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-

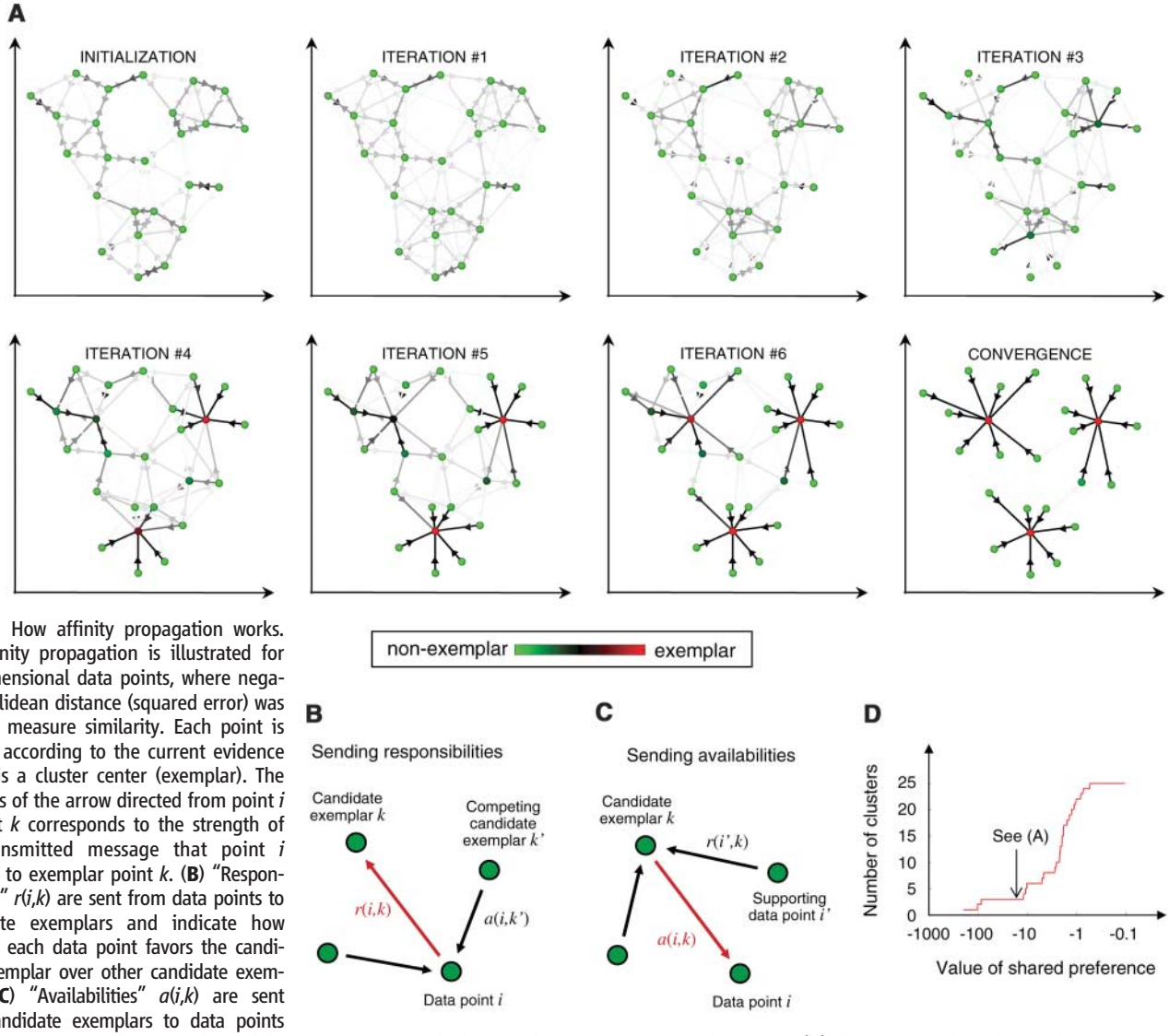


Fig. 1. How affinity propagation works. (**A**) Affinity propagation is illustrated for two-dimensional data points, where negative Euclidean distance (squared error) was used to measure similarity. Each point is colored according to the current evidence that it is a cluster center (exemplar). The darkness 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

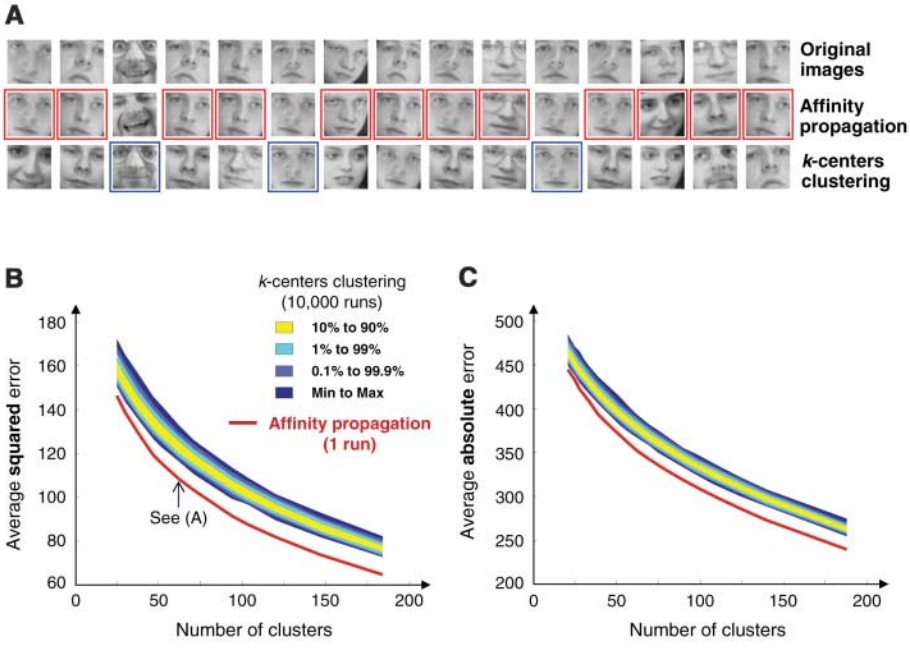


Fig. 2. Clustering faces. Exemplars minimizing the standard squared error measure of similarity were identified from 900 normalized face images (*3*). For a common preference of −600, affinity propagation found 62 clusters, and the average squared error was 108. For comparison, the best of 100 runs of *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*).

