Introducing

ArrayMiner™2 by Optimal Design

A New Class of Algorithm for Gene Expression Clustering

"How many clusters are there, really?"
"What about outliers in my data?"

A limitation of current clustering tools

The objective in clustering of gene expression data is to identify clusters of coexpressed genes, such that genes in one cluster relate to a common biological phenomenon, while genes in different clusters relate to different phenomena.
The classic way of doing this is to follow the *proximity principle*: cluster together genes which expression profiles are *closest* to each other, the distance between any two profiles being measured by some distance measure, e.g. Euclidean distance, correlation, or Pearson Coefficient. The proximity principle expresses the intuitive notion that the closer (more similar) two expression profiles are, the likelier it is that they reflect the same biological phenomenon. Most current clustering algorithms, such as hierarchical dendrograms, k-Means, SOMs, VxInsight™, etc. follow the proximity principle.
An obvious consequence of the proximity principle is that each gene is typically supposed to be classified into the cluster to which it is closest in terms of distance to some representative of the cluster (average profile, median profile, etc.). For instance, both k-Means and the most common dendrogram algorithms construct clusters by repeatedly assigning genes to the closest average profile of extant clusters.
Intuitive as it may seem, the proximity principle nevertheless rests on a controversial assumption: when deciding the membership of any given profile, assigning it to the closest cluster ignores the fact that genes in different clusters may well have different spreads (variances) of activity values.
In order to illustrate why the assumption of proximity principle is indeed controversial, consider the following example. The expression levels of a number of genes are measured under two conditions, 1 and 2. The genes belong to two functional groups. Those in the first group are up-regulated in the first condition, but do not show a common tendency in the second condition, their expression levels varying significantly around zero. The genes in the second group are down-regulated in the second condition, but do not show a common tendency in the first, their expression levels varying significantly around zero. Since there are only two conditions, such results are conveniently represented in a two-dimensional drawing, as in Figure 1 below, where the first group is drawn in red circles and the second in blue rectangles.

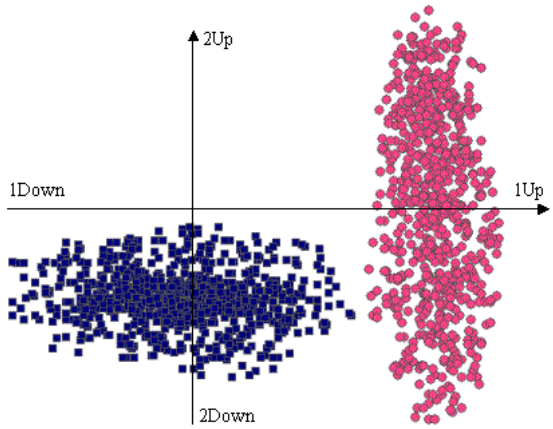

Figure 1: Two groups of genes of different variances

The clusters corresponding to the two functional groups of genes are well separated and should not be difficult to identify. However, the clusters do *not* comply with the proximity principle: Figure 2 reveals that the profiles of a number of genes in the blue cluster are in fact closer to the average (centroid) of the red cluster. This is because the simple criterion of distance disregards the variance (spread) of the clusters.

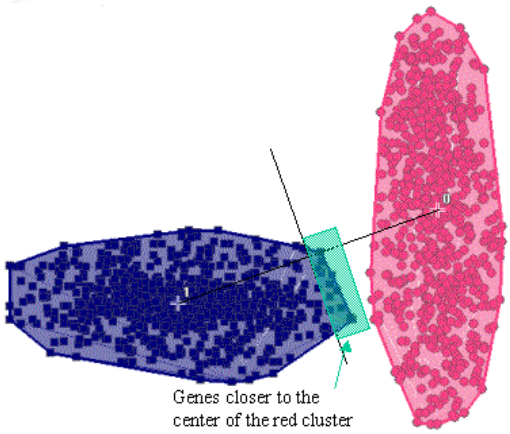

Figure 2: The correct clustering violates the proximity principle

As the example in Figure 2 shows, a clustering algorithm that follows the proximity principle and assigns genes to the closest cluster average, will miss the structure of the two functional groups depicted in the figures. For instance, the best clustering that could be obtained by the k-Means algorithm on these data is shown in Figure 3.

