Affinity propagation Clustering

劝君莫惜金缕衣
劝君须惜少年时
花开堪折直须折
莫待无花空折枝

Clustering by Passing Messages Between Data Points

Brendan J. Frey* and Delbert Dueck

Clustering data by identifying a subset of representative examples is important for processing sensory signals and detecting patterns in data. Such "exemplars" can be found by randomly choosing an initial subset of data points and then iteratively refining it, but this works well only if that initial choice is close to a good solution. We devised a method called "affinity propagation," which takes as input measures of similarity between pairs of data points. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. We used affinity propagation to cluster images of faces, detect genes in microarray data, identify representative sentences in this manuscript, and identify cities that are efficiently accessed by airline travel. Affinity propagation found clusters with much lower error than other methods, and it did so in less than one-hundredth the amount of time.

Clustering data based on a measure of similarity is a critical step in scientific data analysis and in engineering systems. A common approach is to use data to learn a set of centers such that the sum of squared errors between data points and their nearest centers is small. When the centers are selected from actual data points, they are called "exemplars." The popular k-centers clustering technique (*1*) begins with an initial set of randomly selected exemplars and iteratively refines this set so as to decrease the sum of squared errors. k-centers clustering is quite sensitive to the initial selection of exemplars, so it is usually rerun many times with different initializations in an attempt to find a good solution. However, this works well only when the number of clusters is small and chances are good that at least one random initialization is close to a good solution. We take a quite different approach and introduce a method that simultaneously considers all data points as potential exemplars. By viewing each data point as a node in a network, we devised a method that recursively transmits real-valued messages along edges of the network until a good set of exemplars and corresponding clusters emerges. As described later, messages are updated on the basis of simple formulas that search for minima of an appropriately chosen energy function. At any point in time, the magnitude of each message reflects the current affinity that one data point has for choosing another data point as its exemplar, so we call our method "affinity propagation." Figure 1A illustrates how clusters gradually emerge during the message-passing procedure.

Affinity propagation takes as input a collection of real-valued similarities between data points, where the similarity $s(i,k)$ indicates how well the data point with index k is suited to be the exemplar for data point i. When the goal is to minimize squared error, each similarity is set to a negative squared error (Euclidean distance): For points x_i and x_k, $s(i,k) = -\|x_i - x_k\|^2$. Indeed, the method described here can be applied when the optimization criterion is much more general. Later, we describe tasks where similarities are derived for pairs of images, pairs of microarray measurements, pairs of English sentences, and pairs of cities. When an exemplar-dependent probability model is available, $s(i,k)$ can be set to the log-likelihood of data point i given that its exemplar is point k. Alternatively, when appropriate, similarities may be set by hand.

Rather than requiring that the number of clusters be prespecified, affinity propagation takes as input a real number $s(k,k)$ for each data point k so that data points with larger values of $s(k,k)$ are more likely to be chosen as exemplars. These values are referred to as "preferences." The number of identified exemplars (number of clusters) is influenced by the values of the input preferences, but also emerges from the message-passing procedure. If a priori, all data points are equally suitable as exemplars, the preferences should be set to a common value—this value can be varied to produce different numbers of clusters. The shared value could be the median of the input similarities (resulting in a moderate number of clusters) or their minimum (resulting in a small number of clusters).

There are two kinds of message exchanged between data points, and each takes into account a different kind of competition. Messages can be combined at any stage to decide which points are exemplars and, for every other point, which exemplar it belongs to. The "responsibility" $r(i,k)$, sent from data point i to candidate exemplar point k, reflects the accumulated evidence for how well-suited point k is to serve as the exemplar for point i, taking into account other potential exemplars for point i (Fig. 1B). The "availability" $a(i,k)$, sent from candidate exemplar point k to point i, reflects the accumulated evidence for how appropriate it would be for point i to choose point k as its exemplar, taking into account the support from other points that point k should be an exemplar (Fig. 1C). $r(i,k)$ and $a(i,k)$ can be viewed as log-probability ratios. To begin with, the availabilities are initialized to zero: $a(i,k) = 0$. Then, the responsibilities are computed using the rule

$$r(i,k) \leftarrow s(i,k) - \max_{k' \text{ s.t. } k' \neq k} \{a(i,k') + s(i,k')\} \quad (1)$$

In the first iteration, because the availabilities are zero, $r(i,k)$ is set to the input similarity between point i and point k as its exemplar, minus the largest of the similarities between point i and other candidate exemplars. This competitive update is data-driven and does not take into account how many other points favor each candidate exemplar. In later iterations, when some points are effectively assigned to other exemplars, their availabilities will drop below zero as prescribed by the update rule below. These negative availabilities will decrease the effective values of some of the input similarities $s(i,k')$ in the above rule, removing the corresponding candidate exemplars from competition. For $k = i$, the responsibility $r(k,k)$ is set to the input preference that point k be chosen as an exemplar, $s(k,k)$, minus the largest of the similarities between point i and all other candidate exemplars. This "self-responsibility" reflects accumulated evidence that point k is an exemplar, based on its input preference tempered by how ill-suited it is to be assigned to another exemplar.

Whereas the above responsibility update lets all candidate exemplars compete for ownership of a data point, the following availability update gathers evidence from data points as to whether each candidate exemplar would make a good exemplar:

$$a(i,k) \leftarrow \min\left\{0, r(k,k) + \sum_{i' \text{ s.t. } i' \notin \{i,k\}} \max\{0, r(i',k)\}\right\} \quad (2)$$

The availability $a(i,k)$ is set to the self-responsibility $r(k,k)$ plus the sum of the positive responsibilities candidate exemplar k receives from other points. Only the positive portions of incoming responsibilities are added, because it is only necessary for a good exemplar to explain some data points well (positive responsibilities), regardless of how poorly it explains other data points (negative responsibilities). If the self-responsibility $r(k,k)$ is negative (indicating that point k is currently better suited as belonging to another exemplar rather than being an exemplar itself), the availability of point k as an exemplar can be increased if some other points have positive responsibilities for point k being their exemplar. To limit the influence of strong

Department of Electrical and Computer Engineering, University of Toronto, 10 King's College Road, Toronto, Ontario M5S 3G4, Canada.

*To whom correspondence should be addressed. E-mail: frey@psi.toronto.edu

incoming positive responsibilities, the total sum is thresholded so that it cannot go above zero. The "self-availability" $a(k,k)$ is updated differently:

$$a(k,k) \leftarrow \sum_{i' \text{ s.t. } i' \neq k} \max\{0, r(i',k)\} \qquad (3)$$

This message reflects accumulated evidence that point k is an exemplar, based on the positive responsibilities sent to candidate exemplar k from other points.

The above update rules require only simple, local computations that are easily implemented (*2*), and messages need only be exchanged between pairs of points with known similarities. At any point during affinity propagation, availabilities and responsibilities can be combined to identify exemplars. For point i, the value of k that maximizes $a(i,k) + r(i,k)$ either identifies point i as an exemplar if $k = i$, or identifies the data point that is the exemplar for point i. The message-passing procedure may be terminated after a fixed number of iterations, after changes in the messages fall below a threshold, or after the local decisions stay constant for some number of iterations. When updating the messages, it is important that they be damped to avoid numerical oscillations that arise in some circumstances. Each message is set to λ times its value from the previous iteration plus $1 - \lambda$ times its prescribed updated value, where the damping factor λ is between 0 and 1. In all of our experiments (*3*), we used a default damping factor of $\lambda = 0.5$, and each iteration of affinity propagation consisted of (i) updating all responsibilities given the availabilities, (ii) updating all availabilities given the responsibilities, and (iii) combining availabilities and responsibilities to monitor the exemplar decisions and terminate the algorithm when these decisions did not change for 10 iterations.

Figure 1A shows the dynamics of affinity propagation applied to 25 two-dimensional data points (*3*), using negative squared error as the similarity. One advantage of affinity propagation is that the number of exemplars need not be specified beforehand. Instead, the appropriate number of exemplars emerges from the message-passing method and depends on the input exemplar preferences. This enables automatic model selection, based on a prior specification of how preferable each point is as an exemplar. Figure 1D shows the effect of the value of the common input preference on the number of clusters. This relation is nearly identical to the relation found by exactly minimizing the squared error (*2*).

