Clustering Gene Expression Patterns

Amir Ben-Dor[†], Zohar Yakhini[‡]
Hewlett-Packard Laboratories Israel
HPL-98-190
November, 1998

E-mail: zohary@hpl.hp.com
amirbd@cs.washington.edu

gene expression patterns, clustering, random graphs

With the advance of hybridization array technology researchers can measure expression levels of sets of genes across different conditions and over time. Analysis of data produced by such experiments offers potential insight into gene function and regulatory mechanisms. We describe the problem of clustering multi-condition gene expression patterns. We define an appropriate stochastic model of the input, and use this model for performance evaluations. We present a $0(n(\log(n))^c)$-time algorithm that recovers cluster structures with high probability, in this model, where n is the number of genes. In addition to the theoretical treatment, we suggest practical heuristic improvements to the algorithm. We demonstrate the algorithm's performance first on simulated data, and then on actual gene expression data.

[†]Department of Computer Science & Engineering, University of Washington, Seattle, Washington.
[‡]Hewlett-Packard Laboratories Israel, Technion City, Haifa 32000 Israel

Clustering Gene Expression Patterns

Amir Ben-Dor[*] Zohar Yakhini[†]

November 4, 1998

Abstract

With the advance of hybridization array technology researchers can measure expression levels of sets of genes across different conditions and over time. Analysis of data produced by such experiments offers potential insight into gene function and regulatory mechanisms. We describe the problem of clustering multi-condition gene expression patterns. We define an appropriate stochastic model of the input, and use this model for performance evaluations. We present a $O(n(\log(n))^c)$-time algorithm that recovers cluster structures with high probability, in this model, where n is the number of genes. In addition to the theoretical treatment, we suggest practical heuristic improvements to the algorithm. We demonstrate the algorithm's performance first on simulated data, and then on actual gene expression data.

1 Introduction

In any living cell that undergoes a biological process, different subsets of its genes are expressed in different stages of the process. The particular genes expressed at a given stage and their relative abundance are crucial to the cell's proper function. Measuring gene expression levels in different stages, different body tissues, and different organisms is instrumental in understanding biological processes. Such information can help the characterization of gene/function relationships, the determination of effects of experimental treatments, and the understanding of many other molecular biological processes.

Current approaches to measuring gene expression profiles include SAGE [Velculescu et al 97], RT/PCR [Somogyi et al 95], and hybridization based assays. In the latter, a set of oligonucleotides, or a set of appropriate cDNA molecules, is immobilized on a surface to form the hybridization array. When a labeled target DNA (or RNA) mixture is introduced to the array, target sequences hybridize to complementary immobilized molecules. The resulting hybridization pattern (detected, for example, by fluorescence) is indicative of the mixture's content. Hybridization arrays are thus used as molecular recognition tools for nucleic acids (see [Drmanac et al 91, Khrapko et al 91, Lennon Lehrach 91, Pevzner et al 91, Lysov et al 95, Blanchard Hood 96, Lin et al 96].)

These methods accelerate the rate at which gene expression pattern information is accumulated [Kim's Lab, Lockhart et al 96, DeRisi Iyer Brown 97, Wen et al 98, Khan et al 98] (also see Section 3 for more details). As a result, there is an increasing need to elucidate the patterns hidden in the data. However, the nature of studies of multiconditional gene expression patterns may widely vary. Accordingly, we are interested in analysis tools that may be useful in all such contexts. Clustering techniques are applicable as they would cluster sets of genes that "behave similarly" under the set of given conditions.

In cluster analysis, one wishes to partition entities into groups called clusters, so that clusters are homogeneous and well-separated. Clustering problems arise in numerous disciplines including biology, medicine, psychology, economics and others. There is a very rich literature on cluster analysis going back over two decades (cf. [Duda Hart, Everitt, Mirkin].) There are numerous approaches to defining quality criteria for solutions, stipulating the type of clustering sought, and interpreting the solutions. Algorithmic approaches also abound. Most formulations of the problem are NP-hard, so the algorithmics emphasizes heuristics and approximation. Clustering literature lacks concensus on basic definitions, probably due to the diversity of applications of the problem. A common theme in the literature is the need to fit the approach to the problem at hand and the necessity to assess the quality of solutions by subjective impression of experts in each area.

[*]Department of Computer Science & Engineering, University of Washington (amirbd@cs.washington.edu). Supported by the Program in Mathematics and Molecular Biology.

[†]HPLI, Hewlett Packard Laboratories (zohary@hpl.hp.com).

Analyzing multi-conditional gene expression patterns with clustering algorithms involves the following steps:

- Determination of the gene expression data (usually reported as vectors of real numbers).
- Calculation of a similarity matrix S. In this matrix the entry S_{ij} represents the similarity of the expression patterns for genes i and j. Many possible similarity measures can be used here. The actual choice should reflect the nature of the biological question and the technology that was used to obtain the data.
- A clustering algorithm. This is the main concern of this paper. The clustering algorithm should be effective and efficient. Its input is the similarity matrix mentioned above and its output is a set of clusters. Genes that belong to the same cluster have similar expression patterns, under the given conditions.
- Means for visually presenting the constructed solution (exemplified in Section 3).

Current approaches to clustering gene expression patterns ([Brown's Lab, NHGRI, Wen et al 98]) utilize hierarchical methods (constructing phylogenetic trees) or methods that work for Euclidean distance metrics (e.g k-means). We take a graph theoretic approach, and make no assumptions on the similarity function or the number of clusters sought. The cluster structure is produced directly, without involving an intermediate tree stage.

In Section 2.1 we describe the stochastic model used in this work. We then present a provably efficient method of solving the problem with high probability. In Section 2.2 we present a heuristic improvement of the said method and analyze its performance by simulations. In Section 3 we apply it to actual gene expression data, and analyze its output.

2 The Clustering Algorithm

2.1 Theory

We approach the clustering problem at hand by studying a stochastic model. A graph is called a *clique graph* if it is a disjoint union of complete graphs. Given a graph, consider the problem of finding its nearest clique graph, where distance is measured by the number of edges that must be changed (added or removed). Those cliques can be thought of as the underlying cluster structure of the graph. In the case of gene expression patterns it makes (biological) sense to assume that some true underlying cluster structure does exist for a graph that represents correlation between patterns of different genes. The underlying structure is, however, obscured by the complexity of the biological processes and corrupted by experimental errors. For our purposes it makes sense, therefore, to study the clustering problem on a random graph model built upon a cluster structure and corrupted at random. In this section we therefore assume that the input to the clustering problem are distributed according to the *corrupted clique-graph model* defined below. It is reminiscent of the planted bisection model ([Condon Karp 98]) and the planted clique model ([Alon Krivelevich Sudakov 98]).

