Modified Particle Swarm Optimization (PSO) with Laplace Mutation Operator (LMPSO)

By:

Muhammad Imran

Reg No: 01728

Supervised By:

Dr. Hajira Jabeen

Assistant Professor

IQRA University Islamabad Campus

Thesis Submitted as partial fulfillment of requirement for the Degree of Master of Science in Computer Science (MSCS)

DEPARTMENT OF COMPUTING AND TECHNOLOGY

IQRA UNIVERSITY ISLAMABAD CAMPUS

January 2010

ABSTRACT

Particle Swarm Optimization (PSO) algorithm has shown good performance in many optimization problems. However, PSO can get stuck into the local minima. To prevent the problem of early convergence into a local minimum, different researchers have proposed some variants. In this work different variants of PSO are reviewed. Three new variants of PSO are proposed and compared with existing variants for function optimization applications. The analysis in this research shows the effect of different mutation operators on Particle Swarm Optimization (PSO). The result shows that the proposed variants improve the performance of PSO algorithm.

ACKNOWLEDGEMENTS

Up and above everything, I am grateful to Almighty ALLAH, The Beneficent, The Merciful, and His Prophet (Peace be upon him) who is forever a true torch of guidance for whole humanity. I am greatly obliged to “ALLAH” by whose grace I have been able to complete this project successfully. I am especially indebted to my supervisor Dr. Hajira Jabeen, for giving me an initiative to this project. I am very thankful to my supervisor Dr. Hajira Jabeen for her inspiring guidance, remarkable suggestions, constant encouragement, keen interest, constructive criticism, and friendly discussion, which enabled me to complete this project efficiently. Without her support and proper guidance, it would be almost impossible to accomplish this task successfully.

I am grateful to my loving parents and my elder brother for providing me all sort of moral and social support in life. Their prayers have enabled me to reach this stage.

Finally I thank GOD again for his countless blessings.

DECLARATION

I hereby declare that the work done in this research is my original piece of work performed under the supervision of Dr. Hajira Jabeen, Assistant Professor, Iqra University Islamabad. Material from other authors, researchers has been properly acknowledged, where ever included in this report. And further I also certify that no part of this work, separate or as a whole has been presented for award of any other degree program.

Name: Muhammad Imran

(MS (CS), 01728)

PUBLICATIONS

Mubashir Ahmad, **Muhamamd Imran**, Abdul Wahab "Management Issues in Software Development", Proceedings of the 9th WSEAS international conference on Software engineering, parallel and distributed systems., 2010 United Kingdom.

Muhammad Imran, Hajira Jabeen, Mubashir Ahmad, Qamar Abbas,Waqas Bangyal, Qamar Abbas," Opposition based PSO and mutation operators (OPSO with Power Mutation)", 2nd International Conference on Education Technology and Computer (ICETC), 2010 China.

Muhamamd Imran, Qamar Abbas, Abdul Rauf, Fazal Wahab, "Using Ontologies for Software Process Improvement (SPI) in Small & Medium Organizations", International Conference on Intelligence and Information Technology (ICIIT),(28-30 OCT 2010) Pakistan

Muhamamd Imran, Zahid Manzor, Fazal Wahab, Qamar Abbas, Waqas Haider Bangyal, "Risk management Cycle and Risk Types", International Conference on Intelligence and Information Technology (ICIIT), (28-30 OCT 2010) Pakistan.

Muhammad Imran, Mubashir Ahmad, Muhammad Shoaib, Sardar Zafar Iqbal and MuhammadAli Abid, "New Web Portal Quality Model", International Conference on Intelligence and Information Technology (ICIIT), (28-30 OCT 2010) Pakistan.

D EDICATION

To hands،

Shivering and uplifted

Eyes heavy and thoughtful

Of my parents؛

Hands ever praying for me

Eyes with dreams in of my bright tomorrow

*These hands may never fall down_*

*These eyes may never go to asleep_*

This Thesis is dedicated to my most respectable and honorable

Parents، Teachers and Brothers

THESIS APPROVAL SHEET

It is certify that Muhammad Imran, Student of MS (CS), Department of Computing & Technology, Student ID (01728), of IQRA University Islamabad, has submitted the final thesis report on "**To Analyze the Effect of Mutation Operators on Particle Swarm Optimization (PSO)**". We have read the report and it fulfills the partial of Master of Science in Computer Science degree.

INTERNAL EXAMINER:

Name:

Designation:

Organization: Iqra University Islamabad Campus

Signature: _____

EXTERNAL EXAMINER:

Name:

Designation:

Organization:

Signature: _____

SUPERVISOR:

Name: Dr. Hajira Jabeen

Designation: Assistant Professor

Organization: Iqra University Islamabad Campus

Signature: _____

Table of Contents

List of Figure

List of Tables

1. Chapter 1

1.1. Introduction

In computer science the term optimization is referred to find the best solution of a problem [45]. The term “best” here refer to acceptable or satisfactory solution. Function optimization is used to minimize or maximize the function subject to some constraints. Some features of optimization are listed as

- **Cost function**: which represents the quantity to be optimized that is the quantity to be minimized or maximized.
- **Decision variables**: A set of unknowns or variables which affects the value of the objective function.
- **Constraints**: A set of constraints that restrict the values that can be assigned to the unknowns. Most problems define at least a set of boundary constraints which define the domain of values for each variable.

Optimization problems are classified by

- The number of variables that influence the objective function

  A problem which based on a single variable for optimization is referred as uni-variable, while with more variables is referred as multivariable.
- The type of variables
  Different types of problem are like
  1. Continuous problem contain continuous valued variables
  2. Discrete problems consist of integer valued variables
  3. Mixed both continuous and discrete, based on both continues and integer valued variables
  4. Combinatorial optimization problem solutions are permutations of integer valued variables.
- The degree of nonlinearity of the objective function
  Linear problem used linear objective function, quadratic used quadratic objective function while non linear problem used non linear objective function.
- The constraints used

A problem is known as unconstrained problem if it has only boundary constraint. If a problem has additional equality and inequality constraints called constrained problem

- The number of optima
  Some problem has only one optima and some has more than one.
- The number of optimization criteria
  Two types of problems are uni-objective and multi-objective problems.

In evolutionary field different algorithms are compared using a large test set particularly in case of function optimization. The objective of an optimization method is to allocate values from the allowed domain to the unknowns such that the objective function is optimized and all constraints are contented.

Optimization algorithms are search methods where the objective is to find a solution to an optimization problem such that a specified quantity is optimized, possibly subject to a set of constraints. Solutions originated by optimization algorithms are classified by the quality of the solution. The major types of solutions are referred to as local optima or global optima.

An optimization algorithm searches for an optimum solution by iteratively transforming a current candidate solution into a new, expectantly improved solution. Optimization methods can be divided into two major classes based on the sort of solution that is located. Local search algorithms use only local information of the search space surrounding the current solution to produce a new solution. A global search algorithm uses more information about the search space to locate a global optimum. Optimization algorithms are further classified into deterministic and stochastic methods. Stochastic methods use random elements to convert one candidate solution into a new solution. Deterministic methods do not employ random elements.

1.2. Motivation

Function optimization is a generic problem which can be easily resolved to any real life optimization problem like travelling sales man problem (TSP) or time table scheduling problem etc. Particle swarm optimization is a type of swarm intelligence inspired by bird flocking and fish schools. This type of swarm intelligence is used in practical applications such as in artificial neural networks and in grammatical evolution models. As Particle swarm optimization (PSO) is theoretically proved for optimization problem therefore PSO is used in this thesis.

1.3. Objective

The objective of this research is

- To analyze the effect of some mutation operators on PSO.
- To proposed new variants of PSO to improve its performance.
- To compare the results of proposed techniques with some existing techniques of PSO.

- To extend the PSO algorithm so that it becomes a global optimization technique with guaranteed convergence on global optima

1.4. Methodology

Some PSO variants are used in this research. Empirical results were obtained using different benchmark functions. These results are the supporting evidence for the convergence characteristics of the various PSO techniques. Mutation is used for global best particle finding. All the techniques used work on the functionality of mutation using different techniques.

1.5. Contribution

Three PSO variants are proposed and implemented. Their results are compared with other variants. The results of proposed techniques are better as compared to existing techniques.

1.6. Thesis outline

In chapter 2, work done by other researchers is presented. In chapter 2 different variants of PSO are discussed in detail. Chapter 3 contains the detail of some PSO variants with mutation operator and discuses proposed new variants in detail. Chapter 4 contains the all results of 21 Benchmark functions with different PSO techniques. The results are analyzed and discussed in detail in chapter 4. Chapter 5 is about the conclusion of the research work.

2. Chapter 2

Chapter 2 discusses the back ground work; some previous variants of PSO will be discussed in detail. Parameter of PSO will also discuss.

2.1. Swarm Intelligence

Swarm intelligence (SI) is the intelligent behavior of non intelligent species, like ants (going toward search of food) or birds (during flying). SI system is consist of simple independent population entities interacting with their environments and each other. There is no central control dictating the performance of these independent entities.

SI is used o explore discrete problem solving without having a central control structure the artificial intelligence. Real life swarm intelligence can be observed in ant colonies, bacterial growth, bird flocks, animal herds and fish schooling.

Ant colony behavior is one of the popular models of swarm behavior. Through swarm intelligence ants can determine the shortest path to source of food. Ant colony optimization has been used to solve traveling salesman problem, scheduling problem, vehicle routing problem and many other problems.

2.2. Introduction to Particle Swam Optimization (PSO)

PSO is a population based optimization method purposed by Kennedy and Eberhart. The algorithm simulates the behavior of bird flock flying together in multi dimensional space in search of some optimum place, adjusting their movements and distances for better search [1]. PSO is very similar to evolutionary computation such as Genetic algorithm (GA). The swarms are randomly initialized and then search for an optimum by updating generations.

PSO is a combination of two approaches, one is cognition model that is based on self expression and the other is a social model, which incorporates the expressions of neighbors. The algorithm mimics a particle flying in the search space and moving towards the global optimum. A particle in PSO can be defined as $P_i \in [a, b]$ where i=1, 2, 3…. D and a, b $\in R$, D is for dimensions and R is for real numbers [30]. All the particles are initialized with random positions and with random velocities [1], then particles move towards the new position based on their own experience and with neighborhood experience. Each particle in PSO maintains two important positions called p_{best} and g_{best} where p_{best} is the particle's own

best position and g_{best} is the global best position among all the particles. The equations used to update a particle's velocity and position, are the following

$$V_i(t+1)=V_i(t)+C_1*r_1\left(P_{best}-n_i(b)\right)+C_2*r_2\left(g_{best}-x_i(t)\right) \quad \text{..... (2.1)}$$

$$X_i(t+1)=X_i(t)+V_i(t+1) \quad \text{..... (2.2)}$$

Where X_i is the position, V_i is the velocity, P_{best} is the personal best position and g_{best} is the global best position for PSO. Similarly r_1 and r_2 are two random numbers their range is chosen from [0, 1], C_1 and C_2 are learning factors specifically the cognition and cognition component influential respectively.

