# 2.4 数据再表达中的流形学习方法

传统前端学习的数据再表达方法中，数据点和数据点之间的距离和映射函数 $f$ 都是定义在欧氏空间中的，然而在实际情况中，这些数据点可能不是分布在欧氏空间中的，因此传统欧氏空间的度量难以用于真实世界的非线性数据，从而需要对数据的分布引入新的假设。

流形(manifold)是一般几何对象的总称，包括**各种维度的曲线与曲面等**，即**高维样本空间中呈现的一种低维的局部性的结构**，它是指局部具有欧氏空间性质的空间。更准确的说，每个 $n$ 维流形上的点都有一个领域同胚于欧氏空间。流形上的点本身是没有坐标的，所以为了表示这些数据点，我们把流形放入到外围空间(ambient space)，用外围空间上的坐标来表示流形上的点，例如三维空间 $R^3$ 中球面是一个 2 维曲面，即球面上只有两个自由度，但我们一般采用外围空间 $R^3$ 空间中的坐标来表示这个球面。

流形学习方法，自 2000 年在《Science》被首次提出以来，已成为信息科学领域的研究热点。在理论和应用上，流形学习方法都具有重要的研究意义。假设数据是均匀采样于一个高维欧氏空间中的低维流形，**流形学习就是从高维采样数据中恢复低维流形结构，即找到高维空间中的低维流形，并求出相应的嵌入映射，以实现维数约简或者数据可视化。**它是从观测到的现象中去寻找事物的本质，找到产生数据的内在规律。流形算法的主要思想是能够学习高维空间中样本的局部邻域结构，并寻找一种子空间能够保留这种流行结构，使得样本在投影到低维空间后，得到比较好的局部近邻关系。

流形学习可以看成是非线性的 PCA。PCA 得到的是一个线性子空间而舍弃了正交于子空间外的信息，而流形学习得到的是一个非线性子空间而舍弃了流形外的信息。流形学习的目的是维数缩减，从而寻找到数据更本质的内在规律，即数据分布的低维流形。

和一般的降维学习一样，流形学习是把一组在高维空间中的数据在低维空间中再表达。不同之处是，在流形学习中假设：**所处理的数据采样在一个潜在的流形上，或者说对于这组数据存在一个潜在的流形。**

**定义：**给定原空间( $D$ 维)中的 $n$ 个样本点 $x_1, \ldots, x_n \in R^D$，它们分布在一个 $d$ ( $d < D$ )维分布的流形上。流形学习寻找一个映射 $f: M \to R^d$，使得 $y_1, \ldots, y_n \in R^d$，$y_i = f(x_i)$。

前端学习中的流形学习方法具有一些共同的特征：

- 首先构造流形上样本点的局部邻域结构；
- 然后用这些局部邻域结构来将样本点全局的映射到一个低维空间。

方法之间的不同之处主要是在于**构造的局部邻域结构不同以及利用这些局部邻域结构来构造全局的低维嵌入方法的不同**。

对一些主要的方法下面分别进行介绍。

**1. 等距特征映射(Isometric Feature Mapping：Isomap [37])**

所谓等距映射就是多维尺度变换 (MDS：Multidimensional Scaling) 与测地线距离相结合。多维尺度变换是理论上保持欧氏距离的一种经典方法。它最早用来做数据的可视化。MDS 得到的低维表示中心在原地，所以又可以说是保持内积大小，MDS 降维后的任意两点的距离（内积）与高维空间的距离相等或近似相等。因此 MDS 在流形数据处理上，保持欧氏距离不变的理论不可行（失效），等距映射就是改进的 MDS 方法，在流形非线性降维领域的一种应用方法。

Isomap 的理论框架为 MDS，放在流形的理论框架中，原始的高维空间欧氏距离换成了流形上的测地线距离。Isomap 是把任意两点的测地距离(准确地说是最短距离)作为流形的几何描述，用 MDS 理论框架保持点与点之间的最短距离。

如何计算测地线距离是首先要解决的问题。在流形结构未知，数据采样有限的情况下，我们可以通过构造数据点间的邻接图，用图上的最短距离来近似测地线距离，当数据点趋于无穷多时，这个估计近似距离趋向于真实的测地线距离。这种算法是计算机系中常用的图论经典算法。

Isomap 算法是近年来用于非线性降维的一个重要前端学习的数据再表达算法。算法的关键在于利用样本向量之间的欧氏距离 $dx(i,j)$ 计算出样本之间的测地距离 $dG(i,j)$，从而真实再现高维数据内在的非线性几何结构。然后使用经典 MDS 算法构造一个新的 $d$ 维欧氏空间 $Y$ ($d$ 是降维空间的维数)，最大限度地保持样本之间的欧氏距离 $dY(i,j)$ 与 $dG(i,j)$ 误差最小，从而达到降维的目的。两点间的欧氏距离不能表征两点的实际距离。分布于流形面上的曲线是两点的测地线距离。流形未知时可以通过最短路径算法对邻域内的距离进行近似地重构两点间的测地线距离。

Isomap 算法有两个的前提假设：

- 高维数据所在的低维流形与欧氏空间的一个子集是整体等距的；
- 与数据所在的流形等距的欧氏空间的子集是一个凸集。

Isomap 算法的核心是估计两点间的测地距离：

- 离得很近的点间的测地距离用欧氏距离代替；
- 离得较远的点间的测地距离用最短路径来逼近。

整个算法可以分成两步：

（1）估计原空间中的测地距离；

（2）使用 MDS 找低维空间中的点，使它们的距离与（1）中的距离相匹配。

之所以叫“等距”，是因为 Isomap 假设在低维空间中的点在等距坐标图下，即低维空间中的距离度量是普通的欧氏距离。此外，Isomap 还做了局部线性假设，即在局部使用欧氏距离近似测地距离。使用 $k$ 近邻来定义局部。而对 $k$ 近邻之外的点间的距离，通过计算 $k$ 近邻定义的图上的最短路的距离作为测地距离的估计。

MDS 是在给定相似度矩阵 $D \in R^{n\times n}$，找 $n$ 个点的点集，使它们的欧氏距离的矩阵与 $D$ 相近。

算法：cMDS（Classical MDS）算法

**classical Multidimensional Scaling**

input: $D \in \mathbb{R}^{n\times n}(D_{ii}=0,\ \ D_{ij}\geq 0),\ \ d\in\{1,\ldots,n\}$

1. Set $B := -\frac{1}{2}HDH$, where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$ is the centering matrix.
2. Compute the spectral decomposition of $B$: $B = U\Lambda U^\top$.
3. Form $\Lambda_+$ by setting $[\Lambda_+]_{ij} := \max\{\Lambda_{ij}, 0\}$.
4. Set $X := U\Lambda_+^{1/2}$.
5. Return $[X]_{n\times d}$.

cMDS 算法基于下面的定理，它建立了欧氏距离矩阵和内积矩阵的关系。

**Theorem** *A nonnegative symmetric matrix $D \in \mathbb{R}^{n\times n}$, with zeros on the diagonal, is a Euclidean distance matrix if and only if $B \stackrel{\text{def}}{=} -\frac{1}{2}HDH$, where $H \stackrel{\text{def}}{=} I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, is positive semidefinite. Furthermore, this $B$ will be the Gram matrix for a mean-centered configuration with interpoint distances given by $D$.*

该定理将欧氏距离矩阵转化为内积矩阵，再由$B = U\Lambda U^T = XX^T$得到新的表示$X = U\Lambda^{1/2}$。乘 H 是为了将数据的均值移到中心。然而实际上 D 往往不是欧氏距离矩阵。这是由于我们仅仅是在估计测地距离而不是得到真正的测地距离。所以实际上 B 的特征值而未必都是非负的。实际操作中将特征值为负的特征值取为 0。为了降维，X 通常只取前 d 维作为 d 维空间下的新的表示。类似 PCA，剩余的 n-d 维信息被丢弃。具体算法如图所示。

还有一个悬而未决的问题是，d 的选取，即流形的维数应当如何选取。cMDS 的做法是对不同的 d 计算 X，再在新的空间中计算$D^*$，选择 d 使$\|HDH - HD^*H\|_F$最小。而这个目标函数对于噪声十分敏感。

Isomap 算法的主要步骤如下：

**Isomap**

input: $x_1, \ldots, x_n \in \mathbb{R}^D$, $k$

1. Form the $k$-nearest neighbor graph with edge weights $W_{ij} := \|x_i - x_j\|$ for neighboring points $x_i, x_j$.
2. Compute the shortest path distances between all pairs of points using Dijkstra's or Floyd's algorithm. Store the squares of these distances in $D$.
3. Return $Y$ :=cMDS($D$).

