# 维数灾难的提出背景

**维数灾难**（英语：curse of dimensionality，又名**维度的诅咒**）是一个最早由理查德·贝尔曼（Richard E. Bellman）在考虑优化问题时首次提出来的术语，用来描述当（数学）空间维度增加时，分析和组织高维空间（通常有成百上千维），因体积指数增加而遇到各种问题场景。这样的难题在低维空间中不会遇到，如物理空间通常只用三维来建模。维数灾难涉及抽样、组合数学、机器学习、数据挖掘和数据库等诸多领域。[1]

“维数灾难”通常是用来作为不要处理高维数据的无力借口。然而，学术界一直都对其有兴趣，而且在继续研究。另一方面，也由于本征维度的存在，其概念是指任意低维数据空间可简单地通过增加空余（如复制）或随机维将其转换至更高维空间中，相反地，许多高维空间中的数据集也可削减至低维空间数据，而不必丢失重要信息。这一点也通过众多降维方法的有效性反映出来，如应用广泛的主成分分析方法。针对距离函数和最近邻搜索，当前的研究也表明除非其中存在太多不相关的维度，带有维数灾难特色的数据集依然可以处理，因为相关维度实际上可使得许多问题（如聚类分析）变得更加容易。另外，一些如马尔科夫蒙特卡洛或共享最近邻搜索方法经常在其他方法因为维数过高而处理棘手的数据集上表现得很好。

# 维数灾难的内涵

在很多领域中都有维数灾难现象。这些问题的共同特色是当维数提高时，空间的体积提高太快，因而可用数据变得很稀疏。稀疏性对于任何要求有统计学意义的方法而言都是一个问题，为了获得在统计学上正确并且有可靠的结果，用来支撑这一结果所需要的数据量通常随着维数的提高而呈指数级增长。而且，在组织和搜索数据时也有赖于检测对象区域，这些区域中的对象通过相似度属性而形成分组。然而在高维空间中，所有的数据都很稀疏，从很多角度看都不相似，因而平常使用的数据组织策略变得极其低效。

## 在组合学中

在一些问题中，每个变量都可取一系列离散值中的一个，或者可能值的范围被划分为有限个可能性。把这些变量放在一起，则必须考虑很多种值的组合方式，这后果就是常说的组合爆炸。即使在最简单的二元变量例子中，可能产生的组合总数就已经是在维数上呈现指数级的 $\mathcal{O}(2^d)$。一般而言，每个额外的维度都需要成倍地增加尝试所有组合方式的影响。

## 在采样中

当在数学空间上额外增加一个维度时，其体积会呈指数级的增长。如，当要求采样点间距离不超过 $10^{-2} = 0.01$时，$10^2 = 100$ 个均匀间距的样本点足够对一个单位区间（“一个维度的立方体”）采样；但一个10维单元超立方体，则需要 $10^{20}$ 个样本点。一般而言，点距为 $10^{-n}$ 的10维超立方体所需要的样本点数量，是1维超立方体单元区间的 $10^{n(10-1)}$ 倍。在上面的 $n = 2$ 的例子中：当样本距离为0.01时，10维超立方体所需要的样本点数量会比单元区间多 $10^{18}$ 倍。这一影响就是上面所述组合学问题中的组合结果。