2.3. Original PSO

```
Initialize the population randomly
While (Population Size)
{
Loop
      Calculate fitness
      If fitness value is better from the best fitness
      value (pbest) in history then
      Update pbest with the new pbest
End loop
Select the particle with the best fitness value from all
particles as gbest
While maximum iterations or minimum error criteria is not
attained
      {
For each particle
      Calculate particle velocity by equation (2.1)
      Update particle position according to equation (2.2)
Next
      }
}
```

2.4. PSO Variants

PSO proposed by Kennedy et al [1] in 1995, is techniques is theoretically proved therefore researchers are trying to make this technique as best as possible, lot of PSO variants have been proposed by researchers with respect to different parameters and operators, the detail of some of them is given as.

2.4.1. Initialization

Initialization of population plays an important role in the evolutionary and swarm based algorithms. In case of bad initialization, the algorithm may search in unwanted areas and may be unable to search for the optimal solution.

Nguyen et al [14] inspect the some randomized low discrepancy sequence to initialize the swarm to increase the performance of PSO. They used three low discrepancy sequence halton, faur and sobol. halton sequence is actually the extension of van der corput. ven der corput sequence is one dimensional in order to cover search space in n dimension and halton is defined as one of the extension of vender corput sequence. They compare their all three new variants with global best fitness of PSO in which swarms are initialized with pseudo random number. It is observed from the results that S-PSO is dominated among all the four version of the PSO. In case of small search space PSO initialized with faur sequence can perform well while in case of high dimensions Halton performance might be fine. Six benchmark functions are used to evaluate the performance of new three version of PSO.

Pant et al [23] explore the effect of initializing swarm with the vender corput sequence which is a low discrepancy sequence to solve the global optimization problem in large dimension search space. They named the proposed algorithm as VC_PSO. The author claim that PSO performance is very well for problems having low dimensions but as the dimensions increase the performance deteriorates, this problem become more severe in case of multimodal functions. The author says that one of the reasons for this poor performance may be random initialization of the swarm therefore they proposed a PSO technique which initializes the swarm with low discrepancy random number to overcome this problem. They compare their algorithm with PSO using sobol random sequence which is dominated by halton and faur sequence. Vender corput is a low discrepancy sequence over the unit interval proposed by a Dutch mathematician in 1935, which is defined by the radical inverse function.

They used the linear decreasing inertia weight from .9 to .4 with C1=C2=2.0 to test their technique.

Jabeen et al [30] proposed opposition based initialization which calculates opposition of randomly initialized population and selects better among random and opposition as initial population. This population is provided as an input for traditional PSO algorithm. The proposed modification has been applied on several benchmark functions and found successful.

Chang et al [33] proposed an enhanced version of opposition based PSO called quasi-oppositional comprehensive learning PSO (QCLPSO). Instead of calculating traditional opposite of a point, the proposed modification calculates Qausi opposite particle, which is generated from the interval between median and opposite position of the particle. According to authors the Qausi opposite particles have higher chances of being closer to global optima then opposite particle calculated without apriori information.

2.4.2. Constriction Coefficient

This approach to balance the exploration-exploitation trade off has been proposed by Clerc [38], which uses a new parameter 'χ' called the constriction factor. The velocity update scheme proposed by Clerc can be expressed for the d^{th} dimension of i^{th} particle as:

$$v_{ij}(t+1)=\chi\left[v_{ij}(t+1)+\varphi_1\left(v_{ij}(t)-x_{ij}(t)\right)+\varphi_2\left(\hat{y}_{ij}(t)-x_{ij}(t)\right)\right] \quad \text{……..…..} \quad (2.19)$$

Where $\chi = \frac{2}{\left|4-\phi-\sqrt{\phi 2-4\phi}\right|}$

With $\varphi = C_1+C_2, \quad \varphi_1=C_1 r_1 \quad , \quad \varphi_1=C_2 r_2$

Constriction coefficient results in the quick convergence of the particles over time. That is the amplitude of a particle's oscillations decrease as it focuses on the local and neighborhood previous best points. Though the particle converges to a point over time, the constriction coefficient also prevents collapse if the right social conditions are in place. The particle will oscillate around the weighted mean of p_{best} and g_{best}, if the previous best position and the neighborhood best position are near each other the particle will perform a local search.

2.4.3. Inertia Weight

The inertia weight is a scaling factor associated with the velocity during the previous time step resulting in a new velocity update equation as eq (2.1), introduce by Shi [2]. Inertia weight is used to control the exploration and exploitation abilities of the swarm. Large value of inertia weight promotes exploration while small value promotes local exploitation [2]. Some researchers [26, 27, 34] used fixed inertia weight and other [3, 4, 19] used decreasing inertia weight. Linear decrease of inertia weight may have lack of global search ability due to which in complex and complicated problems PSO may fail to find the required optima [44]. To achieve better result using linear decreasing inertia weight need maximum number of iteration.

2.4.3.1. Linear Decreasing

In linearly decreasing inertia weight, large value (0.9) linearly decreased to small value (0.4). Formula used for linearly decreasing inertia weight is following.

$$w_{ldw} = (w_{max} - w_{min}) \times \frac{iteration\,max - iteration\,i}{iteration\,max} + w_{min} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots .... (2.3)$$

Where w_{max} is 0.9 and w_{min} is 0.4 $iteration\,max$ is the maximum number of iterations, $iteration\,i$ is the iteration during which inertia weight is calculated.

According to shi [44] by decreasing linearly inertia weight promotes global search ability at the start of the run and promotes local search at the end of run.

Through literature review it has been observed that linearly decreased inertia weight is better rather than using fixed inertia weight. Exploration and exploitation can be controlled well by linear decreasing inertia weight. The decrease of inertia weight narrows the search space from explorative to exploitative mode.

2.4.3.2. Exponent Decreasing

The exponent decreasing inertia weight is introduced by Hui [28]. The motivation for this idea was to balance the local and global search with overcome on premature convergence. Formula for exponent decreasing inertia weight is as

$$w$$

$$(¿¿ ini - w_{end} - d_1)\exp\left(\frac{1}{1+d_2 t/t_{max}}\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ldots (2.4)$$

$$w = ¿$$

Where t_{max} denotes the max iteration, t denotes the t[th] iteration, w_{ini} denotes the original inertia weight, w_{end} denotes the inertia weight value when the algorithm process run the max iterations, d_1 and d_2 is a factor to control w between $w_{ini} \wedge w_{end}$.

In order to beat the early convergence Hui [29] proposed a variant of PSO with exponent decreasing inertia weight and stochastic mutation. Exponent decreasing inertia weight describe in eq (2.4). Mutation probability p_m is defined as

$$p_m = \begin{cases} \lambda \delta^2 < \delta_d^2 \wedge ¿ F(gbest) - F_m \vee ¿ \varepsilon \\ 0\, other \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.5)}$$

Where ε >0, $F(gbest)$ is the fitness of current global particle, Fm is the theory optimum value of the optimal problem, δ^2 is defined as

$$\delta^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{f(i) - favg}{f} \right)^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

(2.6)

Where $favg = \frac{1}{n} \sum_{i=1}^{n} f(i)$, f is factor of returning. It can be limit δ^2 value of f is defined as

$$\begin{matrix} ¿ f(i) - favg \vee ¿ \\ 1, max(¿), i < [1, n] \\ f = max ¿ \end{matrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.7)}$$

Gbest is mutated as following

$$gbest = \begin{cases} gbest + \eta \times gbest\ t < t_{max1} \\ gbest + 0.1 \times \eta \times gbest\ t_{max1} < t < t_{max} \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(2.8)}$$

Where η is random variable and it denotes the standard normal distribution.

2.4.3.3. Non linear Decreasing

In non linear decreasing inertia weight, a large value decreases to small value but decrease non-linearly. In non linear decreasing of inertia weight search space can be explored in short time but take larger time to exploit the search space. From Naka et all [6]

$$w(t+1)=\frac{(w(t)-0.4)(n_t-1)}{n_t+4} \quad \text{..... (2.9)}$$

From Venter el al [13]

$$w(t+1)=\alpha w(t') \quad \text{..... (2.10)}$$

Where $\alpha=.975$ and t' is the time step when the inertia last changed. The only condition when the inertia is changed is no significant difference between the swarm fitness.

2.4.3.4. Fuzzy Adaptive

Using fuzzy sets and rules, the inertia weight is adjusted dynamically [4]. Fuzzy system for the inertia adaption by Shi and Eberhart consists of the following components.

- Two input variables are used, one for the fitness of the global best position, and the other for the current value of the inertia weight.
- One output variable to represent the change in inertia weight.
- Three fuzzy sets LOW, MEDIUM and HIGH are implemented as a left triangle, triangle and right triangle membership functions respectively [4].
- Nine fuzzy rules are used which calculate the change in inertia is calculated [4].

2.4.3.5. Uniform Random

Zhang et al [10] set the inertia weight as uniform random number between 0 and 1. Author claims that it is more capable to escape from local minima. According to author's proposed method for inertia weight, can overcome two problem of linearly decreasing inertia weight.

1. We can overcome the problem of linearly inertia weight dependency on maximum iteration.
2. Another is avoiding the lack of local search ability at early of run and global search ability at the end of run.

They test their method on three benchmark functions with different dimensions using different number of generations. The result of new proposed inertia weight found best.

2.4.3.6. Dynamic

Wei et al [13] proposed dynamic PSO with dimension mutation. First they introduce dynamic inertia weight which is changed dynamical based on speed and accumulation factor then presented a dimension mutation operator to overcome the premature convergence. The formula for speed factor used is

$$h=\frac{F\left(g_{best_t}\right)}{F\left(g_{best_{t-1}}\right)} \quad , \quad h\in(0,1) \quad \text{……..……} \quad (2.11)$$

'h' is speed factor. The accumulation factor is

$$s=\frac{1}{N.L}.\sum_{i=1}^{N}\sqrt{\sum_{d=1}^{n}\left(P_{id}-\acute{P}_d\right)^2} \quad \in(0,1) \quad \text{……..……..……} \quad (2.12)$$

Where N is the population size, n is the number of variables, L is the length of maximum diagonal and P_{id} is the d^{th} coordinate of the i^{th} particle. They calculate inertia weight as following.

$$\omega=\omega_0-h\omega_h-s\omega_s \quad \text{……...……} \quad (2.13)$$

Where ω_0 =1, $\omega_h\in(0.4,0.6)$, $\omega_s\in(0.1,0.2)$

They named their proposed algorithm as DPSO. They compared the result of DPSO with CEP [42], FEP [42] and LDW [43] using 5 benchmark functions and obtained better result than other.

