Data Clustering Using Almost Parameter Free Differential Evolution Technique

Sai Hanuman A
Associate Professor of CSE, HOD of MCA GRIET Hyd A P India

Dr Vinaya Babu A
Professor of CSE Director of Admissions, JNTU Hyd, AP, India

Dr Govardhan A
Professor of CSE, Principal, JNTUH, Jagitial, A P, India

Dr S C Satapathy
Professor and Head Dept of CSE, ANITS, Vizag A P, India

ABSTRACT

The paper presents a comparative analysis of data clustering by Particle swarm optimization (PSO) and differential evolution (DE) techniques. It is clearly reveled from the simulation results that almost parameter free optimization technique such as Differential evolution could provide better performance compared to PSO where in many parameters are to be tuned. To exhibit the numerical optimizing capability of DE we have demonstrated the capability of this by optimizing few benchmark functions.

Keywords

Data Clustering, PSO, Differential evolution, Function Optimization

1. INTRODUCTION

Cluster analysis seeks to divide a set of objects into a small number of relatively homogeneous groups on the basis of their similarity over N variables. Cluster analysis can be viewed either as a means of summarizing a data set or as a means of constructing a topology [12]. Patterns within a valid cluster are more similar to each other than to a pattern belonging to a different cluster. Clustering is useful in several exploratory pattern-analysis, grouping, decision-making, data mining, document retrieval, image segmentation and pattern classification [8].

Data clustering can be hierarchical or partitional. Hierarchical clustering algorithms can be described in terms of trees. According to Hartigan (1967, p. 1140), "a tree may be regarded as a hierarchical grouping structure, in which the objects are grouped into a set of clusters, these clusters are again grouped into a set of clusters of clusters, and so on." Hartigan regards objects, clusters, and clusters of clusters equally as nodes [12]. Hierarchical clustering algorithms can be agglomerative (bottom-up) or divisive (top-down). Agglomerative algorithms begin with each element as a separate cluster and merge them into larger clusters. Divisive algorithms begin with the whole set of data objects and proceed to divide it into successively smaller clusters [9].

Partitional clustering algorithms relocate instances by moving them from one cluster to another, starting from the initial partitioning. Such method requires the number of clusters to be preset by the user. The simplest and the most commonly used portioning algorithm is K-Means, where K denotes the number of clusters. The reasons for the algorithmic popularity is it's ease of interpretation, simplicity of implementation , speed of convergence and adaptability to sparse data(Dhillon and Modha, 2001)[13]. The disadvantages of this algorithm lies in the fact that the accuracy of the algorithm completely depends on the initial selection of centroids and it more often stuck up at local optima for complex multimodal problems.

Inspired by the process of Darwinian evolution (Back and Weigend, 1998; Eiben and Smith, 2003), a new paradigm called as evolutionary computation came which consists of stochastic search algorithms. Because of its robust, adaptive search methods for performing global search, EAs are applied in the field of Data Mining. In the mid 1990s Eberhart and Kennedy came with a new evolutionary based optimization technique by emulating the collaborative behavior of bird flocks and fish schools and called it Particle Swarm Optimization. Around the same time, Price and Storn proposed a new algorithm based on a differential operator, and called it Differential Evolution.

In this paper we try to bring out the performance variations among the two evolutionary based algorithms PSO and DE by applying them to optimize four standard benchmark functions and also for clustering three real-world data sets. The remainder of this paper is organized as follows. In Section 2, the fundamentals of PSO are described. In Section 3, fundamentals of DE are discussed. Function optimization using PSO and DE is discussed in Section 4. In Section 5, frame work for data clustering is discussed. Experimental setup and results are discussed in Section 6. Conclusion is discussed in Section 7.

2. FUNDAMENTALS OF PSO

Particle swarm optimization (PSO) is an evolutionary computation technique developed by Kennedy and Eberhart (1995) [1]. They were essentially aimed at producing computational intelligence by exploiting simple analogs of social interaction rather than purely individual cognitive abilities. The first simulations (Kennedy & Eberhart, 1995) were influenced by Heppner and Grenander's work (1990), and involved analogs of bird flocks searching for corn. These soon developed into a powerful optimization method, PSO [2].

