Annual Review of Statistics and Its Application

Manifold Learning: What, How, and Why

Marina Meilă[1] and Hanyu Zhang[2]

[1]Department of Statistics, University of Washington, Seattle, Washington, USA; email: mmp@stat.washington.edu

[2]ByteDance, Inc., Bellevue, Washington, USA

Annu. Rev. Stat. Appl. 2024. 11:393–417

First published as a Review in Advance on November 29, 2023

The *Annual Review of Statistics and Its Application* is online at statistics.annualreviews.org

https://doi.org/10.1146/annurev-statistics-040522-115238

Keywords

nonlinear dimension reduction, manifold learning, embedding

Abstract

Manifold learning (ML), also known as nonlinear 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 article reviews the mathematical background, methodology, and recent developments in nonlinear dimension reduction techniques. These techniques have been developed for two decades since the publication of 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 (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 a 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 review, 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 (ML), with three possible subparadigms, each producing a different representation of the data manifold. The rest of the

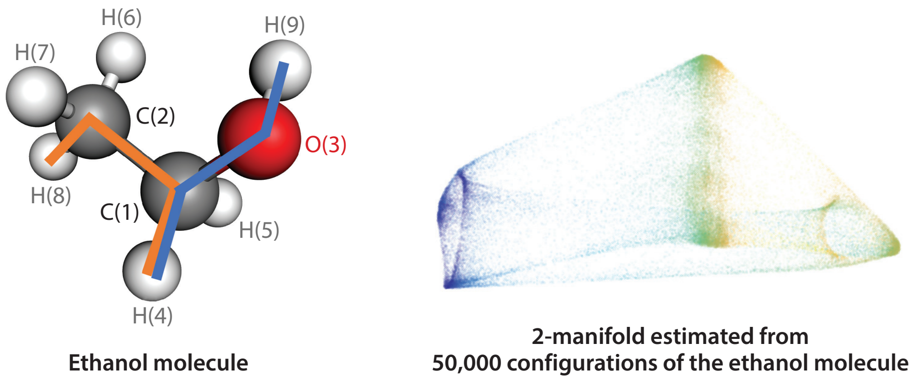
H(6)
H(9)
H(7)
C(2)
O(3)
H(8)
C(1)
H(5)
H(4)
Ethanol molecule
2-manifold estimated from
50,000 configurations of the ethanol molecule

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, and 8) and the OH group (atoms 3 and 9) can rotate with respect to the middle group (atoms 1, 4, and 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). Right panel was created by Samson J. Koelle and adapted with permission from Koelle et al. (2022), copyright 2022.

article focuses on one of these, 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, and intrinsic dimension. Section 6 discusses applications, connecting with related statistics problems, and Section 7 concludes the review. For a more technical treatment of Sections 2, 3, and 5, the reader is referred to Meilă & Zhang (2023).

Smooth: a function f is smooth when it is differentiable and its derivatives are continuous

2. MATHEMATICAL BACKGROUND: MANIFOLDS, COORDINATE CHARTS, EMBEDDINGS

2.1. Manifolds and Coordinate Charts

Readers are referred to Lee (2003) and do Carmo (1992) for a rigorous introduction to manifolds and differential geometry and to Meilă & Zhang (2023) for a slightly more detailed presentation. Intuitively, a manifold is a generalization of curves and surfaces with coordinate systems (called charts). On objects like a sphere or torus (**Figure 2**), 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 charts are and why and when they can be ignored in everyday work with manifold data.

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

Hence, a manifold has a Euclidean coordinate system (the chart) locally around every 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 and a torus (**Figure 2**) are also 2-manifolds, but they cannot be covered by a single chart (they each require at least two), as cartographers well know.

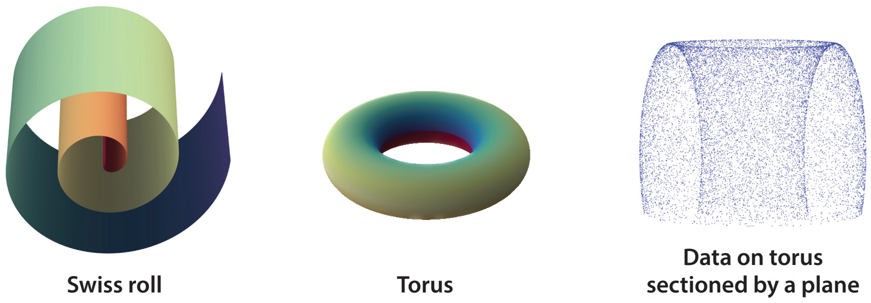

Figure 2

Examples of manifolds with intrinsic dimension $d = 2$. (*Left*) A Swiss roll. (*Middle*) A torus (hollow). (*Right*) 1,000 points sampled from a torus sectioned by a plane.

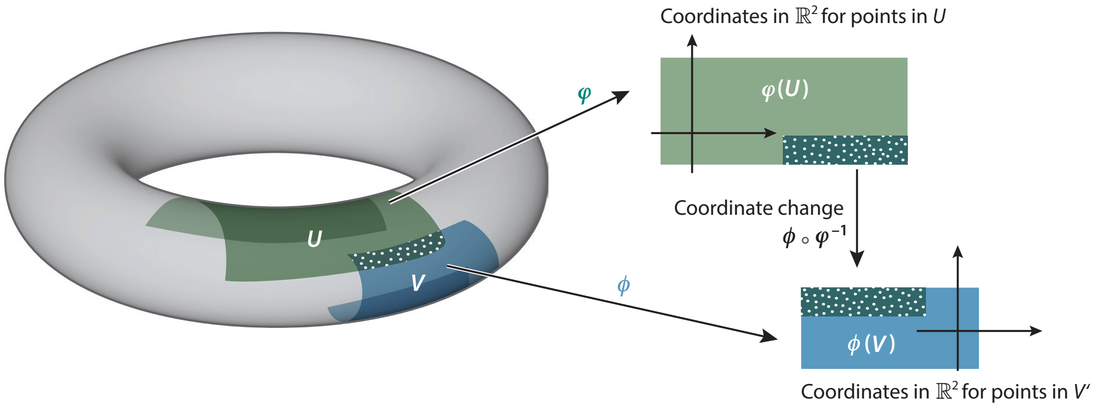
Coordinates in $\mathbb{R}^2$ for points in U
φ
$\varphi(U)$
U
V
Coordinate change
$\phi \circ \varphi^{-1}$
ϕ
$\phi(V)$
Coordinates in $\mathbb{R}^2$ for points in V'

Figure 3

Manifold and charts. The torus is a manifold with 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 V, can. The price paid is that each patch now has a different coordinate system, and to travel on the torus, one must apply coordinate changes. The dotted areas represent, from left to right, the points in $U \cap V$, their mapping into $\mathbb{R}^2$ by φ, and the mapping of the same by ϕ. The coordinate change is $\phi \circ \varphi^{-1}$: $\varphi(U \cap V) \to \phi(U \cap V)$.

