A Repulsive Clustering Algorithm for Gene Expression Data

Chyun-Shin Cheng
Department of Electronic Engineering
Tung Nan Institute of Technology
152, Sec. 3, PeiShen Rd., ShenKeng, Taipei 222
TAIWAN
cscheng@en.ntut.edu.tw

Shiuan-Sz Wang
Department of Electronic Engineering
National Taipei University of Technology
1,Sec. 3, Chung-Hsiao E. Rd., Taipei 106
TAIWAN
s9418035@ntut.edu.tw

Abstract: - Facing the development of microarray technology, clustering is currently a leading technique to gene expression data analysis. In this paper, we propose a novel algorithm called repulsive clustering, which is developed for the use of gene expression data analysis. One common goal to achieve on developing gene expression data clustering algorithms is to acquire a higher quality output. Our performance demonstration on several synthetic and real gene expression data sets show that the repulsive clustering algorithm, compared with some other well-known clustering algorithms, is capable of not only producing even higher quality output, but also easier to implement for immediate use on various situations.

Key-Words: - Microarray, Gene expression data, Clustering, Repulsive clustering algorithm.

1 Introduction

Microarray technology is a rapid method of sequencing and analyzing genes [1]. It is believed that this new technology holds the promise to have significant impact on the diagnosis, treatment, and prevention of disease [2].

Several algorithmic techniques were previously used in clustering gene expression data. Hierarchical clustering [3][4] is widely used because the tree structure is good for visual inspection. Some other algorithms such as k-means [5], self-organizing maps [6], CAST [7] and support vector machines [8], are successfully tested in many applications. Despite for unique features to some application environments such as hierarchical clustering for dendrogram, to gain a better cluster quality for a more sophisticated analysis is a common demand for an appropriate algorithm.

In this paper, we propose a novel algorithm called repulsive clustering for gene expression data analysis. Clusters are automatically formed in a way like magnets repel each other. Namely, repulsive clustering does not require initial specified numbers, or centroids, of clusters. Those will, on the contrary, be determined naturally from the intrinsic nature of data. The simulation result in this paper shows that repulsive clustering can perform dramatic quality boost over compared algorithms on analyzing different gene expression patterns.

The paper is organized as follows: In section 2 we describe the approach of the algorithm. Section 3 specifies the performance simulation environment in this paper. Section 4 contains the simulation results. Finally, section 5 is conclusions.

2 The repulsive clustering approach

2.1 The Idea of Repulsion and Attraction

The essence of repulsive clustering is to have data points repulse each other to different clusters if they can't "see" each other. Fig.1 shows the idea about repulsing. In Fig.1, there are 12 data points marked as G1 to G12 in a two-dimensional space. In order to cluster these data points, we define a regulable range to separate points and to discover groups. In Fig.1 for example, G1 is circled by a closed boundary and the radius of the circle is *w*. Assume the gray area outside this circle is the region that point G1 cannot distinguish from, thus we define that points outside this circle should be pushed into a cluster differs from G1's. This process is therefore called *repulsion* in the algorithm. Moreover, we define that points inside this circle (white area) are don't-cares to G1. Don't-care means that it is neither repulsed nor affiliated to G1 anyway.

After G1 finished pushing points outside, we switch the circle to another point in data set, such as G2, to engage another repulsion. Two simple rules for repulsion are that, only one point can do the repulsion at a time and each point should only do it once. The whole repulsion stage stops when all data points have had their own repulsion. Notice that there is no specified order on data points to follow in the repulsion stage. Therefore you can do random, or simply use the original top-down order from the data matrix as the order of repulsion for data points. The rest of the paper uses the latter way.

Different circle sizes produce different repulsion outcomes. A larger circle covers more not-so-similar

nodes and results in loose output (low in-group average similarity), while smaller one produces compact result (high in-group average similarity). For this reason, we define radius *w* to be a similarity threshold for repulsion.

