Hybrid PSO Algorithm for the Solution of Learning-based Real-Parameter Single Objective Optimization Problems

Batoul Abdulmoti Holoubi

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
January 2018
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Assoc. Prof. Dr. Ali Hakan Ulusoy
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Asst. Prof. Dr. Ahmet Ünveren
Supervisor

Examining Committee

1. Asst. Prof. Dr. Adnan Acan
2. Asst. Prof. Dr. Mehtap Köse Ulukök
3. Asst. Prof. Dr. Ahmet Ünveren

ABSTRACT

During the past 20 years, the community of science have become more interested in Evolutionary Algorithms which have been used in many applications. This thesis proposes hybridized Particle Swarm Optimization (PSO) algorithm that targets to combine the original PSO with a simple local search technique (HPSO-FminLS). FminLS, have been used as a simple local search with original PSO for solving Learning-based-Real-Parameter Single Objective Optimization Problems (LbRPSOOP). These problems are provided in CEC2015 Congress on Evolutionary Computation. Technically, we solved CEC15 in dimensions D10, D30, D50 with HPSO-FminLS then developed 4 different versions by using local search and PSO algorithms. HPSO-FminLS reached optimal solution in Unimodal problems, and the near optimal solution in other problems.

Keywords: Evolutionary Algorithms, Local search, Single Objective Problems.

ÖZ

Son 20 yılda, Bilim Topluluğu, birçok uygulamada kullanılan Metaheuristik yöntemler olarak kullanılan Evrim Algoritmalarına daha fazla ilgi duydu. Bu tez, orijinal Parçacık Sürüsü Optimizasyonu'nu (PSO) basit bir yerel arama tekniği ile birleştirmeyi hedefleyen melezleştirilmiş HPSO-FminLS algoritmasını öneriyor. FminLS, Öğrenme Tabanlı Gerçek Parametre Tek Hedefli Optimizasyon Problemlerini (LbRPSOOP) çözmek için orijinal PSO ile basit bir yerel arama olarak kullanılmıştır. Kullanılan problemler, CEC2015 Evrimsel Hesaplama Kongresi'nden sağlanmaktadır. Teknik olarak, HPSO-FminLS ile üç farklı boyutta, 10, 30 ve 50, CEC15'de verilen problemler, yerel arama ve PSO algoritmaları kullanarak 4 farklı versiyon ile çözülmüşlerdir. HPSO-FminLS, Unimodal problemlerde en iyi çözüme, diğer problemlerde ise en iyi çözüme kabuledilir bir yakınlıkta ulaşmıştır.

Anahrat Kelimeler: Evrimsel Algoritmalar, Yerel Arama, Tek amaçlı eniyileme problemleri.

To my father,Dr.Abdulmoti Holoubi, who gave me confidence and support all the time without hesitation with all he has.

To my Mother, Ghlia Maktabi, who guided and encouraged me with love all the time.

ACKNOWLEDGMENT

I express my deepest gratitude to my supervisor Asst. Prof. Dr. Ahmet Ünveren who guided with encouragement during our work on this thesis. I am also grateful to jury member Asst. Prof. Dr. Adnan Acan for supporting and advising me with my studies in Computer Engineering Department, also I would like to thank jury member Asst. Prof. Dr. Mehtap Köse Ulukök for her feedback.

I would like to thank all my instructors in Computer Engineering department, especially the computer engineering department graduate committee chair Assoc. Prof. Dr. Önsen Toygar for her patience and support.

I would like to thank Mr. Mehmet Topal for helping me in the experimental part of my study, and to all the members of Computer Engineering Department.

I am extremely grateful to the members of Foreign Languages and English Preparatory School Asst. Prof. Dr. Nilgün Hancioğlu, Assist. Prof. Dr. Hicran Bayraktaroğlu Fırat, and Instr. Nurcan Garıp for helping me with my Academic English Language.

I would like to express my sincere gratitude to my first sister Mariam Holoubi, who participated in all the events of this study, for advising, helping, supporting, and evaluating me all the time. I really enjoyed our time together.

To my family members Ibrahim, Ismail, and Tasneem Thank you for helping me and being with me.

To my lovely friends Basma and Zena for supporting and helping me through the past years of our studies of the master, and to my other friends I would like to thank you all for every thing you have done for me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABC	Artificial Bee Colony
ABC-X-LS	Generalized Artificial Bee Colony algorithm
ACO	Ant Colony Optimization
CEC	Congress on Evolutionary Computation
DE	Differential Evolution
DEsPA	Differential Evolution with Success-Based Parameter Adaptation
EAs	Evolutionary Algorithms
FminLS	Fmin Local Search
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
hCC	hybrid Copperative Co-evolution
HPSO	Hybrid Particle Swarm Optimization
ILS	Iterated Local Search
LS	Local Search
LSHADE-ND	Differential Evolution with Linear Population Size Reduction working with Neuro-Dynamic
PSO	Particle Swarm Optimization
SA	Simulated Annealing
sDMS-PSO	Self Adaptive Dynamic Multi-Swarm Particle Swarm Optimizer
VNS	Variable Neighborhood Search

Chapter 1

INTRODUCTION

1.1 Background to the study

1.1.1 History of Metaheuristics:

Evolutionary Algorithms (EAs) are metaheuristics methods that represent a type of stochastic algorithms which go through logical steps for solving linear or nonlinear optimization problems. On the whole, there are two types of evolutionary algorithms; heuristics and metaheuristics. According to the literature, the word heuristic comes from the Greek "heuriskein" meaning "to discover" pertains to the process of gaining knowledge or some desired result by intelligent guesswork rather than by following some preestablished formula [1], while Meta comes from the Greek language with the meaning “higher level”. As an example: the linguistic expression “metalanguage” means language operating on a higher level to describe properties of the plain language [2]. From an optimization searching techniques point of view, heuristics are the algorithms that search for optimal solutions in a reasonable time with no guarantee that it will give the best solutions. In order to make the heuristic algorithms work better, researches combine them with local search techniques. Local search algorithms are based on random search techniques, which is discovering the search space using trial and error to establish a diversiform behavior in the searching process. They are fused with basic EAs to control the searching process of heuristic algorithms and form metaheuristic algorithms. The first scientist who used evolutionary algorithms was Alan Turing who summed up his ideas of neural network, machine intelligence and

learning and evolutionary algorithms in 1948. The development of Evolutionary Algorithms continued to advance and new EAs emerged such as Genetic Algorithm (GA) that was introduced by John Holland. GA was summarized it in Holland book "Adaptation in Natural and Artificial Systems" which was published in 1975. In the same period. Vladimir Vapink worked on support vector machine as a classification technique for linear methods. Vapink and his collaborators developed the nonlinear classification with kernel techniques and then it was summarized in Vapnik's book "The Nature of Statistical Learning Theory" in 1995. Additionally, metaheuristic algorithms were developed on the decades of the 1980s and 1990s, for example, the main algorithm for local search, Simulated Annealing(SA), was developed by Scott Kirkpatrick, C. Daniel Gellat, and Mario P.Vecchi in 1983. Another example that used memory in local search was studied by Fred Glover, the first scientist who used Tabu Search in 1986, then he summarized it in his book "Tabu search" in 1997. The first naturally inspired optimization algorithm was developed in 1992; Ant colony Optimization. In 1995, Particle Swarm Optimization (PSO) was introduced by James Kennedy and Russell C. Eberhart. Then, Rainer Storn and Kenneth Price developed Differential Evolution Algorithm (DE) in 1997. During the decade between 2000s and 2013s, more EAs were developed like Harmony search (HS) in 2001 and Artificial Bee Colony in 2005 [3]. The first hybrid algorithm was used in economics. Oliver, in 1993, used fuzzy hybrid system relating to decision making with "applications of risk assessment, credit evaluation, and insurance underwriting" [4].

1.1.2 Importance of Evolutionary Algorithms

Evolutionary Algorithms (EAs) have been extensively studied in recent years. Yang in 2014 mentioned that the solution space for a given problem can be either a local space 'Intensification' or a global space 'Diversification' according to the given

algorithm strategies [3]. He also classified the algorithms into two basic categories: first, ***the trajectory-based algorithms*** that use a single individual to search for the optimal solution. For example, simulated annealing (SA). The second basic category of algorithms was ***the population-based algorithms*** that use a set of individuals for the search procedure such as Genetic Algorithm (GA). The objective function of any given problem is called fitness function. Fitness functions are the mathematical calculations that each optimization problem use to evaluate the quality of solutions, and they can be classified into single objective functions and multi-objective functions. As a rule, a minimization objective function can be presented in the following mathematical forms in (eq. no 1):

$$\begin{gathered} \textbf{minimize } f_i(x),\ (i=1,2,\ldots\ldots..,M)\ , \\ \textbf{where } x \in R^d \\ \textbf{subject to } h_i(x)=0,\ (j=1,2,\ldots\ldots\ldots,J), \\ g_k(x) \leq 0,\ (k=1,2,\ldots\ldots\ldots K), \end{gathered} \tag{1}$$

Where $f_i(x)$, $h_j(x)$, and $g_k(x)$ are functions of the design vector $x=(x_1,x_2,\ldots,x_d)^T$, Where the components x_i of x are called *decision variables*, they can be real continuous, discrete, or a mix of these two. The objective function $f_i(x)$ where $(i=1,2,\ldots\ldots\ldots,M)$ is called the cost function. $f_i(x)$ objective could be to minimize or maximize the problem solutions. If **M=1**, then $f_i(x)$ is considered a ***single-objective*** function, if **M=2** or more then $f_i(x)$ is considered a ***multi-objective*** function. A group of decision variables are x_i , where $x_i=(x_1,x_2,\ldots\ldots,x_m)$ that form a single solution x, and $x \in R^d$ **,where d is the dimension**; the design space. The range of cost function values is called the solution space. A set of rules corresponding to the equalities of h_i and inequalities of g_k are the constraints of the optimization process.

As a rule, the basic form for EA share common population properties [5]:

1- Each individual in population-based algorithms can be called a ***search point*** in the solution space of the given problem. In addition, each individual may benefit from the usage of 'strategy parameters'.

2- Offspring of population are generated randomly for applying ***mutation*** and ***recombination***. The process of mutation intends to make small modifications on the selected individual to reach a near replicant of it, where recombination is based on sharing information between two or more selected individuals.

3- A measure of quality or fitness value can be assigned to each individual aiming for representing their quality. Fitness values of different individuals should be comparable to show which of the individuals holds superior features according to its fitness measure. The selection process favors better individuals to reproduce more often than those that are relatively worse. The following Pseudocode represents the basic process of population-based Evolutionary Algorithms, while figure 1.1 shows the flowchart of the Evolutionary Algorithms.

Pseudocode for Evolutionary Algorithms [6]

BEGIN

Input: population size, fitness function;

Output: best solution;

1. REPEAT UNTIL (stopping condition is satisfied) DO
2. Initialization individuals randomly;
3. Calculate fitness function;
4. Update
5. Selection;
6. Mutation;
7. Recombination;
8. Selection of new generation;
9. Report the best solution;
10. END

Recently, there has been growing interest in real-world optimization problems that have made scientific community become focused on developing various EAs such as Genetic algorithm(GA), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Water wave optimization and Gravitational search algorithm [7]. The development results successfully found the optimal solutions or near optimal solutions in a reasonable practical time for some EAs, as opposed to the time-consuming problem solving such as mathematical proof of convergence of some bio-inspired algorithms [3].

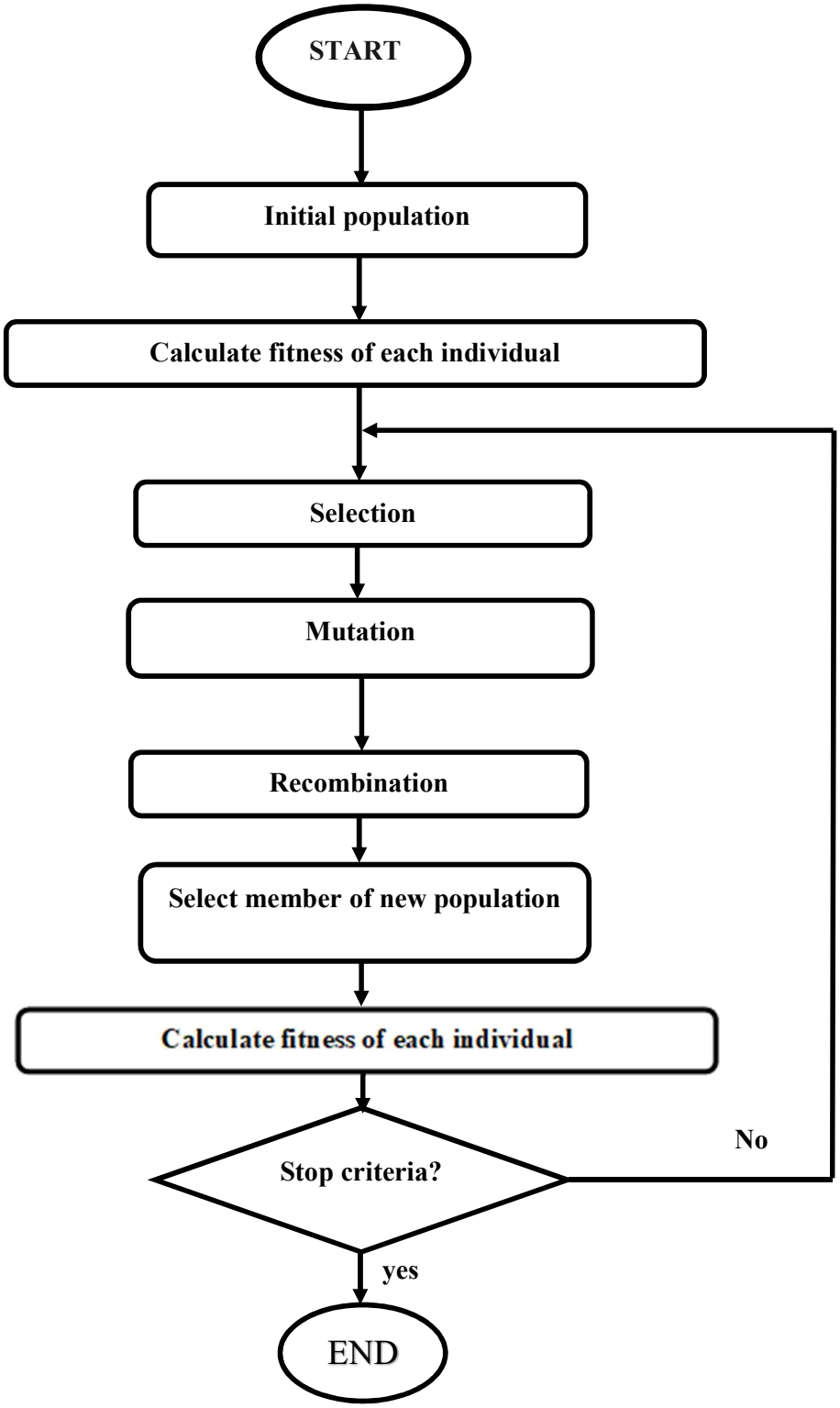

Figure 1.1: Flowchart of Evolutionary Algorithms [6]

1.1.3 Particle Swarm Optimization algorithms

Particle Swarm Optimization (PSO) is inspired by the social behavior of a flock of migrating birds trying to reach a destination. In PSO, each solution is a "bird" in the swarm and is referred to as ***a particle***. A particle ***P*** is a population member that contains two value position ***X*** and velocity ***V***. The evolutionary process in the PSO does not create new birds from parent ones. Rather, the birds in the population only evolve their social behavior and accordingly change their locations towards a destination [8]. As a rule, there are three stages for PSO as an EA:

1. Initial stage.
2. Update stage.
3. Report output stage.

In Initial stage of the evolutionary process is initialized by giving each particle a random value for its position and velocity. The position value will be assigned to the variable called the personal best P_i of the given particle. The cost of the personal best value is compared with the global best P_g value, if it's better than the global best then it will also be assigned as global best of the population. If the personal best is worst than the global best then it will be compared to the global worst solution in the population P_{g_worst}. If the personal best value is found worse than the global worst, the personal best for the current particle will be assigned as global worst solution. The i^{th} particle is represented by its position as a point in a ***d***-dimensional space, where ***s*** is the population size.

During the update stage, the particles values are updated with two eq. (2) for updating the velocity value and eq. (3) for updating the position value.

Accordingly, each particle updates its velocity V_i to catch up with the global best particle P_g, as follows [9] :

New $V_i = \omega$. *current* $V_i + c1$. ***rand()*** . $(P_i - X_i) + c2$. ***Rand()*** x $(P_g - X_i)$ (2)

The algorithm then uses the new velocity V_i for updating the particle's position:

New position X_i = *current position* X_i + *New* V_i ; $Vmax \geq V_i \geq - Vmax$ (3)

where *c1* and *c2* are two positive constants named ***learning factors*** and ($c1 = c2 = 2$). ***rand()*** and ***Rand()*** are two random functions in the range [0, 1], *Vmax* is an upper limit on the maximum change of particle velocity. The operator ω (decreases linearly with time from a value of 1.4 to 0.5) is an inertia weight employed as an improvement proposed by Shi and Eberhart to control the impact of the previous history of velocities on the current velocity and plays the role of balancing the global and the local search [9]. The output stage will give the values of *global best, global worst*, the mean for all the values of global best, the median for all the values of global best and the standard deviation for all the value of global best. In chapter 2 there will be detailed explanation about original PSO.

