A Combinatorial Approach to Clustering Gene Expression Data

Brendan Mumey[1]*, Louise Showe*[2]*, Michael Showe*[2]

[1] *Computer Science Department, Montana State University, Bozeman, MT 59717, USA*
[2] *The Wistar Institute, Philadelphia, PA 19104, USA*

ABSTRACT

We present a new algorithm to discovering natural partitions of a set of samples based on their gene expression patterns found with microarray experiments. The algorithm is based on a bi-criteria combinatorial optimization problem to simultaneously identify an interesting set of genes and a partition of the array samples. Each gene in the gene set should respect the sample partition in the sense that if the gene's values are colored according to the partition class they come from, then the values, when sorted, should have a minimal number of color changes. We refer to this as the full color criterion. It measures how well a particular gene sorts the various partition classes. The other is the black and white criterion that measures how well a gene distinguishes one sample class from the remaining samples. While we show that the optimization problem is NP-complete, we are effectively able to solve problems of interest. Additionally we can calculate a "p-value" to interpret the significance of the results. We have tested the algorithm on a 30 sample Cutaneous T-cell Lymphoma data set; it was able to almost perfectly discriminate short-term survivors from long-term survivors and normal controls.

INTRODUCTION

Microarray clustering is not a new topic; a variety of approaches have been proposed for clustering both genes (based on their expression similarities across multiple samples or conditions) and array samples (usually based on their expression similarities across a set of identified interesting genes). Both supervised and unsupervised methods have been developed [3, 4, 8, 9]. For the former case, techniques such as linear discriminate analysis (and similar methods such as SVM and principal component analysis) have been applied to microarray data to identify a subset of interesting genes that can be used to classify the array samples according to their labels. Many unsupervised clustering methods have also been employed; k-means, self-organizing maps, hierarchical clustering to name a few [13]. The problem of feature selection is particularly relevant as many genes in the microarray set may be uninformative and must somehow be lowered in relevance in order to find good clusters. The CLIFF algorithm [12] uses an expectation-maximization approach to unsupervised clustering that iteratively refines the clustering (using the normalized-cut algorithm) and the set of genes selected. Other randomized approaches to unsupervised clustering have also been proposed, in particular the CLICK algorithm based on a graph-clustering algorithm [1, 2, 11].

We propose a new method for simultaneously identifying a set of interesting genes and a partition of the samples. It can be used in a completely unsupervised fashion, or it can be partially supervised to keep certain samples together (such as controls) or apart (such as separating patients from controls). The algorithm uses a bi-criteria combinatorial optimization search to simultaneously determine an interesting set of genes and a partition of the array samples. Each gene in the gene set should respect the array partition in the sense that if the gene's values are colored according to the array sample partitions they come from, then the values when sorted should have a minimal number of color changes. We refer to this as the full color criterion. It measures how well a particular gene sorts the partitions. The other is the black and white criterion where we color the values of one of the partitions black and the remaining values white and again count the number of color changes. For each gene, we choose the partition to color black that minimizes this count. This criterion measures how well a gene distinguishes one sample class from the remaining samples. In next section, we formally define these criteria and describe a branch-and-bound algorithm to find best partitioning and gene set. Next we prove that the general problem of finding the partitioning and gene set with the minimum number of color changes is NP-complete. Subsequently we discuss how to calculate "p-values" for each gene found; this is just the likelihood of seeing the gene scores found at random. We then describe our results using the algorithm on a 30 sample leukemia data set.

---

[1] To whom correspondence should be addressed.

The results are quite interesting. The algorithm was able to differentiate the patients who were long-term survivors from those who were short-term in an unsupervised way. The set of genes found are of particular interest to cancer researchers who are studying this data set. We conclude with some ideas for improvements and potential new uses for this method.

DEFINITIONS

The input to the algorithm consists of a microarray data set consisting of a set of samples that have been measured against a common microarray. The output of the algorithm is (1) a subset *G* of the genes and (2) a partitioning *P* of the samples. The selected subset of genes supports the partitioning in the following combinatorial sense: If each gene's values are colored according to the array sample partitions they come from, then the values when sorted should have a minimal number of color changes; we call this the *color change* of the gene. We also consider a slightly different scoring scheme called *black and white* change. This statistic looks at a partitioning as if one partition was colored black and remainder were colored white then counts the number of changes that occur; we chose the black partition so as to minimize the number of changes. The idea of also counting B&W changes is to find genes that are good at differentiating one class from all of the remaining classes. We define this formally as follows: Let m be the number of genes, n be the number of array samples and let the matrix $E = (e_{ij})$, where e_{ij} is the expression value of gene i in sample j. Suppose that $P = \{P_1, \ldots, P_k\}$ is a k-partitioning of the arrays samples and that $p(i)$ denotes the partition that sample i belongs to. Let π_g sort the expression values of gene g, i.e. $e_{g,\pi g(1)} \leq e_{g,\pi g(2)} \leq \ldots \leq e_{g,\pi g(n)}$. Let $I[b]$ be a Boolean indicator function, i.e. $I[true] = 1, I[false] = 0$. We define

$$\text{color change(g)} = \sum_{i=1}^{n-1} I[p(\pi_g(i)) \neq p(\pi_g(i+1))]$$

and

$$\begin{aligned}&\text{B \& W change(g)}\\ &= \min_{c=1}^{k} \sum_{i=1}^{n-1} I[\,|\{\pi_g(i), \pi_g(i+1)\} \cap P_c| = 1]\,.\end{aligned}$$