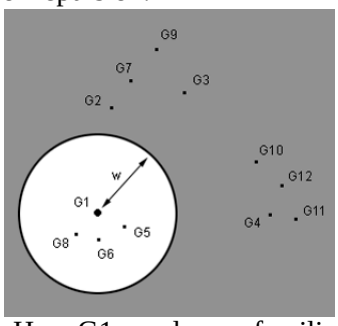

Fig.1: How G1 repulses unfamiliar points.

A visual example of a series of repulsion is demonstrated in Fig.2. We draw four circles from four steps S1 to S4 simultaneously in Fig.2, but only one circle is functioning at a time. The arrows indicate the switch from circles. Table 1 contains 5 steps of move. Column S0 shows that all points were initially assigned no cluster (zero), afterward repulsion stage began. S1 represents repulsion of G1 that pushed all points beside don't-cares, i.e., G5, G6, G8 and G1 itself, into cluster No.1. After the repulsion of G1 was complete, G2 had the circle in S2 and pushed G1, G4, G5, G6, G8, G10, G11 and G12 away into cluster No.2 including points with no cluster. S3 indicates no change in cluster number because no point outside G3's circle had the same cluster number to G3. G4 in S4 finally pushed G1, G5, G6 and G8 away into cluster No.3. We can see that 3 groups in S4 as we can visually tell are formed correctly. The repulsion stage will continue until G12 finished its repulsion.

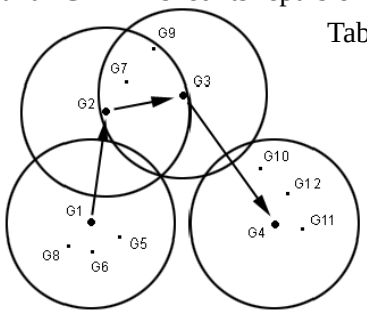

Table 1: Repulsion steps

	S0	S1	S2	S3	S4
G1	0	0	2	2	3
G2	0	1	1	1	1
G3	0	1	1	1	1
G4	0	1	2	2	2
G5	0	0	2	2	3
G6	0	0	2	2	3
G7	0	1	1	1	1
G8	0	0	2	2	3
G9	0	1	1	1	1
G10	0	1	2	2	2
G11	0	1	2	2	2
G12	0	1	2	2	2

Fig.2: A series of repulsion from G1 to G4.

After repulsion stage is complete, an *attraction* stage is applied to further enhance the cluster quality. Data point in attraction stage calculates the average similarity to existing groups discovered in first stage, and rejoins itself into a cluster with maximum average similarity.

The repulsion stage and attraction stage together formed the repulsive clustering algorithm.

2.2 The Algorithm

To ease the further explanations of the algorithm, we give 2 definitions for repulsive clustering.

- A *leading entity* is the point which is currently having the control to repulse its neighbors. There can be only one leading entity at a time.
- When a similarity measure in interval [*p*, *q*], where *p* and *q* are the lower and upper limit of the interval, is applied to a data set. We say that a point *j* is a *repulsive neighbor* to a leading entity *i* if $S(i, j) < w$, where $S(i, j) \in [p, q]$ is an i-to-j similarity, and *w* is the similarity threshold, $w \in [p, q]$.

The algorithm takes a pair of input $\langle S, w \rangle$. *S* is an n-by-n similarity matrix, and *w* is the adjustable similarity threshold. A latest cluster designation number is denoted by *m*, which is initially assigned No.1. C_i is the cluster number of point *i*. All points are initially set to no cluster in the first place, i.e., C_i=0. When repulsion stage begins, each point is in-turn being a leading entity to push its repulsive neighbors away by assigning $C_{repulsive\_neighbors}=m$ when $C_{repulsive\_neighbors}= C_{leading\_entity}$. In the other hand, some will be ignored if $C_{repulsive\_neighbors} \neq C_{leading\_entity}$. Variable *m* should be increased by 1 between every switch of leading entities when repulsion occurred.

With clusters generated in the first stage, most points have reached their place. To further enhance the accuracy, every point in attraction stage can calculate average similarities to all existing clusters and rejoins itself into cluster that has *MaxSim* (maximum similarity) if $MaxSim \geq w$. Otherwise it will be swept away into a new cluster. A leading entity will remain where it is if *MaxSim* falls into its place. Like the previous stage, attraction stops when all points have ran through it once.

