参赛密码 _____

（由组委会填写）

第十二届“中关村青联杯”全国研究生数学建模竞赛

学　　校	重庆理工大学
参赛队号	11660017
队员姓名	1.冉小华
	2.胡猛
	3.陈华良

参赛密码 _____

（由组委会填写）

第十二届“中关村青联杯”全国研究生数学建模竞赛

题 目 数据的多流形结构分析

摘 要：

针对多流形结构的聚类问题，本文提出稀疏子空间聚类模型以及改进优化算法对常规的多流形结构数据进行聚类，并用谱多流形聚类模型（SMMC）对混合流形结构数据进行聚类。

问题一：需将 200 个高维数据分为两类，本文首先建立稀疏子空间聚类模型，在此运用交替方向法求出解稀疏系数，然后利用高维数据的稀疏表示系数构造相似度矩阵，最后利用谱聚类方法，得到数据的聚类结果：1-40 和 141-200 之间的样本点划分到一类，41-140 之间的样本点为另一类。

问题二：针对子问题(a)两条交点不在原点且相互垂直的两条直线进行分类，我们应用稀疏子空间聚类模型并得到较好的分类结果；(b)一个平面和两条直线分成三类和(c)两条不相交的二次曲线分成两类情形，本文建立拉普拉斯特征映射模型，利用拉普拉斯嵌入描述数据的局部几何结构关系进行聚类，最终的分类结果都比较理想；(d)两条相交的螺旋线分成两类，本文采用谱多流形聚类方法进行分类。首先从整个数据中分离出不同的连通或可分离子集，然后进一步将相交的子集分割为相交区域和非相交区域，最后用谱多流形聚类方法得聚类结果比较准确。

问题三：针对子问题(a)对十字上的点位置信息进行分类，本文运用谱多流形聚类方法进行聚类，得出具有较好地分类效果；(b)对已提取特征点后的轨迹进行运动分割。在此对稀疏子空间聚类模型中的交替方向法进行改进。调整交替方向法的惩罚参数，将稀疏子空间聚类应用到视频运动分割中，从而得到特征点轨迹的分类结果：1-138 之间的特征点划为第 2 类，139-214 之间的特征点划为第 3 类，215-297 之间的特征点划为第 1 类；(c)将在不同光照下的人脸图形分成两类，本文同样稀疏子空间聚类模型中的交替方向法进行改进，研究线性化的自适应线性交替方向法。同时把稀疏子空间聚类应用到人脸分类中进行分析，得出分类结果：1-5 幅图和 11-15 幅图都被划分到第 1 类，而 6-10 幅图和 16-20 幅图。被划分到第

2 类,聚类准确率 100%。

问题四：针对子问题(a)对圆台的点分成三类，由于在运用原始数据分类过程中发现内存溢出，所以本文首先对原始数据采样，然后对采样点进行谱多流形聚类，最后运用邻近分类算法对剩下的数据进行分类，得到理想的分类结果；(b)问题，由于机器工件外部边缘轮廓图较为复杂，而且噪音点较多，本文建立谱多流形聚类模型进行分类，最后分别将其划分为二、三、四、五类，实验结果表面可以有效地将轮廓线中不同的直线和圆弧区分开。

关键词：稀疏子空间聚类；交替方向法；拉普拉斯特征映射；谱多流形聚类

一、问题重述

我们已经进入了一个信息爆炸的时代，海量的数据不断产生，迫切需要对这些大数据进行有效的分析，以至数据的分析和处理方法成为了诸多问题成功解决的关键，涌现出了大量的数据分析方法。几何结构分析是进行数据处理的重要基础,已经被广泛应用在人脸识别、手写体数字识别、图像分类、等模式识别和数据分类问题，以及图象分割、运动分割等计算机视觉问题（人脸识别、图像分类、运动分割等实例见下文）中。更一般地，对于高维数据的相关性分析、聚类分析等基本问题，结构分析也格外重要。

大量的数据降维方法被用来挖掘数据集的低维线性子空间结构，这类方法假设数据集采样于一个线性的欧氏空间。针对单一子空间结构假设的后续讨论主要是两个方面，首先是从线性到非线性的扩展，主要的代表性工作包括流形（流形是局部具有欧氏空间性质的空间，欧氏空间就是流形最简单的实例）学习等。其次是流形或子空间从一个到多个的扩展，即假设数据集采样于多个欧氏空间的混合。子空间聚类（是将数据按某种方式分类到其所属的子空间的过程。通过子空间聚类，可以将来自同一子空间中的数据归为一类，由同类数据又可以提取对应子空间的相关性质。

本几何结构分析问题中假设数据分布在多个维数不等的流形上，其特殊情况是数据分布在多个线性子空间上。请按照文献中的方法或以文献中的方法为基础创新新的方法完成以下问题：

1. 当子空间独立时，子空间聚类问题相对容易。附件一中 1.mat 中有一组高维数据，它采样于两个独立的子空间。请将该组数据分成两类。

2. 处理附件二中四个低维空间中的子空间聚类问题和多流形聚类问题：（1）将两条交点不在原点且互相垂直的两条直线，请将其分为两类；（2）对不满足独立子空间的关系时，将一个平面和两条直线，请将其分为三类；(3)将两条不相交的二次曲线划分为两类。(4)将两条相交的螺旋线划分为两类。

3.解决以下三个实际应用中的子空间聚类问题。

（a）受实际条件的制约，在工业测量中往往需要非接触测量的方式，视觉重建是一类重要的非接触测量方法。特征提取是视觉重建的一个关键环节，如（a）所示，其中十字便是特征提取环节中处理得到的，十字上的点的位置信息已经提取出来，为了确定十字的中心位置，一个可行的方法是先将十字中的点按照 “横”和“竖”分两类。请使用适当的方法将（a）中十字上的点分成两类。

（b）运动分割是将视频中有着不同运动的物体分开，是动态场景的理解和重构中是不可缺少的一步。基于特征点轨迹的方法是重要的一类运动分割方法，该方法首先利用标准的追踪方法提取视频中不同运动物体的特征点轨迹，之后把场景中不同运动对应的不同特征点轨迹分割出来。已经有文献指出同一运动的特征点轨迹在同一个线性流形上。（b）显示了视频中的一帧，有三个不同运动的特征点轨迹被提取出来保存在了 3b.mat 文件中，请使用适当方法将这些特征点轨迹分成三类。

（c）3c.mat 中的数据为两个人在不同光照下的人脸图像共 20 幅（X 变量的每一列为拉成向量的一幅人脸图像），请将这 20 幅图像分成两类。

4. 作答如下两个实际应用中的多流形聚类问题

实际应用一:对圆台的点云，请将点按照其所在的面分开(即圆台按照圆台的顶、底、侧面分成三类)。

实际应用二：机器工件外部边缘轮廓的图像，请将轮廓线中不同的直线和圆弧分类，类数自定。

二、基本假设

1、采样密度稀疏时，所寻找的局部邻域结构时产生的偏差在可控范围之内。
2、在子空间分割中，假设高维数据分布于多个低维子空间的并。
3、欧式空间的子集是一个凸集。
4、假设数据分布在多个维数不等的流形上。
5、当数据的低维特征嵌入在高维的特征空间时,同处在小的局部邻域内样本数据会具有相似的特性,即类别标签属性也是相同。

三、符号说明

符号	含义
C	样本 y_i 基于数据集 Y 的稀疏
$\|x\|_0$	向量 x 的 l_0-（伪）范数， x 中非零元个数
$\|x\|_F$	矩阵 X 的 l_1-范数， $\|x\|_F=\sqrt{\sum_i \sum_j \|X_{ij}\|^2}$
A	表示引入的辅助矩阵 $A\in R^{N\times N}$
λ_z	表示平衡两项的权重
ρ	惩罚因子
I	N 阶单位矩阵
Δ	表示 $\Delta\in R^{N\times N}$ 拉格朗日乘子矩阵
W	由数据集构造的相似矩阵
D	$\mathrm{D}=W^{N\times N}$ 对角矩阵
L	图拉普拉斯矩阵， $\mathrm{L}=\mathrm{I}-\mathrm{D}^{-1/2}\mathrm{WD}^{1/2}$
$\langle\bullet\rangle$	表示标准内积
$Knn(x)$	表示 x 的 K 个近邻数据点
ε_m	表示噪声

四、问题 1 分析及模型建立与求解

4.1 问题分析

近年来，由于高维数据在很多领域普遍存在，谱聚类的方法已成为高维数据聚类的主流方法。由于稀疏子空间聚类是实现高维数据集聚类的一种有效途径，在此本文将稀疏子空间聚类的方法用于解决问题一。建模思路如下：首先建立稀疏子空间聚类模型，在此运用交替方向法求出解稀疏系数，然后根据表示系数矩阵构造相似度矩阵，最后利用谱聚类方法，获得数据聚类的结果。

4.2 模型的建立与求解

4.2.1 稀疏子空间聚类模型

稀疏子空间聚类是一种基于稀疏表示的子空间聚类算法[1-3]。假设数据集为 $Y=[y_1,y_2,\cdots,y_N]\in R^{M\times N}$，对于每个数据样本而言，直接使用原始数据集作为冗余字典，得每个数据点的稀疏表示 c_i：

$$\begin{aligned}&\min_{\langle C,Z\rangle}\|C\|_0+\frac{\lambda_z}{2}\|Z\|_F^2\\&s.t.\quad Y=YC+Z\\&C_{ii}=0\quad i=1,2,\cdots N\end{aligned}\tag{4-1}$$

上式优化式是非凸的并且是一个 NP-hard 问题，常使用凸松弛方式（用 ℓ_1 范数替代原有的 ℓ_0 范数），将上述非凸优化转变如下：

$$\begin{aligned}&\min_{\langle C,Z\rangle}\|C\|_1+\frac{\lambda_z}{2}\|Z\|_F^2\\&s.t.\quad Y=YC+Z\\&C_{ii}=0\quad i=1,2,\cdots N\end{aligned}\tag{4-2}$$

求解上式得到每个数据点的稀疏表示，构造相似度矩阵 $\mathrm{W}=[w_1,w_2,\cdots,w_N]\in W^{N\times N}$。为使 W 对称，定义如下：

$$W=|C|+|C|^T\tag{4-3}$$

从另一方面说，节点 i 和节点 j 之间边的权重等于 $|c_{ij}|+|c_{ji}|$。这里 W 由 n 个独立子空间生成，每个样本 y_i 可以由属于同一个子空间的其他样本进行稀疏重构，属于不同子空间的样本的稀疏表示系数为零。

在相似度矩阵 W 上用谱聚类算法[4]进行划分。

算法 1 谱聚类算法

输 入： n 个线性子空间 $\{S_i\}_{i=1}^{n}$ 组成的数据点 $\{y_i\}_{i=1}^{N}$。

输 出：原始数据对应的聚类结果。

Step1　构造基于数据集 Y 的相似矩阵 W；

Step2　计算相似矩阵 W 的 *Laplacian* 矩阵(L)；

Step3　求出 L 的前 n 个特征值, 以及其对应的特征向量 $\{u_i\}_{i=1}^{k}$

Step4　把 k 个特征（列）向量排列在一起组成一个 $N\times k$ 的矩阵

Step5　把生产矩阵每一行看做 k 维空间中的一个向量，利用 $K-means$ 将的行量分组为 n 个聚类。

4.2.2 交替方向法求解算法

稀疏子空间模型一般传统常用内点算法 CVX 优化工具包求解，由于交替算法常常用于具有可分离结构的目标函数凸优化问题。故在此本文用交替方向法[4]对其进行求解。

首先根据（4-2）的等式约束。我们可以消除目标函数的 Z ,并引入 $A\in R^{N\times N}$，那么公式（4-2）可以重写成如下形式：

$$\min_{\langle C,Z\rangle}\|C\|_1+\frac{\lambda_z}{2}\|Z\|_F^2 \qquad (4-4)$$
$$s.t. \quad A=C$$
$$C_{ii}=0 \qquad i=1,2,\cdots N$$

引进矩阵 $\Delta\in \mathrm{R}^{N\times N}$ 拉格朗日乘子，我们可以得到增广拉格朗日函数：

$$L(C,A,\Delta)=\|C\|_1+\frac{\lambda_z}{2}\|Y-YA\|_F^2+\langle\Delta,A-C\rangle+\frac{\rho}{2}\|A-C\|_F^2 \qquad (4\text{-}5)$$

通过固定 $\left(C^k,\Delta^k\right)$，对 L 关于 A 求导可得：

$$-\lambda_z\left[Y^T\left(Y-YA^{(k+1)}\right)\right]+\Delta^k+\rho\left[A^{(k+1)}-C^k\right]=0 \qquad (4-6)$$

化简可得：

$$\left(\lambda_z Y^TY+\rho I\right)A^{(k+1)}=\lambda_z Y^TY+\rho C^k-\Delta^k \qquad (4-7)$$

一般情况下，Y^TY 为对角阵时，可以得到 $A^{(k+1)}$ 的封闭解：

$$A^{(k+1)}=\left(\lambda_z Y^TY+\rho I\right)^{-1}\cdot\left(\lambda_z Y^TY+\rho C^k-\Delta^k\right) \qquad (4-8)$$

固定 $\left(A^{(k+1)},\Delta^k\right)$，对 L 关于 C 求导得到 C^{k+1}，可以得到其封闭解：

$$C^{(k+1)}=J-diag(J) \qquad (4-9)$$

其中 $J = T_{1/\rho}\left(A^{(k+1)} + \Delta^k/\rho\right)$

这里 $T_\eta(\cdot)$ 是收缩阈值操作符，其可以定义成如下形式：

$$T_\eta(x) = \begin{cases} x-\eta, & x > \eta \\ x+\eta, & x < -\eta \\ 0, & others \end{cases} \tag{4-10}$$

这里的 x 可以是数字，向量或者一个矩阵。

当得到 (C^{k+1}, A^{k+1}) 时，可以更新拉格朗日乘子，如下：

$$\Delta^{k+1} = \Delta^k + \rho(A^{k+1} - C^{k+1}) \tag{4-11}$$

这三步不断重复执行直到收敛完成或达到设定的迭代次数。迭代停止标准为 $\left\|\mathrm{A}^{(k)} - C^{(k)}\right\|_\infty \le \varepsilon, \left\|A^{(k)} - A^{(k-1)}\right\|_\infty \le \varepsilon$ 。其中 ε 表示容错率，一般取 $\varepsilon \in \left\{10^{-3}, 10^{-4}\right\}$ 下面算法展示了交替方向法执行式子（4-2）的过程。

算法 2 稀疏求解的交替方向算法

输 入：待分类的数据集 Y

输 出：最优稀疏系数矩阵 $\mathrm{C}^* = C^k$

初始化： $k = 0$，$C^{(0)}$，$A^{(0)}$，$\Delta^{(0)}$ 为 0。

while $\left\|\mathrm{A}^{(k)} - C^{(k)}\right\|_\infty \ge \varepsilon, \left\|A^{(k)} - A^{(k-1)}\right\|_\infty \ge \varepsilon$ 做下面操作：

Step1 通过求解下列等式更新 A^{k+1}：$\left(\lambda_z Y^T Y + \rho I\right) A^{(k+1)} = \lambda_z Y^T Y + \rho C^k - \Delta^k$

Step2 更新 $C^{(k+1)}$ 如下：$C^{(k+1)} = J - diag(J)$，其中 $J = T_{1/\rho}\left(A^{(k+1)} + \Delta^k/\rho\right)$；

Step3 更新 Δ^{k+1} 如何：$\Delta^{k+1} = \Delta^k + \rho(A^{k+1} - C^{k+1})$；

Step4 $k = k + 1$；

end while

4. 2.3 实验结果与分析

考虑嵌入构造在 100 维空间中的两个独立空间的子空间 $\left\{S_i\right\}_{i=1}^2$ ，在子空间里每一个样本点都可以由其所在空间里的点线性表示。在此运用上述模型和算法将 1.mat 中高维数据进行分类。

实验环境为：Windows 7 系统和 Matlab 2012b，CPU 为 i3-2310M CPU 和 2GB 内存。通过多实验调节参数，现做如下设置：最大迭代次数为 300，$n = 2$ ，$\rho = 20$ ，$\lambda_z = 1.2$ ，$\varepsilon = 0.0001$ ，程序见附录 1,。为书写方便，故将第 1 类记为