1.2 Statement of the Problems

In optimization field that uses the EAs to find optimal solutions, there are difficulties in solving traditional mathematical optimization methods, that can be listed as:

* The initialization of the solutions determines the process of reaching near-optimal solution or optimal solution.

* While searching in the solution space, the algorithm could get stuck in the local optimum.

* Some methods require some specific properties such as convexity.

* The difficulties of controlling tuning parameters. Sometimes the cost of tuning the parameters can be much higher than the cost of actual problem solving. As an example,

an algorithm has 5 control parameters and every one of the parameters has 10 possible values, which mean the total candidate setting of the parameters is 10^5. As a result, evaluating each distinct setting can require a certain number of fitness function calls which often takes more time than the actual problem needs.

* EAs can become slower after some generations in popular-based algorithms, also the quality of the solution can be affected by the increasing size of the problem [10] [11] [12].

As a result, in order to overcome the previous obstacles, research studies worked on developing algorithms and demonstrating the feasibility of metaheuristics algorithms, however, there exists no algorithm that solves all the problems and EAs development is still going on until now. Since 1999 until now [13] the Congress on Evolutionary Computation (CEC) has presented a set of problems evaluation every year for the researchers in order to encourage them to develop new EAs then test them on the benchmark problems to get either near optimal solution or the optimal one.

1.3 Aim of the Study

Stagnation in the local minimum is one of the main problems that prevents the algorithm from reaching the optimal solution of a given problem. Therefore, for reaching near-optimal solution or the optimal one using metaheuristics algorithms, the designer must take into account the balance between ***Intensification***; the exploitation of the local space, and the ***Diversification***; the exploration of the global space, in order for the algorithm to be able to search through the entire solution space and not to get stuck in the local optimal points.

This thesis seeks to propose A Hybrid Particle Swarm Optimization metaheuristics algorithm that targets combining the original PSO with a simple local search technique (FminLS). Our aim is to use the power of local search for reaching the near optimal solutions for solving CEC2015 benchmark set of 15 single objective problems [11].

1.4 Significance of the study

In general, the controlling and connection of data devices, such as mobiles and laptops, need some experts in building applications to present the data to the users. In terms of development of these applications, professionals in EAs are needed. Therefore, during the past 20 years, the community of science have become more interested in Evolutionary Algorithms that are used as Metaheuristics methods which have been applied in many applications. According to Ghazali [6] "Optimization is everywhere; optimization problems are often complex; then metaheuristics are everywhere".

1.5 Structure of the Thesis

This thesis is structured as follows, Chapter 2 demonstrates the basic information about PSO, literature review, and a summary of previous work of different EAs on CEC2015. Local search explanation of using Fmin for hybridizing the proposed algorithm PSO-LSFmin are described in the methodology chapter. Experimental Results chapter explains the proposed versions of HPSO-FminLS and demonstrates the results of the implementation. Therefore, showing findings of the four versions of HPSO-LSFminLS and then comparing between HPSO-FminLS versions with the previous work on Congress on Evolutionary Computation CEC2015, then explaining the results of Friedman Ranking test for comparing the 4 versions of the proposed method. Lastly, the conclusion of this dissertation.

Chapter 2

LITERATURE REVIEW

2.1 Overview

Metaheuristics Algorithms can be classified into natural inspired algorithms; PSO, and unnatural inspired algorithms; SA, memory using algorithms; Tabu Search, and memoryless algorithms; local search, population-based algorithms; GA, and trajectory-based algorithms; GRASP [14]. This chapter provides general information about Particle Swarm Optimization, then examples about local search techniques will be shown. Finally, a summary of previous work of applied EAs on CEC2015 respectively.

2.2 Original Particle Swarm Optimization

One of main types of natural swarm intelligent algorithms is Particle Swam Optimization (PSO), which is inspired from social behavior of bird flocking. The mathematical scientists Eberhart and Kennedy developed the mechanism of (PSO) in 1995 to be presented as a population-based stochastic algorithm. PSO bird flocking behave according to this scenario: a group of birds are moving from a source location to a destination. Their aim is to reach the destination through consuming minimum amount of energy. All the birds do not know where the destination is. Hence, they move in random directions. But each bird knows its nearest distance to destination and the bird in its vicinity that is closest to the destination. So, *what's the best strategy to reach the destination?* The effective idea is to follow the bird which is nearest to the destination while also not being too far from the individual's personal best position.

That is the best strategy to be followed; a combination of the individual's best strategy and the best strategy in the neighborhood.

PSO is developed based on the above strategy and hard numerical optimization problems are solved by it. In PSO, the algorithm considers that each "bird" represents a single solution in the search space. The "bird" can be called a "particle". A particle's current position is considered to be a potential solution to the underlying optimization problem. Each one of the particles consists of two values; velocity and position. The position value is evaluated by the fitness function, and the velocity value is the direction of the flying particles. The particles are hypothetically flying through the problem search range according to the direction of the current optimum particles. The following Pseudocode is the original PSO.

Pseudocode for Original PSO:

Input: $\text{Problem}_{\text{Size}}$, $Population_{size}$

Output: P_{g_best}, P_{g_worst}, Median(P_{g_best}), Mean(P_{g_best}), STD(P_{g_best})

1 Population← ∅;

2 P_{g_best} ← ∅;

3 for i=1 to $Population_{size}$ do

4 $P_{velocity}$ ← RandomVelocity();

5 $P_{position}$← RandomPosition($Population_{size}$);

6 P_{p_best} ← $P_{position}$;

7 if Cost(P_{p_best}) ≤ (P_{g_best}) then

8 P_{g_best} ←P_{p_best} ;

9 else

10 if Cost (P_{p_best}) ≥ (P_{g_worst}) then

11 $P_{g\_worst} \leftarrow P_{p\_best}$;

12 end

13 end

14 while ¬ StopCondition() do

15 For each $P \in$ Population do

16 $P_{velocity} \leftarrow$ UpdateVelocity($P_{velocity}$, P_{g_best} ,P_{p_best});

17 $P_{position} \leftarrow$ Update Position($P_{position}$,$P_{velocity}$);

18 If Cost($P_{position}$)≤ Cost(P_{p_best}) then

19 $P_{p\_best} \leftarrow P_{position}$;

20 If Cost(P_{p_best})≤ Cost (P_{g_best}) then

21 $P_{g\_best} \leftarrow P_{p\_best}$;

22 end

23 end

24 end

25 end

26 return P_{g_best} ,P_{g_worst} ,Median(P_{g_best}),Mean(P_{g_best}),STD(P_{g_best}) ;

Where:

ProblemSize : present the number of the given problem.

***Population**$_{size}$:* present the numbers of the particles that equal to 200.

***P**$_{velocity}$:* present velocity value of each particle.

***P**$_{position}$:* present the position value of each particle.

***P**$_{p\_best}$* : present the personal best value for each particle.

***P**$_{g\_best}$*: present the global best value of all population.

***P**$_{g\_worst}$*: present the global worst value of all population.

Median(P_{g_best}): present the middle value of all the global best of all the iterations.

Mean(P_{g_best}): present the average value of all the global best of all the iterations.

STD(P_{g_best}): present the standard deviation value of all the global best of all the iterations.

The variables in the above Pseudocode are $Problem_{Size}$ is the defining of fitness according to the given problem. Where, $Population_{size}$ is used for the number of particles in the PSO population, P_{g_best} is used for storing the global best solution, P_{p_best} is used to store the personal best for each particle in PSO population. The particle in the swarm has two values $P_{velocity}$ and $P_{position}$ that are selected randomly at the begining of each iteration using the method RandomVelocity() and RandomPosition($Population_{size}$). The Cost is represented by the result of fitness function calculation of each particle. StopCondition( ) is the condition for stopping the loop iteration for the swarm.

UpdateVelocity is the same formula of eq. (2), while UpdatePosition is the same formula of eq. (3) that have been described in Chapter1. In the end of the algorithm loop, the value of global best, global worst, mean for all the values of global best, median for all the values of global best, and standard deviation for all the value of global best are demonstrated.

2.3 Local Search

Local search is the trajectory algorithm that works on improving single solution; the current, in the process of searching for a better solution to the given problem [6]. It is also called iterative improvement for a single solution. There are many local search algorithms and the simplest one is to select the best neighborhood N of the solution

(best improvement). Best improvement local search is the process of searching for better solutions among the neighborhood of the current solution in each iteration , that is to check every neighbour of the current solution then compare it with the best solution that the algorithm have already found in the iteration ; if one of the neighbours is better than the best solution, it will be assigned as the current solution and then continue searching the neighborhood of this current solution until stopping criteria is satisfied. The following pseudocode shows the main steps for Local Search Best Accept LS.

Pseudocode Local Search Best Accept:

Input : s_0, N, F

Output : best_neighbor

1: current ← s_0;

2: done ← **false;**

3: while done= false do

4: best_neighbor ← current;

5: for each s∈*N(*current) do

6: if *F*(s) < *F(*best_neighbor) then

7: best_neighbor ← s;

8: end;

9: end;

10: if current =best_neighbor then

11: done← true

12: else

13: current ← best _neighbor

14: end;

15: end;

Where s_o is the starting solution, N is the neighborhood operator, F is the fitness function of the problem, and best neighbor is the best solution among the neighborhood of the current. The previous step (or move) could be applied in the searching process through many iterations by using ***Iterated Local Search*** (ILS) that works according to the mechanism in the following Pseudocode.

Pseudocode Iterated local search (ILS):

Input : $\mathbf{s_0}$,LS

Output: $\mathbf{s}^*$

1: **current** ←LS($\mathbf{s_0}$)

2: while stopping criterion not met do

3: **s← perturbation of current based on the search history**;

4: $\mathbf{s}^*$← LS(**s**);

5: if $\mathbf{s}^*$ is accepted as the new current solution then

6: current ←$\mathbf{s}^*$;

7: end ;

8: end ;

Where s_o is the starting solution from local search procedure, **s** is the result of perturbation process which change the position of the solution randomly, s^* is the best solution in the neighborhood of the current and it is accepted according to its evaluation of fitness function.

Unfortunately, the ILS can get stuck in the local optimum of the problem during the phase of updating only to best solutions, that is why ILS uses **perturbation strategy**

to avoid this problem. The **perturbation strategy is** used to change the location of the current solution ***s*** randomly.

Additionally, there is another local search algorithm that is used to improve the solution which is called ***Variable Neighborhood Search*** (VNS), the combination between LS and ILS is used to employ different neighborhood operators, as shown in the following pseudocode.

Pseudocode for basic Variable Neighborhood Search:

Input : s_o ,ILS, N_k , F, k

Output : s^*

1: **current** ←s_0

2: while stopping criterion not met do

3: $k \leftarrow 1$;

4: while $k \leq k_{max}$ do

5: s← random solution in N_k*(current)*

6: $s^* \leftarrow$ ILS(s) ;

7: if $F(s^*) < F$(current) then

8: current← s^*

9: k← 1;

10: else

11: k← k+1;

12 : end ;

13: end ;

14: end;

Where s_o is the starting solution of the LS procedure, s is the solution which is selected randomly from the neighborhood , s^* is the solution found by (ILS) which is searching for the best solution among the neighbors by using Best Accept strategy. The newly found solution is accepted according the given problem. F is the fitness function, if it's not accepted then we change the neighborhood strategy by $k = k+1$; the number of the given neighborhood strategies, until the counter of k ends, and the running of the algorithm counties until the stopping criterion is satisfied. The order of the neighborhoods

- Forward VNS: start with k=1 and increase
- Backward VNS: start with $k=k_{max}$ and decrease
- Extended version:
  - Parameters k_{min} and k_{step}
  - Set $k = k_{min}$, and increase k by k_{step} if no improvement

- Accepting worse solutions
  - With some probability
  - Skewed VNS: Accept if
    - $f(s^*)-\alpha d(s, s^*) < f(s)$
    - $d(s, s^*)$ measures the distance between the solutions.

2.3 Previous work of EAs on CEC2015

Different studies were proposed for the purpose of optimizing CEC2015 Benchmark problems [11] . Some of the proposed methods are summarized in this section and then in chapter 4 we will compare their results with our proposed algorithm's results. First one of the EAs is the Self Adaptive Dynamic Multi-Swarm Particle Swarm Optimizer (sDMS-PSO) [7] which employed an adaptive way for assigning parameters to different swarms and used a paralleling method between multi swarms to detect the competitive arguments. At the same time, the evolutionary process of sDMS-PSO accelerated the learning speed and shared information about the best parameters for

the purpose of getting a faster convergence using a local search; enhancing the ability of exploitation. Similarly, the quasi-Newton method used a Local Search mechanism in order to improve problems Generalized Artificial Bee Colony algorithm (ABC-X-LS) [12] that presented a flexible, freely configurable framework for ABC that is adaptable to different specific problems with no need for any algorithm re-design. Next, hybrid Cooperative Co-evolution algorithm (hCC) [15] which had two stages. The initial stage was conducted by applying the recently introduced differential grouping for learning the problem variables' inter-dependencies. Then the variables were separated into groups of separable and non-separable ones. The second stage of the method adopted different algorithms (ABC) with cooperative co-evolution (CC) framework for the purpose of simultaneously optimizing the generated groups. The final two literature methods were the Differential Evolution with Success-Based Parameter Adaptation (DEsPA) [16] and the Differential evolution with Linear Population Size Reduction working with Neuro-Dynamic (LSHADE-ND) [17]. DEsPA presented a new mechanism to adapt the control parameters using a memory-based structure of previous successful settings while LSHADE-ND algorithm employed embedding the concept of Neuro-Dynamic into modified success history based parameter adaptation for Differential Evolution with linear population-size reduction.

Chapter 3

METHODOLOGY

3.1 Overview

In this section we will present a brief explanation about the proposed algorithm of Hybrid Particle Swarm Optimization with Fmin Local Search (HPSO-FminLS). Four different versions of HPSO-FminLS were developed and presented in this study.

3.2 Proposed Hybrid Particle Swarm Optimization with FminLS (HPSOFminLS)

3.2.1 Fmin Local Search (FminLS)

Fmin is a function that is employed in MATLAB application for finding the minimum of single-solution function using derivative-free method. In our method, we used two functions, first is Fmin search that minimize function of several variables, second is Fmincon that is a constrained nonlinear multi-variable function that works upon the boundaries of the given problem, such as CEC2015 benchmark problems, the mathematical description of the two functions; Fmin and Fmincon, are as follows.

3.2.2 Fminsearch

General formula, eq. (4):

[x,fval]=fminsearch(…) (4)

x is the solution, *fval* is the objective function fun

Matlab code

[xx,fval]=fminsearch(@function,xx);

fval is the value of objective function fun, that is the description of the function, at the solution xx, Fmin search is an unconstrained nonlinear optimization, which gives the minimum of a scalar function of several variables, starting at an initial estimate.@fucntion is the calling of the given problem ,xx is the calling of current solution (particle) in the current iteration of Fmin search algorithm.

3.2.3 Fmincon

The function of Fmincon is $min_x\ f(x)$ subject to

$$\begin{aligned} c(x) &\leq 0 \\ ceq(x) &= 0 \\ A * x &\leq b \\ Aeq * x &= beq \\ lb \leq x &\leq ub \end{aligned} \tag{5}$$

Where

- x, b, beq, lb and ub are vectors.
- A and Aeq are matrices.
- c(x) and ceq(x) are functions that return vectors.
- f(x) is a function that returns a scalar.

F(x) c(x), and ceq(x) can be nonlinear functions[20].

General formula

[x,fval]= fmincon(…)

x is the solution of the given problem, *fval* the objective function fun.

Matlab code

$[xx, fval] =$

$fmincon(@myfuntion, gbest, [\quad], [\quad], [\quad][\quad], -100{,}100, [\quad], options);$

xx are the solution (particle) of the current iteration ,*fval* is the value of objective function fun ,that is the description of the function, at the solution *xx* ,*fmincon* finds a constrained minimum of scalar unit of several variables starting at an initial estimate, *@myfunction* is one of the given problems ,*gbest* is the global best solution that has been found in the current iteration ,-*100* is the lower bound of the solutions ,*100* is the upper bound of the solutions, *options* are optimization parameters of specified structure.

3.3 Original PSO

We downloaded the code from http://www.ntu.edu.sg/home/EPNSugan/index_files/ for CEC'15 problems then implemented it to find the **global best solution P_{g_best}, global worst P_{g_worst}** solutions. Then calculated the ***median, mean***, and ***standard deviation "std"*** for the global best solution. The Pseudocode of the algorithm was mentioned in chapter 2.

3.4 Four versions of HPSO-FminLS

3.4.1 HPSO-FminLS -E

In this version, we improved the global best solution P_{g_best} with FminLS. Then the solution found was compared with the previous P_{g_best} found in the last iteration. After finding the best of each iteration, if the solution's fitness is improved compared to the previous global best fitness value, then it will be stored as the new P_{g_best}.