第 1 步：构造近邻图。首先计算任意两个样本向量 $x_i$ 与 $x_j$ 的欧氏距离 $dX(i,j)$，然后用全部的样本向量 $x_i(1 \le i \le N)$ 构造无向图 G。对于样本向量 $x_i$，在图 G 中将它与离它最近的 $n$ 个样本向量( $n$ 是可调参数)连接起来，设置连接线的长度分别为它们各自的距离。

第 2 步：计算任意两个样本向量之间的最短路径。在图 G 中，设置任意两个样本向量 $x_i$ 与 $x_j$ 之间的最短距离为 $dG(i,j)$ 。如果 $x_i$ 与 $x_j$ 之间存在连线，$dG(i,j)$ 的初始值设为 $dX(i,j)$，否则令 $dX(i,j)=\infty$。接下来依次更新 $dG(i,j)$ 的数值：

$dG(i,j)=\min\{dG(i,j)$ dG(i，j)=min≤I≤N{dG(i，J)，dG(1，j)} 。

第 3 步：经过多次迭代，样本向量间最短路径矩阵 $DG=\{dG(i,j)\}$ 便可收敛。使用经典 MDS 将样本向量压缩到 $d$ 维，并使压缩之后样本向量之间的欧氏距离尽可能接近已求出的最短路径。

事实上，Isomap 能自适应的选取流形的维数 $d$，在特征值分解步骤选取特征值为正的维数。当然，噪声和不确定性会使小的正特征值出现，所以直接选取所有正特征值往往不好。

在以下条件下，Isomap 的最优性是有保证的：

（1）流形在$R^D$中等距嵌入；

（2）任意两点在$R^d$中的距离由完全在$R^d$中的测地线给出；

（3）流形的采样均匀充分；

（4）流形是紧的。

Silva 等 [38]提出了两种 Isomap 的变种，分别是：映射具有保角性(conformal)的 Clustering-Isomap(C-Isomap)和计算稀疏性的 Landmark Isomap(L-Isomap)。

C-Isomap 的第一步和第三步与 Isomap 相同，在第二步做了调整：边的权重为$\frac{|x_i - x_j|}{\sqrt{M(i)M(j)}}$，其中$M(i)$为$i$到它的$k$个邻点的平均距离。

L-Isomap 使用地标点(landmark point)来减少 cMDS 中对点对间距离的计算。算法思想在于选择$l$个地标点，$l > d, l << n$。在计算点对间距离时，只计算$n$个点到$l$个地标点之间的距离，而谱分解只对地标点距离矩阵进行。在测试阶段求取地维坐标时，以$g_{new,i} = \min\{d_{new,j} + g_{j,i} : j\}$作为$x_{new}$到第$i$个地标点的距离。再用这个距离向量使用地标点学出的矩阵向低维空间投影。

Isomap 算法有如下优点：

(1)能处理非线性流形之类的高维数据；

(2)全局优化；

(3)不管输入空间是高度折叠的，还是扭曲的，或者弯曲的，Isomap 仍然能全局优化低维的欧氏表示；

(4)Isomap 能保证渐进地恢复到真实的维度。

但是，Isomap 也存在如下缺点：

(1)可能在数据拓扑空间是不稳定；

(2)保证渐进恢复到非线性流形的几何结构的时候，当 N 增加的时候，点对距离提供更加接近于测地的距离，但是花更多计算时间；假如 N 是小的，测地距离将会非常不精确。

**2. 局部线性嵌入（Local Linear Embedding：LLE[39]）**

（1）LLE 算法的基本思想

在局部保持结构特征，或者说数据特征的方法有很多种，不同的保持方法对应不同的流形算法。等距映射算法在降维后的数据再表达中希望保持样本之间的测地距离而不是欧氏距离，因为测地距离更能反映样本之间在流形中的真实距离。

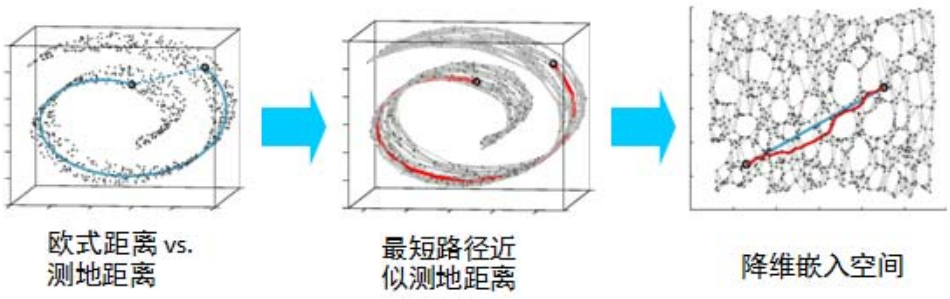


图 2.28 各种距离与降维嵌入空间比较

但是等距映射算法有一个问题就是他要找所有样本全局的最优解，当数据量很大，样本维度很高时，计算量大，非常耗时。鉴于这个问题，LLE 通过放弃所有样本全局最优的降维，只是通过保证局部最优来降维。同时假设样本集在局部是满足线性关系的，进一步减少的降维数据再表达的计算量。

一个流形在很小的局部邻域上可以近似看成是欧氏的，即局部线性的。那么，在小的局部邻域中，一个点就可以用它周围的点在最小二乘意义下的最优线

性来表示。LLE 方法即是把这种线性拟合的系数当成这个流形的局部几何性质的描述。同样的，一个后的低维空间数据表示，也就应该具有同样的局部几何性质，所以利用同样的线性表示的表达式，最终写成一个二次型的形式 。

LLE 的想法与 Isomap 不同，它把流形一块块重叠的有坐标的小块。当邻域足够小、流形足够光滑时，这样的小块接近线性。LLE 寻找保持局部线性几何的映射，并假设这样的小块彼此重叠从而能够重建出整个流形。

LLE 首先假设数据在较小的局部是线性的，也就是说，在一个数据可以由它的领域中的几个样本来线性表示。如，有一个样本$x_1$，我们在它的原始高维邻域里用$K$-近邻思想找到和它最近的三个样本$x_2,x_3,x_4$，然后我们假设$x_1$可以由$x_2,x_3,x_4$线性表示，即

$$x_1=w_{12}x_2+w_{13}x_3+w_{14}x_4$$

其中$w_{12},w_{13},w_{14}$为权重系数。通过 LLE 降维后，我们希望$x_1$再低维空间的投影$x_1'$和$x_2,x_3,x_4$对应的投影$x_2',x_3',x_4'$也尽量保持相同的线性关系，即

$$x_1'\approx w_{12}x_2'+w_{13}x_3'+w_{14}x_4'$$

也就是说，投影前后线性关系的权重系数$w_{12},w_{13},w_{14}$尽量不变或者改变最小。

从上面可以看车，线性关系只在样本附近起作用，离样本远的样本对局部线性关系没有影响，因此降维的复杂度降低了很多。概括的来说就是 “流形在局部可以近似等价于欧氏空间”。

