Cluster Stability and the Use of Noise in Interpretation of Clustering

George S. Davidson, Brian N. Wylie, Kevin W. Boyack
Sandia National Laboratories*
[gsdavid,bnwylie,kboyack]@sandia.gov

Abstract

A clustering and ordination algorithm suitable for mining extremely large databases, including those produced by microarray expression studies, is described and analyzed for stability. Data from a yeast cell cycle experiment with 6000 genes and 18 experimental measurements per gene are used to test this algorithm under practical conditions. The process of assigning database objects to an X,Y coordinate, ordination, is shown to be stable with respect to random starting conditions, and with respect to minor perturbations in the starting similarity estimates. Careful analysis of the way clusters typically co-locate, versus the occasional large displacements under different starting conditions are shown to be useful in interpreting the data. This extra stability information is lost when only a single cluster is reported, which is currently the accepted practice. However, it is believed that the approaches presented here should become a standard part of best practices in analyzing computer clustering of large data collections.

1 Introduction

We are interested in finding unexpected relationships in extremely large collections of experimental data. *Unfortunately, it is too easy to see illusory patterns*. Our minds are constructed to find patterns, and will do so even when we know that the perceived patterns are no more than random artifacts. The patterns we are interested in finding must, therefore, stand some test that shows they would necessarily reoccur if we started with another, similar dataset, or, perhaps, data randomly perturbed by the addition of slight amounts of noise. We use computers to look through these large datasets, so it is essential that we have confidence that our computational tools are reliable, especially since, in our case, they make use of random numbers. We want to know that the results are insensitive to the particulars of the tool's internal stochastic processes; that is, we want to know that the tool we are using is stable from use to use. Of course, we want these tools to be practical, to be useful to practicing scientists, who want to know that the patterns are *real* and that they potentially point at some physical fact or important process in the world.

Several centuries of practicing the Scientific Method have shown that this last question can only be answered by carefully controlled experiments. Here, we must leave the testing and interpretation of relationships uncovered by our tools to the experimentalists. Our intent is, instead, to address the process of uncovering potentially important patterns, and the investigation of the reliability of the process itself, and the sensitivity of the process to slight variations in the starting data. We will report on various computational investigations into the reliability and sensitivity of one particular data mining tool, VxInsight®[1] used with a medium sized microarray dataset[2].

VxInsight® uses a terrain metaphor[3] to describe large collections of data, summarizing clusters of similar elements by placing them physically close to each other in the terrain. The 2-dimensional clusters are visualized as mountains and hills separated by valleys and open spaces. The heights of the mountains indicate the number of elements clustered together under each mountain. The local groupings and separations between mountains also carry information about the inter-cluster similarities. The data elements in widely separated mountains will have less similarity than those in neighboring mountains.

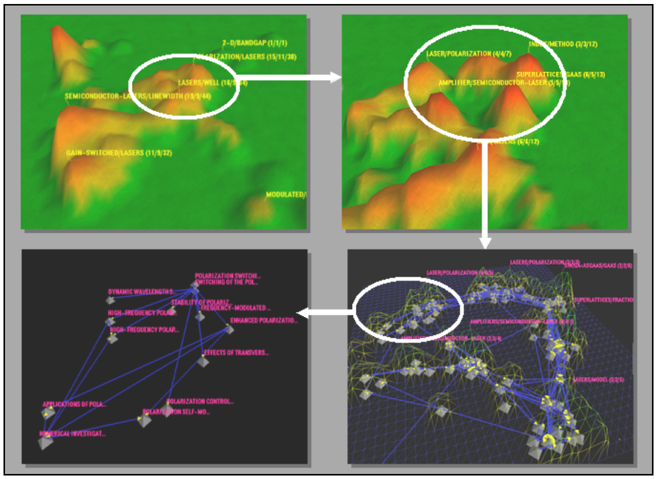


Figure 1: Continuous-level relationships within VxInsight.

Unlike self-organizing maps[4], k-means[5], or York's "Fast Divisive Clustering"[6,7], our approach requires no *a priori* guess at how many clusters should be created. Further, the

*Sandia is a multi-program laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94Al85000.

terrain-like presentation of clusters conveys more information than a technique that merely lists data elements assigned to clusters. The local structure under a mountain reveals finer and finer relationships, which are visible as one zooms into the representation of the terrain (Figure 1). The data objects are not explicitly members in a particular cluster or hierarchy. The positions of the objects are determined through the energy minimization of a connected graph in 2-dimensional space. The assignment of X, Y coordinates to each data element, is a process we call '***ordination***'.

The stability of our ordination process is the main subject of this paper. Briefly, we report on the stability characteristics of the stochastic elements of our force-directed ordination algorithm. These characteristics were studied in a series of experiments that re-ordinated the same dataset with different random starting conditions and compared (both visually and statistically) the results. As described in the Results and Discussion section, the ordination algorithm exhibits predictable, understandable behavior.

