Manifold learning: what, how, and why

Marina Meila,[1] Hanyu Zhang,[2]

[1]Department of Statistics, University of Washington, Seattle, USA, 98195; email: mmp2@stat.washington.edu
[2]Tiktok,Inc. Bellevue, USA, 98004

Xxxx. Xxx. Xxx. Xxx. 2023. AA:1–28

https://doi.org/10.1146/((please add article doi))

Keywords

nonlinear dimension reduction, manifold learning, embedding

Abstract

Manifold learning (ML), also known as non-linear dimension reduction, is a set of methods to find the low-dimensional structure of data. Dimension reduction for large, high-dimensional data is not merely a way to reduce the data; the new representations and descriptors obtained by ML reveal the geometric shape of high-dimensional point clouds and allow one to visualize, denoise and interpret them. This review presents the underlying principles of ML, its representative methods, and their statistical foundations, all from a practicing statistician's perspective. It describes the trade-offs and what theory tells us about the parameter and algorithmic choices we make in order to obtain reliable conclusions.

1. Introduction

Modern data analysis tasks often face challenges in high dimensions. Thus nonlinear dimension reduction techniques emerge as a way to construct maps from high-dimensional data to corresponding low-dimensional representations. Finding such representations is beneficial in several aspects. Reducing dimension, while preserving the relevant geometric features of the data saves space and processing time. More importantly, the low dimensional representation frequently provides a better understanding of the *intrinsic* structure of data, which often leads to better features that can be fed into further data analysis algorithms; Figure 1 illustrates such a case. This survey paper reviews the mathematical background, methodology, and recent nonlinear dimension reduction techniques developments. These techniques have been developed for two decades since two seminal works: Tenenbaum et al. (2000) and Roweis & Saul (2000), and are widely used in various data analysis tasks, especially in scientific research.

Before nonlinear dimension reduction emerged, Principal Component Analysis (PCA) was already widely accepted (I.T.Jolliffe 2002). Intuitively, PCA assumes that high dimensional data living in $\mathbb{R}^D$ lie around a lower-dimensional linear subspace of $\mathbb{R}^D$. It aims to identify an optimal linear subspace such that data points projected onto this subspace have minimal reconstruction error. Nonlinear dimension reduction algorithms extend this idea by assuming data are supported on smooth nonlinear low-dimensional geometric objects (i.e., manifolds embedded in $\mathbb{R}^D$) and find maps that send the samples into lower dimensional coordinates while preserving some intrinsic geometric information.

In this survey, we start with a brief introduction to the central differential geometric concepts underlying ML, elaborating on the geometric information that manifolds carry (Section 2). Then, in Section 3, we describe the paradigm of manifold learning, with three possible sub-paradigms, each producing a different representation of the data manifold. The rest of the paper focuses on one of these, namely on the so-called embedding algorithms. In Section 4, we survey representative embedding algorithms and their variants. We also discuss the parameter choices and some pitfalls, which leads to the discussion in Section 5, where we present the statistical aspects and statistical results supporting these choices. This section also includes the estimation of crucial manifold descriptors from data: the Laplace-Beltrami operator, Riemannian metrics, intrinsic dimension. Section 6 discusses applications, connecting with related statistics problems, and Section 7 concludes the survey.

2. Mathematical background: manifolds, coordinate charts, embeddings

Manifolds and Coordinate Charts Readers are referred to Lee (2003), do Carmo (1992) for a rigorous introduction to manifolds and differential geometry. Intuitively, a manifold is a generalization of curves and surfaces with coordinate systems (called charts). On objects like a sphere or torus, one cannot maintain a globally continuous single coordinate system, hence, a manifold is described by multiple charts, as in Figure 3. Below, we explain what they are and why they can be ignored in everyday work with manifold data.

A function f is **smooth** when it is differentiable and its derivatives are continuous.

Mathematically, $\mathcal{M}$ is a *(smooth) manifold* of dimension d when it can be covered by "patches" (open sets) U so that: (1) For each U there is an invertible mapping $\varphi : U \to \varphi(U) \subset \mathbb{R}^d$, so that both φ, φ^{-1} are smooth. Such pair (U, φ) is called a *chart*; $\varphi(\boldsymbol{p}) \in \mathbb{R}^d$ is the *local coordinate* of $\boldsymbol{p} \in \mathcal{M}$. (2) Whenever two charts (U, φ) and (V, ϕ) overlap, the change of coordinates $\varphi \circ \phi^{-1}$ is smooth on $\phi(U \cap V)$ and has a smooth inverse.

Hence, a manifold has a Euclidean coordinate system (the chart) locally around every

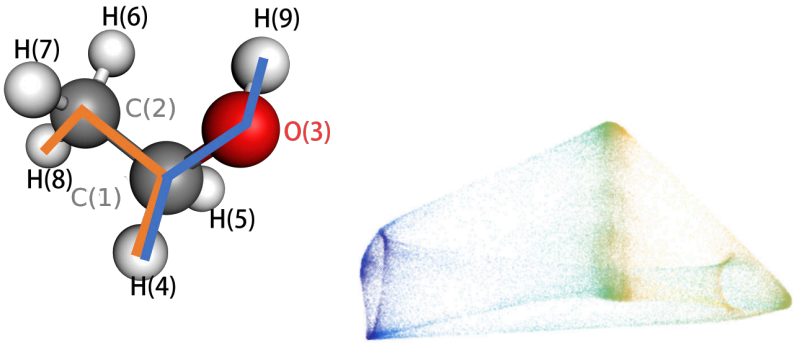

Figure 1: **Left:** The ethanol molecule has 9 atoms; a spatial configuration of ethanol has $D = 3 \times 9$ dimensions. The CH_3 group (comprising atoms 2, 6, 7, 8) and the OH group (atoms 3 and 9) can rotate with respect to the middle group (atoms 1, 4, 5), and the blue and orange lines represent these angles of rotation. **Right:** A 2-manifold estimated from 50,000 configurations of the ethanol molecule. The manifold has the topology of a torus, and the color represents the rotation of the OH group, pointing out that the two above rotation angles are sufficient to approximate any molecular configuration in these data. The sharp "corners" are distortions introduced by the embedding algorithm (explained in Section 5.1). Figure 6 shows the original data; the dataset is from Chmiela et al. (2017).

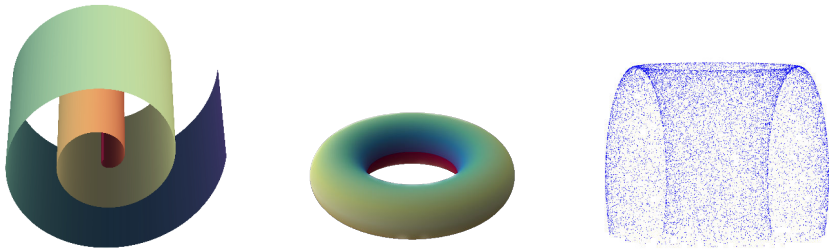

Figure 2: Examples of manifolds. **Left**: A swiss roll; **Middle**: A torus; **Right**: Data sampled from a torus that is chopped by a plane.

point, but the coordinate system may not extend to the whole manifold. In this case, transitions between charts are seamless.

The simplest example of a manifold is $\mathbb{R}^d$ itself, which has a single, global coordinate chart. The "swiss roll" in Figure 2 is a 2-manifold (i.e., a manifold of dimension 2) that also admits a global coordinate chart (into $\mathbb{R}^2$, by simply unrolling it). A sphere, or a torus (Figure 2), is also a 2-manifold, but they cannot be covered by a single chart (they each require at least two), as cartographers well know.

Coordinate charts are not unique; there are infinitely many coverings with patches U, and changes of variables for each φ. While this multiplicity of charts and coordinate functions can be daunting at first sight, the framework of differential geometry is set up so that most geometric quantities related to a manifold $\mathcal{M}$ are independent of the coordinates

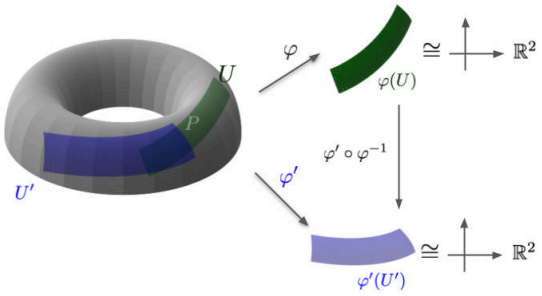

Figure 3: Manifold and charts. The torus is a manifold with an intrinsic dimension $d = 2$ situated within the ambient space $\mathbb{R}^3$. The entire torus cannot be unfolded on the $\mathbb{R}^2$ plane without cutting or collapsing it, but patches of it, such as U and U' can. The price paid is that each patch now has a different coordinate system, and to travel on the torus one must apply coordinates changes.

chosen. For example, the compatibility of charts shows that the dimension d must be the same for all charts. Hence, d is called the *intrinsic dimension* of the manifold $\mathcal{M}$.

For a data scientist, this implies that (1), they can work in the coordinate system of their choice, and intrinsic quantities like d will remain invariant. But, (2), care must be taken when the outputs of two different algorithms or from different samples are being compared because these may not be in the same coordinate system.

Embeddings In differential geometry, an *embedding* is a smooth map $F : \mathcal{M} \to \mathcal{N}$ between two manifolds whose inverse $F^{-1} : \mathcal{F}(\mathcal{M}) \subset \mathcal{N} \to \mathcal{M}$ exists and is also smooth. Commonly in statistics, the high dimensional data lie originally in $\mathbb{R}^D$. Then D is called the *ambient dimension* (of the data). The ML algorithms under consideration aim to find an embedding $F : \mathcal{M} \to \mathbb{R}^m$, where $m \geq d$ and $m \ll D$. Notably, if $m = d$, the embedding F represents a (global) coordinate chart.

An advantage of embeddings is that one can avoid using multiple charts to describe a manifold. Instead, one can find a global mapping $F : \mathcal{M} \subset \mathbb{R}^D \to \mathcal{N} \subset \mathbb{R}^m$, where $\mathcal{N}$ is easier to understand. Whitney's embedding Theorem (Lee 2003) states that every $d-$dimensional manifold can be embedded into $\mathbb{R}^{2d}$. Therefore, if one can find a valid embedding, a significant dimension reduction can be achieved (from D to $O(d)$). This is one of the major targets of manifold learning algorithms.

Tangent spaces The *tangent space* $\mathcal{T}_{\boldsymbol{p}}\mathcal{M}$ at a point $\boldsymbol{p} \in \mathcal{M}$ is a d-dimensional vector space of *tangent vectors* to $\mathcal{M}$. The canonical basis of $\mathcal{T}_{\boldsymbol{p}}\mathcal{M}$ is given by the tangents to the coordinate functions seen as curves on $\mathcal{M}$, while the tangent vectors can be seen as tangents (or velocity vectors) at $\boldsymbol{p}$ to smooth curves on $\mathcal{M}$ passing through $\boldsymbol{p}$.

3. Premises and paradigms in manifold learning

The Manifold Assumption Suppose we are given data $\{\boldsymbol{x}_i\}_{i=1}^n$ where each data point $\boldsymbol{x}_i \in \mathbb{R}^D$. It is assumed that data are sampled from a distribution $\mathbb{P}$ that is supported on, or close to a d dimensional manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$. This is the *Manifold Assumption*.

Throughout this survey, with a few exceptions, we will discuss the no noise case when the data lie on $\mathcal{M}$.

Manifold learning A manifold learning algorithm can be thought of as a mapping F of $\boldsymbol{x}_i \in \mathbb{R}^D$ to $\boldsymbol{y}_i \in \mathbb{R}^m$. The *embedding dimension* m is usually much smaller than D but could be higher than the intrinsic dimension d. In the regime that $\mathbb{P}$ is supported exactly on $\mathcal{M}$, and sample size $n \to \infty$, a valid manifold learning algorithm F should converge to a smooth embedding function F. This implies that the algorithm should be guaranteed to recover the manifold $\mathcal{M}$, regardless of the shape of $\mathcal{M}$.

Can a manifold be estimated? The Manifold Assumption itself is testable. For example Fefferman et al. (2016) tests whether, given an i.i.d. sample, there exists a manifold $\mathcal{M}$ that can approximate this sample with tolerance ε. These results are currently not practically useful, as knowledge of usually unknown manifold parameters (d, volume, etc) must be known or estimated. However, they, as well as Genovese et al. (2012), give us the confidence to develop and use ML algorithms in practice.