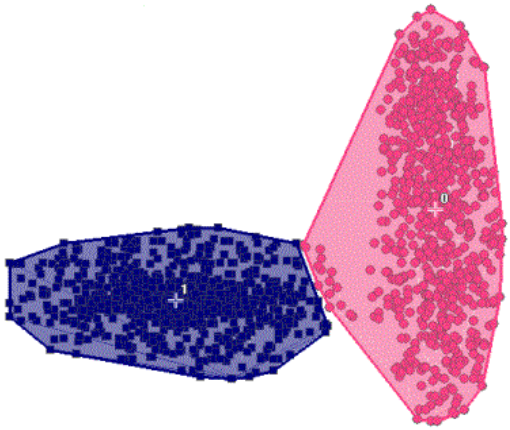

Figure 3: Best k-Means clustering of the above data

The flawed clustering in Figure 3 is due to the fact that the variances of the expression levels in the two functional groups are different, while the k-Means algorithm implicitly supposes that they are equal. In other words, k-Means is an *inadequate method* for clustering such data, and the same holds for many other methods relying on the proximity principle.

The important point is that since the genes in the two groups in Figure 1 belong to different functional groups, their activity is regulated via different pathways, involving different biological phenomena. There is thus no reason why the variances of the activity levels should be equal in the two groups. On the contrary, *most real-world data* can be expected to exhibit different variances of activity levels in different functional groups. Consequently, since clustering by the proximity principle is inadequate in these conditions, one must conclude that most current clustering methods are in fact ill-suited for many real-world data sets.

Accounting for cluster variance

ArrayMiner2 introduces a unique clustering algorithm that takes cluster variance into account. This turns out to be crucial.
The fundamental idea behind ArrayMiner2's approach is to realize that the ultimate aim in gene expression clustering is to identify *distinct functional groups* of genes, each reflecting different biological phenomena that give raise to the observed activity levels. Since they stem from different phenomena, each with its own dynamic, the observed expression level values in each cluster must therefore be expected to follow a distribution different from other clusters. The aim of the clustering thus becomes to identify the distribution for each cluster, such that the resulting statistical model of the data matches the observed data.
Since numerous random factors are involved in gene activation and the measurement process, ArrayMiner2 assumes that the Law of Large Numbers of statistics applies to the distribution of the observed activity levels. The activity level values are therefore modeled by Gaussian distributions, *one for each condition* (experiment) *and each cluster*. So for instance, identifying ten clusters in a data set over four conditions means finding the mean and variance of 40 (10x4) Gaussians that best fit the observed data.

The Algorithm

The standard way of fitting Gaussians to a set of data is to apply the Expectation Maximization (EM) algorithm. Although EM is a well known algorithm, it has two serious drawbacks: its convergence is quite slow, and it is highly sensitive to initialization. While slow convergence could be made for by faster computers, the latter drawback means that EM's results are unreliable, often resulting in clusters that actually do not match the data at hand.
ArrayMiner2 elegantly solves both of EM's drawbacks by exploiting Optimal Design's proprietary technology of Grouping Genetic Algorithm (GGA). The GGA's power, well proven over the last decade, not only allows ArrayMiner2 to identify the best possible clusters with unprecedented reliability, it also enables it to exploit advanced programming "tricks" to significantly speed up the process.