类比标签“1”，将第 2 类记为类别标签“2”，结果如下表 1：

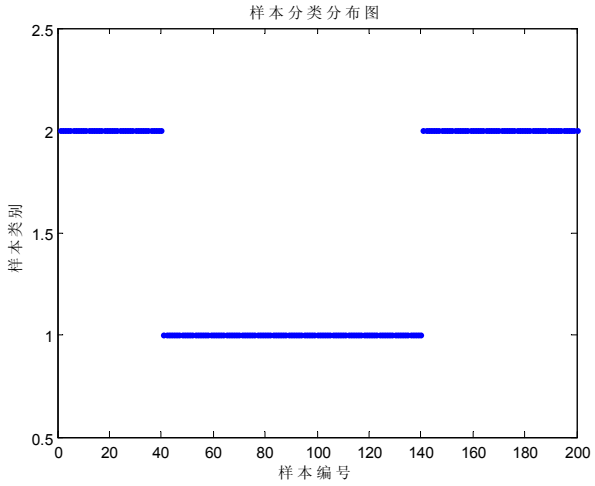


图 1. 问题一高维数据分类图

从图 1 可以大致看出，这 200 个样本点数据大致分成三段，其中第一段样本点和第三段样本点分为第 2 类，第二段的样本点划分为第 1 类。为了详细看出每个样本点的分类情况，故将每个样本点的分类结果列出，即表 1 所示：

表 1 问题一高维数据分类结果

样本编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
类别标签	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
类别标签	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
类别标签	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
类别标签	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
类别标签	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
类别标签	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140
类别标签	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
类别标签	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
类别标签	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
类别标签	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

从表 1 中可以明显看出，从第 1 个样本点到第 40 个样本点之间，全都被划分

到第 2 类。从第 41 个样本点至第 140 个样本点，全部划入第 1 类。从第 141 个样本点到第 200 个样本点全部划入第 2 类。从最终统计结果可以看出，总共 200 个样本点，进行分类后，落入第 1 类和第 2 类分别有 100 个样本点。

五、问题 2 分析及模型建立与求解

5.1 问题分析

在问题 2 中主要是处理四个低维空间中的子空间聚类。针对（a）将两条交点不在原点且相互垂直的两条直线区分成两类，在此运用前面问题 1 所建立的稀疏子空间聚类模型。针对（b）需将一个平面和两条直线分成三类以及将（c）两条不相交的二次曲线分成两类，本文将采用拉普拉斯特征映射(LEM)进行求解分析。针对(d)中两条相交的螺旋线分成两类，在此本文采用谱多流形聚类方法(SMMC)进行分类。首先从整个数据中分离出不同的连通或可分离子集，从而将由单一流形构成的纯粹子集和由相交流形构成的交叠子集区分开来,然后进一步将相交的子集分割为相交区域和非相交区域,

5.2 模型的建立与求解

5.2.1 两条交点不在原点且相互垂直的直线分类

在此运用前面问题 1 所建立的稀疏子空间模型以及运用交替方向对 2a.mat 中数据进行分类。实验环境为：Windows 7 系统和 Matlab 2012b，CPU 为 i3-2310M CPU 和 2GB 内存。经过多次实验，现做如下设置：最大迭代次数为 300，$n=2$，$\rho=20$，$\lambda_z=1.01$，$\varepsilon=0.0001$，程序见附录 2.1。分类结果如下图所示：

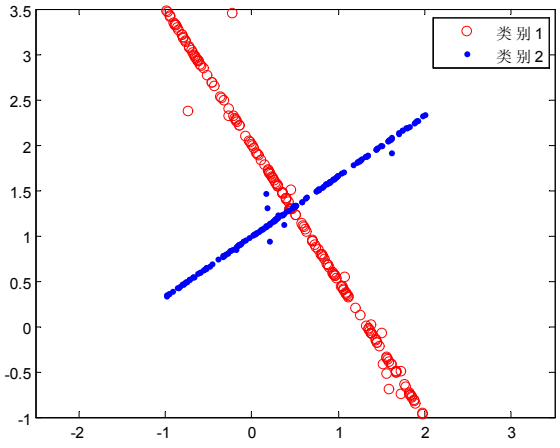


图 2 相交两条直线的分类结果图

从图 2 可以看出，对 340 个样本点运用子空间聚类模型划分成两类。其中红色圆圈表示的类别 1，蓝色实心正方体表示类别 2。在用红色圆圈表示的类别 1 中在图像的左上角有两个点明显偏离于类别 1，得知在类别 1 中有两个离群点。其中在两类别相交处的地方，有几个离群点偏离于类别 2，从图形中可以看出该模型能

够准确的将离群点划分至所属的类别。

5.2.2 不满足独立子空间的一个平面和两条直线分类

由于 2b.mat 数据集中处在多个子空间里，单线性模型处理效果不是很理想。所以本文采用拉普拉斯特征映射(LEM)方法进行聚类[5]。

LEM 方法对非线性流形的分类思想：首先将流形上样本点之间的局部近邻关系形式化为一个无向加权图，其中样本点对应图的顶点，样本之间的局部近邻关系对应图的边，而边的权值等于近邻样本之间的某种距离或相似性度量，然后在最优地保持样本之间的这些局部近邻关系的情况下，通过图嵌入的方式在低维空间得到嵌入坐标表示。其具体过程如下：

Step1. 构造近邻图：LEM 的第一步将所有数据点当做加权图的顶点，然后基于输入空间 N 个数据点之间的欧式距离将最后的数据点连接起来形成加权图的边。通常有两种近邻图的构造方式: $A.K-$近邻图（$K\in\mathbb{R}$）；当 x_i 在 x_j 的 K 个近邻点以内或者 x_j 在 x_i 的 K 个近邻点以内时，连接顶点当 x_i 在 x_j； $B.\varepsilon-$近邻图（$\varepsilon\in\mathbb{R}$）：当 $\left\|x_i-x_j\right\|^2<\varepsilon$ 时，连接 x_i 和 x_j。

Step2. 计算加权图的权值: LEM 的第二步赋予近邻图中的每条边一定的权值以度量数据点之间的近邻关系或相似性程度。通常有两种边的权值赋予方式: $A.$ 热核或高斯核:当顶点 x_i 和 x_j 之间有边连接时, $w_{ij}=w_{ji}=\exp\left(-\frac{\left\|x_i-x_j\right\|}{2}\right)$ 否则 $w_{ij}=0$； $B.$简单方法 $w_{ij}=1$,否则 $w_{ij}=0$。

Step3. 该目标函数的最优解 Y 最终转化为广义特征矩阵 $Ly=\lambda Dy$ 的谱分解问题。具体地,设上述广义特征矩阵的前 $d+1$ 个最小特征值所对应的特征向量为 $u_0,u_1\cdots,u_d$ (其中 u_0 对应特征值 0),则原始数据的低维嵌入表示为 $Y=\left[u_1,\cdots,u_d\right]^T$.

对 2b.mat 数据集中的 300 个样本点，运用上述方法进行分类，结果见图 3。程序见附录 2.2。

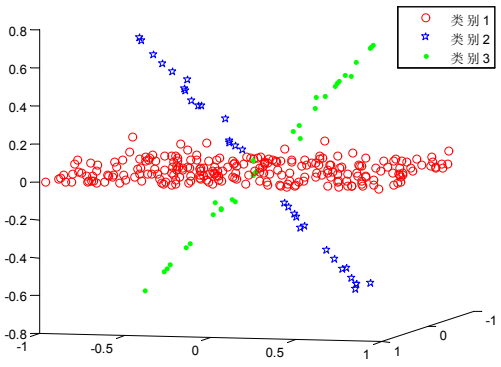


图 3 一个平面和两条直线的分类结果图

从图 3 可以看出，红色圆圈表示平面（类别 1），蓝色五角星表示直线（类别 2），另外一条直线（类别 3）用绿色实心小圆点表示，总体将 300 个样本点分成了三类别。从图中可以看出几乎没有离群点，整体的聚类结果比较清晰和准确。

5.2.3 两条不相交的二次曲线分类

从 2c.mat 数据集中可以看出，样本点的数据结构为 $2\times 400\in\mathbb{R}^2$，属于低维空间，又因为两条曲线不相交也即没有重叠区域，故本文继续采用拉普拉斯特征映射(LEM)方法对该数据进行分类。所得分类结果见图 4，程序见附录 2.3。

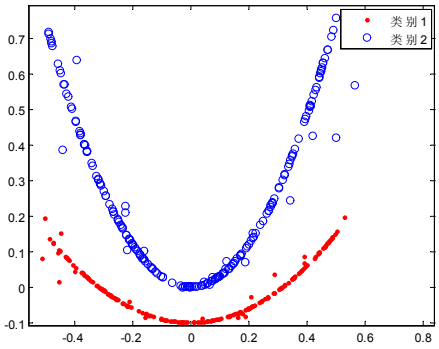


图 4 两条不相交的二次曲线分类结果图

从图 4 中可以看出，红色的实心小圆点表示类别 1，另外蓝色的圆圈表示所分为类别 2，将整体 400 个样本点分成两类。类别 1 和类别 2 中都有几个离群点，特别是在类别 2 中，有几个样本点分离较远。总体的分类结果比较准确。

5.2.4 两条相交的螺旋线分类

由于两条螺旋线相交，它们之间有重叠部分，在此本文选用谱多流形聚类方法（Spectral Multi-Manifold Clustering 简记 SMMC）[6-7]对混合流形聚类。

SMMC 分类的基本思想：从相似性矩阵的角度出发，充分利用流形采样点所内含的自然的局部几何结构信息来辅助构造更合适的相似性矩阵并进而发现正确的流形聚类。

相似性矩阵构造基于下述事实:a)尽管数据在全局上位于或近似位于光滑的非线性流形上，局部地，每个数据点和它的近邻点位于流形的一个局部线性块上;b）每个数据点的局部切空间提供了非线性流形局部几何结构的优良低维线性近似;c）在不同流形聚类的相交区域，来自于同一个流形聚类的数据点有相似的局部切空间而来自不同流形聚类的数据点其切空间是不同的。因此，在此可以利用数据点所内含的局部几何结构信息来辅助构造更合适的相似性矩阵。

在构造相似性矩阵时，既要考虑数据点之间的欧氏距离关系 $q_{ij}=q\left(\left\|x_i-x_j\right\|\right)$（称为局部相似性,），又要考虑数据点局部切空间之间的相似性 p_{ij}（称为结构相似性,）。这两个相似性融合在一起来决定最后的相似性权值：

$$w_{ij}=f(p_{ij},q_{ij}), \qquad (5\text{-}1)$$

其中 f 是一个合适的融合函数。 f 是关于数据点间欧氏距离的一个单调递减

函数同时是局部切空间之间相似性的单调递增函数。

下面给出 SMMC 方法中所采用的函数 p,q 和 f 的具体形式。

假设数据点 $x_i(i=1,\cdots,N)$ 处的局部切空间为 Θ_i，则两个数据点 x_i 和 x_j 的局部切空间之间的结构相似性可以定义为：

$$p_{ij}=p(\Theta_i,\Theta_j)=\left(\prod_{i=1}^{d}\cos(\theta_l)\right)^o \tag{5-2}$$

在（5-2）中，$o\in N^+$ 是一个可调节参数。$0\le\theta_1\le,\cdots\le\theta_d\le\pi/2$ 是两个切空间 Θ_i 和 Θ_j 之前的主角度，递归地定义为：

$$\cos(\theta_1)=\max_{\substack{u_1\in\Theta_i,v_1\in\Theta_j\\ \|u_1\|=\|v_1\|=1}} u_1^T v_1 \tag{5-3}$$

$$\cos(\theta_l)=\max_{\substack{u_l\in\Theta_i,v_l\in\Theta_j\\ \|u_l\|=\|v_l\|=1}} u_l^T v_l,\quad l=2,\cdots,d \tag{5-4}$$

其中 $u_l^T u_i=0,\quad v_l^T v_l=0,\quad i=1,\cdots,l-1$ 。

数据点 x_i 和 x_j 之间的局部相似性简单地定义为：

$$q_{ij}=\begin{cases}1 & if\ x_i\in Knn(x_j)\ or\ x_j\in Knn(x_i)\\ 0 & otherwise\end{cases} \tag{5-5}$$

在此构造近邻图采用 K-近邻图。

最后函数 f 将这两个函数 p 和 q 简单的乘在一起得到相似性权值:

$$w_{ij}=p_{ij}q_{ij}=\begin{cases}\left(\prod_{l=1}^{d}\cos(\theta_l)\right)^o & if\ x_i\in Knn(x_j)\ or\ x_j\in Knn(x_i)\\ 0 & otherwise\end{cases} \tag{5-6}$$

在给定样本点 x 和和它在欧式空间度量下的 n 个近邻点 $N(x)=\{x^1,\cdots,x^n\}$，x 附近的局部信息内含在该点出的局部采样协方差矩阵 Σ_x 中：

$$\Sigma_x=1/n\sum_{i=1}^{n}(x^i-\mu_x)(x^i-\mu_x)^T \tag{5-7}$$

其中 $\mu_x=\frac{1}{n}\sum_{i=1}^{n}x^i$ 。

样本点 x 处的局部切空间 Θ_x 由 Σ_x 的最大 d 个奇异值对应的左奇异向量给出。假设 Σ_x 的 SVD 为：

$$\Sigma_x = \begin{bmatrix} U_d & \tilde{U}_d \end{bmatrix} \begin{bmatrix} \Sigma_d & 0 \\ 0 & \tilde{\Sigma}_d \end{bmatrix} \begin{bmatrix} V_d & \tilde{V}_d \end{bmatrix}^T \tag{5-8}$$

其中$\begin{bmatrix} U_d & \tilde{U}_d \end{bmatrix} \in \mathbb{R}^{D\times D}$ 是正交矩阵并且$U_d \in \mathbb{R}^{D\times d}$ ，则有：

$$\Theta_x = span(U_d) \tag{5-9}$$

下面,我们将给出一个快速有效的方法来逼近每个数据点附近的局部切空间。本文的基本思想基于如下事实：a）全局非线性流形在局部能被一系列局部线性流形很好的逼近[8,9]；b）主成分分析器[10]可以有效地穿过相交线性流形；c）被同一个线性分析器逼近的数据点通常具有相似的局部切空间并且这些切空间可以被局部分析器的主子空间很好地近似。因此,我们可以训练一系列局部线性分析器来逼近潜在的流形,然后估计每个给定数据点的局部切空间为其相应局部分析器的主子空间。

具体地说，我们训练M 个混合概率主成分分析器来估计局部切空间，其中每个分析器由模型参数$\theta_m = \{\mu_m, V_m, \sigma_m^2\}\ (m = 1,\cdots,M)$ 刻画，其中$\mu_m \in \mathbb{R}^D$， $V_m \in \mathbb{R}^{D\times d}$ ，而σ_m^2 是一个标量。需要指出的是, M 是用于逼近所有潜在的线性或非线性流形的局部线性子模型的个数。在第m 个分析器模型下,一个D 维的观测数据向量x 通过下式对应一个相应的d 维潜在向量y :

$$x = V_m y + \mu_m + \varepsilon_m \tag{5-10}$$

其中μ_m是数据的均值向量，潜在变量y 和噪声ε_m 分别是高斯分布$y \sim N(0, I)$ 和$\varepsilon_m \sim N(0, \sigma_m^2 I)$ 。在此模型下， x 的边际分布为：

$$p(x \mid m) = (2\pi)^{-D/2} \left|C_m\right|^{-1/2} \exp\left\{-\frac{1}{2}(x-\mu_m)^T C_m^{-1}(x-\mu_m)\right\} \tag{5-11}$$

其中模型协方差为：

$$C_m = \sigma_m^2 I + V_m V_m^T \tag{5-12}$$

模型参数 μ_m, V_m, σ_m^2 可以通过利用 EM 算法最大化观测数据$\chi = \{x_i, i = 1,\cdots,N\}$ 的对数似然来得到：

$$L = \sum_{i=1}^{N} \ln\{\sum_{m=1}^{M} \pi_m p(x_i \mid m)\} \tag{5-13}$$

其中π_m是混合比例,满足条件$\pi_m \geq 0$和$\sum_{m=1}^{M} \pi_m = 1$。具体地说,EM 学习的主要过程。

$E-step$ ：利用当前模型参数$\theta_m = \{\mu_m, V_m, \sigma_m^2\}$计算：

$$R_{im}=\frac{\pi_m p(x_i\mid m)}{\sum_{m=1}^{M}\pi p(x_i\mid m)} \tag{5-14}$$

$$\pi_m^{new}=\frac{1}{N}\sum_{m=1}^{N}R_{im} \tag{5-15}$$

$$\mu_m^{new}=\frac{\sum_{m=1}^{N}R_{im}x_i}{\sum_{m=1}^{N}R_{im}} \tag{5-16}$$

$M-step$:重新估计参数V_m和$\sigma^2{}_m$为

$$V_m{}^{new}=S_mV_m(\sigma_m^2\mathrm{I}+T_m^{-1}V_m^TS_mV_m)^{-1} \tag{5-17}$$

$$(\sigma_m{}^2)^{new}=\frac{1}{d}tr[S_m-S_mV_mT_m^{-1}(V_m^{new})^T] \tag{5-18}$$