3.1. Neighborhood graphs

Practically all manifold learning algorithms start with finding the neighbors of each data point $\boldsymbol{x}_i$. This leads to the construction of a *neighborhood graph*; this graph, with suitable weights, summarizing the local geometric and topological information in the data, is the typical input to a non-linear dimension reduction algorithm. Every data point $\boldsymbol{x}_i$ represents a node in this graph, and an edge connects two nodes if their corresponding data points are neighbors. Throughout the survey, we use $\mathcal{N}_i$ to denote the neighbors of $\boldsymbol{x}_i$ and $k_i = |\mathcal{N}_i|$ the number of neighbors of $\boldsymbol{x}_i$ (including $\boldsymbol{x}_i$ itself).

There are two usual ways to define neighbors. In a *radius-neighbor graph*, $\boldsymbol{x}_j$ is a neighbor of $\boldsymbol{x}_i$ iff $||\boldsymbol{x}_i - \boldsymbol{x}_j|| \leq r$. Here r is a parameter that controls the neighborhood scale, similar to a bandwidth parameter in kernel density estimation. Consistency of manifold learning algorithms is usually established assuming an appropriately selected neighborhood size that decreases slowly with n (see Section 5.2). In the *k-nearest neighbor (k-NN) graph*, $\boldsymbol{x}_j$ is the neighbor of $\boldsymbol{x}_i$ iff $\boldsymbol{x}_j$ is among the closest k points to $\boldsymbol{x}_i$. Since this relation is not symmetric, usually, the neighborhoods are symmetrized.

The $k-$NN graph has many computational advantages w.r.t. the radius neighbor graph; it is more regular, and often connected when the latter is not. More software is available to construct (approximate) k-NN graphs fast for large samples. Nevertheless, theoretically, it is much more challenging to analyze, and fewer consistency results are known for $k-$NN graphs (Sections 5.1, 5.4). Intuitively, k_i the number of neighbors in the radius graph is proportional to the local data density, and manifold estimation can be analyzed through the prism of kernel regression;. In contrast, the k-NN graph is either asymmetric or if symmetrized, becomes more complicated to analyze.

The distances between neighbors are stored in the distance matrix $\mathbf{A}$, with $\mathbf{A}_{ij}$ being the distance $||\boldsymbol{x}_i - \boldsymbol{x}_j||$ if $\boldsymbol{x}_j \in \mathcal{N}_i$, and infinity if $\boldsymbol{x}_j$ is not a neighbor of $\boldsymbol{x}_i$.

Some algorithms weight the neighborhood graph by weights that are non-increasing with distances; the resulting $n \times n$ matrix is called the *similarity matrix* (or sometimes

kernel matrix). The weights are given by a *kernel function*,

$$\mathbf{K}_{ij} := \begin{cases} K\left(\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||}{h}\right), & \boldsymbol{x}_j \in \mathcal{N}_i, \\ 0, & \text{otherwise.} \end{cases} \quad 1.$$

The kernel function here is almost universally the Gaussian kernel, defined as $K(u) = \exp(-u^2)$ (Belkin et al. (2006), Ting et al. (2010), Coifman & Lafon (2006), Singer & Wu (2012)). In the above, h, the kernel width, is another hyperparameter that must be tuned. Note that, even if $\mathcal{N}_i$ would trivially contain all the data, the similarity $\mathbf{K}_{ij}$ vanishes for far-away data points. Therefore, equation 1. effectively defines a radius-neighbor graph with $r \propto h$. Hence, a rule of thumb is to select r to be a small multiple of h (e.g., 3–10h).

It is sometimes also useful to have kernel function $K(u) = \mathbf{1}$. Then the similarity matrix $\mathbf{K}$ is the same as the unweighted adjacency matrix of the neighborhood graph. By construction, $\mathbf{K}$ is usually a sparse matrix, which is useful to accelerate the computation.

When the data dimension D and sample size n are large—the latter being essential for manifold recovery—constructing the neighborhood graph often becomes the algorithm's most computationally demanding step. Fortunately, much work has been devoted to speeding up this task, and approximate algorithms are now available, which can run in almost linear time in n and have very good accuracy (Ram et al. 2009).

3.2. Linear local approximation and Principal Curves and Surfaces

Here we quickly review two methods for manifold estimation: local linear approximation reduces the dimension locally but offers no global representation, while principal curves produce a global representation but do not reduce dimension. Then, from Section 4, we focus on the third class, consisting of algorithms that produce embeddings, representations of global and low dimensions.

Linear local approximation This idea is derived from classical Principal Component Analysis, which identifies a global optimal linear subspace to approximate the data. In linear local approximation, PCA is performed on a weighted covariance matrix, with weights decaying away from any point $\boldsymbol{x}$; this approximates data locally around $\boldsymbol{x}$ on a curved manifold and can produce a chart around specific fixed reference point. To cover the entire manifold, one needs to obtain multiple such charts.

Principal curves and principal d-manifolds In this paradigm, noise is assumed. Consider data of the form $\boldsymbol{x}_i = \boldsymbol{x}_i^* + \epsilon_i$, where ϵ_i represents 0-mean noise, and the $\boldsymbol{x}_i^*$ are sampled from a curve, for instance. This data density has a *ridge* $\tilde{\mathcal{M}}$, called *principal curve*, and the *Subspace Constrained Mean Shift (SCMS)* algorithm of Ozertem & Erdogmus (2011) maps each $\boldsymbol{x}_i$ iteratively to a point $\boldsymbol{y}_i \in \mathbb{R}^D$ lying on the principal curve. This concept can be extended to principal surfaces and principal d-manifolds.

Usually, the ridge does not coincide with the mean of the data; the bias depends on the manifold's curvature: the density is higher on the "inside" of the curve. However, for their smoothing property, principal d-manifolds are remarkably useful in analyzing manifold estimation in noise (Genovese et al. 2012, Mohammed & Narayanan 2017).

Table 1: Three main paradigms for non-linear dimension reduction

Paradigm	Representation
Linear local approximation	$D \to d$, local coordinates only
Principal Curves and Surfaces	$D \to D$, global coordinates, noise removal
Embedding	$D \to m$, with $D \gg m \geq d$, global coordinates (or charts)

4. Embedding algorithms

The term "manifold learning" was proposed in the works of Roweis & Saul (2000) and Tenenbaum et al. (2000) which introduced the Local Linear Embedding (LLE) and Isomap algorithms, inaugurating the modern era of non-linear dimension reduction. In this section, we introduce classical manifold learning algorithms that aim to find a global embedding $\boldsymbol{y}_1, \dots \boldsymbol{y}_n$, also denoted $\mathbf{Y} \in \mathbb{R}^{n \times m}$ (with $\boldsymbol{y}_i$ representing row i of $\mathbf{Y}$) of data set $\mathcal{D}$.

Algorithms can be broadly categorized into "one-shot", which derive embedding coordinates from principal eigenvectors of a matrix associated with the neighborhood graph or by solving some other global (usually convex) optimization problem, and "attraction-repulsion" algorithms, which proceed from an initial embedding $\mathbf{Y}$ (often produced by a one-shot algorithm) and improve it iteratively. While this taxonomy can rightly be called superficial, at present, it represents a succinct and relatively accurate summary of the state of the art.

No matter what the approach, given the neighborhood information summarized in the weighted neighborhood graph, an embedding algorithm's task is to produce a smooth mapping F of $\boldsymbol{x}_1, \dots \boldsymbol{x}_n$ which distorts the neighborhood information as little as possible. The algorithms that follow differ in their choice of information to preserve and in the sometimes implicit constraints on smoothness.

Laplacian matrix: generalization of the Laplacian differential operator $\Delta f = \sum_j \frac{\partial^2 f}{\partial x_j^2}$ on a graph. To see this, consider the graph 1–2–. . . –i–. . . n (a chain) with edge length h and a function f with $f_i = f(i)$. By finite differences $\Delta f(i) = \frac{1}{h}\left[\frac{f_{i+1}-f_i}{h} - \frac{f_i - f_{i-1}}{h}\right] = \frac{1}{h^2}\left(\sum_{j \in \mathcal{N}_i} f_j - d_i f_i\right)$.

4.1. "One shot" embedding algorithms

In this section, we focus on the best-studied one-shot embedding algorithm, Diffusion Maps (DM, Coifman & Lafon (2006)), and its variant Laplacian Eigenmaps (LE, Belkin & Niyogi (2003)). Other one-shot embedding algorithms include Isomap (Tenenbaum et al. 2000) and Local Tangent Space Alignment (LTSA), Zhang & Zha (2004)), described in Section A of the SI, along with PCA and Multidimensional Scaling (MDS).

DM, as well as most one-shot embedding methods, works with a sparse matrix derived from the similarity $\mathbf{K}$; namely, DM uses the eigenvectors of the *Laplacian* matrix $\mathbf{L}$ to embed the data.

To construct a Laplacian matrix, define $d_i = \sum_{j \in \mathcal{N}_i} \mathbf{K}_{ij}$ as the *degree* of node i and set $\mathbf{D} = \mathrm{diag}\{d_1, \cdots, d_n\}$. Then multiple choices of graph Laplacian exist:

- *Unnormalized* Laplacian: $\mathbf{L}^{un} = \mathbf{D} - \mathbf{K}$
- *Normalized* Laplacian: $\mathbf{L}^{nor} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$
- *Random-walk* Laplacian: $\mathbf{L}^{rw} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{K}$
- *Renormalized* or *Diffusion Maps* Laplacian $\mathbf{L}$ is defined in Algorithm 1 below

Why choose one Laplacian rather than another? The reason is that, even though in simple examples the difference is hard to spot, as more sample are collected, one needs to ensure that the limit of these $\mathbf{L}$ matrices is well defined, and the embedding algorithm is unbiased. It is easy to see that $\mathbf{L}^{norm}$ and $\mathbf{L}^{rw}$ are similar matrices. Moreover, whenever the degrees

Algorithm 1 Renormalized Laplacian

Input: Similarity matrix $\mathbf{K}$, kernel bandwitdth h

Normalize columns: $d_j = \sum_{i=1}^n \mathbf{K}_{ij}$, $\tilde{\mathbf{K}}_{ij} = \mathbf{K}_{ij}/d_j$ for all $i, j = 1, \ldots n$

Normalize rows: $d'_i = \sum_{j=1}^n \tilde{\mathbf{K}}_{ij}$, $\mathbf{P}_{ij} = \tilde{\mathbf{K}}_{ij}/d'_i$ for all $i, j = 1, \ldots n$

Output: $\mathbf{L} = (\mathbf{I} - \mathbf{P})/h^2$

d_i are constant, $\mathbf{L} \propto \mathbf{L}^{rw} \propto \mathbf{L}^{un}$, hence all Laplacians produce the same embedding. Differences arise when data density is non-uniform, making the degrees d_i larger in regions of higher density. The seminal work of Coifman & Lafon (2006), which introduced renormalization, showed that the eigenvectors of $\mathbf{L}^{norm}, \mathbf{L}^{rw}$ are biased by the sampling density and that renormalization removes this bias. Sections 5.4, and Figure 6 illustrate this.

Using the defined graph Laplacian matrix $\mathbf{L}$, we can summarize the DM procedure presented in algorithm 2. Similar to PCA, the data are mapped to the principal directions

Algorithm 2 Diffusion Maps/Laplacian Eigenmaps

Input: Laplacian $\mathbf{L}$ (or $\mathbf{L}^{nor}$), embedding dimension m.

1: Compute $\{\boldsymbol{v}^i\}_{i=0}^m$, eigenvectors of smallest $m+1$ eigenvalues of $\mathbf{L}$, with $\boldsymbol{v}^i \in \mathbb{R}^n$.

2: Discard $\boldsymbol{v}^0$ (this is typically a constant vector (Shi & Malik 2000))

3: Represent each $\boldsymbol{x}_j$ by $\boldsymbol{y}_j = (v_j^1, \cdots, v_j^m)^\top \in \mathbb{R}^m$

Output: $\mathbf{Y}$

of a positive definite matrix. While in PCA, these eigenvectors represent directions of maximum variance, in DM they represent the *smoothest* (least varying) eigenvectors of $\mathbf{L}$; therefore, they correspond to the *lowest eigenvalues* (see also Section 6.1). The Laplacian Eigenmaps (LE) algorithm resembles DM but uses a different Laplacian, namely $\mathbf{L}^{norm}$ above.