We next studied the problem of clustering images of faces using the standard optimiza-

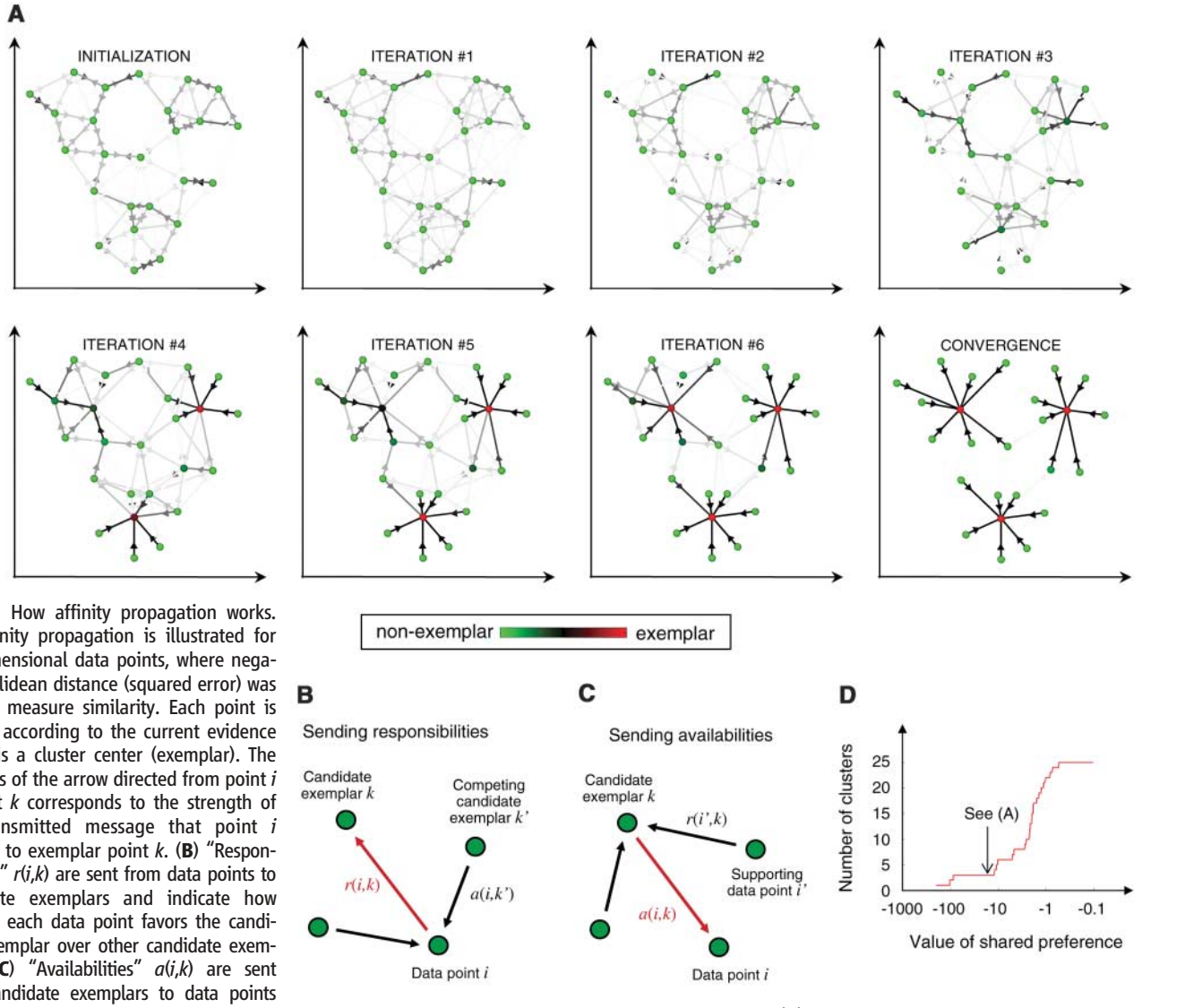

Fig. 1. How affinity propagation works. (**A**) Affinity propagation is illustrated for two-dimensional data points, where negative Euclidean distance (squared error) was used to measure similarity. Each point is colored according to the current evidence that it is a cluster center (exemplar). The darkness of the arrow directed from point i to point k corresponds to the strength of the transmitted message that point i belongs to exemplar point k. (**B**) "Responsibilities" $r(i,k)$ are sent from data points to candidate exemplars and indicate how strongly each data point favors the candidate exemplar over other candidate exemplars. (**C**) "Availabilities" $a(i,k)$ are sent from candidate exemplars to data points and indicate to what degree each candidate exemplar is available as a cluster center for the data point. (**D**) The effect of the value of the input preference (common for all data points) on the number of identified exemplars (number of clusters) is shown. The value that was used in (A) is also shown, which was computed from the median of the pairwise similarities.

tion criterion of squared error. We used both affinity propagation and *k*-centers clustering to identify exemplars among 900 grayscale images extracted from the Olivetti face database (*3*). Affinity propagation found exemplars with much lower squared error than the best of 100 runs of *k*-centers clustering (Fig. 2A), which took about the same amount of computer time. We asked whether a huge number of random restarts of *k*-centers clustering could achieve the same squared error. Figure 2B shows the error achieved by one run of affinity propagation and the distribution of errors achieved by 10,000 runs of *k*-centers clustering, plotted against the number of clusters. Affinity propagation uniformly achieved much lower error in more than two orders of magnitude less time. Another popular optimization criterion is the sum of absolute pixel differences (which better tolerates outlying pixel intensities), so we repeated the above procedure using this error measure. Affinity propagation again uniformly achieved lower error (Fig. 2C).

Many tasks require the identification of exemplars among sparsely related data, i.e., where most similarities are either unknown or large and negative. To examine affinity propagation in this context, we addressed the task of clustering putative exons to find genes, using the sparse similarity matrix derived from microarray data and reported in (*4*). In that work, 75,066 segments of DNA (60 bases long) corresponding to putative exons were mined from the genome of mouse chromosome 1. Their transcription levels were measured across 12 tissue samples, and the similarity between every pair of putative exons (data points) was computed. The measure of similarity between putative exons was based on their proximity in the genome and the degree of coordination of their transcription levels across the 12 tissues. To account for putative exons that are not exons (e.g., introns), we included an additional artificial exemplar and determined the similarity of each other data point to this "non-exon exemplar" using statistics taken over the entire data set. The resulting 75,067 × 75,067 similarity matrix (*3*) consisted of 99.73% similarities with values of $-\infty$, corresponding to distant DNA segments that could not possibly be part of the same gene. We applied affinity propagation to this similarity matrix, but because messages need not be exchanged between point i and k if $s(i,k) = -\infty$, each iteration of affinity propagation required exchanging messages between only a tiny subset (0.27% or 15 million) of data point pairs.

Figure 3A illustrates the identification of gene clusters and the assignment of some data points to the nonexon exemplar. The reconstruction errors for affinity propagation and *k*-centers clustering are compared in Fig. 3B. For each number of clusters, affinity propagation was run once and took 6 min, whereas *k*-centers clustering was run 10,000 times and took 208 hours. To address the question of how well these methods perform in detecting bona fide gene segments, Fig. 3C plots the true-positive (TP) rate against the false-positive (FP) rate, using the labels provided in the RefSeq database (*5*). Affinity propagation achieved significantly higher TP rates, especially at low FP rates, which are most important to biologists. At a FP rate of 3%, affinity propagation achieved a TP rate of 39%, whereas the best *k*-centers clustering result was 17%. For comparison, at the same FP rate, the best TP rate for hierarchical agglomerative clustering (*2*) was 19%, and the engineering tool described in (*4*), which accounts for additional biological knowledge, achieved a TP rate of 43%.

Affinity propagation's ability to operate on the basis of nonstandard optimization criteria makes it suitable for exploratory data analysis using unusual measures of similarity. Unlike metric-space clustering techniques such as *k*-means clustering (*1*), affinity propagation can be applied to problems where the data do not lie in a continuous space. Indeed, it can be applied to problems where the similarities are not symmetric [i.e., $s(i,k) \neq s(k,i)$] and to problems where the similarities do not satisfy the triangle inequality [i.e., $s(i,k) < s(i,j) + s(j,k)$]. To identify a small number of sentences in a draft of this manuscript that summarize other sentences, we treated each sentence as a "bag of words" (*6*) and computed the similarity of sentence i to sentence k based on the cost of encoding the words in sentence i using the words in sentence k. We found that 97% of the resulting similarities (*2*, *3*) were not symmetric. The preferences were adjusted to identify (using $\lambda = 0.8$) different numbers of representative exemplar sentences (*2*), and the solution with four sentences is shown in Fig. 4A.

We also applied affinity propagation to explore the problem of identifying a restricted number of Canadian and American cities that are most easily accessible by large subsets of other cities, in terms of estimated commercial airline travel time. Each data point was a city, and the similarity $s(i,k)$ was set to the negative time it takes to travel from city i to city k by airline, including estimated stopover delays (*3*). Due to headwinds, the transit time was in many cases different depending on the direction of travel, so that 36% of the similarities were asymmetric. Further, for 97% of city pairs i and k, there was a third city j such that the triangle inequality was violated, because the trip from i to k included a long stopover delay

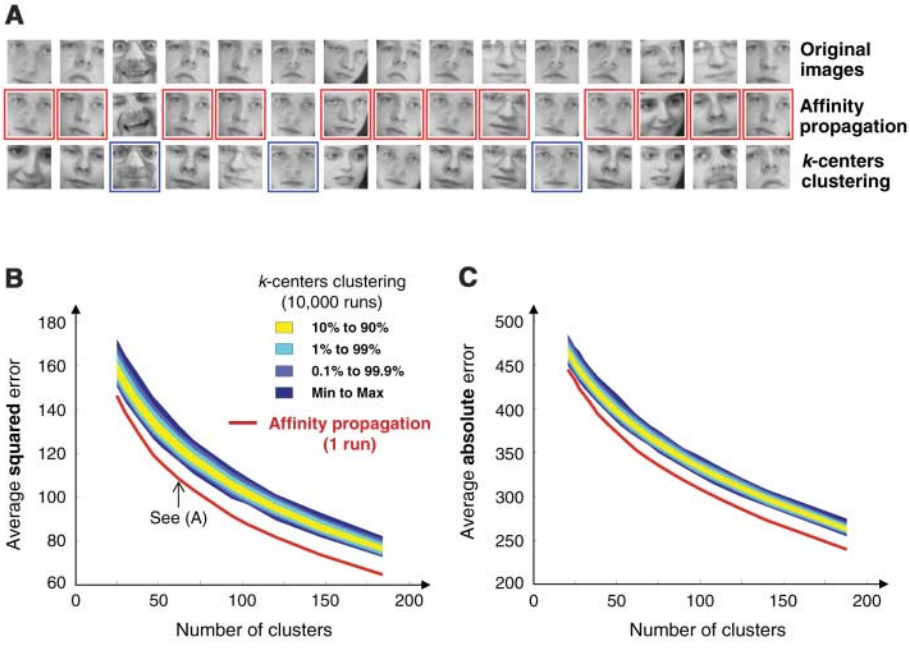

Fig. 2. Clustering faces. Exemplars minimizing the standard squared error measure of similarity were identified from 900 normalized face images (*3*). For a common preference of −600, affinity propagation found 62 clusters, and the average squared error was 108. For comparison, the best of 100 runs of *k*-centers clustering with different random initializations achieved a worse average squared error of 119. (**A**) The 15 images with highest squared error under either affinity propagation or *k*-centers clustering are shown in the top row. The middle and bottom rows show the exemplars assigned by the two methods, and the boxes show which of the two methods performed better for that image, in terms of squared error. Affinity propagation found higher-quality exemplars. (**B**) The average squared error achieved by a single run of affinity propagation and 10,000 runs of *k*-centers clustering, versus the number of clusters. The colored bands show different percentiles of squared error, and the number of exemplars corresponding to the result from (A) is indicated. (**C**) The above procedure was repeated using the sum of absolute errors as the measure of similarity, which is also a popular optimization criterion.