其中

$$S_m=\frac{1}{\pi_m^{new}N}\sum_{i=1}^{N}R_{im}(x_i-\mu_m^{new})(x_i-\mu_m^{new})^T \tag{5-19}$$

$$T_m=\sigma_m^2\mathrm{I}+V_m^TV_m \tag{5-20}$$

本文中采用$K-means$来初始化上述EM学习过程。最后,样本点x_i根据下述

关系分组到第j个局部分析器 ：

$$p(x_i\mid j)=\max_m p(x_i\mid m) \tag{5-21}$$

同时其局部切空间由下式给出:

$$\Theta_i=span(V_j) \tag{5-22}$$

谱多流形聚类方法(SMMC)算法步骤

算法3 谱多流形聚类方法(SMMC)算法

输 入:原始数据集χ,聚类数k,流形维数d,局部化模型数M,近邻点数K,

节参数o

输 出:原始数据对应的聚类结果

Step1 利用MPPCA训练个维的局部线性模型来近似潜在的流形数据;

Step2 根据式(5-22)确定每个点的局部切空间;

Step3 利用式(5-3)计算两个局部切空间之间的结构相似性;

Step4 利用式(5-6)计算相似性矩阵$W \in \mathbb{R}^{N\times N}$，并计算对角矩阵$D$，其中

Step5 计算广义特征矩阵$(D-W)u=\lambda Du$最小k个特征值对应的特征向量

Step6 利用$K-means$将$U=[u_1,\cdots,u_k]\in\mathbb{R}^{N\times k}$的行向量分组为$k$个聚类。

实验环境为：Windows 7 系统和 Matlab 2012b，CPU 为 i3-2310M CPU 和 2GB 内存。原始数据大小 2×988，参数设置：原始数据 $M=[N/(10d)], k=2,\ K=2\log(N), d=2,\ o=8$，程序见附录 2.4。运用上述模型得到图 5 分类结果。

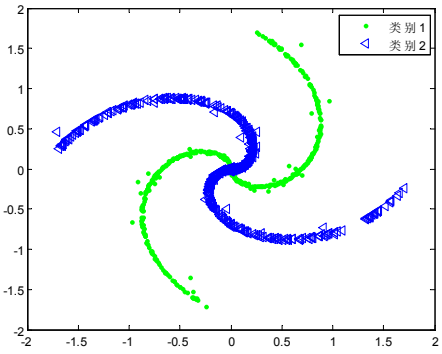


图 5 两相交螺旋线的分类结果

从图 5 可以看出，绿色小圆点表示类别 1，在类别 1 中有一些具有较小偏离的离群点。蓝色三角形表示类别 2，类别 2 中样本点相对比较稳定，几乎没有离群点。整体的分类效果比较理想，从而说明谱多流形聚类对于有重叠的混合流聚类具有较好的效果。

六、问题 3 分析及模型建立与求解

6.1 问题分析

针对（a）问题，十字上的点的位置信息已经提取出来，为了确定十字的中心位置，需先将十字中的点按照“横”和“竖”分成两类。但由于十字上的中心位置为重叠区域，如果运用一般的线性聚类模型，分类效果可能不佳。从问题 2（d）中得知谱多流形聚类对于有重叠的混合流聚类具有较好的效果。故本文继续运用谱多流形聚类对本问题（a）进行分类。

针对（b）问题，涉及到对已提取特征点的轨迹进行运动分割。解决该问题的模型来自于问题 1 中所建立的稀疏子空间聚类模型。在此需要对交替方向法进行改进，调整交替方向法的惩罚参数，将稀疏子空间聚类应用到视频运动分割中，从而进行特征点轨迹分类。

针对（c）问题，需要将在不同光照下的人脸图形分成两类，本文将对交替方向法做进一步改进即研究线性化的自适应线性交替方向法。同时把稀疏子空间聚类应用到人脸分类中进行分析。

6.2 模型的建立与求解

6.2.1 十字点的分类

从 3a.mat 数据集中可以看出，样本点的数据结构为 $2\times 2835\in\mathbb{R}^2$，属于低维空间，又因为十字上的点有重叠区域，故本文继续采用谱多流形聚类(SMMC)方法对该数据进行分类。

经过多次调试，设定参数如下：$M=323, k=2,\ K=6.1, d=1,\ o=7$。程序见附录 3.1。

所得分类结果如图 6 所示：

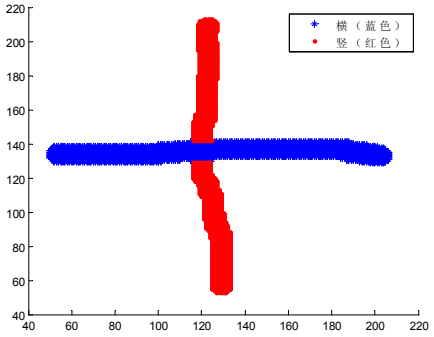


图 6 十字线上点的分类结果

从图 6 可以看出，蓝色星号表示分成的“横”类，红色小圆点表示分成的“竖”类。从分类的结果来看，分类结果比较准确，在横竖类别中几乎没有离群点。从而也证明谱多流形聚类方法对于有重叠区域流的数据分类效果较好。

6.2.2 不同运动物体的运动分割

在此将将惩罚参数自调整交替方向法和稀疏子空间聚类对视频运动分割。

（1）惩罚参数自调整交替方向法

在交替方向算法中，惩罚因子 ρ 是固定不变的。但是可以观察到固定的惩罚因子的交替方向法收敛很慢，而且选择一个最优的惩罚因子是很困难的。所以本文研究了对于惩罚因子 ρ 的自调整策略。

$$\rho_{k+1}=\min(\rho_{\max},\beta\rho_k) \tag{6-1}$$

其中 $\rho_{\max}$ 是 $\{\rho_k\}$ 的上限，β 值定义如下：

$$\beta=\begin{cases}\beta_0, & \rho_k \max\left(\|C_{k+1}-C_k\|,\ \|A_{k+1}-A_k\|\right)<\varepsilon_2\\ 1, & others\end{cases} \tag{6-2}$$

其中 $\beta_0 \geq 1$ 是常数。实际应用中变化的 β 是首选。ε_2 常数，文中此处取 $\varepsilon_2 = 10^{-4}$。这种方法根据迭代停止标准来平衡误差并且引进几个参数。。

算法4 稀疏求解的惩罚参数自调整交替方向法算法

输 入：待分类的数据集 Y

输 出：最优稀疏系数矩阵 $\mathrm{C}^* = C^k$

初始化：$\mathrm{K} = 0$，$\mathrm{C}^{(0)}$，$\mathrm{A}(0)$，和 ρ_0

while $\left\|\mathrm{A}^{(k)} - C^{(k)}\right\|_\infty \geq \varepsilon, \left\|A^{(k)} - A^{(k-1)}\right\|_\infty \geq \varepsilon$ 做下面操作：

Step1 通过求解下列等式更新 A^{k+1}：$\left(\lambda_z Y^T Y + \rho_k I\right) A^{k+1} = \lambda_z Y^T Y + \rho_k C^k - \Delta^k$ ；

Step2 更新 $C^{(k+1)}$ 如下：$C^{(k+1)} = J - diag(J)$，其中 $J = T_{1/\rho}\left(A^{(k+1)} + \Delta^k / \rho\right)$；

Step3 更新 Δ^{k+1} 如下：$\Delta^{k+1} = \Delta^k + \rho_k (A^{k+1} - C^{k+1})$；

Step4 $\rho_{\mathrm{k}+1} = \min(\rho_{\max}, \beta\rho_{\mathrm{k}})$；

Step5 $k = K + 1$;

end while

（2）稀疏子空间聚类的运动分割

本节主要考虑具有跟踪特征点的运动分割问题。数据从 F 帧 N 个特征点轨迹 $\left\{\left(x_{if}, y_{if}\right)\right\}_{i=1,\ldots,N,f,\ldots,F}$，其中 x_{if} 和 y_{if} 是特征点 i 在 f 帧的坐标值。本文主要的目标是估计每一个特征点 i 所所对应的标记 c_i，即把特征点根据不同的运动划分到不同的组。

运动分割线性空间稀疏表示运动分割的研究常假设为仿射摄像机模型。在这种假设下，可以看到同一个物体（相同的运动）下特征点的轨迹在线性空间里至少是四维。事实是这样的，仿射摄像机模型下矩阵 $Y \in R^{2F \times P}$ 包括一个物体上的所有特征点的坐标，可以写成如下：

$$\begin{bmatrix} u_{11} & u_{21} & \cdots & u_{P1} \\ v_{11} & v_{21} & \cdots & v_{P1} \\ \vdots & \vdots & \ddots & \vdots \\ u_{1F} & u_{2F} & \cdots & u_{PF} \\ v_{1F} & v_{2F} & \cdots & v_{PF} \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_F \end{bmatrix} \begin{bmatrix} U_1 & \cdots & U_P \\ V_1 & \cdots & V_P \\ Z_1 & \cdots & Z_P \\ 1 & \cdots & 1 \end{bmatrix} \qquad (6\text{-}3)$$

其中 U_{if}，V_{if} 和 Z_{if} 都是点 i 的三维坐标，$A_f \in R^{2\times 4}$ 是 f 帧的仿射运动矩阵。在

（6-3）式的坐标矩阵的秩为4在左边的矩阵所位于四维的子空间R^{2F}里。在这个框架里，运动分割可以归结为线性子空间划分问题，其中每一个子空间对应一个具体运动，在本文中，轨迹矩阵$Y \in R^{2F\times P}$是由来自等式（6-3），但是由来自不同物体上N个特征点组成。

可以看出，对于每一个数据点位于一个线性子空间集Y中。这里利用数据的自表示性质，即位于子空间集里的每一个点可以由数据集的其他的点有效表示，这里可以把W当成字典，可以写成下式

$$y_i = Yc_i, \quad c_{ii} = 0 \tag{6-4}$$

其中$c_i = \begin{bmatrix} c_{i1} & c_{i2} & \cdots & c_{iN} \end{bmatrix}^T$，约束条件$c_{ii} = 0$消除了自己表示自己的特殊情况。换句话说，数据矩阵Y是一个自表示字典，在这种情况，每个点是由其他点线性表示。可是，数据点用字典Y表示一般不唯一。本文主要目标是找出最稀疏解，数据点是由其他数据表示且个数最少。

对于等式（6-4）有许多解，但是可以通过增加一个目标函数以达到限制并得到所需要的解。写成下式（6-5）

$$\min \left\| c_i \right\|_q, s.t., y_i = Yc_i \tag{6-5}$$

这里选择不同的q，会得到不同解。当q由无穷大到0变化时，得到稀疏解会增加.

当$q = 0$时，可以得最理想的稀疏解。但是当q为零时，所对应优化式是非凸的并且是一个NP–*hard*问题。因此使用凸松弛方式（用L_0范数替代原有的L_1范数），将上述非凸优化转变如下：

$$\min \left\| c_i \right\|_1, s.t., y_i = Yc, c_{ii} = 0 \tag{6-6}$$

上面优化等式可以通过凸优化工具[11,12]有效解决。对于所有的数据，我们可以用矩阵形式重写（6-7）稀疏优化式子，如下，

$$\min \left\| c_i \right\|_1, s.t., y_i = Yc, diag\left(C\right) = 0 \tag{6-7}$$

其中$C \in R^{N\times N}$为稀疏系数矩阵，其第i列对应点y_i的稀疏表示。$C = \begin{bmatrix} c_1 & c_2 & \cdots & c_N \end{bmatrix}$并且这里$diag\left(C\right) \in R^N$表示一个向量，为矩阵$C$的对角元素。面考虑的是理想情况下，数据点的稀疏表示，在某些情况下，数据里包含了噪声，在实际应用中，需要考虑噪声的存在，改写（6-8）式，加上噪声约束如下，

$$\min_{\langle C,Z\rangle} \left\| C \right\|_1 + \frac{\lambda_z}{2} \left\| Z \right\|_F^2 \tag{6-8}$$

$$s.t. \quad Y = YC + Z \qquad C_{ii} = 0 \quad i = 1,2,\ldots,N$$

其中参数$\lambda_z > 0$，求解上式得到每个数据点的稀疏表示。

由（6-8）得到每个数据点的最优稀疏系数表示，然后通过算法利用每个数据的最优稀疏系数把数据点划分到不同的子空间。为了解决这个问题，首先建立带有权重的图$G=(v,\varepsilon,W)$,其中v表示与N个数据点相对应的结点，$\varepsilon\subseteq v\times v$表示结点之间的边。$\mathrm{W}\in R^{N\times N}$是非负对称相似矩阵，表示边的权重。比如节点$i$与节点$j$之间的边的权重为$w_{i_j}$。理想的相似矩阵与理想的相似图$G$情况下，同一个子空间里点相互连接，属于不同子空间里的点不连接。理想情况下，稀疏优化求解稀疏系数对每个点恢复。如一个非零稀疏表示与同一个子空间里的点相对应。因此可以选择相似矩阵$W=|C|+|C|^{\mathrm{T}}$。换句话说，节点i和其相连的节点j之间的权重相等。相似矩阵对称的原因是数据点$y_i\in S_l$可以由包括同一个子空间一些点线性组合。可是，数据点y_i没有必要选择y_i做其稀疏表示系数。通过对权重的特殊选择，可以确保点i和点j之间是相互连接，点i和点j可以相互稀疏表示。

通过这种方式建立相似矩阵，理想情况下，n个连接部分对应着n个子空间，如下式，

$$W=\begin{bmatrix}\mathrm{W}_1 & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \mathrm{W}_n\end{bmatrix}\Gamma \qquad (6\text{-}.9)$$

其中W_ℓ是子空间S_ℓ里的数据点的相似矩阵。对G进行谱聚类[13]把数据划分到对应的子空间里。更具体一点来说，通过把K均值算法[14]应用到矩阵的每一列获得数据的聚类，这个矩阵的每一列为拉普拉斯矩阵的n个特征向量。具体谱聚类算法如下，

算法 5　谱聚类算法（Spectral Clustering）

输 入： n个线性子空间$\{\mathrm{S}_i\}_{i=1}^{n}$组成的数据点$\{y_i\}_{i=1}^{N}$。

输 出：数据集Y的n子集划分$y_1,y_2,\cdots,y_n$。

Step 1　构造基于数据集Y的相似矩阵W；

Step 2　计算相似矩阵W的*Laplacian*矩阵，其中矩阵$\mathrm{L}=\mathrm{I}-\mathrm{D}^{-1/2}\mathrm{W}\mathrm{D}^{1/2}$，

Step 3　求出L的前n个特征值，以及其对应的特征向量$\{u_i\}_{i=1}^{k}$；

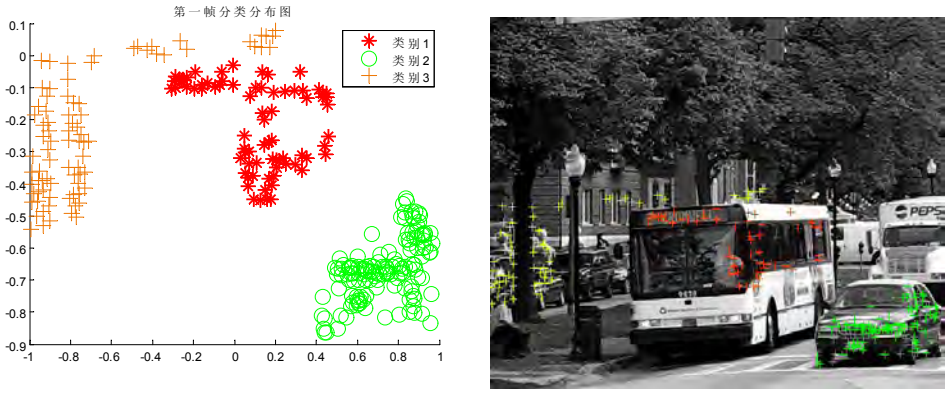
Step 4　把k个特征（列）向量排列在一起组成一个$N\times k$的矩阵；

Step 5　把生成矩阵矩阵每一行看作k维空间中的一个向量，使用$K-means$算法聚类成n类；

（3）分类结果与分析

在此运用模型和算法将 3.b.mat 中高维数据进行分类。

实验环境为：Windows 7 系统和 Matlab 2012b，CPU 为 i3-2310M CPU 和 2GB 内存。经过多次手动调节，现做如下设置：最大迭代次数为 $k=200$，$\rho_0=10$，$n=3$，$\varepsilon=0.0001$，程序见附录 3.2。如图 7 所示给出分类结果图。



(a) **(b)**

图 7 基于运动分割的分类结果图与实际图

从图 7(a)可以看出，红色星点表示为类别 1，绿色空心圆圈表示为类别 2，黄色十字形表示类别 3。从图 7(a)中可以看出，各类别之间没有交错，其中类别 1 中的特征点比较分散。相反，类别 2 中的特征点比较集中。整体来看，对比实际图形 7(b)可以看出，对同一运动视频中的分类结果比较准确。