Next is the steps of improving the global best by FminLS:

1 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

2 *If* $P_{g\_best} \leq FminLS(P_{g\_best})$ then

3 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

4 $Population \leftarrow P_{g\_best}$;

5 end;

Next is the Pseudocode of the algorithm steps.

Pseudocode for PSO for HPSO-FminLS -E:

Input: $\text{Problem}_{\text{Size}}$, $Population_{size}$

Output: P_{g_best}, P_{g_worst}, Median(P_{g_best}), Mean(P_{g_best}), STD(P_{g_best})

1 Population← ∅;

2 $P_{g\_best} \leftarrow \emptyset$;

3 for i=1 to $Population_{size}$ do

4 $P_{velocity} \leftarrow$ RandomVelocity();

5 $P_{position} \leftarrow$ RandomPosition($Population_{size}$);

6 $P_{p\_best} \leftarrow P_{position}$;

7 if Cost(P_{p_best}) ≤ (P_{g_best}) then

8 $P_{g\_best} \leftarrow P_{p\_best}$;

9 end;

10 if Cost (P_{p_best}) ≥ (P_{g_worst}) then

11 $P_{g\_worst} \leftarrow P_{p\_best}$;

12 end;

13 end;

14 while ¬ StopCondition() do

15 For each $P \in$ Population do

16 $P_{velocity} \leftarrow$ UpdateVelocity($P_{velocity}$, P_{g_best} ,P_{p_best});

17 $P_{position} \leftarrow$ Update Position($P_{position}$,$P_{velocity}$);

18 If Cost($P_{position}$)≤ Cost(P_{p_best}) then

19 $P_{p\_best} \leftarrow P_{position}$;

20 If Cost(P_{p_best})≤ Cost (P_{g_best}) then

21 $P_{g\_best} \leftarrow P_{p\_best}$;

22 end;

23 end;

24 end;

25 Improving P_{g_best} with FminLS

26 end;

27 end;

28 return P_{g_best} ,P_{g_worst} ,Median(P_{g_best}),Mean(P_{g_best}),STD(P_{g_best}) ;

Where:

ProblemSize: present the number of the given problem.

Population$_{size}$: *present* the numbers of the particles that equal to 200.

$P_{velocity}$: present velocity value of each particle.

$P_{position}$: present the position value of each particle.

P_{p_best}: present the personal best value for each particle.

P_{g_best}: present the global best value of all population.

P_{g_worst}: present the global worst value of all population.

Median(*P_{g_best}*): present the middle value of all the global best of all the iterations.

Mean(*P_{g_best}*): present the average value of all the global best of all the iterations.

STD(P_{g_best}): present the standard deviation value of all the global best of all the iterations.

3.4.2 HPSO-FminLS-B-E

In this version, we improved P_{g_best} at the beginning of population of PSO with FminLS. If its fitness value was improved then the algorithm considered the new solution as P_{g_best}. next is the steps:

Improving the global best by *FminLS* at the beginning

1 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

2 *If* $P_{g\_best} \leq FminLS(P_{g\_best})$ then

3 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

4 $Population \leftarrow P_{g\_best}$;

5 end;

Then, after updating the velocity and position of each particle in the swarm we improve P_{g_best} again using FminLS. After finding P_{g_best} of each iteration and comparing it with the previous P_{g_best} of the last iteration, if it is improved then it will be stored as the new P_{g_best} in order to reach the optimal solution. next the steps:

Improving the global best by *FminLS* at the end

1 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

2 *If* $P_{g\_best} \leq FminLS(P_{g\_best})$ then

3 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

4 $Population \leftarrow P_{g\_best}$;

5 end;

The steps are shown in the following Pseudocode.

Pseudocode for PSO for HPSO-FminLS-B-E:

Input: ProblemSize, $Population_{size}$

Output:P_{g_best} ,P_{g_worst} ,Median(P_{g_best}),Mean(P_{g_best}),STD(P_{g_best})

```
1    Population← ∅;
2    P_g_best ← ∅;
3      for i=1 to Population_size do
4          P_velocity ← RandomVelocity();
5          P_position← RandomPosition(Population_size);
6          P_p_best ← P_position ;
7           if Cost( P_p_best ) ≤ (P_g_best) then
8              P_g_best ←P_p_best ;
9           end;
10         P_g_worst ← Max(P_g_best );
11   end;
12    Improving global best at the beginning.
13   while ¬ StopCondition() do
14       For each P ∈ Population do
15            P_velocity ← UpdateVelocity(P_velocity, P_g_best ,P_p_best);
16            P_position ← Update Position(P_position ,P_velocity);
17           If  Cost(P_position)≤ Cost(P_p_best ) then
18               P_p_best ←P_position ;
19               If  Cost(P_p_best)≤ Cost (P_g_best) then
20                   P_g_best←P_p_best ;
21               end;
22           end;
```

23 end;

24 Improving global best with FminLS at the end.

25 end;

26 return P_{g_best}, P_{g_worst}, Median(P_{g_best}), Mean(P_{g_best}), STD(P_{g_best}) ;

Explanation of the variables:

ProblemSize: present the number of the given problem.

***Population**$_{size}$: present* the numbers of the particles that equal to 200.

***P**$_{velocity}$:* present velocity value of each particle.

***P**$_{position}$:* present the position value of each particle.

***P**$_{p\_best}$* : present the personal best value for each particle.

***P**$_{g\_best}$*: present the global best value of all population.

***P**$_{g\_worst}$*: present the global worst value of all population.

Median(*P*$_{g\_best}$): present the middle value of all the global best of all the iterations.

Mean(*P*$_{g\_best}$): present the average value of all the global best of all the iterations.

STD(*P*$_{g\_best}$): present the standard deviation value of all the global best of all the iterations.

3.4.3 HPSO-FminLS- W-E

In this version, in each iteration we applied FminLS to the global worst solution P_{g_worst} before updating the velocity and positions of each particle. If the global worst shows improvement, we assign it as a new particle in the population according to the next steps:

Improving the global worst by *FminLS*

1 $P_{g\_worst} \leftarrow FminLS(P_{g\_worst})$;

2 *If* $P_{g\_worst} \leq FminLS(P_{g\_worst})$ then

3 $P_{g\_worst} \leftarrow FminLS(P_{g\_worst})$;

4 $Population \leftarrow P_{g\_worst}$;

5 end;

Additionally, an application of FminLS was conducted to improve the global best P_{g_best} after updating the velocity and positions. These two procedures were implemented in the same iteration in order to reach the optimal solution. The steps are presented next:

Improving the global best by *FminLS* at the end

1 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

2 *If* $P_{g\_best} \leq FminLS(P_{g\_best})$ then

3 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

4 $Population \leftarrow P_{g\_best}$;

5 end;

The steps are described as Pseudocode below.

Pseudocode for PSO for HPSO-FminLS- W-E

Input: ProblemSize, $Population_{size}$

Output:P_{g_best} ,P_{g_worst} ,Median(P_{g_best}),Mean(P_{g_best}),STD(P_{g_best})

```
1    Population← ∅;
2    P_g_best ← ∅;
3      for i=1 to Population_size do
4            P_velocity ← RandomVelocity();
5            P_position← RandomPosition(Population_size);
6           P_p_best ← P_position ;
7           if Cost( P_p_best ) ≤ (P_g_best) then
8              P_g_best ←P_p_best ;
9           end;
10        P_g_worst ← Max(P_g_best );
11    end;
12  Improving global worst by FminLS
13   while ¬ StopCondition() do
14        foreach P ∈ Population do
15            P_velocity ← UpdateVelocity(P_velocity, P_g_best ,P_p_best);
16            P_position ← Update Position(P_position ,P_velocity);
17           If  Cost(P_position)≤ Cost(P_p_best ) then
18               P_p_best ←P_position ;
19               If  Cost(P_p_best)≤ Cost (P_g_best) then
20                   P_g_best←P_p_best ;
21               end;
22           end;
```

23 end;

24 Improving global best with FminLS at the end.

25 end;

26 return P_{g_best} ,P_{g_worst} ,Median(P_{g_best}),Mean(P_{g_best}),STD(P_{g_best}) ;

Where:

ProblemSize : present the number of the given problem.

***Population**$_{size}$: present* the numbers of the particles that equal to 200.

***P**$_{velocity}$:* present velocity value of each particle.

***P**$_{position}$:* present the position value of each particle

***P**$_{p\_best}$* : present the personal best value for each particle.

***P**$_{g\_best}$*: present the global best value of all population.

***P**$_{g\_worst}$*: present the global worst value of all population.

Median(*P*$_{g\_best}$): present the middle value of all the global best of all the iterations.

Mean(*P*$_{g\_best}$): present the average value of all the global best of all the iterations.

STD(*P*$_{g\_best}$): present the standard deviation value of all the global best of all the iterations.

3.4.4 HPSO-FminLS-B-W-E

In this version, we applied FminLS three times, which represents the merging between version 2 and version 3.

First, the global best ***P**$_{g\_best}$* was improved with FminLS at the beginning of PSO algorithm. By the following steps:

Improving the global best by *FminLS* at the beginning

1 $P_{g\_best} \leftarrow FminLS\ (P_{g\_best})$;

2 *If* $P_{g\_best} \leq FminLS(P_{g\_best})$ then

3 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

4 $Population \leftarrow P_{g\_best}$;

5 end;

The second step conducted improving the worst solution with FminLS and then adding it to the population individuals using the steps from **Improving the global worst by *FminLS***

1 $P_{g\_worst} \leftarrow FminLS(P_{g\_worst})$;

2 *If* $P_{g\_worst} \leq FminLS(P_{g\_worst})$ then

3 $P_{g\_worst} \leftarrow FminLS(P_{g\_worst})$;

4 $Population \leftarrow P_{g\_worst}$;

5 end;

Finally, the global best was improved again with FminLS at the end of PSO algorithm in order to reach the optimal solution. By the steps:

Improving the global best by *FminLS* at the end

1 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

2 *If* $P_{g\_best} \leq FminLS(P_{g\_best})$ then

3 $P_{g\_best} \leftarrow FminLS(P_{g\_best})$;

4 $Population \leftarrow P_{g\_best}$;

5 end;

The following Pseudocode presents the steps of this version.

Pseudocode for PSO for HPSO-FminLS-B-W-E

Input: ProblemSize, $Population_{size}$

Output:P_{g_best} ,P_{g_worst} ,Median(P_{g_best}),Mean(P_{g_best}),STD(P_{g_best})

1 Population← ∅;

2 P_{g_best} ← ∅;

3 for i=1 to $Population_{size}$ do

4 $P_{velocity}$ ← RandomVelocity();

5 $P_{position}$← RandomPosition($Population_{size}$);

6 P_{p_best} ← $P_{position}$;

7 if Cost(P_{p_best}) ≤ (P_{g_best}) then

8 P_{g_best} ←P_{p_best} ;

9 else

10 If Cost(P_{p_worst}) ≥ ($P_{\_g\_worst}$)

11 P_{g_worst} ←P_{p_best}

12 end;

13 end;

14 **Improving the global best by *FminLS* at the beginning**

15 **Improving the global worst by *FminLS***

16 while ¬ StopCondition() do

17 foreach P ∈ Population do

18 $P_{velocity}$ ← UpdateVelocity($P_{velocity}$, P_{g_best} ,P_{p_best});

19 $P_{position}$ ← Update Position($P_{position}$,$P_{velocity}$);

20 If Cost($P_{position}$)≤ Cost(P_{p_best}) then

21 P_{p_best} ←$P_{position}$;

22 If Cost(P_{p_best})≤ Cost (P_{g_best}) then

23 $P_{g\_best} \leftarrow P_{p\_best}$;

24 end;

25 end;

26 end;

27 **Improving the global best by *FminLS* at the end**

28 end;

29 return P_{g_best}, P_{g_worst}, Median(P_{g_best}), Mean(P_{g_best}), STD(P_{g_best}) ;

Where:

ProblemSize : present the number of the given problem.

$\boldsymbol{Population_{size}}$*: present* the numbers of the particles that equal to 200.

$\boldsymbol{P_{velocity}}$*:* present velocity value of each particles.

$\boldsymbol{P_{position}}$*:* present the position value of each particles.

$\boldsymbol{P_{p\_best}}$: present the personal best value for each particles.

$\boldsymbol{P_{g\_best}}$: present the global best value of all population.

$\boldsymbol{P_{g\_worst}}$: present the global worst value of all population.

Median($\boldsymbol{P_{g\_best}}$): present the middle value of all the global best of all the iterations.

Mean($\boldsymbol{P_{g\_best}}$): present the average value of all the global best of all the iterations.

STD($\boldsymbol{P_{g\_best}}$): present the standard deviation value of all the global best of all the iterations.

Chapter 4

EXPERIMANTAL RESULTS

4.1 Overview

The experiment was applied on 15 black-box benchmark functions using the four proposed hybrid methods for PSO; HPSOFminLS-E, HPSOFminLS-B-E, HPSOFminLS-W-E, and HPSOFminLS-B-W-E at three dimensions 10, 30 and 50. In general, the results affirmed a noticeable difference between results of the original PSO and the proposed HPSO versions. In this chapter, the results of the experiment conducted on the CEC2015 Benchmark problems are shown and explained. Then, a comparison between the results of Hybrid PSO with Literature is demonstrated. Finally, a ranking test will be applied and a conclusion will be provided.

4.2 CEC2015 Expensive Optimization Test Problems

The Original code of PSO was downloaded from [21]. The set of 15 benchmark single objective problems were used as basis of the more complex optimization algorithms such as multi-objective, niching, dynamic, constrained optimization algorithms. All the problems were tested as black-box optimization problems.

4.2.1 Common definitions

All of the test problems' mathematical equations are minimization functions defined by the next general structure in eq. (6)

$$min\ f(x), x = [x_1, x_2, \dots., x_D]^T \qquad 6$$

Where:

x: vector of variables.

D: dimensions.

$\boldsymbol{o_{i1}} = [o_{i1}, o_{i2}, \ldots\ldots, o_{iD}]^T$: the shifted global optimum, which is randomly divided up in [-80,80]^D. Each mathematical equation has a shift data for CEC'14. All test functions are shifted to $\boldsymbol{o}$ and scalable. For convenience, the same search ranges are defined for all test functions.

Search range: [-100,100]^D .

M_i : rotation matrix. Different rotation matrixes are assigned to each function and each basic function. The variables are randomly split into subcomponents. Production of the rotation matrix for each subcomponent is applied from standard normalized distributive inputs by Gram-Schmidt ortho-normalization with the number *c* either equal to 1 or 2 as a satisfaction rule.

In CEC2015 the benchmark problems were categorized into four groups. First, ***Unimodal Functions*** with problems (**1 and 2**). The second group was ***Simple Multimodal Functions*** that contained three problems:(**3, 4 and 5**). Then, ***Hybrid Functions*** which contained another three problems: (**6, 7 and 8**). The rest of the fifteen functions were entitled as ***Composition functions***. Table 4.1 present the structure of the problems.

4.2.2 CEC'15 Problems.

In this study we used hybrid heuristic Particle Swarm Optimization and Fmin LS 4 versions in dimension 10, 30 and 50 on the CEC'15 Learning-Based Benchmark Suite that are showing in Table 4.1

Table 4.1: summary of the CEC'15 Learning-Based Benchmark Suite

	# problem	Functions	$F_i^* = F_i(x^*)$
Unimodal Functions	1	Rotated High Conditioned Elliptic Function	100
	2	Rotated Cigar Function	200
Simple Multimodal Functions	3	Shifted and Rotated Ackley's Function	300
	4	Shifted and Rotated Rastrigin's Function	400
	5	Shifted and Rotated Schwefel's Function	500
Hybrid Functions	6	Hybrid Function 1 (Ackley's Function)	600
	7	Hybrid Function 2 (Rastrigin's Function)	700
	8	Hybrid Function 3 (Schwefel's Function)	800
Composition Function	9	Composition Function(N=3)	900
	10	Composition Function(N=3)	1000
	11	Composition Function(N=5)	1100
	12	Composition Function(N=5)	1200
	13	Composition Function(N=5)	1300
	14	Composition Function(N=7)	1400
	15	Composition Function(N=10)	1500
Search Range: $[-100,100]^D$			

Where N is the number of the basic function

4.3 Results of the algorithm HPSO-FminLS

The tables from Table 4.2 to Table 4.13 show the results of the proposed four versions of Hybrid PSO Algorithms. For the four versions of ***Unimodal Functions***, the problems 1 and 2 results show that hybridizing PSO with Fmin was able to reach **optimal solutions F*.**

.For D10, ***Simple Multimodal Functions*** with three problems: 3, 4 and 5, the results of all four versions of Hybrid PSO show that near optimal solution had been obtained. ***Problem 4*** was the best one among them with a small difference from the optimal

solution F*.The group of ***Hybrid Functions*** that contained another three problems; 6, 7 and 8, the results of the four versions show that the near optimal solution had been obtained and ***problem 7*** had the minimum result of the group.***Composition functions*** contained the rest of the problems; 9, 10, 11, 12, 13, 14 and 15. All of the results show that the near optimal solution was obtained and the best result was for ***problem 11***.