Coordinate charts are not unique; there are infinitely many coverings of a manifold 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 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 (*a*) they can work in the coordinate system of their choice, and intrinsic quantities like d will remain invariant, but (*b*) 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.

2.2. 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 ML algorithms.

2.3. 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

3.1. 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 review, with a few exceptions, we discuss the no-noise case when the data lie on $\mathcal{M}$.

3.2. Manifold Learning

An ML 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. When $\mathbb{P}$ is supported exactly on $\mathcal{M}$, and the sample size $n \to \infty$, a valid ML 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 $\mathcal{M}$.

3.3. Can a Manifold be Estimated?

The manifold assumption itself is testable. For example, Fefferman et al. (2016) test whether, given an independent and identically distributed sample, there exists a manifold $\mathcal{M}$ that can approximate this sample with tolerance ε. These results are currently not practically useful, as several 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.

3.4. Neighborhood Graphs

Practically all ML 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 nonlinear 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. In the following, 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 defines the neighborhood scale, similar to a bandwidth parameter in kernel density estimation. Consistency of ML 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 with respect to 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 and 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 either is 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 nonincreasing 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, Coifman & Lafon 2006, Ting et al. 2010, 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).

3.5. 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 (more details about these can be found in Meilă & Zhang 2023). Then, from Section 4, we focus on the third class, consisting of algorithms that produce embeddings, global representations in low dimensions. The three paradigms are summarized in **Table 1**.

3.5.1. Linear local approximation. The idea of linear local approximation is derived from classical PCA, 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 a specific fixed reference point. To cover the entire manifold, one needs to obtain multiple such charts.

3.5.2. Principal curves and principal d-manifolds. In the paradigm of principal curves and principal d-manifolds, 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 the 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, and the bias depends on the manifold's curvature: The density is higher on the inside of the curve. However, for their smoothing

Table 1 Three main paradigms for nonlinear 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)

property, principal d-manifolds are remarkably useful in analyzing manifold estimation in noise (Genovese et al. 2012, Mohammed & Narayanan 2017).

4. EMBEDDING ALGORITHMS

The term manifold learning was proposed in the works of Roweis & Saul (2000) and Tenenbaum et al. (2000), who introduced the Locally Linear Embedding (LLE) and Isomap[1] algorithms, inaugurating the modern era of nonlinear dimension reduction. In this section, we introduce classical ML algorithms that aim to find a global embedding $\boldsymbol{y}_1, \ldots \boldsymbol{y}_n$, also denoted $\mathbf{Y} \in \mathbb{R}^{n\times m}$ (with $\boldsymbol{y}_i$ representing row i of $\mathbf{Y}$), of dataset $\mathcal{D}$.

Algorithms can be broadly categorized into one-shot algorithms, 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 $\boldsymbol{x}_1, \ldots \boldsymbol{x}_n$ that 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.

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 **Supplemental Appendix A**, along with PCA and multidimensional scaling.

Supplemental Material >

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

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 $\mathbf{D} = \text{diag}\{d_1, \ldots, d_n\}$. Then, multiple choices of graph Laplacian exist:

- Unnormalized Laplacian: $\mathbf{L}^{\text{un}} = \mathbf{D} - \mathbf{K}$
- Normalized Laplacian: $\mathbf{L}^{\text{norm}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$

LAPLACIAN MATRIX

A Laplacian matrix is a specialization of the Laplacian differential operator $\Delta f = \sum_j \frac{\partial^2 f}{\partial x_j^2}$ to 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, we have $\Delta f(i) = \frac{1}{h}\left[\frac{f_{i+1}-f_i}{h} - \frac{f_i - f_{i-1}}{h}\right] = \frac{1}{h^2}\left(\sum_{j\in\mathcal{N}_i} f_j - d_i f_i\right) = \mathbf{L}^{\text{un}} f(i)$ for all $i = 2, \ldots, n-1$, because $d_i = 2$ for these nodes.

[1]Throughout this article, names in small capitals (e.g., Isomap) designate algorithms that appear in the text or the **Supplemental Appendix**.

- Random-walk Laplacian: $\mathbf{L}^{\text{rw}} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{K}$
- Renormalized or DM Laplacian $\mathbf{L}$, as defined in Algorithm 1 below

```
Algorithm 1 (Laplacian).
Input: Similarity matrix $\mathbf{K}$, kernel bandwidth $h$
  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: $\mathbf{L} = (\mathbf{I} - \mathbf{P})/h^2$
```

Why choose one Laplacian rather than another? Even though in simple examples the difference is hard to spot, as more samples are collected, one needs to ensure that the limit of these $\mathbf{L}$ matrices is well defined and the embedding algorithm is unbiased (see Sections 5.1 and 5.4). It is easy to see that $\mathbf{L}^{\text{norm}}$ and $\mathbf{L}^{\text{rw}}$ are similar matrices. Moreover, whenever the degrees d_i are constant, $\mathbf{L} \propto \mathbf{L}^{\text{rw}} \propto \mathbf{L}^{\text{un}}$, hence all Laplacians produce the same embedding. Differences arise when data density is nonuniform, 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 $\mathbf{L}^{\text{norm}}$ and $\mathbf{L}^{\text{rw}}$ are biased by the sampling density and that renormalization removes this bias. This is illustrated in Section 5.4 (and especially the accompanying **Figure 6**).

Using the defined graph Laplacian matrix $\mathbf{L}$, we can summarize the DM procedure, presented in Algorithm 2.

```
Algorithm 2 (Diffusion Maps/Laplacian Eigenmaps).
Input: Laplacian $\mathbf{L}$ (or $\mathbf{L}^{\text{norm}}$), 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; see Meilă & Shi 2001).
  3. Represent each data point $j$ by $\boldsymbol{y}_j = (v_j^1, \ldots, v_j^m)^\top \in \mathbb{R}^m$.
Output: $\mathbf{Y}$
```

Similar to PCA, the data are mapped to the principal directions 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 LE algorithm resembles DM but uses a different Laplacian, namely $\mathbf{L}^{\text{norm}}$ above.

The idea of spectral embedding also appeared independently in graph visualization, then was used by Shi & Malik (2000) as a method for clustering, and was then generalized as a data representation method by Belkin & Niyogi (2003) as LE. They connect the Laplacian matrix 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 is reviewed in Section 5.4.

