Statistical Machine Learning
Manifold Learning

Horacio Gómez-Acevedo
Department of Biomedical Informatics

February 22, 2024

UAMS

PCA

In a general setup, we can think that we have m vectors in $\mathbb{R}^n$, namely $\{x_1, \ldots, x_m\}$. We try to find the "optimal" linear projection $\theta \colon \mathbb{R}^n \to \mathbb{R}^k$ where $k \ll n$ (meaning k much lower than n)

1. $\tilde{x_j} = x_j - \mu$ where $\mu = \frac{1}{m} \sum x_j$
2. We find the variance-covariance matrix

$$C = \frac{1}{m} \sum_{j=1}^{m} \tilde{x_j} \tilde{x_j}^t$$

3. We compute the top k-eigenvectors $\{v_1, \ldots, v_k\}$ of C.
4. These eigenvectors span a hyperplane (subspace) of $\mathbb{R}^n$. The projection $\theta \colon \mathbb{R}^n \to \mathbb{R}^k$ is precisely the orthogonal projection onto this plane followed by a choice of identification of the plane with $\mathbb{R}^k$.
5. We can think θ as

$$y_j = \theta(\tilde{x_j}) + \mu$$

PCA cont

The process chooses the basis which maximizes the variance captured by the representation; the eigenvector v_1 with the largest eigenvalue is the single direction that captures the maximal amount of information about the variance; the plane spanned by $\{v_1, v_2\}$ is the plane with the most variance, etc.
Another characterization of PCA is to find a projection that minimizes the error function

$$E = \sum_{j=1}^{m} \|x_j - y_j\|^2,$$

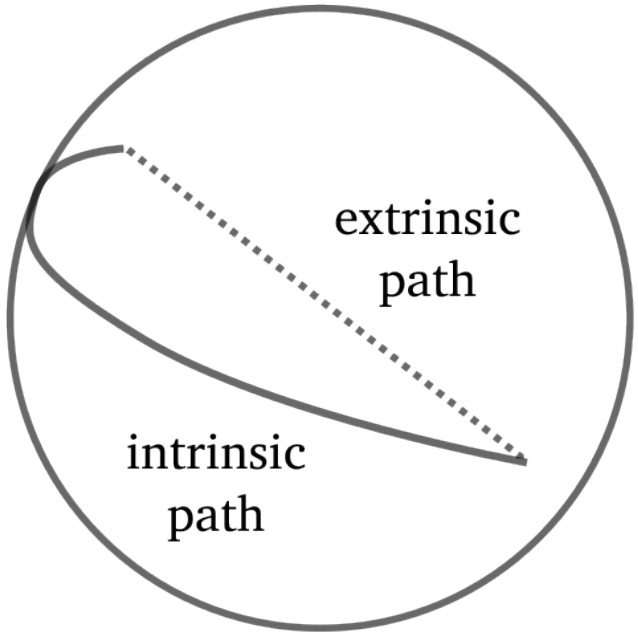
where $\|\cdot\|$ denotes some distance. For PCA this means that it produces the points $\{y_i\}$ that minimize the reconstruction error among all projections onto a k dimensional subspace.

What we learned from linear models

Popular methods of data analysis make the assumption that data lies on a linear k-dimensional subspace of $\mathbb{R}^n$ where $k < n$.
The general problem reduces to search for a linear transformation $\theta\colon \mathbb{R}^n \to \mathbb{R}^k$ such that $\{\theta(x_i)\}$ retains something about the structure of $\{x_i\}$.
Linear models are ubiquitous in sciences, but the assumption of linearity is often unrealistic. One alternative, is to consider that the data has been sampled from a (compact Riemannian) **manifold** $\mathcal{M} \subset \mathbb{R}^n$ of much lower dimension than n.

Manifold Learning (cont))