For D30, the result of all the problems were near optimal solutions in ***Simple Multimodal***, ***Hybrid*** and ***Composition Functions***. For ***Simple Multimodal Functions***, ***problem 3*** result was the best among other problems. Hybrid function ***problem 7*** had obtained the best result among the other problem in the category. Results of ***Composition functions*** show that ***problem 13*** obtained the best result among all other problems.

For D50, the results in all the problems had obtained near optimal solutions in ***Simple Multimodal***, ***Hybrid*** and ***Composition Functions***. For ***Simple Multimodal Functions***, the result of ***problem 3*** was the best among the others problems. Hybrid function ***problem 7*** had obtained the best result among all other problems in the category. Results of ***Composition Functions*** show that ***problem 9*** and ***12*** obtained the best result among all the problems of the same category.

4.3.1 Results of comparing original PSO with Hybrid PSO Version 1 (HPSOFminLS-E)

Table 4.2: Results of Fmin at the End for PSO in D10.

#	F*	PSO	HPSOFminLS-E
1	100	387020	**100.0131**
2	200	82782000	**200.0176**
3	300	320.1884	319.9999
4	400	426.0494	**407.9597**
5	500	1165.5	653.6771
6	600	3343.4	748.8018
7	700	703.3774	**701.8066**
8	800	1987.5	875.5819
9	900	1000.4	1000.2
10	1000	1608.7	1226.9
11	1100	1118.6	**1109.8**
12	1200	1303.3	1301.2
13	1300	1334.2	1330.3
14	1400	2986.6	1500.1
15	1500	1610.2	1600

Table 4.3: Results of Fmin at the End for PSO in D30.

#	F*	PSO	PSOFminLS-E
1	100	64321000	**100.0131**
2	200	5203700000	**200.0013**
3	300	320.7218	**320**
4	400	595.5969	448.7529
5	500	6328.3	2130
6	600	907480	1429.7
7	700	730.6795	**714.7427**
8	800	315980	1383
9	900	1027.5	1004.2
10	1000	336240	1714.2
11	1100	1483.6	1433.6
12	1200	1318	1307.3
13	1300	1419.2	**1398.2**
14	1400	37512	2052.5
15	1500	1633.6	1610.4

Table 4.4: Results of Fmin at the End for PSO in D50.

#	F*	PSO	PSOFminLS-E
1	100	320780000	**100.002**
2	200	17588000000	**200.1692**
3	300	321.0368	**320**
4	400	830.9402	518.3999
5	500	12864	4479.6
6	600	11640000	1379.9
7	700	844.3723	**734.7006**
8	800	3156100	1645.4
9	900	1070.6	**1008.1**
10	1000	4990400	2109.9
11	1100	2860.7	2780.1
12	1200	1337.8	**1308.6**
13	1300	1522.1	1495
14	1400	69339	35302
15	1500	1889.9	1617.2

4.3.2 Results of comparing original PSO with Hybrid PSO Version 2 (HPSOFminLS-B-E)

Table 4.5: Results of Fmin at the Beginning and at the End for PSO in D10.

#	F*	PSO	HPSOFminLS-B-E
1	100	387020	**100**
2	200	82782000	**200.023**
3	300	320.1884	315.6645
4	400	426.0494	**404.9748**
5	500	1165.5	518.4721
6	600	3343.4	735.5666
7	700	703.3774	**702.0541**
8	800	1987.5	818.1371
9	900	1000.4	1000.3
10	1000	1608.7	1226
11	1100	1118.6	**1108**
12	1200	1303.3	1301.9
13	1300	1334.2	1328.9
14	1400	2986.6	1500
15	1500	1610.2	1600

Table 4.6: Results of Fmin at the Beginning and at the End for PSO in D30.

#	F*	PSO	HPSOFminLS-B-E
1	100	64321000	**100.0004**
2	200	5203700000	**200.0061**
3	300	320.7218	**320**
4	400	595.5969	444.7731
5	500	6328.3	2585.9
6	600	907480	1390.9
7	700	730.6795	**714.9418**
8	800	315980	1187.8
9	900	1027.5	1004.7
10	1000	336240	1721.1
11	1100	1483.6	1476.9
12	1200	1318	1307.1
13	1300	1419.2	**1404**
14	1400	37512	2054.6
15	1500	1633.6	1611.4

Table 4.7: Results of Fmin at the Beginning and at the End for PSO in D50.

#	F*	PSO	HPSOFminLS-B-E
1	100	320780000	**100.0018**
2	200	17588000000	**200.1132**
3	300	321.0368	**320**
4	400	830.9402	504.4705
5	500	12864	4739.7
6	600	11640000	2252.5
7	700	844.3723	**734.144**
8	800	3156100	1832.6
9	900	1070.6	**1006.5**
10	1000	4990400	2143.3
11	1100	2860.7	1889.1
12	1200	1337.8	**1308.6**
13	1300	1522.1	1499.1
14	1400	69339	35288
15	1500	1889.9	1616.6

4.3.3 Results of comparing original PSO with Hybrid PSO Version 3 (HPSOFminLS-W-E)

Table 4.8: Results of Fmin for the global worst solution and at the End for PSO in D10

#	F*	PSO	HPSOFminLS-W-E
1	100	387020	**100**
2	200	82782000	**200**
3	300	320.1884	301.6462
4	400	426.0494	**402.9849**
5	500	1165.5	510.3073
6	600	3343.4	665.0878
7	700	703.3774	**701.5606**
8	800	1987.5	817.7843
9	900	1000.4	1000.3
10	1000	1608.7	1118.5
11	1100	1118.6	1109.1
12	1200	1303.3	1301.1
13	1300	1334.2	**1329.6**
14	1400	2986.6	1700
15	1500	1610.2	1600

Table 4.9: Results of Fmin for the global worst solution and at the End for PSO in D30

#	F*	PSO	HPSOFminLS-Worst-E
1	100	64321000	**100.0006**
2	200	5203700000	**200**
3	300	320.7218	**320**
4	400	595.5969	457.7075
5	500	6328.3	2146.4
6	600	907480	1243.7
7	700	730.6795	**712.3987**
8	800	315980	1230.6
9	900	1027.5	1004.6
10	1000	336240	1534.9
11	1100	1483.6	1402.8
12	1200	1318	1307.8
13	1300	1419.2	**1408.2**
14	1400	37512	2029
15	1500	1633.6	1605

Table 4.10: Results of Fmin for the global worst solution and at the End for PSO in D50

#	F*	PSO	HPSOFminLS-Worst-E
1	100	320780000	**100.002**
2	200	17588000000	**200.0013**
3	300	321.0368	**320**
4	400	830.9402	500.4907
5	500	12864	4403.8
6	600	11640000	2381.3
7	700	844.3723	**723.327**
8	800	3156100	1974.9
9	900	1070.6	**1007.2**
10	1000	4990400	1698.3
11	1100	2860.7	2708.6
12	1200	1337.8	**1308.1**
13	1300	1522.1	1494.4
14	1400	69339	18231
15	1500	1889.9	1619.4

4.3.4 Results of comparing original PSO with Hybrid PSO Version 4 (HPSOFminLS-B-W-E)

Table 4.11: Results of Fmin at the beginning, for the global worst solution, and at the End for PSO in D10

#	F*	PSO	HPSOFminLS-B-Worst-E
1	100	387020	**100**
2	200	82782000	**200**
3	300	320.1884	312
4	400	426.0494	**404**
5	500	1165.5	**507**
6	600	3343.4	709
7	700	703.3774	**702**
8	800	1987.5	818
9	900	1000.4	1000
10	1000	1608.7	1100
11	1100	1118.6	**1110**
12	1200	1303.3	1300
13	1300	1334.2	1330
14	1400	2986.6	1500
15	1500	1610.2	1600

Table 4.12: Results of Fmin at the beginning, for the global worst solution, and at the End for PSO in D30

#	F*	PSO	HPSOFminLS-B-Worst-E
1	100	64321000	**100.0011**
2	200	5203700000	**200**
3	300	320.7218	**320**
4	400	595.5969	442.7832
5	500	6328.3	2108.9
6	600	907480	1216.9
7	700	730.6795	**709.6085**
8	800	315980	1198.1
9	900	1027.5	1004.5
10	1000	336240	1562.9
11	1100	1483.6	1403
12	1200	1318	1305
13	1300	1419.2	**1402.1**
14	1400	37512	2044.6
15	1500	1633.6	1603.5

Table 4.13: Results of Fmin at the beginning, for the global worst solution, and at the End for PSO in D50

#	F*	PSO	HPSOFminLS-B-Worst-E
1	100	320780000	**100.002**
2	200	17588000000	**200.0031**
3	300	321.0368	**320**
4	400	830.9402	512.4301
5	500	12864	4040.9
6	600	11640000	2203.6
7	700	844.3723	**712.0535**
8	800	3156100	1368.9
9	900	1070.6	**1008**
10	1000	4990400	1765.4
11	1100	2860.7	1405.1
12	1200	1337.8	**1305.8**
13	1300	1522.1	1407.5
14	1400	69339	15742
15	1500	1889.9	1612..5

Table 4.14 till Table 4.16 show the result of comparing between the 4 versions of Hybrid PSO, Results of optimal solutions were obtained in all dimensions for the 4 versions of **Unimodal functions**.

Table 4.14 for D10 results for problem 15 was same in all the versions with the value 1600. In addition, among all the results of the versions, **Simple Multimodal function** for version HPSOFminLS-Worst-E showed the best two results in problem **3** and **4** where as the last version HPSOFminLS-B-Worst-E obtained the near optimal solution in **problem 4.** In **Hybrid functions,** version HPSOFminLS-Worst-E obtained the best result among the other versions. For the rest of the functions, version HPSOFminLS-B-E obtained the best near optimal solution in **11 and 13** problem ,while in version HPSOFminLS-B-Worst-E obtained the best solution near optimal solution in **problem 9,10,12, and 14.**

Table 4.15 for D30 shows that the results for problem 3 was same in all the versions with 320 value , however, for **Simple Multimodal function,** version HPSOFminLS-B-Worst-E obtained best solution among the other version. Also in **Hybrid function,** the version obtained best solutions in **problem 6 and 7.** The best near optimal in **problem 8** was obtained using version HPSOFminLS-B-E. Moreover, in function version HPSOFminLS-E got best near optimal in **problem 9 and 1.** For HPSOFminLS-Worst-E had it in problem **11 and 14**, while version HPSOFminLS-B-Worst-E had it in problem **12 and 15**.

Table 4.16 for D50 similar with D30 result of problem 3 was the same in all the version with over 20 above the optimal solution.

Best near optimal solution was obtained by HPSOFminLS-E version in **problem 6** and version HPSOFminLS-B-E in **problem 9**, and by version HPSOFminLS-Worst-E for **problem 4 and 10.** The last version HPSOFminLS-B-Worst-E had the majority of best solution in **problem 5,7,8,10,11,12,13,14, and 15.**

4.3.5 Results of Comparing 4 Hybrid PSO Versions

Table 4.14: Results of D10 for Comparing all the versions

#	F*	PSO	HPSOFminLS-E	HPSOFminLS-B-E	HPSOFminLS-Worst-E	HPSOFminLS-B-Worst-E
1	100	387020	**100.0131**	**100**	**100**	**100**
2	200	82782000	**200.0176**	**200.023**	**200**	**200**
3	300	320.1884	319.9999	315.6645	**301.6462**	312
4	400	426.0494	407.9597	404.9748	**402.9849**	404
5	500	1165.5	653.6771	518.4721	510.3073	**507**
6	600	3343.4	748.8018	735.5666	**665.0878**	709
7	700	703.3774	701.8066	702.0541	**701.5606**	702
8	800	1987.5	875.5819	818.1371	**817.7843**	818
9	900	1000.4	1000.2	1000.3	1000.3	**1000**
10	1000	1608.7	1226.9	1226	1118.5	**1100**
11	1100	1118.6	1109.8	**1108**	1109.1	1110
12	1200	1303.3	1301.2	1301.9	1301.1	**1300**
13	1300	1334.2	1330.3	**1328.9**	1329.6	1330
14	1400	2986.6	1500.1	**1500**	1700	**1500**
15	1500	1610.2	**1600**	**1600**	**1600**	**1600**

Table 4.15: Results of D30 for Comparing all the versions

#	F*	PSO	HPSOFminLS-E	HPSOFminLS-B-E	HPSOFminLS-Worst-E	HPSOFminLS-B-Worst-E
1	100	64321000	**100.0131**	**100.0004**	**100.0006**	**100.0011**
2	200	5203700000	**200.0013**	**200.0061**	**200**	**200**
3	300	320.7218	**320**	**320**	**320**	**320**
4	400	595.5969	448.7529	444.7731	457.7075	**442.7832**
5	500	6328.3	2130	2585.9	2146.4	**2108.9**
6	600	907480	1429.7	1390.9	1243.7	**1216.9**
7	700	730.6795	714.7427	714.9418	712.3987	**709.6085**
8	800	315980	1383	**1187.8**	1230.6	1198.1
9	900	1027.5	**1004.2**	1004.7	1004.6	1004.5
10	1000	336240	1714.2	1721.1	**1534.9**	1562.9
11	1100	1483.6	1433.6	1476.9	**1402.8**	**1403**
12	1200	1318	1307.3	1307.1	1307.8	**1305**
13	1300	1419.2	**1398.2**	1404	1408.2	1402.1
14	1400	37512	2052.5	2054.6	**2029**	2044.6
15	1500	1633.6	1610.4	1611.4	1605	**1603.5**

Table 4.16: Results of D50 for Comparing 4 Hybrid PSO Versions

#	F*	PSO	HPSOFminLS-E	HPSOFminLS-B-E	HPSOFminLS-Worst-E	HPSOFminLS-B-Worst-E
1	100	320780000	**100.002**	**100.0018**	**100.002**	**100.002**
2	200	17588000000	**200.1692**	**200.1132**	**200.0013**	**200.0031**
3	300	321.0368	**320**	**320**	**320**	**320**
4	400	830.9402	518.3999	504.4705	**500.4907**	512.4301
5	500	12864	4479.6	4739.7	4403.8	**4040.9**
6	600	11640000	**1379.9**	2252.5	2381.3	2203.6
7	700	844.3723	734.7006	734.144	723.327	**712.0535**
8	800	3156100	1645.4	1832.6	1974.9	**1368.9**
9	900	1070.6	1008.1	**1006.5**	1007.2	1008
10	1000	4990400	2109.9	2143.3	**1698.3**	1765.4
11	1100	2860.7	2780.1	1889.1	2708.6	**1405.1**
12	1200	1337.8	1308.6	1308.6	1308.1	**1305.8**
13	1300	1522.1	1495	1499.1	1494.4	**1407.5**
14	1400	69339	35302	35288	18231	**15742**
15	1500	1889.9	1617.2	1616.6	1619.4	**1612.5**

From table 4.17 till table 4.28, the results of comparing between the 4 versions are demonstrated. Then, results and other work results for solving CEC15 problems. The results were obtained according to the function error values which is the result of (F(X)-F(X*)) X is the result of problems, where X* is the optimal solution for the same problems.

Optimal solution for Unimodal functions problems were obtained by all HPSO-FminLS 4 versions and the others literature functions.
For dimension 10, HPSO-FminLS in the 4 versions obtained the optimal solution in Unimodal problems.

In the last version, HPSO-FminLS-B-W-E, the near optimal solutions were obtained in **Simple Multimodal.** The result was 12 in **problem 3** and 4,7 in **problem 4, 5** in the error function. for **Hybrid problems** in problem **6,7,8**, the results were 109,2,18 in the error function. For **Composition functions,** problem **11** obtained near optimal solution with near 10 for the error function, as for problem 13 gave 30 as a result. In problems **9,10,11,12**, it had obtained results by 100 for error function. sDMS-PSO proposed methods obtained the optimal solution in **Unimodal function** and in **Composition functions** for problem **9,14 and 15**. It had 100 as a result for error function. ABC-X-LS, hCC, DEsPA and LSHADE-ND had optimal solution in problems 1,2,3,4,5,6,7, and 8 in error rate.

For dimension 30, all the algorithms obtained 20 value solution in problem 3 and 100 value solution for problem **15** except for DEsPA which obtained 273 above the optimal solution. The best near optimal solution was obtained by ABC-X-LS algorithms with

6 out of 11 in problem **5,8,9,12,14 and 15**. Also it obtained the optimal one in problem **4**.