Clearly, genes that have low color and B&W changes are more interesting with respect to a given sample partition. We are interested in finding a large set of genes that mutually support one particular partition of the array samples and can now state the partitioning/gene selection problem precisely:

GENEPART OPTIMIZATION PROBLEM

Input: The input consists of a *m x n* expression matrix *E*, *k*, the number of partitions, *s*, a minimum class size, and *p*, the desired size of the selected gene set *G*.
Output: The output is the partition $P = \{P_1, \ldots, P_k\}$ and gene set *G* that is the solution to

$$\arg\min_{(P,G)} \sum_{g \in G} \text{color change(g)} + \text{B \& W change(g)}$$

subject to the constraints $|P_i| \geq s$ for $i = 1 \ldots k$ and $|G|=p$.

ALGORITHM

We employ a combinatorial search algorithm based on branch-and-bound to find P and G. The central data structure used is a search tree T. Each internal node of T represents a partial solution where some of the samples have been assigned to partition classes but some are unset. This is accomplished as follows: We view T as a rooted tree, the level of a node in T is simply the depth it is at from the root. Edges are considered to have the same depth as their originating parent node. We will use edges at level i to indicate the partition of sample i. In general, there will be k edges leaving each node at level i to indicate the k possible partitions that sample i could belong to. (Note: if sample i is constrained to belong or not belong to a particular class then there may be fewer such edges. Also, we remove edges to eliminate partitions that are same up to a relabeling of the classes.) It follows that leaf nodes in T represent complete solutions where the samples have been fully partitioned.

Each new search node x in T is evaluated by considering its (partial) assignment of samples to partitions and then calculating the B&W and color change of each gene in G. If samples are not currently assigned partitions, then they are simple omitted from the scoring calculation. Once this is done, we can sort the genes according to their computed score and keep the best p (in practice, we use a selection algorithm instead of a sorting algorithm to increase efficiency). We sum the scores of these selected genes and assign this score to the node n in question, denoted `score(x)`. Notice that scores can only increase as we descend the search tree; this is because assigning a new sample to a partition can only increase the number of B&W and color changes for a given gene. This means that we can use a node x's current score as a lower bound on any partitioning solutions that are descendants of x. So, if we have already discovered a solution pair (G,P) with a lower score than x, then we can immediately prune the subtree rooted at x.

A further consideration is the order in which nodes are considered for expansion in T. At any point in the search, there is a set of nodes that are on the edge of the search; that is none of their immediate descendants have been examined. We term this set of nodes, the *frontier* of T. We chose which node in the frontier to expand next by considering its current score to depth ratio; this is simply its current score (as defined above) divided by the level of the node in T. The idea is that nodes with a low ratio are more likely to yield good solutions than those with higher ratios and so should be expanded first.

We can describe the tree search algorithm in pseudo-code. Let F represent the search node frontier.

Tree Search Algorithm

```
1. F = { root }
2. best = null
3. while (F not empty) {
     remove node x in F with lowest score/depth
     if (best != null OR score(x) < score(best))
       add the children of x to F
     if (x is a leaf AND
        (best = null OR score(x) < score(best))
       best = x
   }
4. return best
```

Upon complete of this algorithm, we will have found the best search node; associated with this node is the partition P of the samples as well as the associated set of genes G that has the fewest total B&W and color changes.

During the search, we can enforce user-specified constraints on the partitioning. For example, we may wish to exclude partitions that mix particular groups of samples (e.g. patients from controls). Our initial experience with this search algorithm is that it is able to find the optimal *(P,G)* pair relatively quickly. In the experimental results section we will discuss our experience using the algorithm on a 9600 clone by 30 sample leukemia data set involving 12 controls and 18 patient samples (we specified that controls should be kept separate from patients); depending on the number of partitions found, running times would vary from a few minutes to half a day on a moderate Alpha workstation.

COMPUTATIONAL COMPLEXITY

We outline a proof that that GENEPART optimization problem is NP-complete. The reduction is from a graph partitioning problem; the problem of deciding whether a graph G contains a complete bipartite subgraph of a given size. The vertices of such can graphs can be divided into two groups L and R such that the edge set of the graph contains L x R (i.e. every vertex in L is connected to every vertex in R) such that $|L| = |R| = |V|/2$. This problem is NP-complete for general graphs with an even number of vertices.

Lemma: GENEPART is NP-Complete

Proof: We sketch the reduction. Let G=(V,E) be a graph that we would like to decide if it contains a complete bipartite subgraph. We will create an instance of the GENEPART problem as follows: The tissues samples will be the vertices in v; that is V form the columns of the expression matrix. For each edge e = (u,v) in E we create a new row in the expression matrix that is equal to 0 in all positions except at column u will equal -1 and at column v will equal 1. We claim that G contains a complete bipartite subgraph if and only if we can find a 2-partition of the samples (vertices) and $p = |V|^2/4$ genes (edges) with exactly p total color changes (one change per gene). Observe that if an edge's endpoints are in different partitions then there exists a sorting permutation of the corresponding gene values such that there is exactly one color change. On the other hand if an edge's endpoints are in the same partition then there must be at least two color changes (the -1 and +1 are the same color and there will be 0's of the other color in between). Thus if we can find a 2-partition with p genes that have exactly one color we know the corresponding edges form a complete bipartite subgraph of the desired size.

