Subspace Clustering: A Brief Overview

Zhong Zhang

Data Mining Lab, Big Data Research Center, UESTC
Email: zhangzhong0512@163.com
http: //dm.uestc.edu.cn/

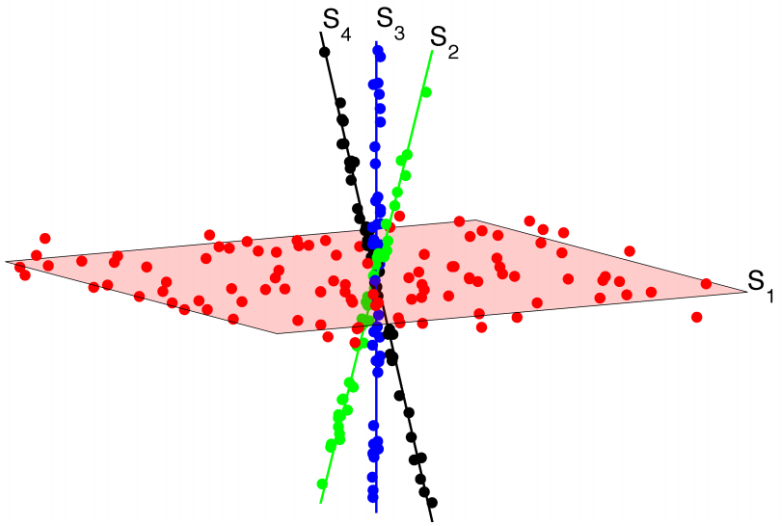
Outline

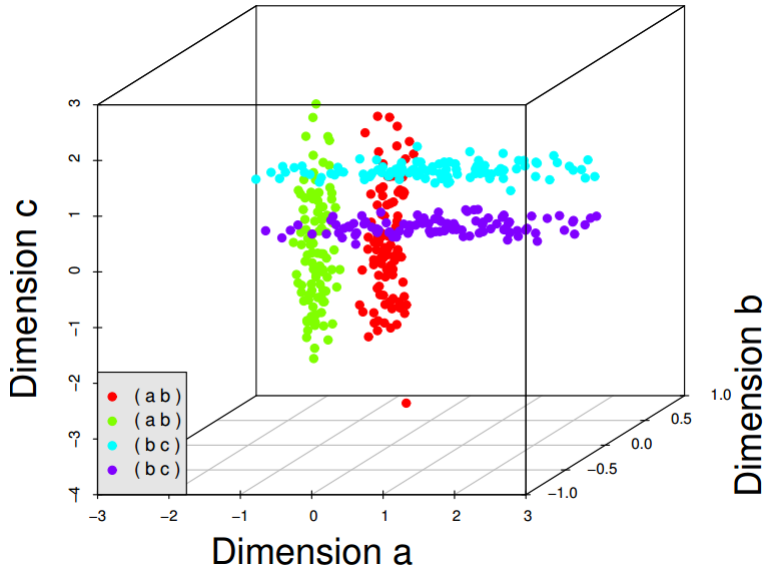
- Motivation and Application
- A Cursory Category
- CLIQUE Algorithm
- Sparse Representation
- Factorization-based methods

Motivation and Application

- What is subspace clustering?

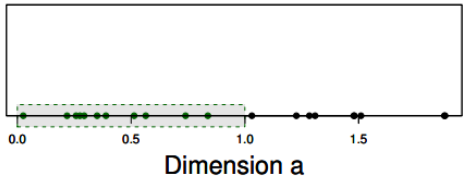
现实中的一些高维数据，不同的属性维度生成机制是不同的，代表了不同的子空间。高维数据则由低维子空间嵌入组合而成。子空间聚类希望能够从高维数据中辨别出子空间，在子空间上实现更好的聚类效果。