（2）LLE 算法的推导

对于 LLE 算法，我们首先要确定邻域大小的选择，即我们需要多少个领域样本来线性表示一个给定的样本，假定这个值为$k$。我们可以通过和 KNN 一样的思想通过距离度量（如欧氏距离）来选择给定样本的$k$个最近邻。

假设有$m$个$n$维样本$\{x_1,x_2,\cdots,x_m\}$，其中第$i$个样本$x_i=\begin{pmatrix}x_{i1}\\x_{i2}\\\vdots\\x_{in}\end{pmatrix}\in R^n$，记

$$X=\begin{pmatrix}x_1^T\\x_2^T\\\vdots\\x_m^T\end{pmatrix}=\begin{pmatrix}x_{11}&x_{12}&\cdots&x_{1n}\\x_{21}&x_{22}&\cdots&x_{2n}\\\vdots&\vdots&\vdots&\vdots\\x_{m1}&x_{m2}&\cdots&x_{mn}\end{pmatrix}\in R^{m\times n}$$

对于第 $i$ 个样本 $x_i$，假设其 $k$ 个最近邻样本集合为 $N_i=\{x_{1_i},x_{2_i},\cdots,x_{k_i}\}$。需要找到 $x_i$ 与这 $k$ 个最近邻样本集合 $\{x_{1_i},x_{2_i},\cdots,x_{k_i}\}$ 之间的线性关系，即求出对应的权重系数矩阵：

$$W=(w_1,w_2,\cdots,w_m)=\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ w_{k1} & w_{k2} & \cdots & w_{km} \end{pmatrix}\in R^{k\times m}$$

第 $i$ 个样本 $x_i$ 用 $N_i=\{x_{1_i},x_{2_i},\cdots,x_{k_i}\}$ 中的样本近似线性表示为：

$$x_i=\begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{im} \end{pmatrix}\approx w_{1i}x_{1_i}+w_{2i}x_{2_i}+\cdots+w_{ki}x_{k_i}=w_{1i}\begin{pmatrix} x_{1_i1} \\ x_{1_i2} \\ \vdots \\ x_{1_im} \end{pmatrix}+w_{2i}\begin{pmatrix} x_{2_i1} \\ x_{2_i2} \\ \vdots \\ x_{2_im} \end{pmatrix}+\cdots+w_{ki}\begin{pmatrix} x_{k_i1} \\ x_{k_i2} \\ \vdots \\ x_{k_im} \end{pmatrix}$$

自然可以定义这种表示的第 $i$ 个样本 $x_i$ 的损失为：

$$J(x_i)=\left\|x_i-(w_{1i}x_{1_i}+w_{2i}x_{2_i}+\cdots+w_{ki}x_{k_i})\right\|_2^2=\left\|x_i-\sum_{j=1}^{k}w_{ji}x_{j_i}\right\|_2^2$$

因此，所有样本的损失即为：

$$J(W)=\sum_{i=1}^{m}J(x_i)=\sum_{i=1}^{m}\left\|x_i-\sum_{j=1}^{k}w_{ji}x_{j_i}\right\|_2^2$$

且满足 $\sum_{j=1}^{k}w_{ji}=1(i=1,2,\cdots,m)$，即要求第 $i$ 个样本 $x_i$ 是其邻近样本 $N_i=\{x_{1_i},x_{2_i},\cdots,x_{k_i}\}$ 的近似凸组合。

为了求出最好的近似线性表示，即要求解下述优化问题：

$$\begin{aligned} &\min J(W)=\sum_{i=1}^{m}\left\|x_i-\sum_{j=1}^{k}w_{ji}x_{i_j}\right\|_2^2 \\ &\text{s.t.}\ \sum_{j=1}^{k}w_{ji}=1 \end{aligned} \qquad (2.4.10)$$

第一步，运用 $k$ 近邻算法得到每个数据的 k 近邻点：

$$N_i=KNN(x_i,k)=\{x_{1_i},x_{2_i},\cdots,x_{k_i}\};$$

第二步，求解优化问题（2.4.10）；

第三步，映射到低维空间。

下面我们先推导第二步优化问题（2.4.10）的求解。假设输入为 $m$ 个 $n$ 维样

本 $\{x_1,x_2,\cdots,x_m\}$，需要求出的权重系数矩阵 $W=(w_1,w_2,\cdots,w_m)\in R^{k\times m}$。下面我们逐步推导出权重系数矩阵的表达式：

$$J(W)=\sum_{i=1}^{m}\left\|x_i-\sum_{j=1}^{k}w_{ji}x_{j_i}\right\|_2^2=\sum_{i=1}^{m}\left\|\sum_{j=1}^{k}\left(x_i-x_{j_i}\right)w_{ji}\right\|_2^2=\sum_{i=1}^{m}\left\|(X_i-N_i)w_i\right\|_2^2$$

其中 $X_i=(x_i,x_i,\cdots,x_i)\in R^{n\times k}$，$N_i=(x_{1_i},x_{2_i},\cdots,x_{k_i})\in R^{n\times k}$，所以

$$J(W)=\sum_{i=1}^{m}\left\|(X_i-N_i)w_i\right\|_2^2=\sum_{i=1}^{m}w_i^T(X_i-N_i)^T(X_i-N_i)w_i \qquad (2.4.11)$$

记局部协方差矩阵 $Z_i=(X_i-N_i)^T(X_i-N_i)$，则

$J(W)=\sum_{i=1}^{m}w_i^TZ_iw_i$，且 $\sum_{j=1}^{k}w_{ij}=1$ 等价于 $e_k^Tw_i=1(i=1,2,\cdots,m)$，

其中 $e_k=(1,1,\cdots,1)^T\in \mathrm{R}^k$。因此，（2.4.10）等价于

$$\begin{aligned}&\min J(W)=\sum_{i=1}^{m}w_i^TZ_iw_i\\&\text{s.t. }e_k^Tw_i=1(i=1,2,\cdots,m)\end{aligned} \qquad (2.4.12)$$

运用拉格朗日乘子法：

$$L(W)=\sum_{i=1}^{m}\left[w_i^TZ_iw_i+\lambda(e_k^Tw_i-1)\right] \qquad (2.4.13)$$

令 $\frac{\partial L(W)}{\partial(w_i)}=0$，得到

$$2Z_iw_i-\lambda e_k=0$$

$$w_i=\lambda' Z_i^{-1}e_k,\quad \lambda'=\frac{1}{2}\lambda$$

利用 $e_k^Tw_i=1$，对 $w_i$ 归一化，得到权重系数 $w_i$ 为：

$$w_i=\frac{Z_i^{-1}e_k}{e_k^TZ_i^{-1}e_k}\in R^k(i=1,2,\cdots,m) \qquad (2.4.14)$$

下面推导第三步，即映射到低维空间。我们得到了高维的权重系数 $w_i$，希望这些权重系数 $w_i$ 对应的线性关系在降维后的低维空间中同样保持。假设输入的样本为：

$$X=\begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{pmatrix}=\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}\in R^{m\times n}$$

降维后的数据为：

$$Y=\begin{pmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_m^T \end{pmatrix}\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1d} \\ y_{21} & y_{22} & \cdots & y_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{md} \end{pmatrix}\in R^{d\times m} \quad (d<<n)$$

