic and administrative staff members of the Department of Econometrics and Business Statistics at Monash University.

I would like to sincerely thank Emeritus Professor Gael Martin, who, together with Professor Rob Hyndman, welcomed me into the department at the time of my application. I also thank Professor George Athanasopoulos for his generous assistance in making it possible for me to complete my visit to the USA. My heartfelt thanks go as well to Professor Brett Inder, Associate Professor Christo Karuna, and Professor Xueyan Zhao, whose warm smiles, encouragement, and constant support have been invaluable in helping me to complete this journey.

I would also like to thank Professor Heike Hofmann, Associate Professor Susan VanderPlas, Associate Professor Ursula Laa, Associate Professor Natalia da Silva, and Professor Eun-Kyung, whom I had the privilege of visiting during my PhD journey. Their generosity in providing opportunities, their guidance in improving my work, and their endless care and kindness have meant a great deal to me. I am also grateful to the University of Nebraska–Lincoln, USA and the University of Natural Resources and Life Sciences, Vienna, Austria, for warmly hosting me during these visits.

I acknowledge the use of Grammarly and ChatGPT for grammar and spelling checks, which helped improve the accuracy of my writing. This thesis was written using Quarto and the Monash University thesis template, which greatly supported the preparation of reproducible documents in both PDF and HTML formats.

I remember with gratitude the University of Sri Jayewardenepura, Sri Lanka, for paving the way for my academic journey. I am especially grateful to my undergraduate research supervisor, Dr Thiyanga S. Talagala, for sparking my interest in visualization, and for broadening my academic horizons through her wisdom and mentorship. Without her guidance, I would never have thought to begin this journey, and I remain deeply grateful for that. I am also grateful to Sigithi Kindergarten, Weligama, Sri Sumangala Balika Vidyalaya, Weligama, and Sujatha Vidyalaya, Matara, Sri Lanka, where I received my early, primary, and secondary education. I extend my heartfelt thanks to all my teachers and private tutors, who continue to check on me even today. The lessons and challenges from those years made me stronger and more resilient, which proved invaluable during my PhD journey.

Parts of this thesis have been prepared for publication. Chapter 2 has been resubmitted to the **Journal of Computational and Graphical Statistics** following a first round of revision, and I am grateful to the two anonymous reviewers for their constructive comments. Chapters 3 and 4 are currently under revision for submission to **The R Journal**.

I presented my research work at 12^{th}-Conference of the Asian Regional Section of the International Association for Statistical Computing (IASC-ARS 2023) (Wollongon, Australia), Australian Statistical

Conference (ASC 2023) (Wollongon, Australia), Bioinformatics Seminar 2024, Victorian branch of the Australian and New Zealand Industrial and Applied Mathematics Society (VicANZIAM) 2024 (RMIT university, Melbourne, Australia), Faculty of BusEco Three Minute Thesis (3MT) competition 2024, useR! 2024 (Salzburg, Austria), Graphics Group Presentation 2024 (Nebraska, USA), UNO Data Science Club 2024 (Omaha, USA), Joint Statistical Meetings (JSM) 2025 (Nashville, USA), useR! 2025 (Durham, USA), Biometrics in the Bush Capital (BIBC2025) (Canberra, Australia), and Australian Statistical Conference (ASC 2025) (Perth, Western Australia). I would like to thank the participants of these seminars, research groups, and conferences for their valuable comments.

I am deeply grateful to my extended family here in Melbourne, Nuwani and Rehan, Heshani and Kanishka, and Shanika (also became my gym partner), who not only stood by me during difficult times but constant listeners. My heartfelt thanks also go to Chaya and Supun, Himasha and Danushka, and Hiruni, for their genuine friendship and unwavering support.

I am thankful to my fellow PhD students at Monash for making this journey such a memorable experience. Special appreciation goes to my "Numbats family", Patrick, Mitch, Fin, Sherry, Cynthia, Harriet, David, Janith, Kris, Hannah, Tina, Javad, and Maliny, for the camaraderie and encouragement. I also treasure the many conversations with Nimni and Gayani during office hours, which brought lightness and motivation to my days. To my office mates: Shelly, Elvis, Floyd, Cash, Minh, and Vis, thank you for creating a supportive and welcoming environment. I am especially grateful to Dr Xiaoqian, both an office mate and a wonderful roommate during conference trips, for sharing this journey with me. I also want to thank Can for the warm welcome and kindness she showed me during my visit to Canberra.

My sincere thanks go to Dovini and Vihanga for their help during my English test and the application process to Australia, and to the MIG family for their kindness and support throughout my PhD. I also extend my gratitude to Danusha, Kalani, Narmada, Malith, and all the friends I met during my visit to the University of Nebraska, USA, who made that experience so enriching. I will never forget the kindness of Dilmi and Nishadi, who welcomed me on my very first day in Melbourne, picked me up, and helped me settle into a new life. I remain truly grateful for their generosity. I am grateful to Pabasara, Vindula, Amaya, Nirmitha, and Malinda for being wonderful friends from my university days until now, and for continuing to support me through the difficult moments of this journey. I also sincerely thank Kavishka, Dinith, Sonal, Thilina, and Tharaka for their enduring friendship and encouragement. I would also like to thank my gym trainer and yoga teachers for their support in maintaining my physical and mental well-being during my PhD. A special thanks to Paton, Michael's dog, whose wagging tail and joyful spirit always lifted my mood.

Last, but by no means least, I would like to thank my family members. I am truly grateful to my parents for their unconditional, selfless love, care, and sacrifices that they made to make my life better. Your inspiration and guidance always kept me motivated to perceive better achievements in my life. I am fortunate to be your daughter. I would also like to thank my grandparents, my sister, my brother, and my sister-in-law for their unconditional love, understanding, and moral support both in this academic journey and in life. No matter how I stumble, you have always lifted me up and reminded me of my worth. I am also grateful to my pet family, even though they are far away, for the joy and comfort they brought throughout this journey. Amma, Thaththa, Achchi, Seeya, Nangi, Malli, Podi nangi, Podi mama, Chuti nanda, Punchi and Bappa, I am endlessly grateful for you, beyond words, beyond life, and beyond everything.

This journey was never mine alone, and for that, I am eternally grateful!

Chapter 1

Introduction

A high-dimensional dataset is one in which each observation is described by many features, or dimensions, often with associations among them. To create visual representations of high-dimensional data, it is common to apply **dimension reduction** techniques. One established approach is **linear projection**, where high-dimensional points are represented as linear combinations of the original features. **Principal Component Analysis (PCA)** (for an overview, see Jolliffe (2011)) is the most familiar method, identifying directions of maximum variance. Extending this idea, **tours** (See Lee et al. (2021) for a review of tour methods) provide dynamic sequences of linear projections, giving views from multiple angles to help reveal hidden structure. Tour methods are implemented in R packages such as `tourr` (Wickham et al. 2011), `langevitour` (Harrison 2023), and `detourr` (Hart and Wang 2025). A key advantage of linear projections is that they preserve the geometric relationships of the original data; they do not introduce distortion. However, linear projections can become cluttered, and global structure may obscure local detail. Furthermore, *piling* (Laa et al. 2022), where points concentrate in the center of projections, can mask important variation.

Because linear projections can reveal only limited aspects of high-dimensional structure, analysts often turn to nonlinear dimension reduction (NLDR) methods in the hope of revealing patterns that may not be visible in any linear view. Common NLDR techniques include t-distributed stochastic neighbor embedding (tSNE) (Maaten and Hinton 2008), uniform manifold approximation and projection (UMAP) (McInnes et al. 2018), potential of heat-diffusion for affinity-based trajectory embedding (PHATE) (Moon et al. 2019), large-scale dimensionality reduction using triplets (TriMAP) (Amid and Warmuth 2019), and pairwise controlled manifold approximation (PaCMAP) (Wang et al. 2021). The methods tSNE, UMAP, TriMAP, and PaCMAP can be considered for producing the 2-D representation by minimizing the divergence between two inter-point distance distributions. PHATE is an example of a diffusion process spreading to capture geometric shapes that include both global and local structure.

(See Coifman et al. (2005) for an explanation of diffusion processes.) These methods are designed to *exaggerate structure*, making it easier for analysts to detect patterns that may not be apparent through linear projections.

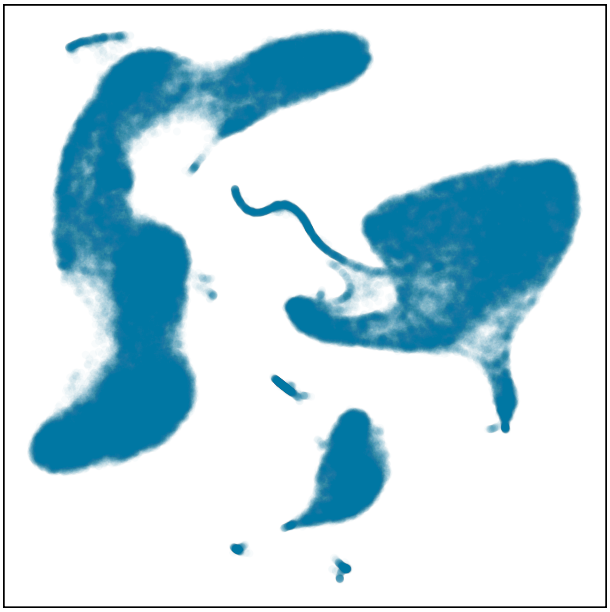

Figure 1.1: *UMAP representation shown in Hao et al. (2021) of human PBMC CITE-seq dataset, generated using n_neighbors* $= 30$ *and min_dist* $= 0.3$*, used as a motivating example of NLDR layouts. The layout shows multiple clusters with distinct shapes, including compact, well-separated groups as well as elongated and partially overlapping structures. This raises the question of whether this layout faithfully represents the underlying high-dimensional structure in the PBMC CITE-seq data.*

Yet this strength also introduces a critical risk: **NLDR can hallucinate structure**, creating patterns in the low-dimensional space that do not exist in the high-dimensional data. This is illustrated in Figure 1.1, where a UMAP layout of a CITE-seq dataset appears to show several distinct clusters with different shapes. While these patterns are visually appealing and easy to interpret, it is not obvious whether they reflect true structure in the high-dimensional data or arise from the choice of method and hyper-parameters. This naturally leads to key questions: *Can this layout be trusted? Does it faithfully represent the structure of the underlying* 10*-D PBMC CITE-seq data?*

Despite the widespread use of NLDR, there is no widely accepted or visually interpretable framework for **diagnosing the reliability of NLDR representations**. Analysts are left to rely on subjective judgment when choosing and interpreting NLDR layouts, without tools to distinguish faithful representations from artifacts. There is also a lack of benchmark clustering datasets for testing, especially NLDR methods.

In addition to technical gaps, **little is known about how people perceive and misperceive structure in NLDR layouts**. It is unclear how different NLDR representations influence analysts' conclusions,

or how distortions introduced by NLDR affect decision-making. Given the critical role of visualization in high-dimensional data analysis, understanding the human perception of NLDR representations is essential.

1.1 Research objectives

This thesis aims to address the key challenges in understanding and evaluating NLDR methods through four main objectives:

1. Develop a new approach and software to evaluate NLDR techniques, providing tools to assess whether low-dimensional representations accurately capture the high-dimensional data structures, using visual and quantitative diagnostics.
2. Design and conduct a user study to explore perception and misperception in NLDR representations, assessing whether participants conceptualize the data structure similarly in NLDR layout compared to tours of linear combinations. This will guide the development of further cognitive perception experiments for assessing NLDR.
3. Generate benchmark clustering data structures in high dimensions with some additional properties like background noise, using the `cardinalR` package, to evaluate the performance of the algorithms, like clustering, NLDR.
4. Provide a web tool for NLDR users to help select the most reasonable NLDR representation among a selection of possible layouts.

1.2 Contribution

This research contributes to a deeper understanding of how NLDR methods can be evaluated and trusted in practice. It provides new tools and software for assessment, benchmark datasets for testing algorithms, and insights from a user study exploring how participants perceive and misperceive structures in NLDR layouts.

1.3 Thesis outline

The rest of the thesis is organized as follows:

Chapter 2 introduces an algorithm to assess the NLDR and decide on which, if any, is the most reasonable representation of the structure(s) present in high-dimensional data. We create a model starting with an NLDR layout that is then used to display as a wireframe in high dimensions.

Chapter 3 presents the implementation of the work, which is available as an R package named `quollr`, an acronym for "**qu**estioning how a high-dimensional **o**bject **l**ooks in **l**ow-dimensions using **r**" (Gamage et al. 2025a). This package also contains a function for performing hexagonal binning using a new approach, for saving `langevitour` results with a specific projection, and link plots to understand the quirks that occur with different NLDR techniques.

Chapter 4 introduces the R package, `cardinalR` (Gamage et al. 2025b) (**c**ollection of v**ar**ious high-**d**imens**i**o**nal** data structures in **R**), which includes functions to generate high-dimensional clustering data structures, with features such as adding noise dimensions and background noise, along with some already generated examples.

Chapter 5 provides empirical evidence on how viewers recognize structure differently when using NLDR layouts versus the tour view, particularly with varying distances between clusters. The findings will help clarify common mistakes made when selecting and reporting structures based on NLDR layouts.

Chapter 6 introduces `menuraR` (**m**onitoring **e**mbeddings of **n**onlinear **u**nfoldings for **r**epresentation and **a**nalysis in **R**), a Shiny web application designed to select and evaluate NLDR layouts.

Chapter 7 concludes the thesis, summarizes the contribution of the work, and discusses some future plans.

Chapter 2

Choosing Better NLDR Layouts by Evaluating the Model in the High-Dimensional Data Space

Nonlinear dimension reduction (NLDR) techniques such as tSNE, and UMAP provide a low-dimensional representation of high-dimensional data (p-D) by applying a nonlinear transformation. NLDR often exaggerates random patterns. But NLDR views have an important role in data analysis because, if done well, they provide a concise visual (and conceptual) summary of p-D distributions. The NLDR methods and hyper-parameter choices can create wildly different representations, making it difficult to decide which is best, or whether any or all are accurate or misleading. To help assess the NLDR and decide on which, if any, is the most reasonable representation of the structure(s) present in the p-D data, we have developed an algorithm to show the 2-D NLDR model in the p-D space, viewed with a tour, a movie of linear projections. From this, one can see if the model fits everywhere, or better in some subspaces, or completely mismatches the data. Also, we can see how different methods may have similar summaries or quirks.

2.1 Introduction

Nonlinear dimension reduction (NLDR) is popular for making a convenient low-dimensional (k-D) representation of high-dimensional (p-D) data ($k < p$). Recently developed methods include t-distributed stochastic neighbor embedding (tSNE) (Maaten and Hinton 2008), uniform manifold approximation and projection (UMAP) (McInnes et al. 2018), potential of heat-diffusion for affinity-based trajectory embedding (PHATE) algorithm (Moon et al. 2019), large-scale dimensionality

reduction using triplets (TriMAP) (Amid and Warmuth 2019), and pairwise controlled manifold approximation (PaCMAP) (Wang et al. 2021).

However, the representation generated can vary dramatically from method to method, choice of hyper-parameter, or even random seed, as illustrated by Figure 2.1. The specific method and hyper-parameters used to produce each layout (see Supplementary materials) are not essential for the discussion. The dilemma for the analyst is which representation to use. The choice might result in different procedures used in the downstream analysis or different inferential conclusions. Various academics have expressed concerns with current practices and procedures for choosing (e.g. Irizarry (2024), Chari and Pachter (2023)). The research described here provides new numerical and visual tools to aid with this decision.

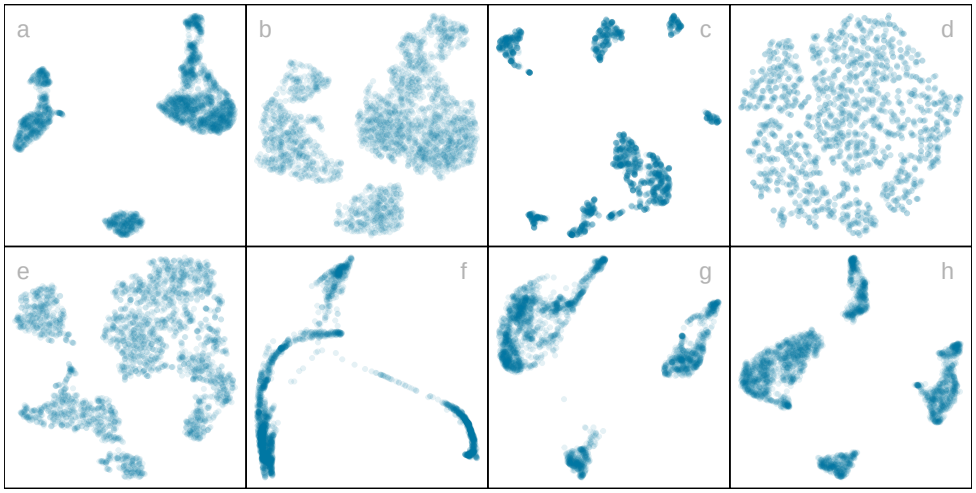

Figure 2.1: *Eight different NLDR representations of the same data, produced by tSNE, UMAP, PHATE, TriMAP, and PaCMAP with a variety of hyper-parameter choices. The variety in layouts makes it difficult to choose which best represents the data distribution.*

The chapter is organized as follows. Section 2.2 provides a summary of the literature on NLDR and high-dimensional data visualization methods. Section 2.3 contains the details of the new methodology, including a simulated data example. In Section 2.4, we describe how to assess the best fit and identify the most accurate 2-D layout based on the proposed model diagnostics. Two applications illustrating the use of the new methodology for bioinformatics and image classification are in Section 2.5. Limitations and future directions are provided in Section 2.6.

2.2 Background

Historically, low-dimensional (k-D) representations of high-dimensional (p-D) data have been computed using multidimensional scaling (MDS) (Kruskal 1964), which includes principal components

analysis (PCA) (for an overview see Jolliffe (2011)). (A contemporary comprehensive guide to MDS can be found in Borg and Groenen (2005).) The *k-D* representation can be considered to be a layout of points in *k-D* produced by an embedding procedure that maps the data from *p-D*. In MDS, the *k-D* layout is constructed by minimizing a stress function that differs distances between points in *p-D* with potential distances between points in *k-D*. Various formulations of the stress function result in non-metric scaling (Saeed et al. 2018) and isomap (Silva and Tenenbaum 2002). Challenges in working with high-dimensional data, including visualization, are outlined in Johnstone and Titterington (2009).

Many new methods for NLDR have emerged in recent years, all designed to better capture specific structures potentially existing in *p-D*. Here we focus on five currently popular techniques: tSNE, UMAP, PHATE, TriMAP, and PaCMAP. The methods tSNE, UMAP, TriMAP, and PaCMAP can be considered for producing the *k-D* representation by minimizing the divergence between two inter-point distance distributions. PHATE is an example of a diffusion process spreading to capture geometric shapes that include both global and local structure. (See Coifman et al. (2005) for an explanation of diffusion processes.)

The array of layouts in Figure 2.1 illustrates what can emerge from the choices of method and hyper-parameters, and the random seed that initiates the computation. Key structures interpreted from these views suggest: (1) highly **separated clusters** (a, b, e, g, h) with the number ranging from 3-6; (2) **stringy branches** (f), and (3) **barely separated clusters** (c, d), which would **contradict** the other representations. These contradictions arise because these methods and hyper-parameter choices provide different lenses on the interpoint distances in the data.

The alternative approach to visualizing the high-dimensional data is to use linear projections. PCA is the classical approach, resulting in a set of new variables that are linear combinations of the original variables. Tours, defined by Asimov (1985), broaden the scope by providing movies of linear projections that provide views of the data from all directions. (See Lee et al. (2021) for a review of tour methods.) There are many tour algorithms implemented, with many available in the R package `tourr` (Wickham et al. 2011), and versions enabling better interactivity in `langevitour` (Harrison 2023) and `detourr` (Hart and Wang 2025). Linear projections are a safe way to view high-dimensional data because they do not warp the space, so they are more faithful representations of the structure. However, linear projections can be cluttered, and global patterns can obscure local structure. The simple activity of projecting data from *p-D* suffers from piling (Laa et al. 2022), where data concentrates in the center of projections. NLDR is designed to escape these issues, to exaggerate structure so that it can be observed. But as a result, NLDR can hallucinate wildly, to suggest patterns that are not actually present in the data.

Our proposed solution is to use the tour to examine how the NLDR is warping the space. It follows what Wickham et al. (2015) describes as *model-in-the-data-space*. The fitted model should be overlaid on the data to examine the fit relative to the spread of the observations. While this is straightforward and commonly done when data is 2-D, it is also possible in p-D, for many models, when a tour is used.

Wickham et al. (2015) provides several examples of models overlaid on the data in p-D. In hierarchical clustering, a representation of the dendrogram using points and lines can be constructed by augmenting the data with points marking the merging of clusters. Showing the movie of linear projections reveals how the algorithm sequentially fitted the cluster model to the data. For linear discriminant analysis or model-based clustering, the model can be indicated by $(p-1)$-D ellipses. It is possible to see whether the elliptical shapes appropriately match the variance of the relevant clusters, and to compare and contrast different fits. For PCA, one can display the model (a k-D plane of the reduced dimension) using wireframes of transformed cubes. Using a wireframe is the approach we take here to represent the NLDR model in p-D.

2.3 Method

2.3.1 What is the NLDR model?

At first glance, thinking of NLDR as a modeling technique might seem strange. It is a simplified representation or abstraction of a system, process, or phenomenon in the real world. The p-D observations are the realization of the phenomenon, and the k-D NLDR layout is the simplified representation. Typically, $k = 2$ is used for the rest of this chapter. From a statistical perspective, we can consider the distances between points in the 2-D layout to be variance that the model explains, and the (relative) difference with their distances in p-D is the error, or unexplained variance. We can also imagine that the positioning of points in 2-D represents the fitted values, which will have some prescribed position in p-D that can be compared with their observed values. This is the conceptual framework underlying the more formal versions of factor analysis (Jöreskog 1969) and MDS. (Note that, for this thinking, the full p-D data needs to be available, not just the interpoint distances.)

We define the NLDR as a function g: $\mathbb{R}^{n\times p} \rightarrow \mathbb{R}^{n\times 2}$, with hyper-parameters $\boldsymbol{\theta}$. These parameters, $\boldsymbol{\theta}$, depend on the choice of g, and can be considered part of model fitting in the traditional sense. Common choices for g include functions used in tSNE, UMAP, PHATE, TriMAP, PaCMAP, or MDS, although in theory any function that does this mapping is suitable.

With our goal being to make a representation of this 2-D layout that can be lifted into high-dimensional space, the layout needs to be augmented to include neighbor information. A simple approach would be to triangulate the points and add edges. A more stable approach is to first bin the data, reducing it from n to $m \leq n$ observations, and connect the bin centroids. We recommend using a hexagon grid because it better reflects the data distribution and has fewer artifacts than a rectangular grid. This process serves to reduce some noisiness in the resulting surface shown in p-D. The steps in this process are shown in Figure 2.2, and documented below.

To illustrate the method and how to use it to choose a reasonable layout, we use 7-D simulated data, which we call the "2NC7" data. It has two separated nonlinear clusters, one forming a 2-D curved shape, and the other a 3-D curved shape, each consisting of 1000 observations. The first four variables hold this cluster structure, and the remaining three are purely noise. We would consider (X_1, X_2, X_3, X_4) to hold the geometric structure (true model) that we hope to capture. This data is sufficiently simple, with just two complexities (two separated curvilinear clusters and two different implicit dimensions), to adequately explain the new method. The applications section contains two practical examples where NLDR has been used in published work. This data has both global and local structure. The two separated clusters would be considered to be a global structure, and the nonlinear low-dimensional shapes could be considered to be a local structure, one being 2-D and the other 3-D. An ideal NLDR layout would reveal the two clusters with moderate separation, and flatten the curvilinear forms while preserving the proximity of points.

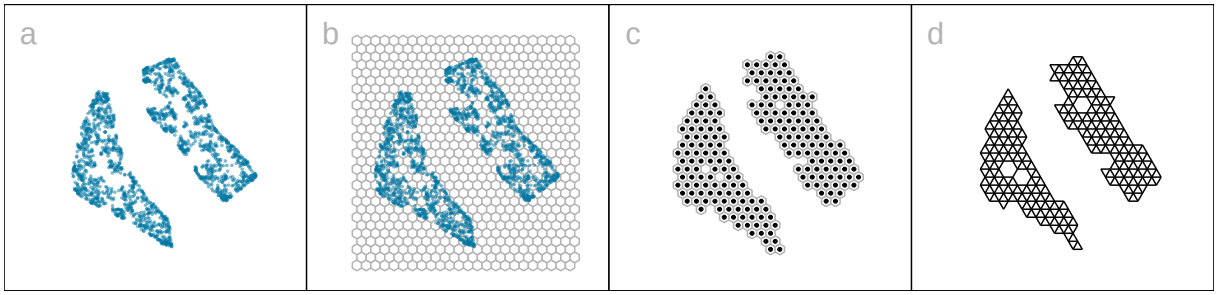

Figure 2.2: *Key steps for constructing the model on the tSNE layout ($k = 2$) of 2NC7: (a) data, (b) hexagon bins, (c) bin centroids, and (d) triangulated centroids. The 2NC7 data is shown.*

2.3.2 Algorithm to represent the model in 2-D

Scale the data

Because we are working with distances between points, starting with data having a standard scale, e.g., $[0, 1]$, is recommended. The default should take the aspect ratio produced by the NLDR $(r_1, r_2, ..., r_k)$ into account. When $k = 2$, as in hexagon binning, the default range is $[0, y_{i,\max}], i = 1, 2$, where $y_{1,\max} = 1$ and $y_{2,\max} = r_2/r_1$ (Figure 2.2). If the NLDR aspect ratio is ignored then set $y_{2,\max} = 1$.

Hexagon grid configuration

Although there are several implementations of hexagon binning (Carr et al. 1987) and a published paper (Carr et al. 2023), surprisingly, none have sufficient detail or components that produce everything needed for this project. So we described the process used here. Figure 2.3 illustrates the notation used.

The 2-D hexagon grid is defined by its bin centroids. Each hexagon, H_h $(h = 1, \ldots, b)$ is uniquely described by centroid, $C_h^{(2)} = (c_{h1}, c_{h2})$. The number of bins in each direction is denoted as (b_1, b_2), with $b = b_1 \times b_2$ being the total number of bins. We expect the user to provide just b_1, and we calculate b_2 using the NLDR ratio to compute the grid.

To ensure that the grid covers the range of data values, a buffer parameter (q) is set as a proportion of the range. By default, $q = 0.1$. The buffer should be extending a full hexagon width (a_1) and height (a_2) beyond the data, in all directions. The lower left position where the grid starts is defined as (s_1, s_2), and corresponds to the centroid of the lowest left hexagon, $C_1^{(2)} = (c_{11}, c_{12})$. This must be smaller than the minimum data value. Because it is one buffer unit, q below the minimum data values, $s_1 = -q$ and $s_2 = -qr_2$.

The value for b_2 is computed by fixing b_1. Considering the upper bound of the first NLDR component, $a_1 > (1 + 2q)/(b_1 - 1)$. Similarly, for the second NLDR component,

$$a_2 \geq \frac{r_2 + q(1 + r_2)}{(b_2 - 1)}.$$

Since $a_2 = \sqrt{3}a_1/2$ for regular hexagons,

$$a_1 \geq \frac{2[r_2 + q(1 + r_2)]}{\sqrt{3}(b_2 - 1)}.$$

This is a linear optimization problem. Therefore, the optimal solution must occur on a vertex. Therefore,

$$b_2 = \left\lceil 1 + \frac{2[r_2 + q(1 + r_2)](b_1 - 1)}{\sqrt{3}(1 + 2q)} \right\rceil. \tag{2.1}$$

Binning the data

Observations are grouped into bins based on their nearest centroid. This produces a reduction in size of the data from n to m, where $m \leq b$ (total number of bins). This can be defined using the function

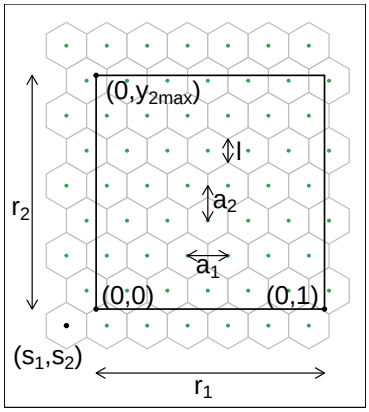

Figure 2.3: *The components of the hexagon grid illustrating notation.*

$u : \mathbb{R}^{n\times 2} \rightarrow \mathbb{R}^{m\times 2}$, where

$$u(i) = \arg \min_{j=1,\ldots,b} \sqrt{(y_{i1} - C_{j1}^{(2)})^2 + (y_{i2} - C_{j2}^{(2)})^2},$$

maps observation i into $H_h = \{i|u(i) = h\}$.

By default, the bin centroid is used for describing a hexagon (as done in Figure 2.2 (c)), but any measure of center, such as a mean or weighted mean of the points within each hexagon, could be used. The bin centers and the binned data are the two important components needed to render the model representation in high dimensions.

Indicating neighborhood

Delaunay triangulation (Gebhardt et al. 2024; Lee and Schachter 1980) is used to connect points so that edges indicate neighboring observations, in both the NLDR layout (Figure 2.2 (d)) and the p-D model representation. When the data has been binned, the triangulation connects centroids. The edges preserve the neighborhood information from the 2-D representation when the model is lifted into p-D.

2.3.3 Rendering the model in p-D

The last step is to lift the 2-D model into p-D by computing p-D vectors that represent bin centroids (Figure 2.4). We use the p-D mean of the points in a given hexagon, H_h, denoted $C_h^{(p)}$, to map the centroid $C_h^{(2)} = (c_{h1}, c_{h2})$ to a point in p-D. Let the j^{th} component of the p-D mean be

$$C_{hj}^{(p)} = \frac{1}{n_h} \sum_{i=1}^{n_h} x_{hij}, \quad h = 1, \ldots, b;\ j = 1, \ldots, p;\ n_h > 0.$$

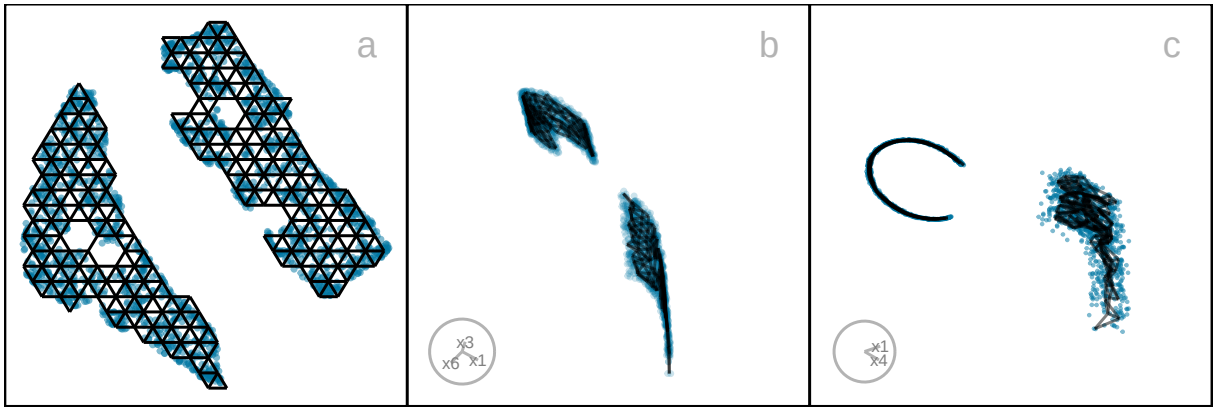

Figure 2.4: *Lifting the 2-D fitted model into p-D. Two projections of the p-D fitted model overlaying the data are shown in b, c. The fit is reasonably tight with the data in one cluster (top one in b), but slightly less so in the other cluster, probably because it is 3-D. Notice also that, in the 2-D layout, the two clusters have internal gaps which create a model with some holes. This lacy pattern happens regardless of the hyper-parameter choice, but this doesn't severely impact the p-D model representation.*

2.3.4 Measuring the fit

All NLDR methods internally optimize a quantity to produce a layout for any particular hyper-parameter set. These are not always made available in the model output, and may not be universally comparable between hyper-parameter choices and methods.

Several common metrics are often used to assess the quality of any NLDR layout, based on the preservation of the global and local structure of the data. The *RNX* curve quantifies the neighborhood agreement between *p-D* and *k-D* spaces, by computing the area under the curve (*ARNX*) across a range of neighborhood scales (Lee et al. 2015). A high value indicates better preservation of a balance of global and local structure. Random Triplet Accuracy (RTA) and Centroid Triplet Accuracy compare the order of 2-*D* and *p-D* distances of random triplets of points (Wang et al. 2021). High values indicate preservation of the geometry, suggesting both local and global structure preservation. The Shepard diagram (Shepard 1962) and its associated Spearman correlation (SC) (Spearman 1961) are used to assess the relationship between *p-D* and *k-D* distances. High values indicate preservation of global structure. The Global Score (GS) measures how well an embedding retains the overall geometry of the data relative to a PCA baseline (Amid and Warmuth 2019). Higher values indicate better preservation of global structure. The metric RTA, SC, GS, and ARNX have been reversed (rRTA, rSC, rGS, and rARNX) so that they align with HBE - the lower the value, the better the layout.

None of the above measures is particularly well-suited to assessing our model fit, as we will show later. Thus, we need a different approach to measuring model fit. Because the model here is similar to a confirmatory factor analysis model (see a general explanation in Brown (2015)), our approach is similar to the ones used in this area. It is based on "residuals" computed as the difference between

the fitted model and observed values in p-D. Observations are associated with their bin center, $C_h^{(p)}$, which are also considered to be the *fitted values*. The error is computed by taking the squared p-D Euclidean distance of points from their bin centroid, which we will call the hexbin error (HBE):

$$HBE = \sqrt{\frac{1}{n}\sum_{h=1}^{m}\sum_{i=1}^{n_h}\sum_{j=1}^{p}(x_{hij} - C_{hj}^{(p)})^2} \tag{2.2}$$

where n is the number of observations, m is the number of non-empty bins, n_h is the number of observations in h^{th} bin, p is the number of variables and x_{hij} is the j^{th} dimensional data of i^{th} observation in h^{th} hexagon. We can consider $e_{hi} = \sqrt{\sum_{j=1}^{p}(x_{hij} - C_{hj}^{(p)})^2}$ to be the residual for each observation. Figure 2.5 shows plots of e as a density (a), coloring the points in the NLDR layout (b), and the points in a tour (c). It can be seen that the biggest residuals are in one cluster, which occurs due to the intentional design that one cluster is slightly 3-D and perfectly captured by a 2-D layout.

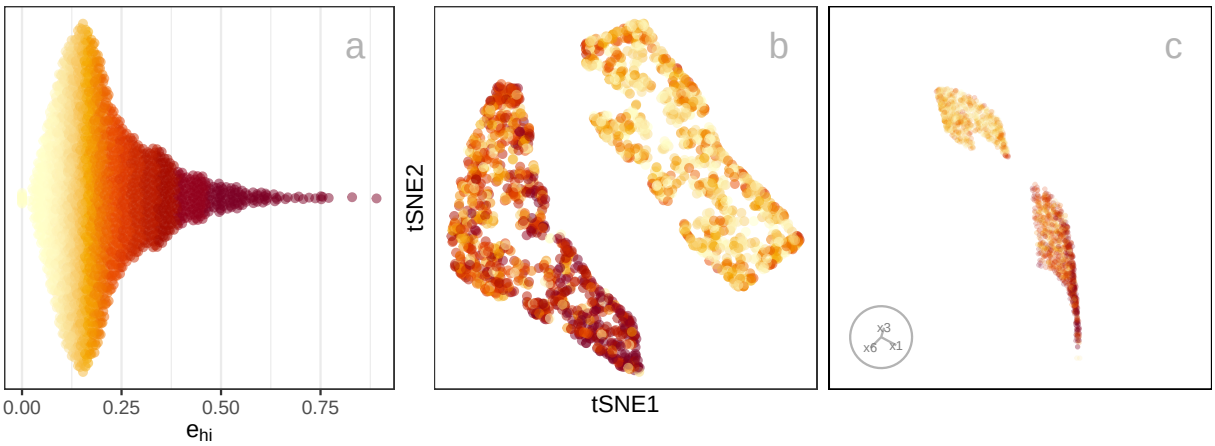

Figure 2.5: *Examining the distribution of residuals in a jittered dotplot (a), 2-D NLDR layout (b), and a tour of 4-D data space (c). Color indicates residual (e_{hi}), dark color indicating high value. Most large residuals are distributed in one cluster (bottom one in c), and most small residuals are distributed in the other cluster.*

2.3.5 Prediction into 2-D

NLDR methods are primarily designed for visualization and exploration rather than reconstruction, and do not explicitly provide out-of-sample prediction. Of the five methods studied here, only UMAP provides a `predict()` function for embedding new data points based on the learned manifold (Konopka 2023). Several other approaches, not used here, PCA, neural network autoencoders (Hinton and Salakhutdinov 2006), and parametric tSNE (Maaten 2009) support prediction.

A benefit of our approach is that for any NLDR method, it provides a way to predict the layout position of a new observation, x'. The steps are (1) determine the closest bin centroid in p-D, $C_h^{(p)}$, and (2) predict the embedding to be the bin centroid in 2-D, $C_h^{(2)}$.

2.3.6 Tuning

The model fitting is based on several parameters, including the hexagon bin parameters and the low-count bin removal process. The hexagon bin parameters define the bottom-left bin position (s_1, s_2), the number of bins in the horizontal direction (b_1), which also determines the number of bins in the vertical direction (b_2), the total number of bins (b), and the total number of non-empty bins (m). Low count bins are removed using standardized bin counts, defined as $w_h = n_h/n, \;\; h = 1, \ldots m$.

Default values are provided for each of these, but deciding on the best model fit is assisted by examining a range of values. The default number of bins $b = b_1 \times b_2$ is computed based on the sample size, by setting $b_1 = n^{1/3}$, consistent with the Diaconis-Freedman rule (Freedman and Diaconis 1981). The value of b_2 is determined analytically by b_1, q, r_2 (Equation 2.1). Values of b_1 between 2 and $b_1 = \sqrt{n/r_2}$ are recommended, where the dependence on r_2 reflects the preservation of aspect ratio in the NLDR layout.

Figure 2.6 shows the hexbin grids for three choices of b_1. While the number of bins is the common parameter to modify, bin start positions (s_1, s_2) can also be worth experimenting with also because they can change bin counts.

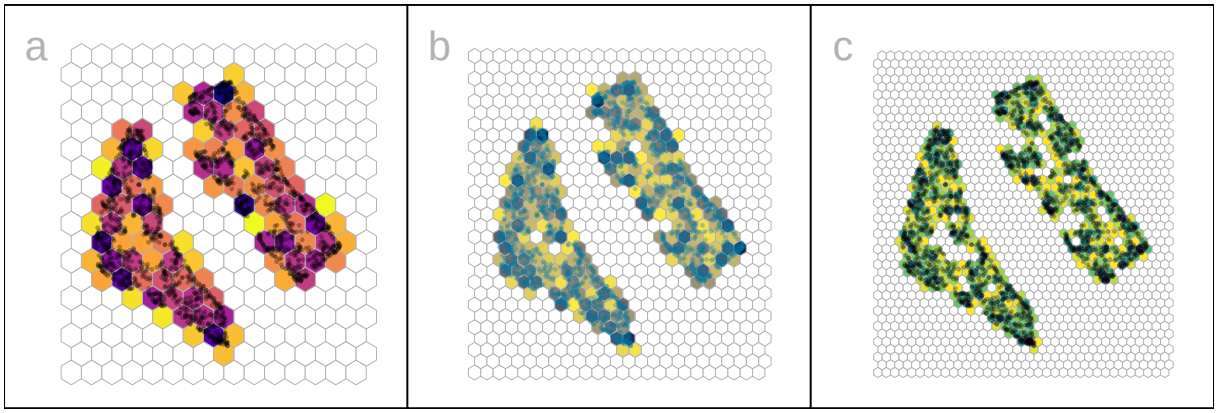

Figure 2.6: *Hexbin density plots of tSNE layout of the 2NC7 data, using three different b_1 specifications yielding different b_2, b, m: (a) **15**, 18, 270, 98, (b) **24**, 29, 696, 209, and (c) **35**, 42, 1470, 386. Color indicates standardized counts, dark indicating high count, and light indicating low count. At the smallest binwidth, the data structure is discontinuous, suggesting that there are too many bins.*

It is worthwhile to consider what desirable aspects of a hexbin result, which maps to summarizing the *p-D* fit well. The binning should capture the underlying data distribution closely, with a minimum number of necessary bins. An ideal binning might be indicated by a more uniform distribution of bin counts or having few relatively empty bins. To help with this assessment average bin count $(\bar{n} = \sum n_h/m)$, average standardized bin count $(\bar{w} = \sum w_h/m)$ and proportion of non-empty bins (m/b), are also computed. Figure 2.7 shows some choices of plots of these quantities for a single

NLDR layout, with three choices of a_1 indicated. Some expectations and reasoning for these plots are:

- HBE will increase as a_1 increases, so good choices will be just before a big increase. In plot a, HBE changes fairly steadily, so there is no easy choice to make.
- HBE can also be examined against the average standardized bin count or the average bin count (plot b). This is similar to the comparison with a_1, but to used when comparing different NLDR layouts. Different layouts might produce different densities of points, which will not be captured well by a comparison of HBE vs a_1.
- The proportion of non-empty bins is interesting to examine across different binwidths (plot c). A good binning should have just the right number of bins to neatly cover the shape of the data, and no more or less. As binwidth gets smaller, m/b should roughly get bigger.
- Bins with a small number of observations might be removed to sharpen the wireframe model. This can have adverse effects, though - failing to extend the wireframe into sparse areas, or resulting in holes in the wireframe. Plot d shows the relationship between HBE (computed for all observations despite some bin removal) and the standardized bin count cutoff used to remove bins. For all three chosen binwidths, a small number of bins can be removed without affecting HBE.

2.3.7 Interactive graphics

Matching points in the 2-D layout with their positions in p-D is useful when tuning the fit. This can be used to examine the fitted model in some subspaces in p-D, in particular in association with residual plots.

The interactive 2-D layout (Sievert 2020) and the `langevitour` (Harrison 2023) view with the fitted model overlaid can be linked using a browsable HTML widget (Cheng and Sievert (2025), Cheng et al. (2024)). A rectangular "brush" is used to select points in one plot, which will highlight the corresponding points in the other plot(s). Because the `langevitour` is dynamic, brush events that become active will pause the animation, so that a user can interrogate the current view. This approach will be illustrated on the examples to show how it can help to understand how the NLDR has organized the observations, and learn where it does not do well.

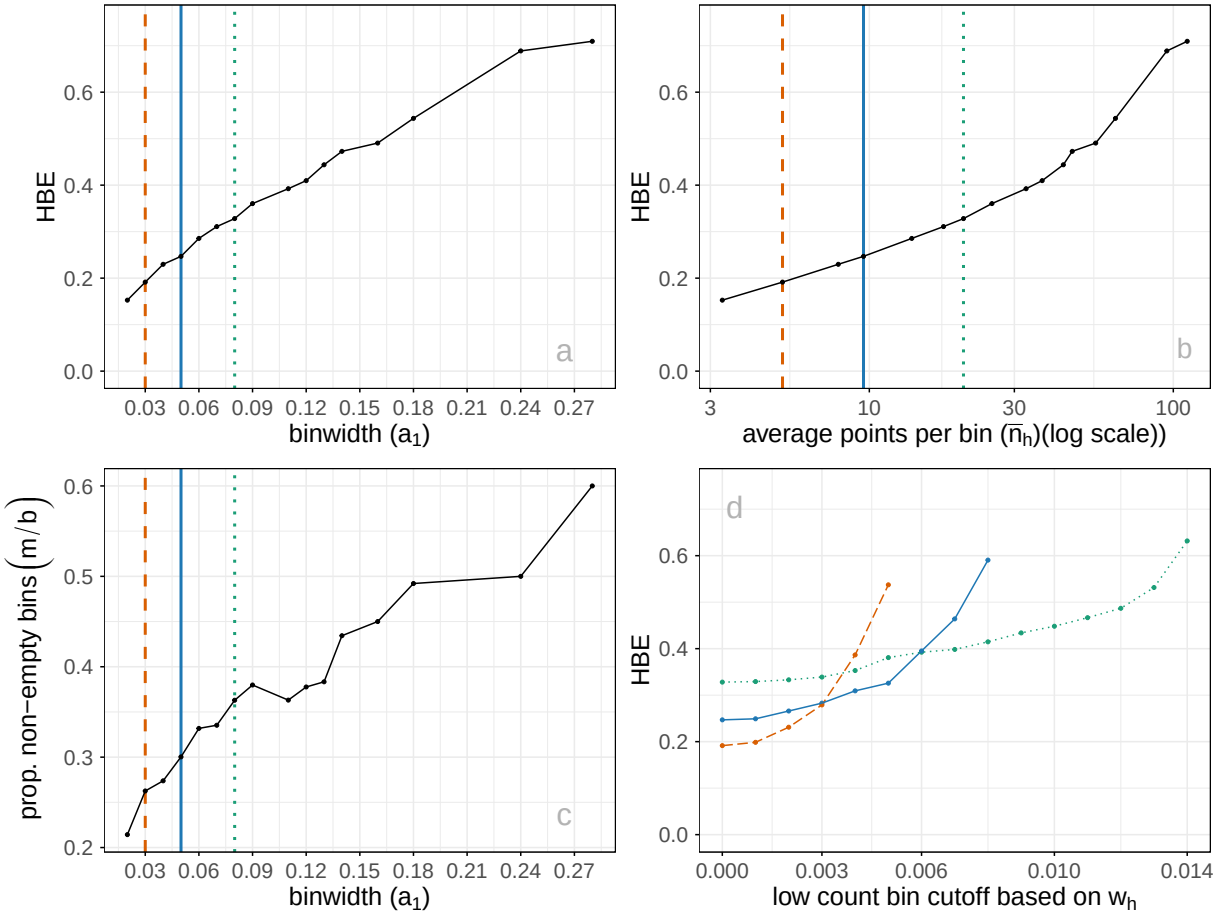

Figure 2.7: *Various plots to help tune the model fit: (a) HBE vs a_1, (b) HBE vs $\bar{n}_h$, (c) proportion of non-empty bins (m/b) vs a_1, (d) HBE vs w_h cutoff used for removing low count bins. Color indicates the three binwidths shown in Figure 6:* 0.03 *(orange dashed),* 0.05 *(blue solid), and* 0.08 *(green dotted). The better model fit will have low HBE, but reasonably sized bins that capture the data sufficiently. Proportion of non-empty bins tends to increase with a_1 (c). Removing a few low-count bins doesn't substantially change HBE, for all three binwidths (d).*

2.4 Choosing the best 2-D layout

Figure 2.8 illustrates the approach to compare the fits for different representations and assess the strength of any fit. What does it mean to be a best fit for this problem? Analysts use an NLDR layout to display the structure present in high-dimensional data in a convenient 2-D display. It is a competitor to linear dimension reduction that can better represent nonlinear associations, such as clusters. However, these methods can hallucinate, suggesting patterns that don't exist, and grossly exaggerate other patterns. Having a layout that best fits the high-dimensional structure is desirable, but more important is to identify bad representations so they can be avoided. The goal is to help users decide on the most useful and appropriate low-dimensional representation of the high-dimensional data.

A particular pattern that we commonly see is that analysts tend to pick layouts with clusters that have big separations between them. When you examine their data in a tour, it is almost always that we see there are no big separations, and actually, often the suggested clusters are not even present. While we don't expect that analysts include animated gifs of tours in their papers, we should expect that any 2-*D* representation adequately indicates the clustering that is present, and honestly shows lack of separation or lack of clustering when it doesn't exist. It is important for analysts to have tools to select the accurate representation, not the pretty but wrong representation.

To decide on a reasonable layout, an analyst needs a selection of NLDR representations generated using a range of hyper-parameter choices and possibly different methods, such as tSNE and UMAP. They also require a range of model fits created by varying the binwidths and the level of low count bin removal, along with the calculated HBE values for each layout after transformation into the *p*-*D* space. Finally, the analyst must be able to visually examine how well each model fits the data in the original data space.

Comparing the HBE to obtain the best fit is appropriate if the same NLDR method is used. However, because the HBE is computed on *p*-*D* data, it measures the fit between model and data, so it can also be used to compare the fit of different NLDR methods. A lower HBE indicates a better NLDR representation.

Figure 2.8 compares the metrics rARNX, rRTA, rSC, rGS, along with HBE computed on $a_1 = 0.05$ for the six layouts shown in Figure 2.8. This is a parallel coordinate plot where the y-axis shows a normalized score to ensure the metrics are on the same scale. Each line corresponds to one layout.

There is some agreement between the metrics. All, except rARNX and HBE, agree that layout d is best. rARNX and HBE agree that layout f is best or very close to best. Layout a is best according to HBE and rARNX but considered to be much less optimal by rRTA, rSC, and rGS. Layout c is considered poor by rRTA, rSC, rGS, and rARNX. This illustrates how difficult it is to use the numerical metrics alone to decide on the best layout.

The problem with rSC is that correlation is not a good measure in the presence of clusters - the further the clusters are apart in the layout, produces the Shepard plot with two clusters of distances will produce a high correlation value. Similar reasoning would explain why rRTA and rGS behave similarly: they put too much emphasis on the global structure. Thus, for the 2NC7 data, further apart clusters score better, overly emphasizing that there are two clusters, even though this separation is not accurately reflecting the difference in *p*-*D*.

When the metrics disagree, it causes confusion for the analyst, and thus provides a temptation to choose the nicest looking layout (very separated clusters), even though it may be a hallucination.

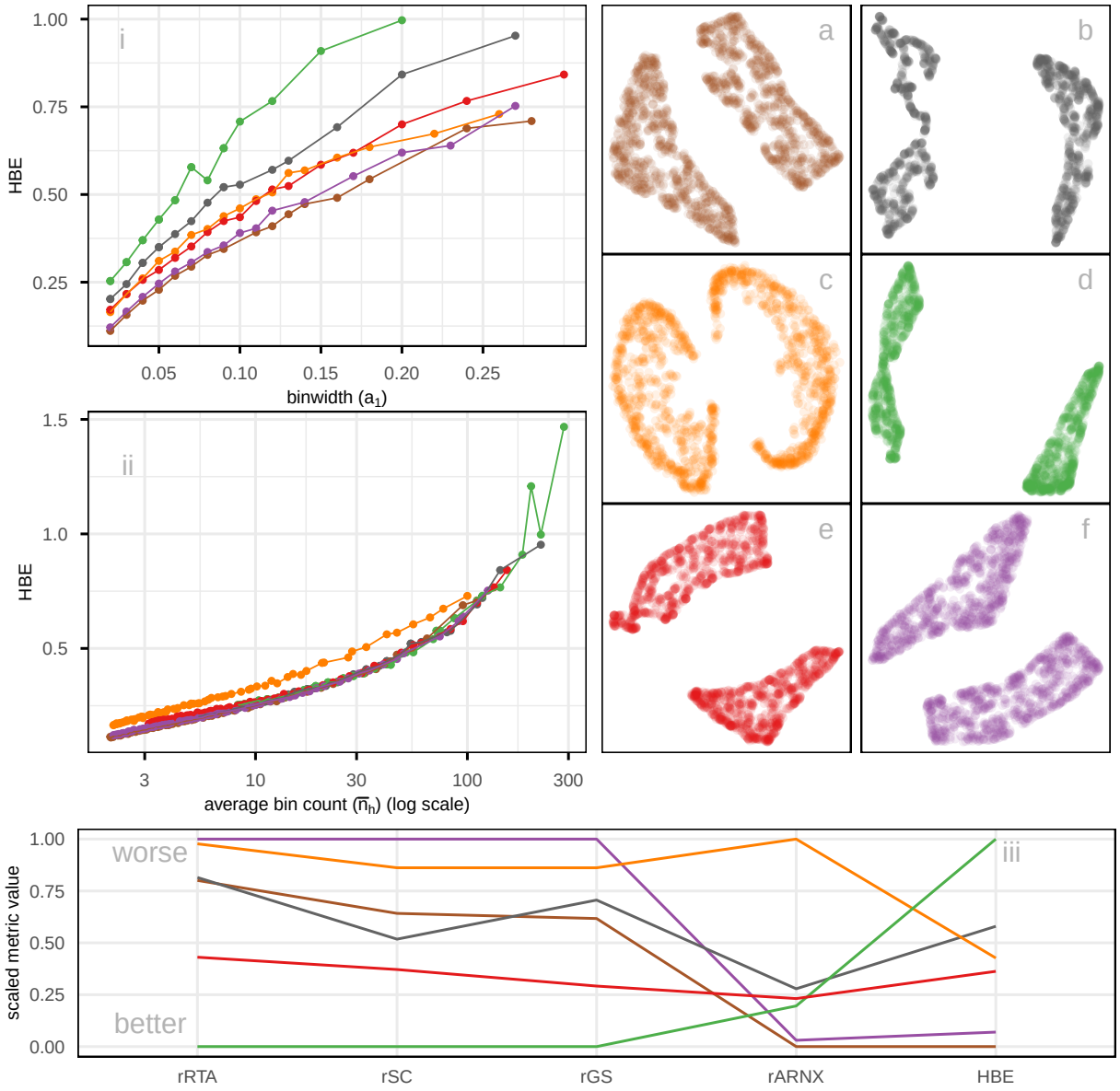

Figure 2.8: *Assessing which of the 6 NLDR layouts (a-f) on the 2NC7 data is the better representation using HBE for varying (i) binwidth (a_1), and (ii) average bin count ($\bar{n}_h$). Color represents the NLDR layout. Layout d is universally poor. Layouts b, e that show two close clusters are universally suboptimal. Layout f with little separation performs well at tiny binwidth (where most points are in their own bin) and poorly as binwidth increases. Layout e has a small separation with oddly shaped clusters. Layout a is the best choice. Plot (ii), which compares HBE values with respect to average bin count, helps account for differences in cluster density; here, the variation among layouts is reduced, showing that some differences observed in (i) arise from density rather than true structure. Comparison of scaled evaluation metrics (rRTA, rSC, rGS, rARNX, and HBE using $a_1 = 0.05$) for the six NLDR layouts computed on the 2NC7 data using a parallel coordinate plot (iii). Color of the line indicates NLDR layout.*

Because HBE is accompanied by a representation of the layout in *p-D* to compare with the observed data, it can help to add more clarity in making decisions. Figure 2.9 shows the fitted models for layouts a (rated high by HBE and rARNX) and c (rated poorly). These are 2-*D* projections from the tour, with black indicating the fitted model overlaid on the blue points of the data. The reason for the poor fit is that the PHATE layout (c) twists extremely along the 2- and 3-*D* manifolds where the data lies. We have learned that all the NLDR methods tend to have twists in the fit in *p-D*, but this is extreme. This is likely why layout c has poor metrics relative to the other layouts, and it suggests that it does not adequately capture the local structure in the 2NC7 data.

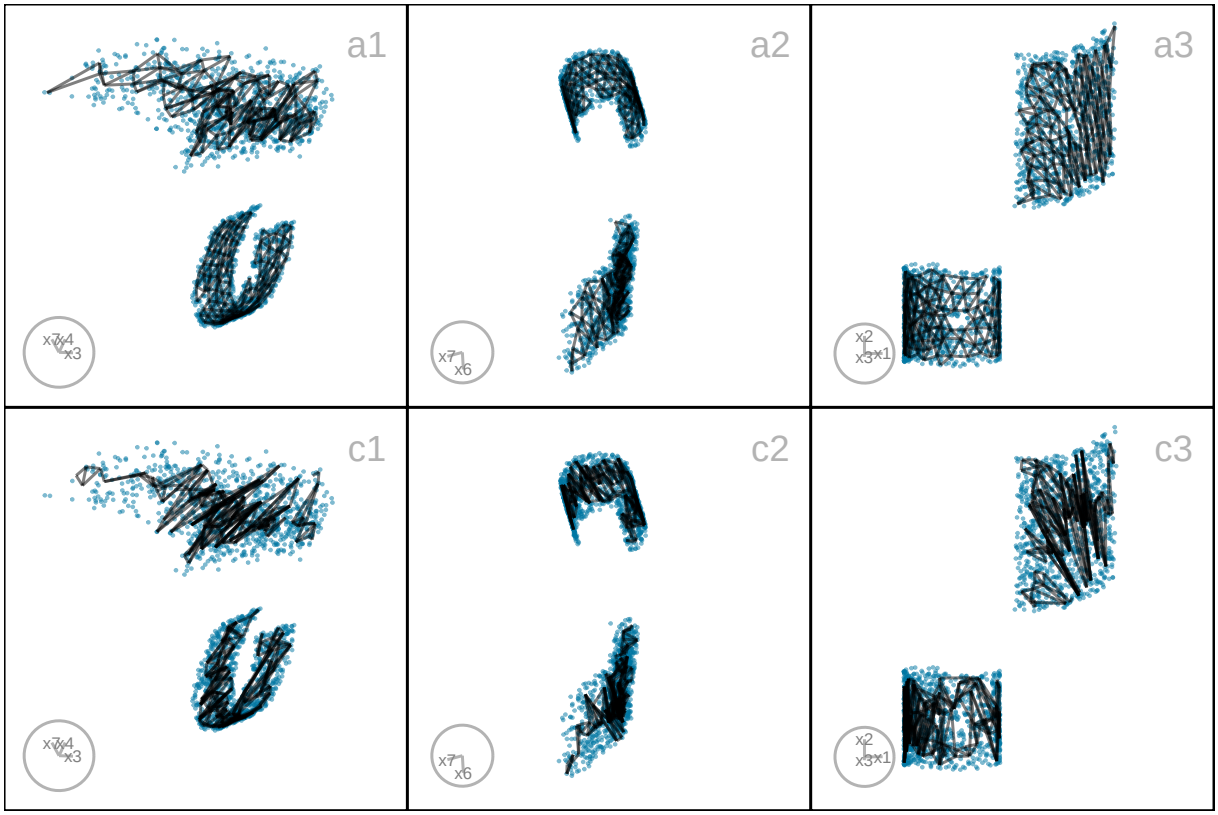

Figure 2.9: *Three 2-D projections from a tour showing the fitted models (black lines) for layouts a (top row) and c (bottom row) of the 2NC7 data (blue points). Layout c, which was poorly rated by rARNX and HBE, covers less of the width of the data than a. The triangular gridding is less visible in c than layout a, and actually corresponds to extreme twisting.*

2.5 Applications

To illustrate the approach, we use two examples: PBMC3k data (single cell gene expression), where an NLDR layout is used to represent cluster structure present in the *p-D* data, and MNIST hand-written digits, where NLDR is used to represent a low-dimensional nonlinear manifold in *p-D*.

2.5.1 PBMC3k

This is a benchmark single-cell RNA-Seq data set collected on Human Peripheral Blood Mononuclear Cells (PBMC3k) as used in 10x Genomics (2016). Single-cell data measures the gene expression of individual cells in a sample of tissue (see, for example, Haque et al. (2017)). This type of data is used to obtain an understanding of cellular level behavior and heterogeneity in their activity. Clustering of single-cell data is used to identify groups of cells with similar expression profiles. NLDR is often used to summarize the cluster structure. Usually, NLDR does not use the cluster labels to compute the layout, but uses color to represent the cluster labels when it is plotted.

In this data, there are 2622 single cells and 1000 gene expressions (variables). Following the same pre-processing as Chen et al. (2024), different NLDR techniques were performed on the first nine principal components. Figure 2.1 shows this data using a variety of methods and different hyper-parameters. You can see that the result is wildly different depending on the choices. Layout a is a reproduction of the layout that was published in Chen et al. (2024). This layout suggests that the data has three very well-separated clusters, each with an odd shape. The question is whether this accurately represents the cluster structure in the data, or whether they should have chosen b or c or d or e or f or g or h. This is what our new method can help with – to decide which is the more accurate 2-D representation of the cluster structure in the p-D data.

Figure 2.10 shows HBE across a range of binwidths (a_1) for each of the layouts in Figure 2.1. The layouts were generated using tSNE and UMAP with various hyper-parameter settings, while PHATE, PaCMAP, and TriMAP were applied using their default settings. Lines are color-coded to match the color of the layouts shown on the right. Lower HBE indicates a better fit. Using a range of binwidths shows how the model changes, with possibly the best model being one that is universally low HBE across all binwidths. It can be seen that layout f is sub-optimal with universally higher HBE. Layout a, the published one, is better, but it is not as good as layouts b, d, or e. With some imagination layout d perhaps shows three barely distinguishable clusters. Layout e shows three, possibly four, clusters that are more separated. The choice reduces from eight to these two. Layout d has slightly better HBE when the a_1 is small, but layout e beats it at larger values. Thus, we could argue that layout e is the most accurate representation of the cluster structure of these eight.

To further assess the choices, we need to look at the model in the data space, by using a tour to show the wireframe model overlaid on the data in the 9-D space (Figure A.7). Here we compare the published layout (a) versus what we argue is the best layout (e). The top row (a1, a2, a3) corresponds to the published layout, and the bottom row (e1, e2, e3) corresponds to the optimal choice according to our procedure. The middle and right plots show two projections. The primary difference between

the two models is that the model of layout e does not fill out to the extent of the data but concentrates in the center of each point cloud. Both suggest that three clusters are a reasonable interpretation of the structure, but the layout e more accurately reflects the separation between them, which is small.

2.5.2 MNIST hand-written digits

The digit "1" of the MNIST dataset (LeCun et al. 1998) consists of 7877 grayscale images of handwritten "1"s. Each image is 28×28 pixels, which corresponds to 784 variables. The first 10 principal components, explaining 83% of the total variation, are used. This data essentially lies on a nonlinear manifold in high dimensions, defined by the shapes that "1"s make when sketched. We expect that the best layout captures this type of structure and does not exhibit distinct clusters.

Figure 2.12 compares the fit of six layouts computed using UMAP (b), PHATE (c), TriMAP (d), PaCMAP (e) with default hyper-parameter setting and two tSNE runs, one with default hyper-parameter setting (a) and the other changing perplexity to 89 (f). The layouts are reasonably similar in that they all have the observations in a single blob. Some (b, c) have a more curved shape than others. Layout e is the most different, having a linear shape and a single very large outlier. Both a and f have a small clump of points, perhaps slightly disconnected from the other points, in the lower to middle right.

The layout plots are colored to match the lines in the HBE vs binwidth (a_1) plot. Layouts a, b, and f fit the data better than c, d, e, and layout f appears to be the best fit. Figure 2.13 shows this model in the data space in two projections from a tour. The data is curved in the 10-D space, and the fitted model captures this curve. The small clump of points in the 2-D layout is highlighted in both displays. These are almost all inside the curve of the bulk of points and are sparsely located. The fact that they are packed together in the 2-D layout is likely due to the handling of density differences by the NLDR.

An interesting aside is that the rather strange layout e, which has what looks like a single point far from the remaining observations, is actually similar to this one. That point is actually a clump of points corresponding to some of the diffuse points interior to the curve of the bulk of points. This is easy to see using the linked brushing tool.