In [Kučera 95] a variety of graph partitioning problems is considered, in the context of random graphs. The author considers the bisection problem and the graph coloring problem. He suggests algorithms for solving these with high probability and studies their expected complexity assuming some specific distributions on the input. In [Condon Karp 98] the authors consider the graph l-partition problem: partition the nodes of an undirected graph into l subsets of predefined sizes so that the total number of inter subset edges is minimal. They then present a linear (in the number of edges) time algorithm that solves the graph l-partition problem with high probability, on the planted l-partition model. Our problem is different: no predefined structure is given (any clique structure is, apriori, a possible candidate) and minimality with respect to inter cluster edges as well as intra cluster non-edges is sought.

Definition 2.1:

(i) A *cluster structure* is a vector $S = (s_1, s_2, ..., s_d)$, where $s_i > 0$ and $\sum s_i = 1$. For a cluster structure S, let $\gamma(S)$ = the smallest entry, $d(S)$ = the dimension, d.

(ii) A cluster structure $S = (s_1, s_2, ..., s_d)$ defines a clique graph on the vertices $\{1...n\}$ in the following way: The number s_i corresponds to a clique of size $\lfloor s_i N \rfloor$ (In our simulations we will choose N and S so that no rounding is needed. For asymptotic results the rounding is irrelevant) on the appropriate vertices. Call this graph $Q_n(S)$. A relabeling of the vertices of $Q_n(S)$ according to a permutation $\sigma \in S_n$ generates the clique graph $Q_n(S, \sigma)$.

∎

Definition 2.2: The random graph model $\mathcal{Q}(n, \alpha, S)$ (representing random corruption of clique graphs) is defined as follows: Given a cluster structure S and a value $0 \leq \alpha \leq \frac{1}{2}$, the random graph $\mathcal{Q}(n, \alpha, S)$ is obtained from $Q_n(S) = (V, E)$ by randomly (1) removing each edge in E with independent probability α; (2) adding each edge not in E with independent probability α; (3) permuting the vertices according to a uniformly chosen random permutation $\sigma \in S_n$. Edge inversions can be represented by a binary vector ζ of length $\binom{n}{2}$, where ij is inverted iff $\zeta_{ij} = 1$. The graph generated as above, from $(\sigma, \zeta) \in S_n \times \{0,1\}^{\binom{n}{2}}$ will be denoted $G(\sigma, \zeta) = (V, E(\sigma, \zeta))$. ∎

Definition 2.3: Consider an algorithm $\mathcal{A}$ that takes arbitrary graphs as inputs and returns clique graphs. Denote the output of $\mathcal{A}$ on $G = (V, E)$ by $\mathcal{A}(G) = (V, F)$. Accordingly, $F(\sigma, \zeta)$ is the edge set of $\mathcal{A}(G(\sigma, \zeta))$.

Let $\delta > 0$. We say that an algorithm $\mathcal{A}$ as above *clusters* $\mathcal{Q}(n, \alpha, S)$ *with probability* $1 - \delta$ if the output graph is, asymptotically, as good a solution as the original cluster graph is, with probability $1 - \delta$. That is,

$$\liminf_{n \to \infty} \mathbb{P}(|E(\sigma, \zeta) \triangle F(\sigma, \zeta)| \leq |E(\sigma, \zeta) \triangle Q_n(S, \sigma)|) > 1 - \delta.$$

Here and throughout this section $\mathbb{P}$ denotes the relevant probability measure (which is clear from the context).

Let $\delta(n) \to 0$. We say that an algorithm $\mathcal{A}$ as above *clusters* $\mathcal{Q}(n, \alpha, S)$ *with failure rate* $\delta(n)$ if

$$\limsup_{n \to \infty} \frac{1 - \mathbb{P}(|E(\sigma, \zeta) \triangle F(\sigma, \zeta)| \leq |E(\sigma, \zeta) \triangle Q_n(S, \sigma)|)}{\delta(n)} < \infty.$$

∎

Theorem 2.4: *Let S be a cluster structure and $\alpha < 1/2$.*

(i) For any fixed $\delta > 0$ there exists a $n(\log(n))^c$-time algorithm that clusters $\mathcal{Q}(n, \alpha, S)$ with probability $1 - \delta$. (c is a constant that depends only on the cluster structure S and on α).

(ii) For any $\delta(n) \in \Omega((\log(n))^{-b})$, where b is some constant, there exists a $n(\log(n))^{c(b)}$-time algorithm that clusters $\mathcal{Q}(n, \alpha, S)$ with failure rate $\delta(n)$. ($c(b)$ is as above but also depends on b).

(iii) For any $\delta(n) \in \Omega(n^{-b})$, where b is some constant, there exists a polynomial-time algorithm that clusters $\mathcal{Q}(n, \alpha, S)$ with failure rate $\delta(n)$.

To prove this theorem we shall present the algorithm and analyze its performance. It uses ideas similar to these presented in [Condon Karp 98] and [Kučera 95]. For the proof we need Theorem 2.5, due to Chernoff ([Chernoff 52], [Dembo Zeitouni, Section 2.2]). We use $D(p\|a)$ to denote the *relative entropy distance from* $(p, 1-p)$ *to* $(a, 1-a)$, That is, $D(p\|a) = p \log(p/a) + (1-p) \log((1-p)/(1-a))$.

Theorem 2.5: *(Chernoff, 1952) Let $X \sim$ Binomial(N, p). Let $a < p < b$. Then*

$$\mathbb{P}(X > b) < \exp(-ND(b\|p)), \quad \textit{and} \quad \mathbb{P}(X < a) < \exp(-ND(a\|p)).$$

We also need a very crude sampling lemma, stated without proof:

Lemma 2.6: *Consider n objects of d different colors, each color represented by at least n/m objects. If s objects are sampled uniformly, and independently without replacement then*

$$\mathbb{P}(\textit{ The sample contains} \geq s/2m \textit{ representatives of each color }) > 1 - \delta,$$

providing $16m^2 \log(d/\delta) \leq s \leq \frac{n}{4m}$.