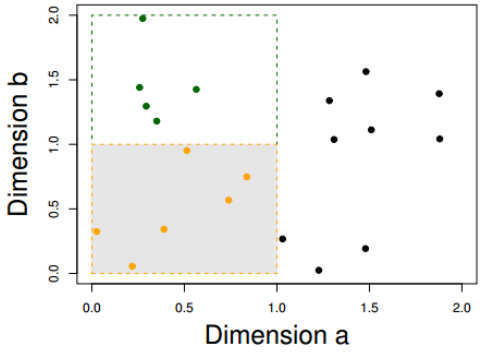


Motivation and Application

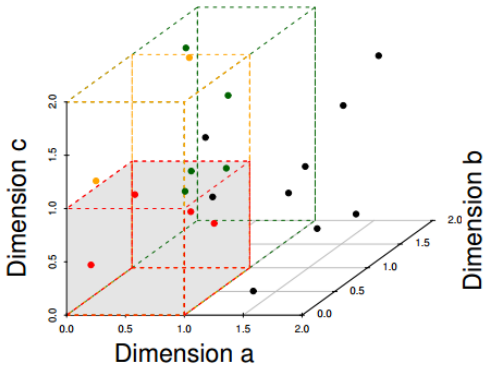
- Why subspace clustering?
  - ➢ Noise robust
  - ➢ Curse of dimensionality



(a) 11 Objects in One Unit Bin



(b) 6 Objects in One Unit Bin



(c) 4 Objects in One Unit Bin

  - ➢ Interpretability of results

Motivation and Application

- Application
  - Computer Vision

    facial recognition, image segmentation, motion detection…
  - Biological Data

    gene clustering, disease classification…
  - Text Categorization
  - Data Fusion and Multi-Source Learning
  - …

A Cursory Category

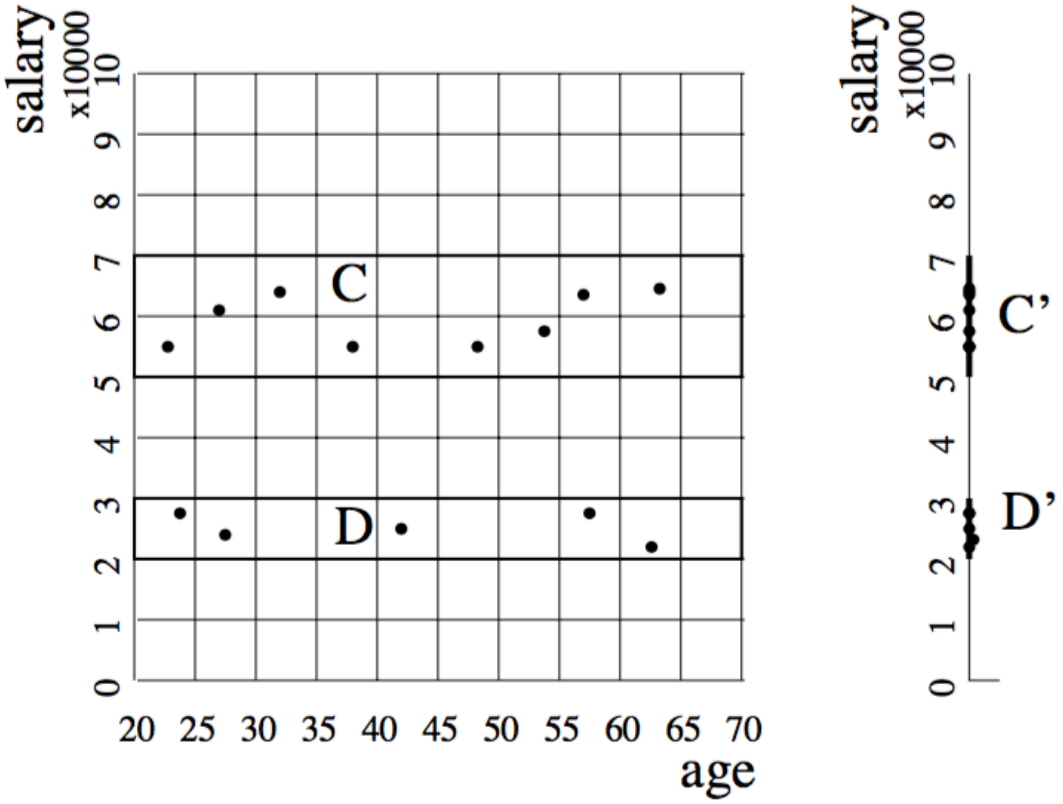
- Existing works on subspace clustering can be divided into seven main categories:
  - **Iterative**: K-subspaces[1], CLIQUE[2]
  - **Statistical approaches**: MPPCA[3], MSL[4]
  - **Factorization-based methods**
  - **Spectral clustering methods**
  - **Algebraic methods**: GPCA[5, 6]
  - **Information-theoretic approaches**: ALC[7]
  - **Sparse representation**