Be aware that when a similarity measure *S* is used and a S_{ij} representing similarity from point *i* to point *j* is generated, the inequality $S_{ij}<w$ in the algorithm means that point *i* and *j* have an below-threshold similarity that point *j* will be repulsed by point *i*. If a dissimilarity or distance measure δ is used instead and a δ_{ij} is generated as dissimilarity from point *i* to point *j*, the inequality in the algorithm should, of course, be adjusted to $\delta_{ij} > w$. Otherwise you should convert the dissimilarity δ_{ij} into similarity S_{ij} by taking, for example, δ_{ij} =1- S_{ij} , in the first place. The rest of the paper uses similarity for representation. The time complexity of the algorithm is O(N^2), and a pseudo-code of the algorithm is given in Fig.3.

Repulsive Clustering Algorithm

Input: $\langle S,w\rangle$, S is an $n\times n$ similarity matrix, w is a similarity threshold.

Initialization:

$m\leftarrow 1$ *//Initialize current cluster designation*

$C(\ )\leftarrow 0$*// all elements have no cluster in the first place*

$E\leftarrow\{1,\ldots,n\}$ *//a set of all elements*

Begin:

Repulse:

for each ($e\in E$) **do**

  for each ($k\in E$, $k\neq e$) **do** *//check all elements*

    if ($S(e, k)<w$) **then** *//repulsive neighbors to e*

      Push $C(k)=C(e)$ or $C(k)=0$ into cluster m

        //push to new cluster

  increase m *//get new cluster designation*

Attract:

for each ($e\in E$) **do**

  pick an *EC* (existing cluster) with *MaxSim* (maximum average similarity) to e

  if ($MaxSim\geq w$) **then**

      //found a cluster that e should belong to

    $C(e)=EC$ *//let e break away and rejoin EC*

  else

    $C(e)=m$ *//let e break away into new cluster*

    increase m

return C

Operation complete.

Fig.3: The repulsive clustering algorithm.

3 Performance Simulation

3.1 Data Sets

3.1.1 Synthetic random data set

The synthetic random data set in our simulation provides randomly generated classes in a two-dimensional Euclidean space. This data set is used to evaluate the basic predictive power of algorithms. It is supposed to provide data with low noise and highly clarified classes. We generate a synthetic random data set with 5 classes, 1184 genes, which yields a 1184×1184 similarity matrix with average similarity of 0.638 using correlation coefficient in interval [0, 1].

3.1.2 The yeast cell cycle data set

We next evaluate the performance of algorithms by analyzing a time course data that has temporal gene expression patterns. We select a yeast microarray data from LBNL [9], retrieving a subset of 101 cell cycle regulated genes in 5-phase criterion: G1, S, S/G2, G2/M and M/G1, and this yields a 101×101 similarity matrix with average similarity of 0.806. We expect clustering results to approximate these five-phase criteria.

3.1.3 The human cancer cells line data sets

We next analyze the data studied by Scherf et al. [10] for 60 human cancer cell lines (NCI60) on gene-drug relationships by integrating large databases on gene expression and molecular pharmacology. Two data sets are selected in this paper for our performance simulation.

3.1.3.1 Cell-cell correlations on the basis of gene expression profiles

In the cell-cell correlations on the basis of gene expression profiles, there is a 1376×60 data matrix with 9 cancer classes in 60 samples. A 60×60 similarity matrix with average similarity of 0.815 is generated. We would like to see if algorithms could tell 9 cancer classes that showed similar patterns.

3.1.3.2 Cell-cell correlations on the basis of drug activity profiles

In the cell-cell correlations on the basis of drug activity profiles, the data formed a 1400×60 matrix with the same 9 classes. A 60×60 similarity matrix with average similarity of 0.891 is also generated in our simulation.

3.2 Evaluation of Cluster Quality

We apply both *external* and *internal* assessment of validity [11] to algorithms on all simulated data.

