Design and development of metaheuristic algorithms for Feature Selection

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Avneet Kaur
(801732008)

Under the supervision of:
Dr. Vijay Kumar
Assistant Professor
TIET, Patiala

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
PATIALA – 147004

June 2019

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "Design and Development of *Binary Metaheuristic Algorithms For Feature Selection*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Vijay Kumar* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Avneet Kaur

(Avneet Kaur)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Vijay Kumar)
Assistant Professor
Computer Science and
Engineering Department
TIET
Patiala

ACKNOWLEDGEMENT

It's an honour for me to work under the respected supervisor, **Dr. Vijay Kumar**, Assistant Professor, Department of Computer Science & Engineering, Thapar University, Patiala, who has provided me all the materials essentials for the preparation of this thesis report. He has been very helpful throughout the year and without his guidance this work might not have been possible. So I wanted to thank him for helping me in every possible manner.

I am also gratefull to **Dr. S.S. Bhatia**, Dean of Academic Affairs, **Dr. Maninder Singh**, Head of Computer Science & Engineering Department and **Dr. Ashutosh Mishra**, P.G. Coordinator, for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there at the need of hour and provided with all the help and facilities, which I required, for the completion of my thesis work.

Most importantly, I would like to thank my parents and the almighty for showing me the right direction out of the blue, to help me stay calm in the oddest of the times and keep moving even at times when there was no hope.

Avneet Kaur
(801732008)

ABSTRACT

There are many real life complex problems existing in this world which need to be taken into consideration. To solve such kind of problems various techniques have been proposed. There are many classical techniques designed to solve them. But there are some drawbacks associated with these techniques. Therefore in order to solve the complex problems efficiently, metaheuristic techniques have been designed and developed. Spotted hyena optimizer (SHO) and Emperor penguin optimizer (EPO) are the metaheuristic algorithms developed to solve the optimization problem.

The main motive of the thesis work is to design and develop the binary version of spotted hyena optimizer (BSHO) and Emperor penguin optimizer (BEPO). Spotted Hyena Optimizer (SHO) is a recently developed metaheuristic technique that mimics the hunting behavior of the spotted hyenas. The three main steps of this algorithm are prey searching, encircling, and attacking. In the BSHO algorithm, tangent hyperbolic function is utilized to squash the continuous position and then these values are used to update the position of spotted hyenas. Emperor Penguin Optimizer is a novel meta-heuristic algorithm which is inspired from the huddling behavior of the emperor penguins. In the binary version of this algorithm, we have used the sigmoidal transfer function to update the positions of the emperor penguins. Both these algorithms have been tested on 29 benchmark test functions and compared with the other binary metaheuristic algorithms. They have been applied on the feature selection problem also.

TABLE OF CONTENTS

LIST OF FIGURES

datasets

LIST OF TABLES

Chapter 1

Introduction

1.1 Preamble

Optimization is the process in which the function is maximized or minimized in order to attain the best solution among the various solutions. There are many real life complex problems existing in this world which need to be taken into consideration. To solve such kind of problems various techniques have been proposed. There are many classical techniques designed to solve them. But there are some drawbacks associated with these techniques as they take the large amount of time to find out the best solutions. Approximate methods used for solving such problems. As these techniques are guaranteed for solving these problems. However the drawback of using these methods is that they may take a large computational time to find the solution. Therefore for such problems metaheuristic techniques have been designed and developed.

Metaheuristic techniques are widely used from past few years. These algorithms remove the complexity of solving the problems. They are capable enough of solving the problems which may not be solved by the approximate methods in considerable amount of time. They are able to generate the solution of complete or incomplete data also. They are not dependent upon the type of the problem. These techniques may not always provide the exact solution to a problem but provides us with the nearest optimal solutions in lesser computational time as compared to the classical techniques which may take infinite time to find out the exact solution. [1]

Some of well-known metaheuristic techniques are Genetic algorithm GA [2], Differential evolution (DE) [3], Ant colony optimization (ACO) [4], Gravitational search algorithm (GSA) [5], Particle swarm optimization (PSO) [6], etc. These algorithms are designed and developed for solving problems which may take large time. Therefore, new metaheuristic algorithms are designed for solving a different optimization problems.

1.2 Definition

Metaheuristics is defined as the problem independent way of providing the solution to wide range of problems. It provides better exploration and exploitation of the search space than the others. It performs well with the complete as well as incomplete information of the problem. [7]

"A metaheuristics formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space" [8]

Important properties of metaheuristics are given below: [9]

- They are problem independent
- Their main aim is exploring search space efficiently
- They avoid local optima
- These methods are flexible
- These are approximate methods
- These are very simple methods

1.3 Classification

There are large numbers of metaheuristic techniques available to solve the complex problems. Metaheuristic algorithms are categorized into different categories based on the parameters. Depending upon the dimensions, the metaheuristic techniques have been classified into four different categories as mentioned below:

Following represents the metaheuristic techniques based upon its classification mentioned above:

1.3.1 Search Based Metaheuristic Algorithm

Based upon the searching techniques, metaheuristic techniques have been categorized into different categories, namely, local search and global search algorithms.

1.3.1.1 Local search algorithm

The Local search algorithms are used for solving hard optimization problems. This algorithm begins with the candidate solution and proceeds towards the adjacent solution after every iteration. The relation of the neighbor must be defined in the search space. Examples of local search algorithms are hill climbing, simulated annealing [10] etc. In these the searching initiates from the candidate solution and proceeds towards adjacent solution after every iteration. Then the solution is compared with the previous one. If the newly found solution is better than the previous one, the solution is updated otherwise no modification will take place. The pros are less memory space. The major drawback is getting stuck in the local optima.

1.3.1.2 Global search algorithm

Global search algorithm is used in finding the global optimal solution. These techniques are population based. This algorithm avoids the solution being stuck in local optima by considering best solution among various optimal solutions. This algorithm is more effective than local search algorithm as it overcomes the problem faced by local search algorithm. Examples of global search algorithms are grey wolf optimization [11], gravitational search algorithm, spotted hyena optimizer [12] etc.

1.3.2 Population based

Based upon the number of solution, the metaheuristic techniques are categorized in different categories. These are single solution and multi solution algorithm.

1.3.2.1 Single solution algorithm

In this category, only one candidate solution is considered in the beginning. Then after every iteration this algorithm proceeds towards the adjacent solution in for finding the best solution. These kinds of algorithms have the chances of getting stuck in local optima. Examples of the single solution algorithms are Guided Local Search, simulated annealing etc.

1.3.2.2 Multi solution algorithm

These are population based algorithms in which there are large number of candidate solutions present in the beginning. These multi solutions are helpful in finding the optimal solutions effectively as compared to the single solution. They have capability

to avoid the local optima. Some of the examples of population based algorithms are grey wolf optimizer, particle swarm optimization, spotted hyena optimizer, gravitational search algorithm, ant colony optimization, binary grey wolf optimization [13], etc. The exploration and exploitation capability of these metaheuristics are superior to single solution metaheuristic algorithm.

1.3.3 Nature inspired algorithm

These algorithms have been inspired by the nature. These algorithms have been categorized into four categories:

1.3.3.1 Swarm based algorithm

Swarm-based metaheuristic are influenced from their collective behavior of swarm. The swarms can be group of animals or birds whose behavior is mimicked. Some of the algorithms included in this category are Grey Wolf Optimizer, Bat Algorithm [14], Ant Colony Optimization, Spotted Hyena Optimizer, Bee collecting pollen algorithm [15], Particle Swarm Optimization, Wolf Pack Search Algorithm [16].

1.3.3.2 Evolutionary based algorithm

The evolutionary based algorithms are motivated from the biological evolution. Examples of the evolutionary algorithms are Genetic Programming [17], Evolution Strategy[18], Biogeography-Based Optimizer (BBO) [19] etc.

1.3.3.3 Physics based algorithm

Physics based algorithm uses the concept of physics in the algorithms such as laws of motion and gravity. Gravitational search algorithm, Simulated Annealing, Big Bang Big Crunch [20], Small world optimization problem [21], Black Hole algorithm [22], Central Force Optimization [23], Artificial chemical reaction optimization algorithm [24], Galaxy-based algorithm [25] are some of the physics based algorithms.

1.3.3.4 Chemical based algorithm

The Chemical based algorithm is based upon chemical reaction. In this, the molecules retaliate with each other and later on goes into low energy state. Chemical reaction optimization is one such algorithm.

1.4 Spotted Hyena Optimizer: Spotted hyena optimizer is the swarm based algorithm. Spotted Hyenas live in a group of more than 100 members. They usually hunt in groups. The hyenas are categorized into four types namely spotted, striped, brown, and aardwolf. Spotted hyenas are the massive hyena species and proficient hunters. They produce a sound like human laugh in order to communicate with each other. They have spots on their body. They form the cohesive clusters for coordination between hyenas.

The three fundamental steps of SHO are encircling, hunting, and attacking prey. The prey is considered as the best solution that is nearest to optimal solution. Rests of spotted hyenas update their positions according to the best solution. Encircling behavior of spotted hyenas is formulated using following equations:

$$\vec{D}_h = |\vec{Y}.\vec{Q}_q\,(s) - \vec{Q}(s)| \tag{1.1}$$

$$\vec{Q}(s+1) = \vec{Q}_q(s) - \vec{Z}.\vec{D}_h \tag{1.2}$$

where $\vec{D}_h$ is considered as the distance between hyena and the target prey. $\vec{Y}$ and $\vec{Z}$ are coefficient vectors. s indicates the current iteration. $\vec{Q}_q$ is the position vector of the prey and $\vec{Q}$ is position vector of the hyena. $\vec{Y}$ and $\vec{Z}$ are calculated below:

$$\vec{Y} = 2.\overrightarrow{rand}_1 \tag{1.3}$$

$$\vec{Z} = 2\vec{g}.\overrightarrow{rand}_2 - \vec{g} \tag{1.4}$$

$$\vec{g} = 5 - (Iteration * (5/Max_{Iteration})) \tag{1.5}$$

where, Iteration=1,2,3,…,$Max_{Iteration}$

Here, $\vec{g}$ is decreased from 5 to 0 which maintains the stability between exploitation and exploration. $\vec{rand}_1$ and $\vec{rand}_2$ are the random vectors in [0,1]. The values of $\vec{Y}$ and $\vec{Z}$ are adjusted so that spotted hyenas can reach different region about the current position.

The mathematical formulation of hunting mechanism is given below:

$$\vec{D}_h = |\vec{Y}.\vec{Q}_h - \vec{Q}_k| \tag{1.6}$$

$$\vec{Q}_k = \vec{Q}_h - \vec{Z}.\vec{D}_h \tag{1.7}$$

$$\vec{C}_h = \vec{Q}_k + \vec{Q}_{k+1} + \cdots + \vec{Q}_{k+N} \tag{1.8}$$

where Q_h indicates the position of first best spotted hyena and Q_k is the position of rest of the spotted hyenas. N is the number of spotted hyenas which can be calculated in the following manner:

$$N = count_{NS}(\vec{Q}_h, \vec{Q}_{h+1}, \vec{Q}_{h+2}, \dots, (\vec{Q}_h + \vec{M})) \quad (1.9)$$

where $\vec{M}$ is the random vector in range [0.5, 1]. NS indicates the count of candidate solutions. $\vec{C}_h$ is the cluster of N best solutions. The attacking mechanism of spotted hyenas is mathematical formulation below:

$$\vec{Q}(s+1) = \frac{\vec{C}_h}{N} \quad (1.10)$$

where $\vec{Q}(s+1)$ stores the best solution. Spotted hyena optimizer (SHO) is depicted in Algorithm 1.

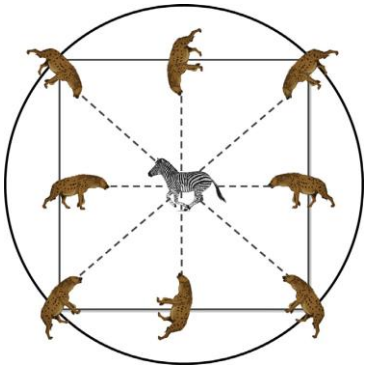


Figure 1.1: Spotted hyenas attacking the prey

Spotted Hyena Optimizer Algorithm

1. Initialize spotted hyenas population $Q_i(i = 1,2 \dots . . n)$
2. Initialize control parameters namely $Y, Z, h, and\ N$
3. Compute the fitness value of each spotted hyena
4. Q_h is set to the best spotted hyena
5. C_h is set to cluster of optimal solutions
6. while (s < Max number of Iterations) do
7. for each spotted hyena do
8. Update the position of the current hyena according to Eq. (10)
9. end for
10. Update control parameters $Y, Z, h, and\ N$
11. Adjust the boundary of hyenas if they go beyond the search space.
12. Compute the fitness value of each spotted hyena
13. Update Q_h if it is better than the previous solution
14. Update the cluster C_h w.r.t Q_h
15. $s = s + 1$
16. end while
17. return Q_h

1.5 Emperor Penguin optimizer: Aptenodytes forsteri is the scientific name of Emperor penguin. As compared to all different types of penguins, they are the largest ones. With respect to the size, all the males and females are very much alike. They have black head, white stomach, wings, and tail. They live in Antarctica. In order to survive in the winters of Antarctica, they huddle together in order to keep themselves warm and protect themselves from cold wind. This behavior of emperor penguins is explained below: [26]

- Create the huddle boundary
- Evaluate the temperature profile of the penguins
- Calculate the distance among the emperor penguins
- Relocate the mover

The huddling behavior of the emperor penguins is mathematically in the following manner. The main motive is finding the efficient mover and accordingly update position of the emperor penguins.

1.5.1 Create huddle boundary

The mathematical formulation of huddle boundary is given below:

$$\varphi = \nabla \emptyset \tag{1.11}$$

Where $\emptyset$ is wind and φ is gradient of $\emptyset$.

$$F = \emptyset + i\mu \tag{1.12}$$

Where F is analytical function and i is imaginary constant. Vector μ is merged with $\emptyset$.

1.5.2 Evaluate the temperature profile of the penguins

The penguins huddle for increasing the temperature around the huddle to survive Antarctica winter. It is assumed that when radius R>1, temperature t=0 and when radius $R < 1$, temperature t=1.

$$T = \left(t - \frac{Max_{iteration}}{x - Max_{iteration}}\right) \tag{1.13}$$

$$t = \begin{cases} 0, & R > 1 \\ 1, & R < 1 \end{cases} \tag{1.14}$$

Where t is temperature profile, $Max_{iteration}$ is the maximum iterations, T indicates the time for finding best optimal solution.

1.5.3 Calculate the distance between the emperor penguins

$$\vec{D} = Abs(\ B(\vec{Y}).\overrightarrow{Q(s)} - \vec{Z}.\overrightarrow{Q_{ep}(s)} \quad (1.15)$$

Where $\vec{D}$ is the distance among emperor penguins and the best solution, $\vec{Y}$ and $\vec{Z}$ are used to avoid collisions between emperor penguins. s is the current iteration. $\vec{Q}$ indicates the best optimal solution and $\overrightarrow{Q_{ep}}$ indicates the position of the emperor penguins. $B()$ designates the social forces with which the emperor penguins move towards the best optimal solution. $\vec{Y}$ and $\vec{Z}$ are computed as follows:

$$\vec{Y} = \left(N \times \left(T + Q_{grid}(Accuracy)\right) \times Rand()\right) - T \quad (1.16)$$

$$Q_{grid}(Accuracy) = Abs(\vec{Q} - \overrightarrow{Q_{ep}}) \quad (1.17)$$

$$\vec{Z} = Rand() \quad (1.18)$$

where N is the movement parameter maintaining gap among emperor penguins and its value is 2. $Q_{grid}(Accuracy)$ is the polygon grid accuracy, $Rand()$ indicates the random function whose value lies between [0,1].

$B()$ is calculated below:

$$\overrightarrow{B(Y)} = \sqrt{(g.e^{-\frac{s}{l}} - e^{-s})^2} \quad (1.19)$$

Where e is the expression function, g and l are control parameters whose value lies in the range [2,3] and [1.5,2] respectively.

1.5.4 Relocate the mover According to the mover that is best optimal solution the position of the emperor penguin is updated. The equation given below is used to update the position.

$$\overrightarrow{Q_{ep}}(s+1) = \vec{Q}(s) - \vec{Y}.\vec{D} \quad (1.20)$$

The Emperor Penguin Optimizer algorithm is depicted below:

Algorithm 1. Emperor Penguin Optimizer Algorithm

Input: Initialize the emperor penguin population $\overrightarrow{Q_{ep}}(i = 1, 2 \ldots . . n)$
Output: best optimal solution $\vec{Q}$
Procedure EPO
Initialize the parameters $T, \vec{Y}, \vec{Z}, B(), R \text{ and } Max_{iteration}$
Calculate the fitness function of each search agent
while(s < $Max_{iteration}$) do
FITNESS(Q_{ep})
$R \leftarrow Rand()$
If($R > 1$) then
$t \leftarrow 0$
Else
$t \leftarrow 1$
End if

$T \leftarrow (t - \frac{Max_{iteration}}{x - Max_{iteration}})$

For $i \leftarrow 1$ to n do
For $j \leftarrow 1$ to n do
Compute the vectors $\vec{Y} \text{ and } \vec{Z}$ using equations (6) and (8)
Compute the functions $B(Y)$ using equation (9)
Update positions of current agent using eq. (10)
End for
End for
Update parameters $T, \vec{Y}, \vec{Z} \text{ and } B()$
Reorganize the search agents that goes beyond the boundary
FITNESS(Q_{ep})
Update $\vec{Q}$ if current solution is better than previous one
$s \leftarrow s + 1$
End while
Return $\vec{Q}$
End Procedure

Procedure FITNESS(Q_{ep})
For $i \leftarrow 1$ to n do
FIT[i]← FITNESS_FUNCTION(Q_{ep})
End for
$FIT_{best} \leftarrow BEST(FIT[])$
End FIT_{best}
End procedure

```
Procedure BEST(FIT[])
  best ← FIT[0]
     For i ← 1 to n do
          if  FIT[i] < best
              best ← FIT[i]
          End if
     End for
   Return best
End procedure
```

1.6 Applications

Metaheuristics has been used for solving problems such as combinatorial problems. Using these algorithms, the optimal solution for discrete search space can be identified. One such example is the travelling salesman problem in which increasing the size of the problem increases the search space of the solution. To deal with such kind of problems metaheuristics have been used. Metaheuristic techniques are applied in different field of engineering, science and technology.