in city j so it took longer than the sum of the durations of the trips from i to j and j to k. When the number of "most accessible cities" was constrained to be seven (by adjusting the input preference appropriately), the cities shown in Fig. 4, B to E, were identified. It is interesting that several major cities were not selected, either because heavy international travel makes them inappropriate as easily accessible domestic destinations (e.g., New York City, Los Angeles) or because their neighborhoods can be more efficiently accessed through other destinations (e.g., Atlanta, Philadelphia, and Minneapolis account for Chicago's destinations, while avoiding potential airport delays).

Affinity propagation can be viewed as a method that searches for minima of an energy function (*7*) that depends on a set of N hidden labels, $c_1,\ldots,c_N$, corresponding to the N data points. Each label indicates the exemplar to which the point belongs, so that $s(i,c_i)$ is the similarity of data point i to its exemplar. $c_i = i$ is a special case indicating that point i is itself an exemplar, so that $s(i,c_i)$ is the input preference for point i. Not all configurations of the labels are valid; a configuration $\mathbf{c}$ is valid when for every point i, if some other point i' has chosen i as its exemplar (i.e., $c_{i'} = i$), then i must be an exemplar (i.e., $c_i = i$). The energy of a valid configuration is $E(\mathbf{c}) = -\sum_{i=1}^{N} s(i,c_i)$. Exactly minimizing the energy is computationally intractable, because a special case of this minimization problem is the NP-hard k-median problem (*8*). However, the update rules for affinity propagation correspond to fixed-point recursions for minimizing a Bethe free-energy (*9*) approximation. Affinity propagation is most easily derived as an instance of the max-sum algorithm in a factor graph (*10*) describing the constraints on the labels and the energy function (*2*).

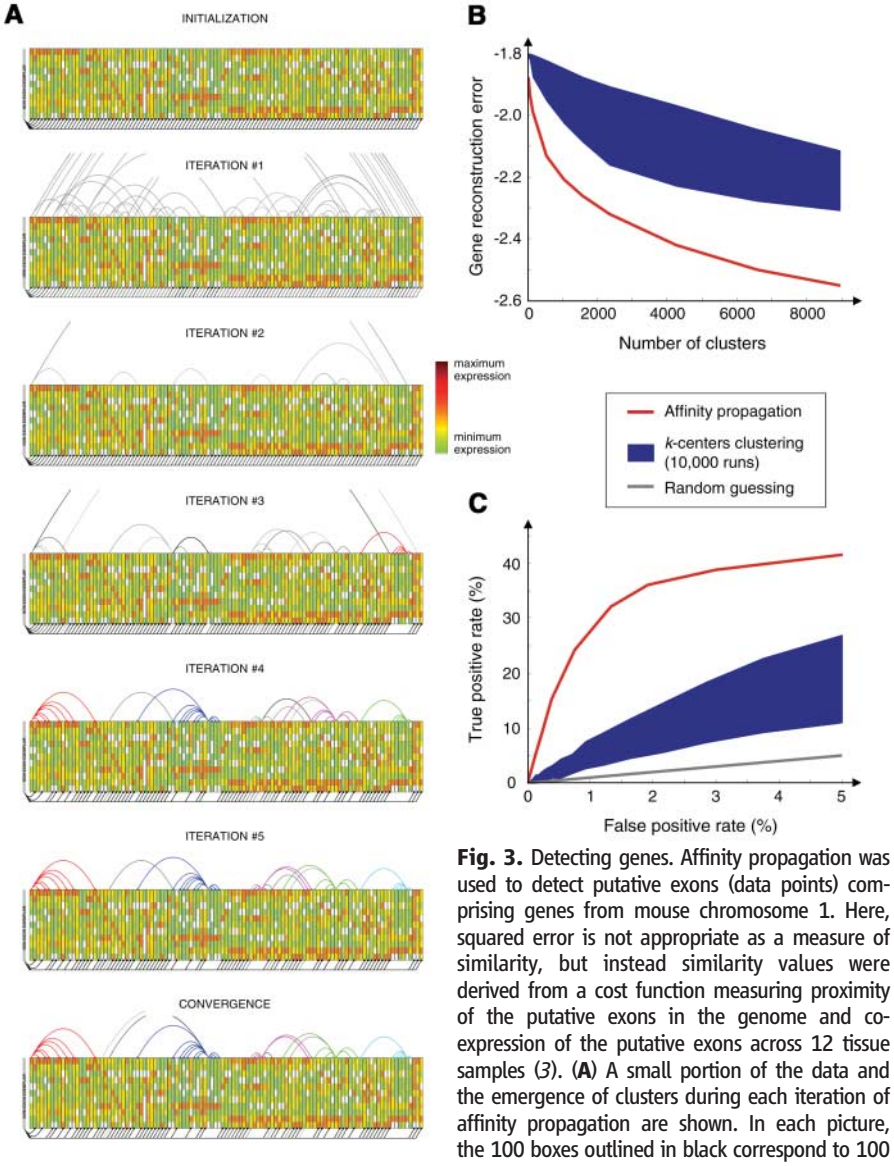

Fig. 3. Detecting genes. Affinity propagation was used to detect putative exons (data points) comprising genes from mouse chromosome 1. Here, squared error is not appropriate as a measure of similarity, but instead similarity values were derived from a cost function measuring proximity of the putative exons in the genome and co-expression of the putative exons across 12 tissue samples (*3*). (**A**) A small portion of the data and the emergence of clusters during each iteration of affinity propagation are shown. In each picture, the 100 boxes outlined in black correspond to 100 data points (from a total of 75,066 putative exons), and the 12 colored blocks in each box indicate the transcription levels of the corresponding DNA segment in 12 tissue samples. The box on the far left corresponds to an artificial data point with infinite preference that is used to account for nonexon regions (e.g., introns). Lines connecting data points indicate potential assignments, where gray lines indicate assignments that currently have weak evidence and solid lines indicate assignments that currently have strong evidence. (**B**) Performance on minimizing the reconstruction error of genes, for different numbers of detected clusters. For each number of clusters, affinity propagation took 6 min, whereas 10,000 runs of k-centers clustering took 208 hours on the same computer. In each case, affinity propagation achieved a significantly lower reconstruction error than k-centers clustering. (**C**) A plot of true-positive rate versus false-positive rate for detecting exons [using labels from RefSeq (*5*)] shows that affinity propagation also performs better at detecting biologically verified exons than k-centers clustering.

In some degenerate cases, the energy function may have multiple minima with corresponding multiple fixed points of the update rules, and these may prevent convergence. For example, if $s(1,2) = s(2,1)$ and $s(1,1) = s(2,2)$, then the solutions $c_1 = c_2 = 1$ and $c_1 = c_2 = 2$ both achieve the same energy. In this case, affinity propagation may oscillate, with both data points alternating between being exemplars and nonexemplars. In practice, we found that oscillations could always be avoided by adding a tiny amount of noise to the similarities to prevent degenerate situations, or by increasing the damping factor.

Affinity propagation has several advantages over related techniques. Methods such as k-centers clustering (*1*), k-means clustering (*1*), and the expectation maximization (EM) algorithm (*11*) store a relatively small set of estimated cluster centers at each step. These techniques are improved upon by methods that begin with a large number of clusters and then prune them (*12*), but they still rely on random sampling and make hard pruning decisions that cannot be recovered from. In contrast, by simultaneously considering all data points as candidate centers and gradually identifying clusters, affinity propagation is able to avoid many of the poor solutions caused by unlucky initializations and hard decisions. Markov chain Monte Carlo techniques (*13*) randomly search for good solutions, but do not share affinity propagation's advantage of considering many possible solutions all at once.

Hierarchical agglomerative clustering (*14*) and spectral clustering (*15*) solve the quite different problem of recursively comparing pairs of points to find partitions of the data. These techniques do not require that all points within a cluster be similar to a single center and are thus not well-suited to many tasks. In particular, two points that should not be in the same cluster may be grouped together by an unfortunate sequence of pairwise groupings.

In (*8*), it was shown that the related metric k-median problem could be relaxed to form a

linear program with a constant factor approximation. There, the input was assumed to be metric, i.e., nonnegative, symmetric, and satisfying the triangle inequality. In contrast, affinity propagation can take as input general nonmetric similarities. Affinity propagation also provides a conceptually new approach that works well in practice. Whereas the linear programming relaxation is hard to solve and sophisticated software packages need to be applied (e.g., CPLEX), affinity propagation makes use of intuitive message updates that can be implemented in a few lines of code (*2*).

