Microarray Data Clustering Using Particle Swarm Optimization K-means Algorithm

Youping Deng [1], Dheeraj Kayarat[2], Mohamed O. Elasri[1], Susan J. Brown[2]

[1] Department of Biological Sciences, The University of Southern Mississippi, Hattiesburg, MS 39406, USA
Email: Youping.Deng@usm.edu , Phone: 601-266-6678. Fax: 601-266-5797
[2] Division of Biology, Kansas State University, Manhattan, Kansas 66506, USA

Abstract

Gene expression data clustering provides a powerful tool for studying the interrelation between various genes. In this paper we present a K-means clustering algorithm based upon particle swarm optimization (PSO K-means) for microarray data clustering. This algorithm discovers clusters in microarray data without prior knowledge of feasible cluster numbers or complex parameter settings, which are required by other clustering methods. PSO K-means is able to bring out the latent structure of microarray data sets. A Significant reduction in within cluster error was obtained using this procedure, without considerable reduction in inter-cluster dissimilarity.

Keywords: particle swarm optimization, K-means, clustering, microarray, gene expression.

Availability:
http://www.bioinformatics.ksu.edu/psok-means.html

1. Introduction

K-means is the simplest and most popular among the various iterative and hill climbing clustering algorithms. But it is prone to convergence near local optima. Since stochastic optimization approaches are good in preventing convergence to local optimum solutions, these methods could be used to find global optima. Stochastic approaches used in clustering include those based on simulated annealing, Genetic algorithms [1], evolutionary programming and Ant algorithms. In this paper we provide a clustering algorithm based on Particle Swarm Optimization and compare its performance to other algorithms. This algorithm considers clustering as an optimization problem with the objective being to reduce the total within cluster error.

Particle Swarm Optimization (PSO) model was first described in 1995 [2,3,4] as a new method for function optimization. In PSO, particles flow in virtual space looking for feasible solutions, which in our case are matching particles. Unlike older partitional clustering algorithms where the number of partitions have to be specified prior to running the algorithm, PSO algorithm is capable of forming clusters on its own depending on the level of similarity or dissimilarity between the data vectors (particles). The PSO K-means algorithm is essentially different from SOM/PSO algorithm [5] in the sense that the later uses PSO to optimize the weights of neural network once it has been trained with training data sets.

1.1. Initialization

This algorithm treats each data vector or particle as an object or entity with its own functions and characteristics. Each particle stores essential information about itself like mean, standard deviation, difference vector, X and Y coordinates, force components, cluster membership etc., which prompts them to interact dynamically to the changing environment.

The initial position of each particle is chosen using its mean and standard deviation with each forming variables for one of the two coordinates. This choice of initial position is based on three assumptions 1) particles will be able to find more partners with similar characteristics in a region of similar mean and standard deviation than in the case of random initialization 2) closer initialization position will reduce the search space for particles 3) closer initialization will help in the formation of early nuclei of cluster aiding in faster convergence.

1.2. Forces and Velocity

Particles in PSO are guided in virtual space by various attractive forces. The two kinds of forces at play are the force of attraction between two particles due to closeness of position and the force of attraction due to similarity of the signal corresponding to the particle.

The force on the particle V_a due to the particle V_i in X direction is given by the following equations.

Table 1. Results of internal validation for K-means and PSO K-means using square error method. PSO K-means shows better performance upon K-means algorithm

Data	K number used for k-means	Major clusters formed by PSO	Within cluster square error per gene. PSO K-means	Within cluster error square per gene. K-means
Eisen	34	49	13.778	26.838
Cho data	30	51	16.196	21.563
Fig2data	10	14	12.714	58.69
synthetic	12	14	1.032	3.451

Table 2. Results obtained for internal validation using same number of clusters in PSO K-means and K-means algorithm

Data	K number used for k-means	Major clusters formed by PSO	Within cluster square error per gene. PSO K-means	Within cluster error square per gene. K-means
Eisen	49	49	13.778	27.035
Cho data	51	51	16.196	17.134
Fig2data	14	14	12.714	56.211
synthetic	14	14	1.032	3.321

Table 3. Results of Silhouette index testing for the data sets using K-means and PSO k-means

Data set	K number used for k-means	Major clusters formed by PSO	K-means	PSO K-means
Fig2data	34	49	0.084122	0.142674
Cho data	30	51	0.13286	0.15433
Eisen	10	14	0.19918	0.19972
Synthetic	12	14	0.1857	0.19283

