# CONNECTIVITY CLUSTERING: A NOVEL ALGORITHM APPLIED TO THE CLUSTERING OF GENE EXPRESSION PATTERNS IN *DICTYOSTELIUM* DEVELOPMENT

R. ŠÁŠIK[†], T. HWA[†], N. IRANFAR[*], W. F. LOOMIS[*]
*Department of Physics[†] and Division of Biology[*]*
*University of California at San Diego, La Jolla, CA 92093, USA*

We present a novel approach to the clustering of gene expression patterns based on the *mutual connectivities* of the patterns. Unlike certain widely used methods (e.g., self-organizing maps and K-means) which essentially *force* gene expression data into a fixed number of predetermined clustering structures, our approach aims to *reveal* the natural tendency of the data to cluster, in analogy to the physical phenomenon of *percolation*. The approach is probabilistic in nature, and as such accommodates the possibility that one gene participates in multiple clusters. The result is cast in terms of the connectivity of each gene to a certain number of (significant) clusters. A computationally efficient algorithm is developed to implement our approach. Performance of the method is illustrated by clustering both constructed data and gene expression data obtained from *Dictyostelium* development.

## 1 Introduction

With the advent of microarray cDNA technology[1,2] and genome sequencing projects, progress in molecular biology will depend more and more on the ability to efficiently process and quantitatively analyze large amounts of data. For instance, in a microarray experiment in which one looks for groups (or clusters) of similarly expressed genes out of a vast number expression patterns containing $10^3$ to $10^5$ data points, visual inspection is inadequate and automated algorithms must be used to reach beyond the most conspicuous clusters. A number of clustering algorithms have been introduced in the past few years, and they have been instrumental in discovering new features contained in large-scale gene expression data[3,4,5].

Despite the versatility and early success of these methods, however, they are limited in scope due to a number of fundamental shortcomings, the most notable one being the dependence of the results on certain *arbitrarily imposed* clustering structures. Such dependences can give rise to misleading results when the assumed clustering structures are far from the truth. For example, the hierarchical method of clustering imposes a rigid hierarchical structure to the clusters which gene expression data may not possess. Moreover, the method is "greedy" (e.g., allows no backtracking), so that small errors in cluster assignment in early stages of the algorithm can be drastically amplified.

The more sophisticated K-means clustering assumes that there are $K$ clus-

ters in the data, and finds the cluster origins and the membership of each cluster by minimizing the cluster variances. The major disadvantage of this method is that the number $K$ is not known in advance and a lot of guesswork is required on the part of the analyst while finding the "best" $K$, usually by judging the tightness of clusters. The problem, of course, is that the larger the $K$ one chooses, the tighter the clusters become. Another potential flaw of this method is that because *each* gene is uniquely assigned to some cluster, it is difficult for the method to accomodate a large number of stray data points, intermediates, or outliers. The latter can in principle be handled by allowing different clusters to have different variances; however optimization over such a large parameter space quickly becomes intractable computationally. Another popular method is the self-organizing maps, which as implemented by Tamayo *et. al.*, is essentially a restricted version of $K$-means: Here the $K$ clusters are linked by some arbitrary user-imposed topological constraints (e.g., a $3 \times 2$ grid), and as such suffers from all of the problems mentioned above for $K$-means (and more), except that the constraints expedites the optimization process.

In this paper, we present a different approach to clustering based on the "mutual connectivity" of the expression profiles. Our method does not assume the structure nor the number of clusters. It is probabilistic in nature and allows each gene to belong to multiple clusters. The algorithm is not greedy but more Monte-Carlo like. In the subsequent sections, we first present a conceptual overview of the approach, followed by a detailed description of the algorithmic implementation of the approach. Then we illustrate the performance of our algorithm by contrasting it with that of the SOM on a constructed test dataset for which the clustering structures are known. Finally, we apply our algorithm to the gene expression data obtained from *Dictyostelium* development, and describe new findings which are of biological interest.

## 2 Connectivity Clustering

### *2.1 Overview:*

Our approach is inspired from the concept of percolation, a well-studied paradigm used by statistical physicists to characterize a diverse range of physical phenomena, from spreading of forest fires to drainage of fluid in porous rocks. The basic idea is very simple: a probe is used to *reveal* the mutual connectivity among a large number of points, with the highly-connected regions identified as clusters. Thus, setting fire to a forest reveals which trees are clustered together, and pushing fluids through porous rocks reveals which pores are connected.