Outliers

Outliers are genes with expression patterns that do not fit any of the clusters – simply put, such genes "do not jibe" with the other genes in the data set. The unique patterns may have any of a number of possible origins, including a truly unique behavior of the genes, sample contamination, laboratory errors, data processing errors, etc. Whatever the origin of the observed unique expression patterns, it is important to identify such outliers, since they may represent *the* interesting finding in the data set, or may signal problems that need to be addressed.
Most current clustering tools are unable to handle outliers adequately. Not only are outliers typically not detected at all, they often perturb the algorithm as it tries to include them into clusters with the other genes at any cost.
ArrayMiner2 detects outliers in the following elegant way. Since by definition an outlier's expression stems from unknown phenomena, it can be really anything. ArrayMiner2 therefore postulates the existence of an additional uniform probability distribution. In other words, ArrayMiner2 models "outlier noise" as an additional *uniform distribution* that "competes" with the Gaussians.
On many data sets, all genes are found to match a set of Gaussians, so ArrayMiner2 reports the corresponding clusters and no outliers. On other data, some expression profiles are found to more probably belong to the uniform "noise" distribution than to any of the Gaussians, and those genes are thus labeled as outliers.

How many clusters?

One of the most frustrating aspects of many current clustering tools is the a priori choice of the number of clusters. The parameter must be supplied for the algorithm to function, but the biologist rarely knows how many clusters there are. Indeed, he or she typically performs the clustering with the aim of discovering the functional groups of genes including their number, so specifying it *before* the functional groups are established is extremely difficult.
The choice of the number of clusters would not be a problem if the structure of the clusters was *robust*. With a robust clustering, increasing the number of clusters would simply reveal more detail in the otherwise stable clusters, enabling the biologist to choose the clustering with the desired level of detail.
Unfortunately, the limited capacity of most current clustering methods to identify expression profile clusters of different variances has a very annoying consequence, routinely observed by users of these methods: the purported structure of the data reported by the method often varies widely with the number of clusters. Instead of refining the

clusters by increasing their number, one often gets completely different clusters. Not only do such results directly contradict the intuitive notion of some kind of *hierarchy* in the underlying physiological phenomena, they also make the choice of the adequate number of clusters all but impossible.

Some clustering tools brush aside this all-important issue by offering "measures of quality" of a clustering, suggesting that the number of clusters that scores best at that criterion is the "right" one. But biologists are usually unwilling to blindly follow such artificial measures, and with good reason: all these measures rest on some evaluation of "information gain" offered by the clustering, but the only real judge of the value of the information are the biologists themselves. Indeed, the question of the "right" number of clusters really boils down to *"how much detail do you need?"* Only the user, not the algorithm, can know the answer to that question.

ArrayMiner2 does ask for the desired number of clusters, because the biologist needs to specify the level of detail he or she requires. However, unlike most other current tools, the clusters ArrayMiner2 supplies are *stable*: their structure is *consistent* across a wide range of the parameter. ArrayMiner2 thus represents the ideal clustering tool: with a low number of clusters, the "big picture" is supplied, and with a high number of clusters, the detailed structure is given. Most importantly, the detailed view is consistent with the "big picture", since the detailed clusters are readily identified as subsets of the clusters in the "big picture". This desirable property is illustrated in the following examples.

In order to illustrate visually the robustness of ArrayMiner2's algorithm and stability of the clusters it supplies, the following two figures show a simple two-dimensional example. The example features three groups of data points, representing three functional groups of genes.

In the left part of, the data has been clustered with the classic k-Means algorithm into two, three and four clusters, respectively. The resulting classifications are shown in the three boxes A, B and C. On the right of the figure, those three clusterings, as well as k-Means clusterings into five through nine clusters, are shown in ArrayMiner'2 Classification Compare facility. That view reveals the correspondences among clusters obtained by clustering into different numbers of clusters.