The next step is to investigate the 2-D layout to understand what information is learned from this representation. Figure 2.14 summarizes this investigation. Plot a shows the layout with points colored by their residual value - darker color indicates larger residual and poor fit. The plots b, c, d, e show samples of hand-written digits taken from inside the colored boxes. Going from top to bottom around the curve shape, we can see that the "1"s are drawn from right slant to a left slant. The "1"s in d (black box) tend to have the extra up stroke, but are quite varied in appearance. The "1"s shown in the plots labelled e correspond to points with big residuals. They can be seen to be more strangely

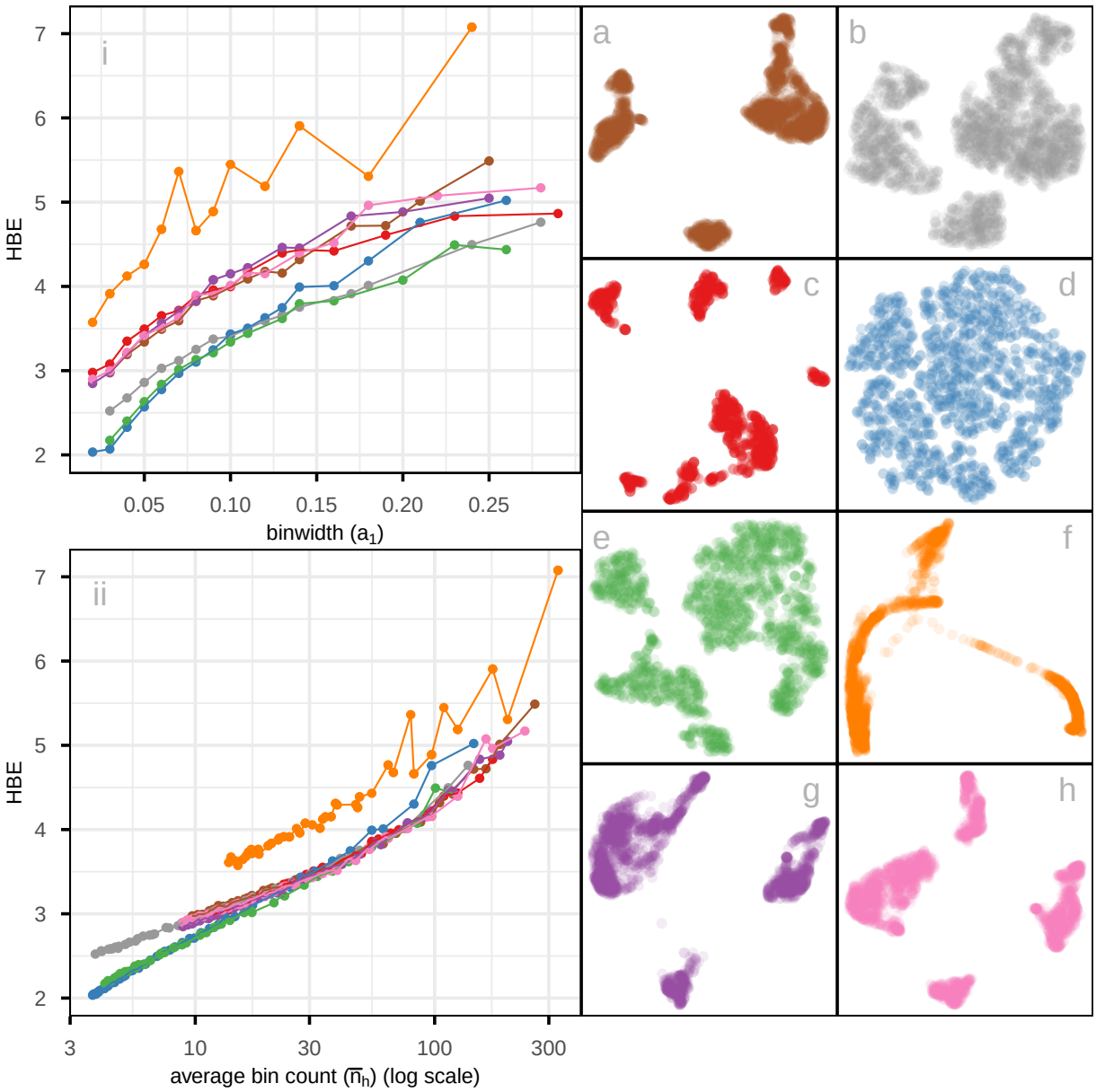

Figure 2.10: *Assessing which of the 8 NLDR layouts on the PBMC3k data (shown in Figure 2.1) is the better representation using HBE for varying (i) binwidth (a_1), and (ii) average bin count ($\bar{n}_h$). Color used for the lines and points in the left plot and in the scatterplots represents the NLDR layout (a-h). Layout f is universally poor. Layouts a, c, g, and h that show large separations between clusters are universally suboptimal. Layout d with little separation performs well at tiny binwidth (where most points are in their own bin) and poorly as binwidth increases. The choice of the best is between layouts b and e, which have small separations between oddly shaped clusters. Layout e is chosen as the best. Plot (ii), which accounts for the density within clusters by using average bin count, shows reduced differences between layouts, indicating that part of the variation in (i) is driven by cluster density rather than true structural differences.*

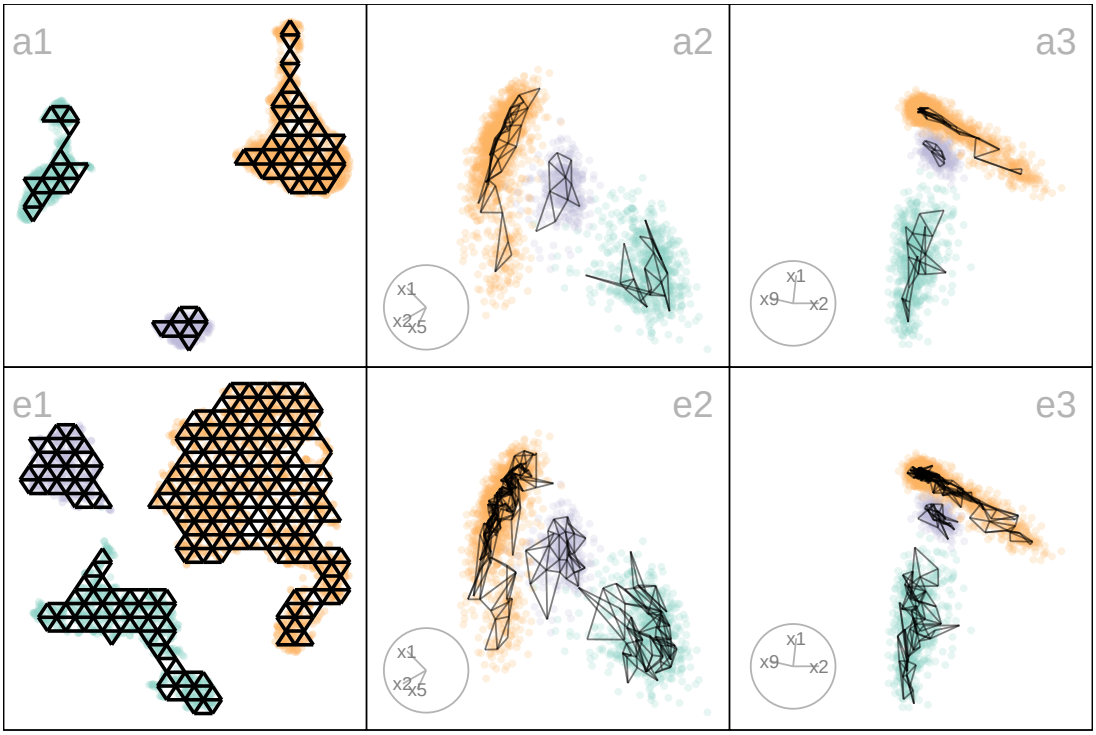

Figure 2.11: *Compare the published 2-D layout (a) made with UMAP and the 2-D layout selected by HBE plot (e) made by tSNE. The two plots on the right show projections from a tour, with the models overlaid. The published layout a suggests three very separate clusters, but this is not present in the data. While there may be three clusters, they are not well-separated. The difference in model fit also indicates this: the published layout a does not spread out fully into the point cloud like the model generated from layout e. This supports the choice that layout e is the better representation of the data, because it does not exaggerate separation between clusters.*

drawn than the others. Overall, this 2-*D* layout shows a useful way to summarize the variation in ways "1"s are drawn.

2.6 Discussion

We have developed an approach to help assess and compare NLDR layouts, generated by different methods and hyper-parameter choice(s). It depends on conceptualizing the 2-*D* layout as a model, allowing for the creation of a wireframe representation of the model that can be lifted into *p*-*D*. The fit is assessed by viewing the model in the data space, computing residuals, and HBE. Different layouts can be compared using the HBE, providing a quantitative metric to decide on the most suitable NLDR layout to represent the *p*-*D* data. Global and local preservation of structure is assessed by examining the HBE across a range of binwidths. It also provides a way to predict the values of new *p*-*D* observations in the 2-*D*, which could be useful for implementing uncertainty checks, such as using training and testing samples.

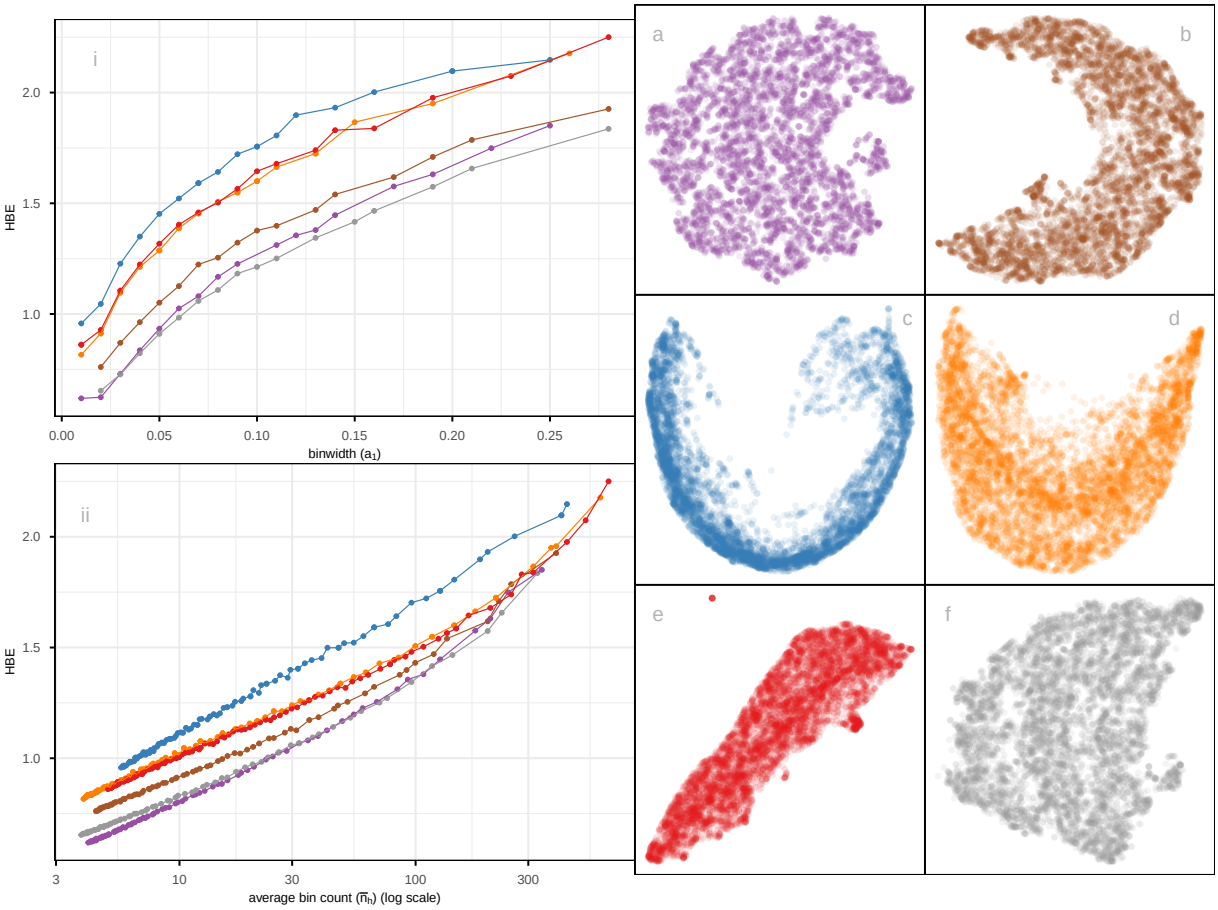

Figure 2.12: *Assessing which of the 6 NLDR layouts of the MNIST digit 1 data is the better representation using HBE for varying (i) binwidth (a_1), and (ii) average bin count ($\bar{n}_h$). Color is used for the lines and points in the left plot to match the scatterplots of the NLDR layouts (a-f). Layout c is universally poor. Layouts a, f that show a big cluster and a small circular cluster are universally optimal. Layout a performs well at tiny binwidth (where most points are in their own bin) and not as well as f with larger binwidth, thus layout f is the best choice. Plot (ii), which accounts for the density within clusters by using average bin count, shows reduced differences between layouts, indicating that part of the variation in (i) is driven by cluster density rather than true structural differences.*

The new methodology is accompanied by an R package called `quollr`, so that it is readily usable and broadly accessible. The package has methods to fit the model, compute diagnostics, and also visualize the results, with interactivity. We have primarily used the `langevitour` software (Harrison 2023) to view the model in the data space, but other tour software such as `tourr` (Wickham et al. 2011) and `detourr` (Hart and Wang 2025) could also be used.

Two examples illustrating usage are provided: the PBMC3k data, where the NLDR is summarizing clustering in *p-D*, and hand-written digits illustrating how NLDR represents an intrinsically low-dimensional nonlinear manifold. We examined a typical published usage of UMAP with the PBMC3k dataset (Chen et al. 2024). As is typical of UMAP layout with default settings, the separation between clusters is grossly exaggerated. The layout even suggests separation where there is none. Our

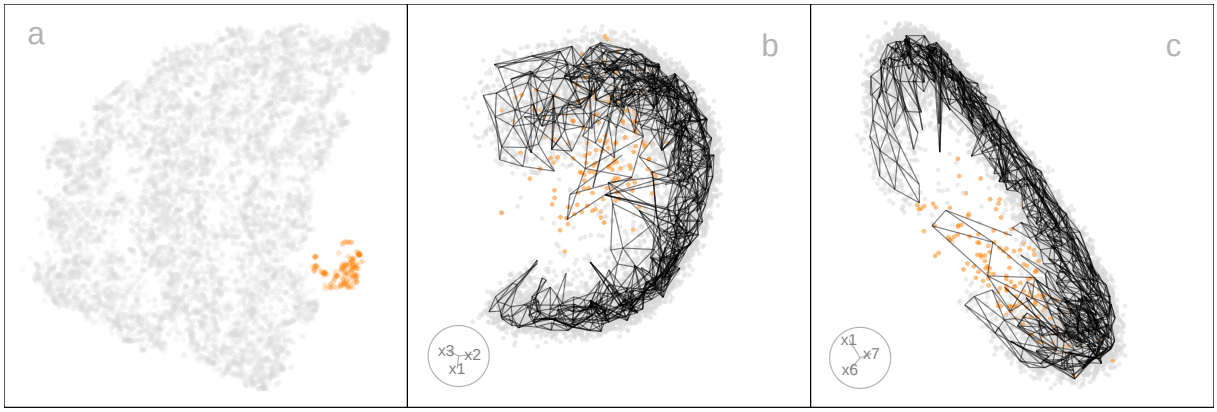

Figure 2.13: *Summary from exploring tSNE layout of the MNIST digit 1 data (Figure 2.12 a) using linked brushing. There is a big nonlinear cluster (grey) and a small cluster (orange) located very close to one corner of the big cluster in 2-D (a). The MNIST digit 1 data has a nonlinear structure in 10-D. Two 2-D projections from a tour on 10-D (b and c) reveal that the small orange cluster is actually a diffuse set of points wrapped within the grey cluster, which is C-shaped in the high dimensions.*

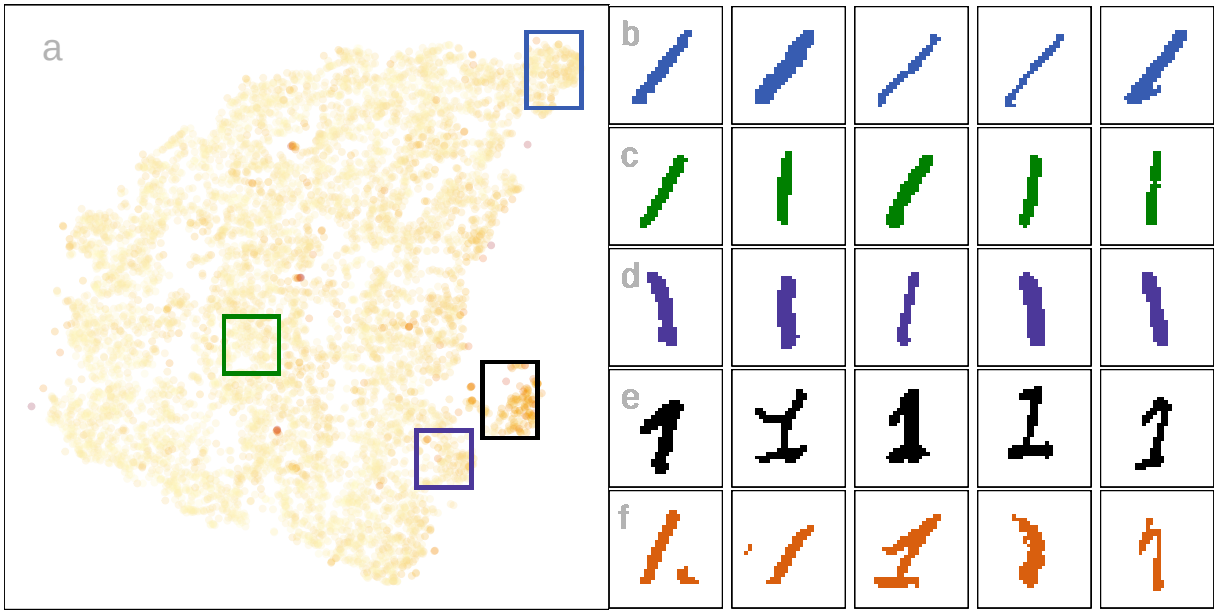

Figure 2.14: *Summary of the layout structure, and large errors, relative to the MNIST digit 1 shape: (a) layout colored by residual value, and at right (b-e) are images of samples of observations taken at locations around the layout, showing similarity in how the 1's were drawn. Set (f) are images corresponding to large residuals in the big cluster (darker orange in plot a). Along the big cluster, the angle of digit 1 changes (b-d). The small cluster has larger residuals, and the images show that these tend to be European style with a flag at the top and a base at the bottom. The set in (f) shows various poorly written digits.*

approach provides a way to choose a reasonable layout and avoids the use of misleading layouts in the future. In the hand-written digits (LeCun et al. 1998), we illustrate how our model fit statistics show that a flat disc layout is superior to the curved-shaped layouts, and how to identify oddly written "1"s using the residuals of the fitted model.

This work can be applied with existing metrics for evaluating NLDR layout, such as ARNX, RTA, SC, and RGS. It provides an additional evaluation metric, and importantly allows any layout to be viewed in the p-D data space. This latter aspect can help to disentangle conflicting suggestions by the different metrics.

Additional exploration of distance measures to summarize the fit could be a valuable direction for future work. We have used Euclidean distance, but other measures, such as geodesic distances (Tenenbaum et al. 2000), may better capture curved or nonlinear relationships in the data and are worth exploring.

This work has also revealed some interesting behaviors of NLDR methods, including twisting, flattened "pancake" clusters in p-D, and severe effects of density differences. These are described in more detail in the supplementary materials.

Researchers usually use 2-D layouts, but if a k-D ($k > 2$) layout is provided, the approach developed here could be extended. Potential approaches include 3-D binning, k-means clustering, or even special implementations of convex hulls.

2.7 Supplementary materials

All the materials to reproduce the chapter can be found at https://github.com/JayaniLakshika/paper-nldr-vis-algorithm.

The supplementary materials provide additional details on the methods and hyper-parameters used to generate layouts, video links of animated p-D tours, notation summaries, and the R and Python scripts used in the study. They also describe the generation of the 2NC7 data, computation of hexagon grid configurations, and data binning procedures. Further sections highlight interesting NLDR behaviors observed in the data space and compare HBE with existing evaluation metrics for the PBMC3k and MNIST datasets.

The R package `quollr`, available on CRAN and at https://jayanilakshika.github.io/quollr/, provides software accompanying this chapter to fit the wireframe model representation, compute diagnostics, visualize the model in the data with `langevitour`, and link multiple plots interactively.

2.8 Acknowledgments

These `R` packages were used for the work: `tidyverse` (Wickham et al. 2019), `Rtsne` (Krijthe 2015), `umap` (Konopka 2023), `patchwork` (Pedersen 2024), `colorspace` (Zeileis et al. 2020),

`langevitour` (Harrison 2023), `conflicted` (Wickham 2023), `reticulate` (Ushey et al. 2024), `kableExtra` (Zhu 2024). These Python packages were used for the work: `trimap` (Amid and Warmuth 2019) and `pacmap` (Wang et al. 2021). The article was created with `R` packages `quarto` (Allaire and Dervieux 2024).

Chapter 3

quollr: An R Package for Visualizing 2-D Models from Nonlinear Dimension Reductions in High-Dimensional Space

Nonlinear dimension reduction methods provide a low-dimensional representation of high-dimensional data by applying a nonlinear transformation. However, the complexity of the transformations and data structures can create wildly different representations depending on the method and hyper-parameter choices. It is difficult to determine whether any of these representations are accurate, which one is the best, or whether they have missed important structures. The R package `quollr` has been developed as a new visual tool to determine which method and which hyper-parameter choices provide the most accurate representation of high-dimensional data. The `scurve` data from the package is used to illustrate the algorithm. Single-cell RNA sequencing (scRNA-seq) data from mouse limb muscles are used to demonstrate the usability of the package.

3.1 Introduction

Nonlinear dimension reduction (NLDR) techniques, such as t-distributed stochastic neighbor embedding (tSNE) (Maaten and Hinton 2008), uniform manifold approximation and projection (UMAP) (McInnes et al. 2018), potential of heat-diffusion for affinity-based trajectory embedding (PHATE) algorithm (Moon et al. 2019), large-scale dimensionality reduction using triplets (TriMAP) (Amid and Warmuth 2019), and pairwise controlled manifold approximation (PaCMAP) (Wang et al. 2021), can create hugely different representations depending on the selected method and hyper-parameter

choices. It is difficult to determine whether any of these representations are accurate, which one is the best, or whether they have missed important structures.

This chapter presents the R package, `quollr`, which is useful for understanding how NLDR warps high-dimensional space and fits the data. Starting with an NLDR layout, our approach is to create a 2-*D* wireframe model representation that can be lifted and displayed in the high-dimensional (*p*-*D*) space (Figure 3.1).

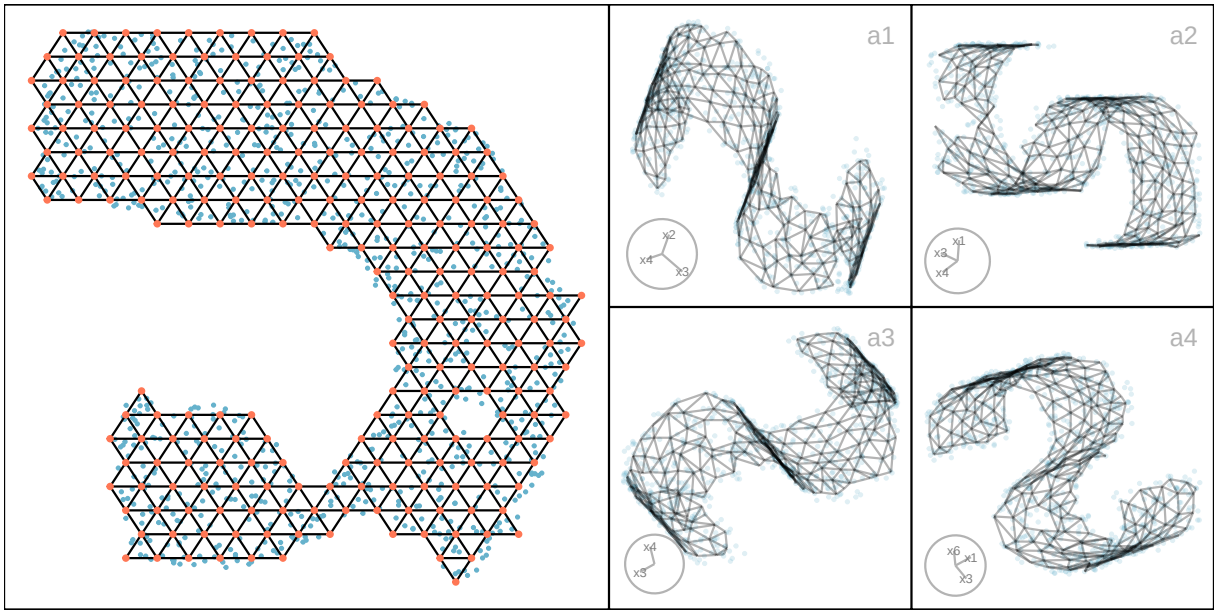

Figure 3.1: *Wireframe model representation of the NLDR layout, lifted and displayed in high-dimensional space. The left panel shows the NLDR layout with a triangular mesh overlay, forming the wireframe structure. This mesh can be lifted into higher dimensions and projected to examine how the geometric structure of the data is preserved. Panels (a1–a4) display different* 2-*D projections of the lifted wireframe, where the underlying curved sheet structure of the data is more clearly visible. The triangulated mesh highlights how local neighborhoods in the layout correspond to relationships in the high-dimensional space, enabling diagnostics of distortion and preservation across dimensions.*

The chapter is organized as follows. The usage section explains how users can fit a complete model pipeline from hexagonal binning of a 2-*D* layout to lifting the representation back into *p*-*D* and how the resulting objects can be explored interactively using tour. The next section introduces the implementation of the `quollr` package on CRAN, including a demonstration of the package's key functions and visualization capabilities. In the application section, we illustrate the algorithm's functionality for studying a clustering data structure. Finally, we conclude the chapter with a brief summary and discuss potential opportunities for using our algorithm.

3.2 Usage

The package is available on CRAN, and the development version is available at https://github.com/JayaniLakshika/quollr.

Our algorithm includes the following steps: (1) scaling the NLDR data, (2) computing configurations of a hexagon grid, (3) binning the data, (4) obtaining the centroids of each bin, (5) indicating neighboring bins with line segments that connect the centroids, and (6) lifting the model into high dimensions. A detailed description of the algorithm can be found in Gamage et al. (2025c).

The user needs two inputs: the high-dimensional dataset and the corresponding NLDR data. The high-dimensional data must contain a unique `ID` column, with data columns prefixed by the letter `x` (e.g., `x1`, `x2`, etc.). The NLDR dataset should include embedding coordinates labeled as `emb1` and `emb2`, ensuring one-to-one correspondence with the high-dimensional data through the shared `ID`. The built-in example datasets, `scurve` and `scurve_umap`, demonstrate these structures.

To run the entire model pipeline, we can use the `fit_high_model()` function. This function requires: the high-dimensional data (`highd_data`), the embedding data (`nldr_data`), the number of bins along the x-axis (`b1`), the buffer amount as a proportion of the data (`q`), and a benchmark value to identify high-density hexagons (`hd_thresh`).

The function returns an object of class `highd_vis_model` containing the scaled NLDR object (`nldr_scaled_obj`) with three elements: the first is the scaled NLDR data (`scaled_nldr`), and the second and third are the limits of the original NLDR data (`lim1` and `lim2`); the hexagonal object (`hb_obj`), the fitted model in both 2-*D* (`model_2d`), and *p*-*D* (`model_highd`), and triangular mesh (`trimesh_data`).

```
model_obj <- fit_highd_model(
  highd_data = scurve,
  nldr_data = scurve_umap,
  b1 = 21,
  q = 0.1,
  hd_thresh = 0)
```

The resulting model can then be shown in a tour using a two-step process:

```
combined_data <- comb_data_model(
  highd_data = scurve,
  model_highd = model_obj$model_highd,
  model_2d = model_obj$model_2d
)

tour_view <- show_langevitour(
  point_data = combined_data,
  edge_data = model_obj$trimesh_data
)
```

which produces the model and data plot shown in Figure 3.3.

3.3 Implementation

The implementation of `quollr` is designed to be efficient and easy to extend. The package is organized into a series of logical components that reflect the main stages of the workflow: data preprocessing, model fitting, low-density bin removal, prediction, visualization, and interactive exploration (Figure 4.1). This package structure makes the code easier to maintain and allows new features to be added without changing the existing functionality.

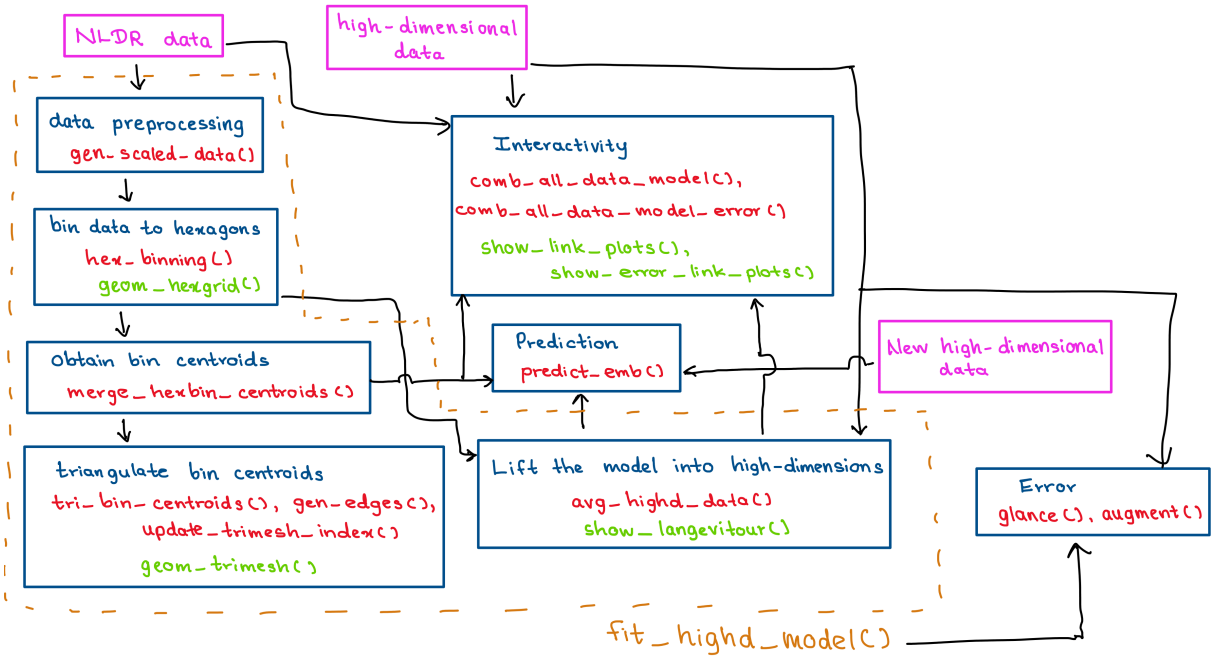

Figure 3.2: *Overview of the* `quollr` *workflow and software architecture. The process begins with NLDR and p-D data inputs, followed by data preprocessing and hexagonal binning. Centroids are computed and triangulated to form the 2-D mesh, which is then lifted into the p-D space. Predictions and error computations are performed on new data, while interactive functions enable dynamic linking between the p-D and 2-D representations.*

3.3.1 Software architecture

The package is organized into seven core modules corresponding to stages of the workflow: preprocessing, 2-*D* model construction, lifting the model into *p*-*D*, prediction, error computation, visualizations, and interactivity. Each module performs a distinct task and communicates through data objects.

1. Data preprocessing: The function `gen_scaled_data()` standardizes the embedding data, manages variable naming, and ensures consistent identifiers across high-dimensional and embedded datasets.

2. Construct 2-*D* model: A series of functions `hex_binning()`, `merge_hexbin_centroids()`, `tri_bin_centroids()`, `gen_edges()`, and `update_trimesh_index()` generate the hexagonal grid, compute bin centroids, and connect the triangular mesh that defines local neighborhoods in the 2-*D* space.

3. Lift the model into *p*-*D*: The function `avg_highd_data()` computes the average of the high-dimensional variables for each bin, linking the 2-*D* representation back to the original data space.

4. Prediction: The function `predict_emb()` estimates the embedding of new high-dimensional observations based on the fitted model.

5. Error computation: The `glance()` and `augment()` function summarizes model performance by comparing the predicted and original embeddings.

6. Visualizations: Functions such as `geom_hexgrid()`, `geom_trimesh()`, and `show_langevitour()` provide tools for exploring the fitted models through static and dynamic visualizations.

7. Interactivity: The functions `comb_all_data_model()` and `show_link_plots()` generate interactive linked visualizations that connect the 2-D NLDR layout, the corresponding tour view, and the fitted model. Similarly, `comb_all_data_model_error()` and `show_error_link_plots()` integrate the error distribution with the 2-D embedding and tour view, enabling interactivity across multiple plots.

Each module is internally independent but connected through data objects (see next section). This modular design simplifies maintenance and allows developers to extend individual components, such as substituting different binning approaches, extracting centroids, or using visualization tools, without altering the overall workflow.

3.3.2 Data objects

The internal data objects follow the tidy data principle: each variable is stored in a column, each observation in a row, and each type of information in its own table. This structure makes the package easy to use with the `tidyverse` and other visualization tools.

Input objects

- `highd_data`: a tibble containing the original high-dimensional observations with a unique identifier (`ID`) and variable columns prefixed with `x` (e.g., `x1`, `x2`, ...).

- `nldr_data`: a tibble containing two-dimensional embeddings, labeled as `emb1` and `emb2`, matched to the same `ID`s.

Generated objects

- `scaled_nldr_obj`: the output of `gen_scaled_data()`, which rescales the embedding to the range $[0, 1] \times [0, y_{2,\max}]$, where $y_{2,\max} = r_2/r_1$ is the ratio of the embedding ranges. It includes the scaled coordinates (`scaled_nldr`) and the original limits (`lim1`, `lim2`).

- `hex_bin_obj`: the object created by `hex_binning()`, which includes hexagon grid configurations. It includes the binwidth (`a1`), binheight (`a2`), the number of bins along each axis (`b1`, `b2`), the centroids of all hexagons, polygon coordinates, and the assignment of each data point to the hexagon.

- `highd_vis_model`: the main model object returned by `fit_highd_model()`. It stores all components of the fitted model, including the scaled NLDR data (`nldr_scaled_obj`), the hexagonal bin configurations (`hb_obj`), the averaged p-D summaries for each bin (`model_highd`), the corresponding 2-D bin centroids (`model_2d`), and the triangulated mesh connecting neighboring bins (`trimesh_data`).

3.3.3 Computational efficiency and optimization

Several core computations within `quollr` are optimized using compiled C++ code via the `Rcpp` and `RcppArmadillo` packages. While the user interacts with high-level R functions, performance-critical steps such as nearest-neighbor searches (`compute_highd_dist()`), error metrics (`compute_errors()`), 2-D distance calculations (`calc_2d_dist_cpp()`), and generation of hexagon coordinates (`gen_hex_coord_cpp()`) are handled internally in C++. This design provides significant speedups when analyzing large datasets while maintaining a user-friendly R interface. These C++ functions are not exported but are bundled within the package and fully accessible for inspection in the source code.

3.3.4 Pipeline implementation

In this section, we demonstrate the implementation of each step of the pipeline discussed in the Usage section (`fit_highd_model` and `comb_data_model`). Each step can be run independently to ensure flexibility in the modelling approach.

The algorithm starts by scaling the NLDR data to the range $[0, 1] \times [0, y_{2,max}]$, where $y_{2,max} = r_2/r_1$ is the ratio of ranges of embedding components. The output includes the scaled NLDR data (`scaled_nldr`) along with the original limits of the embeddings (`lim1`, `lim2`).

```
scurve_umap_obj <- gen_scaled_data(nldr_data = scurve_umap)
```

Hexagon binning

The hexagon binning process builds a complete hexagonal tessellation over the 2-D NLDR embedding and assigns points to bins for downstream modeling. The workflow proceeds through several steps,

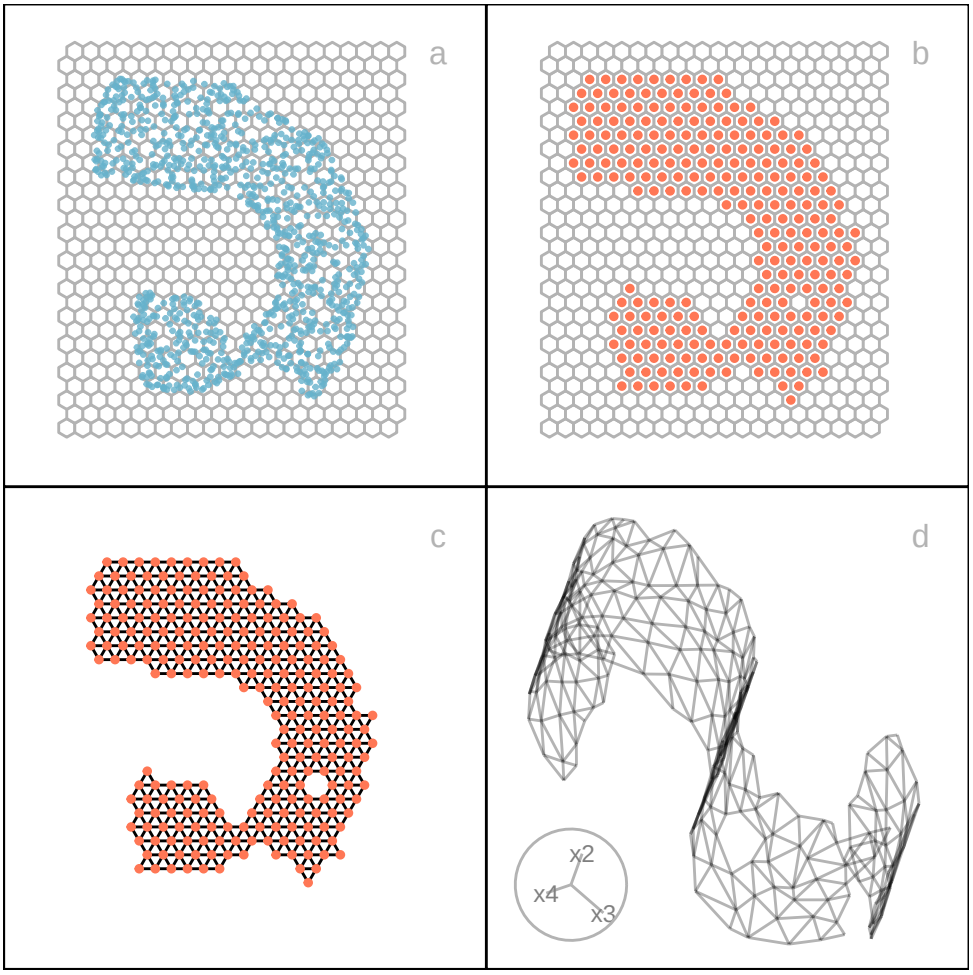

Figure 3.3: *Key steps for constructing the model on the UMAP layout: (a) hexagon bins, (b) bin centroids, (c) triangulated centroids, and (d) lifting the model into high dimensions. The* `scurve` *data is shown.*

beginning with selecting an appropriate grid configuration and ending with assigning NLDR points to their corresponding hexagons.

The first step is to determine the configuration of the hexagonal grid. The function `calc_bins_y()` computes the number of bins along the y-axis (`b2`), the hexagon width (`a1`), and the hexagon height (`a2`), based on three inputs: (i) the scaled NLDR object (`nldr_scaled_obj`, containing the scaled embedding and its 2-*D* limits), (ii) the chosen number of bins along the x-axis (`b1`), and (iii) a buffer proportion (`q`) that expands the grid slightly beyond the observed data range. This buffer ensures that the tessellation fully covers the embedding. By default, `q = 0.1`, but in practice, it must be chosen to avoid exceeding the domain of the scaled NLDR data.

```
bin_configs <- calc_bins_y(
  nldr_scaled_obj = scurve_umap_obj,
  b1 = 21,
```

```
  q = 0.1)


bin_configs
```

```
> $b2
> [1] 28
>
> $a1
> [1] 0.05869649
>
> $a2
> [1] 0.05083265
```

Once the grid configurations are known, the next step is to construct the full set of hexagon centroids. The function `gen_centroids()` uses the bin configuration and the embedding limits to determine the horizontal and vertical spacing of rows, including the staggered row offsets required for hexagonal tiling. Odd rows place centroids directly above one another, while even rows shift by half the hexagon width to create the interlocking structure. Vertical spacing is determined by $v_s = (\sqrt{3}/2)a_1$, with starting positions defined by the buffer-adjusted limits. This produces a complete grid of potential hexagons covering the 2-D space.

```
all_centroids_df <- gen_centroids(
  nldr_scaled_obj = scurve_umap_obj,
  b1 = 21,
  q = 0.1
)


head(all_centroids_df, 5)
```

```
> # A tibble: 5 x 3
>       h     c_x    c_y
>   <int>   <dbl>  <dbl>
> 1     1 -0.1    -0.116
> 2     2 -0.0413 -0.116
> 3     3  0.0174 -0.116
```

```
> 4     4  0.0761 -0.116
> 5     5  0.135  -0.116
```

After determining the centroid locations, each hexagon's polygonal boundary is generated using the `gen_hex_coord()` function, which constructs the six vertices of every hexagon by applying fixed geometric offsets derived from the hexagon width, a_1. Specifically, the horizontal and vertical offsets are defined as $dx = a_1/2$, $dy = a_1/\sqrt{3}$, and $vf = a_1/(2\sqrt{3})$, and these constants are used to position each vertex relative to the centroid, producing the complete set of hexagon boundaries.

These constants define how far each vertex lies from the centroid in the six characteristic directions. For efficiency, the vertex generation is implemented in C++ and returns a tibble listing all polygon vertices, uniquely indexed by hexagons.

```
all_hex_coord <- gen_hex_coord(
  centroids_data = all_centroids_df,
  a1 = bin_configs$a1
)

head(all_hex_coord, 5)
```

```
>   h           x           y
> 1 1 -0.10000000 -0.08179171
> 2 1 -0.12934824 -0.09873593
> 3 1 -0.12934824 -0.13262436
> 4 1 -0.10000000 -0.14956858
> 5 1 -0.07065176 -0.13262436
```

The next step is to assign each NLDR point to its closest hexagon. The `assign_data()` function computes all pairwise Euclidean distances between the 2-D NLDR embedding and the centroid coordinates, identifying the nearest centroid for every observation. Each point is assigned a hexagon index (h), producing a version of the embedding annotated with bin membership.

```
umap_hex_id <- assign_data(
  nldr_scaled_obj = scurve_umap_obj,
  centroids_data = all_centroids_df
)
```

```
head(umap_hex_id, 5)
```

```
> # A tibble: 5 x 4
>    emb1  emb2    ID     h
>   <dbl> <dbl> <int> <int>
> 1 0.277 0.913     1   427
> 2 0.697 0.538     2   287
> 3 0.779 0.399     3   226
> 4 0.173 0.953     4   446
> 5 0.218 0.983     5   468
```

Finally, the list of points within each hexagon can be extracted using `group_hex_pts()`. This step collapses the point-level assignments into a hexagon-level summary by grouping by `h` and collecting the IDs of all points belonging to that polygon. The result is a tidy representation of the binning structure.

```
pts_df <- group_hex_pts(
  scaled_nldr_hexid = umap_hex_id
)

head(pts_df, 5)
```

```
> # A tibble: 5 x 2
>       h pts_list
>   <int> <list>
> 1    58 <int [4]>
> 2    68 <int [1]>
> 3    69 <int [5]>
> 4    70 <int [6]>
> 5    71 <int [9]>
```

Although each component of the workflow can be run independently, the `hex_binning()` function automates the entire sequence—from grid construction to point assignment—and returns a `hex_bin_obj` containing the bin dimensions (`a1`, `a2`), the full grid geometry (`centroids`,

`hex_poly`), bin membership (`data_hb_id`), standardized counts (`std_cts`), the total number of bins (`b`), non-empty bins (`m`), and the list of points per bin (`pts_bins`).

```
hb_obj <- hex_binning(
  nldr_scaled_obj = scurve_umap_obj,
  b1 = 21,
  q = 0.1)
```

Computing the standardized number of points within each hexagon

The `compute_std_counts()` function calculates both the raw and standardized counts of points inside each hexagon.

The function begins by grouping the data by hexagon (`h`) and counting the number of NLDR points falling within each bin. These raw counts are stored as `n_h`. To enable comparisons across bins with varying densities, the function then standardizes these counts by dividing each bin's count by the maximum count across all bins. This yields a standardized bin counts, `w_h`, ranging from 0 to 1.

```
std_df <- compute_std_counts(
  scaled_nldr_h = umap_hex_id
  )

head(std_df, 5)
```

```
> # A tibble: 5 x 3
>       h   n_h   w_h
>   <int> <int> <dbl>
> 1    58     4 0.004
> 2    68     1 0.001
> 3    69     5 0.005
> 4    70     6 0.006
> 5    71     9 0.009
```

Obtaining bin centroids

The `merge_hexbin_centroids()` function combines hexagonal bin coordinates, raw and standardized counts within each hexagons.

This function performs a full join with `centroids_data`, aligning hexagons (`h`) between the two datasets to incorporate both hexagonal bin centroids (`h`) and count metrics. After merging, the function handles missing values in the count columns: any `NA` values in `w_h` or `n_h` are replaced with zeros. This ensures that hexagons with no assigned data points are retained in the output, with zero values for count-related fields. The resulting data contains the full set of hexagon centroids along with associated bin counts (`n_h`) and standardized counts (`w_h`).

```
df_bin_centroids <- merge_hexbin_centroids(
  centroids_data = all_centroids_df,
  counts_data = hb_obj$std_cts
  )

head(df_bin_centroids, 5)
```

```
>   h          c_x         c_y n_h w_h
> 1 1 -0.10000000 -0.1156801   0   0
> 2 2 -0.04130351 -0.1156801   0   0
> 3 3  0.01739298 -0.1156801   0   0
> 4 4  0.07608947 -0.1156801   0   0
> 5 5  0.13478596 -0.1156801   0   0
```

Indicating neighbors by line segments connecting centroids

To represent the neighborhood structure of hexagonal bins in a 2-*D* layout, we employ Delaunay triangulation (Gebhardt et al. 2024; Lee and Schachter 1980) on the centroids of hexagons.

The `tri_bin_centroids()` function generates a triangulation object from the x and y coordinates of hexagon centroids using the `interp::tri.mesh()` function (Gebhardt et al. 2024).

```
tr_object <- tri_bin_centroids(
  centroids_data = df_bin_centroids
  )
```

The `gen_edges()` function uses this triangulation object to extract line segments between neighboring bins. It constructs a unique set of bin-to-bin connections by identifying the triangle edges and filtering duplicate or reversed links. Each edge is then annotated with its start and end coordinates and the Euclidean distance between the coordinates.

```
trimesh <- gen_edges(tri_object = tr_object, a1 = hb_obj$a1)


head(trimesh, 5)
```

```
> # A tibble: 5 x 8
>    from    to  x_from  y_from    x_to    y_to from_count to_count
>   <int> <int>   <dbl>   <dbl>   <dbl>   <dbl>      <dbl>    <dbl>
> 1     1     2 -0.1    -0.116  -0.0413 -0.116           0        0
> 2    22    23 -0.0707 -0.0648 -0.0120 -0.0648          0        0
> 3    22    44 -0.0707 -0.0648 -0.0413 -0.0140          0        0
> 4     3    23  0.0174 -0.116  -0.0120 -0.0648          0        0
> 5    44    45 -0.0413 -0.0140  0.0174 -0.0140          0        0
```

The `update_trimesh_index()` function re-indexes the node IDs to ensure that edge identifiers are sequentially numbered and consistent with downstream analysis. This sequential ordering of edges is essential because software such as `langevitour` and `detourr` rely on one-to-one mapping between edges and their corresponding vertices to visualize the mesh.

```
trimesh <- update_trimesh_index(trimesh_data = trimesh)


head(trimesh, 5)
```

```
> # A tibble: 5 x 10
>    from    to  x_from  y_from    x_to    y_to from_count to_count from_reindexed
>   <int> <int>   <dbl>   <dbl>   <dbl>   <dbl>      <dbl>    <dbl>          <int>
> 1     1     2 -0.1    -0.116  -0.0413 -0.116           0        0              1
> 2    22    23 -0.0707 -0.0648 -0.0120 -0.0648          0        0             22
> 3    22    44 -0.0707 -0.0648 -0.0413 -0.0140          0        0             22
> 4     3    23  0.0174 -0.116  -0.0120 -0.0648          0        0              3
> 5    44    45 -0.0413 -0.0140  0.0174 -0.0140          0        0             44
> # i 1 more variable: to_reindexed <int>
```

Identifying and removing low-density hexagons

Not all hexagons contain meaningful information. Some may have very few or no data points due to the sparsity or shape of the underlying structure. Simply removing hexagons with low counts (e.g.,

fewer than a fixed threshold) can lead to gaps or "holes" in the 2-*D* structure, potentially disrupting the continuity of the representation.

To address this, we propose a more nuanced method that evaluates each hexagon not only based on its own density, but also in the context of its immediate neighbors. The `find_low_dens_hex()` function identifies hexagonal bins with insufficient local support by calculating the average standardized count across their six neighboring bins. If this means neighborhood density is below a user-defined threshold (e.g., 0.05), the hexagon is flagged for removal.

The `find_low_dens_hex()` function relies on a helper, `compute_mean_density_hex()`, which iterates over all hexagons and computes the average density across neighbors based on their hexagon (`h`) and a defined number of bins along the x-axis (`b1`). The hexagonal layout assumes a fixed grid structure, so neighbor IDs are computed by positional offsets.

```
low_density_hex <- find_low_dens_hex(
  model_2d = df_bin_centroids,
  b1 = 21,
  md_thresh = 0.05
)
```

For simplicity, we remove low-density hexagons using a threshold of 0.

```
df_bin_centroids <- df_bin_centroids |>
  dplyr::filter(n_h > 0)

trimesh <- trimesh |>
  dplyr::filter(from_count > 0,
                to_count > 0)

trimesh <- update_trimesh_index(trimesh)
```

Lifting the model into high dimensions

The final step involves lifting the fitted 2-*D* model into *p*-*D*. This is done by modelling a point in *p*-*D* as the *p*-*D* mean of data points in the 2-*D* centroid. This is performed using the `avg_highd_data()` function, which takes *p*-*D* data (`highd_data`) and embedding data with their corresponding hexagonal bin IDs as inputs (`scaled_nldr_hexid`).

```
df_bin <- avg_highd_data(
  highd_data = scurve,
  scaled_nldr_hexid = hb_obj$data_hb_id
)

head(df_bin, 5)
```

```
> # A tibble: 5 x 8
>       h     x1     x2    x3       x4       x5       x6       x7
>   <int>  <dbl>  <dbl> <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
> 1    58 -0.371 1.91    1.92 -0.00827 0.00189   0.0170   0.00281
> 2    68  0.958 0.0854  1.29  0.00265 0.0171    0.0876  -0.00249
> 3    69  0.855 0.0917  1.51  0.00512 0.000325 -0.0130  -0.00395
> 4    70  0.731 0.129   1.68 -0.00433 0.00211  -0.0356  -0.00240
> 5    71  0.474 0.108   1.88 -0.00260 0.000128  0.00785  0.00170
```

3.3.5 Prediction

The `predict_emb()` function is used to predict a point in a 2-*D* embedding for a new *p*-*D* data point using the fitted model. This function is useful to predict 2-*D* embedding irrespective of the NLDR technique.

In the prediction process, first, the nearest *p*-*D* model point is identified for the new *p*-*D* data point by computing *p*-*D* Euclidean distance. Then, the corresponding 2-*D* bin centroid mapping for the identified *p*-*D* model point is determined. Finally, the coordinates of the identified 2-*D* bin centroid are used as the predicted NLDR embedding for the new *p*-*D* data point.

To accelerate this process, the nearest-neighbor search is implemented in C++ using `Rcpp` via the internal function `compute_highd_dist()`.

```
predict_data <- predict_emb(
  highd_data = scurve,
  model_2d = df_bin_centroids,
  model_highd = df_bin
  )

head(predict_data, 5)
```

```
> # A tibble: 5 x 4
>   pred_emb_1 pred_emb_2    ID pred_h
>        <dbl>      <dbl> <int>  <int>
> 1      0.252      0.901     1    427
> 2      0.692      0.545     2    287
> 3      0.780      0.393     3    226
> 4      0.164      0.952     4    446
> 5      0.193      1.00      5    468
```

It is worth noting that while `predict_emb()` provides a general approach that works across methods, some NLDR techniques have their own built-in prediction mechanisms. For example, UMAP (Konopka 2023) supports direct prediction of embeddings for new data once a model is fitted.

3.3.6 Compute residuals and hexbin error (HBE)

Hexbin error (HBE) serves as a goodness-of-fit metric for evaluating the high-dimensional to 2-*D* mapping. The `glance()` function computes these summary diagnostics by combining the fitted model returned by `fit_highd_model()` with the *p*-*D* data. After renaming the model output to avoid join conflicts, `predict_emb()` is used to assign each observation to a hexagon in the 2-*D* embedding. The resulting bin assignments are joined with both the model output and the *p*-*D* data, allowing squared and absolute differences between the true and modeled *p*-*D* coordinates to be computed for every dimension. Total absolute error (`Error`) and hexbin error (`HBE`) are then obtained using an efficient C++ implementation (`compute_errors()`), and returned in a tidy tibble.

```
glance(
  x = scurve_model_obj,
  highd_data = scurve
)
```

```
> # A tibble: 1 x 2
>   Error   HBE
>   <dbl> <dbl>
> 1  196. 0.116
```

The `augment()` function provides point-level diagnostics by appending predictions and residuals to the original *p*-*D* data. Using the same prediction process as `glance()`, it computes dimension-wise

residuals, squared errors, and absolute errors, along with two aggregate measures per observation: the total squared error (`row_wise_total_error`) and the total absolute error (`row_wise_abs_error`). The final output is a tibble containing IDs (`ID`), *p-D* data, predicted hexagons (`h`), modeled coordinates, and all residual diagnostics, with one row per observation.

```
model_error <- augment(
  x = scurve_model_obj,
  highd_data = scurve
)
```

3.3.7 Visualizations

The package offers several 2-*D* visualizations (Figure 3.4), including a full hexagonal grid, a hexagonal grid that matches the data, a full grid based on centroid triangulation, a centroid triangulation grid that aligns with the data, and a triangular mesh for any provided set of points.

The generated *p-D* model, overlaid with the data, can also be visualized using `show_langevitour`. Additionally, it features a function for visualizing the 2-*D* projection of the fitted model overlaid on the data, called `plot_proj`.

Furthermore, there are two interactive plots, `show_link_plots` and `show_error_link_plots`, which are designed to help diagnose the model. Each visualization can be generated using its respective function, as described in this section.

Hexagonal grid

The `geom_hexgrid()` function introduces a custom `ggplot2` layer designed for visualizing a hexagonal grid on a provided set of bin centroids.

To display the complete grid, users should supply all available bin centroids (Figure 3.4 a).

```
full_hexgrid <- ggplot() +
  geom_hexgrid(
    data = hb_obj$centroids,
    aes(x = c_x, y = c_y)
    )
```

If the goal is to plot only the subset of hexagons that correspond to bins containing data points, then only the centroids associated with those bins should be passed (Figure 3.4 b).

```
data_hexgrid <- ggplot() +
  geom_hexgrid(
    data = df_bin_centroids,
    aes(x = c_x, y = c_y)
    )
```

Triangular mesh

The `geom_trimesh()` function introduces a custom `ggplot2` layer designed for visualizing 2-*D* wireframe on a provided set of bin centroids.

To display the complete wireframe, users should supply all available bin centroids (Figure 3.4 c).

```
full_triangulation_grid <- ggplot() +
  geom_trimesh(
    data = hb_obj$centroids,
    aes(x = c_x, y = c_y)
    )
```

If the goal is to plot only the subset of hexagons that correspond to bins containing data points, then only the centroids associated with those bins should be passed (Figure 3.4 d).

```
data_triangulation_grid <- ggplot() +
  geom_trimesh(
    data = df_bin_centroids,
    aes(x = c_x, y = c_y)
    )
```

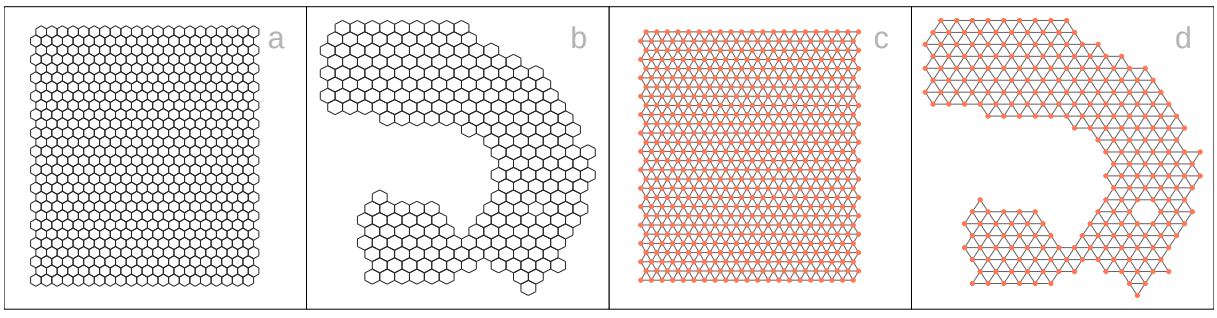

Figure 3.4: *The outputs of `geom_hexgrid` and `geom_trimesh` include: (a) a complete hexagonal grid, (b) a hexagonal grid that corresponds with the data, (c) a full grid based on centroid triangulation, and (d) a centroid triangulation grid that aligns with the data.*

p-D model visualization

To visualize how well the *p-D* model captures the underlying structure of the high-dimensional data, we provide a tour of the model in *p-D* using the `show_langevitour()` function (Figure 3.5). This function renders a dynamic projection of both the high-dimensional data and the model using the `langevitour` R package (Harrison 2023).

Before plotting, the data needs to be organized into a combined format through the `comb_data_model()` function. This function takes three inputs: `highd_data` (the high-dimensional observations), `model_highd` (high-dimensional summaries for each bin), and `model_2d` (the hexagonal bin centroids of the model). It returns a tidy data frame combining both the data and the model.

In this structure, the `type` variable distinguishes between original observations (`data`) and the bin-averaged model representation (`model`).

```
df_exe <- comb_data_model(
  highd_data = scurve,
  model_highd = df_bin,
  model_2d = df_bin_centroids
  )
```

The `show_langevitour()` function then renders the visualization using the `langevitour` interface, displaying both types of points in a dynamic tour. The `edge_data` input defines connections between neighboring bins (i.e., the hexagonal edges) to visualize the model's structure.

```
show_langevitour(
  point_data = df_exe,
  edge_data = trimesh
  )
```

As an alternative to `langevitour`, users can explore the fitted *p-D* model using the `detourr` (Hart and Wang 2025) (Figure 3.6). The combined data object from `comb_data_model()` can be passed directly to the `detour()` function, where `tour_aes()` defines the projection variables and color mapping. The visualization is rendered using `show_scatter()`, which can display both data points and the model's structural edges via the `edges` argument.

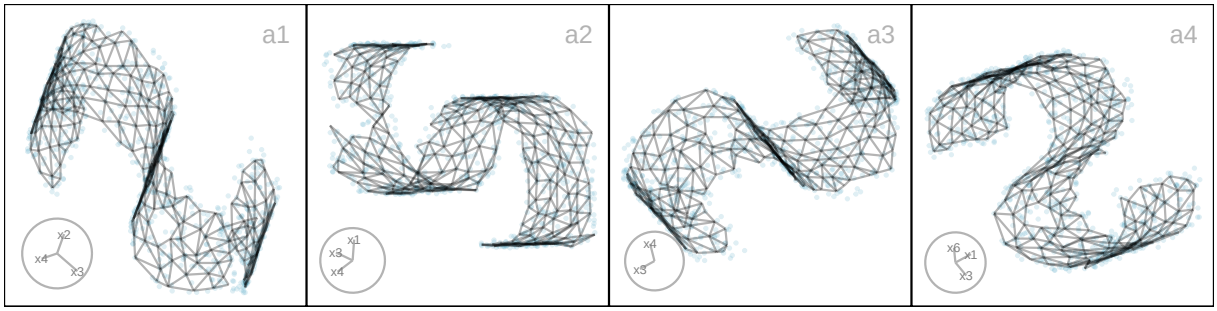

Figure 3.5: *2-D projections of the lifted high-dimensional wireframe model from the* `scurve` *UMAP layout. Each panel (a1–a4) shows the model (black) overlaid on* `scurve` *data (blue) in different projections. These views illustrate how the lifted wireframe model captures the structure of the* `scurve` *data. Regions with sparse or no data in the UMAP layout are also visible in the lifted model.*

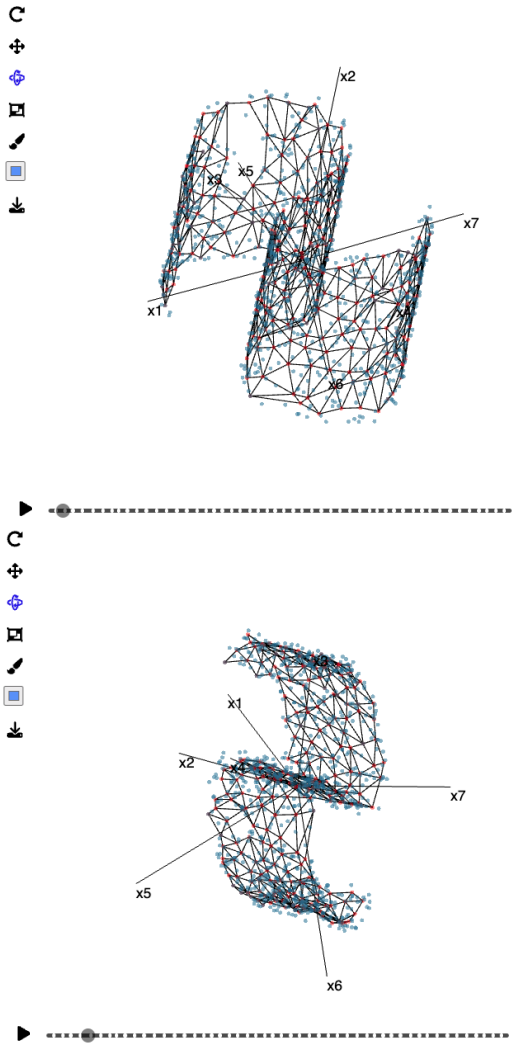

Figure 3.6: *Screenshots of the lifted high-dimensional wireframe model from the* `scurve` *UMAP layout using* `detourr`*. Regions with sparse or no data in the UMAP layout are also visible in the lifted model.*

```
detour(
  df_exe,
  tour_aes(
    projection = starts_with("x"),
    colour = type
  )
) |>
  tour_path(grand_tour(2),
                    max_bases=50, fps = 60) |>
  show_scatter(axes = TRUE, size = 1.5, alpha = 0.5,
               edges = as.matrix(trimesh[, c("from_reindexed",
                                             "to_reindexed")]),
               palette = c("#66B2CC", "#FF7755"),
               width = "600px", height = "600px")
```

In the resulting interactive visualization, blue points represent the high-dimensional data, orange points represent the model centroids from each bin, and the lines between model points reflect the 2-*D* wireframe structure mapped to high-dimensional space.

Linked plots

Two types of interactively linked plots can be generated to assess the model fits everywhere, or better in some subspaces, or completely mismatch the data. The plots are linked using `crosstalk`, which allows interactive brushing: selecting or brushing points in one plot automatically highlights the corresponding points in the other linked views.

First, the function `show_link_plots()` provides linking a 2-*D* NLDR layout and a tour (via `langevitour`) of the model overlaid by the data (Figure 3.7).

The `point_data` for `show_link_plots()` can be prepared using the `comb_all_data_model()` function. This function combines the high-dimensional data (`highd_data`), the NLDR data (`nldr_data`), and the bin-averaged high-dimensional model representation (`model_highd`) aligned to the 2-*D* bin layout (`model_2d`). This combined dataset includes both the original observations and the bin-level model averages, labeled with a `type` variable for distinguishing between them. Also, the `show_link_plots()` function takes `edge_data`, which defines connections between neighboring bins.

```
df_exe <- comb_all_data_model(
  highd_data = scurve,
  nldr_data = scurve_umap,
  model_highd = df_bin,
  model_2d = df_bin_centroids
  )
```

```
nldrdt_link <- show_link_plots(
  point_data = df_exe,
  edge_data = trimesh,
  point_colour = clr_choice
  )

nldrdt_link <- crosstalk::bscols(
    htmltools::div(
        style = "display: grid; grid-template-columns: 1fr;
    gap: 0px; align-items: start; justify-items: center;
    margin: 0; padding: 0;
    height: 420px; width: 100%; overflow: hidden;",
        nldrdt_link
    ),
    device = "xs"
)

class(nldrdt_link) <- c(class(nldrdt_link), "htmlwidget")

nldrdt_link
```

The function `show_error_link_plots()` generates three side-by-side, interactively linked plots: an error distribution, a 2-*D* NLDR layout, and a tour (via `langevitour`) of the model overlaid by the data (Figure 3.8). The function takes the output from `comb_all_data_model_error()` (`point_data`) and `edge_data`, which defines connections between neighboring bins.

