引用格式：何选森, 何帆, 薄喜柱, 等. 亲和力传播聚类算法中最佳聚类数量的确定 [J]. 电子科技大学学报, 2025, 54(5): 740-754.
HE X S, HE F, BO X Z, et al. Determination of the optimal number of clusters in affinity propagation clustering algorithm[J]. Journal of University of Electronic Science and Technology of China, 2025, 54(5): 740-754.

亲和力传播聚类算法中最佳聚类数量的确定

何选森[1,2]，何　帆[3*]，薄喜柱[1]，肖湘萍[1]

（1. 广州商学院 鸿蒙研究院，广州 511363；2. 湖南大学 信息科学与工程学院，长沙 410082；3. 浙江外国语学院 国际商学院，杭州 310012）

摘要：亲和力传播（AP）聚类能自动搜索聚类数量和聚类中心，但它提供的聚类数量与数据固有的聚类结构相差较大。为此，提出一种确定数据集潜在聚类数量的方法。利用任意两个数据点的欧氏距离平方构成相似性矩阵，以数据样本容量和相似性矩阵中非对角元素的中位数为参数，建立偏好的更新公式以确定聚类数量；将相似性与可用性相加构成亲和矩阵，并将亲和矩阵中取正值的主对角元素作为聚类的质心，以实现聚类数量与质心数量的相互验证。通过对随机数据集以及真实数据集的仿真，对多种性能度量以及算法的运行时间进行评估，结果说明该方法不仅能准确地估计聚类的数量，而且能有效地加快算法的收敛，从而适应于大数据应用的要求。

关键词：亲和力传播；相似性；责任性；可用性；偏好值
中图分类号：TP274; TP391　　**文献标志码：**A　　DOI: 10.12178/1001-0548.2024308

Determination of the optimal number of clusters in affinity propagation clustering algorithm

HE Xuansen[1,2], HE Fan[3*], BO Xizhu[1], and XIAO Xiangping[1]

(1. Harmony Research Institute, Guangzhou College of Commerce, Guangzhou 511363, China;
2. College of Information Science and Engineering, Hunan University, Changsha 410082, China;
3. School of International Business, Zhejiang International Studies University, Hangzhou 310012, China)

Abstract: Affinity propagation (AP) clustering can automatically search the number and center of clusters, but the number of clusters provided by AP algorithm is quite different from the inherent clustering structure of dataset. Therefore, a method to determine the number of potential clusters in a dataset is proposed. The Euclidean distance square of any two data points is used to form the similarity matrix, and the sample size and the median of non-diagonal elements of the similarity matrix are used as parameters, a preference update formula is established to determine the number of clusters. Similarity and availability are added to form an affinity matrix, the main diagonal elements with positive values in the affinity matrix are taken as centroids of the clusters to realize the mutual verification of the number of clusters and the number of centroids. Through simulation on both random and real datasets, various performance metrics and algorithm running time are evaluated. The results show that the proposed method not only accurately estimates the number of clusters, but also effectively accelerates the convergence of the algorithm, thus meeting the requirements of big data applications.

Key words: affinity propagation; similarity; responsibility; availability; preference

在大数据时代，数据分析最重要的一项任务是将数据划分成不同聚类。数据的分类包括监督和无监督两种。监督法是对具有标签的数据进行训练，得到一个模型以预测未知数据所属的类[1]。实际中，采用传感器采集的数据是没有标签的，需要无监督分类。聚类[2]或探索性数据分析[3]是无监督分类，聚类的目标是将无标记的观测数据区分成有限个簇，以反映数据潜在的分组结构[4]，因此，它获得了非常广泛的应用。

聚类是按照相似性[5]测度把数据分割成不同的簇，相似的数据聚合在一起形成聚类，而不相似的数据则尽可能地被分离[6]。由所生成数据簇的特

收稿日期：2024−04−20
基金项目：广东省普通高校特色创新项目（2024KTSCX131）
作者简介：何选森，教授，主要从事统计信号处理、盲源分离、数据分析等方面的研究。
*通信作者 E-mail: fanhe_2017@163.com

性，聚类可区分为划分聚类（partition clustering）和层次聚类（hierarchical clustering）。划分聚类[7]是直接将数据划分为预先指定数量的簇。层次聚类是利用嵌套方式依序列对数据分组，它包括凝聚层次聚类[8]和分裂层次聚类[9]两种数据集群的形成过程。

各种聚类算法都有特定的使用范围，应用最广泛的是划分聚类，如著名的 K-均值算法[10]；由于它的聚类数量需要用户事先指定，而且初始聚类中心的选择是随机的，因此就有了很多改进的版本[11-12]。为了提高在复杂数据集上的聚类性能，亲和力传播（affinity propagation, AP）[13]算法被提出，它可应用于面部图像聚类、检测微阵列数据中的基因、在文本中找出代表性句子、确定航空旅行可到达城市等[13]。最近几年，AP 算法的应用[14-16]也在不断地扩大。AP 聚类可以自动搜索出数据的集群和相应的质心（或聚类中心），因而解决了“用户需要事先指定聚类数量”的问题，这也是它的最大优势。然而，在 AP 聚类过程中，算法所形成的聚类结果与原始数据固有的聚类结构可能相差甚远。为此，本文提出一种对 AP 算法输入参数的改进方案，以获得与数据集潜在结构相一致的聚类数量，提高算法的聚类性能和聚类质量，为 AP 聚类算法在不同环境下的应用提供有效的参考。

1　AP 聚类算法

在聚类分析中，数据样本点间的距离是一种相异性测度。聚类算法通过对数据集进行学习获得一组质心，使数据点与其最邻近质心之间的平方误差和（sum-of-squared-errors, SSE）达到最小[17]。K-均值算法的 SSE 被证明收敛于 Kuhn-Tucker 点[18]；但在某些条件下，算法可能无法收敛到局部最小值[18]。为了解决这种问题，AP 聚类采用的策略是将所有数据点等可能地视作聚类中心的候选者，从而使得 AP 算法总能收敛，即它能为任何数据集提供聚类结果。

当从实际的数据样本点中选择聚类中心时，被选中的点称为范例[19]。在 AP 算法中，输入参数为数据点之间的实值的相似性[13]：

$$s_{\mathrm{im}}(i,k) = -\|x_i - x_k\|^2 \tag{1}$$

式中，x_k 与 x_i 为两个数据点；相似度 $s_{\mathrm{im}}(i, k)$ 表示点 x_k 作为 x_i 的范例的适合程度。当以最小平方误差为目标，$s_{\mathrm{im}}(i, k)$ 被定义为负平方误差，是两点间欧氏距离[10,17]的平方。当使用依赖范例概率模型（exemplar-dependent probability model）时，在假定范例是点 x_k 时，$s_{\mathrm{im}}(i, k)$ 可以被设置为 x_i 的对数似然。在整个数据集中，每个样本点 x_k 都对应一个 $s_{\mathrm{im}}(k, k)$ 值，称其为偏好，即 $p(k)=s_{\mathrm{im}}(k, k)$。AP 算法是将 $p(k)$ 作为输入参数，$p(k)$ 值大的点 x_k 被选为范例的概率就大，这意味着聚类质心位置与聚类数量都与 $p(k)$ 的取值有关。

从以上分析可知，AP 算法唯一的输入参数是相似矩阵 $\boldsymbol{S}=\{s_{\mathrm{im}}(i, k), i, k=1, 2, \cdots, N\}$，其中 N 为数据集中样本的数量。为了确定合适的 $p(k)$ 值以获得聚类数量，AP 算法中采用的默认偏好值是所有数据的中位数（median）。严格来讲，中位数只适用于连续性的随机变量。但对于划分型的聚类算法，其数据取值都是离散的。因此 AP 算法采用概率相等法计算数据的中位数 M_{e}：

$$P(\boldsymbol{x} < M_{\mathrm{e}}) = P(\boldsymbol{x} > M_{\mathrm{e}}) \tag{2}$$

式中，P 为概率；$\boldsymbol{x}$ 为数据向量 $\{x_i, i=1, 2, \cdots, N\}$。上式含义是取值大于 M_{e} 和小于 M_{e} 的样本点数是一样的。

式 (2) 中还存在一个问题，就是需要区分样本数量 N 为奇数还是偶数。如果 N 为奇数，则 M_{e} 值可能与某个样本点的取值重合；如果 N 为偶数，则 M_{e} 可能落在两个数据点之间的位置上。

另外，AP 聚类将整个数据集看成一个网络，每个数据点代表网络中的一个结点。在点 x_k 与 x_i 间的边上传播消息：责任性和可用性，并称它们为证据。责任性 $r(i, k)$ 和可用性 $a_v(i, k)$ 用以度量每个点对范例的竞争关系。$r(i, k)$ 反映了点 x_i 发送给候选范例 x_k 的积累证据，即从 x_i 角度来看，x_k 作为 x_i 范例的适合程度。$a_v(i, k)$ 反映了候选范例 x_k 发送给点 x_i 的积累证据，即从候选点 x_k 角度来看，点 x_i 选择 x_k 作为范例的适合程度。AP 算法就是对这两种消息不断迭代更新的过程，在迭代初始时刻，可用性 $a_v(i, k)$ 被初始化为 0，即 $a_v(i, k)=0$；而责任性 $r(i,k)$ 的计算规则[13]为：

$$r(i,k) = s_{\mathrm{im}}(i,k) - \max_{j\,(j\neq k)} \{a_v(i,j) + s_{\mathrm{im}}(i,j)\} \tag{3}$$

在初次的迭代中，由于 $a_v(i, k)=0$，则 $r(i, k)$ 为点 x_k 与 x_i 的输入相似性减去点 x_i 与其他候选范例 $x_j(j\neq k)$ 之间相似性中的最大值。在此后的迭代中，当数据点被分配给其他范例时，可用性 $a_v(i,k)$ 会降到 0 以下[13]：

$$a_v(i,k)=\min\left\{0,\ r(k,k)+\sum_{j,\ j\notin\{i,k\}}\max[0,\ r(j,k)]\right\} \quad (4)$$

显然，负值的 $a_v(i, k)$ 将会降低式 (3) 中 $r(i, k)$ 的一些输入相似性 $s_{\mathrm{im}}(i, j)$ 的有效值，从而将相应的候选样本点从竞争中移除。当 $k=i$ 时，$r(k, k)$ 被称为点 x_k 的自责任性，它是选择点 x_k 作为范例的偏好 $p(k)$ 值减去点 x_i 与所有其他范例之间相似度的最大值：

$$r(k,k)=p(k)-\max_{j,\ j\neq k}\{a_v(k,j)+s_{\mathrm{im}}(k,j)\} \quad (5)$$

自责任性 $r(k, k)$ 反映了点 x_k 作为一个范例积累的证据，说明点 x_k 作为一个范例是基于它的输入偏好（值），而且也是作为它不适合分配给其他范例的证据。