2.4.3.7. Gaussian

Pant et al [16] proposed new version of PSO that uses gaussian inertia weight. Responsible factor for the uniqueness of the modified algorithm was

- Development of new inertia weight using Gaussian distribution
- Use of different distribution then uniform distribution for the generation of the initial swarm.

Inertia weight given by [16]

$$w = |(rand)|/2 \quad \text{……..……..……} \quad (2.14)$$

Where rand is the random number having gaussion distribution..

2.4.3.8. Adaptive Dynamic

Shu-Kai [17] proposed PSO using an adaptive dynamic weight scheme. They propose a novel nonlinear function amendable inertia weight adaptation with an active method for improving the performance of PSO algorithms.

The aim was the determination of the inertia weight through a nonlinear function at each time step. The nonlinear function is given by

$$\omega = d^{r}_{\omega_{initial}} \quad \text{……..……} \quad (2.15)$$

Where d is the decrease rate from 1.0 to 0.1 and r is dynamic adaption rule depending on the following rule. For minimization case it follows.

If $f = (P_{gd-new}) < f(P_{gd-old})$ then $r \to r-1$ ……...…….. (2.16)

If $f = (P_{gd-new}) \geq f(P_{gd-new})$ then $r \to r+1$ …… (2.17)

Where P_{gd-new} and P_{gd-old} represent the global best position at current and previous time step respectively. Author claims that this method desires to make particle take off quickly towards the optimal solution and then perform local enhancement around the neighborhood of finest solution by decreasing inertia weight. They test their technique using different benchmark function and find best results.

2.4.3.9. Dynamic Using Mutation

Xuedan Liu et al [27] proposed the PSO with dynamic inertia weight using mutation, reinitialized the swarm when it get stagnant. Used the linearly decreasing inertia weight with following formula

$$w(t) = .9 - \left(\frac{t}{max_t}\right) \times 0.5 \quad \text{……..……} \quad (2.18)$$

Where max_t is maximum number of iteration then they used the wheel structure. Information of particle is exchanged with only the neighborhood's best position.

2.4.4. Mutations Operators

In PSO Mutation operator is used to change the position of particle from previous position to new positions. The basic role of mutation operator in PSO is quick convergence. It prevents the PSO from stagnation at the local minima. Some researchers used different mutation operators to overcome the premature convergence of the PSO. Using mutation operators researchers mutate the g_{best}, after mutating the g_{best} compare with original one if mutated one is better than replaced with original. Some of the mutation operators are Cauchy Mutation [19, 20], Adaptive Mutation [22], Power Mutation [34] and Gaussian Mutation [39]

To prevent PSO trapping from local optima Li et al [15] introduce Cauchy mutation in PSO. They named their algorithm as FPSO. They combined the natural selection strategy of evolutionary algorithm. They update the particles position by Cauchy mutation as

$V' = V + \exp(\delta)$ …… (2.20)

$X' = X + V'(\delta)$ ……..…….. ……..(2.21)

Where δ is a Cauchy random number. They update the velocity and position of the particle not only with eq (2.1) and (2.2), they also used the eq (2.12) and (2.13) for this purpose. Now choose the one with best fitness then generate next generation according to evolutionary selection strategy. They compare their algorithm with AMPSO using several benchmark function and fine better result.

Wang [19] proposed a new Cauchy mutation operator for PSO. This operator is applied to perform local search around the global best particle. The motivation for using such a mutation operator is to increase the probability of escaping from a local optimum. Several benchmark functions have been used to test the performance of this new operator and better results were achieved. The inertia weight used is 72984 and c1=c2 =1.49618. They named their algorithm as HPSO. HPSO compared with the standard PSO, FDR-PSO, CEP, and FEP using 6 unimodel functions and 4 multimodal functions. HPSO perform well for all functions but in some cases it falls in local minima.

Wang [20] proposed opposition based initialization in PSO coupled with application of Cauchy mutation operator. Cauchy mutation operator is used on the global best particle if newly created global best is better after application of mutation operator then global best is replaced. The proposed modification did not perform realistically well for multimodal functions.

Pant et al [22] proposed two variants of PSO; AMPSO1 and AMPSO2 for global optimization problem; author used adaptive mutation for both techniques. The main goal of author was to improve the diversity of PSO without compromising on the solution quality. In AMPSO1 personal best position of the swarm is mutated while in AMOPSO2 global best particle of the swarm is mutated. To compare the performance of APMSO with Cauchy mutation, Gaussian mutation, EP with having adaptive Gaussian and Cauchy mutation, FEP and CEP (version of EP) FEP with self adaptive Cauchy mutation whereas CEP is classical EP with self adaptive Gaussian mutation is used. Through the numerical results it is observed that AMPSO2 give good performance in7 out of 12 problem and FEP performance better in 4 functions out of 12, one function converged to zero in all algorithms except CEP.

Pant et al [24] explore the Sobol mutation operator in PSO to enhance the performance of basic PSO algorithm, which uses Qausi random Sobol sequence as they claim that random probability distribution cannot cover the search domain as Qausi random can cover. They named their operator Systematic mutation (SM). They proposed two variants of PSO SM-PSO1 and SM-PSO2. In SM-PSO1 mutation is applied to global best particle while SM-PSO2 mutates the worst particle of the swarm. They defined the SM operator as

SM= R1+ (R2/lnR1)……...……... (2.22)

Where R1 and R2 are random number generated by sobol sequence. The aim of mutating the worst particle is to move forward scientifically. Just three multimodal functions are used to test the proposed variants with different population size and dimensions.

Different mutation operators perform well for different types of problems. Li et al [25] Proposed adaptive mutation with three mutation operator to escape the particle from local optima. They apply Cauchy mutation, Gaussian mutation and levy mutation on the position and velocity. Choose the mutation operator on the base of selection ratio and evaluate its offspring fitness. Initially the probability set to 1/3. Latter on probability for each mutation operator is updated. Probability of an operator increased in case of best fitness

offspring and decreased in case of low fitness offspring. At the end only one appropriate mutation operator control the whole search space. 7 benchmark functions are used to test the algorithm with FEP algorithm. Adaptive algorithm performs better than FEP.

Yuelin [28] presented new adaptive mutation operator by fitness variance and space position aggregation degree to give a new PSO with adaptive mutation. They claim that algorithm can be stuck into local convergence when fitness variance of particles is near to zero. To get better this state of affairs they build adaptive mutation. The mutation probability is

$$P_m = \begin{cases} \frac{\exp(-h)}{5.0}, \sigma^2 < \sigma_1 \\ 0, \sigma^2 \geq \sigma_1 \end{cases} \quad \text{………………………………………….…………… ….. (2.23)}$$

Where σ^2 is the fitness value aggregation degree and is defines as

$$\sigma^2 = \sum_{i=1}^{n} \left[\frac{f_1 - f_{avg}}{f} \right]^2 \quad \text{……………………………………...…………………… ….. (2.24)}$$

Where $f_{avg} = ¿ \frac{1}{n} \sum_{i=1}^{n} f_1$ where f is factor of returning f can get any value but need to cate two things

- After returning $|f_1 - f_{avg}| \leq 1$
- f Changes with algorithm's evolution.

In equation 2.23 σ_1 is relative to the objective function and h is space position aggregation degree in t^{th} iteration below.

$$h = max_{1 \leq i \leq m} \{ \| X_i^t - P_g^t \|_2 \} / max_t \{ max_{1 \leq i \leq m} \{ \| X_i^t - P_g^t \|_2 \} \} \quad \text{…………….................. (2.25)}$$

A random number r [0, 1] is generated if r < p_m then mutation operator is implemented as following

$$p_i = p_i (1 + .5\eta) \quad \text{…………………………………………………………….……………… (2.26)}$$

Where p_i is best position of the i^{th} particle so far and η is n dimension random variable that reconciles normal distribution [0, 1].

Wu et al [34] proposed a new variant of PSO called power mutation PSO (PMPSO), which employs a power mutation operator. The core plan of PMPSO is to apply power

mutation on the fittest particle of current swarm. Purpose of power mutation is help particles to jump out from the local optima. The algorithm has been compared with several other PSO variants and better results have been achieved on most of the benchmark functions.

Pant et al [35] introduced the new mutation operator for improving the Quantum Particle Swarm Optimization (QPSO) algorithm. The mutation operator uses the Qausi random Sobol sequence and called as a Sobol mutation operator. Author proposed two versions using Sobol mutation in first best particle is mutated and in other worst particle is mutated. The proposed technique is compared with BPSO, QPSO and two more variants of QPSO also both proposed variants are compared with each other.

Tang [36] proposed adaptive mutation in PSO by mutating the global best particle as

$$gbest_j(t+1) = gbest_j(t) + \left[b(t)_j - a_j(t)\right] * rand() \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.27)$$

$$a_j(t) = Min\left(x_{ij(t)}\right), b_j(t) = Max\left(x_{ij}(t)\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2.28)$$

i=1,2,3…….,popsize ,j=1,2,3……..n

Where $gbest_j$ is the j^{th} vector of global best particle, $a_j(t)$ and $b_j(t)$ are the minimum and maximum values of j^{th} dimensions in current search space respectively, rand () is random number with in [0, 1] and t=1, 2, 3, 4 indicates the generations.

2.4.5. Miscellaneous PSO Variants

Silva et al [7] proposed predator pray optimization technique used for function optimization. New particles known as predators are introduced in the technique to avoid the premature convergence. The particles in the swarm are repelled by the predator particles and attracted towards the best positions of the swarm. This repulsive mechanism ensures the presence of diversity in the swarm and eliminates the phenomenon of premature convergence.

PSO has been applied for constrained non linear optimization problem [8]. Feasibility study has been used to deal with constraints, and feasibility function is used to check the satisfaction of all the constraints. Initial population is a group of feasible solutions that satisfy all the constraints. All particles keep feasible solution in their memory. The proposed modified algorithm successfully solved problems with nonlinear inequality constraints.

Brits et al [9] proposed another variant of PSO which intended to locate multiple best possible solutions in multimodal problems by using sub swarms and the convergent sub swarms algorithm. Niching algorithms find and track various solutions via fitness based principle to discover and mark particle solution. However there are still some issues that need to be solved.