CLIQUE

- CLIQUE algorithm combines density and grid based clustering and uses an APRIORI style technique to find clusterable subspaces

No 2-dimensional unit is dense and no clusters in the original data space.

There are three 2-dimen-sional dense units, two of them are connected.



CLIQUE

- CLIQUE consists of the following steps:
  1. Identification of subspace that contain clusters
  2. Identification of clusters
  3. Generation of minimal description for the clusters

CLIQUE

1. 找出包含密集的子空间

对n维数据空间进行划分，划分为互不相交的矩形单元，并识别其中的密集单元

空间划分：$A=\{A_1, A_2, \ldots, A_d\}$是一组有界域，$S=A_1 \times A_2 \times \ldots \times A_d$是d维空间，$A_1, A_2, \ldots, A_d$视为$S$的属性。

d维数据点集$V=\{v_1, v_2, \ldots, v_m\}$，其中$v_i=\{v_{i1}, v_{i2}, \ldots, v_{id}\}$，第j个分量$v_{ij}$是从$A_j$中抽出。

输入参数ξ，将空间S的每一维分成相同的ξ个区间，从而将整个空间划分为有限的ξ^d个网格，表示为$\{u_1, u_2, \ldots, u_d\}$，其中$u_i=[l_i, h_i)$。

稠密单元：给定稠密阈值τ，若一个单元格中点的密度大于阈值，则称该单元是稠密的。

CLIQUE

- CLIQUE使用一个“自下而上”类似于Apriori的性质来寻找稠密单元：

  如果一个k维单元是密集的，那么他在k-1维空间上的投影也是密集的。

  给定一个k维的候选密集单元，如果检查它的k-1维投影单元，发现任何一个不适密集的，那么该k维单元也不会是密集的。

  算法：首先遍历数据寻找一维密集单元，然后通过使用候选集生成算法从k-1维密集单元得到k维候选密集单元，当没有新的候选集生成时，算法终止。（类似从频繁k-1项集连接成候选频繁k项集，最终得到频繁k项集）

CLIQUE

- 使用最小描述长度MDL剪枝

使用类似Apriori的算法可以减少需要验证的密集单元数，但对于高维数据来看，空间维度的增长依然会造成密集单元的飞速增长。CLIQUE使用了一个基于MDL的剪枝策略。

令$x_{s_j} = \sum_{u_i \in s_j} count(u_i)$，其中$count(u_i)$是单元$u_i$中包含点的数目，$x_{s_j}$称为子空间S$j$的覆盖。覆盖大的被选中，剩余的被剪枝。把子空间按照覆盖进行降序排列，我们希望将子空间分成选择集合I和剪枝集合P。