Proof: (of Theorem 2.4)

Sketch. Before presenting the complete proof, we outline the idea. Consider a simpler scenario - assume that the hidden structure S, consists of only two clusters, red, and blue. We say that a $\log n$-subset of vertices is a *core* if it is monochromatic (either all red or all blue). The algorithm has two phases. In the first phase it forms a list $\mathcal{L}$ of core candidates. In the second it uses each core candidate, $L \in \mathcal{L}$, as a classifier, to partition the rest of the vertices: vertices with at least $\frac{\log n}{2}$ neighbors in L versus those that have fewer neighbors in L. Finally, the partition that is closest (in the symmetric difference sense) to the input graph is returned.

The analysis of the algorithm above is based on the following:

- A list of core candidates, $\mathcal{L}$, that positively contains a core can be generated in polynomial time - choose an arbitrary subset A of size $2\log n$ and let $\mathcal{L}$ be the list of all $\log n$-subsets of A.
- Assume that a core is used as a classifier to produce the vertices partition. Using large deviations bound we show that the produced partition is as good as the original cluster structure with high probability.

Note that the time complexity of the second phase is $O(n \log n)$ times the size of $\mathcal{L}$. To reduce the time complexity order we replace the first phase of the algorithm above by a "recursive" application of the algorithm. We generate a list that contains $O(\log n)$ sub-core candidates, each with $\log\log n$ vertices. Each sub-core candidate is used to grow a core candidate, which in turn is used to grow the complete partition.

Complete proof. For clarity we analyze the case $d(S) = 3$, $\gamma(S) = 1/m$. Generalizing to more clusters is straight forward.

We are given a graph on n vertices that was obtained from a cluster structure S by the process described in Definition 2.2. Call the vertices of the original clusters blue, red and white. Write $V = B \cup R \cup W$. For a vertex $v \in V$ let $C(v)$ denote the subset it belongs to (before corrupting the clique graph). Let $\delta > 0$ (will be related to the tolerated failure probability, at the end). Let $k(\alpha) = \lceil 2/D(1/2\|\alpha)\rceil$.

- Uniformly draw a subset U_1 of vertices of size $r \cdot k(\alpha)\log\log(n)$ where r is determined so that with probability $1-\delta$ each color has at least $k(\alpha)\log\log(n)$ representatives in this chosen subset. By Lemma 2.6 r can be set to be $2m$ providing that n is large enough: $\log\log(n) > 8m\log(1/\delta)$ and $n > 8m^2\log\log(n)$.
- Uniformly over the subsets of $V \setminus U_1$ draw a subset U_2 of vertices with $r \cdot k(\alpha)\log(n)$ elements. Again, r is such that with probability $1-\delta$ each color has at least $k(\alpha)\log(n)$ representatives in this chosen subset and $r = 2m$ suffices under the above assumptions.
- Consider all partitions of U_1 into 3 subsets (there are less than $\log(n)^{r\cdot k(\alpha)\log(3)}$ of them). Call the subsets of each such partition B_1^C, R_1^C and W_1^C. Run the following enumerated steps starting with all these partitions. For the analysis focus on a partition where $B_1^C \subset B$, $R_1^C \subset R$ and $W_1^C \subset W$ (such a partition is, indeed, considered, since we are considering all partitions).

1. For all $u \in U_2$ let $\hat{C}(u)$ be the color that attains $\max(\deg(u, B_1^C)/|B_1^C|, \deg(u, R_1^C)/|R_1^C|, \deg(u, W_1^C)/|W_1^C|)$. Add u to that set. Assume that $C(u) = B$. The collection of edges from u to B_1^C are independent Bernoulli$(1-\alpha)$ (the drawings of U_1 and U_2 were independent of everything else). Therefore $\deg(u, B_1^C) \sim$ Binomial$(|B_1^C|, 1-\alpha)$. Using the Chernoff bound stated above we therefore have

$$\begin{aligned}\boldsymbol{P}\big(\deg(u, B_1^C) \le |B_1^C|/2\big) &< \exp(-|B_1^C| D(\tfrac{1}{2}\|\alpha)) \\ &< \log(n)^{-k(\alpha)D(\frac{1}{2}\|\alpha)} \qquad (1)\\ &< \log(n)^{-2}, \qquad (2)\end{aligned}$$

where (1) follows from $|B_1^C| \ge k(\alpha)\log\log(n)$. Similarly, $\deg(u, R_1^C) \sim$ Binomial$(|R_1^C|, \alpha)$, and thus

$$\boldsymbol{P}\big(\deg(u, R_1^C) \ge |R_1^C|/2\big) < \exp(-|R_1^C| D(\tfrac{1}{2}\|\alpha)) < \log(n)^{-2}. \qquad (3)$$

The same holds for W_1^C, whence $\hat{C}(u) = C(u)$ with high probability: $\boldsymbol{P}\Big(\hat{C}(u) \ne C(u)\Big) < 3\log(n)^{-2}$. Finally, by a union bound

$$\boldsymbol{P}\Big(\hat{C}(u) \ne C(u) \text{ for some } u \in U_2\Big) < 3r \cdot k(\alpha)\log(n)^{-1}. \qquad (4)$$

2. Focusing on the part of the measure space where no error was committed in the previous steps (in particular, all vertices were assigned to their original color), we now have three subsets of vertices $B_2^C \subset B$, $R_2^C \subset R$ and $W_2^C \subset W$, each of size at least $k(\alpha)\log(n)$. We take all other vertices and classify them using these subsets, as in the previous step. Observe that all edges used in this classification are independent of the algebra generated by everything previously done. This is true since in the previous step only edges from U_2 to U_1 were considered, and these are of no interest here. Therefore, the equivalents of (2) and (3) hold, yielding

$$\boldsymbol{P}(\text{any } v \in V \text{ was not assigned to } C(v)) < 3r \cdot k(\alpha) n^{-1}. \qquad (5)$$

- Amongst all outputs of the above, chose the partition which is closest (in the symmetric difference sense) to the input graph.

The total probability of failure in this process is estimated as follows

$$\begin{aligned} \mathbb{P}\left(\begin{array}{c} \text{The original partition } V = B \cup R \cup W \\ \text{is not one of the outputs} \end{array} \right) &< 2\delta + 3r \cdot k(\alpha) \left(n^{-1} + \log(n)^{-1} \right) \\ &\leq 2\delta + 6m \cdot k(\alpha) \left(n^{-1} + \log(n)^{-1} \right), \qquad (6) \end{aligned}$$

which is arbitrarily small for large n. As noted above, we have less than $\log(n)^{2m \cdot k(\alpha) \log(3)}$ parallel processes here. In each one the expensive part (time-wise) is the classification of all vertices in $V \setminus (U_1 \cup U_2)$, using the core clusters B_2^C, R_2^C and W_2^C. In this stage $O(n \log(n))$ edges are considered, each at most once: sums over disjoint subsets of these are compared to a threshold. Thus the time spent here is $O(n \log(n))$ and the total time complexity is $O(n \log(n)^{2m \cdot k(\alpha) \log(3) + 1})$. This proves (i).