根据前面的讨论，我们自然需要最小化损失函数：

$$\Phi(Y)=\sum_{i=1}^{m}\left\| y_i-\sum_{j=1}^{k}w_{ji}y_{j_i}\right\|_2^2 \qquad (2.4.15)$$

为了得到标准化的低维数据，一般加入如下约束条件：

$$\frac{1}{m}\sum_{i=1}^{m}y_i y_i^T=I \qquad (2.4.16)$$

为了得到降维后的数据 $Y$，即要求解下述优化问题：

$$\begin{aligned} &\min\Phi(Y)=\sum_{i=1}^{m}\left\| y_i-\sum_{j=1}^{k}w_{ji}y_{j_i}\right\|_2^2 \\ &\text{s.t. }\frac{1}{m}\sum_{i=1}^{m}y_i y_i^T=I \end{aligned} \qquad (2.4.17)$$

首先，对于样本 $x_i$，需要用一个 $m\times m$ 的稀疏矩阵 $W_i^s=(w_{ji}^s)\in R^{m\times m}$ 来表示 $W\in R^{k\times m}$：

$$w_{ji}^s=\begin{cases} w_{ji}, j\le k \\ 0, \quad \text{否则} \end{cases} \qquad (2.4.17)$$

因此，$\sum_{j=1}^{m} w_{ji}^{s} y_{ji} = \sum_{j=1}^{k} w_{ji} y_{ji} = YW_i^s$。所以，

$$\Phi(Y) = \sum_{i=1}^{m} \left\| y_i - \sum_{j=1}^{k} w_{ji} y_j \right\|_2^2 = \sum_{i=1}^{m} \left\| Y(I_i - W_i^s) \right\|_2^2$$

因为对于对称矩阵 $A = (a_1, a_2 \cdots, a_m)$，总有：

$$\sum_{i=1}^{m} \|a_i\|_2^2 = \sum_{i=1}^{m} a_i^T a_i = tr(AA^T)$$

所以，

$$\Phi(Y) = \sum_{i=1}^{m} \left\| Y(I_i - W_i^s) \right\|_2^2 = tr\left(Y^T (I_i - W_i^s)^T (I_i - W_i^s) Y\right) = tr\left(Y^T M Y\right),$$

其中

$$M = (I_i - W_i^s)^T (I_i - W_i^s)$$

所以，（2.4.17）变为

$$\min \Phi(Y) = tr\left(YMY^T\right)$$
$$\text{s.t. } YY^T = mI$$

运用拉格朗日乘子法，记

$L(Y) = YMY^T - \lambda(YY^T - mI)$，令 $\frac{\partial L(Y)}{\partial Y} = 2MY - 2\lambda Y = 0$，　得到

$$MY = \lambda Y$$

可见，$Y$ 是 $M$ 的特征向量组成的矩阵。为了将数据降到 $d$ 维，我们只需要取 $M$ 的最小的 $d$ 个非零的特征值对应的特征向量 $Y = (y_1, y_2, \cdots, y_d)$，而一般第一个最小的特征值接近零，此时对应的特征向量元素全为 1，不能反映数据的特征。通常

选取 $M$ 的从第二小到第 $d+1$ 小的非零的特征值对应的特征向量 $Y=(y_2,y_3,\cdots,y_{d+1})$ 来得到最终的降维数据。

（3）LLE 算法的流程

LLE 算法可以用一张图可以表示如下：

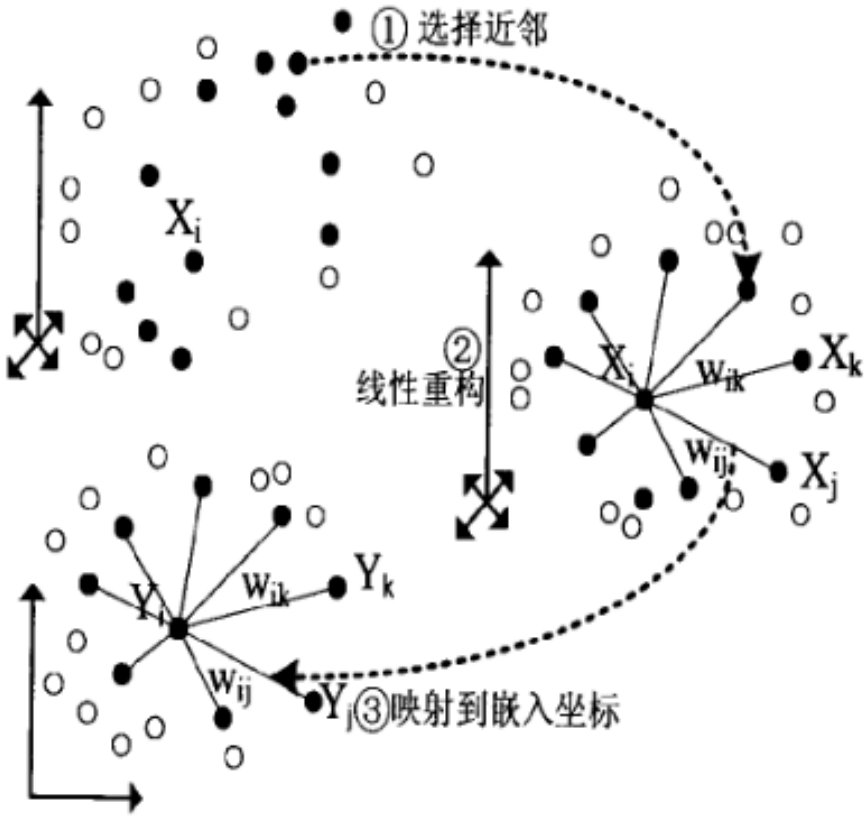


图 2.29 LLE 算法流程图

从图中可以看出，LLE 算法主要分为三步，第一步是求 K 近邻的过程，这个过程使用了和 KNN 算法一样的求最近邻的方法。第二步是对每个样本求它在邻域里的 K 个近邻的线性关系，得到线性关系权重系数 W。第三步是利用权重系数来在低维里重构样本数据。

具体过程如下：

算法：

输入：样本 $n$ 维集合 $X=\{x_1,x_2,\cdots,x_m\}$，最近邻数 $k$，降维后的样本数据的维数 $d$；

输出：降维后的 $d$ 维样本集合矩阵 $Y=\{y_1,y_2,\cdots,y_m\}$

1） for i=1 to m， 计算和 $x_i$ 最近邻的 $k$ 个最近邻：

$$N_i=KNN(x_i,k)=\{x_{1_i},x_{2_i},\cdots,x_{k_i}\};$$

2） for i=1 to m，求出局部协方差矩阵 $Z_i=(X_i-N_i)^T(X_i-N_i)$ 和权重系数向量：$w_i=\dfrac{Z_i^{-1}e_k}{e_k^TZ_i^{-1}e_k}$；