由式 (4) 可知，$a_v(i, k)$ 是 $r(k, k)$ 与候选范例 x_k 从其他点获得的正值 $r(j, k)(j\neq\{i, k\})$ 之和。为了限制较大正值 $r(k, k)$ 的影响，将 $a_v(i, k)$ 表达式的总和设置一个阈值，使得自有效性 $a_v(k, k)$ 不能越过 0[13]：

$$a_v(k,k)=\sum_{j,\ j\neq k}\max\{0,\ r(j,k)\} \quad (6)$$

式中，自有效性 $a_v(k, k)$ 是基于其他数据点发送给候选范例 x_k 的正值的责任性 $r(j, k)$，也是对证据的积累过程，其目的是用于证明点 x_k 是一个范例。

另外，AP 算法在对消息的更新过程中需要进行阻尼，以避免在某些情况下，信息出现周期循环或数值振荡[13]。一般地，阻尼因子 λ 的值介于 0～1 之间。

2 最佳聚类数量的确定

尽管 AP 算法中给出了默认的偏好 $p(k)=M_e$，但采用 M_e 进行聚类，效果很不理想。这说明，在无任何先验知识的情况下，要给出与数据相适应的、合适的 $p(k)$ 值是很困难的。为此，需要对 AP 算法的机理进行深入地剖析，从中寻找相关的线索，并以此为基础来估计数据集潜在的聚类数量与聚类中心。

AP 传播机制使消息沿着网络的边传输与交换，并通过不断递归，直到出现一组好的范例为止。在任何时间点上，每条消息的幅度反映点 x_i 选择 x_k 作为其范例的当前亲和力，因此这种机制也称为亲和力传播。在 $r(i, k)$ 更新中允许所有候选范例来争夺 x_i 的所有权，即 $r(i, k)$ 反映的是由数据驱动的竞争性。而 $a_v(i, k)$ 更新则是从数据点收集证据，以确定每个候选范例是否会成为好的范例。所谓好范例是指它与周围非范例点的相似性极强。两种消息 $r(i, k)$ 和 $a_v(i, k)$ 只在具有已知相似性的数据点对 (x_i, x_k) 之间交换，在消息传播的任何时候，消息 $a_v(i, k)$ 和 $r(i, k)$ 都可结合起来用以识别出范例点以及非范例点应归属的范例。对于数据点 x_i，使 $a_v(i, k)+r(i, k)$ 达到最大值所对应的 k 值，要么在 $k=i$ 时将数据点 x_i 标识为范例，要么在 $k\neq i$ 时将 x_k 标识为范例 x_i 所包含的数据点。因此证据 $r(i, k)$ 和 $a_v(i, k)$ 在聚类过程中被看作是对数概率比（log-probability ratios）。

对于一个范例点 x_k，采用正值的 $r(k, k)$ 来解释它适合周围的非范例点，而不用负值的 $r(k, k)$ 解释它不适合其他的数据点。由式 (4) 和式 (5) 可知，偏好 $p(k)$ 值越大，则 $r(k, k)$ 和 $a_v(i, k)$ 的值也就越大，这样的点 x_k 作为最终聚类中心的概率也就越大。

消息 $r(k, k)$ 和 $a_v(i, k)$ 的传递过程一直进行，直到完成所设定的固定迭代次数之后传递过程才终止；或者在消息的更新中，其值低于某个设定的阈值之后传递过程终止；或者在一定次数的迭代中它们的值保持不变就终止传递过程。为此，对于第 i 次迭代，$r(i, k)$ 和 $a_v(i, k)$ 分别被设置了阻尼因子 λ，其迭代规则为：

$$a_v(i,k)=(1-\lambda)a_v(i,k)+\lambda a_v(i-1,k) \quad (7)$$

$$r(i,k)=(1-\lambda)r(i,k)+\lambda r(i-1,k) \quad (8)$$

阻尼因子 λ 的作用是改进 AP 算法的收敛性，当被识别出的范例处于周期变化时，由于振荡使得算法无法收敛，则可通过增加 λ 值以消除振荡[13]。虽然 AP 算法中默认的 λ 值为 0.5，但对于不同的数据集，还需要通过实验对 λ 值进行选择，才能取得更好的聚类效果。

当采用 AP 算法默认的偏好以及阻尼因子，其聚类结果表现为：算法所产生的聚类数量比数据集本身的固有集群数量要大很多。为此，需要对这两个参数进行修正。由于 λ 用于保证算法的收敛，通过设置一个较大的值就可解决（见实验与仿真部分）这个问题。对 AP 算法的聚类数量起决定性作用的偏好值，则要通过对数据本身的特性进行分析来确定。

设数据集 $\boldsymbol{X}$ 的样本数量为 N，特征数量为 M，则其数据矩阵 $\boldsymbol{X}$ 为：

$$\boldsymbol{X}=\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1M} \\ x_{21} & x_{22} & \cdots & x_{2M} \\ \vdots & \vdots & & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{NM} \end{bmatrix}\in \boldsymbol{R}^{N\times M} \tag{9}$$

式中，$\boldsymbol{X}$ 的每一行 $\boldsymbol{x}_i=[x_{i1},x_{i2},\cdots,x_{iM}]$ 表示一个数据样本（点）；$\boldsymbol{X}$ 的每一列 $\boldsymbol{x}_j=[x_{1j},x_{2j},\cdots,x_{Nj}]$ 是数据的一个属性或特征，称为变量。

为了便于分析，这里针对数据矩阵 $\boldsymbol{X}$ 的某个特征 j (j=1, 2, ⋯, M)，计算出它的相似性矩阵：

$$\boldsymbol{S}_j=\begin{bmatrix} s_{\mathrm{im}}(1,1) & s_{\mathrm{im}}(1,2) & \cdots & s_{\mathrm{im}}(1,N) \\ s_{\mathrm{im}}(2,1) & s_{\mathrm{im}}(2,2) & \cdots & s_{\mathrm{im}}(2,N) \\ \vdots & \vdots & & \vdots \\ s_{\mathrm{im}}(N,1) & s_{\mathrm{im}}(N,2) & \cdots & s_{\mathrm{im}}(N,N) \end{bmatrix}\in \boldsymbol{R}^{N\times N} \tag{10}$$

显然，这样的矩阵 $\boldsymbol{S}_j$ 有 M 个。$\boldsymbol{S}_j$ 主对角线上的元素就构成了数据点 x_k 的偏好 $p(k)=\{s_{\mathrm{im}}(k,k)|k=1,2,\cdots,N\}$。在实际的数据聚类分析中，通过选择不同的特征构成数据集 $\boldsymbol{X}$ 相应的相似性矩阵 $\{\boldsymbol{S}_j|j=1,2,\cdots,M\}$。

为寻找与数据集 $\boldsymbol{X}$ 相适应的偏好，对 $\boldsymbol{S}_j$ 矩阵中的非对角元素 $\{s_{\mathrm{im}}(i,k)|i\neq k\}$，即不同数据样本点之间的相似性按照从小到大的次序排列 $s_{\mathrm{im}}(1)\leqslant s_{\mathrm{im}}(2)\leqslant\cdots\leqslant s_{\mathrm{im}}(n)$，其中 $n=N(N-1)$ 为 $\boldsymbol{S}_j$ 中非对角元素个数，由 $s_{\mathrm{im}}(1), s_{\mathrm{im}}(2), \cdots, s_{\mathrm{im}}(n)$ 构成了相似性 $\{s_{\mathrm{im}}(i,k)|i\neq k\}$ 的顺序统计量。

由于矩阵 $\boldsymbol{S}_j$ 中非对角元素的数量 $n=N(N-1)$ 始终保持为偶数，因此，可由顺序统计量计算出 $\{s_{\mathrm{im}}(i,k)|i\neq k\}$ 的中位数（median）为：

$$\mathrm{Median}=\frac{1}{2}\left[s_{\mathrm{im}}\left(\frac{n}{2}\right)+s_{\mathrm{im}}\left(\frac{n}{2}+1\right)\right] \tag{11}$$

由式 (1) 可知相似性 $s_{\mathrm{im}}(i,k)$ 定义为负值，因此由上式计算的中位数 Median 也是负值。

比较式 (11) 和式 (2) 可知，采用式 (2) 计算的是数据样本取值落在某个范围的概率，它与样本数量有关；式 (11) 计算的是相似性矩阵中非对角元素的中位数，它是 $\{s_{\mathrm{im}}(i,k)|i\neq k\}$ 的一种统计量。Median 的显著特点是受样本中异常值的影响较小，具有稳健性（robust）。然而，由式 (11) 也看出，Median 的计算并未考虑所有样本点的相似性信息，只是利用了中间两个相似性的值，因此其有效性较差。显然，无论是式 (2) 中的 M_e 还是式 (11) 中的 Median，对应的 AP 算法聚类性能都不是最好的，为此需要寻找与数据集本身更适合的偏好。

通过上述分析，得到与数据集聚类数量 K 相关的第一条线索为：AP 算法将数据矩阵 $\boldsymbol{X}$ 中所有样本点都等可能地看作是范例的候选者，意味着范例数量 K 与样本容量 N 是有关的。一般地，样本容量 N 越大，形成的聚类数量 K 可能也就越多（见实验与仿真部分），即范例的数量与样本容量成正比：$K\propto N$。

与聚类数量 K 相关的第二条线索则是与相似性有关。相似性 $s_{\mathrm{im}}(i,k)$ 本质上是数据点对 (x_i, x_k) 的欧氏距离平方的度量。当两个数据点间的距离越小（相似性的绝对值越大），它们越有可能被合并在同一个聚类中；相反地，距离越大（相似性的绝对值越小）的数据点对 (x_i, x_k) 则越有可能被划分在不同的聚类中。也就是说，AP 算法所形成的范例数量 K 与相似性 $s_{\mathrm{im}}(i,k)$ 度量是成反比关系。为了度量数据相似性取值的集中趋势（即集中位置），可采用相似性的均值和中位数。在实际应用中，数据集中的离群值（野值 outlier）或噪声是不可避免的，因此需要选择对异常值不敏感的统计量来度量数据相似性的集中位置。基于这种考虑，本文采用中位数 Median 作为 $s_{\mathrm{im}}(i,k)$ 集中趋势的统计量。因此，数据的聚类数量 K 具有以下的关系：$K\propto 1/\mathrm{Median}$。

把以上两种线索综合起来，则可得到聚类数量 K 满足以下关系：$K\propto N/\mathrm{Median}$。

在 AP 算法中，虽然用户不必指定聚类数量 K，但对 K 值具有决定作用的参数就是偏好。考虑以上两种线索提供的聚类数量 K 与样本量 N 及中位数 Median 的比例关系，本文提出改进偏好的计算公式为：

$$\mathrm{Preference}=\frac{N}{\mathrm{Median}\times\mathrm{abs}(\mathrm{Median})} \tag{12}$$