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

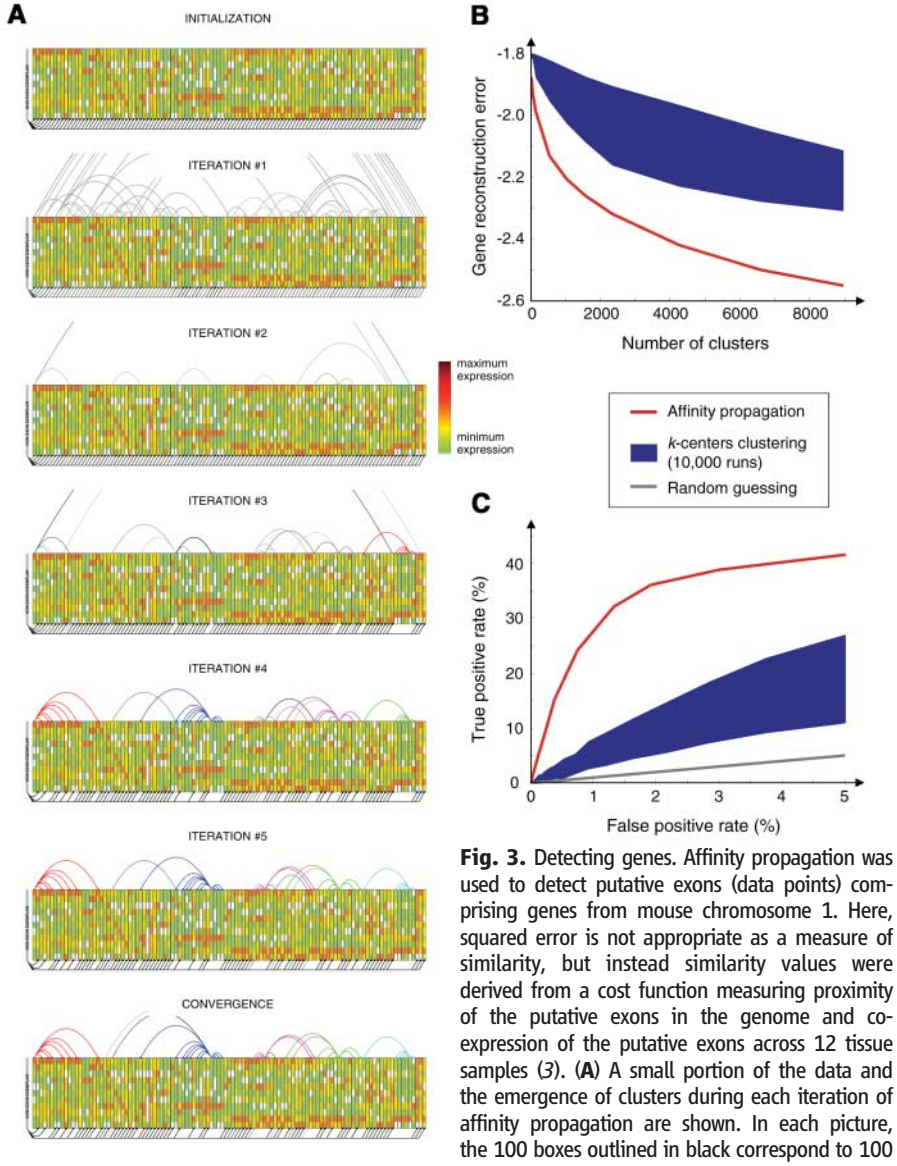


Fig. 3. Detecting genes. Affinity propagation was used to detect putative exons (data points) comprising genes from mouse chromosome 1. Here, squared error is not appropriate as a measure of similarity, but instead similarity values were derived from a cost function measuring proximity of the putative exons in the genome and coexpression 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.

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.

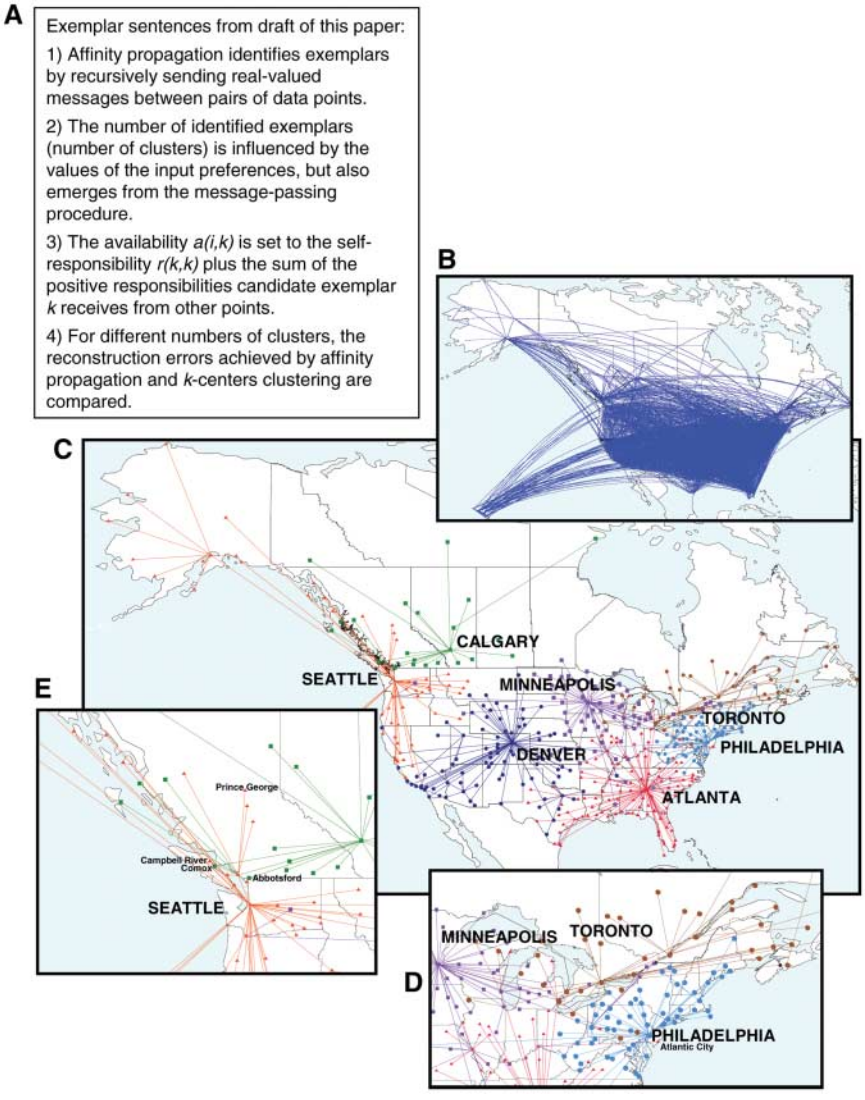


Fig. 4. Identifying key sentences and air-travel routing. Affinity propagation can be used to explore the identification of exemplars on the basis of nonstandard optimization criteria. (**A**) Similarities between pairs of sentences in a draft of this manuscript were constructed by matching words. Four 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.

(*14*). Future work will surely focus on why more of apparently the same neurons seem to have a different function.

References

1. J. O'Keefe, *Exp. Neurol.* **51**, 78 (1976).
2. M. A. Wilson, B. L. McNaughton, *Science* **261**, 1055 (1993).
3. J. K. Leutgeb, S. Leutgeb, M.-B. Moser, E. I. Moser, *Science* **315**, 961 (2007).
4. T. J. Wills, C. Lever, F. Cacucci, N. Burgess, J. O'Keefe, *Science* **308**, 873 (2005).
5. J. J. Hopfield, *Proc. Natl. Acad. Sci. U.S.A.* **79**, 2554 (1982).
6. J. K. Leutgeb *et al.*, *Neuron* **48**, 345 (2005).
7. R. M. Hayman, S. Chakraborty, M. I. Anderson, K. J. Jeffery, *Eur. J. Neurosci.* **18**, 2825 (2003).
8. K. D. Harris, J. Csicsvari, H. Hirase, G. Dragoi, G. Buzsaki, *Nature* **424**, 552 (2003).
9. E. Pastalkova *et al.*, *Science* **313**, 1141 (2006).
10. J. R. Whitlock, A. J. Heynen, M. G. Shuler, M. F. Bear, *Science* **313**, 1093 (2006).
11. R. U. Muller, J. L. Kubie, *J. Neurosci.* **7**, 1951 (1987).
12. M. H. Fyhn, T. F. Hafting, A. Treves, E. I. Moser, M. B. Moser, *Soc. Neurosci. Abstr.* **68**, 9 (2006).
13. M. K. Chawla *et al.*, *Hippocampus* **15**, 579 (2005).
14. E. Gould, A. Beylin, P. Tanapat, A. Reeves, T. J. Shors, *Nat. Neurosci.* **2**, 260 (1999).

110.1126/science.1139146

COMPUTER SCIENCE

Where Are the Exemplars?

A fast way of finding representative examples in complex data sets may be applicable to a wide range of difficult problems.

Marc Mézard

As a flood of data pours from scientific and medical experiments, researchers crave more efficient computational methods to organize and analyze it. When dealing with large, noisy data sets, scientists often use a computational method that looks for data clusters. In the case of gene expression with tens of thousands of sequences, for example, the clusters would be groups of genes with similar patterns of expression. On page 972 of this issue, Frey and Dueck propose a new method for finding an optimal set of clusters (*1*). Their algorithm detects special data points called exemplars, and connects every data point to the exemplar that best represents it. In principle, finding an optimal set of exemplars is a hard problem, but this algorithm is able to efficiently and quickly handle very large problems (such as grouping 75,000 DNA segments into 2000 clusters). An analysis that would normally take hundreds of hours of computer time might now be done in a few minutes.