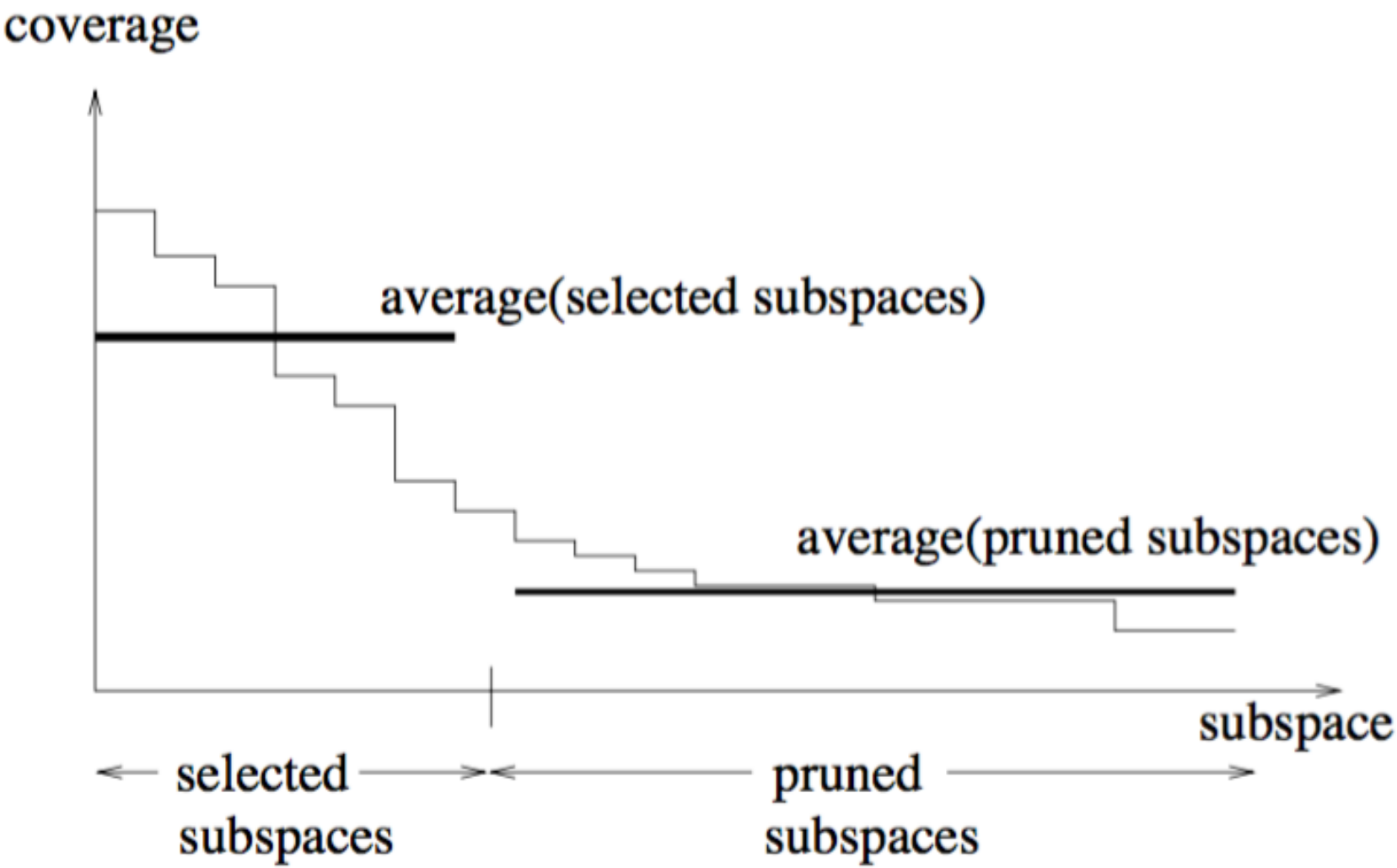
通过让编码长度最小，选择剪枝点i。

$$CL(i) = \log_2(\mu_I(i)) + \sum_{1 \le j \le i} \log_2(|x_{s_j} - \mu_I(i)|) + \log_2(\mu_P(i)) + \sum_{i+1 \le j \le n} \log_2(|x_{s_j} - \mu_P(i)|)$$

$$\mu_I(i) = \lceil (\sum_{1 \le j \le i} x_{s_j})/i \rceil$$

$$\mu_P(i) = \lceil (\sum_{i+1 \le j \le n} x_{s_j})/(n-i) \rceil$$

CLIQUE


coverage
average(selected subspaces)
average(pruned subspaces)
subspace
selected subspaces
pruned subspaces