为了看出特征点轨迹的分类情况，将第 1 类记为类比标签“1”，将第 2 类记为类别标签“2”，将第 3 类记为类别标签“3”，做出基于运动分割的第一帧分类结果图。

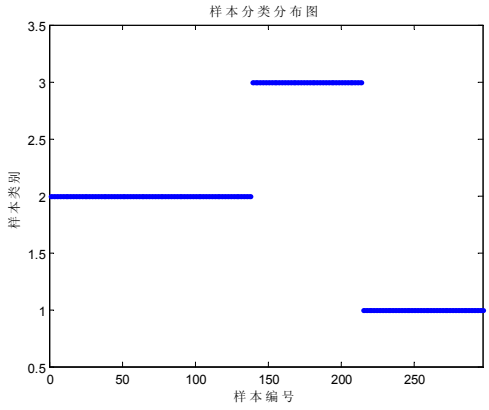


图 8 基于运动分割的第一帧分类图

从图 8 可以看出，将全部的特征点轨迹分成三部分，第一部分被划为到第 2 类，第二部分被划分到第 3 类，最后一部分被划分到第 1 类，且在分类过程中没有出现离群点。为了更加清晰看出第一帧分类的结果，故将每个特征点轨迹的分类结果列出，见表 2。

表 2 基于运动分割的第一帧分类结果

样本编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
样本编号	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140
样本类别	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3
样本编号	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
样本类别	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
样本编号	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180
样本类别	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
样本编号	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200
样本类别	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
样本编号	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220
样本类别	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1	1	1	1	1
样本编号	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
样本类别	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260
样本类别	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280
样本类别	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
样本编号	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297			
样本类别	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			

从表 2 中可以明显看出，从第 1 个特征点轨迹到第 138 个特征点轨迹，全部划分为第 2 类。从第 139 个特征点轨迹起至第 214 个特征点轨迹，全部划分为第 3 类。从第 215 个特征点轨迹到第 297 个特征点轨迹，全部划分为第 1 类。

6.2.3 不同光照下的两人脸图形分类

本文为了解决该问题，主要对交替方向法进行了改进，研究线性化的自适应交替方向法。同时把稀疏子空间聚类应用到人脸分类中。即考虑在不同光照下具有固定的姿势的人脸图像分类，通过稀疏系数表示这些人脸图像集，这些人脸图像可以用低维线性空间表示。

稀疏系数与子空间聚类相结合，一个类别的人脸图片都是在同一个线性子空•间中[15]。人脸子空间模型能够很灵活的获取真实数据中的各种变化，因为同一个人脸在不同的光线条件及表情下的图像处于一个子空间，这个子空间称为人脸子空间[4],因此，子空间聚类模型可以运用到人脸分类中。在稀疏子空间聚类中人脸分类问题可以归结于如下数学形式：

$$\begin{aligned} &\min \ \ f(x)+g(x) \\ &s.t. \quad Ax+Bz=c \end{aligned} \tag{6-10}$$

其中，$x\in R^n, z\in R^m, A\in R^{p\times n}, B\in R^{p\times m}, c\in R^p$， f 和 g 都是凸函数。在此本文主要运用基于交替方向法基础之上的线性化交替方法进行求解。

（1）自适应的线性化交替方向法

在交替方向法中，解决大规模的稀疏表示的问题中使用的非常广泛。当用交替方向法解决（6-10）式时，首先得到下面的增广拉格朗日函数：

$$L_A(x,y,\lambda)=\mathrm{f}(x)+g(y)+\langle\lambda,A(x)\rangle+B(y-c)+\frac{\beta}{2}\left\|A(x)+B(y)-c\right\|^2 \tag{6-11}$$

其中 λ 是拉格朗日乘子，$\langle\bullet,\bullet\rangle$ 为内积运算，$\beta>0$ 是惩罚参数。通常做法是对关于 L_A 同时 x 和 y 最小化。这种做法通常难度很大，而且这种方法并不能利用目标函数是可分离的这种情况。为了解决这个问题，交替方向法把 L_A 的最小化分解为两个子问题，即分别各自关于 x 和 y 最小化，迭代步骤如下：

$$\begin{aligned} x_{k+1} &= \arg\min_x L_A(\mathrm{x},\mathrm{y}_k,\lambda_k) \qquad (6\text{-}12)\\ &= \arg\min_x f(\mathrm{x})+\frac{\beta}{2}\left\|A(\mathrm{x})+B(\mathrm{y}_k)-c+\lambda_k/\beta\right\|^2 \\ y_{k+1} &= \arg\min_y L_A(\mathrm{x}_{k+1},\mathrm{y},\lambda_k) \qquad (6\text{-}13)\\ &= \arg\min_y f(\mathrm{x})+\frac{\beta}{2}\left\|A(\mathrm{x}_{k+1})+B(\mathrm{y})-c+\lambda_k/\beta\right\|^2 \\ \lambda_{k+1} &= \lambda_k+\beta\left[A(x_{k+1})+B(y_{k+1})-c\right] \qquad (6\text{-}14) \end{aligned}$$

在压缩感知和稀疏表示中，f 和 g 通常都是矩阵范数或者向量范数。当 A 和 B 为单位矩阵时，子问题（6-12）和子问题（6-13）有封闭解。在这种情况，交替方向法是很吸引人的。可是，在很多情况下， A 和 B 并不是单位矩阵。例如，完备矩阵 A 为一个选择矩阵，在低秩描述和稀疏表示中， A 为一般矩阵，在这种情况下，对于子问题（6-12）和（6-13）没有封闭解。这样子问题（6-12）和（6-13）必须用迭代的方法解决。为了客服这种难题，一种通常做法是引进辅助变量 u 和 v 那么问题1可以等价于下面的形式：

$$\min_{x,y,u,v} f(x)+g(y) \quad s.t. \quad \mathrm{A}(u)+\mathrm{B}(v)=c, x=u, y=v \tag{6-15}$$

同样的，可以推导出交替方向法迭代类似于式子(5.3)和(5.4)。这样就会有更多的变量和限制，交替方向法的收敛也会更慢。而且更新u和v，它们的子问题为最小二乘问题，求解矩阵逆是必须的。而且，对于超过两个变量的交替方向法的收敛性还没有证明。

为了避免引进辅助变量，并且可以有效解决子问题（6-12）和（6-13），在此提出了对子问题（6-12）和（6-13）进行线性化。同时为了加快算法的收敛性，我们同意提出了一种自适应法则来更新惩罚参数。通过线性化x_k在子问题（6-12）中二次项，并且加上一个近似项，得到如下：

$$\begin{aligned} x_{k+1} &= \arg\min_x f(\mathrm{x})+\left\langle A^*(\lambda_k)+\beta A^*\left(A(\mathrm{x}_k)+B(\mathrm{y}_k)-c\right), \mathrm{x}-\mathrm{x}_k\right\rangle+\frac{\beta\eta_A}{2}\|x-x_k\|^2 \\ &= \arg\min_x f(\mathrm{x})+\frac{\beta\eta_A}{2}\left\|x-x_k+A^*\left(\lambda_k+\beta\left(A(\mathrm{x}_k)+B(\mathrm{y}_k)-c\right)\right)/(\beta\eta_A)\right\|^2 \end{aligned} \tag{6-16}$$

其中A^*为A的邻接矩阵，η_A是一个参数，之后会分析这个参数。上面的近似项类似于参考文献中的加速梯度投影(AGP)，但是这里并不使用AGP解决子问题（6-12）。同样，子问题（6-13）可以被近似如下：

$$y_{k+1}=\arg\min_y g(y)+\frac{\beta\eta_B}{2}\left\|y-y_k+B^*\left(\lambda_k+\beta\left(A(\mathrm{x}_{k+1})+B(\mathrm{y}_k)-c\right)\right)/(\beta\eta_B)\right\|^2 \tag{6-17}$$

子问题（6-14）还是按照以前一样迭代更新。

结合上问题中研究的惩罚参数自调整策略，得到惩罚参数自适应线性交替方向算法。

算法6　自适应线性交替方向算法

输　入：待分类的数据集Y

输　出：最优稀疏系数矩阵$\mathrm{C}^*=C^k$

初始化：$\mathrm{k}=0, \varepsilon_1>0, \varepsilon_2>0, \beta_{\max}\gg\beta_0>0, \eta_A>\|A\|^2, \eta_B>\|B\|^2, x_0, y_0, \lambda_0$

while　$\left\|A(\mathrm{x}_{k+1})+B(\mathrm{y}_{k+1})-c\right\|/\|c\|>\varepsilon_1$　做下面操作：

Step1　通过求解式（6-16）下列等式更新x；

Step2　通过求解式（6-17）下列等式更新y；

Step3　更新$\Delta^{\mathrm{k+1}}$如下：$\Delta^{\mathrm{k+1}}=\Delta^k+\rho\left(A^{k+1}-C^{k+1}\right)$；

Step4　$\rho_{\mathrm{k+1}}=\min\left(\rho_{\max}, \beta\rho_{\mathrm{k}}\right)$；

Step5　$k=k+1$；

end　*while*

（2）稀疏求解和分类

如前所述，用l_0范数求稀疏解，但是l_0范数的优化是非凸的且是一个NP-hard问题。但是当解足够稀疏时，最小l_0的解与最小l_1等价，此时问题转化为如下式：

$$\min \|x\|_1 \quad s.t. \quad Ax = y \quad x_{ii} = 0 \tag{6-18}$$

假设式子（6-18）是精确的，但往往所得到的数据都含有噪声，因此为了精确描述数据，这里在式子（6-18）上引入一个噪声项，含有噪声项的模型如下：

$$y = Ax_0 + z \tag{6-19}$$

其中$z \in R^m$是一个有限能量的噪声项，x_0还是能够通过最小l_1方法来求解：

$$\begin{aligned} &\min_{\langle C,Z \rangle} \|C\|_1 + \frac{\lambda_z}{2}\|Z\|_F^2 \\ &s.t. \quad Y = YC + Z \\ &C_{ii} = 0 \qquad i = 1,2,\cdots,N \end{aligned} \tag{6-20}$$

这里的凸优化问题已经通过前面章节解答。传统的方法是内点算法CVX优化工具包解决，本文采用交替方向法以及改进的算法。

求解上式得到每个数据点的稀疏表示，构造相似度矩阵$\mathrm{W} = [w_1, w_2, \cdots, w_N] \in W^{N \times N}$。为了使W对称，定义如下：

$$W = |C| + |C|^T \tag{6-21}$$

从另一方面说，结点i和结点j之间边的权重等于$|c_{ij}| + |c_{ji}|$。这里W由n个独立子空间生成，每个样本y_i可以由属于同一个子空间的其他样本进行稀疏重构，属于不同子空间的样本的稀疏表示系数为零。在相似度矩阵上使用谱聚类算法[16]进行划分。把每张人脸划分到相应的类别里。

综上所述，对用稀疏表示的人脸分类算法具体步骤如下：

算法7 人脸分类算法

输 入：待分类的数据集Y

输 出：最终分类结果

Step1 给定训练样本矩阵：$A = [A_1, A_2, \cdots, A_k] R^{m \times n}$，$k$类共有$n$个样本，

Step2 求解最小l_1范数。

Step3 根据稀疏系数构造相似矩阵。

Step4 把谱聚类应用于相似矩阵，以用来对人脸图片进行分类。

（3）分类结果与分析

实验环境为：Windows 7 系统和 Matlab 2012b，CPU 为 i3-2310M CPU 和 2GB 内存，程序见附录 3.3。3c.mat 中的数据为两个人在不同光照条件下的人脸图像，数据大小为 2160×20，共 20 幅图片数据，在此运用上述和算法将 1.mat 中高维数据进行分类，为书写方便，故将第 1 类记为类比标签“1”，将第 2 类记为类别标签“2”，分类结果如图 9 所示。

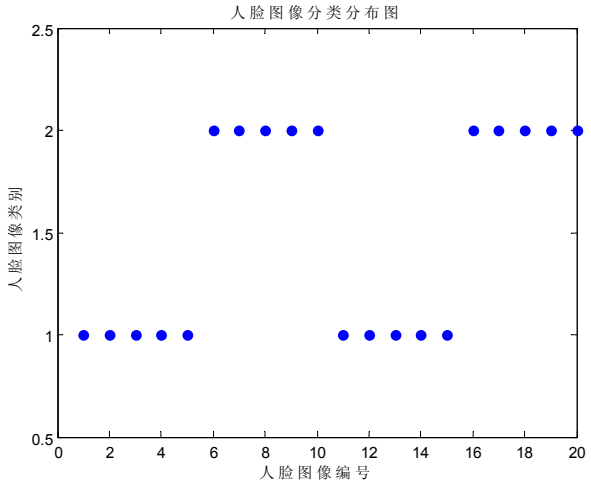


图 9 不同光照下人脸图图像分类图

从图 9 可以看出 1-5 幅图和 11-15 幅图都被划分到第 1 类，而 6-10 幅图和 16-20 幅图被划分到第 2 类。其中这 20 幅图两个人各占了 10 幅。同样的，做出 20 幅图形对应的类别标签见表 3。

表 3 在不同光照下的人脸图像分类

样本编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
样本类别	1	1	1	1	1	2	2	2	2	2	1	1	1	1	1	2	2	2	2	2

为了对我们分类的结果进行检验，我们尝试着将 20 幅人脸图像进行可视化（恢复人脸图像），由于 3c.mat 中的任意一列数据是人脸图像按列拉成的，所以首先将 2016 进行两项相乘的英式分解，即找出所有的 $(A_i, B_i \geq= 2)$ 的点对 $\{A_i, B_i\}$，使得 $A_i * B_i = 2016$，以 A_i 为长度，将 2016 个点分成 B_i 个长度为 A_i 的列向量，并将这些列向量按照分解出来的先后顺序拼接成一个 $A_i * B_i$ 的矩阵，在将这个矩阵中的每个元素除以 255，并将这个结果显示在屏幕上，在所有的 $\{A_i, B_i\}$ 点对应中，有唯一一对 $\{A_i, B_i\}$ 可以将这个人脸图形还原。我们找到 $A_i = 42, B_i = 48$ 。最终将所有的图形还原，见图 10。

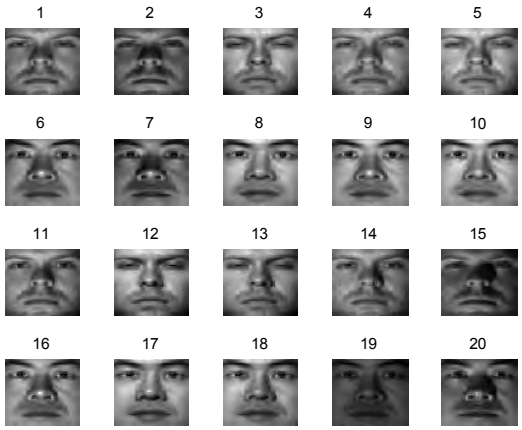


图 10 人脸还原图形

图 10 中对应的数字为原始数据中点的编号，从还原的人脸图形也进一步验证了所建模型对于人脸分类的有效性。所得的结果与人脑识别一致，本算法在此问题聚类的准确率达到 100%。

七、问题 4 分析及模型建立与求解

7.1 问题分析

针对（a）问题，本题要求将一个圆台分成三面（顶、底、侧面）。本题的数据大小为 3×8318，即有 8318 个样本点。子空间聚类的优化式是一个 NP-hard 规划问题，所以时间复杂度和空间复杂度都非常大，我们运用 SMMC 算法对原始数据直接进行聚类，发现内存溢出。所以我们首先对数据进行预处理，然后进行分类，即对原始数据进行采样，然后对采样点进行聚类，最后运用邻近分类算法对剩下的数据进行分类，以达到降低时间复杂度和空间复杂度。

针对（b）问题，机器工件外部边缘轮廓图较为复杂，而且噪音点较多，题目要求将轮廓线中不同的直线和圆弧进行分类,在此，运用问题 2 谱多流形聚类模型，并多次手动调节相关参数，期望达到理想的分类效果并选择最好的结果进行可视化。

7.2 模型的建立与求解

7.2.1 圆台的分类

在聚类时经常由于数据量较大，而无法在规定的时间或者空间内完成时，我们往往是先抽取部分样本点进行聚类，然后再运用分类算法对剩下的样本点进行分类，以达到减少时间和空间复杂度。