Detecting exemplars goes beyond simple clustering, as the exemplars themselves store compressed information. An example with a broad range of possible applications is found in the statistical analysis of language. For instance, take your last scientific paper (and no, I don't really suggest that it is a large, noisy data set) and consider each sentence to be a data point. The similarity between any two sentences can be computed with standard information theory methods (that is, the similarity increases when the sentences include the same words). Knowing the similarities, one can detect the exemplary sentences in the paper, which provide an optimally condensed description. If you are a hasty reader, you can thus go directly to Fig. 4 of Frey and Dueck's report and find the best summary of their own paper in four sentences. But understanding the method requires a bit more effort.

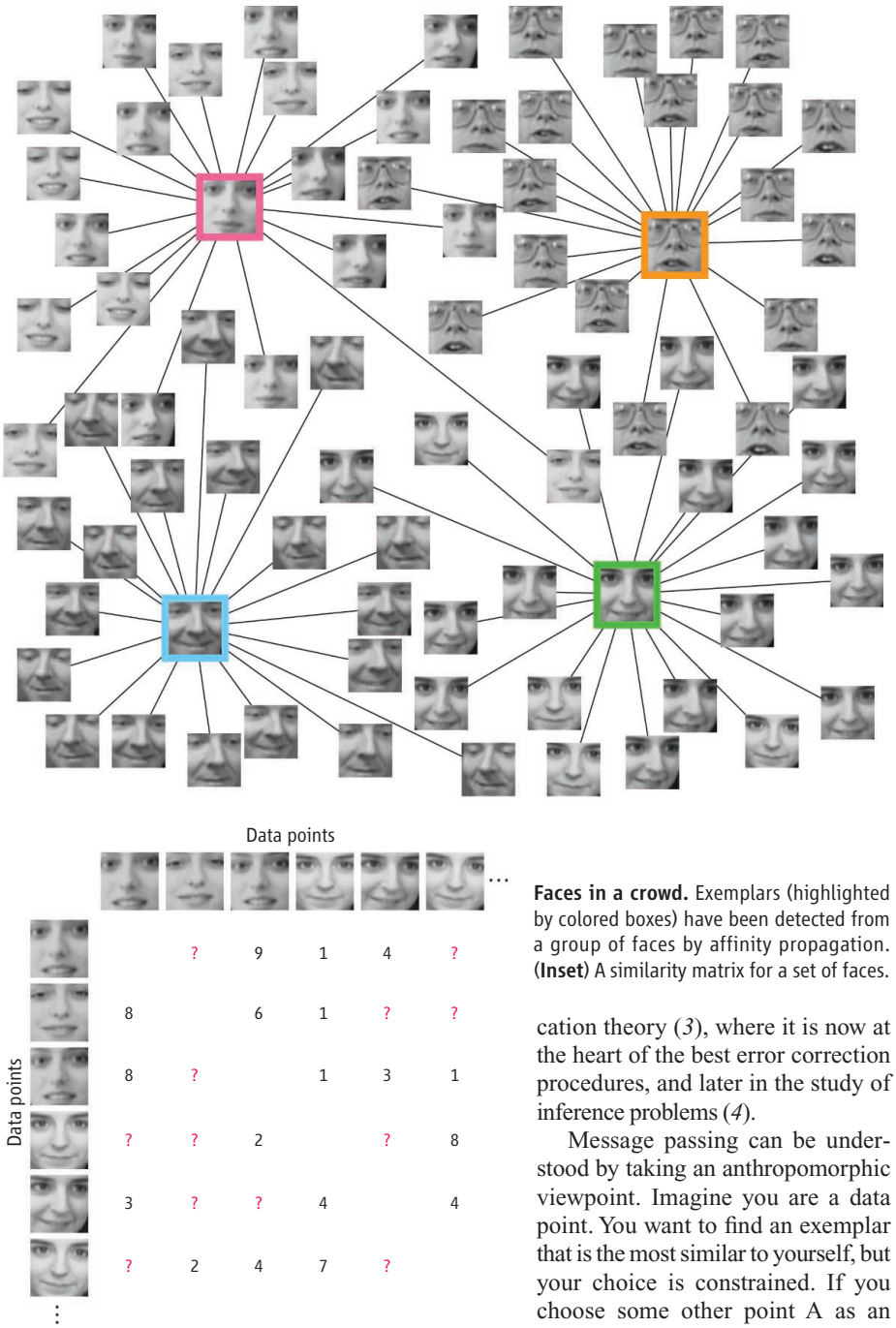
Such methods start with the construction of a similarity matrix, a table of numbers that establishes the relationship of each data point to every other data point. As we saw in the semantics example, $S(B, A)$ is a number that measures how well the data point A represents point B [and it is not necessarily equal to $S(A, B)$]. The optimal set of exemplars is the one for which the sum of similarities of each point to its exemplar is maximized. In the usual clustering methods (*2*), one decides a priori on the number of exemplars, and then tries to find them by iterative refinement, starting from a random initial choice.

The method of Frey and Dueck, called affinity propagation, does not fix the number of exemplars. Instead, one must choose for each point B a number $P(B)$ that characterizes

CREDIT: SCALA/ART RESOURCE

Caravaggio's "Vocazione di San Matteo." How to choose an exemplar through message passing. The messages are exchanged in the directions of the fingers and of the glances, leading to the recognition of San Matteo as the "exemplar."

The author is at the Centre National de la Recherche Scientifique and Laboratoire de Physique Théorique et Modeles Statistiques, Université Paris Sud, 91405 Orsay, France. E-mail: mezard@lptms.u-psud.fr



Faces in a crowd. Exemplars (highlighted by colored boxes) have been detected from a group of faces by affinity propagation. (**Inset**) A similarity matrix for a set of faces.

the a priori knowledge of how good point B is as an exemplar. In most cases all points are equally suitable, so all the numbers take the same value P. This quantity provides a control parameter: The larger P, the more exemplars one is likely to find.

Affinity propagation is known in computer science as a message-passing algorithm (see the first figure) and it aims at maximizing the net similarity. It is in fact an application of a method called "belief propagation," which was invented at least twice: first in communication theory (*3*), where it is now at the heart of the best error correction procedures, and later in the study of inference problems (*4*).

Message passing can be understood by taking an anthropomorphic viewpoint. Imagine you are a data point. You want to find an exemplar that is the most similar to yourself, but your choice is constrained. If you choose some other point A as an exemplar, then A must also decide to be its own exemplar. This creates one constraint per data point, establishing a large network of constraints that must all be satisfied. When the net similarity is maximized with all constraints satisfied, the set of actual exemplars emerges.

Now imagine that next to each point stands a guardian angel telling whether someone else has chosen that point as an exemplar or not. An approximate solution of the complicated web of conflicting constraints is obtained by having all of these characters talk to each other. At a given time, all angels send messages to all points, and all points answer to all angels. One data point tells the angel of every other point his ranked list of favorite exemplars. An angel tells every other point the degree of compatibility of his list with the angel's constraints. Every sent message is evaluated through a simple computation on the basis of the received messages and the similarity matrix. After several message-passing rounds, all the characters reach an agreement and every point knows its exemplar. In practice, the running time of this algorithm scales linearly with the number of similarities.

As an example, affinity propagation can be a powerful method to extract representative faces from a gallery of images (see the second figure). The input is a list of numerical similarities between pairs of data points, which may be measured, computed using a model, or, in the present example, set by visual inspection (missing similarity values indicated with question marks are accepted by the algorithm). Each face is a data point that exchanges messages with all other faces and their guardian angels. After a few iterations of message passing, a global agreement is reached on the set of exemplars.

Such message-passing methods have been shown to be remarkably efficient in many hard problems that include error correction, learning in neural networks, computer vision, and determining the satisfiability of logical formulas. In many cases they are the best available algorithms, and this new application to cluster analysis looks very powerful. Understanding their limits is a main open challenge. At the lowest level this means controlling the convergence properties or the quality of the approximate solutions that they find. A more ambitious goal is to characterize the problems where they can be useful. The concepts and methods developed in statistical physics to study collective behavior offer the most promising perspective in this respect. In physics terms, belief propagation (and therefore affinity propagation) is a mean field–type method (*5*). That is, the complex interaction of a given object (a data point) with all of the others is approximated by an average effective interaction. Although this works well in most cases, it may get into trouble when the system gets close to a phase transition (*6*), where some correlations become extremely long-ranged. The appropriate modification, which requires using more sophisticated messages, has been worked out in some special cases (*7*), but again its full range of applicability is still to be found.