The key difference between PSO and evolving populations is the way in which new samples are generated. In GAs, new samples are produced by some recombination of selected parent solutions which may then go on to replace members of population. In

PSO, new samples are generated by perturbation of existing solutions. Later approach can lead to issues of stability [3].

PSO is a population-based search algorithm and is initialized with a population of random solutions called particles [4].In this, each individual is treated as a volume less particle in a d-dimensional search space.

- The ith particle is represented as $x_i = (x_{i1}, x_{i2},...x_{id})$.
- The best previous position (the position giving the best fitness value) of the ith particle is recorded and represented as $P_i=(P_{i1}, P_{i2},...P_{id})$.
- The index of the best particle among all the particles in the population is represented by the symbol g.
- The rate of position change (velocity) for particle i is represented as $V_i=(V_{i1}, V_{i2}...V_{id})$[5].
- The particle is manipulated according to the following equation:

$$v_{id} = v_{id} * w + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \rightarrow (1)$$

$$x_{id} = x_{id} + v_{id} \rightarrow (2)$$

Where w is the inertia of weight, c_1 and c_2 are two positive constants, and rand( ) is a random function in the range [0,1] [11].

The original process for implementing the global version of PSO includes:

1. initializing a population of particles with random positions and velocities on d dimensions in the problem space,
2. evaluating the desired optimization fitness function in d variables, for each particle,
3. comparing particle's fitness evaluation with particle's pbest. If current value is better than pbest, then set pbest value equal to the current value, and the pbest location equal to the current location in d-dimensional space.
4. comparing fitness evaluation with the population's overall previous best. If the current value is better than gbest, then reset gbest to the current particle's value.
5. changing the velocity and position of the particle according to the equations(1) and(2), respectively:
6. loop to step 2 until the criterion is met, usually a sufficiently good fitness or a maximum number of iterations.

3. FUNDAMENTALS OF DE

Differential Evolution (DE) is a parallel direct search method developed by Storn and Price in 1997 which is a population-based global optimization algorithm .It uses a real-coded representation [6]. This approach for numerical optimization is simple to implement and requires little or no parameter tuning, but gives a remarkable performance. Like all other evolutionary algorithms, the initial population is chosen randomly.

Classical DE

Like all other evolutionary algorithms, DE method also consists of three basic steps:

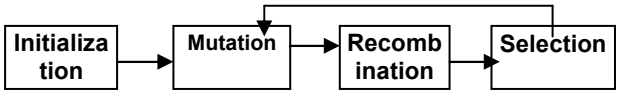
(i) Generation of population with N individuals in the d-dimensional space, randomly distributed over the entire search domain

$$\overrightarrow{X_i}(t) = [x_{i,1}(t), x_{i,2}(t), x_{i,3}(t)....x_{iD}(t)]$$

where t=0,1,2,….t,t+1

(ii) Replacement of this current population by a better fit new population, and

(iii) Repetition of this replacement until satisfactory results are obtained or certain criteria of termination is met.



The basic scheme of evolutionary algorithms is given below:

a) Mutation

After the random generation of population, in each generation, a Donor vector $\overrightarrow{V_i}(t)$ is created for each $\overrightarrow{X_i}(t)$.This donor vector can be created in different ways (see section 3.2).

b) Recombination

Now a trial offspring vector is created by combining components from the Donor vector $\overrightarrow{V_i}(t)$ and the target vector $\overrightarrow{X_i}(t)$. This can be done in the following way

$$U_{i,j}(t) = V_{i,j}(t) \quad \text{if } rand_{i,j}(0,1) <= cr$$
$$= X_{i,j}(t) \quad \text{otherwise}$$

c) Selection

Selection in DE adopts Darwinian principle "Survival Of the Fittest". Here if the trail vector yields a better fitness value, it replaces its target in the next generation; otherwise the target vector is retained in the population. Hence the population either gets better (w.r.t. the fitness function) or remains constant but never deteriorates.