CLIQUE

2. 识别聚类

可运用深度优先算法来发现空间中的聚类，算法描述如下：

输入：在同一个k维空间S中的密集单元集合D

输出：D的一个分割D1，D2，…，Dq，分割中的密集单元是相互连接的，或者说是连通的，同时没有任何两个不同单元ui∈Di， uj∈Dj（i≠j）是连接的。

$\mathbf{dfs}(u, n)$
$u.num = n$
for $(j = 1; j < k; j{+}{+})$ **do begin**
// examine the left neighbor of u in dimension a_j
$u^l = \{[l_1, h_1), \ldots, [(l_j^l), (h_j^l)), \ldots, [l_k, h_k)\}$
if (u^l is dense) **and** ($u^l.num$ is undefined)
$\mathbf{dfs}(u^l, n)$
// examine the right neighbor of u in dimension a_j
$u^r = \{[l_1, h_1), \ldots, [(l_j^r), (h_j^r)), \ldots, [l_k, h_k)\}$
if (u^r is dense) **and** ($u^r.num$ is undefined)
$\mathbf{dfs}(u^r, n)$
end

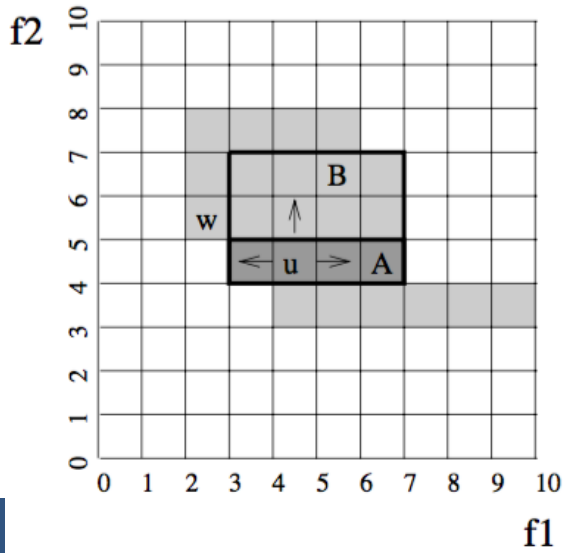
CLIQUE

3. 为每个簇生成最小化的描述

CLIQUE使用每个子空间中的稠密单元来装配可能具有任意形状的簇，确定其覆盖相连的密集单元最大区域。其采用了一种简单的贪心方法：

从一个任意稠密单元开始，找出覆盖该单元的最大矩形区域，然后在尚未被覆盖的剩余的稠密单元上继续这一过程。当所有稠密单元都被覆盖时，贪心方法终止。

确定最小的覆盖区域，从已有的最大覆盖中，移走数目最小的多余最大空间区域，直到没有多余的最大密集区域为止。



CLIQUE

算法总结：

第1步，根据ε的值，把原多维空间数据对象的每一维属性划分成相等的区间

第2步，n=1，经过划分得到的所有矩形单元都为候选稠密单元；

第3步，扫描原数据空间，记录n维子空间中落在每个候选稠密单元的数据对象个数；

第4步，根据设置的阈值找出n维子空间中的密集单元；

第5步，用MDL算法修剪已有的密集子空间；

第6步，由n维子空间中的稠密单元集，找出n+1维子空间中的候选稠密单元集合，若n+1维子空间中的候选稠密单元集不为空，则跳转到第3步；

第7步，用深度优先探索算法找出n维空间中的聚类；

第8步，运用贪婪算法找到覆盖每个子聚类的最大密集；

第9步，确定每个聚类的最小覆盖；

第10步，将聚类信息结果保存到文件中

Sparse Representation

Algorithm 1 Sparse Subspace Clustering (SSC)

Input: A data set $\mathcal{X}$ arranged as columns of $\boldsymbol{X} \in \mathbb{R}^{n \times N}$.