3） 由权重系数向量组成权重系数矩阵 $W$，扩充矩阵 $W$ 为 $W^s\in R^{m\times m}$：

$$w_{ji}^s=\begin{cases}w_{ji},j\le k\\0,\quad \text{否则}\end{cases};$$

4） 计算矩阵 $M=(I_i-W_i^s)^T(I_i-W_i^s)$；

5） 计算 $M$ 的最小的 $d+1$ 个非零的特征值对应的特征向量：

$$Y=(y_1,y_2,\cdots,y_{d+1})$$

6） 选取 $M$ 的从第二小到第 $d+1$ 小的非零的特征值对应的特征向量，得到降维后的样本数据 $Y=(y_2,y_3,\cdots,y_{d+1})$。

（4）LLE 的一些改进算法

LLE 算法很简单高效，但是却有一些问题：

（1）使用单个权重不够稳定；

（2）如果近邻数 k 大于输入数据的维度时，权重系数矩阵不是满秩的。

为了解决这样类似的问题，有一些 LLE 的变种产生出来。如 Heissian LLE [43]和 Modified LLE[44]。HLLE 与 LLE 十分类似，HLLE 也将流形局部线性的块映射到低维空间。HLLE 使用 PCA 对每个点的切空间进行估计，而 LLE 通过邻点线性组合的方式估计局部线性空间。HLLE 使用 Heissian 矩阵的 Frobenius 范数度量弯曲程度，并将其作为目标函数最小化；而 LLE 使用之前学习的线性关系指导嵌入低维流形。HLLE 结果比 LLE 好很多，但计算复杂性也高上很多。MLLE 使用多个权重来表示局部几何关系。另一个比较好的 LLE 的变种是局部切空间对齐方法（ Local tangent space alignment：LTSA[49]），它希望保持数据集局部的几何关系，在降维后希望局部的几何关系得以保持，同时利用了局部几何到整体性质过渡的技巧。

**2. 拉普拉斯特征映射**（Laplacian Eigenmaps: LE[40]）

拉普拉斯特征映射（Laplacian Eigenmaps）是一种基于图的降维算法，它从局部的角度去构建数据之间的关系，它希望**相互间有关系的点（在图中相连的点）在降维后的空间中尽可能的靠近，从而在降维后仍能保持原有的数据结构。**

拉普拉斯特征映射的思想来源于图论中的谱方法。图拉普拉斯矩阵也叫导纳矩阵、基尔霍夫矩阵或离散拉普拉斯算子。主要应用在图论中，作为一个图的矩阵表示。

**定义：**给定一个有 $n$ 个顶点的图 $G$，它的拉普拉斯矩阵定义为：$L=(l_{ij})_{n\times n}$，

定义为 $L=D-W$，其中 $D$ 为图的度矩阵，是对角线元素为度数的对角阵；$W$ 为图的邻接矩阵，也叫权重矩阵。

图的 Laplacian 矩阵包含了许多诸如图的连通性的信息，从而被用来获取流形的局部信息。

（1）LE 算法的基本思想：

使用一个无向有权图来描述一个流形，通过图的嵌入(graph embedding) 来找低维表示。其实就是保持图的局部邻接关系的前提下，把流形数据形成的图从高位空间映射在一个低维空间中。**LE 希望保持流形中的近邻关系，即将原始空间中相近的点映射成目标空间中相近的点。**

（2）LE 算法的推导

拉普拉斯特征映射通过构建邻接矩阵为 $W$ 的图来重构数据流形的局部结构特征。其主要思想是，如果两个数据实例 $i$ 和 $j$ 很相似，那么 $i$ 和 $j$ 在降维后目标

子空间中应该尽量接近。设数据实例的数目为$n$，目标子空间，即最终的降维目标的维度为$d$。定义$n\times d$大小的矩阵$Y$，其中每一个行向量$y_i^T$是数据实例$i$在目标$d$维子空间中的向量表示（即降维后的数据实例$i$）：

$$Y=\begin{pmatrix} y_1^T \\ y_2^T \\ \vdots \\ y_n^T \end{pmatrix}=\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1d} \\ y_{21} & y_{22} & \cdots & y_{2d} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nd} \end{pmatrix}$$

我们的目的是让相似的数据样例$i$和$j$在降维后的目标子空间里仍旧尽量接近，故拉普拉斯特征映射优化的目标函数如下：

$$min\sum_{i,j}||y_i-y_j||^2W_{ij}$$

由于

$$\begin{aligned}
&\sum_{i=1}^{n}\sum_{j=1}^{n}||y_i-y_j||^2W_{ij} \\
&=\sum_{i=1}^{n}\sum_{j=1}^{n}(y_i^Ty_i-2y_i^Ty_j+y_j^Ty_j)W_{ij} \\
&=\sum_{i=1}^{n}(\sum_{j=1}^{n}W_{ij})y_i^Ty_i+\sum_{j=1}^{n}(\sum_{i=1}^{n}W_{ij})y_j^Ty_j-2\sum_{i=1}^{n}\sum_{j=1}^{n}y_i^Ty_jW_{ij} \\
&=2\sum_{i=1}^{n}D_{ii}y_i^Ty_i-2\sum_{i=1}^{n}\sum_{j=1}^{n}y_i^Ty_jW_{ij} \\
&=2\sum_{i=1}^{n}(\sqrt{D_{ii}}y_i)^T(\sqrt{D_{ii}}y_i)-2\sum_{i=1}^{n}y_i^T(\sum_{j=1}^{n}y_jWij) \\
&=2trace(Y^TDY)-2\sum_{i=1}^{n}y_i^T(YW)_i \\
&=2trace(Y^TDY)-2trace(Y^TWY) \\
&=2trace[Y^T(D-W)Y] \\
&=2trace(Y^TLY)
\end{aligned}$$

其中$D$为图的度矩阵，是对角线元素为度数的对角阵；$W$为图的邻接矩阵，$L$为图拉普拉斯矩阵。

变换后拉普拉斯特征映射优化目标函数为：

$$\begin{aligned}&\min tr(Y^T LY)\\&\text{s.t.}\,Y^T DY = I\end{aligned}$$

其中约束条件 $Y^T DY = I$ 保证优化问题有解。用拉格朗日乘子法求解：

$$f(Y) = tr(Y^T LY) + tr\left(\Lambda(Y^T DYT - I)\right)$$

$$\frac{\partial f(Y)}{\partial y} = LY + L^T Y + D^T Y\Lambda^T + DY\Lambda = 2LY + 2DY\Lambda$$

令 $\frac{\partial f(Y)}{\partial y} = 0$，得到

$$LY = -DY\Lambda$$

其中 $\Lambda$ 为乘子矩阵，是一个对角矩阵，$L, D$ 是实对称矩阵。对于单独的向量 $y$ 有 $Ly = \lambda Dy$，是一个广义特征值问题。

由于 $LY = -DY\Lambda$ 和 $Y^T DY = I$，所以目标函数：

$$tr(Y^T LY) = tr(-Y^T DY\Lambda) = tr(-\Lambda)$$

即为特征值之和，最小化目标函数即要最小化特征值之和。因此通过求得最小的特征值对应的特征向量，即可达到降维的目的。

（3）LE 算法的步骤

算法：

**第 1 步：**构建图。使用某种方法，将所有的样本点构成一个图，如使用 KNN 算法，将每个点最近的 $k$ 个点与之相连，$k$ 是一个给定的数；

**第 2 步：**确定权重。确定点与点之间的权重大小，如可以选热核函数来确定，即若 $i$ 和 $j$ 相连，则其之间的权重设定为 $W_{ij}=e^{\frac{\|x_i-x_j\|^2}{t}}$；另外一种可选的简单值为若 $i$ 和 $j$ 相连，则其之间的权重设定为 $W_{ij}=1$，否则设为 $W_{ij}=0$。