For dimension 50, all the algorithms obtained the same result by 20 above the optimal solution except for hCC had the max result with 20.3. The best result was in **problem 1,2, and 3 for four Version of HPSO-FminLS**. ABC-X-LS had the majority of obtaining two optimal solutions as result for **Unimodal problems** and best near optimal solution in all **Simple Multimodal problems** and in Composite function for problems **9,11,12,14, and 15**. However, the best near optimal result for Hybrid functions was obtained by DEsPA in **problem 6**, and algorithm LSHADE-ND in problem **7 and 8.**

4.3.6 Results of Comparing version HPSO-FminLS-E V1 with other works

Table 4.17: Results of D10 for Comparing HPSO-FminLS V1 with others work

#	F*	PSO	HPSO-FminLS-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	386920	**1.31E-02**	**7.41E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
2	200	82781800	**1.76E-02**	**3.57E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
3	300	20.1884	2.00E+01	1.99E+01	**0.00E+00**	4.14E+00	**0.00E+00**	**0.00E+00**
4	400	26.0494	7.96E+00	9.95E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	9.95E-01
5	500	665.5	1.54E+02	3.12E-01	**1.43E-01**	1.87E-01	6.75E-01	2.50E-01
6	600	2743.4	1.49E+02	1.22E+01	**3.27E-04**	0.00E+00	2.08E-01	0.00E+00
7	700	3.3774	1.81E+00	1.29E-01	**1.00E-02**	1.91E-02	4.58E-02	0.00E+00
8	800	1187.5	7.56E+01	1.24E+00	1.36E-01	**2.44E-05**	2.19E-06	4.52E-06
9	900	100.4	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.02E+02	**1.00E+02**
10	1000	608.7	2.27E+02	2.45E+02	1.95E+02	1.41E+02	**6.90E+00**	2.17E+02
11	1100	18.6	9.80E+00	2.34E+00	4.53E+00	1.32E+00	**2.17E-01**	7.12E-01
12	1200	103.3	1.01E+02	1.01E+02	**1.00E+02**	1.01E+02	**1.00E+02**	**1.00E+02**
13	1300	34.2	3.03E+01	2.54E+01	2.11E+01	**3.03E-02**	1.12E+01	3.04E-02
14	1400	1586.6	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.93E+02	**1.00E+02**	**1.00E+02**
15	1500	110.2	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.05E+02	**1.00E+02**

Table 4.18: Results of D30 for Comparing HPSO-FminLS-E with others work

#	F*	PSO	HPSO-FminLS-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	64320900	**1.31E-02**	**5.13E-04**	**0.00E+00**	**1.56E-13**	**0.00E+00**	**0.00E+00**
2	200	5203699800	**1.30E-03**	**8.07E-04**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	20.7218	**2.00E+01**	**2.00E+01**	**2.00E+01**	2.01E+01	**2.00E+01**	**2.00E+01**
4	400	195.5969	4.88E+01	2.49E+01	**0.00E+00**	5.22E+00	3.98E+00	4.98E+00
5	500	5828.3	1.63E+03	1.59E+03	**3.48E+00**	2.59E+02	9.48E+02	7.02E+02
6	600	906880	8.30E+02	5.64E+02	7.41E+01	4.50E+01	**2.72E+01**	4.48E+01
7	700	30.6795	1.47E+01	5.83E+00	3.21E+00	2.25E+00	**1.07E+00**	3.65E+00
8	800	315180	5.83E+02	5.38E+02	**2.05E+00**	1.15E+01	3.40E+00	2.34E+00
9	900	127.5	1.04E+02	1.03E+02	**1.02E+02**	1.06E+02	1.16E+02	**1.02E+02**
10	1000	335240	7.14E+02	2.61E+03	6.22E+02	4.15E+02	**3.50E+01**	3.32E+02
11	1100	383.6	3.34E+02	3.06E+02	3.01E+02	3.18E+02	**2.01E+02**	4.00E+02
12	1200	118	1.07E+02	1.03E+02	**1.02E+02**	1.04E+02	1.08E+02	1.03E+02
13	1300	119.2	9.82E+01	8.97E+01	8.29E+01	**2.51E-02**	6.93E+01	**2.56E-02**
14	1400	36112	**6.53E+02**	1.75E+04	**1.00E+02**	3.11E+04	2.73E+04	3.11E+04
15	1500	133.6	1.10E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.73E+02	**1.00E+02**

Table 4.19: Results of D50 for Comparing HPSO-FminLS-E with others work

#	F*	PSO	HPSO-FminLS -E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	3.21E+08	**2.00E-03**	**1.13E+00**	**0.00E+00**	**1.69E-12**	**3.36E-01**	**8.35E+01**
2	200	1.76E+10	**3.10E-03**	**3.39E-02**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	2.10E+01	**2.00E+01**	**2.00E+01**	**2.00E+01**	2.03E+01	**2.00E+01**	**2.00E+01**
4	400	4.31E+02	1.12E+02	5.77E+01	**2.98E+00**	1.53E+01	*6.96E+00*	9.97E+00
5	500	1.24E+04	3.54E+03	3.55E+03	**2.23E+00**	6.24E+02	2.68E+02	*2.09E+02*
6	600	1.16E+07	1.60E+03	1.68E+03	1.08E+02	3.67E+02	**1.77E+01**	*5.36E+01*
7	700	1.44E+02	1.21E+01	9.80E+00	1.03E+01	*8.93E+00*	3.95E+01	**4.70E+00**
8	800	3.16E+06	5.69E+02	7.70E+02	*1.43E+01*	1.72E+01	5.08E+01	**1.10E+01**
9	900	1.71E+02	1.08E+02	*1.04E+02*	**1.04E+01**	**1.04E+01**	1.18E+02	*1.04E+02*
10	1000	4.99E+06	7.65E+02	6.58E+03	*1.02E+02*	1.10E+02	3.33E+02	**8.04E+01**
11	1100	1.76E+03	3.05E+02	3.07E+02	**3.00E+01**	*3.49E+01*	3.06E+02	4.00E+02
12	1200	1.38E+02	1.06E+02	1.07E+02	**1.03E+01**	**1.07E+01**	1.07E+02	1.04E+02
13	1300	2.22E+02	1.08E+02	1.79E+02	*1.61E+01*	**7.15E-02**	1.29E+02	**7.10E-02**
14	1400	6.79E+04	1.57E+04	*1.49E+04*	**1.00E+02**	4.95E+04	2.98E+04	4.95E+04
15	1500	3.90E+02	*1.15E+02*	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.83E+02	**1.00E+02**

4.3.7 Results of Comparing version HPSO-FminLS-B-E V2 with other works

Table 4.20: Results of D10 for Comparing version HPSO-FminLS-B-E

#	F*	PSO	HPSO-FminLS-B-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	3.87E+05	**0.00E+00**	**7.41E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
2	200	8.28E+07	**2.30E-02**	**3.57E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
3	300	2.02E+01	1.57E+01	1.99E+01	**0.00E+00**	4.14E+00	**0.00E+00**	**0.00E+00**
4	400	2.60E+01	4.97E+00	9.95E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	9.95E-01
5	500	6.66E+02	1.85E+01	3.12E-01	**1.43E-01**	1.87E-01	6.75E-01	2.50E-01
6	600	2.74E+03	1.36E+02	1.22E+01	**3.27E-04**	**0.00E+00**	2.08E-01	**0.00E+00**
7	700	3.38E+00	2.05E+00	1.29E-01	**1.00E-02**	1.91E-02	4.58E-02	**0.00E+00**
8	800	1.19E+03	1.81E+01	1.24E+00	1.36E-01	2.44E-05	**2.19E-06**	**4.52E-06**
9	900	1.00E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.02E+02	**1.00E+02**
10	1000	6.09E+02	2.26E+02	2.45E+02	1.95E+02	1.41E+02	**6.90E+00**	2.17E+02
11	1100	1.86E+01	8.00E+00	2.34E+00	4.53E+00	1.32E+00	**2.17E-01**	7.12E-01
12	1200	1.03E+02	1.02E+02	1.01E+02	1.00E+02	1.01E+02	1.00E+02	1.00E+02
13	1300	3.42E+01	2.89E+01	2.54E+01	2.11E+01	**3.03E-02**	1.12E+01	**3.04E-02**
14	1400	1.59E+03	1.50E+07	**1.00E+02**	**1.00E+02**	2.93E+02	**1.00E+02**	**1.00E+02**
15	1500	1.10E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.05E+02	**1.00E+02**

Table 4.21: Results of D30 for Comparing version HPSO-FminLS-B-E

#	F*	PSO	HPSO-FminLS-B-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	6.43E+07	**4.00E-04**	**5.13E-04**	**0.00E+00**	**1.56E-13**	**0.00E+00**	**0.00E+00**
2	200	5.20E+09	**6.10E-03**	**8.07E-04**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	2.07E+01	**2.00E+01**	**2.00E+01**	**2.00E+01**	2.01E+01	**2.00E+01**	**2.00E+01**
4	400	1.96E+02	4.48E+01	2.49E+01	**0.00E+00**	5.22E+00	3.98E+00	4.98E+00
5	500	5.83E+03	2.09E+03	1.59E+03	**3.48E+00**	2.59E+02	9.48E+02	7.02E+02
6	600	9.07E+05	7.91E+02	5.64E+02	7.41E+01	4.50E+01	**2.72E+01**	4.48E+01
7	700	3.07E+01	1.49E+01	5.83E+00	3.21E+00	2.25E+00	**1.07E+00**	3.65E+00
8	800	3.15E+05	3.88E+02	5.38E+02	**2.05E+00**	1.15E+01	3.40E+00	2.34E+00
9	900	1.28E+02	1.05E+02	1.03E+02	**1.02E+02**	1.06E+02	1.16E+02	**1.02E+02**
10	1000	3.35E+05	721.1	2.61E+03	6.22E+02	4.15E+02	**3.50E+01**	3.32E+02
11	1100	3.84E+02	3.77E+02	3.06E+02	3.01E+02	3.18E+02	**2.01E+02**	4.00E+02
12	1200	1.18E+02	1.07E+02	1.03E+02	**1.02E+02**	1.04E+02	1.08E+02	1.03E+02
13	1300	1.19E+02	1.04E+02	8.97E+01	8.29E+01	**2.51E-02**	6.93E+01	2.56E-02
14	1400	3.61E+04	**6.55E+02**	1.75E+04	**1.00E+02**	3.11E+04	2.73E+04	3.11E+04
15	1500	1.34E+02	1.11E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.73E+02	**1.00E+02**

Table 4.22: Results of D50 for Comparing version HPSO-FminLS-B-E

#	F*	PSO	HPSO-FminLS-B-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	6.43E+07	**1.80E-03**	**1.13E+00**	**0.00E+00**	**1.69E-12**	**3.36E-01**	8.35E+01
2	200	5.20E+09	**1.13E-01**	**3.39E-02**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	320.7218	**2.00E+01**	**2.00E+01**	**2.00E+01**	**2.03E+01**	**2.00E+01**	**2.00E+01**
4	400	595.5969	1.04E+02	5.77E+01	**2.98E+00**	1.53E+01	6.96E+00	9.97E+00
5	500	6.33E+03	4.24E+03	3.55E+03	**2.23E+00**	*6.24E+02*	2.68E+03	2.09E+03
6	600	9.07E+05	1.65E+03	1.68E+03	1.08E+03	3.67E+03	**1.77E+02**	5.36E+02
7	700	730.6795	3.41E+01	9.80E+00	1.03E+01	8.93E+00	3.95E+01	**4.70E+00**
8	800	3.16E+05	1.03E+03	7.70E+02	*1.43E+01*	1.72E+02	5.08E+01	**1.10E+01**
9	900	1.03E+03	1.07E+02	1.04E+02	**1.04E+01**	1.04E+02	1.18E+02	1.04E+02
10	1000	3.36E+05	1.14E+03	6.58E+03	**1.02E+02**	1.10E+03	3.33E+02	8.04E+02
11	1100	1.48E+03	7.89E+02	3.07E+02	**3.00E+02**	3.49E+02	3.06E+02	4.00E+02
12	1200	1.32E+03	1.09E+02	1.07E+02	**1.03E+02**	1.07E+02	1.07E+02	*1.04E+02*
13	1300	1.42E+03	1.99E+02	1.79E+02	1.61E+02	*7.15E-02*	1.29E+02	**7.10E-02**
14	1400	3.75E+04	3.39E+04	1.49E+04	**1.00E+02**	4.95E+04	2.98E+04	4.95E+04
15	1500	1.63E+03	*1.17E+02*	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.83E+02	**1.00E+02**

4.3.8 Results of Comparing version HPSO-FminLS-W-E V3 with other works

Table 4.23: Results of D10 for Comparing version HPSO-FminLS-W-E with other works

#	F*	PSO	HPSO-FminLS-W-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	387020	**0.00E+00**	**7.41E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
2	200	82782000	**0.00E+00**	**3.57E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
3	300	320.1884	*1.65E+00*	1.99E+01	**0.00E+00**	4.14E+00	**0.00E+00**	**0.00E+00**
4	400	426.0494	2.98E+00	*9.95E-01*	**0.00E+00**	**0.00E+00**	**0.00E+00**	*9.95E-01*
5	500	1165.5	1.03E+01	3.12E-01	**1.34E-01**	*1.87E-01*	6.75E-01	2.50E-01
6	600	3343.4	6.51E+01	1.22E+01	**3.27E-04**	*0.00E+00*	2.08E-01	*0.00E+00*
7	700	703.3774	1.56E+00	1.29E-01	1.00E+02	**1.91E-02**	4.58E-02	*0.00E+00*
8	800	1987.5	1.78E+01	1.24E+00	**1.36E-01**	*2.44E-05*	2.19E-06	4.52E+06
9	900	1000.4	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.02E+02	1.00E+02
10	1000	1608.7	1.19E+02	2.45E+02	1.95E+02	1.41E+02	**6.90E+00**	2.17E+02
11	1100	1118.6	9.10E+00	2.34E+00	4.53E+00	1.32E+00	**2.17E-01**	*7.12E-01*
12	1200	1303.3	*1.01E+02*	*1.01E+02*	**1.00E+02**	*1.01E+02*	**1.00E+02**	**1.00E+02**
13	1300	1334.2	2.96E+01	2.54E+01	2.11E+01	**3.03E-02**	1.12E+00	*3.04E-02*
14	1400	2986.6	3.00E+02	**1.00E+02**	**1.00E+02**	2.93E+02	**1.00E+02**	**1.00E+02**
15	1500	1610.2	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.05E+02	**1.00E+02**

Table 4.24: Results of D30 for Comparing version HPSO-FminLS-W-E with other works

#	F*	PSO	HPSO-FminLS-W-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	387020	**6.00E-04**	**5.13E-04**	**0.00E+00**	**1.56E-13**	**0.00E+00**	**0.00E+00**
2	200	82782000	**0.00E+00**	**8.07E-04**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	320.1884	**2.00E+01**	**2.00E+01**	**2.00E+01**	**2.01E+01**	**2.00E+01**	**2.00E+01**
4	400	426.0494	5.77E+01	2.49E+01	**0.00E+00**	5.22E+00	*3.98E+00*	4.98E+00
5	500	1165.5	1.65E+03	1.59E+03	**3.48E+00**	*2.59E+02*	9.48E+02	7.02E+02
6	600	3343.4	6.44E+02	5.64E+02	7.41E+01	4.50E+01	**2.72E+01**	*4.48E+01*
7	700	703.3774	1.24E+01	5.83E+00	3.21E+00	*2.25E+00*	**1.07E+00**	3.65E+00
8	800	1987.5	4.31E+02	5.38E+02	**2.05E+00**	1.15E+01	3.40E+00	*2.34E+00*
9	900	1000.4	1.05E+02	*1.03E+02*	**1.02E+02**	1.06E+02	1.16E+02	**1.02E+02**
10	1000	1608.7	5.35E+02	2.61E+03	6.22E+02	4.15E+02	**3.50E+01**	*3.32E+02*
11	1100	1118.6	3.03E+02	3.06E+02	*3.01E+02*	3.18E+02	**2.01E+01**	4.00E+02
12	1200	1303.3	1.08E+02	1.03E+02	*1.02E+02*	1.04E+02	**1.08E+01**	1.03E+02
13	1300	1334.2	1.08E+02	8.97E+01	8.29E+01	**2.51E-02**	6.93E+01	*2.56E-02*
14	1400	2986.6	*6.29E+02*	1.75E+04	**1.00E+02**	3.11E+04	2.73E+03	3.11E+04
15	1500	1610.2	1.05E+02	*1.00E+02*	*1.00E+02*	*1.00E+02*	**2.73E+01**	*1.00E+02*

Table 4.25: Results of D50 for comparing HPSOFminLS-W-E with other works.