4.2. Horseshoe Effects, Neighbor Embedding Algorithms, and Selecting Independent Eigenvectors

Now we turn to attraction–repulsion algorithms, sometimes known as neighbor embedding algorithms. We start by highlighting a pitfall of standard one-shot algorithms that attraction–repulsion algorithms usually avoid. After these algorithms are presented, we briefly return to describe how one-shot algorithms may be corrected, too.

4.2.1. The repeated eigenvectors problem. Algorithms that use eigenvectors, such as DM, are among the most promising and well-studied in ML (see Sections 5.1, 5.2, and 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) by

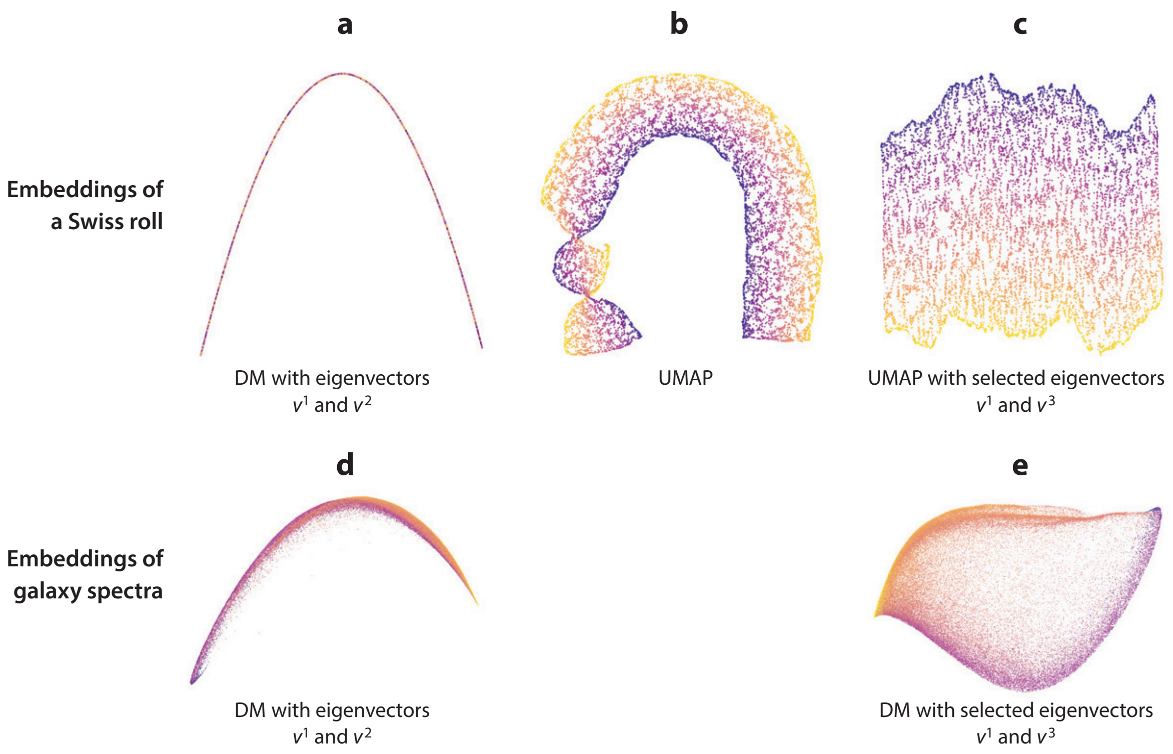

Figure 4

Examples of embedding algorithms failing to find a full-rank mapping, if they greedily select the first $m = 2$ eigenvectors, and of corrections by a more refined choice of eigenvectors. (*a*) Embeddings of a Swiss roll with length seven times the width by DM. The first two eigenvectors form a 1-dimensional curve—a horseshoe. Hence, $\mathbf{v}^2$ does not add a new dimension, but repeats $\mathbf{v}^1$. (*b*) When the same data are embedded with 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. (*c*) UMAP embedding of the same data as in panel *a*, with selection of eigenvectors by Chen & Meilă (2019). In panels *d* and *e*, galaxy spectra from the SDSS (Section 6) are embedded by DM. In panel *d*, the first two eigenvectors, v^1 and v^2, are used, and the embedding results in a horseshoe, while in panel *e*, eigenvectors $\mathbf{v}^1$ and $\mathbf{v}^3$, selected by Chen & Meilă (2019), are used. Plots by Yu-Chia Chen, adapted with permission from Chen & Meilă (2019). Abbreviations: DM, Diffusion Maps; UMAP, Uniform Manifold Approximation and Projection; SDSS, Sloan Digital Sky Survey.

Dsilva et al. (2018). The REP has been demonstrated theoretically for DM/LE, LTSA, and LLE (Goldberg et al. 2008), and in real datasets.

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 $(\mathbf{v}^1, \mathbf{v}^2)$ 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 $(\mathbf{v}_i^1, \mathbf{v}_i^2)_{i=1,\dots n}$ from step 3 of the DM follows a parabola. This is a relevant diagnosis for the REP in practice: When an embedding looks like a horseshoe, this may represent not a property of the data but an artifact signaling 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, Carreira-Perpiñan 2010, Jacomy et al. 2014, Im et al. 2018, McInnes et al. 2018). Usually, the embedding coordinates **Y** are optimized iteratively until equilibrium is reached.

The T-SNE algorithm of van der Maaten & Hinton (2008), one variant of which we briefly describe here (Böhm et al. 2022), 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). van der Maaten & Hinton (2008) proposed using a Student's t-distribution to model the output similarities and, as T-SNE, this algorithm became widely used.

```
Algorithm 3 (T-SNE).
Input: Similarity matrix $\mathbf{K}$ (from $k$-nearest neighbor graph), initial embedding $\boldsymbol{y}_1, \ldots \boldsymbol{y}_n$, step size $\eta$,
       repulsion parameter $\rho$
  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 $\mathbf{A}^{\text{out}}_{ij} = \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2$, for $i, j = 1, \ldots n$
  4.   Compute similarities in embedding space $\mathbf{W}_{ij} = \frac{1}{1+\mathbf{A}^{\text{out}}_{ij}}$, for $i, j = 1, \ldots n$, $w_{\text{tot}} = \sum_{i,j=1}^{n} \mathbf{W}_{ij}$
  5.   Update embedding by $\boldsymbol{y}_i \leftarrow \boldsymbol{y}_i + \eta\left[\sum_{j \in \mathcal{N}_i} \mathbf{V}_{ij}\mathbf{W}_{ij}(\boldsymbol{y}_i - \boldsymbol{y}_j) - \frac{n}{\rho}\sum_{j=1}^{n} \frac{\mathbf{W}_{ij}}{w_{\text{tot}}}(\boldsymbol{y}_i - \boldsymbol{y}_j)\right]$
  6. end while
Output: $\mathbf{Y}$
```

Uniform Manifold Approximation and Projection (UMAP) (McInnes et al. 2018) is another popular heuristic method. On a high level, UMAP minimizes the mismatches between topological representations of high-dimensional dataset $\{\boldsymbol{x}_i\}_{i=1}^{n}$ 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 of Algorithm 3. The first term in the change of $\boldsymbol{y}_i$ is an attraction between graph neighbors, 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. Böhm et al. (2022) show 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). Zhang & Steinerberger (2022) also analyze the attraction–repulsion behavior of T-SNE. 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 by Zhang et al. (2022).

