



Response to Comment on "Clustering by Passing Messages Between Data Points"

Brendan J. Frey, *et al.*
Science **319**, 726d (2008);
DOI: 10.1126/science.1151268

The following resources related to this article are available online at www.sciencemag.org (this information is current as of February 11, 2008):

Updated information and services, including high-resolution figures, can be found in the online version of this article at:

<http://www.sciencemag.org/cgi/content/full/319/5864/726d>

A list of selected additional articles on the Science Web sites **related to this article** can be found at:

<http://www.sciencemag.org/cgi/content/full/319/5864/726d#related-content>

This article **cites 8 articles**, 1 of which can be accessed for free:

<http://www.sciencemag.org/cgi/content/full/319/5864/726d#otherarticles>

This article appears in the following **subject collections**:

Technical Comments

http://www.sciencemag.org/cgi/collection/tech_comment

Information about obtaining **reprints** of this article or about obtaining **permission to reproduce this article** in whole or in part can be found at:

<http://www.sciencemag.org/about/permissions.dtl>

Response to Comment on “Clustering by Passing Messages Between Data Points”

Brendan J. Frey* and Delbert Dueck

Affinity propagation (AP) can be viewed as a generalization of the vertex substitution heuristic (VSH), whereby probabilistic exemplar substitutions are performed concurrently. Although results on small data sets (≤ 900 points) demonstrate that VSH is competitive with AP, we found VSH to be prohibitively slow for moderate-to-large problems, whereas AP was much faster and could achieve lower error.

Affinity propagation (AP) is an algorithm that clusters data and identifies exemplar data points that can be used for summarization and subsequent analysis (1). Dozens of clustering algorithms have been invented in the past 40 years, but in (1) we compared AP with three commonly used methods and found that AP could find solutions with lower error and do so much more quickly. Brusco and Köhn (2) compared AP with the best of 20 runs of a randomly initialized vertex substitution heuristic (VSH) described in 1997 (3), which is based on a previously introduced method (4). They found that for some small data sets (≤ 900 data points), VSH achieves lower error than AP in a similar amount of time. We subsequently confirmed those results but found no factual errors in our original report. Interestingly, when we studied larger, more complex data sets, we found that AP can achieve lower error than VSH in a fraction of the amount of

time (5). VSH took ~ 10 days to find 454 clusters in 17,770 Netflix movies, whereas AP took ~ 2 hours and achieved lower error.

As explained in our original report (1), regardless of whether or not the measure of data similarity is symmetric, “[e]xactly minimizing [AP’s cost function] is computationally intractable, because a special case of this minimization problem is the NP-hard k -median problem” (also known as the p -median model, or PMM). [Also see (6).] Consequently, it is expected that different algorithms may work better for different data sets. In fact, we reported results using a brute force method that can exactly solve trivially small problems (e.g., 70 points and six clusters) in a minute or two on any modern computer. We also pointed out that linear programming relaxations (i.e., Lagrangian relaxations) have been used when the data set is not too large (7).

We were curious about how close AP and VSH could get to the best possible (exact)

solution, so we studied the original Olivetti face data set ($n = 400$ data points) for which the exact clustering solution could be found. Because the error for VSH varies depending on the random initialization, in all of our experiments we used the best of 20 runs. Figure 1A plots squared error versus the number of exemplars (k) for AP (single run), VSH (best of 20 runs), k -centers clustering (all of one million runs), and the exact solution (7). Both AP and VSH performed substantially better than k -centers clustering and, for practical purposes, achieved the exact solution (for $k < 135$, the difference in error between AP or VSH and the exact solution is less than the error reduction obtained by including an additional exemplar) (5).

The results presented by Brusco and Köhn (2) demonstrate that for small data sets (≤ 900 data points), VSH often achieves lower error than AP. However, because linear programming was used to find exact solutions for all but one of those data sets, a question that naturally arises is “how do AP and VSH compare for moderate-to-large problems, where exact clustering is not practically feasible?”