The `point_data` can be generated using the `comb_all_data_model_error()` function. The function requires several arguments: points data, which contain high-dimensional data (`highd_data`),

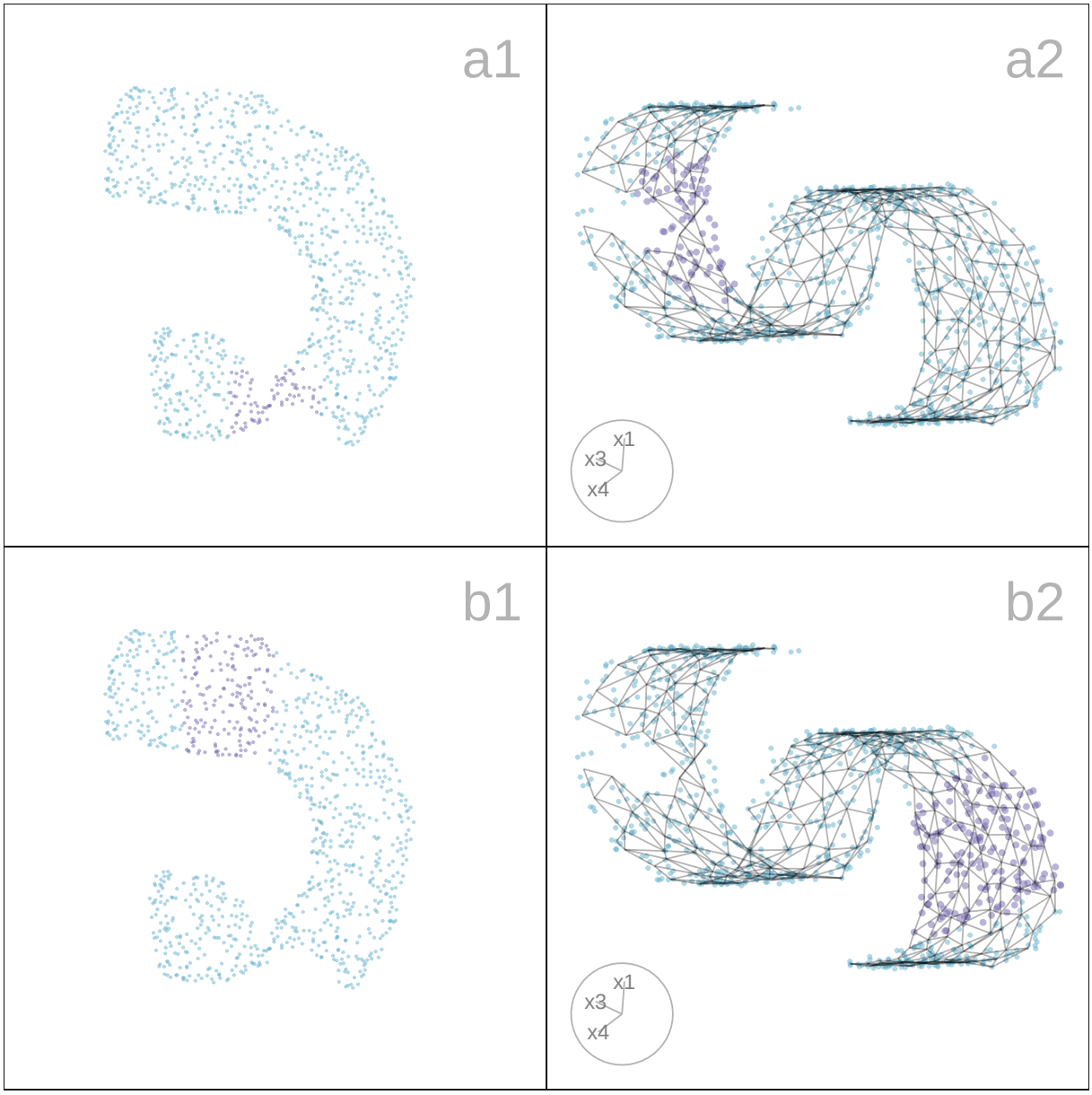

Figure 3.7: *Exploring the correspondence between UMAP layout and* `scurve` *structure in 7-D. Two sets of plots are interactively linked: UMAP layout (a1, b1) and projection of 7-D model and data (a2, b2). The purple points indicate the selected subsets, which differ between rows. In (a1), the lower bridge of the* `scurve` *is highlighted, which corresponds in (a2) to points spanning across both arms of the high-dimensional structure. In (b1), a different region near the upper arm of the* `scurve` *is selected, and in (b2) these points map onto one side of the curved manifold in 7-D projection. While the UMAP layout suggests distinct local clusters, the linked tour views reveal how these selections trace continuous structures in the 7-D space, highlighting distortions introduced by UMAP.*

NLDR data (`nldr_data`), high-dimensional model data (`model_highd`), 2-*D* model data (`model_2d`), and model error (`error_data`). This combined dataset includes both the original observations and the bin-level model averages, labeled with a `type` variable for distinguishing between them.

```
df_exe <- comb_all_data_model_error(
  highd_data = scurve,
  nldr_data = scurve_umap,
  model_highd = df_bin,
  model_2d = df_bin_centroids,
  error_data = model_error
  )
```

```
errornldrdt_link <- show_error_link_plots(
  point_data = df_exe,
  edge_data = trimesh,
  point_colour = clr_choice
)

errornldrdt_link <- crosstalk::bscols(
    htmltools::div(
        style = "display: grid; grid-template-columns: 1fr;
    gap: 0px; align-items: start; justify-items: center;
    margin: 0; padding: 0;
    height: 360px; width: 100%; overflow: hidden;",
        errornldrdt_link
    ),
    device = "xs"
)

class(errornldrdt_link) <- c(class(errornldrdt_link), "htmlwidget")

errornldrdt_link
```

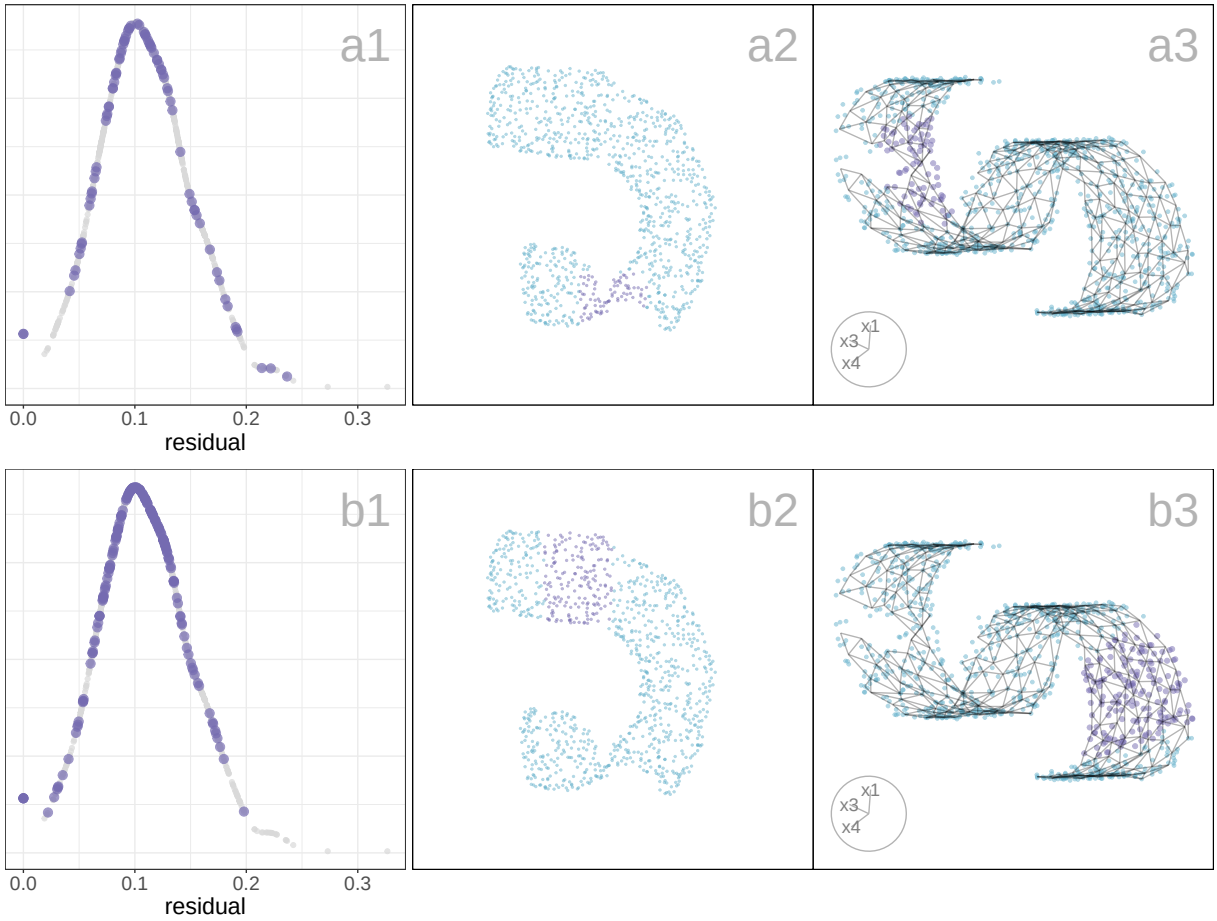

Figure 3.8: *Exploring residuals in relation to UMAP layouts using a 7-D* `scurve` *model. Three views are linked: distribution of residuals (a1, b1), UMAP layout (a2, b2), and projection of the 7-D model with data (a3, b3). The purple points highlight selected subsets of the data, which differ across rows. In the top row (a1–a3), points with higher residuals (a1) are selected, corresponding to the sparse bridging region in the UMAP layout (a2) and the less dense end of the* `scurve` *in the high-dimensional projection (a3). In the bottom row (b1–b3), points with lower residuals (b1) are highlighted, which map to one side of the dense region in the NLDR layout (b2) and to a thicker band of the* `scurve` *in the projection (b3). This comparison illustrates how residuals can help diagnose distortions in UMAP, with high-residual points often concentrated in sparse or stretched regions of the structure.*

In the implementation examples, points are shown without cluster-based coloring unless explicitly stated. When points are colored by cluster, linked plot functionality is currently only partially supported: selections made in the `langevitour` controls are not reflected in the corresponding highlights in the interactive 2-*D* layout. As a result, cluster-specific exploration must be carried out separately in the tour view and the interactive 2-*D* layout. In addition to `langevitour`, linked plots can also be constructed using tour views generated with the `detourr` package. (see https://jayanilakshika.github.io/quollr/articles/quollr8linkeddetourr.html for details.)

3.4 Application

Single-cell RNA sequencing (scRNA-seq) is a popular and powerful technology that allows you to profile the whole transcriptome of a large number of individual cells (Andrews et al. 2021).

Clustering of single-cell data is used to identify groups of cells with similar expression profiles. NLDR is often used to summarize the discovered clusters and help to understand the results. The purpose of this example is to *illustrate how to use our method to help decide on an appropriate NLDR layout that accurately represents the data*.

Limb muscle cells of mice in The Tabula Muris Consortium (2018) are examined. There are 1067 single cells, with 14997 gene expressions. Following their pre-processing, different NLDR methods were performed using ten principal components. Figure 3.9 (a) is the reproduction of the published plot. This was generated using tSNE with perplexity = 30, the default hyper-parameters. The question is whether this accurately represents the cluster structure in the data. Note that the cluster variable is not used to produce the 2-*D* layout.

We illustrate how to use `quollr` to assess whether this is a reasonable layout. Figure 3.9 shows five alternative layouts, and the HBE plot summarizing the resulting model fits. Layout b is produce by UMAP (neighbors = 5, minimum distance = 0.1); layout c was produced by PHATE (knn = 5); layout d was produced by TriMAP (number of inliers = 12, outliers = 4, random = 3); layout e was produced by by PaCMAP (neighbors = 10, init = "random", MN-ratio = 0.5, FP-ratio = 2); layout f was produced by tSNE (perplexity = 15).

The HBE plot indicates that the two tSNE layouts outperform all the other methods across a range of binwidths, but that the result with perplexity of 15 outperforms the other. There are small visual differences between the two layouts. Both support that there are 5 — 6 clusters. Layout a has slightly more space between clusters. Layout d has three small clusters at the top, whereas layout f has only two, and another smaller one at the bottom.

```
design <- gen_design(n_right = 6, ncol_right = 2)

plot_hbe_layouts(plots = list(error_plot_limb, nldr1,
                              nldr2, nldr3, nldr4,
                              nldr5, nldr6), design = design)
```

Figure 3.10 shows how to examine the resulting models of the layouts overlaid on the data in high dimensions. Point color represents the cluster reported in the published paper. In each case, the best

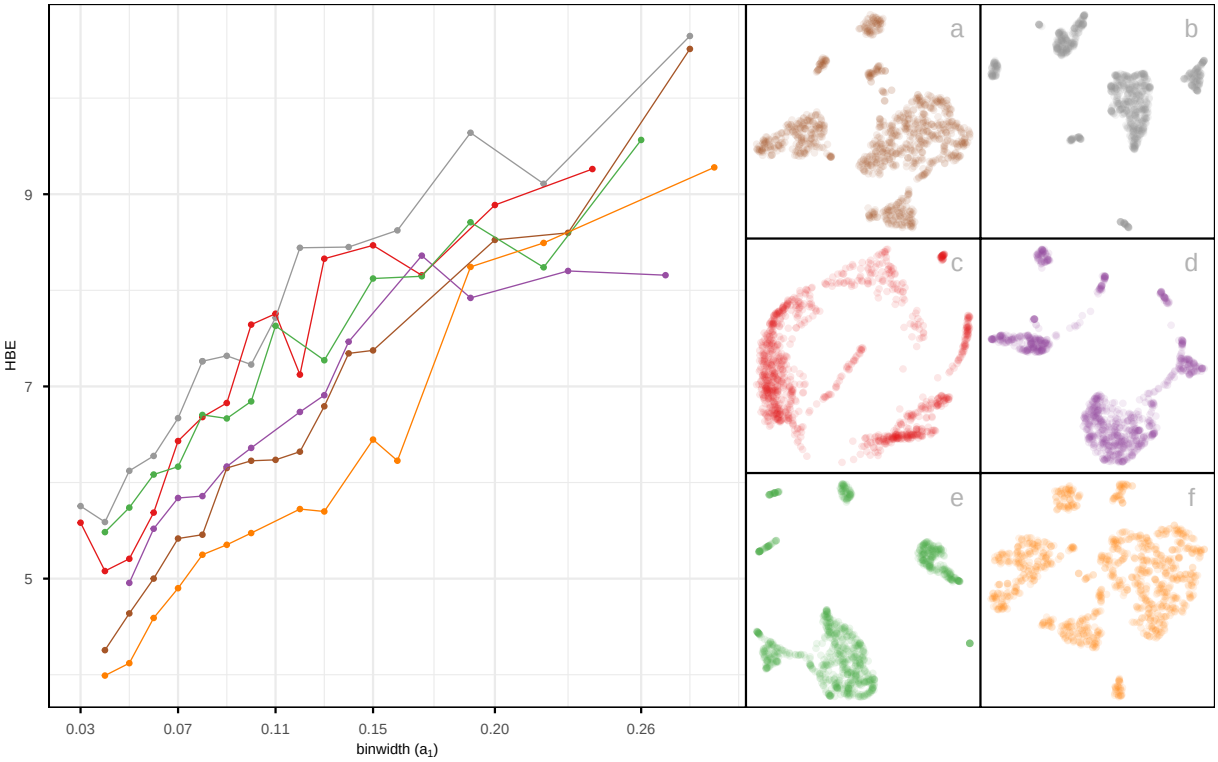

Figure 3.9: *Assessing which of the 6 NLDR layouts on the limb muscle data is the better representation using HBE for varying binwidth (a_1). Color used for the lines and points in the left plot and in the scatterplots represents NLDR layout (a-f). Layout d performs well at large binwidth (where the binwidth is not enough to capture the data structure) and poorly as the bin width decreases. Layout f is the best choice.*

model is produced using binwidth = 0.06. A binwidth of 0.06 is used because it is small enough to show local structure and differences between clusters, but large enough to avoid breaking the layout into too many small pieces that would make the plots hard to interpret. The plots in these figures are best understood using small steps.

1. Examine the model in the data space for each, by looking at the tour views. In each case, the clustering doesn't quite match the separations in the data, and both models help see this. For example, the orange cluster (1) should probably be split into more than one cluster because both models show large stretched lines connecting a small group far from the remaining orange points.

2. Because 6 clusters are hard to examine together, use the menu to select just one cluster to view at a time. Selecting just cluster 1 might help you see the explanation above, that a small group is quite separate from the main group. This suggests both the layouts and the clustering that groups these together might be wrong. Now change to focus on cluster 5 (yellow). This group is a fairly large, sparse cluster, but it is separated from the other points. Both layouts are right in separating these points from the others, but the fit for layout f is slightly better.

3. To examine where the layouts differ, examine clusters 4 (green) and 6 (blue) by selecting just these two. Layout a has them close, but layout f has them far apart. (It might also help to include cluster 5 here because layout a has this group close to the cluster 4 also.) In the tour view, we can see that the three clusters are separated clusters in different directions away from most of the other points. They are both correct in this. It may have been better to place them all in different corners of the layout, but they have preserved the most important aspect that they are separated clusters. That they are all close together in layout could be incorrectly interpreted as close in high-dimensions, though.

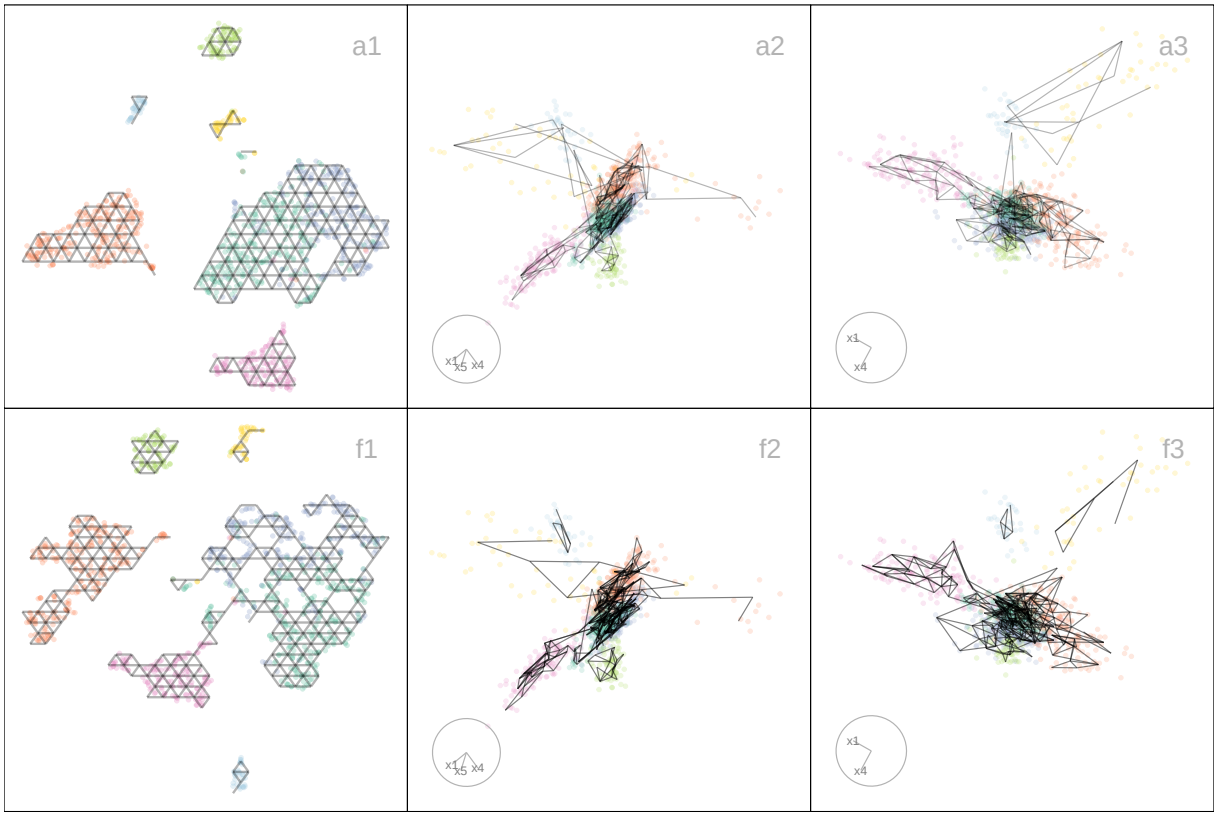

Figure 3.10: *Representative views of two selected NLDR layouts for the Limb muscle dataset ($n = 1067$), shown row-wise. The top row (a1–a3) corresponds to the published 2-D layout (Figure 3.9 a), and the bottom row (f1–f3) corresponds to the 2-D layout selected (Figure 3.9 f) selected using the HBE plot. In each row, the left panel (a1, f1) shows the NLDR embedding with points colored by muscle group and overlaid with triangulated hexagon centroids. The middle (a2, f2) and right (a3, f3) panels show two different 2-D projections of the fitted model and data in the original 10-D space, with the same triangular mesh displayed. Together, these panels summarize how the low-dimensional layouts relate to the underlying high-dimensional structure across different viewing directions.*

3.5 Discussion

The `quollr` package introduces a new framework for interpreting NLDR outputs by fitting a geometric wireframe model in 2-*D* and lifting it into high-dimensional space. This lifted model provides a direct way to assess how well a 2-*D* layout, produced by methods such as tSNE, UMAP, PHATE, TriMAP, or PaCMAP, preserves the structure of the original high-dimensional data. The approach offers both numerical and visual diagnostics to support the selection of NLDR methods and tuning hyper-parameters that produce the most accurate 2-*D* representations.

In contrast to the common practice of visually inspecting scatterplots for clusters or patterns, `quollr` provides a quantitative route for evaluation. It enables the computation of HBE and residuals between the original high-dimensional data and the lifted model, offering interpretable diagnostics. These diagnostics are complemented by interactive linked plots and high-dimensional dynamic visualizations using the `langevitour` package, allowing users to inspect where the model fits well and where it does not.

To support efficient computation, particularly for large-scale datasets, several core functions in `quollr` are implemented in C++ using `Rcpp` and `RcppArmadillo`. These include functions for computing Euclidean distances in high-dimensional and 2-*D* space, identifying nearest centroids, calculating residual errors, and generating polygonal coordinates of hexagons. For instance, `compute_highd_dist()` accelerates nearest neighbor lookup in high-dimensional space, `compute_errors()` calculates HBE and total absolute error efficiently, and `calc_2d_dist_cpp()` speeds up distance calculations in 2-*D*. Additionally, `gen_hex_coord_cpp()` constructs the coordinates for hexagonal bins based on their centroids with minimal overhead. These optimizations result in substantial performance gains compared to native R implementations, making the package responsive even when used in interactive contexts or on large datasets such as single-cell transcriptomic profiles.

The modular structure of the package is designed to support both flexibility and reproducibility. Users can access individual functions to control each step of the pipeline, such as scaling, binning, and triangulation, or use the main function `fit_highd_model()` for end-to-end model construction. The diagnostics can be used not only to compare NLDR methods but also to tune binning parameters, assess layout stability, and detect local distortions in the embedding.

There are several avenues for future development. While hexagonal binning provides a regular structure conducive to modeling, alternative spatial discretizations (e.g., adaptive binning or density-aware tessellations) could be explored to better capture varying data densities. Expanding support for additional distance metrics in the lifting and prediction steps may improve performance across different domains. Additionally, statistical inference tools could be introduced to assess the stability

and robustness of the fitted model, which would enhance interpretability and confidence in the outcomes. A potential extension of the current implementation would be to synchronize cluster selections between the tour view and the linked 2-*D* layout, enabling more direct cluster-specific comparisons across views.

3.6 Acknowledgements

The source code for reproducing this chapter can be found at: https://github.com/JayaniLakshika/paper-quollr. This article is created using `knitr` (Xie 2015) and `rmarkdown` (Xie et al. 2018) in R with the `rjtools::rjournal_article` template. These `R` packages were used for this work: `cli` (Csárdi 2025), `dplyr` (Wickham 2023), `ggplot2` (Wickham 2016), `interp` (>= 1.1-6) (Gebhardt et al. 2024), `langevitour` (Harrison 2023), `detourr` (Hart and Wang 2025), `proxy`(Meyer and Buchta 2022), `stats` (R Core Team 2025), `tibble` (Müller and Wickham 2023), `tidyselect` (Henry and Wickham 2024), `crosstalk` (Cheng and Sievert 2025), `plotly` (Sievert 2020), `htmltools` (Cheng et al. 2024), `kableExtra` (Zhu 2024), `patchwork` (Pedersen 2024), and `readr` (Wickham et al. 2024).

Chapter 4

cardinalR: Generating Interesting High-Dimensional Data Structures

Simulated high dimensional data is useful for testing, validating, and improving algorithms used in dimension reduction, supervised, and unsupervised learning. High-dimensional data is characterized by multiple variables that are dependent or associated in some way, such as linear, nonlinear, clustering, or anomalies. Here, we provide new methods for generating a variety of high-dimensional structures using mathematical functions and statistical distributions organized into the R package `cardinalR`. Several example data sets are also provided. These will be useful for researchers to better understand how different analytical methods work and can be improved, with a special focus on nonlinear dimension reduction methods. This package enriches the existing toolset of benchmark datasets for evaluating algorithms.

4.1 Introduction

Generating synthetic datasets with clearly defined geometric properties is useful for evaluating and benchmarking algorithms in various fields, such as machine learning, data mining, and computational biology. Researchers often need to generate data with specific dimensions, noise characteristics, and complex underlying structures to test the performance and robustness of their methods. There are numerous packages available in R for generating synthetic data, each designed with unique characteristics and focus areas. The `geozoo` package (Schloerke 2016) provides functions to generate standard high-dimensional data like cubes, spheres, and simplexes, along with some prepared datasets. The `snedata` package (Melville 2025) provides functions for generating common examples used in dimension reduction publications and to download benchmark data sets. The `splatter`

package (Zappia et al. 2017) is designed to simulate complex biological data, capturing field-specific nuances such as batch effects and differential expression. The `mlbench` package (Leisch and Dimitriadou 2024) provides access to benchmark datasets commonly associated with established classification or regression challenges. The `surreal` package (Balamuta 2024) implements the "Residual (Sur)Realism" algorithm (Stefanski 2007) to generate datasets that embed hidden images or text into residual plots, providing engaging visual demonstrations for teaching model diagnostics.

The current work implemented in the `cardinalR` R package builds on these approaches. It provides functions to generate a more extensive set of high-dimensional data structures, allowing users to: (i) construct high-dimensional datasets based on geometric shapes, including the option to enhance dimensionality by adding controlled noise dimensions; (ii) introduce adjustable levels of background noise to these structures; and (iii) combine the shapes to produce multiple clusters. The user can control characteristics such as the number of dimensions, shape, and sample size. It is designed to resource researchers with synthetic datasets to evaluate the performance and interpret the fit of NLDR methods, clustering algorithms, and visualization techniques. These datasets can also serve as benchmark examples for exploring how different choices of algorithm parameters affect the identification or representation of cluster and manifold structures in high-dimensional spaces.

The motivation for developing this package originated from our own work in studying nonlinear dimension reduction (NLDR) algorithms. We wanted to conduct a visualization experiment to understand the perception and misperception of a variety of NLDR methods. This required simulated datasets with carefully controlled geometric and clustering properties. While some existing packages provided useful starting points, none fully supported the creation of flexible, high-dimensional data with the specific structural variations needed for our experiment. Developing these generators for research purposes underlies `cardinalR`, which is now a general-purpose package that should be useful for research and teaching.

The example data structures are best viewed using a tour (Asimov 1985). These show the data as a sequence of low-dimensional projections (typically 2-D), providing a good sense of the shape in high dimensions. The interactive tour plots included in this chapter are produced using the software `langevitour` (Harrison 2023).

The next section provides an overview of the usage of the `cardinalR` package, illustrating how its modular components can be combined to generate complex high-dimensional datasets. This is followed by a section describing the implementation of the package, including its design principles and key functions. The Application section then demonstrates how the simulated clustering structures can be used to evaluate and compare dimension reduction and clustering methods. Finally, we give a

brief conclusion of the chapter and discuss potential opportunities for the use of our data collection.

4.2 Usage

The `cardinalR` package is built on a modular framework where individual geometric generators (e.g., Gaussian, cone, sphere) create well-defined shapes (A full list of available shape generators are available at https://jayanilakshika.github.io/cardinalR/reference/index.html.), which can then be combined into a single dataset including scaling, rotation, and translation. The package is available on CRAN, and the source is available on GitHub at https://github.com/JayaniLakshika/cardinalR.

The main function, `gen_multicluster()`, is an all-in-one function that includes generating individual shapes, handling scaling and rotating of these shapes, and combining the result into a single unified dataset. This function and associated workflow allow flexible construction of complex, high-dimensional structures for evaluating clustering and dimension reduction methods. Figure 4.1 illustrates the workflow of `gen_multicluster()`.

Users can control the number of clusters (`k`) and the number of points in each cluster (`n`). Each cluster can take on a different geometric shape (e.g., Gaussian, cone, uniform cube) by specifying the corresponding generator function (`shape`), can be scaled to adjust its spread, rotated using custom rotation matrices (`rotation`), and positioned at defined centroids (`loc`). The function ensures flexibility in cluster location and orientation, allowing users to simulate complex high-dimensional structures.

The following is an example of a three-shape multiclustered dataset. The first shape is Gaussian, the second conical, and the third a cube.

```
clust_data <- gen_multicluster(
  n = c(200, 300, 500),
  k = 3,
  loc = matrix(c(
    0, 0, 0, 0,
    5, 9, 0, 0,
    3, 4, 10, 7
  ), nrow = 3, byrow = TRUE),
  scale = c(3, 1, 2),
  shape = c("gaussian", "cone", "unifcube"),
```

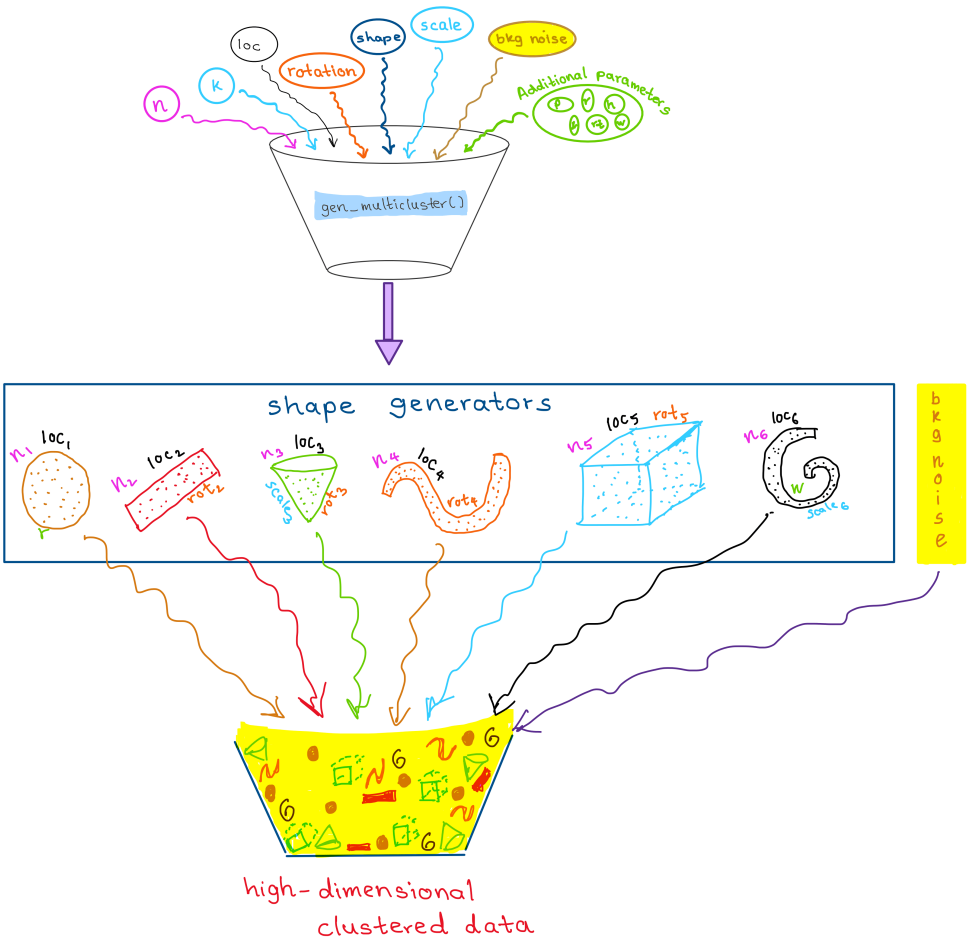

Figure 4.1: *Workflow for generating high-dimensional clustered data. The user specifies input parameters such as the number of points (n), number of clusters (k), cluster locations, shapes, scaling, rotations, and optional background noise. Each cluster shape is generated by a shape generator, optionally rotated or scaled, and combined into a single dataset. Additional background noise can be added, and each observation is labeled by shape.*

```
  is_bkg = FALSE
)
```

Here, the shapes have 200, 300, and 500 points respectively (`n`), are positioned in 4-*D* space according to a location matrix, `loc`, and stretched according to the `scale`. The details of the individual shape generators and the noise elements are contained in the following sections.

4.3 Implementation

The main function of the package is `gen_multicluster()`, which generates datasets consisting of multiple clusters with user-specified characteristics. To maintain consistency across generators, the function identifies the arguments required by each chosen generator function and supplies only

Table 4.1: *The main arguments for* `gen_multicluster()`.

Argument	Type	Explanation
`n`	integer (vector)	Number of points in each cluster.
`k`	integer	Number of clusters.
`loc`	numeric (matrix)	Locations/centroids of clusters.
`scale`	numeric (vector)	Scaling factors of clusters.
`shape`	character (vector)	Shapes of clusters.
`rotation`	numeric (list)	Rotation matrices, one per cluster.
`is_bkg`	boolean	Background noise should exist or not.

those arguments that are valid for that specific generator. This design enables the combination of cluster types with differing parameter requirements within the same dataset. When clusters are generated with fewer dimensions than others, the function augments the lower-dimensional clusters with additional Gaussian noise variables so that all clusters are represented in the same dimensional space. These noise dimensions are drawn independently from normal distributions $X \sim \mathcal{N}(\mu, \sigma^2)$, where the mean ($\mu$) is set to the average of the cluster coordinates and the standard deviation (σ) defaults to 0.2.

An optional argument, `is_bkg`, adds background noise drawn from a multivariate normal distribution centered on the dataset's overall mean with standard deviations matching the observed spread. Extra arguments `(...)` can be passed to cluster generators, allowing further control over per-cluster characteristics like the radius of the sphere. The main arguments of the `gen_multicluster()` function are shown in Table 4.1.

4.3.1 Shape generators

The shape generators form the foundation of the package, providing functions to create synthetic datasets from simple, well-defined geometric forms such as cones, pyramids, spheres, grids, and branching structures. Each generator includes the parameter `n`, which specifies the number of points to generate. Some functions, such as `gen_unifcube()`, also take the dimension `p`, while others include arguments specific to the geometry (e.g., radius for spheres (`r`), width for bands (`w`)). If higher-dimensional data are required, additional noise dimensions can be appended after data generation using any noise generator function. This flexibility allows users to construct both low- and high-dimensional datasets from the same underlying structures. Table 4.2 outlines these functions. The main arguments of the functions described in Table 4.3.

Table 4.2: *Overview of shape-generation functions, including their required parameters and a brief description of each geometric structure produced. The generators cover branching patterns, spheres, spirals, pyramids, Gaussian clouds, and other nonlinear shapes.*

Function	Arguments	Explanation
`gen_expbranches`	`n, k`	Exponential shaped branches.
`gen_linearbranches`	`n, k`	Linear shaped branches.
`gen_curvybranches`	`n, k`	Curvy shaped branches.
`gen_orglinearbranches`	`n, p, k`	Linear-shaped branches originated from one point.
`gen_orgcurvybranches`	`n, p, k`	Curvy shaped branches originated from one point.
`gen_cone`	`n, p, h, ratio`	Cone-shaped structure.
`gen_gridcube`	`n, p`	Cube with specified grid points along each axis.
`gen_unifcube`	`n, p`	Cube with uniform points.
`gen_gaussian`	`n, p, s`	Multivariate Gaussian cloud.
`gen_longlinear`	`n, p`	Long linear structure.
`gen_mobius`	`n`	Möbius strip in 3-*D*.
`gen_quadratic`	`n`	Quadratic pattern in 2-*D*.
`gen_cubic`	`n`	Cubic pattern in 2-*D*.
`gen_pyrrect`	`n, p, l_vec, rt`	Rectangular-base Pyramid, with a sharp or blunted apex.
`gen_pyrtri`	`n, p, h, l, rt`	Triangular-base Pyramid, with a sharp or blunted apex.
`gen_pyrstar`	`n, p, h, rb`	Star-shaped base Pyramid, with a sharp or blunted apex.
`gen_pyrfrac`	`n, p`	Pyramid with triangular pyramid-shaped holes.
`gen_scurve`	`n`	S-curve in 3-*D*.
`gen_circle`	`n, p`	Circle.
`gen_curvycycle`	`n, p`	Curvy cell cycle.
`gen_unifsphere`	`n, r`	Uniform ball.
`gen_hollowsphere`	`n, p`	Hollow sphere.
`gen_gridedsphere`	`n`	Grided sphere.
`gen_clusteredspheres`	`n, k, r, loc`	Multiple small spheres within a big sphere.
`gen_hemisphere`	`n, p`	Hemisphere.
`gen_swissroll`	`n, w`	Swissroll structure.
`gen_trefoil4d`	`n, steps`	Trefoil in 4-*D*.
`gen_trefoil3d`	`n, steps`	Trefoil in 3-*D*.
`gen_crescent`	`n`	Crescent pattern.
`gen_curvycylinder`	`n, h`	Curvy cylinder.
`gen_sphericalspiral`	`n, spins`	Spherical spiral.
`gen_helicalspiral`	`n`	Helical spiral.
`gen_conicspiral`	`n, spins`	Conic spiral.
`gen_nonlinear`	`n, hc, non_fac`	Nonlinear hyperbola.

Table 4.3: *Argument definitions for the shape generators. The table lists each argument, its data type, and a description of its role in controlling geometric structure, dimensionality, scaling, curvature, spacing, and other features of the simulated high-dimensional datasets.*

Argument	Type (positive)	Explanation
`n`	integer	Number of points.
`k`	integer	Number of clusters.
`p`	integer	Number of dimensions.
`h`	real value	Height.
`ratio`	real value	Radius tip to radius base ratio.
`s`	real value	Variance-covariance matrix.
`r`	real value	Radius.
`n_vec`	integers	Sample sizes of the big and small spheres
`k_small`	integer	Number of small spheres.
`r_vec`	real values	Radius of the big and small spheres.
`spe`	real value	How far apart are the small spheres placed?
`w`	real value	Vertical variation
`steps`	integer	Number of steps for the theta parameter.
`spins`	integer	Number of loops of the spiral.
`hc`	real value	Steepness and vertical scaling of the hyperbola.
`non_fac`	real value	Strength of this sinusoidal effect.
`l`	real value	Base length of the pyramid.
`l_vec`	real values	Base lengths along the and y of the pyramid.
`rt`	real value	Tip radius of the pyramid.
`rb`	real value	Base radius of the pyramid.

Branching

A branching structure (Figure 4.2) captures trajectories that diverge or bifurcate from a common origin, similar to processes such as cell differentiation in biology (Trapnell et al. 2014). We introduce a set of data generation functions specifically designed to simulate high-dimensional branching structures with various geometries, total number of points (`n`) generated across all branches, with points allocated approximately evenly among branches, and number of branches (`k`). Although these functions can generate multiple branches, they do not produce a formal *multicluster* dataset: the branches form a single connected structure, with multiple visually distinct arms rather than independent clusters.

The simplest structures are approximately linear branches in 2-D, generated by the `gen_linearbranches(n, k)` function. These consist of k short line segments in the first two dimensions, with added jitter to simulate variability. Mathematically, each branch i is defined as

$$X_1 \sim U(a_i, b_i), \quad X_2 = s_i(X_1 - x_{\text{start},i}) + y_{\text{start},i} + \epsilon, \quad \epsilon \sim U(0, \delta),$$

where $(x_{\text{start},i}, y_{\text{start},i})$ is the starting point of branch i, δ controls local jitter, and s_i is the slope, initialized as

$$s_i = \begin{cases} 0.5 & i = 1, \\ -0.5 & i = 2, \\ \text{randomly sampled from } [s_{\min}, s_{\max}] & i = 3, \ldots, k. \end{cases}$$

The jitter term is sampled from a one-sided uniform distribution to introduce directional variability without altering branch orientation.

Branches 1 and 2 are initialized with fixed slopes and starting points, while later branches are iteratively added at locations chosen to avoid overlap with existing branches, producing a set of connected linear paths.

To introduce curvature, the `gen_curvybranches(n, k)` function generates k curvilinear branches in 2-D. Each branch follows a quadratic trajectory of the form

$$X_1 \sim U(a_i, b_i), \quad X_2 = 0.1X_1 + s_i X_1^2 + \epsilon, \quad \epsilon \sim U(-\delta, \delta),$$

where (a_i, b_i) defines the domain of the branch, s_i controls curvature, and δ introduces local jitter. For the first two branches, the parameters are fixed to establish reference shapes: $(a_1, b_1, s_1) = (0, 1, 1)$, $(a_2, b_2, s_2) = (-1, 0, -2)$. Additional branches are attached iteratively to existing structures. Each new branch i starts at a selected point $(x_{\text{start},i}, y_{\text{start},i})$ from the current structure and extends according to

$$X_1 \sim U(x_{\text{start},i}, x_{\text{start},i} + 1), \quad X_2 = 0.1X_1 - s_i(X_1^2 - x_{\text{start},i}) + y_{\text{start},i},$$

where s_i is a scale factor controlling the curvature of branch i. For the first few initial branches, s_i can be fixed (e.g., $s_1 = 1, s_2 = 2$) to establish reference shapes, while for subsequent branches it is sampled from a predefined set, such as $s_i \in \{-2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5\}$, to control curvature magnitude and direction.

The `gen_expbranches(n, k)` function creates k exponential branches in 2-D, radiating from a central region. Each branch i is defined as

$$X_1 \sim U(-2, 2), \quad X_2 = \exp(\sigma_i s_i X_1) + \epsilon, \quad \epsilon \sim U(0, \delta), \quad s_i \sim U(0.5, 2),$$

where $\sigma_i = (-1)^{i+1}$ alternates the sign of the exponent to produce mirror-symmetric branches. The parameter s_i controls the steepness of branch i, and δ introduces small local jitter.

High-dimensional generalizations are provided by `gen_orglinearbranches(n, p, k)` (Figure 4.2) and `gen_orgcurvybranches(n, p, k)`. For branch i, the active coordinate pair (i_1, i_2) indexes the selected 2-*D* subspace. When `allow_share = TRUE`, multiple branches may share the same subspace; otherwise, subspaces are sampled without replacement until all possible $\binom{p}{2}$ combinations are exhausted, after which additional branches may repeat subspaces.

In both cases, branch i is generated according to

$$X_{i_1} \sim U(a_i, b_i), \quad X_{i_2} = f_i(X_{i_1}) + \epsilon, \quad \epsilon \sim N(0, \sigma^2),$$

where a_i and b_i define the domain of the branch and ϵ introduces smooth variability in the p-*D* space. The function $f_i(\cdot)$ determines the branch geometry:

$$f_i(x) = \begin{cases} s_i x, & \text{linear branches,} \\ -s_i x^2, & \text{curvilinear branches.} \end{cases}$$

The scale factor s_i controls slope (linear branches) or curvature (curvilinear branches) and is assigned as follows: for the first $\binom{p}{2}$ branches, $s_i = 1$; for additional branches when $k > \binom{p}{2}$, s_i is randomly drawn from the set $\{1, 1.5, 2, \ldots, 8\}$.

Across all branching generators, the scale parameter s_i controls the strength of deviation from linearity, determining slope, curvature, or growth rate depending on the branch geometry.

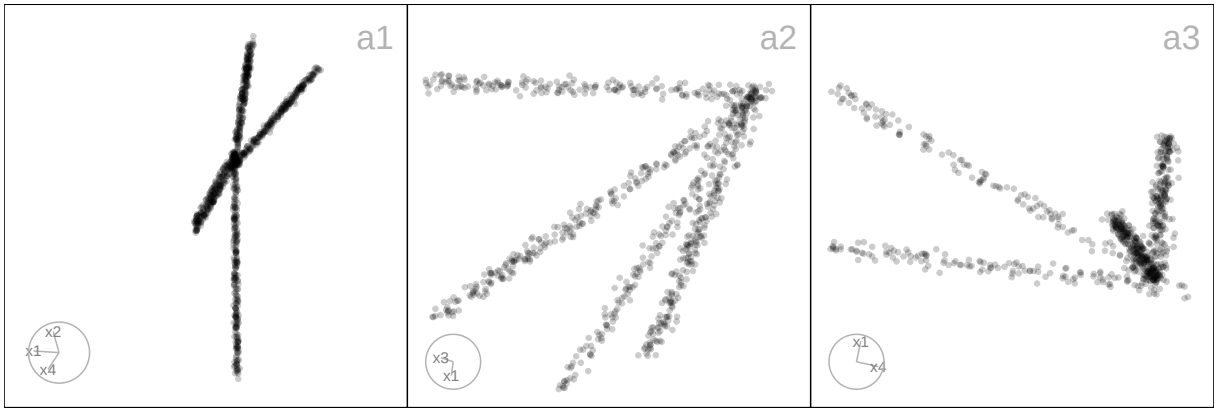

Figure 4.2: *Three 2-D projections from the 4-D `orgcurvybranches` data. Each shows a different projection, illustrating how the linear branches appear from multiple viewing angles. These views highlight the dataset's underlying branching structure and demonstrate how projections reveal patterns that are otherwise hidden in higher dimensions.*

Cone

To simulate a cone-shaped structure in arbitrary dimensions (Figure 4.3), we define a function `gen_cone(n, p, h, ratio)`, which creates a high-dimensional cone with options for a sharp or blunted apex, allowing for a dense concentration of points near the tip.

This function generates n points in p-D, where the last dimension, X_p, represents the height along the cone's axis, and the first $p-1$ dimensions define a shrinking hyperspherical cross-section toward the tip. Heights are sampled from a truncated exponential distribution, $X_p \sim \text{Exp}(\lambda = 2/h)$, truncated to the interval $[0, h]$, producing a higher density of points near the tip. At each height X_p, the radius of the cross-section increases linearly from base to tip according to $r = r_{\min} + (r_{\max} - r_{\min})X_p/h$, where $r_{\min} = \text{ratio} \in [0, 1]$ and $r_{\max} = 1$.

For each point, a direction in the first $p-1$ dimensions is sampled uniformly on a $(p-1)$-dimensional hypersphere using generalized spherical coordinates. The radial coordinates are scaled by the height-dependent radius r, producing the conical taper. In three dimensions ($p = 3$), this results in a classical 3-D cone, while for $p > 3$, additional dimensions provide a smooth embedding into higher-dimensional space, preserving the conical structure.

Cone-shaped structures appear in particle dispersions, light beams, and tapering processes, where spread decreases along one axis. They are also used to benchmark clustering and dimension reduction methods (Hadsell et al. 2006).

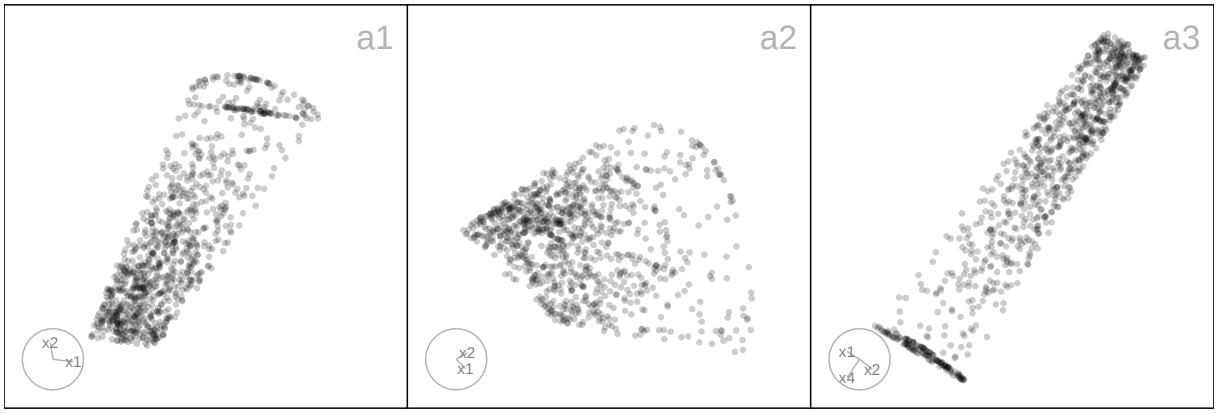

Figure 4.3: *Three 2-D projections from the 4-D cone data. Points are concentrated near the tip along the height dimension, while the radius of the hyperspherical cross-section decreases linearly toward the apex. These projections show how the conical geometry is preserved.*

Cube

A cube structure represents uniformly or systematically distributed points within a high-dimensional hypercube, providing a useful framework for assessing how well algorithms preserve uniformity and

boundary properties in high dimensions. We provide a set of functions to generate high-dimensional cube structures with flexible configurations, including regular grids and uniform random points.

The function `gen_gridcube(n, p)` is a wrapper around `geozoo::cube.solid.grid()`. It generates a regular lattice of points in p-D, producing a uniform hypercube grid. The parameter `n` controls the approximate number of points by determining the grid resolution along each axis.

By contrast, `gen_unifcube(n, p)` wraps `geozoo::cube.solid.random()`, producing uniformly distributed points within a p-D cube. To avoid including the cube's vertices, these points are removed after generation. This results in a hypercube filled with random samples rather than structured lattice points.

Such cube-based structures are commonly used as benchmarks in Monte Carlo sampling, computational geometry, and density estimation, where assessing how algorithms behave under uniform or grid-like distributions is critical (Devroye 1986; Niederreiter 1992).

Gaussian

The `gen_gaussian(n, p, s)` function generates a multivariate Gaussian cloud in p-D, centered at the origin with user-defined covariance structure. For $i = 1, \ldots, n$, each observation is independently drawn from a multivariate normal distribution, $X_i \sim N_p(\mathbf{0}, s)$, where s is a user-defined $p \times p$ positive-definite matrix.

Gaussian clouds are common benchmark structures in statistics and machine learning, used in clustering, classification, and anomaly detection, with applications in image segmentation, speech recognition, and forensic analysis (McLachlan and Peel 2000).

Linear

The `gen_longlinear(n, p)` function generates a high-dimensional dataset representing a single noisy linear trajectory. Let $t_i = i - 1, \quad i = 1, \ldots, n$, denote a common latent index shared across all dimensions. For each dimension $j = 1, \ldots, p$), independent scale and shift parameters are sampled as $a_j \sim U(-10, 10), \qquad b_j \sim U(-300, 300)$. Gaussian noise $\varepsilon_{ij} \sim N(0, (0.03n)^2)$ is added independently across observations and dimensions. The observed variables are then defined as $X_{ij} = a_j\big(t_i + b_j + \varepsilon_{ij}\big), \quad i = 1, \ldots, n$. This construction yields a single elongated linear structure embedded in p-D, with each dimension exhibiting a different orientation, scale, and offset.

This structure appears in p-D data when variation is driven by a single factor, such as time-course or sensor measurements, providing a useful test case for trajectory and regression methods (Trapnell et al. 2014).

Möbius

The `gen_mobius()` function is a wrapper around `geozoo::mobius()`, designed to simplify the generation of a Möbius strip in three dimensions for use in high-dimensional diagnostic studies. The function returns a tibble with n sampled points forming the surface of a Möbius strip.

The Möbius strip structure can model twisted or cyclic surfaces in physics and engineering, such as conveyor belts, molecular structures, or optical systems with non-orientable geometries (Optica - The Optical Society 2023).

Polynomial

A polynomial structure generates data points that follow nonlinear curvilinear relationships, such as quadratic or cubic trends, in 2-D space. To extend these patterns into high-dimensional settings, additional noise dimensions can be added. These patterns are useful for evaluating how well algorithms capture smooth, nonlinear trajectories and curvature in the data. We provide functions for generating quadratic and cubic structures, enabling controlled experiments with different degrees of polynomial complexity.

The first is the quadratic curve of n points in two dimensions. This is generated using `gen_quadratic(n, range)`. Let $range = [a, b]$. The independent variable is defined as $X_1 \sim U(a, b)$, and the response is generated as $X_2 = X_1 - X_1^2 + \varepsilon$, where $\varepsilon \sim U(0, 0.5)$. This produces a smooth parabolic arc opening downward, with vertical jitter introduced by the noise term.

The second is the cubic curve of n points in two dimensions. This is generated using `gen_cubic(n, range)`. Let $range = [a, b]$. The independent variable is defined as $X_1 \sim U(a, b)$, and a raw polynomial basis of degree 3 is applied to construct $X_2 = X_1 + X_1^2 - X_1^3 + \varepsilon_2$, where $\varepsilon_2 \sim U(0, 0.5)$. This produces a more complex curvilinear structure than the quadratic case, with both upward and downward turning points.

Pyramid

A pyramid structure (Figure 4.4) represents data arranged around a central apex and base, useful for exploring how algorithms handle pointed or layered geometries in p-D space. The functions provided allow users to generate pyramids with rectangular, triangular, and star-shaped bases, and sharp or blunted apexes. Additionally, it is possible to create a pyramid with a fractal-like internal structure, enabling the study of non-convex and sparse regions.

Let $X_1, \ldots, X_p$ denote the coordinates of the generated points. For the rectangular, triangular, and star-shaped based pyramid generator functions, the final dimension, X_p, encodes the height of each

point and is drawn from an exponential distribution capped at the maximum height h. That is, $X_p = z \sim \min(\text{Exp}(\lambda = 2/h),\ h)$. This distribution creates a natural skew toward smaller height values, resulting in a denser concentration of points near the pyramid's apex. For the star-shaped base pyramid, the final dimension is drawn from a uniform distribution. That is, $X_p = z \sim U(0,h)$.

The remaining dimensions are based on the specific pyramid shape. For the rectangular-based pyramid, `gen_pyrrect(n, p, h, l_vec, rt)` (Figure 4.4 a), the base shape is a rectangle whose size shrinks linearly with height. Let l_x and l_y denote the half-widths of the rectangular base in the X_1 and X_2 directions, specified via $l_{vec} = (l_x, l_y)$, and let r_t denote the half-width at the pyramid tip. At height $z \in [0,h]$, the half-widths of the rectangular cross-section are $r_x(z) = r_t + (l_x - r_t)z/h$, $r_y(z) = r_t + (l_y - r_t)z/h$. The first three coordinates are then defined as $X_1 \sim U(-r_x(z),\ r_x(z)), \quad X_2 \sim U(-r_y(z),\ r_y(z))$, and $X_3 \sim U(-r_x(z),\ r_x(z))$.

For the triangular-based pyramid, `gen_pyrtri(n, p, h, l, rt)` (Figure 4.4 b), let $r(z)$ denote the scaling factor (distance from the origin to triangle vertices) at height z. That is, $r(z) = r_t + (l - r_t)z/h$. A point in the triangle at height z is generated using barycentric coordinates (u,v) to ensure uniform sampling within the triangular cross-section: $u, v \sim U(0,1), \quad \text{if } u+v > 1 : u \leftarrow 1-u,\ v \leftarrow 1-v$. The first three coordinates (triangle plane) are then: $X_1 = r(z)(1-u-v)$, $X_2 = r(z)u$, and $X_3 = r(z)v$.

For the star based pyramid, `gen_pyrstar(n, p, h, rb)` (Figure 4.4 c), let the radius at height z, $r(z)$, be such that the radius scales linearly from zero (tip) to the base radius r_b. That is, $r(z) = r_b(1 - z/h)$. Each point is placed within a regular hexagon in the plane (X_1, X_2), using a randomly chosen hexagon sector angle $\theta \in \{0, \pi/3, 2\pi/3, \pi, 4\pi/3, 5\pi/3\}$ and a uniformly random radial scaling factor: $\theta \sim DiscreteUniform\{0, \pi/3, \ldots, 5\pi/3\}$, $r_{\text{point}} \sim \sqrt{U(0,1)}$. Then, the first two coordinates are: $X_1 = r(z)r_{\text{point}}\cos(\theta)$, and $X_2 = r(z)r_{\text{point}}\sin(\theta)$.

For rectangular and triangular pyramids, the remaining dimensions X_4 to X_{p-1}, and for star-based pyramids X_3 to X_{p-1}, are treated as noise.

Finally, for the Sierpinski-like pyramid, `gen_pyrfrac(n, p)` (Figure 4.4 d), let $X_1, X_2, \ldots, X_p$ denote the coordinates of the generated points. The generation process begins with an initial point $T_0 \in [0,1]^p$ drawn from a uniform distribution: $T_0 \sim U(0,1)^p$. Let $C_1, C_2, \ldots, C_{p+1}$ denote the corner vertices of a p-D simplex. At each iteration $i = 1, \ldots, n$, a new point is computed by taking the midpoint between the previous point T_{i-1} and a randomly selected vertex C_k: $T_i = 1/2(T_{i-1} + C_k), \quad C_k \in \{C_1, \ldots, C_{p+1}\}$. This recursive midpoint rule generates self-similar patterns with systematic voids (holes) between clusters of points. The points remain bounded inside the convex hull of the simplex. The final output is a $n \times p$ matrix where each row represents a point: $X = \{T_1, T_2, \ldots, T_n\}, \quad X \in \mathbb{R}^{n \times p}$.

Pyramid structures mimic tapering or layered geometries seen in architecture, crystals, and fractal-like natural patterns (Mandelbrot 1983).

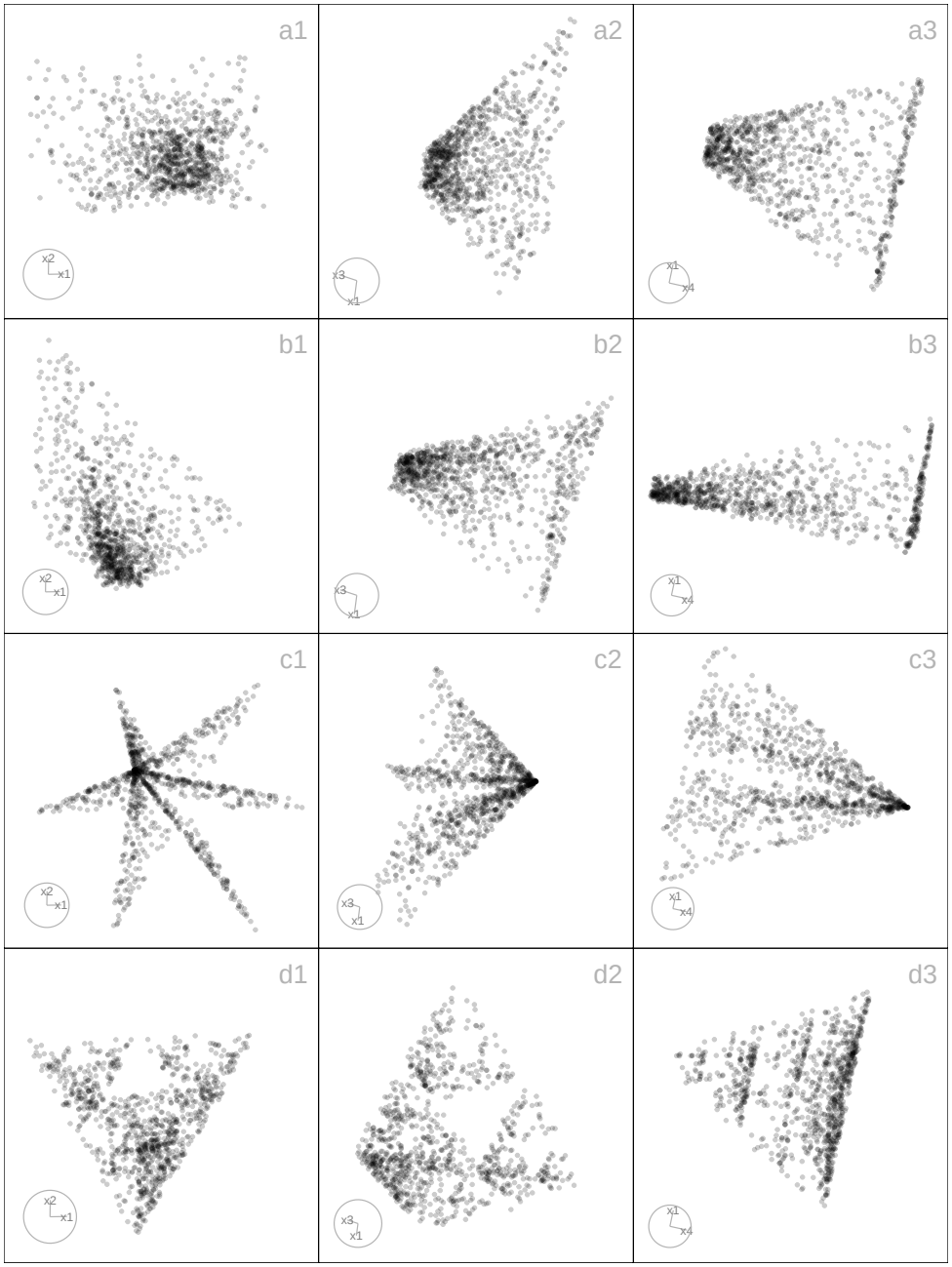

Figure 4.4: *Three 2-D projections from 4-D, for the* `pyrrect` *(a1-a3),* `pyrtri` *(b1-b3),* `pyrstar` *(c1-c3), and* `pyrholes` *(d1-d3) data. The* `pyrrrect` *structure forms a dense rectangular base tapering to a narrow tip, while* `pytri` *shows a more triangular spread with sharper edges.* `pyrstar` *extends into multiple pointed branches radiating from a common core, and* `pyrholes` *reveals hollow or open regions within an otherwise compact shape. These projections illustrate a range of pyramid-like geometries that vary in density and structure.*

S-curve

The S-curve is a smooth, nonlinear manifold in 3-*D* space. Using `gen_scurve(n)`, it is defined by

$$X_1 = \sin(\theta), \quad X_2 \sim U(0,2), \quad X_3 = \text{sign}(\theta)(\cos(\theta) - 1), \quad \theta \sim U(-3\pi/2, 3\pi/2).$$

This follows the `s_curve()` function from snedata (Melville 2025), itself adapted from *scikit-learn*, but differs by returning a tibble with standardized names (`x1`, `x2`, `x3`), excluding the color variable, and omitting built-in noise (which can be added separately). S-curve is commonly used in manifold learning and dimension reduction as benchmarks for unfolding curved structure.

Sphere

Sphere-shaped structures are useful for evaluating how dimension reduction and clustering algorithms handle curved, symmetric manifolds in high-dimensional spaces. Throughout this section, we follow the standard mathematical terminology: a *sphere* refers to the hollow $(p-1)$-dimensional surface in $\mathbb{R}^p$, while a *ball* refers to the filled interior region. The functions generate a variety of spherical forms, including simple circles, uniform and hollow spheres, grid-based spheres, and complex arrangements like clustered spheres within a larger sphere. The first few coordinates define the main geometric form (circle, cycle, sphere, or hemisphere), while higher-dimensional embeddings are achieved by adding noise dimensions. Such spherical or hemispherical structures frequently appear in physical and biological systems, for example, in models of celestial bodies, molecular shells, or cell membranes (Alberts et al. 2014; Tinkham 2003).

The simplest case, `gen_circle(n, p)`, creates a unit circle in two dimensions, with the remaining dimensions forming sinusoidal extensions of the angular parameter at progressively smaller scales (Figure 4.5 a). Let a latent angle variable $\theta \sim U(0, 2\pi)$. Coordinates in the first two dimensions represent a perfect circle on the plane:

$$X_1 = \cos(\theta), \quad X_2 = \sin(\theta).$$

For dimensions X_3 through X_p, sinusoidal transformations of the angle θ are introduced. The first component is a scaling factor that decreases with the dimension index, defined as $s_j = \sqrt{(0.5)^{j-2}}$ for $j = 3, \ldots, p$. The second component is a phase shift that is proportional to the dimension index, specifically designed to decorrelate the curves, given by the formula $\phi_j = (j-2)\pi/2p$. Each additional dimension is computed as: $X_j = s_j \sin(\theta + \phi_j), \quad j = 3, \ldots, p$.