Finally, in Minimum Variance Unfolding (MVU) (Weinberger & Saul 2006, Arias-Castro & Pelletier 2013), repulsion is implemented via a semidefinite program; hence, the embedding **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 RiemannianRelaxation algorithm of McQueen et al. (2016) (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 repeated eigendirections problem in spectral embeddings. The REP has a theoretically straightforward solution for algorithms like DM and LTSA. From the sequence of eigenfunctions $F^1, \ldots, F^{m'}$ on $\mathcal{M}$ (or eigenvectors $\boldsymbol{v}^1, \ldots, \boldsymbol{v}^{m'}$ in the finite sample case), with $m' > m$, sorted by their corresponding eigenvalues, one needs to select $F^{j_1} = F^1$, then (recursively) $F^{j_2}, \ldots F^{j_m}$ so that the rank of the Jacobian matrix $[(\mathrm{d}F^1)_{\boldsymbol{p}}, \ldots (\mathrm{d}F^{j_m})_{\boldsymbol{p}}]$ is d at every point $\boldsymbol{p} \in \mathcal{M}$. For example, 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 the Jacobian at the data points. Dsilva et al. (2018) proposed to measure dependence by regressing $\boldsymbol{v}_{j_{k+1}}$ on the previously selected $\boldsymbol{v}^{j_1, \ldots j_k}$; Chen & Meilă (2019) derived a condition number from the embedding metric (Section 5.5) and used it 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 eigenvectors optimized with respect to rank. Finally, 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 the REP and is a first step toward 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 postprocessing 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 present representative algorithms of two types. One-shot algorithms (typically) embed the data by eigenvectors, of which Isomap, DM, and LTSA are the best understood and most computationally scalable. The main drawback of this class of algorithms is the REP, which requires postprocessing 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 the REP affecting one-shot algorithms. Quantifying the repulsion, smoothness, large-sample limits, and other properties of the neighbor embedding algorithms is 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; see, e.g., Linderman & Steinerberger 2019).

Neither algorithm guarantees against local singularities, such as the knots 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 illustrated in **Figure 5** and 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 large n. To compound this problem, finding k or r in a principled way 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 (McQueen et al. 2016a). 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

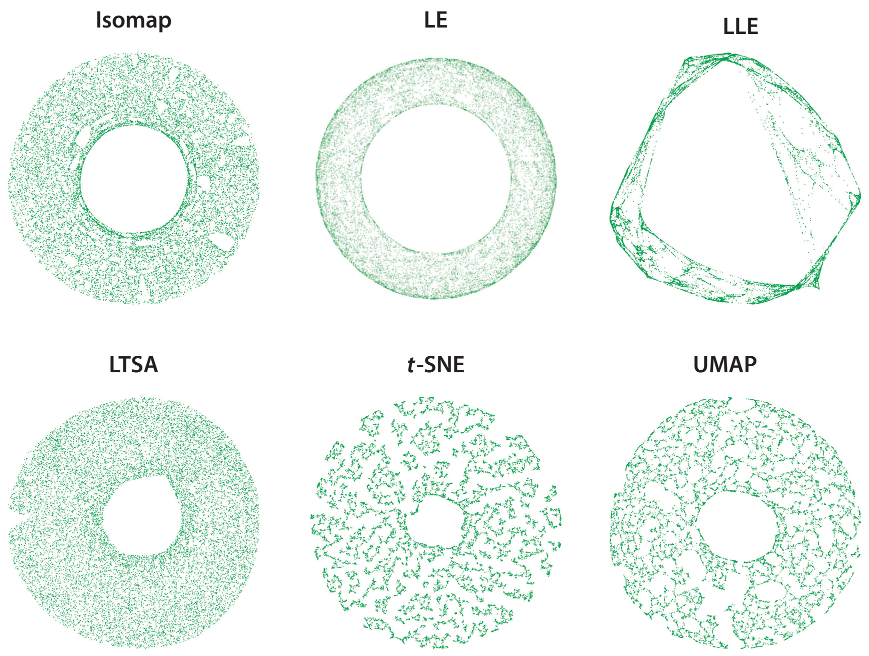

Figure 5

Supplemental Material >

Embeddings of the sectioned torus data from **Figure 2** by various algorithms; Isomap and LTSA are described in **Supplemental Appendix A**. This manifold cannot be embedded isometrically in $d = 2$ dimensions; each algorithm distorts (stretches or contracts) it differently. **Figure 7** visualizes the local distortions. Abbreviations: LE, Laplacian Eigenmaps; LLE, Locally Linear Embedding; LTSA, Local Tangent Space Alignment; UMAP, Uniform Manifold Approximation and Projection.

Barnes-Hut trees approximation (van der Maaten 2014) and the negative sampling heuristic for UMAP (McInnes et al. 2018, Böhm et al. 2022).

5. STATISTICAL BASIS OF MANIFOLD LEARNING

The output or result of ML algorithms depends critically on algorithm parameters such as the type of neighborhood graph (k-NN or radius neighbor), the neighborhood scale (k or r), and the 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 $\Delta_{\mathcal{M}}$ (the limit of $\mathbf{L}$) as the natural representation of the manifold geometry, and the basis for the DM embedding, 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. For a slightly more technical treatment of Sections 5.2–5.5, the reader is referred to Meilă & Zhang (2023).