**第 3 步：**特征映射。计算拉普拉斯矩阵 $L$ 的特征值与特征向量：$Ly=\lambda LD$，使用最小的 $m$ 个特征值对应的特征向量作为降维后的数据再表达输出。

（4）LE 算法的数值实验

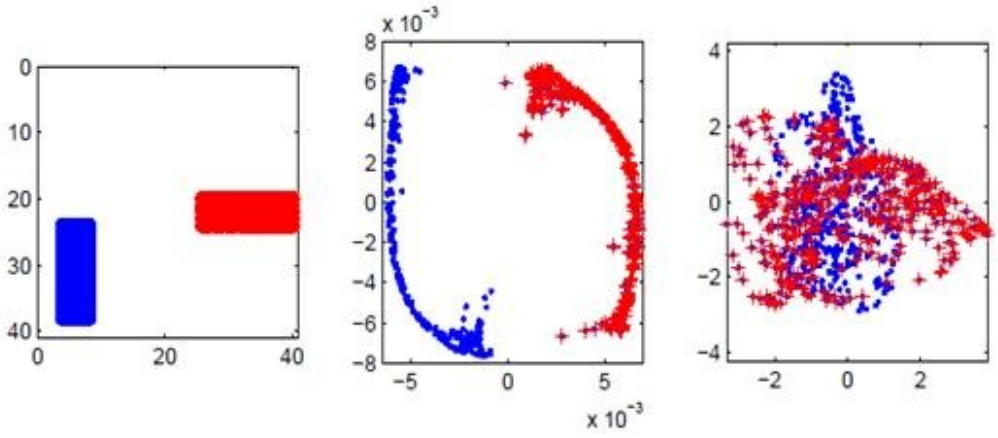


图 2.30 LE 算法的数值实验图

上图所示左边的图表示有两类数据点（数据是图片），中间图表示采用拉普拉斯特征映射降维后每个数据点在二维空间中的位置，右边的图表示采用 PCA 并取前两个主要方向投影后的结果，可以清楚地看到，在此分类问题上，拉普拉斯特征映射的结果明显优于 PCA。

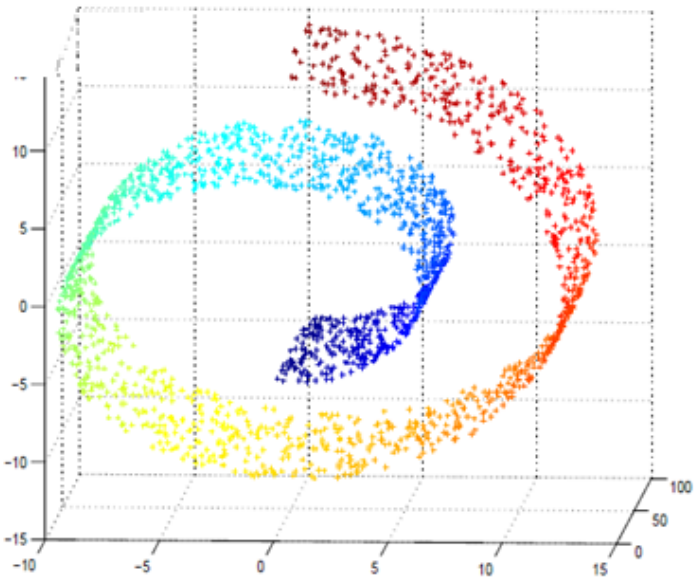


图 2.31 在“swiss roll”上的随机的 2000 个数据点

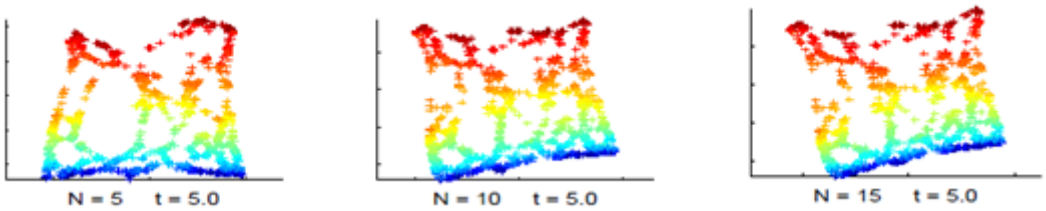


图 2.32 LE 算法高维数据的降维表示

上图说明的是高维数据（图中 3D）也有可能是具有低维的内在属性的（图中 Swiss roll 实际上是 2D 的），但是这个低维不是原来坐标表示，例如如果要保持局部关系，蓝色和下面黄色是完全不相关的，但是如果只用任何 2D 或者 3D 的距离来描述都是不准确的。下面的三个图是拉普拉斯特征映射在不同参数下的展开结果（降维到 2D），可以看到，似乎是要把整个卷拉平了，蓝色和黄色差的比较远，很好地保留了数据原有的结构。

**5. 半定嵌入（Semidefinite embedding ： SDE）/最大方差展开（maximum variance unfolding ： MVU[41]）**

MVU 这个名字更好的展现了该算法的基本思想。我们想象每个点和它的邻点有硬杆相连，MVU 通过最大化不是邻点的点之间的距离来展开流形。SDE 指使用半正定规划算法寻找合适的内积矩阵。约束为邻域距离不变，即 $\|x_i - x_j\| = \|y_i - y_j\|$。而

$$\begin{aligned}\|y_i - y_j\|^2 &= \|y_i\|^2 + \|y_j\|^2 - 2\langle y_i, y_j\rangle \\ &= \langle y_i, y_i\rangle + \langle y_j, y_j\rangle - 2\langle y_i, y_j\rangle \\ &= B_{ii} + B_{jj} - 2B_{ij}.\end{aligned}$$

故约束为：

$$B_{ii} + B_{jj} - 2B_{ij} = \|x_i - x_j\|^2$$

目标函数为最大化点间距离：

$$\max \sum_{i,j} \|y_i - y_j\| = \max \sum_{i,j} B_{ii} + B_{jj} - 2B_{ij} = \max \sum_{i,j} B_{ii} + B_{jj} = \max tr(B)\,,$$

其中 $\sum_{i,j} B_{ij} = 0$ 通过把 y 移到中心点实现。

类似 Isomap 特征值分解中正特征值的数量也可以作为流形的维数。

---

**Semidefinite Embedding**

input: $x_1, \ldots, x_n \in \mathbb{R}^D$, $k$

1. Solve the following semidefinite program:

   maximize $\quad \mathrm{tr}(B)$

   subject to: $\quad \sum_{ij} B_{ij} = 0;$

   $B_{ii} + B_{jj} - 2B_{ij} = \|x_i - x_j\|^2$

   (for all neighboring $x_i, x_j$);

   $B \succeq 0.$

2. Compute the spectral decomposition of $B$: $B = U\Lambda U^\top$.

3. Form $\Lambda_+$ by setting $[\Lambda_+]_{ij} := \max\{\Lambda_{ij}, 0\}$.

4. Return $Y := U\Lambda_+^{1/2}$.

---

然而半正定规划很难解决规模过大( $n \geq 2000$ )的问题，一种方法是使用投影次梯度方法求解。Landmark SDE(lSDE)[42]使用矩阵分解的方法减小半正定规划的规模。算法的思想是把内积矩阵 B 分解成 $QLQ^T$，L 为 l 个地标点的矩阵。由半正定规划求这 l 个地标点，剩余的点由地标点表示。

**6. 局部保留投影**（Locality preserving projection：LPP[45]）

LPP 算法提出的目的是为了实现非线性流形的学习和分析，LPP 可以提取最具有判别性的特征来进行降维，是一种保留了局部信息，降低影响图像识别的诸多因素的降维方法，这种算法本质上是一种线性降维方法，由于其巧妙的结合了拉普拉斯特征映射算法（LE）的思想，从而可以在对高维数据进行降维后有效地保留数据内部的非线性结构。线性注定了它的性能有限，当然速度也快很多。

与其它非线性降维方法相比，LPP 方法可以将新增的测试数据点，通过映射在降维后的子空间找到对应的位置，而其他非线性方法只能定义训练数据点，无法评估新的测试数据。LPP 方法可以很容易地将新的测试数据点根据特征映射关系（矩阵），投影映射在低维空间中。