To further compare AP and VSH, we studied six additional, diverse data sets: a randomly generated 400×400 nonmetric similarity matrix (8), an extended circuit-board data set consisting of 1272 drill-hole coordinates (9), 1965 video

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

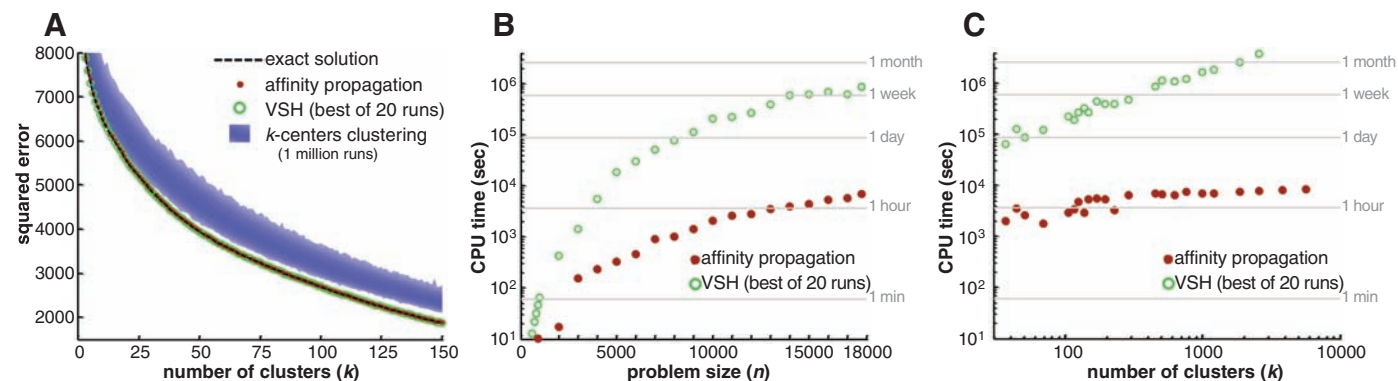
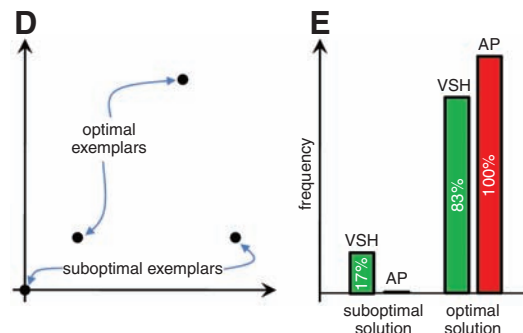


Fig. 1. Performance of AP and VSH on selected data sets. For the Olivetti face image data, (A) plots the squared error achieved by AP, the best of 20 VSH runs, one million k -centers clustering runs, and exact clustering (17). For the Netflix movie rating data, (B) and (C) compare the computational requirements of AP and VSH as a function of the number of data points (n) and the number of clusters or exemplars (k). Because AP updates exemplars in parallel and probabilistically, it can avoid getting stuck in suboptimal solutions. When finding two clusters in a simple data set (D) consisting of points (0,0), (1,1), (3,4), and (4,1), 17% of VSH runs get stuck in the suboptimal solution, whereas AP always finds the correct solution (E).



frames containing faces (10), 10,000 of the 75,066 putative exons from (1), 10,000 of the 22,709 gene expression patterns from (11), and 17,770 Netflix movies (12). We applied AP and VSH to these data sets, and we report error (measured relative to the minimum error, because n was too large to compute the lowest possible error exactly) and central processing unit (CPU) time in Table 1. These results show that although VSH sometimes achieves lower error than AP, the converse is frequently true. These results indicate that AP has a considerable advantage over VSH in terms of speed, especially as the number of data points (n) or exemplars (k) grows. Although using more VSH runs may decrease its error, doing so will require additional CPU time. Conversely, using fewer than 20 runs would likely increase the VSH error; in the one case in which VSH achieved lower error than AP, it did so in only two of 20 runs (if VSH were run once, the AP error would be lower with ~90% probability).

To more closely investigate how the computation times of AP and VSH scale with problem size, we varied n from 1000 to 17,770 for the Netflix movie data (taking nested data sets) while fixing the preference at -1 to obtain moderately sized clusters with 10 to 40 points. Figure 1B shows that AP is much faster than VSH for large problems, where a speedup of roughly 100 is achieved. Because VSH explicitly tracks k exemplars, whereas AP does not, we expected that VSH's CPU time would scale more poorly with k . For all data ($n = 17,770$), Fig. 1C shows that the CPU time required by VSH grows linearly with the number of exemplars (k), whereas the CPU time required by AP is largely independent of k (13).

What is the relationship between AP and VSH and how they reorganize clusters to better account for the data? VSH starts with a randomly selected set of exemplars and then examines every nonexemplar data point in turn and considers replacing a current exemplar with that point. It is well known that VSH can get stuck in a suboptimal solution and that randomly initializing VSH works well only if chances are good that at least one random initialization is close to a good solution (14). As a simple example, if the current exemplars are the "suboptimal exemplars" shown in Fig. 1D, replacing one of these with either of the two other points will not

Table 1. Relative error and CPU time using AP and VSH.