To see that (ii) holds observe that the dominant term in (6) is $\log(n)^{-1}$ and that the degree here can be increased by pushing $k(\alpha)$ up, paying a price in the time complexity (the power of $\log(n)$ there would increase). The proof of (iii) is along similar lines and is omitted here. ∎

2.2 Practice

In this section we take a more practical approach, and present a novel and simple clustering heuristic, called `Cluster Affinity Search Technique`, or, in short, `CAST`. The algorithm uses the same idea as in the theoretical algorithm described in Theorem 2.4, namely it relies on average similarity (affinity) between unassigned vertices and the current cluster seed to make its next decision. However, it differs from the theoretical algorithm in some aspects: (1) The theoretical algorithm repeats the same process for many initial seeds. Here we use "cleaning" steps to remove spurious elements from cluster seeds and avoid the repetition. (2) `CAST` adds (and removes) elements from the current seed one at a time (and not independently, as in the theoretical algorithm). Heuristically, this helps by strengthening the constructed seed, thus improving the decision base for the next step. (3) `CAST` handles more general inputs. Namely, it allows the user to specify both a real-valued similarity matrix, and a threshold parameter which determines what is considered significantly similar. This parameter controls the number and sizes of the produced clusters.

The input to the algorithm is a pair $\langle \Sigma, t \rangle$, where Σ is a n-by-n similarity matrix ($\Sigma(i,j) \in [0,1]$), and t is a similarity cutoff. The clusters are constructed one at a time. The currently constructed cluster is denoted by C_{open}. We define the *affinity* of an element x, denoted by $a(x)$, to be the sum of similarity values between x and the elements in C_{open}. We say that an element x is *of high affinity* if $a(x) \geq t|C_{\text{open}}|$. Otherwise, x is called *of low affinity*. Note that an elements' status (high/low affinity) depends on C_{open}. Roughly speaking, `CAST` alternates between adding high affinity elements to C_{open}, and removing low affinity elements from it. When this process stabilizes C_{open} is closed and a new cluster is started. A pseudo-code of the algorithm is given in Figure 1.

We remark that the "cleaning" steps in `CAST` serve to avoid a common shortcoming shared by many popular clustering techniques (such as `single-linkage`, `complete-linkage`, `group-average`, and `centroid`): due to their "greedy" nature, once a decision to join two clusters is made, it cannot be reversed (see [Everitt, ch. 4]).

2.2.1 Performance Analysis

As is sometimes the case with practical heuristics, it is very hard to prove rigorous performance bounds for `CAST`. Instead, we assess its performance by testing its ability to recover hidden cluster structures in computer generated random data. Recall (Definition 2.2) that the corrupted clique graph random graph model is specified by three parameters: a cluster structure S, an error probability α, and a size parameter n. For different choices of these parameters, we perform the following steps:

- Draw a random graph $G = G(\sigma, \zeta)$, from the distribution $\mathcal{Q}(n, \alpha, S)$.
- Apply `CAST` to G (viewed as a binary similarity matrix), using cutoff $t = 0.5$.

For each such trial we compute the similarity between the output cluster structure and the original clique graph $Q_n(S, \sigma)$ using similarity coefficients (matching coefficient and Jaccard's coefficient [Everitt, p. 41]). Visual judgment can also be used.

Clustering Affinity Search Technique

- Input: An n-by-n similarity matrix Σ, and a cutoff parameter t.
- Initializations:

```
$\mathcal{C} \leftarrow \emptyset$                /* The collection of closed clusters */
$C_{\text{open}} \leftarrow \emptyset$            /* The constructed cluster */
$U \leftarrow \{1, \ldots, n\}$            /* Elements not yet assigned to any cluster */
$a(\cdot) \leftarrow 0$                 /* Reset the affinity (total similarity between $v$ and elements in $C_{\text{open}}$) */
```

- **while**($U \cup C_{\text{open}} \neq \emptyset$) **do**

```
        Let $u$ be an element with maximal affinity in $U$.
        if ($a(u) \geq t|C_{\text{open}}|$)             /* $u$ is of high affinity */
          $C_{\text{open}} \leftarrow C_{\text{open}} \cup \{u\}$        /* Insert u into $C_{\text{open}}$ */
          $U \leftarrow U \setminus \{u\}$                /* Remove u from U */
          For all $x$ in $U \cup C_{\text{open}}$ do
                $a(x) = a(x) + \Sigma(x, u)$    /* Update the affinity */
          end
        else    /* No high affinity elements outside $C_{\text{open}}$ */
            Let $v$ be a vertex with minimal affinity in $C_{\text{open}}$.
            if ($a(v) < t|C_{\text{open}}|$)              /* $v$ is of low affinity */
              $C_{\text{open}} \leftarrow C_{\text{open}} \setminus \{v\}$        /* Remove v from $C_{\text{open}}$ */
              $U \leftarrow U \cup \{v\}$                 /* Insert v into U */
              For all $x$ in $U \cup C_{\text{open}}$ do
                    $a(x) = a(x) - \Sigma(x, v)$    /* Update the affinity */
              end
            else   /* $C_{\text{open}}$ is clean */
                $\mathcal{C} \leftarrow \mathcal{C} \cup C_{\text{open}}$   /* Close the cluster */
                $C_{\text{open}} \leftarrow \emptyset$       /* Start a new cluster */
                $a(\cdot) \leftarrow 0$          /* Reset affinity */
            end
        end
    end
```

- Done, return the collection of clusters, $\mathcal{C}$.

Figure 1: CAST Algorithm

For completeness we define the above mentioned similarity coefficients. Let $M(S)$ be the adjacency matrix of $Q_n(S, \sigma)$ (a n-by-n matrix). That is, $M(i, j) = 1$ if and only if i and j belong to the same cluster. Similarly, let $M(\mathcal{C})$ denote the adjacency matrix of the output cluster structure. Let N_0, N_1, N_2 denote the number of entries that have '0' in both matrices, the number of entries that have '1' in both matrices, and the number of entries that differ in the two matrices, respectively. The matching coefficient is simply the ratio of the total number of entries on which the two matrices agree, to the total number of entries: $(N_0 + N_1)/(N_0 + N_1 + N_2)$. Jaccard's coefficient is the corresponding ratio when "negative" matches (N_0) are ignored: $N_1/(N_1 + N_2)$.