（1）邻近分类算法

由于多流形结构的数据无法运用普通的距离分类算进行分类,因此我们提出一个邻近分类算法，对采样之后剩下的点进行邻近分类。

算法 8 邻近分类算法

输 入： n 个类别 $\{S_i\}_{i=1}^{n}$，未分类的样点集 D。

输 出： 新的 n 个类别 $\{S_i^*\}_{i=1}^{n}$。

初始化：$\{S_i^*\}_{i=1}^{n}=\{S_i\}_{i=1}^{n}$；

while $|D|>0$ 做下面操作： // $|D|$ 表示 D 中本的个数。

Step1 取出 D 中的一个点 x，并将 D 中的 x 删除；

Step2 计算 x 到 S_i 的最短距离 $d_i\ (i=1,\cdots,n)$； // S_i 中离 x 最近点的距离。

Step3 取 $\{d_i\}_{i=1}^{n}$ 中的最小值，并记录其下标 k；

Step4 更新 S_k^*，$S_k^*\leftarrow S_k^*\cup\{x\}$； // 此处并未更新 S_k。

end while

对于圆台按底、顶、侧面分三类问题，由于 4a.mat 中的数据大小为 3*8318。有 8318 个三维数据点。我们尝试着用前面的几种算求解，虽然数据的维数为 3 维，但是数据的样本量过大，由于问题本身就是一个 NP 难问题，前面的几种算法都会导致内存溢出，无法进行求解，所以我们首先对数据进行预处理，从而减少计算量，先将数据点进行抽样，以减少计算所消耗的时间和内存空间。对于这 8318 个样本点，我们采取等间隔采样，间隔步长为 5，即每间隔 4 个点抽取一个点.，这样原来的 8318 个点变成了 1663 个点。我们在运用 SMMC 算法对这 1663 个采样点进行聚类。在运用邻近分类算法将剩下的样本点分到已经聚好的类当中去，具体算流程图见图 11。

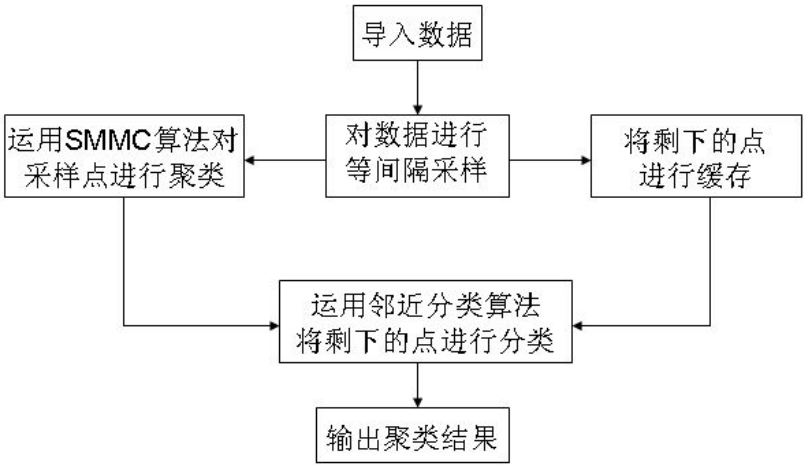


图 11 圆台分类流程图

（2）分类结果与分析

首先只将采样点进行聚类，并对聚类的结果进行可视化，其聚类的结果见图 12，程序见附录 4.1。在运用 SMMC 算法对采样点分类的结果和邻近分类算法对

剩下的点进行分类，分类后的可视化结果见图 13。

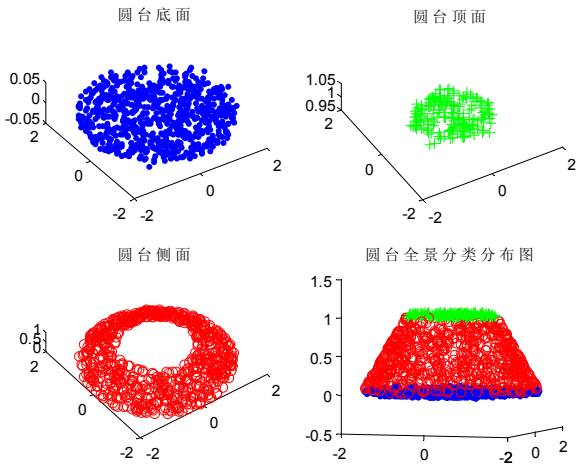


图 12 圆台采样点聚类图

通过图 12 中圆台底面，圆台顶面，圆台侧面三个图可知运用 SMMC 算法可以对采样点进行分类，通过圆台全景分类分布图也可以得出 SMMC 对小样本量的数据精度较高，能够有效的处理线性和非线性的混合流形聚类问题。

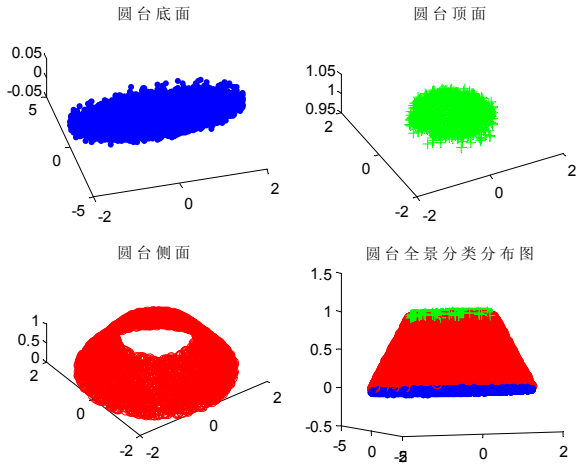


图 13 圆台聚类图

通过图 12 和图 13 对比发现，先对采样点进行分类，再运用邻近分类算法对剩余的点进行分类，可以将圆台的底面、顶面和侧面三个面分开。通过图 13 的圆台全景分类分步图可以得知对采样点进行 SMMC 聚类效果比较理想。从而也说明运用 SMMC 算法能有效的处理线性和非线性的混合流形聚类问题，先采样聚类再分类的方法对处理大规模数据减少时间和空间复杂度是可行的。

7.2.2 机器工作外部边缘轮廓的分类

通过对机器工作外部边缘轮廓图形观察得知，轮廓线中既有直线又有圆弧，并且是封闭的，同时存在较多的噪音点。在此，针对本问题建立谱多流形聚类模型。实验环境为：Windows 7 系统和 Matlab 2012b，CPU 为 i3-2310M CPU 和 2GB 内存。程序见附录 4.2。

第一次将轮廓线分成两类，经过多次手动调节，对参数做如下设置：

$M=351, k=2,\ K=4.2, d=1,\ o=6$

第二次将轮廓线分成三类，经过多次手动调节，对参数做如下设置：

$M=543, k=3,\ K=5.1, d=1,\ o=4.2$

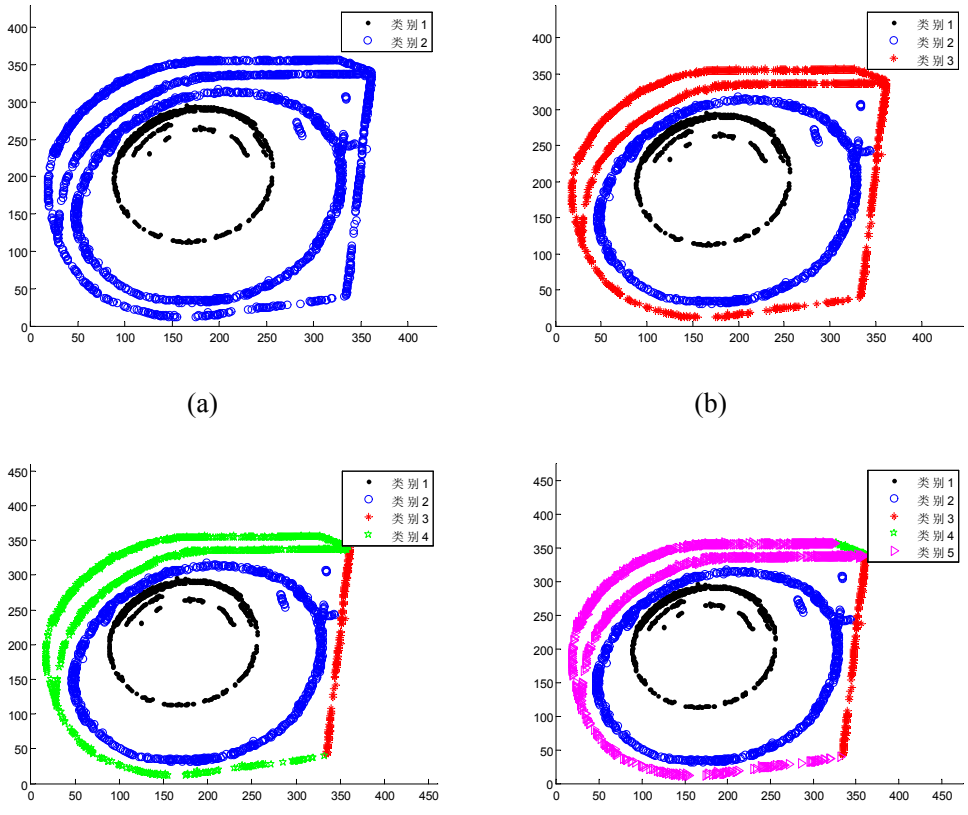
第三次将轮廓线分成四类，经过多次手动调节，对参数做如下设置：

$M=783, k=4,\ K=6.4, d=1,\ o=8$

第四次将轮廓线分成五类，经过多次手动调节，对参数做如下设置：

$M=994, k=5,\ K=5.3, d=1,\ o=5.8$

四种分类结果见图 14(a)-14(d)：



(a)　　(b)

(c)　　(d)

图 14　轮廓线 SMMC 聚类图

从图 14(a)-14(d)中，我们得出用 SMMC 算法可以分别有效地将机器工件外部轮廓分两类、三类、四类、五类，并且可以有效地将轮廓线中不同的直线和圆弧区分开。

八 、模型的总结与改进

1、稀疏子空间聚类模型

总结：本文主要运用稀疏子空间聚类模型以及其优化算法即交替方向法和其改进算法。具体建模过程：对相互独立的子空间进行分类研究中，本文建立稀疏子空间模型并采用交替方向优化算法对聚类进行求解。对交替方法法进行改进，研究了惩罚参数自调整交替方向法，同时把稀疏子空间聚类应用到运动分割中，跟踪特征点根据不同的物体的运动划分到不同的组。进一步对交替方向法进行改进，研究了线性化的交替方向法，同时把稀疏子空间聚类应用到人脸分类中，生成一个空间而且同一类的人脸图像都在同一个子空间中，这样对人脸分类转换成子空间聚类。将稀疏子空间聚类应用到视频运动分割和人脸分类中，都取得了较好的分类结果。

思考：目前稀疏子空间聚类的性能取决于两个方面，一方面是模型对各种噪声、数据缺损、奇异数据的鲁棒性取决于数据项；另一方面是子空间表示系数矩阵的结构取决于正则项。现有模型选择不同的度量来设计数据项与正则项，存在一定的局限性。鉴于此可以做如下改进：一方面，针对数据项，根据实际数据的噪声统计分布情况进行误差建模，从而更好地拟合数据，提高鲁棒性。另一方面，设计适当的正则项，使得相应的子空间聚类模型得到的系数矩阵满足类间稀疏、类内一致的性质，从而提高聚类的性能。

2、谱多流形聚类模型(SMMC)

总结：本文主要运用谱多流形聚类模型来处理具有相互重叠的流行结构中的聚类问题。针对题目中两条相交螺旋线的分类以及在多流形聚类问题中的实际案例中也采用谱多流形聚类模型进行聚类分析，从最终的聚类结果来看，分类比较精确。谱多流形聚类模型不同与一般的流形聚类，它主要从相似性矩阵的角度出发,充分利用流形采样点所内含的局部几何结构信息来辅助构造更合适的相似性矩阵来分组混合结构数据,即来自不同流形结构的数据点之间有相对低的权值方法从相似性矩阵的角度出发,充分利用流形采样点所内含的局部几何结构信息来辅助构造更合适的相似性矩阵实现混合流形聚类，对处理混合流形以及具有多次重叠的流行结构的聚类中具有较好的效果。

思考：从本文对该模型的运来来看，也存在一些不足之处。比如，在计算程中具有较高的计算复杂度，同时对噪声或离群点敏感，针对此问题可以对大规模、噪声复杂、进一步进行探讨和分析。同时也进一步完善其理论，寻求更加广泛地应用领域。

。

参考文献

[1] R. Basri and D. W. Jacobs. Lambertian reflectance and linear subspaces. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(2):218–233, 2003。

[2] R. Vidal. Subspace clustering. IEEE Signal Processing Magazine, 28(2):52–68, 2011。

[3] J. Shi and J. Malik, “Normalized cuts and image segmentation,” IEEE Transactions Pattern Analysis Machine Intelligence, 22(8):888–905, 2000。

[4] 姚刚. 稀疏子空间聚类的交替方向法研究[D].南京邮电大学,2014。

[5] 王勇. 基于流形学习的分类与聚类方法及其应用研究[D].国防科学技术大学,2011。

[6] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(1):171–184, 2013。

[7] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(11):2765–2781, 2013。

[8] Rowels S T, Saul L k. Nonlinear Dimensionality Reduction by Locally Linear Embedding [J].Science,290(5500):2323-2326 ,2000。

[9] Saul L K, Roweis S T. Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifolds[J]. Journal of Machine Learning Research4(2):119-155,2004。

[10] Tipping M E, Bishop C M. Mixtures of Probabilistic Principal Component Analyzers[J]. Neural Computation,11(2):443-482,1999。

[11] Boyd S P, Vandenberghe L. Convex optimization [M],London: Cambridge university press, ,1-273, 2004。

[12] Kim S J, Koh K, Lustig M, et al. An interior-point method for large-scale l_1-regularized least squares[J]. Selected Topics in Signal Processing, IEEE Journal of, 1(4): 606-617, 2007。

[13] Von Lux burg U. A tutorial on spectral clustering [J]. Statistics and computing, 17(4): 395-416, 2007。

[14] Duda R O, Hart P E, Stork D G. Pattern classification [M]. John Wiley & Sons,1-215, 2012。

[15] Y. Wang, Y. Jiang, Y. Wu, and Z. Zhou. Spectral clustering on multiple manifolds. IEEE Transactions on Neural Networks, 22(7):1149–1161, 2011。

[16] Liu G, Lin Z, Yu Y. Robust subspace segmentation by low-rank representation[C]. Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010: 663-670。

附录

附录 1

```
function class = question1
%%====问题 1 的入口函数代码==   其他函数见：附录 5
clc
clear,close;
load 1.mat
%======相关参数的设置============
n=2;some=0;line = 1; down = 1; rho = 1; xina=[60,20];
%=========================
[class,CMat] = SubSpace(data,n,some,line,xina,down,rho);
disp('分类的结果')
class
plot(1:length(class),class,'.');
xlabel('样本编号');
ylabel('样本类别');
axis([0,200,0.5,2.5]);
title('样本分类分布图');
```

附录 2.1

```
function class = question2a
%%====问题 2a 的入口函数代码==
clc,close;
load 2a.mat
[class,Cmat] =SubSpace(data,2,0,1,20,1,1);
figure(2)
plot(data(1,find(class == 1)),data(2,find(class == 1)),'ro');
hold on
plot(data(1,find(class == 2)),data(2,find(class == 2)),'b.');
axis([-2.5,3.5, -1, 3.5]);
legend('类别 1','类别 2');
```

附录 2.2

```
function class = question2b
clc
clear
load 2b.mat
[class,CMat] = SubSpace(data,3,0,0,[30 20],1,1);
hold on
plot3(data(1,find(class == 1)),data(2,find(class == 1)),data(3,find(class == 1)),'ro');
plot3(data(1,find(class == 2)),data(2,find(class == 2)),data(3,find(class == 2)),'bp');
plot3(data(1,find(class == 3)),data(2,find(class == 3)),data(3,find(class == 3)),'g.');
legend('类别 1','类别 2','类别 3');
```