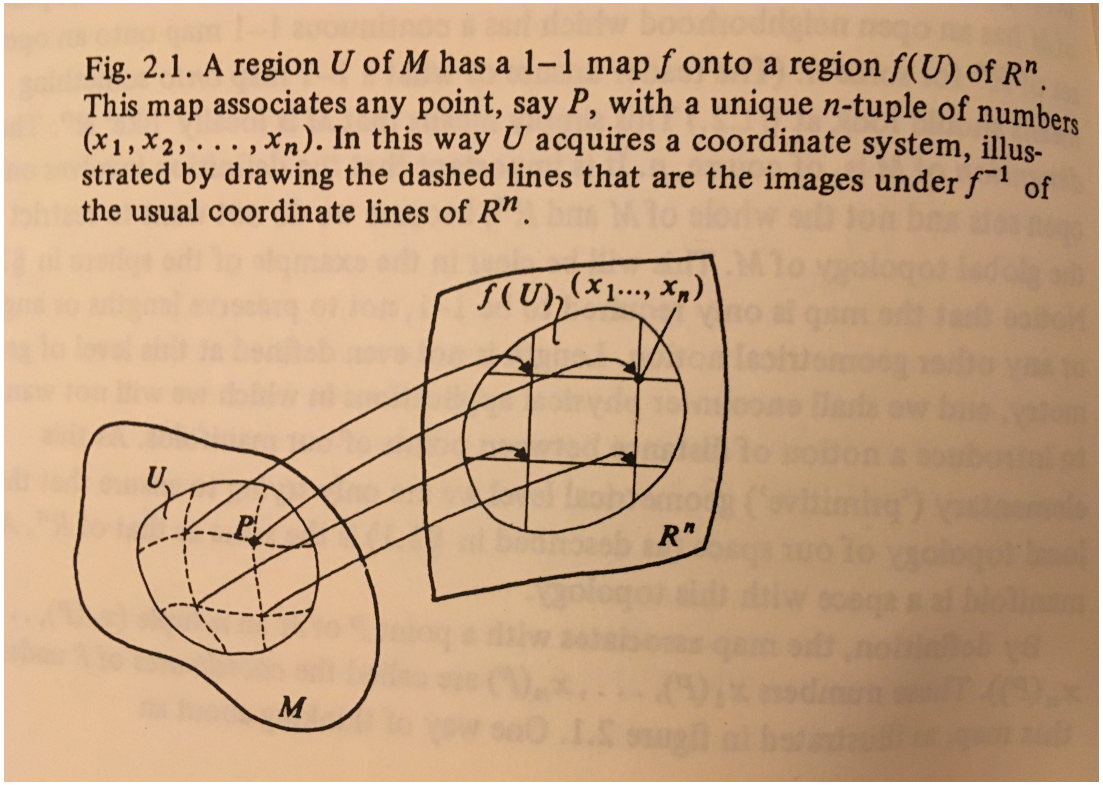
Suppose that we are given data points $\{x_1, \ldots, x_m\}$ in $\mathbb{R}^n$. We will assume that there is a function (embedding) $\gamma : \mathcal{M} \to \mathbb{R}^n$.
Keep in mind that *distances* in manifolds can be different



The manifold structure can be reconstructed by considering the "short distances" as reliable indicators of the local (intrinsic) distances. Extrinsic refers to the paths that "leave" the manifold.

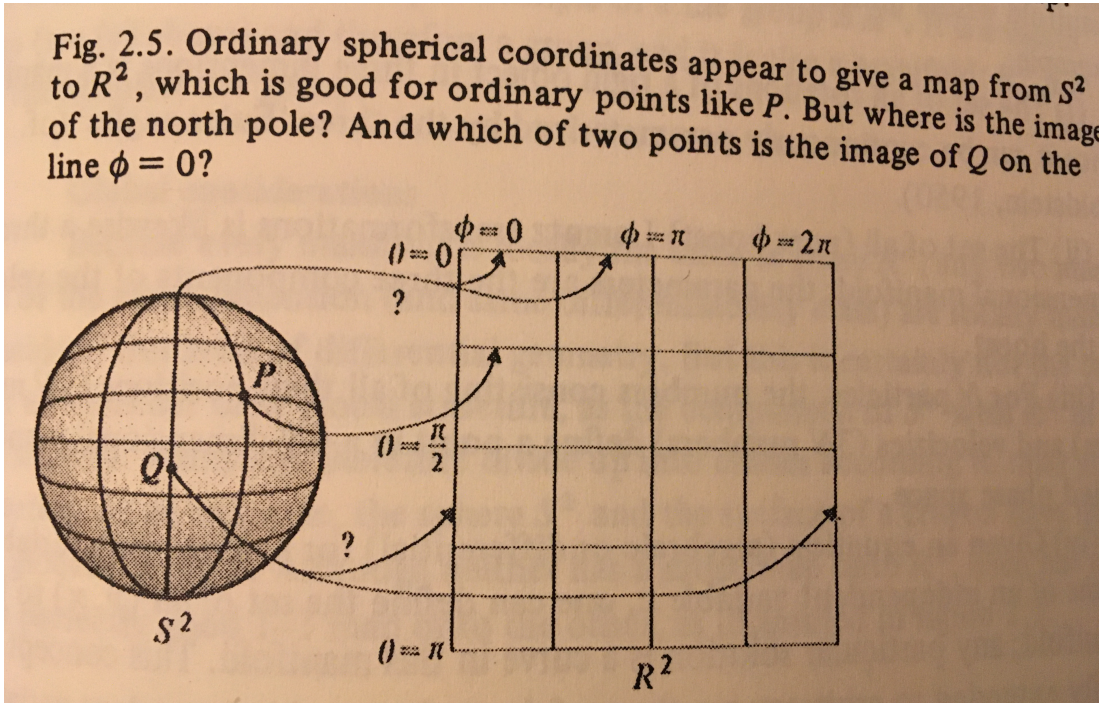
What is a manifold?

Roughly speaking, a manifold (of dimension 2) is a surface that locally behaves as a regular space in $\mathbb{R}^2$. In geometry, we refer to local properties as intrinsic.