Applying this idea to gene expression analysis, we represent each expres-

sion pattern containing $m$ measurements by a point in an $m-$dimensional space. A distance measure $d_{ij}$ is defined between all pairs of points $i$ and $j$ such that small distances correspond to similar expressions, and large distances to dissimilar expressions. In connectivity clustering one takes advantage of the intuitively obvious fact that clusters have a higher density of points than the surrounding background. This is realized by connecting every pair of points that are within a certain threshold distance, say $d_0$, from each other; each connected graph which results can be regarded as a cluster. For $d_0 = 0$, of course every point is a cluster by itself. Now if $d_0$ is gradually increased from zero, then the points from the regions of highest density would get interconnected first to form tight clusters; next, the more dilute clusters will form. Later as connections are made between existing clusters, they merge into even larger clusters, until eventually at some large $d_0$ (typically much smaller than the maximum of $d_{ij}$'s), all points will be interconnected.

So far, we have described a deterministic procedure which in essence is not so different from hierarchical clustering. As already mentioned above, such procedures are susceptible to errors, say in individual pairwise distances. (We know in fact that array data suffer from a good deal of noise.[6]) For instance, one small $d_{ij}$ can prematurely connect two clusters that are otherwise separated far away. To overcome this problem, we adopt a probabilistic procedure, connecting two points $(i, j)$ with probability $P(d_{ij}) = \exp(-d_{ij}^2/d_0^2)$, which varies continuously between 0 and 1. In this way, two points $(i, j)$ are directly connected only if their distance $d_{ij} \ll d_0$. Otherwise, a group of points need to be *multiply connected* to each other in order for this group to remain as a cluster with high probability. Note that the function $P(d_{ij})$ defines a *statistical ensemble*. In order to properly sample the ensemble, one must generate many of its realizations (in practice a hundred seems sufficient). Since two points may belong to the same cluster in some realizations and to different clusters in other realizations, membership of points in each cluster also becomes probabilistic for a given value of $d_0$.

The probabilistic procedure described above can be cast mathematically in the language of percolation[7], or equivalently, as the finite-temperature $q-$state Potts model[8] with $q = 1$. Within the latter framework our work is related to that of Blatt *et al.*[9], which is an application of the $q-$state Potts model with $q \geq 2$. An important feature of percolating system of this kind is that there exist a *percolation transition*: When $d_0$ reaches a critical value $d_c$ (whose precise value can be computed from $P$), a random collection of points becomes globally connected. What is important for the task at hand is the known property that for $d_0 < d_c$, the typical cluster size $S_0$ (e.g., the number of points contained in each cluster) becomes small, with the probability of finding a cluster size

$S \gg S_0$ decaying exponentially. Thus, if one operates below the percolation threshold, the probability of finding large clusters by chance is small, making such findings statistically significant. Despite the existence of this general theoretical framework, the precise statistical evaluation of the clusters require detailed analysis and will be presented elsewhere. In what follows, we will describe the algorithm and illustrate the results.

### *2.2 Algorithm:*

In practice, the connectivity clustering algorithm proceeds in three steps.

**Step One: Global Analysis**

• Generate random numbers $0 < \eta_{ij} < 1$ from a uniform distribution for all pairs of points $(i, j)$.

• Calculate the threshold $d_{ij}^0$ as the value of $d_0$ for which $P(d_0) = \eta_{ij}$.

• Increase $d_0$ continuously from 0. Everytime $d_0$ exceeds a threshold $d_{ij}^0$, a connection is made between the points $i$ and $j$. If two previously disconnected clusters get connected, we assign the *tree distance* $D_{m,n}$ between every point $m$ in the first cluster and every point $n$ in the second cluster to be $d_0$. The process stops when all points are connected.

Step One is repeated many times with different seeds for the random number generator (in practice 100 times is sufficient) in order to calculate the *average tree distance* $\overline{D}_{ij}$ between all pairs of points. Note that $\overline{D}_{ij}$ will be different from the direct pairwise distance $d_{ij}$ we started from. Pairs of points from dense regions of space will all have nearly the same average tree distances, which will generally be smaller than the corresponding direct distances.

Next, we construct the *average* tree by repeating the last bullet above, except with $d_{ij}^0$ replaced by $\overline{D}_{ij}$.