The idea of *spectral embedding* also appeared independently in graph visualization, then in Shi & Malik (2000) as a method for clustering, and was then generalized as a data representation method in Belkin & Niyogi (2003) as LE. They connect the graph Laplacian with the *Laplace-Beltrami* operator $\Delta_{\mathcal{M}}$ of manifold $\mathcal{M}$ (Rosenberg 1997). Estimating the Laplace-Beltrami operator itself is an important geometric estimation problem that will be reviewed in Section 5.4.

4.2. "Horseshoe" effects, neighbor embedding algorithms, and selecting independent eigenvectors

4.2.1. The Repeated Eigenvectors Problem (REP). Algorithms that use eigenvectors, such as DM, are among the most promising and well-studied in ML (see Sections 5.1,5.2,5.4). Unfortunately, such algorithms fail when the data manifold has a large aspect ratio, such as a long, thin strip or a slender torus. This problem has been called *the Repeated Eigendirections Problem (REP)* in **?**. The REP has been demonstrated theoretically for DM/LE, LTSA, LLE (Goldberg et al. 2008), and in real data sets.

From a mathematical standpoint, the REP is due to eigenvectors (or eigenfunctions, in the limit) that are harmonics of previous ones, as shown in Figure 4. Consider, for example, the rectangle $[0, l] \times [0, 1]$ in (x_1, x_2) space, where the length $l > 1$; l in this case, is the aspect ratio. It is easy to show that (in the continuum limit), the first $\lceil l \rceil - 1$ eigenvectors

vary in the x_1 direction, as shown in the top row of Figure 4. Hence, if we use $(\boldsymbol{v}^1, \boldsymbol{v}^2)$ in the DM algorithm, we obtain a 1-dimensional mapping, even though the rectangle is 2-dimensional.

Moreover, in this simple case, the scatterplot of $(\boldsymbol{v}_i^1, \boldsymbol{v}_i^2)_{i=1,\dots n}$ from step 3 of the DM follows a parabola. This is a relevant diagnosis for REP in practice: when an embedding looks like a "horseshoe", this may not represent a property of the data but an artifact signalling that one of the data dimensions is collapsed or poorly reflected in the embedding (Diaconis et al. 2008).

4.2.2. Relaxation-based neighbor embedding algorithms. The pervasiveness of the REP stimulated the development of algorithms that balance attraction between neighbors in the original space, with repulsion between neighbors in the embedding space (van der Maaten & Hinton 2008, McInnes et al. 2018, Jacomy et al. 2014, Carreira-Perpiñan 2010, Im et al. 2018). Usually, the embedding coordinates $\mathbf{Y}$ are optimized iteratively until equilibrium is reached.

The t-SNE algorithm of van der Maaten & Hinton (2008), one variant of which (Böhm et al. 2022) we briefly describe here, exemplifies this approach. Hinton & Roweis (2002), proposed to match the (normalized) data similarities by (normalized) output similarities around each embedded point $\boldsymbol{y}_i$, which motivates the name *Stochastic Neighbor Embedding* (SNE, Hinton & Roweis (2002)). In van der Maaten & Hinton (2008), the authors proposed to use a Student-t distribution to model the output similarities, and, as t-SNE, this algorithm became widely used. Uniform manifold approximation and projection (UMAP,

Algorithm 3 t-SNE

Input: Similarity matrix $\mathbf{K}$ (from k-nearest neighbor graph), initial embedding $\boldsymbol{y}_1, \dots \boldsymbol{y}_n$, step size η, repulsion parameter ρ

1: Compute normalized input similarity $\mathbf{V} = (\mathbf{D}^{-1}\mathbf{K} + \mathbf{K}\mathbf{D}^{-1})/(2n)$

2: **while** not converged **do**

3: Compute all squared distances in embedding space $\mathbf{A}_{ij}^{out} = \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2$, for $i, j = 1, \dots n$

4: Compute similarities in embedding space $\mathbf{W}_{ij} = \frac{1}{1+\mathbf{A}_{ij}^{out}}$, for $i, j = 1, \dots n$, $w_{tot} = \sum_{i,j=1}^{n} \mathbf{W}_{ij}$

5: Update embedding by $\boldsymbol{y}_i \leftarrow \boldsymbol{y}_i + \eta \left[\sum_{j=1}^{n} \mathbf{V}_{ij}\mathbf{W}_{ij}(\boldsymbol{y}_i - \boldsymbol{y}_j) - \frac{n}{\rho}\sum_{j=1}^{n} \frac{\mathbf{W}_{ij}}{w_{tot}}(\boldsymbol{y}_i - \boldsymbol{y}_j)\right]$.

6: **end while**

Output: $\mathbf{Y}$

McInnes et al. (2018)) is another popular heuristic method. On a high level, UMAP minimizes the mismatches between topological representations of high-dimensional data set $\{\boldsymbol{x}_i\}_{i=1}^n$ and its low-dimensional embeddings $\boldsymbol{y}_i$. Theoretical understanding of UMAP is still limited.

The t-SNE algorithm has the advantage of being sensitive to local structure and to clusters in data (Linderman & Steinerberger 2019, Kobak et al. 2020), but does not explicitly preserve the global structure. We note that the propensity for finding clusters comes partly from the choice of neighborhood graph (Section 5.1). However, this is not the whole story. Recently, it has been shown that this property stems from the last term of the update in step 5 above. The first term in the change of $\boldsymbol{y}_i$ is an attraction between graph neighbors,

Figure 4: Embedding algorithms failing to find a full rank mapping, if they greedily select the first $m = 2$ eigenvectors, and correction by a more refined choice of eigenvectors. **Top row**: embeddings of a swiss roll with length seven times the width. **Left:** first two eigenvectors from DM/LEform a 1-dimensional curve; hence $\boldsymbol{v}^2$ does not add a new dimension, but "repeats" $\boldsymbol{v}^1$; **middle** the same after UMAP: repulsion expands the curve to a strip, but is not able to produce a full-rank embedding everywhere; the "knots", the horseshoe and the three clusters are all artifacts. **Right**: UMAP with selection of eigenvectors by Chen & Meila (2021). **Bottom row**: Embeddings of galaxy spectra from the SDSS (Section 6) by DM ; **left** "horseshoe" when first 2 eigenvectors are used; **right** the same data, by eigenvectors $(\boldsymbol{v}^1, \boldsymbol{v}^3)$, selected by Chen & Meila (2021). Plots by Yu-Chia Chen.

while the second represents repulsive forces between the embedded points $\boldsymbol{y}_{1:n}$ (Böhm et al. 2022, Zhang et al. 2022). The parameter ρ (originally called *early exaggeration*) controls the trade-off between attraction and repulsion. In Böhm et al. (2022), it is shown that varying ρ from small to large values decreases the cluster separation and makes the embedding more similar to the LE embedding. Moreover, quite surprisingly, Böhm et al. (2022) show that by varying ρ, the t-SNE can emulate a variety of other algorithms, most notably UMAP (McInnes et al. 2018) and ForceAtlas (Jacomy et al. 2014). Other works that analyze the attraction-repulsion behavior of t-SNE are Zhang & Steinerberger (2021). One yet unsolved issue with t-SNE is the choice of the number of neighbors k. Most applications use the default $k = 90$ (Poličar et al. 2019); this choice, as well as other behaviors of this class of algorithms, are discussed in Zhang et al. (2022).

Finally, in *Minimum Variance Unfolding (MVU)*, proposed in Weinberger & Saul (2006), Arias-Castro & Pelletier (2013), repulsion is implemented via a *Semidefinite Program*, hence the embedding $\mathbf{Y}$ is obtained by solving a convex optimization. This algorithm can be seen both as a one-shot and as an attraction-repulsion algorithm; Diaconis et al. (2008) show that MVU is related to the fastest mixing Markov chain on the neighborhood graph. Note also that since the REP can be interpreted as extreme distortion, the RiemannianRelaxationof Perrault-Joncas & Meila (2014) (see also Section 5.5) can also be used to improve the conditioning of an embedding in an iterative manner.

4.2.3. Avoiding the REP in spectral embeddings. The REP has a theoretically straightforward solution for algorithms like DM, and LTSA. From the sequence of eigenfunctions $F^1, \dots, F^{m'} \dots$ on $\mathcal{M}$ (or eigenvectors $\boldsymbol{v}^1, \dots \boldsymbol{v}^{m'}$ in the finite sample case), with $m' > m$, sorted by their corresponding eigenvalues, one needs to select $F^{j_1} = F^1$, then (recursively) $F^{j_2}, \dots F^{j_m}$ so that the rank of the Jacobian $[(\mathrm{d}F^1)_{\boldsymbol{p}}, \dots (\mathrm{d}F^{j_m})_{\boldsymbol{p}}]$ is d at every point $\boldsymbol{p} \in \mathcal{M}$. E.g., for the $l \times 1$ rectangle, eigenvectors $\boldsymbol{v}^1$ and $\boldsymbol{v}^{\lceil l \rceil}$ should be selected. This is called *Independent Eigendirection Selection (IES)*. In a finite sample, the rank condition must be replaced with the well-conditioning of $\mathrm{d}F$ at the data points. Dsilva et al. (2018) proposed to measure dependence by regressing $\boldsymbol{v}_{j_{k+1}}$ on the previously selected $\boldsymbol{v}_{j_1,\dots j_k}$; in Chen & Meila (2021), a condition number derived from the embedding metric (Section 5.5) is used to evaluate entire sets of m eigenvectors. The *manifold deflation* method (Ting & Jordan 2020) proposes to bypass eigenvector selection by choosing a linear combination of all optimized eigenvectors w.r.t. rank. Finally, the *Low Distortion Local Eigenmaps (LDLE)* (Kohli et al. 2021) solves the REP by essentially covering the data manifold with contiguous patches (discrete versions of the U neighborhoods) and performing IES on each patch separately. LDLE avoids REP and is a first step towards the algorithmic use of charts and atlases to complement global embeddings.

In summary, attraction-repulsion algorithms such as t-SNE, which are heuristic, enjoy large popularity due in part to their immunity to the REP, while eigenvector-based methods, although better grounded in theory, are less useful in practice without post-processing by an IES method. On the other hand, unlike global search in eigenvector space, a local relaxation algorithm cannot resolve the rank deficiency globally, and it may become trapped in a local optimum (Figure 4).

4.3. Summary of embedding algorithms

A variety of embedding algorithms have been developed. Here we presented representative algorithms of two types. One-shot algorithms that (typically) embed the data by eigenvectors, of which Isomap, DM and LTSA are the best understood and computationally scalable. The main drawback of this class of algorithms is the Repeated Eigendirections Problem, which requires post-processing of the eigenvectors. Neighbor embedding algorithms are (typically) iterative, starting with the output of a one-shot algorithm (LE for UMAP) or even PCA. The presence of repulsion makes these algorithms robust to REP, affecting one-shot algorithms. Quantifying the repulsion, smoothness, large-sample limits, and other properties of the neighbor embedding algorithms are less developed. Hence, for the moment, neighbor embedding algorithms remain heuristic for ML, while they remain useful for visualization, and clustering (for which guarantees exist, e.g., in Linderman & Steinerberger (2019)).

Neither algorithm guarantees against local singularities, such as the "crossing" in Figure 4. It is not known how these can be reliably detected or avoided. Additionally, all algorithms distort distances except in special cases (as discussed in Section 5.5).

All algorithms depend on hyperparameters: intrinsic dimension d (Section 5.3) or embedding dimension m, and k or r for the neighborhood scale (Section 5.2). Iterative algorithms often depend on additional parameters controlling the repulsion (such as ρ in t-SNE) or the step size η.