式中，abs(·) 为取绝对值的运算符，这里主要考虑到中位数 Median 本身是负值，因此，偏好的参数 Preference 也必须为负值。

在式 (12) 的计算中，既兼顾了偏好 Preference 与数据样本容量 N 的线性正比例关系；同时 Preference 与相似性中位数 Median 的平方成反比，这就充分强调了数据相似性度量对偏好的影响程度。以 Preference 为输入参数，就得到了本文提出的改进 AP 聚类算法。

为了验证以 Preference 为参数的 AP 算法所得到的聚类数量是否与数据集本身的聚类结构相适应，在每次迭代过程中，将相似度 $s_{\mathrm{im}}(i,k)$ 和可用性 $a_v(i,k)$ 相加以计算出证据（亲和）矩阵：

$$\boldsymbol{E}=\begin{bmatrix} s_{\mathrm{im}}(1,1)+a_v(1,1) & \cdots & s_{\mathrm{im}}(1,N)+a_v(1,N) \\ s_{\mathrm{im}}(2,1)+a_v(2,1) & \cdots & s_{\mathrm{im}}(2,N)+a_v(2,N) \\ \vdots & & \vdots \\ s_{\mathrm{im}}(N,1)+a_v(N,1) & \cdots & s_{\mathrm{im}}(N,N)+a_v(N,N) \end{bmatrix} \tag{13}$$

$\boldsymbol{E}$ 的对角元素 $e(k,k)=\{s_{\mathrm{im}}(k,k)+a_v(k,k)|k=1,2,\cdots,N\}$ 是 $\boldsymbol{E}$ 每行中的最大值，即 $e(k,k)$ 为范例的候选者。

由式 (6) 可知，$a_v(k,k)$ 是基于其他数据点发送给候选范例 x_k 的正值责任性的累计，因此采用亲和矩阵 $\boldsymbol{E}$ 中正值的对角元素 $e(k,k)$ 作为最终形成的范例，即聚类中心所对应的数据点向量为：

$$\boldsymbol{L}=\mathrm{find}\{\mathrm{diag}(\boldsymbol{E})>0\} \tag{14}$$

式中，diag 是求矩阵 $\boldsymbol{E}$ 的对角元素，而 find 则是从其中找到具有正值的主对角元素。

由式 (14) 得到的 $\boldsymbol{L}$ 是一个由范例组成的矢量，$\boldsymbol{L}$ 中元素的个数就是 AP 算法最终所形成的聚类数量：

$$K=\mathrm{length}(\boldsymbol{L}) \tag{15}$$

式中，length($\boldsymbol{L}$) 是求向量 $\boldsymbol{L}$ 的长度。如果数据集中所有的数据都属于同一个聚类，则 K=1。一般地，AP 聚类算法所产生的聚类数量 K 满足 $1\leqslant K\leqslant N-1$。

利用构建的改进 AP 算法就可自动提供数据集的聚类数量和聚类中心。此外，在改进 AP 算法的聚类过程中，利用所获得的聚类数量与式 (15) 的向量 $\boldsymbol{L}$ 的长度 K 值进行相互验证，以确保聚类结果的有效性。

3 实验与仿真

为了验证本文方法对数据聚类数量估计的有效性以及聚类结果的可信性，利用不同类型的数据集进行仿真和实验。仿真平台为个人电脑，其中 CPU 为 Intel（R） Core（TM） i7-8700CPU@3.20 GHz 3.19 GHz，内存为 16 GB，操作系统为 Windows 10，仿真软件 Spyder5 （Python3.11）。

3.1 聚类性能指标与算法的运行时间

一般地，划分聚类算法都采用迭代的方式来搜索数据点构成的聚类空间，给出聚类数目，并产生相应的聚类中心位置。对算法产生聚类结果是否有效进行分析，就是聚类有效性分析（clustering validity）[20-21]。为此就需要采用一些性能度量指标来评价算法的聚类质量。当同一种聚类算法采用不同输入参数产生多个不同的聚类结果时，利用这些性能指标可以确定哪个结果才是最适合数据的自然划分。在本文的仿真与实验中，通过比较数据本身的聚类结构与算法产生的聚类数量（聚类质心位置）来评价聚类的质量。

在数据本身仅有两个集群的情况下，为度量算法的聚类准确程度，采用兰德分数（rand index, RI）[22]；而对于多个集群（3 个或 3 个以上）情况则采用调整的（adjusted）兰德分数（ARI）[23]。

基于互信息分数（mutual information-based score, MI）[24] 用于衡量算法的聚类结果与真实数据类标签之间的相似性。调整互信息（AMI）以及归一化互信息（NMI）则比 MI 具有更强的鲁棒性[24]。

均匀性指数（homogeneity index, HI）[25] 用于衡量一个集群内是否只包含同一类别的样本点，即算法对数据的划分是否同质。完备性（completeness）指数（CI）[25] 用于衡量同一聚类的样本点是否被划分到同一个集群中，即聚类是否完整。显然 HI 和 CI 是从不同的角度对聚类效果进行测度，如果综合考虑 HI 和 CI，则可得到一种基于条件熵的外部评价方法即 V 测度（V-Measure, VM）[25] 指标，VM 用于衡量聚类算法对数据划分的质量。为简单起见，本文采用 $\mathrm{VM}=2\dfrac{\mathrm{HI}\times\mathrm{CI}}{\mathrm{HI}+\mathrm{CI}}$ 计算 VM 的值。HI 和 CI 的差值越小，VM 值就越大。

Fowlkes-Mallows index （FMI）[23, 26] 利用成对精度和召回率的几何平均值来度量聚类质量。

轮廓系数（silhouette coefficient, SC）[27] 用于度量数据聚类的内部紧密性和不同聚类间的分离性，一般采用所有数据点的平均轮廓系数来度量聚类质量。

以上各个指标的取值范围为 [0, 1]，值越接近于 1 则聚类效果越好，越接近 0 表示聚类效果越差。

考虑 Calinski-Harabasz index（CHI）[28] 即方差比准则（variance ratio criterion）来度量算法聚类效果，指标 CHI 越大，表示聚类效果越好。

为了全面评估 AP 算法，除了以上的聚类性能

指标之外，计算复杂度也是重要的评价指标。算法的计算复杂度主要包括空间复杂度和时间复杂度，由于传统的 AP 算法和本文改进的 AP 算法均采用完全同样的数据集进行仿真和实验，因此它们所消耗的存储空间一样；而运行时间的开销（表示为 Time，单位为 s）可以反映出不同算法的收敛性能。

3.2　实验流程

本文的实验与仿真过程以可视化方式呈现聚类的结果，并将各个聚类性能指标的值、算法运行的时间开销也同时显示在可视化的图中。

为了便于观察在不同情况下的聚类效果，对于算法所形成的聚类数量、聚类质心位置，采用辐射状连线构成的数据集群（簇）来表示，即从每个范例点到本集群中的所有非范例点之间都产生一条边，不同的集群用不同颜色的边来区分。整个实验的流程如图 1 所示。

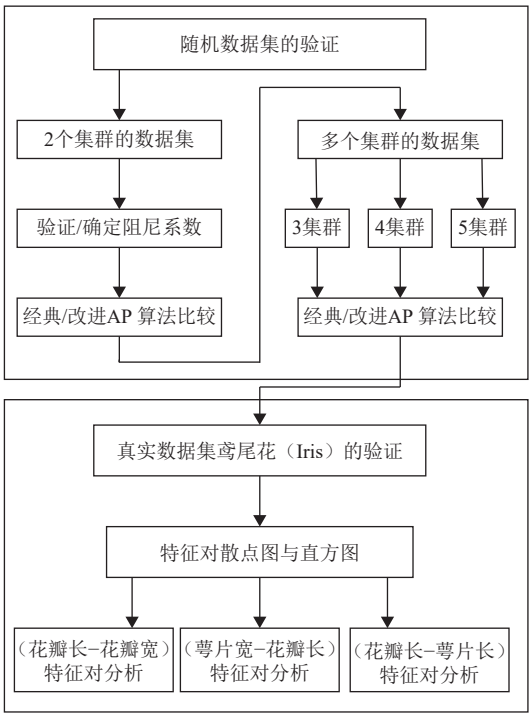


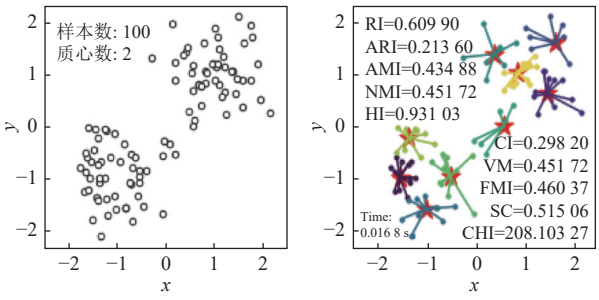
图 1　实验和仿真的流程图

3.3　随机数据的验证

在聚类算法的性能测试中，利用随机种子技术产生的伪随机数具有广泛的应用。在以下仿真实验中，为了便于分析和比较，采用事先指定聚类中心位置的二维 Blobs 随机数（聚类的标准差统一设置为 std=0.5），并分别用基本 AP 聚类算法和本文提出的改进 AP 算法对数据进行聚类分析。

3.3.1　两个聚类的数据验证

随机生成样本容量为 N=100 的数据集，这些数据分别属于质心位置为（−1, −1）和（1, 1）的两个聚类。采用默认参数的 AP（简称基本 AP）算法对该数据进行聚类，其结果如图 2 所示，其中所产生的聚类质心用红色五角星标明。从图中可看出，基本 AP 算法对这个简单的数据集花费了 0.016 8 s 的时间，并产生了 9 个集群。该结果明显是不合理的，因而性能指标也不理想。如 ARI 仅为 0.21，AMI 和 NMI 也仅为 0.43～0.45，其数值很小；而 HI 与 CI 的指标差值却有 0.63，其值很大，SC 也只有 0.51，这些指标值都说明聚类质量较差。