LPP 算法分为三步：

第 1 步：建立邻接矩阵，即建立矩阵描述点对之间是否相连。可以选择 $\varepsilon$ 邻域法，即将距离小于 $\varepsilon$ 的点对间相连。也可以选择 k 近邻法，即将每个点与它的 k 个近邻相连。

第 2 步：选择边的权重。可以选择 $W_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}$，或者有边相连则权重为 1。

第 3 步：特征映射。计算 $XLX^T\mathbf{a} = \lambda XDX^T\mathbf{a}$ 的前 l 个最小的特征值对应的特征向量 a 作为线性映射的变换矩阵。其中 L 为拉普拉斯矩阵，D 为度矩阵。

第 3 步来源于最小化低维表示下加权的距离平方和。想法在于原空间中距离近的点权重大，在低维空间表示中最小化目标时所占的比重大。即

$$\min \sum_{ij}(y_i - y_j)^2 W_{ij}$$

等价变形：

$$\frac{1}{2}\sum_{ij}(y_i - y_j)^2 W_{ij} = \frac{1}{2}\sum_{ij}(\mathbf{a}^T\mathbf{x}_i - \mathbf{a}^T\mathbf{x}_j)^2 W_{ij}$$

$$= \sum_i \mathbf{a}^T\mathbf{x}_i D_{ii}\mathbf{x}_i^T\mathbf{a} - \sum_{ij}\mathbf{a}^T\mathbf{x}_i W_{ij}\mathbf{x}_j^T\mathbf{a} = \mathbf{a}^T X(D-W)X^T\mathbf{a} = \mathbf{a}^T XLX^T\mathbf{a}$$

每个点的度标志着每个点的重要程度，约束

$$\mathbf{y}^T D\mathbf{y} = 1 \Rightarrow \mathbf{a}^T XDX^T\mathbf{a} = 1$$

是加权意义下的表示规范化。求投影方向 a 化为优化问题

$$\arg\min_{\mathbf{a}^T XDX^T\mathbf{a}=1} \mathbf{a}^T XLX^T\mathbf{a}$$

再利用拉格朗日乘子法，其中 a 就是

$$XLX^T\mathbf{a} = \lambda XDX^T\mathbf{a}$$

的最小特征值的特征向量。

**7. 黎曼流形学习（Riemannian Manifold Learning：RML[46]）**

在介绍黎曼流形学习之前，首先总结将上述主要算法进行比较，见表 2.2。

表 2.2 主要的流形学习算法

| Algorithm | Property | Description and Comments |
|---|---|---|
| ISOMAP | Isometric mapping | Computes the geodesic distances, and then uses MDS. Computationally expensive. |
| LLE | Preserving linear reconstruction weights | Computes the reconstruction weights for each point, and then minimizes the embedding cost by solving an eigenvalue problem. |
| C-ISOMAP and L-ISOMAP | Conformal ISOMAP and landmark ISOMAP | C-ISOMAP preserves angles. L-ISOMAP efficiently approximates the original ISOMAP by choosing a small number of landmark points. |
| Laplacian eigenmaps | Locality preserving | Minimizing the squared gradient of an embedding map is equal to finding eigenfunctions of the Laplace-Beltrami operator. |
| HLLE or Hessian eigenmaps | Locally isometric to an open, connected subset | A modification of Laplacian eigenmaps by substituting the Hessian for the Laplacian. Computationally demanding. |
| Manifold charting | Preserving local variance and neighborhood | Decomposes the input data into locally linear patches, and then merges these patches into a single low-dimensional coordinate system by using affine transformations. |
| LTSA | Minimizing the global reconstruction error | First constructs the tangent space at each data point, and then aligns these tangent spaces with a global coordinate system. |
| SDE | Local isometry | Maximizing the variance of the outputs, subject to the constraints of zero mean and local isometry. Computationally expensive by using semidefinite programming. |
| Laplacianfaces | Linear version of Laplacian eigenmaps | The minimization problem reduces to a generalized eigenvalues problem. |
| Diffusion maps | Preserving diffusion distances | Given a Markov random walk on the data, the diffusion map is constructed based on the first few eigenvalues and eigenvectors of the transition matrix $P$. |
| Conformal eigenmaps | Angle-preserving embedding | Maximizing the similarity of triangles in each neighborhood. More faithfully preserving the global shape and the aspect ratio. Semidefinite programming is used to for optimization. |
| Incremental ISOMAP | Data are collected sequentially. | Efficiently updates all-pair shortest path distances, and solves an incremental eigenvalue problem. |

黎曼流形学习的目的是解决上述方法(上表中)的一些问题：（1）保持度量问题，在高斯曲率不为 0 的流形上表现不好，且相邻关系发生了很大的变化；（2）总体代价函数使降维后的图像出现的不正确的重叠(如图 4)；（3）不能增量学习；（4）全局问题难解，如使用 eigenmaps method 虽简化了计算但会导致问题（1）。

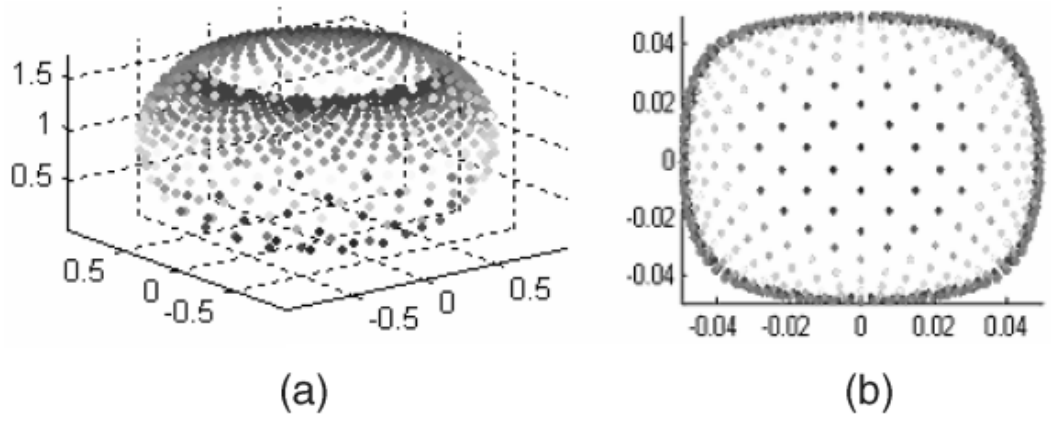


图 2.33 平均成本问题

（a）一个被刺穿的球体数据集 （b）使用谱方法 LTSA 嵌入降维表达结果

文章的思想来源于[47]：黎曼流形 M 能等度嵌入欧式空间 $R^m$。通过使用黎

曼法坐标框架(normal coordinate chart)保持测地距离的性质，获得保持测地距离的低维特征。为了解决问题（2）-（4），文章采用由起点出发逐渐延伸的局部方法计算低维坐标。

算法首先确定近邻。近邻大小自适应，即每个点的近邻大小不一。流形的本质维数与近邻大小一同确定。这些参数通过流形的重构来确定。算法首先寻找 K 近邻(K 足够大)，然后定义“可见”邻居

$$VN(x_i) = \{y \in KNN(x_i) | \langle x_i - z, y - z \rangle \geq 0, \forall z \in KNN(x_i)\},$$

并且仅保留可见邻居。接下来移除短回路的邻点(类似虫洞这样的)，将由一点出发的所有边升序排列，用 PCA 计算前 j 个点的局部本质维数(非零分量个数)，然后计算带来维数变化的边与前一条边的长度之差，如果超过某一阈值，则后一条边被认为是短回路的边。(带来维数变化的同时距离变化不小)。确定完邻点以后就得到了一个单纯复形，单纯复形的维数作为 M 的本质维数的估计。