STATISTICAL SIGNIFICANCE

In this section we develop a means to evaluate the significance of each gene found in *(P,G)* using a "*p*-value" style calculation. Such a calculation asks "what is the likelihood of observing the genes found and their associated color changes in random data?" For this analysis, we assume that each row in *E* is independently randomly generated and that each sorting permutation π_g of the values of gene *g* is equally likely. Suppose that a gene *g* has *c* color changes for some fixed partition $P = \{P_1, ..., P_k\}$. We are interested in calculating the probability that this would occur by chance. Let $s_i = |P_i|$. We can view the problem as calculating the chance that exactly *c* color changes occur in a random permutation of colored objects, where there are *k* colors present and exactly s_i objects with color *i*. To solve this we first need to count the number of distinguishable ways of permuting the objects with exactly *c* color changes. Dividing by the number of distinguishable permutations in total will yield the desired probability. Let $CC(\{s_1, ..., s_k\}, c)$ be the number of distinguishable permutations of the objects above with exactly *c* color changes. It is relatively straightforward to determine a recursive expression for this quantity. The base case occurs when $k = 2$, that is two colors are present. If *c* is even, then endpoints of the permutation must have the same color (either black or white). We can view the process of creating a distinguishable permutation as first placing all the objects of one color in a row and then selecting *c/2* internal gaps between consecutive objects to insert the objects of the remaining color into. Once the gaps are chosen we need to partition the remaining objects into *c/2* nonempty groups to insert into these gaps. This is tantamount to picking *c/2-1* internal "split points" in the list of the remaining objects. Accounting for the fact that then endpoints are either black or weight, we arrive at the top case in the expression below. When *c* is odd, analogous reason holds (the leading *2* in the expression results from the fact that the distinguishable permutation can start with either color first).

$$CC(\{s_1, s_2\}, c) = \begin{cases} \binom{s_1 - 1}{c/2}\binom{s_2 - 1}{c/2 - 1} + \binom{s_2 - 1}{c/2}\binom{s_1 - 1}{c/2 - 1} & \text{if c is even} \\ 2\binom{s_1 - 1}{\frac{c-1}{2}}\binom{s_2 - 1}{\frac{c-1}{2}} & \text{if c is odd} \end{cases}$$

For $k > 2$, we apply a similar argument to the one given above. Suppose that we have placed all of the objects having the first *k-1* colors and that this has yielded $d < c$ color changes. We need to then consider the number of ways to place the remaining objects of color *k* in order to reach a total of *c* color changes upon completion. Observe that if one of the remaining objects is inserted in between two objects of the same color, then two color changes are created. However, if one of the remaining objects is inserted at a location that already has a color change, then there is only a net increase of one color change. Suppose there are *e* insertions of the latter case and *f* insertions of the former case. This yields a net increase of $e + 2f$ colors, so we require that $d + e + 2f = c$. There are exactly *d+2* positions that will yield a net increase of one color change (the existing internal color change locations and the two endpoints), we choose *e* of them. This leaves $n - s_k - d - 1$ internal locations to choose that will increase the color change count by *2*; we choose *f* of them. Finally we need multiply by the number of ways we can divide up the color *k* objects into *e+f* nonempty groups; the final factor in the following formula accounts for this:

$$CC(\{s_1, \ldots, s_k\}, c) = \sum_{\substack{d+e+2f=c \\ d,e,f\geq 0}} CC(\{s_1, \ldots, s_{k-1}\}, d)\binom{d+2}{e}\binom{n-s_k-d-1}{f}\binom{s_k-1}{e+f-1}$$

We can now state the likelihood of finding gene *g* with exactly *c* color changes given a partition *P*. This is simply the number of distinguishable permutations with c color changes divided by the total number of distinguishable permutations:

$$\Pr(\text{ color change(g)} = c \mid P) = CC(\{s_1, \ldots, s_k\}, c) \Big/ \binom{n}{s_1, \ldots, s_k}$$

In the case of B&W changes, suppose that *P* is again given and that we select partition *i* to be black and remaining partitions are colored white. Applying the above formula we have:

$$\Pr(\text{ B \& W change(g)} = c \mid P, i \text{ selected}) = CC(\{s_i, n - s_i\}, c) \Big/ \binom{n}{s_i, n - s_i}$$

To improve the readability of the results, we take the negative logarithm (base 10) of all of the probabilities calculated. So, a larger value indicates more significance. Typical values can be seen in Tables 1B and 2B at the end of the paper.

EXPERIMENTAL RESULTS

To test our GENEPART algorithm (implemented in C++ and primarily executed on an Alpha workstation) we have used a 30 sample data set created by the Showe laboratory at the Wistar Institute. The data set consists of microarrays for 18 samples from 17 patients (one patient sampled twice) and 12 controls previously described and analyzed by the Showe laboratory at the Wistar Institute. The data is obtained from single-channel cDNA microarrays spotted with 9600 features. The RNA samples were derived from ficoll-gradient purified peripheral blood mononuclear cells (PBMC). Patients with SS are clinically identified as having T-cells with aberrant cerebriform nuclei in excess of 5% of morphologically normal T-cells. In these 18 patient samples, SS cells comprised 65-99% of the total PBMC. Survival in this group of patients ranged from 1 month to >5 years. There is no correlation between percent circulating malignant cells and time of survival, so the derivation of a prognostic indicator based on gene expression will be clinically important. The patients have been previously described elsewhere [6].