The Sphere as a manifold

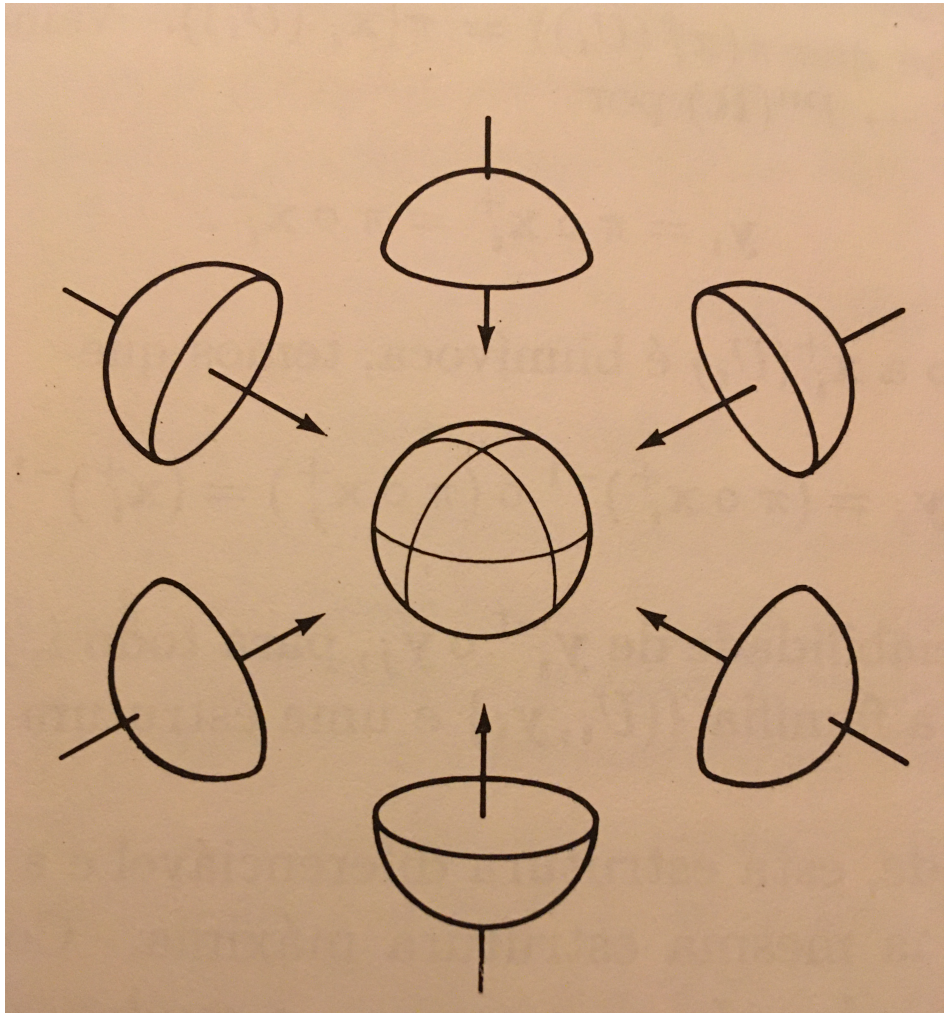
Another typical example is the sphere (often called S^2).



Intuitively, we cannot cover the whole sphere (in a unique way) with only one sheet. The north and south poles are a problem.

Sphere as a patch work

Instead, we use several overlapping sheets



The manifold hypothesis

The **Manifold hypothesis** states the most "naturally occuring" high dimensional dataset lie in a low dimensonal manifold.