## 在优化中

当用数值逆向归纳法解决动态优化问题时，目标函数针对每个可能的组合都必须计算一遍，当状态变量的维度很大时，这是极其困难的。

## 在机器学习中

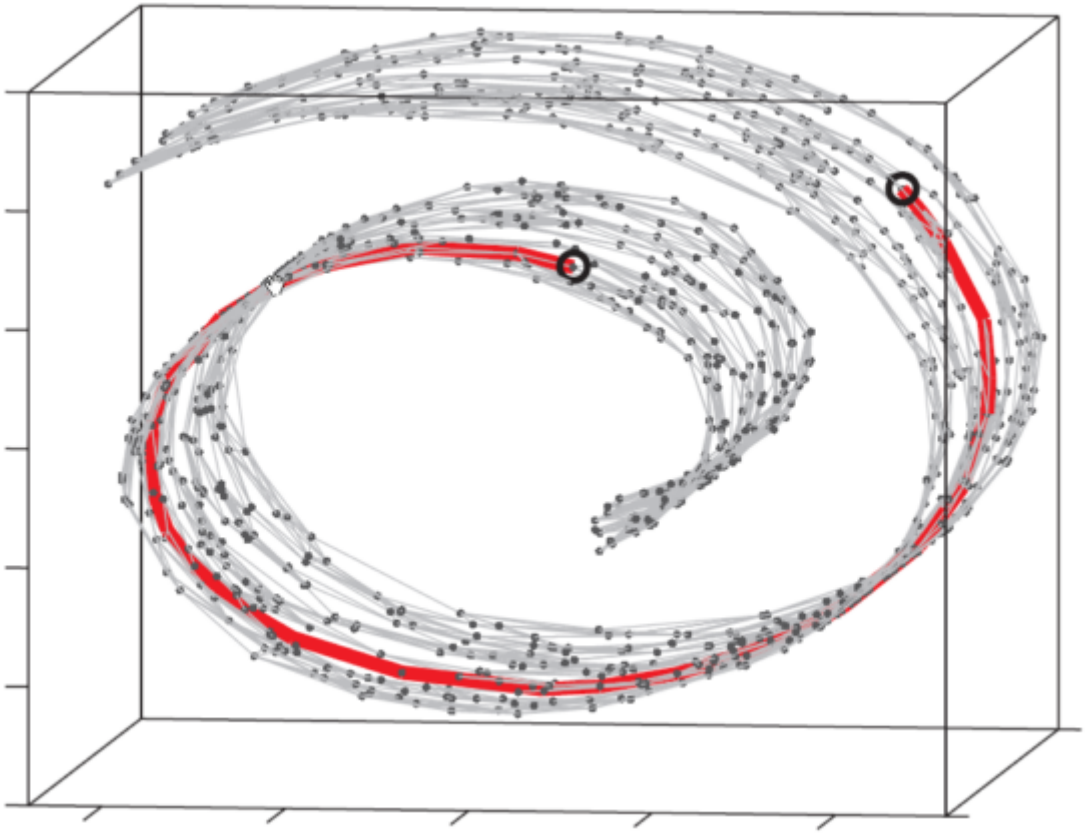
在机器学习问题中，需要在高维特征空间（每个特征都能够取一系列可能值）的有限数据样本中学习一种“自然状态”（可能是无穷分布），要求有相当数量的训练数据含有一些样本组合。给定固定数量的训练样本，其预测能力随着维度的增加而减小，这就是所谓的*Hughes影响*或*Hughes现象*（以Gordon F. Hughes命名）。

# 有关的研究工作——流形学习

维度灾难问题产生的根源是维度太大。但事实上，尽管数据的维度可能很大，但并不代表这些维度都是有用的。解释数据本质特征的实际维度可能很小，远远小于原始数据的维度。如果能够用合适的降维方法进行尽可能不损失数据信息的维度约减，就能有效地解决维度灾难问题。但传统的线性降维方法（如PCA）只是粗暴地保留了若干个数据中方差最大的方向，并没有考虑到数据中蕴含的结构信息，特别是无法处理和体现非线性的结构信息。针对此问题，本次我介绍*流形学习*这一研究方向。

**流形学习**（manifold learning）是机器学习的一种方法。它的主要观点是，我们在现实生活中观测到的高维数据实质上是一个低维流形到高维欧式空间的嵌入。流形学习的目的就是找到合适的映射，将数据还原到原本的低维流形空间中，从而实现维度约减与更好描述其性质的效果。

在方法上，流形学习实质上属于“非线性降维”。通过流形的观点，将传统基于线性映射的PCA、LDA、MDS等降维方法扩展到非线性映射，更好地抓住数据的实质结构，解决如下图（瑞士卷）等线性降维方法难以解决的问题。



在这里，我具体介绍两种流形学习的方法，*局部线性嵌入*（Locally Linear Embedding, LLE）和*最大方差展开*（Maximum Variance Unfolding, MVU）。

## 局部线性嵌入（LLE）[2]

# 原理

LLE的思路是，*任何一点都可以通过其临近点之间的线性加权组合得到。*其权重具有在平移、旋转、缩放映射下保持不变的特性，可以看作是反映数据几何结构的内蕴量。通过在原空间中对这组内蕴量的学习（误差最小化原则），可以求出这组权重 $W$。计算方法为：