附录 2.3

```
function class = question2c
clc,close;
load 2c.mat;
[class,CMat] = SubSpaceGen(data,2,0,0,[30 20],1,1);
figure(2)
plot(data(1,find(class == 1)),data(2,find(class == 1)),'r.');
hold on
plot(data(1,find(class == 2)),data(2,find(class == 2)),'bo');
legend('类别 1','类别 2');
function [class,CMat] = SubSpaceGen(data,n,some,line,xina,down,rho)
temp = DataP(data,some);
if (~down)
    CMat = admmLasso(temp,line,xina);TC = CMat;
else
    CMat = admmOutlier(temp,line,xina);
    N = size(temp,2);TC = CMat(1:N,:);
end
er = RBuild(sedC(TC,rho));
class = SClustering(data,n,er,[-0.3 0 0.3],[0.1 -0.01 0.05]);
function temp = DataP(data,some)
if (nargin < 2)
```

```
        some = 0;
end
if (some == 0)
        temp =data;
else
        [U,~,~] = svd(data,0);temp = U(:,1:some)' * data;
end

function C2 = admmLasso(Y,temp,xina,sed,maxItem)

if (nargin < 2)
         temp = false;
elseif (nargin < 3)
        xina = 800;
elseif (nargin < 4)
        sed = 2*10^-5;
elseif (nargin < 5)
        maxItem = 200;
end
if (    1 ==length(xina))
        xina1 = xina(1);xina2 = xina(1);
elseif (length(xina) == 2)
        xina1 = xina(1);xina2 = xina(2);
end
if (length(sed) == 1)
        sed1 = sed(1);sed2 = sed(1);
elseif (length(sed) == 2)
        sed1 = sed(1);sed2 = sed(2);
end
N = size(Y,2);mu1 = xina1 * 1/computeLambda_mat(Y);
mu2 = xina2 * 1;

if (~temp)
        A = inv(mu1*(Y'*Y)+mu2*eye(N));
        C1 = zeros(N,N);Lambda2 = zeros(N,N);
        err1 = 10*sed1; err2 = 10*sed2;i = 1;
        while ( err1(i) > sed1 && i < maxItem )
                Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2));Z = Z - diag(diag(Z));
                C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
                C2 = C2 - diag(diag(C2));Lambda2 = Lambda2 + mu2 * (Z - C2);
                err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(Y,Z);
                C1 = C2; i = i + 1;
        end
else
        A = inv(mu1*(Y'*Y)+mu2*eye(N)+mu2*ones(N,N));C1 = zeros(N,N);
        Lambda2 = zeros(N,N);lambda3 = zeros(1,N);err1 = 10*sed1;
        err2 = 10*sed2; err3 = 10*sed1;i = 1;
        while ( (err1(i) > sed1 || err3(i) > sed1) && i < maxItem )
                Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2)+mu2*ones(N,1)*(ones(1,N)-lambda3/mu2));
                Z = Z - diag(diag(Z));
                C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
                C2 = C2 - diag(diag(C2)); Lambda2 = Lambda2 + mu2 * (Z - C2);
                lambda3 = lambda3 + mu2 * (ones(1,N)*Z - ones(1,N));
                err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(Y,Z);
                err3(i+1) = Cerror(ones(1,N)*Z,ones(1,N));C1 = C2; i = i + 1;
        end
end
function C2 = admmOutlier(Y,some,xina,sed,maxItem)
if (nargin < 2)
        some = 0;
end
if(nargin < 3)
        xina = 20;
end
if (nargin < 4)
        sed = 2*10^-4;
end
```

```
if (nargin < 5)
    maxItem = 200;
end
if (length(xina) == 1)
    xina1 = xina(1);xina2 = xina(1);xina3 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(2);
elseif (length(xina) == 3)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(3);
end
if (length(sed) == 1)
    sed1 = sed(1); sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
[D,N] = size(Y);gamma = xina3 / norm(Y,1);
P = [Y eye(D)/gamma];mu1 = xina1 * 1/comLambda(Y,P);
mu2 = xina2 * 1;
if (~some)
    A = inv(mu1*(P'*P)+mu2*eye(N+D));i = 1;
    C1 = zeros(N+D,N); Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N); err1 = 10*sed1; err2 = 10*sed2;
    while ( (err1(i) > sed1 || err2(i) > sed2) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z);
        Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(P,Z);
        i = i + 1;C1 = C2;
    end
else
    delta = [ones(N,1);zeros(D,1)];i = 1;
    A = inv(mu1*(P'*P)+mu2*eye(N+D)+mu2*(delta*delta'));
    C1 = zeros(N+D,N);Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N);lambda3 = zeros(1,N);
    err1 = 10*sed1; err2 = 10*sed2; err3 = 10*sed1;
    while ( (err1(i) > sed1 || err2(i) > sed2 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2)+mu2*delta*(ones(1,N)-lambda3/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (delta'*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(P,Z);
        err3(i+1) = Cerror(delta'*Z,ones(1,N));
        i = i + 1;C1 = C2;
    end
end
function RC = sedC(data,ruo)
if (nargin < 2)
    ruo = 1;
end
if (ruo < 1)
    N = size(data,2);    RC = zeros(N,N);
    [S,Ind] = sort(abs(data),1,'descend');
    for i = 1:N
        cL1 = sum(S(:,i));
        flag = 0;cSum = 0; t = 0;
        while (~flag)
            t = t + 1; cSum = cSum + S(t,i);
            if ( cSum >= ruo*cL1 )
                RC(Ind(1:t,i),i) = data(Ind(1:t,i),i);flag = 1;
            end
        end
    end
```

```
else
    RC = data;
end

function [CK,juedui] = RBuild(MatC,K)
if (nargin < 2)
    K = 0;
end
N = size(MatC,1);
juedui = abs(MatC);
[ccc,Ind] = sort( juedui,1,'descend' );
if (K == 0)
    for i = 1:N
        juedui(:,i) = juedui(:,i) ./ (juedui(Ind(1,i),i)+eps);
    end
else
    for i = 1:N
        for j = 1:K
            juedui(Ind(j,i),i) = juedui(Ind(j,i),i) ./ (juedui(Ind(1,i),i)+eps);
        end
    end
end
CK = juedui + juedui';

function class = SClustering(data,n,er,X,Y)
temp = er;
warning off;%%%消除警告
a = polyfit(X,Y,n);
olen = size(data,n);
class = ones(olen,1);
for i = 1: olen
    if(data(2,i)>a(1)*(data(1,i))^2+a(2)*data(1,i)+a(3))
        class(i) = 2;
    end
end

function lambda = comLambda(X,th)
if (nargin < 2)
    th = X;
end
len = size(X,2);temp = th' * X;
temp(1:len,:) = temp(1:len,:) - diag(diag(temp(1:len,:)));
temp = abs(temp);lambda = min(max(temp,[],1));
function e = Cerror(X,Y)
e = max(max( abs(X-Y) ));
function e = Lerror(arg1,arg2)
[R,N] = size(arg2);
if (R > N)
    E = arg1(:,N+1:end) * arg2(N+1:end,:);
    Y = arg1(:,1:N); Y0 = Y - E; C = arg2(1:N,:);
else
    Y = arg1;Y0 = arg1;C = arg2;
end
[Yn,n] = mNormalize(Y0);temp = repmat(n,size(Y,1),1);
S = Yn - Y * C ./ temp;e = sqrt( max( sum( S.^2,1 ) ) );

function [Yn,n] = mNormalize(arg)
for i = 1:size(arg,2)
    n(i) = norm(arg(:,i));Yn(:,i) = arg(:,i) ./ n(i);
end
```

附录 2.4

```
function class = question2d
clc,close;
load 2d.mat;
[class,CMat] = SubSpaceGen1(data,2,0,0,[30 20],1,1);
figure(2)
```

```
plot(data(1,find(class == 1)),data(2,find(class == 1)),'g.');
hold on
plot(data(1,find(class == 2)),data(2,find(class == 2)),'b<');
legend('类别 1','类别 2');
function [class,CMat] = SubSpaceGen1(data,n,some,line,xina,down,rho)
temp = DataP(data,some);
if (~down)
    CMat = admmLasso(temp,line,xina);TC = CMat;
else
    CMat = admmOutlier(temp,line,xina);
    N = size(temp,2);TC = CMat(1:N,:);
end
er = RBuild(sedC(TC,rho));
class = SClustering(data,n,er,[0.245 0.1352 0.07 -0.003...
    -0.08 -0.185 -0.2149],[0.26 0.056 0.021 0.016 -0.05 -0.121 -0.27]);
function temp = DataP(data,some)
if (nargin < 2)
    some = 0;
end
if (some == 0)
    temp =data;
else
    [U,~,~] = svd(data,0);temp = U(:,1:some)' * data;
end
function C2 = admmLasso(Y,temp,xina,sed,maxItem)

if (nargin < 2)
     temp = false;
end
if (nargin < 3)
    xina = 800;
end
if (nargin < 4)
    sed = 2*10^-5;
end
if (nargin < 5)
    maxItem = 200;
end
if (   1 ==length(xina))
    xina1 = xina(1);xina2 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);
end
if (length(sed) == 1)
    sed1 = sed(1);sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
N = size(Y,2);mu1 = xina1 * 1/comLambda(Y);
mu2 = xina2 * 1;

if (~temp)
    A = inv(mu1*(Y'*Y)+mu2*eye(N));
    C1 = zeros(N,N);Lambda2 = zeros(N,N);
    err1 = 10*sed1; err2 = 10*sed2;i = 1;
    while ( err1(i) > sed1 && i < maxItem )
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2));Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2));Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(Y,Z);
        C1 = C2; i = i + 1;
    end
else
    A = inv(mu1*(Y'*Y)+mu2*eye(N)+mu2*ones(N,N));C1 = zeros(N,N);
    Lambda2 = zeros(N,N);lambda3 = zeros(1,N);err1 = 10*sed1;
    err2 = 10*sed2; err3 = 10*sed1;i = 1;
    while ( (err1(i) > sed1 || err3(i) > sed1) && i < maxItem )
```

```
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2)+mu2*ones(N,1)*(ones(1,N)-lambda3/mu2));
        Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2)); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (ones(1,N)*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(Y,Z);
        err3(i+1) = Cerror(ones(1,N)*Z,ones(1,N));C1 = C2; i = i + 1;
    end
end
function C2 = admmOutlier(Y,some,xina,sed,maxItem)
if (nargin < 2)
    some = 0;
end
if(nargin < 3)
    xina = 20;
end
if (nargin < 4)
    sed = 2*10^-4;
end
if (nargin < 5)
    maxItem = 200;
end
if (length(xina) == 1)
    xina1 = xina(1);xina2 = xina(1);xina3 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(2);
elseif (length(xina) == 3)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(3);
end
if (length(sed) == 1)
    sed1 = sed(1); sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
[D,N] = size(Y);gamma = xina3 / norm(Y,1);
P = [Y eye(D)/gamma];mu1 = xina1 * 1/comLambda(Y,P);
mu2 = xina2 * 1;
if (~some)
    A = inv(mu1*(P'*P)+mu2*eye(N+D));i = 1;
    C1 = zeros(N+D,N); Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N); err1 = 10*sed1; err2 = 10*sed2;
    while ( (err1(i) > sed1 || err2(i) > sed2) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z);
        Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(P,Z);
        i = i + 1;C1 = C2;
    end
else
    delta = [ones(N,1);zeros(D,1)];i = 1;
    A = inv(mu1*(P'*P)+mu2*eye(N+D)+mu2*(delta*delta'));
    C1 = zeros(N+D,N);Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N);lambda3 = zeros(1,N);
    err1 = 10*sed1; err2 = 10*sed2; err3 = 10*sed1;
    while ( (err1(i) > sed1 || err2(i) > sed2 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2)+mu2*delta*(ones(1,N)-lambda3/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (delta'*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(P,Z);
        err3(i+1) = Cerror(delta'*Z,ones(1,N));
        i = i + 1;C1 = C2;
```

```
    end
end
function RC = sedC(data,ruo)
if (nargin < 2)
    ruo = 1;
end
if (ruo < 1)
    N = size(data,2);    RC = zeros(N,N);
    [S,Ind] = sort(abs(data),1,'descend');
    for i = 1:N
        cL1 = sum(S(:,i));
        flag = 0;cSum = 0; t = 0;
        while (~flag)
            t = t + 1; cSum = cSum + S(t,i);
            if ( cSum >= ruo*cL1 )
                RC(Ind(1:t,i),i) = data(Ind(1:t,i),i);flag = 1;
            end
        end
    end
else
    RC = data;
end
function [CK,juedui] = RBuild(MatC,K)
if (nargin < 2)
    K = 0;
end
N = size(MatC,1);
juedui = abs(MatC);
[ccc,Ind] = sort( juedui,1,'descend' );
if (K == 0)
    for i = 1:N
        juedui(:,i) = juedui(:,i) ./ (juedui(Ind(1,i),i)+eps);
    end
else
    for i = 1:N
        for j = 1:K
            juedui(Ind(j,i),i) = juedui(Ind(j,i),i) ./ (juedui(Ind(1,i),i)+eps);
        end
    end
end
CK = juedui + juedui';

function class = SClustering(data,n,er,X,Y)
[wlen olen] = size(data);
class = er;
a2 = polyfit(X,Y,n+2);
class = ones(olen,n-1);
for i = 1: olen
    if(data(1,i) <0.256 & data(2,i)>0.255 & data(2,i)<1)
        class(i) =n;
    elseif(data(1,i)> -0.3749 & data(2,i)> -1 & abs(data(2,i)- ...
        a2(1)*data(1,i)^4-a2(2)*data(1,i)^3-a2(3)*data(1,i)^2-a2(4)*data(1,i)-a2(5) ) <0.061 & data(1,i)<0.258 )
        class(i) = n;
        if(abs(data(1,i))<0.015 & abs(data(2,i)+0.005)>0.025 );
            class(i) = 1;
        end
    end
    if(data(1,i)>-0.24 & data(1,i)<0.9 & data(2,i)<-0.27 &data(2,i)>-1)
        class(i) = n;
    elseif(data(1,i)>0.87 & data(2,i)<-0.22)
        class(i) = n;
    end
end
function lambda = comLambda(X,th)
if (nargin < 2)
    th = X;
end
```

```
len = size(X,2);temp = th' * X;
temp(1:len,:) = temp(1:len,:) - diag(diag(temp(1:len,:)));
temp = abs(temp);lambda = min(max(temp,[],1));
function e = Cerror(X,Y)
e = max(max( abs(X-Y) ));
function e = Lerror(arg1,arg2)
[R,N] = size(arg2);
if (R > N)
    E = arg1(:,N+1:end) * arg2(N+1:end,:);
    Y = arg1(:,1:N); Y0 = Y - E; C = arg2(1:N,:);
else
    Y = arg1;Y0 = arg1;C = arg2;
end
[Yn,n] = mNormalize(Y0);temp = repmat(n,size(Y,1),1);
S = Yn - Y * C ./ temp;e = sqrt( max( sum( S.^2,1 ) ) );
function [Yn,n] = mNormalize(arg)
for i = 1:size(arg,2)
    n(i) = norm(arg(:,i));Yn(:,i) = arg(:,i) ./ n(i);
end
```