In Table 1 we report results (based on at least 100 executions) for various choices of the model parameters. Figure 2 visually presents clustering results.

Cluster structure (S)	n	α	Matching coeff.	Jaccard's coeff.
$\langle 0.4, 0.2, 0.1, 0.1, 0.1, 0.1 \rangle$	500	0.2	1.0	1.0
$\langle 0.4, 0.2, 0.1, 0.1, 0.1, 0.1 \rangle$	500	0.3	0.999	0.995
$\langle 0.4, 0.2, 0.1, 0.1, 0.1, 0.1 \rangle$	500	0.4	0.939	0.775
$\langle 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1 \rangle$	1000	0.3	1.0	1.0
$\langle 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1 \rangle$	1000	0.35	0.994	0.943

Table 1: Results from simulations.

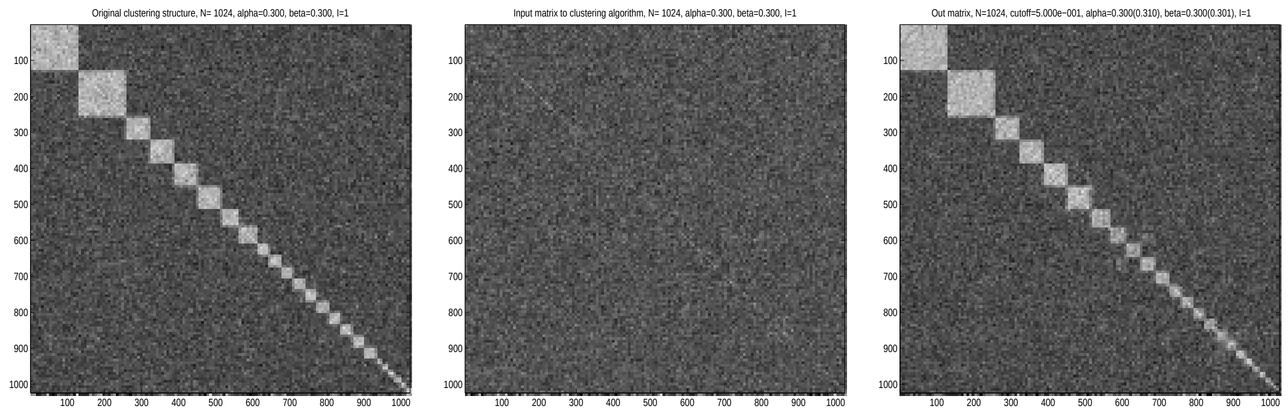

Figure 2: The visual effect of clustering. The figure on the left depicts the adjacency matrix of the corrupted clique graph, before vertices are permuted. The middle figure constitutes the actual input to the algorithm (vertices are randomly permuted). The figure on the right is the output. Note that all large clusters were recovered. Some of the very small ones were lost, but these differences are hard to detect.

2.2.2 Implementation Notes

All of the software developed (including the `CAST` algorithm, the synthetic data generation, and the visualization tools) was implemented using MATLAB. The expression matrix for the data in [Kim's Lab] is 1246×146. Running one clustering execution on it takes under ten seconds on a HP Vectra XU 6/180MHz (after a one time preprocessing step that computes the similarity matrix). The similarity matrix for actual gene expression data can take large memory space. When this is a problem it is possible to compute all similarity values when they are needed. Since (in `CAST`) some entries of the matrix are accessed more than once this increases the computation time. The expression data in [DeRisi Iyer Brown 97] (a $\sim 6000 \times 7$ matrix) was analyzed in this manner.

3 Applications to Biological Data

3.1 Temporal Gene Expression Patterns

As a first example of applying our clustering techniques to gene expression data we analyze the data reported, analyzed and discussed in [Wen et al 98]. In this study the authors establish some relationships between temporal gene expression patterns of 112 rat CNS (Central Nervous System) genes and the development process of the rat's CNS. Three major gene families are considered: Neuro-Glial Markers family (NGMs), Neurotransmitter Receptors family (NTRs) and Peptide Signaling family (PepS). All other genes measured in this study are lumped by the authors into a fourth family: Diverse (Div). All families are further subdivided by the authors, based on apriori biological knowledge.

Gene expression patterns for the 112 genes of interest were measured (using RT/PCR: [Somogyi et al 95]) in cervical spinal cord tissue, at nine different developmental time points. This yields a 112×9 matrix of gene expression data. To capture the temporal nature of this data, the authors transform each (normalized) 9-dimensional expression vector into a 17-dimentional vector - 8 difference values (between time adjacent expression levels) were included. This transformation emphasizes the similarity between genes with closely parallel, but offset, expression patterns. Euclidean distances between the augmented vectors were computed, yielding a 112×112 distance matrix. Next, A phylogenetic tree was constructed for this distance matrix (using FITCH, [Felsenstein 93]). Finally, Cluster boundaries were determined by visual inspection of the resulting tree. Some correlation between the resulting clusters and the apriori family information was observed.

We analyze the same data in the following way. The raw expression data is preprocessed in a similar manner - first the normalized expression levels are augmented with the derivative values. Then, a similarity matrix is computed based on the L_1 distance between the augmented 17-dimensional vectors. A hands-off version of our algorithm, which automatically searches for a good cutoff value, is applied to the similarity matrix (the eventual cutoff for the presented data was 0.647). Clusters are directly inferred.