As an initial test, we first constrained the program to find a list of the genes that had the fewest changes when the partition consisted of two classes; the patient samples and the controls. Of the top 10 genes (CD8B1, TLE1, TPR, PLS3, TNFRSF5, QSCN6, TNSF10, SCYA4, KLF4, FCER2), five are over expressed and five are under expressed in patients relative to controls. Four of these (PLS3, TNFRSF5 TNSF10, and SCYA4) are also among the most informative genes found by Penalized discriminant analysis (PDA) [5, 10] on a data set of 6,000 features representing 4500 genes [6]. Of the next 10, two more, IL1R1 and CD26 are also informative for PDA in this dataset. Two other genes which are highly informative using PDA, STAT4 and DUSP1 appear slightly lower on the list. Many of these genes, e.g. CD8B1 and STAT4 are important T-cell specific proteins. The progressive loss of CD8 cells is an indicator of poor prognosis in these patients and the loss of STAT4 is consistent with the T-helper cell type 2 phenotype of the malignant cells. PLS3 is a unique single-gene marker for 70% 0f patients with SS, since it is never expressed in normal lymphocytes. Other T-cell specific genes include IL4R; the receptor for interleukin 4, and IL-11Ralpha.

The 3-partition shown in Tables 1A & 1B distinguishes the short-term (ST) from long-term (LT) survivors with only two classification errors out of 18. Interestingly, the mean of expression values of controls is, with the exception of 3 genes, always intermediate between that of the ST survivors, which are down regulated and that of the LT-survivors, which are up regulated. The exceptions are PLS3, and KLF4 up regulated in both patient classes, and SATB1, down in both disease classes. Among the most consistent classifying genes with regulatory functions are FOXO1A, a forkhead family transcription factor and MAP4K1, and DLG4, regulatory kinases. The best scoring gene, HEM1, is highly expressed in T-cells.

The 4-partition is, perhaps the most interesting. A small group, consisting of four long-term and one short-term survivor has mean expression values similar to the controls. The remaining groups are mainly ST and mainly LT patients respectively. The gene expression of the former are uniformly down regulated by about 2-fold, while the latter are up regulated relative to controls between 50 and 100%. Interestingly, the most consistent genes, showing fewest color changes, show little overlap with the genes which are informative using PDA. Although the differences between these two classes and controls are not large, the removal of a class of LT survivors (nearer normal) from the remainder of them leaves a larger difference between the ST and LT which show a ratio of 3-4 in the most informative cases. It may be possible to produce a smaller classifier from these genes than was possible with genes selected using PDA. However, some note must be made of the fact that the original PDA studies were carried out using fewer genes. Some of the genes from the 3-partition are also informative in the 4-partition. Among the "new" genes are LIF, a macrophage induction cytokine, and CHD1, a DNA helicase.

DISCUSSION AND FUTURE WORK

We have developed a new combinatorial approach to the problem of finding interesting genes and partitioning the samples from microarray experiments. Based on the two criteria of counting both color changes and B&W changes we are able to compute the optimal sample partition and gene set. On a fairly large data set, the partitioning that was found separated long-term survivors from short-term with only two misclassifications. This was substantially better than a hierarchical clustering algorithm applied to a set of genes found using discriminant analysis [6].

We have contemplated and are currently testing some refinements to the algorithm. Currently, one must specify the number of partitions to look for and a minimum partition size. This is somewhat artificial as ultimately we are interested in finding the most statistically significant partitions of the samples. It should be possible to develop bounds on the likelihood of finding a particular *(P,G)* pair at random; these bounds could then be used to prune the search to find only the most statistically significant partitions. For example, a data set might have a very statistically significant four-way partition but have no interesting five-way partitions. Along these lines, it would be interesting to investigate suboptimal solutions. There may be several interesting *(P,G)* pairs possibly involving different genes in one data set.

Once an interesting *(P,G)* pair has been found, one could also look for patterns in the genes. Some genes may similar expression patterns across the partitions; potentially suggestive of common regulatory pathways. One idea is to perform a phylogenetic analysis on the samples (for example to study and indicate a evolutionary progression in cancer). Briefly, we could identify "states" for a gene by using select color changes as state boundaries (i.e. a gene with a color change of 1 would two states). For each array sample (or perhaps sample partition) we could then associate a gene state vector that records what state each gene is in for that sample (or partition, on average). The number of gene states would be fairly small, so a standard phylogeny search algorithm could be then be used to search for evolutionary relationships among the samples. Finally, another feature of the algorithm that may be useful is that it is tolerant to missing data; blank entries in the expression matrix E can simply be ignored for the purposes of calculating color changes.