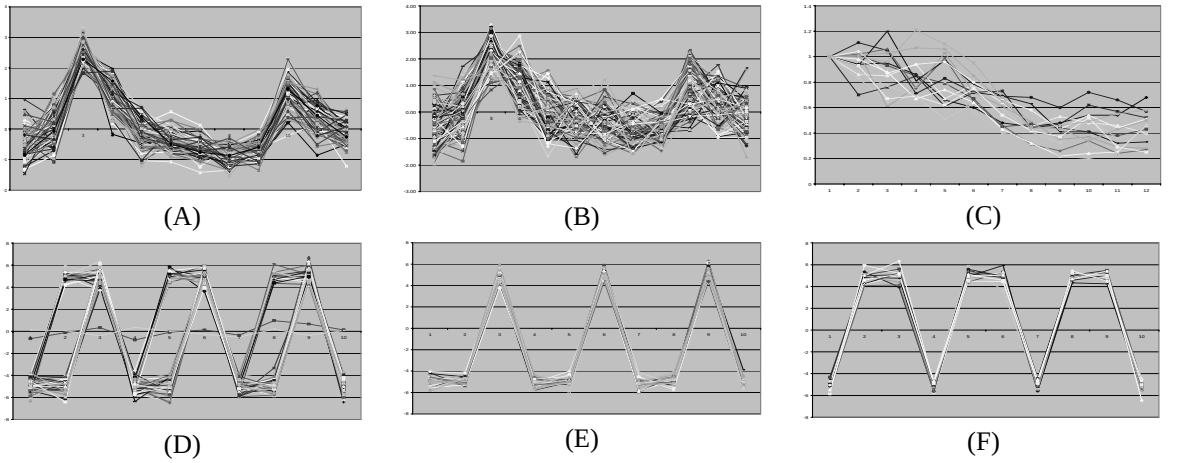

Fig. 1: Cluster results comparison (A) cluster obtained using PSO K-means during the test on Chow data. The graph shows visually a very close cluster formation. (B) Similar cluster obtained from K-means algorithm from Chow data. Clearly the cluster contains more noise than PSO K-means. (C)Cluster obtained in fig2Data using PSO K-means. The algorithm was able to bring out the transient trends in the input data. (D) Cluster obtained on synthetic data using K-means. Dissimilar genes having comparable means have been forced in to same cluster, resulting in high values of within cluster square error. (E,F) Clusters obtained from PSO K-means algorithm in Synthetic data. The same set of genes in Fig 1.D has been divided in to two clusters in PSO k-means. The clustering resulted in closely fitting genes with significantly low within cluster error. (G) Cluster formed from Eisen data Using PSO K-means.

$$FDx_{a,i} = \min[Q, (1/d_{a,i})\cos\Phi] \quad \ldots (1)$$

Where $FDx_{a,i}$ is the force on V_a by V_i in X direction, $d_{a,i}$ is distance between the two particles, Φ is the angle with X-axis and Q is a constant. The total force on the particle due to distance factor will be $FDX_{(a)}$ such that

$$FDX_{(a)} = \sum_{\substack{i=1 \\ i \neq a}}^{n} FDx_{(a,i)} \quad \ldots\ldots\ldots (2)$$

Where n is the number of particles in the system. In the same manner force in Y direction is calculated

Particles also experience force due to the similarity of genes. A low value of squared error of corresponding values of data vector is considered as a

measure of similarity. For a signal vector V1 ($a_{1,1}, a_{2,1,\ldots\ldots\ldots\ldots}, a_{m,1}$) the square error term with respect to the particle V_i($a_{1,i}, a_{2,i,\ldots\ldots\ldots\ldots}, a_{m,i}$) is calculated as

$$\varepsilon_1 = \sum_{j=1}^{m} (x_{j,1} - x_{j,i},)^2$$

ε_1 value is computed for each particle in the environment.

The force is inversely proportional to the squared error term. Force in X direction is given by

$$FX_{a,i} = \min[C, (1/\varepsilon_i) * A * \cos\Phi] \quad \ldots.(3)$$

Where C is a large constant corresponding to the value of force,when the value of ε_i equal to zero, A is a constant.