**Step Two: Construction of the Average Tree**

• Set $d_0 = 0$. Every point is a cluster of size 1 and also a *cluster origin.*

• Increase $d_0$ continuously from zero. Every time $d_0$ exceeds a threshold $\overline{D}_{ij}$, a connection is made between points $i$ and $j$. If this connection links two previous disconnected clusters, then the new cluster results as the union of the two. The size of the new cluster is the sum of the two previous clusters, and the origin of the new cluster is taken to be the origin of the larger of the previous two. The process stops when all $N$ points are connected.

At the end of this step, we have a list of cluster origins, and a record of which points (and at which value of $d_0$) belong to each origin.

This list defines an "average tree", and provides a global view of the mutual relationship among the different data points. It can be regarded qualitatively

as a more reliable version of the kind of trees produced by the hierarchical clustering algorithm. This result is used to filter out the uninteresting clusters (say, those whose sizes are below certain limit set by users). In the next step, one then investigates each of the remaining clusters closely using the full power of this approach. Specifically, we use the concept of *invasion percolation* to determine probable connectivity of all points to each of the cluster origins:

**Step Three: Calculation of Connectivity to Cluster Origins**

- Choose a cluster origin, say, $X$, as the point of "invasion".
- Generate random numbers $0 < \eta_{ij} < 1$ from a uniform distribution for all pairs of points $(i, j)$.
- Calculate the threshold $d^0_{ij}$ for which $P(d^0_{ij}) = \eta_{ij}$ for each pair $(i, j)$.
- Set $d_0 = 0$. Point $X$ is the only member of the invading cluster.
- Increase $d_0$ continuously. Whenever there is a pair of points $(i, j)$ such that one point belongs to the invading cluster and the other point does not, *and* if $d^0_{ij} \leq d_0$, then make a connection between these two points (thereby enlarging the invading cluster size by 1). The process stops when an infinitesimal increase of $d_0$ would cause any cluster origin (other than $X$) to become part of the invading cluster.
- Record membership of the invading cluster.

Step Three is again repeated many times (in practice 100 times is again sufficient) with different seeds for the random number generator, for each cluster origin the user decides to pursue at the end of step two. At the end of this step, one obtains a list containing "connectivity", i.e., the probablility that each point is connected to a cluster origin. This list provides the complete description of a probablistic cluster containing the origin X. Statistical significance (e.g. the $p$-value) of finding such a cluster by chance can be provided once a null model is specified; a detailed discussion of statistical significance will be provided elsewhere. For now, we will examine the resulting clusters obtained from the above algorithm.

### *2.3 Result and Comparison:*

The principles of connectivity clustering and comparison of its performance with another common clustering algorithm, the self-organizing map (SOM), are best illustrated in a graphical form (Figs. 1a–1d). Figure 1a shows the constructed test data set in two spatial dimensions, which represents a putative array experiment on $N = 325$ genes, with just two measurements taken (we are limited to two dimensions for presentation purposes). In order to simulate the experimental situation, the set is created as follows: 50 points (red) are drawn from a spherically symmetric Gaussian cluster, 40 points (green) are

drawn from an elliptically symmetric Gaussian cluster, 10 points (blue) are drawn from another spherically symmetric Gaussian cluster of smaller width; superimposed on these "data" points are 225 "noise" points (black) which are uniformly distributed inside the box. Note that some of the "noise" points overlap with the "data", so these should naturally cluster together (remember that real-life experimental data do not come color coded), and some "data" points are so far removed from their respective cluster centers that they should cluster with differently-colored "data". Of importance are noise points that one might wish to leave unclustered based on a purely visual inspection.

We executed the connectivity clustering algorithm of Sec. 2.2 on this test dataset. The complete run (including 100 realizations of ensemble average) took 22 sec of CPU time on a 500 MHz Pentium computer. [Note that the algorithm scales quadratically with the number of data points but is essentially independent of the number of components of each point.] The algorihtm clearly identifies the three "data" clusters we generated (Fig. 1b–1d). In addition to these, however, there was also a cluster found (Fig. 1e), for the particular value of minimal cluster size we asked the program to report. (Of course, many smaller clusters exist had we asked for them.) Clearly, the cluster in Fig. 1e results from random agglomeration of Poissonian "noise" and contains no information. We can distinguish the true and false clusters by characterizing the statistics of obtaining clusters such as that in Fig. 1e; this important issue along with the choice of null models will be reported elsewhere. Here, we proceed with the comparison of the clusters obtained with those of SOM.