Along with its pedagogical virtue, the anthropomorphic explanation of message passing also underlines its main features. This

strategy can find an excellent approximate solution to some of the most difficult computational problems with a very simple recipe: It uses basic messages which are exchanged in a distributed system, together with simple update rules that are purely local. This realizes in practice a new scheme of computation, based on distributed simple elements that operate in parallel, in the spirit of neurocomputation. One might expect to find that some of its principles are at work in living organisms or social systems. Each new successful application of message passing, such as affinity propagation, thus adds to our understanding of complex systems.

References

1. B. J. Frey, D. Dueck, *Science* **315**, 972 (2007); published online 11 January 2007 (10.1126/science.1136800).
2. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, L. Le Cam, J. Neyman, Eds. (Univ. of California Press, Berkeley, CA, 1967), p. 281.
3. R. G. Gallager, *Low Density Parity Check Codes* (MIT Press, Cambridge, MA, 1963).
4. J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo, CA, 1988).
5. J. S. Yedidia, W. T. Freeman, Y. Weiss, *IEEE Trans. Infor. Theory* **51**, 2282 (2005).
6. O. Dubois, R. Monasson, B. Selman, R. Zecchina, Eds., special issue on Phase Transitions in Combinatorial Problems, *Theor. Comp. Sci.* **265** (2001).
7. M. Mézard, G. Parisi, R. Zecchina, *Science* **297**, 812 (2002); published online 27 June 2002 (10.1126/science.1073287).

10.1126/science.1139678

GEOLOGY

On the Origins of Granites

Granites make up a large part of the continental crust. New data reveal their complex and diverse formation history, calling for a revision of the geological histories of many granites.

John M. Eiler

Geology spent the 19th and much of the 20th century fighting a scientific civil war over the origin of granites—the coarsely crystalline, feldspar-rich rocks that make such excellent building stones and kitchen counters. The ultimate losers (*1*) held that granites precipitated from aqueous fluids that percolate through the crust, or formed by reaction of preexisting rocks with such fluids; the winners (*2*) recognized that granites crystallized from silicate melts.

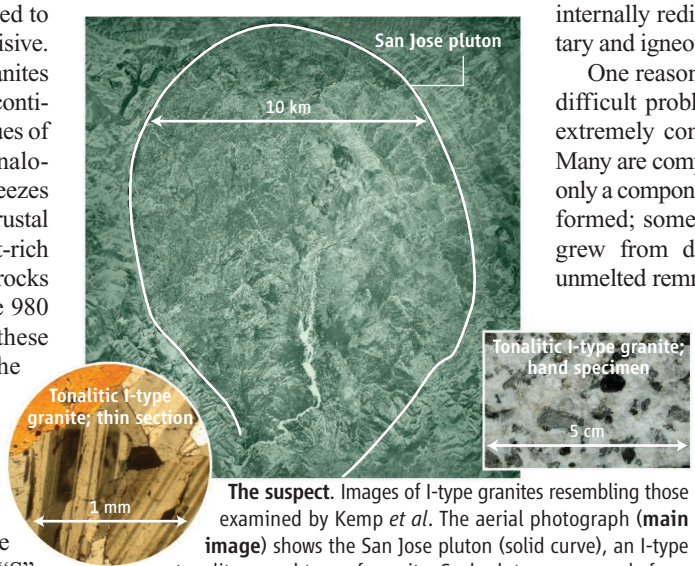
Yet, the resolution of this argument led to various others that remain almost as divisive. Are the silicate melts that give rise to granites partial melts of preexisting rocks in the continental crust, or are they instead the residues of crystallizing mantle-derived basalts, analogous to the brine that is left when ice freezes out of salty water? If granites form by crustal melting, do they come from the sediment-rich upper crust or from preexisting igneous rocks that dominate the lower crust? On page 980 of this issue, Kemp *et al*. (*3*) examine these questions through the lens of two of the newest analytical tools developed for the earth sciences.

Clear answers to the above questions have been found previously for some extreme types of granite. There is little debate that upper-crustal sediments are the sources of S-type granites (*4*) (where "S" stands for sediment) and that mantle-derived basalts give rise to M-type granites (*5*) ("M" for mantle). However, members of a third class—the I-type (*4*)—are abundant, widely distributed, and diverse, and their origins are up for grabs. A popular view holds that these granites are melts of deep-crustal igneous rocks (hence the "I" for igneous) (*4*, *6*). A minority dissenting view suggests that they are instead largely mantle-derived and only modified by passage through the crust (*7*).

The stakes in this argument are high: I-type granites (or their metamorphosed or eroded derivatives) make up a large fraction of the continental crust. Therefore, our thoughts regarding their origins are key to understanding the mechanisms by which the continents differentiate from the rest of the silicate earth, and the consequences of that differentiation for the composition of the mantle. If I-type granites are descended from basalts, then their formation represents net growth of the continents and net removal from the mantle of elements that are highly concentrated in the crust (such as the heat-producing radioactive isotopes, ^{40}K and ^{238}U). If, instead, they form by melting preexisting crustal rocks, they represent a mechanism by which the continents internally redistribute their various sedimentary and igneous constituents.

One reason the origin of granite is such a difficult problem is that these rocks can be extremely complicated (see the figure) (*8*). Many are composed of minerals that represent only a component of the melts from which they formed; some are mixtures of minerals that grew from different melts; some contain unmelted remnants of their sources; and individual minerals often have heterogeneous chemical and isotopic compositions, reflecting the evolution of their parental magmas over the course of their crystallization.

Kemp *et al*. (*3*) examine the origin and evolution of I-type granites from the Lachlan belt in Australia. Their work draws on several recent microanalytical innovations, including high-precision, in situ measurements of oxygen isotope ratios with a large-radius ion microprobe and in situ measurements of hafnium isotopes using laser ablation joined with an inductively coupled plasma mass spec-



The suspect. Images of I-type granites resembling those examined by Kemp *et al*. The aerial photograph (**main image**) shows the San Jose pluton (solid curve), an I-type tonalite, or subtype of granite. Such plutons commonly form kilometer-scale bodies intruded into rocks of the upper crust. Kemp *et al*. suggest that assimilation of enveloping rocks influences the compositions of such bodies. The insets show a specimen of a similar tonalite from the Chihuahua Valley, California. The visible light photograph (**right inset**) reveals dark laths of amphibole and hexagonal crystals of biotite embedded in a white matrix of interlocking feldspar and quartz. The transmitted-light photomicrograph (**left inset**) shows twinning, compositional zoning, overgrowths, and inclusions in plagioclase (complex light and dark pattern), adjacent to a crystal of amphibole (brown). The micro-analytical techniques employed by Kemp *et al*. aim to avoid artifacts that arise from mixing different components of these compositionally and texturally complex rocks.

CREDIT: MAIN IMAGE, J. D. MURRAY AND L. T. SILVER; INSETS, J. M. EILER

The author is in the Division of Geological and Planetary Sciences, California Insititute of Technology, Pasadena, CA 91125, USA. E-mail: eiler@gps.caltech.edu

Supporting Online Material for

Clustering by Passing Messages Between Data Points

Brendan J. Frey* and Delbert Dueck

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

Published 11 January 2007 on *Science* Express
DOI: 10.1126/science.1136800

This PDF file includes:

MATLAB implementation of affinity propagation

Here, we provide a MATLAB implementation of affinity propagation that does not account for sparse similarities matrices. (See http://www.psi.toronto.edu/affinitypropagation for software that efficiently processes sparse similarity matrices).