#	F*	PSO	HPSO-FminLS-W-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	387020	**2.00E-03**	**1.13E+00**	**0.00E+00**	**1.69E-12**	***3.36E-01***	**8.35E+01**
2	200	82782000	***1.30E-03***	**3.39E-02**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	320.1884	**2.00E+01**	**2.00E+01**	**2.00E+01**	2.03E+01	**2.00E+01**	**2.00E+01**
4	400	426.0494	1.00E+02	5.77E+01	**2.98E+00**	1.53E+01	*6.96E+00*	9.97E+00
5	500	1165.5	3.90E+03	3.55E+03	**2.23E+00**	6.24E+02	2.68E+02	2.09E+02
6	600	3343.4	1.78E+03	1.68E+03	1.08E+02	3.67E+02	**1.77E+01**	5.36E+01
7	700	703.3774	2.33E+01	9.80E+00	1.03E+01	8.93E+00	3.95E+01	**4.70E+00**
8	800	1987.5	1.17E+03	7.70E+02	*1.43E+01*	1.72E+01	5.08E+01	**1.10E+01**
9	900	1000.4	1.07E+02	1.04E+02	1.04E+01	1.04E+01	1.18E+02	1.04E+02
10	1000	1608.7	6.98E+02	6.58E+03	*1.02E+02*	1.10E+02	3.33E+02	**8.04E+01**
11	1100	1118.6	1.61E+03	3.07E+02	**3.00E+01**	*3.49E+01*	3.06E+02	4.00E+02
12	1200	1303.3	1.08E+02	1.07E+02	**1.03E+01**	*1.07E+01*	1.07E+02	1.04E+02
13	1300	1334.2	1.94E+02	1.79E+02	*1.61E+01*	**7.15E-02**	1.29E+02	7.10E-02
14	1400	2986.6	1.68E+04	*1.49E+04*	**1.00E+02**	4.95E+04	2.98E+04	4.95E+04
15	1500	1610.2	*1.19E+02*	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.83E+02	**1.00E+02**

4.3.9 Results of Comparing version HPSO-FminLS-B-W-E V4 with other works

Table 4.26: Results of D10 for Comparing version HPSO-FminLS-B-W-E V4 with other works

#	F*	PSO	HPSO-FminLS-B-W-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	3.87E+05	**0.00E+00**	**7.41E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
2	200	8.28E+07	**0.00E+00**	**3.57E-05**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
3	300	2.02E+01	*1.20E+01*	*1.99E+01*	**0.00E+00**	4.14E+00	**0.00E+00**	**0.00E+00**
4	400	2.60E+01	4.00E+00	*9.95E-01*	**0.00E+00**	**0.00E+00**	**0.00E+00**	*9.95E-01*
5	500	6.66E+02	7.00E+00	3.12E-01	**1.43E-01**	1.87E-01	6.75E-01	2.50E-01
6	600	2.74E+03	1.09E+02	1.22E+01	***3.27E-04***	**0.00E+00**	2.08E-01	**0.00E+00**
7	700	3.38E+00	2.00E+00	1.29E-01	**1.00E-02**	**1.91E-02**	4.58E-02	**0.00E+00**
8	800	1.19E+03	1.80E+01	1.24E+00	1.36E-01	**2.44E-05**	**2.19E-06**	**4.52E-06**
9	900	1.00E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.02E+02	**1.00E+02**
10	1000	6.09E+02	*1.00E+02*	2.45E+02	1.95E+02	1.41E+02	**6.90E+00**	2.17E+02
11	1100	1.86E+01	1.00E+01	2.34E+00	4.53E+00	1.32E+00	**2.17E-01**	*7.12E-01*
12	1200	1.03E+02	**1.00E+02**	1.01E+02	**1.00E+02**	1.01E+02	**1.00E+02**	**1.00E+02**
13	1300	3.42E+01	3.00E+01	2.54E+01	2.11E+01	**3.03E-02**	1.12E+01	*3.04E-02*
14	1400	1.59E+03	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.93E+02	**1.00E+02**	**1.00E+02**
15	1500	1.10E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.05E+02	**1.00E+02**

Table 4.27: Results of D30 for Comparing version HPSO-FminLS-B-W-E V4 with other works

#	F*	PSO	HPSO-FminLS-B-W-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	6.43E+07	**1.10E-03**	**5.13E-04**	**0.00E+00**	**1.56E-13**	**0.00E+00**	**0.00E+00**
2	200	5.20E+09	**0.00E+00**	**8.07E-04**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	2.07E+01	**2.00E+01**	**2.00E+01**	**2.00E+01**	**2.01E+01**	**2.00E+01**	**2.00E+01**
4	400	1.96E+02	4.28E+01	2.49E+01	**0.00E+00**	5.22E+00	*3.98E+00*	4.98E+00
5	500	5.83E+03	1.61E+03	1.59E+03	**3.48E+00**	*2.59E+02*	9.48E+02	7.02E+02
6	600	9.07E+05	6.17E+02	5.64E+02	7.41E+01	4.50E+01	**2.72E+01**	*4.48E+01*
7	700	3.07E+01	9.61E+00	5.83E+00	3.21E+00	*2.25E+00*	**1.07E+00**	3.65E+00
8	800	3.15E+05	3.98E+02	5.38E+02	**2.05E+00**	1.15E+01	3.40E+00	*2.34E+00*
9	900	1.28E+02	1.05E+02	*1.03E+02*	**1.02E+02**	1.06E+02	1.16E+02	**1.02E+02**
10	1000	3.35E+05	5.63E+02	2.61E+03	6.22E+02	4.15E+02	**3.50E+01**	*3.32E+02*
11	1100	3.84E+02	3.03E+02	3.06E+02	*3.01E+02*	3.18E+02	**2.01E+02**	4.00E+02
12	1200	1.18E+02	1.05E+02	*1.03E+02*	**1.02E+02**	1.04E+02	1.08E+02	*1.03E+02*
13	1300	1.19E+02	1.02E+02	8.97E+01	8.29E+01	**2.51E-02**	6.93E+01	*2.56E-02*
14	1400	3.61E+04	*6.45E+02*	1.75E+04	**1.00E+02**	3.11E+04	2.73E+04	3.11E+04
15	1500	1.34E+02	*1.04E+02*	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.73E+02	**1.00E+02**

Table 4.28: Results of D50 for Comparing version HPSO-FminLS-B-W-E V3with other works

#	F*	PSO	HPSO-FminLS-B-W-E	sDMS-PSO	ABC-X-LS	hCC	DEsPA	LSHADE-ND
1	100	3.21E+08	**2.00E-03**	**1.13E+00**	**0.00E+00**	**1.69E-12**	**3.36E-01**	**8.35E+01**
2	200	1.76E+10	**3.10E-03**	**3.39E-02**	**0.00E+00**	**2.84E-14**	**0.00E+00**	**0.00E+00**
3	300	2.10E+01	**2.00E+01**	**2.00E+01**	**2.00E+01**	2.03E+01	**2.00E+01**	**2.00E+01**
4	400	4.31E+02	1.12E+02	5.77E+01	**2.98E+00**	1.53E+01	*6.96E+00*	9.97E+00
5	500	1.24E+04	3.54E+03	3.55E+03	**2.23E+00**	6.24E+02	2.68E+02	*2.09E+02*
6	600	1.16E+07	1.60E+03	1.68E+03	1.08E+02	3.67E+02	**1.77E+01**	*5.36E+01*
7	700	1.44E+02	1.21E+01	9.80E+00	1.03E+01	*8.93E+00*	3.95E+01	**4.70E+00**
8	800	3.16E+06	5.69E+02	7.70E+02	*1.43E+01*	1.72E+01	5.08E+01	**1.10E+01**
9	900	1.71E+02	1.08E+02	*1.04E+02*	**1.04E+01**	**1.04E+01**	1.18E+02	*1.04E+02*
10	1000	4.99E+06	7.65E+02	6.58E+03	*1.02E+02*	1.10E+02	3.33E+02	**8.04E+01**
11	1100	1.76E+03	3.05E+02	3.07E+02	**3.00E+01**	*3.49E+01*	3.06E+02	4.00E+02
12	1200	1.38E+02	1.06E+02	1.07E+02	**1.03E+01**	**1.07E+01**	1.07E+02	1.04E+02
13	1300	2.22E+02	1.08E+02	1.79E+02	*1.61E+01*	**7.15E-02**	1.29E+02	**7.10E-02**
14	1400	6.79E+04	1.57E+04	*1.49E+04*	**1.00E+02**	4.95E+04	2.98E+04	4.95E+04
15	1500	3.90E+02	*1.15E+02*	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.83E+02	**1.00E+02**

4.4 Ranking for D30 for all the versions

This ranking had been conducted by Friedman Test [18]. Friedman is a high statistical test that concludes to a specific difference between median of the given data values for null hypothesis. *P*-value is the value of probability that evaluate the decision of accepting the hypothesis or not [19]. Table 4.29 presents the results of Ranking for D30 of four HPSO-FminLS proposed versions.

Table 4.29 Friedman's test results of HPSO-FminLS versions comparison in D30

Ranking	Algorithm hybrid PSO with Fmin versions
1 Best version	**HPSO-FminLS-B-Worst-E**
2	HPSO-FminLS-Worst-E
3	HPSO-FminLS-B-E
4	HPSO-FminLS-E
5	PSO

To evaluate the 4 versions of HPSO-FminLS by checking the statistical similarity of the algorithms results, we implemented the Friedman aligned ranks test for global best value of each CEC2015 benchmark problems. The *p*-value of the test was 8.9899E-8. Table 4.29 shows that version **HPSO-FminLS-B-W-E** had the best performing algorithm among the other versions. Meanwhile, the *p*-value is very close to zero, indicating that there is significant statistical difference among the results of all versions, such that the **HPSO-FminLS-B-W-E** is statistically different from the other versions [22].

Chapter 5

CONCLUSION

5.1 Summary of the Study

Optimizing the 15 Benchmark of single objective problems of **CEC*2015*** [11] with Hybrid Particle Swarm Optimization using Fmin function as a local search was proposed. The main idea is searching for the global best solution using simple FminLS to balance between diversification-based and intensification-based searches in the proposed algorithm. In this thesis we developed four different versions of HPSO-FminLS that was applied to 3 dimensions D10, D30, D50. Moreover, we compared findings of HPSO-FminLS with the previously proposed algorithms for solving single objective optimization CEC2015 problems. Finally, we compared the findings of the four versions by using Friedman Ranking Test.

5.2 Conclusions

Two main conclusions are derived. The proposed algorithms reached optimal best solutions **F*** in **Unimodal problems,** and Global best solution had been better improved using **HPSO-FminLS** algorithm compared with the proposed PSO algorithm. On the other hand, the findings of the proposed algorithm could not improve the optimization solutions compared with literature.

5.3 Limitation of the Study

According to the findings of employing the Particle Swarm Optimization searching techniques, the results of the four proposed versions of HPSO-FminLS could not improve experiment results compared to other works in the Simple Multimodal functions, Hybrid functions, and Composition function categories.

5.4 Implications for Further Research

Further research work should focus on improving the Global Best solution of PSO by using another Local Search mechanism.

REFERENCES

[1] Margaret Rouse, Computer Science, Computing fundamentals Wide Web: http://whatis.techtarget.com/definition/heuristic.

[2] Wikipedia, the free encyclopedia Wide Web: https://en.wikipedia.org/wiki/Meta.

[3] Xin-She Yang, Nature-Inspired Optimization Algorithms,1-21, ResearchGate, March 2014.

[4] Alec Banks, Jonathan Vincent, Chukwudi Anyakoha," A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications, Springer,DOI: 10.1007/s11047-007-9050-z,2008.

[5] Thomas Back, David.B.Fogel, Zbigniew Michalewicz, Evolutionary Computation 1 Basic Algorithms and Operators ,Institute of Physics Publishing, ANSI Z39.48-1984,2000.

[6] El -Ghazali Talbi, METAHEURISTICS form Design to Implementation, A Joha Wiley & Sons, INC., Publication Canada, 2009.

[7] J Liang, L. Guo, R. Liu,B . Y. Qu,"A Self-adaptive Dynamic Particle Swarm Optimizer", IEEE, Dot:978-1-4799-7492-4/15/$31.00, 2015.

[8] Kennedy J, Eberhart R. (1995). “Particle Swarm Optimization.” Proceedings of the IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, 1942-1948.

[9] Shi Y, Eberhart R. (1998). “A Modified Particle Swarm Optimizer.” Proceedings of the IEEE International Conference on Evolutionary Computation, Piscataway, NJ; IEEE Press, 69-73.

[10] Yu-Jun Zheng,Xiao-Bei Wu,Tuning Maturity Model of Ecogeography-Based Optimization On CEC 2015 Single-Objective Optimization Test Problems, IEEE,978-1-4799-7492-4/15/$31.00,2015.

[11] J. J . Liang, B. Y. Qu , P . N . Suganthen ,Q . Chen, Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-based Real-Parameter Single Objective Optimization, Technical Report, November 2014.

[12] Dogan Aydin,Thomas Stuzle,”A Configurable Generalized Artificial Bee Colony Algrithm with Local Search Strategies”,IEEE, Dot:978-1-4799-7492-4/15/$31.00, 2015.

[13] Cenaero, software ,Evolution Algorithm ,2010,From the World Wide Web website : http://www.cenaero.be/Page.asp?docid=27092&langue=EN.

[14] Zhang LB, WJ,Xie XF(2003) DEPSO: hybrid particle swarm with differential evolution operator . In:IEEE international conference on systems, man and cybernetics (SMCC), Washington DC,USA,pp 3816,3821.

[15] Mohammed El-Abd, “Hybrid Cooperative Co-evolution For The CEC15 Benchmarks”, IEEE, Dot:978-1-4799-7492-4/15/$31.00, 2015.

[16] Noor Awad, Mostafa Z. Ali, Robert G. Reynolds,”A Differential Evolution Algorithm with Success-based Parameter Adaptation for CEC2015 Learning-based Optimization”,IEEE, Dot:978-1-4799-7492-4/15/$31.00, 2015.

[17] Karm M. Sallam, Ruhul A. Sarker, Daryl L. Essam, Saber M.Elsayed,”Neurodynamic Differential Evolution Algoritm and Solving CEC2015 Competition Problems”,IEEE,Dot:978-1-4799-7492-4/15/$31.00, 2015.

[18] Friedman Ranking Test website: http://vassarstats.net/textbook/ch15a.html .

[19] Interpret all statistics and graphs for Friedman Test website: http://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modeling-statistics/anova/how-to/friedman-test/interpret-the-results/all-statistics-and-graphs/#p-value.

[20] Fmincon, Optimization Toolbox PDF. Available at http://bwrcs.eecs.berkeley.edu/Classes/icdesgn/ee141_s03/Project/Project1_solutions/fmincon.pdf.

[21] Benchmarks for Evaluation of Evolutionary Algorithms,CEC2015 Special Session Competition,
website: http://www.ntu.edu.sg/home/EPNSugan/index_files/

[22] Zahavat Sherinov, Ahmet Ünveren, Multi-objective imperialistic competitive algorithm with multiple non-dominated sets for the solution of global optimization problems,Springer,Soft Comput,DOI 10.1007/s00500-017-2773-6,August 2017.

APPENDIX

Introduction of the CEC'15 expensive optimization test problems

For all the CEC'15 expensive optimization test problems the variable bounds are (-100,100) [11].

1) High Conditioned Elliptic Function

$$f_1(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2 \quad \textbf{(1)}$$

2) **Cigar Function**

$$f_2(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2 \quad \textbf{(2)}$$

3) Discus Function

$$f_3(x) = 10^6 \, x_1^2 + \sum_{i=2}^{D} x_i^2 \quad \textbf{(3)}$$

4) **Rosenbrock's Function**

$$f_4(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \quad \textbf{(4)}$$

5) **Ackley's Function**

$$f_5(x) = -20 \exp(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i)) + 20 + e} \quad \textbf{(5)}$$

6) **Weierstrass Function**

$$f_6(x) = \sum_{i=1}^{D} (\sum_{k=0}^{k\,max} [a^k \cos(2\pi b^k (x_i + 0.5))]) - D \sum_{k=0}^{k\,max} [a^k \cos(2\pi b^k . 0.5)] \quad \textbf{(6)}$$

$$a = 0.5 \, , b = 3, kmax = 20$$

7) **Griewank's Function**

$$f_7(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad \textbf{(7)}$$

8) **Rastrigin's Function**

$$f_8(x) = \sum_{i=1}^{D} (x_i^2 - 10 \cos(2\pi x_i) + 10) \quad \textbf{(8)}$$

9) **Modified Schwefel's Function**

$$f_9(x) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i), \qquad z_i = x_i + 4.209687462275036e + 002$$

$$g(z_i) = \begin{cases} z_i \sin(|z_i|^{1/2}) & if\ |z_i| \le 500 \\ (500 - mod(z_i, 500)) \sin\left(\sqrt{|500 - mod(z_i, 500)|}\right) - \frac{(z_i - 500)^2}{10000D} & if\ z_i > 500 \\ (mod(|z_i|, 500) - 500) \sin\left(\sqrt{|mod(|z_i|, 500) - 500|}\right) - \frac{(z_i - 500)^2}{10000D} & if\ z_i < -500 \end{cases} \quad \textbf{(9)}$$

10) **Katsuura Function**

$$f_{10}(x) = \frac{10}{D^2} \prod_{i=1}^{D} (1 + i \sum_{j=1}^{32} \frac{\left|2^j x_i - round(2^j x_i)\right|}{2^j})^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} \quad \textbf{(10)}$$

11) **HappyCat Function**

$$f_{11}(x) = \left|\sum_{i=1}^{D} x_i^2 - D\right|^{1/4} + (0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i) \,/\, D + 0.5 \quad \textbf{(11)}$$

12) **HGBatFunction**

$$f_{12}(x) = \left|(\sum_{i=1}^{D} x_i^2)^2 - (\sum_{i=1}^{D} x_i)^2\right|^{1/2} + (0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i)/D + 0.5 \quad \textbf{(12)}$$

13) **Expanded Griewank's plus Rosenbrock's Function**

$$f_{13}(x) = f_7(f_4(x_1, x_2)) + f_7(f_4(x_2, x_3)) + \ldots\ldots + f_7(f_4(x_{D-1}, x_D)) + f_7(f_4(x_D, x_1)) \quad \textbf{(13)}$$