By considering the particles previous best position and mistakes, Yang et al [11] proposed a new variant of PSO. The author says that the individual can learn not only from its previous best but it can improve its learning by using its mistakes. Author used 4 Benchmark functions to compare their technique with original PSO.

To share the information of particles Zhi-Feng [18] proposed a PSO with cross over operator. The crossover take place only in one dimension, which is randomly selected. In the meantime, fitness of two offspring produced by cross over is compared. Then enhanced one is selected. Crossover is done as

$$p_{it}^{'} = r\, p_{it}^{'} + (1-r)\, p_{kt} \quad \text{.....} \quad (2.29)$$

$$p_{it}^{''} = (1-r)\, p_{it} + r\, p_{kt} \quad \text{.....} \quad (2.30)$$

Where i=1,2,3……N, t is random integer in the range (1,D). $p_{it}^{''}$ is equal to p_i in any dimention except in t dimension. R is random number in the range (0, 1). K is the particle's best position equivalent to better fitness produced by i^{th} particle with the fitness-proportionate selection. This technique is proposed just to prevent the trapping of algorithm into local minima by sharing the information of particles. Five benchmark functions with two uni-models and three multi-models are used to test the algorithm.

Omran et al [26] used an opposition based learning to improve the performance of PSO. In each iteration the particle with lowest strength of fitness is replaced by its opposite, the speed and individual experience of the anti-particle are reset. After that a global best solution is updated. They did not introduced any new parameter to PSO. The only modification is the use of opposition based learning to enhance the performance of PSO.

Shahzad et al [31] proposed another variant of OPSO with velocity clamping (OVCPSO). The authors control the velocity by velocity clamping to speed up convergence and to stay away from impulsive convergence. Velocity clamping changes the search direction of particles. Linearly decreasing inertia weight between 0.4 and 0.9 has been used. The proposed algorithms has been tested on various benchmark functions and results revealed its success.

Tang et al [32] proposed an enhance opposition based PSO, called EOPSO. According to the authors opposite point is closer to global optima then current point. This provides more chances to get close to global optima. The enhanced opposition of a population is calculated based on opposition probability and best among original and enhanced population are selected for further exploration of the search space using traditional PSO. Prominent results have been achieved using the proposed modification to traditional PSO.

3. Chapter 3

In this chapter the mutation operator for PSO has been discussed. There will be total of 6 mutation operator in this chapter, 3 out of them are proposed and three are existing.

3.1. Introduction

PSO proposed by Kennedy [1] in 1995 was a simple PSO, later researchers proposed different variants of PSO, Shi [2] introduced the inertia weight to control the exploration and exploitation and Clerc [3] developed coefficient to balance the exploration-exploitation trade off, which improved the performance of PSO.

Following to these enhancements, researchers have been doing more work to improve the performance of PSO. Therefore, some researchers introduced mutation operators by using different distribution in PSO to improve its performance. Mutation is important for PSO as it helps to escape from local minima. It is different to other techniques as after mutating the g_{best} or l_{best} mutated one is compare with original and better one is replaced with original. Below we will see some of Mutations.

3.2. PSO Mutations

Many of the researchers have presented some PSO techniques, firstly these existing techniques are studied and later proposed techniques in this research are presented.

3.2.1. Cauchy Mutation

Cauchy distribution is a continuous distribution and is defined on the open interval of [0, 1]. It is a bell shaped and a symmetric distribution. Cauchy distribution is a distribution which has no mean, variance or moments.

PSO with Cauchy mutation was proposed by Wang et al [19]. The author claimed to prevent PSO from immature convergence; they proposed the new variant of PSO. They used Cauchy distribution for this purpose. Search space of the best particle can be extended by adding the searching neighbor of the global best particle in each generation. The author claimed that this can be achieved by having a Cauchy mutation on the global best particle in each generation. The Cauchy density function given by

$$f(x)=\frac{1}{\pi}\frac{t}{t^2+x^2}, -\infty<x<\infty$$ ……………………….……………………….. …. (3.1)

Where t>0 is scale parameter. The Cauchy distributed function is

$$f_t(x) = \frac{1}{2}+\frac{1}{\pi}arctn(\frac{x}{t})$$ ………………………………………………..……….. .. ……. (3.2)

This mutation operator is used to increase the probability of escaping from a local optimum. The Cauchy mutation operator is:

$$gbest_i(i)=gbest_i(i)+W(i)*N(x_{min},x_{max})$$ ………………………..…………… (3.3)

Where N is a Cauchy distributed function with scale parameter t=1, $N(x_{min},x_{max})$ is a random number with in (x_{min},x_{max}) of defined domain of test function and

$$W(i)=\left(\sum_{j=1}^{popsize}V[j][i]\right)/popsize$$ ……………………………...………………….. (3.4)

Where $V[j][i]$ is the i th velocity vector of j th particle in the population popsize is the size of population

3.2.2. Adaptive Mutation

Pant [22] proposed a variant of PSO with adaptive mutation. The author proposed two variant of PSO; AMPSO1 and AMPSO2. They worked on mutation of the global best and personal best particle of the swarm. In AMPSO1 they transformed the personal best position while in AMPSO2 they transformed the global best position of the particle. Following formula is used to mutate the particle.

$$g_{best}=g_{best}+\sigma'*Betarand()$$ …………………………………………………….. (3.5)

Where $\sigma'=\sigma*\exp(\tau N(0,1)+\tau' N_j(0,1))$, N(0,1) denotes a normally distributed function with mean zero and standard deviation one, $N_j(0,1)$ that a different random number is generated for each value of j, τ and τ' are set as $\frac{1}{\sqrt{2n}}$ and $\frac{1}{\sqrt{2\sqrt{n}}}$ respectively

and value of σ is originally set as 3.Betarand() is a random number generated by beta distribution with parameter les then 1.

Beta distribution is a continuous probability distribution that is defined on the open interval of (0, 1). It has two parameters α, β that define the shape of the distribution. The probability density function of the Beta distribution is:

$$f(x)=\frac{x^{\alpha-1}(1-x)^{\beta-1}}{B[\alpha,\beta]} \quad \text{……..} \quad (3.6)$$

Where $B[\alpha,\beta]$ is the beta function with parameter α and β given by

$$B[\alpha,\beta]=\int_0^1 x^{\alpha-1}(1-x)^{\beta-1}dx \quad \text{……... (3.7)}$$

3.2.3. Power Mutation

Power distribution is also a continuous probability distribution. It has two parameters α, β that define the shape of the distribution. The probability density function of the Beta distribution is:

$$f(x)=\alpha\beta^{\alpha}x^{\alpha-1} \quad \text{……..... (3.8)}$$

Wu [34] proposed the variant of PSO using power mutation based on power distribution. The distribution function is:

$$f(x)=P\,x^{p-1},0\le x\le 1 \quad \text{…….. (3.9)}$$

The density function is given by

$$f(x)=x^{p},0\le x\le 1 \quad \text{……...……. ….. (3.10)}$$

Where p is the index of the distribution function. The power mutation is defined as

$$x_j^{'}=\begin{cases} x_j-S\left(x_j+x^{l}\right), if t<r \\ x_j+s\left(x^{u}-x_j\right),\wedge if\, t\ge r \end{cases} \quad \text{……... (3.11)}$$

$x^l = min\{x_j\}$ and $x^u = max\{x_j\}, j = 1,2,3.....D$ ……...…… (3.12)

Where $t = \frac{(x - x^l)}{(x^u - x^l)}$ and [x^l, x^u] is the boundary of the decision variables in the current search space, r is random number between 0, 1 and s is calculated according to equation 3.10.

3.3. Proposed Techniques

3.3.1. Laplace Mutation (LMPSO)

Laplace mutation is continuous and double exponential distribution. It is the distribution of differences between two independent variables with identical exponential distributions. A random variable has a Laplace (μ, b) distribution if its probability density function is

$$F(x \vee \mu, b) = \frac{1}{2b} \exp ¿ \; -¿ \frac{x - \mu}{b} \vee ¿ \quad \text{……..……. (3.11)}$$

Where μ is a location parameter and b>0 is a scale parameter

To escape PSO from local minima, PSO with Laplace mutation (LMPSO) is proposed in this research. In this technique global best particle is mutated by Laplace mutation. By using Laplace in PSO, PSO takes a long jump to escape from local minima. In LMPSO g_{best} is mutated by following way.

$$g_{best} = \begin{cases} g_{best} + t, g_{best} is + ve \\ g_{best} - t, \wedge g_{best} is - ve \end{cases} \quad \text{……..……. (3.14)}$$

Where $t = \frac{x^l}{x^u} * L$ ……..…… (3.15)

Where x^l, x^u are the boundaries of the current search space and L is the Laplace random number generated by Laplace distribution.

3.3.2. Normal Mutation (NMPSO)

The normal distribution is a bell shaped distribution with peak at mean. This distribution is also known as gaussian distribution. It is the most commonly used distribution. This distribution is completely described by the two parameters of mean and variance. The normal distribution N(0, 1) is also called unit normal function. The probability density function of the normal distribution is

$$f(x)=\frac{e^{\frac{-1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}}{\sigma\sqrt{2\pi}} \quad \text{……..…….…. ... ...} \quad (3.16)$$

In NMPSO g_{best} is mutated using following formula

$$P_g(i)=P_g(i)+W(i)*NormalDist() \quad \text{……..…….. ..} \quad (3.17)$$

Where $W(i)$ is the average velocity of swarm, $NarmolDist()$ is used to generate the random number with Normal distribution with $\mu=1.0$ and Sigma=1.