One external measure is *entropy* [12], which evaluates the accordance of a cluster set with known classes. Here we briefly describe it below. For each cluster C_j, we shall compute the data that it contains for each different class i so that p_{ij} is equal to the probability a randomly drawn data from cluster C_j to be of class i. The entropy H_j of each cluster C_j is calculated as:

$$H_j = -\sum_i p_{ij} \log_2(p_{ij}) \qquad (1)$$

For gene expression data analysis, we introduce a modified entropy M_j for external gene data assessment by adding a variable $p_{(not\ i)j}$ equals to the probability a randomly drown data from cluster C_j to not be of class i, and a $p_{i(not\ j)}$ equals to the probability a randomly drown data from clusters other than C_j to be of class i.The former variable detects if multiple classes exist in single cluster, and the latter one ensures that class i does not spread across multiple clusters, thus M_j will be:

$$M_j = -\sum_i (p_{ij} \log_2(p_{ij}) - p_{(not\ i)\ j} - p_{i(not\ j)}) \qquad (2)$$

Modified entropy is 0 only when data points are exactly separated as their class. Finally, we weigh the modified entropy by the size of each cluster. The weighted modified entropy M_w is:

$$M_w = \sum_j \frac{n_j \times M_j}{n} \quad (3)$$

Where n_j is the size of cluster j, and n is the total number of data in that data set. M_w is used for external assessment in this paper.

One internal measure is *Adjusted Figure Of Merit* (Adjusted FOM) [13]. Adjusted FOM is calculated as follows. A typical gene expression data set contains measurements of expression levels of n genes measured under m experimental conditions. Suppose there are k clusters, C_1, C_2,…,C_k. Let $R(g, e)$ be the expression level of gene g under condition e in the row data matrix. Let $\mu C_i(e)$ be the average expression level in condition e of genes in cluster C_i. The 2-norm figure of merit, *Adjusted FOM*(k) for k clusters under condition e is

$$\text{Adjusted FOM}(k) = \frac{\sum_{e=1}^{m} \sqrt{\frac{1}{n} \sum_{i=1}^{k} \sum_{x \in C_i} (R(x,e) - \mu c_i(e))^2}}{\sqrt{\frac{n-k}{n}}} \quad (4)$$

4 Simulation results

In this section, we demonstrate how repulsive clustering performed under both internal and external assessments on synthetic and real gene expression data. The w for repulsive clustering (RC) in the simulation runs from 0 to 1 (step 0.001) to produce different number of clusters. Hierarchical clustering approaches like single-link (SL), complete-link (CL) and average- link (AL), and heuristic-based clustering approach like CAST, are selected for co-evaluation in our simulation.

4.1 The synthetic random data set

Fig.5 shows the adjusted FOM of the synthetic random data set with 5 classes over a range of different number of clusters. The lower the curve, the better the internal quality for specified data. From the output of respective algorithms, we compare the difference in internal quality at output that is equal to known classes. In this synthetic data set, RC (square) and CAST (line) both perform almost the same internal quality at 5 clusters output while AL (diamond), SL (circle) and CL (down triangle) fall behind.

Fig.6 shows modified entropy for the synthetic random data set with 5 classes over a range of different number of clusters. This graph shows if each class is gathered respectively (lower curve), scattered (higher curve), or mixed (higher curve) in different cluster outputs. The lower the curve, the better the external quality. The lowest point of the curve represents the best cluster output an algorithm can make. RC and CAST in Fig.6 both produce lowest point at 5 clusters output that matches 5 classes' criteria.

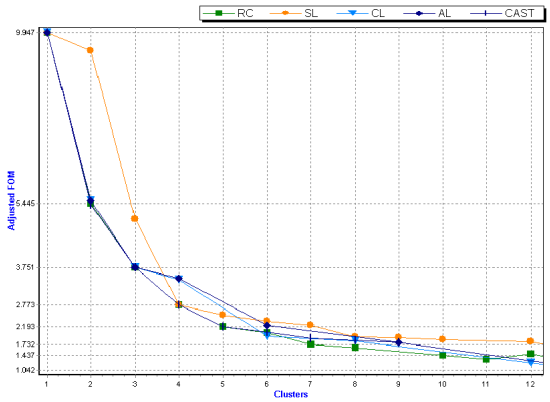

Fig.5: Adjusted FOM for the synthetic data.