记

$$\eta_{ij}=\begin{cases}1, & \text{if } x_i, x_j \text{ 是} k\text{近邻}\\ 0, & \text{if } x_i, x_j \text{ 不是} k\text{近邻}\end{cases}$$

则权重为

$$\begin{aligned}\min_{W}\quad & \varepsilon(X,W)=\sum_{\mathrm{i}}\left|\vec{X}_{\mathrm{i}}-\Sigma_{\mathrm{j}}W_{\mathrm{ij}}\vec{X}_{\mathrm{j}}\right|^2 && (1.0)\\ s.t.\quad & W_{ij}=0,\forall\eta_{ij}=0 && (1.1)\quad (\mathrm{P})\\ & \sum_j W_{ij}=1 && (1.2)\end{aligned}$$

其中优化目标(1.0)为最小化线性近似的总误差，约束(1.1)意为k近邻以外的点不参与加权，约束(1.2)说明 $W_{ij}$ 是一组“权重”。

而后对于降维后的数据，其权重 $W_{ij}$ 作为内蕴量应是保持不变的，因此用同样的方式（误差最小化原则），可以求出这组向量 $Y$。计算方法为：

$$\min_{Y}\quad \varepsilon(Y,W)=\sum_{\mathrm{i}}\left|\vec{Y}_{\mathrm{i}}-\Sigma_{\mathrm{j}}W_{\mathrm{ij}}\vec{Y}_{\mathrm{j}}\right|^2 \qquad (2.0)$$

注意到两次计算的损失函数(1.0)与(2.0)是一致的，只不过第一次是已知数据 $X$，通过最小化损失求权重 $W$；第二次是已知权重，通过最小化损失求降维后的数据 $Y$。而这两次求解的问题都是简单的二次规划，且参数只有近邻数 $k$ 和 目标维数 $r$。

故综上，LLE算法的具体流程为：

> Step1. 选取超参数k，求解二次问题(P)，得到权重W
>
> Step2. 选取超参数r，求解二次问题(2.0)，得到r维空间中的向量组Y

下图是论文中给出的算法流程图示：

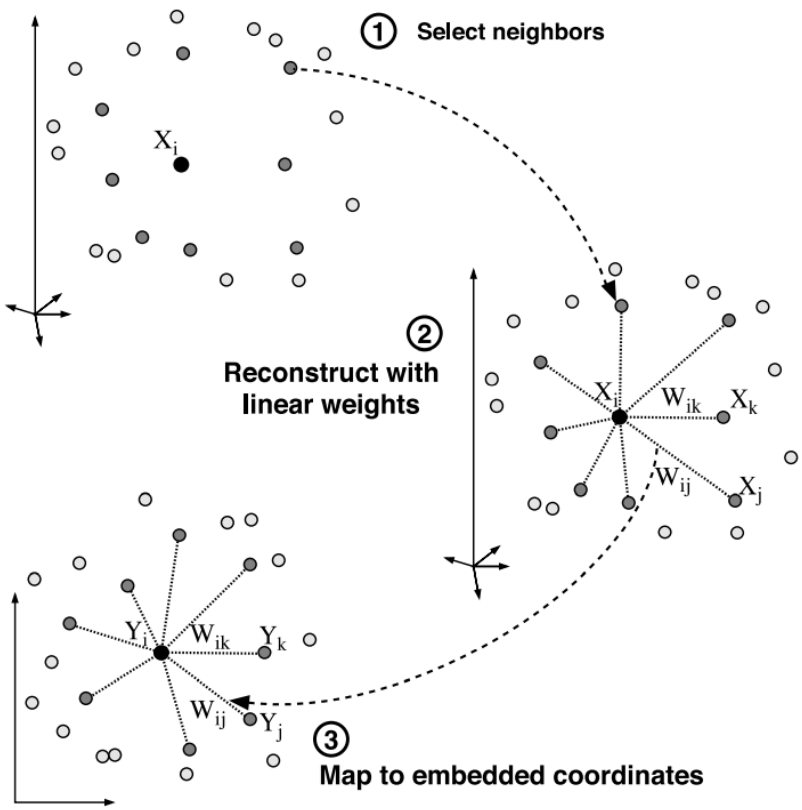



**Fig. 2.** Steps of locally linear embedding: (1) Assign neighbors to each data point $\vec{X}_i$ (for example by using the $K$ nearest neighbors). (2) Compute the weights $W_{ij}$ that best linearly reconstruct $\vec{X}_i$ from its neighbors, solving the constrained least-squares problem in Eq. 1. (3) Compute the low-dimensional embedding vectors $\vec{Y}_i$ best reconstructed by $W_{ij}$, minimizing Eq. 2 by finding the smallest eigenmodes of the sparse symmetric matrix in Eq. 3. Although the weights $W_{ij}$ and vectors $Y_i$ are computed by methods in linear algebra, the constraint that points are only reconstructed from neighbors can result in highly nonlinear embeddings.