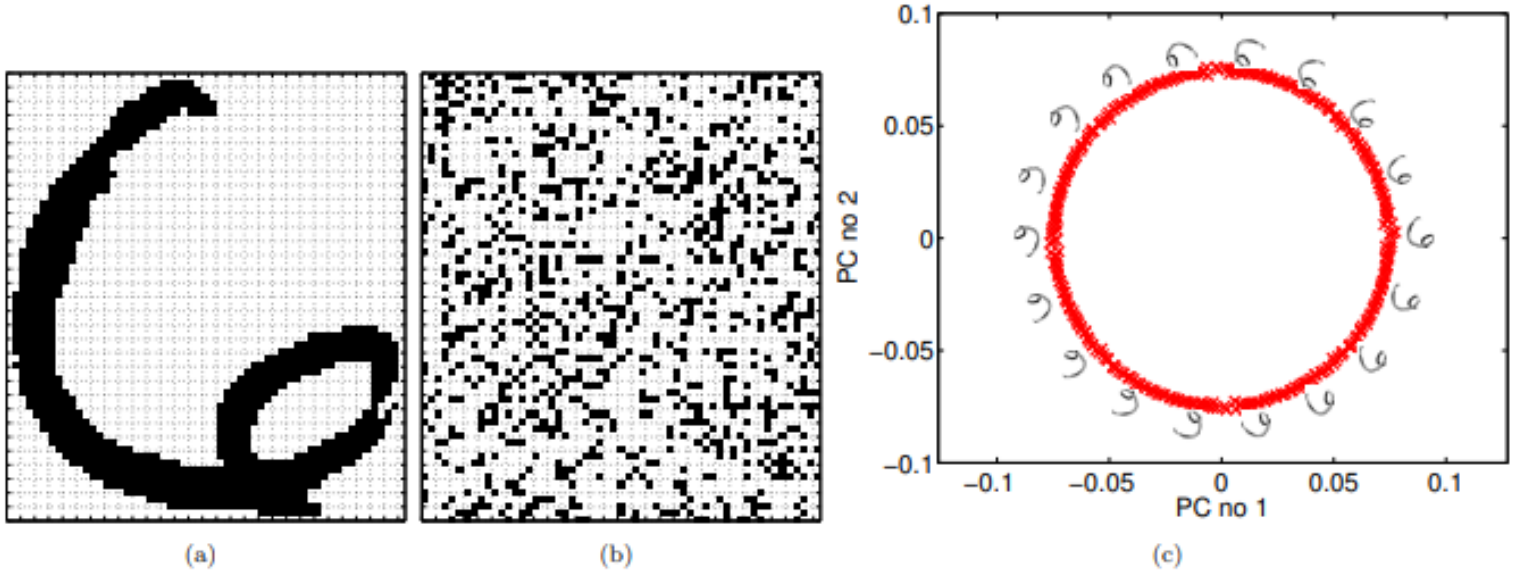


Figure 1: The storage capacity of high dimensional spaces. (a) A six from the USPS digit data set. (b) A sample from a simple independent pixel model of the six. There are $2^{3,648}$ possible images. Even with an enormous number of samples from such a model we would never see the original six. (c) A data set generated by rotating the original six from (a) 360 times. The data is projected onto its first two principal components. These two principal components show us that the data lives on a circle in this high dimensional space. There is a small amount of noise due to interpolation used in the image rotation. Alongside the projected points we show some examples of the rotated sixes.

Manifold Learning

Loosely speaking, the basic idea of most manifold learning techniques is to take the k-nearest neighbors of a point x and use the vectors specified by the line segments from x to its neighbors as an approximation for the tangent plane at x.
Global optimization then sews these local approximations together to produce a low-dimensional representation of the data.

Manifold learning is also known as dimensionality reduction.

Multidimensional scaling (MDS)

From Lapedes and Farber (2001)

> *"Multidimensional scaling" algorithms are a class of algorithms initially developed in the computational psychology literature (Shepherd, 1963, 1964) which reconstruct the true dimension of the space, and the relative coordinates of points, given only distances, or more generally monotonic transformations of distances, between the points. "Relative" means that coordinates are reconstructed from the distance data up to global translation, reflection, scale and rotation which leave the relative relation of points invariant. Since interest centers on relative relationships, such global transformations are irrelevant.*

MDS (cont)

The MDS tries to find a set of low dimensional vectors $\{\mathbf{z}_i \in \mathbb{R}^k\}_{i=1}^N$ such that the pairwise distances between these vectors is as similar as possible to a set of pairwise dissimilaritites $\mathbf{D} = \{d_{ij}\}$ provided by the user.
Let's considered the centered similarity matrix as follows

$$\tilde{K}_{ij} = \langle \mathbf{x}_i - \bar{\mathbf{x}}, \mathbf{x}_j - \bar{\mathbf{x}} \rangle$$

Digression

Let's consider a matrix $\mathbf{X}$ of type $N \times D$. The **sum of squares matrix** is a $D \times D$ matrix defined by

$$\mathbf{X}^t\mathbf{X} = \sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^t = \sum_{n=1}^{N} \begin{pmatrix} x_{n,1}^2 & \cdots & x_{n,1}x_{n,D} \\ \vdots & \ddots & \vdots \\ x_{n,D}x_{n,1} & \cdots & x_{n,D}^2 \end{pmatrix}$$

The **scatter matrix** is defined by

$$\mathbf{S}_{\bar{\mathbf{x}}} = \sum_{n=1}^{N} (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^t$$

The last expression means that the sum of squares matrix is applied to mean-centered data.

Digression (cont)

The **centering matrix** is defined by

$$\mathbf{C}_N = \mathbf{I}_N - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^t$$

For instance

$$\mathbf{C}_3 = \begin{pmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \end{pmatrix}$$

It can be shown that

$$\mathbf{S}_{\overline{\mathbf{x}}} = \mathbf{X}^t\mathbf{C}_N\mathbf{X}$$

MDS (cont)

Using matrix notation, we have

$$\tilde{\mathbf{X}} = \mathbf{C}_N\mathbf{X}(\mathbf{C}_N\mathbf{X})^t$$

Moreover, if we define $\tilde{\mathbf{z}}_i = \mathbf{z}_i - \bar{\mathbf{z}}$, and $\tilde{Z}_{ij}$ denotes elements of the centered matrix $\tilde{\mathbf{Z}}$.
The goal will be to minimize

$$\mathcal{E} = \sum_{i,j}(\tilde{K}_{ij} - \tilde{Z}_{ij})^2 = \|\tilde{\mathbf{K}} - \tilde{\mathbf{Z}}\tilde{\mathbf{Z}}^t\|^2$$

This process is done using a method called **singular value decomposition** of $\tilde{\mathbf{K}}$. Thus, $\tilde{\mathbf{K}} = \mathbf{USU}^t$, and hence $\tilde{\mathbf{Z}} = \mathbf{US}^{\frac{1}{2}}$

MDS (cont)

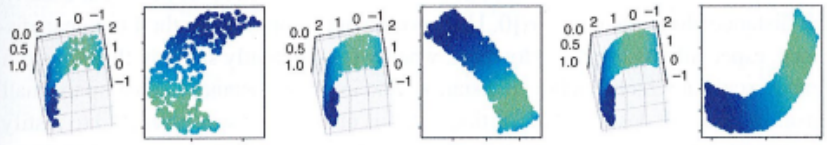


Figure 4.3 When the data lies on a single curved ribbon, the embedding into $\mathbb{R}^2$ exhibits distortion arising from the curvature.