The only input is the MATLAB matrix of similarities, `S`, where `S(i,k)` is the similarity $s(i,k)$. The preferences should be placed on the diagonal of this matrix; if appropriate preferences are not known, usually, a good choice for the preferences is to set them all equal to $\text{median}_{i,k:i\neq k}s(i,k)$. The following MATLAB code executes 100 iterations of affinity propagation. After execution, the combined evidence $r(i,k)+a(i,k)$ is stored in the $N\times N$ matrix `E`, the number of exemplars is stored in `K`, and the indices of the exemplars for the data points are stored in the N-vector `idx` (point `i` is assigned to the data point with index `idx(i)`.)

```
N=size(S,1); A=zeros(N,N); R=zeros(N,N); % Initialize messages
S=S+1e-12*randn(N,N)*(max(S(:))-min(S(:))); % Remove degeneracies
lam=0.5; % Set damping factor
for iter=1:100
    % Compute responsibilities
    Rold=R;
    AS=A+S; [Y,I]=max(AS,[],2);
    for i=1:N AS(i,I(i))=-realmax; end;
    [Y2,I2]=max(AS,[],2);
    R=S-repmat(Y,[1,N]);
    for i=1:N R(i,I(i))=S(i,I(i))-Y2(i); end;
    R=(1-lam)*R+lam*Rold; % Dampen responsibilities

    % Compute availabilities
    Aold=A;
    Rp=max(R,0); for k=1:N Rp(k,k)=R(k,k); end;
    A=repmat(sum(Rp,1),[N,1])-Rp;
    dA=diag(A); A=min(A,0); for k=1:N A(k,k)=dA(k); end;
    A=(1-lam)*A+lam*Aold; % Dampen availabilities
end;
E=R+A; % Pseudomarginals
I=find(diag(E)>0); K=length(I); % Indices of exemplars
[tmp c]=max(S(:,I),[],2); c(I)=1:K; idx=I(c); % Assignments
```

Mostly, the above code directly follows the updates in the main text. Implemented naively, the updates in the main text would use order N^3 scalar binary operations per iteration. However, if certain computations are re-used, only order N^2 scalar binary operations

are needed, and if only a subset of J similarities are provided, the sparse affinity propagation software available at http://www.psi.toronto.edu/affinitypropagation uses only order j operations.. In the above code, when computing the responsibility sent from i to k, the maximum value of $a(i,k')+s(i,k')$ w.r.t. k' *and* the next-to-maximum value are computed. Then, the maximum value of $a(i,k')+s(i,k')$ w.r.t. $k' \neq k$ needed in equation (1) of the main text can be found in a single operation, by checking to see if k gives the maximum (in which case the next-to-maximum value is used) or not (in which case the maximum value is used). The implementation provided above could be made more efficient and does not take into account sparse data networks, where many input similarities are $-\infty$.

Comparison to exact inference for model selection

For the data shown in Fig. 1C and available at *(S1)*, we ran affinity propagation using a preference (common for all data points) ranging from -200 to -0.1. In this case, the number of points is small enough that we were also able to find the solution that exactly minimizes the energy, for a given input preference. Fig. S1 plots the number of detected exemplars versus the input preference for affinity propagation and the exact method.

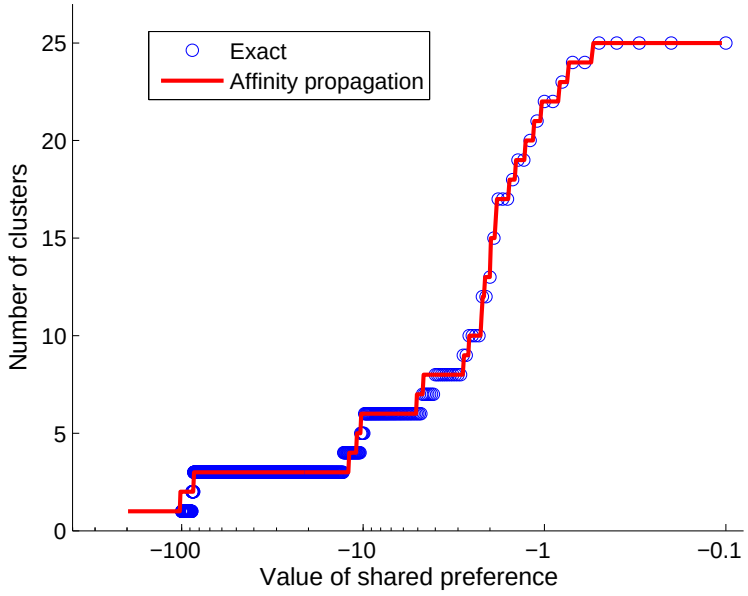


Figure S1: Comparison to exact inference

The number of exemplars detected by affinity propagation is nearly identical to the exact solution.

In a plot of the number of clusters versus the shared preference value, high rates of change correspond to the existence of multiple subsets of data that have approximately the same intra-subset similarities and approximately the same inter-subset similarities. When the preferences are below a certain value, these subsets are grouped together with other clusters. When the preferences rise above that value, it becomes beneficial (in energy) for the multiple subsets to simultaneously form distinct clusters. In a task where the data has a hierarchical similarity structure, different plateaus would correspond to the extraction of different levels of structure.

Detecting genes and comparing against REFSEQ annotations

It was previously shown that when nearby segments of DNA undergo coordinated transcription across multiple tissues, they are likely to come from transcribed regions of the same gene *(S2)*. In that work, $75,066$ DNA segments corresponding to possible transcribed regions of genes were mined from the genome for mouse Chromosome 1. All $75,066$ segments were indexed according to genomic order, and a matrix of similarities was constructed as described in *(S2)*, so that clusters of segments would correspond to predicted genes.

Briefly, the input similarity $s(i,k)$ between DNA segment (data point) i and DNA segment k measures similarity between the expression of the DNA segments across 12 tissues (as measured by a microarray) and proximity of the DNA segments in the genome. To account for non-transcribed regions, an additional artificial data point was included (index $75,067$) and the similarity of each other point to this 'non-transcribed exemplar' was determined using average statistics of the entire data set. The preference for this artificial data point was set to ∞ to ensure it was chosen as an exemplar, while the preference for every other data point was set by comparing its corresponding expression levels to the distribution of expression levels for the entire data set. After clustering the $75,067 \times 75,067$ sparse similarity matrix, DNA segments assigned to exemplars other than the non-transcribed exemplar were considered to be parts of genes. All DNA segments were separately mapped

to the REFSEQ database of annotated genes *(S2)*, to produce labels used to report true positive and false positive rates. These labels, along with the sparse similarity matrix (and sparse affinity propagation implementation), the pre-processed data, and the predictions made by the engineered gene discovery tool reported in *(S2)* are available at *(S1)*.

In addition to using both affinity propagation and over $100,000$ runs of K-centers clustering to detect transcribed regions (as described in the main text), we also used hierarchical agglomerative clustering, or HAC *(S4)*. We used the MATLAB 6.1 implementation of HAC with the 'single linkage' technique. This program could not be applied to the entire $75,067 \times 75,067$ similarity matrix, so the genome was divided into windows containing 600 DNA segments each, in steps of 400. We obtained results using HAC with a variety of linkage functions applied to the same similarity matrix as was used by affinity propagation (data not shown). We found we could obtain better results for HAC using a pair-wise linkage distance equal to one minus the Pearson correlation of the mRNA concentrations for the two query mRNAs (based on 12 conditions). HAC was applied (single linkage worked best) and a threshold was used cut the HAC tree. Points belonging to non-singleton clusters were labeled as being transcribed, and the central 400 labels in each window were kept before moving on to the next window. The threshold was varied to obtain different false detection rates. To prevent distant query mRNAs from being linked together, the linkage distance for training cases i and j was set to infinity if $|i - j| > d$. We let d range from 1 to 10 and found that $d = 3$ gave highest sensitivity. The results for HAC are reported in the main text.

Identifying representative sentences using affinity propagation

For each sentence in the main text of this manuscript, words delimited by spaces were extracted, punctuation was removed, and words with fewer than 5 characters were discarded. The similarity of sentence i to sentence k was set to the negative sum of the information-theoretic costs *(S5)* of encoding every word in sentence i using the words in sentence k and a dictionary of all words in the manuscript. For each word in sentence i, if the word matched a word in sentence k, the coding cost for the word was set to the negative logarithm of the number of words in sentence k (the cost of coding the index of the

matched word), and otherwise it was set to the negative logarithm of the number of words in the manuscript dictionary (the cost of coding the index of the word in the manuscript dictionary). A word was considered to match another word if either word was a substring of the other.

The preference for each sentence as an exemplar was set to the number of words in the sentence times the negative logarithm of the number of words in the manuscript dictionary (the cost of coding all words in the exemplar sentence using the manuscript dictionary), plus a number shared across all sentences that was adjusted to select the number of detected exemplars. This number was set to -90 to detect the four exemplar sentences reported in the main text. For different settings of this number, the following sentences were identified as exemplars.