1.5 Organization of Thesis

The Thesis work has been constituted into six chapters. The descriptions of these six chapters are mentioned below:

Chapter 1 illustrates the introduction of optimization problems and metaheuristics. This chapter contains definition and its various properties. The various categories of metaheuristics have been explained in this chapter. The detailed description of spotted hyena optimizer (SHO) and emperor penguin optimizer (EPO) has been given.

Chapter 2 constitutes literature survey in which various metaheuristic algorithms have been discussed. The continuous as well as the binary versions of the metaheuristic techniques have been discussed in this chapter.

Chapter 3, the motivation behind the proposed algorithm has been discussed. Then the problem statement of the algorithm has been mentioned. The main objectives of the thesis are also mentioned.

The proposed algorithm binary spotted hyena optimizer (BSHO) is discussed in chapter 4. This algorithm is compared with various binary metaheuristic algorithms. The experiments on various benchmark functions have been described in this chapter.

The proposed algorithm binary emperor penguin optimizer (BEPO) is discussed in chapter 5. It is compared with various binary metaheuristic algorithms. Various benchmark functions have been used to evaluate the performance of proposed BEPO. Beside this, mathematical formulation of BEPO is also mentioned.

Chapter 6 constitutes the conclusions and future scopes.

Chapter 2

Literature Survey

This chapter represents the study of different metaheuristic algorithms namely continuous as well as the binary version of the metaheuristic algorithms.

2.1 Continuous Metaheuristic Algorithm

Spotted Hyena Optimizer is developed by Gaurav Dhiman and Vijay Kumar in 2017. Spotted Hyenas live in a group of more than 100 members. They usually hunt in groups. Hyenas are categorized into different categories such as spotted, striped, brown, and aardwolf. Among them the spotted hyenas are the largest ones. They have fur on their body. Spotted hyenas are the massive hyena species and proficient hunters. They produce a sound like human laugh in order to communicate with each other. They have spots on their body. They form the cohesive clusters for coordination between hyenas. SHO algorithm works by searching, encircling, hunting, and attacking the prey. The spotted hyenas will search for their prey. After that they will encircle them in order to trap the prey. And in the end, they will attack the prey. The best solution is the prey that is nearest to optimal solution. Rests of spotted hyenas update their positions accordingly. This will be done with each course of iterations. This algorithm is verified on 29 well-known benchmark functions and further applied to engineering problems.

Emperor penguin optimizer is developed by Gaurav Dhiman and Vijay Kumar. As compared to all different types of penguins, they are the largest ones. With respect to the size, all the males and females are very much alike. They have black head, white stomach, wings and tail. They live in Antarctica. In order to survive in the winters of Antarctica they huddle together in order to keep themselves warm and protect themselves from cold wind. This huddling behavior of emperor penguins is emphasized below:

- Create the huddle boundary

- Evaluate the temperature profile of the penguins
- Calculate the distance among the emperor penguins
- Relocate the mover

The main motive is detecting the effective mover and accordingly updates the position of the emperor penguins. This algorithm has been verified on 44 benchmark functions and has been compared with the eight other algorithms which show that this algorithm outperforms than the other metaheuristic algorithms. This algorithm has also applied on real-life engineering problems.

Grey Wolf Optimizer is designed and developed by Seyedali Mirjalili and Andrew Lewis. This algorithm is based upon the hunting mechanism of the grey wolves. In this algorithm, the wolves search their prey, then encircle them and at the end attack their prey. This whole process is performed in a group by the wolves. The wolves are categorized in various categories i.e. alpha, beta, delta and omega wolves. The alphas act as the leaders of the wolves. They can be male or a female. Their main duty is to make decisions of the hunting process, sleeping time etc. They are the dominating wolves. The next category is of the beta wolves. In case there is no alpha wolf, the beta wolves can hold their place. Third category is of omega wolves. They are considered as the scapegoats. They are treated as babysitters in the pack and maintain peace in the pace so that there is no fighting between the wolves. Last category is of delta wolf. Alpha and beta wolf dominate the delta wolves but delta in turn dominates the omega wolves.
The alphas (α) are considered as the best solution and the beta (β) is considered as second best solution and delta (δ) is considered as the third best solution. The candidates other than this are assumed as omega (ω). The omega wolves follow these three best solutions. The result shows that this algorithm outperforms than the other metaheuristic algorithms.

Xin-She Yang in 2013 proposed a metaheuristic algorithm named as Bat Algorithm. This algorithm uses the concept of echolocation quality of bats. In this algorithm, the bats use the echolocation to find out their prey and distinguish them from obstacles. They emit the sound pulse which when reflected back is used to estimate the

distance of the prey and also distinguish that whether it is a prey or any other obstacle. When bats find out their target, the loudness will get decreased and the pulse emission will get increased.

Esmat Rashedi and Saeid Saryazdi developed a metaheuristic algorithm in 2009 named as Gravitational Search Algorithm. It is a physics based metaheuristic algorithm. This algorithm uses physics concept i.e. law of gravity. The searching agents are considered as the mass which interconnect with one another using Newton's gravity law. By increasing the distance between the masses the force of attraction decreases and by decreasing the distance between the mass the force of attraction increases. Each mass has four specifications associated with it.

Ant colony optimization algorithm (ACO) is one of the metaheuristic algorithm. Initially proposed by Marco Dorigo in 1992[11]. Each ant is placed different or the same corners at the beginning of the problem. They find the shortest path to reach their food. The ants secrete the chemical called pheromone. Whenever an ant goes for the search of their food they leave behind the pheromone trails which help the other ants to follow the same path. When the path is too long, the liquid will vaporize and ants will not able to reach their destination. This will help them to find the shortest path.

Holland in 1975 proposed a novel metaheuristic algorithm names as Genetic Algorithm. The two main things required in this algorithm are the fitness function and the genetic representation of the solutions. Initializing randomly the population is the first step. Then next step is the process of selection where fittest candidate solutions got selected from the population. The function is problem dependent. After selection process, the next step is crossover and mutation to produce new population. In mutation only one parent is needed but in crossover two parents are required. The algorithm will continue till desired solution is found.

James Kennedy and Russell Eberhart developed a metaheuristic algorithm in 1995 named as Particle swarm optimization. In this, the group of birds are

searching their food. All the birds are following one bird which is considered as the current optimal solution. There are two values gbest known as group best and pbest known as personal best. The optimal solution of an individual bird will be considered pbest value and the best optimal solution among all the birds is considered as the gbest value.

Seyedali Mirjalili proposed a new metaheuristic algorithm in 2015 named as Moth Flame Algorithm [28]. This algorithm depends upon navigation method of the moths. Moths usually fly at the night by maintain the angle according to the moon. By maintain the angle according to moon they traverse in the straight line for longer distances. If they will not maintain a fixed angle then they will get stuck in the artificial lights and will fly in a spiral way around the artificial light.

Mucherino and Seref in 2008 developed a metaheuristic algorithm named as Monkey algorithm. This algorithm depends upon the climbing process of monkey. In this, the optimal solution is found step by step. First the monkey will climb on the mountain and then have a look around. If the monkey finds a mountain higher than the current mountain, then he will jump to the higher mountain. In this way the monkey will update its position. Depending upon this, each monkey will update their position according to the pivot.

2.2 Binary Metaheuristic Algorithms

E. Emary, Hossam M. Zawbaa and Aboul Ella Hassanien in 2016 proposed this metaheuristic algorithm named as Binary Grey Wolf Optimization [29]. It depends upon the hunting process of the wolves. In this algorithm, all the wolves hunt for their prey in groups in continuous search space. The wolves are categorized in various ways i.e. alpha, beta, delta and omega wolves. The alpha acts as leaders. Their main duty is to make decisions of the hunting process, sleeping time etc. They are the dominating wolves. The next category is of the beta wolves. In case there is no alpha wolf, the beta wolves can hold their place. The third category is of omega wolves. They are considered

as the scapegoats. They are treated as babysitters in the pack and maintain peace in the pace so that there is no fighting between the wolves. Last category is of delta wolf. Alpha and beta dominate the delta wolves but delta in turn dominates the omega wolves. The alphas (α) are the best solution and the beta (β) is considered as the second best solution and delta (δ) as third best solution. The candidates other than this are assumed as omega (ω). The omega wolves follow these three best solutions. The binary algorithm came into existence where the wolves search for their prey in discrete search space, that is, they search for their prey at the corners of the hypercube. They change their position in the binary form i.e. in the (0,1) form. In this algorithm, sigmoid function is used to update the positions of the wolves.

Seyedali Mirjalili, Seyed Mohammad Mirjalili and Xin-She Yang in 2014 developed a metaheuristic algorithm named as Binary Bat Algorithm [30]. It was developed in 2010. In this algorithm, the bat emits the sound pulse which emits back after striking through an obstacle or prey which helps them to calculate the distance between them and the obstacle or prey. It also helps them to identify whether it's a prey or obstacle. The binary bat is the extended version of this algorithm in which the bats update their position using sigmoid function.

Binary Gravitational Search Algorithm [31] is invented by Esmat Rashedi, Hossein Nezamabadi-pour and Saeid Saryazdi in 2010. The gravitation search algorithm is the physics based metaheuristic algorithm. It uses the law of force and gravity. In this algorithm, the masses act as the search agents. By increasing the distance between the masses the force of attraction decreases and by decreasing the distance between the mass the force of attraction increases. Each mass has four specifications associated with it. The binary algorithm uses discrete binary space that is in the form of 0 and 1. This algorithm uses the tangent function as the transfer function to update the positions.

Mojtaba Ahmadieh Khanesar, Mohammad Teshnehlab and Mahdi Aliyari Shoorehdeli proposed the Binary Particle Swarm Optimization algorithm in 2007 [32]. It is a swarm based algorithm. In this algorithm, the flock of birds are searching their food. All the birds are following one bird which is considered as the best solution. In Binary algorithm, the personal best, group best values are calculated in the same way as that of the continuous PSO. The only difference is the calculation of the velocity which is calculated using the sigmoidal transfer function.

Seyedali Mirjalili proposed a binary metaheuristic algorithm named as Binary Dragon fly algorithm in 2016 for feature selection problem [33]. This algorithm is inspired from thee swarming behavior of the dragonflies. They hunt their prey and migrate. The objective of the dragonflies is the survival, therefore they move towards the food and away from the enemies. The solution to the problem is found in the binary values.

Nadia Abd-Alsabour in 2015 proposed the binary version of the Ant Colony Optimization. Each ant is placed different or the same corners at the beginning of the problem. They find the shortest path to reach their food. The ants secrete the chemical called pheromone. Whenever an ant goes for the search of their food they leave behind the pheromone trails which help the other ants to follow the same path. When path is longer then the liquid will vaporize and the ants will not able to reach their destination. In the binary version, all solutions are binarized.

Chapter 3

Problem Statement

This chapter incorporates the motivation of the proposed algorithms, i.e., binary spotted hyena optimizer and binary emperor penguin optimization algorithm. The problem statement has been mentioned below in this chapter. The objectives of this thesis work are discussed below.

3.1 Motivation

Metaheuristic techniques can solve most of the real-life problems but not for all of them. To solve such problems, the binary versions of various metaheuristic algorithms came into existence. The binary metaheuristic algorithms are able to solve the discrete optimization problems such as feature selection [34], data mining [35] and unit commitment problem [36]. In such kind of problems, the search space is discrete in nature. The binary metaheuristic algorithms provide better solutions in discrete search space. These algorithms utilize a discretization mechanism to change the continuous values into discrete values.

In both proposed idea, the search space, location of the prey, and search agents are modified into binary form using transfer function. All the solutions are confined to binary values, i.e., 0 or 1. The tangent hyperbolic test function has been utilized to distort the continuous position and then these values are used to update the position of spotted hyenas in case of BSHO and emperor penguins in case of BEPO. The hyenas and penguin update their position by switching their values between 0 and 1 in their respective algorithms.

3.2 Problem Statement

The metaheuristic algorithms have become popular for solving the real life complex problems in lesser amount of time. Though the continuous metaheuristic algorithms can solve most of these problems but there are some problems which need binary search space. Therefore to solve such kind of problems, binary metaheuristics came into existence.

3.3 Objectives of Thesis

1. To study and analyze of existing metaheuristic techniques.

2. To develop binary spotted hyena optimizer and binary emperor penguin optimizer metaheuristic algorithms.

3. To test and validate the proposed algorithms on benchmark test functions.

4. To apply the developed binary metaheuristic algorithms on feature selection problems.

Chapter 4

Binary Spotted Hyena Optimizer

This chapter includes binary spotted hyena optimizer (BSHO) which is inspired from the hunting behavior of spotted hyenas. In this, the hyenas use discrete binary search space and can solve discrete optimization problems.

As per knowledge of authors, the binary version of spotted hyena optimizer is not present till date. Therefore, the binary version of spotted hyena optimizer (BSHO) is developed which can deal with the discrete problems.

4.1 Mathematical formulation

The major difference between the original spotted hyena optimizer and its binary version (BSHO) is in the position updating mechanism of spotted hyenas. The tangent hyperbolic function is used to update the position of spotted hyenas according to the optimal solution obtained. In BSHO, the positions are switched between "0" and "1" only. Hence, the dimension of search space lies in range of 0 or 1. The mathematical representation of cluster formation is given below.

$$C_h = Q_k + Q_{k+1} + \cdots + Q_{k+N} \tag{4.1}$$

Where C_h is the cluster of optimal solutions obtained from proposed algorithm.

$$x_d^{s+1} = \begin{cases} 1\,, & T(C_h) \geq rand \\ 0\,, & Otherwise \end{cases} \tag{4.2}$$

$$T(C_h) = \tanh(C_h) = (exp^{-\tau(C_h)} - 1)/(exp^{-\tau(C_h)} + 1) \tag{4.3}$$

where rand represents random number [0,1]. x_d^{s+1} specifies binary position of spotted hyena, *d* is the dimension, *s* as number of iteration and τ is 1.

4.2 Proposed algorithm

The concept behind this work is how the spotted hyena looks for their target in discrete binary search space. It mimics hunting behavior of spotted hyenas. The spotted hyena hunts in a group and form the group of trusted members. They search the prey, encircle them, and then attack. The search space in binary spotted hyena optimizer is examined as the hypercube and hyenas will be able to shift only to the corners of the hypercube. All the solutions are in binary form, i.e., either 0 or 1. The positions of spotted hyenas are updated according to the best-spotted hyena position. Proposed tangent hyperbolic function is used for updating the position of the spotted hyenas. Updated positions lie between "0" and "1" using the proposed tangent function.

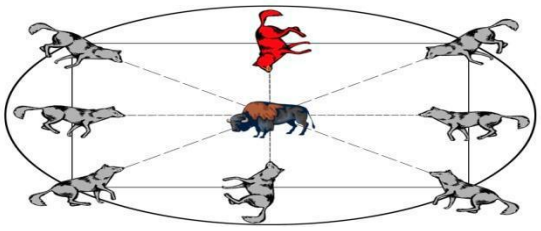


Figure 4.1. Movement of spotted Hyenas around the prey

Binary Spotted Hyena Optimizer Algorithm (BSHO)

Input: Number of Spotted hyenas, $\boldsymbol{Q_i(i = 1, 2 \ldots . . n)}$
Output: Best Spotted Hyena

1. The population of n spotted hyenas is initialized randomly
2. Calculate fitness value of each search agent.
3. **while** (s < Maximum no of Iterations)
4. **for** every spotted hyena
5. position of the search agent is updated
6. **end for**
7. Upgrade control parameters $\boldsymbol{Y, Z, h, and\ N}$
8. Enumerate fitness value of every spotted hyena
9. Update $\boldsymbol{Q_h}$ if its better than the previous one
10. Update cluster $\boldsymbol{C_h}$ w.r.t $\boldsymbol{Q_h}$
11. $\boldsymbol{s = s + 1}$
12. end while

4.3 Experimental results

4.3.1 Benchmark test functions

The benchmark functions are categorized into four categories unimodal, multimodal, fixed-dimensional multimodal and composite functions. All these benchmark functions are described in Appendix A.1 to A.4 In these tables, *Dim* designates the dimensions of benchmark test function. *Range* indicates the boundary.

The first category consists of 7 benchmark test functions (F1-F7). They have only one global optimum value and do not have any local optima. Multimodal functions consist of nine functions (F8-F13). These have many local solutions along with global optimum. The next category is fixed-dimension multimodal, that consists of ten functions (F14-F23). The last category is composite functions that have six test functions (F24-F29).