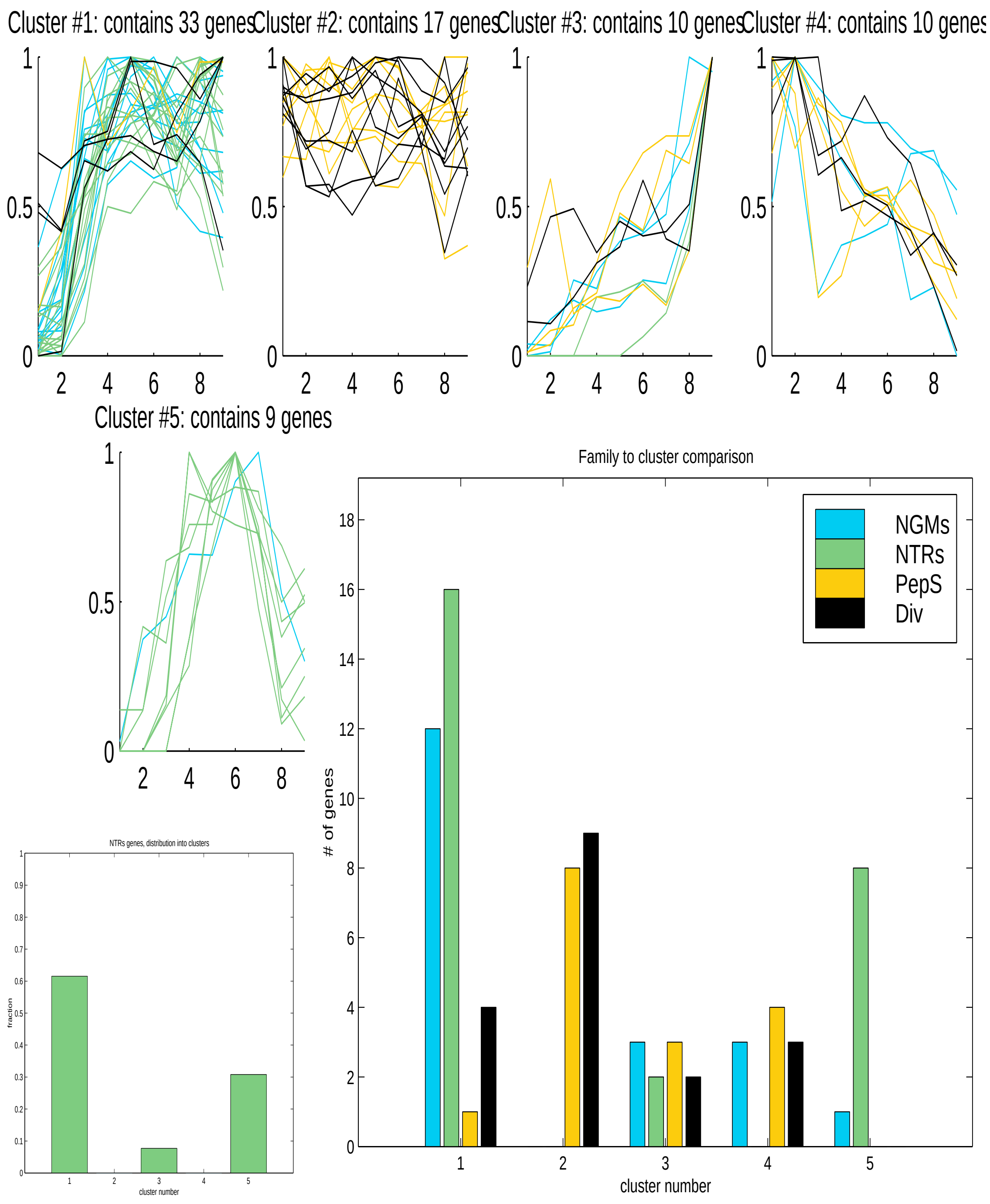

Figure 3: In the top figure the expression patterns of genes in each of the clusters are depicted. The graphs are color coded so as to distinguish between members of the various families (see legend in the bottom figure). Our software enables comparison to any user defined partition into families. Note a single NGM in cluster #5 that is dominated by NTRs. The bottom figure summerizes the composition of all clusters, in terms of the defined families. The distribution into clusters, within each one of the individual families can also be displayed, as in the example here.