With respect to computation, constructing the neighborhood graph is the most expensive step, typically for n large. To compound this problem, finding k or r in a principled way

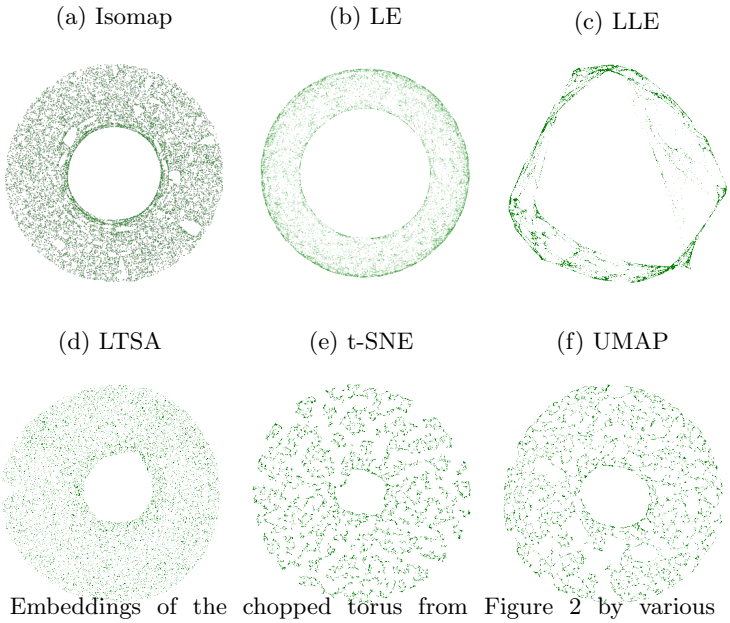

Figure 5: Embeddings of the chopped torus from Figure 2 by various algorithms; ISOMAP and LTSA are described in the Supplementary Materials. This manifold cannot be embedded isometrically in $d = 2$ dimensions; each algorithm stretches/contracts (distorts) it differently. Figure 7 in the SI visualizes the local distortions.

often requires constructing multiple graphs, one for each scale. One-shot algorithms that compute eigenvectors are quite efficient for n up to 10^6 when the neighborhood graph is not dense (**?**). Neighbor embedding algorithms work, in theory, with dense matrices (e.g., $\mathbf{W}$); however, accelerated approximate versions for these algorithms have been developed, such as the Barnes-Hut trees approximation (van der Maaten 2014), and the negative sampling heuristic for UMAP (Böhm et al. 2022, McInnes et al. 2018).

5. Statistical basis of manifold learning

The output or result of manifold learning algorithms depends critically on algorithm parameters such as the type of neighborhood graph (k-nearest neighbor or radius neighbor), the neighborhood scale (k or r), and embedding dimension m (and intrinsic dimension d, in some cases).

This section is concerned with making these choices in a way that ensures some statistical consistency, whenever possible. Neglecting statistical consistency and theoretical guarantees in general, is risky. In the worst case, it can lead to methods that have no limit when $n \to \infty$ (e.g. for LLE without any regularization (Ting et al. 2010)), and in milder cases to biases (e.g., due to variations in data density), and artifacts, i.e., features of the embedding, such as clusters, arms, and horseshoes that have no correspondence in the data.

Here we discuss in more general terms what is known about graph construction methods (Section 5.1), the neighborhood scale (Section 5.2), and the intrinsic dimension (Section 5.3). We revisit the estimation of the manifold Laplacian (the limit of $\mathbf{L}$), as the natural representation of the manifold geometry, and the basis for the DIFFUSION MAPS embedding,

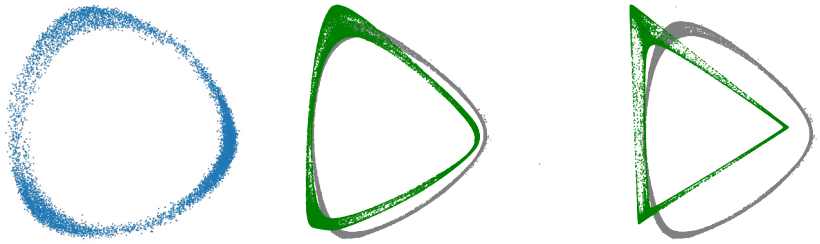

Figure 6: Effects of graph construction and renormalization, when the sampling density is highly non-uniform, exemplified on the configurations of the ethanol molecule. **Left:** original data, after preprocessing, is a noisy torus (shown here in the first two principal components), with three regions of high density, around local minima of the potential energy. **Center:** Embeddings by DM (gray), and by the same algorithm with $\mathbf{L}$ constructed from the k-nearest neighbor graph (green). The sparse regions are stretched, while the dense regions appear like "corners" of the embedding. Note that DM *should* remove the effects of the density; in this case, the variations in density are so extreme that the effect persists. The effect is somewhat stronger for the k-nearest neighbor graph. **Right:** Embedding by DM (gray) and by LE (green), which uses the singly normalized $\mathbf{L}^{rw}$.

which can be seen as the archetypal embedding, in Section 5.4. Finally, in Section 5.5, we turn to mitigating the distortions induced by embedding algorithms.

5.1. Biases in ML. Effects of sampling density and graph construction

Biases due to non-uniform density Many embedding algorithms tend to contract regions of $\mathcal{M}$ where the data are densely sampled and to stretch the sparsely sampled regions. In attraction-repulsion algorithms, such as T-SNE, this is explained by the repulsive forces between every pair of embedding points $\boldsymbol{y}_i, \boldsymbol{y}_j$, while the attractive forces act only along graph edges, between neighbors. If fewer graph edges connect two dense regions, repulsion will push them apart, exaggerating clusters.

For one-shot algorithms, the effect is similar, albeit less intuitive to explain, as shown in Figure 6. For DM, LE, and their Laplacian matrices, the effect was calculated in Coifman & Lafon (2006); they also showed that renormalization removes this bias (asymptotically). Moreover, the degree values d'_i obtained in the LAPLACIAN algorithm are estimators of the density around data point $\boldsymbol{x}_i$. An alternative method, applicable to low d, is to use a simple estimator of the local density and to use it to renormalize $\mathbf{L}^{rw}$ (Luo et al. 2009).

If enough samples are available, one can resample the data to obtain an approximately uniform distribution. For example, the *farthest point heuristic* chooses samples sequentially, with the next point being the farthest away from the already chosen points.

Effect of neighborhood graph (Figure 6) Radius neighbor graph of k-nearest neighbors? Ting et al. (2010) and later Calder & Trillos (2019) show that the k-nearest neighbor graph, with the similarity matrix with constant kernel $K(u) = 1$ exhibits qualitatively similar biases from non-uniform sampling as the normalized radius-neighbor graphs.

5.2. Choosing the scale of neighborhood

Whatever the task, a manifold learning method requires the user to provide an external parameter, be it the number of neighbors k or the kernel bandwidth h, that sets the scale of the local neighborhood.

Asymptotic results and what they mean The asymptotic results of Giné & Koltchinskii (2006), Hein et al. (2007), Ting et al. (2010) and Singer (2006) provide the necessary rates of change for h with respect to n to guarantee convergence of the respective estimate. For instance, Singer (2006) proves that the optimal bandwidth parameter for Laplacian estimation is given by $h \sim n^{-\frac{1}{d+6}}$ using a random-walk Laplacian. For the k-neareast neighbor graph, Calder & Trillos (2019) show that the number of neighbors k must grow slowly with n, and a recommended rate is $k \sim n^{\frac{4}{d+4}}(\log n)^{\frac{d}{d+4}}$, again for Laplacian estimation. The hidden constant factors in these rates are not completely known, but they depend on the (typically not known) manifold volume, curvature, and *injectivity radius* τ. Even so, these statistical results suggest that, in practice, the number of neighbors k should be sufficiently large and grow with n (Linderman & Steinerberger 2019).

With these rate-wise optimal selections of k or r, the convergence rate for estimating Laplacian operators, their eigenvectors, and so on, can be established. These rates are *non-parametric*, implying that the sample size n must grow *exponentially* with the dimension d. For example, using the previously mentioned rate of k, one can calculate that, for a 10-fold decrease in error, n must increase $\approx 10^{(d+4)/3}$-fold.

For neighbor embedding algorithms, such as t-SNE, less is known theoretically; however, practically, the defaults are for larger values of k, e.g., $k = 90$ (Poličar et al. 2019) and some research (Linderman & Steinerberger 2019) suggests $k \sim n$, which would create very dense graphs.

Practical methods Unfortunately, cross-validation (CV), a widely valuable model selection method in, e.g., density estimation, is not applicable in manifold learning for the lack of a criterion to cross-validate. (However, CV is still applicable in *semi-supervised* learning on manifolds (Belkin et al. 2006).) The ideas we describe below each mimic CV by choosing a criterion that measures the "self-consistency" of an embedding method at a particular scale.

For the k-nearest neighbor graph, Chen & Buja (2009) evaluates a given k with respect to the preservation of k' neighborhoods in the original graph. A problem to be aware of with this approach is that (see Section 5.5) most embeddings distort the data geometry. Hence Euclidean neighborhoods will not be preserved, even at the optimal k. A variable k method based on Topological Data Analysis (**?**) was proposed by Berry & Sauer (2019).

For the radius-neighbor graph, Perraul-Joncas & Meila (2013) exploit the connection between manifold geometry, represented by the Riemannian metric (see Section 5.5), and the Laplacian $\mathbf{L}$. The radius neighbor graph width h affects the Laplacian's ability to recognize the identity mapping. This method is specific to the DM algorithm, but the h obtained can be used by other embedding algorithms. Finally, we mention a dimension estimation algorithm proposed in Chen et al. (2013); a by-product of this algorithm is a range of scales r where the manifold looks locally linear, hence these scales would also be correct for the neighborhood graph.

5.3. Estimating the intrinsic dimension

Knowing the intrinsic dimension of data is important in itself. Additionally, some embedding algorithms (t-SNE, Isomap, LTSA) and all local PCA and Principal d-manifolds algorithms require the intrinsic dimension d as input.

How hard is dimension estimation? The dimension of a manifold is a non-negative integer, and therefore, intuitively, it should require fewer samples to estimate than a real-valued geometric parameter. Indeed, it is known (Kim et al. 2019, Genovese et al. 2012, Koltchinskii 2000) that the *minimax rate* for dimension estimation is exponential (i.e., the error is proportional to q^n for some $q < 1$) or faster. Unfortunately, the empirical experience belies the optimistic theoretical results. Due primarily to the presence of noise, which does not conform to simple assumptions, and secondarily to non-uniform sampling, estimating d for real data is a hard problem for which no satisfactorily robust solutions have been found yet (see Altan et al. (2020) for some empirical results).

Principles and methods for estimating d An idea that appears in various forms through the dimension estimation literature is to find a *local statistic* that scales with d by a known law. For example, the volume of a ball of radius r contained in a manifold $\mathcal{M}$ is proportional to r^d. Hence, $\log k_{i,r} \approx d \log r + \text{constant}$ (where $k_{i,r}$ is the number of radius r neighbors of data point x_i), and a regression line of $(\log r, \log \text{avg}(k_{i,r}))$ should have slope d. This is known as *correlation dimension* (Grassberger & Procaccia 1983). Other methods consider statistics such as $\frac{k_{i,2r}}{k_{i,r}} \approx 2^d$, or *covering number*, which lead respectively to the so-called *doubling dimensions* (Assouad 1983), and *Box Counting dimension* (Falconer 2003).

Modern estimators consider other statistics, such as distance to k-th nearest neighbor (Pettis et al. 1979, Costa et al. 2005), the volume of a spherical cap (Kleindessner & von Luxburg 2015) (both statistics can be computed without knowing actual distances, just comparisons between them), or Wasserstein distance between two samples of size n on $\mathcal{M}$, which scales like $n^{-1/d}$(Block et al. 2022); the algorithm of Levina & Bickel (2004), analyzed in Farahmand et al. (2007), proposes a Maximum Likelihood method based on k-nearest neighbor graphs.

An algorithm for dimension estimation in noise is proposed by Chen et al. (2013). The algorithm is based on the maximum eigengap of the local covariance matrix at multiple scales. This algorithm can be simplified by using a neighborhood radius selection algorithm such as Joncas et al. (2017) (Section 5.2).

5.4. Estimating the Laplace-Beltrami operator

We have seen that the eigenvectors of Laplacian-Beltrami operator $\Delta_{\mathcal{M}}$ can embed the data in low dimensions by the DM algorithm. Additionally, graph Laplacian estimators of $\Delta_{\mathcal{M}}$ are used in many different scenarios, described in Section 6.1. The question is which of the Laplacian matrices $\mathbf{L}$, $\mathbf{L}^{\text{norm}}$, $\mathbf{L}^{\text{rw}}$, etc., converge to $\Delta_{\mathcal{M}}$ when the sample size n tends to infinity?

Laplace-Beltrami operator: $\Delta_{\mathcal{M}} f \equiv \text{div grad}(f)$ plays a central role in modern differential geometry. See Sogge (2014), Rosenberg (1997) for details.