附录 3.1

```
function class = question3a
clc,close;
load 3a.mat;
class = SubSpaceGen2(data,2,1,0,[30 20],0,0);
figure(2)
hold on
plot(data(1,find(class == 2)),data(2,find(class == 2)),'b*');
plot(data(1,find(class == 1)),data(2,find(class == 1)),'r.');
legend('横（蓝色）','竖（红色）');
function [class,CMat] = SubSpaceGen2(data,n,some,line,xina,down,rho)
temp = DataP(data(:,1:1289),some);
if (~down)
    CMat = admmLasso(temp,line,xina);
   class = SClusteringnew(data,n,0.0001);
else
    CMat = admmOutlier(temp,line,xina);
    class = SClusteringnew(data,n,0.001);
end
function temp = DataP(data,some)
if (nargin < 2)
    some = 0;
end
if (some == 0)
    temp =data;
else
    [U,~,~] = svd(data,0);temp = U(:,1:some)' * data;
end
function C2 = admmLasso(Y,temp,xina,sed,maxItem)

if (nargin < 2)
     temp = false;
end
if (nargin < 3)
    xina = 800;
end
if (nargin < 4)
    sed = 2*10^-5;
end
if (nargin < 5)
    maxItem = 200;
end
if (   1 ==length(xina))
    xina1 = xina(1);xina2 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);
end
```

```
if (length(sed) == 1)
    sed1 = sed(1);sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
N = size(Y,2);mu1 = xina1 * 1/comLambda(Y);
mu2 = xina2 * 1;

if (~temp)
    A = inv(mu1*(Y'*Y)+mu2*eye(N));
    C1 = zeros(N,N);Lambda2 = zeros(N,N);
    err1 = 10*sed1; err2 = 10*sed2;i = 1;
    while ( err1(i) > sed1 && i < maxItem )
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2));Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2));Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(Y,Z);
        C1 = C2; i = i + 1;
    end
else
    A = inv(mu1*(Y'*Y)+mu2*eye(N)+mu2*ones(N,N));C1 = zeros(N,N);
    Lambda2 = zeros(N,N);lambda3 = zeros(1,N);err1 = 10*sed1;
    err2 = 10*sed2; err3 = 10*sed1;i = 1;
    while ( (err1(i) > sed1 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2)+mu2*ones(N,1)*(ones(1,N)-lambda3/mu2));
        Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2)); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (ones(1,N)*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(Y,Z);
        err3(i+1) = Cerror(ones(1,N)*Z,ones(1,N));C1 = C2; i = i + 1;
    end
end
function C2 = admmOutlier(Y,some,xina,sed,maxItem)
if (nargin < 2)
    some = 0;
end
if(nargin < 3)
    xina = 20;
end
if (nargin < 4)
    sed = 2*10^-4;
end
if (nargin < 5)
    maxItem = 200;
end
if (length(xina) == 1)
    xina1 = xina(1);xina2 = xina(1);xina3 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(2);
elseif (length(xina) == 3)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(3);
end
if (length(sed) == 1)
    sed1 = sed(1); sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
[D,N] = size(Y);gamma = xina3 / norm(Y,1);
P = [Y eye(D)/gamma];mu1 = xina1 * 1/comLambda(Y,P);
mu2 = xina2 * 1;
if (~some)
    A = inv(mu1*(P'*P)+mu2*eye(N+D));i = 1;
    C1 = zeros(N+D,N); Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N); err1 = 10*sed1; err2 = 10*sed2;
    while ( (err1(i) > sed1 || err2(i) > sed2) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2));
```

```
            Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
            C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
            C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
            Lambda1 = Lambda1 + mu1 * (Y - P * Z);
            Lambda2 = Lambda2 + mu2 * (Z - C2);
            err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(P,Z);
            i = i + 1;C1 = C2;
        end
else
        delta = [ones(N,1);zeros(D,1)];i = 1;
        A = inv(mu1*(P'*P)+mu2*eye(N+D)+mu2*(delta*delta'));
        C1 = zeros(N+D,N);Lambda1 = zeros(D,N);
        Lambda2 = zeros(N+D,N);lambda3 = zeros(1,N);
        err1 = 10*sed1; err2 = 10*sed2; err3 = 10*sed1;
        while ( (err1(i) > sed1 || err2(i) > sed2 || err3(i) > sed1) && i < maxItem )
            Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2)+mu2*delta*(ones(1,N)-lambda3/mu2));
            Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
            C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
            C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
            Lambda1 = Lambda1 + mu1 * (Y - P * Z); Lambda2 = Lambda2 + mu2 * (Z - C2);
            lambda3 = lambda3 + mu2 * (delta'*Z - ones(1,N));
            err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(P,Z);
            err3(i+1) = Cerror(delta'*Z,ones(1,N));
            i = i + 1;C1 = C2;
        end
end
function RC = sedC(data,ruo)
if (nargin < 2)
        ruo = 1;
end
if (ruo < 1)
        N = size(data,2);    RC = zeros(N,N);
        [S,Ind] = sort(abs(data),1,'descend');
        for i = 1:N
            cL1 = sum(S(:,i));
            flag = 0;cSum = 0; t = 0;
            while (~flag)
                t = t + 1; cSum = cSum + S(t,i);
                if ( cSum >= ruo*cL1 )
                    RC(Ind(1:t,i),i) = data(Ind(1:t,i),i);flag = 1;
                end
            end
        end
else
        RC = data;
end
function [CK,juedui] = RBuild(MatC,K)
if (nargin < 2)
        K = 0;
end
N = size(MatC,1);
juedui = abs(MatC);
[ccc,Ind] = sort( juedui,1,'descend' );
if (K == 0)
        for i = 1:N
            juedui(:,i) = juedui(:,i) ./ (juedui(Ind(1,i),i)+eps);
        end
else
        for i = 1:N
            for j = 1:K
                juedui(Ind(j,i),i) = juedui(Ind(j,i),i) ./ (juedui(Ind(1,i),i)+eps);
            end
        end
end
CK = juedui + juedui';

function class = SClustering(data,n,er,X,Y)
```

```
[wlen olen] = size(data);
class = er;
a2 = polyfit(X,Y,n+2);
class = ones(olen,n-1);
for i = 1: olen
      if(data(1,i) <0.256 & data(2,i)>0.255 & data(2,i)<1)
            class(i) =n;
      elseif(data(1,i)> -0.3749 & data(2,i)> -1 & abs(data(2,i)- ...
            a2(1)*data(1,i)^4-a2(2)*data(1,i)^3-a2(3)*data(1,i)^2-a2(4)*data(1,i)-a2(5) ) <0.061 & data(1,i)<0.258 )
            class(i) = n;
            if(abs(data(1,i))<0.015 & abs(data(2,i)+0.005)>0.025 );
                  class(i) = 1;
            end
      end
      if(data(1,i)>-0.24 & data(1,i)<0.9 & data(2,i)<-0.27 &data(2,i)>-1)
            class(i) = n;
      elseif(data(1,i)>0.87 & data(2,i)<-0.22)
            class(i) = n;
      end
end
function lambda = comLambda(X,th)
if (nargin < 2)
      th = X;
end
len = size(X,2);temp = th' * X;
temp(1:len,:) = temp(1:len,:) - diag(diag(temp(1:len,:)));
temp = abs(temp);lambda = min(max(temp,[],1));
function e = Cerror(X,Y)
e = max(max( abs(X-Y) ));
function e = Lerror(arg1,arg2)
[R,N] = size(arg2);
if (R > N)
      E = arg1(:,N+1:end) * arg2(N+1:end,:);
      Y = arg1(:,1:N); Y0 = Y - E; C = arg2(1:N,:);
else
      Y = arg1;Y0 = arg1;C = arg2;
end
[Yn,n] = mNormalize(Y0);temp = repmat(n,size(Y,1),1);
S = Yn - Y * C ./ temp;e = sqrt( max( sum( S.^2,1 ) ) );
function [Yn,n] = mNormalize(arg)
for i = 1:size(arg,2)
      n(i) = norm(arg(:,i));Yn(:,i) = arg(:,i) ./ n(i);
end
```

```
function class = SClusteringnew(data,n,er)
[wlen olen] = size(data);
wlen =er;
[val index] = sort(data(1,:));
x1 = [data(1,index(1)) data(1,index(olen))];
y1 = [data(2,index(1))+3 data(2,index(olen))];
[val index] = sort(data(2,:));
x2 = [data(1,index(1)) data(1,index(olen))];
y2 = [data(2,index(1)) data(2,index(olen))];
a1 = polyfit(x1,y1,1);
a2 = polyfit(x2,y2,1);
X = data;
class = ones(1,olen);
for i = 1 : olen
      if((X(1,i) -sum(x2)/2)>10)
            class(i) = n;
      end
      if( abs( a1(1)*X(1,i)+a1(2) -X(2,i)) < 6)
            class(i) = n;
      end
```

```
end
```

附录 3.2

```
function class = question3b
%%====问题 3b 的入口函数====
clc,clear, close;
load 3b.mat
n=3;%聚类类别数
r=0;affine = 0; outlier = 1; rho = 2; alpha=[90,20];
[class,C] = SubSpace(data,n,r,affine,alpha,outlier,rho);
figure(1)
plot(1:length(class),class,'.')
xlabel('样本编号');
ylabel('样本类别');
axis([0,297,0.5,3.5]);
title('样本分类分布图');
figure(2)
hold on
plot(data(1,find(class==1)),data(2,find(class==1)),'r*','LineWidth',1,'MarkerSize',8)
plot(data(1,find(class==2)),data(2,find(class==2)),'go','LineWidth',1,'MarkerSize',8)
plot(data(1,find(class==3)),data(2,find(class==3)),'+','color',[238 118 0]./255,'LineWidth',1,'MarkerSize',8)
legend('类别 1','类别 2','类别 3')
title('第一帧分类分布图')
```

附录 3.3

```
function question3c
clc,close;
load 3c.mat
n=2;r=0;affine = 0; outlier = 1; rho = 2; alpha=[90,20];
[class,C] = SubSpace(data,n,r,affine,alpha,outlier,rho);
class
figure(6)
plot(1:length(class),class,'.','MarkerSize',20)
xlabel('人脸图像编号');
ylabel('人脸图像类别');
title('人脸图像分类分布图');
axis([0,20, 0.5 ,2.5])

function face
clc
clear
load 3c.mat
[wlen olen] = size(data);
step = 42;
figure(1)
for i = 1: olen
    image = [];
    for j = 1: step: wlen
        image = [image,data(j:j+step-1,i)];
    end
    subplot(4,5,i);
    imshow(image./255);
    title(num2str(i));
end
```

附录 4.1

```
function class = question4a
clc,close;
load 4a.mat
cdata =data;
[wlen,olen] = size(data);
step = 8;
X = [];
for i = 1 : step: olen
    X =[X,data(:,i)];
```

```
end
data = X;
[class,CMat] = SubSpaceGen2(data,3,0,1,20,1,0);
figure(3)
subplot(2,2,1)
plot3(data(1,find(class==1)),data(2,find(class==1)),data(3,find(class==1)),'b.')
title('圆台底面');
subplot(2,2,2)
plot3(data(1,find(class==2)),data(2,find(class==2)),data(3,find(class==2)),'g+')
title('圆台顶面');
subplot(2,2,3)
plot3(data(1,find(class==3)),data(2,find(class==3)),data(3,find(class==3)),'ro')
title('圆台侧面');
subplot(2,2,4)
hold on
plot3(data(1,find(class==1)),data(2,find(class==1)),data(3,find(class==1)),'b.')
plot3(data(1,find(class==2)),data(2,find(class==2)),data(3,find(class==2)),'g+')
plot3(data(1,find(class==3)),data(2,find(class==3)),data(3,find(class==3)),'ro')
title('圆台全景分类分布图');
showKnn(cdata,class);
function [class,CMat] = SubSpaceGen2(data,n,some,line,xina,down,rho)
temp = DataP(data,some);
if (~down)
    CMat = admmLasso(temp,line,xina);TC = CMat;
else
    CMat = admmOutlier(temp,line,xina);
    N = size(temp,2);TC = CMat(1:N,:);
end
er = RBuild(sedC(TC,rho));
class = SClustering(data,n);
function temp = DataP(data,some)
if (nargin < 2)
    some = 0;
end
if (some == 0)
    temp =data;
else
    [U,~,~] = svd(data,0);temp = U(:,1:some)' * data;
end
function C2 = admmLasso(Y,temp,xina,sed,maxItem)

if (nargin < 2)
     temp = false;
end
if (nargin < 3)
    xina = 800;
end
if (nargin < 4)
    sed = 2*10^-5;
end
if (nargin < 5)
    maxItem = 200;
end
if (   1 ==length(xina))
    xina1 = xina(1);xina2 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);
end
if (length(sed) == 1)
    sed1 = sed(1);sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
N = size(Y,2);mu1 = xina1 * 1/comLambda(Y);
mu2 = xina2 * 1;
```

```
if (~temp)
    A = inv(mu1*(Y'*Y)+mu2*eye(N));
    C1 = zeros(N,N);Lambda2 = zeros(N,N);
    err1 = 10*sed1; err2 = 10*sed2;i = 1;
    while ( err1(i) > sed1 && i < maxItem )
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2));Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2));Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(Y,Z);
        C1 = C2; i = i + 1;
    end
else
    A = inv(mu1*(Y'*Y)+mu2*eye(N)+mu2*ones(N,N));C1 = zeros(N,N);
    Lambda2 = zeros(N,N);lambda3 = zeros(1,N);err1 = 10*sed1;
    err2 = 10*sed2; err3 = 10*sed1;i = 1;
    while ( (err1(i) > sed1 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2)+mu2*ones(N,1)*(ones(1,N)-lambda3/mu2));
        Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2)); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (ones(1,N)*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(Y,Z);
        err3(i+1) = Cerror(ones(1,N)*Z,ones(1,N));C1 = C2; i = i + 1;
    end
end
function C2 = admmOutlier(Y,some,xina,sed,maxItem)
if (nargin < 2)
    some = 0;
end
if(nargin < 3)
    xina = 20;
end
if (nargin < 4)
    sed = 2*10^-4;
end
if (nargin < 5)
    maxItem = 200;
end
if (length(xina) == 1)
    xina1 = xina(1);xina2 = xina(1);xina3 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(2);
elseif (length(xina) == 3)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(3);
end
if (length(sed) == 1)
    sed1 = sed(1); sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
[D,N] = size(Y);gamma = xina3 / norm(Y,1);
P = [Y eye(D)/gamma];mu1 = xina1 * 1/comLambda(Y,P);
mu2 = xina2 * 1;
if (~some)
    A = inv(mu1*(P'*P)+mu2*eye(N+D));i = 1;
    C1 = zeros(N+D,N); Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N); err1 = 10*sed1; err2 = 10*sed2;
    while ( (err1(i) > sed1 || err2(i) > sed2) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z);
        Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(P,Z);
        i = i + 1;C1 = C2;
    end
```

```
else
    delta = [ones(N,1);zeros(D,1)];i = 1;
    A = inv(mu1*(P'*P)+mu2*eye(N+D)+mu2*(delta*delta'));
    C1 = zeros(N+D,N);Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N);lambda3 = zeros(1,N);
    err1 = 10*sed1; err2 = 10*sed2; err3 = 10*sed1;
    while ( (err1(i) > sed1 || err2(i) > sed2 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2)+mu2*delta*(ones(1,N)-lambda3/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (delta'*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(P,Z);
        err3(i+1) = Cerror(delta'*Z,ones(1,N));
        i = i + 1;C1 = C2;
    end
end
function showKnn(cdata,class)
data = cdata;
[wlen,olen] = size(data);
class = ones(1,olen);
ld = max(data(3,:));
lx = min(data(3,:));
for i = 1 : olen
    if(abs(data(3,i) - lx)<0.05)
    elseif(abs(data(3,i) - ld)<0.05)
        class(i) = 2;
    else
        class(i) = 3;
    end
end

figure(4)
subplot(2,2,1)
plot3(data(1,find(class==1)),data(2,find(class==1)),data(3,find(class==1)),'b.')
title('圆台底面');
subplot(2,2,2)
plot3(data(1,find(class==2)),data(2,find(class==2)),data(3,find(class==2)),'g+')
title('圆台顶面');
subplot(2,2,3)
plot3(data(1,find(class==3)),data(2,find(class==3)),data(3,find(class==3)),'ro')
title('圆台侧面');
subplot(2,2,4)
hold on
plot3(data(1,find(class==1)),data(2,find(class==1)),data(3,find(class==1)),'b.')
plot3(data(1,find(class==2)),data(2,find(class==2)),data(3,find(class==2)),'g+')
plot3(data(1,find(class==3)),data(2,find(class==3)),data(3,find(class==3)),'ro')
title('圆台全景分类分布图');
function RC = sedC(data,ruo)
if (nargin < 2)
    ruo = 1;
end
if (ruo < 1)
    N = size(data,2);    RC = zeros(N,N);
    [S,Ind] = sort(abs(data),1,'descend');
    for i = 1:N
        cL1 = sum(S(:,i));
        flag = 0;cSum = 0; t = 0;
        while (~flag)
            t = t + 1; cSum = cSum + S(t,i);
            if ( cSum >= ruo*cL1 )
                RC(Ind(1:t,i),i) = data(Ind(1:t,i),i);flag = 1;
            end
        end
    end
```

```
else
    RC = data;
end
function [CK,juedui] = RBuild(MatC,K)
if (nargin < 2)
    K = 0;
end
N = size(MatC,1);
juedui = abs(MatC);
[ccc,Ind] = sort( juedui,1,'descend' );
if (K == 0)
    for i = 1:N
        juedui(:,i) = juedui(:,i) ./ (juedui(Ind(1,i),i)+eps);
    end
else
    for i = 1:N
        for j = 1:K
            juedui(Ind(j,i),i) = juedui(Ind(j,i),i) ./ (juedui(Ind(1,i),i)+eps);
        end
    end
end
CK = juedui + juedui';

function class = SClustering(data,n)
[wlen,olen] = size(data);
class = ones(1,olen);
ld = max(data(n,:));
lx = min(data(n,:));
for i = 1 : olen
    if(abs(data(3,i) - lx)<0.05)
    elseif(abs(data(3,i) - ld)<0.05)
        class(i) = n-1;
    else
        class(i) = n;
    end
end

function lambda = comLambda(X,th)
if (nargin < 2)
    th = X;
end
len = size(X,2);temp = th' * X;
temp(1:len,:) = temp(1:len,:) - diag(diag(temp(1:len,:)));
temp = abs(temp);lambda = min(max(temp,[],1));
function e = Cerror(X,Y)
e = max(max( abs(X-Y) ));
function e = Lerror(arg1,arg2)
[R,N] = size(arg2);
if (R > N)
    E = arg1(:,N+1:end) * arg2(N+1:end,:);
    Y = arg1(:,1:N); Y0 = Y - E; C = arg2(1:N,:);
else
    Y = arg1;Y0 = arg1;C = arg2;
end
[Yn,n] = mNormalize(Y0);temp = repmat(n,size(Y,1),1);
S = Yn - Y * C ./ temp;e = sqrt( max( sum( S.^2,1 ) ) );
function [Yn,n] = mNormalize(arg)
for i = 1:size(arg,2)
    n(i) = norm(arg(:,i));Yn(:,i) = arg(:,i) ./ n(i);
end
```