When the constructed data are processed with the SOM algorithm as implemented by Tamayo *et al.*, the results vary greatly depending on what structure of centroids one chooses. For instance, with a $3 \times 1$ structure (Fig. 1f) which corresponds exactly to the actual number of "data" clusters, the result is quite discouraging, as the two original (red and green) clusters are assigned together with many "noise" points to a single cluster, the red one in Fig. 1f; the other two clusters being made up essentially of "noise". Obviously this particular SOM algorithm does not deal well with a situation in which the clusters are close together in a large background of non-informative noise. Unfortunately the real expression data we have share the same character as we will see below. When a $2 \times 2$ structure is used instead (Fig. 1g), the result is better. Nevertheless, comparison with Figs. 1b-1e shows that much of the "noise" left out by our algorithm gets clustered along with the "data" by this SOM. This illustrates a generic weakness of the K-means-type approach to clustering. Of course the "data" can be better separated from the outlying "noise" by choosing a higher number of centroids, but then we are left with a large number of clusters, most of which carry no information whatsoever.

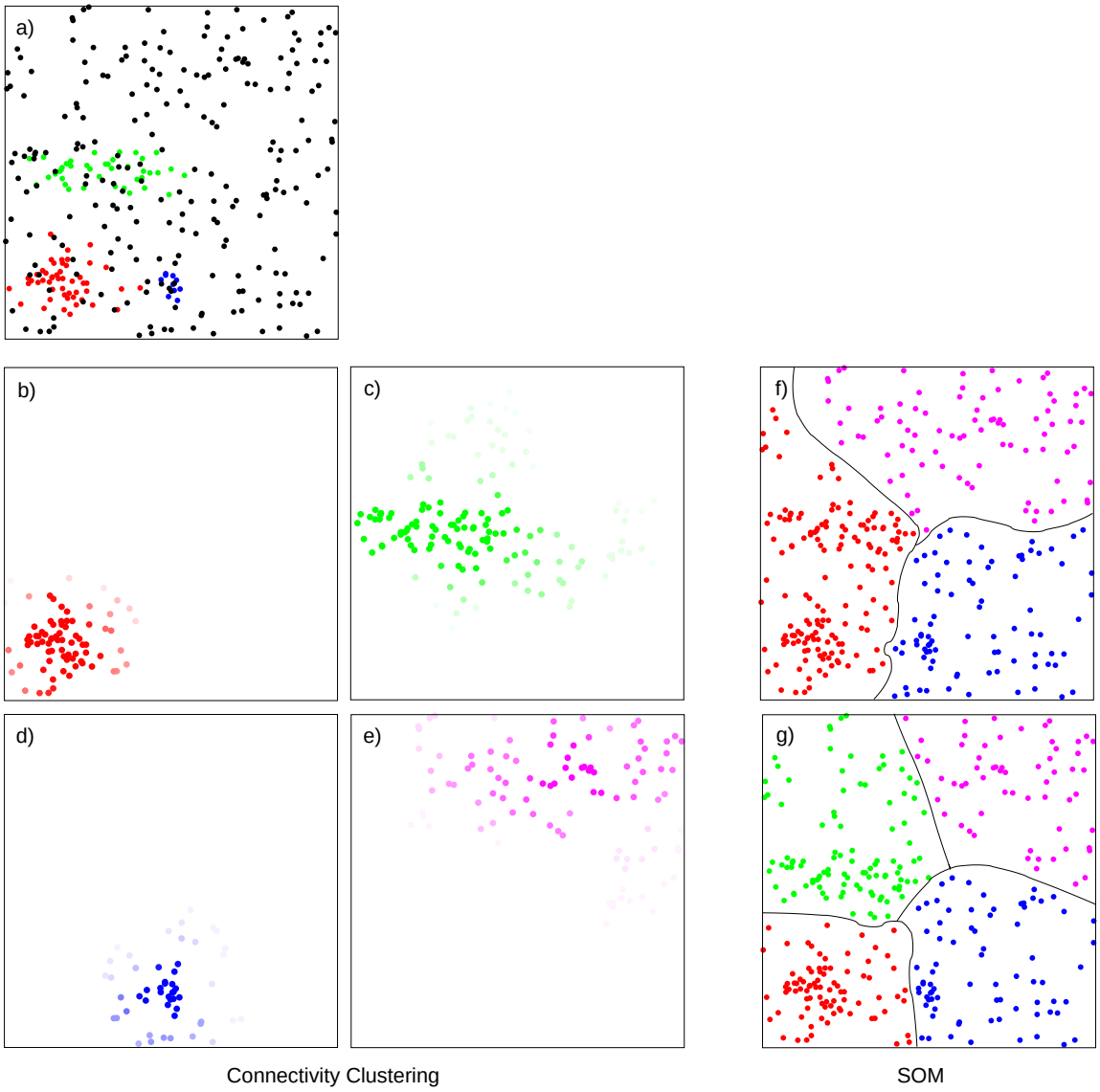


Figure 1: a) The constructed data set: 50 points (red) are drawn from a spherically symmetric Gaussian cluster, 40 points (green) are drawn from an elliptically symmetric Gaussian cluster, 10 points (blue) are drawn from a spherically symmetric Gaussian cluster of smaller width, and 225 points (black) representing noise are drawn from a uniform Poisson distribution. b)–e) Clusters of size larger than 20 as identified by connectivity clustering. The color intensity of every point is linearly proportional to the connectivity of that point to the cluster origin. f) Result of SOM clustering with a $3 \times 1$ structure of centroids. g) Result of SOM clustering with a $2 \times 2$ structure of centroids.