14) **Expanded Scaffer's F6 Function**

$$\text{Scaffer's F6 Function: } g(x,y) = 0.5 + \frac{(\sin^2(\sqrt{x^2+y^2})-0.5)}{(1+0.001(x^2+y^2))^2}$$

$$f_{14}(x) = g(x_1,x_2) + g(x_2,x_3) + \ldots + g(x_{D-1},x_D) + g(x_D,x_1) \quad \textbf{(14)}$$

1.5 Definitions of the CEC'15Learning-Based Benchmark Suite

A. Unimodal Functions:

1) **Rotated High Conditioned Elliptic Function**

$$F_1(x) = f_1\big(M_1(x-0)\big) + F_1^* \quad \textbf{(15)}$$

$$x \in (-100,100), \quad F^* = 100$$

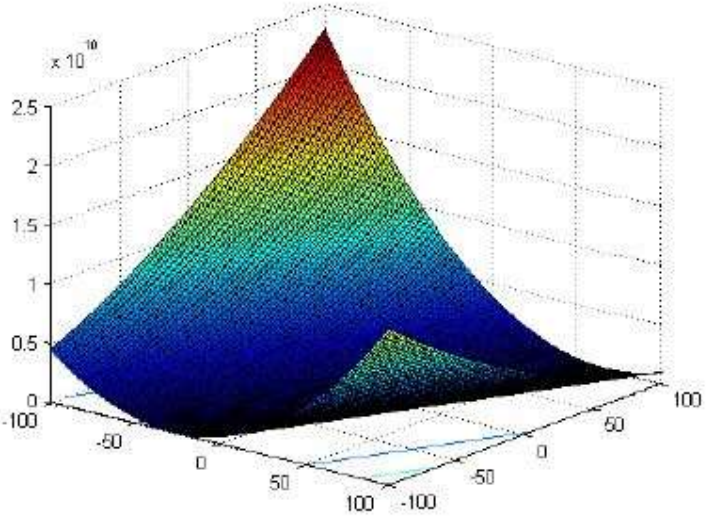

Figure 1.3-D map for 2-D function

Properties:

- Unimodal
- Non-separable
- Quadratic ill-conditioned

2) Rotated Cigar Function

$$F_2(x) = f_2\big(M_2(x - o_2)\big) + F_2^* \qquad \textbf{(16)}$$

$$x \in (-100,100), \qquad F^* = 200$$

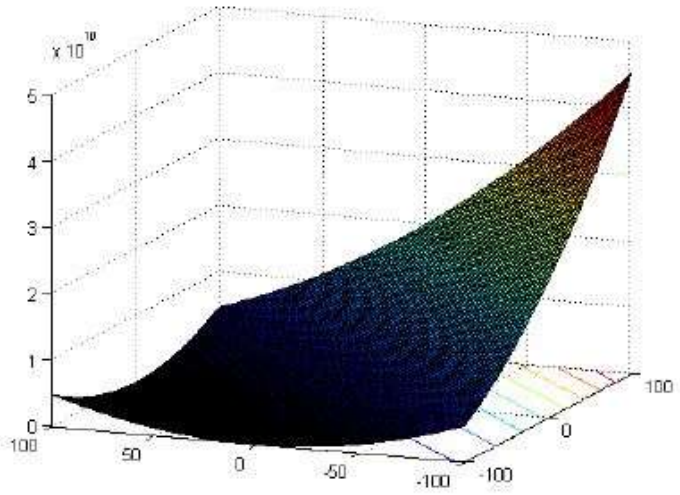

Figure 2. 3-D map for 2-D function

Properties:

- Unimodal
- Non-separable
- Smooth but narrow ridge

B. Multimodal Functions

3) Shifted and Rotated Ackley's Function

$$F_3(x) = f_3\big(M_3(x - o_3)\big) + F_3^* \qquad \textbf{(17)}$$

$$x \in (-100,100), \qquad F^* = 300$$

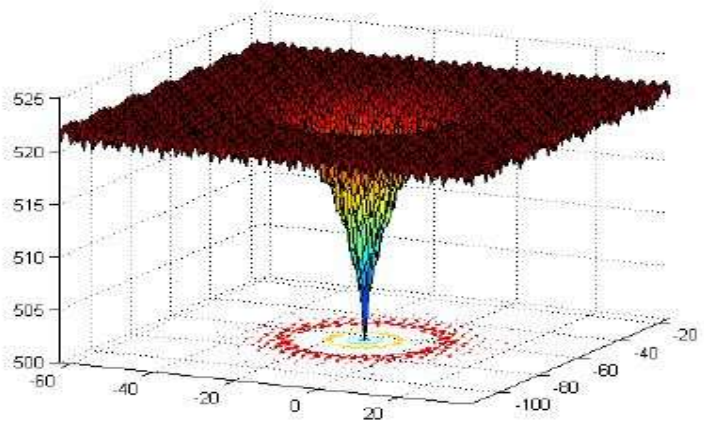

Figure 3. 3-D map for 2-D function

Properties:

- Multi-modal
- Non-separable

4) Shifted and Rotated Rastrigin's Function

$$F_4(x) = f_8\left(M_4\left(\frac{5.12(x - o_4)}{100}\right)\right) + F_4^* \quad \textbf{(18)}$$

$$x \in (-100,100), \qquad F^* = 400$$

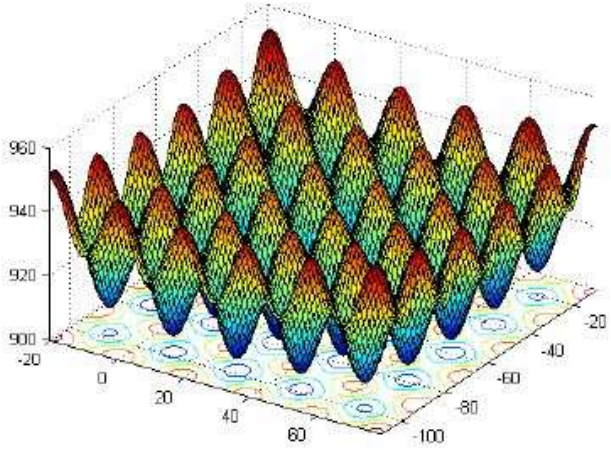

Figure 4 3-D map for 2-D function

Properties:

- Multi-modal
- Non-separable
- Local optima's number is huge

5) Shifted and Rotated Schwefel's Function

$$F_5(x) = f_9\left(M_5\left(\frac{5.12(x - o_5)}{100}\right)\right) + F_5^* \quad \textbf{(19)}$$

$$x \in (-100,100), \qquad F^* = 500$$

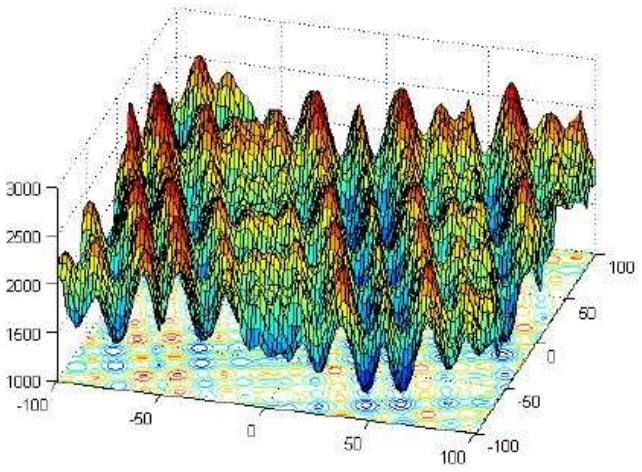

Figure 5(a). 3-*D* map for 2-*D* function

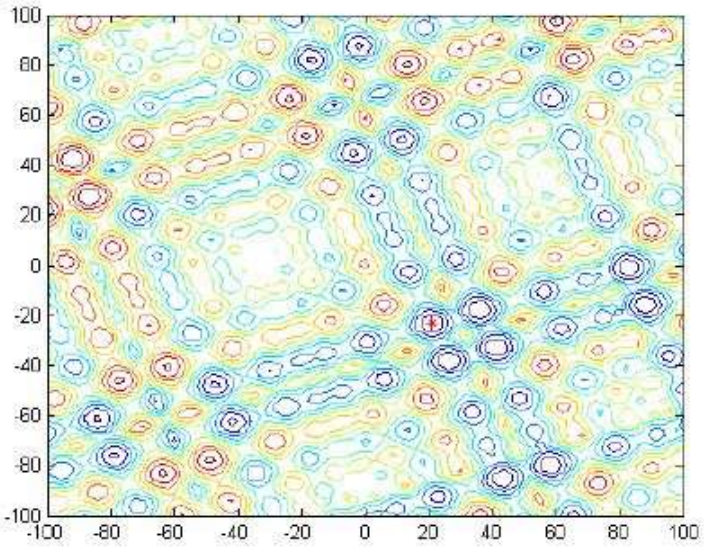

Figure 5(b).Contour map for 2-*D* function

Properties:

- Multi-modal
- Non-separable
- Local optima's number is huge and second better local optimum is far from the global optimum.

C. Hybrid Functions

Considering that in the real-world optimization problems, different subcomponents of the
variables may have different properties. In this set of hybrid functions, the variables are
randomly divided into some subcomponents and then different basic functions are used for
different subcomponents.

$$F(x) = g_1(M_1 z_1) + g_2(M_2 z_2) + \ldots + g_N(M_N z_N) + F^*(x) \qquad \textbf{(20)}$$

$x \in (-100,100)$
$F^*(x)$: best solution
$F(\boldsymbol{x})$: hybrid function
$g_i(\boldsymbol{x})$: i_{th} basic function used to construct the hybrid function

N: number of basic functions

$$\boldsymbol{z} = [\boldsymbol{z_1}, \boldsymbol{z_2}, \ldots\ldots, \boldsymbol{z_N}]$$

$$\boldsymbol{z_1} = [\boldsymbol{y_{S_1}}, \boldsymbol{y_{S_2}}, \ldots\ldots, \boldsymbol{y_{S_m}}], \boldsymbol{z_2} = \left[\boldsymbol{y_{S_{m+1}}}, \boldsymbol{y_{S_{m+2}}}, \ldots\ldots, \boldsymbol{y_{S_{m+n_2}}}\right], \ldots\ldots, \boldsymbol{z_N} = [\boldsymbol{y_{S_{\sum_{i=1}^{N-1} n_i+1}}}, \boldsymbol{y_{S_{\sum_{i=1}^{N-1} n_i+2}}}, \ldots\ldots, \boldsymbol{y_{S_D}}]$$

$y=x-o_i$, $S=randperm(1: D)$

p_i : used to control the percentage of $g_i(\boldsymbol{x})$

n_i: dimension for each basic function $\sum_{i=1}^{N} n_i = D$

$n_1=[p_1 D]$, $n_2=[p_2 D]$,….,$n_{N-1} =[p_{N-1} D], n_N = D - \sum_{i=1}^{N-1} n_i$

Properties:

- Multi-modal or Unimodal, depending on the basic function
- Non-separable subcomponents
- Different properties for different variables subcomponents

6) Hybrid Function 1

$F^* = 600$
$N= 3$
p = [0.3,0.3,0.4]
g_1: Modified Schwefel's Function f_9
g_2: Rastrigin's Function f_8
g_3: High Conditioned Elliptic Function f_1

7) Hybrid Function 3
$F^* = 700$
$N= 4$
p =[0.2,0.2,0.3,0.3]
g_1: Griewank's Function f_7
g_2: Weierstrass Function f_6
g_3: Rosenbrock's Function f_4
g_4: Scaffer's F6 Function f_{14}

8) Hybrid Function 5

$F^* = 800$

N= 5

p = [0.1,0.2,0.2,0.2,0.3]

g_1: Scaffer's F6 Function f_{14}

g_2: HGBat Function f_{12}

g_3: Rosenbrock's Function f_4

g_4: Modified Schwefel's Function f_9

g_5: High Conditioned Elliptic Function f_1

D. Composition Functions

$$F(x) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(x) + bias_i]\} + F^* \quad \textbf{(21}$$

$$x \in (-100,100)$$

F^*: best solution.

$F(\boldsymbol{x})$: composition function.

$g_i(\boldsymbol{x})$: i^{th} basic function used to construct the composition function.

N: number of basic functions.

o_i: new shifted optimum position for each $g_i(\boldsymbol{x})$, define the global and local optima's position.

$bias_i$: defines which optimum is global optimum.

σ_i: used to control each $g_i(\boldsymbol{x})$'s coverage range, a small σ_i give a narrow range for that $g_i(x)$.

λ_i: used to control each $g_i(\boldsymbol{x})$'s height

w_i : weight value for each $g_i(\boldsymbol{x})$, calculated as below:

$$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_{ij})^2}} \exp\left(-\frac{\sum_{j=1}^{D}(x_j - o_{ij})^2}{2D\sigma_i^2}\right) \quad \textbf{(22)}$$

Then normalize the weight $\omega_i = w_i / \sum_{i=1}^{n} w_i$

So when x=o_i, $\omega_j = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$ for j=1,2,…..,N, $f(x) = bias_i + f^*$

The local optimum which has the smallest bias value is the global optimum. The composition function merges the properties of the sub-functions better and maintains continuity around the global/local optima.

Functions $Fi' = Fi - Fi^*$ are used as g_i. In this way, the function values of global optima of g_i are equal to 0 for all composition functions in this report.

For some composition functions, the hybrid functions are also used as the basic functions.

With hybrid functions as the basic functions, the composition function can have different properties for different variables subcomponents.

9) Composition Function 1

$F^* = 900$

N=3

$\sigma = [20,20,20]$

$\lambda = [1,1,1]$

bias=[0,100,200]+F_9^*

g_1

- Schwefel's Function

g_2,g_3:

- Rotated Rastrigin's Function
- Rotated HGBat Function

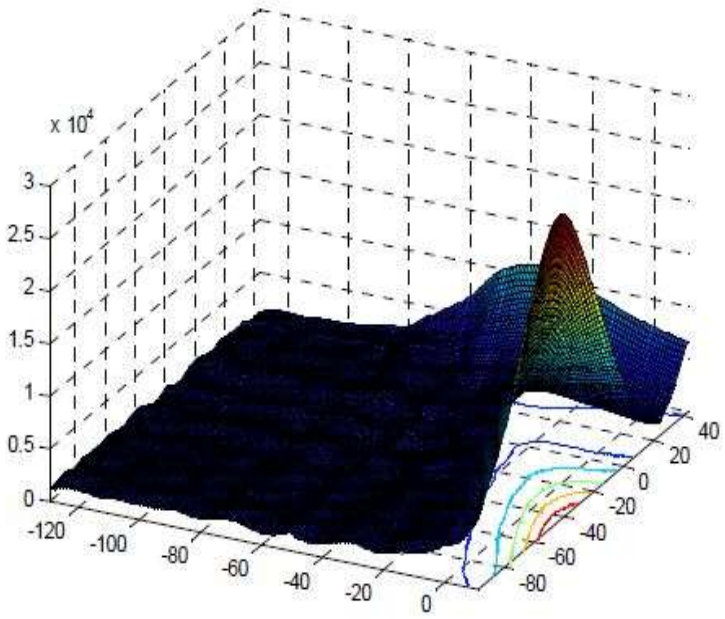

Figure6(a). 3-D map for 2-D function (example)

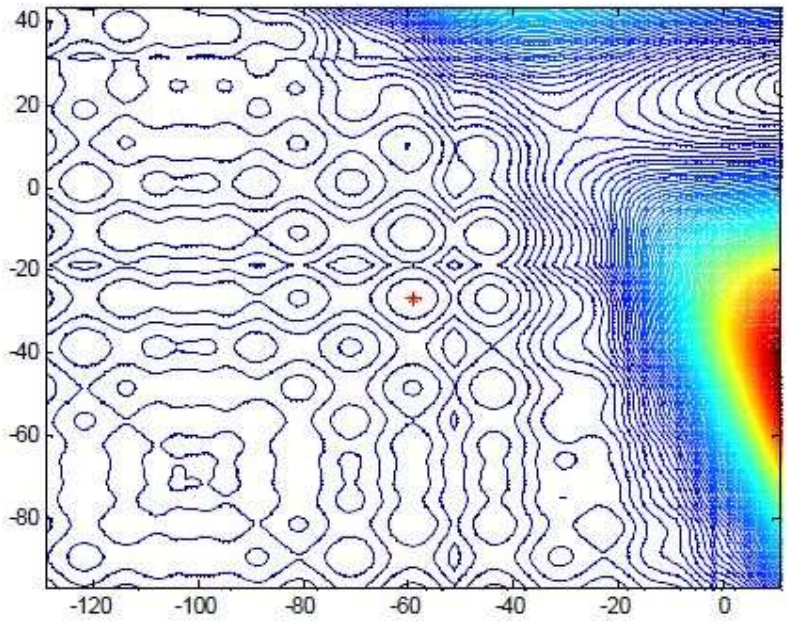

Figure6(b). Contour map for 2-D function (example)

Properties:

- Multi-modal
- Non-separable
- Different properties around different local optima
- The basic function of which the global optimum belongs to is fixed. The sequence of
  the other basic functions can be randomly generated.