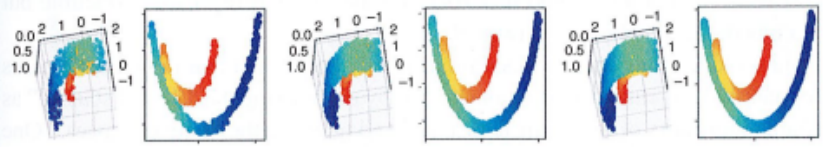


Figure 4.4 When the data lies on nested ribbons, the embedding is further distorted by the proximity of the two components.

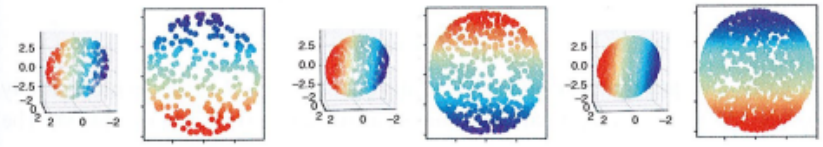


Figure 4.5 When the data lies on the standard sphere S^2, the embedding flattens the sphere and distorts the distances along an arbitrary axis.

Theorem

Given $\{x_1, \ldots, x_l\} \subset \mathbb{R}^n$ and $k < n$, the results of metric MDS and PCA embedding $\{x_i\}$ into $\mathbb{R}^k$ are isometric.

Isomap

Isomap applies MDS to an empirical approximation of the intrinsic metric. The procedure goes as follows. We fix a scale parameter ε and a target dimension parameter k.

1. Form the weighted graph G with
   - vertices the points $\{x_i\}$, and
   - edges (i, j) with weights given by $w_{ij} = \|x_i - x_j\|$ when $\|x_i - x_j\| \leq \varepsilon$.
2. We form a new space X' with points $\{x_i\}$ but distances given by the graph metric on G. Meaning that the distances between two edges is given by the shortest path in the graph.
3. Use MDS to embed this space into $\mathbb{R}^k$ producing points $y_i = \theta(x_i)$

Isomap

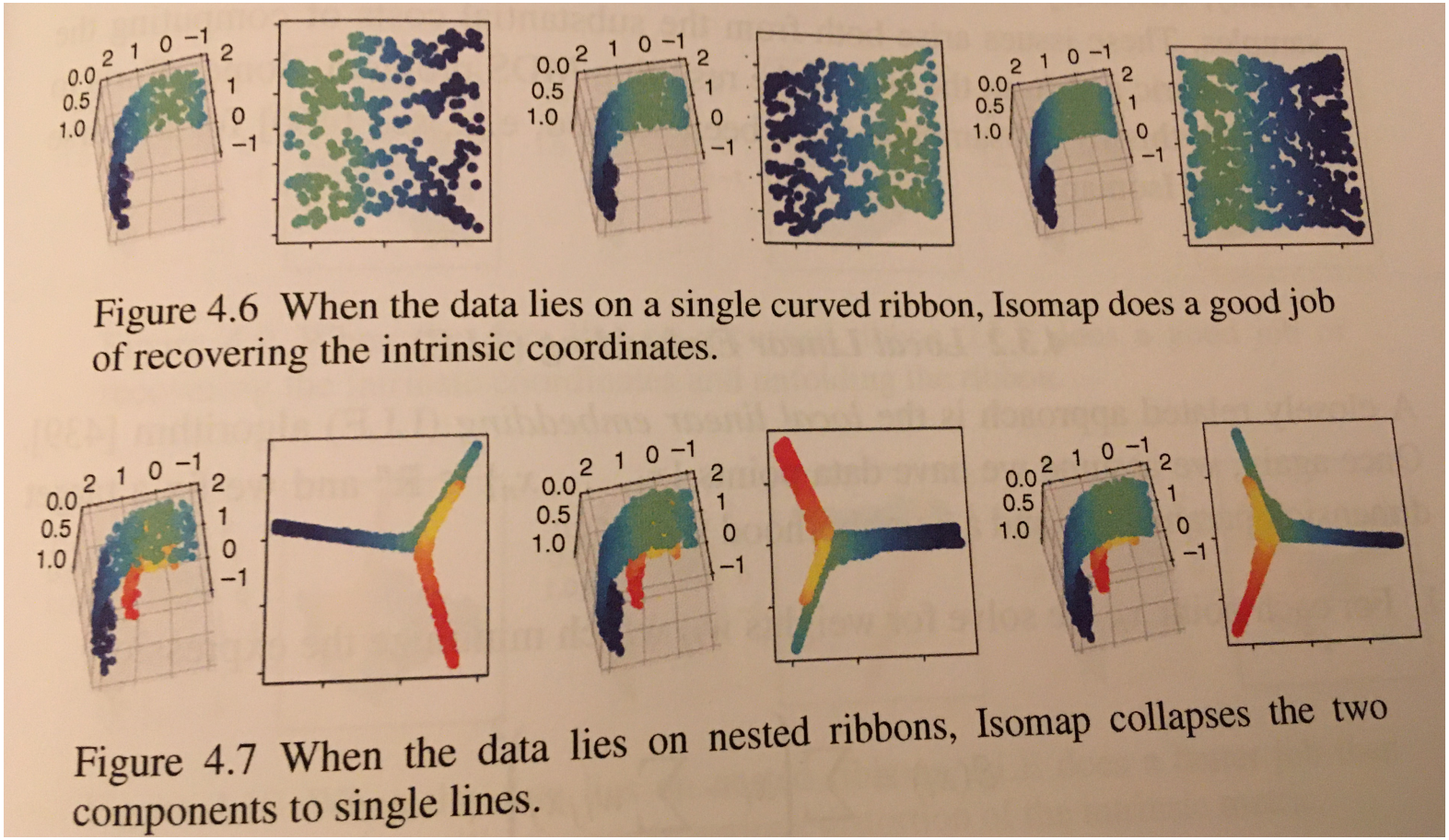


Figure 4.6 When the data lies on a single curved ribbon, Isomap does a good job of recovering the intrinsic coordinates.