Force in Y direction is given by

$$FY_{a,i} = \min[C, (1/\varepsilon_i) * A * \sin\Phi] \quad \ldots.(4)$$

The total force on the particle is calculated by finding the vector sum of the force induced by the collective attraction of all the particles.

$FTXa = \sum_{i=1}^{n} FX_{a,i}$ where FTX_a is the total force experienced by the particle V_a in X direction. The total force on the particle due to all the forces combined is calculated as

$$FFx = A1 * FDX + A2 * FTXa \quad \ldots\ldots (5)$$

$$FFy = B1 * FDY + B2 * FTYa \quad \ldots\ldots(6)$$

FFx, FFy are the total force in X and Y-axis respectively.

Both the forces in play and level of randomness influence the velocity and the position of the particles.

$$X_{i,t} = X_{i,t-1} + FFx + random() * C1 \quad \ldots. (7)$$

Where $X_{i,t}$ is the position of particle i at time t, C1<1 is a constant. Randomness increases the search space of the particle but may delay the time for convergence, so C1 is given a very low value.

1.3. Clustering

Initial clusters or nuclei clusters are formed by the combination of two particles. When the distance between two particles flying in space gets reduced to a predefined value the particles check for similarity. If the square error is low the particles tend to combine together to form a cluster, under a threshold value. Second level cluster formation occurs when single particles get added up to the nuclei cluster or a well-developed cluster.

$$\vec{V}c = \begin{bmatrix} \sum_{i=1}^{p} v_{i,1} \\ \sum_{i=1}^{p} v_{i,2} \\ . \\ . \\ \sum_{i=1}^{p} v_{i,j} \end{bmatrix} * 1/n \qquad \ldots\ldots\ldots\ldots (8)$$

$\vec{V}_C$ is the mean value vector of cluster C, $\sum_{i=1}^{p} v_{i,j}$ is the sum of j terms in the particles of cluster C.

Mega clusters are formed when one clusters combines with another. If the mean value of cluster having higher cluster ID is within 1**sigma* limit of the other cluster, the two clusters are merged together. *Sigma* is the standard deviation of distances of the particles from the mean value.

$$\vec{V}_j = \sum_{i=1}^{n} v_{i,j} / n \qquad \ldots\ldots\ldots (9)$$

Where $\vec{V}_j$ is the mean value of the cluster at data point j $v_{i,j}$ is the value of i[th] particle at data location j. square error associated with each particle is calculated as follows.

$$\varepsilon_i = \sum_{j=1}^{d} (\vec{V}_j - v_{i,j})^2 \qquad \ldots\ldots\ldots(10)$$

The particles in the cluster are made to experience an inertia force [6] proportional to the number of particles in the cluster, which reduces the velocity of the cluster. This is done so that clusters will tend to form anchor point and searching is done by the particle to find a suitable cluster not the cluster moving in search of particles.

To eliminate the chance of nuclei clusters conducting self-cleansing, a limit could be set on the minimum cluster size required to start this operation. The numbers of nuclei clusters or the second level clusters that fail to merge with other clusters to form mega clusters are reduced by including a one step K-means operation in the end of the algorithm. The particles in clusters with number of entities less than a limit is compared with the bigger clusters and merged with the cluster which results in least increase in square error.

PSO algorithm demands a few parameters to evaluate the given data set. The parameters being 1) the maximum permissible square error between the two genes to form nuclei cluster E_{max} 2) the force coefficients for distance and similarity 3) small cluster size limit.

2. Implementation

We conducted experiments using particle swarm optimization K-means algorithm on three data sets. Viz., 1) Supplementary material for Eisen *et al*. [7], 2) Yeast data *Cho_Data_2070* [8], 3) *fig2data* introduced in 1999 [9] and 4) a synthetic data set [10].

The clusters obtained using POS K-means algorithm were found to have reduced within cluster error and good fit compared to those formed by K-means algorithm (Fig 1).

The result of POS K-means clustering and their statistical validity has also been tested in comparison with K-means algorithm using Euclidian distance. Reduction in within cluster error $\varepsilon_i = \sum_{j=1}^{d} (\vec{V}_j - v_{i,j})^2$ is taken as the test criteria.

It was found that the within cluster error associated with PSO K-means clusters are much less compared to those formed by K-means (Table 1). The results obtained when the K number used in K-means algorithm is set to the same value as the number of main clusters formed using PSO K-means algorithm is summarized in Table 2.