1. Solve (the optimization variable is the $N \times N$ matrix $\boldsymbol{Z}$)

$$\begin{array}{ll} \text{minimize} & \|\boldsymbol{Z}\|_{\ell_1} \\ \text{subject to} & \boldsymbol{X}\boldsymbol{Z} = \boldsymbol{X} \\ & \operatorname{diag}(\boldsymbol{Z}) = \boldsymbol{0}. \end{array}$$

2. Form the affinity graph G with nodes representing the N data points and edge weights given by $\boldsymbol{W} = |\boldsymbol{Z}| + |\boldsymbol{Z}|^T$.
3. Sort the eigenvalues $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_N$ of the normalized Laplacian of G in descending order, and set

$$\hat{L} = N - \underset{i=1,\ldots,N-1}{\arg\max} \ (\sigma_i - \sigma_{i+1}).$$

4. Apply a spectral clustering technique to the affinity graph using $\hat{L}$ as the estimated number of clusters.

Output: Partition $\mathcal{X}_1, \ldots, \mathcal{X}_{\hat{L}}$.

Sparse Representation

Why does sparse representation recover subspace ?

- Union of disjoint model

Ref: Elhamifar, Ehsan, and Rene Vidal. "Sparse subspace clustering: Algorithm, theory, and applications." *IEEE transactions on pattern analysis and machine intelligence* 35.11 (2013): 2765-2781.

- Independent subspace model

Theorem 1: *Consider a collection of data points drawn from n independent subspaces $\{\mathcal{S}_i\}_{i=1}^n$ of dimensions $\{d_i\}_{i=1}^n$. Let $\boldsymbol{Y}_i$ denote N_i data points in $\mathcal{S}_i$, where $\operatorname{rank}(\boldsymbol{Y}_i) = d_i$, and let $\boldsymbol{Y}_{-i}$ denote data points in all subspaces except $\mathcal{S}_i$. Then, for every $\mathcal{S}_i$ and every nonzero $\boldsymbol{y}$ in $\mathcal{S}_i$, the ℓ_q-minimization program*

$$\begin{bmatrix} \boldsymbol{c}^* \\ \boldsymbol{c}_-^* \end{bmatrix} = \operatorname{argmin} \left\| \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{c}_- \end{bmatrix} \right\|_q \quad \text{s.t.} \quad \boldsymbol{y} = [\boldsymbol{Y}_i \;\; \boldsymbol{Y}_{-i}] \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{c}_- \end{bmatrix}, \quad (46)$$

for $q < \infty$, recovers a subspace-sparse representation, i.e., $\boldsymbol{c}^* \neq \boldsymbol{0}$ and $\boldsymbol{c}_-^* = \boldsymbol{0}$.

Sparse Representation

Proof:

Assume $c_{-}^{*} \neq 0$. Then we can write $\boldsymbol{y} = \boldsymbol{Y}_i \boldsymbol{c}^* + \boldsymbol{Y}_{-i} \boldsymbol{c}_{-}^*$.

Since y is a data point in subspace Si , there exists a c such that $\boldsymbol{y} = \boldsymbol{Y}_i \boldsymbol{c}$

So we get $\boldsymbol{Y}_i(\boldsymbol{c} - \boldsymbol{c}^*) = \boldsymbol{Y}_{-i} \boldsymbol{c}_{-}^*$.

Note that left hand side of equation corresponds to a point in the subspace Si, while the right hand side is not.

By the independence assumption, the two subspaces Si and S_i are independent hence disjoint and intersect only at the origin.

Thus, we must have $\boldsymbol{Y}_{-i} \boldsymbol{c}_{-}^* = \boldsymbol{0}$ and $\boldsymbol{y} = \boldsymbol{Y}_i \boldsymbol{c}^*$, so $[\boldsymbol{c}^{*\top} \quad \boldsymbol{0}^\top]^\top$ is a feasible solution.