Figure 4.7 When the data lies on nested ribbons, Isomap collapses the two components to single lines.

Isomap cont.

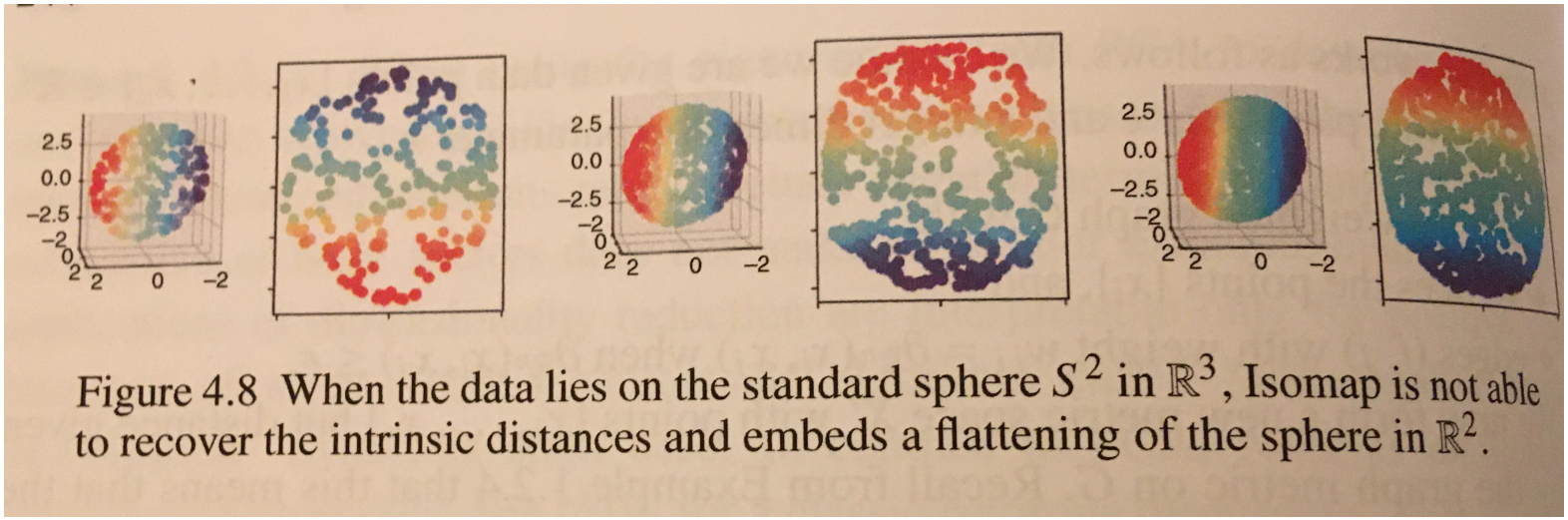


Figure 4.8 When the data lies on the standard sphere S^2 in $\mathbb{R}^3$, Isomap is not able to recover the intrinsic distances and embeds a flattening of the sphere in $\mathbb{R}^2$.

Local Linear Embedding LLE

We fix a target dimension parameter k and a neighborhood size K. The algorithm goes as follows:

1. For each point x_i, we solve for weight w_{ij} which minimize the expression

$$\mathcal{E}(x_i) = \sum_i \left(x_i - \sum_j w_{ij} x_j \right)^2,$$

subject to the constrains

$$\begin{cases} w_{ij} = 0 & x_j \text{ not a } K \text{ nearest neighbor of } x_i. \\ \sum_j w_{ij} = 1 \end{cases}$$

We are solving for weights that optimally reconstruct each point x_i from its K nearest neighbors. The weights can efficiently computed via least squares.

2. Embedding points $\{y_i = \theta(x_i)\} \subset \mathbb{R}^k$ are computed so that

LLE cont.

$$\mathcal{E} = \sum_i \left(y_i - \sum_j w_{ij} y_j \right)^2$$

is minimized. To solve this problem we can calculate the top k eigenvectors of the corresponding matrix of the quadratic form.

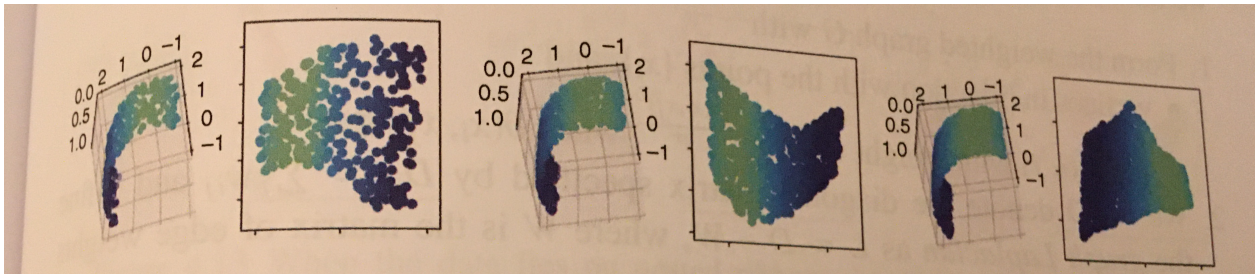


Figure 4.9 When the data lies on a curved ribbon, LLE does a good job of recovering the intrinsic coordinates and unfolding the ribbon.