10) Composition Function 2

$F^* = 1000$

$N = 3$

$\sigma = [10,30,50]$

$\lambda = [1.1,1]$

bias=[0,100,200]+F_{10}^*

g_1, g_2, g_3:

- Hybrid Function 1
- Hybrid Function 2
- Hybrid Function 3

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- Different properties for different variables subcomponents
- The sequence of the basic functions can be randomly generated.

11) Composition Function 3

$F^* = 1100$

$N = 5$

$\sigma = [10,10,20,20]$

$\lambda = [10,10,2.5,25,1e-6]$

bias=[0,100,200,300,400]+ F_{11}^*

g_1:

- Rotated HGBat Function

g_2,g_3,g_4,g_5:

- Rotated Rastrigin's Function
- Rotated Schwefel's Function
- Rotated Weierstrass Function
- Rotated High Conditioned Elliptic Function

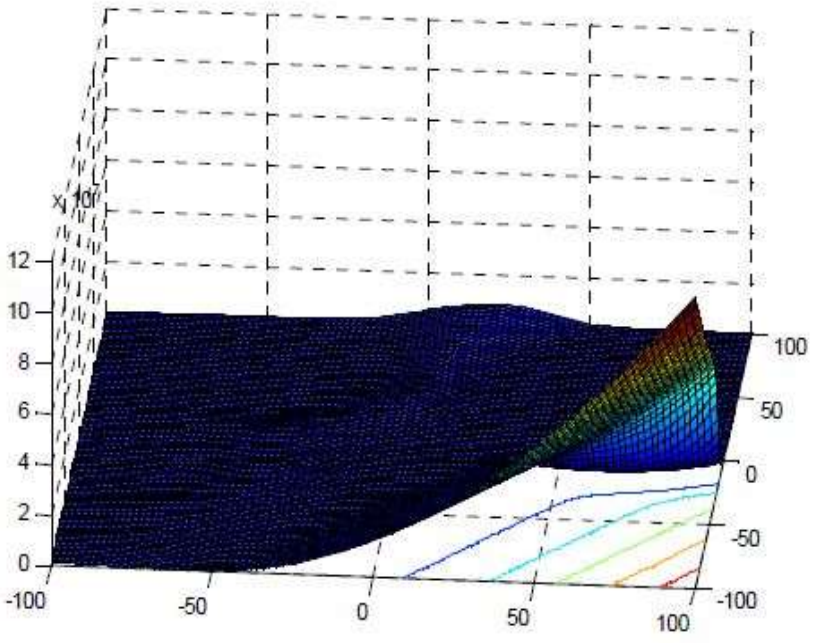

Figure 8(a). 3-D map for 2-D function(example)

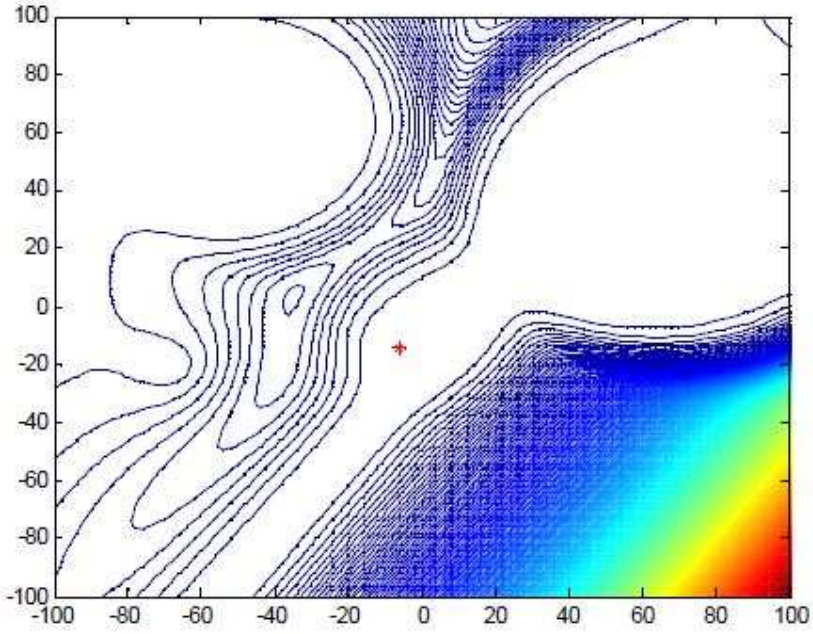

Figure8(b) Contour map for 2-D function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The basic function of which the global optimum belongs to is fixed. The sequence of
- the other basic functions can be randomly generated.

12) Composition Function 4

$F^* = 1200$

N=5

$\sigma = [10,20,20,30,30]$

$\lambda = [0.25,1,1e-7,10,10]$

bias=[0,100,100,200,200]+ F_{12}^*

g_1, g_2, g_3, g_4, g_5:

- Rotated Schwefel's Function
- Rotated Rastrigin's Function
- Rotated High Conditioned Elliptic Function
- Rotated Expanded Scaffer's F6 Function
- Rotated HappyCat Function

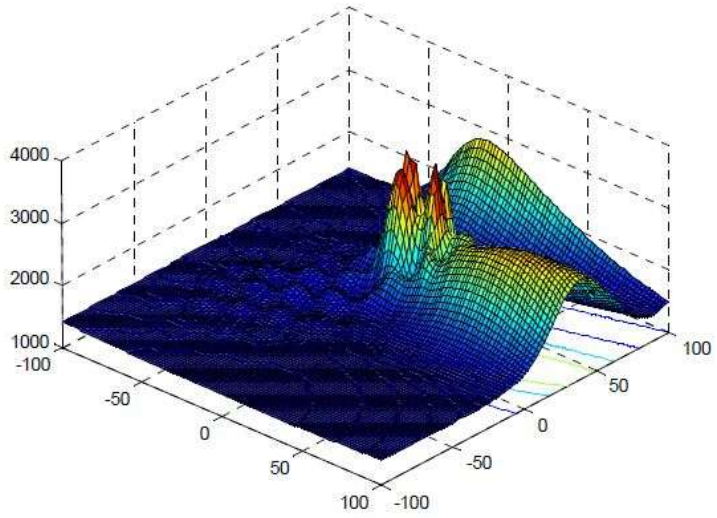

Figure9(a). 3-D map for 2-D function (example)

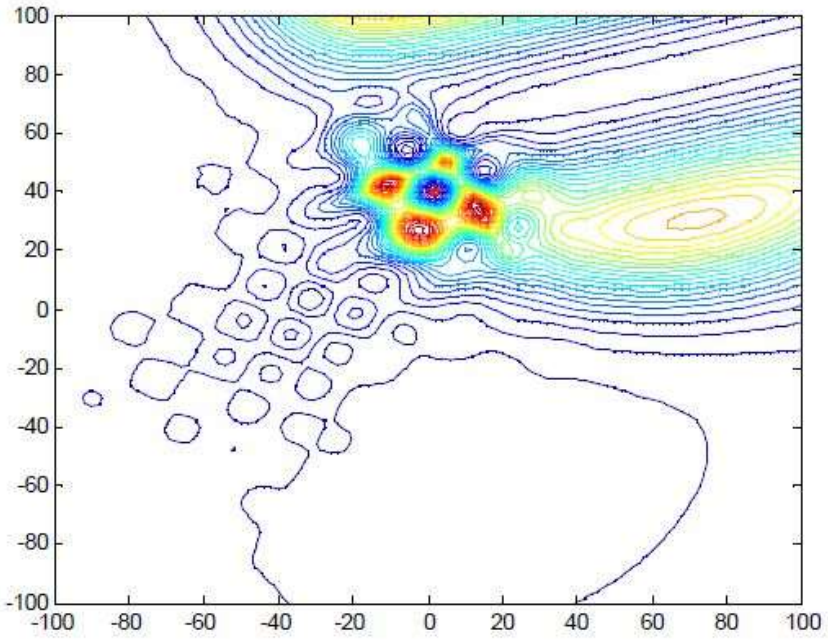

Figure9(b). Contour map for 2-D function(example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- Different properties for different variables subcomponents
- The sequence of the basic functions can be randomly generated

13) Composition Function 5

$F^* = 1300$

N =5

$\sigma = [10,10,10,20,20]$

$\lambda = [1,10,1,25,10]$

bias=[0,100,200,300,400]+F_{13}^*

g_1, g_2, g_3, g_4, g_5:

- Hybrid Function 3
- Rotated Rastrigin's Function
- Hybrid Function 1
- Rotated Schwefel's Function
- Rotated Expanded Scaffer's F6 Function

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The sequence of the basic functions can be randomly generated

14) Composition Function 6

$F^* = 1400$

N = 7

$\sigma = [10,20,30,40,50,50,50]$

$\lambda = [10,2.5,2.5,10,1e-6,1e-6,10]$

bias=[0,100,200,300,300,400,400]+ F_{14}^*

g_1:

- Rotated HappyCat Function

g_2, g_3, g_4, g_5:

- Rotated Expanded Griewank's plus Rosenbrock's Function
- Rotated Schwefel's Function
- Rotated Expanded Scaffer's F6 Function
- Rotated High Conditioned Elliptic Function
- Rotated Cigar Function
- Rotated Rastrigin's Function

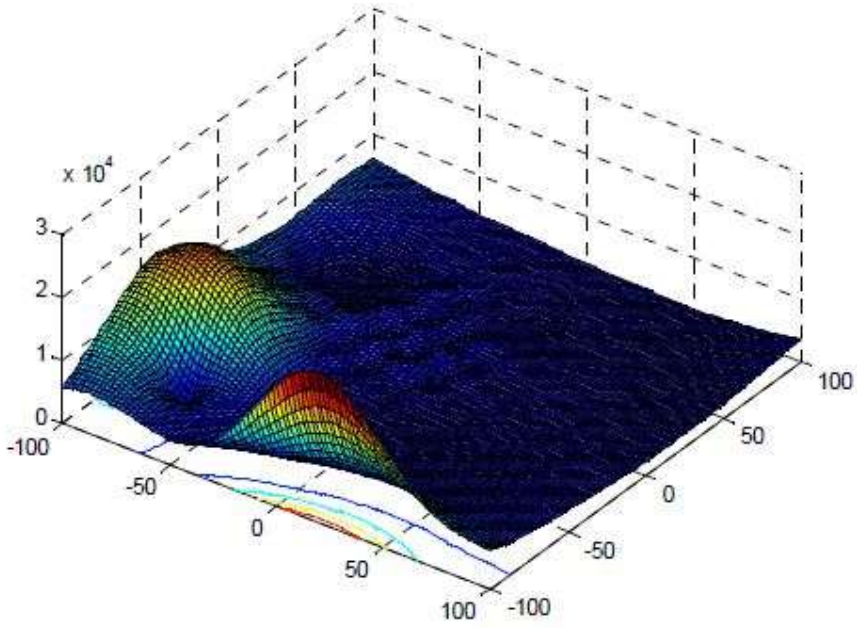

Figure 10(a). 3-D map for 2-D function (example)

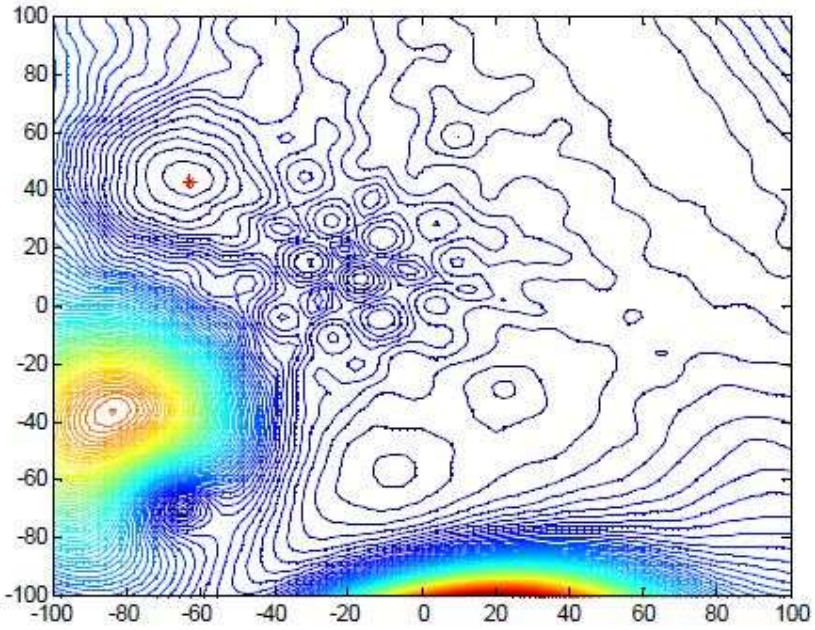

Figure 10(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The basic function of which the global optimum belongs to is fixed. The sequence of
- the other basic functions can be randomly generated.

15) Composition Function 7

$F^* = 1500$

$N = 10$

$\sigma = [10,10,20,20,30,30,40,40,50,50]$

$\lambda = [0.1,2.5e-1,0.1,2.5e-2,1e-3,0.1,1e-5,10,2.5e-2,1e-3]$

bias=[0,100,100,200,200,300,300,400,400,500]+ F_{15}^*

g_1, g_2, g_3, g_4, g_5:

- Rotated Rastrigin's Function
- Rotated Weierstrass Function
- Rotated HappyCat Function
- Rotated Schwefel's Function
- Rotated Rosenbrock's Function
- Rotated HGBat Function
- Rotated Katsuura Function
- Rotated Expanded Scaffer's F6 Function
- Rotated Expanded Griewank's plus Rosenbrock's Function
- Rotated Ackley Function

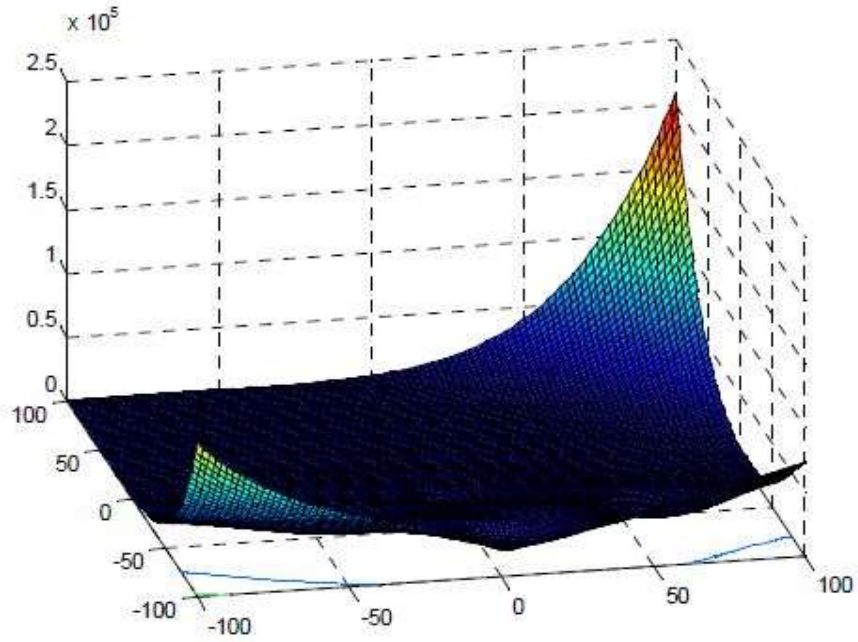

Figure 11(a). 3-*D* map for 2-*D* function (example)

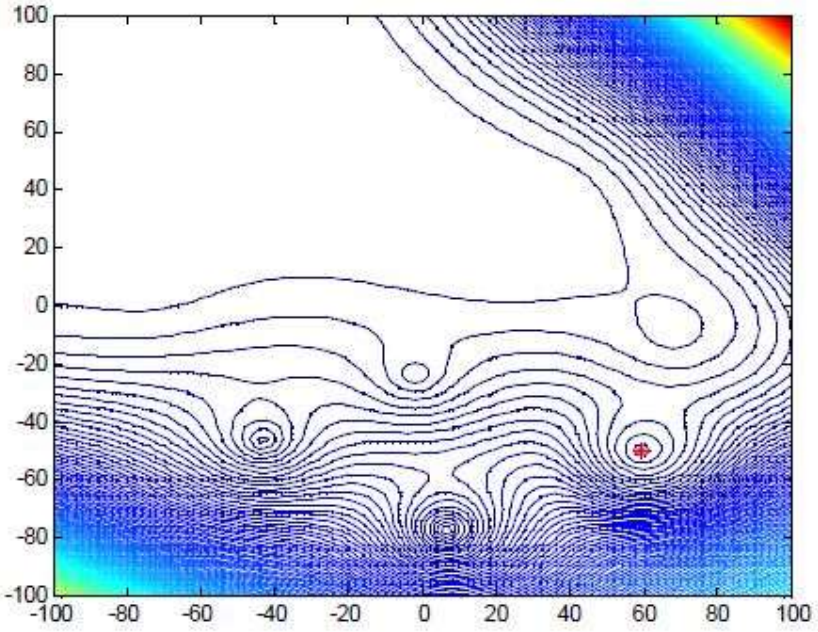

Figure 11(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The sequence of the basic functions can be randomly generated