$$\overrightarrow{X_i}(t+1) = \overrightarrow{U_i}(t) \quad \text{if } f(U_i(t)) \le f(X_i(t)),$$
$$= \overrightarrow{X_i}(t) \quad \text{if}$$
$$f(X_i(t)) < f(U_i(t)).... \rightarrow (3)$$

DE mutation Schemes

The five different mutation schemes suggested by Price [6] is as follows:

Scheme 1-DE/rand/1

In this scheme, to create a donor vector $\overrightarrow{V_i}(t)$ for each ith member, three other parameter vectors (say the o_1, o_2, and o_3th vectors) are chosen randomly from the current population. A scalar number F is taken. This number scales the difference of any two of the three vectors and the resultant is added to the third one. For the ith donor vector, this process can be given as

$$\overrightarrow{V_i}(t+1) = \overrightarrow{X_{o_1}}(t) + F * (\overrightarrow{X_{o_2}}(t) - \overrightarrow{X_{o_3}}(t)) \rightarrow (4)$$

Scheme 2-DE/rand to best/1

This scheme follows the same procedure as that of the Scheme1. But the difference is, now the donor vector is generated by randomly selecting any two members of the population (say the $\vec{X}_{0_2}(t)$, and $\vec{X}_{0,3}(t)$ vectors) and the best vector of the current generation (say $\vec{X}_{best}(t)$). For the ith donor vector, at time t=t+1, this can be expressed as

$$\vec{V}_i(t+1) = \vec{X}_i(t) + \lambda * \left(\vec{X}_{best}(t) - \vec{X}_i(t)\right) + F * \left(\vec{X}_{o_2}(t) - \vec{X}_{o_3}(t)\right) \rightarrow (5)$$

Where λ is a control parameter in DE and ranges between [0, 2] . To reduce the number of parameters, we consider λ =F.

Scheme 3-DE/best/1

This scheme is identical to Scheme 1 except that the result of the scaled difference is added to the best vector of the current population. This can be expressed as

$$\vec{V}_i(t+1) = \vec{X}_{best}(t) + F * \left(\vec{X}_{o_1}(t) - \vec{X}_{o_2}(t)\right) \rightarrow (6)$$

Scheme 4-DE/best/2

In this scheme, the donor vector is formed by using two difference vectors as shown below

$$\vec{V}_i(t+1) = \vec{X}_{best}(t) + F * \left(\vec{X}_{o_1}(t) - \vec{X}_{0_2}(t)\right) + F * \left(\vec{X}_{o_3}(t) - \vec{X}_{o_4}(t)\right) \rightarrow (7)$$

Scheme 5-DE/rand/2

Here totally five different vectors are selected randomly

from the population, in order to generate the donor

vector. This is shown below

$$\vec{V}_i(t+`) = \vec{X}_{o_1}(t) + F_1 * \left(\vec{X}_{o_2}(t) - \vec{X}_{o_3}(t)\right) + F_2 * \left(\vec{X}_{o_4}(t) - \vec{X}_{o_5}(t)\right) \rightarrow (8)$$

Here *F*1 and *F*2 are two weighing factors selected in the range from 0 to 1. To reduce the number of parameters we may choose *F*1 = *F*2 = *F*.

The experiment we conducted in this study uses Scheme 1-DE/rand/1(equation 4).

Procedure for DE

1. Randomly initialize the position of the particles
2. Evaluate the fitness for each particle
3. For each particle, create Difference-Offspring
4. Evaluate the fitness of the Difference-Offspring
5. If an offspring is better than its parent then replace the parent by offspring in the next generation;
6. Loop to step 2 until the criterion is met, usually a sufficiently good fitness or a maximum number of iterations.