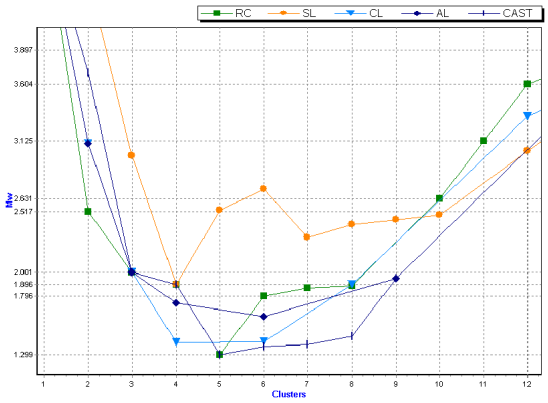

Fig.6: Modified entropy for the synthetic data.

4.2 The yeast cell cycle data

Fig.7 shows the adjusted FOM for the yeast cell cycle data with 5 classes over a range of different number of clusters. RC here achieves highest internal quality at 5 clusters output. It also shows comparable overall internal quality against other algorithms. CAST has lower overall internal quality in compared to RC, but is still out performance over hierarchical clustering family quite a range.

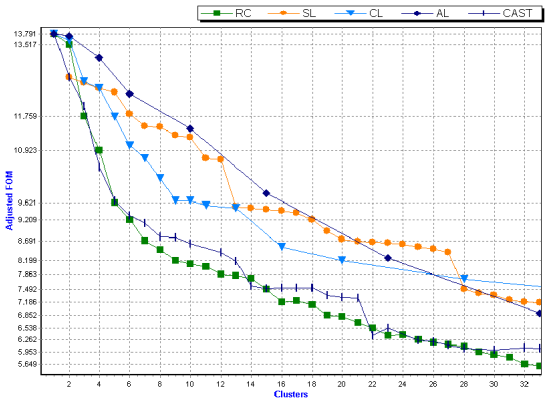

Fig.7: Adjusted FOM for the yeast cell cycle data

Fig.8 shows modified entropy for the yeast cell cycle data with 5 classes over a range of different number of clusters. The yeast cell cycle data here is quite noisy except the S-phase. RC has best external quality at 12 clusters output, which is closest to 5 classes, while CL is at 16, AL at 23, CAST at 28 and SL at 41.

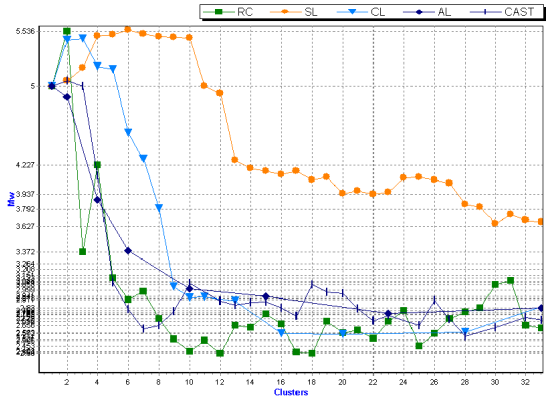

Fig.8: Modified entropy for the yeast cell cycle data

4.3 The human cancer cells line data sets

4.3.1 Cell-cell correlations on the basis of gene expression profiles (human cancer data (1))

Fig.9 shows the adjusted FOM for the cell-cell human cancer correlations on the basis of gene expression profiles with 9 classes over a range of different number of clusters. RC again, achieves best internal cluster quality at 9 clusters output in compared with other algorithms.