4.3.2 Parameter setting and compared algorithms

For analyzing the performance of binary spotted hyena optimizer (BSHO), it is compared with four metaheuristic algorithms namely Binary Grey Wolf Optimizer (BGWO), Binary Gravitational Algorithm (BGSA), Binary Particle Swarm Optimization (BPSO), Binary Bat Algorithm (BBA). The parameter settings are shown in Table 5. Experiments are conducted on MATLAB R2018a running on Window 7 having 2.30 GHz and 8GB RAM.

For all the above-mentioned algorithms, the size of the population and maximum number of iterations are set to 30 and 500, respectively. Mean and standard deviation of the results obtained from metaheuristic algorithms are mentioned in the tables.

Table 4.1 Parameter settings

Algorithms	Parameters	Values
BSHO	Search Agents	30
	Control Parameter($\vec{h}$)	[5,0]
	Constant $\vec{M}$	[0.5,1]
	Iterations	500
	Runs	30
BGWO (Emary et al., 2016) [29]	Search Agents	30
	Control Parameter($\vec{a}$)	[2,0]
	Iterations	500
	Runs	30
BGSA (Rashedi et al., 2009b) [31]	Search Agents	30
	Gravitational Constant G_0	100
	Iterations	500
	Runs	30
BPSO (Chuang et al., 2008) [32]	Search Agents	30
	Maximum velocity	6
	Constants c_1 and c_2	2,2
	Inertia weight	Increase from 0.4 to 0.9
	Iterations	500
	Runs	30
BBA (Nakamura et al., 2012) [30]	Search Agents	30
	Range of frequency	[0,2]
	Loudness	0.25
	Pulse rate	0.5
	α	0.9
	γ	0.9
	Iterations	500
	Runs	30
BWOA (Kumar and Kumar, 2018) [37]	Search Agents	30
	Iterations	500
	Runs	30
	au	Decreased from 2 to 0
	b	1
BDA (Mirjalili, 2016) [33]	Search Agents	30
	Iterations	500
	Runs	30
	Cohesion Weight	0.7
	Separation Weight	0.1
	Alignment Weight	0.1
	Inertia Weight	Decreased from 0.9 to 0.2
	Enemy Factor	1

4.3.3 Performance comparison

4.3.3.1 Evaluation of unimodal test functions (F1-F7)

The proposed BSHO is tested on unimodal benchmark functions (i.e., F1-F7). Table 4.2 reveals the results of the BSHO and other algorithms. The results which are marked as bold are better than the others. The BSHO has the best optimal solutions and better exploitation capability than the compared algorithms. For most of the functions the result of BSHO is better than the others as it has lower values for average and standard deviation. For F4 function, BGWO gives better result. Therefore, the proposed BSHO is able to provide better exploitation capability for the majority of the functions.

4.3.3.2 Evaluation of multimodal test functions (F8-F13)

The proposed BSHO is tested on multimodal benchmark functions (i.e., F8-F13). Table 4.3 reveals the results obtained from the proposed BSHO and other algorithms. It has been seen that BSHO has superior exploration capability as compared to others for the multimodal functions (i.e., F8-F12). The results reveal that BSHO excels in terms of exploration.

4.3.3.3 Evaluation of fixed-dimension multimodal test functions (F14-F23)

The proposed BSHO has been further tested on fixed-dimension multimodal benchmark functions (i.e., F14-F23). Table 4.4 reveals the results of proposed BSHO and other algorithms. BSHO algorithm provides better exploration capability for the fixed dimensional multimodal functions.

Results obtained from Tables 4.3 and 4.4 confirm the exploration capability of BSHO. The solutions generated from BSHO are not trapped into the local optima. The cluster of hyenas avoids the solution being stuck in local optima.

4.3.3.4 Evaluation of composite test functions (F24-F29)

The composite functions are very complex than the above-mentioned test functions. The proper balance between exploitation and exploration is required to solve these composite functions. Table 4.5 shows the results of BSHO and other algorithms. The results in the table 4.5 shows BSHO performs better for functions 25, 26, 28 and 29.

Table 4.2: Results of BSHO and other algorithms on unimodal functions

	BSHO		**BGWO**		**BGSA**		**BPSO**		**BBA**		**BWOA**		**BDA**	
	Avg	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**
F1	**0.00E+00**	**0.00E+00**	0.14E+01	0.57E+01	9.54E+03	1.35E+05	0.07E+01	0.18E+01	0.12E+01	0.13E+01	0.71E+01	2.71E+01	9.45E-02	6.34E-02
F2	**0.00E+00**	**0.00E+00**	0.14E+01	0.57E+01	1.50E+22	7.05+E21	0.06E+01	0.17E+01	0.11E+01	0.10E+01	1.85E+02	8.00E+01	4.87E-03	2.11E-03
F3	**0.00E+00**	**0.00E+00**	1.56E+01	2.88E+02	2.67E+06	8.90E+06	1.11E+01	2.47E+01	2.23E+01	1.55E+01	2.87E+01	0.25E+01	2.10E+01	4.76E+01
F4	1.82E-01	8.33E-01	2.53E-01	**3.33E-02**	8.65E-02	9.98E+01	**0.00E+00**	0.10E+01	0.03E+01	0.08E+01	3.45E+03	5.61E+02	9.98E-01	9.00E-01
F5	**0.00E+00**	2.90E+01	0.01E+00	**0.00E+00**	6.11E+07	7.27E+08	9.93E+01	3.04E+02	1.63E+02	3.39E+01	7.31E+01	6.19E+01	7.44E+01	8.91E+01
F6	**0.10E+01**	**0.77E+01**	0.25E+01	1.92E+01	7.93E+03	1.34E+05	0.11E+01	1.11E+01	0.19E+01	0.93E+01	3.23E+02	6.00E+01	2.98E+03	1.94E+02
F7	**7.13E-05**	**6.75E-05**	1.57E+01	1.74E+01	4.19E+01	4.11E+02	0.52E+01	1.43E+01	2.42E+01	3.75E+01	2.00E-04	3.98E-04	9.81E-04	7.67E-04

Table 4.3: Results of BSHO and other algorithms on multimodal functions

	BSHO		BGWO		BGSA		BPSO		BBA		BWOA		BDA	
	Avg	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**
F8	5.61E+02	**2.22E+04**	**1.08E-14**	2.52E+01	5.22E+02	2.44E+03	4.64E+02	-1.9E+03	0.11E+01	-1.7E+01	2.71E+03	3.99E+03	1.23E+06	2.21E+06
F9	**0.00E+00**	**0.00E+00**	0.14E+01	0.45E+01	2.85E+01	6.79E+02	0.07E+01	0.16E+01	0.10E+01	0.08E+01	1.87E+02	5.70E+02	1.65E+01	2.00E+01
F10	**0.00E+00**	**8.88E-16**	0.02E+01	0.16E+01	7.43E-02	2.15E+01	0.01E+01	0.09E+01	0.04E+01	0.05E+01	4.45E-05	9.97E-07	1.08E+02	2.35E+02
F11	**0.00E+00**	**0.00E+00**	7.92E-02	0.02E+01	7.29E+01	1.18E+03	0.18E-01	0.48E-01	0.31E-01	0.45E+01	2.57E+04	2.89+05	5.97E+03	2.90E+03
F12	**0.64E-01**	**0.17E+01**	0.03E+01	0.26E+01	2.08E+08	2.02E+09	0.96E-01	0.19E+01	0.01E+01	0.18E+01	1.29E+01	4.98E+02	2.87E+01	4.12E+01
F13	8.58E-02	8.76E-01	**5.56E-48**	**1.34E-32**	3.01E+08	3.42E+09	0.53E-01	0.01E+01	1.15E-01	0.99E+00	8.61E-01	9.45E-01	4.14E-01	0.11E-01

Table 4.4: Results of BSHO and other algorithms on fixed-dimension multimodal functions

	BSHO		**BGWO**		**BGSA**		**BPSO**		**BBA**		**BWOA**		**BDA**	
	Avg	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**
F14	**0.31E-16**	**1.13E+01**	3.61E-15	1.26E+01	2.01E-04	4.99E+02	**3.61E-15**	1.26E+01	3.61E-15	1.26E+01	7.12E-02	6.09E-07	2.79E+04	5.61E+06
F15	**0.52E-01**	**9.31E-02**	0.11E+01	1.48E-01	3.32E+10	6.09E+09	1.02E+01	1.48E-01	2.33E+01	1.48E-01	3.21E+01	5.84E+01	7.69E+02	7.14E+02
F16	**3.00E-01**	**-2.09E-01**	0.10E+01	9.08E+01	8.14E+02	3.77E+03	1.01E+02	2.20E+02	4.28E+02	1.01E+02	3.65E+01	4.01E+01	2.97E+03	3.41E-+03
F17	2.04E+02	3.49E+01	1.24E+00	1.09E+03	**9.5E-02**	**9.17E-03**	5.10E-01	4.30E+00	5.41E-01	9.10E-01	2.09E+01	9.21E+01	2.60E+04	4.23E+03
F18	**1.48E+01**	**2.04E+01**	0.23E+03	1.60E+03	1.94E+07	5.61E+06	1.05E+04	0.60E+04	2.03E+03	0.60E+03	1.87E+03	2.01E+04	4.15E+04	4.60E+04
F19	**3.74E-21**	**-0.32E+01**	1.69E-16	-3.34E-01	2.06E-20	-3.77E-05	1.69E-16	3.34E-01	1.69E-16	3.34E-01	2.84E-05	3.67E-04	5.41E-04	7.84E+01
F20	**4.99E-19**	**-0.15E+01**	0.43E-01	-1.45E-01	6.7E-08	-4.53E-08	2.82E-17	-1.65E-01	0.17E-01	-1.57E-01	0.98E-05	2.90E-01	6.36E-07	4.67E-02
F21	4.52E-01	-5.57E-01	**4.51E-15**	**-0.50E+01**	0.22E-02	-4.54E-01	**4.51E-15**	**-0.50E+01**	0.12E+01	-0.46E+01	3.80E+04	5.79E+04	3.64E+04	6.78E+04
F22	**1.91E-03**	**-6.02E-04**	2.08E+01	0.50E+01	0.41E-02	-0.61E-01	2.44E+01	0.50E+01	7.62E-01	0.49E+01	3.03E-02	2.00E-01	6.49E-01	4.64E-02
F23	3.81E-01	-8.26E-01	**2.71E-15**	**-0.51E+01**	0.53E-02	-0.95E-01	**2.71E-15**	**-0.51E+01**	0.12E+01	-0.49E+01	1.45E+01	2.45E+01	6.31E+04	3.67E+04

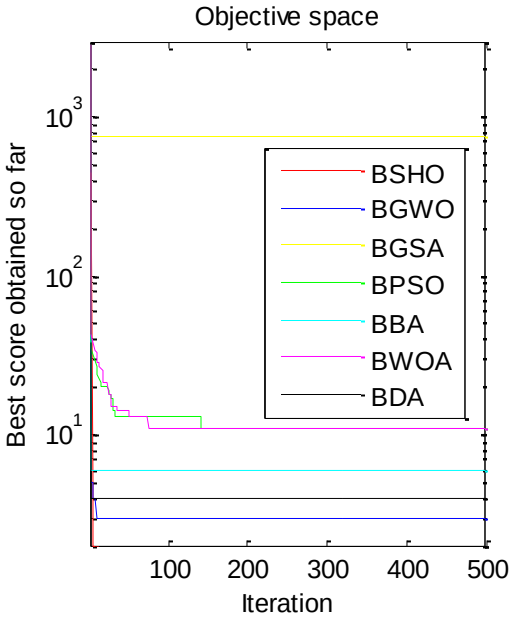
Table 4.5: Results of BSHO and other algorithms on composite functions

	BSHO		BGWO		BGSA		BPSO		BBA		BWOA		BDA	
	Avg	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**
F24	1.13E+02	5.20E+02	1.50E+01	6.46E+02	6.19E+01	**2.98E+02**	**3.06E+00**	6.38E+02	3.31E+01	7.21E+02	4.43E+03	3.75E+04	6.98E+05	1.81E+05
F25	**0.28E+01**	**1.23E+01**	1.11E+01	2.11E+01	5.51E+01	3.32E+02	4.02E+02	3.78E+02	1.99E+03	9.21E+03	3.09E+04	8.11E+03	9.09E+08	6.87E+08
F26	**1.41E+02**	**1.91E+01**	9.89E+02	2.18E+03	9.33E+03	1.65E+03	4.13E+04	1.98E+03	1.75E+02	2.22E+03	9.34E+05	1.87E+04	2.98E+08	4.00E+09
F27	5.11E+01	2.41E+03	4.56E+04	7.12E+03	1**.23E+01**	4.32E+02	1.98E+04	3.32E+03	3.27E+03	**0.45E+01**	9.45E+04	2.90E+01	7.09E+03	3.23E+03
F28	**1.63E+01**	**1.29E+01**	3.37E+02	1.07E+03	1.67E+02	2.55E+03	7.46E+01	1.03E+03	3.87E+01	1.19E+03	2.98E+03	5.43E+03	2.34E+02	3.13E+02
F29	**0.00E+00**	**0.12E+00**	0.37E+01	0.88E+01	0.37E+02	4.62E+03	9.29E+01	0.20E+02	3.37E+00	5.71E+00	3.65E-02	1.54E-01	1.43E-02	3.34E-02

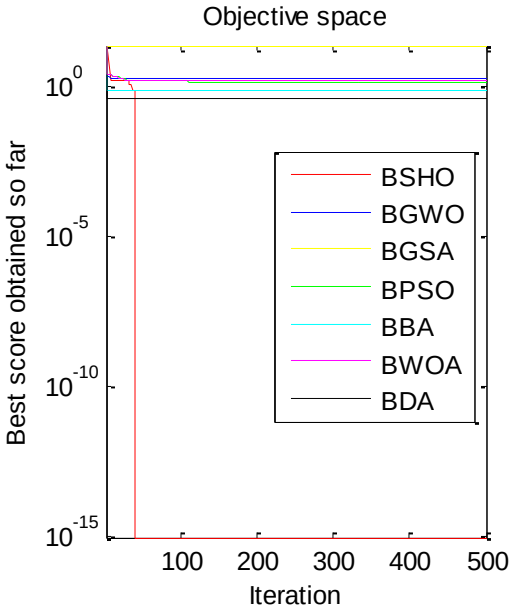
4.4 Convergence analysis

Analysis of convergence has been done to determine the exploration and exploitation capability of BSHO. Fig. 4.2 shows the convergence curves obtained from BSHO, BGWO, BGSA, BPSO, and BBA. It can be seen from Fig. 2 that the spotted hyenas in BSHO explore the most promising area of search space. Initially, BSHO converges more for most regions of search space. The behavior of BSHO is depicted in F6, F10, F18, and F29 test functions. BSHO converges towards the optimal solution at the end. The results reveal that BSHO provides a better balance between exploitation and exploration to obtain an optimal solution.

The proposed updated mechanism of spotted hyenas provides better exploration than the other metaheuristic algorithms such as BGWO, BGSA, BPSO, and BBA. BSHO avoids the local optimum solution during the iteration process. The above-mentioned algorithms are unable to explore the search space as seen in F10, F18, and F29 test functions.