For the one-dimensional nonlinear cycle embedded in p-D space, `gen_curvycycle(n, p)` (Figure 4.5 b), let a latent angle variable $\theta \sim U(0, 2\pi)$. The first three dimensions define a non-circular closed curve, referred to as a `"curvy cycle"`. In this configuration, $X_1 = \cos(\theta)$ represents horizontal oscillation, while $X_2 = \sqrt{3}/3 + \sin(\theta)$ introduces a vertical offset to avoid centering the curve at the origin. Additionally, $X_3 = 1/3\cos(3\theta)$ introduces a third harmonic perturbation that intricately folds the curve three times along its path, creating a unique and complex shape that oscillates in both dimensions while incorporating the effects of the harmonic perturbation.

Together, these define a periodic, non-trivial, closed curve in 3-D with internal folds that produce a more complex geometry than a standard circle or ellipse. For dimensions X_4 through X_p, additional structured variability is introduced through decreasing amplitude scaling and phase-shifted sine waves. The scaling factor is defined as $s_j = \sqrt{(0.5)^{j-3}}$ for j ranging from 4 to p, which means that the amplitude decreases as the dimension increases. Each dimension X_j is then calculated using the formula $X_j = s_j \sin(\theta + \phi_j)$, where the phase shift ϕ_j is given by $\phi_j = (j-2)\pi/(2p)$.

Building on simple circular structures, the `gen_unifsphere(n, r)` function extends the idea to three dimensions by generating n observations approximately uniformly distributed on the surface of a sphere of radius r. Each observation is computed from spherical coordinates, with $u \sim U(-1, 1)$ representing $\cos(\phi)$ and $\theta \sim U(0, 2\pi)$ the azimuthal angle. Cartesian coordinates are then defined as

$$X_1 = r\sqrt{1-u^2}\cos(\theta), \quad X_2 = r\sqrt{1-u^2}\sin(\theta), \text{ and } X_3 = ru,$$

ensuring uniform distribution on the surface (not within) of the sphere.

In contrast, the `gen_hollowsphere(n, p)` function, a wrapper around `geozoo::sphere.hollow()`, generates n points uniformly distributed only on the surface of the $(p-1)$-dimensional sphere embedded in $\mathbb{R}^p$. This results in a hollow shell-like structure with no interior points. For example, when $p = 3$, `gen_unifsphere()` produces a solid ball in 3-D space, whereas `gen_hollowsphere()` produces only the spherical boundary. These paired structures allow controlled experiments to investigate how algorithms behave when data is concentrated throughout the full volume versus constrained to the boundary.

In addition, the `gen_gridedsphere(n)` function constructs a p-dimensional dataset consisting of approximately n points arranged on the surface of the unit $(p-1)$-sphere embedded in $\mathbb{R}^p$ (Figure 4.5 d). Rather than sampling points uniformly, this function creates a deterministic grid in spherical coordinates, using $(p-1)$ angular variables: the first $(p-2)$ angles are taken from $[0, \pi]$, and the final angle from $[0, 2\pi]$. The number of grid points along each angular dimension is determined by decomposing n into $(p-1)$ approximately equal integer factors via `gen_nproduct(n, p - 1)`.

Each grid point is subsequently mapped into Cartesian space via the standard hyperspherical-to-Cartesian transformation,

$$
\begin{aligned}
X_1 &= \cos(\theta_1), \\
X_2 &= \sin(\theta_1)\cos(\theta_2), \\
X_3 &= \sin(\theta_1)\sin(\theta_2)\cos(\theta_3), \\
&\vdots \\
X_{p-1} &= \sin(\theta_1)\sin(\theta_2)\cdots\sin(\theta_{p-2})\cos(\theta_{p-1}), \\
X_p &= \sin(\theta_1)\sin(\theta_2)\cdots\sin(\theta_{p-2})\sin(\theta_{p-1}).
\end{aligned}
$$

The result is a deterministic grid of points lying exactly on the surface of the unit $(p-1)$-sphere, without any additional noise dimensions.

For more heterogeneous structures, the `gen_clusteredspheres(n, k, r, loc)` function generates one large sphere of radius r_1 and k smaller spheres of radius r_2, each centered at a different random location (Figure 4.5 e). A large Uniform ball centered at the origin is created by sampling n_1 points uniformly on the surface of a p-D sphere with a radius of r_1. The sampling is executed using the function `gen_unifsphere(n_1, r_1)`, which generates the desired points in the specified dimensional space. In the generation of k smaller Uniform balls, each sphere contains n_2 points that are sampled uniformly on a sphere with a radius of r_2. These spheres are positioned at distinct random locations in p-D, with the center of each sphere being drawn from a normal distribution $N(0, \texttt{loc}^2 I_p)$. Points on spheres are generated using the standard hyperspherical method, which involves sampling $u \sim U(-1, 1)$ to determine the cosine of the polar angle, and sampling $\theta \sim U(0, 2\pi)$ to determine the azimuthal angle (for 3-D). Each observation is classified by cluster, with labels such as "big" for the large central sphere and "small_1" to "small_k" for the smaller spheres.

Finally, the `gen_hemisphere(n, p)` function restricts sampling to a hemisphere of a 4-D sphere (Figure 4.5 f). Using spherical coordinates, the azimuthal angle $\theta_1 \sim U(0, \pi)$ in the (x_1, x_2) plane, while the elevation angle $\theta_2 \sim U(0, \pi)$ in the (x_2, x_3) plane. Additionally, $\theta_3 \sim U(0, \pi/2)$ in the (x_3, x_4) plane, ensuring that the points remain restricted to a hemisphere. The coordinates are transformed into 4-D Cartesian space:

$$
X_1 = \sin(\theta_1)\cos(\theta_2), \quad X_2 = \sin(\theta_1)\sin(\theta_2), \quad X_3 = \cos(\theta_1)\cos(\theta_3), \quad X_4 = \cos(\theta_1)\sin(\theta_3).
$$

This produces points on one side of a 4-D unit sphere, effectively generating a 4-D hemisphere.

Swiss Roll

The Swiss roll is a plane curled into 3-D, and is a commonly used example of a nonlinear manifold. The `gen_swissroll(n, w)` generates points as $X_1 = t\cos(t), \quad X_2 = t\sin(t), \quad X_3 \sim U(w_1, w_2), \quad t \sim$

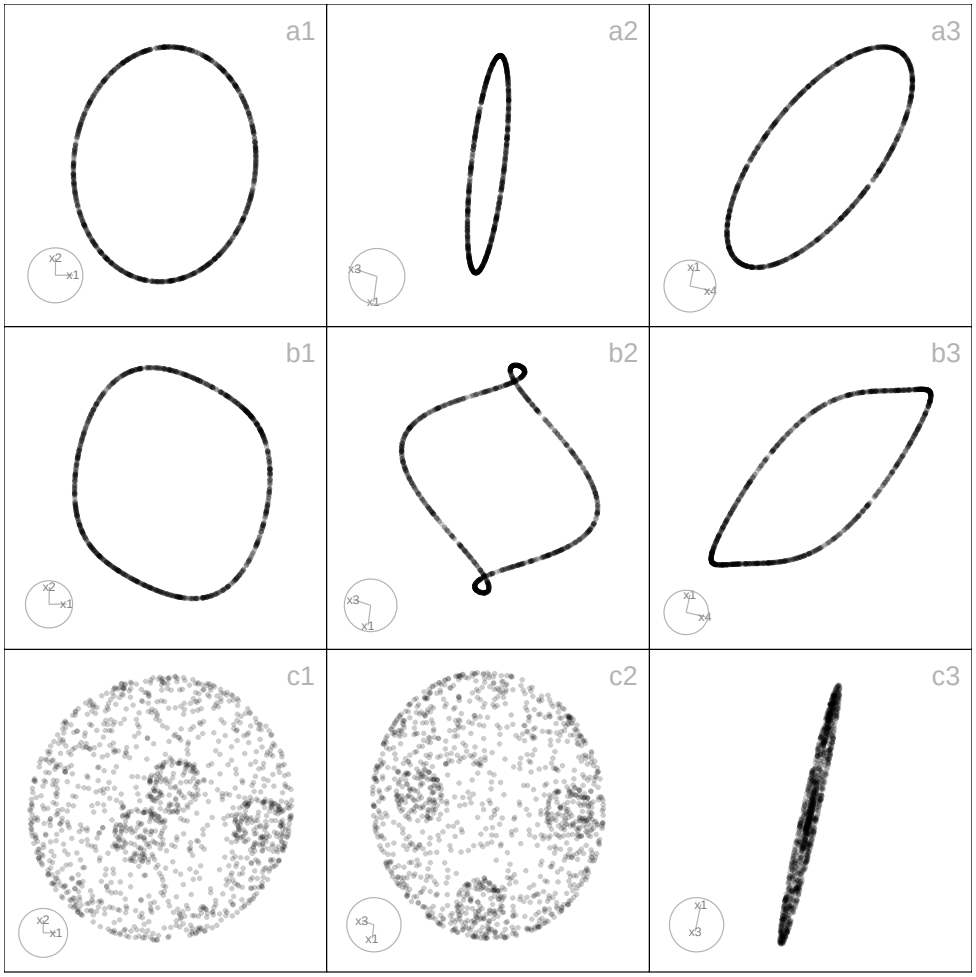

Figure 4.5: *Three 2-D projections from 4-D, `circle` (a1-a3), `curvycycle` (b1-b3), and, 3-D `clusteredspheres` (c1-c3). The `circle` structure forms a smooth, closed loop, while `curvycycle` shows a wavy, continuous pattern forming a twisted ring. The `clusteredspheres` dataset displays multiple compact spherical groups that are clearly separated in higher dimensions but overlap slightly in some 2-D projections, highlighting how projection can distort spatial relationships. These projections show how simple cyclic, wavy curvilinear, and clustered structures appear in 2-D, emphasizing the effects of projection on density, continuity, and separation*

$U(0, 3\pi)$.

Compared with `snedata::swiss_roll()` (Melville 2025), this implementation (i) samples t over $[0, 3\pi]$ instead of $[1.5\pi, 4.5\pi]$, (ii) allows a flexible vertical range $w = (w_1, w_2)$ rather than fixing $z \in [0, z_{\max}]$, and (iii) returns a tibble with `x1, x2, x3` instead of adding a color variable.

The Swiss roll is a classic benchmark for manifold learning, illustrating how a curved surface can be "unrolled" into lower dimensions. Similar spiral-like forms appear in galaxies, protein folding, and coiled materials (Agrafiotis and Xu 2002).

Trefoil knots

The Trefoil is a closed, nontrivial one-dimensional manifold embedded in 3-*D* or 4-*D* space (Figure 4.6). The trefoil features topological complexity in the form of self-overlaps, making it a valuable test case for evaluating the ability of nonlinear dimension reduction methods to preserve global structure, loops, and embeddings in high-dimensional data.

For the 4-*D* trefoil knot (Laurent 2024), the function `gen_trefoil4d(n, steps)` generates the structure on the 3-sphere ($S^3 \subset \mathbb{R}^4$) using two angular parameters, θ and ϕ. A band of thickness around the knot path is controlled by the `steps` argument, while the number of θ and ϕ values is determined by the `steps` and `n` arguments, respectively (Figure 4.6 a). The coordinates are defined as

$$X_1 = \cos(\theta)\cos(\phi), \quad X_2 = \cos(\theta)\sin(\phi), \quad X_3 = \sin(\theta)\cos(1.5\phi), \text{ and } X_4 = \sin(\theta)\sin(1.5\phi),$$

where θ parameterizes the band thickness and ϕ parameterizes the knot trajectory.

For the 3-*D* stereographic projection (Laurent 2024), `gen_trefoil3d(n, steps)` maps each point $(X_1, X_2, X_3, X_4) \in \mathbb{R}^4$ to $(X_1', X_2', X_3') \in \mathbb{R}^3$ using $X_1' = X_1/(1-X_4)$, $X_2' = X_2/(1-X_4)$, and $X_3' = X_3/(1-X_4)$, excluding points where $X_4 = 1$ to avoid division by zero (Figure 4.6 b).

The trefoil knot appears in molecular biology (DNA/protein knotting), fluid dynamics (knotted vortices), and physics (topological phases), making it a useful benchmark for testing whether dimension reduction preserves global loops and topology (Arsuaga et al. 2002; Witten 1985).

Trigonometric

Trigonometric-based structures provide flexible ways to simulate complex curved patterns and spirals that often arise in real-world high-dimensional data, such as in biological trajectories, or physical systems (Figure 4.7). The main geometry is defined by the first few coordinates: crescents ($p = 2$), cylinders, spirals, and helices ($p = 4$). These structures are particularly valuable for testing how well dimension reduction and clustering algorithms preserve intricate geometric and topological features (Calladine et al. 2004; Gershenfeld 2000).

First, the `gen_crescent(n, p)` function generates a p-dimensional dataset of n observations based on a 2-*D* crescent-shaped manifold with optional structured high-dimensional noise (Figure 4.7 a). Let $\{\theta_i\}_{i=1}^n$ be a sequence of n evenly spaced angles on the interval $[\pi/6, 2\pi]$, defined as $\theta_i = \frac{\pi}{6} + (i-1)\frac{2\pi - \pi/6}{n-1}$, $\quad i = 1, \ldots, n$. The corresponding 2-*D* coordinates are defined by:

$$X_{i1} = \cos(\theta_i), \quad X_{i2} = \sin(\theta_i).$$

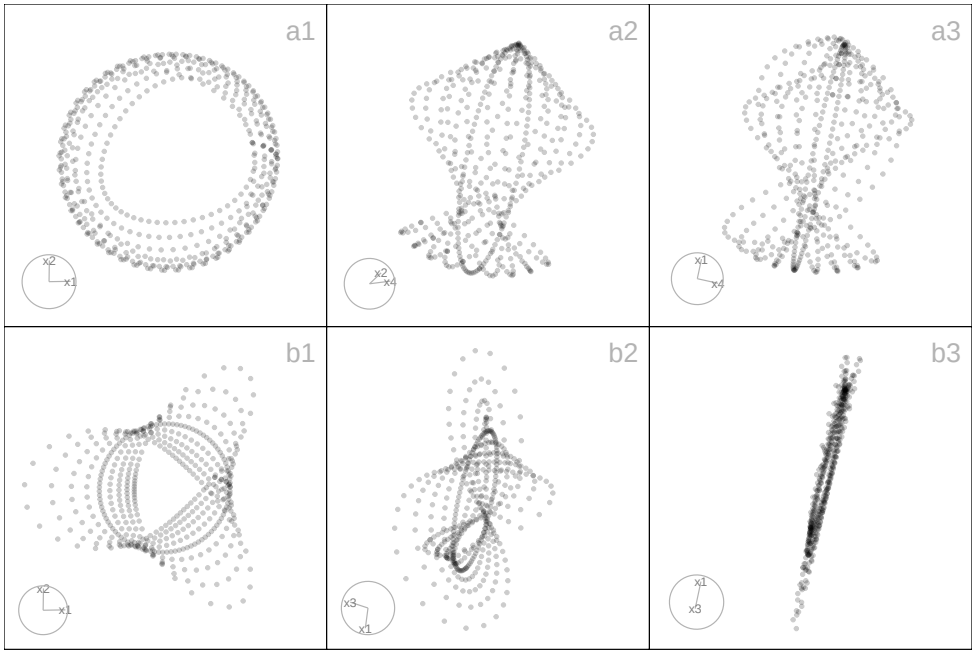

Figure 4.6: *Three 2-D projections from 4-D, `trefoil4d` (a1-a3) and 3-D `trefoil3d` (b1-b3) data. The `trefoil4d` structure represents a higher-dimensional extension of the classic trefoil knot, revealing complex twisting and looping patterns that remain continuous across projections. In contrast, the `trefoil3d` dataset maintains a simpler, more compact knot-like form, showing how dimensional extension adds curvature and separation in the embedded space. These projections illustrate a range of looping structures in high-dimensions.*

Second, the `gen_curvycylinder(n, p, h)` function generates a p-D dataset of n observations structured as a 3-D cylindrical manifold with an added nonlinear curvy dimension, and optional noise dimensions when $p > 4$ (Figure 4.7 b). The core structure consists of a circular base and height values, extended by a nonlinear fourth dimension. Let $\theta \sim U(0, 3\pi)$ represent a random angle on a circular base and $z \sim U(0, h)$ represent the height along the cylinder. The coordinates are defined as: $X_1 = \cos(\theta)$ (Circular base, x-axis), $X_2 = \sin(\theta)$ (Circular base, y-axis), $X_3 = z$ (Linear height), and $X_4 = \sin(z)$ (Nonlinear curvy variation along height).

For a spiraling path on a spherical surface in the first four dimensions, `gen_sphericalspiral(n, p, spins)` (Figure 4.7 c), let $\theta \in [0, 2\pi \times \text{spins}]$ be the azimuthal angle (longitude), controls the number of spiral turns and the $\phi \in [0, \pi]$ be the polar angle (latitude), controls the vertical sweep from the north to the south pole. Cartesian coordinates from spherical conversion: $X_1 = \sin(\phi)\cos(\theta)$, $X_2 = \sin(\phi)\sin(\theta)$, $X_3 = \cos(\phi) + \varepsilon$, where $\varepsilon \sim U(-0.5, 0.5)$ introduces vertical jitter, and $X_4 = \theta / \max(\theta)$: a normalized progression along the spiral path. This generates a spherical spiral curve embedded in 4-D space, combining both circular and vertical movement, with gentle curvature and nonlinear progression.

For a helical spiral in four dimensions, `gen_helicalspiral(n, p)` (Figure 4.7 d), let $\theta \in [0, 5\pi/4]$ be a sequence of angles controlling rotation around a circle. Cartesian coordinates; $X_1 = \cos(\theta)$:

circular trajectory along the x-axis, $X_2 = \sin(\theta)$: circular trajectory along the y-axis, $X_3 = 0.05\theta + \varepsilon_3$, with $\varepsilon_3 \sim U(-0.5, 0.5)$: linear progression (height) with vertical jitter, simulating a helix, and $X_4 = 0.1\sin(\theta)$: oscillates with θ, representing a periodic "wobble" along the fourth dimension.

Similarly, the `gen_conicspiral(n, p, spins)` function generates a dataset of n points forming a conical spiral in the first four dimensions of p-D (Figure 4.7 e). The geometry combines radial expansion, vertical elevation, and spiral deformation, simulating a structure that fans out like a 3-D conic helix. The shape is defined by parameter $\theta \in [0, 2\pi \times \text{spins}]$, controlling the angular progression of the spiral. The Archimedean spiral in the horizontal plane is represented by; $X_1 = \theta\cos(\theta)$ for radial expansion in x, and $X_2 = \theta\sin(\theta)$ for radial expansion in y. The growth pattern resembles a cone, with the height increasing according to $X_3 = 2\theta/\max(\theta) + \varepsilon_3$, with $\varepsilon_3 \sim U(-0.1, 0.6)$. Spiral modulation in the fourth dimension is represented by $X_4 = \theta\sin(2\theta) + \varepsilon_4$, with $\varepsilon_4 \sim U(-0.1, 0.6)$, which simulates a twisting helical component in a non-radial dimension.

Finally, the `gen_nonlinear(n, p, hc, non_fac)` function simulates a nonlinear 2-D surface embedded in higher dimensions, constructed using inverse and trigonometric transformations applied to independent variables (Figure 4.7 f). The $X_1 \sim U(0.1, 2)$: base variable (avoids zero to prevent division errors), $X_3 \sim U(0.1, 0.8)$: independent auxiliary variable, $X_2 = hc/X_1 + \text{non\_fac}\sin(X_1)$: nonlinear combination of hyperbolic and sinusoidal transformations, creating sharp curvature and oscillation, and $X_4 = \cos(\pi X_1) + \varepsilon$, with $\varepsilon \sim U(-0.1, 0.1)$: additional nonlinear variation based on cosine, simulating more subtle periodic structure. These transformations together result in a nonlinear surface warped in multiple ways: sharp vertical shifts due to inverse terms, smooth waves from sine and cosine, and additional jitter.

4.3.2 Generate a spherical or hyperspherical hole within a structure

The package provides functionality for generating datasets with spherical hole (in 2-D/3-D) or, more generally, hyperspherical hole (in higher dimensions). These structures are valuable for evaluating how dimension reduction methods and clustering algorithms handle incomplete manifolds or missing regions of the data space. A hyperspherical hole introduces topological complexity: the structure remains continuous but contains excluded regions (voids) that algorithms must correctly represent in lower-dimensional embeddings.

The core function `gen_hole(df, anchor, r)` removes points from a dataset that fall within a user-specified hypersphere. Formally, given data points ($x \in \mathbb{R}^p$), a center ($a \in \mathbb{R}^p$), and radius ($r > 0$), only points satisfying $\sqrt{\sum_{i=1}^{n}(x_i - a_i)^2} > r$ are retained. The anchor point (a) can either be user-specified or default to the dataset mean, and radius (r) is controlled by the user, with safeguards

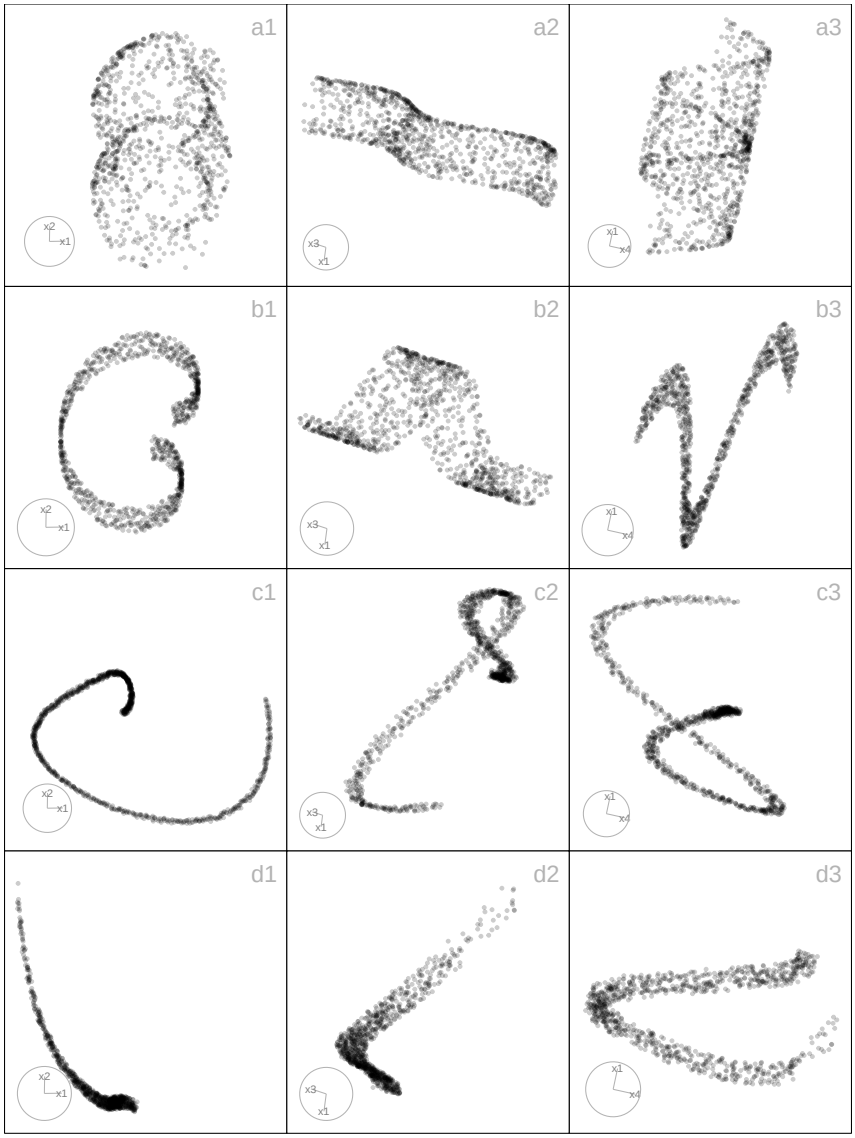

Figure 4.7: *Three 2-D projections from 4-D, for the* `curvycylinder` *(a1-a3),* `sphericalspiral` *(b1-b3),* `conicspiral` *(c1-c3), and* `nonlinear` *(d1-d3) data. The* `curvycylinder` *shows a cylindrical manifold with a nonlinear twist along its height, producing smooth, continuous curvature. The* `sphericalspiral` *forms a spiral path on a spherical surface, combining circular and vertical motion in a helical form. The* `conicspiral` *spreads radially while ascending, forming a conical helix with twisting variations in a non-radial dimension. The* `nonlinear` *dataset exhibits a warped 2-D surface with sharp oscillations and smooth waves, reflecting complex nonlinear interactions. Each shows variations in curvature, density, and continuity.*

to avoid trivial or degenerate cases. Because it operates generically on any dataset, spherical or hyperspherical holes can be embedded in a wide range of geometric structures.

Two specialized wrappers illustrate this idea. The function `gen_scurvehole(n, r_hole)` generates an S-curve with a spherical hole by applying `gen_hole()` to the output of `gen_scurve()`. This structure has been used in prior diagnostic studies of NLDR methods (Maaten et al. 2007; Tenenbaum et al. 2000), since it tests the ability of algorithms to capture nonlinear manifolds that are not

Table 4.4: *`cardinalR` noise dimensions generation functions.*

Function	Explanation
`gen_noisedims`	Gaussian noise dimensions with optional mean and standard deviation.
`gen_wavydims1`	Wavy noise dimensions based on a user-specified theta sequence with added jitter.
`gen_wavydims2`	Wavy noise dimensions using polynomial transformations of an existing dimension vector.
`gen_wavydims3`	Wavy noise dimensions using a combination of polynomial and sine transformations based on the first three dimensions of a dataset.

simply connected. The second wrapper, `gen_unifcubehole(n, p, r_hole)`, generates uniformly sampled cube data with a hyperspherical hole. By embedding a hyperspherical void inside a convex high-dimensional structure, this creates non-convex regions that challenge algorithms in terms of separability and neighborhood preservation.

4.3.3 Generate noise dimensions

High-dimensional data structures often benefit from the addition of auxiliary noise dimensions, which can be used to assess the robustness of dimension reduction and clustering algorithms. The functions in this section provide flexible ways to generate random noise dimensions, ranging from purely random Gaussian variables to more structured, wavy patterns that mimic nonlinear distortions in high-dimensional space. These functions can be applied independently or combined with other geometric structures to create complex simulated datasets. Table 4.4 details these functions.

The `gen_noisedims(n, p, m, s)` function generates p independent Gaussian noise dimensions,

$$X_j \sim N(m_j, s_j^2), \quad j = 1, \ldots, p,$$

with odd-numbered dimensions multiplied by -1. This does not affect independence, since all noise dimensions are generated independently. The sign alternation is included only to avoid consistent directional drift and to ensure a symmetric appearance of noise when visualized or combined with other simulated structures.

For scenarios where noise should follow a smooth, wavy pattern, `gen_wavydims1(n, p, theta)` generates dimensions as

$$X_j = \alpha_j \theta + \varepsilon_j, \quad \varepsilon_j \sim N(0, \sigma^2), \quad j = 1, \ldots, p,$$

where each dimension is scaled by a different factor α_j, producing structured noise that oscillates along the latent parameter θ, mimicking trends or trajectories observed in real-world data.

The `gen_wavydims2(n, p, x_1)` function extends this approach by applying a nonlinear transformation to an existing dimension vector x_1:

$$X_j = \beta_j (-1)^{\lfloor j/2 \rfloor} x_1^{k_j} + \varepsilon_j, \quad j = 1, \ldots, p,$$

where k_j is a randomly chosen polynomial power, β_j is a scaling factor, and ε_j is small uniform noise.

Finally, `gen_wavydims3(n, p, data)` generates noise for datasets with multiple correlated dimensions. The first three dimensions are small perturbations of the original coordinates (X_1, X_2, X_3), while higher dimensions are constructed via nonlinear combinations, including polynomial and trigonometric transformations, e.g.,

$$X_j = f_j(X_1, X_2, X_3) + \varepsilon_j, \quad j > 3,$$

producing high-dimensional noise that preserves some geometric correlation with the base structure while introducing additional complexity.

4.3.4 Rotating shape generators

In *p-D* space, a rotation is an orthogonal transformation that changes the orientation of data while preserving its total variance and pairwise distances. The function `gen_rotation()` generates such rotation matrices for any dimension, given a list of rotation planes (axis pairs) and angles.

4.3.5 Multiple cluster examples

By using the shape generators mentioned above, we can create various examples of multiple clusters. The package includes some of these examples, which are described in Table 4.5.

4.3.6 Additional functions

The package includes various supplementary tools in addition to the shape-generating functions mentioned earlier. These tools allow users to create background noise, randomize the rows of the data, relocate clusters, generate a vector whose product and sum are approximately equal to a target value, rotate structures, and normalize the data. Table 4.6 details these functions. More detailed explanations are available in https://jayanilakshika.github.io/cardinalR/articles/03additionalfun.html.

Table 4.5: *`cardinalR` multiple clusters generation functions.*

Function	Explanation
`make_mobiusgau`	Möbius-like cluster combined with a Gaussian.
`make_multigau`	Multiple Gaussian clusters in high-dimensional space.
`make_curvygau`	Curvilinear cluster with a Gaussian cluster.
`make_klink_circles`	K-link circular clusters (nonlinear circular patterns).
`make_chain_circles`	Chain-like circular clusters connected sequentially.
`make_klink_curvycycle`	K-link curvy cycle clusters (curvilinear loop structures).
`make_chain_curvycycle`	Chain-like curvy cycle clusters connected sequentially.
`make_gaucircles`	Circular clusters with a Gaussian cluster in the middle.
`make_gaucurvycycle`	Curvy circular clusters with a Gaussian in the middle.
`make_onegrid`	Single grid in two dimensions.
`make_twogrid_overlap`	Two overlapping grids.
`make_twogrid_shift`	Two grids shifted relative to each other.
`make_shape_para`	Parallel shaped clusters.

Table 4.6: *`cardinalR` additional functions.*

Function	Explanation
`gen_bkgnoise`	Adds background noise.
`randomize_rows`	Randomizes the rows of input data.
`relocate_clusters`	Relocates the clusters.
`gen_nproduct`	Generates a vector of positive integers whose product is approximately equal to a target value.
`gen_nsum`	Generates a vector of positive integers whose summation is approximately equal to a target value.
`normalize_data`	Normalizes data.

4.4 Application

This section demonstrates how the package can be used to generate complex high-dimensional datasets, evaluate dimension reduction (DR) and clustering methods. The example shows how diverse geometric structures can be simulated and analyzed to assess algorithmic behavior.

To illustrate how high-dimensional clustered data can be generated using `cardinalR`, we generate a dataset with five clusters in 4-*D*, each representing distinct geometric characteristics: a *helical spiral* (elongated and twisted), a *hemisphere* (curved surface), a *uniform cube* (isotropic distribution), a *cone* (density gradient), and a *Gaussian* cluster (compact and spherical) (Figure 4.8). Each cluster has a unique number of points and scaling factor, representing variation in cluster size and spread across the 4-*D* space.

```
positions <- geozoo::simplex(p=4)$points
positions <- positions * 0.3

five_clusts <- gen_multicluster(n = c(2250, 1500, 750, 1250, 1750), k = 5,
                        loc = positions,
                        scale = c(0.25, 0.35, 0.3, 1, 0.3),
                        shape = c("helicalspiral", "hemisphere", "unifcube",
                                  "cone", "gaussian"),
                        rotation = NULL,
                        is_bkg = FALSE)
```

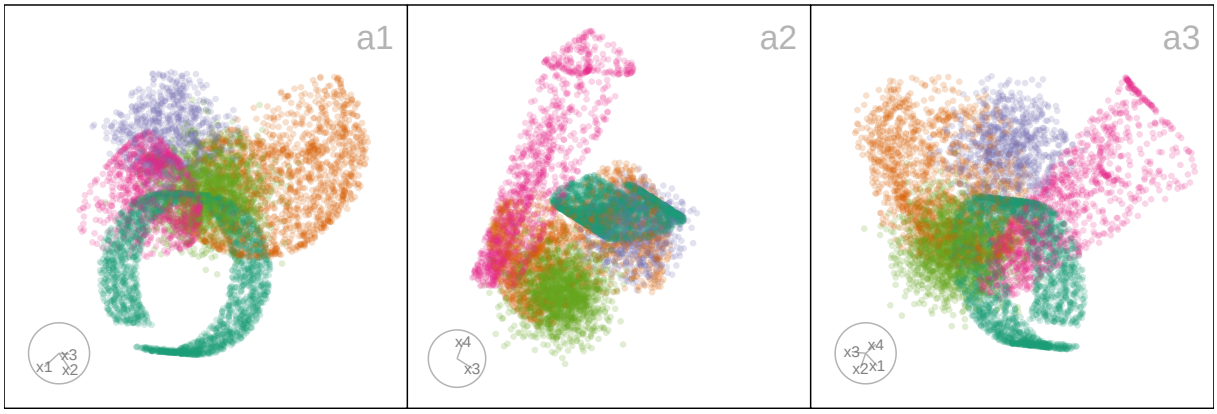

Figure 4.8: *Three 2-D projections from 4-D, for the five clusters data. The helical spiral cluster is represented in dark green, the hemisphere cluster in orange, the uniform cube-shaped cluster in purple, the blunted cone cluster in pink, and the Gaussian-shaped cluster in light green.*

4.4.1 Evaluating dimension reduction (DR) methods

We applied six popular DR techniques to the generated dataset: Principal Component Analysis (PCA) (Jolliffe 2011), tSNE, uniform manifold approximation and projection (UMAP) (McInnes et al. 2018), potential of heat-diffusion for affinity-based trajectory embedding (PHATE) algorithm (Moon et al. 2019), large-scale dimensionality reduction using triplets (TriMAP) (Amid and Warmuth 2019), and pairwise controlled manifold approximation (PaCMAP) (Wang et al. 2021).

To assess their performance, we computed the hexbin error (HBE) between the observed high-dimensional data and the fitted values, defined as the high-dimensional mappings of the bin centroids (Gamage et al. 2025c). A lower HBE indicates that the method better preserves the high-dimensional structure in its low-dimensional embedding.

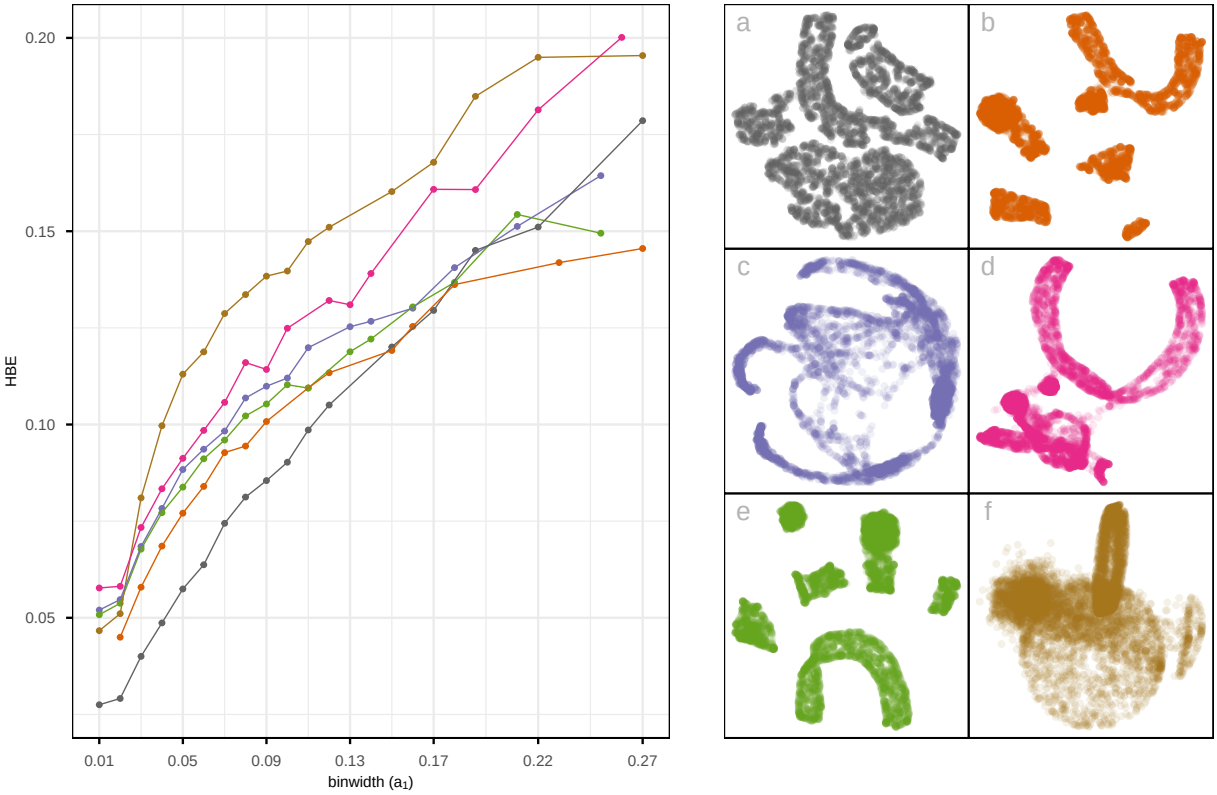

Figure 4.9: *Assessing which of the 6 NLDR layouts ((a) tSNE, (b) UMAP, (c) PAHTE, (d) TriMAP, (e) PaCMAP, and (f) PCA) of the five clusters data is the better representation using HBE for varying binwidth (a_1). Color is used for the lines and points in the left plot to match the scatterplots of the NLDR layouts (a-f). Layout f is universally poor. Layouts a and b are universally optimal. Layout b shows six well-separated clusters and layout a shows close clusters, thus layout a is the best choice.*

As shown in Figure 5.1, tSNE (Figure 5.1 a) achieved the lowest HBE across bin widths (mostly tiny), indicating high preservation of both local and global structures. Its layout displays well-separated clusters with minimal inter-cluster distances, making it the most faithful representation of the underlying data structure. UMAP and PaCMAP (Figure 5.1 b and e) produced moderately accurate embeddings, although the six clusters appear more well-separated, while PHATE (Figure 5.1 c) shows nonlinear cluster structures irrespective of the original structure. Also, TriMAP (Figure 5.1 d) has high HBE and shows three clusters with small distances. PCA (Figure 5.1 f) failed to capture the nonlinear geometry, leading to the highest HBE.

4.4.2 Benchmarking clustering algorithms

To further evaluate the structure of the generated data, we benchmarked three clustering algorithms: *k*-**means** (Chapter 20 of Boehmke and Greenwell 2019), **hierarchical** (Murtagh and Contreras 2012), and **model-based clustering** (Fraley and Raftery 2002; Scrucca et al. 2023) using the simulated dataset. Model-based clustering performed the `"EII"` covariance structure. Under this

parameterization, clusters are spherical with equal volume and equal shape, and no orientation parameter is estimated. Cluster validity statistics were computed using the `cluster.stats()` function from the `fpc` package (Hennig 2024).

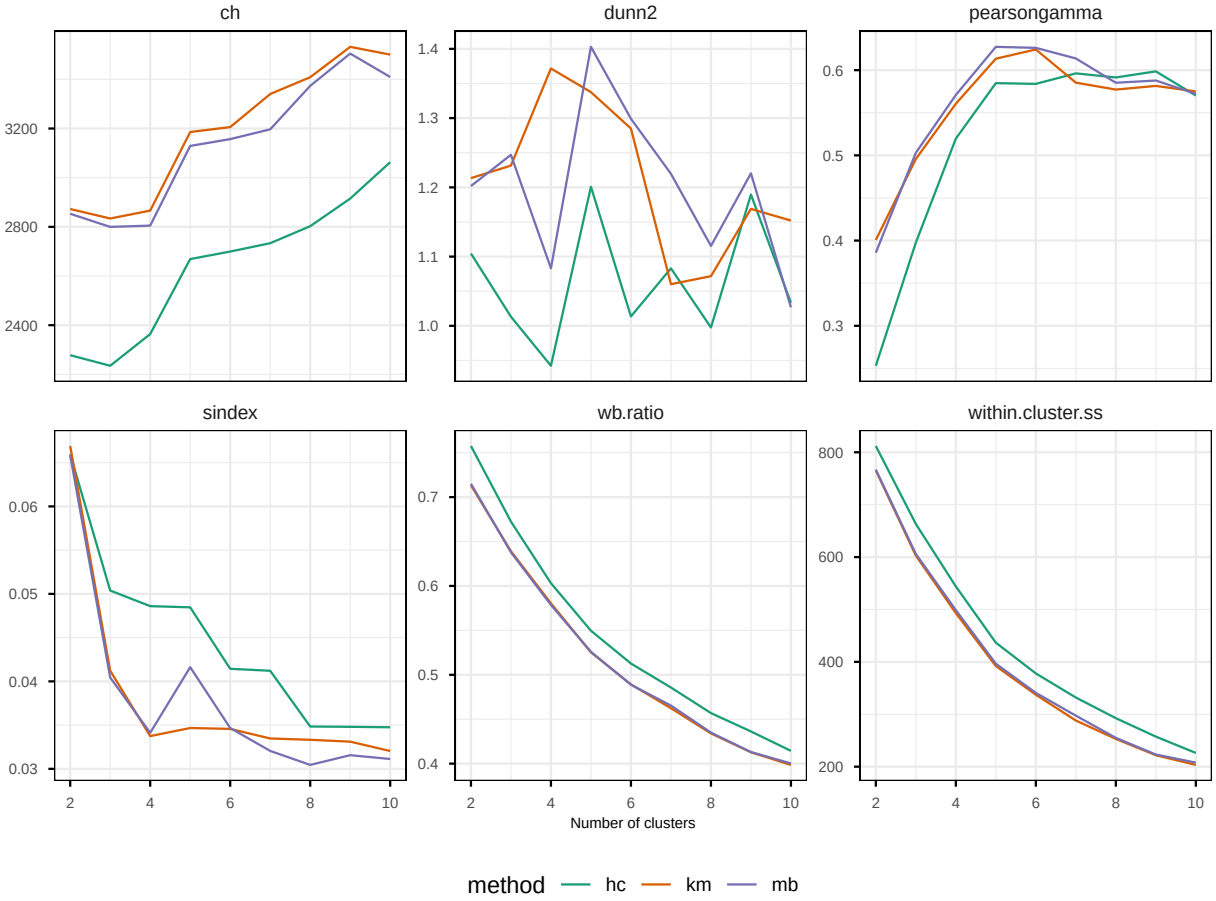

Figure 4.10: *Cluster validity metrics for solutions with* $2-10$ *clusters obtained using k-means, hierarchical, and model-based clustering. Several indices consistently suggest that* $4-5$ *clusters provide the best balance of separation and compactness, with k-means performing slightly better across metrics.*

Figure 4.10 shows a selection of cluster metrics for $2-10$ clusters for each of the methods, *k*-means, hierarchical, and model-based. As is typical, the suggestion of the best solution varies between cluster statistics. Although the metrics differ in their preferences, several show consistent support for a $4-5$ cluster solution. Pearson gamma (`pearsongamma`) increases sharply up to five clusters before leveling off, Calinski–Harabasz index (`ch`) increases sharply from 4 to 5 clusters, and Dunn (`dunn2`) has a maximum at 5 for two methods and at 4 for *k*-means. All of these are interpreted as higher is better. With the other three, lower is better. WB ratio (`wb.ratio`) and within-cluster sum of squares (`within.cluster.ss`) steadily decline with the number of clusters, possibly elbowing around 5 clusters. The S-index (`sindex`) is optimized at 4 clusters for *k*-means, 3, 6, or 8 for hierarchical clustering, and 4 or 8 for model-based. Overall, *k*-means performs slightly better than the hierarchical and model-based clustering across most metrics and number of clusters.

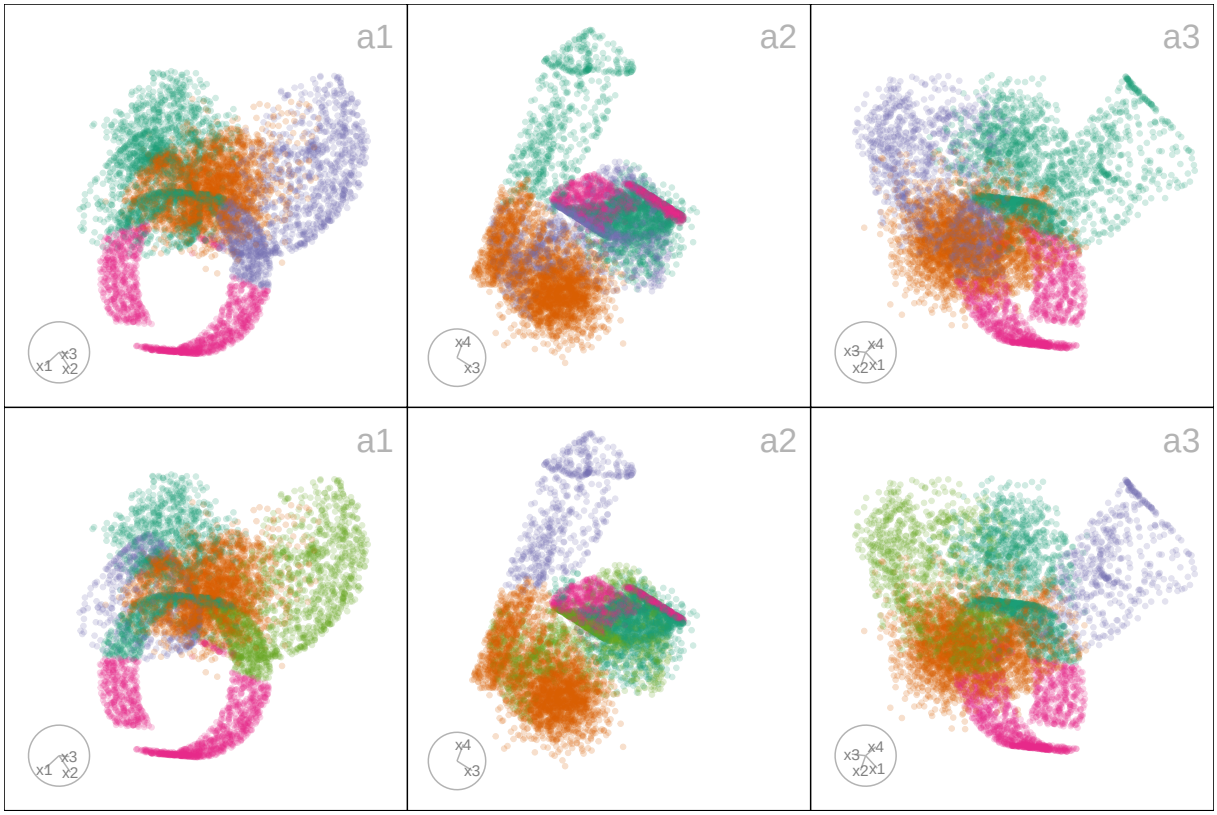

Figure 4.11: *Three 2-D projections from 4-D, for the five clusters data colored by the k-means four- (a1-a3) and five-cluster (b1-b3) solutions. The intermixing of colors within each projection reflects misclassification in both solutions, showing the difficulty of using k-means to capture the dataset's nonlinear and heterogeneous shapes.*

Figure 4.11 shows the four- and five-cluster *k*-means solutions, with cluster id used to color the points. Neither solution captures the geometric nature of the true clusters, but they are both reasonable partitions of the data. To examine either one, it is best to subset to a single cluster to view in the tour. With each solution, the five original shapes are each split by the clustering. More than 5 clusters would be needed to better capture the original shapes.

4.5 Conclusion

The `cardinalR` package introduces a flexible framework for generating high-dimensional data structures with well-defined geometric properties. It addresses an important need in the evaluation of clustering, machine learning, and DR methods by enabling the construction of customized datasets with interpretable structures, noise characteristics, and clustering arrangements. In this way, `cardinalR` complements existing packages such as `geozoo`, `snedata`, and `mlbench`, while extending the scope to higher dimensions and more complex shapes.

The included structures cover a wide range of diagnostic settings. Branching shapes facilitate the study of continuity and topological preservation, the S-curve with a hole allows investigation of incomplete manifolds, and clustered spheres assess separability on curved surfaces. The Möbius strip introduces challenges from non-orientable geometry, while gridded cubes and `pyrholes` test spatial regularity and clustering in sparse, non-convex regions.

These structures are designed to support not only algorithm diagnostics, but also teaching high-dimensional concepts, benchmarking reproducibility, and evaluating hyper-parameter sensitivity. By allowing users to adjust dimensionality, sample size, noise, and clustering properties, the package promotes transparent experimentation and comparative model evaluation. Together, these capabilities make `cardinalR` a versatile tool for generating interpretable, high-dimensional datasets that advance research, teaching, and evaluation of data-analytic methods.

Future extensions of `cardinalR` may include biologically inspired or application-driven data structures that would further broaden its utility in domains such as bioinformatics, forensic science, and spatial analysis.

4.6 Acknowledgements

The source material for this chapter, including the full datasets and figures, is available at https://github.com/JayaniLakshika/paper-cardinalR. This article is created using `knitr` (Xie 2015) and `rmarkdown` (Xie et al. 2018) in R with the `rjtools::rjournal_article` template. These `R` packages were used for this work: `cli` (Csárdi 2025), `tibble` (Müller and Wickham 2023), `gtools` (Warnes et al. 2023), `dplyr` (Wickham 2023), `stats` (R Core Team 2025), `tidyr` (Wickham et al. 2024), `purrr` (Wickham et al. 2025), `mvtnorm` (Genz and Bretz 2009), `geozoo` (Schloerke 2016), and `MASS` (Venables and Ripley 2002).

Chapter 5

Perception and Misperception of Clustering in Nonlinear Dimension Reduction: A User Study

Nonlinear dimension reduction (NLDR) methods such as tSNE, UMAP, PHATE, TriMAP, and PaCMAP are popular ways to visualize high-dimensional data, yet their effectiveness for conveying structure remains mysterious. Many factors might contribute to perceptual miscommunication, which for cluster structure, may include how their shapes are represented, the degree of separation, or even the number of clusters. This study evaluates how well NLDR methods preserve perceptually meaningful cluster structure using a human subject experiment with simulated data having three clusters with distinct geometries, unequal sizes, and varying inter-cluster separation. Subjects were asked whether a 2-*D* NLDR layout and a tour of linear projections showed the same high-dimensional data. Cluster separation was controlled for the study to be the distance between means, but for analyzing the results, two additional measures, the between-within (BW) ratio and the exponentially scaled minimum inter-cluster distance, were used to account for highly nonlinear shapes. The results suggest interesting differences across methods. For example, UMAP and tSNE represent the distance between clusters distinctly differently, resulting in data being interpreted differently. These findings highlight the need for more studies to assess NLDR methods based on how effectively their visualizations support human perception of high-dimensional structure.

5.1 Introduction

Nonlinear dimension reduction (NLDR) is popular for making a suitable 2-*D* representation of high-dimensional (*p*-*D*) data by applying nonlinear transformations. Recently developed methods include t-distributed stochastic neighbor embedding (tSNE) (Maaten and Hinton 2008), uniform manifold approximation and projection (UMAP) (McInnes et al. 2018), potential of heat-diffusion for affinity-based trajectory embedding (PHATE) algorithm (Moon et al. 2019), large-scale dimensionality reduction using triplets (TriMAP) (Amid and Warmuth 2019), and pairwise controlled manifold approximation (PaCMAP) (Wang et al. 2021).

Nonlinear transformations allow for multiple shape-varying clusters to be represented in a single 2-*D* layout. In contrast, classical linear projection will often require multiple projections to show multiple clusters. Figure 5.1 illustrates this: a1-a4 show linear projections revealing three well-separated clusters, one spherical, one ribbon-like, and one like a star-shaped pyramid. The NLDR layout (left) is generated using tSNE and has a mostly reasonable display of the three clusters in a single view, although it struggles with the star pyramid. It does place the clusters very close to each other, which does not reflect the large separation in the high-dimensional space.

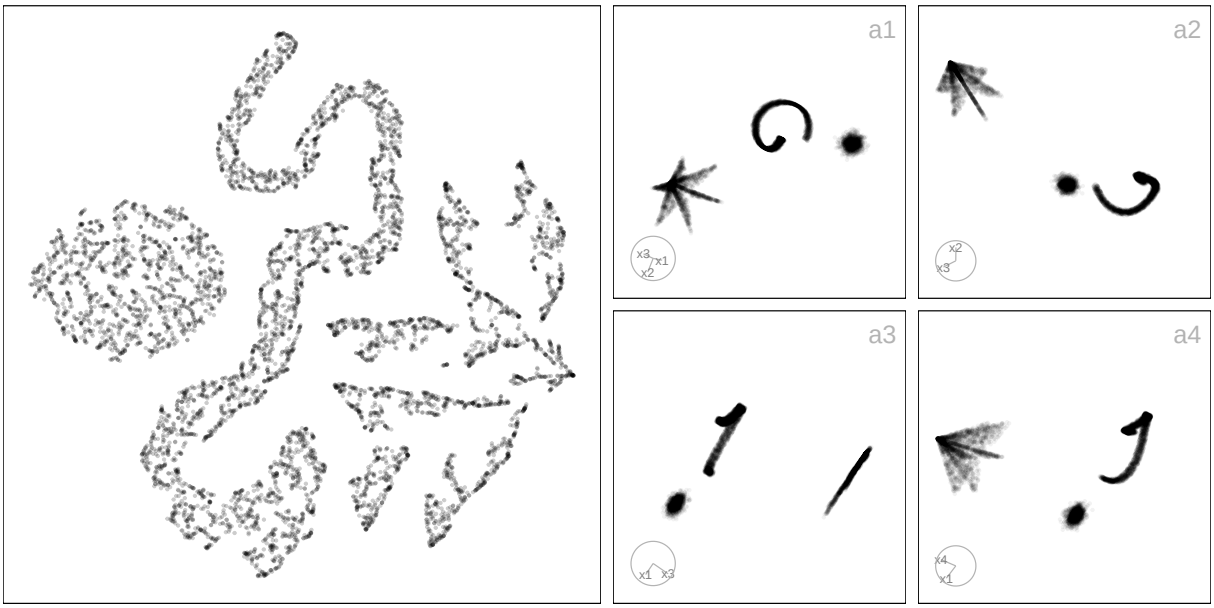

Figure 5.1: *A 2-D tSNE layout (left) and four 2-D projections (a1–a4) of the same 4-D data. The data consist of three main structures: a star-shaped pyramid, a curvilinear cluster, and a Gaussian-shaped cluster. While the tour consistently shows the star-shaped cluster as a single coherent group, the 2-D tSNE layout fragments this structure into several smaller clusters. This illustrates how NLDR may distort global structure, making the same 4-D cluster appear as multiple clusters in the 2-D layout.*

In general, the dilemma for the analyst is to make the conceptual leap from the structure displayed in the NLDR layout to what exists in high dimensions. From Figure 5.1 we might find that the analyst

correctly conceptualizes the existence of the spherical and ribbon clusters, but mistakenly considers them close in high dimensions. The star-shaped pyramid might be incorrectly conceptualized as a lot of small clusters, possibly triangular in shape. This is what the work presented here is attempting to assess: whether the conceptualization from the NLDR reasonably matches that gained by viewing the same data using a tour of linear projections.

The chapter is organized as follows. Section 5.2 provides a summary of the literature on NLDR, high-dimensional data, and visualization methods. Section 5.3 describes the experiment designed to examine people's perception to assess how viewers recognize structure differently from a 2-*D* NLDR layout and the tour view. Section 5.4 discusses the collected data, results, and reasons for misperception. Limitations are provided in Section 5.5. A discussion of the presented work and ideas for future directions is described in Section 5.6.

5.2 Background

Historically, 2-*D* nonlinear representations of *p*-*D* data have been obtained through versions of multidimensional scaling (MDS) (originally defined by Kruskal (1964), and see Borg and Groenen (2005) for a modern overview) and linear representations using principal component analysis (PCA) (for an overview see Jolliffe (2011)). MDS aims to construct a low-dimensional (usually 2-*D*) layout that preserves pairwise distances between observations in the original space by minimizing a stress function. Challenges such as distance concentration that lead to difficulties for interpretation have been documented by Johnstone and Titterington (2009).

NLDR methods have been developed to improve on MDS with varying degrees of preserving local and/or global structures of *p*-*D* data, with some modern methods being tSNE, UMAP, PHATE, TriMAP, and PaCMAP. Each method uses different underlying principles. For example, tSNE and PHATE emphasize local relationships, while TriMAP and PaCMAP are designed to better capture global structure. As a result, these methods can produce very different 2-*D* layouts of the same data, potentially leading to misinterpretation of structures such as cluster separation.

An alternative to NLDR for visualizing *p*-*D* data is to use linear projections. PCA is the classical approach, producing new variables as linear combinations of the original dimensions. While PCA provides a single static projection that maximizes variance, tours introduced by Asimov (1985) extend this idea by generating smooth sequences of linear projections, effectively creating a movie of the data viewed from multiple directions. Tours can reveal structures that may be hidden in any single projection by continuously changing the viewing angle through high-dimensional space. Many tour

algorithms have since been developed and are implemented in the R package `tourr` (Wickham et al. 2011), with interactive variants available in `langevitour` (Harrison 2023) and `detourr` (Hart and Wang 2025). Tours are valuable because they preserve the geometry of the data, unlike NLDR methods - they do not warp distances or angles. This makes them faithful but sometimes visually cluttered representations: global structure can obscure local detail, and the phenomenon of piling (Laa et al. 2022), where high-dimensional points project toward the center, can make clusters harder to distinguish.

Quantifying clusters in shape and separation is not simple. For this experiment, a variety of shapes were generated using the functions in the `cardinalR` package (Gamage et al. 2025b). Measuring distance between clusters is classically done using the between-within (BW) ratio, which captures global separability if the cluster shape is elliptical. A variety of distance-based metrics have been proposed in the clustering and visualization literature (Calinski and Harabasz 1974; Davies and Bouldin 1979; Rousseeuw 1987), including minimum, maximum, and average distances between clusters, centroid distances, and ratios that combine between- and within-cluster variation. Although the data sets were created with a fixed process, the results will be examined with a variety of distance metrics to capture NLDR behavior using different lenses of separation.

The objective of this research is to study analyst perception of clustering structure in a 2-*D* NLDR layout comparison with that from a tour of the same high-dimensional data. The tour is generated using `langevitour`. The primary factor of interest is how the perception changes when cluster separation increases.

5.3 Methods

Although there are many aspects of NLDR and perception of data structure to assess, for this work, we restrict attention to the distance between clusters. For a range of cluster shapes, the distance between clusters is varied, and NLDR layouts are generated by the commonly used methods with default settings. The conceptualization of clustering is tested by showing subjects two views (one NLDR layout and the tour of linear projections) and asking whether both show the same data. When the response is that they are the same, it is interpreted as that they conceptualize the clustering in both similarly. Conversely, if the response is that the two are different, it is interpreted as a different conceptualization.

5.3.1 Data generation

A total of 30 4-D data sets are generated. Two are reserved as an attention check used to determine if the subject conscientiously attempted the task. All data sets were standardized prior to NLDR and are shown in the tour.

Non-attention check data

For the experiment, three cluster data sets are generated. The three clusters contain different numbers of points and shapes. Let C_1, C_2, and C_3 denote the centroids of three clusters. The pairwise distances between these centroids are calculated as: $d(C_1, C_2) = c_{12}$, $d(C_1, C_3) = c_{13}$, and $d(C_2, C_3) = c_{23}$. At the original distance scale (scale factor 1, referred to as medium-large), clusters C_1 and C_2 are in close proximity, while cluster C_3 is positioned farther away, creating an asymmetric separation pattern. Centroid distances were used because they provide a simple and controllable way to adjust overall cluster separation.

In the SAME trials, the degree of separation between clusters was varied by multiplying the original centroid distances by four scale factors: 0.1 (small), 0.6 (small-medium), 0.9 (medium), and 1.1 (large). These values were chosen to span a range of perceptual difficulty from cases where clusters are expected to overlap strongly and be hard to distinguish (0.1), through intermediate levels where separation is visible but ambiguous (0.6 and 0.9), to cases where clusters are clearly separated (1.1). Using proportional scaling ensures that the relative geometry of the data is preserved while systematically controlling how strongly separation cues are expressed.

In contrast, data structures used for the DIFFERENT trials retained the original centroid distances (scale factor 1) without modification. This allows the DIFFERENT trials to serve as stable reference cases while ensuring that variation in separation is introduced only in trials where participants are asked to judge whether two displays show the same data.

Shapes for each cluster were selected randomly from a predefined set of curved, linear, and volumetric structures, including S-curves, crescents, spirals, hyperbolic and cylindrical shapes, as well as geometric solids such as cubes, hemispheres, pyramids, cones, and Gaussian clusters.

Attention check data

There are two sets of attention check data: one consisting of three Gaussian clusters and the other consisting of four Gaussian clusters. Each cluster is generated using a multivariate normal distribution where the mean vectors and variances were predefined. Specifically, for the three-cluster case, the mean vectors were set as $[1, 0, 0, 0]$, $[0, 1, 0, 0]$, and $[0, 0, 1, 1]$, with a common variance of 0.1 for all clusters. For the four-cluster case, the mean vectors were defined as $[1, 0, 0, 1]$, $[0, 1, 1, 0]$, $[1, 0, 1, 0]$,

and $[0, 1, 0, 1]$, also using a variance of 0.1. This approach ensures that data points are normally distributed around the specified centroids, with the spread controlled by the variance parameter. Each Gaussian cluster dataset consists of 4-*D* data with a sample size of 7500, and each cluster contains an equal number of data points.

5.3.2 Organization of SAME and DIFFERENT trials

Although the main analysis focuses on trials where the same data are shown in both displays, it is essential to include DIFFERENT trials in the experiment. Without them, participants could rely on a trivial strategy—such as always responding "SAME" and still achieve high accuracy. DIFFERENT trials, therefore, act as a necessary control, ensuring that correct responses in SAME trials reflect genuine perceptual agreement between the NLDR layout and the tour rather than response bias or guessing.

Therefore, the experiment was designed to include a mixture of SAME, DIFFERENT, and attention check trials. In total, 28 non–attention check data structures were used. Of these, 18 data structures were assigned to SAME trials, where the same high-dimensional data structure was used to generate both the 2-*D* NLDR plot and the tour. These trials are the primary focus of the analysis.

The remaining 10 data structures were used to create DIFFERENT trials. In these cases, the NLDR plot and the tour were generated from two distinct but related data structures. For example, when data structure `three_clust_19` appeared in the NLDR plot, `three_clust_20` was shown in the tour. Although these DIFFERENT trials are not analyzed directly, they play a crucial role in maintaining the integrity of the task by preventing systematic response strategies.

In addition, two clearly separable Gaussian cluster data sets were included as attention checks. These appear as both SAME and DIFFERENT trials and are used to verify that participants are paying attention and are able to perform the task under easy conditions.

To avoid learning and familiarity effects, each participant sees each data structure only once. Data sets were therefore assigned to subjects randomly but without replacement at the subject level. This ensures that participants cannot rely on memory from earlier trials and that each judgment is based solely on the visual information presented.

5.3.3 Experiment design

The visual layout of the experiment for five subjects is shown in Figure 5.2. Each subject completed 20 trials: 15 SAME trials, in which the same data structure was shown in both the 2-*D* NLDR plot and the tour; 4 DIFFERENT trials, showing DIFFERENT data structures; and one attention check trial that could be either SAME or DIFFERENT. The purpose of the DIFFERENT trials was to ensure that

subjects didn't get too familiar with the task, which might happen if the data were always the same in both graphics. For the SAME, five NLDR methods (*tSNE, UMAP, PHATE, PaCMAP, and TriMAP*) were each paired with three of five distance scale factors (*small, small-medium, medium, medium-large, and large*), giving 15 balanced combinations. In the DIFFERENT, four NLDR methods were randomly selected, with the remaining method assigned to the attention check trial. All DIFFERENT and attention check trials used a distance scale factor of *medium-large*.