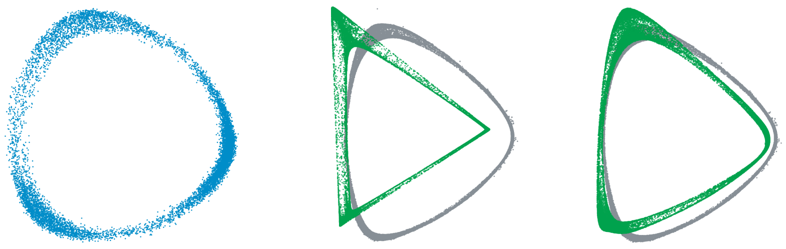

Figure 6

Effects of graph construction and renormalization, when the sampling density is highly nonuniform, exemplified on the configurations of the ethanol molecule shown in **Figure 1**. (*Left*) The 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. (*Middle*) Embeddings by DM (*gray*), and by the LE, which uses the singly normalized $\mathbf{L}^{\text{rw}}$ (*green*). The sparse regions are stretched, while the dense regions appear like corners of the embedding. Note that DM should be immune to the effects of the density; in this case, the variations in density are so extreme that a slight effect persists. (*Right*) Embedding by DM (*gray*) and Algorithm 2 with $\mathbf{L}$ constructed from the k-nearest neighbor graph (*green*). Abbreviations: DM, Diffusion Maps; LE, Laplacian Eigenmaps. Data are from Chmiela et al. (2017).

5.1. Biases in Manifold Learning: Effects of Sampling Density and Graph Construction

It is little known in practice and not intuitively obvious that the shape of the embedding $y_1, \ldots, y_n$ reflects not only the shape of the original manifold, but also the way the data were sampled, and whether a radius-neighbor or a k-NN graph was used.

5.1.1. Biases due to nonuniform 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 by 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 $\mathbf{L}^{\text{rw}}$ (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.

5.1.2. Effect of neighborhood graph. Ting et al. (2010), and later Calder & Trillos (2022), showed that the k-NN graph, with the similarity matrix with constant kernel $K(u) = 1$, exhibits qualitatively similar biases from nonuniform sampling as the normalized radius-neighbor graphs, as seen in **Figure 6** (right panel).

5.2. Choosing the Scale of Neighborhood

Whatever the task, an ML 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.

5.2.1. 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 $h \sim n^{-\frac{1}{d+6}}$ when using a random-walk Laplacian. For the k-NN graph, Calder & Trillos (2022) 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 (Lee 2003). 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 nonparametric, 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.

5.2.2. Practical methods. Unfortunately, cross-validation (CV), a widely valuable model selection method in, e.g., density estimation, is not applicable in ML due to the lack of a criterion to cross-validate. [However, CV is still applicable in semisupervised 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-NN graph, Chen & Buja (2009) evaluate 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 (see Wasserman 2018) was proposed by Berry & Sauer (2019).

For the radius-neighbor graph, Perrault-Joncas et al. (2017) 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 by Chen et al. (2013); a by-product of this algorithm is a range of scales, r, where the manifold looks locally linear, and 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.

5.3.1. How hard is dimension estimation? The dimension of a manifold is a nonnegative integer, and therefore, intuitively, it should require fewer samples to estimate than a real-valued geometric parameter. Indeed, the minimax rate for dimension estimation is exponential (i.e., the error is proportional to q^n for some $q < 1$) or faster (Koltchinskii 2000, Genovese et al. 2012, Kim et al. 2019). 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 nonuniform sampling, estimating d for real data is a hard problem for which no satisfactorily robust solutions have been found yet (for some empirical results, see Altan et al. 2021).

Laplace–Beltrami operator: $\Delta_{\mathcal{M}} f \equiv \text{div grad}(f)$, the extension of the Laplace operator Δ for functions $f : \mathbb{R} \to \mathbb{R}$, plays a central role in modern differential geometry

5.3.2. 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, $\log k_{i,r} \approx d \log r +$ 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 the kth 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 by Farahmand et al. (2007), proposes a maximum likelihood method based on k-NN 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 that of Perrault-Joncas et al. (2017) (Section 5.2).

5.4. Estimating the Laplace–Beltrami Operator

We have seen that the eigenvectors of the Laplace–Beltrami operator $\Delta_{\mathcal{M}}$ (for details, see Rosenberg 1997, Sogge 2014) 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.

Denote the limit of the discrete operator $\mathbf{L}^{\text{rw}}$ 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 $(\mathbf{L}^{\text{rw}}\boldsymbol{f})_i$ to $\mathbf{L}^{\infty} f(\boldsymbol{x}_i)$, while spectral convergence involves the similarity between the eigenvalues of $\mathbf{L}^{\text{rw}}$ and the eigenvalues of $\mathbf{L}^{\infty}$.

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

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

5.5.1. Geodesic distances, intrinsic geometry and isometry.

For the data in **Figure 1**, a scientist may be interested in the distance between two molecular configurations $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, 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.

5.5.2. 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 DM 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; MVU aims to preserve local distances (Weinberger & Saul 2006); Conformal Eigenmap maps triangles in each neighborhood, thus preserving angle (Sha & Saul 2005); and LTSA (Zhang & Zha 2004) and Locally Linear 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) makes 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 that 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.

5.5.3. 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 $m = d$, $\mathbf{H}_i = \mathbf{I}_d$ denotes 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., Perrault-Joncas & Meilă 2013, Lee 2003). The local correction at $\boldsymbol{y}_i$ is

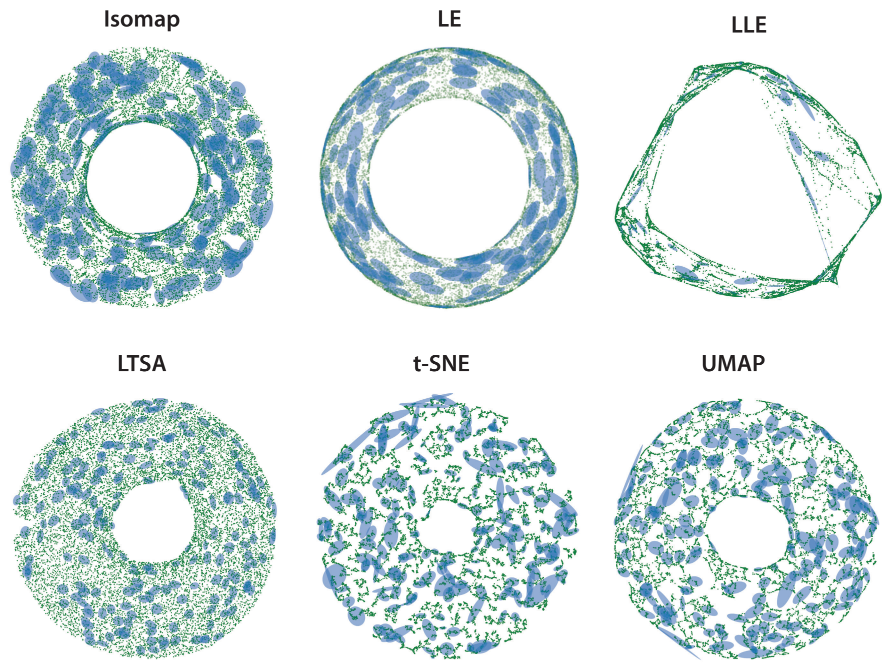

Figure 7

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. Note the very elongated ellipses indicating extreme stretching especially in the LLE, t-SNE, and UMAP embeddings, as well as the random directions of stretch for t-SNE. Abbreviations: LE, Laplacian Eigenmaps; LLE, Locally Linear Embedding; LTSA, Local Tangent Space Alignment; UMAP, Uniform Manifold Approximation and Projection.