# 最大方差展开（MVU）[3][4]

## 原理

MVU的理念是简单的认为，*一个好的对数据结构的表示，应该保持输入数据中邻近点之间的距离在输出后保持不变*（k近邻距离不变）。再借鉴PCA的方式，在满足"k近邻距离不变"的条件下，进行最大方差展开。这一过程可以形象地理解为，首先将每个点和它周围的k个点用棍子连接起来，然后在保持棍子不断裂的情况下，尽可能地把这些点压平摊开。由于数据在高维空间中是曲折的，而在低维空间中是展平的，这一过程实际上就是在保持结构信息的基础上（k近邻距离不变）进行维度约减的过程。

将上述想法写成数学公式即为：

仍记

$$\eta_{ij} = \begin{cases} 1, & \text{if } x_i, x_j \text{ 是} k\text{近邻} \\ 0, & \text{if } x_i, x_j \text{ 不是} k\text{近邻} \end{cases}$$

则我们的目标为

$$\begin{aligned} \max_{i,j} \quad & \sum_{ij} \|y_i - y_j\|^2 && (3.0) \\ s.t. \quad & \|y_i - y_j\|^2 = \|x_i - x_j\|^2, \forall \eta_{ij} = 1 && (3.1) \\ & \sum_i y_i = 0 && (3.2) \end{aligned} \quad (\text{P1})$$

其中优化目标(3.0)为最大化方差，约束(3.1)为k近邻距离不变。约束(3.2)是一个中心化条件，表示我们希望最终获得的 $\{y_i\}$ 是均值为0的，因为对向量整体做平移是不会改变方差和距离的，因此加入这个条件可以使我们期待解的唯一性（不考虑方向旋转的话）。而且后面我们会看到，条件(3.2)对优化目标的进一步简化很有帮助。

优化问题(P1)的确是我们所希望求解的对象。但是(P1)是一个*非凸问题*，这意味着我们难以找到该问题的全局最小解。因此MVU对问题(P1)进行了数学上的处理，将问题(P1)转化为了一个等价的凸问题(P2)。

$$
\begin{aligned}
\max \quad & tr(K) && (4.0)\\
s.t. \quad & K_{ii}-2K_{ij}+K_{jj}=\|x_i-x_j\|^2, \forall \eta_{ij}=1 && (4.1)\\
& \sum_{ij}K_{ij}=0 && (4.2)\\
& K\geq 0 && (4.3)
\end{aligned}
\tag{P2}
$$

其中方阵 $K$ 定义为 $K_{ij}=\langle y_i,y_j\rangle$。于是 $K$ 是一个半正定矩阵，即条件(4.3)；也容易看出条件(4.1)等价于条件(3.1) 。至于条件(4.2)，注意到

$$
\sum_{ij}K_{ij}=\sum_{ij}\langle y_i,y_j\rangle=\sum_{i}\left\langle y_i,\sum_j y_j\right\rangle=0
$$

说明(4.2)是一个弱于(3.2)的条件。由于条件(3.2)本身就是为了缩减解空间大小而加入的附加条件，因此条件(4.2)也是如此，它不会改变问题(P2)的最优值，只是减少了符合条件的解的数量。那为什么要加入这一条件呢，原因是：

$$
\sum_{ij}\|y_i-y_j\|^2=\sum_{ij}(K_{ii}-2K_{ij}+K_{jj})=2tr(K)-2\sum_{ij}K_{ij}
$$

故加入条件(4.2)是为了使得目标函数(3.0)等价于形式简单的目标函数(4.0)。因此，综上所述，我们可以得到(P2)与(P1)等价这一结论。但不同于(P1)，(P2)是一类经典的凸优化问题，叫做**半定优化**（semidefinite program, SDP）。我们可以在多项式时间内求出其全局最小解。