-100

- Affinity propagation identifies exemplars by recursively sending real-valued messages between pairs of data points.
- 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.
- For different numbers of clusters, the reconstruction errors achieved by affinity propagation and k-centers clustering are compared in Fig. 3B.

-90

- Affinity propagation identifies exemplars by recursively sending real-valued messages between pairs of data points.
- The number of detected exemplars (number of clusters) is influenced by the values of the input preferences, but also emerges from the message-passing procedure.
- 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.

- For different numbers of clusters, the reconstruction errors achieved by affinity propagation and k-centers clustering are compared in Fig. 3B.

-80

- Affinity propagation identifies exemplars by recursively sending real-valued messages between pairs of data points.
- The number of detected exemplars (number of clusters) is influenced by the values of the input preferences, but also emerges from the message-passing procedure.
- 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.
- Fig. 1A shows the dynamics of affinity propagation applied to 25 two-dimensional data points using negative squared error as the similarity.
- At a false positive rate of 3% affinity propagation achieved a true positive rate of 39% whereas the best k-centers clustering result was 17%.

-60

- Affinity propagation identifies exemplars by recursively sending real-valued messages between pairs of data points.
- The number of detected exemplars (number of clusters) is influenced by the values of the input preferences, but also emerges from the message-passing procedure.
- 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.
- Fig. 1A shows the dynamics of affinity propagation applied to 25 two-dimensional data points using negative squared error as the similarity.
- At a false positive rate of 3% affinity propagation achieved a true positive rate of 39% whereas the best k-centers clustering result was 17%.

- In fact it can be applied to problems where the similarities are not symmetric and even to problems where the similarities do not satisfy the triangle inequality.

-50

- The most frequently used technique for learning exemplars is k-centers clustering which starts with an initial set of exemplars usually a randomly selected subset of the data points and iteratively refines this set so as to decrease the sum of squared errors in each iteration.

- Affinity propagation identifies exemplars by recursively sending real-valued messages between pairs of data points.

- The number of detected exemplars (number of clusters) is influenced by the values of the input preferences, but also emerges from the message-passing procedure.

- 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.

- Fig. 1A shows the dynamics of affinity propagation applied to 25 two-dimensional data points using negative squared error as the similarity.

- Instead, the measure of similarity between putative exons was based on a cost function measuring their proximity in the genome and the degree of coordination of their transcription levels across the 12 tissues.

- At a false positive rate of 3% affinity propagation achieved a true positive rate of 39% whereas the best k-centers clustering result was 17%.

- In fact, it can be applied to problems where the similarities are not symmetric and even to problems where the similarities do not satisfy the triangle inequality.

Derivation of affinity propagation as the max-sum algorithm in a factor graph

As described in the main text, identifying exemplars can be viewed as searching over valid configurations of the labels $\mathbf{c} = (c_1, \ldots, c_N)$ so as to minimize the energy $E(\mathbf{c}) =$

$-\sum_{i=1}^{N} s(i, c_i)$. It is easier to think of *maximizing* the net similarity, $\mathcal{S}$, which is the negative energy plus a constraint function that enforces valid configurations:

$$\begin{aligned} \mathcal{S}(\mathbf{c}) &= -E(\mathbf{c}) + \textstyle\sum_{k=1}^{N} \delta_k(\mathbf{c}) \\ &= \textstyle\sum_{i=1}^{N} s(i, c_i) + \sum_{k=1}^{N} \delta_k(\mathbf{c}) \end{aligned} \qquad \text{where } \delta_k(\mathbf{c}) = \begin{cases} -\infty, \text{ if } c_k \neq k \text{ but } \exists i\text{: } c_i = k \\ 0, \text{otherwise} \end{cases} \tag{S1}$$

Here, $\delta_k(\mathbf{c})$ is a penalty term that equals $-\infty$ if some data point i has chosen k as its exemplar (*i.e.*, $c_i = k$), without k having been correctly labeled as an exemplar (*i.e.*, $c_k = k$).

This function (Eq. S1) can be represented using a factor graph *(S6)*. Each term in $\mathcal{S}(\mathbf{c})$ is represented by a 'function node' and each label c_i is represented by a 'variable node'. Edges exist only between function nodes and variable nodes, and a variable node is connected to a function node if and only if its corresponding term depends on the variable. So, the term $s(i, c_i)$ appearing in the above expression has a corresponding function node that is connected to the single variable c_i. The term $\delta_k(\mathbf{c})$ has a corresponding function node that is connected to all variables $c_1, \ldots, c_N$. In a factor graph, the 'global function' — in this case $\mathcal{S}(\mathbf{c})$ — is given by the sum of all the functions represented by function nodes. (Factor graphs are also used to represent a global function that is a product of simpler functions, but here we use the summation form.)

The max-sum algorithm (the log-domain version of the max-product algorithm) can be used to search over configurations of the labels in the factor graph that maximize $\mathcal{S}(\mathbf{c})$. This algorithm is identical to the sum-product algorithm (a.k.a. loopy belief propagation), except that it computes maximums of sums, instead of sums of products. These algorithms are provably exact for trees, but have been used to obtain record-breaking results for highly constrained search problems including error-correcting decoding *(S7,S8)*, random satisfiability *(S9)*, stereo vision *(S10)* and two-dimensional phase-unwrapping *(S11)*. The max-sum algorithm for the factor graph in Fig. S2A can be derived in a straightforward fashion (c.f. *(S6)*) and consists of sending messages from variables to functions and from functions to variables in a recursive fashion.

The message sent from c_i to $\delta_k(\mathbf{c})$ consists of N real numbers — one for each possible value, j, of c_i — and can be denoted $\rho_{i \to k}(j)$ (Fig. S2B). Later, we show that these

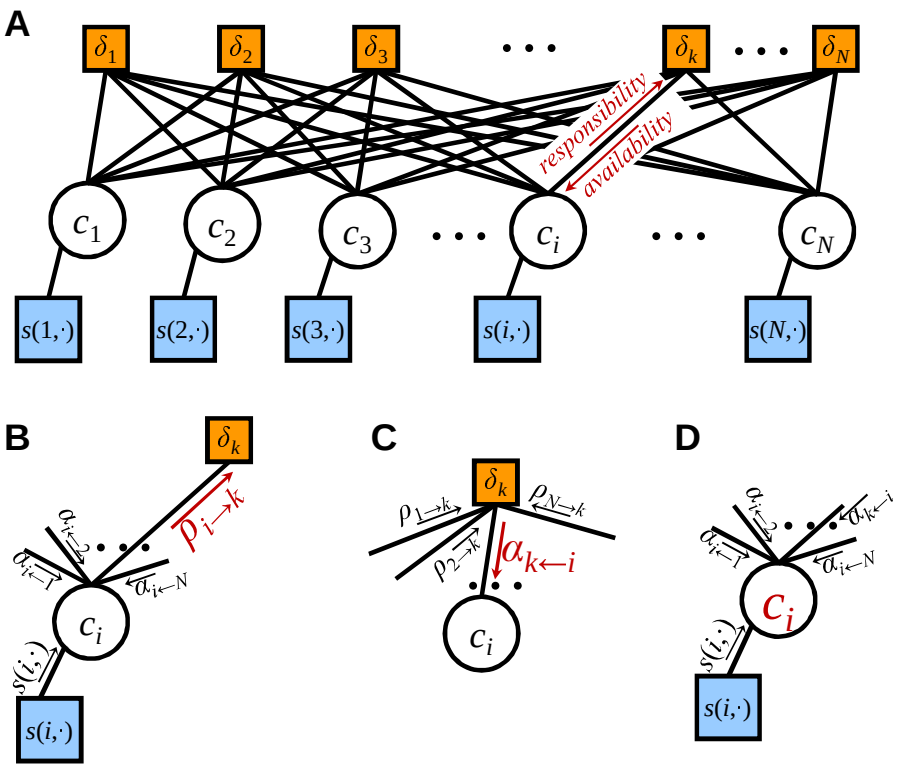


Figure S2: Factor Graph for Affinity Propagation

N numbers can be reduced to a single number, making affinity propagation efficient as a message-passing algorithm. The message sent from $\delta_k(\mathbf{c})$ to c_i also consists of N real numbers and can be denoted $\alpha_{i\leftarrow k}(j)$ (Fig. S2C). At any time, the value of c_i can be estimated by summing together all incoming availability and similarity messages (Fig. S2D).