F6



F10

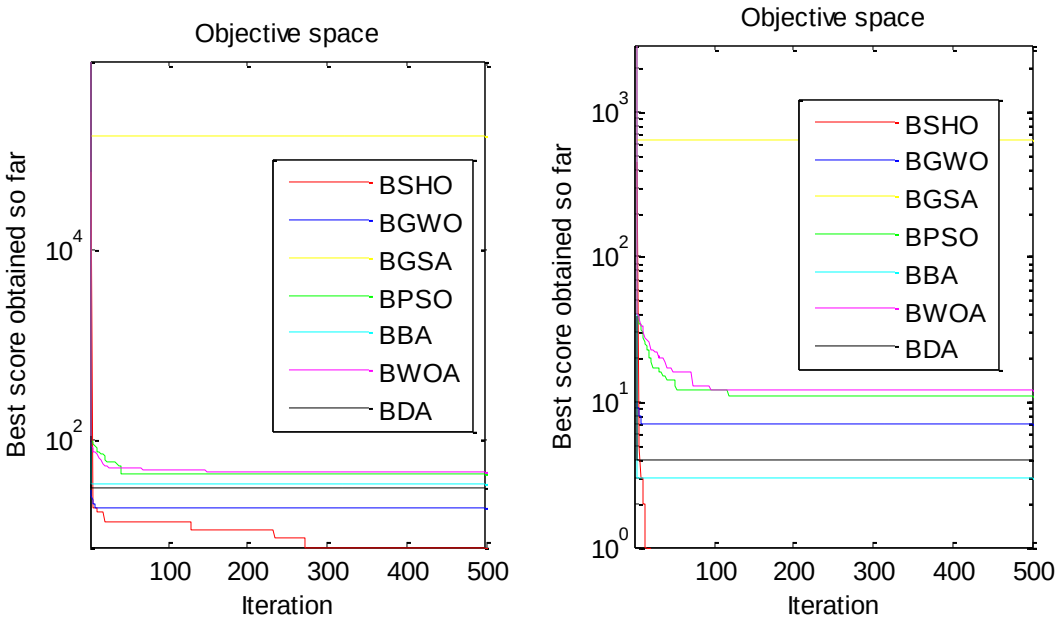
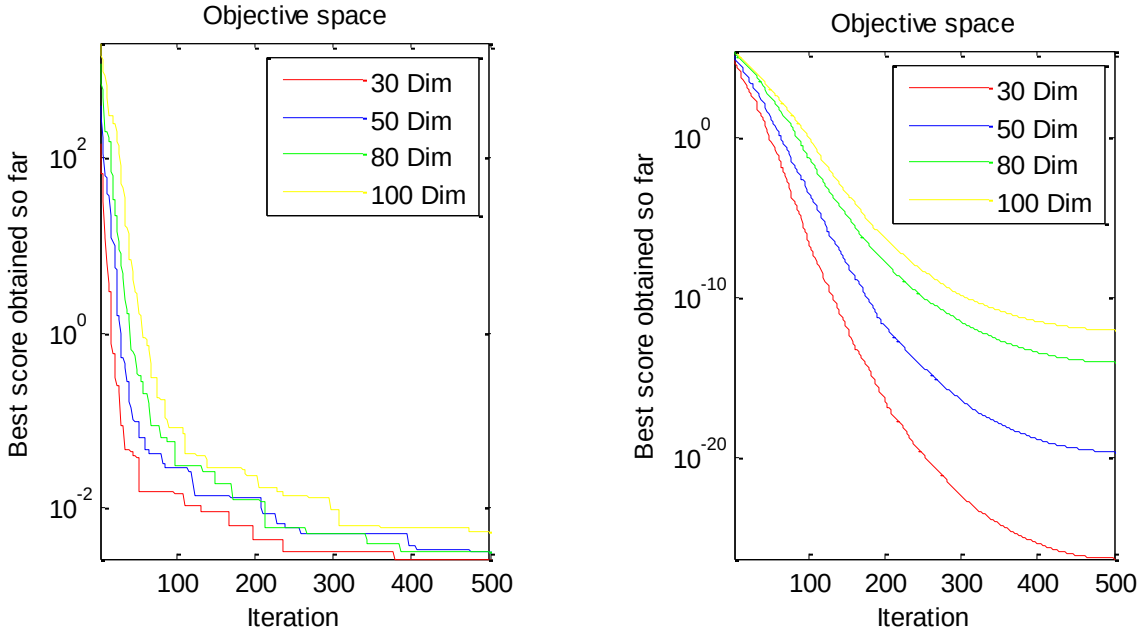


Figure 4.2: Convergence curves of BSHO and other algorithms on benchmark functions

4.5 Scalability analysis

The scalability test has been done on the proposed BSHO to confirm its applicability in complex problems. Different dimensions of test functions has been taken. We have taken 30, 50, 80, and 100. Fig. 4.3 depicts scalability effect of BSHO. It is observed from Fig. 4.3 that the BSHO is not affected much when the dimension is increased. Therefore, BSHO is suitable for immensely scalable environment.



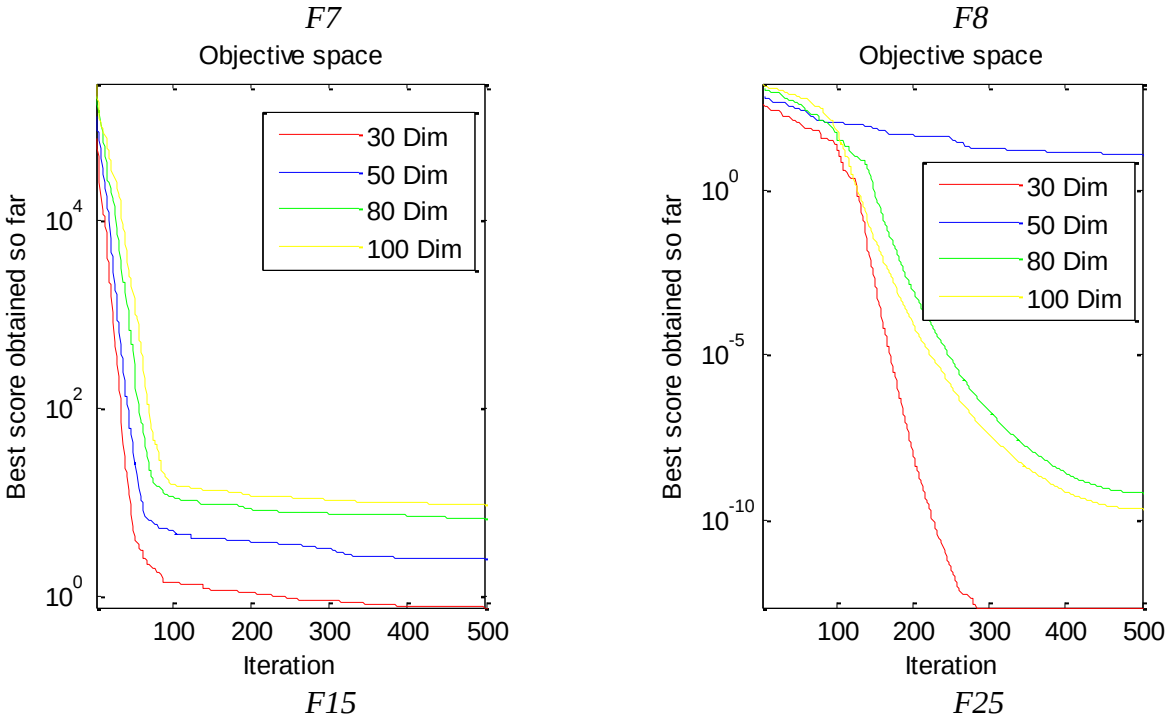


Figure 4.3: Effect of scalability analysis on proposed BSHO algorithm

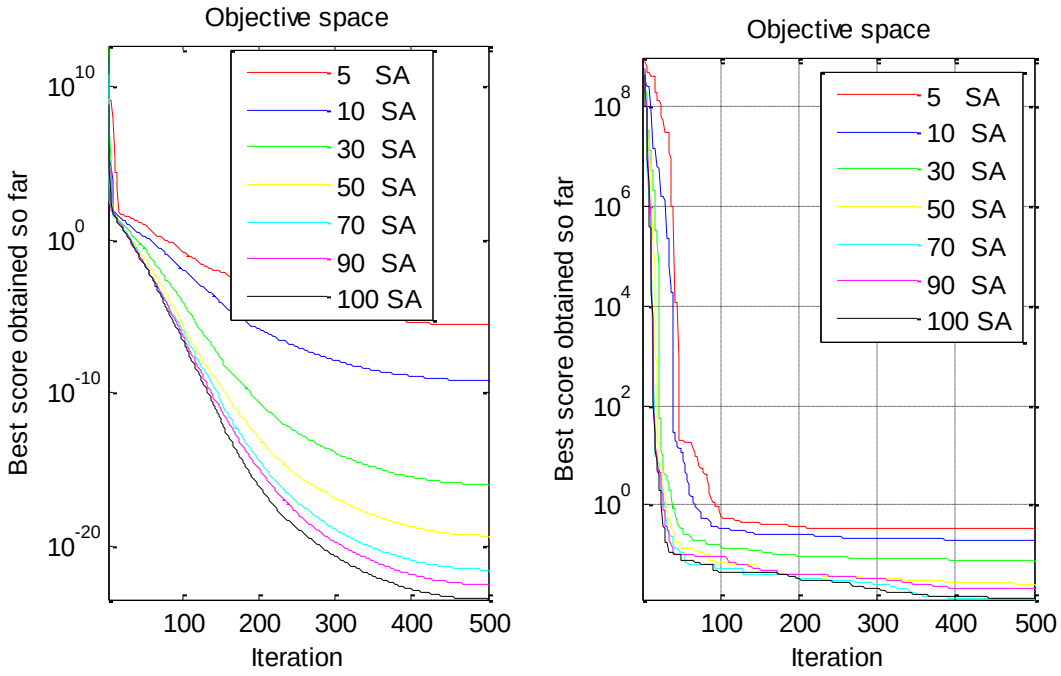
4.6 Sensitivity analysis

BSHO has different parameters such as number of search agents, maximum number of iterations, and parameter $\vec{h}$. The sensitivity analysis of has been introduced in this section.

4.6.1 Number of search agents

Search agents are increased from 30 to 50, 50 to 80, and 80 to 100 for analyzing performance of BSHO on benchmark test functions. Fig. 4.4 shows the effect of number of search agents. The results unveils that the BSHO offers best optimal solution when number of search agents are increased.

F7 F8



F15 *F25*

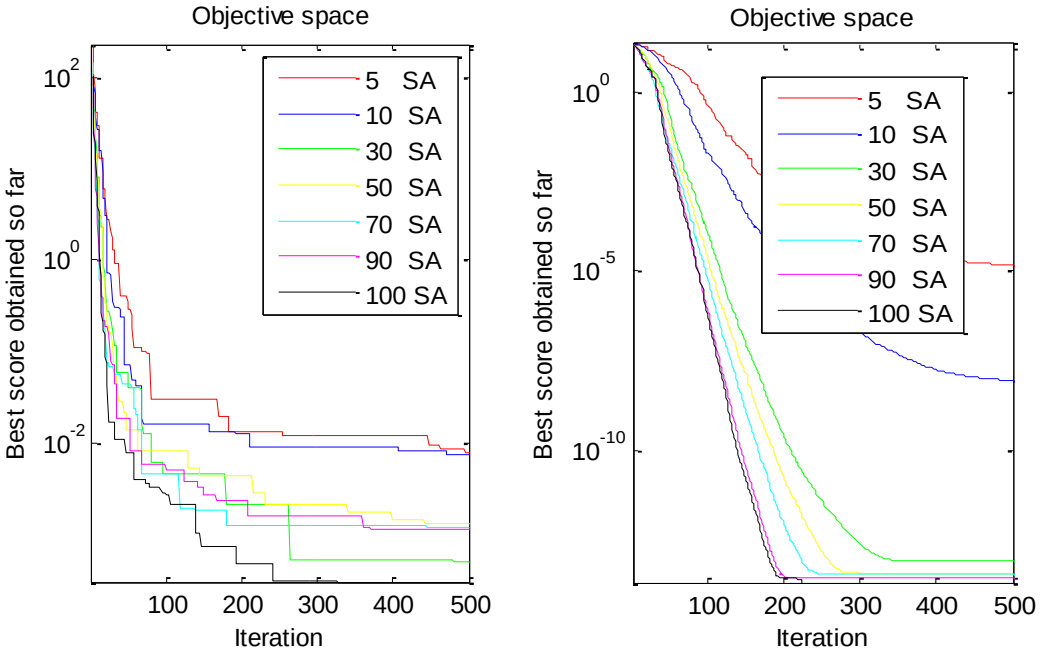
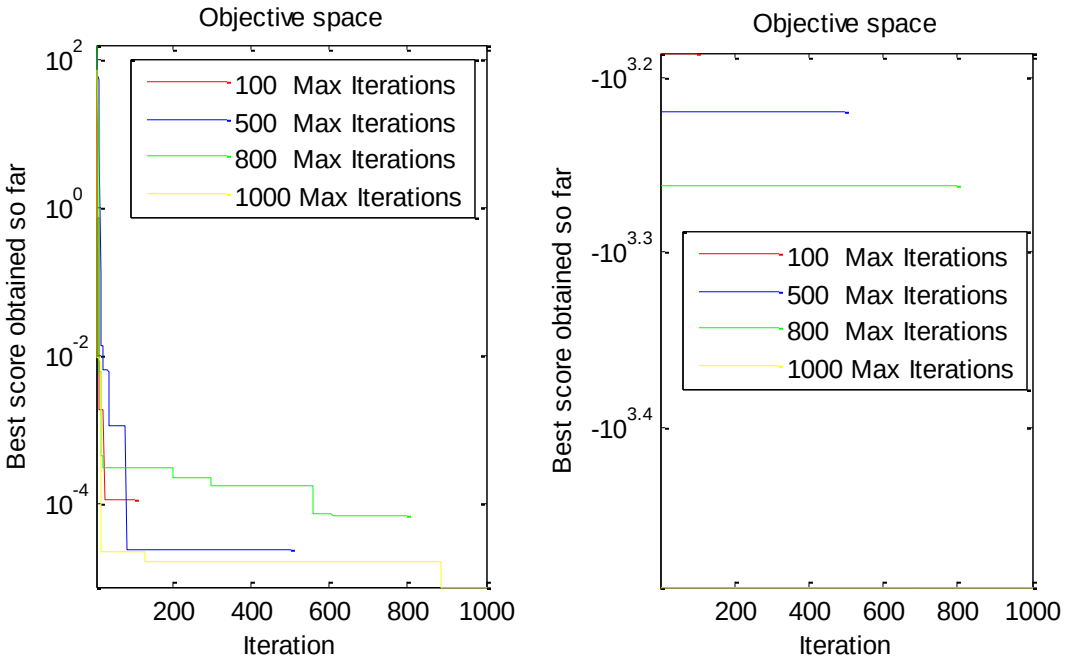


Figure 4.4: Effect of different search agents number on BSHO algorithm

4.6.2 Maximum number of iterations

The number of iterations varies between 100, 500, 800 and 1000. Fig. 4.5 reveals the effect when iterations changes on BSHO. It can be seen from Fig. 4.5 that BSHO converges more towards best solution when the iterations increases.

F7 *F8*



F15 *F25*

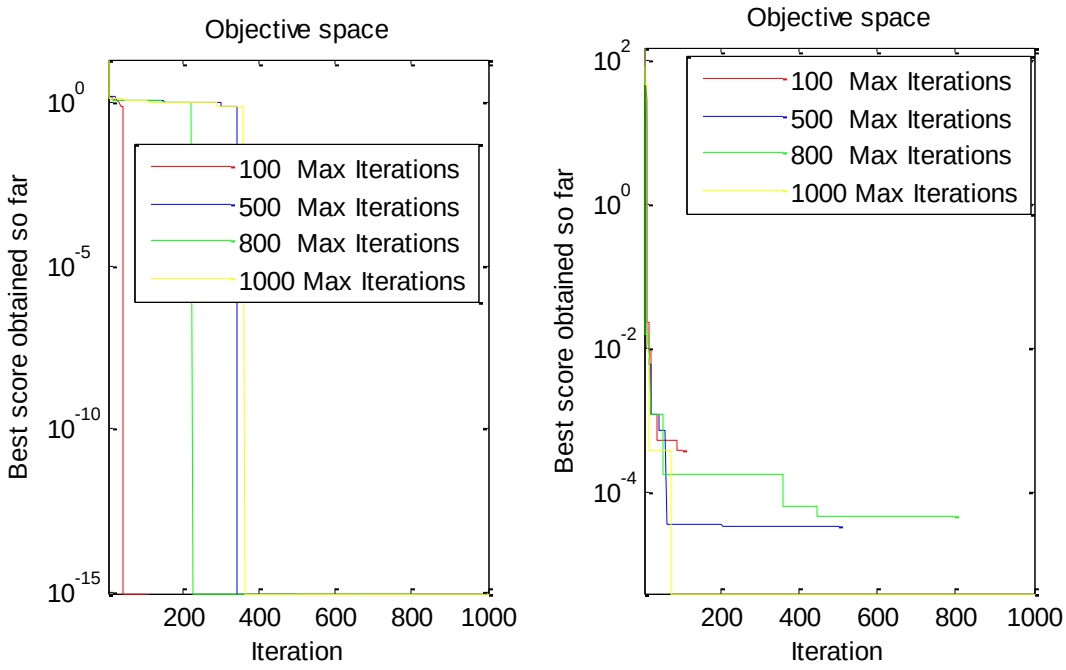
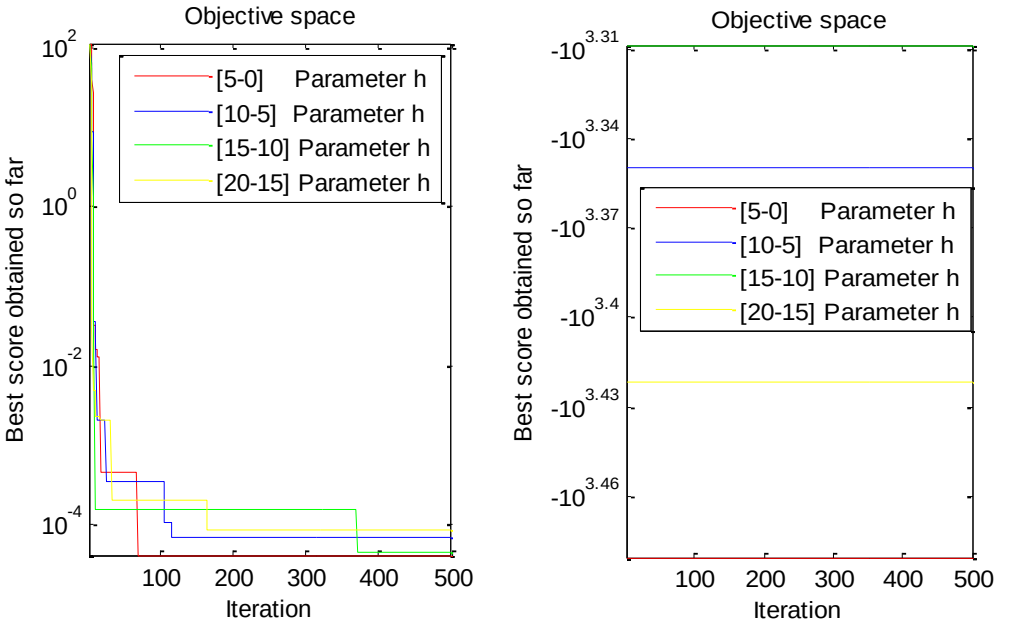


Figure 4.5: Effect of number of iterations on the proposed BSHO algorithm

4.6.3 Parameter $\vec{h}$

The value of $\vec{h}$ used in this experimentation lies in the range of [5,0], [10,5], and [15,10]. Fig. 4.6 shows the effect of $\vec{h}$. It can be seen that BSHO gives results better than others when the value of $\vec{h}$ is set to [5,0].