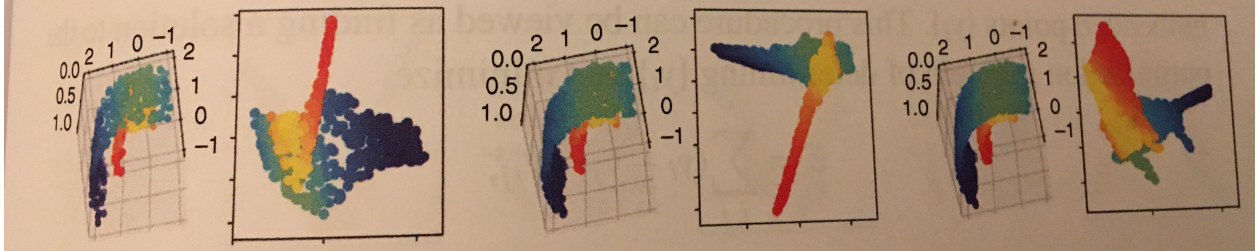


Figure 4.10 When the data lies on nested ribbons, LLE does a better job than

LEE cont

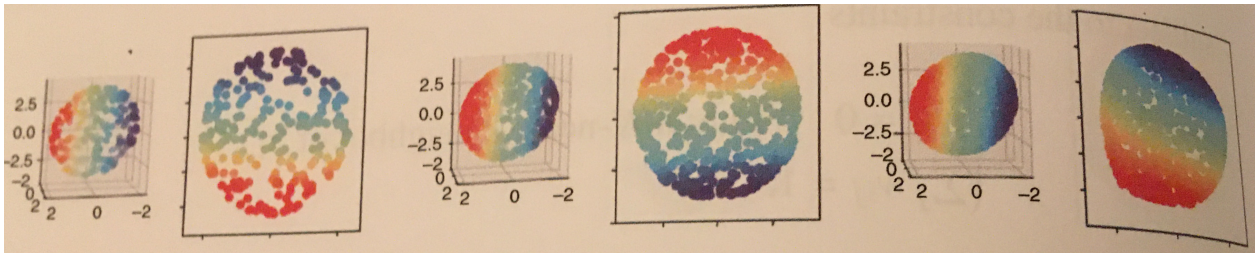


Figure 4.11 When the data lies on the standard S^2 in $\mathbb{R}^3$, LLE does not do a good job of capturing the intrinsic distances and simply embeds a flattening of the sphere.

Neighbor Embedding Algorithms

These algorithms aim to address the problem when data points have non-uniform density. The **stochastic neighbor embedding (SNE)** and the *t*-**distributed stochastic neighbor embedding (t-SNE)** aim to address that issue.
We begin with our traditional setup for data where $\{x_i\} \subset \mathbb{R}^n$ and $\{y_i\} \subset \mathbb{R}^k$, where $k \leq n$ (hopefully considerably smaller). Then we define

$$p_{j|i} = \frac{\exp(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2})}{\sum_{k \neq i} \exp(\frac{-\|x_i - x_j\|^2}{2\sigma_i^2})}$$

and

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_j\|^2)}$$

where the variances σ_i are obtained by an optimization process, and in the second equation we are fixing all of the variances to be identically $\sqrt{2}/2$.

SNE cont.

The idea behind SNE is that good image points $\{y_i\}$ have the property that the difference between $p_{j|i}$ and $q_{j|i}$ is minimized in the sense of the so-called summed Kullback-Leibler divergences cost function

$$C = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

This equation ensures that the local distributions have similar means.

SNE cont

In practice, SNE algorithm proceeds by solving the minimizing point $\{y_i\}$ via a gradient descent in order to find a good local minimum for C. Convergence is often slow and depends critically on good choices of the variances σ_i. In principle, differing choices of $\sigma_i$amount to enforcing variable numbers of neighbors used to do the local estimation of the coordinates; this is expressed here via the use of the **perplexity** which is computed as

$$P = 2^{-\sum_j p_{j|i} \log_2 p_{j|i}}$$

The desired perplexity is typically between 10 and 100 is a parameter and we solve for values σ_i that achieve the perplexity.

t-SNE

One problem with the original SNE is that the gradient descent optimization is slow and difficult to achieve convergence. The proposed solution goes as follows

1. We symmetrize $p_{j|i}$ as
$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2}$$
2. We define a symmetrize variant of $q_{j|i}$ as follows
$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}}$$
In the original definition, the $q_{j|i}$ was defined using a Gaussian; this expression replaces that with the Student t-distribution with one degree of freedom, which has more weight in the tails.
3. Finally, the cost function is replaces with the expression
$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

t-SNE advantages over SNE

Why do we complicate things this way?