REFERENCES

1. Ben-Dor, A. N. Friedman, Z.Yakhini. 2001. *Class discovery in gene expression data.* Proceedings of the fifth annual international conference on Computational biology.
2. Ben-Dor, A., B. Chor, R. Karp, Z. Yakhini. 2002. *Discovering local structure in gene expression data: the order-preserving submatrix problem*. Proceedings of the sixth annual international conference on Computational biology.
3. Bickel, D. 2003. *Robust cluster analysis of microarray gene expression data with the number of clusters determined biologically*. Bioinformatics, 19(7):818-24.
4. Eisen, M B., P T. Spellman, P.O Brown, D. Botstem. 1998. *Cluster analysis and display of genome-wide expression patterns.* Proc. Nat. Acad. Sc. USA, 95:14863-14868.
5. Hastie, T., A. Buja, R. Tibshirani. 1995. *Penalized Discriminant Analysis.* Annals of Statistics 23:73-102.
6. Kari, L., A. Loboda, M. Nebozhyn, A.H. Rook, E.C.Vonderheid , C. Nichols, D. Virok, C. Chang, W.-H. Horng, J. Johnston, M. Wysocka, M.K. Showe, and L.C. Showe. *Classification and Prediction of Survival in Patients with the Leukemic Phase of Cutaneous T-Cell Lymphoma.* J. Exp Med 197: 1477-1488.
7. Krishnapuram, B., L. Carin, A.J. Hartemink. 2003. *Joint classifier and feature optimization for cancer diagnosis using gene expression data.* Proceedings of the seventh annual international conference on Computational molecular biology.
8. Li, W., I. Grosse. 2003. *Gene selection criterion for discriminant microarray data analysis based on extreme value distributions.* Proceedings of the seventh annual international conference on Computational molecular biology.
9. Madeira, S.C., Arlindo L. Oliveira. 2004. *Biclustering Algorithms for Biological Data Analysis: A Survey.* IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB).
10. Raychaudhuri, S. 2001. *Penalized Discriminant Analysis.* TIBS 19:189-193.
11. Sharan, R., A. Maron-Katz, R. Shamir. 2003. *CLICK and EXPANDER: A System for Clustering and Visualizing Gene Expression Data.* Bioinformatics Vol. 19 No. 14 pp. 1787-1799.
12. Xing, E. Karp, R. 2001. *CLIFF: clustering of high dimensional microarray data via iterative feature filtering using normalized cuts.* Bioinformatics, 17, 306-315.
13. Yeung, K., D. Haynor, W. Ruzzo. 2001. *Validating clustering for gene expression data.* Bioinformatics, 17:309-318.
14. Yeung, K., W. Ruzzo. 2001. Principal component analysis for clustering gene expression data. Bioinformatics, 17:763-774.

Table 1A – Results for the B30 data set, 3-partition case

This table shows the optimal partitioning of the B30 CTCL data set into three classes. Partition 0 was constrained to contain all of the controls. All but one sample in partition 1 was a short-term survivor. All but one sample in partition 2 was a long-term survivor.

Partition 0:	Partition 1:	Partition 2:
sample 0: Th0-C022 1	sample 12: ST-S123 1	sample 16: ST-S114 3
sample 1: Th0-C021 1	sample 13: ST-S127 1	sample 18: LT-S116 1
sample 2: Th0-C004 1	sample 14: ST-S112 5	sample 19: LT-S110 3
sample 3: Th2-C002 2	sample 15: ST-S115 1	sample 20: LT-S118 3
sample 4: Th2-C003 2	sample 17: ST-S107 2	sample 21: LT-S138 1
sample 5: Th2-C010 2	sample 28: LT-S109 1	sample 22: LT-S119 2
sample 6: Th2-C011 2		sample 23: LT-S136 1
sample 7: Th2-C018 6		sample 24: LT-S150 1
sample 8: Th2-C019 6		sample 25: LT-S118 2
sample 9: Th2-C020 6		sample 26: LT-S113 4
sample 10: Th2-C021 3		sample 27: LT-S149 1
sample 11: Th2-C022 3		sample 29: LT-S111 1

Table 2A – Results for the B30 data set, 4-partition case

This table shows the optimal partitioning of the B30 CTCL data set into four classes. Partition 0 was constrained to contain all of the controls. Partition 1 consists of four long-term and one short-term survivor has mean expression values similar to the controls. Partition 2 consists of short-term survivors and one long-term. Partition 3 contains the remaining long-term survivors and one short-term.

Partition 0:	Partition 1:	Partition 2:	Partition 3:
sample 0: Th0-C022 1	sample 12: ST-S123 1	sample 13: ST-S127 1	sample 16: ST-S114 3
sample 1: Th0-C021 1	sample 21: LT-S138 1	sample 14: ST-S112 5	sample 18: LT-S116 1
sample 2: Th0-C004 1	sample 23: LT-S136 1	sample 15: ST-S115 1	sample 19: LT-S110 3
sample 3: Th2-C002 2	sample 24: LT-S150 1	sample 17: ST-S107 2	sample 20: LT-S118 3
sample 4: Th2-C003 2	sample 27: LT-S149 1	sample 28: LT-S109 1	sample 22: LT-S119 2
sample 5: Th2-C010 2			sample 25: LT-S118 2
sample 6: Th2-C011 2			sample 26: LT-S113 4
sample 7: Th2-C018 6			sample 29: LT-S111 1
sample 8: Th2-C019 6			
sample 9: Th2-C020 6			
sample 10: Th2-C021 3			
sample 11: Th2-C022 3			

Table 2B – Results for the B30 data set, 3-partition case

In this table, we show a subset of interesting genes from the top 200 genes with the fewest color changes for a 3-partition of the B30 CTCL data set. PLS3 was also the most informative gene found using discriminant analysis [6]. See text for a discussion of the biological relevance of these genes.