3.3.3. Random Mutation (RMPSO)

In RMPSO g_{best} is mutated using following formula

$$P_g(i)=P_g(i)+W(i)*Random(0,1)$$

Where $W(i)$ is the average velocity of swarm and $Random(0,1)$ is used to generate the random number between 0 and 1

PSO Variants	Use Population Min	Use Population Max	Initialization Randomly
CPSO	No	No	Yes
AMPSO	No	No	Yes
PMPSO	Yes	Yes	Yes
LMPSO	Yes	Yes	Yes
NMPSO	No	No	Yes
RMPSO	No	No	Yes

3. 1 PSO Mutation Variants

The flowchart of PSO with proposed variant is shown in figure (3.1)

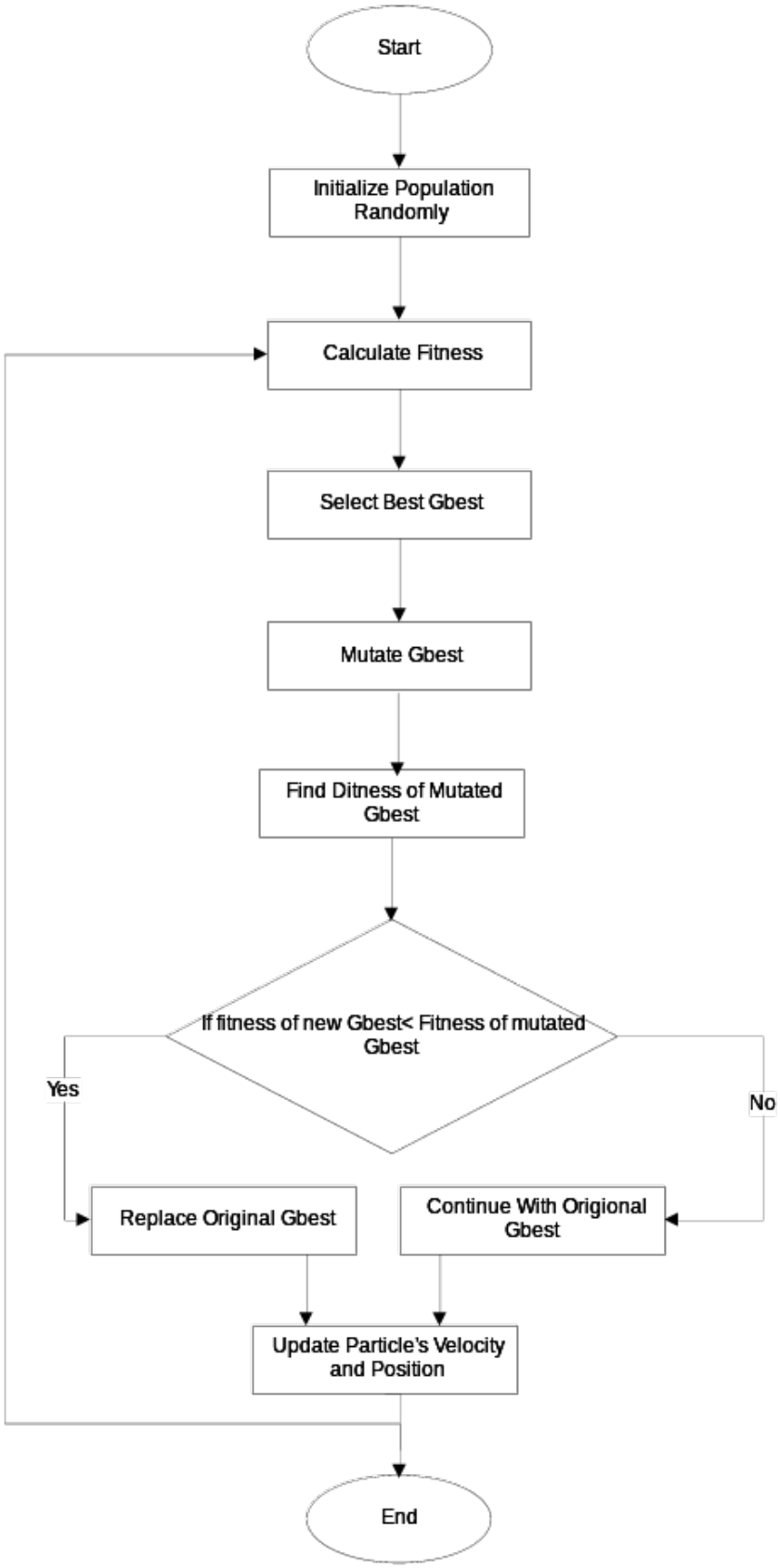


Figure 1 : Flow Chart of Proposed PSO

4. Chapter 4

In this chapter the results of some PSO techniques are compared using some standard Benchmark functions. Following paragraphs will give the description about the comparison of techniques. The analysis of these results will also discuss in this chapter.

4.1. PSO Techniques

Following techniques were used for comparison

1. PSO with Cauchy Mutation (CPSO)
2. PSO with Adaptive Mutation (AMPSO)
3. PSO with Power Mutation (PMPSO)
4. Proposed Techniques

    4.1. PSO with Laplace Mutation (LMPPSO)

    4.2. Normal Mutation (NMPSO)

    4.3. Random Mutation (RMPSO)

4.2. Experimental Setting

Search space boundary is [-100,100] population size kept 30 with 30 number of runs. 10, 20 and 30 dimensions are used for 1000, 1500 and 2000 iterations respectively.

Parameter	**Value**
Search Space	[100,-100]
Dimensions	10
	20
	30

Iterations	1000
	1500
	2000
Population size	30
Number of PSO Runs	30

Table 4.1 Experimental setting of parameters

For some functions, the dimensions and iterations are changed which are listed in table 4.1.

Function	**Dimension**	**Iterations**
f_8	2	100
f_9	2	100
f_{13}	2	50
f_{14}	2	50
f_{16}	2	100
f_{17}	2	1000
f_{18}	2	100

Table 4.2 Functions with Different Dimensions

Following benchmark functions are used to generate the results for optimization problem.

1. $f_1(x)=\sum_{i=0}^{n} x_i^2$

2. $f_2(x)=\sum_{i=0}^{n} i*x_i^2$

3. $f_3(x)=\sum_{i=0}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$

4. $f_4(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$

5. $f_5(x)=\sum_{i=1}^{n}\left[100\left(x_{i+1}-x_i^{\,2}\right)^2+(1-x_i^2)^2\right]$

6. ¿ $x_i \vee$ ¿
   $-1\sqrt{¿}$
   ¿
   $f_6(x)=\sum_{i=1}^{n} -x_i*\sin$ ¿

7. $f_7(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)+20+e$

8. $f_8(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_2-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$

9. $f_9(x)=-\cos(x_1)-\cos(x_2)\exp\left(-(x_1-\pi)^2-(x_2-\pi)^2\right)$

10. $f_{10}(x)=max|x_i|,0\le i\le n$

11. $f_{11}(x)=\frac{\pi}{n}\left\{10\sin^2(\pi y_i)+\sum_{i=1}^{n}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_n-1)^2\right\}+\sum_{i=1}^{n} u(x_i,10,100,4)$,

    a. $y_i=1+\frac{1}{4}(x_i+1)$

    b. $u(x_i,a,k,m)=\begin{cases} x \\ (¿¿i-a)^m, x_i>a_i, \\ k¿0, -a<x_i<a, \\ k(-x_i-a)^m, \end{cases}$

12. $$x$$
$$(¿¿i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right]+(x_n-1)\left[1+\sin^2(2\pi x_n)\right]$$
$$\sin^2(3\pi x_i)+\sum_{i=1}^{n-1} ¿$$
$$¿$$
$$f_{12}(x)=.1¿$$

13. $$f_{13}(x)=\left[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right]^{-1}$$

14. $$f_{14}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6x_1x_2-4x_2^2+4x_2^4$$

15. $$\left|x_i\right|+\prod_{i=1}^{n} ¿x_i\vee¿$$
$$f_{15}(x)=\sum_{i=1}^{n} ¿$$

16. $$x$$
$$(¿¿1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+16x_1x_2+3x_2^2)$$
$$1+¿$$
$$¿$$
$$f_{16}(x)=¿$$

17. $$f_{17}(x)=\frac{-1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{\frac{1}{2}\left(x_1^2+x_2^2\right)+2}$$

18. $$f_{18}(x)=(4-2.1x_1^2+\frac{x_1^4}{3})x_1^2+x_1x_2+(-4+4x_2^2)x_2^2$$

19. $$f_{19}(x)=\sum_{i=1}^{n} \lfloor x_i+1\rfloor^2$$

20. $$f_{20}=\sum_{i=1}^{n} x_i^2+\left(\sum_{i=1}^{n} 0.5ix_i\right)^2+\left(\sum_{i=1}^{n} 0.5ix_i\right)^4$$

21. $$f_{21}=\sum_{i=1}^{n}\left|x_i\sin\left(x_i\right)+0.1x_i\right|$$

22. $$f_{22}=0.5+\frac{\sin^2\sqrt{\left(x_1^2+x_2^2\right)}-0.5}{1+0.01\left(x_1^2+x_2^2\right)^2}$$

4.3. Test Functions

In this section we will present the result and description of some benchmark functions which are commonly used in the literature.

4.3.1. De Jong's Function

De Jong's Function is the simplest test function also known as sphere model. It is continues, convex and uni-model function. It is called the first function of the De Jong's. General definition of the function is given below.

$$f_1(x)=\sum_{i=0}^{n} x_i^2 \qquad \text{..... (4.8)}$$

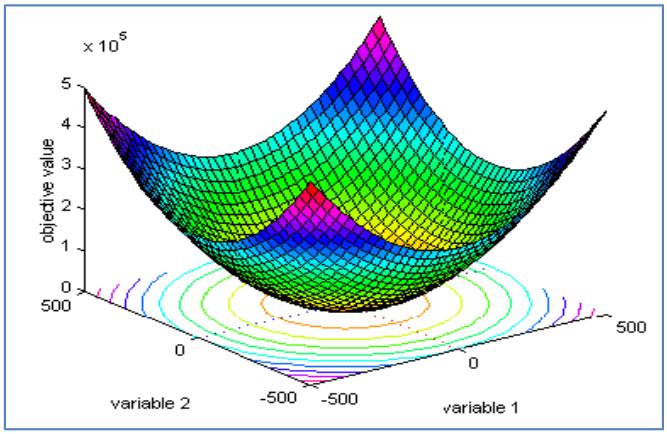


Figure 2: A Graphical Overview of f_1 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	6.91E-57	1.09E-64	1.65E-52	2.05E-96	**2.00E-97**	7.28E-58	1.08E-52
20	1500	2.22E-17	3.23E-25	1.37E-17	2.73E-81	**4.25E-85**	9.68E-21	5.72E-17
30	2000	1.54E-07	4.13E-13	2.34E-09	1.05E-63	**1.65E-78**	2.99E-11	1.00E-07

Table 4.3 Result of f_1

In this table dim represent the dimensions used in PSO, PSO is the traditional PSO, CPSO is Cauchy Mutation PSO, AMPSO is Adaptive Mutation PSO, PMPSO is Power Mutation PSO, LMPSO is Laplace Mutation PSO, NMPSO is Normal Mutation PSO, RRMPSO is Random Mutation PSO. Avg fitness is the average fitness of best 20 runs out of 30 runs and avg mutation is the mutation rate. Similar pattern is used in all tables of this chapter.