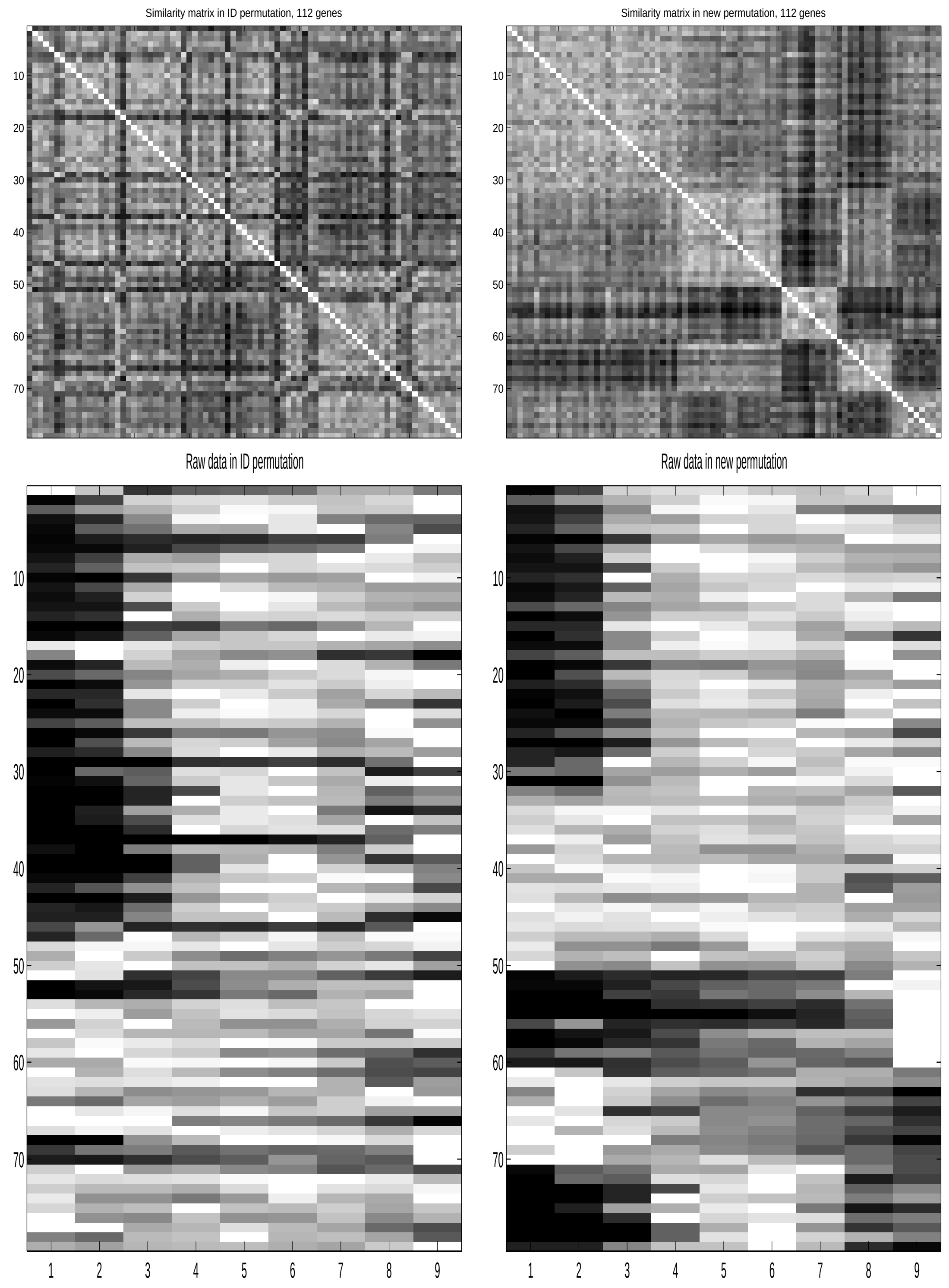

Figure 4: The unprocessed data is compared to the output of the clustering algorithm. Top: the two similarity matrices are depicted. Since the original unprocessed data is ordered according to the four families, some pattern can be detected in the raw data as well. Bottom: The raw gene expression matrix is ordered according to the permutation produced by the clustering algorithm and compared to the original order.

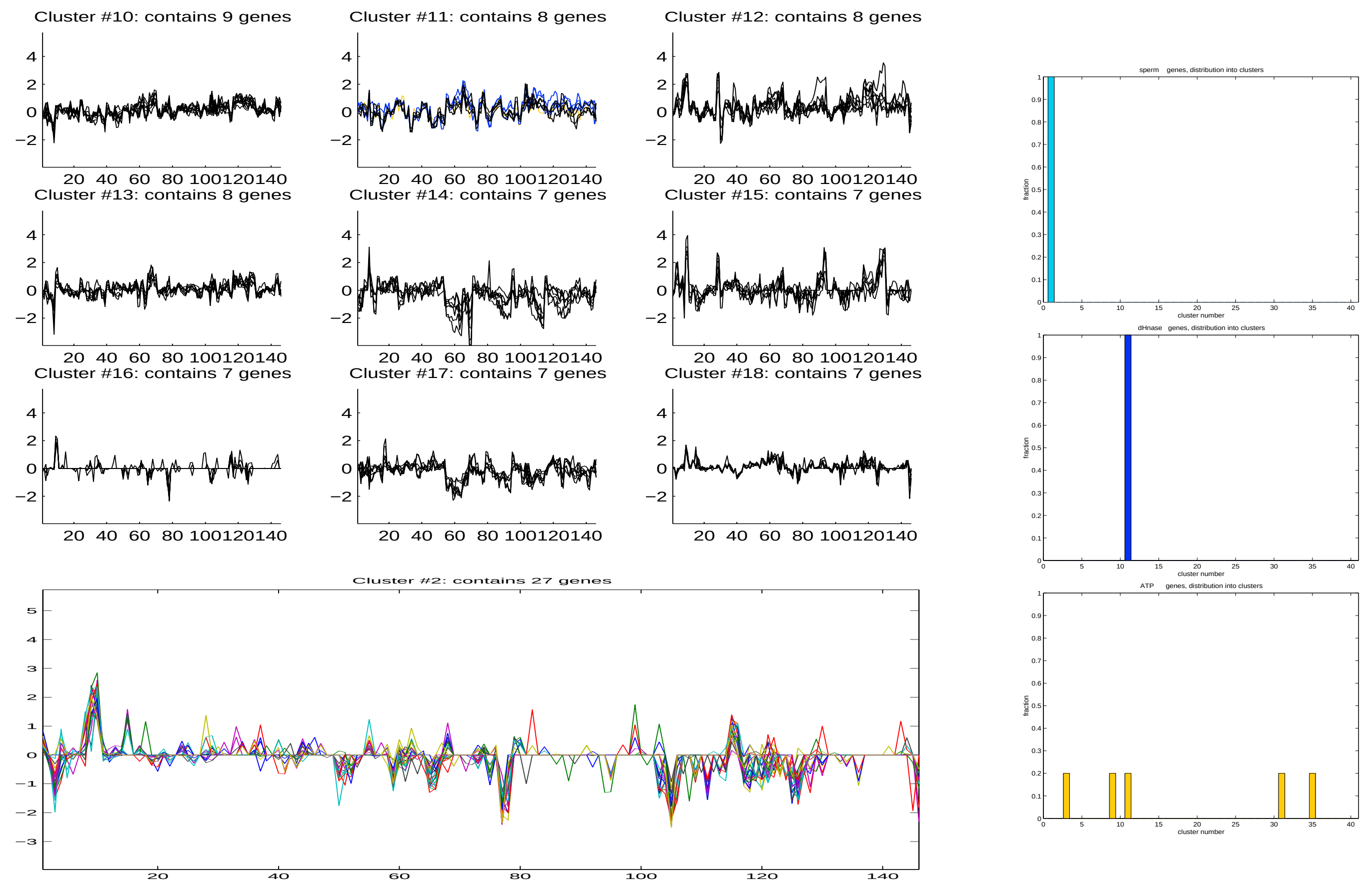

Figure 5: Top left: examples of the clusters found in analyzing the data in [Kim's Lab]. Below we have enlarged cluster No. 2. Note that it is possible to identify the regions that significantly contribute to correlations within a cluster and then analyze the corresponding sub matrix. We are currently working on automating and benchmarking this process. Right: not much information is available about how the genes studied are grouped into families. Therefore, the family comparison utility is presented here mostly for the purpose of validation. Genes coding sperm proteins (8 genes) were all clearly clustered together. The same is true for dehydrogenase related genes (3 of them). ATP related genes don't specifically correlate with any other pattern. This is expected since ATP is involved in all cell processes and is not correlated with specific conditions.

3.2 Multi Experiment Analysis

Clustering gene expression patterns is useful even if the experiments' enumeration has no physical meaning (as opposed to temporal patterns). In [Kim's Lab] studies of gene regulation mechanisms in the nematode *C. elegans* using cDNA microarrays hybridization assays are described. Some software tools (Acacia Biosciences, Inc.) for analyzing the raw data are also accessible from [Kim's Lab]. Using our methods and tools we analyzed the data for 1246 genes, from 146 experiments. The data is in the form $\log\left(\frac{\text{Red}}{\text{Green}}\right)$ (representing the log-ratio of the two sample intensity values at the corresponding array feature), per experiment. Some experiments are parts of time courses and some compare certain mutants to a reference cell. Here we only present some initial clustering results, without further pursuing any of the implied relationships.