Affinity propagation is related in spirit to techniques recently used to obtain record-breaking results in quite different disciplines (*16*). The approach of recursively propagating messages (*17*) in a "loopy graph" has been used to approach Shannon's limit in error-correcting decoding (*18*, *19*), solve random satisfiability problems with an order-of-magnitude increase in size (*20*), solve instances of the NP-hard two-dimensional phase-unwrapping problem (*21*), and efficiently estimate depth from pairs of stereo images (*22*). Yet, to our knowledge, affinity propagation is the first method to make use of this idea to solve the age-old, fundamental problem of clustering data. Because of its simplicity, general applicability, and performance, we believe affinity propagation will prove to be of broad value in science and engineering.

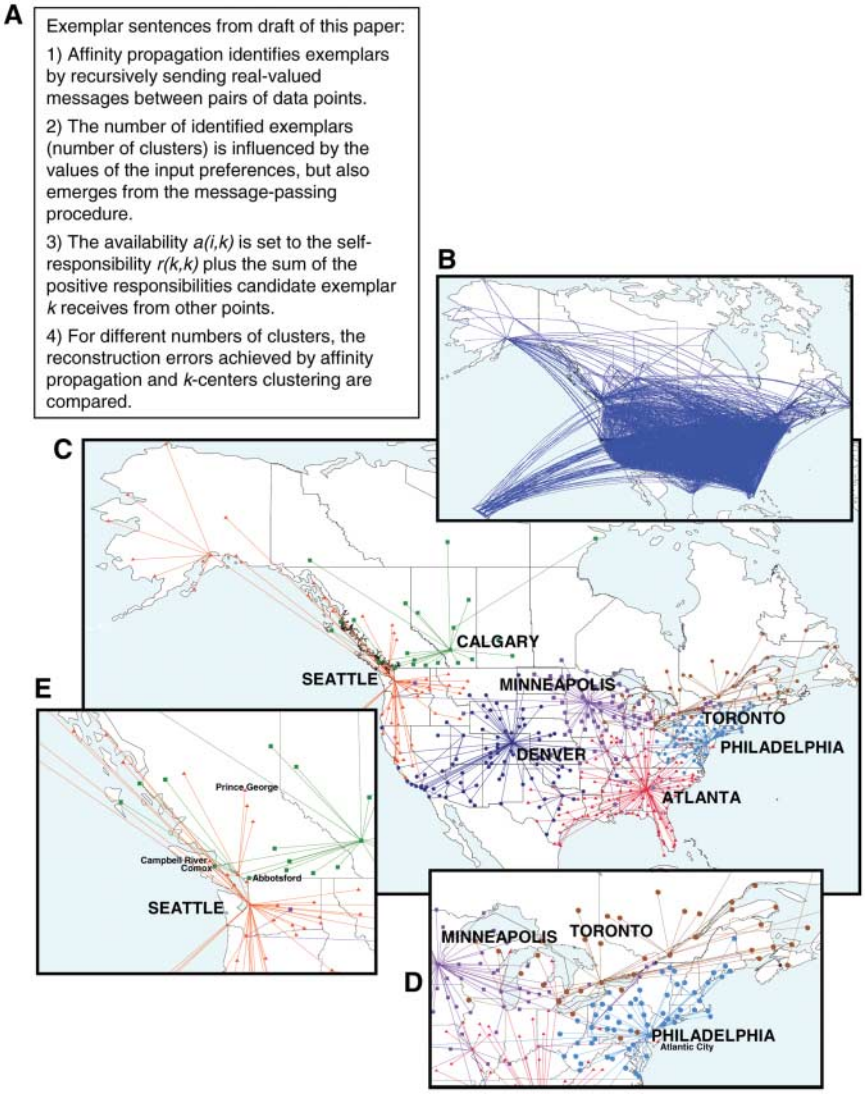

Fig. 4. Identifying key sentences and air-travel routing. Affinity propagation can be used to explore the identification of exemplars on the basis of nonstandard optimization criteria. (**A**) Similarities between pairs of sentences in a draft of this manuscript were constructed by matching words. Four exemplar sentences were identified by affinity propagation and are shown. (**B**) Affinity propagation was applied to similarities derived from air-travel efficiency (measured by estimated travel time) between the 456 busiest commercial airports in Canada and the United States—the travel times for both direct flights (shown in blue) and indirect flights (not shown), including the mean transfer time of up to a maximum of one stopover, were used as negative similarities (*3*). (**C**) Seven exemplars identified by affinity propagation are color-coded, and the assignments of other cities to these exemplars is shown. Cities located quite near to exemplar cities may be members of other more distant exemplars due to the lack of direct flights between them (e.g., Atlantic City is 100 km from Philadelphia, but is closer in flight time to Atlanta). (**D**) The inset shows that the Canada-USA border roughly divides the Toronto and Philadelphia clusters, due to a larger availability of domestic flights compared to international flights. However, this is not the case on the west coast as shown in (**E**), because extraordinarily frequent airline service between Vancouver and Seattle connects Canadian cities in the northwest to Seattle.

References and Notes

1. J. MacQueen, in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, L. Le Cam, J. Neyman, Eds. (Univ. of California Press, Berkeley, CA, 1967), vol. 1, pp. 281–297.
2. Supporting material is available on *Science* Online.
3. Software implementations of affinity propagation, along with the data sets and similarities used to obtain the results described in this manuscript, are available at www.psi.toronto.edu/affinitypropagation.
4. B. J. Frey *et al.*, *Nat. Genet.* **37**, 991 (2005).
5. K. D. Pruitt, T. Tatusova, D. R. Maglott, *Nucleic Acids Res.* **31**, 34 (2003).
6. C. D. Manning, H. Schutze, *Foundations of Statistical Natural Language Processing* (MIT Press, Cambridge, MA, 1999).
7. J. J. Hopfield, *Proc. Natl. Acad. Sci. U.S.A.* **79**, 2554 (1982).
8. M. Charikar, S. Guha, A. Tardos, D. B. Shmoys, *J. Comput. Syst. Sci.* **65**, 129 (2002).
9. J. S. Yedidia, W. T. Freeman, Y. Weiss, *IEEE Trans. Inf. Theory* **51**, 2282 (2005).
10. F. R. Kschischang, B. J. Frey, H.-A. Loeliger, *IEEE Trans. Inf. Theory* **47**, 498 (2001).
11. A. P. Dempster, N. M. Laird, D. B. Rubin, *Proc. R. Stat. Soc. B* **39**, 1 (1977).
12. S. Dasgupta, L. J. Schulman, *Proc. 16th Conf. UAI* (Morgan Kaufman, San Francisco, CA, 2000), pp. 152–159.
13. S. Jain, R. M. Neal, *J. Comput. Graph. Stat.* **13**, 158 (2004).
14. R. R. Sokal, C. D. Michener, *Univ. Kans. Sci. Bull.* **38**, 1409 (1958).
15. J. Shi, J. Malik, *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 888 (2000).
16. M. Mézard, *Science* **301**, 1685 (2003).
17. J. Pearl, *Probabilistic Reasoning in Intelligent Systems* (Morgan Kaufman, San Mateo, CA, 1988).
18. D. J. C. MacKay, *IEEE Trans. Inf. Theory* **45**, 399 (1999).
19. C. Berrou, A. Glavieux, *IEEE Trans. Commun.* **44**, 1261 (1996).
20. M. Mézard, G. Parisi, R. Zecchina, *Science* **297**, 812 (2002).
21. B. J. Frey, R. Koetter, N. Petrovic, in *Proc. 14th Conf. NIPS* (MIT Press, Cambridge, MA, 2002), pp. 737–743.
22. T. Meltzer, C. Yanover, Y. Weiss, in *Proc. 10th Conf. ICCV* (IEEE Computer Society Press, Los Alamitos, CA, 2005), pp. 428–435.

23. We thank B. Freeman, G. Hinton, R. Koetter, Y. LeCun, S. Roweis, and Y. Weiss for helpful discussions and P. Dayan, G. Hinton, D. MacKay, M. Mezard, S. Roweis, and C. Tomasi for comments on a previous draft of this manuscript. We acknowledge funding from Natural Sciences and Engineering Research Council of Canada, Genome Canada/Ontario Genomics Institute, and the Canadian Institutes of Health Research. B.J.F. is a Fellow of the Canadian Institute for Advanced Research.

Supporting Online Material

www.sciencemag.org/cgi/content/full/1136800/DC1
SOM Text
Figs. S1 to S3
References

26 October 2006; accepted 26 December 2006
Published online 11 January 2007;
10.1126/science.1136800
Include this information when citing this paper.

Clustering by Passing Messages Between Data Points

Brendan J. Frey and Delbert Dueck (January 11, 2007)
Science **315** (5814), 972-976. [doi: 10.1126/science.1136800]
originally published online January 11, 2007

Editor's Summary

Science (print ISSN 0036-8075; online ISSN 1095-9203) is published weekly, except the last week in December, by the American Association for the Advancement of Science, 1200 New York Avenue NW, Washington, DC 20005.