Denote the limit of the discrete operator $\mathbf{L}^{rw}$ by $\mathbf{L}^{\infty}$, a continuous differential operator acting on smooth functions. Two types of convergence, have been investigated. Pointwise convergence indicates the proximity of $(\mathbf{L}^{rw}\boldsymbol{f})_i$ to $\mathbf{L}^{\infty} f(\boldsymbol{x}_i)$, while spectral convergence involves the similarity between $\boldsymbol{f}\mathbf{L}^{un}\boldsymbol{f}/\boldsymbol{f}\mathbf{D}\boldsymbol{f}$ and the eigenvalues of $\mathbf{L}^{\infty}$.

When a radius neighbor graph is used, $\mathbf{L}^{\infty} = \Delta_{\mathcal{M}}$ is established for pointwise conver-

gence in the case of uniform sampling density, while $\mathbf{L}^\infty$ will be $\Delta_\mathcal{M}$+some density-related bias term in non-uniform case. Ting et al. (2010) demonstrated the pointwise convergence of the random-walk graph Laplacian to $\Delta_\mathcal{M}$ scaled by $p^{2/d}$ for $k-$nearest neighbor graphs, where p denotes the sampling density. Spectral convergence is similarly discussed in Belkin & Niyogi (2007), Berry & Sauer (2019), García Trillos & Slepčev (2018), García Trillos et al. (2020).

More broadly, an entire class of manifold learning algorithms can be studied by similar theoretical methods. Many embedding algorithms , including LE(Belkin & Niyogi 2003), DM(Coifman & Lafon 2006), LTSA(Zhang & Zha 2004), etc, that use matrices derived from the similarity $\mathbf{K}$ (called *linear smoothing* algorithms) are related to Laplacian-like second-order differential operator on $\mathcal{M}$. On the other hand, unregularized LLE fails to converge to any differential operator. Details can be found in Ting & Jordan (2018).

5.5. Embedding distortions. Is isometric embedding possible?

Figure 5 shows the outputs of various embedding algorithms on a simple 2-manifold $\mathcal{M} \subset \mathbb{R}^3$. It is easily seen that the results depend on the algorithm (and parameter choices) and the input (manifold and sampling density on $\mathcal{M}$). While most embedding algorithms work well, in the sense of producing smooth embeddings, the algorithm-dependent distortions, i.e., the local stretching or contraction – which amount to different coordinate systems – make these embeddings irreproducible and incomparable.

Empirical observations commonly reveal the presence of distortion. The distortions *do not disappear* when the sample size n increases, when the sampling density is uniform, or even when the consistent graph and Laplacian are used. This section is concerned with recovering reproducibility, by preserving the intrinsic geometry of the data.

Geodesic distances, intrinsic geometry and isometry For the data in Figure 1, a scientist may be interested in the distance between two molecular configurations $\boldsymbol{x}_1, \boldsymbol{x}_2$, seen as points of $\mathcal{M} \subset \mathbb{R}^D$. Their Euclidean distance $\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|$ is readily available. However, this value may not be of physical interest since most of the putative configurations along the segment $\boldsymbol{x}_1$ to $\boldsymbol{x}_2$ in $\mathbb{R}^D$ are not physically possible. To deform from state $\boldsymbol{x}_1$ to $\boldsymbol{x}_2$, the ethanol molecule must follow a path contained in (or near) the manifold $\mathcal{M}$ of possible configurations, and the distance $d_\mathcal{M}(\boldsymbol{x}_1, \boldsymbol{x}_2)$ shall naturally be defined as the shortest possible length of such a path; this is the *geodesic distance*. Geodesic distances, angles between curves in a manifold $\mathcal{M}$, and volumes of subsets of $\mathcal{M}$ represent *intrinsic* geometric quantities that can be defined without reference to the ambient space $\mathbb{R}^D$, and are independent of the choices of coordinate charts. Ideally, we would like an embedding (algorithm) to preserve these, and we call such an embedding an *isometric embedding*.

Attempts at isometric embedding Isometric (i.e., distortionless) embedding is possible, as proved by the celebrated Nash embedding theorem (Lee 2003) and more recently for Diffusion Maps by Bérard et al. (1994) and Portegies (2016). Unfortunately, these remarkable mathematical results are not easily amenable to numerically stable implementation.

Many ML methods focus on promoting isometry in local neighborhoods; Minimum Variance Unfolding aims to preserve local distances (Weinberger & Saul 2006), Conformal Eigenmap maps triangles in each neighborhood, thus preserving angle (Sha & Saul 2005), LTSA (Zhang & Zha 2004) and Local Linar Embedding (Roweis & Saul 2000)

preserve linear reconstructions. The works of Yu & Zhang (2010) and Lin et al. (2013) approach global isometry by means of constructing *normal coordinates* recursively from a point $\boldsymbol{p} \in \mathcal{M}$, or, respectively, by mutually orthogonal *parallel vector fields*, and Verma (2011) is the first attempt to implement Nash's construction. The ISOMAP algorithm (Tenenbaum et al. 2000) aims to preserve all shortest paths. We note that, with the exception of Verma (2011), and of ISOMAP for *flat* manifolds (i.e., manifolds that can be "unrolled" into $\mathbb{R}^d$ without stretching), these methods do not guarantee isometric embedding except in limited special cases.

Preserving isometry by estimating local distortion While finding a practical isometric embedding algorithm has been unsuccessful so far, estimating the local distortions is possible. Once the distortions are known, whenever a distance, angle, or volume is calculated, one applies local corrections that amount to obtaining the same result as if the embedding was isometric. The distortion at embedding point $\boldsymbol{y}_i = F(\boldsymbol{x}_i) \in \mathbb{R}^m$ is a symmetric, positive $m \times m$ matrix $\mathbf{H}_i$ of rank d. In Figure 7 the same embeddings of Figure 5 are shown, with $\mathbf{H}_i$ at selected points visualizing the local distortion induced by each algorithm. When the embedding F is isometric, and $m = d$, $\mathbf{H}_i = \mathbf{I}_d$ the unit matrix; otherwise, $\mathbf{H}_i$'s eigenvalues and vectors define the principal axes of stretch or compression around point i. A matrix function such as $\mathbf{H}$ on a manifold is called a *Riemannian metric* (see e.g. Perraul-Joncas & Meila (2013), Lee (2003)). The local correction at $\boldsymbol{y}_i$ is the pseudoinverse $\mathbf{G}_i$ of $\mathbf{H}_i$; $\mathbf{G}_i$ is also a Riemannian metric, called the *embedding (push-forward)* Riemannian metric.

With $\mathbf{G}_i$, the geodesic distance between $\boldsymbol{y}_i$ and a neighbor $\boldsymbol{y}_j$ is given[1] by

$$\hat{d}_{\mathcal{M}}(\boldsymbol{y}_i, \boldsymbol{y}_j)^2 \;\equiv\; \|\boldsymbol{y}_j - \boldsymbol{y}_i\|^2_{\mathbf{G}_i} \;=\; (\boldsymbol{y}_j - \boldsymbol{y}_i)^\top \mathbf{G}_i (\boldsymbol{y}_j - \boldsymbol{y}_i). \qquad 2.$$

For any other $\boldsymbol{y}_i, \boldsymbol{y}_j$, the geodesic distance is the shortest path length from $\boldsymbol{y}_i$ to $\boldsymbol{y}_j$ with the corrected distances above. The resulting distance is an undistorted approximation of the original. Perraul-Joncas & Meila (2013) proposed a method to estimate $\mathbf{H}_i$ for every embedded data point $\boldsymbol{y}_i$, using the renormalized Laplacian $\mathbf{L}$ described in Algorithm 1[2]. Hence, for any $\mathbf{Y} = F(\mathbf{X})$ output by an embedding algorithm, it is sufficient to estimate, at all points $\boldsymbol{y}_{1:n}$, the matrices $\mathbf{G}_{1:n}$, which represent the auxiliary information allowing one to correct distance computations in the non-isometric embedding F. The same $\mathbf{G}_{1:n}$ can be used to preserve not only geodesic distances but also other geometric quantities such as angles between curves in $\mathcal{M}$ or volumes of subsets of $\mathcal{M}$.

Normal coordinate chart: a specifically designe coordinate chart (U, φ) at each point $\boldsymbol{p} \in \mathcal{M}$. In $\varphi(U)$ (recall that this is a Euclidean space), the line given in polar coordinates $\theta = \theta_0$ must correspond to a geodesic on the manifold.

Estimating the metrics $\mathbf{H}_{1:n}$ and $\mathbf{G}_{1:n}$, offers even more insights into the embedding. For instance, the singular values of $\mathbf{H}_i$ (which has numeric rank m, but theoretical rank d) may offer a window into estimating d by looking for a singular value gap. The d singular vectors form an orthonormal basis of the tangent space to $F(\mathcal{M})$ at point $\boldsymbol{y}_i$, providing a natural framework for constructing a *normal* coordinate chart around $\boldsymbol{p}$.

The singular values of $\mathbf{H}_{1:n}$ can be used to evaluate the global distortion for an embedding as a criterion for comparing various embeddings. By iteratively minimizing this, one

[1] This is first-order approximation.

[2] To obtain $\mathbf{H}_i$, Perraul-Joncas & Meila (2013) apply $\mathbf{L}$ to a suitably chosen set of *test functions* $f_{kl,i}$, with $1 \le k \le l \le m$, where $f_{kl,i} = (F_k - F_k(\boldsymbol{x}_i))(F_l - F_l(\boldsymbol{x}_i))$ are pairwise products of coordinate functions, centered at point $\boldsymbol{x}_i$. They show that $\frac{1}{2}\Delta_{\mathcal{M}} f_{kl,i}(\boldsymbol{x}_i) = (\mathbf{H}_i)_{k,l}$, the k, l entry in $\mathbf{H}_i$ (algorithmically, this operation can be easily vectorized).

can get a more isometric embedding, such as in the RiemannianRelaxation of McQueen et al. (2016), which can be seen as an alternative to UMAP or t-SNE.

6. Applications of manifold learning

6.1. Manifold learning in statistics

Manifold learning with DM is closely related to *spectral clustering* (Shi & Malik 2000, Meilă & Shi 2001, Ng et al. 2001, von Luxburg 2007, Meilă 2016) as both methods map data to lower dimensions using the eigenvectors of a Laplacian. For clustering, it is preferable to employ $\mathbf{L}^{rw}$, the random walks Laplacian, which considers data density and enhances cluster separation. By mapping data to lower dimensions with $\mathbf{L}^{rw}$, a continuum between separated clusters (in clustered data) and smooth embedding (in regions where data lie on a manifold) can be observed. It even enables simultaneous embedding and clustering. Sufficient eigenvectors need to be calculated in such cases: $K-1$ eigenvectors indicate clustering for K clusters, and additional eigenvectors are required for low-dimensional mapping within each cluster. Using fewer eigenvectors may recover the clusters but not the intrinsic geometry within each cluster.

For a function $f : \mathcal{M} \to \mathbb{R}$, with $\boldsymbol{f} = [f(\boldsymbol{x}_i)]_{i=1:n}$, the functional $\frac{1}{2}\boldsymbol{f}^T\mathbf{L}\boldsymbol{f}$ approximates $\|\nabla f\|_2^2$ on the manifold, a measure of the *smoothness* of f (a function being smoother when its rate of change is lower). This smoothness measure can serve as a regularizer in supervised or semi-supervised learning on manifolds (Belkin et al. 2006, Slepčev & Thorpe 2019), Bayesian priors (Kirichenko & van Zanten 2017), and modeling Gaussian Processes on manifolds (Borovitskiy et al. 2020). If $\mathbf{L}^{nor}$ is used instead of $\mathbf{L}$ then the smoothness is calculated with respect to the sampling distribution on $\mathcal{M}$ (i.e. the rate of change is weighted more in regions with denser data).

6.2. Manifold learning for visualization

Embedding algorithms are often used in the sciences for data visualization. The scientists, as well as the statisticians, need to distinguish between an embedding as defined in Section 2, which preserves the geometric and topological data properties, and other mappings (occasionally also called "embeddings") into low dimensions using embedding algorithms. The latter kind of dimension reduction is hugely popular, and its value for the sciences cannot be underestimated. However, the users of dimension reduction for visualization should be cautioned that the scientific conclusions drawn from these visualizations must be subject to additional careful scrutiny or a more rigorous statistical and geometric analysis. One pitfall is that when data are mapped into $m = 2$ or 3 dimensions, for visualization, without estimating the intrinsic dimension d, the mapping may collapse together data regions that are not close in the original manifold. When clusters are present, because separating the clusters usually requires at least 2 dimensions, most of the clusters' geometric structure is collapsed. Hence, once the data is separated into clusters, the cluster structure needs to be studied by additional dimension reduction. A second pitfall is the presence of artifacts – interesting geometric features caused by the embedding algorithm but not supported by the data. These can be clusters (Figure 4), arms, holes or circles, and so on.