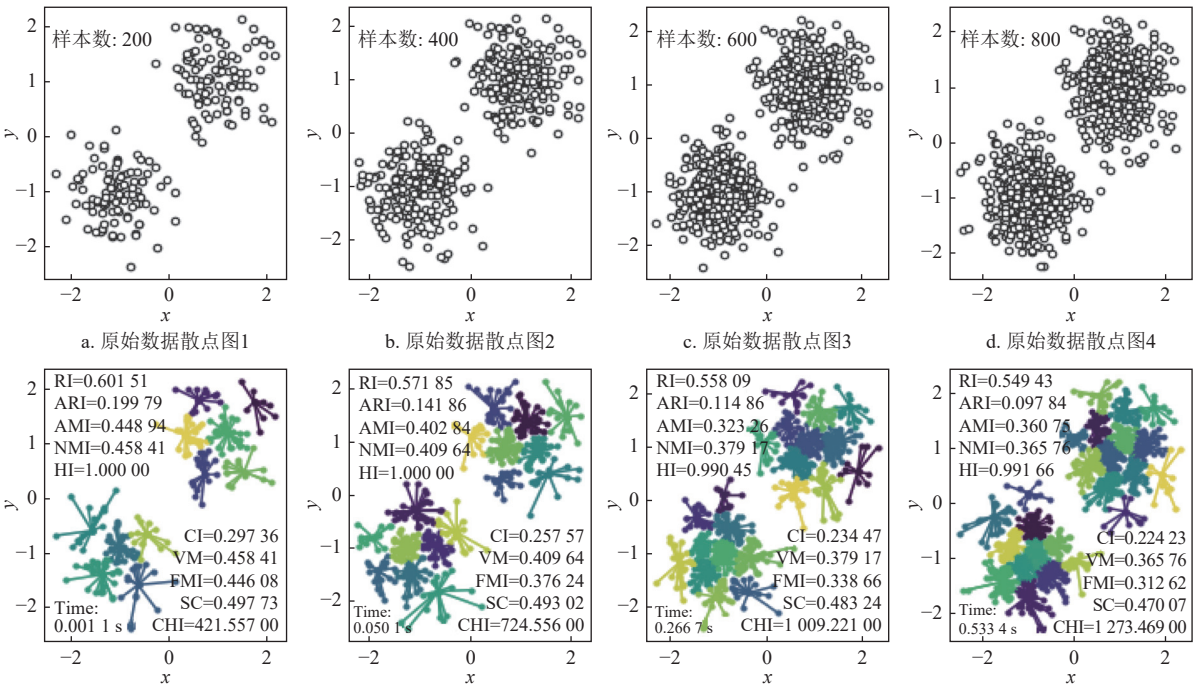


a. 原始数据散点图　　b. 估计的聚类数量: 9

图 2　基本 AP 算法对 Blobs 数据（N=100）的聚类结果

在指定两个聚类中心位置不变的前提下，为了说明样本容量与聚类结果的关系，分别采用 N=200, 400, 600, 800 的 Blobs 数据。由于数据量的增加，数据点之间的距离更加靠近，因此聚类质心位置不再用红色的五角星，而直接用圆点表示。基本 AP 算法对不同样本容量的数据所产生的聚类结果如图 3 所示。

从图 3 可以看出，算法的运行时间随着样本数量 N 的增加而增加。在指定了聚类中心的前提下，算法产生的聚类数量应该与聚类中心数量是一致的。然而从图 3 可以看出，随着数据样本容量增加，算法给出的聚类数量也随之增加。这说明基本 AP 算法的聚类结果不可靠。对于聚类的性能指标，除了 CHI 值随着样本量 N 的增加而增加之外，其他的性能指标基本都是随着 N 的增加而减少。特别是 HI 和 CI 指标的差值越来越大，造成 VM 指标越来越小。这种性能指标的变化意味着两重含义：1）AP 算法的聚类结果与数据的样本容量 N 是有关的；2）随着数据样本容量 N 的增加，基本 AP 算法的聚类质量变差。

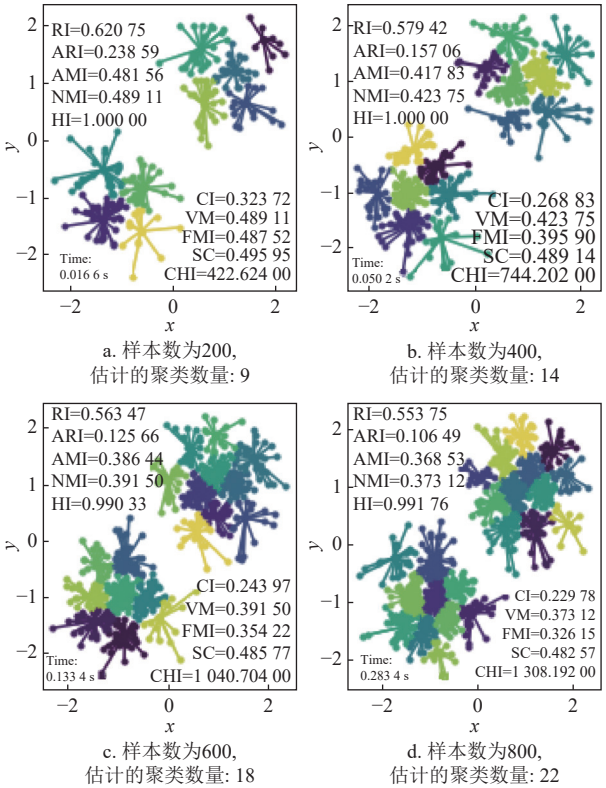


a. 原始数据散点图1　b. 原始数据散点图2　c. 原始数据散点图3　d. 原始数据散点图4

e. 估计的聚类数量: 11　f. 估计的聚类数量: 16　g. 估计的聚类数量: 21　h. 估计的聚类数量: 24

图 3　基本 AP 算法对不同样本容量的聚类结果

为了提高 AP 算法的收敛性能，将默认的阻尼系数 λ=0.5 增加为 λ=0.9。仍然使用上述 4 种样本容量的数据集 Blobs，则 AP 算法的聚类结果如图 4 所示。



a. 样本数为200, 估计的聚类数量: 9　b. 样本数为400, 估计的聚类数量: 14

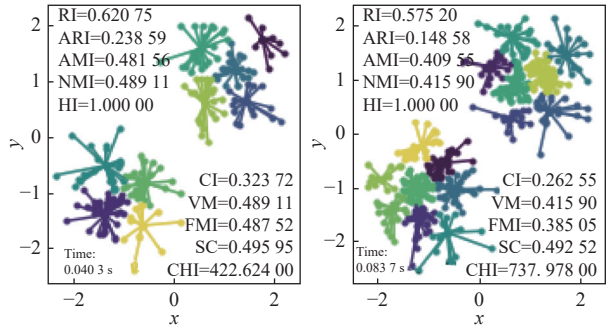
c. 样本数为600, 估计的聚类数量: 18　d. 样本数为800, 估计的聚类数量: 22

图 4　AP 算法设置 λ=0.9 的聚类结果

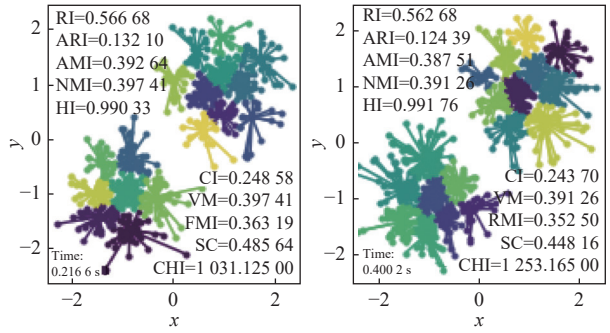
比较图 3 和图 4 可知，当增加阻尼系数 λ 时，AP 算法的运行时间总体上是下降的，而且估计的聚类数量具有朝着合理方向发展的趋势。观察图中的性能指标变化，除了 SC 有升也有降之外，其他的性能指标均有不同程度的提高，这说明 AP 算法的聚类质量在不断地改进。如果继续增加阻尼系数，如设置 λ=0.95，其对应的聚类结果如图 5 所示。

从图 5 可以看出，对于样本数量 N=200 的情况，阻尼因子 λ=0.95 与 λ=0.9 的聚类质量是完全一样的；而当 N=600, 800 时，从聚类数量来看，λ=0.95 比 λ=0.9 的效果要好，但聚类性能指标中有提高也有降低的；当 N=400 时，无论从产生的聚类数量还是聚类的性能指标，λ=0.95 的聚类结果反而比 λ=0.9 的效果差。比较图 4 与图 5 也可知，随着阻尼因子增加，算法的运行时间也在增加。以上这些结果说明，AP 算法的聚类效果与阻尼因子的选取是有关的，但也并非是 λ 越大越好。因此，在下面的仿真实验中，统一采用参数 λ=0.9，并将在 λ=0.9 条件下的 AP 算法简称为经典 AP 算法。

同样地，在以上 4 种样本容量的情况下，阻尼因子设置为 λ=0.9，采用本文提出的改进 AP 算法，对随机数据集 Blobs 的聚类结果如图 6 所示。

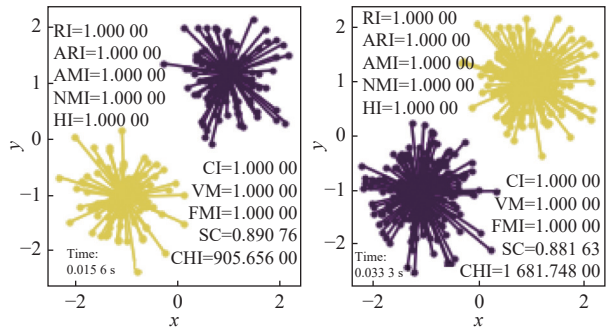


a. 样本数为200，估计的聚类数量: 9　　b. 样本数为400，估计的聚类数量: 15

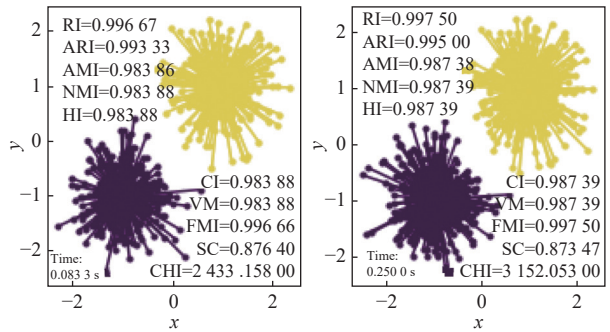


c. 样本数为600，估计的聚类数量: 17　　d. 样本数为800，估计的聚类数量: 18

图 5　AP 算法设置 λ=0.95 的聚类结果



a. 样本数为200，估计的聚类数量: 2　　b. 样本数为400，估计的聚类数量: 2



c. 样本数为600，估计的聚类数量: 2　　d. 样本数为800，估计的聚类数量: 2

图 6　改进 AP 算法的聚类结果

从图 6 的结果可知，改进 AP 算法所获得的聚类数量与原始数据的聚类结构是完全一致的，并且聚类的性能指标都得到了非常明显的提高。其中 RI、ARI、AMI、NMI、HI、CI、VM、FMI 指标在 N=200, 400 的情况下已达到理想值 1，即使对于 N=600, 800 情况，这些指标值也非常接近 1。随着数据样本容量 N 的增加，指标 SC 的总趋势是下降的，但即使在 N=800 这种情况下，SC 的值也有 0.87，它远离 0 而靠近 1。指标 CHI 也是随着 N 的增加而增加，这些性能指标都说明改进 AP 算法的聚类质量是很高的。另外，对比图 4 与图 6 可知，在相同条件下，本文算法的收敛速度比经典 AP 算法快，说明改进 AP 算法的计算效率更高。

为了便于比较改进 AP 算法与经典 AP 算法在 λ=0.9 的同等情况下，聚类质量提升的百分比，将对不同的样本数量 N 和不同性能指标度量分别进行计算。需要说明：指标 HI 和 CI 是从不同角度考察聚类效果，因此它们的可比性较差，将其综合起来使用 VM 指标度量聚类质量更为有效（以下仅考虑 VM）。把图 4 和图 6 中相应的数据分别用于计算改进 AP 算法性能提升、运行时间减少的百分比，其结果如表 1 所示。