AP 聚类算法

1.分类与聚类

1.1 分类算法简介

分类(classification)是找出描述并区分数据类或概念的模型(或函数)，以便能够使用模型预测类标记未知的对象类。

在分类算法中输入的数据，或称训练集（Training Set），是一条条的数据库记录（Record）组成的。每一条记录包含若干条属性（Attribute），组成一个特征向量。训练集的每条记录还有一个特定的类标签（Class Label）与之对应。该类标签是系统的输入，通常是以往的一些经验数据。一个具体样本的形式可为样本向量:(v_1, v_2, ..., v_n; c)。在这里 v_i 表示字段值,c 表示类别。

分类的目的是：分析输入的数据，通过--在训练集中的数据表现出来的特性，为每一个类找到一种准确的描述或者模型。这种描述常常用谓词表示。由此生成的类描述用来对未来的测试数据进行分类。尽管这些未来的测试数据的类标签是未知的，我们仍可以由此预测这些新数据所属的类。注意是预测，而不能肯定。我们也可以由此对数据中的每一个类有更好的理解。也就是说：我们获得了对这个类的知识。

下面对分类流程作个简要描述：

训练：训练集——>特征选取——>训练——>分类器

分类：新样本——>特征选取——>分类——>判决

常见的分类算法有：**决策树、KNN 法(K-Nearest Neighbor)、SVM 法、VSM 法、Bayes 法、神经网络**等。

1.2 聚类算法简介

聚类(clustering)是指根据“物以类聚”的原理，将本身没有类别的样本聚集成不同的组，这样的一组数据对象的集合叫做簇，并且对每一个这样的簇进行描述的过程。

与分类规则不同，进行聚类前并不知道将要划分成几个组和什么样的组，也不知道根据哪些空间区分规则来定义组。

它的目的是使得属于同一个簇的样本之间应该彼此相似，而不同簇的样本应

该足够不相似。

聚类分析的算法可以分为：**划分法（Partitioning Methods）、层次法（Hierarchical Methods）、基于密度的方法（density-based methods）、基于网格的方法（grid-based methods）、基于模型的方法（Model-Based Methods）**。

经典的 K-means 和 K-centers 都是划分法。

分类与聚类的区别

聚类分析也称无监督学习或无指导学习，聚类的样本没有标记，需要由聚类学习算法来自动确定;在分类中，对于目标数据库中存在哪些类是知道的，要做的就是将每一条记录分别属于哪一类标记出来。聚类学习是观察式学习，而不是示例式学习。

可以说**聚类分析可以作为分类分析的一个预处理步骤**。

2.K-MEANS 算法

k-means 算法接受输入量 k；然后将 n 个数据对象划分为 k 个聚类以便使得所获得的聚类满足：同一聚类中的对象相似度较高；而不同聚类中的对象相似度较低。簇的相似度是关于簇中对象的均值度量，可以看作簇的质心(centriod)或重心(center of gravity)。

k-means 算法的工作过程说明如下：首先从 n 个数据对象任意选择 k 个对象作为初始聚类中心；而对于所剩下其它对象，则根据它们与这些聚类中心的相似度（距离），分别将它们分配给与其最相似的（聚类中心所代表的）聚类；然后再计算每个所获新聚类的聚类中心（该聚类中所有对象的均值）；不断重复这一过程直到标准测度函数开始收敛为止。一般都采用均方差作为标准测度函数，其定义如下：

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} | p - m_i |^2 \qquad (1)$$

其中，E 是数据集中所有对象的平方误差和，p 是空间中的点，表示给定对象，m_i 是簇 C_i 的均值（p 和 m_i 都是多维的）。换句话说，对于每个簇中的每个对象，求对象到其簇中心距离的平方，然后求和。这个准则试图使生成的 k 个结果簇尽可能的紧凑和独立。

例 1:我们在二维空间中随机的生成 20 个数据点，将聚类数目指定为 5 个，并随机生成一个聚类中心(用“×”来标注)，根据对象与簇中心的距离，每个对象分成于最近的簇。初始示例图如下：

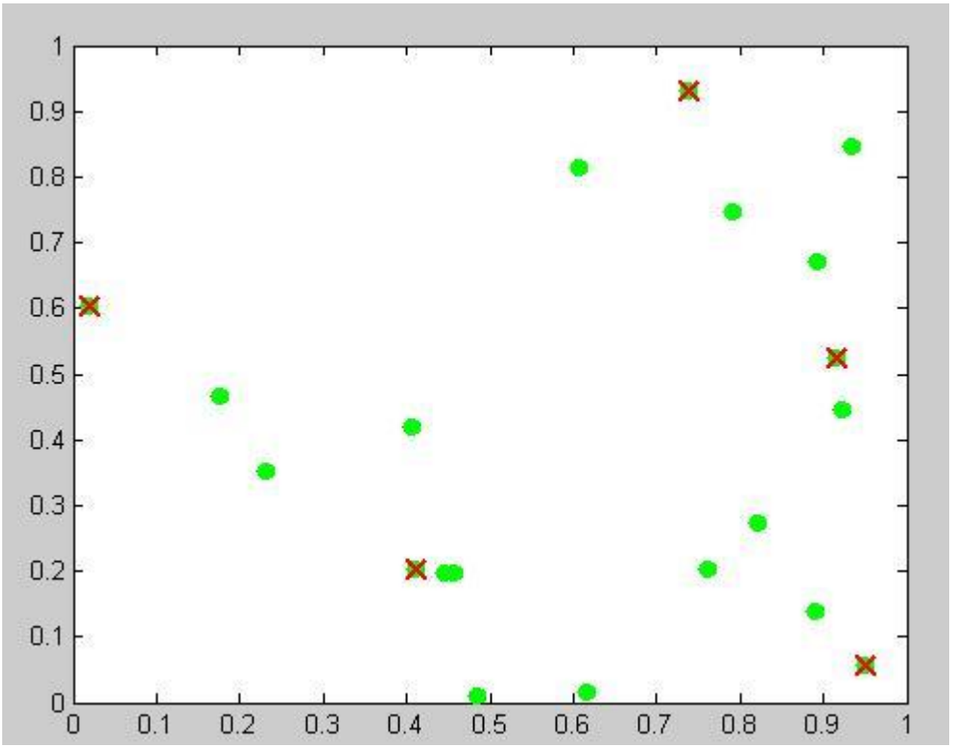

图 1.随机生成的数据点及初始聚类中心示例图

下一步，更新簇中心。也就是说，根据簇中的当前对象，重新计算每个簇的均值。使用这些新的簇中心，将对象重新分成到簇中心最近的簇中。

不断迭代上面的过程，直到簇中对象的重新分布不再发生，处理结束。最终的聚类结果示例图如下：

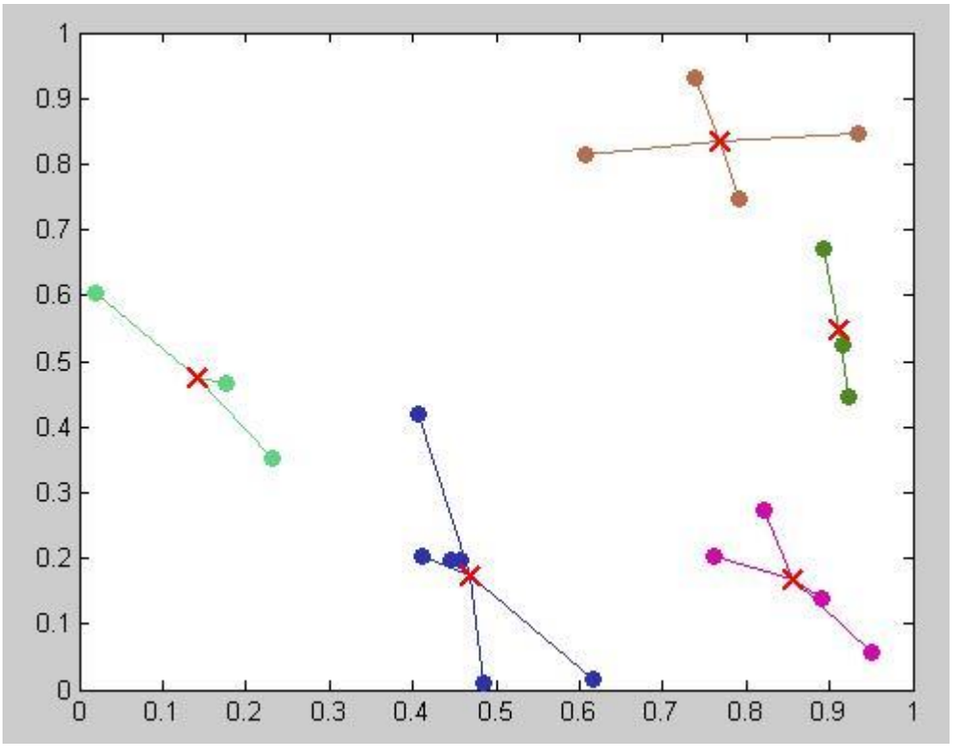

图 2. 最终聚类结果示例图

从上图中我们可以看到，最终的聚类结果受初始聚类中心的影响很大，而且最后的簇质心点不一定是在数据点上。

K 均值算法试图确定最小化平方误差的 k 个划分。当结果簇是紧凑的，并且簇与簇之间明显分离时，它的效果较好。**对处理大数据集，该算法是相对可伸缩的和有效率的**，因为它的计算复杂度是 O（nkt），其中 n 是对象的总数，k 是簇的个数，t 是迭代的次数。通常地，k<<n 并且 t<<n。该方法经常终止于局部最优解。

然而，只有**当簇均值有定义的情况下 k 均值方法才能使用**。在某些应用中，例如当涉及具有分类属性的数据时，均值可能无定义。用户必须事先给出要生成的簇的数目 k 可以算是该方法的缺点。**K 均值方法不适合于发现非凸形状的簇**，或者**大小差别很大的簇**。此外，它**对于噪声和离群点数据是敏感**的，因为少量的这类数据能够对均值产生极大的影响。