Before assigning scientific meaning to these features, a researcher should examine whether they are stable by repeating the embedding with different initial points, algorithms, and algorithm parameters, as well as by perturbing or resampling the original data.

To assess if the interesting features are not large distortions, visualizing the distortion (Figure 7) can provide valuable diagnostics. For example, when a "filament" is produced by stretching a low-density region, a very common effect (see Section 5.1), the estimated distortion will show the stretching (Figure 7), while for a true filament, the distortion will be moderate .

6.3. Manifold learning in the sciences

Astronomy and astrophysics Manifold learning has been used to study data from extensive astronomical surveys, like the Sloan Digital Sky Survey (SDSS)[3]. The mass distribution in the universe reveals *filaments*, i.e., one-dimensional manifolds, and dimension reduction methods, most often Principal Curves, have been used to estimate them (Chen et al. 2015).

Spectra of galaxies are measured in thousands of frequency bands; they contain rich data about galaxies' chemical and physical composition. By embedding these spectra in low dimensions, as in Figure 4, one can analyze the main constraints and pathways in the evolution of galaxies (Vanderplas & Connolly 2009).

Dynamical systems Dynamical systems described by Ordinary or Partial Differential Equations are intimately related to manifolds and exhibit multiscale behavior. Extensions of manifold learning can be used to understand PDE with geometric structure (Nadler et al. 2006), study the long-term behavior of the system or the ensemble of its solutions (Dsilva et al. 2016, 2018).

Chemistry The accurate simulation of atomical and molecular systems plays a significant role in modern chemistry. *Molecular Dynamics (MD)* simulations from carefully designed, complex quantic models can take millions of computer hours; however, simulations can still be less expensive than conducting experiments, and they return data at a level of detail not achievable in most experiments. Manifold learning is used to discover *collective coordinates*, i.e., low dimensional descriptors that approximate well the larger scale behavior of atomic, molecular, and other large particle systems (Boninsegna et al. 2015, A. et al. 2012, Noé & Clementi 2017). In these examples, the systems can be in equilibrium or evolving in time, and in the latter case, the collective coordinates describe the saddle points in the trajectory or the folding mechanism of a large molecule (Rohrdanz et al. 2011, Das et al. 2006).

Manifold embedding is also used to create low dimensional maps of families of molecules and materials by the similarity of their properties (Ceriotti et al. 2013, Isayev et al. 2015).

Biological sciences In neuroscience and the biological sciences, manifold embeddings are widely used to summarize neural recordings (Connor & Rozell 2016, Cunningham & Yu 2014), or to describe cell evolution (Herring et al. 2018)

7. Conclusion

In practice, ML is overwhelmingly used for visualization (Section 6) and with small data sets. But ML can do much more. Efficient software now exists (McQueen et al. 2016, Poličar et al. 2019) which can embed huge, high-dimensional data (for example, SDSS). In these

[3] `www.sdss.org`

cases, ML helps practitioners understand the data, by e.g., its intrinsic dimension, or by interpreting the manifold coordinates (Koelle et al. 2022, Boninsegna et al. 2015, Vanderplas & Connolly 2009). For real data, a manifold learning algorithm has the effect of smoothing the data and suppressing variation orthogonal to the manifold, which can be regarded as noise, just like in PCA. Finally, again similarly to PCA, ML can effectively reduce the data to $m \ll D$ dimensions while preserving features predictive for future statistical inferences. Some inferences, such as regression, can be performed on manifold data without manifold estimation by, for example, local linear regression (Aswani et al. 2011), or via Gaussian Processes (Borovitskiy et al. 2020). Even when only visualization is desired, care must be taken that the results are reproducible and free of artifacts, as discussed in Section 6.2.

What we omitted Among the topics we had to leave out, manifold learning in noise is perhaps the most important one. Noise makes ML significantly more difficult by introducing biases and slowing the convergence of estimators. This is an active area of research, but the estimation of geometric quantities like tangent space and reach in the presence of noise have been studied by Aamari & Levrard (2018, 2019); the theoretical results of manifold recovery in noise were mentioned in Section 3.

The *reach*, or *injectivity radius* $\tau(\mathcal{M})$ of manifold measures how close to itself $\mathcal{M}$ can be. In other words, $\tau(\mathcal{M})$ is the largest radius a ball can have, so that, for any $\boldsymbol{p} \in \mathcal{M}$, if it is tangent to the manifold in $\boldsymbol{p}$, it does not intersect $\mathcal{M}$ in any other point. Large τ implies larger curvature (a plane has infinite τ) and easier estimation of $\mathcal{M}$ (Genovese et al. 2012, Fefferman et al. 2016, Aamari & Levrard 2018, 2019). A manifold can have *borders*; ML with borders is studied. For example, in Singer & Wu (2012), different convergence rates appear when data are sampled close to the border.

A helpful task is to map a new data point $\boldsymbol{x} \in \mathbb{R}^D$ onto an existing embedding $F(\mathcal{M})$; this is often called *Nystrom* embedding (Chatalic et al. 2022). Conversely, if $\boldsymbol{y} \in \mathbb{R}^m$ is a new point on the embedding $F(\mathcal{M})$, obtained, e.g., by following a curve in the low dimensional representation of $\mathcal{M}$, how do we map it back to $\mathbb{R}^D$? This is usually done by interpolation.

Finally, a few words about *neural network representations*, such as *auto-encoders* (**?**), which could be seen as the fourth paradigm for manifold learning. We have left them out, partly for mathematical reasons. Although these mappings are generally smooth, there are no guarantees that they have constant rank d, even if the original data lie on a d-manifold. However, the main reason is that we could not do them justice in this review. Deep learning is an entirely different paradigm for non-linear dimension reduction. The intuitions and formal techniques for understanding neural networks' internal representations are entirely different from those surveyed here.

We surveyed the state-of-the-art knowledge on the main problems and methods of manifold learning, focusing on the algorithms proven to recover the manifold structure through learning a smooth embedding. There are many open problems in this field, though. Statistically, understanding of t-SNE and UMAP algorithms is still very limited, despite the fact that they are among the most popular visualization algorithms used today. More fundamentally, interpretation and validation of the output of an ML algorithm are also of importance to practitioners. An essential input to any ML algorithm is the distance used in finding neighbors and calculating the similarities. Currently, defining this distance (for example, by selecting which features of data point i should be included in $\boldsymbol{x}_i$, and in what units) is left entirely to the user.

Acknowledgements

The authors acknowledge the support from the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the Solar Energy Technologies Office Award Number DE-EE0008563 and from the National Science Foundation award DMS 2015272. They thank the Tkatchenko and Pfaendtner labs, particularly Stefan Chmiela and Chris Fu, for the molecular dynamics data and hours of discussions and brainstorming. MM gratefully acknowledges the Institute for Pure and Applied Mathematics (IPAM), and a Simons Fellowship from IPAM, during Fall 2019.

LITERATURE CITED

A. TG, Ceriotti M, Parrinello M. 2012. Using sketch-map coordinates to analyze and bias molecular dynamics simulations. *Proceedings of the National Academy of Science, USA* 109:5196—-201

Aamari E, Levrard C. 2018. Stability and minimax optimality of tangential delaunay complexes for manifold reconstruction. *Discrete & Computational Geometry* 59(4):923–971

Aamari E, Levrard C. 2019. Nonasymptotic rates for manifold, tangent space and curvature estimation. *Ann. Stat.* 47(1):177–204

Altan E, Solla SA, Miller LE, Perreault EJ. 2020. Estimating the dimensionality of the manifold underlying multi-electrode neural recordings. *bioRxiv*

Arias-Castro E, Pelletier B. 2013. On the convergence of maximum variance unfolding. *Journal of Machine Learning Research* 14:1747–1770

Assouad P. 1983. Plongements lipschitziens dans $\{\{r\}\}^n$. *Bulletin de la Société Mathématique de France* 111:429–448

Aswani A, Bickel P, Tomlin C. 2011. Regression on manifolds: Estimation of the exterior derivative. *The Annals of Statistics* 39(1):48–81

Belkin M, Niyogi P. 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation* 15(6):1373–1396

Belkin M, Niyogi P. 2007. Convergence of laplacian eigenmaps. In *Advances in Neural Information Processing Systems 19*, eds. B Schölkopf, JC Platt, T Hoffman. MIT Press, 129–136

Belkin M, Niyogi P, Sindhwani V. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research* 7(85):2399–2434

Bérard P, Besson G, Gallot S. 1994. Embedding Riemannian manifolds by their heat kernel. *Geometric Functional Analysis* 4(4):373–398

Bernstein M, de Silva V, Langford JC, Tennenbaum J. 2000. Graph approximations to geodesics on embedded manifolds. `http://web.mit.edu/cocosci/isomap/BdSLT.pdf`

Berry T, Sauer T. 2019. Consistent manifold representation for topological data analysis

Block A, Jia Z, Polyanskiy Y, Rakhlin A. 2022. Intrinsic dimension estimation using wasserstein distance. *Journal of Machine Learning Research* 23(313):1–37

Boninsegna L, Gobbo G, Noé F, Clementi C. 2015. Investigating molecular kinetics by variationally optimized diffusion maps. *Journal of chemical theory and computation* 11(12):5947–5960

Borovitskiy V, Terenin A, Mostowsky P, Deisenroth (he/him) M. 2020. Matérn gaussian processes on riemannian manifolds, In *Advances in Neural Information Processing Systems*, eds. H Larochelle, M Ranzato, R Hadsell, M Balcan, H Lin, vol. 33, pp. 12426–12437, Curran Associates, Inc.

Böhm JN, Berens P, Kobak D. 2022. Attraction-repulsion spectrum in neighbor embeddings. *Journal of Machine Learning Research* 23(95):1–32

Calder J, Trillos NG. 2019. Improved spectral convergence rates for graph laplacians on epsilon-graphs and k-nn graphs. *ArXiv* abs/1910.13476

Carreira-Perpiñan MA. 2010. The elastic embedding algorithm for dimensionality reduction, ICML'10, p. 167–174, Madison, WI, USA: Omnipress

Ceriotti M, Tribello GA, Parrinello M. 2013. Demonstrating the transferability and the descrip-

tive power of sketch-map. *Journal of Chemical Theory and Computation* 9(3):1521–1532PMID: 26587614

Chatalic A, Schreuder N, Rosasco L, Rudi A. 2022. Nyström kernel mean embeddings, In *Proceedings of the 39th International Conference on Machine Learning*, eds. K Chaudhuri, S Jegelka, L Song, C Szepesvari, G Niu, S Sabato, vol. 162 of *Proceedings of Machine Learning Research*, pp. 3006–3024, PMLR

Chen G, Little AV, Maggioni M. 2013. Multi-resolution geometric analysis for data in high dimensions. Boston: Birkhäuser Boston, 259–285

Chen L, Buja A. 2009. Local Multidimensional Scaling for nonlinear dimension reduction, graph drawing and proximity analysis. *Journal of the American Statistical Association* 104(485):209–219

Chen YC, Genovese CR, Wasserman L. 2015. Asymptotic theory for density ridges. *The Annals of Statistics* 43(5):1896–1928

Chen YC, Meila M. 2021. The decomposition of the higher-order homology embedding constructed from the k-laplacian, In *Advances in Neural Information Processing Systems*, eds. M Ranzato, A Beygelzimer, Y Dauphin, P Liang, JW Vaughan, vol. 34, pp. 15695–15709, Curran Associates, Inc.