4.3.2. Axis Parallel Hyper-ellipsoid Function

It is similar to the function of De jong. It is also a convex, continues and uni model function. It is called weighted sphere model. Function general definition is given as

$$f_2(x)=\sum_{i=0}^{n} i*x_i^2 \quad \text{..... (4.9)}$$

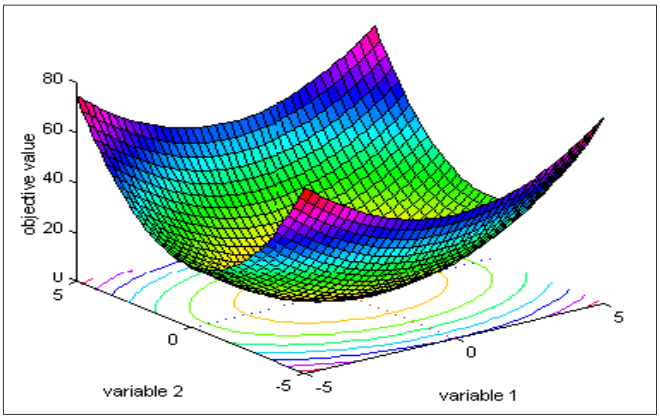


Figure 3: A Graphical Overview of f_2 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	1.21E-64	2.98E-71	5.05E-68	**1.10E-94**	5.85E-89	7.46E-71	2.43E-65
20	1500	1.29E-06	3.42E-13	1.70E-17	2.15E-57	**3.09E-70**	1.22E-21	2.18E-18
30	2000	4.19E-06	8.82E-14	6.79E-09	3.58E-27	**1.23E-51**	6.69E-11	2.02E-06

Table 4.4 Result of f_2

4.3.3. Rastrigin's Function

This function is based on the function of de jong with addition of cosine modulation to produce many local minima and thus it is a multi-model function. The function definition is

$$f_3(x)=\sum_{i=0}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right] \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(4.10)}$$

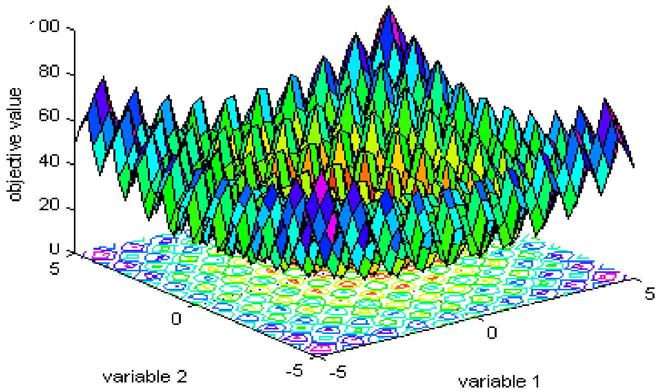


Figure 4: Graphical Overview of f_3 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	5.68E+00	6.82E+0	6.42E+0	4.78E+00	**2.84E+0**	5.18E+00	5.58E+00
20	1500	3.29E+01	3.87E+1	3.84E+1	1.55E+01	**1.29E+1**	3.39E+01	9.21E+01
30	2000	9.37E+01	1.01E+2	1.12E+2	5.06E+1	**2.03E+1**	9.82E+1	9.20E+1

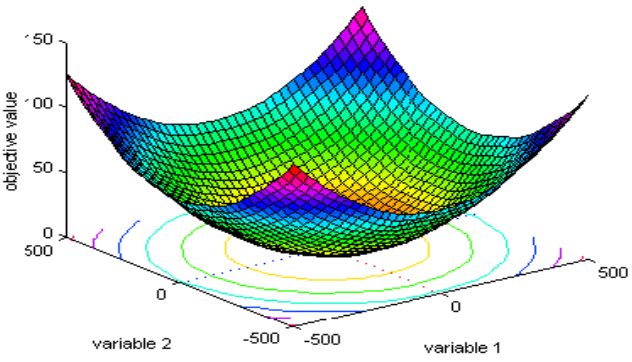
Table 4.5 Result of f_3

4.3.4. Griewangk's function

This function is similar to Rastrigin's function. It has many widespread local minima, while the locations of minima are regularly distributed. The function definition is

$$f_4(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(4.11)}$$

Figure 5: Graphical Overview of f_4 [41]



Dim	Iteration	PS Avg Fitn	NMPSO Avg Fitness	RMPSO Avg Fitness
10	1000	.99	.99975	.9997
20	1500	.99	.99975	.9997
30	2000	.99	.99975	.9997

Table 4.6 Result of function f_4

4.3.5. Rosenbrock's Valley

This function is known as banana function or the second function of de jong. It is non convex function introduced by Rosenbrock in 1960. The global minimum is inside a long, narrow, parabolic shaped flat valley. The function definition is

$$f_5(x)=\sum_{i=1}^{n}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(1-x_i^2\right)^2\right] \quad \text{..... (4.12)}$$

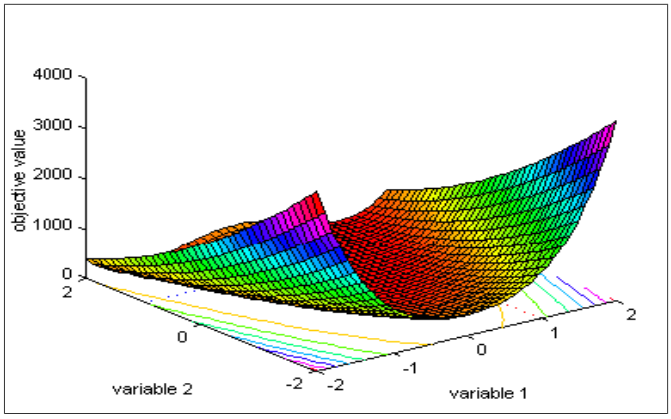


Figure 6: Graphical Overview of f_5 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	2.27E+0	2.68E+0	1.20E+00	**3.38E-03**	4.88E+00	2.53E+00	1.67E+00
20	1500	1.32E+1	1.23E+1	**9.49E+0**	3.94E+01	4.36E+01	1.18E+01	1.91E+01
30	2000	2.92E+1	**2.06E+1**	4.16E+01	6.57E+01	6.15E+01	2.30E+01	4.52E+01

Table 4.7 Result of f_5

4.3.6. Schwefel's Function

Schwefel's function is deceptive in that the global minimum is geometrically distant over the parameter space from the next best local minima. Therefore the search algorithms are potentially prone to convergence in the wrong direction. The function definition is

$$f_6(x)=\sum_{i=1}^{n} -x_i * \sin\left(\sqrt{|x_i|}\right) \qquad \text{..... (4.13)}$$

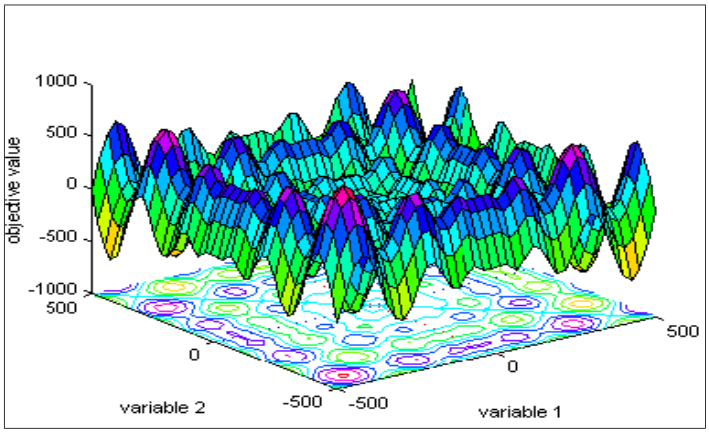


Figure 7: Graphical Overview function of f_6 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	50	**4.99E+4**	1.99E+05	1.47E+05	1.09E+05	7.46E+04	2.87E+05	5.15E+04
20	80	7.95E+6	3.82E+12	2.18E+06	**5.02E+5**	2.04E+06	1.15E+08	3.89E+06
30	100	4.68E+7	1.23E+15	3.59E+07	**4.90E+6**	1.78E+07	2.59E+08	5.19E+07

Table 4.8 Result of function f_6

4.3.7. Ackley's function

It is a multimodal function which is widely used. Originally this problem was defined for two dimensions, but the problem has been generalized to *N* dimensions. Its definition is given below.

$$f_7(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + e \quad \text{…...……. (4.14)}$$

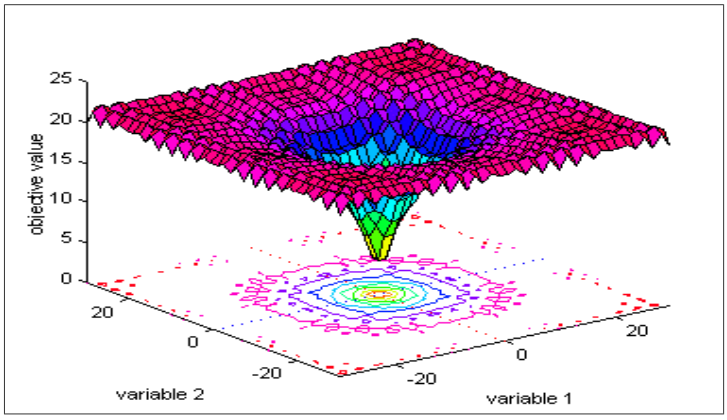


Figure 8: Graphical Overview of f_7 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	50	1.87E+1	1.78E+01	1.84E+01	1.13E+01	**3.31E-01**	1.88E+01	1.66E+01
20	80	2.00E+1	2.00E+01	2.00E+01	2.00E+01	**2.51E+0**	2.00E+01	2.00E+01
30	100	2.00E+1	2.00E+01	2.00E+01	2.00E+01	**3.20E+0**	2.00E+01	2.00E+01

Table 4.9 Result of function f_7

4.3.8. Branins's Function

It is a global optimization test function, which have two variables, containing three equally global optima, the function definition is given below.

$$f_8(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_2 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10 \quad \text{……..…….... (4.15)}$$

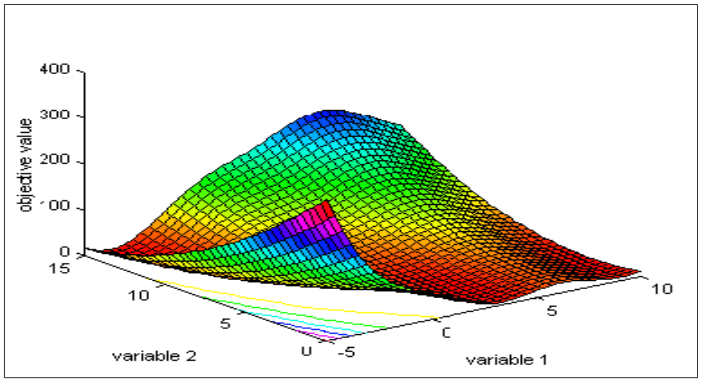


Figure 9: Graphical Overview of function f_8 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	100	**3.98E-1**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**