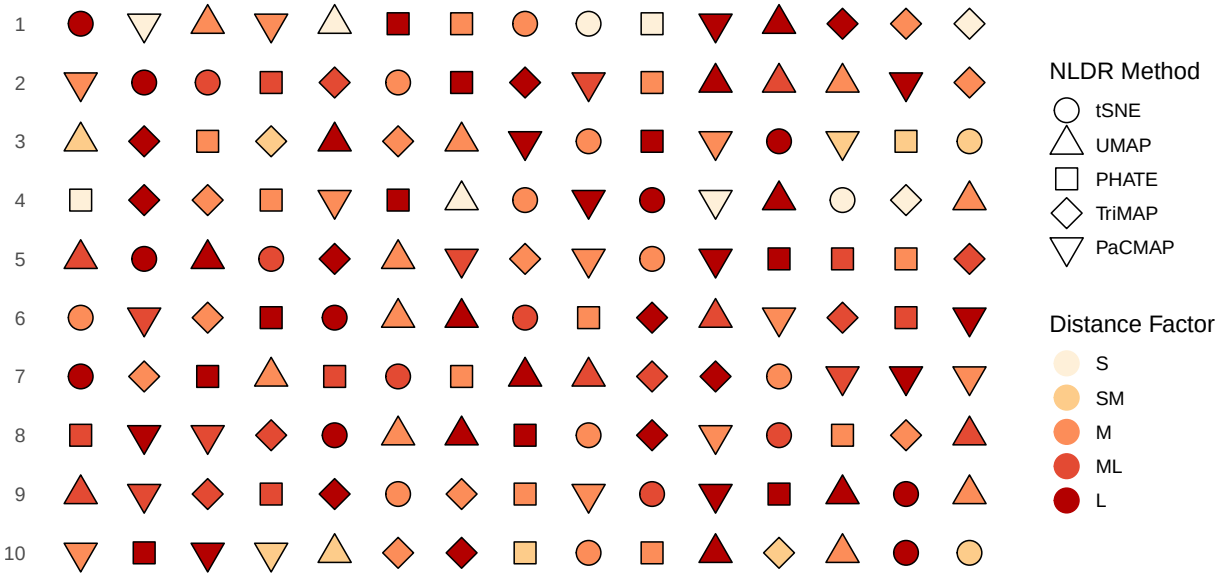

Figure 5.2: *Experimental design for ten subjects. Shapes represent NLDR methods, and fill colors denote distance scale factors, ranging from low to high separability and mapped from light to dark. The figure shows only the SAME trials, making it easier to see the balanced design: for each subject, all five NLDR methods (tSNE, UMAP, PHATE, TriMAP, and PaCMAP) are equally represented, and each method appears with three of the five distance scale factors (small, small–medium, medium, medium–large, and large), distributed across subjects. The order of trials is randomized within each subject. In the full experiment (not shown), DIFFERENT trials and attention checks were inserted at random positions. Each subject completed* 20 *trials in total:* 15 *SAME trials,* 4 *DIFFERENT trials comparing different data structures, and* 1 *attention check (SAME or DIFFERENT). All DIFFERENT and attention-check trials used a medium–large distance scale factor.*

5.3.4 Experimental factors

Two factors of interest were considered in the experiment: the NLDR method and the distance scale factor.

The first factor consisted of five NLDR methods: *tSNE, UMAP, PHATE, PaCMAP, and TriMAP*, each producing a 2-*D* representation.

The second factor, the distance scale factor, controlled the degree of cluster separation in the high-dimensional space. Five categorical levels: *small, small–medium, medium, medium–large, and large* were defined to represent increasing degrees of separability. This categorical design enhances inter-

pretability and perceptual distinctness, allowing subjects to discern meaningful structural differences while maintaining robustness against minor data variations.

In our analysis of the results, we decided to quantify the distances between clusters numerically rather than using the distance scale factor levels directly. Cluster separability was quantified using two complementary measures: the *between-to-within (BW) ratio* and the *minimum inter-cluster distance*. A higher value of either metric indicates greater separation among clusters (Figure 5.3). To ensure comparability across datasets with different underlying structures, all distance-based metrics were min–max scaled prior to analysis.

The BW ratio, defined as

$$\text{BW ratio} = \frac{\sum_{i=1}^{K} n_i \; d(\bar{\mathbf{x}}_i, \bar{\mathbf{x}})}{\sum_{i=1}^{K} \sum_{\mathbf{x}_j \in C_i} d(\mathbf{x}_j, \bar{\mathbf{x}}_i)},$$

where $d(\cdot, \cdot)$ denotes the Euclidean distance, C_i is the i^{th} cluster with n_i observations, $\bar{\mathbf{x}}_i$ is the centroid of cluster C_i, and $\bar{\mathbf{x}}$ is the overall centroid of the dataset.

In addition, the minimum distance was used as a complementary measure of global separation:

$$\text{minimum distance} = \min_{k \neq \ell} \min_{\mathbf{x}_i \in C_k, \mathbf{x}_j \in C_l} d(\mathbf{x}_i, \mathbf{x}_j),$$

which captures the closest proximity between any two clusters. The scaled minimum distance was exponentiated so that, where it agrees with the BW ratio, the relationship between the two measures is approximately linear. The transformation increases separation among larger distance values while leaving small distances largely unchanged, facilitating more comparable variation across datasets.

5.3.5 Subject recruitment

Subjects were recruited from the Prolific crowd-sourcing platform (Palan and Schitter 2018). The study expects that the subjects are uninvolved judges with no prior knowledge of the data to avoid inadvertently affecting results. Pre-screening procedures were applied the recruitment: potential subjects needed with fluent in English and have completed at least 10 Prolific studies with a 98% approval rate.

5.3.6 Data collection

The survey web application, Match-a-roo, was used for data collection. Subjects provided the introduction and instructions for the survey. Before starting the survey, the subjects can be led to

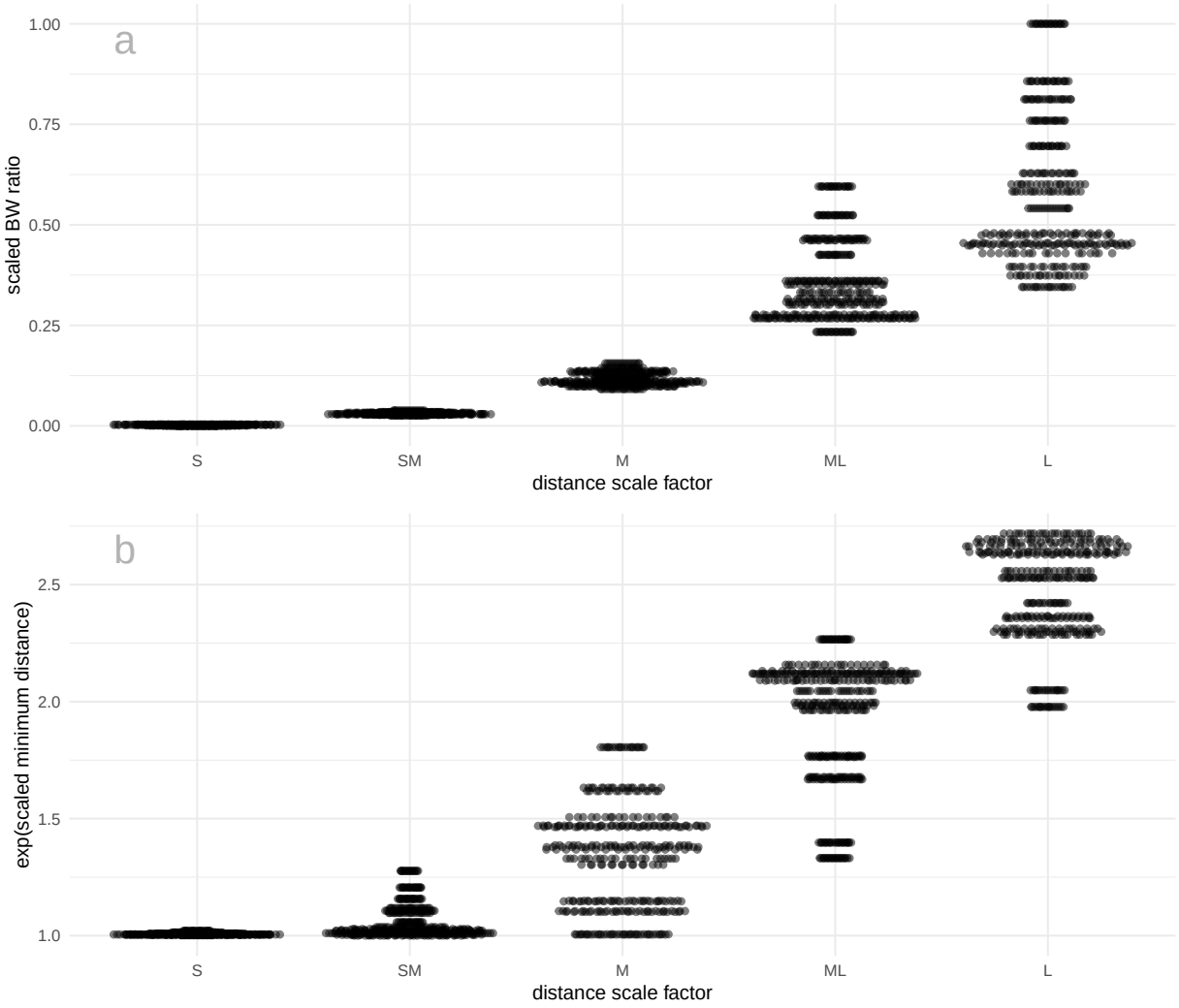

Figure 5.3: *Distribution of distance metric values across distance scale factors used as treatments in the experiment. (a) scaled between-to-within (BW) ratio and (b) exp(scaled minimum inter-cluster distance), each plotted against five categorical distance scale factors: small (S), small–medium (SM), medium (M), medium–large (ML), and large (L). Both metrics increase systematically with the scale factor, confirming that the distance scale treatment effectively controls cluster separability in the high-dimensional space.*

the "example" page, which allows them to experiment with the data collection interface and practice deciding whether the two displays show the same data or not. The main purpose of using the "example" was merely to familiarize the subjects with the questions that would be asked, as well as the process of deciding whether the two displays showed the same data or not. The interface did not provide any numeric feedback as to subject correctness.

The subjects were asked to provide their Prolific ID and their consent to the responses being used for analysis. After giving consent, the subject can start the trials. Two visual displays of data were shown, where the data may be the SAME or DIFFERENT. One of the visual displays is a 2-*D* NLDR plot, and the other is a tour. The subjects were asked to decide whether the data was the same in both displays and to report their confidence about their choice and any comments about the answer.

After completing 20 evaluations, they were asked for their demographics, which included preferred pronoun, the highest level of education achieved, their age category, whether they used principal component analysis in their work, and whether they applied NLDR techniques such as tSNE and UMAP.

5.3.7 Generalized linear mixed-effects models

Two generalized linear mixed effects models (McCulloch et al. 2001) were fitted to model the likelihood of detecting the data structure in both the 2-D NLDR layout and the tour (Equation 5.1). Both models accounted for subject-level variability and the effect of distance measures under different NLDR methods. The general form of the model is given by:

$$\text{logit}(P(y_{ijm} = 1)) = \mu_m + \beta_m d_i + \gamma_j, \tag{5.1}$$

where μ_m is the intercept, d_i is the distance measure for the data structure $i = 1, \ldots, 18$, β_m is the fixed effect of distance metric under NLDR method m, γ_j is the random effect of the subject $j = 1, 2, \ldots, 127$, where $\gamma_j \sim N(0, \sigma^2_\gamma)$. Separate models were fitted using d_i as either the scaled BW ratio or the exp(scaled minimum distance). The NLDR methods denoted by m can include TriMAP, UMAP, PaCMAP, tSNE, and PHATE. The models were fitted using the `lme4` package (Bates et al. 2015) and examined with the `emmeans` package (Lenth 2025).

5.4 Results

The data was collected from 127 subjects, resulting in $127 \times 15 = 1905$ evaluations, excluding the attention check trials and the trials showing the different data in two displays.

5.4.1 Effect of method and distance between clusters

The proportion of correct identifications across the NLDR methods and distance conditions was analysed to evaluate how effectively each method preserves cluster separation. Results are summarized using two generalized linear mixed-effects models, with either the scaled BW ratio (Figure 5.4, Table 5.1) or the exp(scaled minimum distance) (Figure 5.5, Table 5.2) as the distance predictor. Both models accounted for subject-level variability through random effects and included the NLDR method as a fixed factor interacting with the distance measure.

Results from the model using the scaled BW ratio (Table 5.1) indicate that cluster separability positively influences correct identification for some NLDR methods. As shown in Figure 5.4, UMAP exhibits

Table 5.1: *Estimated trends of correct identification probability with respect to scaled BW ratio by NLDR method. The table shows method-specific slope estimates (log-odds scale) for the effect of the scaled BW ratio on the probability of correct identification, obtained from a generalized linear mixed-effects model. Estimates represent the change in log-odds of correct identification per unit increase in scaled BW ratio for each NLDR method, along with standard errors (SE), 95% confidence intervals, Wald z-statistics, and corresponding p-values. p-values and Confidence Intervals are calculated assuming normally distributed errors in the estimates. Positive estimates indicate improved identification accuracy with increasing cluster separation, while negative estimates indicate declining accuracy. Significance codes: (*$p \leq 0.001$ *'⋆⋆⋆',* $p \leq 0.01$ *'⋆⋆',* $p \leq 0.05$ *'⋆',* $p \leq 0.1$ *'. ').*

Method	Slope	SE	95% CI	z	p
TriMAP	0.03	0.49	[-0.92, 0.99]	0.07	0.95
UMAP	1.15	0.49	[0.19, 2.11]	2.35	0.02 *
PaCMAP	0.51	0.48	[-0.43, 1.45]	1.06	0.29
tSNE	-2.61	0.62	[-3.83, -1.4]	-4.20	<0.001 ***
PHATE	-0.92	0.54	[-1.97, 0.13]	-1.71	0.09 .

a clear increase in accuracy as the scaled BW ratio increases, suggesting that this method benefits from greater between-cluster separation. PaCMAP shows a positive but weaker trend, while TriMAP maintains stable performance across the range of separations. In contrast, tSNE and PHATE display declining accuracy at higher BW ratios, indicating that increased separation may distort or obscure structural cues for these methods.

To assess whether these patterns depend on how separation is quantified, we fitted a second model using the exp(scaled minimum distance) as an alternative measure of cluster separability (Table 5.2). The results closely mirror those obtained with the BW ratio (Figure 5.5). In particular, UMAP again shows a significant positive association between separation and correct identification probability, confirming that greater spatial distance between clusters enhances its ability to reveal the underlying structure. Conversely, tSNE demonstrates a strong negative association, with performance deteriorating as minimum distance increases, while PHATE exhibits a weaker but consistent negative trend. The effects for PaCMAP and TriMAP are not statistically significant, indicating comparatively stable performance across varying levels of separation.

Taken together, these results demonstrate that the impact of cluster separability on correct identification is robust to the choice of distance measure but varies substantially across NLDR methods. Methods such as UMAP benefit from increased separation, whereas tSNE and PHATE appear sensitive to over-separation, potentially leading to distortions in the low-dimensional representation. TriMAP, by contrast, shows little sensitivity to changes in separation, suggesting robustness across a wide range of cluster configurations.

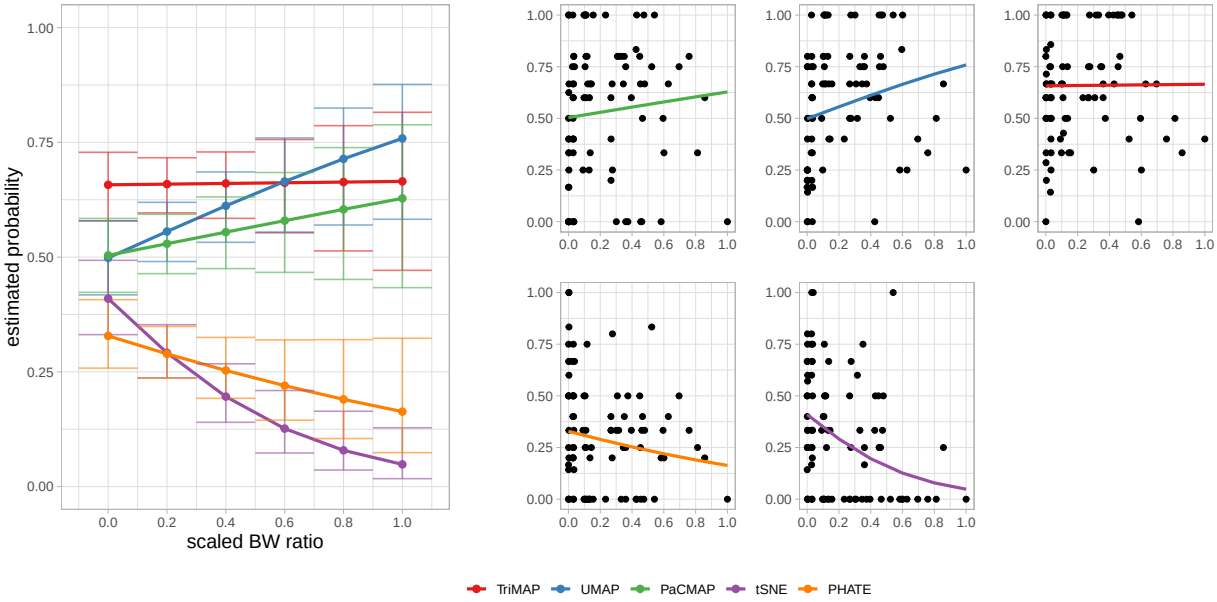

Figure 5.4: *Estimated probability of correct identification as a function of the scaled BW ratio for five NLDR methods. The left panel shows model-based estimated probabilities with 95% confidence intervals across values of the scaled BW ratio. The right panels show observed proportions of correct identification (black points) and fitted logistic regression curves for each method. Each black point represents the proportion of SAME responses from a distinct combination of data structure, distance scale factor, and NLDR method. The scaled BW ratio measures relative cluster separation, with larger values indicating greater separability. Performance trends differ across methods, with UMAP showing increasing accuracy, tSNE and PHATE decreasing accuracy, and TriMAP exhibiting relatively stable performance.*

5.4.2 Patterns conceptualization

The difference between tSNE and UMAP embeddings is curious: the further apart clusters are in high dimensions, the more often subjects reported that the data between the views was different when the embedding was tSNE. The UMAP results are more as expected, that the further apart the clusters, the more likely the subject is to report that they are the same data. Figure 5.6 shows the results for one data set called `three_clust_07`. Plots on the left (a1, a2, b1, b2) show linear projections from a tour, and plots on the right show embeddings by tSNE and UMAP. Rows correspond to small and large distances, respectively. The proportion of correct responses is shown in each embedding plot. (The total number of evaluations for each was 3, 4, 4, and 4, respectively. While there are relatively few evaluations for any single example like this one, this example serves to illustrate the general pattern.)

The reason for the difference in conceptualization from the different embeddings here is quite clear. Firstly, UMAP represents the data with large separation as three unusually shaped clusters that are well-separated. On the other hand, tSNE de-emphasizes the separation, and also does something worse - splits one cluster into two to make four clusters. It is understandable that a different conceptualization would be made from this embedding relative to that from the tour of linear projections, which clearly

Table 5.2: *Estimated trends of correct identification probability with respect to exp(scaled minimum distance) by the NLDR method. The table shows method-specific slope estimates (log-odds scale) for the effect of the exp(scaled minimum distance) on the probability of correct identification, obtained from a generalized linear mixed-effects model. Estimates represent the change in log-odds of correct identification per unit increase in exp(scaled minimum distance) for each NLDR method, along with standard errors (SE), 95% confidence intervals, Wald z-statistics, and corresponding p-values. p-values and Confidence Intervals are calculated assuming normally distributed errors in the estimates. Positive estimates indicate improved identification accuracy with increasing cluster separation, while negative estimates indicate declining accuracy. Significance codes:* ($p \leq 0.001$ '⋆⋆⋆', $p \leq 0.01$ '⋆⋆', $p \leq 0.05$ '⋆', $p \leq 0.1$ '.').

Method	Slope	SE	95% CI	z	p
TriMAP	0.20	0.20	[-0.2, 0.59]	0.97	0.33
UMAP	0.59	0.20	[0.2, 0.98]	2.99	0.00 **
PaCMAP	0.22	0.19	[-0.16, 0.6]	1.12	0.26
tSNE	-0.78	0.22	[-1.2, -0.35]	-3.60	<0.001 ***
PHATE	-0.35	0.21	[-0.76, 0.06]	-1.68	0.09 .

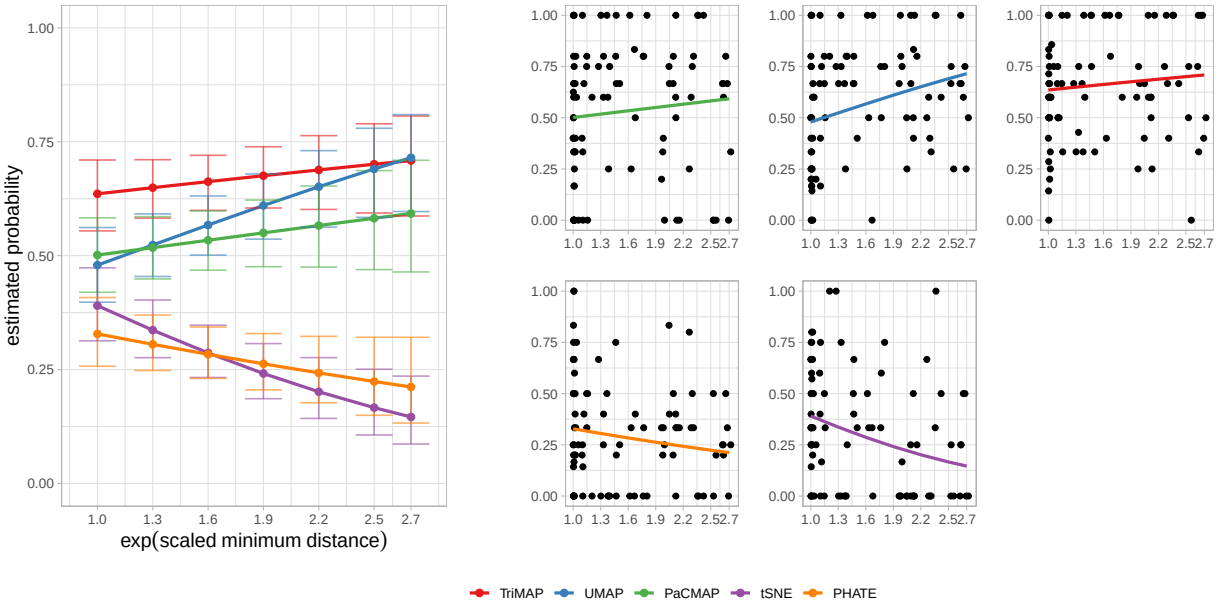

Figure 5.5: *Estimated probability of correct identification as a function of the exp(scaled minimum distance) for five NLDR methods. The left panel shows model-based estimated probabilities with 95% confidence intervals across values of the exp(scaled minimum distance). The right panels show observed proportions of correct identification (black points) and fitted logistic regression curves for each method. Each black point represents the proportion of SAME responses from a distinct combination of data structure, distance scale factor, and NLDR method. Larger values correspond to greater spatial separation between clusters. UMAP shows increasing accuracy with increasing separation, whereas tSNE and PHATE show declining trends, and TriMAP exhibits relatively stable performance.*

shows three clusters.

A different pattern is seen for the data set `three_clust_13`, which consists of a curvy cylinder, a cube, and a blunted cone, shown under small and large separation. (The total number of evaluations

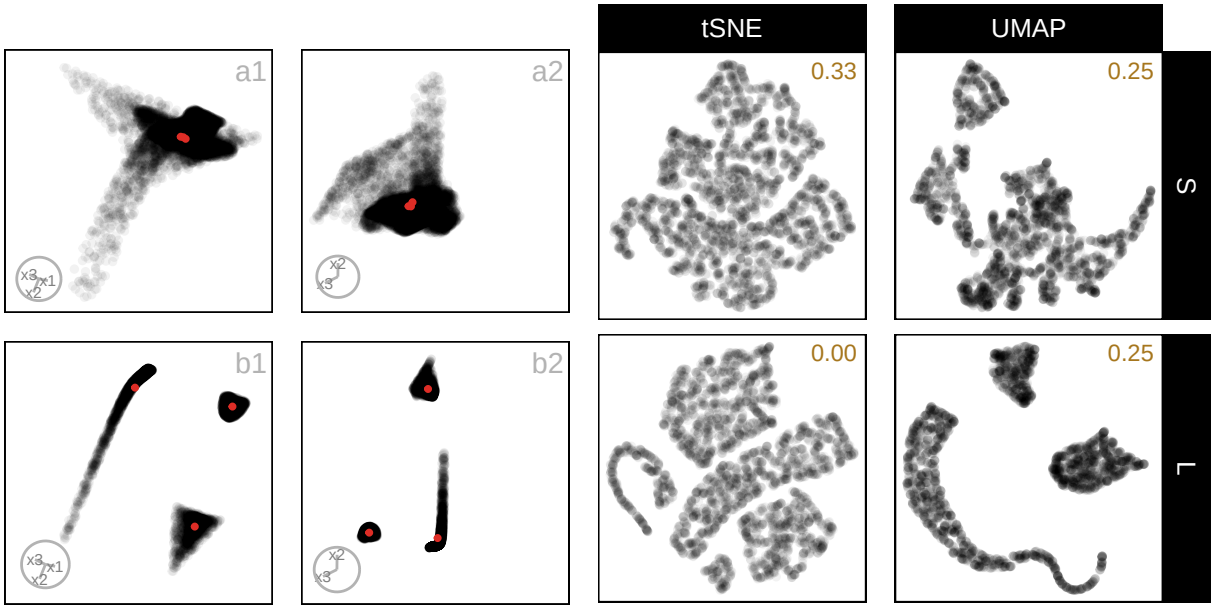

Figure 5.6: *tSNE and UMAP layouts and 2-D projections for the data structure* `three_clust_07`*, composed of a nonlinear hyperbola, a hemisphere, and a triangular pyramid, shown under small and large cluster separation. Panels (a1–a2) show two fixed 2-D projections at small separation, and panels (b1–b2) show the same projections at large separation; the corresponding tSNE and UMAP layouts are shown in the right panels. For each NLDR layout, the proportion of correct identifications for the corresponding method and distance factor is reported in the top-right corner of the plot. At a small separation, curved and rounded components overlap substantially in both methods, making the structure difficult to distinguish. With increased separation, UMAP yields smoother, more continuous representations that retain the curvature of the hyperbolic component and improve spacing between clusters. In contrast, tSNE bends and breaks the hyperbolic structure and introduces irregular gaps between points, weakening global shape cues.*

for each case was 3, 5, 6, and 6, respectively. While there are relatively few evaluations for any single example like this one, the figure is intended to illustrate a general pattern observed across multiple data sets.) Here, tSNE aligns more closely with the tour, particularly under small separation, where the proportion of correct responses is higher for tSNE (0.67) than for UMAP (0.00). In these layouts, tSNE preserves the overall grouping without introducing artificial splits, making it easier to reconcile the embedding with the linear projections. UMAP, on the other hand, emphasizes shape and density in ways that depart from the tour, especially when clusters are close together, leading to lower accuracy. Even at large separation, where UMAP improves (0.60), the visual cues remain less consistent with the tour than those produced by tSNE. This example highlights that which method leads to better conceptual alignment can depend strongly on the underlying data structure, and that neither embedding consistently dominates across all cases.

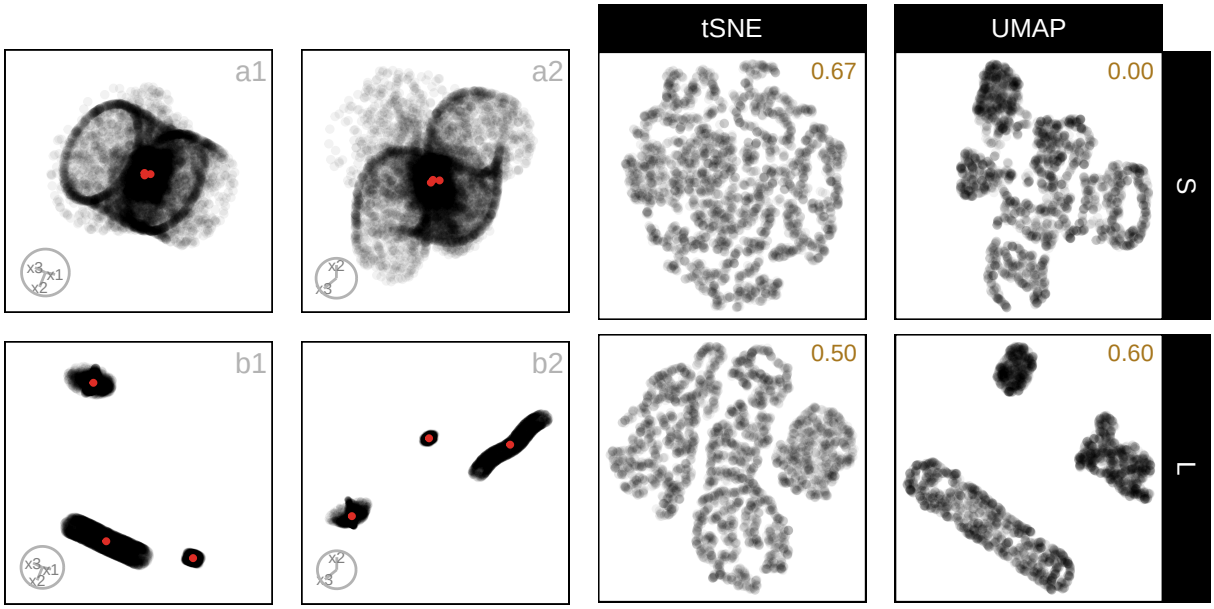

Figure 5.7: *tSNE and UMAP layouts and 2-D projections for the data structure* `three_clust_13`*, composed of a curvy cylinder, a cube, and a blunted cone, shown under small and large cluster separation. Panels (a1–a2) show two fixed 2-D projections at small separation, and panels (b1–b2) show the same projections at large separation; the corresponding tSNE and UMAP layouts are shown in the right panels. For each NLDR layout, the proportion of correct identifications for the corresponding method and distance factor is reported in the top-right corner of the plot. At small separation, the cube and blunted cone partially overlap in the linear projections, but tSNE preserves their separation more clearly than UMAP, leading to a higher proportion of correct responses. With increased separation, both methods improve in interpretability; however, UMAP still compresses the curvy cylinder toward the other components, while tSNE maintains clearer boundaries between the three clusters, supporting more consistent identification across separation levels.*

5.5 Limitations

One of the main drawbacks of visual experiments is their reliance on human judgments. In this context, the effectiveness of identifying the 2-*D* NLDR plot and the tour from the same data can be dependent on the perceptual ability and visual skills of the individual. However, when the results from multiple individuals are combined, the overall quality and robustness of the outcome are considerably higher.

In this study, we chose to use only three clusters, each with unique shapes, and placed two close together with the third located farther away. We also used different sample sizes for each cluster. The purpose was to ensure a manageable experiment as an initial project. Using 4-*D* data allows tours to convey structural information effectively without imposing excessive cognitive or visual load on viewers. Some of the results should hold for more clusters, different arrangements, and higher dimensions, but it would be interesting to expand the scope to check these factors in the future.

5.6 Conclusions

This study examined whether people can correctly identify that a static 2-*D* NLDR layout and a dynamic tour represent the same high-dimensional data, and how this ability depends on both cluster separation and the NLDR method used. Using three clusters with different shapes, numbers of points, and unequal separation, we were able to directly test whether increasing high-dimensional separation improves perceptual identification, and whether this effect varies across methods.

The results show that cluster separation does matter, but its impact is strongly method-dependent. The results show that UMAP and tSNE lead to significantly different conceptualizations as the distance between clusters increases. There is a hint that PaCMAP behaves like UMAP, and PHATE behaves like tSNE, and TriMAP conceptualization is not affected by distance, but these are not statistically significant patterns.

This experiment is best viewed as a template for further studies on how people interpret NLDR layouts. Future work could extend the factors studied by considering different numbers of clusters, sample size, varying noise levels, and dimensionality. For three clusters, including PCA (a linear method) as an embedding is not necessary because almost always it makes a useful display of three clusters in 2-*D* that is easily recognized as the same data when viewed with a tour. PCA is not needed to reveal three clusters; we included it as a positive control. PCA provides a case where the embedding is expected to closely match the tour, helping confirm that participants can correctly identify the same data when distortions are minimal. This makes it easier to interpret errors observed for nonlinear methods, where mismatches are more likely. For more than three clusters, this would not be true, and it would be important to include PCA as a comparison method.

Overall, this work highlights the need for more experiments that systematically assess the perception of structure in 2-*D* NLDR layouts with respect to structure present in high-dimensional data. With results from more human subject experiments, it may be possible to develop better metrics that could be used to automate the assessment of visual similarity measures. These would be helpful to use alongside NLDR layouts to assist with providing more faithful representations.

5.7 Supplementary materials

The appendix provides additional details on the experimental materials and process, including the three-cluster data structures, 2-*D* NLDR layouts, inter-cluster distance metrics, and the data collection and analysis processes, along with links to videos and scripts.

5.8 Acknowledgments

A pilot study was conducted with sample subjects from the working group of the Department of Econometrics and Business Statistics, Monash University. This pilot study allowed us to estimate the study's completion time and the effect size and fine-tune the application.

These R packages were used for the work: `tidyverse` (Wickham et al. 2019), `lme4` (Bates et al. 2015), `broom.mixed` (Bolker and Robinson 2024), `ggbeeswarm` (Clarke et al. 2023), `emmeans` (Lenth 2025), `patchwork` (Pedersen 2024), `colorspace` (Zeileis et al. 2020), `kableExtra` (Zhu 2024), `conflicted` (Wickham 2023), `Rtsne` (Krijthe 2015), `umap` (Konopka 2023), `phateR`(Moon et al. 2019), `reticulate` (Ushey et al. 2024), `langevitour` (Harrison 2023), `binom` (Dorai-Raj 2022), `gridExtra` (Auguie 2017), `shiny` (Chang et al. 2025), `shinydashboard` (Chang and Borges Ribeiro 2025), `shinythemes` (Chang 2021), `bslib` (Sievert et al. 2025), `shinyjs` (Attali 2021), `DT` (Xie 2016), `googledrive` (D'Agostino McGowan and Bryan 2025), `googleAuthR` (Edmondson 2024), `googlesheets4` (Bryan 2025), `shinyalert` (Attali and Edwards 2024), `shinypop` (Meyer and Perrier 2024), `randomNames` (Betebenner 2024), `shinyfullscreen` (Bacher 2021), `shinyWidgets` (Perrier et al. 2025), `hms` (Müller 2025), `shinythemes` (Chang 2021), and `shinycssloaders` (Attali and Sali 2024). These Python packages were used for the work: `trimap` (Amid and Warmuth 2019) and `pacmap` (Wang et al. 2021).

Chapter 6

menuraR: An R Shiny App to Help Select the Best Nonlinear Dimension Reduction Representation

Nonlinear dimension reduction (NLDR) methods such as tSNE and UMAP are widely used to visualize high-dimensional biological data, including single-cell RNA-seq and genomics datasets in two dimensions. However, choosing an appropriate method and tuning hyper-parameters typically requires iterative experimentation and expert knowledge. Existing tools offer limited support for systematically comparing multiple NLDR layouts or diagnosing how well each layout reflects the underlying high-dimensional structure. We present `menuraR`, an interactive Shiny web application for evaluating and comparing multiple NLDR layouts quantitatively and qualitatively. Built on the `quollr` package, `menuraR` provides a graphical user interface for generating, visualizing, and diagnosing NLDR representations without programming. Users can compare multiple layouts, assess representation using the hexbin error (HBE), and view the model fitted in high dimensions. Linked brushing helps to investigate where the NLDR model has challenges representing the high-dimensional data. An example workflow using a PBMC single-cell dataset demonstrates how `menuraR` supports more informed, transparent, and reproducible analysis of high-dimensional biological data.

6.1 Introduction

Nonlinear dimension reduction (NLDR) methods such as tSNE (Maaten and Hinton 2008) and UMAP (McInnes et al. 2018) have become essential tools for exploring and visualizing high-dimensional

data across diverse scientific disciplines. These techniques enable researchers to uncover structures, clusters, and patterns that are not immediately visible in the original feature space. However, the flexibility and power of these methods come with challenges: the quality and interpretability of low-dimensional embeddings are often highly sensitive to hyper-parameter choices, random initialization, and characteristics of the underlying data. As a result, identifying the most meaningful and faithful representation typically requires iterative experimentation, systematic evaluation, and domain expertise.

To address these challenges, we introduce `menuraR` (**m**onitoring **e**mbeddings of **n**onlinear **u**nfoldings for **r**epresentation and **a**nalysis in **R**), an interactive Shiny application created to facilitate the evaluation of NLDR layouts. Building on the functionality of the `quollr` package (Gamage et al. 2025a), `menuraR` provides a graphical user interface that enables users to compare multiple NLDR layouts, explore the effects of different hyper-parameter settings, and apply diagnostic tools for evaluating NLDR layout(s). The `quollr` package is useful for understanding how NLDR warps high-dimensional space and fits the data. Starting from a two-dimensional NLDR layout, `quollr` constructs a wireframe representation that is lifted back into the high dimensions (see Gamage et al. (2025c) for algorithmic details) and viewed using a tour (Asimov (1985), a continuous sequence of linear projections). This model-based view helps reveal how NLDR methods warp high-dimensional geometry, where the embedding fits the data well, and where distortions or mismatches occur.

These capabilities are delivered through an intuitive interface that eliminates the need for programming, thereby lowering the technical barrier for users.

A key advantage of `menuraR` is its accessibility. The application is fully web-based and does not require a local installation of R or package management. Centralized hosting ensures that users always access the most up-to-date version, while reproducibility is supported through logging and open availability of the underlying code. In this way, `menuraR` enhances transparency in NLDR evaluation and fosters broader adoption of rigorous visualization practices.

This chapter introduces `menuraR`, describing its implementation, core features, and intended use cases. We demonstrate how the application can inform NLDR choices, highlight key visual diagnostics, and support exploratory data analysis and teaching.

6.2 User-informed design

To ensure `menuraR` is intuitive and practical, we conducted a usability study with members of the Business Analytics research group at Monash University (NUMBATs). The goal was to observe how

users interact with the app, identify confusing aspects, and gather suggestions for improvement.

We provided two slightly different run sheets for two groups of participants:

1. **Generate default layouts group:** Participants in this group were instructed to choose *"Generate default tSNE and UMAP layouts"* as the source of NLDR layouts.
2. **Upload own layouts group:** Participants in this group were instructed to *"Upload your own NLDR data"*, and we provided the necessary metadata and precomputed NLDR layouts for them to upload.

Both groups were asked to complete tasks that simulate real-world usage: uploading high-dimensional data, generating or uploading NLDR layouts, comparing embeddings using Hexbin Error (HBE), exploring model diagnostics, and downloading results. They also recorded which layouts they used, the binwidth (a_1) they selected, the layouts suggested as "best" by the app, and whether they agreed with that suggestion. Background information, such as experience with PCA or NLDR methods and subject area, was also collected.

The feedback we received led to several key improvements:

- **Data Upload Tab Layout and Numbering:** Initially, all upload tiles were in one column, with "Add more layout" and "Ready to Analyze?" in a separate column, which confused participants. Additionally, the numbering of steps was inconsistent: in the default-layout group, there was no "step 3", which participants found confusing. We rearranged the tiles and renumbered the steps for a more logical and consistent workflow.
- **Displaying Uploaded NLDR Layouts:** When users uploaded NLDR layouts, the titles' showing method and hyper-parameters were being cut off. This was fixed so that all layout titles are fully visible, improving clarity.
- **Understanding Binwidth** (a_1): Users were confused about how changing the binwidth affected the analysis. Previously, only the NLDR layout was drawn without any overlay, so changes to a_1 were not visually clear. We added a hexagon grid overlay on the NLDR layout, making the effect of the binwidth immediately visible.
- **Seeing the Model in High Dimensions:** Some participants did not understand how the 2-*D* layout relates to the high-dimensional model. To address this, we added a 2-*D* wireframe step, which allows users to see the underlying structure before lifting it into high dimensions.

- **Navigating to Model Diagnostics:** Model Diagnostics was originally accessible only from the sidebar, so users did not naturally explore it from the Compare NLDR Layouts tab. We added a small redirect tile to guide users directly to the diagnostics tab.

- **HBE vs Binwidth Plot:** Participants were confused when the full HBE vs binwidth plot was redrawn each time they selected a new a_1. Ideally, we would have liked to show a fixed plot with just a vertical line indicating the chosen binwidth, but this was not feasible due to the way Shiny handles reactivity combined with the loading spinner (`withSpinner`). The plot is generated dynamically from the computed HBE values for the selected layouts, and separating the vertical line from the main plot would require a substantial rewrite of the reactive logic, potentially slowing the app for larger datasets and complicating maintenance. Therefore, the full plot is redrawn each time to ensure the visualization is accurate and the app remains stable and responsive, even though this behavior may appear confusing to users.

Overall, the study confirmed that `menuraR` significantly improves the process of comparing and selecting NLDR layouts, especially for users without programming experience. With clearer workflow, more informative visualizations, and better interactivity, the app is now more intuitive and practical for both research and teaching.

This paper presents `menuraR` version 1.0.2, which incorporates all these user-informed changes and reflects the latest improvements from the usability study.

6.3 Methods

The `menuraR` application is implemented in R using the `shiny` package (Chang et al. 2025), which provides the reactive framework required for interactive web applications. Supporting packages, including `shinycssloaders` (Attali and Sali 2024), are used to indicate progress during computationally intensive tasks.

The application enables users to generate and compare two-dimensional NLDR layouts in high-dimensional space. Users can either upload their own pre-computed NLDR layouts or compute layouts using tSNE (Krijthe 2015) and UMAP (Melville 2025) as part of the application. Core computations, including layout generation, model fitting, and diagnostic evaluation, are handled by the `quollr` package (Gamage et al. 2025a). This includes construction of two-dimensional wireframe representations, lifting these structures into the original high-dimensional space, and computing the hexbin error (HBE) across a range of binwidths.

`menuraR` is deployed on the `shinyapps.io` (RStudio, PBC n.d.) platform, allowing users to access the application through a web browser without local installation or dependency management. This provides a consistent environment for users within a given deployment, while long-term reproducibility is supported through version-controlled code and documented workflows.

The combination of an interactive Shiny interface with the `quollr` back end allows users to explore multiple embeddings, assess hyper-parameter effects, and examine diagnostic measures within a single workflow, without requiring programming expertise.

6.4 The Shiny application

The `menuraR` app contains three main tabs: (1) Data Upload, (2) Compare NLDR Layouts, and (3) Model diagnostics. Each tab includes numbered steps and clear instructions that guide users from data input to interpretation of results.

6.4.1 Data upload

Analysis in `menuraR` begins in two ways: by uploading user-provided high-dimensional data or by using one of the built-in example datasets (Figure 7.3). Two datasets are provided within the application: C-shaped Clusters, a synthetic dataset illustrating nonlinear structure, and PBMC, a biological single-cell dataset for real-world exploration (Satija et al. 2025). If the user uploads their own high-dimensional data, the file should be a CSV and the CSV must have a unique ID column, with data columns prefixed by the letter `x` (e.g., `x1`, `x2`, etc.).

Once the high-dimensional data is uploaded, under “Choose the source of NLDR layouts”, users select an NLDR layout source: “Upload your own NLDR data”, or “Generate default tSNE and UMAP layouts”. Selecting “Upload your own NLDR data” activates the uploaded NLDR layouts and metadata for comparison. Precomputed NLDR layouts are uploaded as a CSV file. For each layout, the two embedding dimensions are labeled `emb1` and `emb2`. If multiple layouts are included, embedding columns are prefixed with the layout number (e.g., `1_emb1`, `1_emb2`). Also, the metadata CSV file includes the NLDR layout name (e.g., 1, 2, etc.), the method used (like UMAP or tSNE), and any hyper-parameters formatted with the parameter name followed by its value, separated by a dash (e.g., perplexity-30 for tSNE). All uploaded files must be under 100 MB in size, and it is essential that each dataset follows the variable naming conventions required by the web application. Alternatively, users may choose “Generate default tSNE and UMAP layouts”, in which case the application automatically computes two embeddings using default hyper-parameter settings for tSNE and UMAP.

Once loaded, all available NLDR layouts appear in the "Your Loaded NLDR Layouts" box. Users can select or deselect specific layouts to include in the comparison.

Adding additional layouts

The application also allows users to generate additional layouts directly within the interface. Users select the NLDR method (tSNE or UMAP), specify hyper-parameters, and click "Show Layout" to generate the embedding. If satisfied, they can add it to the comparison using "Add Layout"; otherwise, they may adjust the parameters and regenerate the layout. Multiple additional layouts can be created and compared in this manner.

Once the desired layouts are finalized, users click "Start Analysis" to proceed automatically to the next tab, Compare NLDR Layouts, where the evaluation and comparison of embeddings take place.

6.4.2 Compare NLDR Layouts

The comparison begins by selecting the binwidth (a_1), which controls the width of the hexagons in the hexagonal grid (Figure 7.4). For the chosen binwidth, a_1, the Shiny application visualizes hexagonal grids overlaid on each selected 2-D NLDR layout. Also, the app constructs a 2-D wireframe representation for each layout, which forms the basis for subsequently lifting the model into high-dimensional space. The app also generates a plot showing the Hexbin Error (HBE) against the binwidth parameter (a_1) and identifies the "best" representation that yields the lowest HBE for that specific a_1. Users can modify the a_1 value to see what layout performs best for the chosen bandwidth.

Furthermore, users have the option to download the 2-D layouts, corresponding data, the HBE versus binwidth plot, and the summary table, which contains error, HBE, the number of bins along the x-axis (b_1), the number of bins along the y-axis (b_2), the total number of bins (b), the number of non-empty bins (m), the binwidth (a_1), the bin height (a_2), standardized bin counts (w_h), and NLDR method id.

6.4.3 Model diagnostics

Once the best representation is selected, interactive plots are generated to display the high-dimensional model error, the best 2-D layout, and a tour view of the model overlaying the high-dimensional data (Figure 6.3). This interactivity allows users to identify where the model fits well, where it is better in some areas, and where it fails to match the data. Importantly, model diagnostics are not limited to the best NLDR layout; other layouts can also be selected and examined for comparison.

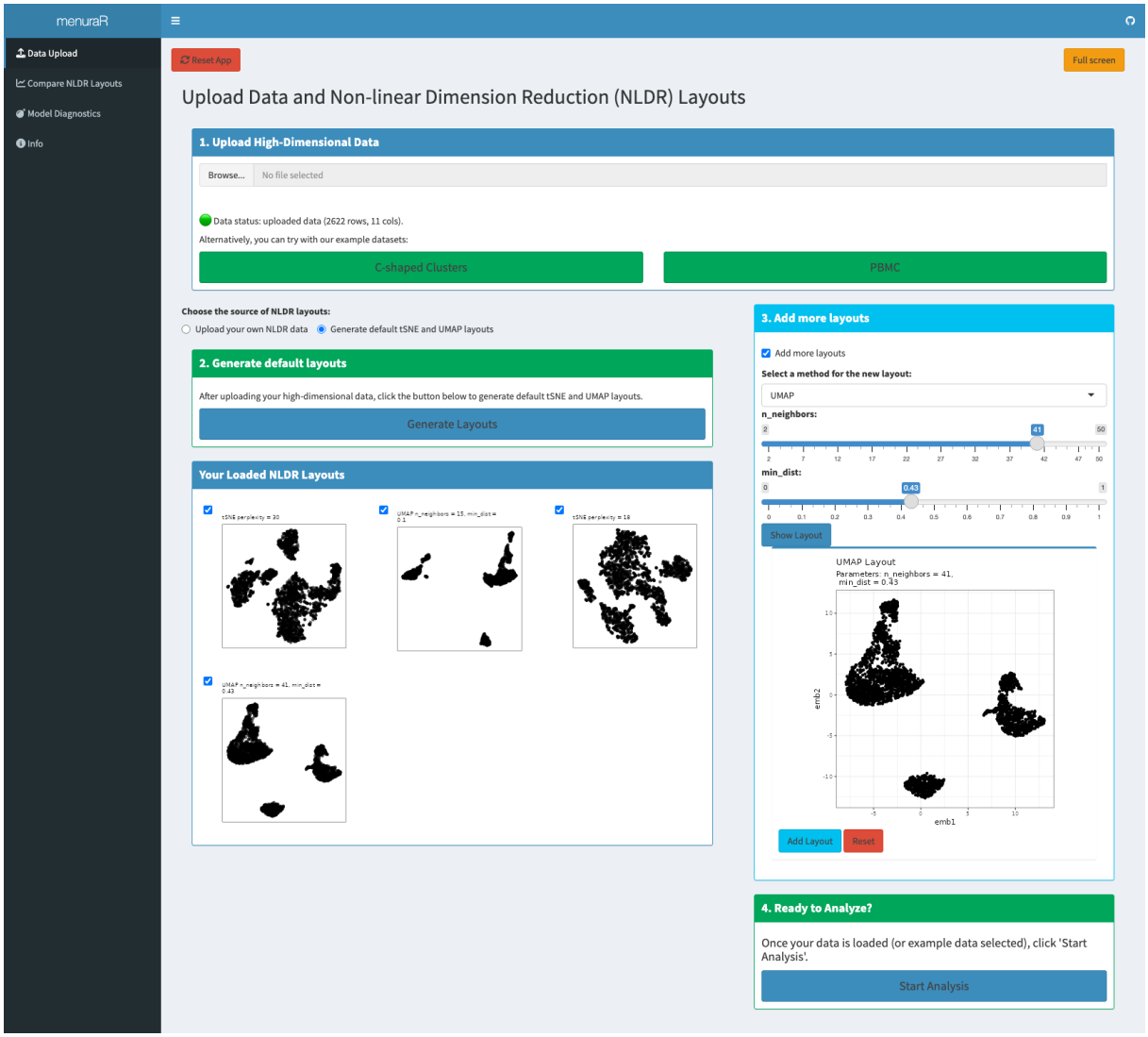

Figure 6.1: *Data Upload and NLDR layout Configuration in* `menuraR`*. The* Data Input *tab enables users to upload high-dimensional datasets or use built-in examples, and generate or upload NLDR layouts. Users can create additional layouts with custom hyper-parameters, and manage loaded embeddings for downstream comparison.*

6.5 Example workflow

We evaluated `menuraR` using the PBMC3k single-cell RNA-seq dataset (Satija et al. 2025), a widely used benchmark for assessing dimension reduction methods in single-cell analysis. This dataset contains 2622 human peripheral blood mononuclear cells (PBMCs) measured across 1000 gene expression variables and is commonly used to study cellular heterogeneity and population structure at the single-cell level.

In single-cell RNA-seq analysis, clustering is typically used to identify groups of cells with similar expression profiles, while nonlinear dimension reduction (NLDR) methods are employed to summarize and visualize this structure in two dimensions. Importantly, NLDR methods do not use cluster labels

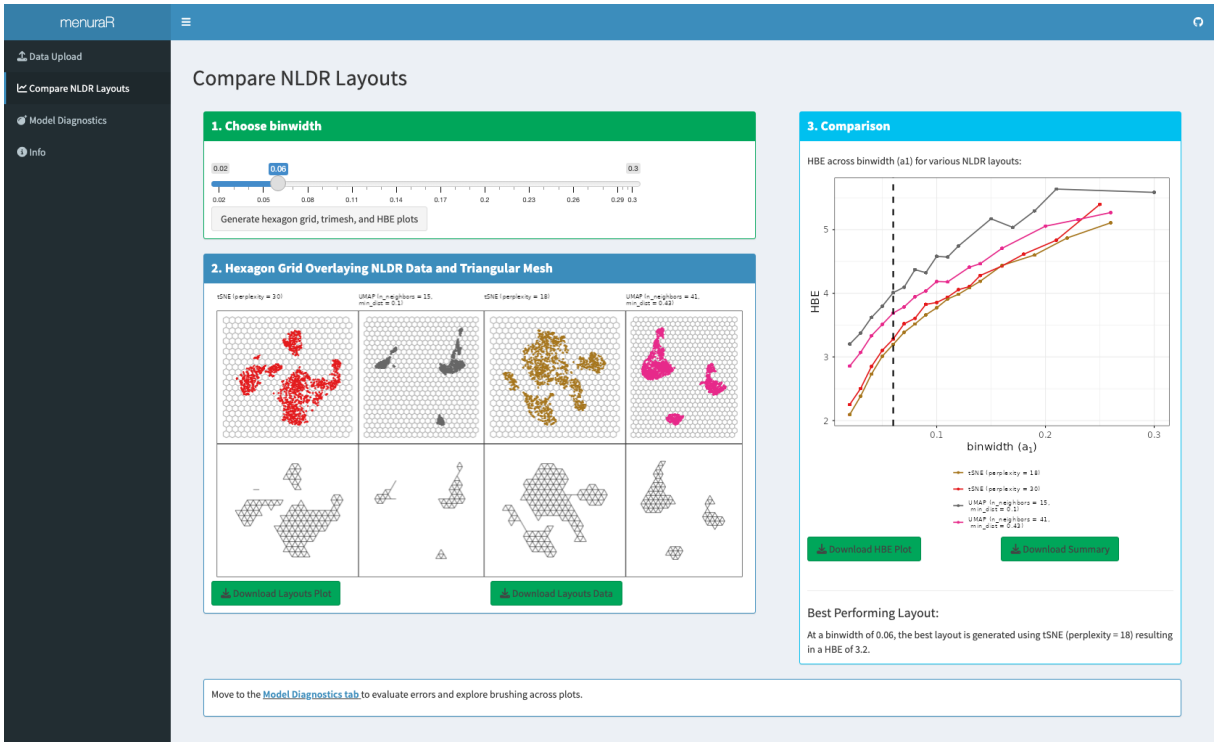

Figure 6.2: *NLDR Layout Comparison and Hexbin Error Evaluation in* `menuraR`*. The Compare NLDR Layouts tab allows users to visualize selected* 2*-D NLDR embeddings overlaid with hexagonal grids and wireframe representations. Users can explore the effect of the binwidth parameter* (a_1) *on Hexbin Error (HBE), identify the most reasonable layout, and download layouts, HBE plots, and summary tables for further analysis.*

to compute embeddings; labels are instead used post hoc for interpretation and visualization.

We applied NLDR methods to the first nine principal components of the gene expression matrix. Using the *Compare NLDR Layouts* tab of `menuraR`, we generated four embeddings commonly used in practice: tSNE with perplexity values of 30 (default) and 18, and UMAP with (n_neighbors, min_dist) set to $(15, 0.1)$ (default) and $(41, 0.43)$. Visual comparison showed consistent separation of major immune cell populations across all layouts, with clear differences in cluster separation and neighborhood continuity. Overall, tSNE produced smaller inter-cluster separation, while UMAP yielded more distinct clusters. For both methods, hyper-parameter choices controlled the local–global trade-off: smaller neighborhood sizes or lower perplexity emphasized tight local groupings, whereas larger neighborhood sizes or higher perplexity produced smoother global transitions (Figure 7.4).

The *Comparison* panel enabled a quantitative comparison of these layouts using the hexbin error (HBE). At a binwidth of $a_1 = 0.06$, the tSNE layout with perplexity $= 18$ achieved the lowest HBE, indicating the best agreement between the two-dimensional embedding and the fitted high-dimensional model at this binwidth.

Linked brushing in the *Model Diagnostics* tab showed that the model fits the data well and highlighted

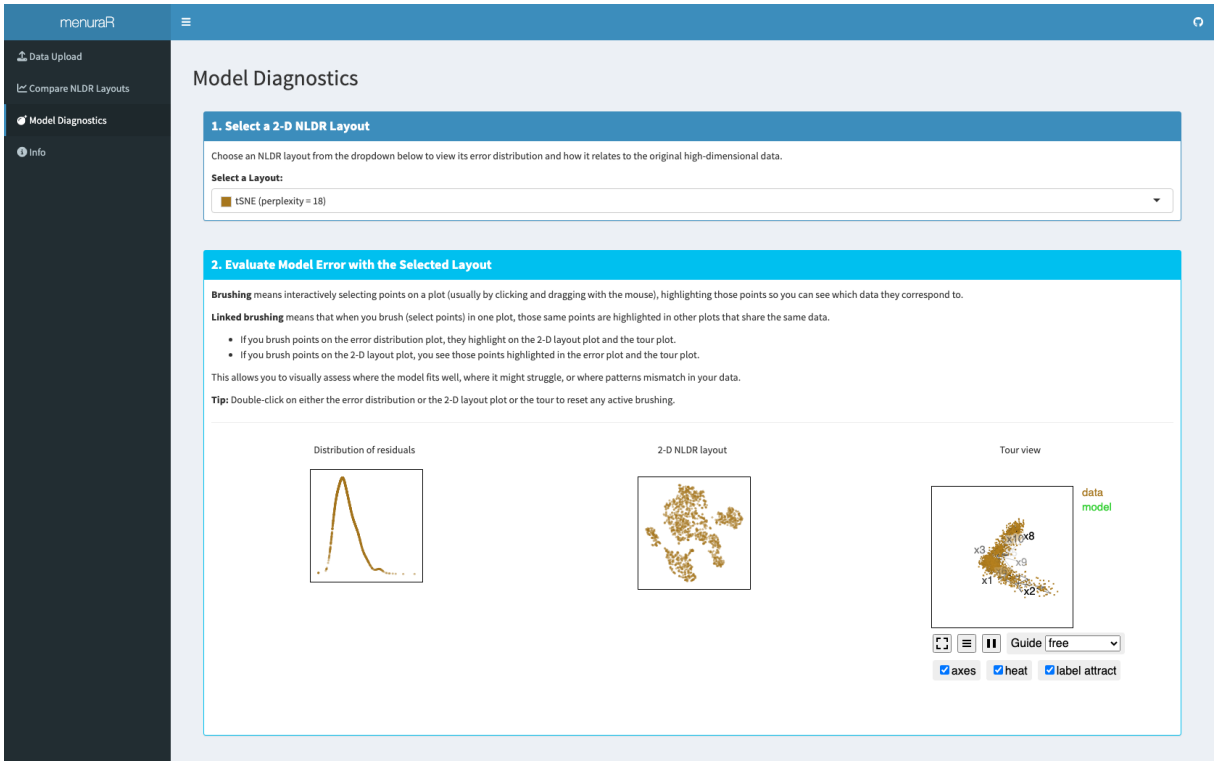

Figure 6.3: *Interactive Model Diagnostics in `menuraR`. The interface provides interactivity between the distribution of model error, the NLDR layout, and the model overlaid on the high-dimensional data. The best 2-D layout is automatically selected, but users can explore other layouts as well. This helps to identify regions where the model fits well or exhibits distortions and some quirks.*

filled-out and dense clusters that are not visible from the NLDR layout (Figure 6.3).

6.6 Conclusions

This chapter introduces `menuraR`, a web-based interface designed to assist in the evaluation and selection of the most reasonable NLDR layout(s). Although NLDR methods such as tSNE and UMAP are widely used for visualizing high-dimensional data, interpreting and selecting the most representative layout can be complex. The `menuraR` application addresses this challenge by providing an accessible, intuitive, and interactive environment that encapsulates the diagnostic features of the `quollr` package, making NLDR selection feasible for users with varying levels of technical expertise.

Developed using the R Shiny framework, `menuraR` eliminates many of the technical barriers traditionally associated with advanced statistical software. Users do not need to install additional packages or configure language-specific environments, which is particularly valuable for interdisciplinary research teams and educational settings. The platform helps users to compare NLDR layouts, select the ones that most accurately represent the high-dimensional data structure, and assess NLDR results.

Currently, `menuraR` supports only two NLDR methods: tSNE and UMAP for computing additional layouts within the app. Performance may vary depending on dataset size and browser memory limits, as all computations are handled server-side. Users working with very large datasets may experience slower response times, and expanding support to other NLDR methods is a potential direction for future development.

6.7 Supplementary materials

The `menuraR` application is available online at https://ebsmonash.shinyapps.io/menuraR/. All data and materials used in the study are openly available. The survey data collected from participants can be accessed at https://github.com/JayaniLakshika/Monash_PhD_thesis/blob/main/data/menuraR/usability_study_data.csv. The example datasets provided within the menuraR app are available at https://github.com/JayaniLakshika/Monash_PhD_thesis/tree/main/data/menuraR.

Additionally, the run sheets used to guide participants through the usability study are publicly available at https://github.com/JayaniLakshika/Monash_PhD_thesis/tree/main/scripts/menuraR/run_sheets.

6.8 Acknowledgments

We thank members of NUMBATs, the working group of the Department of Econometrics and Business Statistics, Monash University, Australia, for their participation in the usability survey and for providing valuable feedback that helped improve this research.

Chapter 7

Conclusion and Future Plans

7.1 Contributions

The primary contributions of this research are fivefold. We introduced a novel method for visualizing how NLDR warps data, thereby improving the diagnostics of NLDR techniques. This methodology is available on an R package, `quollr`, which implements the proposed diagnostic method. The `cardinalR` package generates high-dimensional clustering data with a variety of cluster shapes and enhanced features, such as added noise and background noise. A human subjects experiment showed that different methods create a systematically different conceptualization of the same cluster structure from each other and from what would be imagined from tour views. Finally, this methodology has been made available in a user-friendly web interface. Overall, this work supports better exploratory data analysis and visualization of high-dimensional data.

7.2 How the chapters fit together

In the Introduction, we showed a **published UMAP layout of a human PBMC CITE-seq dataset** (Hao et al. 2021) (Figure 1.1) as a motivating example. This layout shows multiple clusters with distinct shapes: some appear compact and well-separated, while others are elongated, curved, or partially overlapping. In total, six clusters can be observed, including three with nonlinear shapes, two roughly Gaussian clusters, and one elliptical cluster, along with a small amount of background noise. We asked the question: *Does it faithfully represent the structure of the underlying* 10*-D PBMC CITE-seq data?* Here is how we can check that given the various contributions of this thesis.

Figure 7.1 shows the data using the tour. It also suggests that there are six clusters, and they are fairly close to one another, with three nonlinear-shaped clusters, two Gaussian-like blobs, and one closer

to an ellipse, alongside some scattered background points. The full dataset contains approximately 160,000 observations, so for computational efficiency for visualization purposes, we display a random subset of 10,000 points. Such cluster arrangements are commonly seen in other bioinformatics data.

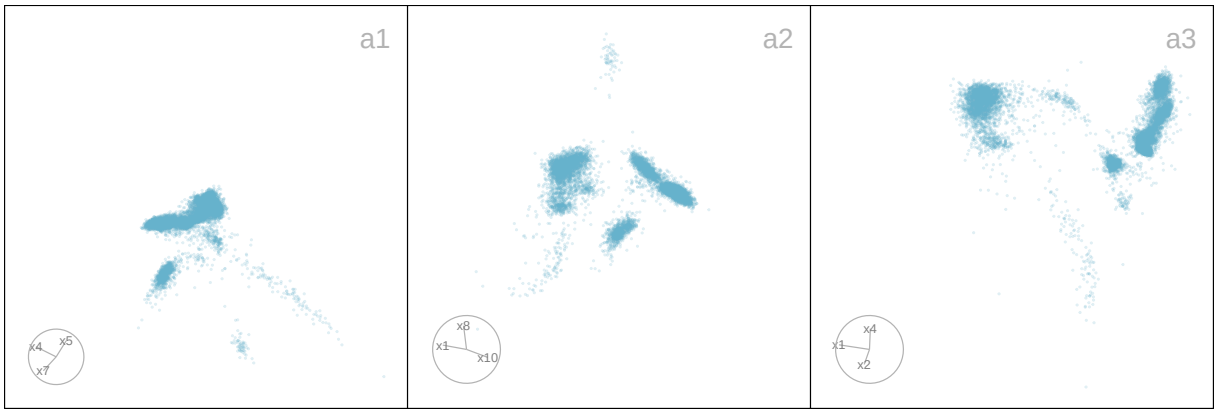

Figure 7.1: *Three 2-D projections of the 10-D PBMC CITE-seq data produced using a tour. The dataset shows six well-separated yet closely positioned clusters, including three with nonlinear geometric structures and three approximately Gaussian clusters, along with a small amount of background noise.*

Using the `quollr` framework, we evaluated how well the UMAP layout reflects the structure of the 10-*D* PBMC CITE-seq data (Figure 7.2). With a model fitted using a binwidth of 0.03, the layout appears reasonable overall, but some limitations become clear. In particular, the roughly Gaussian clusters look more squeezed than expected, and background noise seems to form a separate cluster that likely does not represent a true group in the data. In addition, the nonlinear shaped clusters could benefit from being more spread out to better reflect their underlying structure. Also, clusters should be closer. These observations suggest that, while the current layout is informative, there is good potential to find an alternative layout that represents the data structure even more clearly.

This motivates the comparison of multiple NLDR layouts, rather than relying on a single embedding. The Shiny app `menuraR` makes this comparison easier by allowing different layouts and parameter settings.

Rather than computing embeddings on the fly, it is also helpful to precompute the NLDR layouts. In this case, four layouts are of interest:

- the published UMAP layout (*n-neighbors = 30, min_dist = 0.3*),
- a tSNE layout (*perplexity = 84*),
- a TriMAP layout (*n-inliers = 12, n-outliers = 4, n-random = 3*), and
- a PaCMAP layout (*n-neighbors = 10, init = random, MN-ratio = 0.5, FP-ratio = 2*).