F7 *F8*



F18 *F25*

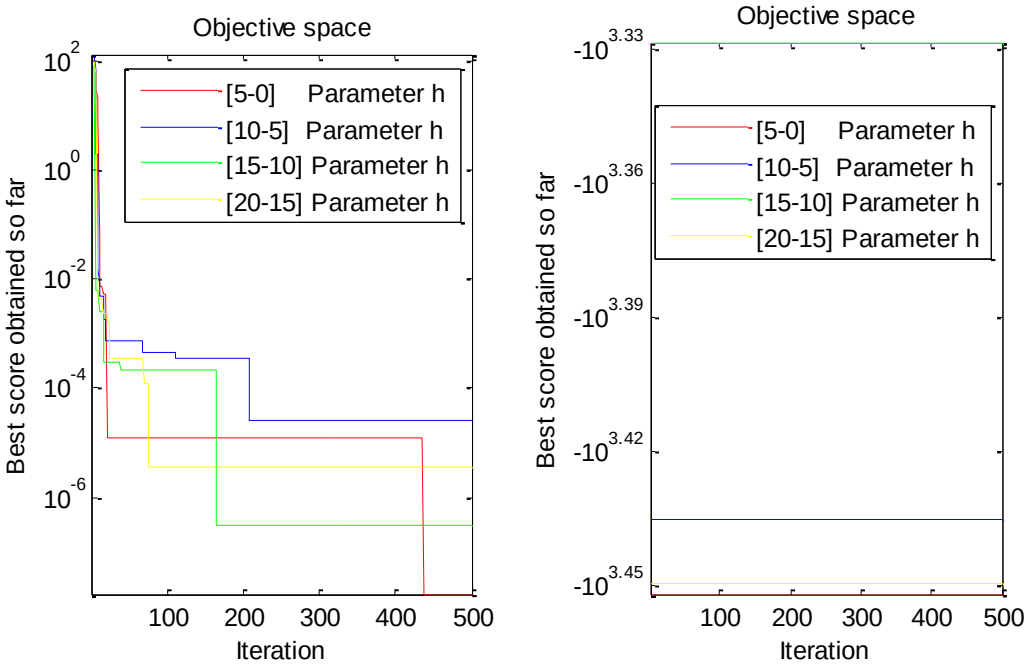


Figure 4.6: Effect of $\vec{h}$ parameter on the proposed BSHO algorithm

4.7 Application on feature selection

The proposed algorithm is used as a wrapper approach for feature selection. The well-known classification technique is K-nearest neighbor method [37]. The classification rules are generated by training the sample data without any additional information. This classification method can estimate the test's sample category based on the minimum distance between training data and testing data [37].

4.7.1 Datasets used and performance measure

Eleven datasets have been used from UCI machine learning repository to compare BSHO with four well-known binary metaheuristic techniques Table 4.1 shows the parameter setting of BSHO. The datasets used for comparison are listed in Table 4.7 The performance of above-mentioned algorithms is tested on following fitness function.

$$Fitness = \lambda \times Error_{rate}(D) + \omega \times \frac{|len_{fsub}|}{|Total\_fea|} \qquad (14)$$

where $Error_{rate}\ (D)$ is error rate obtained from the classifier on feature set *D*, $|len_{fsub}|$is length of selected feature set, and $|Total_{fea}|$ is the total number of features. The values of λ, ω are set to 0.99 and 0.01 respectively, as reported in literature [28].

Table 4.6: Parameter settings for feature selection

Algorithms	Parameters
Proposed BSHO	Control Parameter($\vec{h}$)
	Constant $\vec{M}$
BGWO (Emary et al., 2016)	Control Parameter($\vec{a}$)
BGSA (Rashedi et al., 2009b)	Gravitational Constant G_0
BPSO (Chuang et al., 2008)	Maximum velocity
	Constants c_1 and c_2
	Inertia weight
BBA (Nakamura et al., 2012)	Range of frequency
	Loudness
	Pulse rate
	α
	γ
BWOA (Kumar and Kumar, 2018)	au
	b
BDA (Mirjalili, 2016)	Cohesion Weight
	Separation Weight
	Alignment Weight
	Inertia Weight

Algorithms	Enemy Factor Parameters
Proposed BSHO	Control Parameter($\vec{h}$)
	Constant $\vec{M}$

Table 4.7: Description of datasets

Datasets	**No. of instances**	**No. of attributes**
Wine	178	13
Lymphography	148	18
Zoo	101	17
Bupa	345	07
Glass	214	09
Planning relax	182	13
Haberman	306	06
CMC	1473	09
WDBC	576	30
SonarEW	208	60
PenglungEW	325	73

4.7.2 Results and discussions

Tables 4.8 and 4.9 show the standard deviation and average of fitness function obtained from compared algorithms. It has been observed from table 4.8 and 4.9 that results which are marked as bold are better. The proposed algorithm provides superior results for the datasets.

Table 4.10 shows the mean computation time obtained from the proposed BSHO over all datasets. For the all datasets, the computational time is less than 1 minute. Hence, BSHO is applicable on high-dimensional datasets.

Table 4.8. Standard deviation

Datasets	BSHO	BGWO	BGSA	BPSO	BBA	BWOA	BDA
Wine	**0.14E-02**	1.11E-02	0.96E-02	0.31E-02	1.26E-02	0.55E-02	1.98E-02
Lymphography	**1.11E-02**	2.19E-02	2.12E-02	1.95E-02	1.48E-02	3.65E-02	1.60E-02
Zoo	**1.43E-02**	2.11E-02	2.96E-02	1.98E-02	2.25E-02	1.87E-02	2.72E-02
Bupa	**1.13E-16**	0.75E-02	1.15E-02	0.69E-02	1.61E-02	0.96E-02	6.91E-02
Glass	**0.00E+00**	7.84E-04	4.79E-04	0.11E-01	8.50E-04	0.86E-03	9.57E-07
Planning relax	**1.86E-04**	0.99E-02	2.50E-04	3.91E-04	1.17E-02	8.55E-04	0.60E-03
Haberman	**5.69E-17**	0.29E-02	0.43E-02	4.33E-16	0.85E-02	6.76E-01	2.95E-02
Cmc	**0.11E-01**	0.27E-01	0.34E-01	3.38E-01	0.56E-01	0.61E-01	1.92E-01
WDBC	**2.11E-07**	1.34E-03	0.31E-04	6.90E-01	8.65E-05	1.76E-04	348-03
SonarEW	**9.11E-05**	2.17E-01	3.91E-02	0.11E-01	1.11E-03	2.09E-03	4.64E-02
PenglungEW	**1.05E-04**	0.11E-03	1.76E-02	1.89E-02	0.22E-03	7.64E-03	0.87E-01

Table 4.9: Average

Datasets	BSHO	BGWO	BGSA	BPSO	BBA	BWOA	BDA
Wine	**8.73E-01**	9.05E-01	8.82E-01	8.76E-01	9.00E-01	8.87E-01	9.84E-01
Lymphography	**5.29E-01**	5.50E-01	5.37E-01	5.36E-01	5.83E-01	6.50E-01	5.98E-01
Zoo	**0.77E-01**	0.98E-01	0.79E-01	0.78E-01	1.37E-01	1.09E-01	0.96E-01
Bupa	**3.25E-01**	3.42E-01	3.34E-01	3.28E-01	3.50E-01	3.56E-01	3.61E-01
Glass	**2.98E-02**	3.08E-02	3.01E-02	2.95E-02	3.04E-02	3.09E-02	3.00E-02
Planning relax	**2.41E-01**	2.58E-01	2.45E-01	2.42E-01	2.62E-01	3.98E-01	2.74E-01
Haberman	**2.59E-01**	2.60E-01	2.61E-01	2.63E-01	2.62E-01	2.78E-01	2.61E-01
CMC	**4.74E-01**	4.78E-01	4.88E-01	4.80E-01	4.96E-01	4.95E-01	4.88E-01
WDBC	**0.23E-03**	1.41E-01	2.31E-01	8.23E-02	8.65E-02	4.98E-02	2.87E-02
SonarEW	**0.34E-02**	1.24E-01	4.00E-01	2.55E-01	8.09E-01	3.85E-01	1.91E-01
PenglungEW	**0.58E-04**	7.41E-02	2.38E-02	3.96E-02	7.15E-02	4.12E-03	0.93E-02

Table 4.10: Average computational time obtained from BSHO over 20 independent runs

Datasets	Wine	Lymphography	Zoo	Bupa	Haberman	Glass	WDBC	SonarEW	PenglungEW
Time (min:sec)	00:04.19	00:05.62	00:04.86	00:03.21	00:01.92	00:02.67	00.34.75	00.39.43	00.41.02

4.8 Summary

This chapter includes a BSHO. It utilizes the tangent function. BSHO is compared with four well-known binary versions of metaheuristic algorithms. The effects of sensitivity, scalability, and statistical analysis are also done on BSHO. The experiments reveal the superiority of BSHO over other compared algorithms. Moreover, it is also applied on feature selection problem. The results reveal the applicability of BSHO in feature selection problem.

Chapter 5

Binary Emperor Penguin Optimization

In this chapter Binary Emperor Penguin Optimization algorithm (BEPO) is proposed that uses the binary search space for updating position of the emperor penguins in accordance with the best optimal solutions.

5.1 Mathematical foundation of BEPO

The binary metaheuristic algorithms have been used for solving the discrete problems. Proposed algorithm uses the discrete search space instead of the continuous search space. This algorithm is used for solving discrete optimization problems. BEPO is applied on feature selection problem. Sigmoidal function is utilized for updating emperor penguins position according to the best optimal solutions

$$S\left(\overrightarrow{Q_{ep}}(s)\right) = \frac{1}{1+e^{-\overrightarrow{Qep}(s)}} \tag{5.1}$$

$$\vec{D} = \begin{cases} 1, & if\ S\left(\overrightarrow{Q_{ep}}(s)\right) > 0 \\ 0, & otherwise \end{cases} \tag{5.2}$$

Where $\overrightarrow{Q_{ep}}(s)$ is the updated position of the emperor penguin.

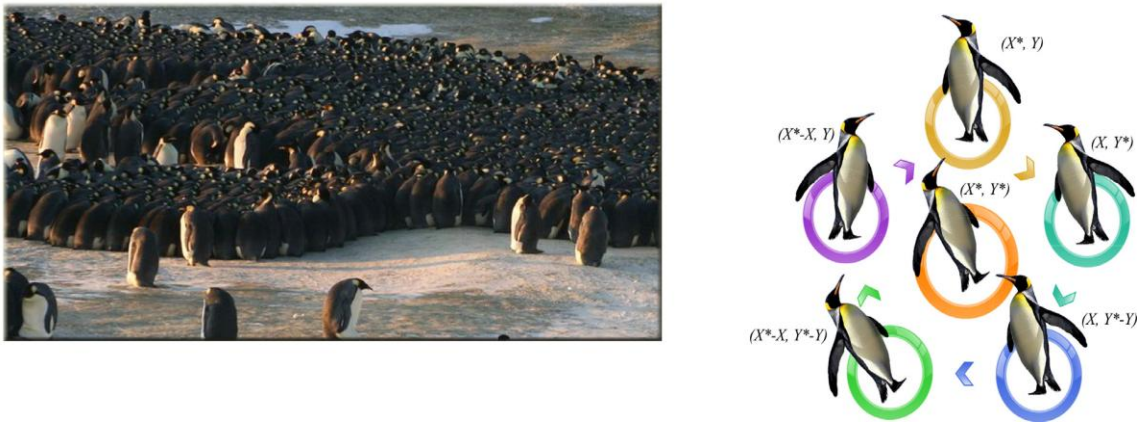


Figure 5.1: Huddling behavior of emperor penguins

The algorithm for BEPO is depicted below:

Binary Emperor Penguin Optimizer Algorithm

Input: Initialize the emperor penguin population $\overrightarrow{Q_{ep}}(i = 1, 2 \ldots\ldots n)$
Output: best optimal solution $\vec{Q}$
Procedure EPO
Initialize the parameters $T, \vec{Y}, \vec{Z}, B(), R \text{ and } Max_{iteration}$
Compute fitness value of each search agent
while(s < $Max_{iteration}$) do
FITNESS(Q_{ep})
$R \leftarrow Rand()$
If($R > 1$) then
$t \leftarrow 0$
Else
$t \leftarrow 1$
End if

$T \leftarrow (t - \frac{Max_{iteration}}{x - Max_{iteration}})$

For $i \leftarrow 1$ to n do
For $j \leftarrow 1$ to n do
Compute the vectors $\vec{Y} \text{ and } \vec{Z}$ using equations (6) and (8)
Compute the functions $B(Y)$ using equation (9)
Upgrade the current agent position using equation (10)
End for
End for
Upgrade parameters $T, \vec{Y}, \vec{Z} \text{ and } B()$
Reorganize the search agents that goes beyond the boundary
Compute (Q_{ep}) using sigmoid function
Update $\vec{Q}$ if there will be better solution than previous one
$s \leftarrow s + 1$
End while
Return $\vec{Q}$
End Procedure

Procedure FITNESS(Q_{ep})
For $i \leftarrow 1$ to n do
FIT[i]← FITNESS_FUNCTION(Q_{ep})
End for
$FIT_{best} \leftarrow BEST(FIT[])$
End FIT_{best}
End **procedure**

```
Procedure BEST(FIT[])
  best ← FIT[0]
    For i ← 1 to n do
        if  FIT[i] < best
            best ← FIT[i]
        End if
    End for
   Return best
End procedure
```

5.2 Experimental Results

5.2.1 Benchmark Test Functions

BEPO is verified on 29 benchmark test functions to evaluate its performance. The test functions are classified into different categories such as unimodal, multimodal, fixed dimensional multimodal and composite functions. The functions are detailed in Appendix A where f_{min} indicates the minimization function and Dim and Range indicates the dimensionality and boundary range of search space.

5.2.2 Algorithms used for comparison

In order to measure the performance of the BEPO, existing metaheuristic algorithms are used for comparison which are BGWO,BGSA, BPSO and BBA.

5.2.3 Experimental setup

Performance of these algorithms are conducted on the MATLAB R2018a on Windows 7 with 4GB RAM. Parameter settings of all metaheuristic algorithms are mentioned in table 5.1. All the algorithms have 30 independent runs each one of with 1000 iterations. The standard deviation and the average are calculated and compared for all the algorithms.

Table 5.1: Parameter settings

Algorithms	Parameters	Values
Binary Emperor Penguin Optimizer (BEPO)	Search Agents	30
	Temperature Profile(T)	[1,1000]
	Constant $\vec{Y}$	[-1.5,1.5]
	Function $B()$	[0,1.5]
	Parameter N	2
	Parameter g	[2,3]
	Parameter l	[1.5,2]
	Runs	30
	Iterations	1000
Binary Grey Wolf Optimizer (BGWO) [29]	Runs	30
	Search Agents	30
	Control Parameter($\vec{a}$)	[2,0]
	Iterations	1000
Binary Gravitational Search Algorithm (BGSA) [31]	Runs	30
	Search Agents	30
	Gravitational Constant G_0	100
	Iterations	1000
Binary Particle Swarm Optimization (BPSO) [32]	Runs	30
	Search Agents	30
	Maximum velocity	6
	Constants c_1 and c_2	2,2
	Inertia weight	Increase from 0.4 to 0.9
	Iterations	1000
Binary Bat Algorithm (BBA) [30]	Runs	30
	Search Agents	30
	Range of frequency	[0,2]
	Loudness	0.25
	Pulse rate	0.5
	α	0.9
	γ	0.9
	Iterations	500
	Runs	30

5.2.4 Performance Evaluation

5.2.4.1 Analysis on Function F1 to F7

There are seven benchmark test functions listed under the category of unimodal functions. Among them F4, F5, F6 and F7 performs well for the BEPO as compared to the others. Results are listed in the table 5.2.

5.2.4.2 Analysis on Function F8 to F23

Multimodal and Fixed dimension multimodal benchmark test functions comprises of sixteen functions from F8 toF23. In this category four among all performs well for the binary emperor penguin algorithm. Results of these functions have been listed in the table 5.3 and 5.4.

5.2.4.3 Analysis on Functions F24 to F29

The benchmark test function which comes under this category are composite functions which are of complex nature. The functions F25, F26, F27, F29 performs well for BEPO. Results are depicted in table 5.5.