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 (to first-order approximation) 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. Perrault-Joncas & Meilă (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 nonisometric 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 $\mathcal{M}$.

[2]To obtain $\mathbf{H}_i$, Perrault-Joncas & Meilă (2013) apply $\mathbf{L}$ to a suitably chosen set of test functions $f_{kl,i}$, with $1 \le k \le l \le m$, and $i = 1, \ldots n$, 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).

Normal coordinate chart: at point $\boldsymbol{p}$, a special coordinate system such that geodesics emanating from $\boldsymbol{p}$ are straight lines

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 enabling us to look 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 normal coordinate charts.

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 can get a more isometric embedding, such as in the RiemannianRelaxation algorithm 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

ML with DM is closely related to spectral clustering (Shi & Malik 2000, Meilă & Shi 2001, Ng et al. 2001, von Luxburg 2007, Meilă 2015) as both methods map data to lower dimensions using the eigenvectors of a Laplacian. For clustering, it is preferable to employ $\mathbf{L}^{\text{rw}}$, the random-walk Laplacian, which considers data density and enhances cluster separation. By mapping data to lower dimensions with $\mathbf{L}^{\text{rw}}$, 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 $\frac{1}{2}\boldsymbol{f}^\top \mathbf{L}\boldsymbol{f}$ 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 semisupervised 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 $\mathbf{L}^{\text{norm}}$ 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 are 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, 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

Manifold learning has been used as a tool for scientific research due to its ability to reveal the shape of complex data. Here, we provide some examples.

6.3.1. Astronomy and astrophysics. ML has been used to study data from extensive astronomical surveys, like the Sloan Digital Sky Survey (SDSS). The mass distribution in the universe reveals filaments, i.e., 1-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 compositions. 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).

6.3.2. Dynamical systems. Dynamical systems described by ordinary or partial differential equations are intimately related to manifolds and exhibit multiscale behavior. Extensions of ML can be used to understand partial differential equations with geometric structure (Nadler et al. 2006) and to study the long-term behavior of the system or the ensemble of its solutions (Dsilva et al. 2016, 2018).

6.3.3. Chemistry. The accurate simulation of atomic and molecular systems plays a significant role in modern chemistry. Molecular dynamics 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. ML 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 (Tribello et al. 2012, Boninsegna et al. 2015, 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).

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

7. CONCLUSION

In practice, ML is overwhelmingly used for visualization (Section 6) and with small datasets. But ML can do much more. Efficient software now exists (McQueen et al. 2016b, Poličar et al. 2019) that can embed huge, high-dimensional data (for example, from SDSS). In these cases, ML helps practitioners understand the data, e.g., by its intrinsic dimension, or by interpreting the manifold coordinates (Vanderplas & Connolly 2009, Boninsegna et al. 2015, Koelle et al. 2022). For real data, an ML 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, e.g., by 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.

7.1. What We Omitted

Among the topics we had to leave out, ML in noise is perhaps the most important. 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 a 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 less 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, and ML with borders has been studied. For example, Singer & Wu (2012) show that 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, we add a few words about neural network representations, such as auto-encoders (Goodfellow et al. 2016), which could be seen as the fourth paradigm for ML. 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 nonlinear dimension reduction. The intuitions and formal techniques for understanding neural networks' internal representations are entirely different from those surveyed here.

7.2. Open Problems

We surveyed the state-of-the-art knowledge of the main problems and methods of ML, 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.

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

The authors acknowledge the support from the US 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

Aamari E, Levrard C. 2018. Stability and minimax optimality of tangential Delaunay complexes for manifold reconstruction. *Discrete Comput. Geom.* 59(4):923–71

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. 2021. Estimating the dimensionality of the manifold underlying multi-electrode neural recordings. *PLOS Comput. Biol.* **https://doi.org/10.1371/journal.pcbi.1008591**

Arias-Castro E, Pelletier B. 2013. On the convergence of maximum variance unfolding. *J. Mach. Learn. Res.* 14:1747–70

Assouad P. 1983. Plongements lipschitziens dans $\{\{r\}\}^n$. *Bull. Soc. Math. France* 111:429–48

Aswani A, Bickel P, Tomlin C. 2011. Regression on manifolds: estimation of the exterior derivative. *Ann. Stat.* 39(1):48–81

Belkin M, Niyogi P. 2003. Laplacian Eigenmaps for dimensionality reduction and data representation. *Neural Comput.* 15(6):1373–96

Belkin M, Niyogi P. 2007. Convergence of Laplacian Eigenmaps. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, ed. B Schölkopf, JC Platt, T Hoffman, pp. 129–36. Cambridge, MA: MIT Press

Belkin M, Niyogi P, Sindhwani V. 2006. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* 7(85):2399–434

Bérard P, Besson G, Gallot S. 1994. Embedding Riemannian manifolds by their heat kernel. *Geom. Funct. Anal.* 4(4):373–98

Berry T, Sauer T. 2019. Consistent manifold representation for topological data analysis. *Found. Data Sci.* 1(1):1–38

Böhm JN, Berens P, Kobak D. 2022. Attraction-repulsion spectrum in neighbor embeddings. *J. Mach. Learn. Res.* 23(95):1–32

Block A, Jia Z, Polyanskiy Y, Rakhlin A. 2022. Intrinsic dimension estimation using Wasserstein distance. *J. Mach. Learn. Res.* 23(313):1–37

Boninsegna L, Gobbo G, Noé F, Clementi C. 2015. Investigating molecular kinetics by variationally optimized diffusion maps. *J. Chem. Theory Comput.* 11(12):5947–60

Borovitskiy V, Terenin A, Mostowsky P, Deisenroth MP. 2020. Matérn Gaussian processes on Riemannian manifolds. In *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, ed. H Larochelle, M Ranzato, R Hadsell, M Balcan, H Lin, pp. 12426–37. Red Hook, NY: Curran

Calder J, Trillos NG. 2022. Improved spectral convergence rates for graph Laplacians on ϵ-graphs and k-NN graphs. *Appl. Comput. Harmon. Anal.* 60:123–75