1. The use of a heavier tailed distribution means that outliers have less impact on the overall results and the compression effects around the center of mass are alleviated (to some degree)
2. The adjusted formula for q_{ij} also has the effect of substantially improving the efficiency and quality of the gradient descend procedure.

t-SNE cont

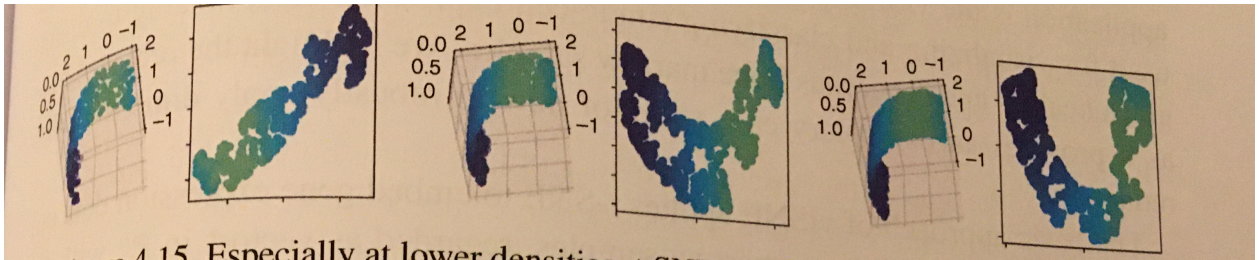


Figure 4.15 Especially at lower densities, t-SNE unfolds the ribbon to recover its intrinsic coordinates.

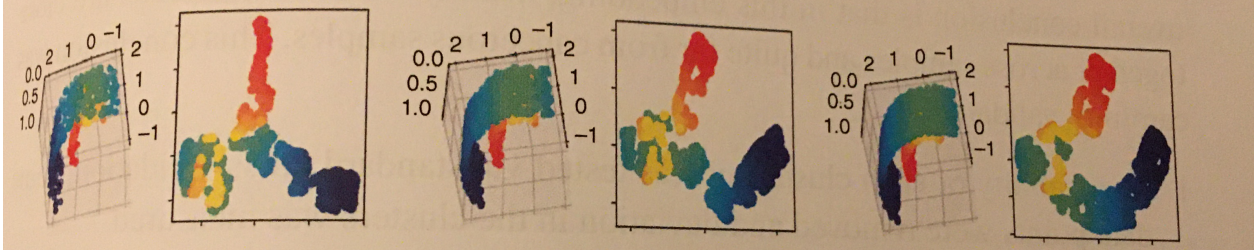


Figure 4.16 When the data lies on the nested arcs, t-SNE actually does a reasonable job at recovering the intrinsic coordinates.

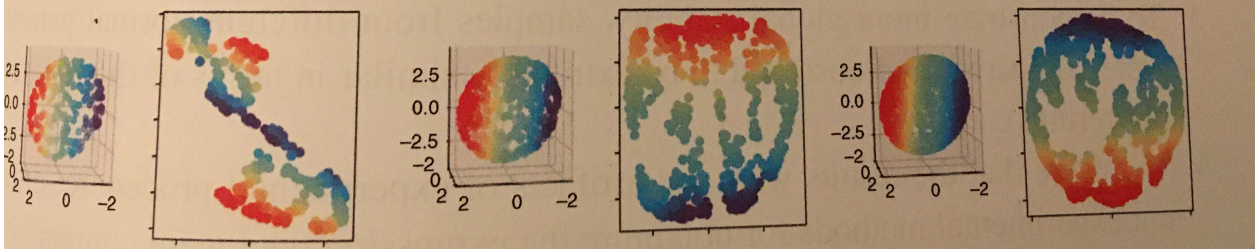


Figure 4.17 When the data lies on the standard S^2 in $\mathbb{R}^3$, t-SNE does not do a good job of capturing the intrinsic distances and instead flattens the sphere.

What we have learned?

- Manifold learning and dimensionality reduction are similar concepts.
- The idea is to reduce dimensionality taking advantage of the manifold structure of our data.
- Most of these methods imply that we can understand the local behavior of our data and classical methodologies are applied based on this assumption.

References

Some of the pictures are from (Do Carmo, 1988), and (Schutz, 1980), and (Géron, 2019). Main reference for manifold learning (Rabadán and Blumberg, 2020), and (Murphy, 2022). The cited paper is (Lawrence, 2012)
I have used some of the graphs by hacking TiKz code from StakExchange, Inkscape for more aesthetic plots and other old tricks of TEX

References (cont)

- Do Carmo, M. P. (1988). *Geometria Riemanniana*. Projeto Euclides. IMPA. ISBN: 85-244-0036-6.
- Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow. Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd. Edition. O'Reilly. ISBN: 978-1-492-03264-9.
- Lawrence, Neil D. (2012). "A Unifying Probabilistic Perspective for Spectral Dimensionality Reduction: Insights and New Models". In: *Journal of Machine Learning Research* 13.51, pp. 1609–1638.
- Murphy, K. P. (2022). *Probabilistic Machine Learning*. The MIT Press. ISBN: 978-262-04682-4.
- Rabadán, R. and A. J. Blumberg (2020). *Topological Data Analysis for Genomics and Evolution*. Cambridge University Press. ISBN: 978-1107-159549.
- Schutz, B. (1980). *Geometrical methods in mathematical physics*. Cambridge University Press. ISBN: 978-0521-298872.