Table 5.2: Results of BEPO and other algorithms on unimodal functions

	BEPO		**BGWO**		**BGSA**		**BPSO**		**BBA**	
	Std	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**Std**	**Avg**
F1	**0.00E+00**	**0.00E+00**	0.14E+01	0.57E+01	9.54E+03	1.35E+05	0.07E+01	0.18E+01	0.12E+01	0.13E+01
F2	**0.00E+00**	**0.00E+00**	0.14E+01	0.57E+01	1.50E+22	7.05+E21	0.06E+01	0.17E+01	0.11E+01	0.10E+01
F3	**0.00E+00**	**0.00E+00**	1.56E+01	2.88E+02	2.67E+06	8.90E+06	1.11E+01	2.47E+01	2.23E+01	1.55E+01
F4	1.82E-01	8.33E-01	2.53E-01	**3.33E-02**	8.65E-02	9.98E+01	**0.00E+00**	0.10E+01	0.03E+01	0.08E+01
F5	**0.00E+00**	2.90E+01	0.01E+00	**0.00E+00**	6.11E+07	7.27E+08	9.93E+01	3.04E+02	1.63E+02	3.39E+01
F6	**0.10E+01**	**0.77E+01**	0.25E+01	1.92E+01	7.93E+03	1.34E+05	0.11E+01	1.11E+01	0.19E+01	0.93E+01
F7	**7.13E-05**	**6.75E-05**	1.57E+01	1.74E+01	4.19E+01	4.11E+02	0.52E+01	1.43E+01	2.42E+01	3.75E+01

Table 5.3: Results of BEPO and other algorithms on multimodal functions

	BEPO		**BGWO**		**BGSA**		**BPSO**		**BBA**	
	Std	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**std**	**Avg**	**Std**	**avg**
F8	5.61E+02	**-2.22E+04**	**1.08E-14**	-2.52E+01	5.22E+02	2.44E+03	4.64E+02	-1.9E+03	0.11E+01	-1.71E+01
F9	**0.00E+00**	**0.00E+00**	0.14E+01	0.45E+01	2.85E+01	6.79E+02	0.07E+01	0.16E+01	0.10E+01	0.08E+01
F10	**0.00E+00**	**8.88E-16**	0.02E+01	0.16E+01	7.43E-02	2.15E+01	0.01E+01	0.09E+01	0.04E+01	0.05E+01
F11	**0.00E+00**	**0.00E+00**	7.92E-02	0.02E+01	7.29E+01	1.18E+03	0.18E-01	0.48E-01	0.31E-01	0.45E+01
F12	**0.64E-01**	**0.17E+01**	0.03E+01	0.26E+01	2.08E+08	2.02E+09	0.96E-01	0.19E+01	0.01E+01	0.18E+01
F13	8.58E-02	8.76E-01	**5.56E-48**	**1.34E-32**	3.01E+08	3.42E+09	0.53E-01	0.01E+01	1.15E-01	0.99E+00

Table 5.4: Results of BEPO and other algorithms on fixed-dimension multimodal functions

	BEPO		**BGWO**		**BGSA**		**BPSO**		**BBA**	
	Std	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**std**	**Avg**	**Std**	**Avg**
F14	**0.31E-16**	**1.13E+01**	3.61E-15	1.26E+01	2.01E-04	4.99E+02	**3.61E-15**	1.26E+01	3.61E-15	1.26E+01
F15	**0.52E-01**	**9.31E-02**	0.11E+01	1.48E-01	3.32E+10	6.09E+09	1.02E+01	1.48E-01	2.33E+01	1.48E-01
F16	**3.00E-01**	**-2.09E-01**	0.10E+01	9.08E+01	8.14E+02	3.77E+03	1.01E+02	2.20E+02	4.28E+02	1.01E+02
F17	2.04E+02	3.49E+01	1.24E+00	1.09E+03	**9.5E-02**	**9.17E-03**	5.10E-01	4.30E+00	5.41E-01	9.10E-01
F18	**1.48E+01**	**2.04E+01**	0.23E+03	1.60E+03	1.94E+07	5.61E+06	1.05E+04	0.60E+04	2.03E+03	0.60E+03
F19	**3.74E-21**	**-0.32E+01**	1.69E-16	-3.34E-01	2.06E-20	-3.77E-05	1.69E-16	3.34E-01	1.69E-16	3.34E-01
F20	**4.99E-19**	**-0.15E+01**	0.43E-01	-1.45E-01	6.7E-08	-4.53E-08	2.82E-17	-1.65E-01	0.17E-01	-1.57E-01
F21	4.52E-01	-5.57E-01	**4.51E-15**	**-0.50E+01**	0.22E-02	-4.54E-01	**4.51E-15**	**-0.50E+01**	0.12E+01	-0.46E+01
F22	**1.91E-03**	**-6.02E-04**	2.08E+01	0.50E+01	0.41E-02	-0.61E-01	2.44E+01	0.50E+01	7.62E-01	0.49E+01
F23	3.81E-01	-8.26E-01	**2.71E-15**	**-0.51E+01**	0.53E-02	-0.95E-01	**2.71E-15**	**-0.51E+01**	0.12E+01	-0.49E+01

Table 5.5: Results of BEPO and other algorithms on composite functions

	BEPO		**BGWO**		**BGSA**		**BPSO**		**BBA**	
	Std	**Avg**	**Std**	**Avg**	**Std**	**Avg**	**std**	**Avg**	**Std**	**Avg**
F24	1.13E+02	5.20E+02	1.50E+01	6.46E+02	6.19E+01	**2.98E+02**	**3.06E+00**	6.38E+02	3.31E+01	7.21E+02
F25	**0.28E+01**	**1.23E+01**	1.11E+01	2.11E+01	5.51E+01	3.32E+02	4.02E+02	3.78E+02	1.99E+03	9.21E+03
F26	**1.41E+02**	**1.91E+01**	9.89E+02	2.18E+03	9.33E+03	1.65E+03	4.13E+04	1.98E+03	1.75E+02	2.22E+03
F27	5.11E+01	2.41E+03	4.56E+04	7.12E+03	1**.23E+01**	4.32E+02	1.98E+04	3.32E+03	3.27E+03	**0.45E+01**
F28	**1.63E+01**	**1.29E+01**	3.37E+02	1.07E+03	1.67E+02	2.55E+03	7.46E+01	1.03E+03	3.87E+01	1.19E+03
F29	**0.00E+00**	**0.12E+00**	0.37E+01	0.88E+01	0.37E+02	4.62E+03	9.29E+01	0.20E+02	3.37E+00	5.71E+00

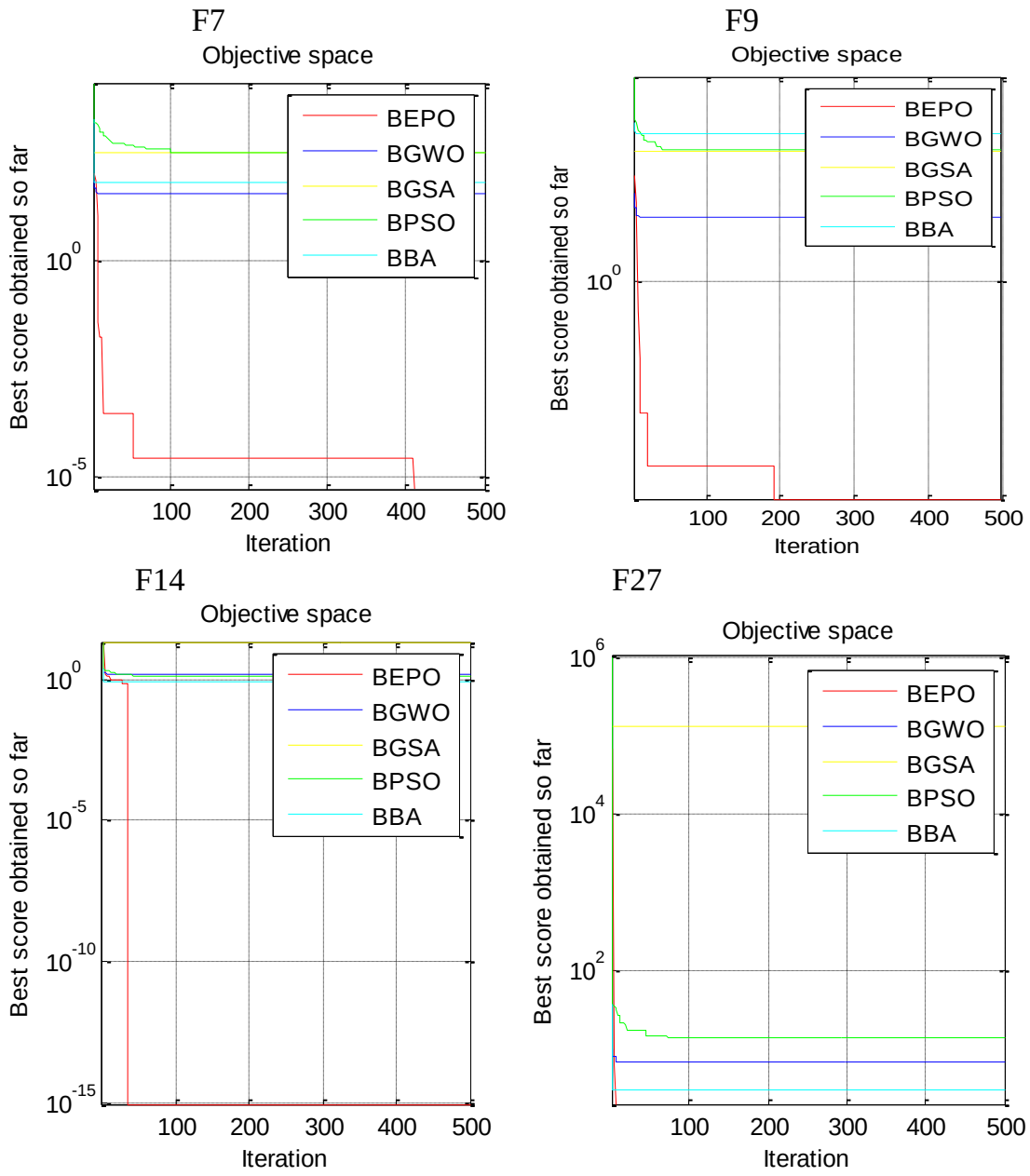


Figure 5.2: Convergence curves of BEPO and other algorithms on benchmark test functions

5.2.5 Scalability Analysis

The effect of scalability has been shown for different functions on the BEPO. Dimensions differ in the range 30, 50, 80 and 100. The convergence curves shown in the figure 2 describes the performance of the BEPO when the dimensions vary. It has been

seen from the figure that the performance of BEPO is not affected much when the dimensions are changed.

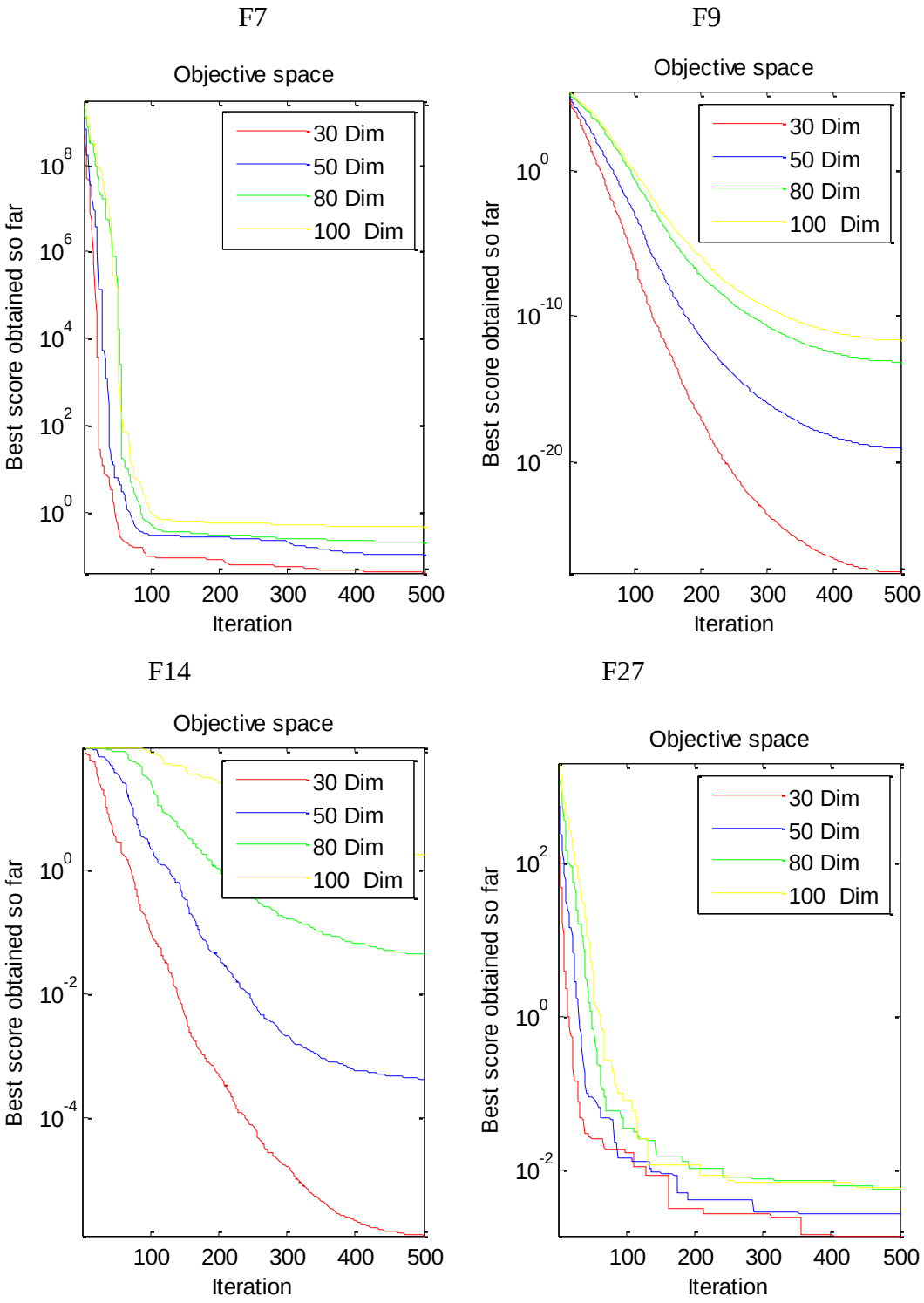


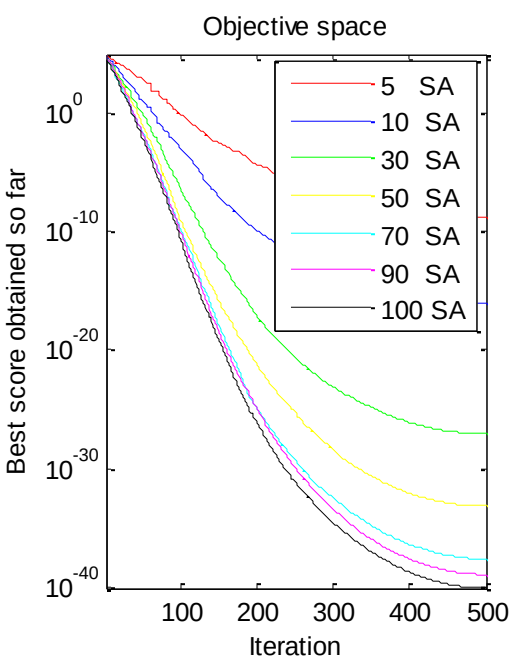
Figure 5.3: Effect of scalability analysis on proposed BEPO

5.2.6 Sensitivity Analysis

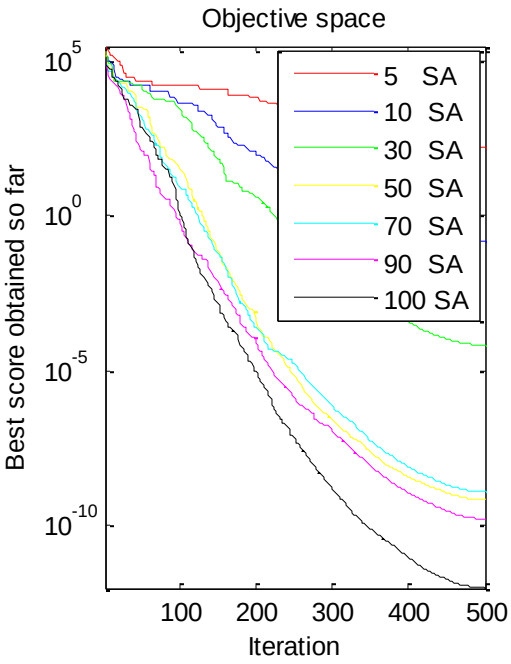
5.2.6.1 Number of Search Agents

For analyzing the performance of BEPO, number of search agents varied between 5, 10, 30, 50, 70, 90, 100 and tested on different benchmark functions. The curves are shown in the figure 3. It is seen from curves that BEPO performs better when number of search agents increases.