4. FUNCTION OPTIMIZATION USING PSO AND DE

The aim of an optimization problem is to determine a best-suited solution under a given set of constraints. Mathematically an optimization problem involves a fitness function, which needs to be maximized or minimized. Most of the traditional optimization techniques employ derivative approach to locate the optima. This approach fails to determine the global optima for multimodal optimization problems. More recently, the optimization problem is being represented as an intelligent search problem, where one or more agents are employed to determine the optima on a search space[7]. This technique of searching is called as evolutionary based searching

All evolutionary based algorithms follow a single approach for solving an optimization problem. The approach is given below:

1. Initialize the population with particles or vectors
2. Calculate the fitness function for each particle or vector
3. Based on the best fitness value, update the position of the particles or vectors

The difference among different evolutionary based algorithms lies in the way they update the particles by changing the parameters.

In our experiment, we have taken two such evolutionary based search techniques PSO(Kennedy and Eberhart 1995) and DE(Storn and Price 1997). These two techniques are first tested on standard four benchmark functions[Table 1] and the comparative analysis is given with plots.[See Table 2 and Fig1]. We have coded all the programs in MATLAB 7.5 software and run with 1.86 Core2 DUO processor , 1GB RAM.

5. Frame work for Data Clustering

Data clustering is a process of grouping a set of data vectors into a number of clusters or bins such that elements or data vectors within the same cluster are similar to one another and are dissimilar to the elements in other clusters. Clustering algorithms can be grouped into two main classes of algorithms, namely Supervised and Unsupervised. With Supervised clustering, the learning algorithm has an external teacher that indicates the target class to which the data vector should belong. For unsupervised clustering, a teacher does not exist, and data vectors are grouped based on distance from one another [8]. Data clustering can be hierarchical or partitional. We here, instead, confine ourselves to the field of evolutionary based partitional Clustering.

Partitional clustering algorithms, attempt to decompose the dataset into a set of disjoint clusters by optimizing a criteria (such as intra cluster distance which is to be detailed below) . Hence clustering can also be treated as an optimization problem and we can apply the evolutionary techniques to get the optimum solutions. In this paper we have taken PSO and DE for clustering three real-world datasets (see Section 6.1 for data description). Comparative analysis is given in Table 3 and a sample plot is also shown(See fig 2 and fig 3). All the algorithms have been coded in MATLAB 7.5

5.1 Basic K-Means clustering algorithm

K-Means algorithm falls under partitional based clustering technique. It was introduced by MacQueen'67[9].K in K-Means, signifies the number of clusters into which data is to be partitioned. This algorithm aims at assigning each pattern of a given dataset to the cluster having the nearest centroid. K-Means algorithm uses similarity measure to determine the closeness of two patterns. Similarity can be measured using Euclidean Distance or Manhattan Distance or Minkowski Distance. In this paper, Euclidean Distance is considered as the similarity measure.

The algorithm for K-Means is as follows:

i. Initialize randomly N_k cluster centroid vectors

ii. Repeat

a. Assign each data vector to the cluster with closest centroid vector. The distance from the data vector to the centroid vector is determined using the equation

$$d(z_p, a_j) = \sqrt{\sum_{k=1}^{N_d} (z_{pk} - a_{jk})^2} \rightarrow (9)$$

b. Recalculate the cluster centroid vectors, using

$$a_j = \frac{1}{n_j} \sum_{\forall z_p \in C_j} z_p \rightarrow (10)$$, where n_j is the number of data vectors in cluster j **Until** a stopping criterion is reached

The process of K-Means clustering can be stopped when one of the following criteria are satisfied:

i. When the maximum number of iterations has reached

ii. When there is no change in the newly obtained centroid vectors

For the purpose of this study, the second criteria got used

5.2 PSO Clustering

There are different versions of PSO models[10]. In this work we stick to the basic PSO model called gbest model wherein every particle will interact with every other particles to decide its optimum direction. This section now presents a standard gbest PSO clustering algorithm.