3.AP 算法

Affinity Propagation(AP)聚类是最近在Science杂志上提出的一种新的聚类算法。它根据N个数据点之间的相似度进行聚类,这些相似度可以是对称的,即两个数据点互相之间的相似度一样(如欧氏距离);也可以是不对称的,即两个数据点互相之间的相似度不等。这些相似度组成N ×N的相似度矩阵S(其中N为有N个数据点)。AP

算法不需要事先指定聚类数目,相反它将所有的数据点都作为潜在的聚类中心,称之为**exemplar**。

以S矩阵的对角线上的数值s(k,k)作为k点能否成为聚类中心的评判标准,这意味着该值越大,这个点成为聚类中心的可能性也就越大,这个值又称作**参考度p (preference)** 。聚类的数量受到参考度p的影响,如果认为每个数据点都有可能作为聚类中心,那么p就应取相同的值。如果取输入的相似度的均值作为p的值,得到聚类数量是中等的。如果取最小值,得到类数较少的聚类。AP算法中传递两种类型的消息,(responsiility)和(availability)。r(i,k)表示从点i发送到候选聚类中心k的数值消息,反映**k点是否适合作为i点的聚类中心**。a(i,k)则从候选聚类中心k发送到i的数值消息,反映**i点是否选择k作为其聚类中心**。r(i,k)与a (i,k)越强,则k点作为聚类中心的可能性就越大,并且i点隶属于以k点为聚类中心的聚类的可能性也越大。**AP算法通过迭代过程不断更新每一个点的吸引度和归属度值,直到产生m个高质量的exemplar,同时将其余的数据点分配到相应的聚类中**。

在这里介绍几个文中常出现的名词：

exemplar：指的是聚类中心。

similarity：数据点i和点j的相似度记为S(i，j)。是指点j作为点i的聚类中心的相似度。一般使用欧氏距离来计算，如$-\|(x_i-x_j)^2+(y_i-y_j)^2\|$。文中，所有点与点的相似度值全部取为负值。因为我们可以看到，相似度值越大说明点与点的距离越近，便于后面的比较计算。

preference：数据点i的参考度称**为P(i)**或**S(i,i)**。是指点i作为聚类中心的参考度。一般取S相似度值的中值。

Responsibility:R(i,k)用来描述点k适合作为数据点i的聚类中心的程度。

Availability:A(i,k)用来描述点i选择点k作为其聚类中心的适合程度。

两者的关系如下图：

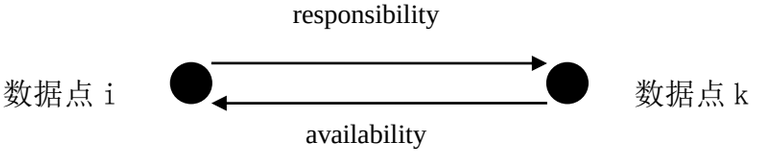

图3. 数据点之间传递消息示意图

下面是R与A的计算公式：

$$R(i,k)=S(i,k)-\max\{A(i,j)+S(i,j)\}(j\in\{1,2,\ldots\ldots,N,但j\neq k\}) \quad (2)$$

$$A(i,k)=\min\{0,R(k,k)+\sum_{j}\{\max(0,R(j,k))\}\} \quad (j\in\{1,2,\ldots\ldots,N,但j\neq i且j\neq k\}) \quad (3)$$

$$R(k,k)=P(k)-\max\{A(k,j)+S(k,j)\}\ (j\in\{1,2,\ldots\ldots,N,但j\neq k\}) \quad (4)$$

由上面的公式可以看出，当P(k)较大使得R(k, k)较大时, A(i, k)也较大,从而类代表k作为最终聚类中心的可能性较大;同样, 当越多的P(i)较大时,越多的类代表倾向于成为最终的聚类中心。因此,增大或减小P可以增加或减少AP输出的聚类数目。

Damping factor(阻尼系数):主要是起收敛作用的。文中讲述，每次迭代，吸引度 R_i 和归属度 A_i 要与上一次的 R_{i-1} 和 A_{i-1} 进行加权更新。公式如下：

$$R_i=(1\text{-lam})*R_i+\text{lam}*R_{i-1} \quad (5)$$

$$A_i=(1\text{-lam})*A_i+\text{lam}*A_{i-1} \quad (6)$$

其中，$\text{lam}\in[0.5,1)$。

AP算法的具体工作过程如下：先计算N个点之间的相似度值，将值放在S矩阵中，再选取P值(一般取S的中值)。设置一个最大迭代次数(文中设默认值为1000)，迭代过程开始后，计算每一次的R值和A值，根据R(k,k)+A(k,k)值来判断是否为聚类中心(文中指定当(R(k,k)+A(k,k))＞0时认为是一个聚类中心)，当迭代次数超过最大值(即maxits值)或者当聚类中心连续多少次迭代不发生改变(即convits值)时终止计算(文中设定连续50次迭代过程不发生改变是终止计算)。

例2：我们在二维空间中随机的生成20个数据点，将P值设为S矩阵的中值，将convits值设置成20, maxits值设置成1000，用AP算法进行计算，最终聚类结果如下图：

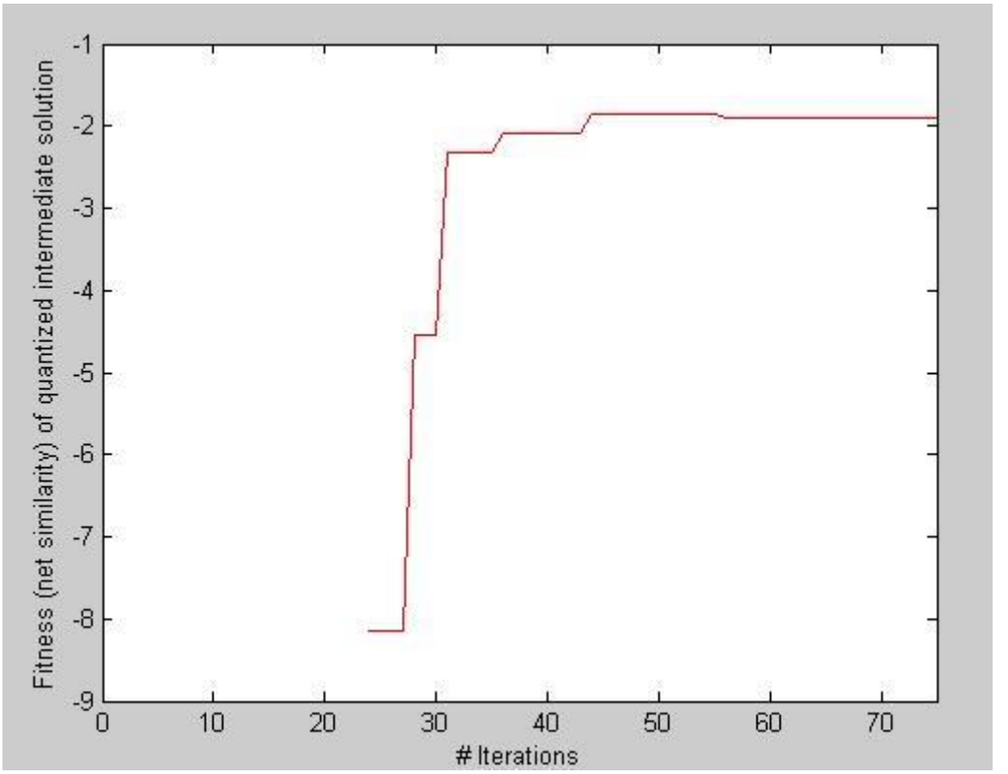

图4. AP算法迭代过程示意图

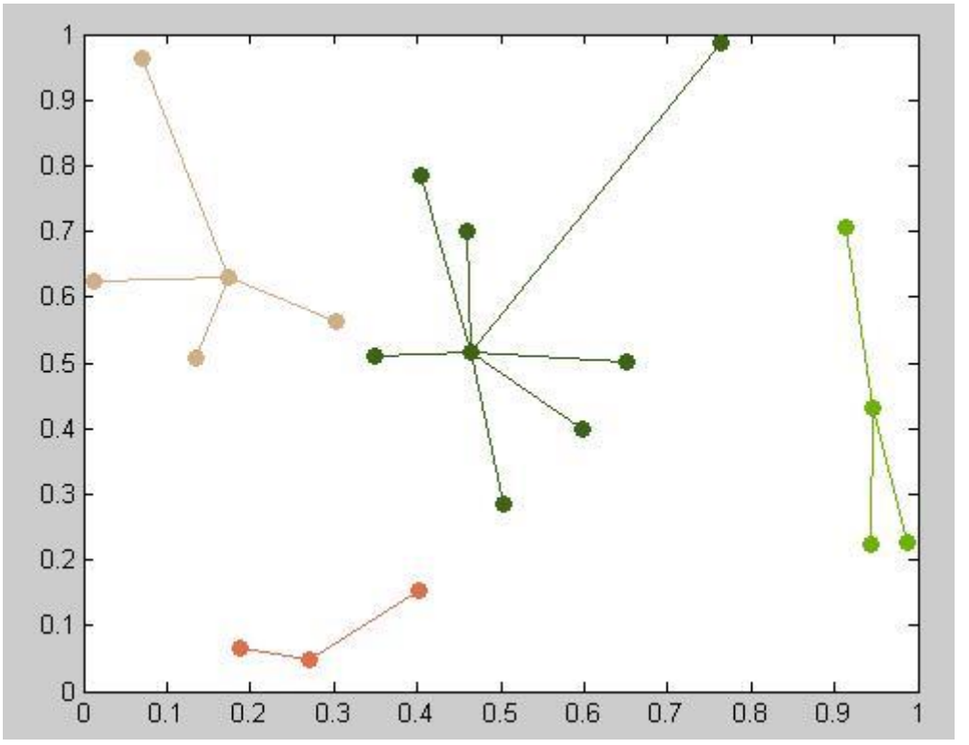

图5 最终聚类结果示意图

在AP 算法中，迭代次数和聚类数目主要受到两个参数的影响。其中，聚数数目主要受preference值（负值）的影响。下面对同一组数据集(200个数据点)进行计

算，取不同的preference值得到的聚类数目如下：

Preference值	聚类数目
$\frac{median(S)}{2}$	16
median(S)	11
2×median(S)	8

表1.不同的preference得到的聚类数目比较

由表1，我们可以看出，当preference越大时，得到的聚类数目越多。

当取不同的**lam（阻尼系数）**值时，迭代次数和迭代过程中数据的摆动都会有很大的不同，下面同样是对同一组数据集(200个数据点)进行计算，取有代表性的两个值（0.5和0.9）进行比较结果如下：