表 1　改进 AP 算法对两个聚类数据聚类性能提升及时间减少对比　　%

N	RI	ARI	AMI	NMI	VM	FMI	SC	CHI	Time
200	61.1	319.1	107.7	104.5	104.5	105.1	114.3	79.6	6.0
400	72.6	535.7	139.3	136.0	136.0	152.6	126.0	80.2	33.7
600	76.9	691.0	154.6	151.3	151.3	181.4	133.8	80.4	37.6
800	80.1	834.3	168.0	164.5	164.5	205.9	141.0	81.0	10.4
μ	72.7	595.0	142.4	139.1	139.1	161.3	128.7	80.3	21.9

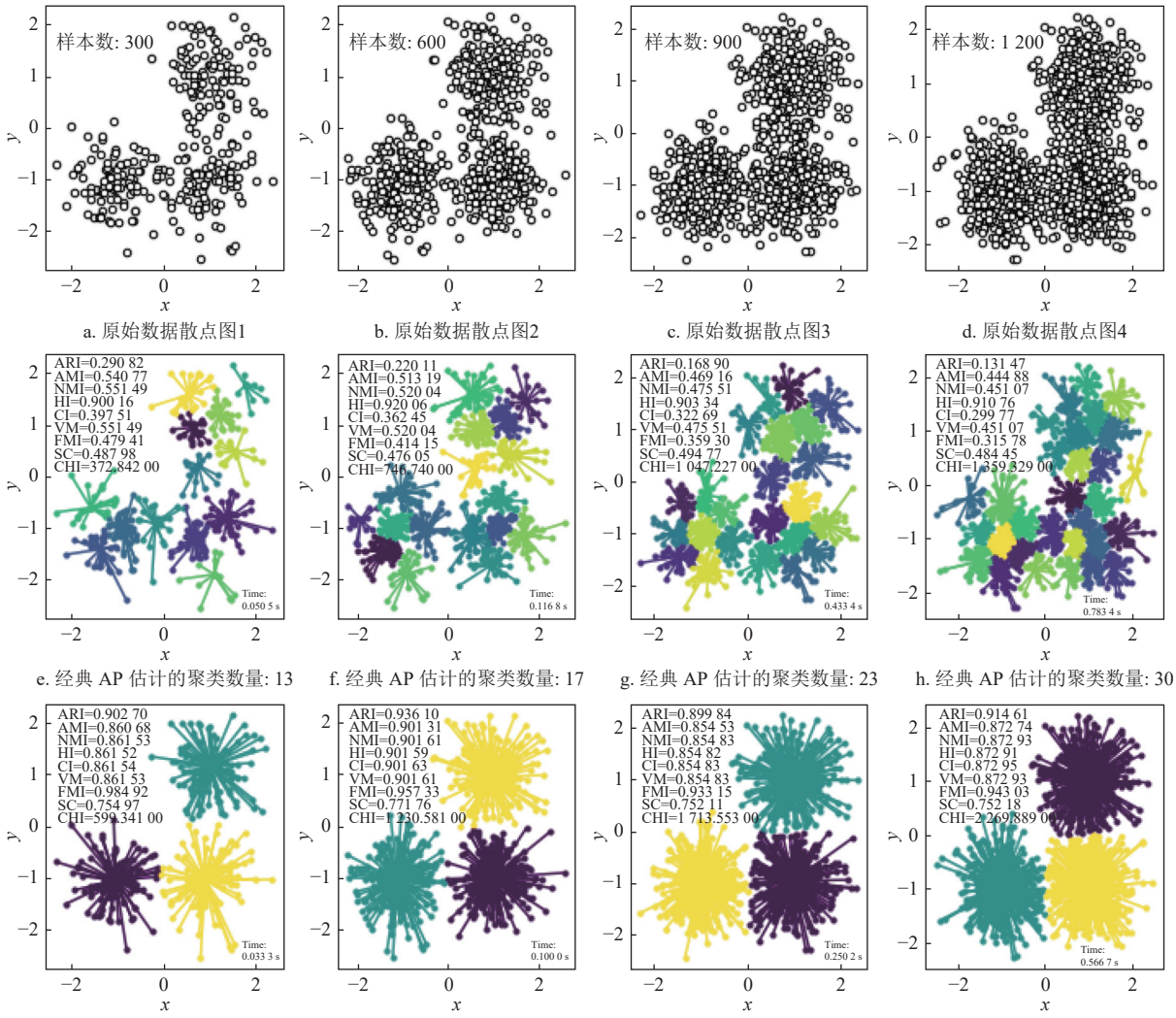
表 1 中的最后一行为各指标百分比的平均值 μ（下同）。从表 1 可以得出以下结论。1）随着样本数量的增加，改进 AP 算法的各项性能指标都在提升，意味着本文算法对于样本容量变化具有很强的鲁棒性。2）对于两个集群的数据集，改进 AP 算法对 ARI 指标平均提升 595%，其他指标平均提升的范围为 72%～278%。3）对于运行时间，改进 AP 算法平均降低了 21.9%。

3.3.2　多个聚类的数据验证

在以下的仿真实验中，仍采用 λ=0.9 来分析经典 AP 算法与改进 AP 算法的聚类性能。为了全方位考察算法的聚类质量，分别对于 3、4、5 个聚类中心和不同样本容量的情况进行仿真。由于这里指定的聚类数量确定大于 2，因此针对两个聚类的性能指标 RI 的度量就没有含义，也就不再计算。

3 个聚类中心：随机生成样本量分别为 N=300,

600, 900, 1 200 的 Blobs 数据集，这些数据分别归属于质心为（−1, −1）、（1, −1）和（1, 1）的 3 个聚类。在这种情况下，经典 AP 和改进 AP 算法的聚类结果如图 7 所示。



a. 原始数据散点图1　b. 原始数据散点图2　c. 原始数据散点图3　d. 原始数据散点图4

e. 经典 AP 估计的聚类数量: 13　f. 经典 AP 估计的聚类数量: 17　g. 经典 AP 估计的聚类数量: 23　h. 经典 AP 估计的聚类数量: 30

i. 改进 AP 估计的聚类数量: 3　j. 改进 AP 估计的聚类数量: 3　k. 改进 AP 估计的聚类数量: 3　l. 改进 AP 估计的聚类数量: 3

图 7　3 个聚类中心数据的聚类结果

从图 7 的结果可以看出，对于算法的收敛性能，改进 AP 算法的运行时间比经典 AP 算法要少，随着样本容量的变化，改进算法收敛速度提升幅度是不同的。从估计的聚类数量来看，经典 AP 算法估计的聚类数量远远大于数据集本身所固有的集群数量，而改进 AP 算法则能够准确地估计出数据集固有的集群数量。从聚类性能指标上来看，随着数据样本数量的增加，经典 AP 算法的指标 ARI、AMI、NMI、VM、和 FMI 逐渐下降；而改进 AP 算法所对应的性能指标并没有随着数据样本量的增加而下降。特别强调：当 HI 和 CI 指标值相差很大时，就会造成综合指标 VM 的恶化。从图 7 中的指标数据可知，随着 N 值的增加，经典 AP 算法所获得的指标 HI 和 CI 的差值也在不断加大，造成指标 VM 变得越来越差；而改进 AP 算法的 HI 和 CI 指标基本不随 N 值而变化，得到稳定的 VM 值。另外，改进 AP 算法的 SC 和 CHI 指标比经典 AP 算法也都获得了明显地提高。以上这些结果说明，改进 AP 算法与经典 AP 算法相比，其聚类的收敛速度和聚类质量都得到了大幅度提高。

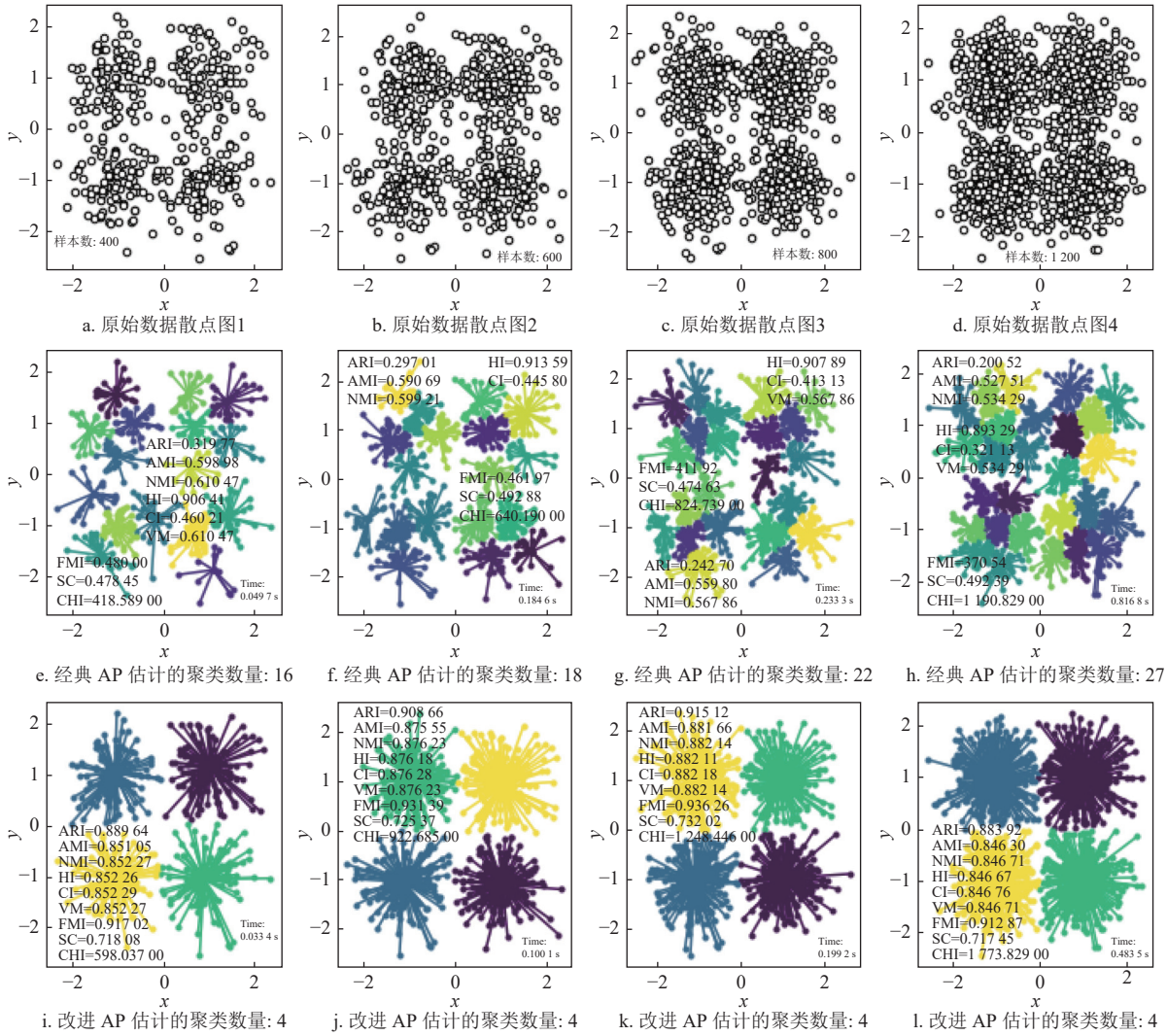
同样地，在聚类数量 K=3 时，改进 AP 算法比经典 AP 算法各个性能指标提升的百分比以及运行时间降低的百分比如表 2 所示。