Data vectors can be clustered using standard gbest PSO as follows:

i. Randomly select N_k cluster centroids to initialize each particle

ii. For I =1 to I_{max} do

a) For each particle i do

b) For each data vector z_p

i. calculate Euclidean distance $d(z_p, a_{ij})$ to all cluster centroids C_{ij} using equation 9

ii. assign z_p to the cluster C_{ij} such that $d(z_p, a_{ij}) = \min_{\forall k=1.....N_k} \{d(z_p, a_{ik})\}$

iii. calculate the fitness using the equation

$$\frac{\sum_{j=1}^{N_k} \left[\sum_{\forall z_p \in C_{ij}} d(z_p, a_j) \right]}{N_k} \rightarrow (11)$$

c) Update the pbest and gbest positions

d) Update the cluster centroids using the equations (1) and (2)

Where I_{max} is the maximum number of iterations.

5.3 DE Clustering

There are different DE Schemes available [6]. Here we stick to the classical DE algorithm (Scheme 1, See section 3.2), which is presented below:

Data vectors can be clustered using classical DE as follows:

i. Initialize each vector to contain K number of randomly selected cluster centers

ii. For *I=1 to* I_{max} do

a. For each vector i do

b. For each object in the data set Z_p

i. Calculate the Euclidean distance $d(z_p, a_{ij})$ to all cluster centroids C_{ij} using equation 3

ii. Assign Z_p to the cluster C_{ij} such that $d(z_p, a_{ij}) = \min_{\forall k=1.....N_k} \{d(z_p, a_{ik})\}$

c. Change the population members according to the DE algorithm outlined in(section 3.3). Use the vectors fitness to guide the evolution of the population.

iii. Report cluster centers and the partition obtained by globally best vector at time $I=I_{max}$

6. Experimental set-up and Results

The algorithms have been tested on four benchmark functions: Sphere,Rosenbrock, Rastrigin and Griewank. Table1 gives the descriptions of these functions. A comparison of the two algorithms, PSO and DE, has been reported for the afore mentioned benchmark functions(see Table2 and Fig1)

These algorithms are applied for clustering three real world data sets described below. All these data sets are available at www.ics.**uci**.edu/~mlearn/ML**Repository**.html. Fitness comparison of the two algorithms, PSO and DE has been reported in Table3. Fig 2 and Fig 3 shows the three-dimensional plots of observations for PSO and DE algorithms respectively on Iris dataset.\

6.1 Dataset Description

Three well-known real-world datasets from the machine learning repository have been considered for this experiment. They are:

1. Fisher's iris dataset (n=150,p=4,c=3), which consists of n objects characterized by p features (sepal length, sepal width, petal length, petal width). There are c categories in this data: Iris setosa (50), Iris versicolor (50), Iris virginica (50)
2. Wisconsin breast cancer dataset (n=683,p=9,c=2), which consists of n objects characterized by p features (clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoi and mitoses). There are c categories in the data: malignant (444 objects) and benign (239 objects)
3. Wine recognition dataset (n=178,p=13,c=3), which consists of n objects, characterized by p features(Alcohol, Malicacid, Ash, Alcalinity of ash, Magnesium, Total Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines,Proline) with c categories in the data: class 1(59 objects), class 2 (71 objects), class 3 (48 objects)

For PSO, we have set the inertia of weight, w=0.7 and c1=c2=2 for our experiment
For DE, we have set the cross over rate cr =0.9, the weighting factor is chosen as F=0.8[6].

Table 1 Descriptions of Benchmark functions used in the experiment

Function	Definition	Search domain	Global minima
Sphere	$f(x)=\sum_{i=1}^{n} x_i^2$	$-5.12 \le x_i \le 5.12$	$x^* = (0)^*, f(x^*) = 0.$
Rosenbrock	$f(x)=\sum_{i=1}^{n-1}\left[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right]$	$-5 \le x_i \le 10$	$x^* = (1)^*, f(x^*) = 0.$
Rastrigin	$f(x)=10n+\sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\Pi x_i)\right)$	$-5.12 \le x_i \le 5.12$	$x^* = (0)^*, f(x^*) = 0.$
Griewank	$f(x)=\sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$-600 \le x_i \le 600$	$x^* = (0)^*, f(x^*) = 0.$