Table 4.10 Result of function f_8

4.3.9. Easom's Function

It is a uni-model test function where the global minimum has a small area relative to the search space; the function was inverted for minimization, function definition is below.

$$f_9(x) = -\cos(x_1)\cos(x_2)\exp\left(-(x_1-\pi)^2-(x_2-\pi)^2\right) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.16)$$

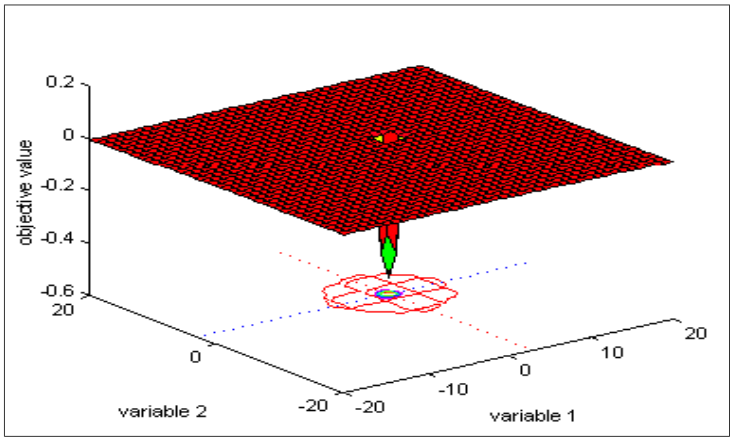


Figure 10: Graphical Overview of function f_9 [41]

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	100	**1.03E+0**	**1.03E+0**	**1.03E+0**	**1.03E+0**	**1.03E+0**	**7.61E-1**	**6.19E-1**

Table 4.11 Result of function f_9

4.3.10. Other Functions

$$f_{10}(x) = max|x_i|, 0 \le i \le n \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.17)$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	3.91E-12	8.38E-18	7.97E-12	4.83E-16	1.54E-18	**0**	**0**
20	1500	3.09E-01	2.66E-02	3.91E-01	1.09E+0	4.85E-06	**0**	**0**
30	2000	9.26E+00	4.11E+00	9.33E+0	1.51E+1	1.75E-04	**0**	**0**

Table 4.12 Result of function f_{10}

$$f_{11}(x) = \frac{\pi}{n}\left\{\begin{array}{c} 10\sin^2(\pi y_i) + \sum_{i=1}^{n}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right] \\ +(y_n-1)^2 \end{array}\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4) \quad \ldots. \quad (4.18)$$

$$y_i = 1+\frac{1}{4}(x_i+1)$$

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a_i, \\ k\,0, -a < x_i < a, \\ k(-x_i - a)^m, \end{cases}$$

Dim	**Itera tion**	**PSO**	**CPSO**	**AMPSO**	**PMPSO**	**LMPSO**	**NMPSO**	**RMPSO**
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	3.14E-01	3.14E-01	3.14E-01	3.14E-01	3.14E-01	**0**	**0**
20	1500	1.57E-01	1.57E-01	1.42E-01	1.18E-01	1.57E-01	**0**	**0**
30	2000	3.55E-02	9.44E-02	5.26E-02	8.19E-02	1.02E-01	**0**	**0**

Table 2.13 Result of function f_{11}

$$f_{12}(x) = .1\left\{ \sin^2(3\pi x_i) + \sum_{i=1}^{n-1} (x_i - 1)^2\left[1 + \sin^2(3\pi x_{i+1})\right] + (x_n - 1)\left[1 + \sin^2(2\pi x_n)\right] \right\}$$

. ……………………………………... ……………………………….. (4.19)

Dim	**Iterati on**	**PSO**	**CPSO**	**AMPSO**	**PMPSO**	**LMPSO**	**NMPSO**	**RMPSO**
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	8.34E+02	3.48E+11	4.73E+02	4.79E+00	1.62E+00	**0**	**0**
20	1500	1.42E+02	2.57E+07	2.07E+03	1.39E-01	8.49E-01	**0**	**0**
30	2000	1.35E+02	1.17E+02	1.87E+02	2.51E+00	9.76E-01	**0**	**0**

Table 4.14 Result of function f_{12}

$$f_{13}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right]^{-1} \quad \text{……………………………………….……………….. (4.20)}$$

Dim	Iterati on	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	50	4.77E-27	1.07E-25	3.50E-26	2.62E-15	1.38E-18	0	0

Table 4.15 Result of function f_{13}

$$f_{14}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 x_1 x_2 - 4x_2^2 + 4x_2^4 \quad \text{……………………...……………… (4.20)}$$

Dim	**Iterati on**	**PSO**	**CPSO**	**AMPSO**	**PMPSO**	**LMPSO**	**NMPSO**	**RMPSO**
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	50	**1.03E+0**	**1.03E+00**	**1.03E+0**	**1.03E+00**	**1.03E+0**	**1.03E+0**	**1.03E+0**

Table 4.16 Result of function f_{14}

$$\left|x_i\right|+\prod_{i=1}^{n} ¿\, x_i \vee ¿$$
$$f_{15}(x)=\sum_{i=1}^{n} ¿ \qquad \text{..... (4.21)}$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	50	2.07E-20	1.64E-27	4.80E-19	3.97E-50	**6.30E-51**	2.56E-23	9.79E-19
20	80	4.62E-05	2.84E-08	3.17E-05	2.30E-50	**5.84E-55**	2.81E-06	3.04E-05
30	100	5.80E-0	7.72E-04	8.25E-03	4.99E-45	**1.12E-50**	2.45E-03	3.71E-02

Table 4.17 Result of f_{15}

$$x$$
$$(¿¿1+x_2+1)^2(19-14\,x_1+3\,x_1^2-14\,x_2+16\,x_1\,x_2+3\,x_2^2)$$
$$1+¿$$
$$¿$$
$$f_{16}(x)=¿ \qquad \text{..... (4.22)}$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	100	**3.00**	**3.00**	**3.00**	**3.00E+**	**3.00**	**0**	**0**

Table 4.18 Result of function f_{16}

$$f_{17}(x)=\frac{-1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{\frac{1}{2}\left(x_1^2+x_2^2\right)+2} \qquad \text{.....}$$

(4.23)

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	100	1.96E-13	3.49E-13	9.14E-15	2.48E-13	4.81E-13	**0**	**0**

Table 4.19 Result of function f_{17}

$$f_{18}(x)=(4-2.1\,x_1^2+\frac{x_1^4}{3})\,x_1^2+x_1\,x_2+(-4+4\,x_2^2)\,x_2^2 \qquad \text{..... (4.24)}$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	1000	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**

Table 4.20 Result of function f_{18}

$$f_{19}(x)=\sum_{i=1}^{n} \lfloor x_i+1 \rfloor^2 \qquad \text{..... (4.25)}$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	50	0	0	0	0	0	**0**	**0**
20	80	3.66E-16	6.76E-24	4.23E-17	0	0	**0**	**0**

30	100	2.77E-07	7.15E-13	2.07E-09	0	0	**0**	**0**

Table 4.21 Result of function f_{19}

$$f_{20}=\sum_{i=1}^{n} x_i^2+\left(\sum_{i=1}^{n} 0.5\,i\,x_i\right)^2+\left(\sum_{i=1}^{n} 0.5\,i\,x_i\right)^4 \quad \text{..... (4.26)}$$

Dim	**Iteration**	**PSO**	**CPSO**	**AMPSO**	**PMPSO**	**LMPSO**	**NMPSO**	**RMPSO**
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	2.172E-14	1.45E-20	8.586E-16	8.227E-45	**3.203E-47**	1.189E-17	9.5142E-14
20	1500	2.78E+02	9.34E+01	3.45E+02	3.19E+00	**2.23E-02**	1.31E+02	3.50E+02
30	2000	2.71E+03	1.48E+03	1.71E+03	2.56E+02	**6.15E+02**	2.35E+03	2.93E+03

Table 4.22 Result of function f_{20}

$$f_{21}=\sum_{i=1}^{n}\left|x_i\sin\left(x_i\right)+0.1x_i\right| \quad \text{..... (4.27)}$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
10	1000	9.79E-07	5.74E-07	6.05E-08	1.91E-09	**4.86E-15**	1.17E-07	2.79E-07
20	1500	4.58E-01	1.74E-01	1.61E+00	4.03E-02	**1.21E-10**	1.11E-01	4.75E-01
30	2000	3.15E+00	3.60E+00	9.21E+00	2.92E+00	**3.79E-14**	7.11E+00	5.91E+00

Table 4.23 Result of function f_{21}

$$f_{22}=0.5+\frac{\sin^2\sqrt{\left(x_1^2+x_2^2\right)}-0.5}{1+0.01\left(x_1^2+x_2^2\right)^2} \quad \text{.....} \quad (4.28)$$

Dim	Iteration	PSO	CPSO	AMPSO	PMPSO	LMPSO	NMPSO	RMPSO
		Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness	Avg Fitness
2	100	2.17E-14	1.46E-20	8.59E-16	8.23E-45	**3.20E-47**	1.19E-17	9.51E-14
2	400	3.82E-15	5.71E-10	1.38E-11	1.33E-13	**6.11E-17**	7.78E-15	5.48E-12
2	800	1.30E-15	8.30E-13	2.36E-12	1.28E-16	**0**	3.03E-12	5.53E-14

Table 4.23 Result of function f_{22}

4.4. Detail Results

The detail results are available on next page.

4.5. Analysis

Detail Results of all the benchmark functions are given in table 4.22. The results of De Jong's Function show that LMPSO has better performance as compared to other techniques.
In axis parallel hyper-ellipsoid function, when 10 dimensions with 1000 iterations are used then PMPSO performance is best, whereas by increasing the dimensions and iterations, LMPSO performance can be boost up as compare to other techniques.
Rastrigin's function's results show that LMPSO has better performance than other techniques.
By Result of Griewangk's function we can infer that all techniques have equal performance in all three cases.
The results of Rosenbrock's Valley function shows that PMPSO has better performance in case of 10 dimensions and 1000 iterations, while AMPSO perform better in case of 20 dimensions and 1500 iterations and CPSO perform better in case of 30 dimensions and 2000 iterations.