Chmiela S, Tkatchenko A, Sauceda H, Poltavsky I, Schütt KT, Müller KR. 2017. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*

Coifman RR, Lafon S. 2006. Diffusion maps. *Applied and Computational Harmonic Analysis* 30(1):5–30

Connor M, Rozell C. 2016. Unsupervised learning of manifold models for neural coding of physical transformations in the ventral visual pathway, In *Neural Information Processing Systems (NIPS) Workshop, Brains and Bits: Neuroscience Meets Machine Learning*. Barcelona, Spain

Costa J, Girotra A, Hero A. 2005. Estimating local intrinsic dimension with k-nearest neighbor graphs, In *IEEE/SP 13th Workshop on Statistical Signal Processing, 2005*, pp. 417–422

Cunningham JP, Yu BM. 2014. Dimensionality reduction for large-scale neural recordings. *Nature Neuroscience* 16:1500—-1509

Das P, Moll M, Stamati H, Kavraki L, Clementi C. 2006. Low-dimensional, free-energy landscapes of protein-folding reactions by nonlinear dimensionality reduction. *Proceedings of the National Academy of Sciences* 103(26):9885–9890

Diaconis P, Goel S, Holmes S. 2008. Horseshoes in multidimensional scaling and local kernel methods. *The Annals of Applied Statistics* 2(3):777 – 807

do Carmo M. 1992. Riemannian geometry. Springer

Donoho DL, Grimes C. 2003. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences* 100(10):5591–5596

Dsilva CJ, Talmon R, Coifman RR, Kevrekidis IG. 2018. Parsimonious representation of nonlinear dynamical systems through manifold learning: A chemotaxis case study. *Appl. Comput. Harmon. Anal.* 44(3):759–773

Dsilva CJ, Talmon R, Gear CW, Coifman RR, Kevrekidis IG. 2016. Data-driven reduction for a class of multiscale fast-slow stochastic dynamical systems. *SIAM J. Appl. Dyn. Syst.* 15(3):1327–1351

Falconer K. 2003. Alternative definitions of dimension, chap. 3. John Wiley & Sons, Ltd, 39–58

Farahmand Am, Szepesvári C, Audibert JY. 2007. Manifold-adaptive dimension estimation, In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, p. 265–272, New York, NY, USA: Association for Computing Machinery

Fefferman C, Mitter S, Narayanan H. 2016. Testing the manifold hypothesis. *J. Amer. Math. Soc.* 29(4):983–1049

García Trillos N, Gerlach M, Hein M, Slepčev D. 2020. Error estimates for spectral convergence of the graph laplacian on random geometric graphs toward the laplace–beltrami operator. *Foundations of Computational Mathematics* 20(4):827–887

García Trillos N, Slepčev D. 2018. A variational approach to the consistency of spectral clustering.

Applied and Computational Harmonic Analysis 45(2):239–281

Genovese CR, Perone-Pacifico M, Verdinelli I, Wasserman LA. 2012. Minimax manifold estimation. *Journal of Machine Learning Research* 13:1263–1291

Giné E, Koltchinskii V. 2006. Concentration inequalities and asymptotic results for ratio type empirical processes. *The Annals of Probability* 34(3):1143 – 1216

Goldberg Y, Zakai A, Kushnir D, Ritov Y. 2008. Manifold learning: The price of normalization. *Journal of Machine Learning Research* 9(63):1909–1939

Grassberger P, Procaccia I. 1983. Measuring the strangeness of strange attractors. *Physica D: Nonlinear Phenomena* 9(1):189–208

Hein M, Audibert J, von Luxburg U. 2007. Graph laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research* 8:1325–1368

Herring CA, Banerjee A, McKinley ET, Simmons AJ, Ping J, et al. 2018. Unsupervised trajectory analysis of Single-Cell RNA-Seq and imaging data reveals alternative tuft cell origins in the gut. *Cell Syst* 6(1):37–51.e9

Hinton GE, Roweis S. 2002. Stochastic neighbor embedding, In *Advances in Neural Information Processing Systems*, eds. S Becker, S Thrun, K Obermayer, vol. 15. MIT Press

Im DJ, Verma N, Branson K. 2018. Stochastic neighbor embedding under f-divergences

Isayev O, Fourches D, Muratov EN, Oses C, Rasch K, et al. 2015. Materials cartography: Representing and mining materials space using structural and electronic fingerprints. *Chemistry of Materials* (27):735–743

I.T.Jolliffe. 2002. Principal component analysis. Springer Series in Statistics. Springer New York, NY

Jacomy M, Venturini T, Heymann S, Bastian M. 2014. Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PLOS ONE* 9(6):1–12

Joncas D, Meila M, McQueen J. 2017. Improved graph laplacian via geometric Self-Consistency. In *Advances in Neural Information Processing Systems 30*, eds. I Guyon, UV Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, R Garnett. Curran Associates, Inc., 4457–4466

Kim J, Rinaldo A, Wasserman LA. 2019. Minimax rates for estimating the dimension of a manifold. *J. Comput. Geom.* 10(1):42–95

Kirichenko A, van Zanten H. 2017. Estimating a smooth function on a large graph by Bayesian Laplacian regularisation. *Electronic Journal of Statistics* 11(1):891 – 915

Kleindessner M, von Luxburg U. 2015. Dimensionality estimation without distances, In *AISTATS*

Kobak D, Linderman G, Steinerberger S, Kluger Y, Berens P. 2020. Heavy-tailed kernels reveal a finer cluster structure in t-sne visualisations, In *Machine Learning and Knowledge Discovery in Databases*, eds. U Brefeld, E Fromont, A Hotho, A Knobbe, M Maathuis, C Robardet, pp. 124–139, Cham: Springer International Publishing

Koelle SJ, Zhang H, Meila M, Chen YC. 2022. Manifold coordinates with physical meaning. *Journal of Machine Learning Research* 23(133):1–57

Kohli D, Cloninger A, Mishne G. 2021. Ldle: Low distortion local eigenmaps. *Journal of Machine Learning Research* 22(282):1–64

Koltchinskii VI. 2000. Empirical geometry of multivariate data: a deconvolution approach. *The Annals of Statistics* 28(2):591 – 629

Lee JM. 2003. Introduction to smooth manifolds. Springer-Verlag New York

Levina E, Bickel PJ. 2004. Maximum likelihood estimation of intrinsic dimension, In *Advances in Neural Information Processing Systems 17 NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pp. 777–784

Lin B, He X, Zhang C, Ji M. 2013. Parallel vector field embedding. *Journal of Machine Learning Research* 14(90):2945–2977

Linderman GC, Steinerberger S. 2019. Clustering with t-sne, provably. *SIAM Journal on Mathematics of Data Science* 1(2):313–332

Luo C, Safa I, Wang Y. 2009. Approximating gradients for meshes and point clouds via diffusion metric. *Computer Graphics Forum* 28(5):1497–1508

McInnes L, Healy J, Melville J. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*

McQueen J, Meila M, Joncas D. 2016. Nearly isometric embedding by relaxation, In *Advances in Neural Information Processing Systems*, eds. D Lee, M Sugiyama, U Luxburg, I Guyon, R Garnett, vol. 29. Curran Associates, Inc.

McQueen J, Meila M, VanderPlas J, Zhang Z. 2016. Megaman: Scalable manifold learning in python. *Journal of Machine Learning Research* 17

Meilă M, Shi J. 2001. A random walks view of spectral segmentation, In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, eds. TS Richardson, TS Jaakkola, vol. R3 of *Proceedings of Machine Learning Research*, pp. 203–208, PMLR. Reissued by PMLR on 31 March 2021.

Meilă M. 2016. Spectral clustering : a tutorial for the 2010 ' s

Mohammed K, Narayanan H. 2017. Manifold learning using kernel density estimation and local principal components analysis. *arxiv* 1709.03615

Nadler B, Lafon S, Coifman R, Kevrekidis I. 2006. Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators, In *Advances in Neural Information Processing Systems 18*, eds. Y Weiss, B Schölkopf, J Platt, pp. 955–962, Cambridge, MA: MIT Press

Ng A, Jordan M, Weiss Y. 2001. On spectral clustering: Analysis and an algorithm, In *Advances in Neural Information Processing Systems*, eds. T Dietterich, S Becker, Z Ghahramani, vol. 14. MIT Press

Noé F, Clementi C. 2017. Collective variables for the study of long-time kinetics from molecular trajectories: theory and methods. *Current Opinion in Structural Biology* 43:141–147

Ozertem U, Erdogmus D. 2011. Locally defined principal curves and surfaces. *Journal of Machine Learning Research* 12(34):1249–1286

Perraul-Joncas D, Meila M. 2013. Non-linear dimensionality reduction: Riemannian metric estimation and the problem of geometric discovery. *ArXiv e-prints*

Perrault-Joncas D, Meila M. 2014. Improved graph laplacian via geometric self-consistency. *ArXiv e-prints*

Pettis KW, Bailey TA, Jain AK, Dubes RC. 1979. An intrinsic dimensionality estimator from near-neighbor information. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1(1):25–37

Poličar PG, Stražar M, Zupan B. 2019. opentsne: a modular python library for t-sne dimensionality reduction and embedding. *bioRxiv*

Portegies JW. 2016. Embeddings of Riemannian manifolds with heat kernels and eigenfunctions. *Communications on Pure and Applied Mathematics* 69(3):478–518

Ram P, Lee D, March W, Gray A. 2009. Linear-time algorithms for pairwise statistical problems, In *Advances in Neural Information Processing Systems*, eds. Y Bengio, D Schuurmans, J Lafferty, C Williams, A Culotta, vol. 22. Curran Associates, Inc.

Rohrdanz MA, Zheng W, Maggioni M, Clementi C. 2011. Determination of reaction coordinates via locally scaled diffusion map. *The Journal of chemical physics* 134(12)

Rosenberg S. 1997. The laplacian on a riemannian manifold: An introduction to analysis on manifolds. London Mathematical Society Student Texts. Cambridge University Press

Roweis S, Saul L. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290(5500):2323–2326

Sha F, Saul LK. 2005. Analysis and extension of spectral methods for nonlinear dimensionality reduction, ICML '05. New York, NY, USA: Association for Computing Machinery

Shi J, Malik J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8):888–905

Singer A. 2006. From graph to manifold laplacian: The convergence rate. *Applied and Computa-*

tional Harmonic Analysis 21(1):128–134Special Issue: Diffusion Maps and Wavelets

Singer A, Wu HT. 2012. Vector diffusion maps and the connection laplacian. *Communications on Pure and Applied Mathematics* 65(8):1067–1144

Slepčev D, Thorpe M. 2019. Analysis of p-laplacian regularization in semisupervised learning. *SIAM Journal on Mathematical Analysis* 51(3):2085–2120

Sogge CD. 2014. Hangzhou lectures on eigenfunctions of the laplacian. Princeton University Press

Tenenbaum JB, de Silva V, Langford JC. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–2323

Ting D, Huang L, Jordan MI. 2010. An analysis of the convergence of graph laplacians, In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 1079–1086

Ting D, Jordan MI. 2018. On nonlinear dimensionality reduction, linear smoothing and autoencoding. *arXiv: Machine Learning*

Ting D, Jordan MI. 2020. Manifold learning via manifold deflation

van der Maaten L. 2014. Accelerating t-sne using tree-based algorithms. *Journal of Machine Learning Research* 15(93):3221–3245

van der Maaten L, Hinton G. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research* 9:2579–2605

Vanderplas J, Connolly A. 2009. Reducing the dimensionality of data: Locally linear embedding of sloan galaxy spectra. *The Astronomical Journal* 138(5):1365

Verma N. 2011. Towards an algorithmic realization of nash ' s embedding theorem

von Luxburg U. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17(4):395–416

Weinberger KQ, Saul LK. 2006. An introduction to nonlinear dimensionality reduction by maximum variance unfolding, In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pp. 1683–1686, AAAI Press

Yu K, Zhang T. 2010. Improved local coordinate coding using local tangents, In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, p. 1215–1222, Madison, WI, USA: Omnipress

Zha H, Zhang Z. 2007. Continuum isomap for manifold learnings. *Computational Statistics & Data Analysis* 52(1):184–200

Zhang Y, Gilbert AC, Steinerberger S. 2022. May the force be with you, In *58th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2022, Monticello, IL, USA, September 27-30, 2022*, pp. 1–8, IEEE

Zhang Y, Steinerberger S. 2021. t-sne, forceful colorings and mean field limits. *CoRR* abs/2102.13009

Zhang Z, Zha H. 2004. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM J. Scientific Computing* 26(1):313–338

Supplementary Materials

A. Details of "one-shot" manifold learning algorithms

A.1. Principal Component Analysis (PCA)

Linear dimension reduction methods find a global embedding of the data in a low-dimensional linear subspace. One way of understanding Principal Component Analysis is to find a d dimensional linear subspace $\mathcal{V}$ such that the data $\{\boldsymbol{x}_i\}$ projected onto it have the smallest reconstruction error. Let $\mathcal{V}$ have an orthogonal basis $\mathbf{T} \in \mathbb{R}^{D\times d}$ such that $\mathbf{T}^\top\mathbf{T} = \mathbf{I}_d$. Then $\boldsymbol{x}_i$ projected onto $\mathcal{V}$ has low dimensional representation $\boldsymbol{y}_i = \mathbf{T}^\top\boldsymbol{x}_i$ under basis $\mathbf{T}$. In $\mathbb{R}^D$, projection of $\boldsymbol{x}_i$ onto $\mathcal{V}$ is given by $\mathbf{T}\mathbf{T}^\top\boldsymbol{x}_i$. If we introduce the data matrix $\mathbf{X} \in \mathbf{R}^{n\times D}$, with $i-$th row being $\boldsymbol{x}_i^\top$, then the low dimensional representation matrix $\mathbf{Y} \in \mathbb{R}^{n\times d}$ is given by $\mathbf{X}\mathbf{T}$ and in $\mathbb{R}^D$ the projected data matrix is $\mathbf{X}\mathbf{T}\mathbf{T}^\top$.