6.2 Results

Table 2: Optimized results of PSO and DE on the afore mentioned four Benchmark functions

Function Name	Best PSO	Best DE	Mean PSO	Mean DE	Standard Deviation-PSO	Standard Deviation-DE
Sphere	0	0	0	0	0	0
Rosenbrock	0.0042	0	0.0283	0	0.034	0
Rastrigin	0	0	0	0	0	0
Griewank	0	0	0	0	0	0

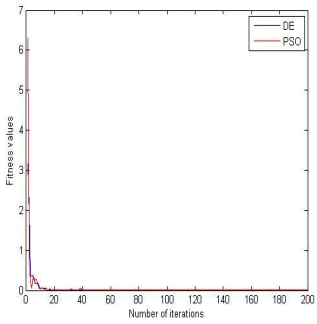
Number of particles/vectors taken=10
Number of variables taken = 2
Number of Iterations = 200
Number of runs = 2

Table 3: Mean Intracluster Distances of PSO and DE on the given three real-world data sets

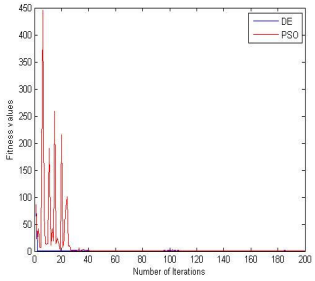
Dataset Name	Clusters	Mean Intracluster Distance PSO	Mean intracluster Distance DE
Iris	3	46.9929	38.047
Wine	3	5600.2	5846
Breast Cancer	2	75444	75285

Number of iterations=50
Number of vectors/particles =5

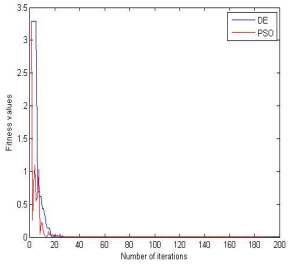
Fig1. Diagram of convergence for the two algorithms applied to all four benchmark functions.