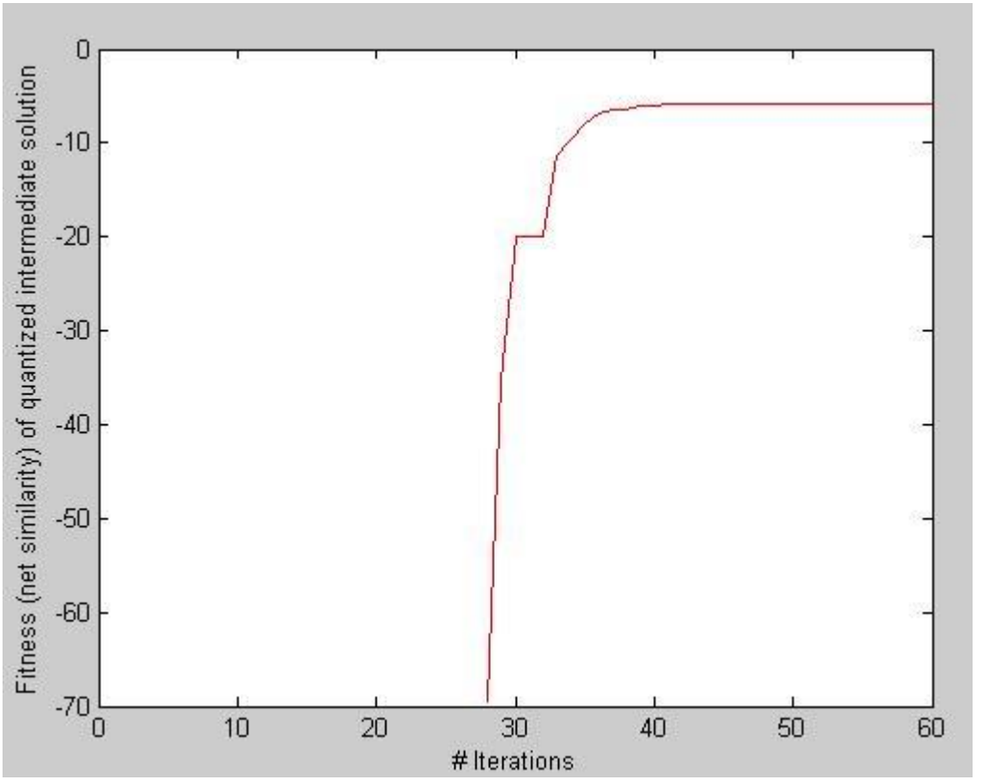

图6 lam取0.9时的迭代示意图

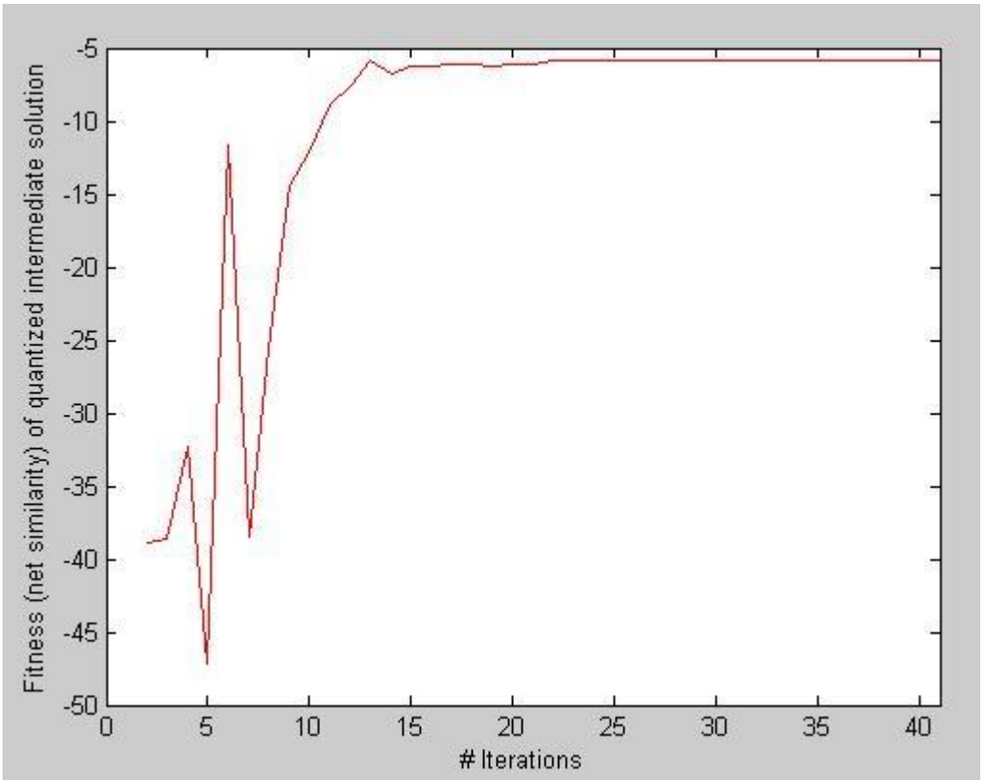

图 7.lam取0.5时的迭代示意图

从上面两个图对比中我们可以发现，当lam值越小时，迭代次数会减少，但是迭代过程中net Similarity值波动会很大，当要聚类的数据点比较大时，这样难于收敛。当lam值较大时，迭代次数会增加，但是总的net Similarity比较平稳。

根据式（5）和式（6），我们也可以看到，每一次迭代的 R_i 和 A_i 受到lam的影响。当lam取较小的值时，R_i 和 A_i 相比上一次迭代的 R_{i-1} 和 A_{i-1} 会发生较大的变化，这也是为什么net Similarity值摆动比较大的原因;当lam取较大值时，R_i 和 A_i 和上一次迭代的 R_{i-1} 和 A_{i-1} 比较接近，这也是导致迭代次数比较多的原因。