Then we can write the PCA problem as

$$\min_{\mathbf{T}:\mathbf{T}\in\mathbb{R}^{D\times d},\mathbf{T}^\top\mathbf{T}=\mathbf{I}_d} \sum_{i=1}^{n} ||\boldsymbol{x}_i - \mathbf{T}\mathbf{T}^\top\boldsymbol{x}_i||^2 = \min_{\mathbf{T}:\mathbf{T}\in\mathbb{R}^{D\times d},\mathbf{T}^\top\mathbf{T}=\mathbf{I}_d} ||\mathbf{X} - \mathbf{X}\mathbf{T}\mathbf{T}^\top||_F^2 \tag{3.}$$

Consider the singular value decomposition preserving only the first d singular values of $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{n\times d}, \mathbf{V} \in \mathbb{R}^{D\times d}$ are orthogonal matrices and $\mathbf{\Sigma}$ is $d \times d$ diagonal matrix, then the solution to problem 3. is $\mathbf{T} = \mathbf{V}$. The low dimensional representation of original data is $\mathbf{Y} = \mathbf{X}\mathbf{T} = \mathbf{U}\mathbf{\Sigma}$; the coordinates of $\mathbf{Y}$ are called *principal components*. In the terminology of PCA, columns of $\mathbf{V}$ are called principal vectors.

When the data $\boldsymbol{x}_{1:n}$ are centered, the (unnormalized) sample covariance matrix of the data is $\mathbf{C} = \mathbf{X}^\top\mathbf{X}$. The principal vectors characterize the d directions that explain the most variance in the data. The solution to PCA can also be found by eigendecomposition of $\mathbf{C}$. The first d eigenvectors of $\mathbf{C}$ are just the matrix $\mathbf{V}$. If the dimension $D \gg n$, it will be easier first to compute the Gram matrix $\mathbf{C} = \mathbf{X}^\top\mathbf{X}$ and then perform a truncated eigendecomposition $\mathbf{C} = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^\top$; the low dimensional representation is still $\mathbf{X}\mathbf{V}$.

A.2. Multidimensional Scaling (MDS)

MDS is dealing with a different problem compared with PCA: given an $n \times n$ distance matrix $\mathbf{M} = [\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2]_{ij}$, find a low dimensional representations $\{\boldsymbol{y}_i\}_{i=1}^n$ in $\mathbb{R}^d$ such that all distances are preserved. Mathematically, we want to minimize

$$\min_{\boldsymbol{y}_1,\cdots,\boldsymbol{y}_n\in\mathbb{R}^d} \sum_{i,j=1}^{n} \text{Loss}(\text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 - ||\boldsymbol{y}_i - \boldsymbol{y}_j||^2) \tag{4.}$$

The MDS problem is hard to solve for generic loss function and metric function. However, it is always possible to use numerical optimization methods to obtain local solutions to such problems.

In the case that $\mathbf{M}$ contains squared Euclidean distances, one can double center the squared pairwise distance matrix by constructing matrix $\mathbf{B} = \mathbf{H}_n\mathbf{M}\mathbf{H}_n$ with $\mathbf{H}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, then the solution of the MDS problem will be given by finding eigenvectors of $-\frac{1}{2}\mathbf{B}$.

The objective of MDS differs from that of ML. In Section 5.5, we explain that when data $\boldsymbol{x}_i$ are high dimensional points in $\mathbb{R}^D$, the Euclidean distances $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ do not represent geodesic distances in the manifold, unless $\boldsymbol{x}_i, \boldsymbol{x}_j$ are neighbors. (Otherwise, $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$, as a "shortcut" through ambient space, will be typically shorter than the geodesic distance.)

Since by equation 4., MDS aims to preserve all distances, MDS would not be able to "unfold" a manifold onto a lower dimensional space. The following section will show how the Isomap algorithm uses MDS to achieve unfolding.

A.3. Isomap

ISOMAP is a generalization of Multidimensional Scaling that preserves distances between data points while finding low dimensional coordinates. Instead of Euclidean distance in classical MDS, ISOMAP use shortest path distances in the neighborhood distance graph to approximate geodesic distance on a manifold. Intuitively, the shortest graph distance

Algorithm 4 ISOMAP

Input: : Neighborhood distance matrix $\mathbf{A}$, embedding dimension m

1: Compute shortest path distance matrix $\tilde{\mathbf{A}}_{ij}$:

$$\tilde{\mathbf{A}}_{ij} = \begin{cases} \mathbf{A}_{ij} & \mathbf{A}_{ij} < \infty \,, \\ \text{shortest path distance between } i, j & \mathbf{A}_{ij} = \infty \,. \end{cases}$$

2: Multidimensional Scaling $\mathbf{Y} = \text{MDS}(\mathbf{M}, d)$ with $\mathbf{M} = [\tilde{\mathbf{A}}_{ij}^2]$

Output: : m dimensional coordinates $\mathbf{Y}$ for $\mathcal{D}$

approximates the geodesic distance in a neighborhood provided that data are sufficiently dense in this region and neighborhood size is appropriately chosen (Bernstein et al. 2000). In the limit of large n, ISOMAP was shown to produce isometric embeddings for $m = d$, whenever the data manifold is *flat*, i.e. admits an isometric embedding in $\mathbb{R}^d$, and data space is convex. Empirically, ISOMAP embeddings are close to isometric also when $m > d$ and m is sufficient for isometric embedding.

The computation complexity of ISOMAP is $O(n^3)$, with the most computational burden for computing all pairs of shortest path distance. Space complexity is $O(n^2)$. Since ISOMAP works with dense matrices, this space complexity cannot be improved.

There are variants of Isomap that improve it in different ways: Hessian Eigenmaps (Donoho & Grimes 2003) enables non-convex data where they introduce the use of Hessian operator; Continuum Isomap (Zha & Zhang 2007) generalizes Isomap to a continuous version such that out-of-sample extension of Isomap is possible.

A.4. Local Tangent Space Alignment (LTSA)

This algorithm, proposed in (Zhang & Zha 2004) seeks to find a local representation in the tangent space b$\mathcal{T}_{\boldsymbol{x}_i}\mathcal{M}$ at each point $\boldsymbol{x}_i$, then aligns these to obtain global coordinates.

The first stage of LTSA finds the local representation of neighboring points $j \in \mathcal{N}_i$ via projections on the tangent space $\mathcal{T}_{\boldsymbol{x}_i}\mathcal{M}$; thus $\boldsymbol{y}_j - \boldsymbol{y}_i$ can locally be approximated by an affine transformation of orthogonal projections of $\boldsymbol{x}_j$ onto tangent space at $\boldsymbol{x}_i$ through Taylor expansion. The optimal affine transformation is obtained by minimizing the reconstruction error near each $\boldsymbol{x}_i$

$$\min_{\tilde{\boldsymbol{x}}_i, \Theta, \mathbf{Q}} \sum_{j \in \mathcal{N}_i} \|\boldsymbol{x}_j - (\tilde{\boldsymbol{x}}_i + \mathbf{Q}\theta_j^{(i)})\|^2 \,, \qquad 5.$$

where $\boldsymbol{x}, \mathbf{Q}$ are translation and rotation that parametrize this affine transformation; θ_j is

a local coordinate of each neighbor $\boldsymbol{x}_j$ projected on this linear subspace.

In the second stage of LTSA, one obtains global embedding coordinates $\mathbf{Y}$ while θ_j that preserves local geometry information through minimizing a global reconstruction error.

$$\min_{\{\boldsymbol{y}_i\}_{i=1}^n, \{\mathbf{P}_i\}_{i=1}^n} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \|\boldsymbol{y}_j - \tilde{\boldsymbol{y}}_i - \mathbf{P}_i \theta_j^{(i)}\|^2 \tag{6.}$$

The optimization in both steps can be transformed into eigenvalue problems. Hence the algorithmic procedure of LTSA is displayed in Algorithm 5

Algorithm 5 Local tangent space alignment

Input: Dataset $\mathcal{D}$, embedding dimension m.
  $\mathbf{B} = 0$
  for $i = 1, 2, \cdots, n$ **do**
    Find the k nearest neighbors of $\boldsymbol{x}_i$: $\boldsymbol{x}_j, j \in \mathcal{N}_i$.
    Find local dataset $\boldsymbol{\Xi}_i = [\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_i]_{j \in \mathcal{N}_i}$, where $\tilde{\boldsymbol{x}}_i$ is the average of all neighbors of $\boldsymbol{x}_i$.
    Compute the m largest eigenvectors $\tilde{\boldsymbol{v}}^1, \cdots, \tilde{\boldsymbol{v}}^d$ of $\boldsymbol{\Xi}^\top \boldsymbol{\Xi}$, set $\mathbf{G}_i = [\mathbf{1}/\sqrt{k}, \tilde{\boldsymbol{v}}^1, \cdots, \tilde{\boldsymbol{v}}^d]$
    $\mathbf{B} = \mathbf{B} + \mathbf{I} - \mathbf{G}_i \mathbf{G}_i^\top$.
  end for
  Compute the 2 to $m+1$ smallest eigenvectors of $\mathbf{B}$, $\{\boldsymbol{v}^j\}_{j=1}^m$, each eigenvector $\boldsymbol{v}^j \in \mathbb{R}^n$.
Output: m dimensional embeddings $\boldsymbol{y}_i = (v_i^1, \cdots, v_i^m)$, for $i = 1, \ldots n$.

B. Illustration of the local distortion (Section 5.5)

While the matrices $\mathbf{g}_i$, not shown, represent the local correction, their inverses $\mathbf{H}_i$ measure the local distortion. They are show here as ellipses, whose principal axes are along the principal directions of stretch/contraction. Hence, the shape and size of the ellipses represents the directional strectching at the givent point.

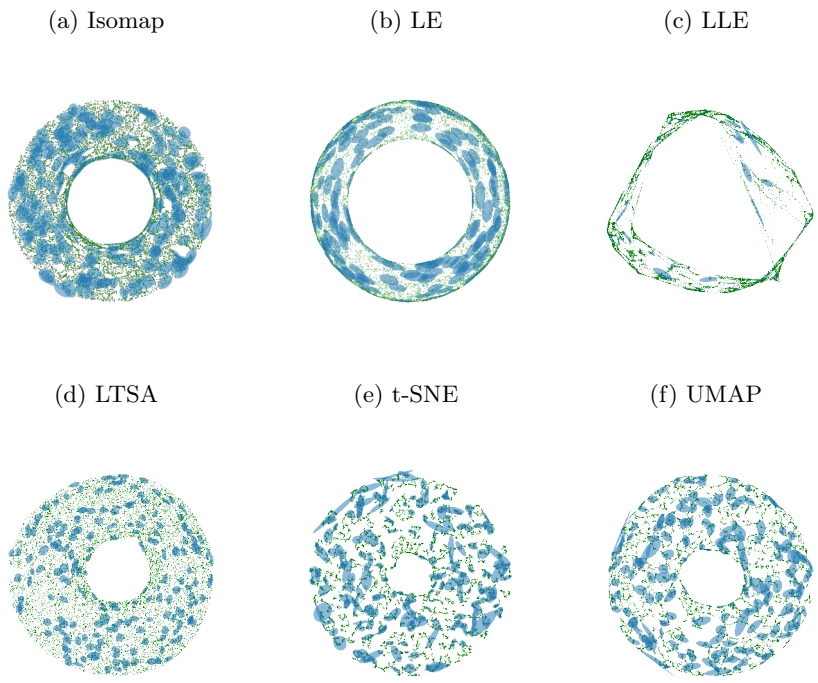

Figure 7: The embeddings from Figure 5, with the distortion **H** estimated at a random subset of points. The principal axes of the ellipses are proportional to the singular values of **H** at each point.