Finally, assumption $c_{-}^{*} \neq 0$ contradicts the solution, thus we must have $c^* \neq 0$ and $c_{-}^{*} = 0$

Sparse Representation

稀疏的本质是什么？低秩呢？

Sparse Representation

近端梯度下降（Proximal Gradient Descent）解L1正则化问题

假设我们要求解以下的最小化问题：

$$\min_x f(x)$$

最简单方法：求导用梯度下降！

但由于加入正则项后，导致x在某些点不可导，GD没法直接推广，因此需要一些技巧。

Sparse Representation

如果$\nabla f(x)$满足L-Lipschitz，即：

$$\|\nabla f(x') - \nabla f(x)\| \leq L\|x' - x\|$$

那么我们可以在点x_k附近把$f(x)$通过二阶泰勒近似为：

$$\hat{f}(x, x_k) \doteq f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{L}{2}\|x - x_k\|^2。$$

重新排列得到：

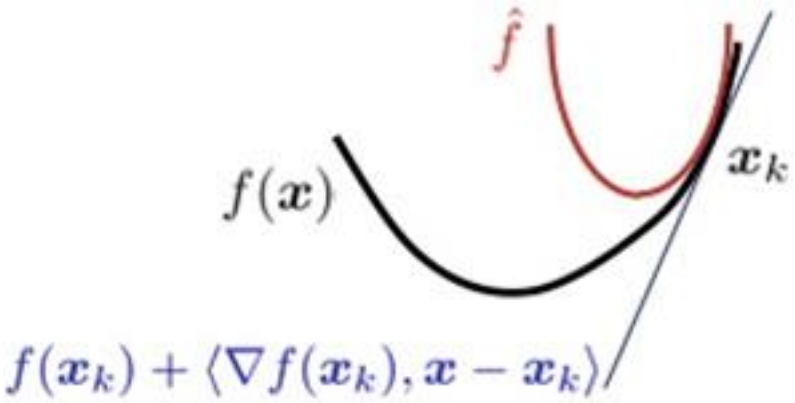
$$\begin{aligned}\hat{f}(x, x_k) &\doteq f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{L}{2}\|x - x_k\|^2 \\ &= \frac{L}{2}\|x - (x_k - \frac{1}{L}\nabla f(x_k))\|_2^2 + \varphi(x_k)。\end{aligned}$$

$\varphi(x_k)$是与x无关的常数可忽略，显然 $\hat{f}(x, x_k)$ 的最小值在

$$x_{k+1} = x_k - \frac{1}{L}\nabla f(x_k)$$

Sparse Representation

从这个角度看的话，GD的每次迭代是在最小化原目标函数的一个二次近似函数 $\hat{f}(x, x_k)$



在很多最小化问题中，我们往往会加入非光滑的惩罚项$g(x)$，比如常见的$L1$惩罚：$g(x)=||x||_1$。这个时候，GD就不好直接推广了。但上面的二次近似思想却可以推广到这种情况：

$$
\begin{aligned}
x_{k+1} &= \arg\min_x \hat{F}(x, x_k) \\
&= \arg\min_x \frac{L}{2} \|x - (x_k - \frac{1}{L} \nabla f(x_k))\|_2^2 + g(x)
\end{aligned}
$$

Sparse Representation

令 z= $(x_k - \frac{1}{L}\nabla f(x_k))$

则： $prox_{ug}(z) = \arg\min_x \frac{1}{2}\|x - z\|_2^2 + \mu g(x)$。

当$g(x)=||x||_1$时，易得：

$$prox_{ug}(z) = sign(z)\max\{|z| - \mu, 0\}.$$

Factorization-based methods

- Overview of NMF

  Let $\mathbf{X} \in R^{M \times N}$, each column represents a data point, each row represents one attribute.

  NMF aims to find two non-negative matrix factors:

$$X \approx UV^T$$

  Optimization problem:

$$\min_{U,V} ||X - UV^T||_F^2, \ s.t. \ U \geq 0, V \geq 0$$

Factorization-based methods

- Multi-View NMF

  Let $\{\mathbf{X}^{(1)},\mathbf{X}^{(2)},...,\mathbf{X}^{(nv)}\}$ denote the data of all the views, for $\mathbf{X}^{(v)}$ we have factorizations:

$$X^{(v)} \approx U^{(v)}(V^{(v)})^T$$

  To learn the joint consensus matrix V^* , we have objective function:

$$\sum_{v=1}^{n_v} \|X^{(v)} - U^{(v)}(V^{(v)})^T\|_F^2 + \sum_{v=1}^{n_v} \lambda_v \|V^{(v)} - V^*\|_F^2$$

$$s.t.\ \forall 1 \le k \le K, ||U_{\cdot,k}^{(v)}||_1 = 1 \text{ and } U^{(v)}, V^{(v)}, V^* \ge 0$$