F7



F9



F14 F27

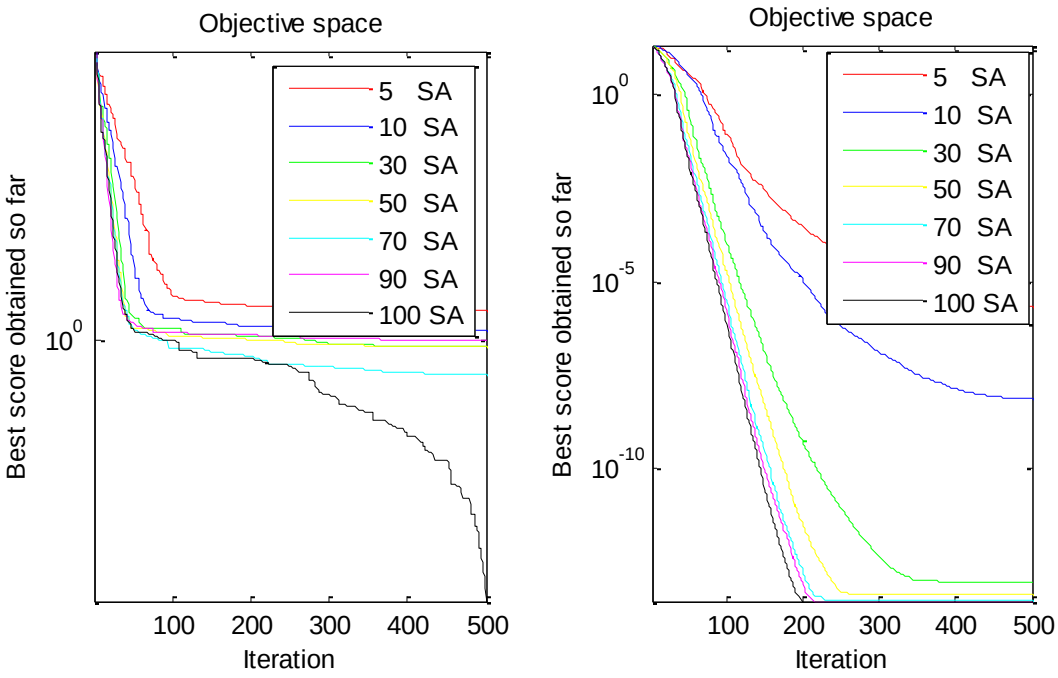


Figure 5.4: Effect of different search agents number on BEPO

5.2.6.2 Maximum number of iterations

The BEPO is verified on different test functions by varying number of iterations between 100, 500, 800 and 1000. Figure 4 shows convergence curves of BEPO on different number of iterations. BEPO performance is better when the number of iterations increases.

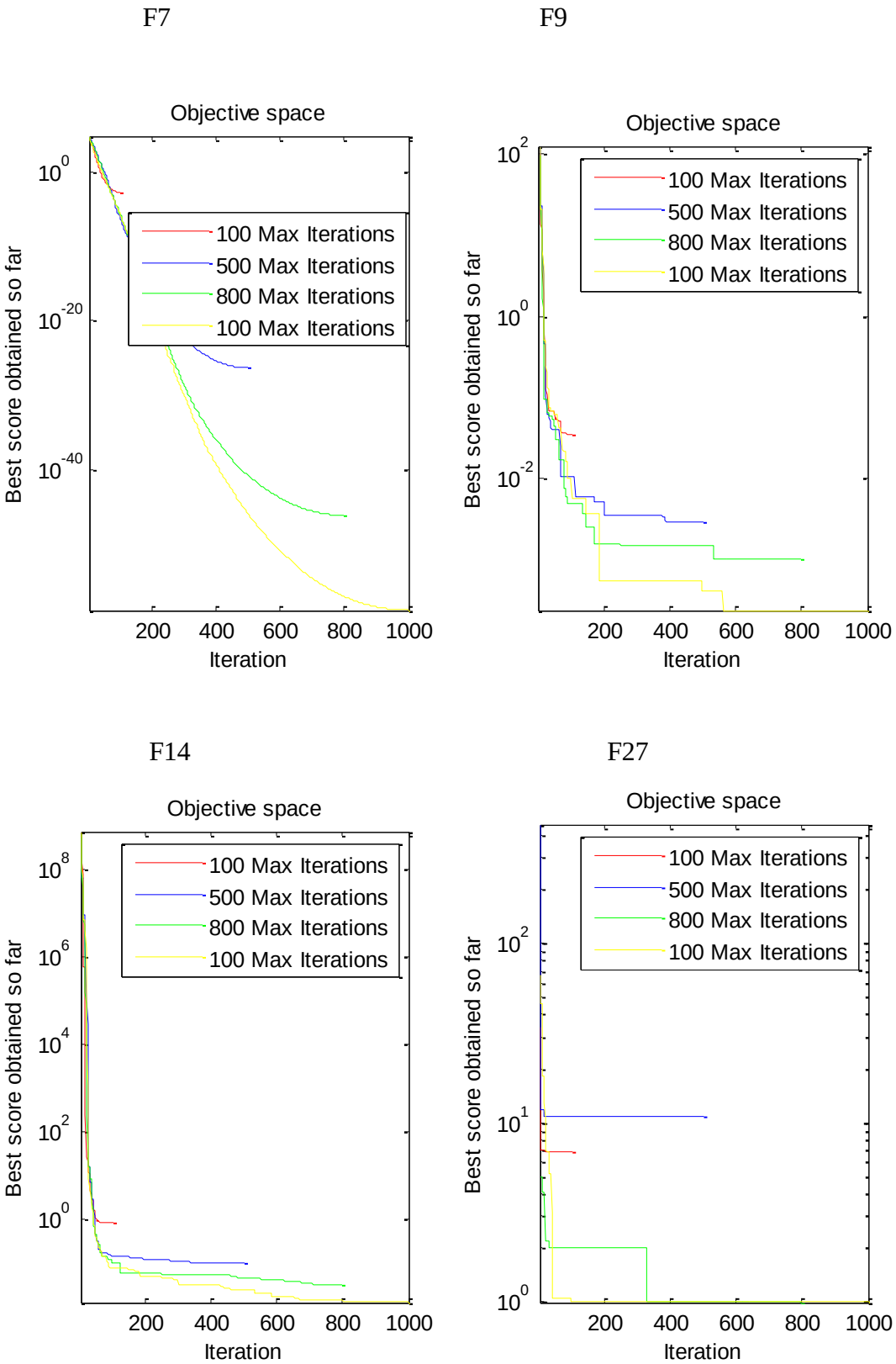
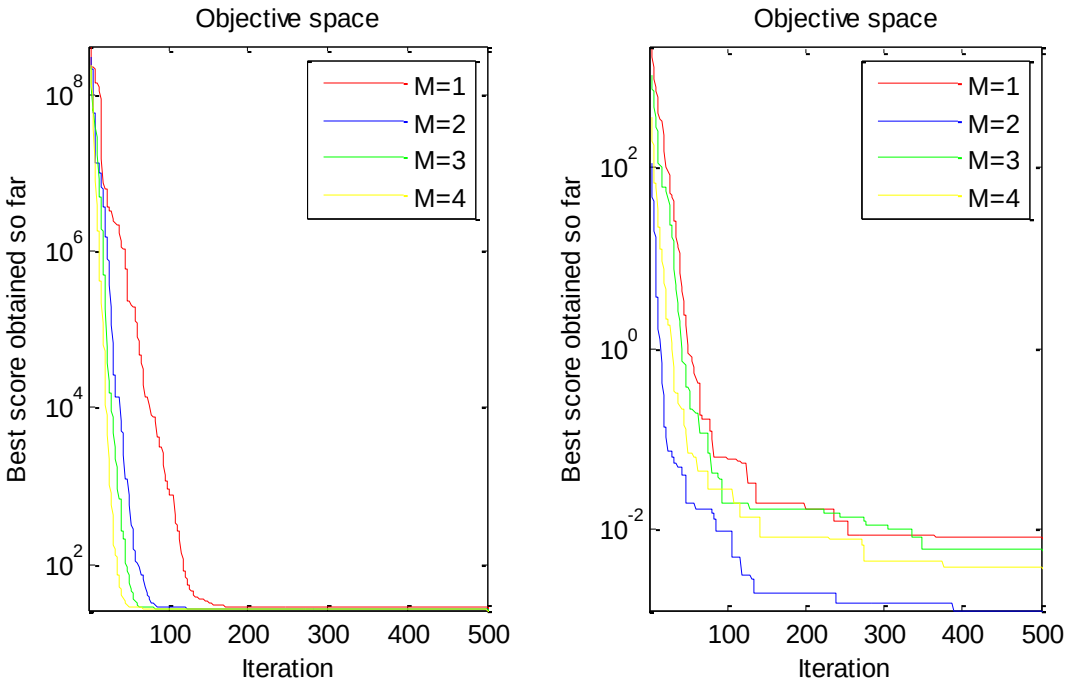


Figure 5.5: Effect of number of iterations on proposed BSHO

5.2.6.3 Parameter M

In the proposed algorithm, the value of M varies in the range [1, 2, 3, 4]. Figure 4 shows the convergence curves obtained. It is observed from the convergence curves that the BEPO performs better when the value of parameter M is 2.

F7 F8



F14 F27

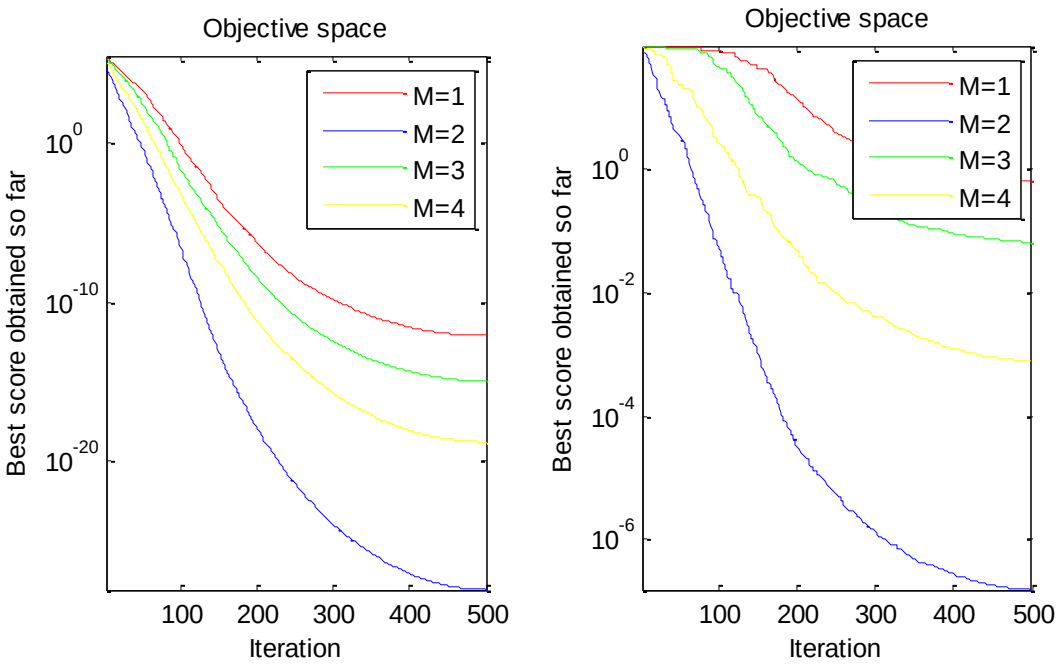


Figure 5.6: Effect of parameter M on proposed BEPO

5.2.6.4 Parameter f

The BEPO is verified on different test functions by varying the value of f in the range [1, 2], [2, 3], [3, 4] and [4, 5] by keeping rest of the variables same. Curves are shown in the fig 6. It has been observed from the convergence curves that the BEPO performs better when the f lies in the range [2, 3].

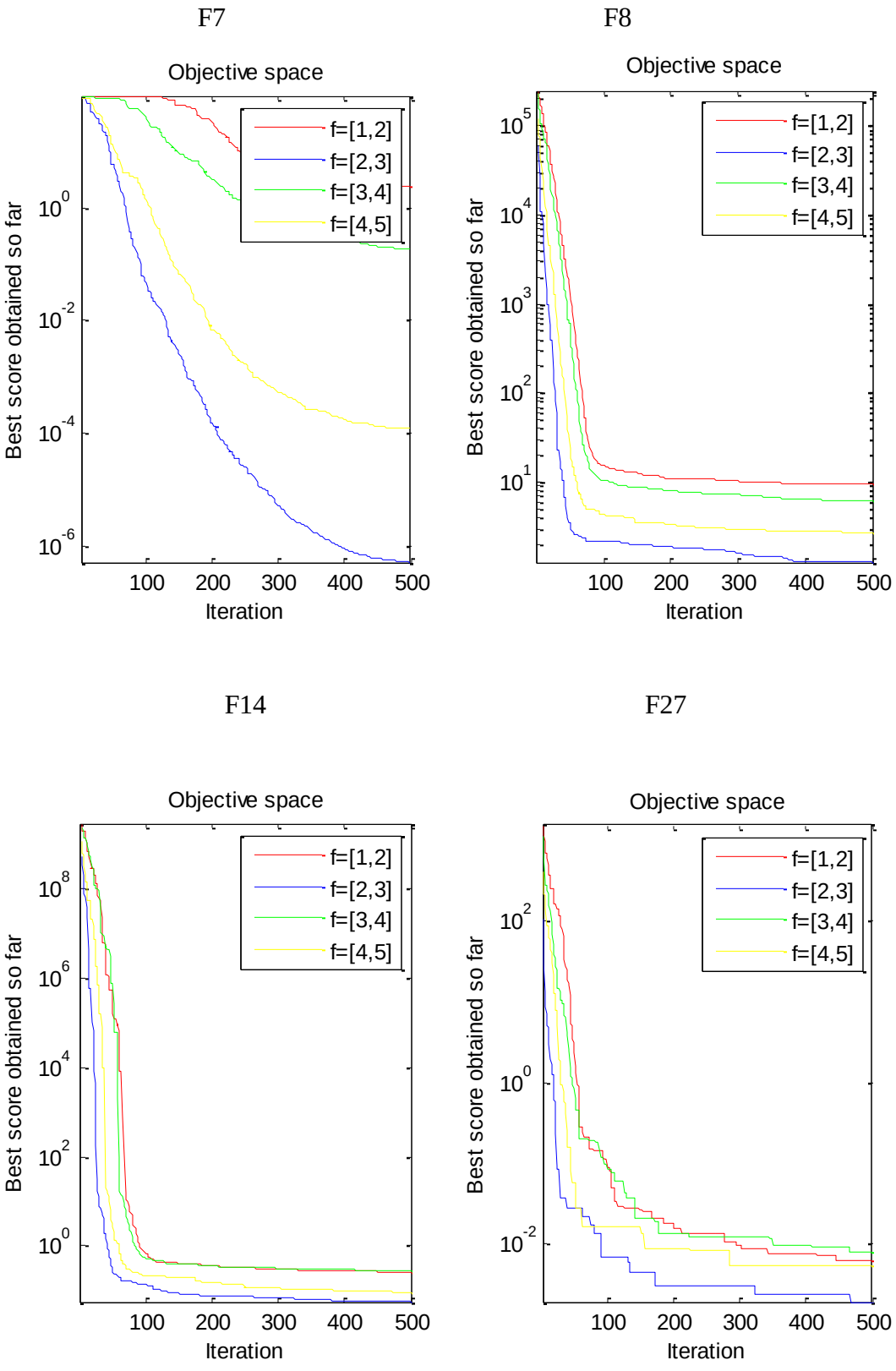
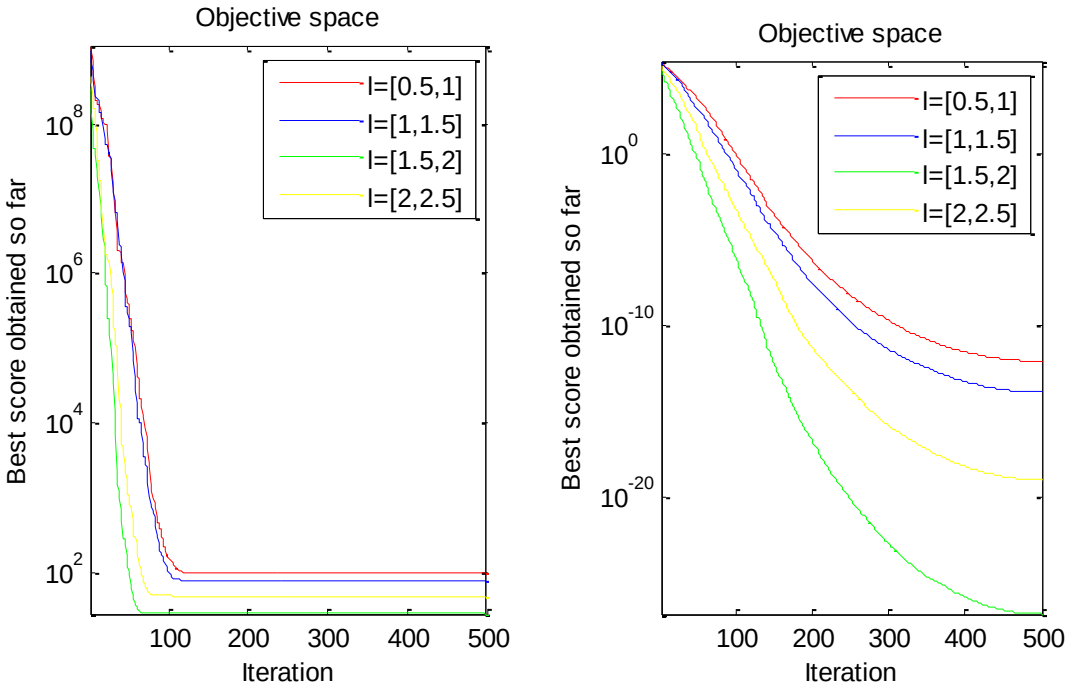


Figure 5.7: Effect of parameter f on BEPO

5.2.6.5 Parameter l

The BEPO is verified on different benchmark functions by varying the value of l in the range [0.5, 1], [1, 1.5], [1.5, 2] and [2, 2.5] by keeping rest of the variables same. The convergence curves are shown in the figure 7. It has been observed from the convergence curves that the BEPO performs better when the f lies in the range [1.5, 2].