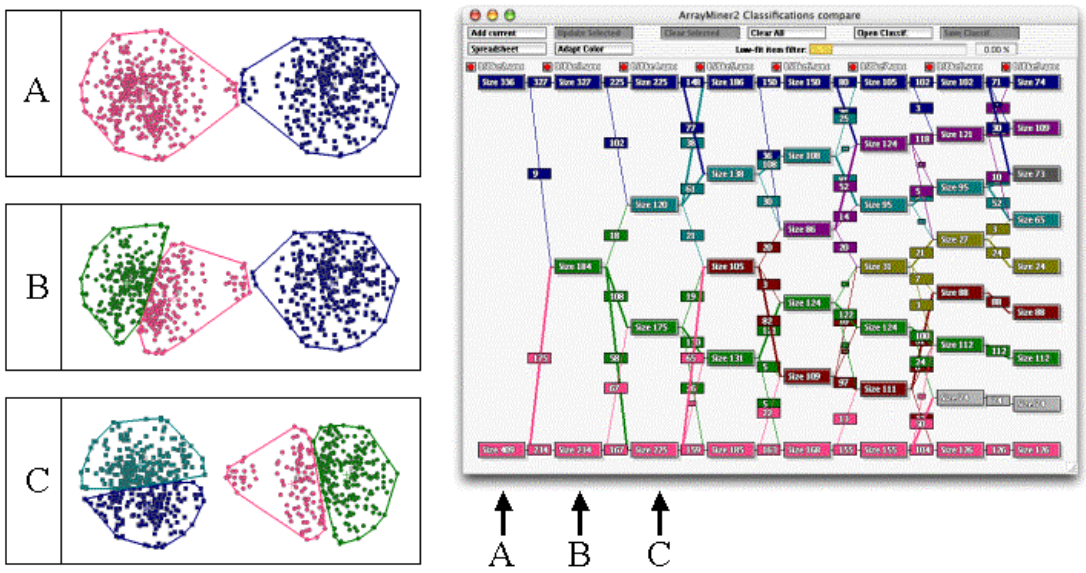

Figure 4: K-Means clustering of a simple data set

As the contents of the three boxes A, B and C in Figure 4 shows, k-Means completely misses the structure of the data. As explained above, this is due to the fact that the three groups have different variances, a property that k-Means is unable to cope with. This failure leads to extremely unstable results – note in particular how the data points in the middle group are first split in two, then clustered together with the points on the left, and then clustered with the points on the right. Furthermore, as the Classification Compare image reveals, this does not get any better when the number of clusters is increased.

The conclusion to be drawn from Figure 4 is that classifications obtained with k-Means can vary wildly with different numbers of clusters. Identifying any stable structure in a series of k-Means clusterings is often hazardous, at best. Worse, the clusters are highly inconsistent across the classifications, which means that the purported functional groups detected by the clustering differ significantly from one classification to another: large numbers of genes are associated in some classifications and dissociated in others. So which one is the "right" classification, given that it is typically not a priori known how many clusters there are? *Few, if any* of them.

ArrayMiner2's results are strikingly different. Since its algorithm takes cluster variance into account, it is substantially better in detecting the true structure of the data. And since the structure remains the same whatever the level of detail, the clusters supplied by ArrayMiner2 are *stable* as well. This is illustrated in Figure 5, where the data from Figure 4 were clustered into two through nine clusters.