Since the ρ-messages are outgoing from variables, they are computed as the element-wise sum of all incoming messages:

$$\rho_{i\rightarrow k}\left(c_i\right) = s\left(i, c_i\right) + \sum\nolimits_{k':k'\neq k} \alpha_{i\leftarrow k'}\left(c_i\right) \tag{S2a}$$

Messages sent from functions to variables are computed by summing incoming messages and then maximizing over all variables except the variable the message is being sent to.

The message sent from function δ_k to variable c_i is:

$$\alpha_{i\leftarrow k}(c_i) = \overbrace{\max_{j_1,\ldots,j_{i-1},j_{i+1},\ldots,j_N}\left[\delta_k\left(j_1,\ldots,j_{i-1},c_i,j_{i+1},\ldots,j_N\right)+\sum\nolimits_{i':i'\neq i}\rho_{i'\to k}\left(j_{i'}\right)\right]}^{\text{best possible configuration satisfying } \delta_k \text{ given } c_i}$$

$$= \begin{cases} \overbrace{\sum\nolimits_{i':i'\neq k}\max_{j'}\rho_{i'\to k}(j')}^{\text{best configuration with or without cluster } k}, \text{ for } c_i = k = i \\ \overbrace{\sum\nolimits_{i':i'\neq k}\max_{j':j'\neq k}\rho_{i'\to k}(j')}^{\text{best configuration with no cluster } k}, \text{ for } c_i \neq k = i \\ \overbrace{\rho_{k\to k}\left(k\right)}^{k \text{ is an exemplar}} + \overbrace{\sum\nolimits_{i':i'\notin\{i,k\}}\max_{j'}\rho_{i'\to k}\left(j'\right)}^{\text{best configuration of others}}, \text{ for } c_i = k \neq i \\ \max\left[\overbrace{\max_{j':j'\neq k}\rho_{k\to k}(j')+\sum_{i':i'\notin\{i,k\}}\max_{j':j'\neq k}\rho_{i'\to k}(j')}^{\text{best configuration with no cluster } k}, \overbrace{\rho_{k\to k}(k)+\sum_{i':i'\notin\{i,k\}}\max_{j'}\rho_{i'\to k}(j')}^{\text{best configuration with a cluster } k}\right], \text{ for } c_i \neq k \neq i \end{cases} \tag{S2b}$$

These vector messages are easier to interpret if we view them as the sum of constant and variable (with respect to c_i) components, *i.e.* $\rho_{i\to k}(c_i) = \tilde{\rho}_{i\to k}(c_i) + \bar{\rho}_{i\to k}$ and $\alpha_{i\leftarrow k}(c_i) = \tilde{\alpha}_{i\leftarrow k}(c_i) + \bar{\alpha}_{i\leftarrow k}$. This changes the messages to:

$$\rho_{i\to k}(c_i) = s\left(i,c_i\right) + \sum\nolimits_{k':k'\neq k}\tilde{\alpha}_{i\leftarrow k'}(c_i) + \sum\nolimits_{k':k'\neq k}\bar{\alpha}_{i\leftarrow k'} \tag{S3a}$$

$$\alpha_{i\leftarrow k}(c_i) = \begin{cases} \sum_{i':i'\neq k}\max_{j'}\tilde{\rho}_{i'\to k}(j') + \sum_{i':i'\neq k}\bar{\rho}_{i'\to k}, \text{ for } c_i = k = i \\ \sum_{i':i'\neq k}\max_{j':j'\neq k}\tilde{\rho}_{i'\to k}(j') + \sum_{i':i'\neq k}\bar{\rho}_{i'\to k}, \text{ for } c_i \neq k = i \\ \tilde{\rho}_{k\to k}(k) + \sum_{i':i'\notin\{i,k\}}\max_{j'}\tilde{\rho}_{i'\to k}(j') + \sum_{i':i'\neq i}\bar{\rho}_{i'\to k}, \text{ for } c_i = k \neq i \\ \max\left[\begin{array}{r}\max_{j':j'\neq k}\tilde{\rho}_{k\to k}(j') + \sum_{i':i'\notin\{i,k\}}\max_{j':j'\neq k}\tilde{\rho}_{i'\to k}(j') + \sum_{i':i'\neq i}\bar{\rho}_{i'\to k}, \\ \tilde{\rho}_{k\to k}(k) + \sum_{i':i'\notin\{i,k\}}\max_{j'}\tilde{\rho}_{i'\to k}(j') + \sum_{i':i'\neq i}\bar{\rho}_{i'\to k}\end{array}\right], \text{ for } c_i \neq k \neq i \end{cases} \tag{S3b}$$

For convenience, if we let $\bar{\rho}_{i\to k} = \max_{j:j\neq k}\rho_{i\to k}(j)$ then $\max_{j':j'\neq k}\tilde{\rho}_{i\to k}\left(j'\right) = 0$ and $\max_{j'}\tilde{\rho}_{i\to k}\left(j'\right) = \max\left(0,\tilde{\rho}_{i\to k}(k)\right)$. Also note that in the update for $\alpha_{i\leftarrow k}(c_i)$ (Eq. S3b), none of the expressions on the right depend directly on the value of c_i, rather only the choice

of expression depends on c_i. Consequently, in the N-vector of messages $\alpha_{i\leftarrow k}(c_i)$, there are only two values — one for $c_i = k$ and another for $c_i \neq k$. We set $\bar{\alpha}_{i\leftarrow k} = \alpha_{i\leftarrow k}(c_i\colon c_i \neq k)$ which will make $\tilde{\alpha}_{i\leftarrow k}(c_i)$ zero for all $c_i \neq k$. This also means that $\sum_{k':k'\neq k} \tilde{\alpha}_{i\leftarrow k'}(c_i) = \tilde{\alpha}_{i\leftarrow c_i}(c_i)$ for $c_i \neq k$ and the summation is zero for $c_i = k$, leading to further simplification:

$$\rho_{i\to k}(c_i) = \begin{cases} s(i,k) + \sum_{k':k'\neq k} \bar{\alpha}_{i\leftarrow k'}, \text{ for } c_i = k \\ s(i,c_i) + \tilde{\alpha}_{i\leftarrow c_i}(c_i) + \sum_{k':k'\neq k} \bar{\alpha}_{i\leftarrow k'}, \text{ for } c_i \neq k \end{cases} \tag{S4a}$$

$$\alpha_{i\leftarrow k}(c_i) = \begin{cases} \sum_{i':i'\neq k} \max(0, \tilde{\rho}_{i'\to k}(k)) + \sum_{i':i'\neq k} \bar{\rho}_{i'\to k}, \text{ for } c_i = k = i \\ \sum_{i':i'\neq k} \bar{\rho}_{i'\to k}, \text{ for } c_i \neq k = i \\ \tilde{\rho}_{k\to k}(k) + \sum_{i':i'\notin\{i,k\}} \max(0, \tilde{\rho}_{i'\to k}(k)) + \sum_{i':i'\neq i} \bar{\rho}_{i'\to k}, \text{ for } c_i = k \neq i \\ \max\Big[0, \tilde{\rho}_{k\to k}(k) + \sum_{i':i'\notin\{i,k\}} \max(0, \tilde{\rho}_{i'\to k}(k))\Big] + \sum_{i':i'\neq i} \bar{\rho}_{i'\to k}, \text{ for } c_i \neq k \neq i \end{cases} \tag{S4b}$$