Having determined that the tool was acceptably stable, we investigated the impact of adding noise to the similarity relationships, which are the input to the ordination process. As expected, with real data, certain clusters are more robustly stable than others. Importantly, some clusters retain not only the same memberships, but remain physically close to each other in the terrain map. Other clusters make large movements that can be understood by examining the strong similarity linkages extending between clusters in the map.

In the Results and Discussion section, we describe how this analysis suggests important strategies for testing the robustness of clustering algorithms.

2 Methods

2.1 How a VxInsight® map is generated

Figure 2 shows the general process through which data must pass to produce a VxInsight® map. A typical database, represented as a spreadsheet in the figure, would consist of a few thousand elements (the rows), with one or more attributes arranged as tables (the columns). These must be processed to compute similarities for each pair of data elements, which are then used to construct an abstract graph. In this graph of nodes and arcs, the nodes represent individual data elements and the arcs are the similarities between the elements. The ordination process assigns to each data element an X,Y location on the abstract visualization surface. Finally, these coordinates are used to generate the mountain terrains.

2.2 Choosing a data set

For our experiments, we chose a readily available dataset http://genomewww.stanford.edu/cellcycle/data/rawdata, a spreadsheet with about 6000 data elements (the genes in the yeast *S. cereviseae*). We chose a subset of this dataset, 18 measurements of the relative activity of those genes as the cell grows and divides. These data are sufficiently large that they offer opportunities for discovery by data mining techniques, and are well beyond the 'toy' problems often used to test clustering approaches. Further, yeast has been well studied and certain genes are known to work together, and should cluster together as a simple test of our algorithms. Finally, studying this data set allowed the possibility of important predictions about the function of unstudied genes that clustered near those genes with known functions. Importantly, these predictions can be verified by examining the literature published since these data were initially released; much of which is available online, indexed by gene name, see, for example, either [http://www.proteome.com or the Stanford site http://genome-www.stanford.edu/cgi-bin/SGD/search].

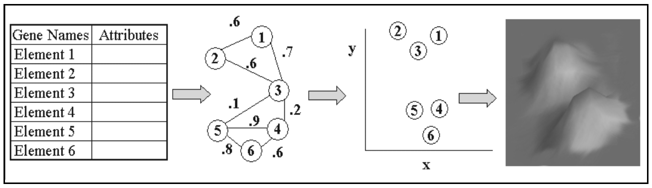


Figure 2: Data processed into a VxInsight map.

2.3 Computing the gene similarities

Each column in the spreadsheet recorded the relative brightness of 6000 spots on a single microscope slide. Various conditions besides the controlled variables will systematically vary these measurements. For example, the overall brightness of one slide may vary due to different amounts of material in the spots, slightly different processing conditions, or differences in scanning the light intensity. To compensate for these effects, the measurements from each slide (a column in the spread sheet) were normalized by subtracting the median value for that slide, and then divided by the inter-quartile range (the difference between the 75th percentile and the 25th percentile brightness value). This robust normalization is less sensitive to outliers than normalization by subtracting the average and dividing by the standard deviation[8]. Pearson's correlation coefficient[9] was used to compute a similarity between each pair of the genes.

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})\left(Y_i - \overline{Y}\right)}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2 \sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \qquad \textbf{(1)}$$

Genes with no similarity will have a value near 0.0, while genes that are strongly similar will have a value near 1.0. Using the raw correlations unduly weights the low similarities and does not adequately represent the information content contained in a strong similarity. The non-linearity of this information, or rareness, is extreme and can change the total range of observed similarity weights by orders of magnitude. We created all of the clusters reported here using gene pair similarities based on the t-statistic of the correlation coefficient, not on the correlation coefficient itself:

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} . \qquad \textbf{(2)}$$

This transformation has logical support, works well in practice, and is easy to compute. We feel it should, at least for microarray experiments, replace the use of straight correlation-based similarities in all clustering analyses.

For this experiment, the twenty strongest positive correlations were recorded for each of the 6000 genes. Finally, for each of the 6000 genes, the gene name, the name of the gene to which it was correlated and the t-statistic of the correlation were written to a file to be used by the ordination program.

It is important to use a large number of similarities to ensure that the fine structure of the ordination is captured. However, we have found that visual inspection of the placement of genes with the strongest similarities provides a valuable tool for evaluating the quality of the ordinations and in understanding their structures. For example, in Figure 3, the strong links suggest that the red cluster can equally well be placed on either side of the ridge (as defined by yellow, pink and light blue).

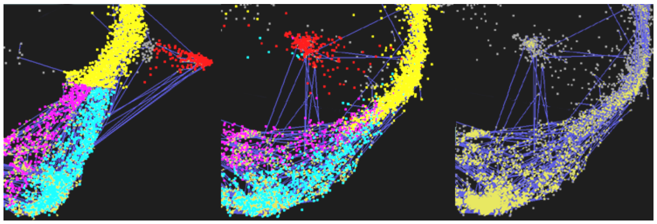


Figure 3: Two random runs (left, middle) show the red cluster switching positions. The strong links (blue lines) suggest either ordination could be acceptable. The third image more clearly shows the high density of strong links within the ridge.