Factorization-based methods

- JSNMF1

  Objective function:

$$\min_{\boldsymbol{W},\boldsymbol{H}} (1-\alpha)\|\boldsymbol{X}-\boldsymbol{W}\boldsymbol{H}\|_F^2 + (\alpha)\|\boldsymbol{Y}-\boldsymbol{V}\boldsymbol{H}\|_F^2$$

$$s.t \quad \boldsymbol{V},\boldsymbol{W},\boldsymbol{H} \succcurlyeq 0.$$

  Solution:

$$\boldsymbol{W} = \boldsymbol{W} \odot \frac{\boldsymbol{X}\boldsymbol{H}^{\boldsymbol{T}}}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^{\boldsymbol{T}}} \qquad \boldsymbol{H} = \boldsymbol{H} \odot \frac{(1-\alpha)\boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{X} + (\alpha)\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{Y}}{((1-\alpha)\boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{W} + \alpha\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{V})\boldsymbol{H}}$$

$$\boldsymbol{V} = \boldsymbol{V} \odot \frac{\boldsymbol{Y}\boldsymbol{H}^{\boldsymbol{T}}}{\boldsymbol{V}\boldsymbol{H}\boldsymbol{H}^{\boldsymbol{T}}}$$

Factorization-based methods

- JSNMF2

$$\boldsymbol{X} \approx \underbrace{[\mathbf{W} \mid \boldsymbol{U}]}_{\mathbf{F}} \mathbf{H} = \mathbf{F}\mathbf{H} \qquad \boldsymbol{Y} \approx \underbrace{[\mathbf{W} \mid \boldsymbol{V}]}_{\mathbf{G}} \mathbf{L} = \mathbf{G}\mathbf{L}$$

W is a M × K matrix whose columns span the common subspace

U and V represent the remaining subspaces having di- mension of M × (R1 − K) and M × (R2 − K)respectively

Objective function:

$$\min D \triangleq \frac{1}{2}\left\{\|\boldsymbol{X} - [\mathbf{W} \mid \boldsymbol{U}]\mathbf{H}\|_F^2 + \lambda \|\boldsymbol{Y} - [\mathbf{W} \mid \boldsymbol{V}]\mathbf{L}\|_F^2\right\}$$

$$\text{subject to } \mathbf{W}, \boldsymbol{U}, \boldsymbol{V}, \mathbf{H}, \mathbf{L} \geq 0$$

Factorization-based methods

- JSNMF3

$$W_i = \begin{bmatrix} W_{i,c} & W_{i,d} \end{bmatrix} \qquad H_i = \begin{bmatrix} H_{i,c} & H_{i,d} \end{bmatrix}$$

Objective function:

$$\min_{W_1,H_1,W_2,H_2 \geq 0} \frac{1}{n_1} \left\| X_1 - W_1 H_1^{\mathsf{T}} \right\|_F^2 + \frac{1}{n_2} \left\| X_2 - W_2 H_2^{\mathsf{T}} \right\|_F^2 + \alpha \left\| W_{1,c} - W_{2,c} \right\|_F^2 + \beta \left\| W_{1,d}^{\mathsf{T}} W_{2,d} \right\|_{1,1}$$

$$\text{subject to } \|(W_1)_{\cdot l}\|_2 = 1, \|(W_2)_{\cdot l}\|_2 = 1 \text{ for } l = 1, \cdots, k.$$

Factorization-based methods

矩阵分解是用来寻找多个数据源共同子空间的有力工具，常用套路是将矩阵分解为两个基矩阵与系数矩阵的乘积，并且多个数据源共享基矩阵或系数矩阵，再添加相应约束、正则项等。

优点是表示简单，特定情境下分解的矩阵有对应解释，噪声不敏感。

缺点也很明显，求解很困难，涉及矩阵的优化求解，通常难以得到最优解，通常是固定某个变量，再优化另一个变量。