4 个聚类中心：设置聚类中心的位置为（−1, −1）、（1, −1）、（−1, 1）和（1, 1），产生样本容量分别为 N=400、600、800、1 200 的随机数据集 Blobs。利用两种 AP 聚类算法对该数据集进行聚类，其结果如图 8 所示。

表 2　改进算法对 3 个集群数据聚类性能提升及时间降低对比　　%

N	ARI	AMI	NMI	VM	FMI	SC	CHI	Time
300	210.3	59.2	56.2	56.2	95.0	54.7	60.7	34.1
600	323.1	74.2	72.0	72.0	130.4	62.1	64.8	14.4
900	432.7	82.2	79.8	79.8	159.7	51.9	63.5	42.3
1 200	595.7	96.2	93.5	93.5	198.6	55.3	67.0	27.7
μ	390.5	78.0	75.4	75.4	145.9	56.0	64.0	29.6

从图 8 的结果可以看出，随着样本数量 N 的增加，改进 AP 算法的运行时间相对于经典 AP 算法来说，都有一定程度的降低，而经典 AP 算法估计的聚类数量却随着 N 的增加而增加。特别是对于样本容量较大的情况，如 N=1 200，经典 AP 算法所形成的聚类质心位置之间的距离相差很小，使得各聚类的辨识度较差，显然这是不合理的划分。而改进 AP 算法所产生的聚类数量和聚类中心位置，既符合数据集本身的分组结构，同时在样本数量 N 不断增加时，各项性能指标仍保持较理想的值，这说明改进 AP 算法的性能具有稳定性。与经典 AP 算法相比较，改进 AP 算法聚类性能提升以及运行时间降低的百分比如表 3 所示。



a. 原始数据散点图1　b. 原始数据散点图2　c. 原始数据散点图3　d. 原始数据散点图4

e. 经典 AP 估计的聚类数量: 16　f. 经典 AP 估计的聚类数量: 18　g. 经典 AP 估计的聚类数量: 22　h. 经典 AP 估计的聚类数量: 27

i. 改进 AP 估计的聚类数量: 4　j. 改进 AP 估计的聚类数量: 4　k. 改进 AP 估计的聚类数量: 4　l. 改进 AP 估计的聚类数量: 4

图 8　对 4 个聚类中心数据的聚类结果

5 个聚类中心：为了进一步考察 AP 算法在恶劣环境下的聚类结果，将数据的聚类数量设置为 5，而样本容量分别设置为 N=1 000, 1 500, 2 000, 2 500 而产生随机数 Blobs，其质心分别在（−1, −1）、（0, 0）、（1, −1）、（−1, 1）和（1, 1）的位置上。对于这样的数据集，采用经典 AP 算法和改进的 AP 算法在同等条件下对这些数据分别进行聚类分析，其结果如图 9 所示。

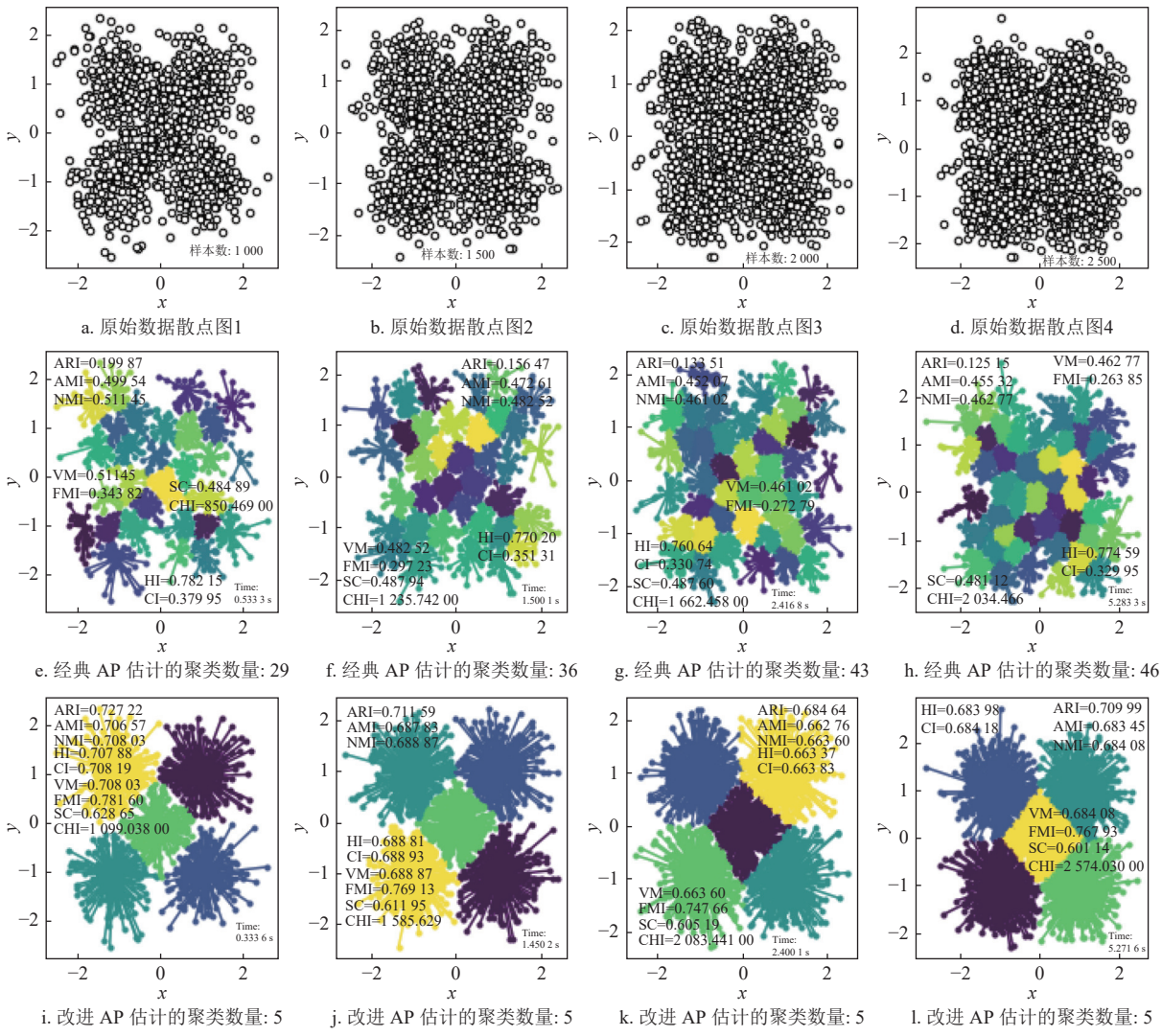
从图 9 的结果可以看出：1）在聚类数量和样本数量都比较大的情况下，算法的运行时间明显地增加了很多；这是因为随着样本数量的剧增，找到聚类质心位置就需要更多的迭代次数。尽管如此，改进 AP 算法的运行时间仍然比经典 AP 算法有优势。2）由于样本量很大（特别是 N=2 000,

2 500），从原始数据散点图中已经无法分辨出数据所形成的聚类形状和集群数量，致使数据的聚类分析环境变得恶劣。3）随着样本容量 N 的增加，经典 AP 算法所产生的聚类数量也在不断地增加，特别是在 N=2 000, 2 500 的情况下，经典 AP 算法估计的聚类数量实在是太多了，造成了对算法结果的解释变得异常困难，甚至无法给出合理的解释。4）对于改进的 AP 算法来说，无论样本容量如何变化，算法所产生的聚类数量与数据集本身的聚类结构总是保持完全一致；同时也清晰、合理地给出了数据集的聚类中心位置。显然，这使得聚类数量和聚类中心二者完美地配合，否则任何一个出错都会造成聚类失效。改进的 AP 算法能够获得这样好的聚类结果，是因为在对偏好的计算过程中，既考虑了数据的样本容量，同时又考虑了数据相似性的统计量（中位数），因而在对聚类数量和聚类中心的估计方面显得与数据本身结构更为贴近。

表 3　改进 AP 算法对 4 个集群数据聚类性能提升及时间减少对比

%

N	ARI	AMI	NMI	VM	FMI	SC	CHI	Time
400	178.2	42.1	39.6	39.6	91.0	50.1	42.9	32.8
600	205.9	48.2	46.2	46.2	101.6	47.2	44.3	45.8
800	275.8	56.7	54.6	54.6	126.7	54.2	51.4	14.6
1 200	340.8	60.4	58.5	58.5	146.4	45.7	49.0	40.8
μ	250.2	51.9	49.7	49.7	116.4	49.3	46.9	33.5



a. 原始数据散点图1　b. 原始数据散点图2　c. 原始数据散点图3　d. 原始数据散点图4

e. 经典 AP 估计的聚类数量: 29　f. 经典 AP 估计的聚类数量: 36　g. 经典 AP 估计的聚类数量: 43　h. 经典 AP 估计的聚类数量: 46

i. 改进 AP 估计的聚类数量: 5　j. 改进 AP 估计的聚类数量: 5　k. 改进 AP 估计的聚类数量: 5　l. 改进 AP 估计的聚类数量: 5

图 9　对 5 个聚类中心数据的聚类结果

根据图 9 中的各项指标，可计算出改进 AP 算法聚类性能提升的百分比以及运行时间降低的百分比，其结果如表 4 所示。

在以上仿真实验的基础上，在不同聚类数量 K=3、4、5 情况下，将改进 AP 算法性能指标提升百分比以及运行时间下降百分比的均值进行汇总，

并计算出各项指标的总平均值，其结果如表 5 所示。

表 4　改进 AP 算法对 5 个集群数据聚类性能提升及时间减少对比　%

N	ARI	AMI	NMI	VM	FMI	SC	CHI	Time
1 000	263.8	41.1	38.4	38.4	127.3	29.6	29.2	37.4
1 500	354.8	45.5	42.8	42.8	158.7	25.4	28.3	3.3
2 000	412.8	46.6	43.9	43.9	174.1	24.1	25.3	0.7
2 500	467.3	50.1	47.8	47.8	191.1	24.9	26.5	0.2
μ	347.7	45.8	43.2	43.2	162.8	26.0	27.3	10.4