Determining an appropriate critical value for identifying *highly correlated* genes is problematic. The common practice [Ostel, 1963] for reporting the statistical significance of a correlation is to test the hypothesis

H$_o$: The observed n-sample correlation is consistent with observing two processes with a true correlation $\rho = 0$,

using a t-test with $n-2$ degrees of freedom and reject the hypothesis with some level of confidence α. However, with 6000 genes we have 18 million pairs of correlations. Even using a confidence level of $\alpha = 0.001$ we would expect some 36,000 correlations to exceed the critical value by chance alone when the true correlation was 0.0.

To identify the set of highly correlated genes (especially when n is large), a better approach is to do a power analysis, which requires the selection of some assumed actual correlation, ρ_0, and some acceptable chance of not detecting pairs of genes which truly have a correlation of ρ_0 due to variation in the observed values. For instance, we selected $\rho_0 = 0.9$ as the actual correlation and decided that we would not want to miss genes pairs having this sample correlation more than one time in twenty (that is $\beta = 0.05$).

The formula given above for testing H$_o$ is only valid when $\rho_0 = 0$. When, $\rho_0 \neq 0$, an approximation due to Fisher[10] can be used, which transforms r into a normally distributed Z statistic with mean z_{ρ_0} and variance $\sigma^2 = \frac{1}{n-3}$

$$z_r = \frac{1}{2}\ln\frac{1+r}{1-r} . \qquad \textbf{(5)}$$

Hence, the critical value for accepting a pair as being strongly correlated, given our specification that we will mistakenly reject a correlation as being significant one time out of twenty, when the true underlying correlation is $\rho_0 = 0.9$ and when $n = 18$ is:

$$\frac{z_r - \frac{1}{2}\ln\frac{1+0.9}{1-0.9}}{\sqrt{\frac{1}{n-3}}} \geq -1.64 \text{, or when } r \geq 0.78 . \qquad \textbf{(6)}$$

This critical value corresponds to $\alpha < 0.0005$. Pairs of genes matching this specification were saved for later display in VxInsight.

2.4 VxInsight Ordination Routine

The ordination program determines the spatial location for the data objects by considering all of the similarities between objects in the entire set. Figure 4 shows that objects with many similarity links (edges) are clustered together on the map; and objects with little, or no, similarity links are separated.

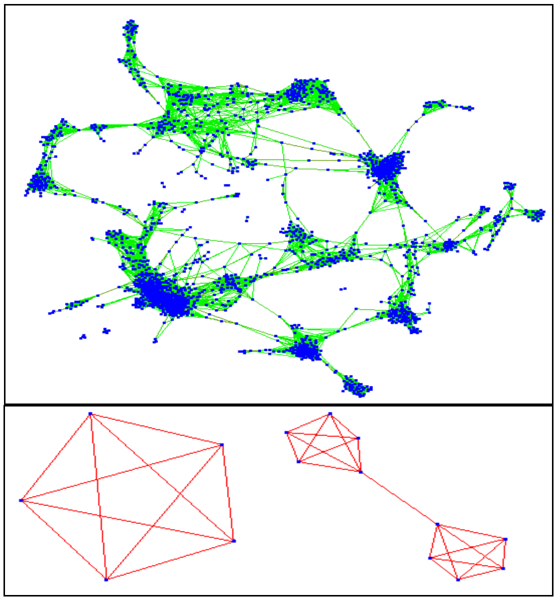


Figure 4: Layout of a 2000 vertex graph (top), and solutions for the well known K$_5$ and Twin K$_5$ (bottom).

An abstract, edge-weighted graph, G = (V, E), is generated using a list of nodes and their similarities, where the vertices, V, correspond to the data objects, and the similarities correspond to the weighted edges, E. An extensive literature exists for graph drawing and layout algorithms[11,12,13,14,15,16,17,18,19]. The work of Fruchterman and Reingold[12] is particularly relevant to our approach.

In developing and implementing our algorithm we were guided by four important principles:

1. Vertices connected by an edge should be drawn near each other.
2. Non-connected vertices should be forced away from each other.
3. The results should be insensitive to random starting conditions.
4. The complexity of computation should be reduced to a minimum.

These principles are so important that we will address each of them in detail.

2.4.1 Principles 1 and 2

Fruchterman *et al.* compute a 'force' term for both attraction and repulsion. These terms are then used to generate new positions for the graph vertices. Our algorithm combines the attraction and repulsion terms into one potential energy equation (Equation 3). The first term, in brackets, is due to the attraction between connected vertices; the second term is a repulsion term.

$$K_{i(x,y)} = \left[\sum_{j=1}^{n_i} \left(w_{i,j} \times l_{i,j}^2 \right) \right] + D_{x,y} \qquad \textbf{(3)}$$

K $_{i\,(x,y)}$ = The energy of a vertex at a specific x, y location
n_i = The number of edges connected to vertex i
$w_{i,j}$ = The edge weight between vertex i and the vertex connected by edge j.
l^2 $_{i,j}$ = The squared distance between vertex i and the vertex at the other end of edge j.
D $_{x,y}$ = A force term proportional to the density of vertices near x,y.