算法分为三步：

第 1 步：选择一个基准点 p 作为法坐标框架的原点；

第 2 步：建立流形的切空间 $T_pM$ 下的笛卡尔坐标系；

第 3 步：用 Dijkstra 算法找单源(p)最短路并求法坐标下最短路终点的对应坐标。

作者先前的算法[48]选择使到所有其它点测地距离中最大的距离最小的点作为原点，并使用 Dijkstra 近似测地距离。

第 2 步计算 $T_pM = x_0 + \mathrm{span}\{x_1 - x_0, \ldots, x_d - x_0\}$，$\{x_1 - x_0, \ldots, x_d - x_0\}$是线性无关向量组。$(p; \vec{e}_1, \cdots, \vec{e}_d)$，是$\{x_1 - x_0, \ldots, x_d - x_0\}$的 Gram-Schmidt 正交化。第 3 步，用 Dijkstra 算法求单纯复形上的最短路。对于与 p 相连的 q，通过求

$$\min_x \left\| q - \left( p + \sum_{i=1}^{d} x_i \vec{e}_i \right) \right\|_{R^n}^2$$

找 q 在切空间上的投影坐标。q 的法坐标为 $\frac{\|q-p\|_{R^n}}{\|x\|_{R^d}} x$。对不与 p 直接相连的 q，b 为p到q最短路上倒数第二个点。设$\{c_1, \ldots, c_k\}$ 与b相连且法坐标已经计算(k>d，否则将 q 当做与 p 相连)。为了保持 b 的邻点的角度和 pb 之间的距离，求解

$$\cos\theta_i = \frac{\langle q-b, c_i-b \rangle}{\|q-b\| \cdot \|c_i-b\|} \approx \cos\theta_i' = \frac{\langle q'-b', c_i'-b' \rangle}{\|q'-b'\| \cdot \|c_i'-b'\|},$$

$$i = 1, 2, \ldots, k$$

$$\|q-b\| = \|q'-b'\|,$$

q’等为法坐标。写成标准形式

$$\min_x \|Ax - y\|^2 \text{ subject to } \|x\| = \alpha,$$

其中

$$x = q' - b', \alpha = \|q - b\|, y = [\cos\theta_1, \ldots, \cos\theta_k]^T.$$

算法有两个限制：（1）第 2 步中的线性无关组只能通过试错寻找，改进的算法[46]中将 Gram-Schmidt 正交化改成了 PCA，只需多取一些点进行 PCA 保留 d 维就不必再试错，与 p 直接相连的点的法坐标也是用 PCA 计算；（2）使用最短路近似测地线。如果样本足够稠密这将是很好的估计，但样本往往不够稠密。新算法[10]使用测地锚点(geodesic anchor point)替代 a 替代原算法中的 b，如图 7。测地锚点通过最小化 p 到 q 的曲线(过 a)长度，其中 a 位于(d-1)维子流形 M'中。M'中的点到 p 的测地距离与 b 相同。而 b 的局部的测地距离由一个二次多项式进行估计，例如二维点$(x_1, x_2)$到 p 的测地距离就由

$$u(x_1, x_2) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_1 x_2 + w_5 x_2^2.$$

估计。这些系数由已求得测地距离的 b 的邻点来确定(要求 k>(d+2)(d+1)/2 以正确求解这个估计函数)。
下图中左图为[48 的算法，右图为[46 的算法。

图 2.33

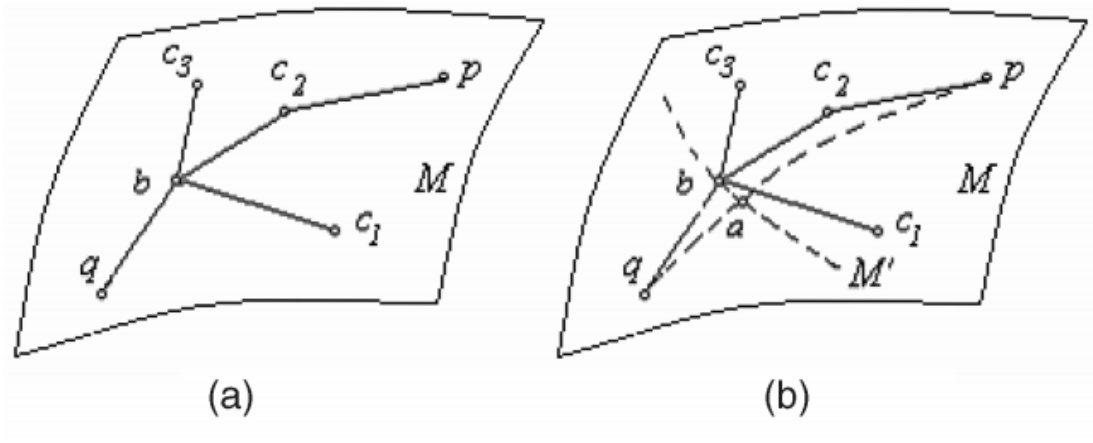


图 2.34

An example illustrating how to compute the normal coordinates of point $q$, which is not in the local region of the base point $p$. (a) Previous algorithm. Point $b$ is the preceding node on the shortest path from $p$ to $q$. Points $c_1$, $c_2$, and $c_3$ are neighbors of $b$. Suppose that the normal coordinates of $b$, $c_1$, $c_2$, and $c_3$ have been computed. The normal coordinates of $q$ are computed based on geometric relations to $b$, $c_1$, $c_2$, and $c_3$. (b) The proposed new algorithm based on the geodesic anchor point $a$. $M'$ is the level set passing through points $b$ and $a$. The normal coordinates of $q$ are computed based on geometric relations to $a$, $c_1$, $c_2$, and $c_3$, where $a$ replaces the role of $b$ in the new algorithm.

**数据再表达的流形学习总结**

流形学习可以概括为：在保持流形上点的某些几何性质特征的情况下，找出一组对应的内蕴坐标(intrinsic coordinate)，将流形尽量好的展开在低维平面上，这种低维表示也叫内蕴特征(intrinsic feature)，外围空间的维数叫观察维数，其表示叫自然坐标，在统计上称为观察（observation）。

总的来说流行学习的大体框架是相似的：

（1）创建数据的局部相似性表示；

（2）保持这种相似性将数据嵌入低维空间。

流形学习的优点在于它的理论很漂亮，而且可以用于无监督的数据再表达学习。缺点也很明显，流行学习目前为止没有真正能够实用的算法。谱方法对噪声十分敏感。而且为了得到准确估计的流形，需要进行大量采样，在现实问题的高维数据中是很难实现的。此外，由于实际数据位于高维空间，任意两个样本的距离其实差距不大。而流形学习需要近邻点来刻画局部几何，并且近邻点的测地距离为欧氏距离。这也会给流行学习在实际问题的应用中带来很大困难。此外，要将局部空间近似为切空间，也需要足够稠密的数据。在维数诅咒下，稠密的数据采样是很难实现的。

| 方法简称 | 所保持的几何属性 | 全局/局部关系 | 计算复杂度 |
|---|---|---|---|
| ISOMAP | 点对测地距离 | 全局 | 非常高 |
| LLE | 局部线性重构关系 | 局部 | 低 |
| LE | 局部邻域相似度 | 局部 | 低 |
| HLLE | 局部等距性 | 局部 | 高 |
| LTSA | 局部坐标表示 | 全局+局部 | 低 |
| MVU | 局部距离 | 全局+局部 | 非常高 |
| Logmap | 测地距离与方向 | 局部 | 非常低 |
| Diffusion Maps | diffusion距离 | 全局 | 中等 |