Problem	n	k	Relative error		CPU time VSH:AP
			AP (%)	VSH (%)	
Random S	400	34	0	1.110	0.57:1
Random S*	400	34	0	0.264	3.11:1
Extended circuit board	1,272	103	0	0.176	5.19:1
Face video	1,965	239	0	0.0021	10.84:1
Putative exons	10,000	542	0.0003	0	29.42:1
Gene expression	10,000	566	0	0	86.07:1
Netflix movies	17,770	454	0	0.019	123.09:1

*We allowed VSH to use 100 random initializations instead of 20.

reduce the total squared error. If two initial exemplars are chosen at random, VSH will get stuck in that suboptimal solution 17% of the time (Fig. 1E). Interestingly, AP can be viewed as a generalization of VSH, wherein all points simultaneously compete for replacing every exemplar, but in a probabilistic manner. AP's availabilities (l) represent a "soft exemplar set," and responsibilities provide probabilistic evidence that a candidate exemplar should replace an existing exemplar. Indeed, AP always avoids the poor solution in the above example. If a data set contains many similar data point configurations, chances are high that VSH will get stuck in a suboptimal solution because of exemplars involved in some of those configurations. When we replicated the data from Fig. 1D 100 times with horizontal and vertical shifts drawn uniformly from [-50, +50], for $k = 168$, the best error in 20 VSH runs was 0.951% (relative to optimal), whereas the best error using AP was 0.102%.

Another difference is that AP automatically determines the number of clusters (k) by taking into account exemplar costs (negative preferences), whereas VSH finds a prespecified number of clusters. Automatic model selection is widely considered to be advantageous (15). Many tasks in science, engineering, and medicine require that the solution satisfy a stringency threshold based on a desired false detection rate, a Bayesian prior probability (15), or an information-theoretic coding cost (16). In AP, the exemplar cost exactly accounts for this stringency threshold.

Based on the results we obtained on moderate-to-large problems and AP's ease of implementation and extendability, we believe that AP offers considerable advantages over existing methods.

References and Notes

1. B. J. Frey, D. Dueck, *Science* **315**, 972 (2007).
2. M. J. Brusco, H.-F. Köhn, *Science* **319**, 726 (2008); www.sciencemag.org/cgi/content/full/319/5864/726c.
3. P. Hansen, N. Mladenovic, *Location Sci.* **5**, 207 (1997).
4. M. B. Teitz, P. Bart, *Oper. Res.* **16**, 955 (1968).
5. The data sets and software used to obtain our results, as well as a web application that enables users to upload their own data sets for analysis, are available at www.psi.toronto.edu/affinitypropagation.
6. N. Megiddo, K. Supowit, *SIAM J. Comput.* **13**, 182 (1984).
7. For $n < 500$, we found exact clustering solutions using CPLEX 7.1 to solve a linear programming relaxation of a binary integer program representing the clustering problem (17). All solutions for the $n = 400$ Olivetti data were exact.
8. Entries of the 400×400 similarity matrix were sampled from a beta distribution with parameters $\alpha = 1$, $\beta = 2$.
9. Five problems (III to VII) studied by Brusco and Köhn in (2) consist of physical layout data (drill-hole locations or city coordinates), which have the special property that no two data points are very close together. To test whether this introduced a bias, we replicated the drill-hole coordinates from problem III four times while adding Gaussian noise ($\sigma = 1$), producing a data set with 1272 points.
10. S. Roweis, www.cs.toronto.edu/~roweis/data.html.
11. W. Zhang *et al.*, *J. Biol.* **3**, 21 (2004).
12. Data are available at www.netflixprize.com. Similarities between movies were set to the negative mean-squared-difference between their ratings (scale of 1 to 5) for viewers common to both movies.
13. For the five largest values of k , we were unable to complete the full 20 restarts of VSH (which would likely have taken months of CPU time), so we extrapolated based on no fewer than five completed restarts in each case.
14. S. Eilon, R. Galvao, *Manage. Sci.* **24**, 1763 (1978).
15. D. J. C. MacKay, *Information Theory, Learning and Inference Algorithms* (Cambridge Univ. Press, Cambridge, 2003).
16. J. Rissanen, *Automatica* **14**, 465 (1978).
17. M. Charikar, S. Guha, A. Tardos, D. B. Shmoys, *J. Comput. Syst. Sci.* **65**, 129 (2002).
18. The authors thank U. Frey and I. Givoni for feedback on drafts of this manuscript.

5 November 2007; accepted 15 January 2008
10.1126/science.1151268