In our ordinations, Equation 3 is gradually minimized in three phases in an iterative fashion. The first phase reduces the free energy in the system by expanding vertices toward the general area where they will ultimately belong. The next phase is similar to the 'quenching' step that occurs in simulated annealing algorithms, the nodes take smaller and smaller random jumps to minimize their energy equations. Last is the simmering phase that makes detailed local corrections.

All movements are random; each vertex is allowed to 'jump' from its current position to a new, random location. If the move reduces the potential energy for the vertex then the vertex is allowed to stay at the new location. Otherwise, the vertex remains where it was until the next iteration. Other, more complicated techniques, including gradient descent and methods with momentum terms, are theoretically appealing. However, the energy 'surface' for thousands of vertices is so chaotic (both spatially and temporally), that, in practice, we have found the simpler method performs better. Notice that for each vertex only its own energy is considered, a characteristic of a 'greedy' algorithm, which only indirectly leads to a global minimization for the entire system. However, the total energy of the system, see Equation 4, can still be used as a criterion for algorithm termination.

$$G = (V, E) : TotalEnergy(G) = \sum_{i=1}^{|V|} K_i \qquad \textbf{(4)}$$

The literature[11,16,17]discusses many other termination criteria, some of which do not explicitly follow the total energy. Eades[11], for example, suggests simply running a fixed number of iterations, in their case 100. We have found that 800 iterations work well for our more complex graphs. We typically deal with graphs having on the order of 10,000 vertices. The graphs discussed in this paper, which have 6000 vertices, require 90 seconds to complete 800 iterations on a 600MHz Pentium III.

Clearly, minimizing the potential energy should lead to ordinations that are consistent with our first two principles. The attraction term rewards movements that minimize the edge lengths between strongly weighted vertices. While the second term, $D_{x,y,}$ which is a force based on the local density of nearby vertices, is minimized when vertices move to less crowded areas. In order to reduce both terms, a vertex must be close to its connected vertices and at a distance from non-connected vertices.

2.4.2 Principle 3

An ordination process can easily get started in ways that prevent smooth transitions to correct answers. That is, the algorithm can get trapped in local minima, and is likely to be forced toward local minima early in the computation. The problem is that an initial configuration can result in some vertices that belong near each other being initially separated by a large barrier. Various stochastic techniques are used to avoid this problem. For instance simulated annealing, which involves the probabilistic decision to take moves that actually increase the energy associated with the node. This technique allows vertices to overcome the barriers associated with local minima, in the effort to find lower energy states. Upon examination of our energy equation it becomes clear that 'barrier jumping' can be achieved by directly solving for the location that minimizes the energy for a single vertex, which can rapidly move a node through an energy barrier. We have successfully used this analytical approach for avoiding local minima early in our algorithm. Achieving a favorable configuration early in the process, independent of the starting configuration, is essential for efficient ordinations that are consistent with our third principle.

We achieve this result by moving vertices in the direction specified by Equation 3 most of the time. However, to jump over energy barriers a small fraction of the vertices ignore the repulsion term and minimize the attraction term analytically. This is accomplished by computing a weighted centriod over all connected vertices. The vertex then 'jumps' to that computed centroid, regardless of any possible energy increase, as shown in Figure 5.

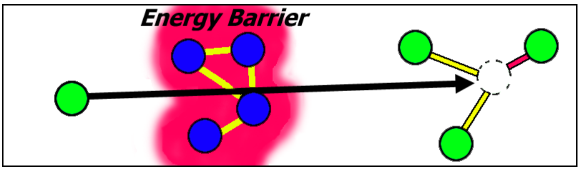


Figure 5: Barrier jumping by ignoring density term.

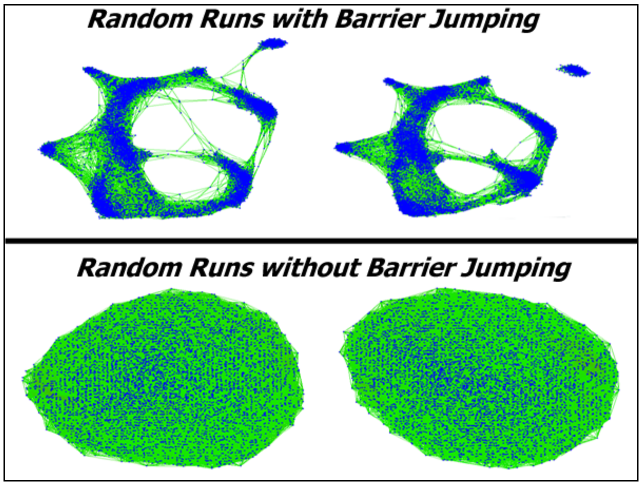


Figure 6: Two random runs with and without barrier jumping.