In case of Schwefel's Function traditional PSO remains good then other techniques when dimensions are 10 and iterations are 50. But if we increase dimensions to 20, 30 and iterations to 80,100 iterations respectively, PMPSO's performance is better. LMPSO's performance was on the second number.
The performance of LMPSO is better than other techniques for Ackley's function in all three cases.
The result of Branins's function remains same for all variants.
The performances of all techniques remain same for Easom's function.
Results of $f10$ show that the performance of LMPSO is better than all other techniques.
For $f11$ the performance of all techniques remain same when dimensions are 10 and iterations are 1000, while in case of 20 dimension and 1500 iterations the performance of RMPSO is good than other techniques and in case of 30 dimensions and 2000 iterations performance of all techniques is almost same.
The performance of LMPSO is good in case of 10 dimensions and 1000 iterations, while in case of 20 dimensions and 1500 iterations performance of PMPSO is good, and again LMPSO re-sustain its performance in case of 30 dimensions and 2000 iterations.
For function $f13$ the performance of traditional PSO is better than all other techniques.
The Performance of all the techniques is same for function $f14$.
From results of function $f15$ it can be seen that Performance of LMPSO remains best in all three cases while PMPSO was the second best technique during this function.
All the techniques have same performance for function $f16$.
The performance of AMPSO is better than all other techniques for function $f17$
For function $f18$ performance of all techniques remains same.
The performance of LMPSO remains well for functions 19 $f20$, $f20$ and $f21$

5. Chapter 5

This chapter consists of conclusion and future work

5.1. Conclusion

From the results that are given in the chapter 4, we observed that in more than 50% cases the performance of LMPSO is better than other techniques. Power mutation also remains good in some cases. CPSO and AMPSO have performed better only in few cases. The performance of each technique remains same for function f 4, f9, f 14, f 16 and f 18.

The performance of LMPSO remains well as it uses some statistics of search space while mutating the g_{best}. The result of power mutation is well from CPSO and AMPSO because they don't use the statistics of search space while mutating the g_{best}.

The functions where performance of all techniques remains same all are simple 2 dimension functions due to which the performance of all techniques remains same.

5.2. Future Work

Future work of this paper may be to implement all the proposed techniques by using the all available benchmark function to generalize the conclusion about the proposed techniques.

References

[1] J. Kennedy and R. Eberhart. "Particle Swarm Optimization," In Proceedings of IEEE International Conference on Neural Networks, 1995, PP.1942-1948.

[2] Y.Shi and R. Eberhart. "A modified particle swarm optimizer," In Proceedings of the IEEE international conference on evolutionary computation, 1998, pp. 69–73.

[3] M. Clerc. "The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization," In Proceedings of the IEEE Congress on Evolutionary Computation, 1999, PP. 1951-1957.

[4] Y. Shi and R. C. Eberhart, "Fuzzy Adaptive particle Swarm Optimization", In Proceedings of the IEEE Congress on Evolutionary Computation, 2001, pp. 101-106.

[5] M. Lovbjerg, T.K. Rasmussen and T. Krink. “Hybrid Particle Optimizer with breeding and subpopulation,” Proceedings of the Genetic and Evolutionary Computation Conference, 2001.

[6] S. Naka, T. Genji, T. Yura, and Y. Fukuyama. “Practical Distribution State Estimation using Hybrid Particle Swarm Optimization.” In Proceedings IEEE Power Engineering Society Winter Meeting, 2001, pp 815-820.

[7] A. Silva, A. Neves and E.Costa. “Chasing The Swarm: A Predator Pray Approach to Function Optimization,” In Proceedings of MENDEL 8th International Conference on Soft Computing, 2002.

[8] X. Hu and R. Eberhart. “Solving Constraint non-linear optimization problems with Partilcle Swarm Optimization,” 6th World Multiconference on Systemics, Cybernetics and Informatics, 2002.

[9] R. Brits .A.P. Engelbrecht, F. v. Den Bergh. “A Niching Particle Swarm Optimizer,” In Proceedings of the Conference on Simulated Evolution and Learning, 2002.

[10] L. Zhang, H. Yu, and S. Hu. (2003). “A new approach to improve particle swarm optimization,” Proceedings of the 2003 international conference on Genetic and evolutionary computation, 2003, pp134-139.

[11] C. Yang and D. Simon. (2005). “A New Particle Swarm Optimization Technique,” Proceedings of the 18th International Conference on Systems Engineering, 2005, pp 164-169.

[12] A.P. Engelbrecht, Fundamentals of Computational Swarm intelligence, England: John Wily and Sons Ltd, 2005.

[13] J. Wei and Y. Wang. (2006). “A Dynamical Particle Swarm Algorithm with Dimension Mutation,” IJCSNS International Journal of Computer Science and Network Security, Vol. 6, pp.221-224, July 2006.

[14] N. Q. Uy, N. X. Hoai, R. McKay and P. M. Tuan. “Initializing PSO with randomized low-discrepancy sequences: the comparative results,” In Proceedings of the IEEE Congress on Evolutionary Computation, 2007, pp 1985-1992.

[15] C. Li et al (2007). “A Fast Particle Swarm Optimization Algorithm with Cauchy Mutation and Natural Selection Strategy,” Advances in Computation and Intelligence Lecture Notes in Computer Science, Vol. 4683, pp 334-343, 2007.

[16] M. Pant and T. Thangaraj, V.P. Singh. “Particle Swarm Optimization Using Gaussian Inertia Weight,” International Conference on Computational Intelligence and Multimedia Applications, 2007, pp. 97-102.

[17] S. Kai, S. Fan and J. M. Chang. “A Modified Particle Swarm Optimizer Using an Adaptive Dynamic Weight Scheme,” Proceedings of the 1st international conference on Digital human modeling. 2007, pp. 56-65.

[18] Z. HAO and Z. W. “A Particle Swarm Optimization Algorithm with Crossover Operator,” Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, 2007, 1036 – 1040.

[19] H, Wang et al. “A Hybrid Particle Swarm Algorithm with Cauchy Mutation,” Proceeding of IEEE Swarm Intelligence Symposium, 2007, pp 356 - 360

[20]H. Wang et al. “Opposition-based Particle Swarm Algorithm with Cauchy Mutation,” IEEE Congress on Evolutionary Computation, 2007, pp 4750 – 4756.

[21] R. Poli, J. Kennedy and T. Blackwell. “Particle swarm optimization an overview,” Swarm Intelligence, 2007, pp 33-57.

[22] M. Pant, R. Thangaraj, and A. Abraham. (2008). “Particle Swarm Optimization Using Adaptive Mutation,” 19th International Conference on Database and Expert Systems Application, 2008, pp. 519-523.

[23] M. Pant, R. Thangaraj, C. Grosan, and A. Abraham .(2008). “Improved Particle Swarm Optimization with Low-Discrepancy Sequences,” IEEE Cong. on Evolutionary Computation, 2008, pp 3011 – 3018.

[24] M. Pant, R. Thangaraj, V.P. Singh, and A. Abraham. (2008). “Particle Swarm Optimization Using Sobol Mutation,” First International Conference on Emerging Trends in Engineering and Technology, 2008, pp. 367-372.

[25] C. Li, S. Yang, and I. A. Korejo. “An adaptive mutation operator for particle swarm optimization,” Proceeding of the 2008 UK Workshop on Computational Intelligence, 2008, pp. 165-170.

[26] M. G. H. Omran and S.al-Sharhan. “Using Opposition-based Learning to improve the Performance of Particle Swarm Optimization.” IEEE Swarm Intelligence Symposium , 2008, pp 1 – 6.

[27] X. Liu et al (2009). “Particle Swarm Optimization with Dynamic Inertia Weight and Mutation,” Third International Conference on Genetic and Evolutionary Computing, 2009, pp 620-623.

[28] Y. Gao and Y. Duan. “A New Particle Swarm Optimization Algorithm with Adaptive Mutation Operator,” Second International Conference on Information and Computing Science, 2009, pp. 58-61.

[29] H-R LI and Y-L Gao. “Particle swarm optimization algorithm with exponent decreasing inertia weight and stochastic mutation,” Second International Conference on Information and Computing Science, 2009, pp. 66-69.

[30] H. Jabeen, Z. Jalil and A.R Baig “Opposition Based Initialization in Particle Swarm Optimization,” Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, 2009, pp 2047-2052.

[31] F. Shahzad et al. “Opposition Based Particle Swarm Optimization with velocity clamping,”advances in computational intelligence, 2009, pp 339-348.

[32] J.Tang and X.Zhao (2009).”An Enhanced Opposition-based Particle Swarm Optimization” Proceedings of the 2009 WRI Global Congress on Intelligent Systems, 2009, pp 149 – 153.

[33] C. Zhang et al, “A Novel Swarm Model With Quasi-Oppositional Particle,” International Forum on Information Technology and Applications, 2009, pp 325 – 330.

[34] X. Wu and M. Zhong. “Particle Swarm Optimization Based on Power Mutation,” ISECS International Colloquium on Computing, Communication, Control, and Management, 2009, pp 464 – 467.

[35] M. Pant, R. Thangaraj. “Sobol Mutated Quantum Particle Swarm Optimization,” International Journal of Recent Trends in Engineering, Vol. 1, 2009, pp 95-99.

[36] J. Tang and X. Zhao. “Particle Swarm Optimization with Adaptive Mutation,” WASE International Conference on Information Engineering, 2009, pp 234-237.

[37]http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_cauchy.htm

[38] Clerc M and Kennedy J. “The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space,” IEEE transactions on evolutionary computation, vol. 6,2002 ,pp 58–73.

[39] N. Higashi, H. Iba.“Particle swarm optimization with Gaussian mutation,” Proceedings of the IEEE Swarm Intelligence Symposium,2003, 72 – 79.

[40] L. Zhen-su, H. Zhi-rong and D.Juan.(2006). “Particle Swarm Optimization with Adaptive Mutation,” Frontiers of electrical and electronic engineering,VOL. 1, 2006, pp 99-104

[41]H. Pohlheim GEATbx (www.geatbx.com) December 2006

[42] Xin Yao, Yong Liu, “Evolutionary programming made faster”, IEEE Trans. On Evolutionary Computation, vol.3, no3, July 1999:82-102.

[43] Shi Y, Eberhart R, “A Modified Particle Swarm Optimizer” [C]. IEEE Int. Conf. on Evolutionary Computation, Piscataway: NJ, IEEE Service Center, 1998, 69-73.

[44] Y.Shi and R. Eberhart. “Empirical study of Particle Swarm Optimization,” Proceedings of the 1999 Congress on evolutionary computation, 1999.

[45]http://en.wikipedia.org/wiki/Optimization_problem November 2010

[46] J. Blondon, “Particle Swarm Optimiztion: A tutorial,” September2009.

[47] F. van den Bergh. “An Analysis of Particle Swarm Optimizers,” PhD thesis Department of Computer Science, University of Pretoria, South Africa, 2002.

[48] X. Yao, Y. Liu and G. Lin. “Evolutionary Programming Made Faster,” IEEE Transactions on Evolutionary Computation, 199, pp 82 – 102.