## 3 Dictyostelium development

*Dictyostelium discoideum* is a unicellular amoeba that lives in the soil, feeding on bacteria[10]. Upon starvation, a 24-hour-long developmental program ensues, leading eventually to the formation of a differentiated multicellular structure in which 80% of the original cells have become resistant spores and the remaining 20% have become stalk cells that physically support the spore. During development, *Dictyostelium* shares many of the physiological functions seen in mammalian cells such as directed amoeboid movement, cell-cell adhesion, tissue differentiation, proportioning, and sorting[11].

We have carried out the first cDNA microarray study of *Dictyostelium*, simultaneously measuring expression levels of nearly 700 selected genes at 2-hour intervals throughout development. Gene probes were microarrayed robotically on glass slides. The entire genome of *Dictyostelium* is estimated to carry 8,000–10,000 genes, and is in the process of being sequenced (at the present time about 3,000 genes have been identified). The genes used in this study included previously characterized developmentally regulated genes as well as genes encoding proteins with significant similarity to proteins characterized in other organisms but not previously encountered in *Dictyostelium*. A large number ($\sim 10^9$) of identical vegetative cells of wild-type strain NC4 were induced to synchronously initiate development by removal of nutrients, and spreading on a buffered surface. Standard methods were used to isolate total RNA from $\sim 10^8$ cells at each time point and to subsequently collect polyA$^+$ RNA. The RNA preparations were used as templates for reverse transcriptase to generate single-stranded cDNA copies of each mRNA. In order to determine relative temporal changes in specific gene expression free of slide or probe specific properties, a reference *mixture* of RNA collected from all experimental time points was copied into DNA in the presence of dCTP-Cy5 (red) fluorescent dye, while the RNAs collected during development were copied in the presence of dCTP-Cy3 (green) fluorescence dye. Microarrays were hybridized for 18 hours with a mixture of approximately equal amounts of Cy3 and Cy5 labelled DNA. The fluorescent intensities from each spot was measured and processed. The ratio of intensities in Cy3 and Cy5 channels for each spot were normalized to the value of this ratio at 2 hours of development. In order to find genes with similar *temporal* expression pattern, the relative ratios were all normalized to the same maximum, and processed by the connectivity clustering algorithm.