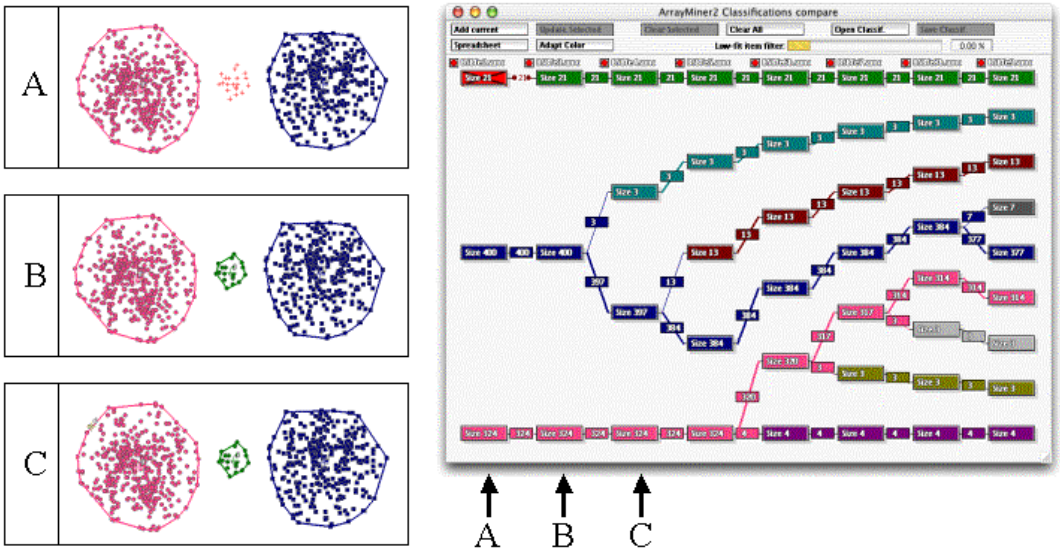

Figure 5: The above data clustered with ArrayMiner2

As in Figure 4, the left part of Figure 5 shows the clusters in the first three classifications of the data, into two, three and four clusters respectively. When clustering into two clusters (box A in Figure 5), ArrayMiner2 detected the two large groups of data points as the most salient feature (the "big picture") in the data. The middle group was correctly identified as *not* being part of either of the two, but since only two clusters were requested, those data points were classified as outliers. When clustering into three clusters (box B), the three groups were correctly identified and no outliers were reported. When clustering into four clusters (box C), a small number of highly similar data points were detected inside the large red group and classified as an extra cluster. These three classifications are also depicted in the right part of Figure 5, together with classifications into four through nine clusters. The diagram confirms the stability of the clusters supplied by ArrayMiner2, as the structure of the data is preserved across all classifications.

Although the successive classifications depicted in the right part of Figure 5 show clusters so stable that one could think that ArrayMiner2 performs a hierarchical clustering (dendrogram), it is important to note that that is not the case: ArrayMiner performs a nonhierarchical clustering, i.e., each of the classifications was computed completely separately from the others. Indeed, one could compute the classification in two clusters and then in nine, without computing the others. So we may ask the same question as above, namely what is the "right" classification among those in Figure 5? *Most* of them in fact: only the level of detail of the view changes. The biologists can at last safely choose the level of detail that will help them discover the interesting clusters of genes.

Real-world data

The examples above illustrate the advantages of ArrayMiner2 in an easy-to-see manner, using two-dimensional examples. Real-world data are in most cases more complex, featuring

a higher number of dimensions (conditions or experiments, or time points). It turns out that with increasing number of dimensions, clustering becomes even more difficult for methods based on the proximity principle. The reason is simple: there are as many variances per cluster as there are dimensions, so a higher number of dimensions means more trouble for methods that disregard cluster variance.

This is illustrated in Figure 6 showing the yeast cell cycle data set "ACGCGT in all ORFs" (507 profiles, 16 time points), available in the demo version of GeneSpring™. For a fair comparison, the data were clustered using k-Means clustering of a leading expression analysis software, into four through eleven clusters. As can easily be seen, no clear structure emerges: as the number of clusters in the k-Means classifications changes from four to eleven, the purported associations among genes change widely as well, and identifying the "right" number of clusters is extremely difficult. Yet choosing any of the classifications invalidates nearly all of the others, because they match each other so poorly. The natural question of whether *any* of the eight k-Means classifications captures the actual structure of the data becomes quite up to the point in face of these results.

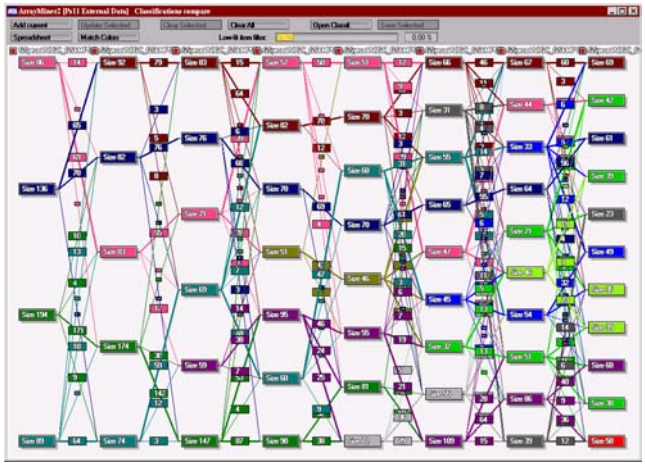

Figure 6: Comparison of k-Means classifications of Yeast data

The same data set was then clustered with ArrayMiner2, using the same distance measure (Pearson Coefficient) as in Figure 6. The resulting classifications are again compared with ArrayMiner's Classification Compare facility, in Figure 7. The high stability of the resulting clusters is clearly visible in the figure.