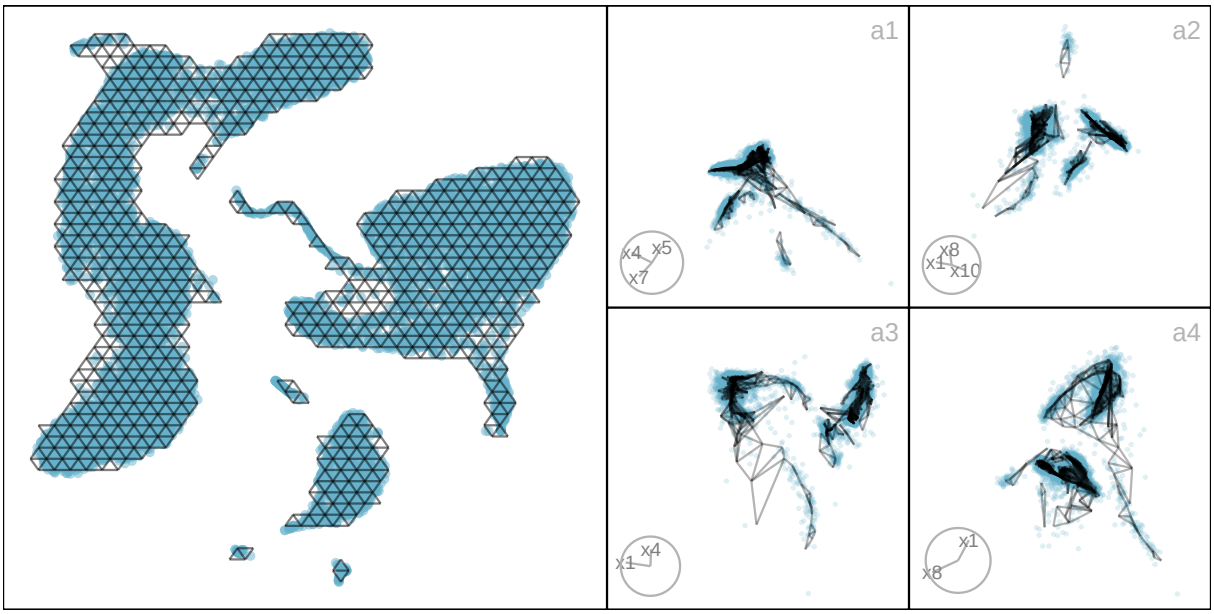

Figure 7.2: *2-D wireframe model representation of the UMAP layout, lifted and displayed in 10-D space. The left panel shows the UMAP layout with a triangular mesh overlay forming the wireframe structure. This mesh is lifted into higher dimensions and projected to examine how the geometric structure of the data is preserved. Panels (a1–a4) show different 2-D projections of the lifted wireframe. The model fits well, and varying densities along the nonlinear-shaped clusters are also observable.*

These layouts can be saved as a single CSV file, following `menuraR`'s naming conventions (`emb1`, `emb2`, or prefixed versions for multiple layouts), along with a small metadata file describing the method and hyper-parameters used. Uploading precomputed layouts avoids long computation times and makes it easy to focus on comparison rather than setup.

Once the data and layouts are loaded in the *Data Upload* tab, all three embeddings appear in the "Your Loaded NLDR Layouts" panel (Figure 7.3). From there, they can be selected together and passed into the *Compare NLDR Layouts* tab.

This allows the layouts to be viewed side by side, overlaid with hexagonal grids, and evaluated using the hexbin error across different bin widths. With `menuraR`, at a binwidth of 0.03, the most reasonable layout is tSNE with $perplexity = 84$ (Figure 7.4).

The TriMAP layout is universally poor. The tSNE layout with little separation performs well at tiny binwidth (where most points are in their own bin) and poorly as binwidth increases. Also, the layout shows more clusters than it should. Both UMAP and PaCMAP perform better in this case because they produce more clearly separated clusters. However, PaCMAP spreads the Gaussian clusters too far apart, with some clusters even overlapping, which makes the structure harder to interpret. In contrast, UMAP provides well-separated clusters without introducing these issues, making it the most reasonable choice here. This addresses the question posed in the Introduction: while NLDR layouts

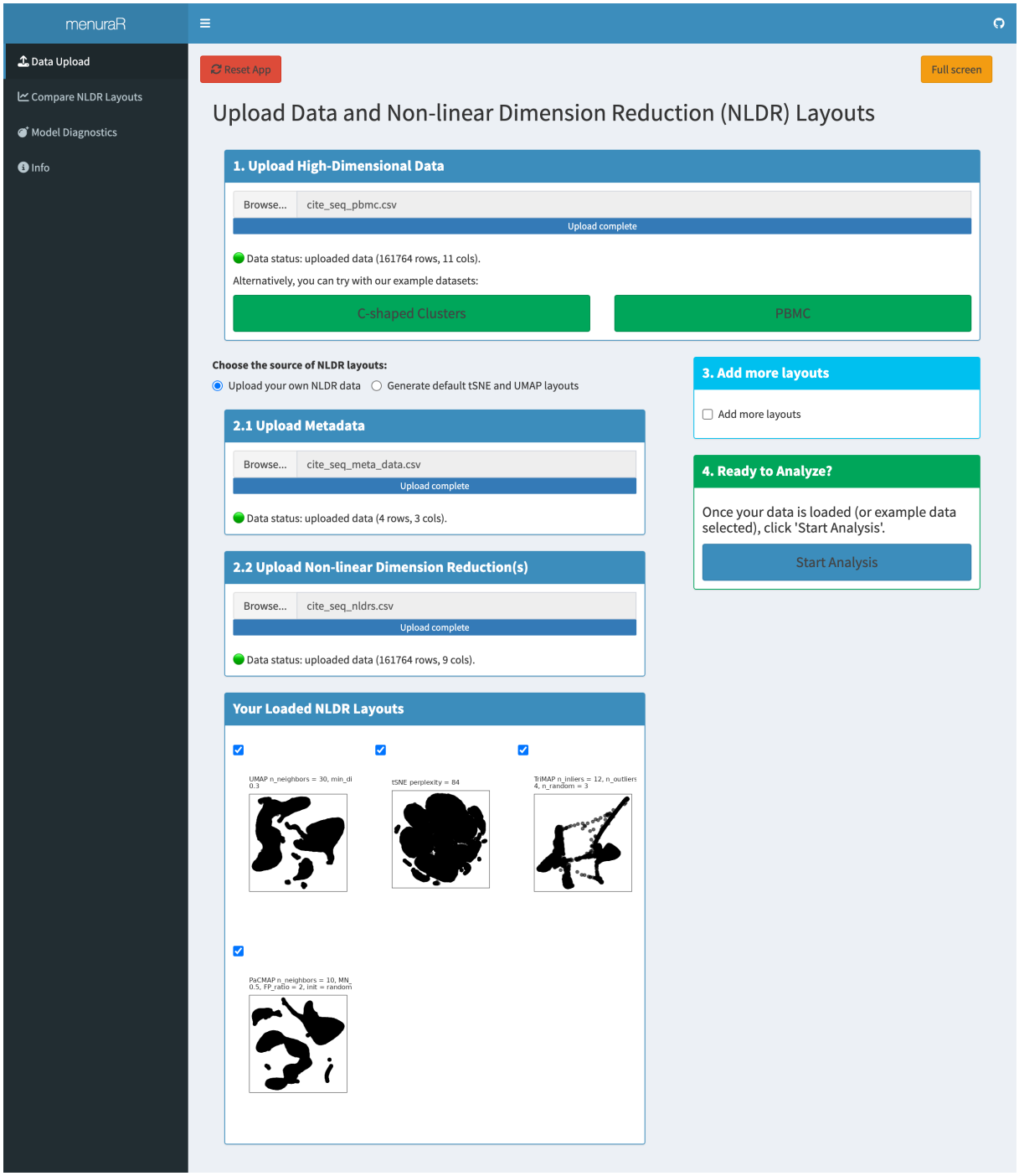

Figure 7.3: *Data upload and NLDR layout configuration in* `menuraR`*. The* Data Upload *tab shows the PBMC CITE-seq dataset together with several precomputed NLDR layouts, including the published UMAP layout and alternative embeddings generated using tSNE, TriMAP, and PaCMAP with different hyper-parameter settings. Uploaded layouts appear in the* Your Loaded NLDR Layouts *panel.*

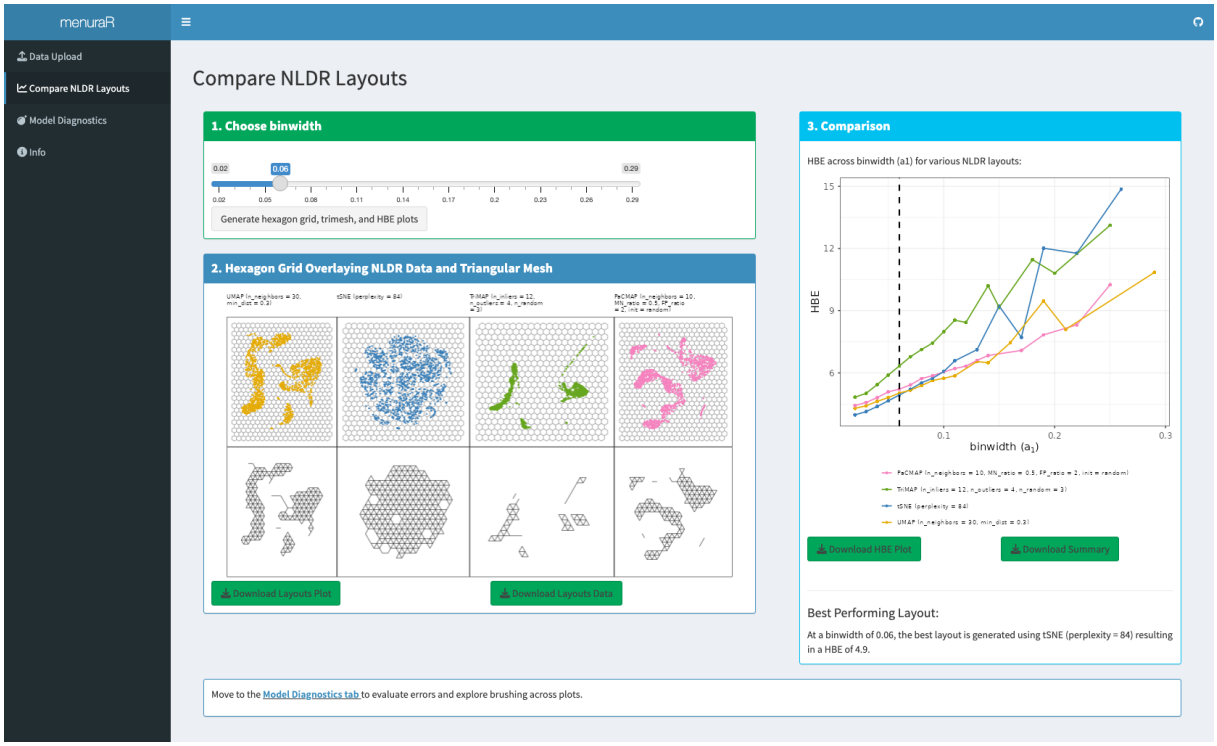

Figure 7.4: *NLDR layout comparison and hexbin error evaluation in* `menuraR`*. The* Compare NLDR Layouts *tab displays multiple 2-D NLDR embeddings side by side, overlaid with hexagonal grids and corresponding wireframe representations. Users can explore how the hexbin error (HBE) changes with the binwidth parameter (a_1) and compare layouts directly. For this example, at a binwidth of* 0.03*, the tSNE layout with* perplexity $=$ 84 *is identified as the most reasonable representation. Layouts, HBE plots, and summary tables can be downloaded for further analysis.*

can distort structure in general, the published UMAP layout provides a faithful and visually reasonable representation of the PBMC CITE-seq dataset.

To better understand how these structures can be represented, we generated a synthetic 10-*D* dataset using the `cardinalR` package (Figure 7.5). The dataset contains six clusters with 500, 700, 700, 500, 700, and 700 points, respectively, for a total of 3,800 points. Cluster centers were positioned according to a custom distance matrix to control their relative arrangement, and each cluster was assigned a distinct geometric shape: three approximately Gaussian clusters, one quadratic cluster, one spherical spiral, and one conic spiral, with additional background noise. This synthetic dataset provides a controlled benchmark for assessing how well NLDR layouts capture cluster structure.

7.3 Future work

This thesis opens up several directions for future work that build directly on its methodological, experimental, and software contributions. These include extending the proposed methods beyond

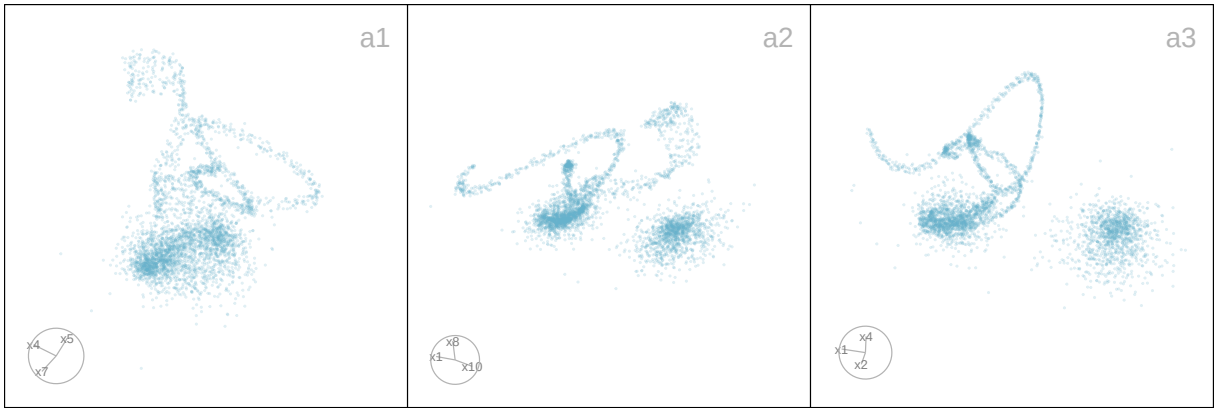

Figure 7.5: *Three 2-D projections of a 10-D simulated dataset generated using* `cardinalR`*. To create a synthetic dataset comparable to PBMC CITE-seq data,* `cardinalR` *first generated a 4-D structure comprising three Gaussian clusters, one quadratic cluster, one spherical spiral, and one conic spiral, with added background noise. Six additional noise dimensions were then added to produce a 10-D dataset retaining similar structural characteristics.*

2-*D* NLDR representations, exploring additional factors that influence how NLDR layouts are perceived, and developing more interactive and diagnostic tools to support evaluation. These directions aim to deepen our understanding of NLDR behavior and improve how these methods are assessed and used in practice.

7.3.1 Extending our algorithm to NLDR representations beyond 2-*D*

A potential direction for future work is extending the current algorithm to NLDR results that project into more than two dimensions. While most existing tools, including those developed in this thesis, focus on 2-*D* embeddings, exploring projections into higher dimensions like 3-*D* or 5-*D* spaces could provide richer structural information in some settings.

Binning into cubes (3-*D* or higher) could be performed relatively easily and used as the basis for constructing a wireframe representation of the fitted model. The algorithm for convex hull computation in p-dimensions, as described by Barber et al. (1996) and implemented in related software (Laurent 2023), serves as inspiration for this approach. Alternatively, a simpler method using k-means clustering to obtain centroids in higher-dimensional embeddings might be feasible; however, the challenge would lie in determining how to connect these centroids to form an appropriate wireframe structure.

7.3.2 Scagnostics to evaluate NLDR

One promising direction for future work is the integration of scagnostics (Dang and Wilkinson 2014; Wilkinson and Wills 2008) as an additional tool for evaluating NLDR results. Scagnostics provide a set of quantitative shape-based metrics (e.g., convexity, skewness, stringiness, clumpiness) that

describe the geometric characteristics of scatterplots. By applying these metrics to 2-*D* scatterplots generated by NLDR methods, we could obtain an objective assessment of how well these methods preserve or distort data structures, particularly in relation to their characteristics (eg: nonlinearity).

Moreover, investigating how scagnostic profiles vary with different sample sizes for specific underlying data structures would provide valuable insight into the stability and robustness of NLDR methods. This could help identify which methods are more resilient to changes in sample size and which structures are more prone to distortion under small sample sizes.

7.3.3 Compare prediction approaches

Future work includes evaluating and comparing the prediction capabilities of different NLDR methods. Only some methods, such as UMAP, provide built-in functionality (Konopka 2023) to project new high-dimensional observations into an existing low-dimensional embedding. Our approach introduces a general prediction framework that can be applied to any NLDR method. It works by identifying the nearest high-dimensional bin centroid for a new observation and assigning its corresponding 2-*D* centroid from the fitted model.

Having predictions from both the built-in functions (when available) and our centroid-based method allows for direct performance comparisons. This enables a systematic evaluation of how well different approaches preserve structure when projecting new observations into an existing NLDR space.

7.3.4 Interactive diagnostic tool for NLDR evaluation

A promising direction for future work is to extend the interactive capabilities developed in this thesis into a more comprehensive diagnostic tool for evaluating NLDR methods, particularly in the context of clustering. In this work, we have already demonstrated the value of interactive visualization—through tours, linked views, and brushing—for understanding how different NLDR layouts align with structure in the original high-dimensional data. Building on these ideas, a natural next step is the development of an interactive diagnostic tool that supports more targeted exploration of where and why NLDR methods succeed or fail.

This functionality could be developed as an extension of the existing menuraR Shiny application, which already provides a framework for exploring NLDR layouts alongside high-dimensional structure. The proposed tool would allow users to upload 2-*D* embeddings, high-dimensional data or distance matrices, and results from spin-and-brush analysis (Cook et al. 2000; Wilhelm et al. 1999), enabling deeper inspection of discrepancies between representations.

Spin-and-brush is a dynamic visual method for exploring clustering structure in high-dimensional numerical data and has proven effective for identifying the influence of nuisance variables, differences in cluster shape or variance, and the presence of low-dimensional manifolds. Support for recording and replaying brushing sequences, as implemented in the `detourr` package (Hart and Wang 2025), aligns closely with the interactive strategies used throughout this thesis and would integrate naturally into the proposed diagnostic workflow.

The envisioned interface would build on linked-view concepts already explored here. For example, users could select a cluster and a specific data point within a 2-*D* NLDR layout and examine how that point relates to others in the same cluster through linked distance-based views. One panel could display the selected cluster and point in the embedding, while a second panel shows the distribution of high-dimensional distances from that point to other cluster members.

Linked brushing between these panels would allow users to directly investigate where NLDR methods preserve local and global structure and where distortions occur. By extending the interactive ideas developed in this thesis, such a tool would support more intuitive diagnosis of NLDR performance and provide a foundation for developing automated evaluation measures that are better aligned with human interpretation.

7.3.5 Visualizing experimental designs

An important practical challenge encountered throughout this thesis is understanding, validating, and diagnosing results from complex experimental designs involving multiple factors and conditions. To support this process, a useful direction for future work is the development of an interactive visualization tool specifically designed to explore and validate experimental designs and their outcomes, including those used in the perceptual studies presented in this thesis.

The main objective of this tool is to visualize and validate experimental results. It includes a web application that allows users to upload experimental design data and corresponding results for visualization. Interactive features such as linked selections and filtering would support exploration of relationships between factors and responses. While the tool primarily targets categorical experimental factors, continuous variables could be transformed into intervals to enable consistent visualization.

The proposed workflow includes importing experimental design and results data, preprocessing, 2-*D* static visualization, 2-*D* interactive visualization, and dynamic visualization. Preprocessing steps involve mapping design variables, identifying missing responses, transforming data to wide format to compute response counts for each factor-level combination (with missing combinations recorded as 0), and converting the data into a long format suitable for visualization. Static plots created

using `ggplot2` (Wickham 2016) provide an overview of response distributions across factor levels, while `plotly` (Sievert 2020) adds interactivity through hover-based detail. Dynamic visualizations represent each factor-level combination as a vertex, with jittered points indicating response counts and edges connecting combinations that differ by a single factor level. This functionality is currently supported through the `detourr` package (Hart and Wang 2025).

7.3.6 Investigating perception and misperception in NLDR with additional factors

One of the contributions of this thesis is the development of a controlled experimental framework for studying how people perceive and interpret NLDR layouts. With this framework in place and shown to work well for measuring perceptual accuracy and misperception, many additional experimental questions can now be explored systematically.

While the current user study focused primarily on how cluster separation influences human perception of NLDR layouts, many additional factors are likely to affect how users conceptualize and interpret low-dimensional embeddings. An important direction for future work is therefore to extend this experimental paradigm to investigate how variations in data characteristics and algorithmic choices influence perception and misperception.

Specifically, we propose extending the perceptual study to consider:

- Background noise: Adding uniformly or normally distributed noise to the data can obscure true structure, and it is important to understand how different NLDR methods handle such interference and how users respond to it visually.
- Number of clusters: As the number of clusters increases, distinguishing them in 2-D may become more challenging, particularly if the separation is subtle or overlaps occur.
- Noise dimensions: Including additional high-dimensional features that contain no signal (i.e., noise variables) can affect NLDR outcomes. We aim to evaluate how this impacts perceived structure.
- Sample size: Varying the number of observations may change both the visual density and the stability of the NLDR projection, influencing the interpretability of patterns.
- Random seed: Since many NLDR methods are stochastic (e.g., tSNE, UMAP), different seeds can lead to different embeddings. It is valuable to understand whether these differences are perceptible to users and how they affect interpretability.

- NLDR hyper-parameters: Algorithm-specific hyper-parameters (such as perplexity in tSNE or the number of neighbors and minimum distance in UMAP) have a strong influence on the resulting embedding. By systematically varying these settings within the experimental framework developed in this thesis, future studies could examine how hyper-parameter choices change the visual appearance of layouts and whether these changes are noticeable or misleading to users.

By extending the study to incorporate these data-driven variables, we can build a more comprehensive understanding of when and why human misperception occurs in NLDR layouts, and which methods are more resilient to such distortions. This work will support the development of more robust diagnostics and improve the practical use of NLDR.

7.3.7 Lineup protocols to evaluate NLDR sensitivity and structure preservation

In this thesis, particularly in Chapter 5, we developed and applied a controlled experimental framework to study how people judge whether a static 2-*D* NLDR layout represents the same underlying high-dimensional data as a tour of linear projections. This task enabled us to quantify perceptual accuracy and misperception, and to examine how these outcomes depend on factors such as cluster separation and the choice of NLDR method.

As discussed in the previous section, this framework can be extended by introducing additional data- and algorithm-related factors that may influence perception. A complementary direction is to modify the experimental task itself, with the goal of probing different aspects of how NLDR layouts are perceived and potentially increasing statistical power. One promising approach in this regard is the use of lineup-based evaluation protocols (Buja et al. 2009).

Lineups were originally introduced as a graphical inference tool for assessing whether visual structure in a plot is stronger than what would be expected under a null model. In a lineup, a plot generated from the true data is randomly embedded among a set of null plots, and observers are asked to identify the plot that appears most different. Successful identification provides evidence that the visual structure is perceptually salient and not attributable to chance.

When combined with the experimental framework developed in this thesis, lineups could be used to evaluate how well a 2-*D* NLDR layout preserves structure from the original high-dimensional data. For example, a lineup could include one NLDR layout computed from the true data alongside several null layouts generated from shuffled data or noise-perturbed versions. If participants consistently identify the true layout, this would suggest that the NLDR method preserves meaningful structure in a way that is perceptually accessible to human viewers.

Lineups could also be used to study the sensitivity of NLDR methods to hyper-parameter choices. Layouts generated under different hyper-parameter settings—such as perplexity in tSNE or the number of neighbors and minimum distance in UMAP—could be embedded within a lineup to assess whether small parameter changes lead to perceptually distinguishable differences. This would allow the robustness and stability of NLDR methods to be evaluated from a human-centered perspective and could help guide more reliable parameter selection.

7.3.8 Comparative perceptual study of PCA and NLDR methods

Another valuable direction for future work is to investigate how PCA compares to NLDR methods in terms of human perception and interpretability. PCA is a linear method widely used for its simplicity and mathematical transparency, whereas NLDR methods often involve nonlinear transformations and hyper-parameter tuning.

By comparing how users interpret and misinterpret PCA layouts versus NLDR-generated layouts, we can gain insights into whether linear techniques are inherently easier to understand or whether they may lead to different types of visual distortions. This work would help clarify when PCA is sufficient for visual analysis and when the added complexity of NLDR is warranted, particularly for exploratory tasks that rely on visual intuition.

7.3.9 Extension for `quollr`

A useful extension to `quollr` would be to link cluster selections between the tour view and the 2-D NLDR layout. This would let users select a cluster in one view and immediately see how it appears in the other, making it easier to compare cluster structure across views.

Chapter 8

Reproducibility and Availability

All materials associated with this thesis are openly available for transparency and following reproducible practice. The thesis is written in Quarto (Allaire and Dervieux 2024) and is available in both **HTML** and **PDF** formats. The **HTML formatted** thesis, which includes interactive and linked plots can be read at https://jayani-lakshika-phd-thesis.netlify.app, and the **PDF formatted** thesis can be downloaded from https://github.com/JayaniLakshika/Monash_PhD_thesis/blob/main/_book/New-Interactive-Visual-Tools-and-Statistical-Methodology-for-Selecting-and-Evaluating-Non-linear-Dimension-Reduction-Layouts-of-High-Dimensional-Data.pdf. All source code, data, and software used to generate the thesis is available on the public GitHub repository at https://github.com/JayaniLakshika/Monash_PhD_thesis.

8.1 Accessibility of figures

To support accessibility, all figures are supplemented with alt text, which allows screen readers and vision-impaired readers to access their content. The `autoAlt` (Maliny Po 2025) package was used as a starting point for generating these descriptions, which were then reviewed and refined to better reflect the content of each figure and its caption.

8.2 Software availability and usage

Some of the software developed has been packaged and is been available on the Comprehensive R Archive Network (CRAN). The R package `quollr`, introduced and used in Chapter 2 and Chapter 3, has been on CRAN since March 2024 and has received 5181 downloads from the CRAN mirror as of

14^{th} January 2026; its development version is hosted on GitHub at https://github.com/jayanilakshika/quollr. The R package `cardinalR`, discussed in Chapter 4, has been available on CRAN since April 2024 and has received 4416 downloads from the CRAN mirror as of 14^{th} January 2026, with the latest development version at https://github.com/jayanilakshika/cardinalR. Figure 8.1 gives an overview of my Git commits to these repositories.

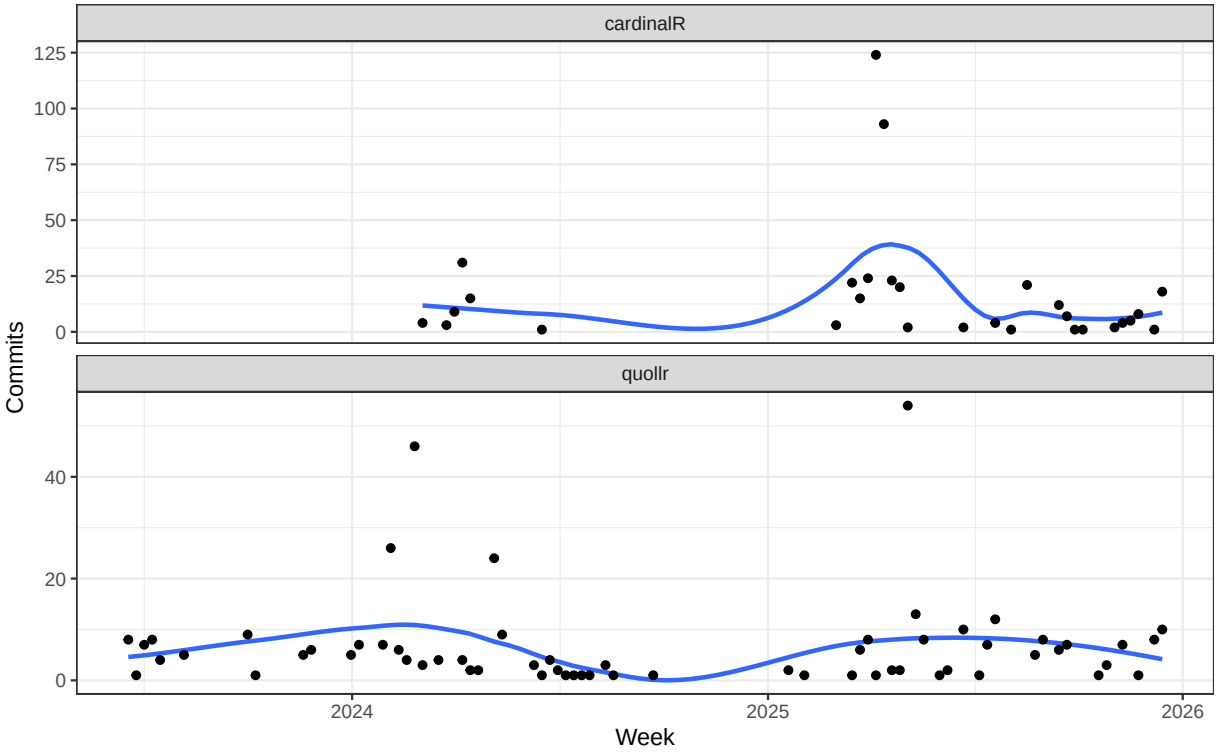

Figure 8.1: *Weekly commit activity for the `cardinalR` and `quollr` packages.*

8.3 Web applications

A Shiny application described in Chapter 6, is accessible via one of the mirror sites at https://menurar.netlify.app/, with its source code available at https://github.com/JayaniLakshika/menuraR. The survey web application, **Match-a-roo** (https://ebsmonash.shinyapps.io/Match-a-roo/), was designed and implemented in `Shiny` to collect the data for the experiment discussed in Chapter 5, participant responses, and demographic information. Each subject accessed the survey through the `shinyapps.io` (RStudio, PBC n.d.) server.

8.4 Supporting R packages

In addition, a number of R packages were essential in the development of this work, including `tidyverse` (Wickham et al. 2019), `ggbeeswarm` (Clarke et al. 2023), `ggrepel` (Slowikowski 2024), `GGally` (Schloerke et al. 2025), `colorspace` (Zeileis et al. 2020), `scales` (Wickham et al. 2025), `patchwork` (Pedersen 2024), `plotly` (Sievert 2020), `crosstalk` (Cheng and Sievert 2025), `htmltools` (Cheng et al. 2024), `quollr` (Gamage et al. 2025a), `cardinalR` (Gamage et al. 2025b), `detourr` (Hart and Wang 2025), `geozoo` (Schloerke 2016), `knitr` (Xie 2015), `kableExtra` (Zhu 2024), `lme4` (Bates et al. 2015), `broom.mixed` (Bolker and Robinson 2024), `emmeans` (Lenth 2025), `mclust` (Scrucca et al. 2023), `fpc` (Hennig 2024), `binom` (Dorai-Raj 2022), `conflicted` (Wickham 2023), `ggforce` (Pedersen 2025), `here` (Müller 2025), `grid` (R Core Team 2025), `gridExtra` (Auguie 2017), and `png` (Urbanek 2022).

8.5 Research workflow and project organization

Presentations, package development, and writing are the three primary types of activities that shape this thesis. Figure 8.2 summarizes my GitHub commits documenting these activities since the start of my PhD, with commits grouped by activity type and annotated with important milestones. It has been a fruitful program.

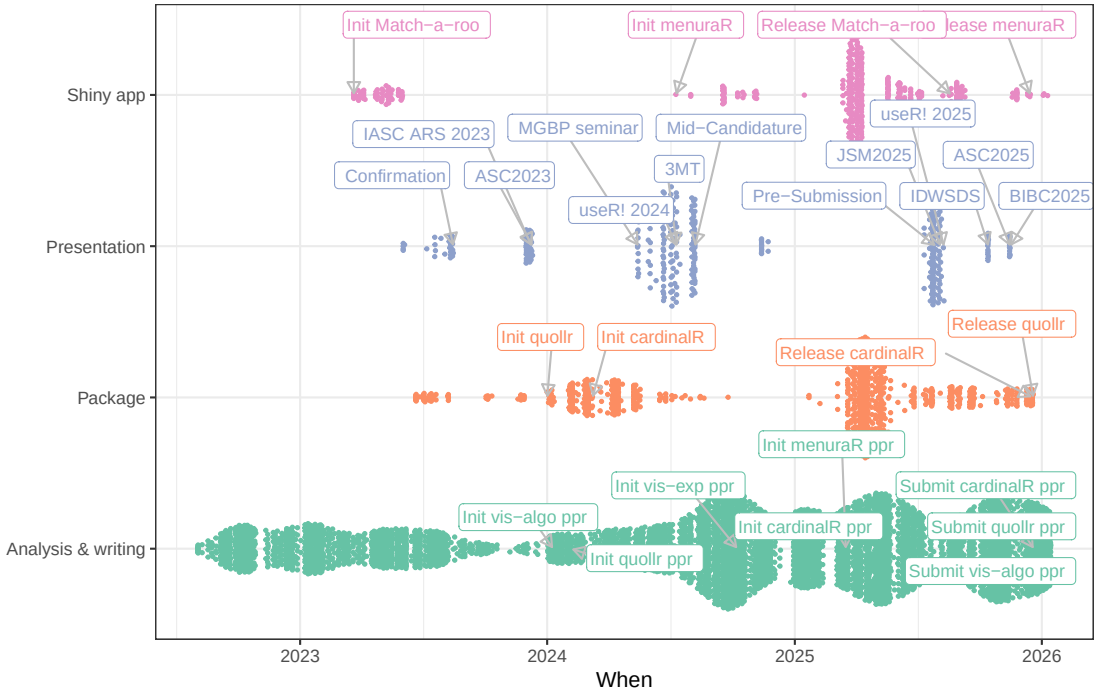

Figure 8.2: *Plots of my Git commits split by the activity types during my PhD years, labeled with some milestones.*

Bibliography

10x Genomics (2016), "3k PBMCs from a Healthy Donor: Universal 3' Gene Expression Dataset," 10x Genomics, Accessed: 2025-06-09.

Agrafiotis, D. K., and Xu, H. (2002), "A self-Organizing Principle for Learning Nonlinear Manifolds," *Proceedings of the National Academy of Sciences*, 99, 15869–15872. https://doi.org/10.1073/pnas.242424399.

Alberts, B., Johnson, A., Lewis, J., Morgan, D., Raff, M., Roberts, K., and Walter, P. (2014), *Molecular Biology of the Cell*, 6th ed., Garland Science.

Allaire, J., and Dervieux, C. (2024), *quarto: R Interface to Quarto Markdown Publishing System*, R package version 1.4.4.

Amid, E., and Warmuth, M. K. (2019), "TriMap: Large-scale Dimensionality Reduction Using Triplets," *ArXiv*, abs/1910.00204.

Andrews, T. S., Kiselev, V. Y., McCarthy, D., and Hemberg, M. (2021), "Tutorial: Guidelines for the Computational Analysis of Single-Cell RNA Sequencing Data," *Nature Protocols*, 16, 1–9.

Arsuaga, J., Vazquez, M., Trigueros, S., Sumners, D. W. L., and Roca, J. (2002), "Characterizing the Entanglement of DNA Molecules," *PNAS*, National Academy of Sciences, 99, 5373–5377. https://doi.org/10.1073/pnas.032095099.

Asimov, D. (1985), "The Grand Tour: A Tool for Viewing Multidimensional Data," *SIAM Journal of Scientific and Statistical Computing*, 6, 128–143. https://doi.org/10.1137/0906011.

Attali, D. (2021), *shinyjs: Easily Improve the User Experience of Your Shiny Apps in Seconds*, R package version 2.1.0.

Attali, D., and Edwards, T. (2024), *shinyalert: Easily Create Pretty Popup Messages (Modals) in 'Shiny'*, R package version 3.1.0.

Attali, D., and Sali, A. (2024), *shinycssloaders: Add Loading Animations to a 'shiny' Output While It's Recalculating*, R package version 1.1.0.

Auguie, B. (2017), *gridExtra: Miscellaneous Functions for "Grid" Graphics*, R package version 2.3.

Bacher, E. (2021), "shinyfullscreen: Display 'HTML' Elements on Full Screen in 'Shiny' Apps," R package version 1.1.0.

Balamuta, J. J. (2024), *surreal: Create Datasets with Hidden Images in Residual Plots*, R package version 0.0.1.

Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. (1996), "The Quickhull Algorithm for Convex Hulls," *ACM Trans. Math. Softw.*, 22, 469–483.

Bates, D., Mächler, M., Bolker, B., and Walker, S. (2015), "Fitting Linear Mixed-Effects Models Using lme4," *Journal of Statistical Software*, 67, 1–48. https://doi.org/10.18637/jss.v067.i01.

Betebenner, D. W. (2024), *randomNames: Function for Generating Random Names and a Dataset*, R package version 1.6-0.0.

Boehmke, B., and Greenwell, B. M. (2019), *Hands-On Machine Learning with R*, 1st ed., Chapman; Hall/CRC. https://doi.org/10.1201/9780367816377.

Bolker, B., and Robinson, D. (2024), *broom.mixed: Tidying Methods for Mixed Models*, R package version 0.2.9.6.

Borg, I., and Groenen, P. J. F. (2005), *Modern Multidimensional Scaling*, 2nd ed., New York, NY: Springer.

Brown, T. A. (2015), *Confirmatory Factor Analysis for Applied Research*, 2nd ed., New York, NY, US: The Guilford Press.

Bryan, J. (2025), *googlesheets4: Access Google Sheets using the Sheets API V4*, R package version 1.1.2.

Buja, A., Cook, D., Hofmann, H., Lawrence, M., Lee, E.-K., Swayne, D. F., and Wickham, H. (2009), "Statistical Inference for Exploratory Data Analysis and Model Diagnostics," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367, 4361–4383.

Calinski, T., and Harabasz, J. (1974), "A Dendrite Method for Cluster Analysis," *Communications in Statistics*, 3, 1–27. https://doi.org/10.1080/03610927408827101.

Calladine, C. R., Drew, H. R., Luisi, B. F., and Travers, A. A. (2004), "Understanding DNA: the Molecule and How It Works," Elsevier.

Carr, D. B., Littlefield, R. J., Nicholson, W. L., and Littlefield, J. S. (1987), "Scatterplot Matrix Techniques for Large N," *Journal of the American Statistical Association*, 82, 424–436.

Carr, D., Lewin-Koh, N., Maechler, M., and Sarkar, D. (2023), *hexbin: Hexagonal Binning Routines*, R package version 1.28.3.

Chang, W. (2021), *shinythemes: Themes for Shiny*, R package version 1.2.0.

Chang, W., and Borges Ribeiro, B. (2025), *shinydashboard: Create Dashboards with 'Shiny'*, R package version 0.7.3.

Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., and Borges, B. (2025), *shiny: Web Application Framework for R*, R package version 1.11.1.

Chari, T., and Pachter, L. (2023), "The Specious Art of Single-Cell Genomics," *PLoS Computational*

Biology, 19, e1011288.

Chen, Z., Wang, C., Huang, S., Shi, Y., and Xi, R. (2024), "Directly Selecting Cell-type Marker Genes for Single-cell Clustering Analyses," *Cell Reports Methods*, 4, 100810.

Cheng, J., and Sievert, C. (2025), *crosstalk: Inter-Widget Interactivity for HTML Widgets*, R package version 1.2.2.

Cheng, J., Sievert, C., Schloerke, B., Chang, W., Xie, Y., and Allen, J. (2024), *htmltools: Tools for HTML*, R package version 0.5.8.1.

Clarke, E., Sherrill-Mix, S., and Dawson, C. (2023), *ggbeeswarm: Categorical Scatter (Violin Point) Plots*, R package version 0.7.2.

Coifman, R. R., Lafon, S., Lee, A. B., Maggioni, M., Nadler, B., Warner, F., and Zucker, S. W. (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, 7426–7431.

Cook, D., Buja, A., Cabrera, J., and Hurley, C. (2000), "Grand Tour and Projection Pursuit," *Journal of Computational and Graphical Statistics*, 4. https://doi.org/10.1080/10618600.1995.10474674.

Csárdi, G. (2025), *cli: Helpers for Developing Command Line Interfaces*, R package version 3.6.4.

D'Agostino McGowan, L., and Bryan, J. (2025), *googledrive: An Interface to Google Drive*, R package version 2.1.2.

Dang, T. N., and Wilkinson, L. (2014), "ScagExplorer: Exploring Scatterplots by Their Scagnostics," in *2014 IEEE Pacific Visualization Symposium*, pp. 73–80. https://doi.org/10.1109/PacificVis.2014.42.

Davies, D. L., and Bouldin, D. W. (1979), "A Cluster Separation Measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1, 224–227. https://doi.org/10.1109/TPAMI.1979.4766909.

Devroye, L. (1986), *Non-Uniform Random Variate Generation*, 1 ed., New York, NY: Springer. https://doi.org/10.1007/978-1-4613-8643-8.

Dorai-Raj, S. (2022), *binom: Binomial Confidence Intervals for Several Parameterizations*, R package version 1.1-1.1.

Edmondson, M. (2024), *googleAuthR: Authenticate and Create Google APIs*, R package version 2.0.2.

Fraley, C., and Raftery, A. E. (2002), "Model-Based Clustering, Discriminant Analysis, and Density Estimation," *Journal of the American Statistical Association*, ASA Website, 97, 611–631.

Freedman, D. A., and Diaconis, P. (1981), "On the Histogram as a Density Estimator: L2 Theory," *Probability Theory and Related Fields*, 57, 453–476.

Gamage, J. P., Cook, D., Harrison, P., Lydeamore, M., and Talagala, T. S. (2025c), "Choosing Better

NLDR Layouts by Evaluating the Model in the High-dimensional Data Space," *arXiv preprint*, arXiv:2506.22051.

Gamage, J. P., Cook, D., Harrison, P., Lydeamore, M., and Talagala, T. S. (2025a), "quollr: An R Package for Visualizing 2-D Models from Nonlinear Dimension Reductions in High-Dimensional Space," *arXiv preprint*, arXiv:2512.18166.

Gamage, J. P., Cook, D., Harrison, P., Lydeamore, M., and Talagala, T. S. (2025b), "cardinalR: Collection of Data Structures," *arXiv preprint*, arXiv:2512.18172.

Gebhardt, A., Bivand, R., and Sinclair, D. (2024), *interp: Interpolation Methods*, R package version 1.1-6.

Genz, A., and Bretz, F. (2009), *Computation of Multivariate Normal and t Probabilities*, Springer-Verlag.

Gershenfeld, N. (2000), "The Physics of Information Technology," Cambridge University Press.

Hadsell, R., Chopra, S., and LeCun, Y. (2006), "Dimensionality Reduction by Learning an Invariant Mapping," pp. 1735–1742. https://doi.org/10.1109/CVPR.2006.100.

Hao, Y., Hao, S., Andersen-Nissen, E., Mauck, W. M., Zheng, S., Butler, A., Lee, M. J., Wilk, A. J., Darby, C., Zager, M., Hoffman, P., Stoeckius, M., Papalexi, E., Mimitou, E. P., Jain, J., Srivastava, A., Stuart, T., Fleming, L. M., Yeung, B., Rogers, A. J., McElrath, J. M., Blish, C. A., Gottardo, R., Smibert, P., and Satija, R. (2021), "Integrated Analysis of Multimodal Single-Cell Data," *Cell*, 184, 3573–3587.e29. https://doi.org/10.1016/j.cell.2021.04.048.

Haque, A., Engel, J., Teichmann, S. A., and Lönnberg, T. (2017), "A Practical Guide to Single-cell RNA-sequencing for Biomedical Research and Clinical Applications," *Genome Medicine*, 9, 75.

Harrison, P. (2023), "langevitour: Smooth Interactive Touring of High Dimensions, Demonstrated With scRNA-Seq Data," *The R Journal*, 15, 206–219.

Hart, C., and Wang, E. (2025), *detourr: Portable and Performant Tour Animations*, R package version 0.2.0.

Hennig, C. (2024), *fpc: Flexible Procedures for Clustering*, R package version 2.2-13.

Henry, L., and Wickham, H. (2024), *tidyselect: Select from a Set of Strings*, R package version 1.2.1.

Hinton, G. E., and Salakhutdinov, R. R. (2006), "Reducing the Dimensionality of Data With Neural Networks," *Science*, American Association for the Advancement of Science, 313, 504–507. https://doi.org/10.1126/science.1127647.

Irizarry, R. (2024), "Biologists, Stop Putting UMAP Plots in Your Papers," Simply Statistics blog, Available at https://simplystatistics.org/posts/2024-12-23-biologists-stop-including-umap-plots-in-your-papers/.

Johnstone, I. M., and Titterington, D. M. (2009), "Statistical Challenges of High-Dimensional Data," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*,

367, 4237–4253.

Jolliffe, I. (2011), "Principal Component Analysis," in *International encyclopedia of statistical science*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1094–1096.

Jöreskog, K. G. (1969), "A General Approach to Confirmatory Maximum Likelihood Factor Analysis," *Psychometrika*, 34, 183–202.

Konopka, T. (2023), *umap: Uniform Manifold Approximation and Projection*, R package version 0.2.10.0.

Krijthe, J. H. (2015), *Rtsne: t-Distributed Stochastic Neighbor Embedding Using Barnes-Hut Implementation*, R package version 0.16.

Kruskal, J. B. (1964), "Nonmetric Multidimensional Scaling: A Numerical Method," *Psychometrika*, 29, 115–129.

Laa, U., Cook, D., and Lee, S. (2022), "Burning Sage: Reversing the Curse of Dimensionality in the Visualization of High-Dimensional Data," *Journal of Computational and Graphical Statistics*, 31, 40–49.

Laurent, S. (2023), *cxhull: Convex Hull*, R package version 0.7.4.

Laurent, S. (2024), "Four-Dimensional Torus Knots," Blog post, Accessed: 2025-11-17, Available at https://laustep.github.io/stlahblog/posts/TorusKnot4D.html.

LeCun, Y., Cortes, C., and Burges, C. J. C. (1998), "The MNIST Database of Handwritten Digits," Accessed: 2025-06-09.

Lee, D. T., and Schachter, B. J. (1980), "Two Algorithms For Constructing a Delaunay Triangulation," *International Journal of Computer & Information Sciences*, 9, 219–242.

Lee, J. A., Peluffo-Ordóñez, D. H., and Verleysen, M. (2015), "Multi-Scale Similarities in Stochastic Neighbour Embedding: Reducing Dimensionality While Preserving Both Local and Global Structure," *Neurocomputing*, 169, 246–261. https://doi.org/10.1016/j.neucom.2014.12.095.

Lee, S., Cook, D., Silva, N. da, Laa, U., Wang, E., Spyrison, N., and Zhang, H. S. (2021), "A Review of the State-of-the-Art on Tours for Dynamic Visualization of High-Dimensional Data," *arXiv preprint*, arXiv:2104.08016.

Leisch, F., and Dimitriadou, E. (2024), *mlbench: Machine Learning Benchmark Problems*, R package version 2.1-6.

Lenth, R. V. (2025), *emmeans: Estimated Marginal Means, aka Least-Squares Means*, R package version 1.11.2-8.

Maaten, L. V. D., and Hinton, G. E. (2008), "Visualizing Data Using t-SNE," *Journal of Machine Learning Research*, 9, 2579–2605.

Maaten, L. van der (2009), "Learning a Parametric Embedding by Preserving Local Structure," in

Artificial intelligence and statistics, PMLR, pp. 384–391.

Maaten, L. van der, Postma, E., and Herik, H. (2007), "Dimensionality Reduction: A Comparative Review," *Journal of Machine Learning Research - JMLR*, 10.

Maliny Po, J. V., Dianne Cook (2025), *autoAlt: Automatic Alt Text Generation*, R package version 0.1.0.

Mandelbrot, B. B. (1983), "The Fractal Geometry of Nature," *Earth Surface Processes and Landforms*, 8, 406–406. https://doi.org/10.1002/esp.3290080415.

McCulloch, C. E., Searle, S. R., and Neuhaus, J. M. (2001), *Generalized, Linear, and Mixed Models*, 2nd ed., Wiley Online Library. https://doi.org/10.1002/0471722073.

McInnes, L., Healy, J., Saul, N., and Großberger, L. (2018), "UMAP: Uniform Manifold Approximation and Projection," *Journal of Open Source Software*, 3, 861.

McLachlan, G. J., and Peel, D. (2000), "Finite Mixture Models," in *Wiley Series in Probability and Statistics*.

Melville, J. (2025), *snedata: SNE Simulation Dataset Functions*, R package version 0.0.0.9001.

Meyer, D., and Buchta, C. (2022), *proxy: Distance and Similarity Measures*, R package version 0.4-27.

Meyer, F., and Perrier, V. (2024), *shinypop: Collection of Notifications for 'Shiny'Applications*, R package version 0.1.1.

Moon, K. R., Dijk, D. van, Wang, Z., Gigante, S. A., Burkhardt, D. B., Chen, W. S., Yim, K., Elzen, A. van den, Hirn, M. J., Coifman, R. R., Ivanova, N. B., Wolf, G., and Krishnaswamy, S. (2019), "Visualizing Structure and Transitions in High-Dimensional Biological Data," *Nature Biotechnology*, 37, 1482–1492.

Müller, K. (2025), "here: A Simpler Way to Find Your Files," R package version 1.0.2, https://github.com/r-lib/here.

Müller, K., and Wickham, H. (2023), *tibble: Simple Data Frames*, R package version 3.2.1.

Murtagh, F., and Contreras, P. (2012), "Algorithms for Hierarchical Clustering: An Overview," *WIREs Data Mining and Knowledge Discovery*, 2, 86–97. https://doi.org/10.1002/widm.53.

Niederreiter, H. (1992), *Random Number Generation and Quasi-Monte Carlo Methods*, Society for Industrial; Applied Mathematics.

Optica - The Optical Society (2023), "Optical Möbius Strips Yield New Secrets," Accessed: 2025-10-06.

Palan, S., and Schitter, C. (2018), "Prolific. Ac—A Subject Pool for Online Experiments," *Journal of Behavioral and Experimental Finance*, Elsevier, 17, 22–27.

Pedersen, T. L. (2024), *patchwork: The Composer of Plots*, R package version 1.2.0.

Pedersen, T. L. (2025), *ggforce: Accelerating 'ggplot2 '*, R package version 0.5.0.

Perrier, V., Meyer, F., and Granjon, D. (2025), *shinyWidgets: Custom Inputs Widgets for Shiny*, R package version 0.9.0.

R Core Team (2025), *R: A Language and Environment for Statistical Computing*.

Rousseeuw, P. J. (1987), "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis," *Journal of Computational and Applied Mathematics*, 20, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7.

RStudio, PBC (n.d.). "shinyapps.io: Hosted web application service for shiny," Accessed: 2025-10-20.

Saeed, N., Nam, H., Haq, M. I. U., and Muhammad Saqib, D. B. (2018), "A Survey on Multidimensional Scaling," *ACM Comput. Surv.*, 51.

Satija, R., Hoffman, P., and Butler, A. (2025), *SeuratData: Install and Manage Seurat Datasets*, R package version 0.2.2.9002.

Schloerke, B. (2016), *geozoo: Zoo of Geometric Objects*, R package version 0.5.1.

Schloerke, B., Cook, D., Larmarange, J., Briatte, F., Marbach, M., Thoen, E., Elberg, A., and Crowley, J. (2025), *GGally: Extension to 'ggplot2'*, R package version 2.4.0.

Scrucca, L., Fraley, C., Murphy, T. B., and Raftery, A. E. (2023), *Model-Based Clustering, Classification, and Density Estimation Using mclust in R*, Chapman & hall/CRC R series, 1st ed., CRC Press.

Shepard, R. N. (1962), "The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. I." *Psychometrika*, 27, 125–140. https://doi.org/10.1007/BF02289630.

Sievert, C. (2020), *Interactive Web-Based Data Visualization with R, plotly, and shiny*, 1st ed., Chapman; Hall/CRC. https://doi.org/doi.org/10.1201/9780429447273.

Sievert, C., Cheng, J., and Aden-Buie, G. (2025), *bslib: Custom 'Bootstrap' 'Sass' Themes for 'shiny' and 'rmarkdown'*, R package version 0.9.0.

Silva, V., and Tenenbaum, J. (2002), "Global Versus Local Methods in Nonlinear Dimensionality Reduction," *Advances in Neural Information Processing Systems*, 15, 721–728.

Slowikowski, K. (2024), *ggrepel: Automatically Position Non-Overlapping Text Labels with 'ggplot2'*, R package version 0.9.6.

Spearman, C. (1961), "The Proof and Measurement of Association Between Two Things," Appleton-Century-Crofts.

Stefanski, L. A. (2007), "Residual (Sur) Realism," *The American Statistician*, American Statistical Association, Taylor & Francis, Ltd., 61, 163–177. https://doi.org/10.1198/000313007X208407.

Tenenbaum, J. B., Silva, V. de, and Langford, J. C. (2000), "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, 290, 2319–2323. https://doi.org/10.1126/science.290.5500.2319.

The Tabula Muris Consortium (2018), "Single-Cell Transcriptomics of 20 Mouse Organs Creates a Tabula Muris," *Nature*, 562, 367–372. https://doi.org/10.1038/s41586-018-0590-4.

Tinkham, M. (2003), *Group Theory and Quantum Mechanics*, Courier Corporation.

Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., Lennon, N. J., Livak, K. J., Mikkelsen, T. S., and Rinn, J. L. (2014), "The Dynamics and Regulators of Cell Fate Decisions are Revealed by Pseudotemporal Ordering of Single Cells," *Nature Biotechnology*, 32, 381–386. https://doi.org/10.1038/nbt.2859.

Urbanek, S. (2022), *png: Read and Write PNG Images*, R package version 0.1-8.

Ushey, K., Allaire, J., and Tang, Y. (2024), *reticulate: Interface to Python*, R package version 1.38.0.

Venables, W. N., and Ripley, B. D. (2002), *Modern Applied Statistics with S*, Fourth ed., Springer, ISBN 0-387-95457-0.

Wang, Y., Huang, H., Rudin, C., and 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, 1–73.

Warnes, G. R., Bolker, B., Lumley, T., Magnusson, A., Venables, B., Ryodan, G., and Moeller, S. (2023), *gtools: Various R Programming Tools*, R package version 3.9.5.

Wickham, H. (2016), *ggplot2: Elegant Graphics for Data Analysis*, 2nd ed., Springer-Verlag New York. https://doi.org/10.1007/978-3-319-24277-4.

Wickham, H. (2023), *conflicted: An Alternative Conflict Resolution Strategy*, R package version 1.2.0.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019), "Welcome to the Tidyverse," *Journal of Open Source Software*, 4, 1686.

Wickham, H., Cook, D., and Hofmann, H. (2015), "Visualizing Statistical Models: Removing the Blindfold," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 8, 203–225.

Wickham, H., Cook, D., Hofmann, H., and Buja, A. (2011), "tourr: An R Package For Exploring Multivariate Data With Projections," *Journal of Statistical Software*, 40, 1–18.

Wickham, H., Hester, J., and Bryan, J. (2024), *readr: Read Rectangular Text Data*, R package version 2.1.5.

Wickham, H., Pedersen, T. L., and Seidel, D. (2025), *scales: Scale Functions for Visualization*, R package version 1.4.0.

Wilhelm, A. F. X., Wegman, E. J., and Symanzik, J. (1999), "Visual Clustering and Classification: The Oronsay Particle Size Data Set Revisited," *Comput. Stat.*, USA: Kluwer Academic Publishers, 14, 109--146. https://doi.org/10.1007/PL00022701.

Wilkinson, L., and Wills, G. (2008), "Scagnostics Distributions," *Journal of Computational and Graphical Statistics*, ASA Website, 17, 473–491. https://doi.org/10.1198/106186008X320465.

Witten, E. (1985), "Non-Commutative Geometry and Knot Theory," *Communications in Mathematical*

Physics, Springer, 121, 351–399. https://doi.org/10.1007/BF01210791.

Xia, L., Lee, C., and Li, J. J. (2024), "Statistical Method scDEED For Detecting Dubious 2D Single-Cell Embeddings and Optimizing t-SNE and UMAP Hyperparameters," *Nature Communications*, 15, 1753. https://doi.org/10.1038/s41467-024-45891-y.

Xie, Y. (2015), *Dynamic Documents with R and knitr*, 2nd ed., Chapman; Hall/CRC. https://doi.org/10.1201/9781315382487.

Xie, Y. (2016), *DT: A Wrapper of the JavaScript Library 'DataTables '*, R package version 0.2.10.

Xie, Y., Allaire, J. J., and Grolemund, G. (2018), *R Markdown: The Definitive Guide*, 1st ed., Chapman; Hall/CRC. https://doi.org/10.1201/9781138359444.

Yadav, D., Laa, U., Gamage, J. P., and Lee, E.-K. (2025), *polarisR: Non-Linear Dimensionality Reduction Visualization Tool*, R package version 0.1.4.

Zappia, L., Phipson, B., and Oshlack, A. (2017), "Splatter: simulation of single-cell RNA sequencing data," *Genome Biology*. https://doi.org/10.1186/s13059-017-1305-0.

Zeileis, A., Fisher, J. C., Hornik, K., Ihaka, R., McWhite, C. D., Murrell, P., Stauffer, R., and Wilke, C. O. (2020), "colorspace: A Toolbox for Manipulating and Assessing Colors and Palettes," *Journal of Statistical Software*, 96, 1–49.

Zhu, H. (2024), *kableExtra: Construct Complex Table with kable and Pipe Syntax*, R package version 1.4.0.

Appendix A

Appendix to "Choosing Better NLDR Layouts by Evaluating the Model in the High-Dimensional Data Space"

A.1 Methods and hyper-parameters used to generate layouts

Table A.1 contains the list of methods and hyper-parameters used for each of the layouts shown in the paper.

Table A.1: *NLDR methods and hyper-parameters used for each Figure in the main paper.*

Figure	NLDR method	Hyper-parameter(s)
1a	UMAP	n_neighbors = 30, min_dist = 0.3
1b	UMAP	n_neighbors = 5, min_dist = 0.8
1c	UMAP	n_neighbors = 5, min_dist = 0.01
1d	tSNE	perplexity = 5
1e	tSNE	perplexity = 30
1f	PHATE	knn = 5
1g	TriMAP	n_inliers = 12, n_outliers = 4, n_random = 3
1h	PaCMAP	n_neighbors = 30, init = random, MN_ratio = 0.9, FP_ratio = 5
2	tSNE	perplexity = 47

(continued)

Figure	NLDR method	Hyper-parameter(s)
4a	tSNE	perplexity = 47
5b	tSNE	perplexity = 47
6	tSNE	perplexity = 47
8a	tSNE	perplexity = 47
8b	tSNE	perplexity = 62
8c	UMAP	n_neighbors = 15, min_dist = 0.1
8d	PHATE	knn = 5
8e	TriMAP	n_inliers = 12, n_outliers = 4, n_random = 3
8f	PaCMAP	n_neighbors = 10, init = random, MN_ratio = 0.5, FP_ratio = 2
10a	UMAP	n_neighbors = 30, min_dist = 0.3
10b	UMAP	n_neighbors = 5, min_dist = 0.8
10c	UMAP	n_neighbors = 5, min_dist = 0.01
10d	tSNE	perplexity = 5
10e	tSNE	perplexity = 30
10f	PHATE	knn = 5
10g	TriMAP	n_inliers = 12, n_outliers = 4, n_random = 3
10h	PaCMAP	n_neighbors = 30, init = random, MN_ratio = 0.9, FP_ratio = 5
11a	UMAP	n_neighbors = 30, min_dist = 0.3
11e	tSNE	perplexity = 30
12a	tSNE	perplexity = 30
12b	tSNE	perplexity = 89
12c	UMAP	n_neighbors = 15, min_dist = 0.1
12d	PHATE	knn = 5
12e	TriMAP	n_inliers = 12, n_outliers = 4, n_random = 3
12f	PaCMAP	n_neighbors = 10, init = random, MN_ratio = 0.5, FP_ratio = 2

(continued)

Figure	NLDR method	Hyper-parameter(s)
13a	tSNE	perplexity = 30
14a	tSNE	perplexity = 30
*A*4a	tSNE	perplexity = 71
*A*4b	UMAP	n_neighbors = 15, min_dist = 0.1
*A*4c	PaCMAP	n_neighbors = 10, init = random, MN_ratio = 0.5, FP_ratio = 2
*A*5	tSNE	perplexity = 52
*A*6a	UMAP	n_neighbors = 30, min_dist = 0.3
*A*6b	tSNE	perplexity = 30
*A*7a	UMAP	n_neighbors = 30, min_dist = 0.3
*A*7b	tSNE	perplexity = 30
*A*8a	UMAP	n_neighbors = 30, min_dist = 0.3
*A*8b	tSNE	perplexity = 30
*A*9a	UMAP	n_neighbors = 30, min_dist = 0.3
*A*9b	UMAP	n_neighbors = 5, min_dist = 0.8
*A*9c	UMAP	n_neighbors = 5, min_dist = 0.01
*A*9d	tSNE	perplexity = 5
*A*9e	tSNE	perplexity = 30
*A*9f	PHATE	knn = 5
*A*9g	TriMAP	n_inliers = 12, n_outliers = 4, n_random = 3
*A*9h	PaCMAP	n_neighbors = 30, init = random, MN_ratio = 0.9, FP_ratio = 5
*A*10a	tSNE	perplexity = 30
*A*10b	tSNE	perplexity = 89
*A*10c	UMAP	n_neighbors = 15, min_dist = 0.1
*A*10d	PHATE	knn = 5
*A*10e	TriMAP	n_inliers = 12, n_outliers = 4, n_random = 3

(continued)

Figure	NLDR method	Hyper-parameter(s)
*A*10f	PaCMAP	n_neighbors = 10, init = random, MN_ratio = 0.5, FP_ratio = 2

A.2 Videos links

Animations of the *p-D* tours that produced specific projections shown in some figures in the main paper are available on YouTube at the links given in Table A.2.

Table A.2: *Videos of the langevitour animations and the linked plots.*

Figure	URL
4	youtu.be/yHKTHK4UBiU
5	youtu.be/FukiminrO90
11	youtu.be/3VfK3M2gnZM, youtu.be/Es84bwQcndU
13	youtu.be/sUcGd57Swdg, youtu.be/QiklCjELUxo

A.3 Notation

Table A.3: *Summary of notation for describing new methodology.*

Notation	Description
n, p, k	number of observations, variables, embedding dimension, respectively
$\boldsymbol{X}, \boldsymbol{x}$	p-dimensional data (population, sample)
$\boldsymbol{y}$	k-dimensional layout
P	orthonormal basis, generating a $d\text{-}dimensional$ linear projection of p-dimensional data
T	true model
g	functional mapping from p-D to k-D, especially as prescribed by NLDR
$\boldsymbol{\theta}$	(Hyper-) parameters for NLDR method
r	ranges of the embedding components
$C^{(j)}$	j-dimensional bin centers
(b_1, b_2)	number of bins in each direction
(a_1, a_2)	binwidths, distance between centroids in each direction
$(s_1,\ s_2)$	starting coordinates of the hexagonal grid
q	buffer to ensure hexgrid covers data, proportion of data range, 0-1
m	number of non-empty bins
b	number of hexagons in the grid
h	hexagonal id
l	side length
A	area
n_h	number of points in hexagon h (bin count)
w_h	standardized number of points in hexagon h (standardized bin counts)

A.4 Scripts

Table A.4: *R and Python script files used to generate outputs in the main paper.*

Folder	Script	Description
script	additional_functions.R	Helper functions to render the main paper.
script	evaluation.py	Python script implementing additional evaluation metrics such as RTA and GS.
script	nldr_code.R	Wrapper functions for running multiple NLDR methods (UMAP, tSNE, PHATE, PaCMAP, TriMAP) with different parameters.
two_nonlinear	01_gen_data.R	Generates the 2NC7 dataset.
two_nonlinear	02_gen_true_model.R	Creates the true structure of 2NC7 data.
two_nonlinear	03_gen_embeddings.R	Computes multiple NLDR embeddings for the 2NC7 data.

(continued)

Folder	Script	Description
two_nonlinear	04_gen_mse_for_diff_methods.R	Computes HBE with varying bin widths (a_1) for all NLDR embeddings.
two_nonlinear	05_gen_rm_lwd_mse.R	Computes HBE with varying low density bin cutoff for all three binwidth (a_1) choices.
two_nonlinear	06_gen_model_with_tSNE.R	Fits the model for the layout a.
two_nonlinear	07_example_evaluation_metrics.R	Calculates evaluation metrics for all NLDR layouts.
two_nonlinear	08_gen_model_with_PHATE.R	Fits the model for the layout c.
five_gau_clusters	01_five_gaussian_cluster_data_emb.R	Generates data and multiple NLDR embeddings.
five_gau_clusters	02_gen_model_with_tSNE.R	Fits the model for the layout a.
five_gau_clusters	03_gen_model_with_UMAP.R	Fits the model for the layout b.
five_gau_clusters	04_gen_model_with_PaCMAP.R	Fits the model for the layout c.