A second cluster validation measure called Silhouette index is also used for cluster validation [11]. It is observed that though the number of clusters formed using PSO K-means is higher than the K-means the resultant reduction in Silhouette index value for PSO K-means is not significant (Table 3). This is a noteworthy achievement considering the fact that the small clusters formed could cause significant reduction in the Silhouette index value.

3. Discussion and conclusion

Particle swarm optimization provides a novel approach to clustering without the need of any prior specification of cluster number or very complex parameter setting. Self- cleansing strategy provides the cluster ability to reduce the within cluster error and remove dissimilar particles from them without compromising the versatility and generic nature of the algorithm. Extremely deviant values are prevented from falling in to good clusters thus increasing the closeness of cluster formation. The PSO K-means algorithm in general performed much better than the K-means algorithm as seen in the results of validation procedure (Table 1, 2, 3).

One shortcoming of the PSO algorithm is the formation of number of small clusters which was overcome by introducing a one setp K-means operator that forces the small clusters into the bigger ones and finishes the clustering operation. But this goes against the spirit of the algorithm and the user will have to specify the minimum number below which the cluster will be considered as a small cluster. Another approach could be to introduce a second phase of PSO in which the parameters are reinitialized and positions reallocated. New force functions could be defined which will increase the attractive force between larger and smaller clusters and reduces the force between larger clusters. Particle swarm optimization provides number of future applications and improvements in the domain of clustering.

4. Acknowledgements

This work was supported by Dean's Research Initiative award of the University of Southern Mississippi to Youping Deng, the K-INBRE Bioinformatics Core, NIH grant number P20 RR016475 and the Mississippi Functional Genomics Network (DHHS/NIH/NCRR Grant# 2P20RR016476-04). The authors sincerely thank Vanka Phaneendra, Vijayaraj Nagarajan, Drs. William Hsu and Sanjoy Das for critical review of the manuscript.

5. References

[1] K. Krishna, and Murthi, "Genetic k-means algorithm", IEEE Transactions on systems, man, and cybernetics-Part B, 29, 433-4, 1999.

[2] R.C. Eberhart, and J. Kennedy, "New optimizer using Particle Swarm theory", In Procedings of Sixth International Symposium on Micro Machine and HumanScience(Nagoya, Japan), IEEE Service center, Piscacataway, NJ, pp. 39-43, 1995.

[3] R.C. Eberhart, R.W. Dobbins, and P. Simpson, Computational Intelligence PC tools. Boston Academic Press, 1996.

[4] J. Kennedy, "The Particle Swarm: social adaptation of knowledge", In Proceedings of IEEE International Conference on Evolutionary Computation, Indianapolis, Indiana,IEEE Service Center, Piscataway, NJ, pp. 303-308, 1997.

[5] X. Xiao, E.R. Dow, R Eberhart, Z. B. Miled and R.J. Oppelt, "Gene clustering using self-organizing maps and particle swarm optimization", Proceedings of Second IEEE International Workshop on High Performance Computational Biology, Nice, France, 2003.

[6] R.C. Eberhart, and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization", Proceedings of the 2000 Congress on Evolutionary Computation, 1, San Diego, pp. 84 -88, 2000.

[7] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Bostein, "Cluster analysis and display of genome wide expression patterns", Proc. Natl. Acad. Sci. , 95, 14 863-14 868, 1998.

[8] R.J. Cho, M.J. Cambell, E.A. WInzer, L. Stenmetz, A. Conway, L. Wodicka, T.G. Wolfsberg, A.E. Garielian, D. Landman, D.J. Lockhart, and R.W. Davis, “A Genome wide transcriptional analysis of the mitotic cell cycle”, Molecular Cell, 2, 65-73, 1998.

[9] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Sculer, T. Moore, J.C.F. Lee, J.M. Trent, L.M. Straudt, Jr.J. Hudson, and MS. Bogosk et al., “The transcriptional program in the response of human fibroblast to serum”, Science, 238, 83-87, 1999.

[10] D. Dembele, and P. Kastner, “Fuzzy C-means method for clustering microarray data”, Bioinformatics, 19, 973-980, 2003.

[11] N. Bolshakova, and F. Azuaje, “Cluster validation techniques for genome expression data”, Signal Processing, 83, 825 – 833, 2002