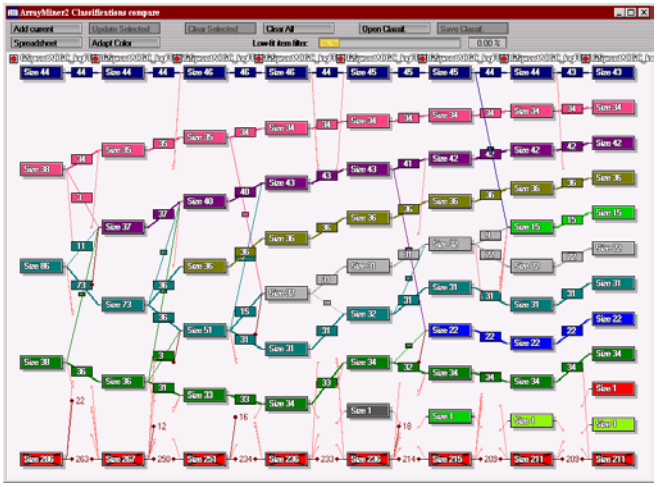

Figure 7: Comparison of ArrayMiner2 classifications of the data in Figure 6

ArrayMiner2 vs. Hierarchical Clustering

The notion of structure that should stay stable whatever the number of clusters is intuitively a strong one. So much so in fact, that "clustering" tools based on that notion are in widespread use. These *hierarchical clustering* tools explicitly enforce the concept of stable structure by constructing a tree of genes (a dendrogram), such that the "clusters" in the upper levels of the tree are reunions of "clusters" in the levels below. The user obtains clusters of genes by selecting disjoint subtrees in the dendrogram (not necessarily all in the same level of the tree).

While such a bottom-up construction does lead to "clusters" that are stable by definition of the dendrogram, the approach has a serious drawback. Since the dendrogram is constructed from bottom up, the whole tree is strongly contingent on associations of genes in the lowest levels. Unfortunately, the "clusters" in the lowest levels are those that are constructed with very little if any regard to the *global* structure of the data. In particular, the variance of functional groups in the data cannot be taken into account at that stage of the construction of the dendrogram. As a result, the upper levels of the dendrogram often miss the true structure of the data.

Although ArrayMiner2's clusters show a near tree-like stability, it is not a result of a hierarchical construction. Instead of recursively uniting "clusters" within a dendrogram, ArrayMiner2 performs a rigorous *non-hierarchical* clustering, with the level of detail depending on the requested number of clusters. The fact that tree-like comparisons of the resulting clusters can be subsequently obtained, as in Figure 5 and Figure 7, is simply due to the fact that a stable structure in the data is correctly identified regardless of the requested number of clusters.

Conclusions

ArrayMiner2 constitutes a significant advance in gene expression clustering. Its capacity to identify the structure of many gene expression data sets significantly better than methods based on the proximity principle yields a *non-hierarchical* clustering method that supplies remarkably stable clusters when the requested number of clusters changes. As a result, the nearly-intractable problem of the "right" number of clusters all but disappears: ArrayMiner2's clusters are highly reliable whatever their number in a classification.

Furthermore, ArrayMiner2 solves the problem of outliers that perturbs most other clustering methods. Outliers are detected as not belonging to any cluster, allowing the biologists to spot these unique patterns, leading them to discovery of previously unsuspected phenomena or data collection problems.

Completed by a unique graphical user interface featuring clusters shown as 3D translucent solids viewable from any angle by a simple mouse drag, the innovative Classification Compare facility for visually easy comparison of various classifications, and many other intuitive viewing tools, ArrayMiner2 offers an unequaled power and user experience in gene expression analysis.

For availability and/or to request a fully functional demo of ArrayMiner, visit our website at http://www.optimaldesign.com.