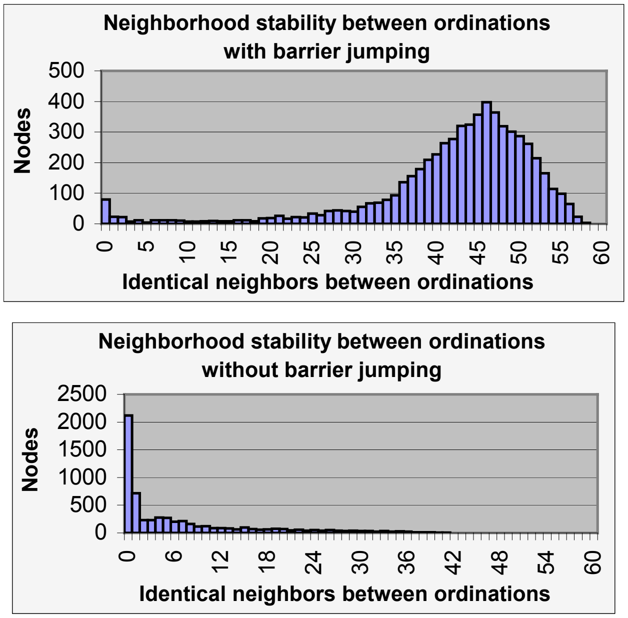


Figure 7: These histograms demonstrate that without barrier jumping local neighborhoods are severely distorted.

Barrier jumping is tied to the cooling schedule, and the frequency of barrier jumping linearly declines from 25% to 10% during the 'quenching' period and is not used at all during the simmer phase. The high frequency at the beginning is required for stability with respect to random initial conditions. The poor initial placement or initial bad jumps that would otherwise irrevocably change the outcome of a purely random algorithm are greatly mitigated by the correcting nature of this process. Figure 6 shows images from two pair of random runs. Ordinations in the first row use barrier jumping, ordinations in the second row do not. We can see the excellent repeatability achieved by using the barrier jump technique. The second row shows that the 6000 vertices become hopelessly trapped in a web of local minima. The histograms in Figure 7 provide further support that barrier jumping improves the repeatability of the random iterative solver. For the histograms in this paper we wanted to measure the stability of the ordination algorithms by counting the number of identical 'neighbors' within a small population of the map (1%). The maps contain 6000 genes so for every gene, we measured how many of the 60 nearest genes remained the same between runs.

2.4.3 Principle 4

The brute force approach for computing $D_{x,y}$ is certainly not consistent with our fourth principle. Because each vertex would have to check its position against all other vertices, this unsophisticated approach would take |V| comparisons for each determination of $D_{x,y}$. As every node must compute $D_{x,y}$ when determining its energy at a specific location x,y, the algorithm would require total running time $\Theta(|V|^2)$.

For real world problems an $\Theta(|V|^2)$ algorithm is prohibitively expensive. We have developed a grid-based method for computing $D_{x,y}$ that allows each vertex to determine an *approximate* value for this term in constant time, $\Theta(1)$, thereby reducing the total running time to a satisfactory $\Theta(|V|)$.

The grid-variant algorithm discussed by Fruchterman[8] uses a binning technique to consider only those vertices within a certain neighborhood. An approach that, with a uniform distribution of the vertices, will reduce the calculation to $\Theta(|V|)$. However, a graph will only have a uniform distribution if the number of edges is small. Highly connected graphs will have dense concentrations of vertices in small areas, and the run time is no longer linear with the number of vertices. To be effective for all graphs, our repulsion term utilizes a 'non-specific' density measure. Vertices are not repulsed by other *specific* vertices, but are repulsed by a general overcrowding. This minor modification to the repulsion criteria allows a dramatic reduction in computational complexity.

This *density field* algorithm is implemented by having each node place an energy footprint onto a two dimensional (density field) array. The energy footprint may be any function in two-space. Our implementation uses a circle with radius r and a function that peaks at the center of the circle, while falling off quadratically with increasing distance from the center of the circle. The total density field is the sum of the contributions of each vertex in the region.

Given the density field, a node can determine an approximate $D_{x,y}$ value using a constant time table lookup method. This method reduces the computation of the repulsion term from $\Theta(|V|^2)$ to $\Theta(|V|)$, and is consistent with our fourth principle, an important result for using our algorithms with real applications.

2.5 The computational experiments

To test the stability of the algorithm to random starting points, *we* ran 100 re-ordinations with different seeds; visually marked the elements of a cluster in one ordination and looked to see if they were visually still clustered together in the other ordinations. We then computed the neighborhood statistics as described below.

To determine if small changes or noise in the similarities would give small changes in the ordination results we ran eighty re-ordinations where we added noise drawn from a gaussian distribution with mean zero, and standard deviations 0.001, 0.010, 0.050, and 0.100, and recomputed the ordinations (these noisy correlations were clipped to remain in the valid range of the correlation coefficient [-1.0, +1.0]. These different ordinations were compared, visually and statistically.