在求解问题(P2)后，MVU还需要通过Gram矩阵 $K$ 计算向量组 $Y=(y_i,\dots,y_n)_{r\times n}$ 。通过奇异值分解，我们可以知道 $Y^TY=K_r=Q_rD_rQ_r^T$ ，其中 $D_r=diag\{\lambda_1,\dots,\lambda_r\}$。此时如果我们不进行降维，即取 $r=d$ 时，$K=Q_nD_nQ_n^T$ 即 $K$ 的正交相似分解。由上课所学的知识，此时对矩阵 $K$ 的最佳秩 $r$ 近似 $K_r$ 为在分解式中选取其最大的r个奇异值和对应的奇异向量所得，而 $Y=\sqrt{D_r}Q_r^T$。

故综上，MVU算法的具体流程为：

> Step1. 选取超参数k，求解SDP问题(P1)，得到矩阵K
>
> Step2. 选取超参数r，对K进行奇异值分解，得到r维空间中的向量组Y

## 实验

在文章[3]中，作者使用了3个可视化性较强的实验来体现MVU算法的效果。

第一个实验的数据对象是从不同角度观察一个茶壶所得到的一系列图片。每个图像有 $76\times 101$ RGB像素，因此这个实验中的数据维度是 $76*101*3=23028$ 维。但通过MVU算法降维到2维，我们可以容易地观察到MVU的结果可以在原空间排列成一个圆周（见[3]中Figure 3），反映了这一组数据中最有价值的信息其实是茶壶的放置角度。

第二个实验的数据对象是同一个人不同表情的1960张灰度图像。每个图像有 $28\times 20$ 像素，因此这个实验中的数据维度是 $28*20=560$ 维。通过MVU算法降低到4维后，可视化的结果发现MVU选取的4个自由度是表情的左移、右移、微笑、皱纹（见[3]中Figure 4），非常接近人们判断表情的方式。

第三个实验是单词的2维表示，MVU可以将60000维的单词矢量急剧缩小到2维但仍然保留许多语义关系（见[3]中Figure 5）。

下表是在保留95%数据方差情况下分别使用线性降维方法PCA和非线性降维方法MVU所需最小维数的对比，可以发现MVU在所有数据集上都比PCA的降维效果明显更好。

| | teapots | faces | words |
|---|---|---|---|
| initial | 23028 | 560 | 60000 |
| linear | 59 | 80 | 23 |
| nonlinear | 2 | 4 | 6 |

# 总结

LLE算法是较早出现的流形学习算法，它的优势有参数数量少（只有近邻数 $k$ 和 目标维数 $r$ ），且求解的优化问题简单（二次优化）等。LLE的思想体现了流形学习中的一个重要原理：*对数据中覆叠的局部近邻间的关系进行分析可以提供数据全局几何结构的信息*。这一从局部到全局的思想也与流形这一数学概念本身一脉相承。

MVU算法的灵感来源于*核方法*。核方法也是一种能够实现非线性降维的方法。其基本思路是通过核映射这一非线性映射将原始数据映到更高维的隐空间中，再在隐空间使用PCA等线性方法降维，以此达到非线性降维的效果。但核方法的最大问题在于第一步进行映射的核函数难以选择，依赖于人类经验。从这一角度看，MVU所计算的矩阵K就相当于核方法中的核矩阵，相当于MVU从数据本身的信息中学习适当的核映射，从而保证了其优秀的降维效果。在文章[4]中作者详细介绍了MVU算法的设计思路，如条件(3.1)中还蕴含了作者将局部等距映射的思路扩展到离散数据中的想法，在此不再赘述。MVU继承了LLE算法的许多优点（参数少、只需求解凸问题等），但缺点是SDP问题的求解相对于二次优化来说复杂度还是提高了许多，当样本数量N很大时效率不够高。

# 参考文献

[1] https://zh.wikipedia.org/wiki/%E7%BB%B4%E6%95%B0%E7%81%BE%E9%9A%BE#%E4%BC%98%E5%8C%96

[2] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by Locally Linear Embedding. Science, 290(5500):2323–2326, 2000.

[3] K. Q. Weinberger and L. K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In AAAI, pages 1683–1686, 2006.

[4] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.