表 5　改进 AP 算法对不同集群数量的平均性能提升及时间减少对比　%

K	ARI	AMI	NMI	VM	FMI	SC	CHI	Time
3	390.5	78.0	75.4	75.4	145.9	56.0	64.0	29.6
4	250.2	51.9	49.7	49.7	116.4	49.3	46.9	33.5
5	347.7	45.8	43.2	43.2	162.8	26.0	27.3	10.4
μ	329.5	58.6	56.1	56.1	141.7	43.8	46.1	24.5

从表 5 中的数据可知，对于不同的集群数量，仅从运行时间上来看，改进 AP 算法在 K=3, 4 的情况下具有显著的优势。但随着数据聚类数量 K 的进一步增加，如 K=5，改进 AP 算法在运行时间降低的幅度上明显变小；这是因为数据样本的取值范围并没有增加，当指定的样本容量和聚类中心数量增多时，对于处在不同集群之间数据点的判断就具有很大的不确定性，要精确地给出聚类数量和聚类质心位置，就需要更多的运算时间。尽管如此，改进 AP 算法在 K=5 时的收敛速度仍比经典 AP 算法快 10%。另外，改进 AP 算法的聚类性能与聚类质量仍能保持在较高的水平上。当然，对不同的聚类性能指标，改进 AP 算法的提升幅度也不完全相同，其中 ARI 指标在 K=3, 4, 5 情况下的总平均提升幅度为 329%，FMI 指标的总平均提升幅度为 142%，而其他指标的总平均提升幅度在 43%～59%之间。

3.4　真实数据集验证

鸢尾花 Iris 数据集包括 3 种花 Setosa、Versicolor 和 Virginica 的 4 个特征：萼片（sepal）长、萼片宽、花瓣（petal）长、花瓣宽的测量数据（单位为 cm）。在 Iris 数据集中总共存储了 150 组样本的特征值，按照 setosa、versicolor 和 virginica 的顺序依次存放了各 50 组的观测值。

为了便于比较，把 3 种花卉的 4 个特征对之间的数据散点图绘制出来，其结果如图 10 所示。可以看出，在不同特征对的数据散点图中，都不同程度的存在数据点相互渗透的现象，即不同品种花卉的样本特征值之间是有交叉的，难以分类。另外从图 10 主对角线的直方图可以看出，3 种花卉的特征大致服从于不同均值和不同方差的正态分布。

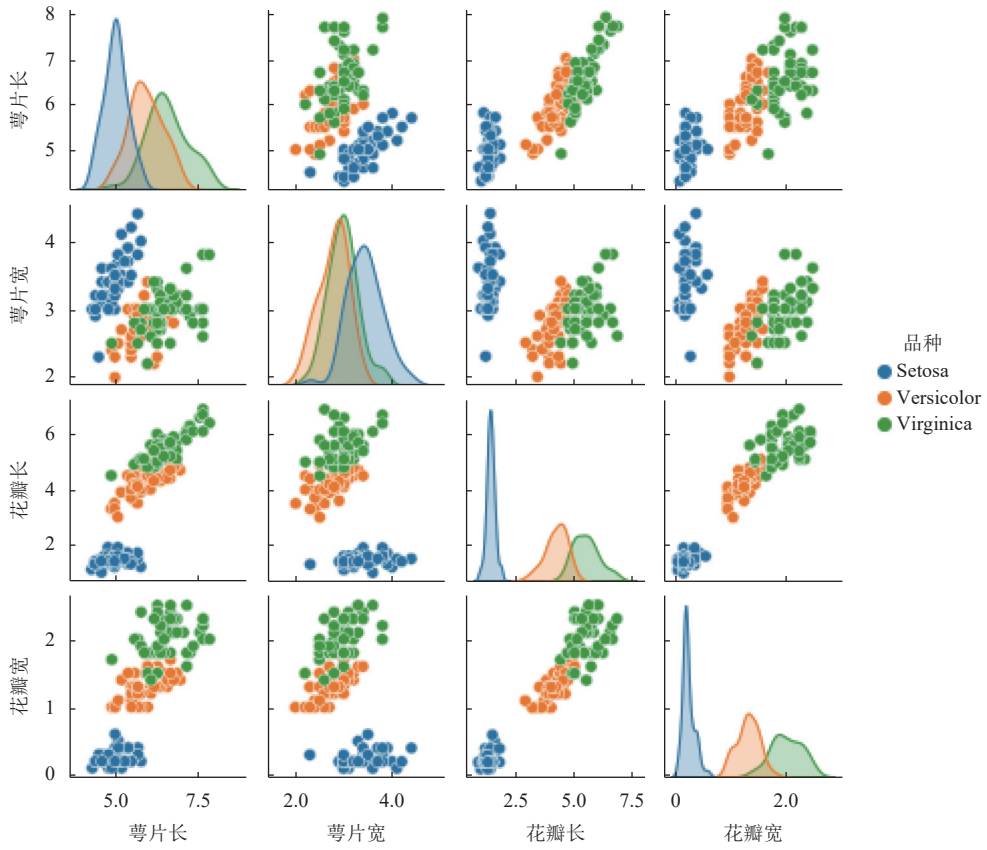
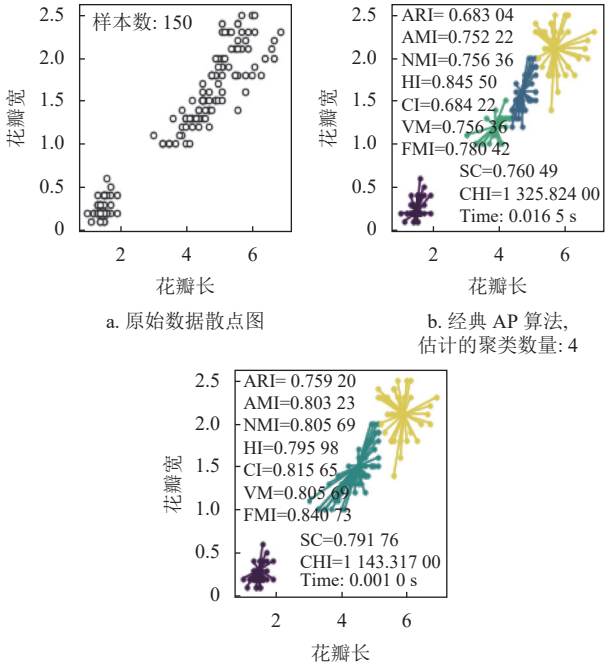


图 10　Iris 数据集的特征对散点图及分布直方图

为了验证 AP 算法对真实数据集 Iris 的聚类性能，将特征对（花瓣长、花瓣宽）的 3 种花卉的数据进行聚类分析，其结果如图 11 所示。从图 11 的结果可知，经典 AP 算法将原本 3 种花卉的数据错误地划分为 4 个聚类，通过比较图 10 中的（花瓣长、花瓣宽）彩色特征对的散点图可知，经典 AP 算法是将原本属于第 2 种花 versicolor 的数据点强制地划分成了 2 个不同的聚类，因此其聚类的性能指标较差，并且算法所花费的运行时间也长。与之相对应地，改进 AP 算法形成的聚类数量为 3，与 Iris 数据集本身的聚类结构完全一致，并且各个聚类性能指标都要优于经典 AP 算法。为便于比较，将改进 AP 算法在性能指标提升及运行时间减少的百分比列在表 6 中。



a. 原始数据散点图　　b. 经典 AP 算法, 估计的聚类数量: 4

c. 改进 AP 算法, 估计的聚类数量: 3

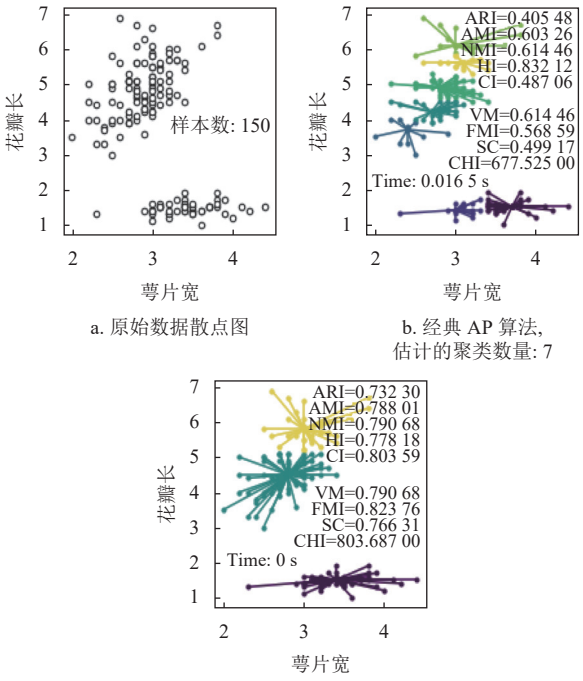
图 11　特征对（花瓣长、花瓣宽）的聚类结果

表 6　改进 AP 算法对（花瓣长、花瓣宽）特征对性能提升及时间减少结果

%

ARI	AMI	NMI	VM	FMI	SC	CHI	Time
11.15	6.78	6.52	6.52	7.72	15.51	12.11	93.9

从表 6 中的数据可知，对于特征对（花瓣长、花瓣宽）来说，改进 AP 算法相对于经典 AP 算法，各个性能指标提升的百分比在 6%～16% 之间，而算法的运行时间减少了 93.9%。类似地，将特征对（萼片宽, 花瓣长）的数据进行聚类分析，其结果如图 12 所示。在图 12 中可知，改进 AP 算法的运行时间为 0，实际上算法运行是需要时间的，出现这种现象的原因是采用保留小数点后面 4 位数字的浮点数所导致的，这也说明了改进 AP 算法的运行时间是很快的。对于特征对（萼片宽、花瓣长），经典 AP 算法产生聚类数量为 7，对照图 10 中数据特征对的彩色散点图可知，它将 setosa 数据点分成了两个聚类，将 versicolor 和 virginica 的数据点分裂成了 5 个聚类，由于所产生的聚类中心位置相互之间距离较近，造成聚类形状的扭曲。而改进 AP 算法不仅获得了完全准确的聚类数量，而且聚类性能指标比经典 AP 算法也提高很多。改进 AP 算法性能提升与运行时间减少的百分比如表 7 所示。



a. 原始数据散点图　　b. 经典 AP 算法, 估计的聚类数量: 7

c. 改进 AP 算法, 估计的聚类数量: 3

图 12　特征对（萼片宽、花瓣长）的聚类结果

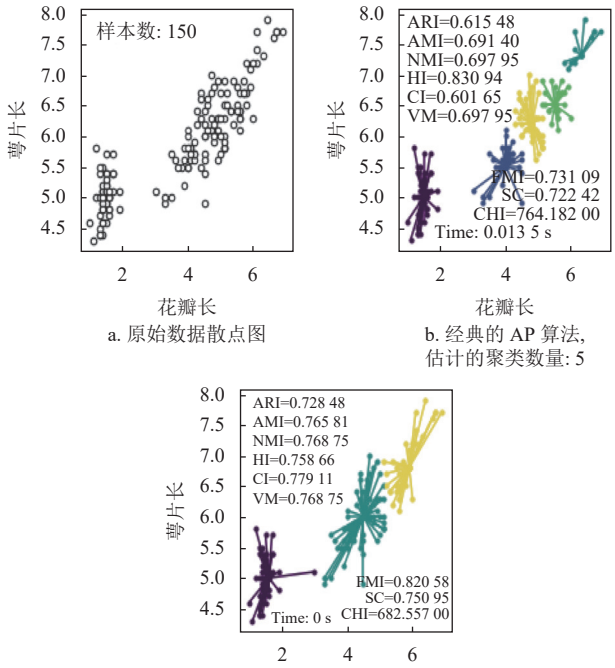
表 7　改进 AP 算法对（萼片宽、花瓣长）特征对性能提升及时间减少结果

%

ARI	AMI	NMI	VM	FMI	SC	CHI	Time
80.60	30.63	28.68	28.68	44.88	83.62	43.96	99.8

从表 7 可以看出，对于特征对（萼片宽、花瓣长）的数据，改进 AP 算法各个指标提升的百分比在 28%～84% 之间。显然，这与表 6 中的（花瓣长、花瓣宽）特征对的聚类性能有较大幅度的提高，同时也意味着对不同特征对的数据聚类，其难易程度是不同的。对于运行时间，在考虑保留小数点后 4 位有效值的情况下，假设改进 AP 算法的运行时间为 0.000 04（因为真实的运行时间会四舍五入），则其降低幅度为 99.8%。