2.6 Evaluation methods

We compared the various ordinations using a neighborhood analysis. When two ordinations are very similar it is reasonable to expect that for every gene, the set of its nearest, say 60, genes would be almost identical in both ordinations. In fact, we would expect the same thing for every gene in the entire ordination. On the other hand, if the ordinations have almost nothing in common, it should be rare to observe a gene that had the same neighbors in both ordinations. We computed these neighborhood statistics for each gene, in each of the two ordinations. For each gene, we first identified the 60 nearest genes, and then counted of the number of genes in both neighborhoods. This number was used to increment the value in a table, so that in the end, we had a histogram showing how many genes had no common neighbors in the two ordinations, how many had one common neighbor, etc., up to the number of genes with exactly the same 60 neighbors in both ordinations, and histograms were prepared, as shown in Figure 8.

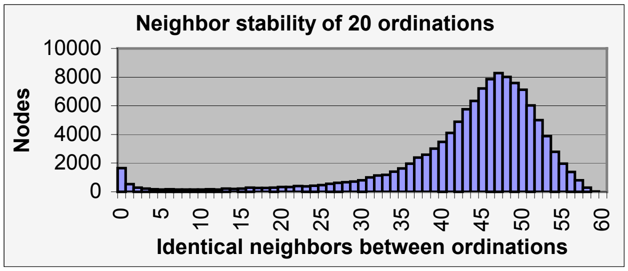


Figure 8: Distribution of neighbors between ordinations with random starting conditions, 20 replicates.

We then visually compared the results of the two ordinations by coloring all of the genes in a cluster found in the first ordination and seeing where those colored genes were placed in the other ordination (so that a similar ordination would not break up the group of colored genes, but would still have them co-located; see Figures 9 and 10).

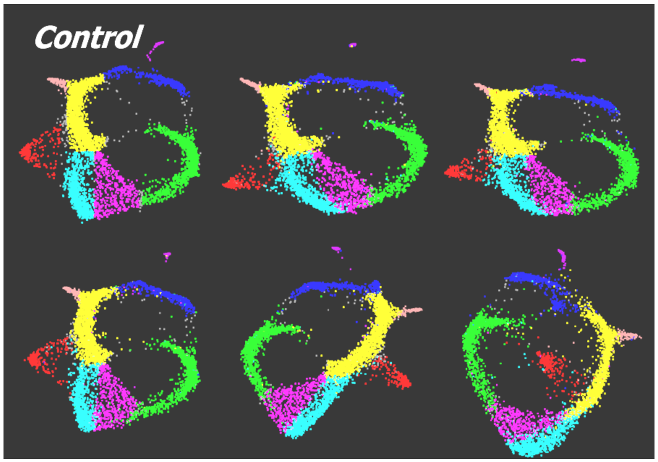


Figure 9: Ordinations with different random starting conditions.

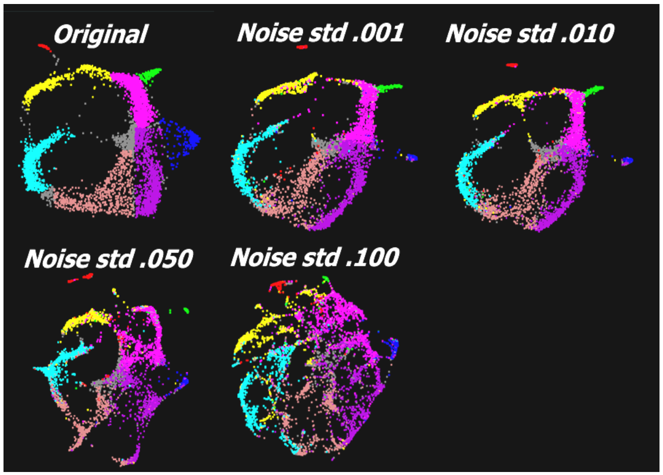
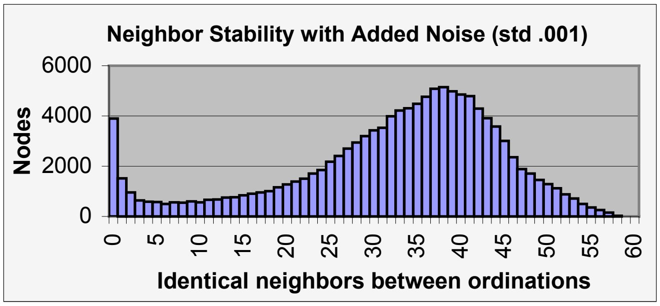


Figure 10: Demonstrates the affect of increasing edge noise on cluster stability.



Figures 11: Histogram of neighborhood stability with added noise (std .001).

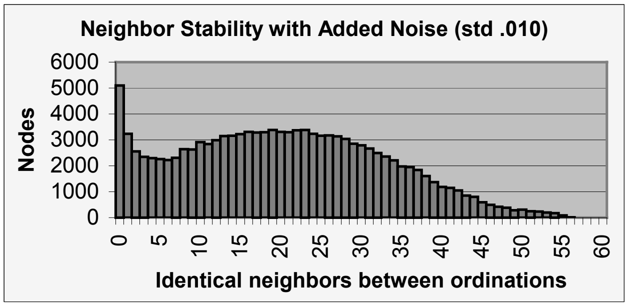


Figure 12: Histogram of neighborhood stability with added noise (std .010).