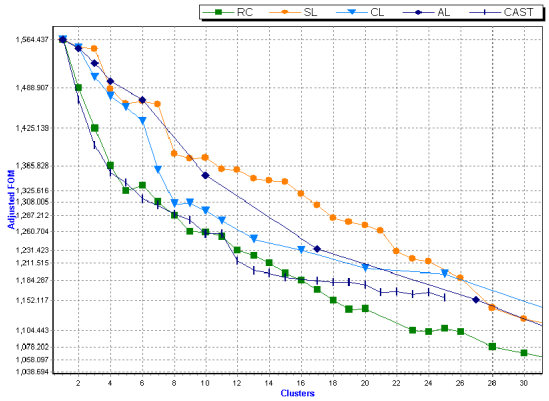

Fig.9: Adjusted FOM for human cancer data (1)

Fig.10 shows modified entropy for the cell-cell human cancer correlations on the basis of gene expression profiles with 9 classes over a range of different number of clusters. CL has its lowest point at 7 clusters output, which is closer to 9 classes, is closer than RC (lowest at 12) by 1, but the actual quality values at either 7 or 12 outputs are far behind RC. RC in Fig.10 can maintain lowest overall modified entropy because it is able to handle outliers carefully without interfering main classes.

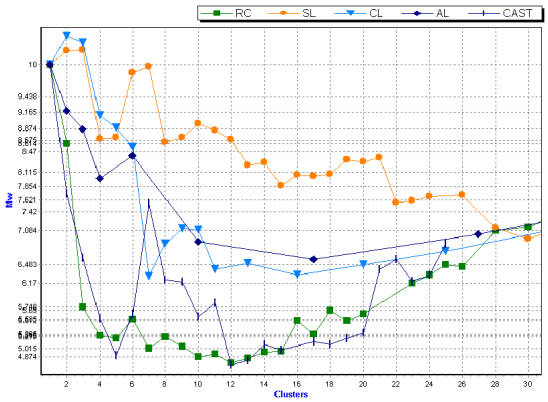

Fig.10: Modified entropy for human cancer data (1)

4.3.2 Cell-cell correlations on the basis of drug activity profiles (human cancer data (2))

Fig.11 shows adjusted FOM for the cell-cell human cancer correlations data on the basis of drug activity profiles with 9 classes over a range of different number of clusters. In this high average similarity data set, RC takes the lead in overall internal quality and has the highest quality at 9 clusters output. SL this time tends to perform better then it was and its internal quality at all range of outputs is similar to CAST.

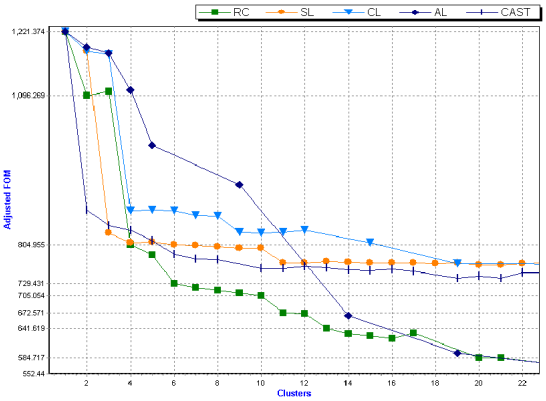

Fig.11: Adjusted FOM for human cancer data (2)

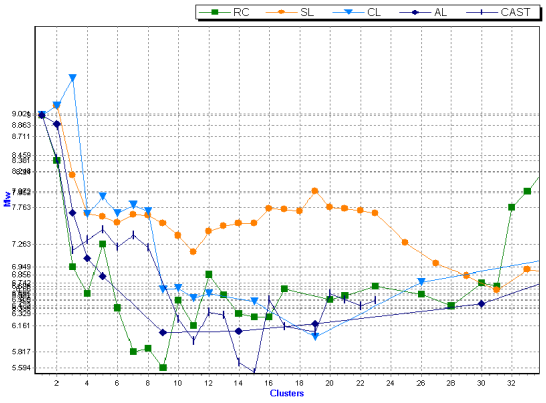

Fig.12: Modified entropy for human cancer data (2)

Fig.12 shows Modified entropy for the cell-cell

human cancer correlations data on the basis of drug activity profiles with 9 classes over a range of different number of clusters. RC produces highest quality at exact 9 clusters output that matches known classes.