(continued)

Folder	Script	Description
c_shaped_dens_str	01_gen_data.R	Generates the 2-*D* curved sheet dataset.
c_shaped_dens_str	02_gen_embeddings_uni_dens.R	Generates multiple NLDR embeddings.
c_shaped_dens_str	03_gen_model_with_tSNE.R	Fits the model for the tSNE layout.
pbmc3k	01_obtain_pca_author.R	Obtains author PCA results.
pbmc3k	02_obtain_umap_authors.R	Obtains author UMAP embeddings.
pbmc3k	03_gen_umap_diff_param.R	Generates multiple UMAP embeddings with different hyperparameter values.
pbmc3k	04_gen_tsne_diff_param.R	Generates multiple tSNE embeddings with different hyperparameter values.

(continued)

Folder	Script	Description
pbmc3k	05_gen_phate.R	Generates a PHATE embeddings with default hyper-parameters.
pbmc3k	06_gen_trimap.R	Generates a TriMAP embeddings with default hyper-parameters.
pbmc3k	07_gen_pacmap.R	Generates a PaCMAP embeddings with default hyper-parameters.
pbmc3k	08_gen_mse_for_diff_methods.R	Computes HBE with varying bin widths (a_1) for all NLDR embeddings.
pbmc3k	09_gen_scDEED.R	Generates UMAP embeddings from scDEED results.
pbmc3k	10_pre_process_for_embedding.R	Generates PBMC3k data used for scDEED results.

(continued)

Folder	Script	Description
pbmc3k	11_gen_mse_for_diff_tsne_scD.R	Computes HBE with varying bin widths (a_1) for tSNE embeddings.
pbmc3k	12_gen_mse_for_diff_umap_scD.R	Computes HBE with varying bin widths (a_1) for UMAP embeddings.
pbmc3k	13_gen_model_with_UMAP.R	Fits the model for the layout a.
pbmc3k	14_gen_model_with_tSNE.R	Fits the model for the layout e.
pbmc3k	15_gen_model_with_UMAP_scD.R	Fits the model for the layout a.
pbmc3k	16_gen_model_with_tSNE_scD.R	Fits the model for the layout b.
pbmc3k	17_evaluation_metrics.R	Calculates evaluation metrics for all NLDR layouts.
pbmc3k	18_evaluation_metrics_scD.R	Calculates evaluation metrics for all NLDR layouts.
mnist	01_data_preprocessing.R	Computes first 10 principal components and save data.

(continued)

Folder	Script	Description
mnist	02_gen_diff_embeddings.R	Generates multiple NLDR embeddings.
mnist	03_gen_mse_for_diff_methods.R	Computes HBE with varying bin widths (a_1) for all NLDR embeddings.
mnist	04_gen_model_with_tSNE.R	Fits the model for the layout a.
mnist	05_evaluation_metrics.R	Calculates evaluation metrics for all NLDR layouts.
mnist	06_link_brush_layout_e.R	Creates interactive linked brushing with layout e.

A.5 Generating the 2NC7 data

This data is constructed by simulating two clusters, each consisting of 1000 observations. The C-shaped cluster is generated from $\theta \sim U(\text{-}3\pi/2, 0)$, $X_1 = \sin(\theta)$, $X_2 \sim U(0, 2)$ (adding thickness to the C), $X_3 = \text{sign}(\theta) \times (\cos(\theta) - 1)$, $X_4 = \cos(\theta)$. Observations lie on a 2-D manifold in 7-D. The other cluster is from $X_1 \sim U(0, 2)$, $X_2 \sim U(0, 3)$, $X_3 = \text{-}(X_1^3 + X_2)$, and $X_4 \sim U(0, 2)$. It is also curved, but observations lie on a 3-D manifold in 7-D. Three more variables, X_5, X_6, X_7, that are small amounts of pure noise, are added. We would consider (X_1, X_2, X_3, X_4) to be the geometric structure (true model) that we hope to capture (Figure A.1).

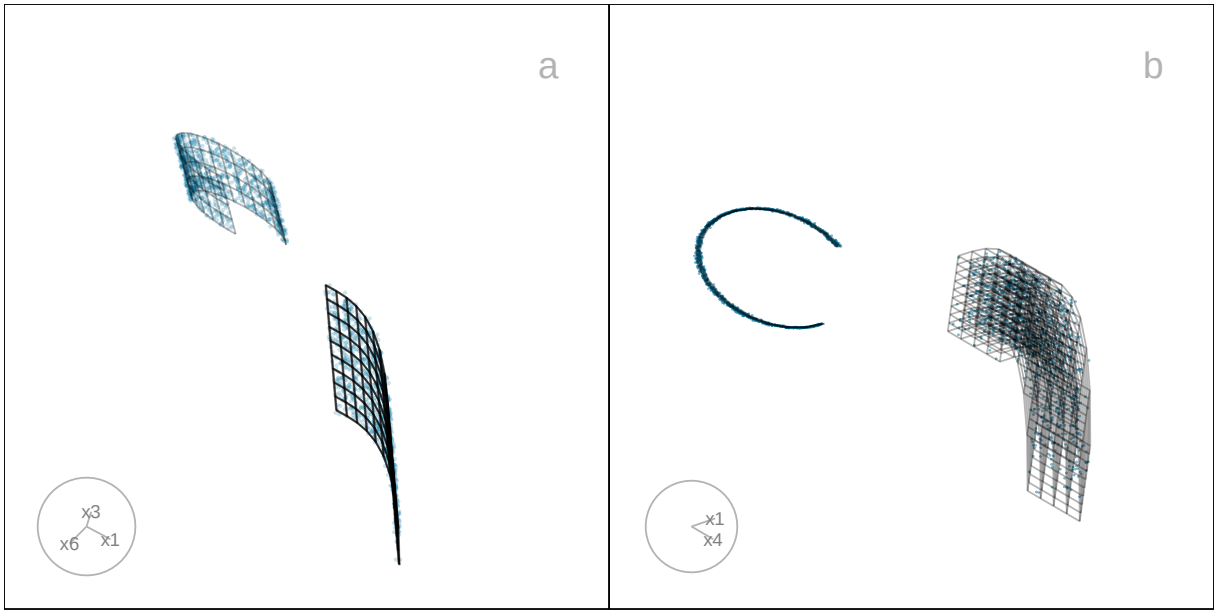

Figure A.1: *Two projections of the p-D true model overlaying the data are shown in a, b. Video of the langevitour animations is available at https://youtu.be/35TrnYJsUUI.*

A.6 Computing hexagon grid configurations

Given range of embedding component, r_2, number of bins along the x-axis, b_1, and buffer proportion, q, hexagonal starting point coordinates, $s_1 = -q$, and $s_2 = -qr_2$. The purpose is to find the width of the hexagon, a_1, and the number of bins along the y-axis, b_2.

Geometric arguments give rise to the following constraints.

min a_1 s.t.

$$s_1 - \frac{a_1}{2} < 0, \tag{A.1}$$

$$s_1 + (b_1 - 1) \times a_1 \geq 1, \tag{A.2}$$

$$s_2 - \frac{a_2}{2} < 0, \tag{A.3}$$

$$s_2 + (b_2 - 1) \times a_2 \geq r_2. \tag{A.4}$$

Since a_1 and a_2 are distances,

$$a_1, a_2 > 0.$$

Also, $(s_1, s_2) \in$ (-0.1, -0.05) as these are multiplicative offsets in the negative direction.

Equation A.1 can be rearranged as,

$$a_1 > 2s_1$$

which given $s_1 < 0$ and $a_1 > 0$ will *always* be true. The same logic follows for Equation A.3 and substituting $a_2 = \sqrt{3}a_1/2$, and $s_2 =$ -qr_2 to Equation A.3 can be written as,

$$a_1 > -\frac{4}{\sqrt{3}} qr_2$$

Also, substituting $a_2 = \sqrt{3}a_1/2$, $s_2 =$ -qr_2 and rearranging Equation A.4 gives:

$$a_1 \geq \frac{2(r_2 + qr_2)}{\sqrt{3}(b_2 - 1)}. \tag{A.5}$$

Similarly, substituting $s_1 =$ -q Equation A.2 becomes,

$$a_1 \geq \frac{(1+q)}{(b_1 - 1)}. \tag{A.6}$$

This is a linear optimization problem. Therefore, the optimal solution must occur on a vertex. So, by setting Equation A.5 equals to Equation A.6 gives,

$$\frac{2(r_2 + qr_2)}{\sqrt{3}(b_2 - 1)} = \frac{(1+q)}{(b_1 - 1)}.$$

After rearranging this,

$$b_2 = 1 + \frac{2r_2(b_1 - 1)}{\sqrt{3}}$$

and since b_2 should be an integer,

$$b_2 = \left\lceil 1 + \frac{2r_2(b_1 - 1)}{\sqrt{3}} \right\rceil. \tag{A.7}$$

Furthermore, with known b_1 and b_2, by considering Equation A.2 or Equation A.4 as the *binding* or *active constraint*, can compute a_1.

If Equation A.2 is active, then,

$$\frac{(1+q)}{(b_1-1)} < \frac{2(r_2+qr_2)}{\sqrt{3}(b_2-1)}.$$

Rearranging this gives,

$$r_2 > \frac{\sqrt{3}(b_2-1)}{2(b_1-1)}.$$

Therefore, if this equality is true, then

$$a_1 = \frac{(1+q)}{(b_1-1)},$$

otherwise,

$$a_1 = \frac{2r_2(1+q)}{\sqrt{3}(b_2-1)}.$$

A.7 Binning the data

Points are assigned to the bin they fall into based on the nearest centroid (Figure A.2). If a point is equidistant from multiple centroids, it is assigned to the centroid with the smallest bin ID.

A.8 Area of a hexagon

The area of a hexagon is defined as $A = 3\sqrt{3}l^2/2$, where l is the side length of the hexagon (Figure A.3). l can be computed using a_1 and a_2.

By applying the Pythagorean theorem, we obtain,

$$l^2 = \left(\frac{a_1}{2}\right)^2 + \left(\frac{a_2-l}{2}\right)^2.$$

Next, rearranging the terms, we get,

$$l^2 - \left(\frac{a_2-l}{2}\right)^2 = \left(\frac{a_1}{2}\right)^2,$$

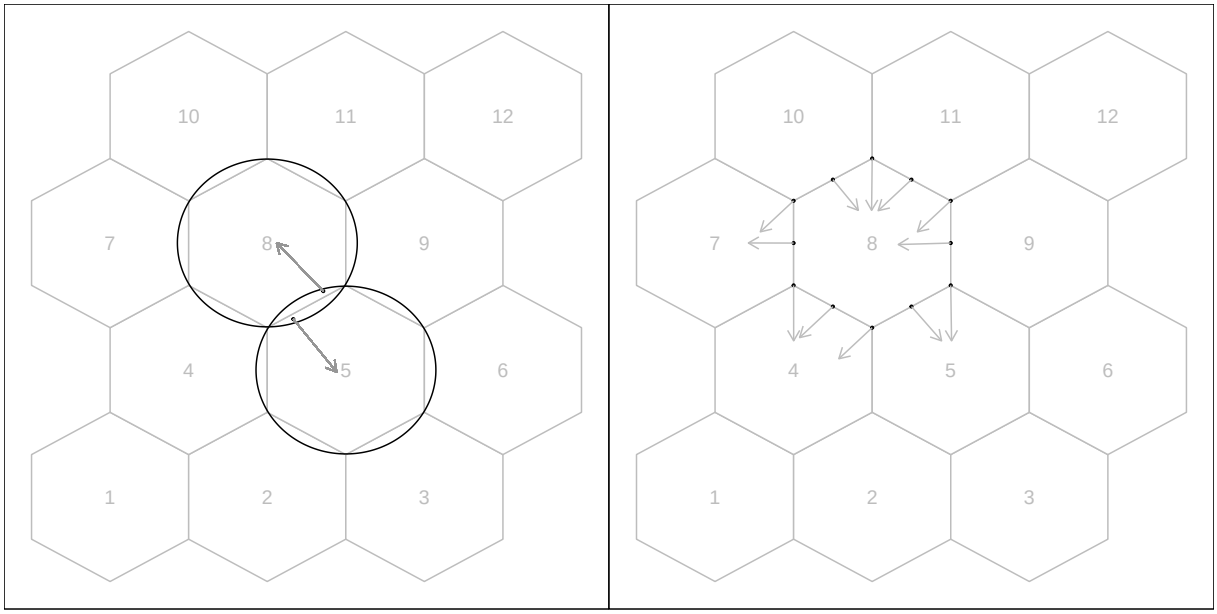

Figure A.2: *Binning the data. Points are assigned to the nearest centroid. If a point is equidistant from multiple centroids, assigned to the centroid with the smallest bin ID.*

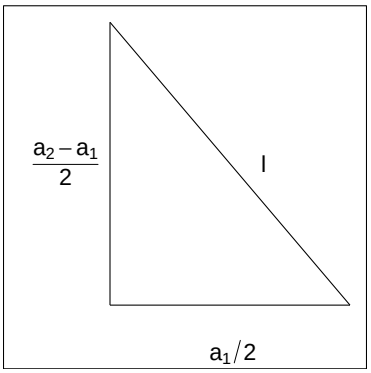

Figure A.3: *The components of the right triangle illustrating notation.*

$$\left[l-\left(\frac{a_2-l}{2}\right)\right]\left[l+\left(\frac{a_2-l}{2}\right)\right]=\left(\frac{a_1}{2}\right)^2,$$

$$3l^2+2a_2l-(a_1^2+a_2^2)=0.$$

Finally, by solving the quadratic equation, we compute,

$$l=\frac{-2a_2\pm\sqrt{4a_2^2-24[-(a_1^2+a_2^2)]}}{6},$$

$$l=\frac{-a_2\pm\sqrt{a_2^2-6[-(a_1^2+a_2^2)]}}{3},$$

where $l > 0$.

A.9 Curiosities about NLDR results discovered by examining the model in the data space

With the drawing of the model in the data, several interesting differences between the NLDR methods can be observed.

A.9.1 Some methods appear to order points in the layout

The 2-*D* model representations generated from some NLDR methods, especially PaCMAP, are unreasonably flat or like a pancake. A simple example of this can be seen with data simulated to contain five 4-*D* Gaussian clusters. Each cluster is essentially a ball in 4-*D*, so there is no 2-*D* representation; rather, the model in each cluster should resemble a crumpled sheet of paper that fills out 4-*D*.

Figure A.4 a1, b1, c1 show the 2-*D* layouts for (a) tSNE, (b) UMAP, and (c) PaCMAP, respectively. The default hyper-parameters for each method are used. In each layout, we can see an accurate representation where all five clusters are visible, although with varying degrees of separation.

The models are fitted to each of these layouts. Figure A.4 a2, b2, c2 show the fitted models in a projection of the 4-*D* space, taken from a tour. These clusters are fully 4-*D* in nature, so we would expect the model to be a *crumpled sheet* that stretches in all four dimensions. This is what is mostly observed for tSNE and UMAP. The curious detail is that the model for PaCMAP is closer to a *pancake* in shape in every cluster! This single projection only shows this in three of the five clusters, but if we examine a different projection, the other clusters also exhibit the pancake. While we don't know what exactly causes this, it is likely due to some ordering of points in the 2-*D* PaCMAP layout that induces the flat model. One could imagine that if the method used principal components on all the data, it might induce some ordering that would produce the flat model. If this were the reason, the pancaking would be the same in all clusters, but it is not: The pancake is visible in some clusters in some projections, but in other clusters it is visible in different projections. It might be due to some ordering by nearest neighbors in a cluster. The PaCMAP documentation doesn't provide any helpful clues. That this happens, though, makes the PaCMAP layout inadequate for representing the high-dimensional data.

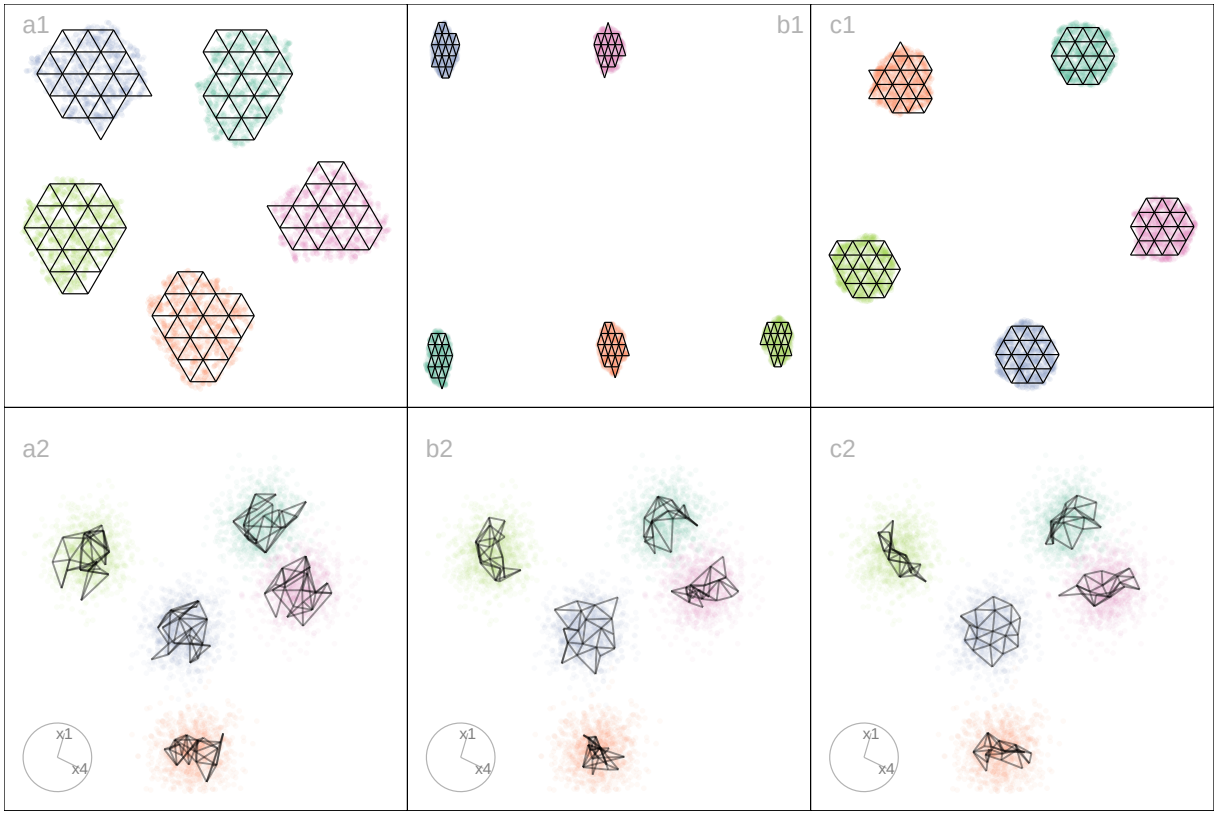

Figure A.4: *NLDR's organize points in the 2-D layout in different ways, possibly misleadingly, illustrated using three layouts: (a) tSNE, (b) UMAP, (c) PaCMAP. The data has five Gaussian clusters in 4-D. The bottom row of plots shows a 2-D projection from a tour on 4-D, revealing the differences generated by the layouts on the model fits. We would expect the model fit to be like that in (a2), where it is distinctly separate for each cluster but like a hairball in each. This would indicate the distinct clusters, each being fully 4-D. With (c2), the curiosity is that the model is a 2-D pancake shape in 4-D, indicating that there is some ordering of points done by PaCMAP, possibly along some principal component axes. Videos of the langevitour animations are available at https://youtu.be/I-kxCwVfqiQ, https://youtu.be/gD1P01FUPyU, and https://youtu.be/MxJ_srOFQNk respectively.*

A.9.2 Sparseness creates a contracted 2-*D* layout

Differences in density can arise from sampling at different rates in different subspaces of *p-D*. For example, the data shown in Figure A.5 all lie on a 2-*D* curved sheet in 4-*D*, but one end of the sheet is sampled densely and the other very sparsely. It was simulated to illustrate the effect of the density difference on layout generated by an NLDR, illustrated using the tSNE results, but it happens with all methods.

Figure A.5 (a2, b2) shows a 2-*D* layout for tSNE created using the default hyper-parameters. One would expect to see a rectangular shape if the curved sheet is flattened, but the layout is triangular. The other two displays show the residuals as a dot density plot (a1, b1), and a 2-*D* projection of the data and the model from 4-*D* (a3, b3). Using linked brushing between the plots, we can highlight points in the tSNE layout, and examine where they fall in the original 4-*D*. The darker (maroon)

points indicate points that have been highlighted by linking. In row a, the points at the top of the triangle are highlighted, and we can see these correspond to higher residuals, and also to all points at the low density end of the curved sheet. In row b, points at the lower left side of the triangle are highlighted, which corresponds to smaller residuals and one corner of the sheet at the high-density end of the curved sheet.

The tSNE behaviour is to squeeze the low-density area of the data together into the layout. This is common in other NLDR methods also, which means analysts need to be aware that if their data is not sampled relatively uniformly, apparent closeness in the 2-*D* may correspond to sparseness in *p*-*D*.

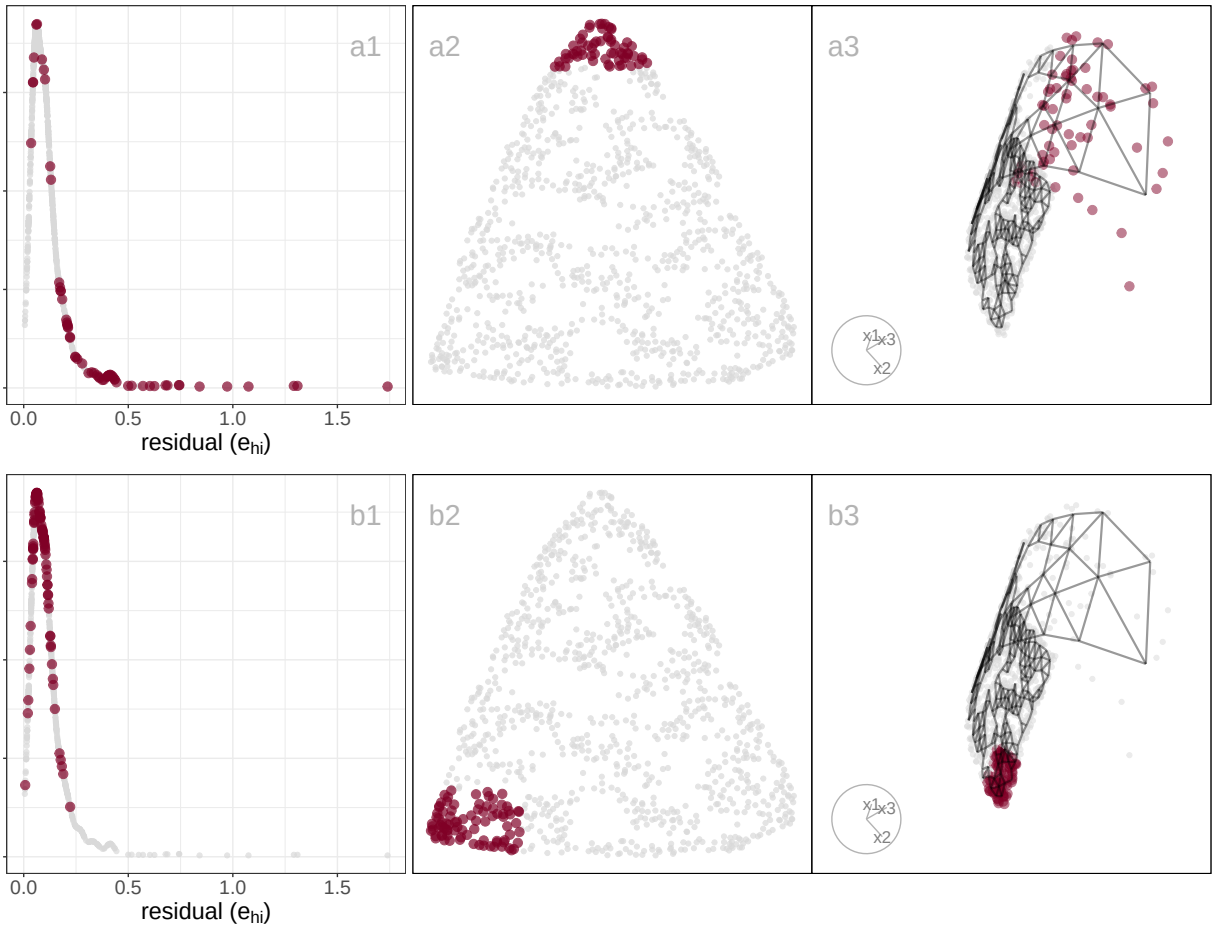

Figure A.5: *Exploring the effect of density on the NLDR layout using a 2-D curved sheet in 4-D with different densities at each end. Three plots are linked: density plot of residuals (a1, b1), NLDR layout (a2, b2), projection of 4-D model and data (a3, b3). The brown points indicate the selected set, which is different in each row. In (a2), the top part of the triangular shape is selected, which corresponds to higher residuals (a1) and the sparse end of the structure (a3). In (b2), one of the other corners is highlighted, which can be seen to correspond to low residuals (b1) and one side of the dense end of the data (b3). While the tSNE layout represents the dense end of the sheet correctly as two corners in the layout, it contracts the sparse end of the sheet into a single corner. Video of the langevitour animation is available at https://youtu.be/-KsQH0rII2A.*

A.10 PBMC3k: comparison with results of scDEED recommendations

As we were writing this paper Xia et al. (2024) appeared proposing a new method called scDEED, helping to assess the validity of a 2-D embedding. scDEED calculates a reliability score for each cell embedding based on the similarity between the cell's 2-D embedding neighbors and its neighbors prior to embedding. A low reliability score suggests a dubious embedding. It can help in deciding on optimal hyper-parameters. Here, we illustrate how our method compares with the results from scDEED.

Note that Xia et al. (2024) uses a different PBMC dataset than that used by Chen et al. (2024), shown by us in the main paper example, which is why this comparison is shown here and not in the main paper. Their data contains 31,021 cells including cell type labels, and the gene expression levels were in the unit of log-transformed UMI count per 10,000. They focused on three sequencing methods (inDrops, DropSeq, and SeqWell) and four common cell types: Cytotoxic T cell, CD4+T cell, CD14+ Monocyte, and B cell. Pre-processing follows the process in Xia et al. (2024) again using the Human Peripheral Blood Mononuclear Cells (PBMC) data.

For illustration purposes, we only selected cells generated with inDrops ($n = 5858$ cells). Also, Xia et al. (2024) used the first 9 principal components to generate the UMAP and tSNE with default hyper-parameters. The objective is to determine what scDEED suggests is the best layout with what HBE would choose. Layout a (Figure A.6) is generated from the hyper-parameters suggested by Chen et al. (2024), and layout b (Figure A.6) is with suggested hyper-parameters by scDEED to be more accurate. The HBE vs binwidth (a_1) plot (Figure A.6) illustrates that our approach would suggest that scDEED is correct here, that layout b is more accurately reflecting the cluster structure in the PBMC data. This is also supported by examining the models in the data space, as shown in Figure A.7.

A.11 Compare HBE with existing evaluation metrics

Figure A.8 and Figure A.9 compare HBE with commonly used evaluation metrics such as rRTA, rARNX, rSC, and rGS across multiple NLDR layouts. These visual comparisons highlight that HBE behaves differently from these existing metrics due to the different settings involved.

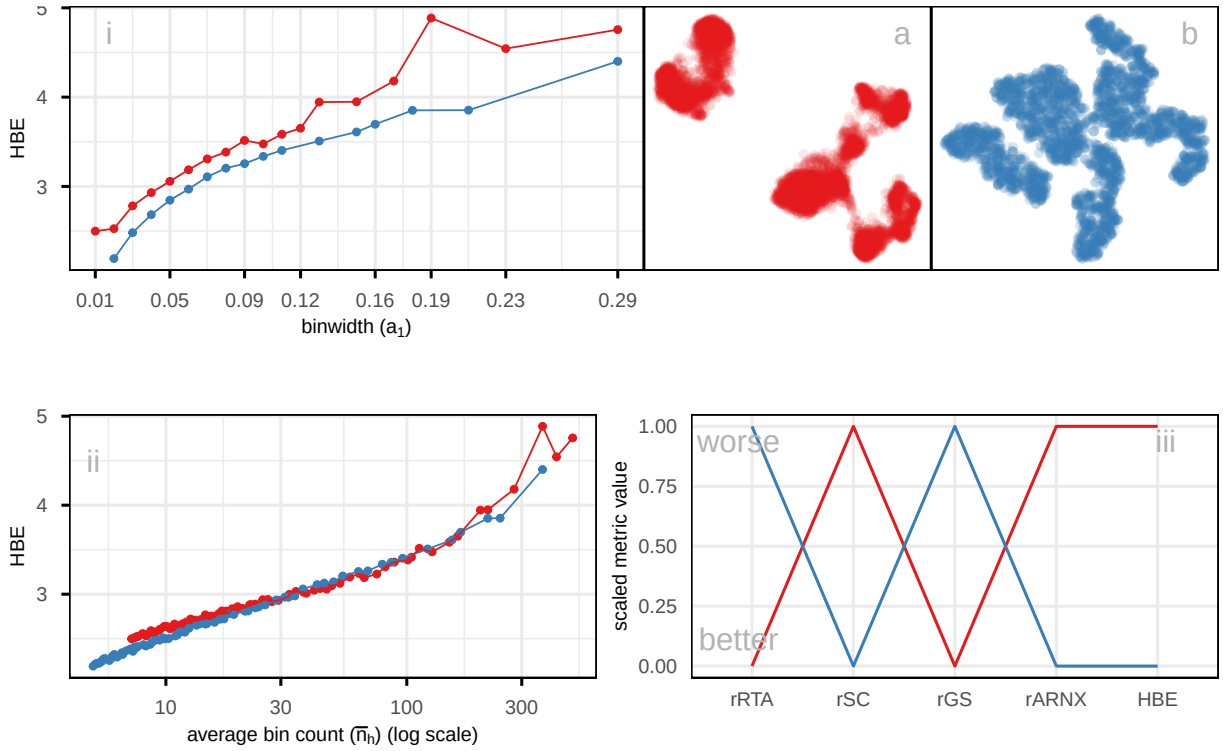

Figure A.6: *Comparing the published layout (a) with what would be suggested to be optimal by scDEED (b), using HBE for varying (i) binwidth (a_1), and (ii) average bin count ($\bar{n}_h$), on a subset of PBMC3k data. Color represents NLDR layouts. HBE would corroborate that the scDEED optimized layout is better than what was originally published. Plot (ii), which accounts for the density within clusters by using average bin count, shows reduced differences between layouts, indicating that part of the variation in (i) is driven by cluster density rather than true structural differences. Comparison of scaled evaluation metrics (iii) (rRTA, rSC, rGS, rARNX, and HBE using $a_1 = 0.04$) for two NLDR layouts of the PBMC3k data, the originally published layout (a) and the scDEED optimized layout (b). Each line represents a layout, with color matching the corresponding scatterplots. Most metrics (rSC, rARNX, and HBE) consistently indicate that the optimized layout (b) provides a better representation, while rRTA and rGS slightly favor the published layout.*

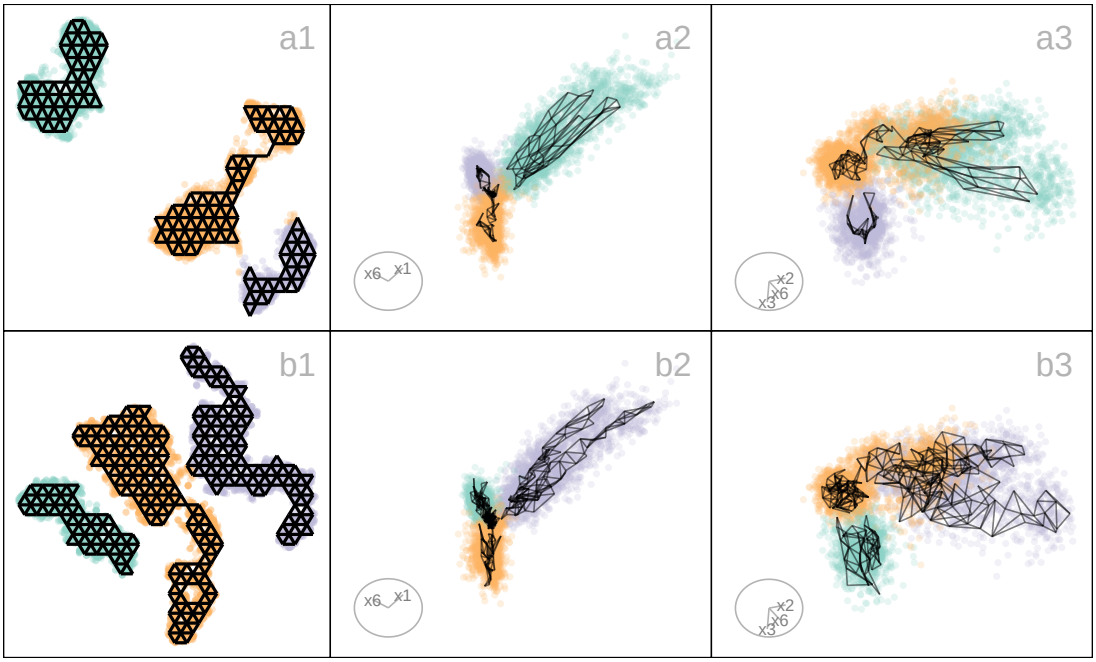

Figure A.7: *Compare the published 2-D layout (Figure A.6 a) made with UMAP and the 2-D layout made with tSNE selected as optimal by scDEED (Figure A.6 b) and also HBE (Figure A.6). The two plots on the right show projections from a tour, with the models overlaid. The published layout a suggests three separated clusters, with two of them are close, but this is not present in the data. While there may be three clusters, they are not well-separated. The difference in model fit also indicates this: the published layout a does not capture the nonlinear structure of the clusters like the model generated from layout b. This supports the choice that layout b is the better representation of the data, because it shows close clusters. Videos of the langevitour animations are available at https://youtu.be/ffiB4MGWyn8 and https://youtu.be/e7XNL18co1c respectively.*

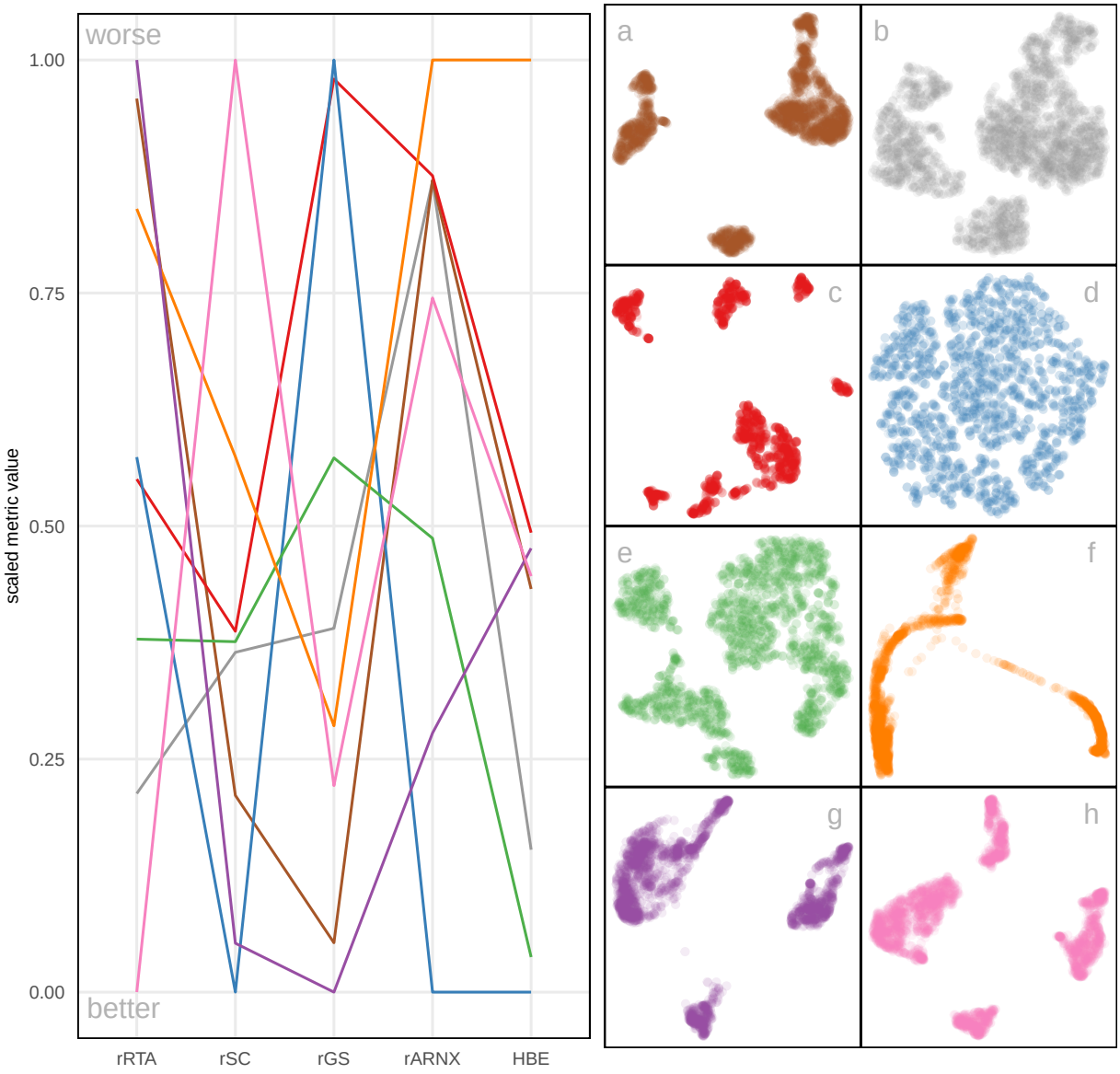

Figure A.8: *Comparison of scaled evaluation metrics (rRTA, rSC, rGS, rARNX, and HBE with $a_1 = 0.06$) for the eight NLDR layouts computed on the PBMC3k data, shown as a parallel coordinate plot. The color of each line corresponds to an NLDR layout. All, except rGS and rARNX agree that layout e is best or very close to best. Layout d is best according to HBE and rARNX, but considered to be much less optimal by rRTA, rSC, and rGS. Layout f is considered poor by rARNX and HBE. Layout a is considered close to the best by rGS and rSC.*

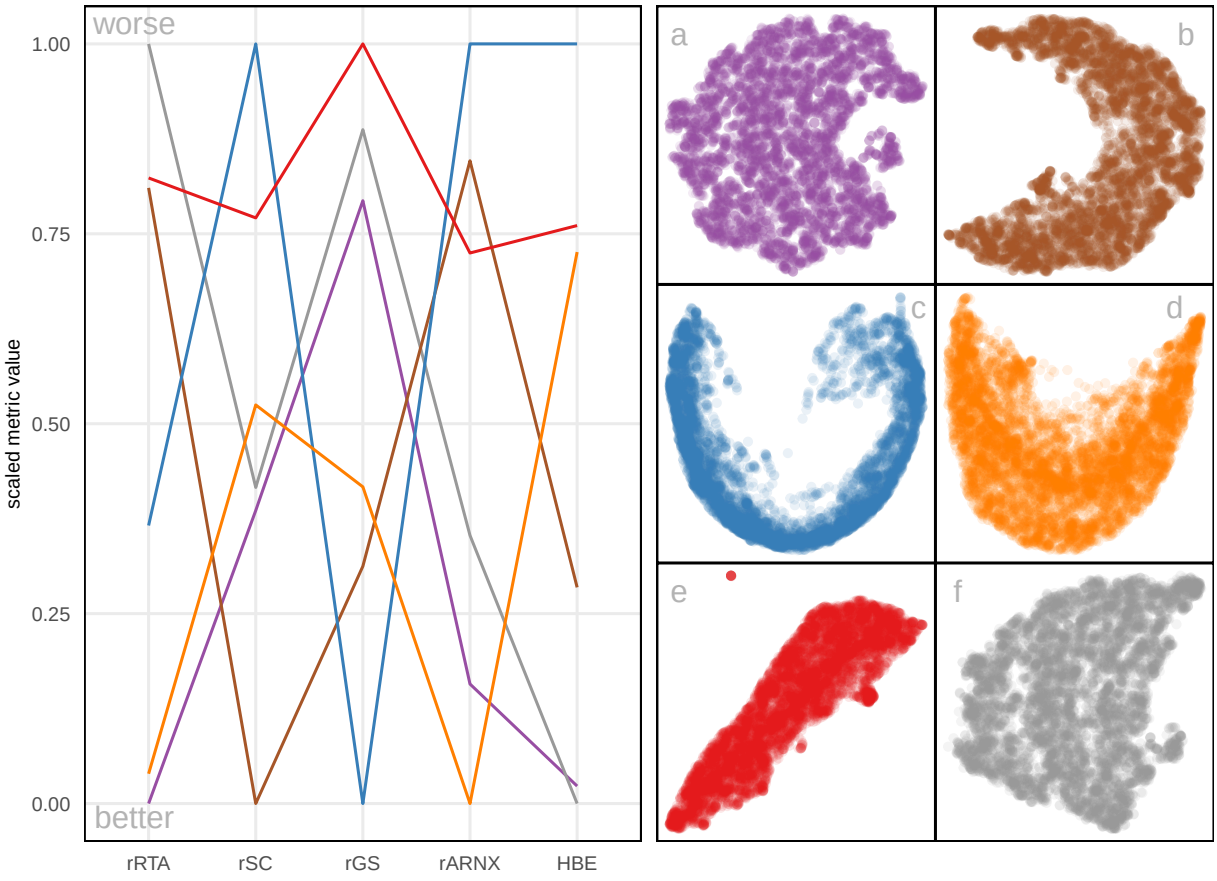

Figure A.9: *Comparison of scaled evaluation metrics (rARNX, rRTA, rSC, rGS, and HBE using $a_1 = 0.04$) for six NLDR layouts computed on the MNIST digit 1 data using a parallel coordinate plot. Each line represents a layout (a–f), with colors corresponding to the scatterplots shown on the right. The metrics display different ranking patterns, indicating that no single measure fully captures embedding quality. Layout a is identified as the best according to HBE and rRTA, but is considered much less optimal by rARNX, rSC, and rGS. Layout e is considered the worst, or close to the poorest, by all metrics. Layouts a and f show similar patterns of agreement across metrics, except for rRTA, where layout a performs the best and layout f the worst. Layout c is the worst in rARNX, rSC, and HBE.*

Appendix B

Appendix to "Perception and Misperception of Clustering in Nonlinear Dimension Reduction: A User Study"

B.1 Scripts

Table B.1: *R script files used to generate outputs in the main paper.*

Script	Description
additional_functions.R	Helper functions to render the main paper.
01_attention_check_data_structures.R	Function to generate three- and four- Gaussian clusters data for attention check.
01_data_structure_components.R	Functions to generate data structure components for non-attention check.
02_data_structures.R	Functions to generate three clusters data structure for non-attention check.

(continued)

Script	Description
03_exp_design_with_method_and_distance_factor.R	Creates the experimental design, varying NLDR method and distance scale factors.
04_exp_design_with_new_ds_factors.R	Extends the experimental design to include additional distance scale factor.
05_gen_clust3_attention_check_data.R	Generates three-cluster data for attention check.
05_gen_cluster3_high_d_data.R	Generates three-cluster data with medium-large distance scale factor.
06_gen_clusters3_with_diff_dist.R	Generates three-cluster data with varying inter-cluster distance scale factors.
09_gen_clusters_merge_all_data.R	Merges all generated cluster data (attention and non-attention check) into a single combined dataset.
10_gen_embeddings.R	Computes multiple NLDR embeddings for specific distance scale factor.
11_comb_emb_default_data.R	Combines NLDR embeddings for all distance scale factors.
12_comb_data.R	Merge all NLDR embeddings generated for attention and non-attention check.
13_data_processing_method_ds_factor_missings.R	Processes collected experimental data and generates the file, containing all relevant details for the same data structure shown in both displays.

(continued)

Script	Description
13_data_processing_method_ds_factor.R	Processes collected experimental data and generates the file, containing all relevant details for the same data structure shown in both displays.
17_compute_distance_btw_centroids.R	Computes different distance metrics between cluster in the high-dimensional space.
19_find_which_replicates_missing.R	Identifies missing responses across experimental conditions.
pwr_analysis_umap_0.1_0.6.R	Power analysis to decide the number of responses needed to detect the difference between UMAP 0.1 and 0.6 distance scale factors.
pwr_analysis_tsne_0.1_0.6.R	Power analysis to decide the number of responses needed to detect the difference between tSNE 0.1 and 0.6 distance scale factors.

B.2 Data sets

Table B.2 summarizes the three-cluster data sets used in the experiment. Each data set was generated using the `cardinalR` package (Gamage et al. 2025b) and comprises three clusters with distinct structures. The collection of structures spans a wide range of nonlinear, curved, and density-based configurations in 4-*D* space, providing controlled yet varied settings for assessing perceptual differences across NLDR methods. All data sets used in this experiment are available at https://github.com/JayaniLakshika/Monash_PhD_thesis/blob/main/data/vis-exp/high_d_data_three_clust_all.rds.

Animations of the 4-*D* tours that were used for the study's non-attention check SAME trials, non-

Table B.2: *Description of the simulated three-cluster data structures. Each data structure consists of three clusters with different geometric shapes.*

Data structure	Cluster1	Cluster2	Cluster3
three_clust_01	curv	elliptical	blunted_cone
three_clust_02	s_curve	cube	pyramid_rectangular_base
three_clust_03	curvy_cylinder	hemisphere	pyramid_triangular_base
three_clust_04	curv2	Gaussian	filled_hexagonal_pyramid
three_clust_05	nonlinear_hyperbola	elliptical	blunted_cone
three_clust_06	crescent	cube	pyramid_rectangular_base
three_clust_07	nonlinear_hyperbola2	hemisphere	pyramid_triangular_base
three_clust_08	conic_spiral	Gaussian	filled_hexagonal_pyramid
three_clust_09	helical_hyper_spiral	cube	blunted_cone
three_clust_10	spherical_spiral	Gaussian	pyramid_triangular_base
three_clust_11	curv	elliptical	pyramid_rectangular_base
three_clust_12	s_curve	hemisphere	filled_hexagonal_pyramid
three_clust_13	curvy_cylinder	cube	blunted_cone
three_clust_14	curv2	Gaussian	pyramid_triangular_base
three_clust_15	nonlinear_hyperbola	elliptical	pyramid_rectangular_base
three_clust_16	crescent	hemisphere	filled_hexagonal_pyramid
three_clust_17	nonlinear_hyperbola2	cube	blunted_cone
three_clust_18	conic_spiral	Gaussian	pyramid_triangular_base
three_clust_19	helical_hyper_spiral	hemisphere	filled_hexagonal_pyramid
three_clust_20	spherical_spiral	elliptical	blunted_cone
three_clust_21	curv	Gaussian	pyramid_rectangular_base
three_clust_22	s_curve	cube	pyramid_triangular_base
three_clust_23	curvy_cylinder	hemisphere	filled_hexagonal_pyramid
three_clust_24	curv2	elliptical	blunted_cone
three_clust_25	nonlinear_hyperbola2	Gaussian	pyramid_rectangular_base
three_clust_26	crescent	cube	pyramid_triangular_base
three_clust_27	nonlinear_hyperbola2	hemisphere	filled_hexagonal_pyramid
three_clust_28	conic_spiral	elliptical	blunted_cone
three_clust_29	Gaussian	Gaussian	Gaussian
three_clust_30	Gaussian	Gaussian	Gaussian

attention check DIFFERENT trials, and attention-check trials are available on YouTube at the links given in Table B.3, Table B.4, and Table B.5.

Table B.3: *Videos of datasets used for non-attention check SAME trials.*

Data structure	**Small**	**Small-medium**	**Medium**	**Medium-large**	**Large**
three_clust_01	youtu.be/kZyZxujDz58	youtu.be/Jz3k4uIAiRo	youtube.com/shorts/QqMDQxShke0	youtu.be/E9msE_XX0KA	youtube.com/shorts/07Ya6SjNDV0

(continued)

Data structure	Small	Small-medium	Medium	Medium-large	Large
three_clust_02	youtu.be/CLMlOU4Fb2w	youtu.be/TFj0satlBBE	youtube.com/shorts/jKarI60euSw	youtu.be/f2WvtD2xog	youtube.com/shorts/Vk7K5vlXiVM
three_clust_03	youtu.be/K2oKM4mUBXM	youtu.be/b-43HKN30ws	youtube.com/shorts/edCnIfgfoU0	youtu.be/7NwNcD4qlLc	youtube.com/shorts/-aB3PwE676E
three_clust_04	youtu.be/7yvvpPgiWNw	youtu.be/1PhZO7cUEaI	youtu.be/XO61YVXAdr8	youtu.be/XO61YVXAdr8	youtube.com/shorts/7e60CeOM50Q
three_clust_05	youtu.be/pbI7UXFgc0k	youtu.be/G-TvOIBj-14	youtube.com/shorts/I9xxCinW4Ec	youtu.be/ardE0G7zevk	youtube.com/shorts/21xj8nnnvec
three_clust_06	youtu.be/Mxylk4M67iA	youtu.be/ABrxozu8F-A	youtube.com/shorts/FISgM4T2xEI	youtu.be/soFQR9UwNsg	youtube.com/shorts/pX0E8E-Dbxc
three_clust_07	youtu.be/2a89BQGK_iU	youtu.be/Wt4NwZSACmo	youtube.com/shorts/MbGOoTrvVXk	youtu.be/hVwIjSxACoo	youtube.com/shorts/y0kitUPbAoQ
three_clust_08	youtu.be/eID-dwpgU44	youtu.be/ILwnlZUMj_U	youtube.com/shorts/6k20OE3Fkcg	youtu.be/oSBaMH9HJZ4	youtube.com/shorts/B4OiM4sfZ4g

(continued)

Data structure	Small	Small-medium	Medium	Medium-large	Large
three_clust_09	youtu.be/6uGCDUSL60Q	youtu.be/RvlSY3drV5I	youtube.com/shorts/TIyP-a75YmQ	youtu.be/mh_rG2qy2Pc	youtube.com/shorts/hlaX3J8ibsA
three_clust_10	youtu.be/CX5O4eNZW5o	youtu.be/fHxflXa9i-s	youtube.com/shorts/hUzYOFS8o4M	youtu.be/R6vD1xJH21w	youtube.com/shorts/lM2sohLJS2s
three_clust_11	youtu.be/1f8S7HiZ8dc	youtu.be/Fki5vIuPupE	youtube.com/shorts/Ar0gbKEfzQk	youtu.be/ciVOD8_sWR0	youtube.com/shorts/jMWtm5gh-wU
three_clust_12	youtu.be/AZv45NGkuC4	youtu.be/qQ4LqHYH_c4	youtube.com/shorts/-WtgmbfY_Qo	youtu.be/Y2sfVoemVZo	youtube.com/shorts/NsDzGdKsCyw
three_clust_13	youtu.be/U-bbZjzvaiE	youtu.be/0MznMYr5gfo	youtube.com/shorts/PtAWhAz8bz8	youtu.be/E7ge3kw5Q0Q	youtube.com/shorts/6yc5gzPi6to
three_clust_14	youtu.be/ynu2oUxv08I	youtu.be/gHDLMn5AG-8	youtube.com/shorts/OSakdYTdbmU	youtu.be/HyCJEiwCVv0	youtube.com/shorts/UJXFzkfoMH0
three_clust_15	youtu.be/xsdWsBek0eQ	youtu.be/SDY64MrcWQg	youtube.com/shorts/w7V49k4GkEI	youtu.be/CFIyW7ftF9M	youtube.com/shorts/SsyhvN6L7ks

(continued)

Data structure	Small	Small-medium	Medium	Medium-large	Large
three_clust_16	youtu.be/VyYyYOqhOVs	youtu.be/zi-TvgVR8a4	youtube.com/shorts/bRVl9y0JT8k	youtu.be/hdQmD499yo8	youtube.com/shorts/Uuy8xrnP_HU
three_clust_17	youtu.be/yojgjcf2NQk	youtu.be/UJPMJ5irRbQ	youtube.com/shorts/zI-JNpMRYxY	youtu.be/zdQYQvqTyGA	youtube.com/shorts/P4E78ewAEJs
three_clust_18	youtu.be/r-Z1Yyf2c4s	youtu.be/2Rf2L8iey2w	youtube.com/shorts/_x7kGF4xRz4	youtu.be/e_-IQycglVE	youtube.com/shorts/rY8hIqDaHKw

Table B.4: *Videos of datasets used for non-attention check DIFFERENT trials.*

Data structure	URL
three_clust_19	youtube.com/shorts/fb-gQ064JdI
three_clust_20	youtube.com/shorts/5Lm03LMiC2s
three_clust_21	youtube.com/shorts/BmKzrqTWUbI
three_clust_22	youtube.com/shorts/wrn6lj7-RrQ
three_clust_23	youtube.com/shorts/AWgG3tbfYpA
three_clust_24	youtube.com/shorts/JR_6QorZjj8
three_clust_25	youtube.com/shorts/gKEZGGZcE6c
three_clust_26	youtube.com/shorts/Ar7OAtuwWsc
three_clust_27	youtube.com/shorts/BXcLP-qqPWo
three_clust_28	youtube.com/shorts/e6cP4jC2xGM

Table B.5: *Videos of datasets used for attention check trials.*

Data structure	URL
three_clust_29	youtube.com/shorts/bqZporzHQ5U
three_clust_30	youtube.com/shorts/onAg2AgT2P4

B.3 2-*D* NLDR layouts

All 2-*D* NLDR layouts used in the experiment are available in the supplementary repository: https://github.com/JayaniLakshika/Monash_PhD_thesis/tree/main/figures/vis-exp/layouts. These include all 2-*D* embeddings generated under different NLDR methods (tSNE, UMAP, PHATE, TriMAP, and PaCMAP) with default hyper-parameter settings for the simulated 4-*D* data sets. All embedding data used to generate the 2-*D* NLDR layouts are available at https://github.com/JayaniLakshika/Monash_PhD_thesis/blob/main/data/vis-exp/embedding_data_three_clust_all.rds.

B.4 Distance metrics

To quantify cluster separation in the high-dimensional space, we considered several inter-cluster distance metrics that capture different aspects of separability (Figure B.1). Together, these metrics reflect both global separation between clusters and more local boundary proximity. All distance metrics were computed using standard implementations provided by the `fpc` (Hennig 2024) R package.

Because the metrics operate on different scales and respond differently to changes in cluster geometry, all distance-based measures were min–max scaled prior to analysis. Several metrics were additionally transformed (using exponential, square-root, or squared transformations) to improve comparability across datasets. These transformations were not intended to alter the interpretation of the measures, but rather to reduce strong nonlinearities and place the metrics on roughly similar scales.

As shown in Figure B.1, most metric pairs are strongly positively correlated, indicating that they respond similarly as cluster separation increases. This suggests that the distance scaling used in the simulations effectively controls separability and that the metrics capture related structural changes. The scatterplots also show differences in sensitivity across scaling levels, with some metrics responding more clearly at smaller separations and others providing better discrimination at larger separations.

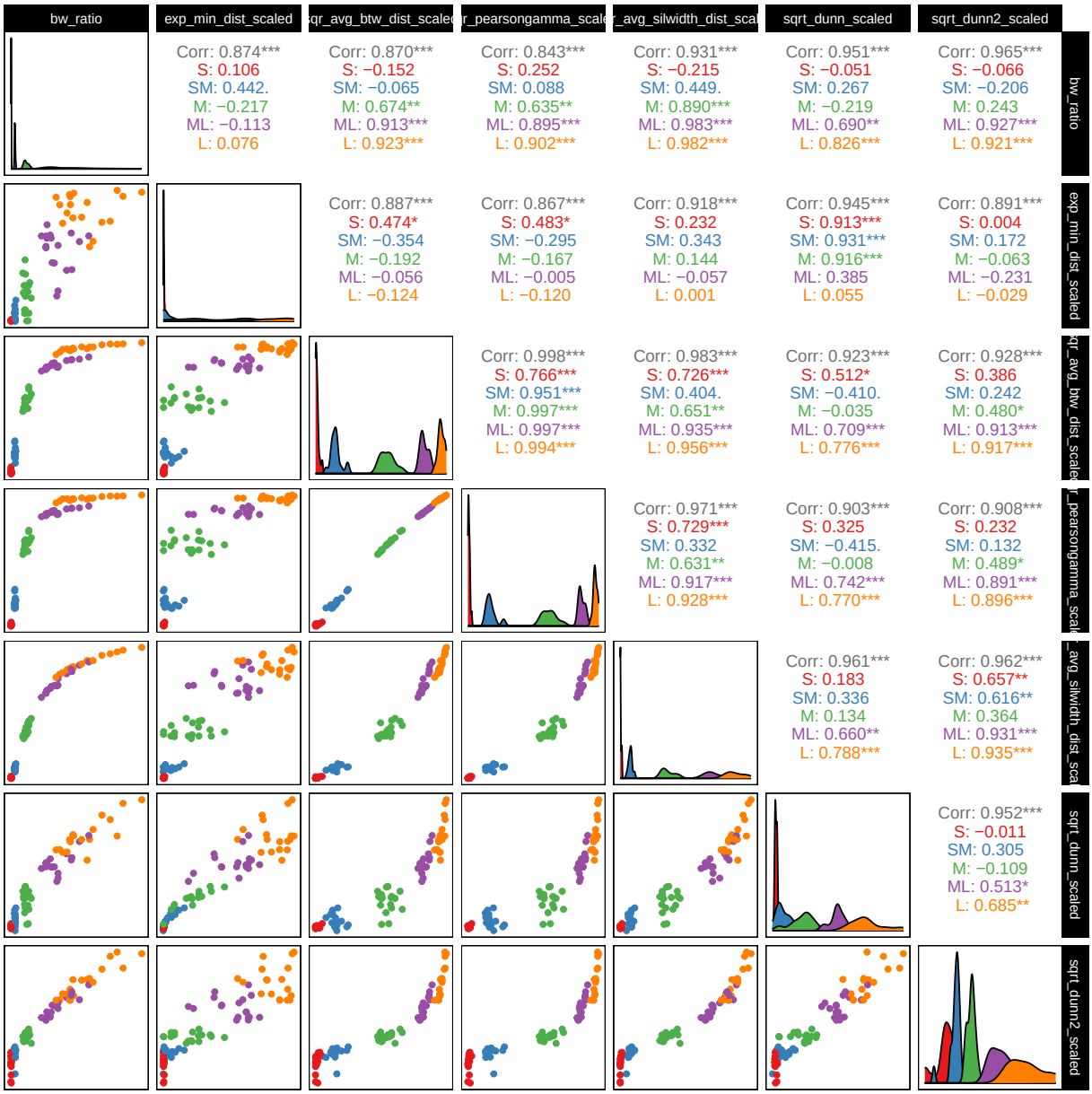

Figure B.1: *Pairwise relationships among six distance metrics used to quantify cluster separation in the high-dimensional space: between–within (BW) ratio, exponentiated scaled minimum distance, quantile-ranked average between-cluster distance, Pearson–Gamma coefficient, average silhouette distance, and square-root–transformed Dunn and Dunn2 indices. The diagonal panels show the distribution of each metric, while the lower panels show scatterplots colored by distance scaling factor (S, SM, M, ML, L). Upper panels report Pearson correlation coefficients for all pairs, with significance indicated by asterisks ($p < 0.001$ '$\star\star\star$'). Metrics show high positive correlation, confirming that they capture consistent structural variation. The BW ratio and exponentiated minimum distance were chosen for the main analysis because they provide complementary summaries of global cluster separation and local boundary distance.*

Based on these patterns, we selected the BW ratio and the exponentiated scaled minimum distance for the main analyses. The BW ratio captures overall separation by contrasting between-cluster and within-cluster dispersion, while the exponentiated minimum distance focuses on the closest boundaries between clusters. Both measures are strongly correlated with the other metrics (upper panels of Figure B.1) but reflect complementary aspects of separability, allowing us to assess whether perceptual accuracy is driven more by global structure, local proximity, or both.

B.5 Determining the number of responses per treatment

Before running the main experiment, we examined how many responses were needed for each treatment (method $\times$ distance factor) to reliably detect meaningful differences in performance. Rather than attempting to cover all possible combinations, we focused on representative comparisons that are most informative for the study. In particular, we compared UMAP and tSNE under two distance conditions (0.1 and 0.6), which showed clear differences in correct identification rates in the pilot data.

Using pilot estimates of the correct proportion, we conducted a simulation-based power analysis based on a difference in proportions framework. The baseline probability was taken from the estimated performance at the smaller distance factor (0.1), and a range of effect sizes was explored. We focused on an effect size of approximately 0.22, which corresponds to a change of about 20 percentage points in correct identification and reflects a perceptually meaningful improvement in the ability to distinguish whether two views show the same data.

The results show that (Figure B.2), for this effect size, UMAP reaches a detection probability of 0.8 with around 70 responses per condition, while tSNE requires approximately 80 responses to achieve the same level of power. This difference reflects the higher variability observed in tSNE responses compared to UMAP. Importantly, these results indicate that the number of responses collected in the main experiment (typically between 75 and 80 per condition) is sufficient to detect moderate to large effects for both methods.

B.6 Data collection process

B.6.1 Recruit subjects

Subjects were recruited from Prolific (Palan and Schitter 2018), an online platform, to evaluate the trials. The study expects that the subjects are uninvolved judges with no prior knowledge of the data

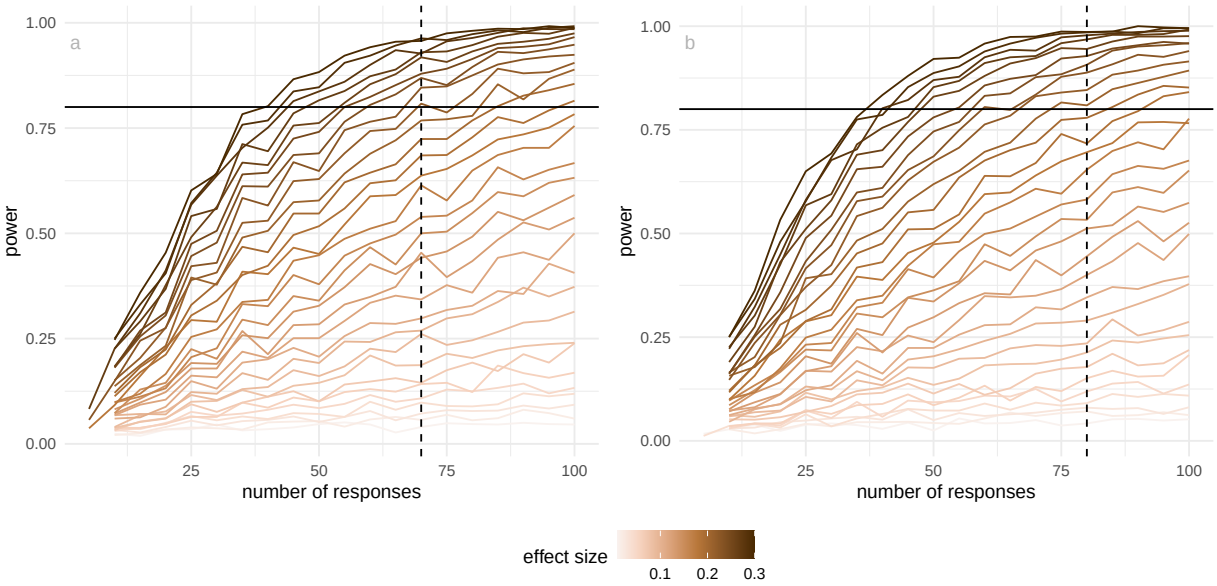

Figure B.2: *Power curves showing the relationship between the number of responses and detection probability for differences in correct identification rates between distance factors* 0.1 *and* 0.6*. Panel (a) shows results for tSNE and panel (b) for UMAP. Curves correspond to different effect sizes (difference in proportions), with darker lines indicating larger effects. The horizontal line marks the target power of* 0.8*, and the vertical dashed lines indicate the approximate number of responses required to reach this level for a moderate–large effect (≈* 0.22*). UMAP reaches the target power with fewer responses than tSNE, reflecting lower variability in participant responses.*

to avoid inadvertently affecting results. Potential subjects needed with fluent in English and have completed at least 10 Prolific studies with a 98% approval rate. The Prolific server only considers subjects who are age 18 and older.