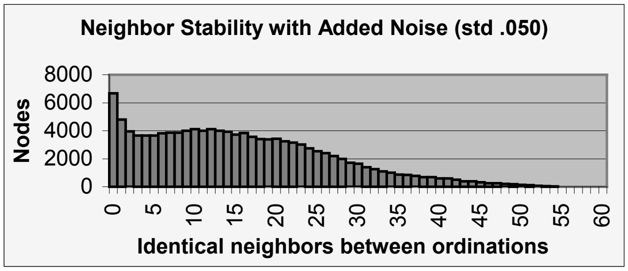


Figure 13: Histogram of neighborhood stability with added noise (std .050).

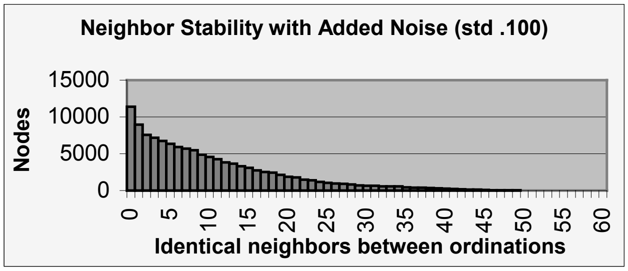


Figure 14: Histogram of neighborhood stability with added noise (std .100).

3 Results and Discussion

The computational experiments revealed two types of information. First, we discovered that large-scale structures were often very robust to starting with different initial conditions. Second, where there were differences, the insights about why the cluster positions changed were as interesting as the fact that they did change. We present two measures of the stability of these structures: a visual interpretation, and the results of our neighborhood analysis. The visual interpretations are striking in their clarity, but are also supported by the numerical results shown in the histograms.

The histogram numbers can be interpreted as data drawn from a binomial distribution. For example, if the two ordinations were totally random, then the neighbors of a gene in the second ordination would be randomly drawn from all the rest of the genes. Given that we had about 6000 genes, and used a neighborhood size of 60, about 1% of the total genes, the probability of exactly k neighbors in the intersection would be

$$\binom{60}{k}\left(\frac{1}{100}\right)^{k}\left(\frac{99}{100}\right)^{60-k} \quad . \qquad \textbf{(7)}$$

When the size of the neighborhood is 1% of the total number of genes the expected frequency for observing 0 neighbors is about 0.547; the expected frequency for observing 1 neighbor is about 0.332; and the frequency for two neighbors is about 0.099, which leaves the expected frequency for observing three or more neighbors in common to be only 0.022. For 6000 genes, only 132 genes would be expected to have more than two neighbors in common between two random ordinations, which is more while several thousand are actually observed. Hence, the histograms and the visual comparisons show that the differences between pairs of our ordinations are very far from being random.

Figure 8 shows six typical ordinations from different starting conditions. Groups in the first ordination were outlined by hand and colored. These same genes were followed in the other ordinations to observe how their relative positions changed. Two striking patterns emerged. In one case the clusters were almost identical to the initial cluster despite different random seeds. In the second case the resulting clusters are a mirror image of the initial clusters. This mirroring is very reasonable, as there is no reason to expect any preferred natural placement as long as the relative distances are preserved, so rotations and reflections should be, and were observed. The histograms showed good neighborhood agreements between mirrored images.

Closer attention to the structures does reveal a few large changes, for example in Figure 8, where we note that the red cluster has flipped from the inside to an outside configuration. This red cluster has a few strong similarity links tying it to the ridge as shown in Figure 3. As a result, it can easily be mirrored with respect to the ridge. Note that the neighborhood analysis would only detect a few differences along the frontiers of the two clusters. As expected, the histograms show very little difference between the two ordinations with respect to the neighborhood analysis. The most encouraging fact is that most groups not only maintain their relative positions given different starting conditions, but that they maintain similar cluster shapes as well, which indicates good interior agreement, which is, again, supported by the histograms. These results indicate that the ordination tool has robust stability when presented with the same dataset. With that information in hand, we began the investigation of how small changes in the similarity data effected the clustering.

Ideally, one would want an ordination algorithm that responded to slight changes in the similarities by producing slight changes in the ordination and that, in some way, moved smoothly from well ordered groupings to totally unordered, high entropy groupings as the similarities are mixed with more and more noise. Figure 10 shows a starting cluster based on the actual similarities, together with four cases where increasing amounts of noise were added to the correlations. Figures 11-14 are the corresponding histograms, reflecting the changes associated with the increasing noise. Note that several large structures remain intact as noise is added, but that some, for example the purple and brown clusters become more disordered. They essentially melt with increasing noise. Also, note the red and green clusters are apparently more resistant to noise. This melting metaphor is particularly appropriate, because it reflects the internal order that must be 'randomized' or melted before the cluster can begin to break apart.