Gene	2710 Ha02 Hs.239176 insulin-like growth factor 1 receptor IGF1R	3727 Ha02 Hs.170133 forkhead box O1A (rhabdomyosarcoma) FOXO1A	3461 Ha02 Hs.86575 mitogen-activated protein kinase 1 MAP4K1	3649 Ha02 Hs.23731 discs, large (Drosophila) homolog 4 DLG4	3178 Ha02 Hs.91093 chitinase 1 (chitotriosidase) CHIT1	3732 Ha02 Hs.13495 requiem, apoptosis response zinc finger gene REQ	4012 Ha02 Hs.132834 hematopoietic protein 1 HEM1	2222 Ha01 Hs.74592 (binds to nuclear matrix/scaffold-associating DNA's) SATB1	5687 Ha03 Hs.4114 plastin 3 (T isoform) PLS3	199 Ha01 Hs.356370 Kruppel-like factor 4 (gut) KLF4
Coloring	0.69^{13}	2.56^{13}	13.70^{13}	0.58^{13}	7.00^{17}	3.14^{13}	22.81^{14}	1.93^{25}	0.29^{10}	0.24^{10}
	0.78^{14}	3.00^{17}	15.92^{17}	1.41^{15}	7.15^{14}	3.28^{17}	24.59^{15}	2.30^{22}	0.49^{28}	0.30^{11}
	0.88^{15}	3.71^{14}	22.76^{15}	1.46^{17}	8.97^{13}	3.35^{15}	28.15^{17}	2.50^{20}	0.54^{11}	0.31^{6}
	0.98^{17}	3.86^{12}	22.97^{14}	1.62^{14}	9.44^{15}	4.69^{14}	29.61^{13}	2.56^{18}	0.55^{9}	0.35^{9}
	1.08^{12}	4.16^{15}	28.52^{28}	1.94^{12}	12.24^{28}	4.89^{12}	33.79^{28}	4.08^{29}	0.59^{4}	0.40^{4}
	1.24^{28}	4.52^{28}	28.71^{12}	2.03^{28}	14.22^{10}	5.12^{28}	43.46^{6}	4.52^{16}	0.63^{0}	0.42^{5}
	1.28^{5}	4.60^{6}	30.74^{7}	2.11^{9}	16.78^{11}	6.26^{10}	45.03^{7}	4.78^{19}	0.64^{7}	0.42^{7}
	1.38^{10}	4.79^{5}	31.79^{6}	2.13^{2}	18.98^{7}	6.37^{7}	55.20^{12}	5.00^{27}	0.67^{3}	0.43^{8}
	1.42^{7}	4.99^{7}	33.50^{11}	2.15^{10}	19.21^{9}	6.54^{6}	56.49^{9}	5.15^{26}	0.76^{8}	0.48^{3}
	1.58^{2}	5.26^{2}	36.57^{10}	2.44^{6}	20.05^{12}	7.28^{5}	56.62^{11}	7.07^{24}	0.78^{5}	0.81^{12}
	1.60^{6}	5.26^{11}	38.34^{2}	2.64^{11}	22.95^{0}	7.55^{11}	57.40^{2}	7.23^{12}	0.79^{12}	0.82^{19}
	1.62^{9}	5.48^{10}	40.56^{9}	2.65^{5}	26.16^{5}	7.96^{2}	58.49^{10}	7.80^{13}	0.86^{1}	0.93^{16}
	1.65^{11}	5.51^{9}	43.09^{5}	3.19^{7}	27.07^{8}	8.81^{23}	59.76^{5}	7.88^{17}	0.91^{6}	0.99^{22}
	1.70^{0}	5.91^{23}	48.73^{21}	3.38^{3}	27.48^{6}	9.73^{26}	62.65^{27}	8.14^{28}	0.95^{2}	1.09^{25}
	1.79^{3}	6.40^{27}	49.10^{27}	3.57^{26}	27.67^{2}	10.25^{27}	66.18^{21}	10.66^{23}	0.97^{13}	1.17^{29}
	1.82^{26}	6.94^{3}	50.68^{23}	3.58^{27}	28.70^{1}	10.31^{0}	69.14^{23}	10.74^{1}	1.13^{15}	1.21^{23}
	1.83^{23}	8.06^{21}	55.88^{3}	3.61^{23}	29.14^{20}	10.50^{9}	72.14^{24}	11.08^{5}	1.19^{18}	1.32^{21}
	2.03^{1}	8.15^{4}	56.91^{24}	3.74^{18}	30.42^{26}	10.69^{24}	72.60^{26}	14.12^{8}	1.78^{22}	1.63^{28}
	2.05^{24}	8.41^{22}	59.34^{26}	3.91^{21}	30.44^{24}	10.94^{21}	75.53^{20}	14.65^{14}	4.72^{26}	1.72^{13}
	2.09^{22}	8.53^{24}	61.56^{22}	3.94^{24}	30.61^{25}	10.94^{4}	83.62^{1}	14.70^{21}	4.82^{27}	1.79^{15}
	2.20^{27}	9.08^{20}	65.14^{25}	4.04^{1}	32.07^{29}	11.03^{20}	84.04^{4}	15.96^{11}	5.31^{21}	1.87^{1}
	2.24^{20}	9.72^{26}	69.13^{16}	4.11^{0}	33.46^{18}	11.89^{3}	85.31^{3}	16.24^{7}	5.58^{17}	2.01^{0}
	2.36^{21}	9.79^{8}	69.19^{0}	4.56^{25}	35.94^{3}	12.08^{22}	86.47^{0}	16.84^{4}	8.90^{29}	2.09^{20}
	2.44^{29}	9.95^{1}	72.17^{4}	4.72^{4}	37.04^{23}	13.86^{29}	87.03^{25}	17.69^{3}	10.82^{19}	2.22^{2}
	2.66^{8}	10.19^{0}	77.05^{20}	4.85^{22}	40.52^{19}	14.01^{25}	95.91^{22}	17.76^{2}	12.19^{24}	2.34^{14}
	2.72^{18}	10.88^{25}	77.75^{1}	4.88^{16}	41.06^{27}	16.49^{1}	97.83^{16}	19.22^{0}	15.52^{25}	2.44^{18}
	2.82^{25}	12.17^{16}	78.93^{8}	5.74^{20}	43.97^{22}	16.85^{8}	104.44^{29}	20.61^{6}	17.60^{16}	3.68^{26}
	3.69^{19}	12.45^{29}	79.98^{29}	5.82^{8}	44.01^{16}	17.43^{16}	105.05^{8}	20.66^{10}	19.11^{23}	4.00^{27}
	4.16^{16}	13.94^{18}	89.44^{18}	6.34^{29}	48.31^{21}	17.71^{19}	112.05^{19}	21.01^{15}	24.30^{20}	6.62^{24}
	4.42^{4}	16.76^{19}	98.60^{19}	7.06^{19}	57.81^{4}	19.52^{18}	118.38^{18}	28.48^{9}	28.54^{14}	12.52^{17}
B&W changes	1	1	1	1	3	1	3	5	4	5
-log prob	5.47	5.47	5.47	5.47	3.41	5.47	3.41	3.76	4.55	3.76
Color changes	7	8	8	8	7	10	8	8	9	9
-log prob	5.25	4.70	4.70	4.70	5.25	3.79	4.70	4.70	4.22	4.22
Mean	1.93	6.74	50.71	3.28	26.92	9.91	68.48	17.45	0.68	0.79
Mean	0.94	3.63	22.10	1.51	10.81	4.08	32.36	11.12	6.25	3.47
Mean	2.53	10.19	67.14	4.65	36.75	13.00	86.16	5.44	10.52	2.20
Fold change	2.70	2.80	3.04	3.08	3.40	3.19	2.66	3.21	15.43	4.41