Carreira-Perpiñan MA. 2010. The elastic embedding algorithm for dimensionality reduction. In *ICML'10: Proceedings of the 27th International Conference on International Conference on Machine Learning*, ed. J Fürnkranz, T Joachims, pp. 167–74. Madison, WI: Omnipress

Ceriotti M, Tribello GA, Parrinello M. 2013. Demonstrating the transferability and the descriptive power of sketch-map. *J. Chem. Theory Comput.* 9(3):1521–32

Chatalic A, Schreuder N, Rosasco L, Rudi A. 2022. Nyström kernel mean embeddings. *Proc. Mach. Learn. Res.* 162:3006–24

Chen G, Little AV, Maggioni M. 2013. Multi-resolution geometric analysis for data in high dimensions. In *Excursions in Harmonic Analysis*, Vol. 1, ed. TD Andrews, R Balan, JJ Benedetto, W Czaja, KA Okoudjou, pp. 259–85. Boston: Birkhäuser

Chen L, Buja A. 2009. Local multidimensional scaling for nonlinear dimension reduction, graph drawing and proximity analysis. *J. Am. Stat. Assoc.* 104(485):209–19

Chen YC, Genovese CR, Wasserman L. 2015. Asymptotic theory for density ridges. *Ann. Stat.* 43(5):1896–928

Chen YC, Meilă M. 2019. Selecting the independent coordinates of manifolds with large aspect ratios. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, ed. H Wallach, H Larochelle, A Beygelzimer, F d'Alché-Buc, E Fox, R Garnett, pp. 1086–95. Red Hook, NY: Curran

Chmiela S, Tkatchenko A, Sauceda H, Poltavsky I, Schütt KT, Müller KR. 2017. Machine learning of accurate energy-conserving molecular force fields. *Sci. Adv.* 3:e1603015

Coifman RR, Lafon S. 2006. Diffusion Maps. *Appl. Comput. Harmon. Anal.* 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*. Presented at Neural Information Processing Systems (NIPS) Workshop, Brains and Bits: Neuroscience Meets Machine Learning, Barcelona, Spain, Dec. 9–10

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. 417–22. Piscataway, NJ: IEEE

Cunningham JP, Yu BM. 2014. Dimensionality reduction for large-scale neural recordings. *Nat. Neurosci.* 16:1500–9

Das P, Moll M, Stamati H, Kavraki L, Clementi C. 2006. Low-dimensional, free-energy landscapes of protein-folding reactions by nonlinear dimensionality reduction. *Proc. Natl. Acad. Sci.* 103(26):9885–90

Diaconis P, Goel S, Holmes S. 2008. Horseshoes in multidimensional scaling and local kernel methods. *Ann. Appl. Stat.* 2(3):777–807

do Carmo M. 1992. *Riemannian Geometry*. New York: Springer

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):759–73

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):1327–51

Falconer K. 2003. Alternative definitions of dimension. In Fractal Geometry: Mathematical Foundations and Applications, ed. K Falconer, pp. 39–58. New York: John Wiley & Sons

Farahmand AM, Szepesvári C, Audibert JY. 2007. Manifold-adaptive dimension estimation. In *Proceedings of the 24th International Conference on Machine Learning, ICML '07*, pp. 265–72. New York: ACM

Fefferman C, Mitter S, Narayanan H. 2016. Testing the manifold hypothesis. *J. Am. 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. *Found. Comput. Math.* 20(4):827–87

García Trillos N, Slepčev D. 2018. A variational approach to the consistency of spectral clustering. *Appl. Comput. Harmon. Anal.* 45(2):239–81

Genovese CR, Perone-Pacifico M, Verdinelli I, Wasserman LA. 2012. Minimax manifold estimation. *J. Mach. Learn. Res.* 13:1263–91

Giné E, Koltchinskii V. 2006. Concentration inequalities and asymptotic results for ratio type empirical processes. *Ann. Probab.* 34(3):1143–216

Goldberg Y, Zakai A, Kushnir D, Ritov Y. 2008. Manifold learning: the price of normalization. *J. Mach. Learn. Res.* 9(63):1909–39

Goodfellow I, Bengio Y, Courville A. 2016. *Deep Learning*. Cambridge, MA: MIT Press

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

Hein M, Audibert J, von Luxburg U. 2007. Graph Laplacians and their convergence on random neighborhood graphs. *J. Mach. Learn. Res.* 8:1325–68

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 15 (NIPS 2002)*, ed. S Becker, S Thrun, K Obermayer, pp. 857–64. Cambridge, MA: MIT Press

Im DJ, Verma N, Branson K. 2018. Stochastic neighbor embedding under f-divergences. arXiv:1811.01247 [cs.LG]

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. *Chem. Mater.* (27):735–43

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):e98679

Jolliffe IT. 2002. *Principal Component Analysis*. New York: Springer

Joncas D, Meilă M, McQueen J. 2017. Improved graph Laplacian via geometric self-consistency. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, ed. I Guyon, UV Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, R Garnett, pp. 4457–66. Red Hook, NY: Curran

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. *Electron. J. Stat.* 11(1):891–915

Kleindessner M, von Luxburg U. 2015. Dimensionality estimation without distances. *J. Mach. Learn. Res.* 38:471–79

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*, ed. U Brefeld, E Fromont, A Hotho, A Knobbe, M Maathuis, C Robardet, pp. 124–39. New York: Springer

Koelle SJ, Zhang H, Meilă M, Chen YC. 2022. Manifold coordinates with physical meaning. *J. Mach. Learn. Res.* 23(133):1–57

Kohli D, Cloninger A, Mishne G. 2021. LDLE: low distortion local eigenmaps. *J. Mach. Learn. Res.* 22(282):1–64

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

Lee JM. 2003. *Introduction to Smooth Manifolds*. New York: Springer-Verlag

Levina E, Bickel PJ. 2004. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems 17 (NIPS 2004)*, ed. L Saul, Y Weiss, L Bottou, pp. 777–84. Red Hook, NY: Curran

Lin B, He X, Zhang C, Ji M. 2013. Parallel vector field embedding. *J. Mach. Learn. Res.* 14(90):2945–77

Linderman GC, Steinerberger S. 2019. Clustering with t-SNE, provably. *SIAM J. Math. Data Sci.* 1(2):313–32

Luo C, Safa I, Wang Y. 2009. Approximating gradients for meshes and point clouds via diffusion metric. *Comput. Graph. Forum* 28(5):1497–508

McInnes L, Healy J, Saul N, Grossberger L. 2018. UMAP: uniform manifold approximation and projection. *J. Open Source Softw.* 3(29):861