Mixing increasing amounts of noise with the similarities allows one to quickly see which clusters are more likely to be an artifact; these are the clusters that melt out with the smallest amount of noise. This information is so easy to obtain that we believe it should be part of every analysis based on clustering.

4 Conclusion

Understanding, and using stability has been the important theme presented here. In particular, it is important: (1) to make sure the clustering tools are stable with respect to random starting conditions, (2) to ensure that the range for the possible numbers of clusters is adequately covered (either by systematically searching through a large range of choices, or by using a tool that does not require *a priori* determinations of the number of clusters), and (3) to use the clustering tool's response to the gradual addition of noise to gain insight into the actual strength of the clusters.

The two important analysis strategies presented here are: (1) use a probability weighted transformation of the correlation coefficients for the similarities, and (2) compute a small series of clusters with similarities mixed with increasing noise. The first strategy leads to better separation of clusters, and the second gives insight into the strength of individual clusters. We also showed a helpful visual way to track the results of alternate clusters by coloring the genes in a base ordination and following the relative movements of those colored genes in other ordinations. Finally, we suggested a statistical metric based on the intersections of local neighbors of genes under different clusterings.

5 Acknowledgements

We would like to thank Chuck Meyers for starting and for funding the work in this fruitful area; Margaret Werner-Washburne, and Stuart Kim for helping make this a useful tool for microarray analysis; the rest of the development team including Bruce Hendrickson, David Johnson, and Helen Koller. We also wish to thank the reviewers for their useful comments and corrections.

6 References

[1] Davidson, G.S., Hendrickson, B., Johnson, D.K., Meyers, C.E. & Wylie, B.N. "Knowledge mining with VxInsight: discovery through interaction". *Journal of Intelligent Information Systems* 11, 1998, 259-285.

[2] Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D. and Futcher, B., "Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization", *Mol. Biology of the Cell*, 1998, **9**:3273-3297.

[3] Wise, J.A., Thomas, J.J., Pennock, K., Lantrip, D., Pottier, M., Schur, A., & Crow, V. "Visualizing the Non-Visual: Spatial Analysis and Interaction with Information from Text Documents", *Proceedings of InfoVis '95*, IEEE, 1995, 51-58.

[4] Kohonen, T. "Self-organized formation of topologically correct feature maps". *Biological Cybernetics*, 1982, 43:59-69.

[5] MacQueen, J. "Some methods for classification and analysis of multivariate observations". *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I: Statistics*,. University of California Press, Berkeley and Los Angeles, CA, 1967, pages 281-297

[6] York, J., Bohn, S., Pennock, K., & Lantrip, D. "Clustering and Dimensionality Reduction in SPIRE". *Symp. on Advanced Intelligence Processing and Analysis*, (1995), 73

[7] Wise, J.A. "The ecological approach to text visualization". *Journal of the American Society for Information Science* 50(13), (1999), 1224-1233.

[8] Wilcox, R.R., "Introduction to Robust Estimation and Hypothesis Testing", *Academic Press*, 1997, ISBN 0-12-751545-3

[9] Ostel, B., "Statistics In Research Basic Concepts and Techniques for Research Workers", *Iowa State University Press*, Ames, Iowa, USA, 1963

[10] Fisher, R.A., "On the probable error of a coefficient of correlation deduced from a small sample". *Metron*., 1921, 1 (No.4):3.

[11] Eades, P., "A heuristic for graph drawing", *Congressus Numerantium*, **42**, 1984, 149-160

[12] Fruchtermann, T. and Rheingold, E. "Graph drawing by force-directed placement". *Technical Report UIUCDCS-R-90-1609*, Computer Science, Univ. Illinois, Urbana-Champagne, Il., 1990

[13] Quinn, N. and Breur M., "A force directed component placement procedure for printed circuit boards", *IEEE Trans on Circuits and Systems*, 1979, **CAS-26**, (6), 377-388

[14] Otten, R. and van Ginneken, L., "The Annealing Algorithm", *Kluwer Academic Publishers*, Boston MA., 1989

[15] Kamada, T. and Kawai, S., "Automatic display of network structures for human understanding", *Technical Report 88-007*, Department of Information Science, Tokyo University, 1988

[16] Davidson, R. and Harel, D., "Drawing graphs nicely using simulated annealing", *Technical Report CS89-13*, Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehovot, Israel., 1989

[17] Kamada, T. and Kawai, S., "An algorithm for drawing general undirected graphs", *Information Processing Letters*, 1989, **31**, (1), 7-15

[18] Kamada, T. and Kawai, S., "A simple method for computing general position is displaying three-dimensional objects", *Computer Vision, Graphics, and Image Processing*, 1988, **41**, 43-56

[19] Kirkpatrick, S. Gelatt, C.D. and Vecchi, M.P., "Optimization by simulated annealing", *Science*, 1983, 220, (4598), 671-680