Table 2B – Results for the B30 data set, 4-partition case

In this table, we show a subset of the top 200 genes with the fewest color changes for a 3-partition of the B30 CTCL data set. LIF, and CHD1 are regulatory kinases of interest that do not appear in the 3-parition list.

Gene	4012 Ha02 Hs.132834 hematopoietic protein 1 HEM1	3461 Ha02 Hs.86575 mitogen-activated protein kinase kinase kinase kinase 1 MAP4K1	3191 Ha02 Hs.79187 coxsackie virus and adenovirus receptor CXADR	3054 Ha02 Hs.100030 telomeric repeat binding factor 2 TERF2	2773 Ha02 Hs.2250 leukemia inhibitory factor (cholinergic differentiation factor) LIF	5643 Ha03 Hs.9950 Sec61 gamma SEC61G	3369 Ha02 Hs.22670 chromodomain helicase DNA binding protein 1 CHD1	2996 Ha02 Hs.4953 golgi autoantigen, golgin subfamily a, 3 GOLGA3	3767 Ha02 Hs.9222 estrogen receptor binding site associated, antigen, 9 EBAG9	2903 Ha02 Hs.83727 cleavage and polyadenylation specific factor 1, 160kD subunit CPSF1
Coloring	22.81^{14} 24.59^{15} 28.15^{17} 29.61^{13} 33.79^{28} 43.46^{6} 45.03^{7} 55.20^{12} 56.49^{9} 56.62^{11} 57.40^{2} 58.49^{10} 59.76^{5} 62.65^{27} 66.18^{21} 69.14^{23} 72.14^{24} 72.60^{26} 75.53^{20} 83.62^{1} 84.04^{4} 85.31^{3} 86.47^{0} 87.03^{25} 95.91^{22} 97.83^{16} 104.44^{29} 105.05^{8} 112.05^{19} 118.38^{18}	13.70^{13} 15.92^{17} 22.76^{15} 22.97^{14} 28.52^{28} 28.71^{12} 30.74^{7} 31.79^{6} 33.50^{11} 36.57^{10} 38.34^{2} 40.56^{9} 43.09^{5} 48.73^{21} 49.10^{27} 50.68^{23} 55.88^{3} 56.91^{24} 59.34^{26} 61.56^{22} 65.14^{25} 69.13^{16} 69.19^{0} 72.17^{4} 77.05^{20} 77.75^{1} 78.93^{8} 79.98^{29} 89.44^{18} 98.60^{19}	0.45^{15} 0.56^{14} 0.70^{17} 0.86^{28} 0.86^{13} 0.94^{7} 0.99^{10} 1.03^{11} 1.14^{9} 1.26^{26} 1.29^{20} 1.30^{0} 1.66^{1} 1.73^{5} 1.79^{6} 1.80^{8} 1.80^{12} 1.88^{24} 1.93^{18} 1.95^{25} 2.01^{29} 2.01^{23} 2.27^{2} 2.35^{3} 2.66^{16} 2.82^{21} 3.07^{27} 3.09^{22} 3.11^{19} 3.82^{4}	11.85^{13} 18.28^{17} 20.71^{15} 21.00^{14} 24.31^{12} 25.92^{23} 28.30^{28} 28.80^{6} 29.24^{7} 30.39^{10} 31.53^{5} 33.08^{11} 34.81^{2} 35.33^{21} 39.75^{27} 47.04^{9} 48.18^{3} 48.32^{4} 50.97^{1} 51.88^{24} 52.32^{22} 53.79^{0} 60.48^{8} 64.95^{26} 65.73^{25} 71.50^{20} 75.53^{29} 80.10^{16} 87.53^{18} 98.69^{19}	9.06^{17} 11.07^{13} 12.90^{14} 13.38^{15} 18.92^{28} 