All subjects were trained using three example displays to orient them to the evaluation trials and provided introductory materials. All subjects who completed the task were compensated 9.96 GBP per hour for their time via the Prolific payment system.

B.6.2 Web application to collect responses

The survey web application, Match-a-roo, is designed to collect survey responses and demographics using the `shiny` (Chang et al. 2025) package in R. Each subject had access to the survey via the `shiny.io server` (RStudio, PBC n.d.). The first interface of the survey app contained an introduction, instructions for the survey (Figure B.4), a consent form (Figure B.5), and buttons to access, for example, actual trials. Subjects can try three examples prior to the study where the answers were not recorded (Figure B.6). The subjects were first asked for their consent for the responses to be used for analysis.

A total of 150 participants took part in the study. Of these, 127 completed the attention check correctly,

while 23 provided incorrect responses. The analysis was therefore conducted using data from the 127 participants who passed the attention check.

After giving consent, the participant can start the trials. Two visual displays of data are shown, where the data may be the same or different (Figure B.7). One of the visual displays is a 2-*D* NLDR plot, and the other is a tour made of many 2-*D* plots. The subjects were asked to decide whether the data was the same in both displays and to report their confidence about their choice and any comments about the answer.

When the subjects completed the twenty evaluations, they were asked for their demographics, which included preferred pronoun, the highest level of education achieved, their age category, whether they used principal component analysis in their work, and whether they applied NLDR techniques such as tSNE and UMAP (Figure B.8). Finally, the subjects need to click on the prolific URL (https://app.prolific.co/submissions/) to redirect back to the Prolific app (Figure B.9).

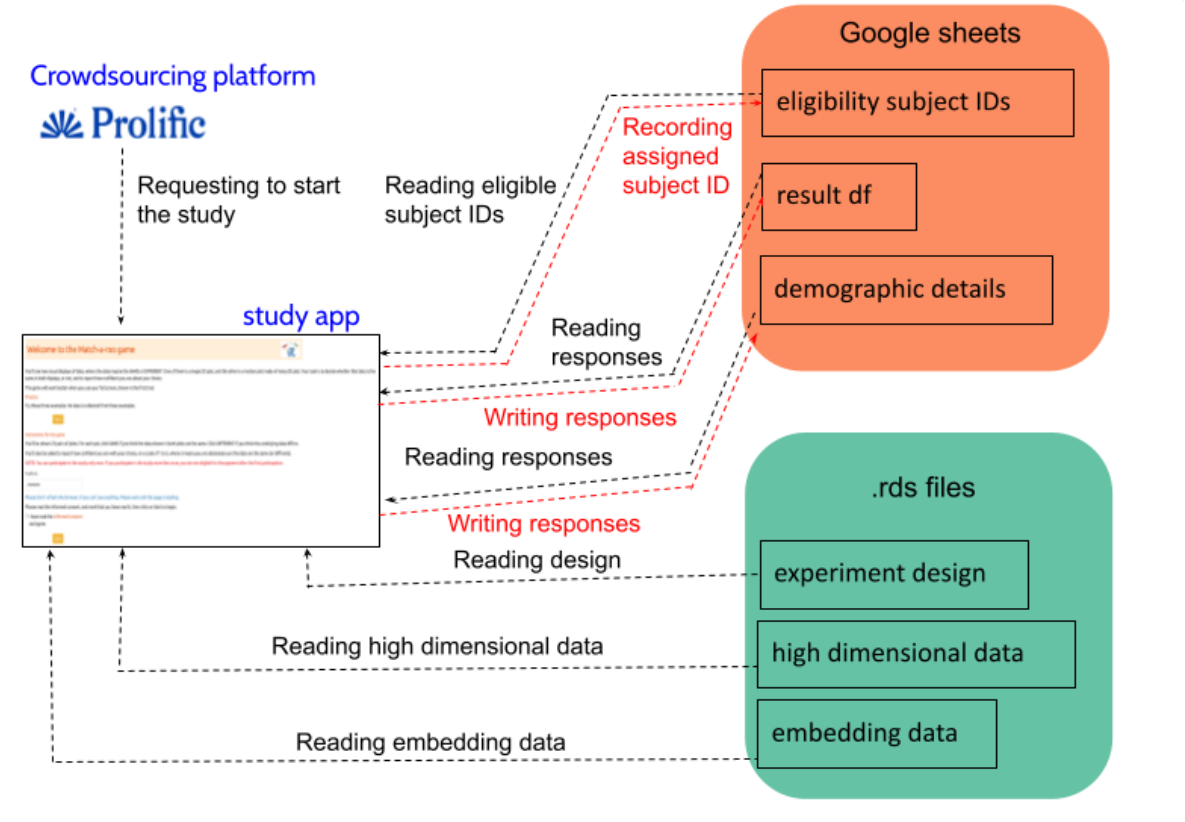

Figure B.3: *Diagram of online experiment setup.*

Once a participant starts the study (Figure B.3), the "eligibility_subject_IDs" Google Sheet is connected and read in the Shiny app to identify which subject IDs have not yet been assigned to anyone, as indicated by the "used" column. If the "used" column is marked as NA, it means that the subject ID

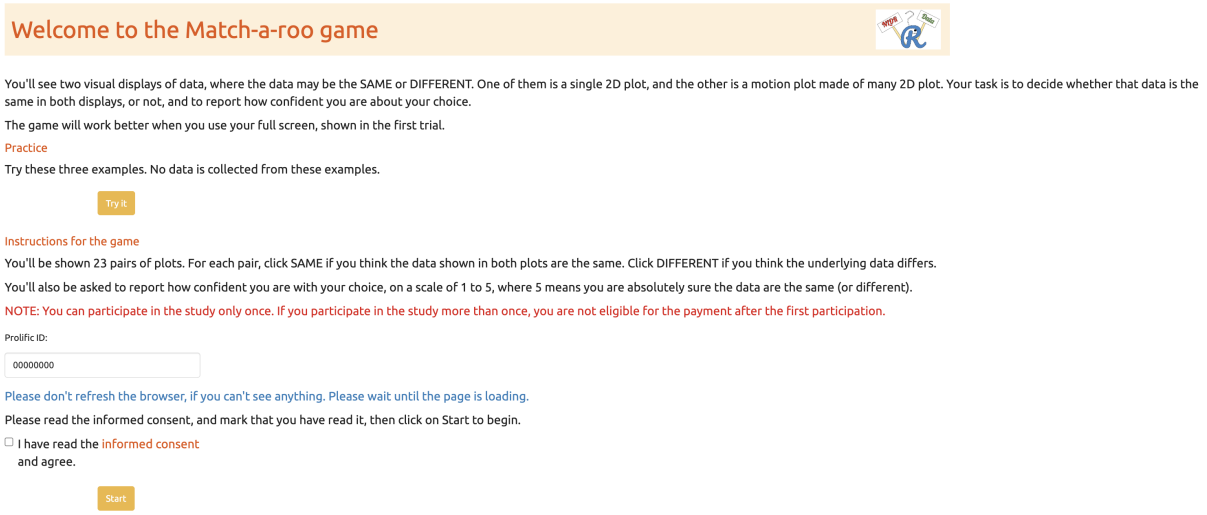

Figure B.4: *The introduction page of the study app.*

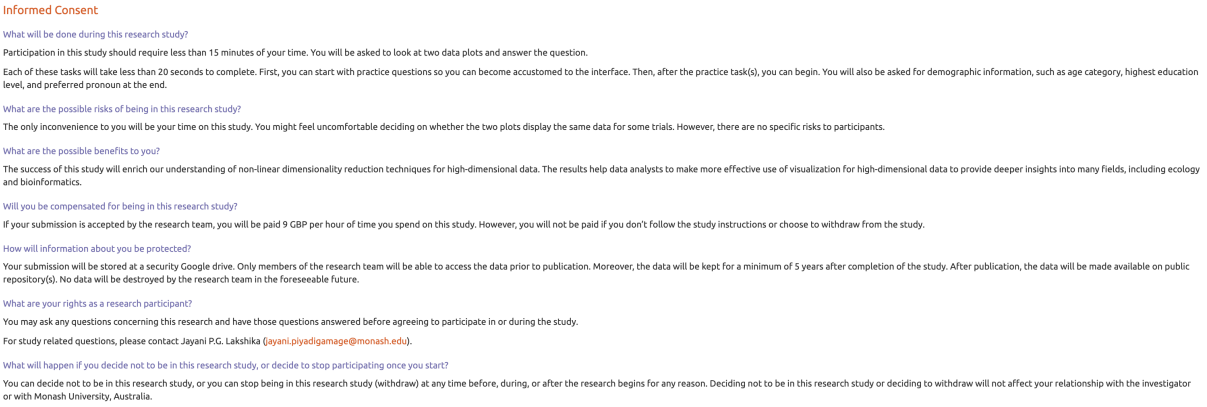

Figure B.5: *The consent form provided in the study app.*

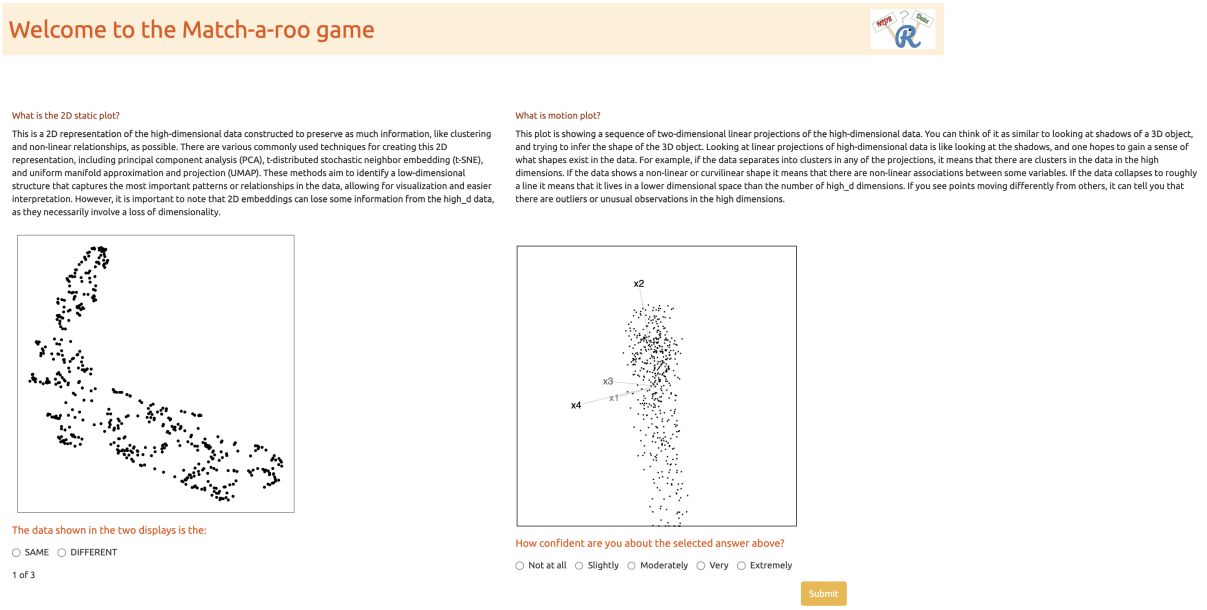

Figure B.6: *The example trial page of the study app.*

Figure B.7: *The actual trial page of the study app.*

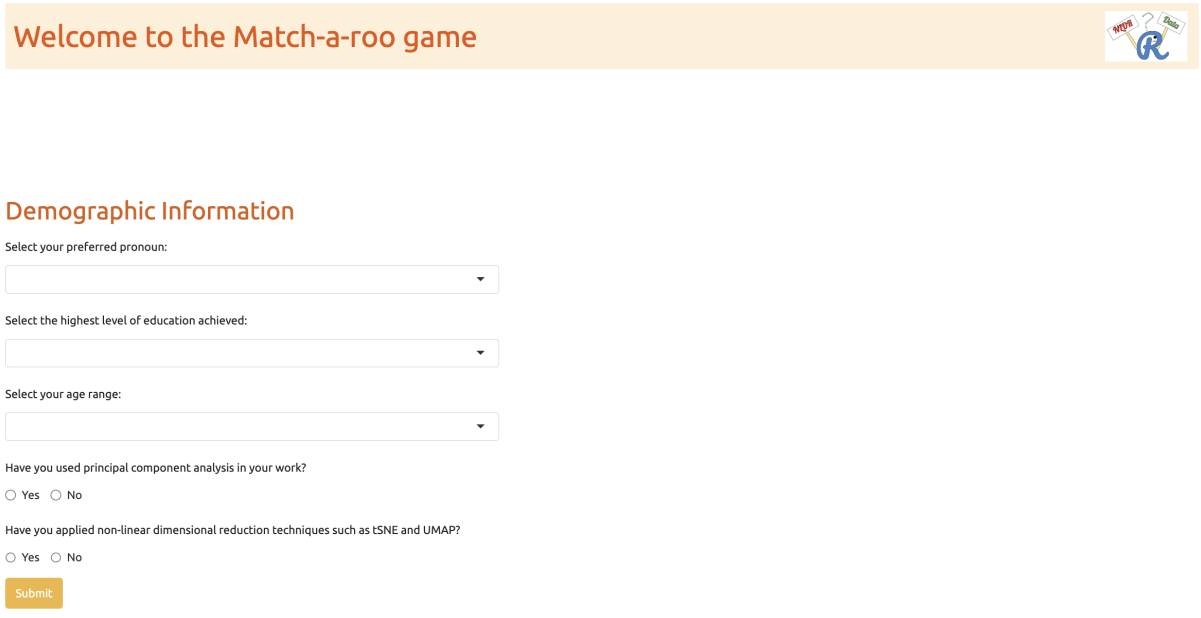

Figure B.8: *The demographics page of the study app.*

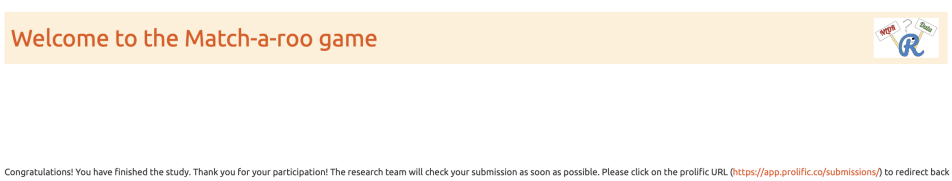

Figure B.9: *The end page of the study app.*

has not been assigned.

After identifying the eligible subject IDs, one is randomly assigned to the participant, and "1" is recorded in the "used" column corresponding to that subject ID. This subject ID will later assist in connecting the experiment design, high-dimensional data, and embedding data.

Once a subject ID is allocated to a participant, the experiment design data are loaded, and the relevant attempts, data structure, and methods are presented to the participant. This process continues until the participant completes all attempts. After determining the data structure and methods, the relevant high-dimensional and embedding data are loaded from "high_d_data_three_clust_all.rds" and "embedding_data_three_clust_all.rds", respectively, and displayed in both tour and 2-*D* NLDR plots.

Once the participant records their answers, a new row is added to the "result_df" Google Sheet with their responses. This continues until the participant finishes the study. Finally, after completing the evaluations, subjects are asked to fill out a demographics questionnaire. Their responses are then recorded in a new row of the "demographic_details" Google Sheet.

B.7 Variability across data sets and subjects

Two sources of variability in the experimental design that are important to assess relative to the fitted model: data sets and subjects. Data sets are effectively treated as replicates in the experiment, providing random samples of a range of types of clusters. Humans have different perceptual skills, which is why it is important to include a subject random effect in the model.

Across the data sets used in the experiment, the proportion of correct responses ranges from approximately 0.3 to 0.7 (Figure B.10 a). Because data sets were assigned at random, in a way unrelated to other factors in the experiment, this represents a source of variation that can safely be treated as noise.

The proportion correct across subjects is symmetric and unimodal, reasonably consistent with the assumption that they are normally distributed random effects (Figure B.10 b). Some subjects performed extremely well, and others poorly. This is similar to what has been observed in other human subject experiments involving visual tasks. A high score could be obtained by selecting SAME on each trial, but this was not the case when all their data was examined.

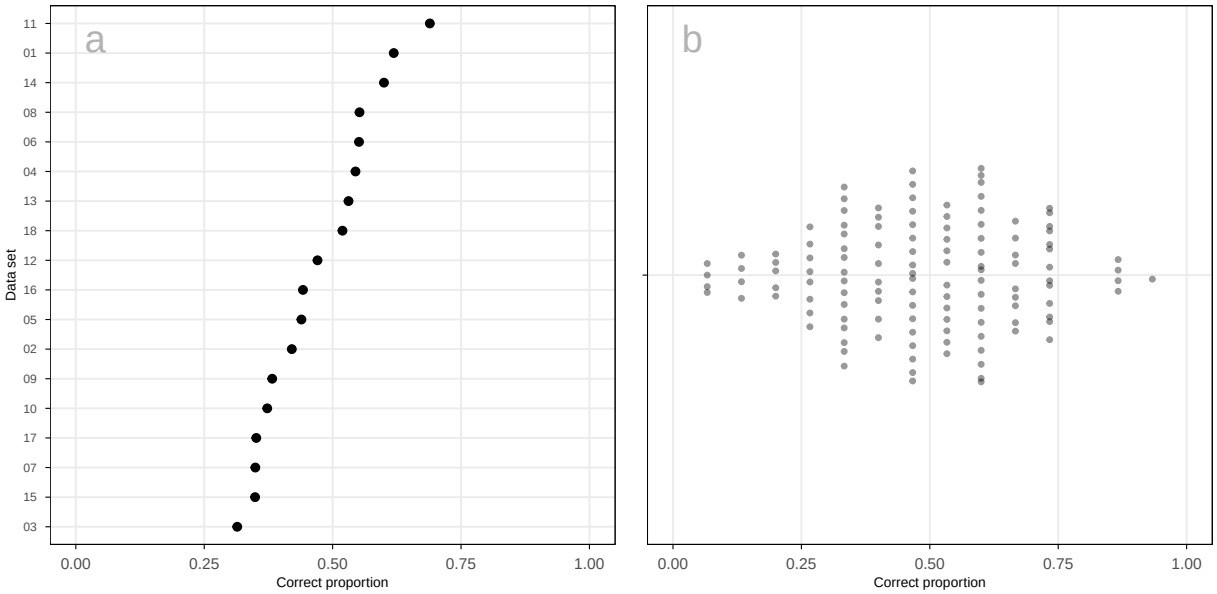

Figure B.10: *Examining the variability of proportion correct across data sets and subjects. Panel (a) shows the proportion of correct responses for each data set. The variation in correct response rates ranges from* 0.3 *to* 0.7*. Given the randomized and balanced design, this variation is largely consistent with expected replication variability and does not add a substantial amount of random noise to the overall results. Panel (b) shows the distribution of proportion correct across subjects. It is relatively Gaussian, with a few subjects performing exceptionally well and some poorly. This is consistent with other human subject experiments and reflects individual visual skills, illustrating the need to include subject-specific random effects in the model.*

B.8 Analysis of results relative to the data collection process

B.8.1 Data cleaning

The initial step in the data cleaning process involves the selection of subjects who have completed the requisite twenty trials, including the demographics and the attention check trial. Subjects who exceeded the average time of $5-10$ minutes were excluded, as determined from the pilot study. Following this, individuals who didn't accurately detect the attention check trial were also removed. Furthermore, the attention check trials were removed, as they did not contribute to the further analyses. Finally, the collected data set is further refined by filtering out all the responses which showed the same data structures in 2-*D* NLDR plot and tour.

B.8.2 Demographics

Along with the responses to the trials, we have collected a series of demographic information, including preferred pronoun, age range category, educational background, and previous experience in PCA and Non-linear dimension reduction techniques. Table B.6, Table B.7, Table B.8, Table B.9, and Table B.10 provide summaries of the demographic data.

Table B.6: *Summary of the pronoun distribution of subjects recruited for this study.*

Pronoun	Period I	Period II	Total	%
he/him	7	54	61	48.03
she/her	11	53	64	50.39
they/them	0	2	2	1.57
Total	18	109	127	100.00

Table B.7: *Summary of the age distribution of subjects recruited for this study.*

Age group	Period I	Period II	Total	%
18 - 24	3	23	26	20.47
25 - 34	9	36	45	35.43
35 - 44	3	22	25	19.69
45 - 54	1	12	13	10.24
Over 55	2	16	18	14.17
Total	18	109	127	100.00

The subjects are fairly balanced in terms of pronouns, with similar proportions identifying as *she/her* (50.4%) and *he/him* (48.0%), and a small number identifying as *they/them* (1.6%). Subjects cover a wide age range, with most between 25 and 34 years old (35.4%), followed by those aged $18-24$ (20.5%) and $35-44$ (19.7%). The sample has more younger and mid-adult age groups, while still including representation from older subjects.

Most subjects have completed an undergraduate degree (44.9%) or a postgraduate qualification (26.8%), with others reporting some undergraduate study (21.3%). Only a small proportion did not complete high school. Prior experience with dimension reduction methods is limited: the majority report no previous experience with PCA (84.2%) or nonlinear dimension reduction techniques (86.6%). This suggests that most subjects approached the task without strong prior familiarity, allowing the results to reflect general perceptual interpretation rather than expert knowledge.

Table B.8: *Summary of the educational distribution of subjects recruited for this study.*

Education	Period I	Period II	Total	%
Completed some undergraduate courses	4	23	27	21.26
Did not complete high school	0	4	4	3.15
Higher degree master or doctorate	3	31	34	26.77
Prefer not to answer	3	2	5	3.94
Undergraduate degree (A bachelor)	8	49	57	44.88
Total	18	109	127	100.00

Table B.9: *Summary of the previous experience in PCA of subjects recruited for this study.*

Experience with PCA	Period I	Period II	Total	%
No	15	92	107	84.25
Yes	3	17	20	15.75
Total	18	109	127	100.00

Table B.10: *Summary of the previous experience in Nonlinear dimension reduction techniques of subjects recruited for this study.*

Experience with NLDR	Period I	Period II	Total	%
No	15	95	110	86.61
Yes	3	14	17	13.39
Total	18	109	127	100.00

Appendix C

Academic Contributions and Professional Engagement

C.1 Planning and design software

In addition to the completed methods and software presented in this thesis, a large amount of exploratory planning and design work went into the development of the R packages `quollr` (Figure C.1) and `cardinalR` (Figure C.2), as well as the Shiny application `menuraR` (Figure C.3). This includes personal working sheets, sketches, and early conceptual diagrams that show how initial ideas gradually evolved into the implemented software tools.

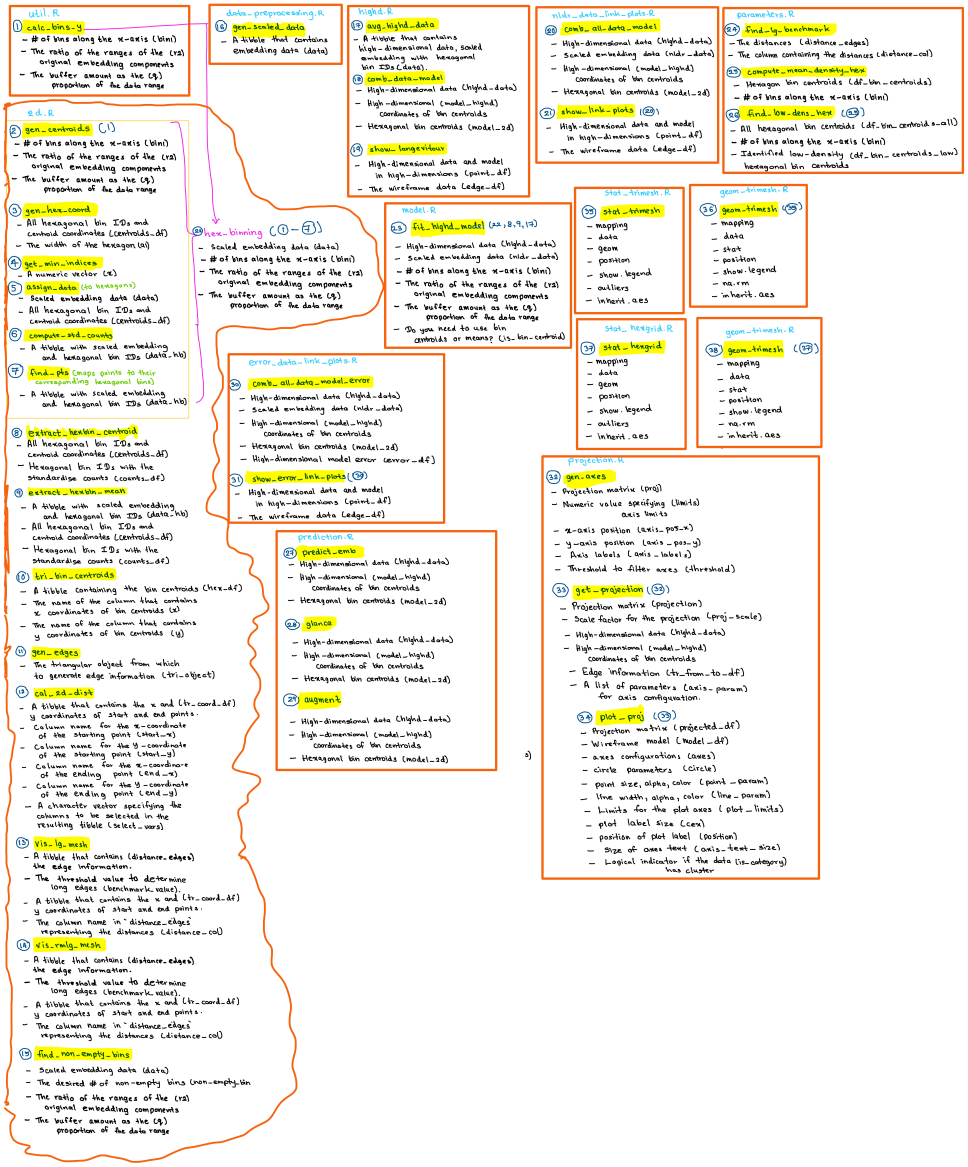

Figure C.1: *Working sheets used during the planning and development of* `quollr`*, showing how early ideas evolved into a diagnostic R package.*

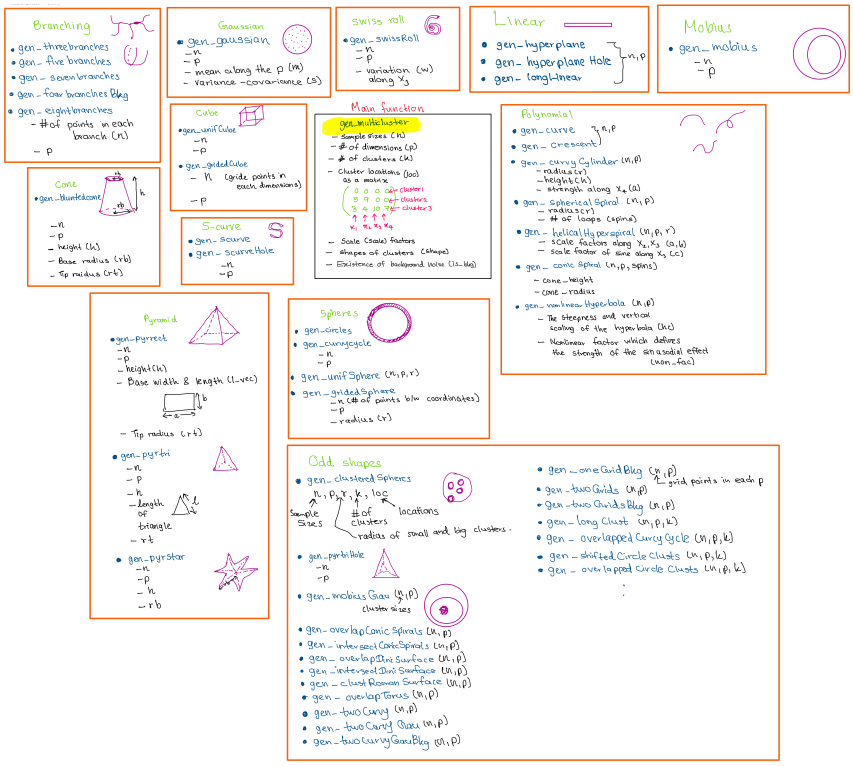

Figure C.2: *Working sheets used during the planning and development of* `cardinalR`*, documenting the evolution of data generation strategies into software.*

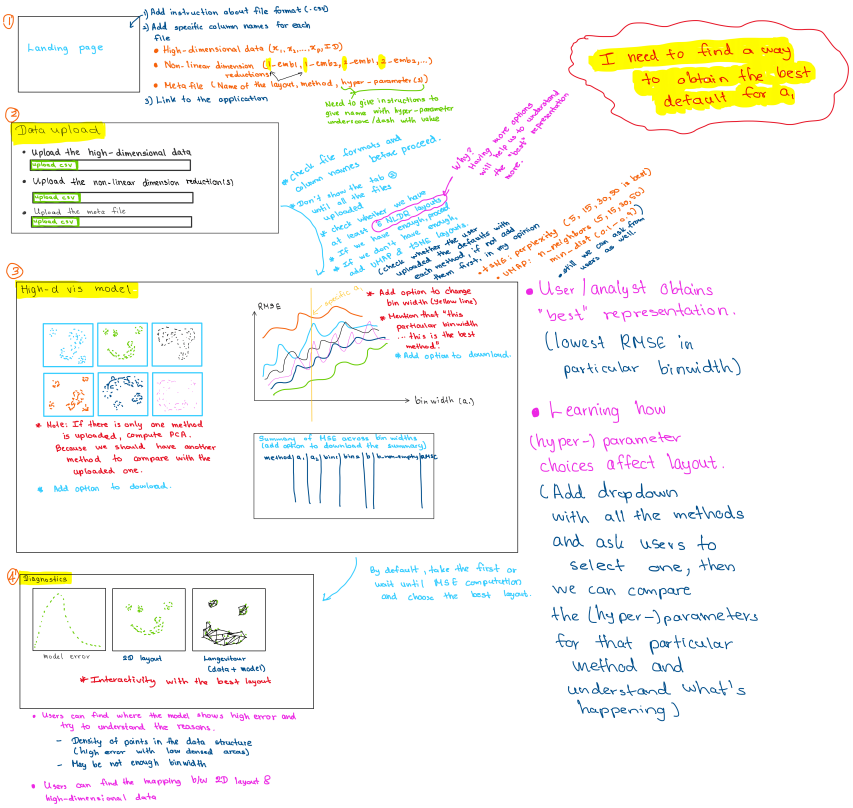

Figure C.3: *Working sheets used in the planning and design of* `menuraR`*, showing how initial concepts were refined into a functional Shiny application.*

C.2 Software names

Each software name is inspired by an animal. `quollr` is named after the **quoll**, a carnivorous, curious, and endangered marsupial from Australia. `cardinalR` is inspired by the North American **cardinal** bird. `menuraR` comes from Australia's lyrebirds (**Menura**), famous for their elaborate courtship displays and extraordinary ability to mimic sounds.

C.3 Presentations

I presented my research work at 12^{th}-Conference of the Asian Regional Section of the International Association for Statistical Computing (IASC-ARS 2023) (Wollongon, Australia), Australian Statistical Conference (ASC 2023) (Wollongon, Australia), Bioinformatics Seminar 2024, Victorian branch of the Australian and New Zealand Industrial and Applied Mathematics Society (VicANZIAM) 2024 (RMIT university, Melbourne, Australia), Faculty of BusEco Three Minute Thesis (3MT) competition 2024, useR! 2024 (Salzburg, Austria), Graphics Group Presentation 2024 (Nebraska, USA), UNO Data Science Club 2024 (Omaha, USA), Joint Statistical Meetings (JSM) 2025 (Nashville, USA), useR! 2025 (Durham, USA), Biometrics in the Bush Capital (BIBC2025) (Canberra, Australia), and Australian Statistical Conference (ASC 2025) (Perth, Western Australia) (Figure C.6).

C.4 Visiting

In July 2024, I had the privilege of visiting A/Prof Ursula Laa at the University of Natural Resources and Life Sciences, Vienna (BOKU University), accompanied by Prof Di Cook, Prof Eun-Kyung Lee, and Dr Natalia da Silva. During this visit, I engaged with academic staff, students, and fellow visitors at BOKU University, gaining valuable insights into their research and receiving constructive feedback on my work and its potential contributions to ongoing projects (Figure C.6 Vienna).

From late October to late December 2024, I visited Prof Heike Hofmann, A/Prof Susan VanderPlas, and Dr Michelle Graham at the University of Nebraska, Lincoln, USA (UNL) (Figure C.6 Nebraska). During this time, I presented my research on high-dimensional data visualization and dimension reduction techniques, participated in the Nebraska R User Group meetings, and joined discussions with the Graphics Group, which provided rich opportunities for collaboration and learning.

These visits were invaluable for broadening my perspective, fostering meaningful exchanges with experts, and deepening my understanding of dynamic visualization and multivariate data analysis.

I also explored several resources that informed my work, including research on dynamic tours for high-dimensional data, parallel coordinate plots, perceptual accuracy in visualizations, and interactive visualization tools such as *langevitour* and *tourr*.

C.5 Academic service & community engagement

During my PhD, I contributed to the academic and statistical communities through service, leadership, and outreach, supporting inclusive research and knowledge exchange. My roles include NUMBATs Seminar Organizer (Monash University, 2025), Session Chair at useR! 2024 (Salzburg) and ASC 2023 (Wollongong), Tutorial Helper for WOMBATs Tutorials (Monash University, 2022), and organizer for R-Ladies Melbourne (2023). These activities let me connect with diverse audiences, support early-career researchers, and share ideas about stats and computational methods.

C.6 Workshops

I have been part of delivering and preparing materials for workshops on *Reproducible Reporting and Research with Quarto* (September 2025) and *Reproducible Reporting, Academic Papers, Presentations, and Theses with Quarto* (July 2025), contributing to hands-on training for researchers on reproducible practices and effective research communication (Figure C.4).

Figure C.4: *Moments from delivering the Reproducible Reporting and Research with Quarto workshop in September 2025, highlighting interactive, hands-on training in reproducible research and academic writing.*

C.7 Mentoring

In July 2025, I had the privilege of serving as a coach in the Monash Innovation Guarantee Postgraduate (MIG-P) program (Figure C.5). Over three inspiring weeks, I worked with a diverse cohort of master's students as they tackled real-world, industry-defined challenges. It was an incredible experience to support their journey from exploration and ideation through to prototyping and pitching—witnessing their creativity, resilience, and ability to thrive in ambiguity.

Figure C.5: *Moments from mentoring master's students in the Monash Innovation Guarantee Postgraduate (MIG-P) program (July 2025), highlighting collaboration, creativity, and team-based problem solving.*

C.8 Additional contributions

I contributed to open-source software development by co-supervising the creation of the `polarisR` package (Yadav et al. 2025) during a Google Summer of Code project with Dr. Ursula Laa and Prof. Eun-Kyung Lee, whom I met during my visit to the University of Natural Resources and Life Sciences in Vienna, Austria. `polarisR` is a Shiny application for diagnosing 2-*D* NLDR layouts using the `quollr` implementation. It also supports comparing how the data appear in high dimensions through various tour methods, including scatter, sage, and slice.

C.9 Teaching

I have contributed to teaching a range of undergraduate and postgraduate courses in statistics, data analysis, and machine learning. These include *Statistical Thinking* ([ETC5242], 2025; [ETC2420],

2025), *Introduction to Data Analysis* ([ETC5510], 2024; [ETC1010], 2024), *Introduction to Machine Learning* ([ETC3250], 2023–2024; [ETC5250], 2023–2025), and *Exploratory Data Analysis* ([ETC5521], 2023).

C.10 Final thoughts

This journey has been as much about exploring the unknown as it has been about developing resilience and insight along the way. I am deeply grateful for the people, places, and lessons that have shaped both this work and the path forward.

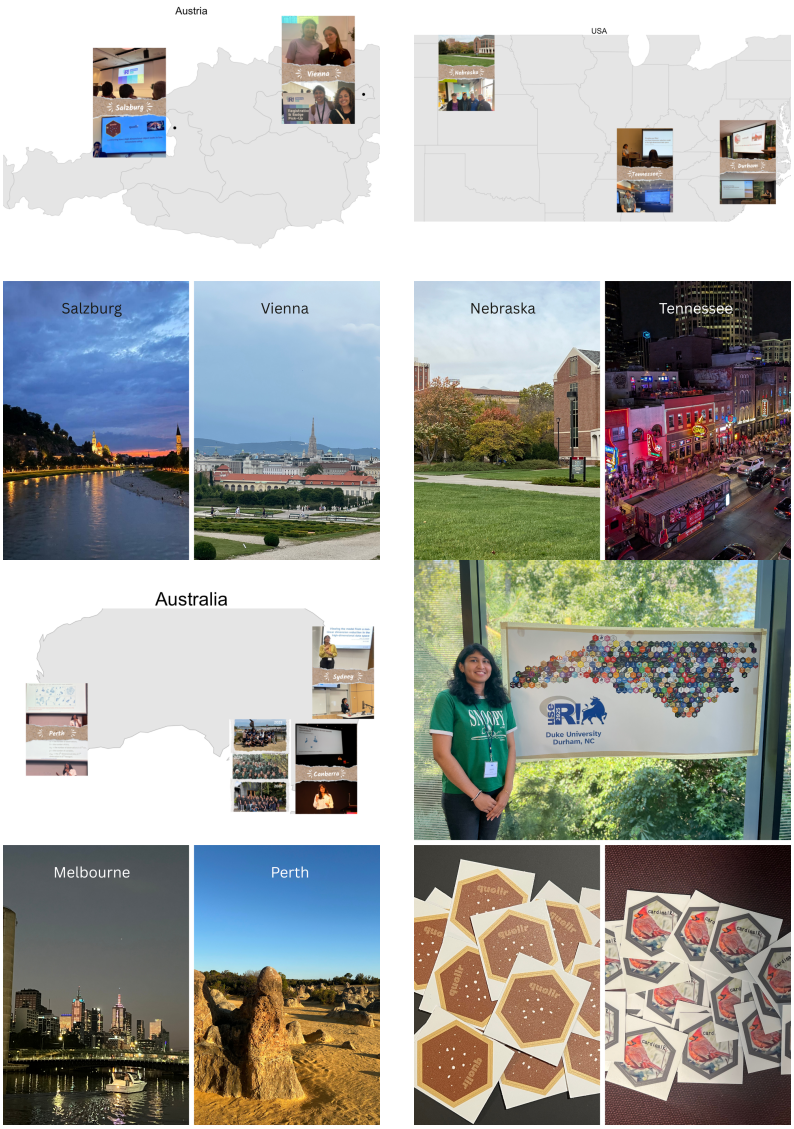

Figure C.6: *Geographic footprint of the PhD journey, highlighting research visits, conferences, and academic engagements across Australia, Austria, and the United States. Locations include Salzburg and Vienna (Austria), Nebraska and Tennessee (USA), and Melbourne, Sydney, Canberra, and Perth (Australia), alongside moments from conferences, collaborations, and software dissemination.*

Appendix D

Glossary

Table D.1: *Glossary.*

Term	Description
2-D model	The fitted representation of the NLDR layout obtained after hexagonal binning and centroid aggregation, used as the basis for lifting into high-dimensional space.
a_1 (Hexagon width)	The horizontal distance between adjacent hexagon centroids in the hexagonal grid used for binning a 2-D NLDR layout.
a_2 (Hexagon height)	The vertical distance between rows of hexagon centroids in the hexagonal grid, related to a_1 by $a_2 = \sqrt{3}a_1/2$.
b_1 (Number of bins along x-axis)	The number of hexagon columns spanning the horizontal range of a 2-D embedding.
b_2 (Number of bins along y-axis)	The number of hexagon rows spanning the vertical range of a 2-D embedding.
k-means clustering	A clustering algorithm that partitions data into a fixed number of clusters by minimizing within-cluster variance.
2NC7 data	A simulated 7-D dataset consisting of two nonlinear clusters with different intrinsic dimensions and added noise variables.
Active (binding) constraint	In optimization, a constraint that holds with equality at the optimal solution and determines the final parameter values.

(continued)

Term	Description
Alt-text	Textual descriptions of figures and visual content that support accessibility, particularly for screen readers and users with visual impairments.
Anchor point	The center of a hole that is removed from a dataset. Often the data mean, but can be user-defined.
Apex	The tip or pointy end of a shape (like the top of a cone or pyramid) where many points can be concentrated.
Application (example study)	A worked example demonstrating how cardinalR can be used to generate data, apply dimension reduction and clustering methods, and evaluate their performance.
Archimedean spiral	A spiral where the distance from the center increases steadily as it winds outward.
Area under the RNX curve (ARNX)	A summary measure of neighborhood preservation, balancing local and global structure.
Aspect ratio	The ratio of the ranges of the NLDR axes, preserved when constructing hexagonal binning grids.
Attention check/ trial	A trial included to verify that participants are paying attention. These trials use very clear data structures where the correct answer should be obvious.
Azimuthal angle	The angle that controls rotation around an axis (like longitude on a globe).
Background noise	Additional points drawn from a distribution that do not belong to any specific geometric structure or cluster, used to simulate unstructured variation in the data.
Ball	A filled-in sphere. Points occupy the whole volume, not just the surface.
Barycentric coordinates	A way of picking points uniformly inside a triangle by mixing the triangles corners with random weights.

(continued)

Term	Description
Baseline probability	The estimated probability of a correct response under a reference condition, used as the starting point in power analysis. In this study, it corresponds to performance at the smallest distance factor (0.1).
Benchmark datasets/ structure	Standard or reference datasets used to evaluate, compare, and validate the performance of analytical algorithms (e.g. clustering or dimension reduction).
Between-cluster distance	A measure of separation between clusters, typically based on distances between cluster centroids or boundaries.
Between-to-within (BW) ratio	A measure of cluster separability comparing how far clusters are from each other (between-cluster variation) relative to how spread out points are within clusters. Higher values indicate better-separated clusters.
Bin (hexagon)	A spatial region in a hexagonal grid used to aggregate points from a 2-*D* NLDR embedding.
Bin centroid	The center point representing a hexagonal bin, either defined geometrically or computed as the mean of observations within the bin.
Binheight (a_2)	The vertical height of a hexagon, determined by the geometry of hexagonal tiling.
Binning function	A mapping that assigns each observation in the 2-*D* layout to its nearest hexagon centroid.
Binwidth/ Bin width (a_1)	The horizontal width of a hexagon, controlling the resolution of the hexagonal grid.
Blunted apex	A tip thats flattened or rounded instead of sharp.
Branching structure	A connected geometric structure consisting of multiple arms or trajectories that diverge from a common origin, often used to represent bifurcation or developmental processes.

(continued)

Term	Description
Browser memory limits	Constraints imposed by web browsers on the amount of memory available for client-side rendering and interaction, which can affect performance when visualizing large datasets in web-based applications.
Brushing / Linked brushing	An interactive technique where selecting points or regions in one view (e.g., a 2-*D* layout) highlights the corresponding points in another view (e.g., high-dimensional model or tour).
Buffer parameter (q)	A proportional margin added around the data range to ensure the hexagon grid fully covers the NLDR layout.
C-shaped cluster	A nonlinear cluster with observations arranged along a curved manifold resembling the letter C.
Centroid	The mean position of all points in a cluster. Centroids are used to control and measure distances between clusters in the simulations.
Cluster	A group of data points generated from the same underlying geometric shape or distribution, representing a coherent structure in the dataset.
Cluster separability/ separation	The degree to which clusters are distinct from one another in the data space, quantified using distance-based metrics.
Cluster validity statistic	A numerical measure used to assess the quality of a clustering solution, often balancing within-cluster compactness and between-cluster separation.
Clustered spheres	A structure made of one large sphere plus several smaller spheres placed around it, each treated as a separate group.
Clustering	The task of grouping observations so that points within the same group are more similar to each other than to points in other groups.
Clustering algorithm	A method that groups observations into clusters based on similarity, without using class labels.

(continued)

Term	Description
Cone	A shape that narrows toward one end. In high dimensions, its formed by shrinking hyperspherical cross-sections along one axis.
Cone cluster	A cluster whose density varies along one axis, typically denser near the apex and more diffuse toward the base.
Confidence rating	A self-reported measure indicating how confident a participant is in their judgment for a given trial.
Conic spiral	A spiral that expands outward and upward, forming a cone-like helix.
Convex hull	The smallest convex set that contains all points in a dataset; referenced as inspiration for extending binning to higher dimensions.
Correct identification rate / proportion correct	The proportion of trials in which participants correctly judged whether the 2-*D* NLDR plot and the tour represented the same data.
Covariance structure (EII)	A model-based clustering assumption where clusters are spherical, have equal volume, equal shape, and no orientation differences.
Crescent	A curved, moon-shaped arc formed by points along part of a circle.
Cube / Hypercube	Points filling a square (2-*D*), cube (3-*D*), or higher-dimensional box, either on a grid or randomly.
Curvy cycle	A closed loop that isnt a simple circle, with extra folds or oscillations.
Cylinder (curvy)	A cylindrical shape with circular cross-sections, extended with a nonlinear bending dimension.
DIFFERENT trials	Trials in which the 2-*D* NLDR embedding and the tour are generated from different (but related) high-dimensional datasets. These trials act as controls to prevent trivial response strategies.

(continued)

Term	Description
Data availability	The practice of making datasets publicly accessible to support transparency, reproducibility, and independent validation of results.
Data set	A simulated configuration of points in high-dimensional space with predefined geometric shapes, densities, and cluster arrangements.
Data set variability	Variation in performance attributable to differences among the simulated data sets, such as cluster shape, size, or arrangement. In this experiment, data sets act as replicates and contribute random noise to the response.
Delaunay triangulation	A geometric method that connects points into triangles such that no point lies inside the circumcircle of any triangle, used to define neighborhood structure.
Demographics questionnaire	A set of questions collecting background information about participants, such as age range, education level, and prior experience with dimension reduction methods.
Diffusion process	A mathematical process modeling how information spreads across a graph or manifold, used in NLDR to capture intrinsic geometry.
Dimension reduction (DR)	A technique that maps high-dimensional data into a lower-dimensional space while attempting to preserve important structure.
Distance scale factor	A multiplier applied to centroid distances to control how close or far apart clusters are. Levels include small, smallmedium, medium, mediumlarge, and large.
Dynamic visualization	Visualization techniques that use motion or interaction (e.g., tours, brushing, animation) to explore high-dimensional or complex data structures.

(continued)

Term	Description
Effect size	The magnitude of a difference in performance between two experimental conditions. Here, it is defined as a difference in proportions, with an effect size of approximately 0.22 corresponding to a 20 percentage-point increase in correct identification.
Embedding	A low-dimensional representation (typically 2-*D*) of high-dimensional data produced by an NLDR method such as tSNE or UMAP.
Exploratory planning	Early-stage conceptual work involving sketches, diagrams, and working notes that guide the development of methods and software.
Exponential distribution (truncated)	A distribution that produces many small values and few large ones, here limited to a fixed range.
Exponential scaled minimum inter-cluster distance	A transformed version of the smallest distance between any two points from different clusters, used to emphasize differences in close cluster proximity.
Fitted values	The high-dimensional bin centroids associated with observations, treated as model-predicted values.
Fractal (Sierpinski-like)	A self-similar pattern with repeating holes or gaps, created using a recursive rule.
Gaussian cloud/ cluster	A cluster of points generated from a multivariate normal distribution, typically dense in the center and sparse at the edges.
Gaussian noise	Random variation drawn from a normal distribution, often added to make data more realistic.
Generalized linear mixed-effects model (GLMM)	A statistical model that accounts for both fixed effects (e.g., NLDR method, distance) and random effects (e.g., participant variability). Used here to model correct identification probabilities.

(continued)

Term	Description
Geometric relationships	Spatial relationships among data points (such as distances and angles) in the original high-dimensional space.
Geometric shape	A mathematically defined structure (e.g., Gaussian, cone, sphere, cube, spiral) used as a building block for generating synthetic data.
Git	A distributed version control system used to track changes in code, documents, and research materials.
GitHub	A web-based platform for hosting Git repositories, enabling collaboration, version control, and public dissemination of software and research materials.
Global Score (GS)	A metric measuring preservation of overall geometry relative to a PCA baseline.
Global structure	Large-scale relationships in the data, such as relative positions and distances between clusters.
Grid-based structure	Points placed in a regular, evenly spaced pattern instead of randomly.
Hallucinated structure	Patterns observed in an NLDR layout that do not correspond to true structure in the original high-dimensional data.
Helical spiral	A twisted, elongated structure that winds around an axis while progressing forward.
Hemisphere	Half of a sphere, created by restricting angles so points lie on only one side.
Hexagon (bin)	A single hexagonal cell in the tessellation of the 2-D NLDR layout.
Hexagon grid	A tessellation of the 2-D embedding space into regular hexagons used for binning and model fitting.
Hexagonal binning (hexbin/ hexbinning)	A spatial aggregation method that partitions a $2-D$ layout into hexagonal cells to summarize local structure and density.

(continued)

Term	Description
Hexbin Error (HBE)	A diagnostic metric that measures how well a 2-*D* NLDR layout represents the underlying high-dimensional data by comparing bin centroids lifted back into high-dimensional space. Lower HBE indicates a better representation.
Hierarchical clustering	A clustering method that builds a tree of nested clusters by successively merging or splitting groups.
High-dimensional centroid	The mean of the high-dimensional observations assigned to a given hexagonal bin.
High-dimensional data (*p*-*D* data)	Data in which each observation is described by a large number of features (dimensions), often making direct visualization and interpretation difficult.
High-dimensional noise	Extra dimensions added to data that introduce variability without changing the main structure.
High-dimensional space (*p*-*D*)	The original data space where each observation is represented by *p* variables.
Hole (spherical / hyperspherical)	A region removed from the data in the shape of a circle, sphere, or higher-dimensional sphere.
Hyper-parameter(s)	A method-specific tuning parameter (e.g., perplexity in tSNE, number of neighbors in UMAP) that influences the resulting NLDR embedding.
Hypersphere	The higher-dimensional equivalent of a circle (2-*D*) or sphere (3-*D*).
Interdisciplinary users	Researchers or students from diverse disciplinary backgrounds who may have varying levels of programming or statistical expertise.
Intrinsic dimensionality	The effective dimensionality of a data structure, independent of the ambient number of variables.
Inverse transformation	A nonlinear operation involving division (e.g. 1/x1/x1/x) that creates sharp curvature.
Isotropic distribution	A distribution with equal variability in all directions, such as points uniformly sampled in a cube.

(continued)

Term	Description
Latent parameter	An underlying variable (like an angle or index) that drives the shape of the data.
Latent variable	An underlying variable (like an angle or time index) that drives the observed structure but isnt directly observed.
Lifted model (*p-D* model)	The representation of the 2-D wireframe model mapped back into the original high-dimensional space by averaging observations within each bin.
Linear optimization problem	An optimization problem where the objective and constraints are linear functions, implying solutions occur at vertices of the feasible region.
Linear projection	A mapping from high- to low-dimensional space using linear combinations of the original variables.
Linear structure	Points arranged roughly along a straight line, possibly with noise and different scales across dimensions.
Linked plots/views	Interactive visualizations where selections in one view (e.g., 2-D layout) are reflected in other views (e.g., tours or error plots).
Local structure	Relationships among nearby points in high-dimensional space, such as nearest neighbors within a cluster.
Localglobal trade-off	The balance between preserving small-scale neighborhood structure and large-scale relationships in NLDR embeddings, controlled by hyper-parameters.
Low-count bin removal	The process of excluding bins with few observations to sharpen the wireframe representation.
Low-density hexagon	A bin containing few points and whose neighboring bins also have low density, indicating weak local support for structure.
Low-dimensional representation/ embedding	A reduced-dimensional embedding (typically 2-*D*) of high-dimensional data used for visualization and interpretation.
Manifold	A low-dimensional shape (curve or surface) embedded inside a higher-dimensional space.

(continued)

Term	Description
Match-a-roo	A Shiny-based web application developed to collect participant responses, confidence ratings, and demographic information for the user study.
Minimum inter-cluster distance	The smallest distance between any point in one cluster and any point in another cluster. It captures the closest approach of clusters.
Misidentification	A participant response indicating that two displays show different data when they actually show the same data, or vice versa.
Model diagnostics	Visual and quantitative tools used to assess how well an NLDR-based model fits the high-dimensional data, including error views, tours, and linked brushing.
Model fitting pipeline	The sequence of steps in quollr, including scaling, hexagonal binning, centroid extraction, triangulation, lifting into p-D, and diagnostic computation.
Model-based clustering	A probabilistic clustering approach that assumes data are generated from a mixture of distributions.
Model-in-the-data-space	A visualization principle in which a fitted model is overlaid directly on the observed data in the original high-dimensional space to assess model fit.
Multicluster dataset	A dataset composed of multiple clusters, each potentially generated from a different geometric shape or distribution.
Multidimensional scaling (MDS)	A family of methods that create low-dimensional representations by preserving pairwise distances from high-dimensional space.
Mbius strip	A twisted surface with only one side and one edge, often used to test how algorithms handle non-orientable geometry.
NLDR (Nonlinear Dimension Reduction)	A class of methods that project high-dimensional data into lower dimensions while preserving nonlinear structure.

(continued)

Term	Description
NLDR layout	The two-dimensional embedding produced by an NLDR method (e.g., tSNE, UMAP), where each point represents a high-dimensional observation.
NLDR selection	The process of choosing one or more nonlinear dimension reduction layouts that best represent the structure of high-dimensional data, based on visual and quantitative diagnostics.
Nearest-neighbor ordering	A possible effect of NLDR methods where points are arranged in a way that preserves local neighborhoods but may impose unintended global ordering.
Neighborhood preservation	The extent to which local proximity relationships among observations are maintained between high- and low-dimensional spaces.
Neighborhood structure	The pattern of local adjacency relationships among bins or points in the 2-*D* layout.
Noise dimensions	Additional variables added to a dataset, typically drawn from random distributions, to increase dimensionality without adding structure.
Non-empty bin	A hexagonal bin containing at least one observation.
Nonlinear dimension reduction (NLDR)	A class of methods that map high-dimensional data into a lower-dimensional space using nonlinear transformations, often to reveal structure not visible through linear projections.
Nonlinear geometry	Data structures that cannot be adequately represented by linear projections, such as spirals or curved surfaces.
Nonlinear surface	A warped 2-*D* surface embedded in higher dimensions, showing bends, waves, or sharp changes.
Orthogonal rotation	A transformation that rotates data while preserving distances and overall shape.
PBMC dataset	A single-cell RNA-seq dataset of peripheral blood mononuclear cells commonly used to benchmark dimension reduction and clustering methods.

(continued)

Term	Description
PHATE (Potential of Heat-diffusion for Affinity-based Trajectory Embedding)	An NLDR method based on diffusion processes, designed to capture both global geometry and continuous transitions in the data.
PaCMAP (Pairwise Controlled Manifold Approximation)	An NLDR method that uses different types of point pairs to control local, mid-range, and global structure preservation.
Pancake effect	An observed artifact where a fitted model collapses into a near-flat structure in high-dimensional space, indicating loss of intrinsic dimensionality.
Participant (subject)	An individual recruited to take part in the user study and complete the evaluation trials.
Participant-level random effect	A model component capturing individual differences in accuracy or response behavior across participants.
Perception and misperception	The ways in which viewers correctly or incorrectly interpret visual patterns in data visualizations.
Perceptual identification/ accuracy	The task performed by participants: deciding whether a 2-*D* NLDR plot and a tour represent the same underlying data.
Piling	A phenomenon in linear projections where many points overlap or concentrate near the center of the display, potentially obscuring important structure.
Pilot data	Preliminary experimental data used to estimate performance levels and inform the design of the main experiment, including sample size and effect size selection.
Point-level diagnostics	Residual and error measures computed for individual observations to identify local regions of poor model fit.
Polar angle	The angle controlling vertical position on a sphere (like latitude).
Polynomial structure	A curved pattern (quadratic or cubic) defined by polynomial relationships between variables.

(continued)

Term	Description
Power analysis	A procedure used to determine the number of responses required per treatment to reliably detect a specified effect size with high probability.
Prediction	The process of assigning a new high-dimensional observation to a location in the 2-D layout based on nearest neighbors in the lifted model.
Principal Component Analysis (PCA)	A linear dimension reduction method that identifies orthogonal directions (principal components) capturing the maximum variance in the data.
Prolific	An online crowd-sourcing platform used to recruit participants for the experiment.
Pyramid	A shape with a broad base that narrows toward an apex, with different possible base shapes (rectangular, triangular, star-shaped).
Quarto	A scientific and technical publishing system used to create reproducible documents, presentations, and websites combining text, code, and visualizations.
Radius scaling	Changing the size of a cross-section as you move along an axis (e.g. shrinking toward a tip).
Random Neighborhood Preservation (RNX) curve	A metric that quantifies neighborhood agreement across scales between high- and low-dimensional spaces.
Random Triplet Accuracy (RTA)	A metric assessing how well relative distances among random triplets of points are preserved in the embedding.
Random effect	A component of a statistical model that captures variability attributable to random differences across experimental units, such as subjects or data sets.
Reliability of NLDR representations	The degree to which a low-dimensional embedding faithfully reflects the structure present in the high-dimensional data.

(continued)

Term	Description
Residual	The Euclidean distance between an observation and its corresponding fitted bin centroid in high-dimensional space.
Residual-based evaluation	An approach to assessing NLDR quality by examining differences between fitted model values and observed high-dimensional data.
Resolution parameter	Any parameter (such as binwidth or neighborhood size) that controls the granularity at which structure is modeled or visualized.
Rotation	A transformation applied to data that changes its orientation in space while preserving distances and geometric relationships.
Rotation matrix	A matrix that changes the orientation of data without changing distances or variance.
Run sheets	Step-by-step task instructions provided to participants during a usability study to ensure consistency in task execution and to facilitate systematic feedback collection.
S-curve	A smooth, bent surface in 3D often used to test whether algorithms can unfold nonlinear structure.
S-curve with hole	An S-shaped manifold with a missing spherical region, creating topological complexity.
Same (SAME) trials	Trials in which the 2-*D* NLDR embedding and the tour are generated from the same high-dimensional dataset. These trials form the primary basis for analysis.
Scaled distance	A distance measure adjusted to account for differences in scale across simulated data structures.
Scaling	A transformation that adjusts the spread or magnitude of a geometric shape along one or more dimensions.
Scaling (data scaling)	Standardizing data before analysis so that variables have comparable ranges.
Scaling of NLDR data	Rescaling the 2-*D* embedding to a standardized range to preserve aspect ratio and ensure consistent binning.

(continued)

Term	Description
Server-side computation	A computational model in which data processing and analysis are performed on a remote server rather than on the users local machine. In menuraR, NLDR generation and diagnostics are executed on the Shiny server.
Shape generator	A function that produces synthetic data points according to a specified geometric form and set of parameters.
Shepard diagram	A scatterplot comparing pairwise distances in the original space to distances in the embedded space.
Simulation-based power analysis	A power analysis approach that uses simulated data, rather than closed-form formulas, to assess the ability to detect effects under realistic experimental conditions.
Single-cell RNA sequencing (scRNA-seq)	A technology that measures gene expression at the level of individual cells, producing extremely high-dimensional datasets.
Sparse sampling	Uneven sampling density across different regions of a manifold, which can distort NLDR layouts by contracting low-density regions.
Spearman correlation (SC)	A rank-based correlation used to assess monotonic agreement between high- and low-dimensional distances.
Sphere	The surface of a ball. Points lie only on the boundary, not inside.
Spherical spiral	A spiral path that wraps around the surface of a sphere.
Spins	The number of turns or revolutions in a spiral structure.
Standardization	A preprocessing step in which variables are rescaled to have comparable ranges (typically mean zero and unit variance) before applying NLDR methods.
Standardized bin count (w_h)	The proportion of total observations contained in a bin, used to identify low-density regions.
Static plot	A single, fixed 2D visualization of an NLDR embedding.

(continued)

Term	Description
Stress function	An objective function used in MDS to quantify the mismatch between distances in the original space and the embedded space.
Structure exaggeration	The tendency of NLDR methods to amplify apparent patterns, such as clusters or separations, in low-dimensional embeddings.
Structured noise	Noise that follows smooth or patterned trends instead of being purely random.
Subject variability	Differences in performance attributable to individual participants perceptual abilities, attention, and strategies. This variability is modeled using a subject-level random effect.
Swiss roll	A flat surface rolled up into a spiral in 3D, commonly used as a nonlinear manifold example.
Synthetic dataset	An artificially generated dataset designed to exhibit specific structural or statistical properties.
Technical barrier	Practical or knowledge-based obstacles (e.g., software installation, programming requirements) that can limit access to advanced analytical tools.
Topological complexity	Features like holes, loops, or twists that affect connectivity but not local smoothness.
Tour	A dynamic visualization technique that presents a continuous sequence of 2-*D* linear projections of high-dimensional data, allowing structure to be explored from multiple viewing angles.
Treatment	A distinct experimental condition formed by a specific combination of factors, here defined as an NLDR method paired with a distance scale factor.
Trefoil knot	A closed loop with self-crossings, forming a nontrivial knot used to test preservation of topology.
TriMAP	An NLDR method that preserves relative distances among triplets of points to balance local and global structure.

(continued)

Term	Description
Trial	A single evaluation task in which a participant compares two visual displays and judges whether they represent the same data.
Triangular mesh (triangulation)	A network of edges connecting neighboring bin centroids, derived using Delaunay triangulation to encode local neighborhood relationships.
Trigonometric structure	A geometric pattern generated using sine and cosine functions.
True model (geometric structure)	The subset of variables and relationships that define the underlying data-generating manifold, which NLDR aims to capture.
Twisting	A distortion in NLDR where the fitted manifold rotates or bends excessively in high-dimensional space.
UMAP (Uniform Manifold Approximation and Projection)	An NLDR method that balances local and global structure preservation using manifold learning principles.
Uniform distribution	All values within a range are equally likely.
Unimodal distribution	A probability distribution with a single dominant peak. The distribution of subject-level proportions correct is approximately unimodal, supporting the use of normally distributed random effects.
Unsupervised learning	A class of methods that identify patterns or structures in data without using labeled outcomes.
Usability survey	A structured evaluation method used to assess how easily users can learn, navigate, and complete tasks within an application. In this study, it was used to inform interface design and feature refinement in menuraR.
User interaction logs	Records of user actions within an application, such as layout generation and comparison steps, used to evaluate usability and identify design improvements.

(continued)

Term	Description
User study	An empirical experiment involving human participants, designed to assess how people interact with and interpret visual representations.
Version control	A system for tracking and managing changes to files, enabling collaboration, rollback, and transparent development history.
Visual conceptualization	The mental interpretation formed by a viewer when observing a visualization, such as perceiving the number of clusters or their relationships.
Visualization technique	A method for graphically representing data to aid interpretation, exploration, or comparison of structures.
Wavy noise dimensions	Noise variables that oscillate smoothly, often following sine or polynomial patterns.
Web-based interface	A graphical user interface accessed through a web browser that enables interaction with computational tools without requiring local software installation.
Wireframe model	A geometric representation of an NLDR layout constructed by connecting neighboring bin centroids with line segments, forming a mesh that approximates the layouts structure.
Within-cluster dispersion/ distance	A measure of how spread out points are within each cluster.
Wrapper function	A helper function that calls another function but simplifies inputs or outputs.
`cardinalR`	An R package designed to generate high-dimensional clustering data structures with controlled geometric and noise properties.
`detourr`	An R package for dynamic visualization of high-dimensional data using tour methods, supporting interactive exploration through projection sequences.
`knitr`	An R package that supports dynamic report generation by integrating R code with documents written in LaTeX, Markdown, or Quarto.

(continued)

Term	Description
`langevitour`	An R package used to generate smooth, continuous tours for exploring high-dimensional data and fitted models.
`menuraR`	A Shiny web application for comparing, diagnosing, and selecting the most reasonable NLDR layouts.
`quollr`	An R package developed to diagnose and evaluate NLDR layouts using visual and quantitative methods.
`shiny`	An R framework for building interactive web applications directly from R code.
autoAlt	An R package developed by the NUMBATs group that provides automated suggestions for generating alt-text for data visualizations.
scDEED	A method for evaluating NLDR embeddings by assigning reliability scores based on neighborhood preservation before and after embedding.
shinyapps.io	A cloud-based hosting platform for R Shiny applications that allows users to access interactive tools such as menuraR through a web browser without local installation.
tSNE (t-distributed Stochastic Neighbor Embedding)	An NLDR method that prioritizes preservation of local neighborhoods, often at the expense of global structure.