正是因为如此，有人提出了**自适应仿射传播聚类**（在文献2中可以看到），文中主要提出了如何**根据数据集自动生成preference值和lam值**的方法。

4. k-means 算法与 AP 算法比较

例 3：下面，我们随机在二维空间中生成 50 个数据点，分别用上面讲述的两种聚类算法进行聚类计算。

我们先进行 AP 算法聚类，将生成的聚类数量用于 k-means 算法中，将结果示意图进行比较，具体结果如下：

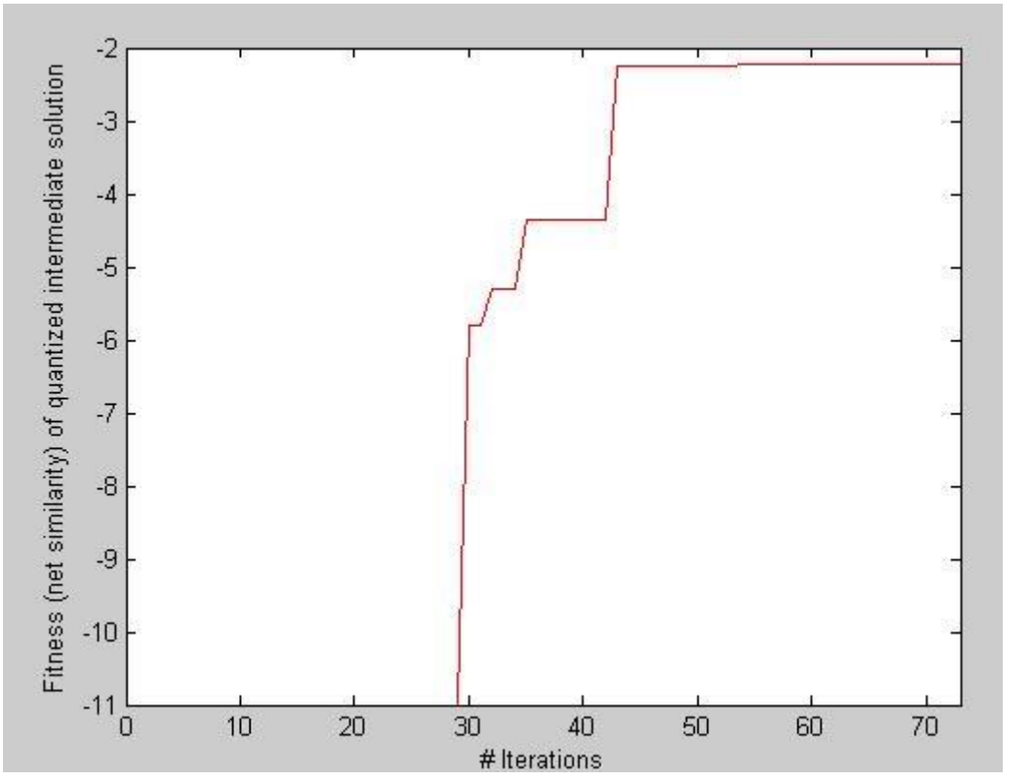

图 8.AP 算法迭代过程

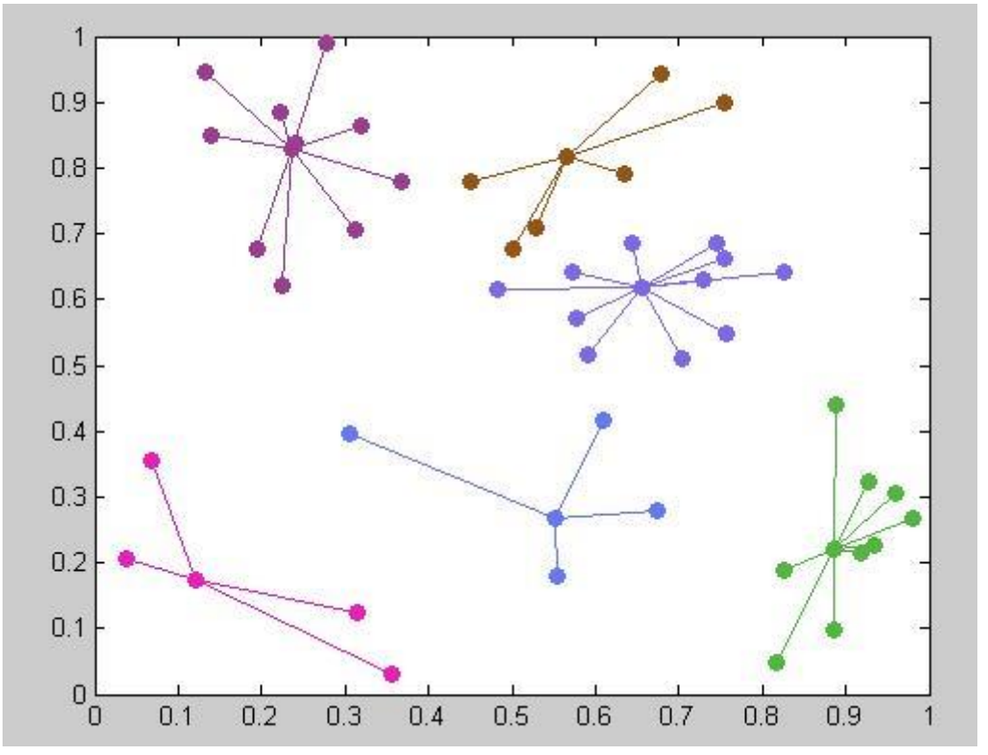

图 9.AP 算法最终计算结果

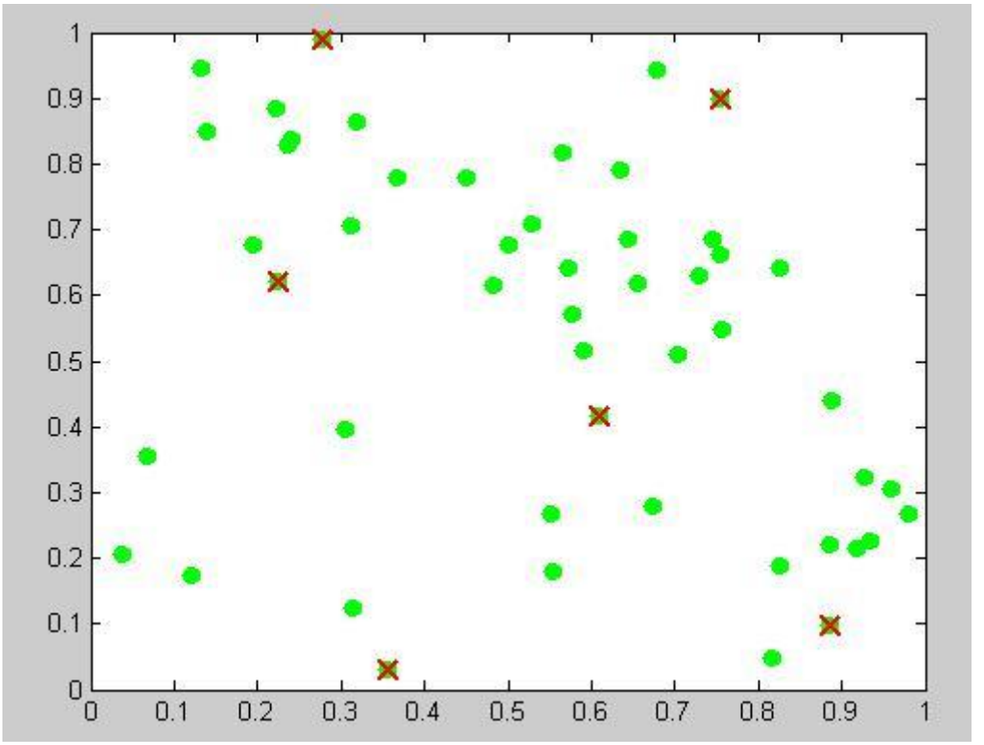

图 10. k-means 算法初始聚类中心示意图

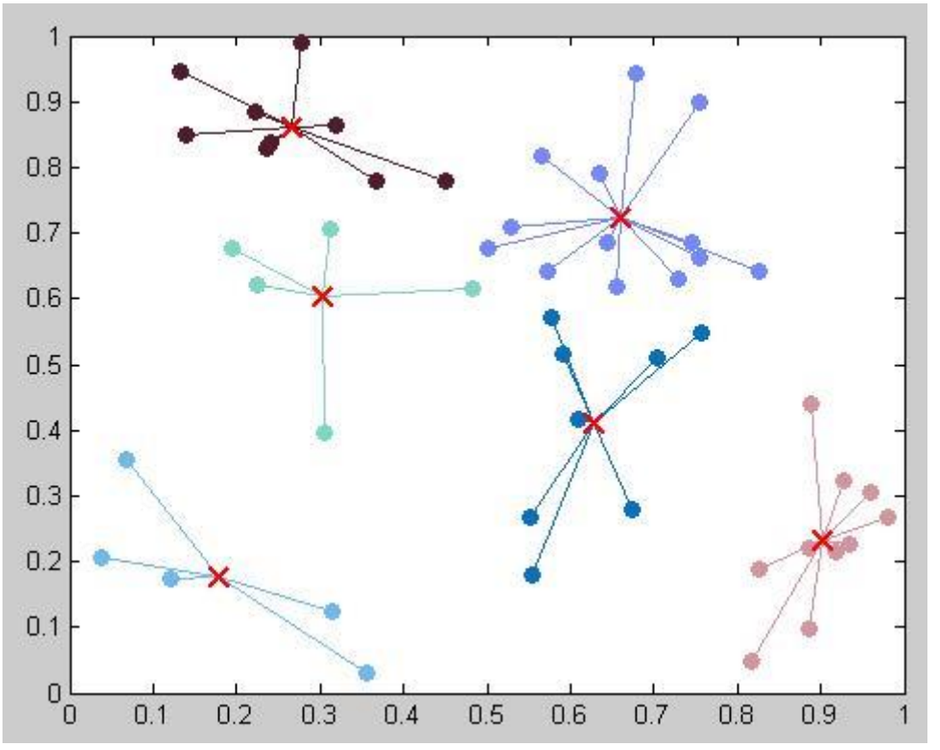

图 11.k-means 算法最终聚类结果

5.总结与展望

k-means 算法对于离散和噪声数据比较敏感，对于初始聚类中心的选择很关

键，因为初始聚类中心选择的好坏直接影响到聚类结果，而且这个算法要求进行聚类时输入聚类数目，这也可以说是对聚类算法的一种限制。不过，这种算法**运行速度相对于 AP 算法要快一些**，因此，对于那些**小**而且数据比较密集的数据集来说，这种聚类算法还是比较好的。

AP 算法对于 **P 值**的选取比较关键，这个值的大小，直接影响都最后的聚类数量。值越大，生成的聚类数越多，反之如此。而且，那个**阻尼系数(lam)**迭代也是很关键的。在文献[2]中有人提及，此算法可能会出现数据震荡现象，即迭代过程中产生的聚类数不断发生变化不能收敛。增大 lam 可消除震荡现象。但根据式[5]和式[6]来看，一味的增大 lam 会使 R 和 A 的更新变的缓慢，增加了计算时间。因此，如何选取一个合适 lam 的来进行计算也成了一个提升算法运行速度的重要因素。

将 AP 算法运用于**图像检索系统**中来进行**初始分类**，我觉得挺有用的。这个想法还有待实现。

6. 参考文献

[1] Frey B J, Dueck D. Clustering by passing messages between data points. Science, 2007, 315(5814): 972~976.

[2] 王开军 张军英等 自适应仿射传播聚类 自动化学报 2007年12月 第33 卷第12 期

[3] Jiawei Han Micheline Kamber 范明 孟小峰 译 数据挖掘概念与技术 机械工业出版社 2008年 251~265.