22.40^{7} 24.82^{11} 25.38^{10} 25.63^{9} 25.68^{12} 26.74^{2} 26.93^{5} 30.87^{6} 35.99^{0} 37.97^{20} 37.97^{1} 39.30^{23} 40.11^{21} 41.20^{24} 42.12^{26} 43.27^{18} 44.34^{27} 49.44^{25} 50.47^{3} 52.74^{8} 53.08^{4} 58.07^{22} 59.14^{29} 79.97^{19} 87.76^{16}	1.02^{26} 1.48^{18} 1.87^{29} 1.94^{19} 1.94^{16} 2.48^{22} 2.51^{20} 2.97^{25} 6.17^{27} 6.34^{14} 6.54^{28} 6.96^{1} 7.59^{8} 8.40^{21} 8.49^{4} 8.59^{17} 9.06^{15} 9.32^{23} 9.64^{5} 10.31^{7} 10.78^{6} 11.42^{3} 12.06^{11} 12.43^{0} 12.99^{2} 14.84^{13} 15.62^{24} 21.61^{12} 22.08^{10} 33.23^{9}	6.71^{13} 9.32^{14} 10.35^{15} 11.39^{17} 14.17^{28} 15.98^{6} 17.23^{7} 17.80^{5} 19.37^{27} 19.85^{12} 20.43^{23} 20.77^{2} 21.13^{11} 23.65^{10} 26.34^{24} 27.92^{26} 28.70^{29} 29.50^{22} 30.85^{21} 31.34^{0} 31.64^{4} 32.87^{3} 32.90^{9} 33.82^{20} 36.02^{25} 37.84^{8} 40.58^{16} 43.47^{18} 43.51^{19} 51.02^{1}	17.74^{15} 20.23^{13} 20.56^{14} 21.80^{17} 26.90^{28} 36.06^{7} 38.65^{5} 39.33^{12} 39.87^{6} 40.89^{10} 41.51^{11} 42.86^{0} 44.67^{1} 46.94^{23} 48.29^{2} 54.58^{9} 57.26^{21} 58.18^{27} 59.24^{24} 60.98^{20} 63.07^{25} 64.96^{26} 69.10^{3} 74.38^{4} 74.39^{18} 86.11^{29} 88.13^{22} 91.95^{8} 105.98^{16} 110.19^{19}	0.79^{15} 1.04^{13} 1.24^{14} 1.25^{17} 1.76^{28} 1.77^{12} 1.82^{5} 1.96^{6} 2.04^{7} 2.35^{23} 2.46^{10} 2.46^{11} 2.47^{2} 2.61^{21} 2.64^{27} 2.89^{24} 2.95^{26} 3.08^{22} 3.09^{3} 3.23^{9} 3.23^{4} 3.38^{25} 3.48^{0} 3.53^{1} 3.65^{20} 3.92^{18} 3.96^{8} 4.07^{29} 5.60^{19} 6.06^{16}	0.80^{15} 1.01^{13} 1.18^{14} 1.32^{17} 1.33^{28} 1.58^{5} 1.72^{3} 1.81^{6} 1.85^{12} 1.87^{23} 2.09^{10} 2.15^{2} 2.57^{27} 2.70^{21} 2.71^{7} 2.78^{9} 2.79^{4} 2.80^{11} 2.91^{25} 2.92^{20} 2.95^{24} 3.00^{18} 3.19^{1} 3.20^{0} 3.21^{26} 3.62^{22} 3.85^{8} 5.03^{29} 5.94^{16} 7.59^{19}
B&W changes	1	1	1	3	1	1	1	1	1	1
-log prob	4.85	4.85	4.85	2.87	4.85	6.47	4.85	4.85	4.85	4.85
Color changes	9	10	11	9	11	11	11	11	12	12
-log prob	6.03	5.50	5.05	6.03	5.05	5.05	5.05	5.05	4.66	4.66
Mean	68.48	50.71	1.74	41.39	34.42	13.17	27.85	51.90	2.81	2.56
Mean	65.06	46.82	2.32	35.44	38.13	12.23	23.37	52.19	2.45	2.39
Mean	27.79	20.77	0.68	20.03	13.06	9.07	10.39	21.45	1.22	1.13
Mean	95.47	75.03	2.16	74.54	57.22	2.03	35.44	81.73	4.09	4.28
Fold change	3.44	3.61	3.39	3.72	4.38	6.50	3.41	3.81	3.36	3.79