4.4 A Performance Summary

Table 2 shows the internal quality ranking on simulated data sets at known classes. We can see that the performance of repulsive clustering (RC) is very comparable under internal quality review.

Table 2: Summary of internal quality

data set	**# of classes**	**Adjusted FOM at # of classes**
synthetic random data	5	RC=CAST=SL>CL>AL
cell cycle data	5	RC>CAST>CL>SL>AL
human cancer data(1)	9	RC>CAST>CL>SL>AL
human cancer data(2)	9	RC>CAST>SL>CL>AL

Table 3 shows the ranking of accuracy between the highest external quality output and real classes. Though CL looks closer to real classes in the 3rd data set, its external quality at 7 is far worse than RC (see fig.10). The external quality of AL at 9 in the 4th data set is also worse than RC at 9 (see Fig.12). We can conclude from simulation results that RC has the best overall performance to distinguish real classes from different data sets.

Table 3: Summary of external quality

data set	**# of classes**	**accuracy of the Modified Entropy to # of classes**
synthetic random data	5	RC(5)=CAST(5)>CL(4) =AL(6)=SL(4)
cell cycle data	5	RC(12)>CL(16)>AL(23)>CAST(28)>SL(41)
human cancer data(1)	9	CL(7)>RC(12)=CAST(12)>AL(17)>SL(30)
human cancer data(2)	9	RC(9)=AL(9)>SL(11)>CAST(15)>CL(19)

5 Conclusions

In this paper, we propose a repulsive clustering algorithm for gene expression data analysis. Repulsive clustering automatically decides the number of clusters according to the nature of data. The boundaries of clusters are automatically spanned as well. These features are important to avoid initially improper assumptions. The algorithm has been implemented and tested on synthetic and real biological data sets in our simulation program. The result demonstrates good performance in all runs. Another to note in simulation is that repulsive clustering has impressive power on distinguishing outliers. Outliers are not specially marked, but are separated in independent clusters respectively.

Finally, since repulsive clustering can deal with any n-by-n similarity matrix generated from different data sets, it could be used in clustering applications from different fields.

References:

[1] KK Jain, Applications of biochip and microarray systems in pharmacogenomics, *Pharmacogenomics*, 2000, Vol.1, pp. 289-307.

[2] Arun Jagota, Microarray Data Analysis and Visualization, UCSC, 2001.

[3] E.M. Voorhees, Implementing agglomerative hierarchical clustering algorithms for use in document retrieval, *Information Proceeding & Management*, 1986, 22:465-476.

[4] J. B. McQueen, Some Methods of Classification and Analysis of Multivariate Observations, *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967, pp. 281-297.

[5] Tavazoie, S., Hughes, J. D., Campbell, M. J., Cho, R. J. and Church, G. M., Systematic determination of genetic network architecture. *Nature Genetics*, 1999, Vol. 22, 281–285.

[6] Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E. S. and Golub, T. R., Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation, *Proceedings of the National Academy of Science USA*, 1999, Vol. 96, pp. 2907–2912.

[7] Amir Ben-Dor and Zohar Yakhini, Clustering Gene Expression Patterns, *Proceedings of the 3rd Annual International Conference on Computational Molecular Biology*, 1999.

[8] Brown et al., Knowledge-based analysis of microarray gene expression data using support vector machines, *Proceedings of the National Academy of Science USA*, 97, pp. 262–267.

[9] Lawrence Berkeley National Lab (LBNL), Michael Eisen's Lab (http://rana.lbl.gov/).

[10] Scherf et al., A gene expression database for the molecular pharmacology of cancer, *Nature Genetics*, march 2000, Vol. 24.

[11] A.K. Jain, M.N. Murty and P.J. Flynn. Data Clustering: A Review. ACM Computing Surveys, September 1999, Vol. 31, No. 3.

[12] C. E. Shannon, A Mathematical Theory of Communication, *The Bell System Technical Journal*, July, October, 1948, Vol. 27, pp. 379–423, 623–656.

[13] Yeung, K. Y., Haynor, D. R., Validating Clustering for Gene Expression Data, *Bioinformatics*, 2001, Vol. 17, pp. 309-318.