Three basic expression patterns were identified: early upregulated genes (Fig. 2a), late upregulated genes (Fig. 2b) and downregulated genes (not shown). [Note that these represent only a small number out of the total of 700 genes arrayed. Most of these genes are thus uninformative on the basis

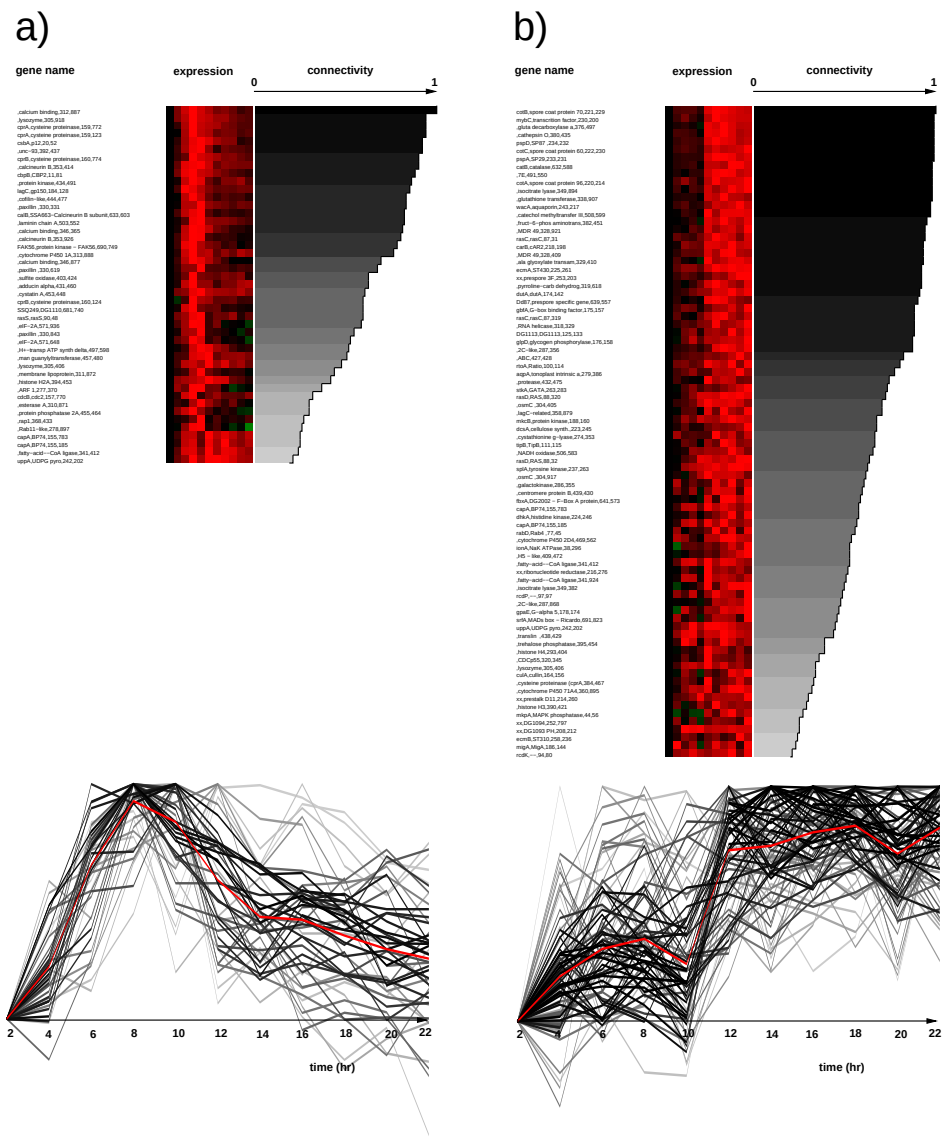


Figure 2: a) The two basic clusters of a) early and b) late upregulated genes as identified by connectivity clustering. Color coding of the expression profiles is as follows: black means gene expression is the same as it was at 2 hours of development; increasing tint of red color means increasing expression relative to 2 hours; and increasing tint of green color means decreasing expression relative to 2 hours. The bottom portions of the figure display expression profiles of the corresponding genes; the red curves are the mean expression. Only genes whose connectivity to the cluster origins is greater than 20% were included in these plots.

Table 1: Early upregulated genes (connectivity > 70%)

| locus | protein product/homolog | cell-type specificity |
|---|---|---|
| cprA | cysteine proteinase CP1 | established prestalk genes |
| cprB | cysteine proteinase CP2 | |
| lagC | adhesion factor gp150 | |
| - - | calcium binding | unknown |
| - - | lysozyme | |
| csbA | p12 | |
| - - | unc-93 | |
| calB | calcineurin B | |
| cbpB | $Ca^{++}$-binding protein CBP2 | |
| - - | protein kinase | |
| - - | cofilin-like | |
| paxB | paxillin | |
| - - | laminin chain A | |
| - - | protein kinase-FAK56 | |
| - - | cytochrome P450 1A | |

of the simplest experiments done here, and further experiments (e.g., response to drugs) are needed to resolve the functions of these unclusterable genes.] The clear separation of upregulated genes into early and late classes has not been previously noted. Analysis of the temporal patterns of two dozen developmentally regulated genes determined by Northern blotting confirms the dichotomy.