同样地，利用 AP 算法对特征对（花瓣长、萼片长）的数据进行聚类分析，其结果如图 13 所示。



a. 原始数据散点图　　b. 经典的 AP 算法，估计的聚类数量: 5

c. 改进的 AP 算法, 估计的聚类数量: 3

图 13　特征对（花瓣长、萼片长）的聚类结果

从图 13 可看出，改进 AP 算法的运行时间仍为 0，这意味着与经典 AP 算法相比，改进 AP 算法的运行速度非常快。从聚类的结果可知，经典 AP 算法给出的聚类数量为 5，通过对比图 10 中数据的彩色散点图可知，它将 versicolor 和 virginica 的数据点分别划分成 2 个聚类，显然与数据本身结构不相匹配。而改进 AP 算法仍能准确地估计出聚类数量，同时也保持了较高的聚类性能指标。改进 AP 算法性能指标提升及运行时间减少（与上述情况类似）的百分比如表 8 所示。

表 8　改进 AP 算法对（花瓣长、萼片长）特征对性能提升及时间减少结果

%

ARI	AMI	NMI	VM	FMI	SC	CHI	Time
18.35	10.76	10.14	10.14	12.24	9.74	10.63	99.7

从表 8 的数据可以看出，对于特征对（花瓣长、萼片长）来说，改进 AP 算法相对于经典 AP 算法性能提升的百分比在 9%～19% 之间；在考虑改进 AP 算法的运行时间为 0.000 04 的情况下，运行时间降低了 99.7%。把以上 3 组特征对的仿真结果进行平均，即可得到聚类性能指标提升以及运行时间降低的百分比的总平均值。其结果如表 9 所示。

从表 9 的数据可知，对于真实数据集 Iris，由于不同花卉的特征数据点之间存在有相互交叉现象，使得经典 AP 算法的聚类效果不很理想。然而，改进 AP 算法的聚类性能指标仍然平均提升了 15%～37%，而算法的运行时间平均减少 97.8%。仅从算法的收敛性角度考虑，改进 AP 算法对 Iris 数据的处理速度很快。

表 9　改进 AP 算法对 Iris 数据聚类性能的平均提升及时间减少结果

%

ARI	AMI	NMI	VM	FMI	SC	CHI	Time
36.7	16.06	15.11	15.11	21.61	36.29	22.23	97.8

通过以上对真实 Iris 数据集的仿真结果可知，无论选择哪两个特征对，本文提出的改进 AP 算法都能够快速给出数据集中真实聚类数量的准确估计，这说明改进 AP 算法对特征对的选择具有较强的鲁棒性。

4　结束语

通过对不同聚类数量与不同样本容量的随机数进行仿真，并采用多种指标进行评估，验证了本文改进 AP 算法比经典 AP 算法的聚类性能、运行时间都有较大幅度的改进。通过对真实数据集 Iris 不同特征对的仿真，验证了改进 AP 算法比经典 AP 算法在聚类质量上具有一定程度的提高。

另外，通过比较随机数据集和 Iris 数据集的仿真实验结果，存在这样一个疑问：本文方法对小数据集（Iris）的收敛性能提升幅度很大，而对大样本的数据集，其收敛性能提升幅度似乎还有空间可挖掘。鉴于本文研究的局限性并考虑未来可能的应用拓展场景，将在以下两个方向继续深入地研究。

1）探索更多的数据集，这包括两个方面的研究：一方面，将本文方法应用于多特征（特别是分类特征）的数据集，提高本文方法的适应性；另一方面，挖掘行业的实际数据集，利用真实数据集验证本文方法的有效性。

2）探索算法收敛性能的进一步提升。为了适应大数据的应用要求，对本文方法需要从理论分析、搜索策略优化等方面进行深入研究，通过不断改进，使得本文方法能对超大的数据集提供有效的聚类分析。

参 考 文 献

[1] TANG L, LI J Y, DU H C, et al. Big data in forecasting research: A literature review[J]. Big Data Research, 2022, 27: 100289.

[2] MITTAL M, GOYAL L M, HEMANTH D J, et al.

Clustering approaches for high-dimensional databases: A review[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2019, 9(3): e1300.
[3] ALMINAGORTA O, LOEWEN C J G, DE KERCKHOVE D T, et al. Exploratory analysis of multivariate data: Applications of parallel coordinates in ecology[J]. Ecological Informatics, 2021, 64: 101361.
[4] CARUSO G, GATTONE S A, FORTUNA F, et al. Cluster analysis for mixed data: An application to credit risk evaluation[J]. Socio-Economic Planning Sciences, 2021, 73: 100850.
[5] SOLEIMANI G, ABESSI M. DLCSS: A new similarity measure for time series data mining[J]. Engineering Applications of Artificial Intelligence, 2020, 92: 103664.
[6] HUANG D, WANG C D, PENG H X, et al. Enhanced ensemble clustering *via* fast propagation of cluster-wise similarities[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2021, 51(1): 508-520.
[7] LIU T M, YU H, BLAIR R H. Stability estimation for unsupervised clustering: A review[J]. Wiley Interdisciplinary Reviews: Computational Statistics, 2022, 14(6): e1575.
[8] ZHOU S B, XU Z Y, LIU F. Method for determining the optimal number of clusters based on agglomerative hierarchical clustering[J]. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(12): 3007-3017.
[9] JI H C, ZUO Z Y, HAN Q L. A divisive hierarchical clustering approach to hyperspectral band selection[J]. IEEE Transactions on Instrumentation and Measurement, 2022, 71: 5014312.
[10] STEINLEY D. K-means clustering: A half-century synthesis[J]. British Journal of Mathematical and Statistical Psychology, 2006, 59(1): 1-34.
[11] AY M, LALE Ö, KULLUK S, et al. FC-kmeans: Fixed-centered K-means algorithm[J]. Expert Systems with Applications, 2023, 211: 118656.
[12] 何选森, 何帆, 徐丽, 等. K-Means 算法最优聚类数量的确定[J]. 电子科技大学学报, 2022, 51(6): 904-912.
HE X S, HE F, XU L, et al. Determination of the optimal number of clusters in K-means algorithm[J]. Journal of University of Electronic Science and Technology of China, 2022, 51(6): 904-912.
[13] FREY B J, DUECK D. Clustering by passing messages between data points[J]. Science, 2007, 315(5814): 972-976.
[14] 黄鹤, 李文龙, 杨澜, 等. 跳跃跟踪 SSA 交叉迭代 AP 聚类算法[J]. 电子学报, 2024, 52(3): 977-990.
HUANG H, LI W L, YANG L, et al. Jump tracking SSA hybrid iterative AP clustering algorithm[J]. Acta Electronica Sinica, 2024, 52(3): 977-990.
[15] 何青松, 谭荣辉, 杨俊. 基于近邻传播聚类元胞自动机模型的武汉城市扩散和聚合过程同步模拟[J]. 地理学报, 2021, 76(10): 2522-2535.
HE Q S, TAN R H, YANG J. Synchronized simulation of urban diffusional and aggregational process based on the affinity propagation cellular automata: A case study of Wuhan city[J]. Acta Geographica Sinica, 2021, 76(10): 2522-253.
[16] 党倩, 崔阿军, 尚闻博, 等. 采用欧式形态距离的负荷曲线近邻传播聚类方法[J]. 西安交通大学学报, 2022, 56(1): 165-176.
DANG Q, CUI A J, SHANG W B, et al. Affinity propagation clustering method of typical load curve with euclidean morphological distance[J]. Journal of Xi'an Jiaotong University, 2022, 56(1): 165-176.
[17] STEINLEY D. Stability analysis in K-means clustering[J]. British Journal of Mathematical and Statistical Psychology, 2008, 61(2): 255-273.
[18] SELIM S Z, ISMAIL M A. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1984, PAMI-6(1): 81-87.
[19] MÉZARD M. Where are the exemplars?[J]. Science, 2007, 315(5814): 949-951.
[20] ABDALAMEER A K, ALSWAITTI M, ALSUDANI A A, et al. A new validity clustering index-based on finding new centroid positions using the mean of clustered data to determine the optimum number of clusters[J]. Expert Systems with Applications, 2022, 191: 116329.
[21] HE X S, HE F, CAI W H. Underdetermined BSS based on K-means and AP clustering[J]. Circuits, Systems, and Signal Processing, 2016, 35(8): 2881-2913.
[22] HUBERT L, ARABIE P. Comparing partitions[J]. Journal of Classification, 1985, 2(1): 193-218.
[23] STEINLEY D. Properties of the hubert-arabie adjusted rand index[J]. Psychological Methods, 2004, 9(3): 386-396.
[24] STREHL A, GHOSH J. Cluster ensembles—A knowledge reuse framework for combining multiple partitions[J]. Journal of Machine Learning Research, 2002, 2(2002): 583-617.
[25] ROSENBERG A, HIRSCHBERG J. V-Measure: A conditional entropy-based external cluster evaluation measure[C]//Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. [S.l.]: Association for Computational Linguistics, 2007, 410-420.
[26] FOWLKES E B, MALLOWS C L. A method for comparing two hierarchical clusterings[J]. Journal of the American Statistical Association, 1983, 78(383): 553-569.
[27] ROUSSEEUW P J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis[J]. Journal of Computational and Applied Mathematics, 1987, 20: 53-65.
[28] CALINSKI T, HARABASZ J. A dendrite method for cluster analysis[J]. Communications in Statistics, 1974, 3(1): 1-27.

编 辑 叶 芳