Next, we solve for $\tilde{\rho}_{i\to k}(c_i{=}k) = \rho_{i\to k}(c_i{=}k) - \bar{\rho}_{i\to k}$ and $\tilde{\alpha}_{i\leftarrow k}(c_i{=}k) = \alpha_{i\leftarrow k}(c_i{=}k) - \bar{\alpha}_{i\leftarrow k}$ to obtain simple update equations where the $\bar{\rho}$ and $\bar{\alpha}$ terms cancel:

$$\begin{aligned} \tilde{\rho}_{i\to k}\,(c_i{=}k) &= \rho_{i\to k}(c_i{=}k) - \bar{\rho}_{i\to k} = \rho_{i\to k}(k) - \max_{j:j\neq k} \rho_{i\to k}(j) \\ &= s(i,k) + \sum_{k':k'\neq k} \bar{\alpha}_{i\leftarrow k'} - \max_{j:j\neq k} \Big[s(i,j) + \tilde{\alpha}_{i\leftarrow j}(j) + \sum_{k':k'\neq k} \bar{\alpha}_{i\leftarrow k'} \Big] \end{aligned} \tag{S5a}$$

$$\begin{aligned} \tilde{\alpha}_{i\leftarrow k}(c_i{=}k) &= \alpha_{i\leftarrow k}(c_i{=}k) - \bar{\alpha}_{i\leftarrow k} = \alpha_{i\leftarrow k}(k) - \alpha_{i\leftarrow k}(j\colon j\neq k) \\ &= \begin{cases} \sum_{i':i'\neq k} \max(0, \tilde{\rho}_{i'\to k}(k)) + \sum_{i':i'\neq k} \bar{\rho}_{i'\to k} - \sum_{i':i'\neq k} \bar{\rho}_{i'\to k}, \text{ for } k = i \\ \tilde{\rho}_{k\to k}(k) + \sum_{i':i'\notin\{i,k\}} \max(0, \tilde{\rho}_{i'\to k}(j')) + \sum_{i':i'\neq i} \bar{\rho}_{i'\to k} \\ \quad - \max\Big[0, \tilde{\rho}_{k\to k}(k) + \sum_{i':i'\notin\{i,k\}} \max(0, \tilde{\rho}_{i'\to k}(j'))\Big] - \sum_{i':i'\neq i} \bar{\rho}_{i'\to k}, \text{ for } k \neq i \end{cases} \end{aligned} \tag{S5b}$$

Noting that $\tilde{\rho}_{i\to k}(c_i)$ and $\tilde{\alpha}_{i\leftarrow k}(c_i)$ for $c_i \neq k$ are not used in the updates (particularly because $\tilde{\alpha}_{i\leftarrow k}(c_i \neq k) = 0$), messages can be considered to be scalar with responsibilities

defined as $r(i,k) = \tilde{\rho}_{i\to k}(k)$ and availabilities as $a(i,k) = \tilde{\alpha}_{i\leftarrow k}(k)$.

$$
\begin{aligned}
r(i,k) &= \tilde{\rho}_{i\to k}(c_i{=}k) = s(i,k) - \max_{j:j\neq k}\left[s(i,j) + a(i,j)\right] \\
a(i,k) &= \tilde{\alpha}_{i\leftarrow k}(c_i{=}k) = \begin{cases} \sum_{i':i'\neq k} \max(0, r(i',k)), \text{ for } k{=}i \\ \min\Big[0, r(k,k) + \sum_{i':i'\notin\{i,k\}} \max(0, r(i',k))\Big], \text{ for } k{\neq}i \end{cases}
\end{aligned}
\tag{S6}
$$

This is the form seen in the main text; the $\min[0,\cdot]$ in the availability update comes from the fact that $x - \max(0,x) = \min(0,x)$.

To estimate the value of a variable c_i after any iteration, we sum together all incoming messages to c_i and take the value, $\hat{c}_i$, that maximizes this:

$$
\begin{aligned}
\hat{c}_i &= \underset{j}{\operatorname{argmax}}\left[\textstyle\sum_k \alpha_{i\leftarrow k}(j) + s(i,j)\right] \\
&= \underset{j}{\operatorname{argmax}}\left[\textstyle\sum_k \tilde{\alpha}_{i\leftarrow k}(j) + \sum_k \bar{\alpha}_{i\leftarrow k} + s(i,j)\right] \\
&= \underset{j}{\operatorname{argmax}}\left[a(i,j) + s(i,j)\right]
\end{aligned}
\tag{S7}
$$

An alternative form for this that includes both availabilities and responsibilities can be obtained by including an additional term inside the $\operatorname{argmax}_j[\cdot]$ that leaves the result unchanged as follows:

$$
\begin{aligned}
\hat{c}_i &= \underset{j}{\operatorname{argmax}}\left[a(i,j) + \overbrace{s(i,j) - \max_{j':j'\neq j}\left[s(i,j') + a(i,j')\right]}^{r(i,j) \text{ from responsibility update equation}}\right] \\
&= \underset{j}{\operatorname{argmax}}\left[a(i,j) + r(i,j)\right]
\end{aligned}
\tag{S8}
$$

This is discussed further in the main text where availabilities are added to responsibilities to determine if a data point is an exemplar or not.

The sum-product algorithm on the affinity propagation factor graph

If we use data likelihoods, $S(i,k) = e^{s(i,k)}$, instead of log-domain similarities we can

derive an analogous set of update equations with the sum-product algorithm:

$$R(i,k) = S(i,k) \Big/ \sum_{k':k' \neq k} \left(S(i,k') \cdot \ A(i,k') \right) \tag{S9a}$$

$$A(i,k) = \begin{cases} \left[\prod_{j:j \notin \{i,k\}} \frac{1}{1+R(j,k)} + 1 \right]^{-1}, \text{ for } i \neq k \\ \prod_{j:j \neq k} \left[1 + R(j,k) \right], \text{ for } i = k \end{cases} \tag{S9b}$$

Here, we use $R(i,k) = e^{r(i,k)}$ to refer to an responsibility probability or proportion, and $A(i,k) = e^{a(i,k)}$ for an availability. These update equations are not as straightforward to implement in N^2 time due to numerical precision issues, and the algorithm is no longer invariant to arbitrary scaling of the S-matrix.

An alternative factor graph

The energy and constraint functions can be rephrased in terms of N^2 binary variables rather than N N-ary variables by considering the factor graph topology shown in Fig. S3.

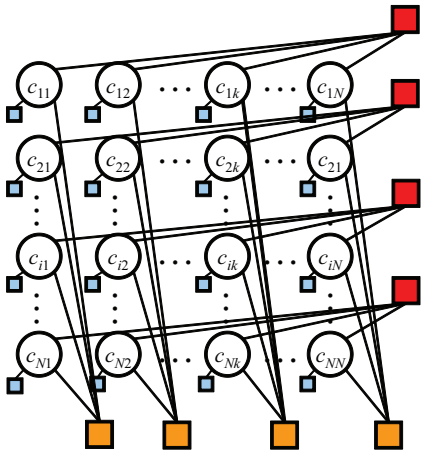


Figure S3: Alternative factor graph topology

Here, binary variable $c_{i,k}$ is one if $c_i = k$ and zero otherwise. There are functions similar to δ_k for each column, constraining that if other data points indicate k is their exemplar, point k needs to be labeled as such. In addition, there are constraints that exactly one variable in each row must be one and all others zero. Message-passing updates can also be

obtained from the max-sum algorithm in this factor graph.

References

S1. Software implementations of affinity propagation, along with the data sets and similarity matrices used to obtain the results described in this manuscript are available at http://www.psi.toronto.edu/ affinitypropagation.

S2. B. J. Frey, *et al.*, *Nature Gen.* **37**, 991 (2005).

S3. K. D. Pruitt, T. Tatusova, D. R. Maglott, *Nucleic Acids Res.* **31**, 34 (2003)

S4. R. R. Sokal, C. D. Michener, *Univ. Kans. Sci. Bull.* **38**, 1409 (1948)

S5. T. M. Cover, J. A. Thomas, *Elements of Information Theory*, (John Wiley & Sons, New York, NY, 1991).

S6. F. R. Kschischang, B. J. Frey, H.-A. Loeliger, *IEEE Trans. Inform. Theory* **47**, 1 (2001).

S7. D. J. C. MacKay, *IEEE Trans. Info. Theory* **45**, 399 (1999).

S8. C. Berrou, A. Glavieux, *IEEE Trans. Comm.* **44**, 1261 (1996).

S9. M. Mézard, G. Parisi, R. Zecchina, *Science* **297**, 812 (2002).

S10. T. Meltzer, C. Yanover, Y. Weiss, *Proc. ICCV*, 428 (2005).

S11. B. Frey, R. Koetter, N. Petrovic, *NIPS* **15**, 737 (2001).