The leading members of the two clusters of upregulated genes are shown in Tables 1 and 2. Of the early genes the only ones whose cell-type specificity has been determined are pre-stalk. The majority, but not all, of the known cell-type specific late genes are pre-spore. Our microarray study reveals an abundance of other genes that strongly cluster within either the early or the late groups. Some of these genes have been previously studied, but not for cell-type specificity, while others are recently isolated and uncharacterized genes whose function can be presently inferred only by homology. These genes are the obvious target for a molecular genetic study. Determination of the cell-

type specificity of these genes is of great importance for further understanding of the genetic network that underlies functional changes during *Dictyostelium* development. Investigation of cell-type specificity of genes pointed out by this study is currently under way in our laboratory and the results will be reported elsewhere.

## Acknowledgments

TH and RS are grateful for helpful discussions with Nick Socci and Shoudan Liang in the initial stage of this project. This research is supported by the Burroughs-Wellcome Fund through an innovation award in functional genomics to TH and a LJIS post-doctoral training fellowship to RS. NI and WL acknowledge the support of the NSF grant No. 9728463.

## References

1. D. J. Lockhart *et al.*, Expression monitoring by hybridization to high-density oligonucleotide arrays, *Nat. Biotechnol.* **14**, 1675 (1996).
2. J. De Risi *et al.*, Exploring the metabolic and genetic control of gene expression on a genomic scale, *Science* **278**, 680 (1997).
3. M. B. Eisen *et al.*, Cluster analysis and display of genome-wide expression patterns, *Proc. Natl. Acad. Sci. USA 95* 14863 (1998).
4. P. Tamayo *et al.*, Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation, *Proc. Natl. Acad. Sci. USA* **96**, 2907 (1999).
5. U. Alon *et al.*, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, *Proc. Natl. Acad. Sci. USA* **96**, 6745 (1999).
6. J. Schuchhardt *et al.*, Normalization strategies for cDNA microarrays, *Nucl. Acids Res.* **28** (2000).
7. See, e.g., D. Stauffer and A. Aharony, *Introduction to percolation theory* (2nd ed., Taylor and Francis, London, 1992).
8. C. M. Fortuin and P. W. Kasteleyn, On the random-cluster model I. Introduction and relation to other models, *Physica* **57**, 536 (1972).
9. M. Blatt, S. Wiseman, and E. Domany, Superparamagnetic clustering of data, *Phys. Rev. Lett.* **76**, 3251 (1996).
10. W. F. Loomis, *Dictyostelium discoideum: a developmental system* (Academic Press, New York, 1975).
11. W. F. Loomis, Genetic networks that regulate development in Dictyostelium cells, *Microbiol. Rev.* **60**, 135 (1996).

Table 2: Late upregulated genes (connectivity > 85%)

| locus | protein product/homolog | cell-type specificity |
|---|---|---|
| cotA | spore coat protein | established prespore genes |
| cotB | spore coat protein | |
| cotC | spore coat protein | |
| pspD | spore coat protein (SP87) | |
| pspA | SP29 | |
| pspG | 3F | |
| wacA | aquaporin MIP | |
| - - | Dd87 | |
| - - | 7E | |
| ecmA | ST430 | established prestalk genes |
| mybC | transcription factor | |
| carB | CAR2 | |
| gadA | glutamate decarboxylase A | unknown |
| cprF | cathepsin O | |
| - - | isocitrate lyase | |
| - - | glutathione transferase | |
| - - | catechol methyltransfer III | |
| - - | fructose-6-P aminotransferase | |
| - - | MDR 49 | |
| rasC | RasC | |
| - - | alanine glyoxilate transaminase | |
| - - | pyrroline-carboxylate dehydrogenase | |
| dutA | DutA | |
| gbfA | G-box binding factor | |
| - - | RNA helicase | |
| - - | DG1113 | |
| glpD | glycogen phosphorylase | |
| - - | 2C-like | |