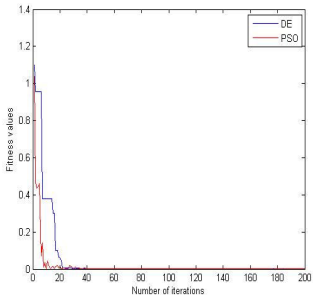
a. Sphere



b. Rosenbrock

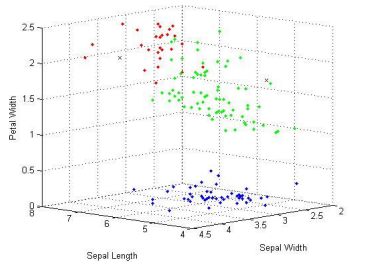


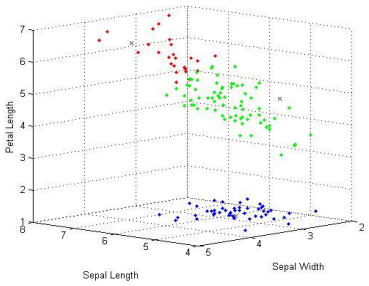
c. Rastrigin

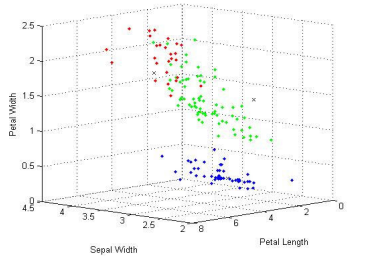


d. Griewank

Fig2. PSO generated Three-Dimensional Clusters of IRIS dataset







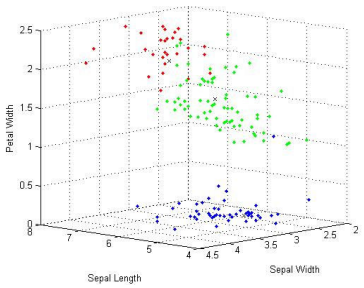
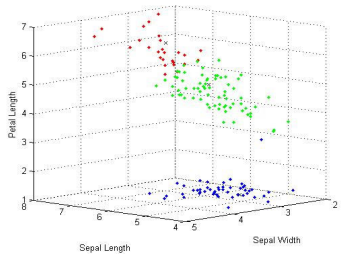
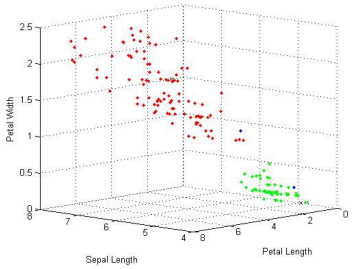
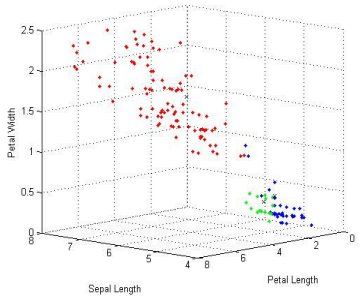
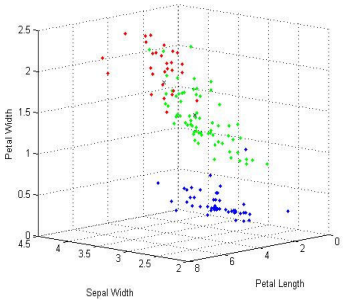


Fig 3:DE generated Three-Dimensional Clusters of IRIS dataset









7. CONCLUSION

In this paper, the performance of PSO and DE for finding the optimum solution is compared on four benchmark functions(Table 1) and also on clustering 3 real-world data sets. From our experiments it turns out that DE is clearly superior compared to that of PSO with respect to the consistency and robustness of results. Apart from performance issues, DE is very easy to implement with very little parameter tuning when compared to that of PSO. As further enhancement we will like to see how different mutation models perform for clustering results.

8. REFERENCES

[1] R. C. Eberhart, and Y. Shi. Particle swarm optimization: developments, applications and resources. Proceedings of the IEEE congress on evolutionary computation. 2001.

[2] R. Poli, J. Kennedy, and T. Blackwell. Particle Swarm Optimization: An overview. Swarm Intelligence. July, 2007.

[3] A. Banks, J. Vincent, and C. Anyakoha. A review of particle swarm optimization, part 1: Background and Development. Nat Comput. 2007.

[4] Y. Shi. Particle swarm optimization. IEEE Neural network society. February, 2004.

[5] Y. Shi, and R. C. Eberhart. Empirical study of particle swarm optimization. IEEE. 1999.

[6] Storn R and Price K (1997), Differential Evolution – A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces, Journal of Global Optimization, 11(4), 341–359..

[7] S. Das, A. Abraham, and A. Konar. Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives. Studies in Computational Intelligence (SCI) 116, 1-38, 2008.

[8] A. K. Jain, M. N. Murthy, and P. J. Flynn. Data clustering: a review. ACM Computing surveys, Volume 31, No. 3. September, 1999.

[9] S. Das, and A. Abraham. Automatic clustering using an improved differential evolution algorithm. IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans, Volume 38, No. 1, January, 2008.

[10] Kennedy JF, Eberhart RC. Particle swarm optimization. In: Proceedings of the IEEE International conference on neural networks, vol. 4. Perth, Australia; p. 1942–48,1995.

[11] Shi Y and Eberhart RC ,Parameter Selection in Particle Swarm Optimization, Evolutionary Programming VII, Springer, Lecture Notes in Computer Science 1447, 591–600, (1998).

[12] Kenneth D. Bailey, Cluster Analysis. Sociological Methodology, Vol. 6 pp. 59-128, American Sociological Association, 1975.

[13] Oded Z. Maimon, Lior Rokach, The Data Mining and Knowledge Discovery Handbook, Springer Science & Business, 2005.