McQueen J, Meilă M, Joncas D. 2016. Nearly isometric embedding by relaxation. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, ed. D Lee, M Sugiyama, U Luxburg, I Guyon, R Garnett, pp. 2631–39. Red Hook, NY: Curran

McQueen J, Meilă M, VanderPlas J, Zhang Z. 2016a. Megaman: manifold learning with millions of points. arXiv:1603.02763 [cs.LG]

McQueen J, Meilă M, VanderPlas J, Zhang Z. 2016b. Megaman: scalable manifold learning in Python. *J. Mach. Learn. Res.* 17(148):1–5

Meilă M. 2015. Spectral clustering. In *Handbook of Cluster Analysis*, ed. C Hennig, M Meilă, F Murtagh, R Rocci, pp. 125–39. New York: Chapman and Hall/CRC

Meilă M, Shi J. 2001. A random walks view of spectral segmentation. *Proc. Mach. Learn. Res.* R3:203–8

Meilă M, Zhang H. 2023. Manifold learning: what, how, and why. arXiv:2311.03757 [stat.ML]

Mohammed K, Narayanan H. 2017. Manifold learning using kernel density estimation and local principal components analysis. arXiv:1709.03615 [math.ST]

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 (NIPS 2005)*, ed. Y Weiss, B Schölkopf, J Platt, pp. 955–62. 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 14 (NIPS 2001)*, ed. T Dietterich, S Becker, Z Ghahramani, pp. 849–56. Cambridge, MA: MIT Press

Noé F, Clementi C. 2017. Collective variables for the study of long-time kinetics from molecular trajectories: theory and methods. *Curr. Opin. Struct. Biol.* 43:141–47

Ozertem U, Erdogmus D. 2011. Locally defined principal curves and surfaces. *J. Mach. Learn. Res.* 12(34):1249–86

Perrault-Joncas D, Meilă M. 2013. Non-linear dimensionality reduction: Riemannian metric estimation and the problem of geometric discovery. arXiv:1305.7255 [stat.ML]

Perrault-Joncas D, Meilă M, McQueen J. 2017. Improved graph Laplacian via geometric self-consistency. In *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, ed. U von Luxburg, I Guyon, S Bengio, H Wallach, R Fergus, pp. 4457–66. Red Hook, NY: Curran

Pettis KW, Bailey TA, Jain AK, Dubes RC. 1979. An intrinsic dimensionality estimator from near-neighbor information. *IEEE Trans. Pattern Anal. Mach. Intell.* 1(1):25–37

Poličar PG, Stražar M, Zupan B. 2019. opentSNE: a modular Python library for t-SNE dimensionality reduction and embedding. bioRxiv 731877. **https://doi.org/10.1101/731877**

Portegies JW. 2016. Embeddings of Riemannian manifolds with heat kernels and eigenfunctions. *Commun. Pure Appl. Math.* 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 22 (NIPS 2009)*, ed. Y Bengio, D Schuurmans, J Lafferty, C Williams, A Culotta, pp. 1527–35. Red Hook, NY: Curran

Rohrdanz MA, Zheng W, Maggioni M, Clementi C. 2011. Determination of reaction coordinates via locally scaled diffusion map. *J. Chem. Phys.* 134(12):124116

Rosenberg S. 1997. *The Laplacian on a Riemannian Manifold: An Introduction to Analysis on Manifolds.* Cambridge, UK: Cambridge Univ. Press

Roweis S, Saul L. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290(5500):2323–26

Sha F, Saul LK. 2005. *Analysis and Extension of Spectral Methods for Nonlinear Dimensionality Reduction (ICML'05)*. New York: ACM

Shi J, Malik J. 2000. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(8):888–905

Singer A. 2006. From graph to manifold Laplacian: the convergence rate. *Appl. Comput. Harmon. Anal.* 21(1):128–34

Singer A, Wu HT. 2012. Vector Diffusion Maps and the connection Laplacian. *Commun. Pure Appl. Math.* 65(8):1067–144

Slepčev D, Thorpe M. 2019. Analysis of p-Laplacian regularization in semisupervised learning. *SIAM J. Math. Anal.* 51(3):2085–120

Sogge CD. 2014. *Hangzhou Lectures on Eigenfunctions of the Laplacian*. Princeton, NJ: Princeton Univ. Press

Tenenbaum JB, de Silva V, Langford JC. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–23

Ting D, Huang L, Jordan MI. 2010. An analysis of the convergence of graph Laplacians. In *ICML'10: Proceedings of the 27th International Conference on Machine Learning*, ed. J Fürnkranz, T Joachims, pp. 1079–86. Madison, WI: Omnipress

Ting D, Jordan MI. 2018. On nonlinear dimensionality reduction, linear smoothing and autoencoding. arXiv:1803.02432 [stat.ML]

Ting D, Jordan MI. 2020. Manifold learning via manifold deflation. arXiv:2007.03315 [stat.ML]

Tribello GA, Ceriotti M, Parrinello M. 2012. Using sketch-map coordinates to analyze and bias molecular dynamics simulations. *Proc. Natl. Acad. Sci.* 109:5196–201

van der Maaten L. 2014. Accelerating t-SNE using tree-based algorithms. *J. Mach. Learn. Res.* 15(93):3221–45

van der Maaten L, Hinton G. 2008. Visualizing data using t-SNE. *J. Mach. Learn. Res.* 9:2579–605

Vanderplas J, Connolly A. 2009. Reducing the dimensionality of data: locally linear embedding of Sloan galaxy spectra. *Astron. J.* 138(5):1365

Verma N. 2011. *Towards an algorithmic realization of Nash's embedding theorem*. Work. Pap., Univ. Calif., San Diego

von Luxburg U. 2007. A tutorial on spectral clustering. *Stat. Comput.* 17(4):395–416

Wasserman L. 2018. Topological data analysis. *Annu. Rev. Stat. Appl.* 5:501–32

Weinberger KQ, Saul LK. 2006. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference*, pp. 1683–86. Washington, DC: AAAI Press

Yu K, Zhang T. 2010. Improved local coordinate coding using local tangents. In *ICML'10: Proceedings of the 27th International Conference on International Conference on Machine Learning*, ed. J Fürnkranz, T Joachims, pp. 1215–22. Madison, WI: Omnipress

Zhang Y, Gilbert AC, Steinerberger S. 2022. May the force be with you. In *58th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1–8. Piscataway, NJ: IEEE

Zhang Y, Steinerberger S. 2022. t-SNE, forceful colorings and mean field limits. *Res. Math. Sci.* 9:42

Zhang Z, Zha H. 2004. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM J. Sci. Comput.* 26(1):313–38