Contrary to Section 3.1, where the similarity measure needed to reflect the temporal nature of the data, the order of experiments here, in the total set, has little or no importance. Therefore, we use a Pearson correlation based similarity measure here. Figure 5 summerizes the results. For time courses it makes sense to use other similarity measures when the corresponding sub matrices are clustered. Clustering the columns (rather than the rows) of the expression matrix is also possible and contains biologically meaningful information.

Acknowledgements We thank Amir Dembo for helping us in rigorously handling the stochastic model. We thank Stuart Kim for access to the *C. elegans* data and for stimulating discussions on computational approaches to gene expression data interpretation. We thank Mel Kronick for introducing us to the problem and for valuable advice and information. We thank Ron Shamir for stimulating discussions and for contributing valuable ideas and insight.

References

[Alon Krivelevich Sudakov 98] N. Alon, M. Krivelevich and B. Sudakov, *Finding a large hidden clique in a random graph*, *Proc. Ninth Annual ACM-SIAM Sympoium on Discrete Algorithms* (SODA 98), San Francisco, California, pp 594-598, 1998.

[Blanchard Hood 96] A. P. Blanchard and L. Hood, *Sequence to array: probing the genome's secrets*, Nature Biotechnology, vol 14, p 1649, 1996.

[Brown's Lab] P. Brown's laboratory web page: `http://cmgm.stanford.edu/pbrown/` .

[Chernoff 52] H. Chernoff, *A measure for the asymptotic efficiency for tests of a hypothesis based on the sum of observations*, Ann. of Math. Stat., vol 23, pp 493-509, 1952.

[Condon Karp 98] A. Condon and R. M. Karp, *Algorithms for Graph Bisection on the Planted Bisection Model*, manuscript, personal communication, 1998.

[Dembo Zeitouni] A. Dembo, O. Zeitouni, **Large Deviations Techniques and Applications**, 2nd edition, Springer-Verlag, New York, 1998.

[DeRisi Iyer Brown 97] J. DeRisi, V. Iyer and P. Brown, *Exploring the Metabolic Genetic Control of Gene Expression on a Genomic Scale*, Science, vol 278, pp 680-686, 1997.

[Drmanac et al 91] R. Drmanac, G. Lennon, S. Drmanac, I. Labat, R. Crkvenjakov and H. Lehrach, *Partial sequencing by oligohybridization: Concept and applications in genome analysis*, *Proceedings of the first international conference on electrophoresis supercomputing and the human genome. Edited by C. Cantor and H. Lim*, (World Scientific, Singapore), pp 60-75, 1991.

[Duda Hart] R. O. Duda and P. E. Hart, **Pattern classification and scene analysis**, Wiley-interscience, NY, 1973.

[Everitt] B. Everitt, **Cluster analysis**, 3rd edition, Edward Arnold, London, 1993.

[Felsenstein 93] J. Felsenstein, PHYLIP version 3.5c, Univ. of Washington, Seattle, 1993.

[Khan et al 98] J. Khan, R. Simon, M. Bittner, Y. Chen, S. B. Leighton, T. Pohida, P. D. Smith, Y. Jiang, G. C. Gooden, J. M. Trent, and P. S. Meltzer, *Gene Expression Profiling of Alveolar Rhabdomyosarcoma with cDNA Microarrays,* manuscript, accepted to Cancer Research, 1998.

[Khrapko et al 91] K. R. Khrapko, A. Khorlin, I. B. Ivanov B. K. Chernov, Y. P. Lysov, S. Vasilenko, V. Floreny'ev, A. Mirzabekov, *Hybridization of DNA with Oligonucleotides Immobilized in Gel: A Convenient Method for Detecting Single Base Substitutions,* Molecular Biology 25, number 3, pp 581-591, 1991.

[Kim's Lab] Stuart Kim's laboratory web page: `http://cmgm.stanford.edu/∼kimlab/` .

[Kučera 95] L. Kučera, *Expected Complexity of Graph Partitioning Problems.* Discrete Applied Math, 57, pp 193-212, 1995.

[Lennon Lehrach 91] G.S. Lennon and H. Lehrach, *Hybridization analysis of arrayed cDNA libraries*, Trends Genet, Vol 7, pp 60-75, 1991.

[Lin et al 96] C. Y. Lin, K. H. Hahnenberger, M. T. Cronin, D. Lee, N. M. Sampas, and R. Kanemoto, *A Method for Genotyping CYP2D6 and CYP2C19 Using GeneChip probe Array Hybridization*, 1996 ISSX Meeting.

[Lockhart et al 96] D. J. Lockhart, H. Dong, M. C. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Horton, and E. L. Brown, *DNA Expression monitoring by hybridization to high density oligonucleotide arrays*, Nature Biotechnology 14, pp 1675-1680, DEC 1996.

[Lysov et al 95] Y. Lysov, A. Chernyi, A. Balaev, F. Gnuchev, K. Beattie, A. Mirzabekov, *DNA sequencing by contiguous stacking hybridization on modified oligonucleotide matrices*, Molecular Biology 29, number 1, pp 62-66, 1995.

[Mirkin] B. Mirkin, **Mathematical classification and clustering**, Kluwer academic publishers, 1996.

[NHGRI] NHGRI's laboratory web page: `http://www.nhgri.nih.gov` .

[Pevzner et al 91] P. A. Pevzner, Y. Lysov, K. R. Khrapko, A. Belyavsky, V. Floreny'ev, A. Mirzabekov, *Improved Chips for Sequencing by Hybridization*, J Biomolecular Str. Dyn., Vol 9, number 2, pp 399-410, 1991.

[Somogyi et al 95] R. Somogyi, X. Wen, W. Ma and J. L. Barker, *Developmental kinetics of GAD family mRNAs parallel neurogenesis in the rat spinal cord*, J. Neurosci, Vol 15, No 4, pp 2575-2591, 1995.

[Velculescu et al 97] V. E. Velculescu et al, *Characterization of the yeast transcriptome*, Cell, Vol 88, pp 243-251, 1997.

[Wen et al 98] X. Wen, S. Fuhrman, G. S. Michaels, D. B. Carr, S. Smith, J. L. Barker and R. Somogyi, *Large-scale temporal gene expression mapping of central nervous system development*, PNAS, Vol 95, No 1, pp 334-339, 1998.