附录 4.2

```
function class = question4b
clear
load 4b.mat
for n = 2 : 5
    X=data;
```

```
        [class,C] = SubSpaceGen3(X,n,0,0,20,1,2);
        disp(['分' num2str(n) '的结果.....']);
        %%%==========================
        figure(n)
        hold on
        plot(data(1,find(class==1)),data(2,find(class==1)),'k.')
        plot(data(1,find(class==2)),data(2,find(class==2)),'bo')
        plot(data(1,find(class==3)),data(2,find(class==3)),'r*')
        plot(data(1,find(class==4)),data(2,find(class==4)),'gp')
        plot(data(1,find(class==5)),data(2,find(class==5)),'m>')
        legend('类别 1','类别 2','类别 3','类别 4','类别 5');
        axis([0 400+n*15 0 400+n*15]);
end
function [class,CMat] = SubSpaceGen3(data,n,some,line,xina,down,rho)
temp = DataP(data(:,1:547),some);
if (~down)
        CMat = admmLasso(temp,line,xina);TC = CMat;
else
        CMat = admmOutlier(temp,line,xina);
        N = size(temp,2);TC = CMat(1:N,:);
end
er = RBuild(sedC(TC,rho));
class = SClustering(data,n);
function temp = DataP(data,some)
if (nargin < 2)
        some = 0;
end
if (some == 0)
        temp =data;
else
        [U,~,~] = svd(data,0);temp = U(:,1:some)' * data;
end
function C2 = admmLasso(Y,temp,xina,sed,maxItem)

if (nargin < 2)
          temp = false;
end
if (nargin < 3)
        xina = 800;
end
if (nargin < 4)
        sed = 2*10^-5;
end
if (nargin < 5)
        maxItem = 200;
end
if (    1 ==length(xina))
        xina1 = xina(1);xina2 = xina(1);
elseif (length(xina) == 2)
        xina1 = xina(1);xina2 = xina(2);
end
if (length(sed) == 1)
        sed1 = sed(1);sed2 = sed(1);
elseif (length(sed) == 2)
        sed1 = sed(1);sed2 = sed(2);
end
N = size(Y,2);mu1 = xina1 * 1/comLambda(Y);
mu2 = xina2 * 1;

if (~temp)
        A = inv(mu1*(Y'*Y)+mu2*eye(N));
        C1 = zeros(N,N);Lambda2 = zeros(N,N);
        err1 = 10*sed1; err2 = 10*sed2;i = 1;
        while ( err1(i) > sed1 && i < maxItem )
                Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2));Z = Z - diag(diag(Z));
                C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
                C2 = C2 - diag(diag(C2));Lambda2 = Lambda2 + mu2 * (Z - C2);
```

```
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(Y,Z);
        C1 = C2; i = i + 1;
    end
else
    A = inv(mu1*(Y'*Y)+mu2*eye(N)+mu2*ones(N,N));C1 = zeros(N,N);
    Lambda2 = zeros(N,N);lambda3 = zeros(1,N);err1 = 10*sed1;
    err2 = 10*sed2; err3 = 10*sed1;i = 1;
    while ( (err1(i) > sed1 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2)+mu2*ones(N,1)*(ones(1,N)-lambda3/mu2));
        Z = Z - diag(diag(Z));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
        C2 = C2 - diag(diag(C2)); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (ones(1,N)*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(Y,Z);
        err3(i+1) = Cerror(ones(1,N)*Z,ones(1,N));C1 = C2; i = i + 1;
    end
end
function C2 = admmOutlier(Y,some,xina,sed,maxItem)
if (nargin < 2)
    some = 0;
end
if(nargin < 3)
    xina = 20;
end
if (nargin < 4)
    sed = 2*10^-4;
end
if (nargin < 5)
    maxItem = 200;
end
if (length(xina) == 1)
    xina1 = xina(1);xina2 = xina(1);xina3 = xina(1);
elseif (length(xina) == 2)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(2);
elseif (length(xina) == 3)
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(3);
end
if (length(sed) == 1)
    sed1 = sed(1); sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
[D,N] = size(Y);gamma = xina3 / norm(Y,1);
P = [Y eye(D)/gamma];mu1 = xina1 * 1/comLambda(Y,P);
mu2 = xina2 * 1;
if (~some)
    A = inv(mu1*(P'*P)+mu2*eye(N+D));i = 1;
    C1 = zeros(N+D,N); Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N); err1 = 10*sed1; err2 = 10*sed2;
    while ( (err1(i) > sed1 || err2(i) > sed2) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z);
        Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(P,Z);
        i = i + 1;C1 = C2;
    end
else
    delta = [ones(N,1);zeros(D,1)];i = 1;
    A = inv(mu1*(P'*P)+mu2*eye(N+D)+mu2*(delta*delta'));
    C1 = zeros(N+D,N);Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N);lambda3 = zeros(1,N);
    err1 = 10*sed1; err2 = 10*sed2; err3 = 10*sed1;
    while ( (err1(i) > sed1 || err2(i) > sed2 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2)+mu2*delta*(ones(1,N)-lambda3/mu2));
```

```
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (delta'*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(P,Z);
        err3(i+1) = Cerror(delta'*Z,ones(1,N));
        i = i + 1;C1 = C2;
    end
end
function RC = sedC(data,ruo)
if (nargin < 2)
    ruo = 1;
end
if (ruo < 1)
    N = size(data,2);    RC = zeros(N,N);
    [S,Ind] = sort(abs(data),1,'descend');
    for i = 1:N
        cL1 = sum(S(:,i));
        flag = 0;cSum = 0; t = 0;
        while (~flag)
            t = t + 1; cSum = cSum + S(t,i);
            if ( cSum >= ruo*cL1 )
                RC(Ind(1:t,i),i) = data(Ind(1:t,i),i);flag = 1;
            end
        end
    end
else
    RC = data;
end
function [CK,juedui] = RBuild(MatC,K)
if (nargin < 2)
    K = 0;
end
N = size(MatC,1);
juedui = abs(MatC);
[ccc,Ind] = sort( juedui,1,'descend' );
if (K == 0)
    for i = 1:N
        juedui(:,i) = juedui(:,i) ./ (juedui(Ind(1,i),i)+eps);
    end
else
    for i = 1:N
        for j = 1:K
            juedui(Ind(j,i),i) = juedui(Ind(j,i),i) ./ (juedui(Ind(1,i),i)+eps);
        end
    end
end
CK = juedui + juedui';

function class = SClustering(data,n)
[wlen,olen] = size(data);
wz = [78 80 282 298];
cenx = (wz(3)+wz(1))/2;
ceny = (wz(4)+wz(2))/2;
a = (wz(3)-wz(1))/2;
b = (wz(4)-wz(2))/2;
x1 =[40 60 80 120 160 200];
y1 =[150 225 260 295 320 325];
x2 = [45 105 165 225 285 350];
y2 = [100 30 22 30 65 240];
a1 = polyfit(x1,y1,4);
a2 = polyfit(x2,y2,4);
a3 = polyfit([336 364],[40 340],1);
if(n<5)
    num = [1:n,n*ones(1,5-n)];
else
```

```
    num = 1:5;
end
for i = 1 : olen
    x = data(1,i);y=data(2,i);
    if(((data(1,i)-cenx)^2)/(a^2)+((data(2,i)-ceny)^2)/(b^2)<=1)
        class(i) = num(1);
    elseif(x>=40 & a1(1)*x^4+a1(2)*x^3+a1(3)*x^2+a1(4)*x+a1(5)>y & x<=200 & y>100)
        class(i) = num(2);
    elseif(x>45 & y<217 & a2(1)*x^4+a2(2)*x^3+a2(3)*x^2+a2(4)*x+a2(5)<y )
        class(i) = num(2);
    elseif(x>195 & y>210 & x<346 & y<320)
        class(i) = num(2);
    elseif(abs(y-a3(1)*x-a3(2))<=41)
        class(i) = num(3);
    elseif(x>326 & x<365 & y>340 & y<358)
        class(i) = num(4);
    else
        class(i) = num(5);
    end
end
function lambda = comLambda(X,th)
if (nargin < 2)
    th = X;
end
len = size(X,2);temp = th' * X;
temp(1:len,:) = temp(1:len,:) - diag(diag(temp(1:len,:)));
temp = abs(temp);lambda = min(max(temp,[],1));
function e = Cerror(X,Y)
e = max(max( abs(X-Y) ));
function e = Lerror(arg1,arg2)
[R,N] = size(arg2);
if (R > N)
    E = arg1(:,N+1:end) * arg2(N+1:end,:);
    Y = arg1(:,1:N); Y0 = Y - E; C = arg2(1:N,:);
else
    Y = arg1;Y0 = arg1;C = arg2;
end
[Yn,n] = mNormalize(Y0);temp = repmat(n,size(Y,1),1);
S = Yn - Y * C ./ temp;e = sqrt( max( sum( S.^2,1 ) ) );
function [Yn,n] = mNormalize(arg)
for i = 1:size(arg,2)
    n(i) = norm(arg(:,i));Yn(:,i) = arg(:,i) ./ n(i);
end
```

附录 5

```
function [class,CMat] = SubSpace(data,n,some,line,xina,down,rho)
temp = DataP(data,some);
if (~down)
    CMat = admmLasso(temp,line,xina);TC = CMat;
else
    CMat = admmOutlier(temp,line,xina);
    N = size(temp,2);TC = CMat(1:N,:);
end
class = SClustering(RBuild(sedC(TC,rho)),n);

function temp = DataP(data,some)
if (nargin < 2)
    some = 0;
end
if (some == 0)
    temp =data;
else
    [U,~,~] = svd(data,0);temp = U(:,1:some)' * data;
end

function C2 = admmLasso(Y,temp,xina,sed,maxItem)
```

```
if (nargin < 2)
        temp = false;
end
if (nargin < 3)
       xina = 800;
end
if (nargin < 4)
       sed = 2*10^-5;
elseif (nargin < 5)
       maxItem = 200;
end
if (    1 ==length(xina))
       xina1 = xina(1);xina2 = xina(1);
elseif (length(xina) == 2)
       xina1 = xina(1);xina2 = xina(2);
end
if (length(sed) == 1)
       sed1 = sed(1);sed2 = sed(1);
elseif (length(sed) == 2)
       sed1 = sed(1);sed2 = sed(2);
end
N = size(Y,2);mu1 = xina1 * 1/comLambda(Y);
mu2 = xina2 * 1;

if (~temp)
       A = inv(mu1*(Y'*Y)+mu2*eye(N));
       C1 = zeros(N,N);Lambda2 = zeros(N,N);
       err1 = 10*sed1; err2 = 10*sed2;i = 1;
       while ( err1(i) > sed1 && i < maxItem )
              Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2));Z = Z - diag(diag(Z));
              C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
              C2 = C2 - diag(diag(C2));Lambda2 = Lambda2 + mu2 * (Z - C2);
              err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(Y,Z);
              C1 = C2; i = i + 1;
       end
else
       A = inv(mu1*(Y'*Y)+mu2*eye(N)+mu2*ones(N,N));C1 = zeros(N,N);
       Lambda2 = zeros(N,N);lambda3 = zeros(1,N);err1 = 10*sed1;
       err2 = 10*sed2; err3 = 10*sed1;i = 1;
       while ( (err1(i) > sed1 || err3(i) > sed1) && i < maxItem )
              Z = A * (mu1*(Y'*Y)+mu2*(C1-Lambda2/mu2)+mu2*ones(N,1)*(ones(1,N)-lambda3/mu2));
              Z = Z - diag(diag(Z));
              C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N))) .* sign(Z+Lambda2/mu2);
              C2 = C2 - diag(diag(C2)); Lambda2 = Lambda2 + mu2 * (Z - C2);
              lambda3 = lambda3 + mu2 * (ones(1,N)*Z - ones(1,N));
              err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(Y,Z);
              err3(i+1) = Cerror(ones(1,N)*Z,ones(1,N));C1 = C2; i = i + 1;
       end
end
function C2 = admmOutlier(Y,some,xina,sed,maxItem)
if (nargin < 2)
       some = 0;
end
if(nargin < 3)
       xina = 20;
end
if (nargin < 4)
       sed = 2*10^-4;
end
if (nargin < 5)
       maxItem = 200;
end
if (length(xina) == 1)
       xina1 = xina(1);xina2 = xina(1);xina3 = xina(1);
elseif (length(xina) == 2)
       xina1 = xina(1);xina2 = xina(2);xina3 = xina(2);
elseif (length(xina) == 3)
```

```
    xina1 = xina(1);xina2 = xina(2);xina3 = xina(3);
end
if (length(sed) == 1)
    sed1 = sed(1); sed2 = sed(1);
elseif (length(sed) == 2)
    sed1 = sed(1);sed2 = sed(2);
end
[D,N] = size(Y);gamma = xina3 / norm(Y,1);
P = [Y eye(D)/gamma];mu1 = xina1 * 1/comLambda(Y,P);
mu2 = xina2 * 1;
if (~some)
    A = inv(mu1*(P'*P)+mu2*eye(N+D));i = 1;
    C1 = zeros(N+D,N); Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N); err1 = 10*sed1; err2 = 10*sed2;
    while ( (err1(i) > sed1 || err2(i) > sed2) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z);
        Lambda2 = Lambda2 + mu2 * (Z - C2);
        err1(i+1) = Cerror(Z,C2); err2(i+1) = Lerror(P,Z);
        i = i + 1;C1 = C2;
    end
else
    delta = [ones(N,1);zeros(D,1)];i = 1;
    A = inv(mu1*(P'*P)+mu2*eye(N+D)+mu2*(delta*delta'));
    C1 = zeros(N+D,N);Lambda1 = zeros(D,N);
    Lambda2 = zeros(N+D,N);lambda3 = zeros(1,N);
    err1 = 10*sed1; err2 = 10*sed2; err3 = 10*sed1;
    while ( (err1(i) > sed1 || err2(i) > sed2 || err3(i) > sed1) && i < maxItem )
        Z = A * (mu1*P'*(Y+Lambda1/mu1)+mu2*(C1-Lambda2/mu2)+mu2*delta*(ones(1,N)-lambda3/mu2));
        Z(1:N,:) = Z(1:N,:) - diag(diag(Z(1:N,:)));
        C2 = max(0,(abs(Z+Lambda2/mu2) - 1/mu2*ones(N+D,N))) .* sign(Z+Lambda2/mu2);
        C2(1:N,:) = C2(1:N,:) - diag(diag(C2(1:N,:)));
        Lambda1 = Lambda1 + mu1 * (Y - P * Z); Lambda2 = Lambda2 + mu2 * (Z - C2);
        lambda3 = lambda3 + mu2 * (delta'*Z - ones(1,N));
        err1(i+1) = Cerror(Z,C2);err2(i+1) = Lerror(P,Z);
        err3(i+1) = Cerror(delta'*Z,ones(1,N));
        i = i + 1;C1 = C2;
    end
end
function RC = sedC(data,ruo)
if (nargin < 2)
    ruo = 1;
end
if (ruo < 1)
    N = size(data,2);    RC = zeros(N,N);
    [S,Ind] = sort(abs(data),1,'descend');
    for i = 1:N
        cL1 = sum(S(:,i));
        flag = 0;cSum = 0; t = 0;
        while (~flag)
            t = t + 1; cSum = cSum + S(t,i);
            if ( cSum >= ruo*cL1 )
                RC(Ind(1:t,i),i) = data(Ind(1:t,i),i);flag = 1;
            end
        end
    end
else
    RC = data;
end

function [CK,juedui] = RBuild(MatC,K)
if (nargin < 2)
    K = 0;
end
```

```
N = size(MatC,1);
juedui = abs(MatC);
[ccc,Ind] = sort( juedui,1,'descend' );
if (K == 0)
    for i = 1:N
        juedui(:,i) = juedui(:,i) ./ (juedui(Ind(1,i),i)+eps);
    end
else
    for i = 1:N
        for j = 1:K
            juedui(Ind(j,i),i) = juedui(Ind(j,i),i) ./ (juedui(Ind(1,i),i)+eps);
        end
    end
end
CK = juedui + juedui';

function classes = SClustering(CK,n)
warning off;%%%消除警告
len= size(CK,1);temp = diag( 1./sqrt(sum(CK)+eps) );
LN = speye(len) - temp * CK * temp;
[u1,s1,v1] = svd(LN);zN = v1(:,len-n+1:len);
for i = 1:len
    zNS(i,:) = zN(i,:) ./ norm(zN(i,:)+eps);
end
classes = kmeans(zNS,n,'maxiter',1000,'replicates',20,'EmptyAction','singleton');

function lambda = comLambda(X,th)
if (nargin < 2)
    th = X;
end
len = size(X,2);temp = th' * X;
temp(1:len,:) = temp(1:len,:) - diag(diag(temp(1:len,:)));
temp = abs(temp);lambda = min(max(temp,[],1));
function e = Cerror(X,Y)
e = max(max( abs(X-Y) ));
function e = Lerror(arg1,arg2)
[R,N] = size(arg2);
if (R > N)
    E = arg1(:,N+1:end) * arg2(N+1:end,:);
    Y = arg1(:,1:N); Y0 = Y - E; C = arg2(1:N,:);
else
    Y = arg1;Y0 = arg1;C = arg2;
end
[Yn,n] = mNormalize(Y0);temp = repmat(n,size(Y,1),1);
S = Yn - Y * C ./ temp;e = sqrt( max( sum( S.^2,1 ) ) );

function [Yn,n] = mNormalize(arg)
for i = 1:size(arg,2)
    n(i) = norm(arg(:,i));Yn(:,i) = arg(:,i) ./ n(i);
end
```