F7 F8



F14 F27

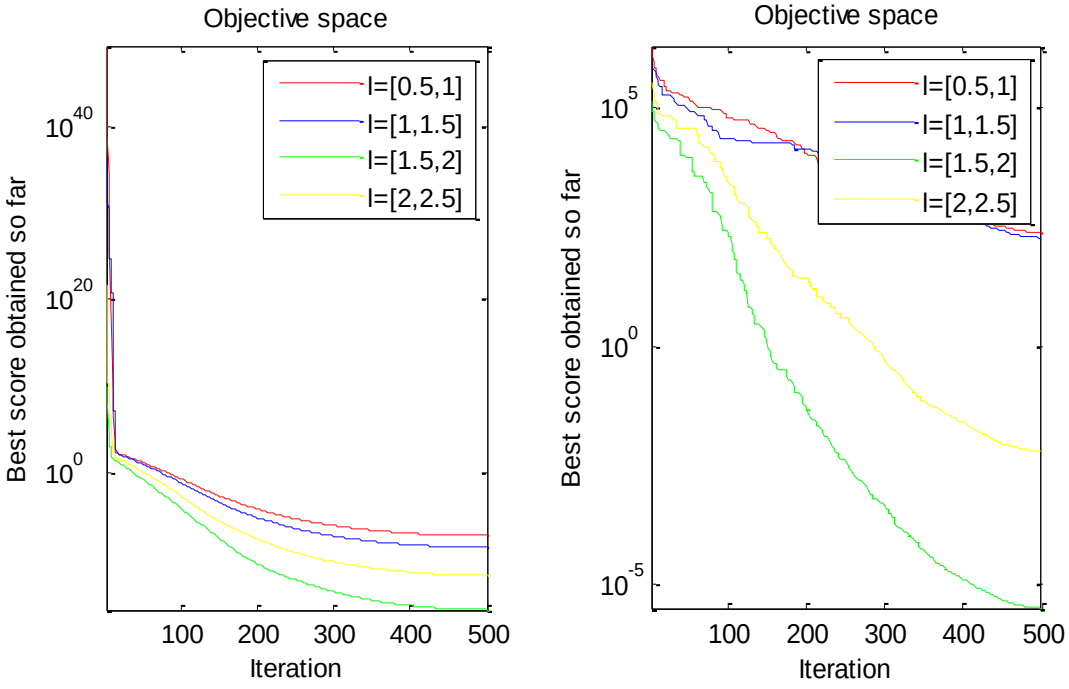


Figure 5.8: Effect of parameter l on proposed BEPO

5.3 Application on Feature Selection

The Binary emperor penguin algorithm is applied on feature selection problem. It is the way to distinguish relevant feature from the irrelevant ones and retrieving only the features which are important. The purpose of the feature selection is improving prediction performance, data dimensionality reduction.

In the proposed algorithm, KNN method is utilized. BEPO is verified on the datasets. BEPO is compared with four binary metaheuristic and results are given in table 5.7 and 5.8. Table 5.6 depicts the parameter settings of the algorithms

Table 5.6: Parameter settings for feature selection

Parameters	Values
Search Agents	30
Iterations	70
Runs	30
Search domain	[0,1]
Problem dimensions	Number of features in the data

5.3.1 Results and discussions

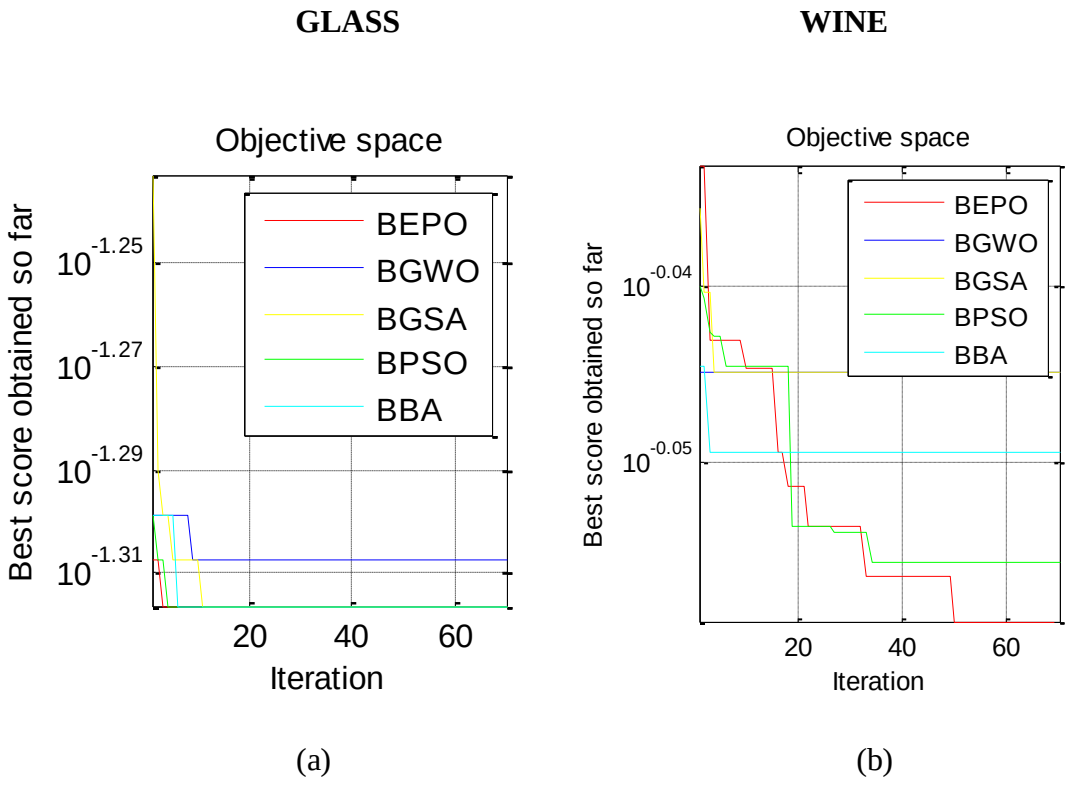
The standard deviation and average are listed below in table 5.7 and 5.8. The values which are marked as bold are better than the others which reveal that the BEPO is superior to other algorithms. Curves of the algorithms are shown in figure 5.9. The BEPO performs better in contrast to the others

Table 5.7: Standard deviation

Datasets	**BEPO**	**BGWO**	**BGSA**	**BPSO**	**BBA**
Wine	**0.14E-02**	1.11E-02	0.96E-02	0.31E-02	1.26E-02
Zoo	**1.43E-02**	2.11E-02	2.96E-02	1.98E-02	2.25E-02
Glass	**0.00E+00**	7.84E-04	4.79E-04	0.11E-01	8.50E-04
Haberman	**5.69E-17**	0.29E-02	0.43E-02	4.33E-16	0.85E-02
Lymphography	**1.14E-02**	5.51E-01	0.43E-01	3.31E-01	3.26E-01
Bupa	**4.43E-03**	0.11E-02	2.45E-02	1.08E-02	3.55E-02
Planning relax	**0.40E-05**	0.84E-04	4.44E-03	0.41E-03	1.40E-03
Cmc	**3.39E-06**	2.59E-02	5.43E-02	4.09E-04	2.56E-04
WDBC	**1.00E-02**	6.44E-01	3.56E-01	0.51E-01	0.50E-01
Sonar EW	**1.00E-12**	5.26E-07	4.57E-03	0.46E-03	6.67E-01
Penglung EW	**5.55E-04**	1.87E-03	0.53E-03	7.33E-02	4.65E-02

Table 5.8: Average

Datasets	BEPO	BGWO	BGSA	BPSO	BBA
Wine	**8.73E-01**	9.05E-01	8.82E-01	8.76E-01	9.00E-01
Zoo	**0.77E-01**	0.98E-01	0.79E-01	0.78E-01	1.37E-01
Glass	**2.98E-02**	3.08E-02	3.01E-02	2.95E-02	3.04E-02
Haberman	**2.59E-01**	2.60E-01	2.61E-01	2.63E-01	2.62E-01
Lymphography	**6.03E-02**	4.01E-01	0.22E-01	2.56E-01	7.11E-01
Bupa	**0.45E-03**	1.38E-02	4.59E-02	0.11E-02	7.54E-01
Planning relax	**2.41E-04**	4.51E-03	6.51E-03	6.65E-02	5.45E-03
Cmc	**2.44E-03**	1.60E-02	2.30E-02	0.11E-01	4.34E-02
WDBC	**6.33E-05**	6.75E-04	8.00E-03	0.99E-02	4.33E-02
Sonar EW	**1.44E-03**	2.38E-02	5.00E-02	5.91E-02	4.00E-01
Penglung EW	**6.90E-02**	1.05E-01	4.56E-01	5.65E-01	6.87E-01



(a) (b)

Figure 5.9: Convergence curves of BEPO and compared algorithms on datasets

5.4 Summary

In this chapter, the binary version of Emperor penguin optimizer has been proposed which works well for the discrete optimization problems. Unlike the continuous emperor penguin optimizer, the binary version can perform well. Binary emperor penguin optimizer is computed on 29 benchmark test functions. Results shows that BEPO is better than the other binary algorithms such as BGWO, BGSA, BPSO, and BBA. BEPO is also verified on the Feature selection and its performance is measured by conducting experiments on different datasets from UCI Machine Repository.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

For solving optimization problems, metaheuristics have been utilized. Metaheuristic techniques have been applied as they are effective for solving the complex problems in finite amount of time.

In this thesis, the binary metaheuristic algorithms have been studied. The original SHO and EPO have been studied and their binary versions have been proposed. The proposed binary algorithms are verified on 29 benchmark test functions. They are also compared with other binary metaheuristic algorithms. They have also been applied on the feature selection problem and tested on various datasets.

6.2 Future Scope

In the future, the thesis can be extended by applying the BSHO and BEPO to various optimization problems such as dimensionality reduction, electrical problems and unit commitment problems.

References

[1] Kumar V, Chhabra JK, Kumar D. Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems. J Comput Sci 2014a: 5(2): 144–55.

[2] Bonabeau E , Dorigo M , Theraulaz G . Swarm intelligence: from natural to artificial systems. Oxford University Press, Inc; 1999.

[3] Storn R, Price K. Differential evolution –a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 1997;11(4):341–59.

[4] Dorigo M , Birattari M , Stutzle T . Ant colony optimization - artificial ants as a computational intelligence technique. IEEE Comput Intell Mag 2006;1:28–39.

[5] Rashedi E, Nezamabadi-pour H, Saryazdi S. GSA: A gravitational search algorithm. Inf Sci 2009;179(13):2232–48.

[6] Kennedy J , Eberhart RC . Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks; 1995. p. 1942–8.

[7] C.Blum and A.Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys (CSUR), 35(3), 268-308, 2003.

[8] I.H.Osman and G.Laporte. Metaheuristics: A bibliography. Annals of Operations research, 63(5), 511-623, 1996.

[9] C.Blum and A.Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys (CSUR), 35(3), 268-308, 2003.

[10] Kirkpatrick S , Gelatt CD , Vecchi MP. Optimization by simulated annealing. Science 1983;220(4598):671–80.

[11] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Software 2014;69:4661. doi: 10.1016/j.advengsoft.2013.12.007.

[12] Gaurav Dhiman, Vijay Kumar. Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. Advances in Engineering Software 1144(2017) 48-70.

[13] E. Emary, H. M. Zawbaa, and A. E. Hassanien (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371-381.

[14] Yang X-S. A new metaheuristic bat-inspired algorithm. Springer Berlin Heidelberg.p.65–74.

[15] Lu X, Zhou Y. A novel global convergence algorithm: bee collecting pollen algorithm. In: 4th International Conference on Intelligent Computing. Springer; 2008. p. 518–25.

[16] Yang C, Tu X, Chen J. Algorithm of marriage in honey bees optimization based on the wolf pack search. In: International Conference on Intelligent Pervasive Computing; 2007. p. 462–7. doi: 10.1109/IPC.2007.104.

[17] Koza JR . Genetic programming: on the programming of computers by means of natural selection. MIT Press 1992.

[18] Beyer H-G, Schwefel H-P. Evolution strategies –a comprehensive introduction. Nat Comput 2002;1(1):3–52. doi: 10.1023/A:1015059928466.

[19] Simon D. Biogeography-based optimization. IEEE Trans Evol Comput 2008;12(6):702–13.

[20] Erol OK, Eksin I. A new optimization method: big bang-big crunch. Adv Eng Softw 2006;37(2):106–11.

[21] Du H, Wu X, Zhuang J. Small-world optimization algorithm for function optimization. Springer Berlin Heidelberg; 2006. p. 264–73.

[22] Hatamlou A. Black hole: a new heuristic optimization approach for data clustering. Inf Sci 2013;222:175–84.

[23] Formato RA. Central force optimization: a new deterministic gradient- like optimization metaheuristic. Opsearch 2009;46(1):25–51.

[24] Alatas B. Acroa: artificial chemical reaction optimization algorithm for global optimization. Expert Syst Appl 2011;38(10):13170–80.

[25] Shah Hosseini H. Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation. Int J Comput Sci Eng 2011;6:132–40.

[26] Gaurav Dhiman, Vijay Kumar. Emperor penguin optimizer. A bio-inspired algorithm for engineering problems. Knowledge-based systems.

[27] Seyedali Mirjalili, Andrew Lewis. The Whale Optimization Algorithm. Advances in Engineering Software 95 (2016) 51–67.

[28] Seyedali Mirjalili. Moth-Flame Optimization Algorithm: A Novel Nature-inspired Heuristic Paradigm. Knowledge-Based Systems 11-07-2015.

[29] E. Emary, H. M. Zawbaa, and A. E. Hassanien (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371-381.

[30] R.Y.M. Nakamura, L.A.M. Pereira, K.A. Costa, D. Rodrigues, J.P. Papa, and X.-S. Yang (2012) BBA: A binary bat algorithm for feature selection, In: IEEE Conference on Graphics, Patterns and Images, pp. 291–297.

[31] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi (2009) BGSA: binary gravitational search algorithm. Nat Comput 9:727-745.

[32] L.Y. Chuang, H.W. Chang, C.J. Tu, C.H. Yang (2008) Improved binary PSO for feature selection using gene expression data, Comput. Biol. Chem. 32:29–38.

[33] Majdi Mafarja, Abdelaziz I. Hammouri, Iyad Jaber and Derar Eleyan. Binary Dragonfly Algorithm for Feature Selection. Conference Paper · November 2017.

[34] B.Chizi,L.Rokach,O.Maimon,Asurveyoffeatureselectiontechniques, Encyclopedia of Data Warehousing and Mining, seconded., IGIGlobal,2009, pp. 1888–1895.

[35] Srinivasa KG, Venugopal KR, Patnaik LM (2007) A self-adaptive migration model genetic algorithm for data mining applications. Inf Sci 177(20):4295–4313.

[36] Yuan X, Nie H, Su A, Wang L, Yuan Y (2009) An improved binary particle swarm optimization for unit commitment problem. Expert Syst Appl 36(4):8049–8055.

[37] L.Y.Chuang, H.W.Chang ,C.J.Tu,C.H.Yang, Improved binary PSO for feature selection using gene expression data,Comput.Biol.Chem.32(2008)29–38.

[38] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010.

List of publications

Accepted:

1. Kaur, Avneet and Kaleka, Kamalinder Kaur and Kumar, Vijay, Design and Development of Soft Computing Toolbox Using Metaheuristics (March 14, 2019).
2. Vijay Kumar, Avneet Kaur. Binary spotted hyena optimizer and its application to feature selection. Journal of Ambient Intelligence and Humanized Computing

Communicated:

3. A Conceptual Comparison of Metaheuristic Algorithms and Applications to Engineering Design Problems, Kamalinder Kaur Kaleka, Avneet Kaur, Vijay Kumar International Journal of Intelligent Information and Database Systems.

Appendex

Table A.1 Unimodal benchmark test functions

Benchmark Test Function	Dim	Range	F_{min}
$F_1(Y)=\sum_{i=1}^{D} y_i^2$	5	[−100,100]	0
$F_2(Y)=\sum_{i=1}^{D} \lvert y_i \rvert + \prod_{i=1}^{D} \lvert y_i \rvert$	5	[−10,10]	0
$F_3(Y)=\sum_{i=1}^{D} \left(\sum_{j=1}^{i} y_j \right)^2$	5	[−100,100]	0
$F_4(Y)=\max_i \{ \lvert y_i \rvert, 1 \le i \le D \}$	5	[−100,100]	0
$F_5(Y)=\sum_{i=1}^{D-1} \left[100\left(y_{i+1}-y_i^2\right)^2 + \left(y_i-1\right)^2 \right]$	5	[−30,30]	0
$F_6(Y)=\sum_{i=1}^{D} \left(\lfloor y_i + 0.5 \rfloor \right)^2$	5	[−100,100]	0
$F_7(Y)=\sum_{i=1}^{D} i y_i^4 + random[0,1)$	5	[−1.28,1.28]	0

Table A.2 Multimodal benchmark test functions

Objective Function	Dim	Range	F_{min}
$F_8(Y)=-\frac{1}{D}\sum_{i=1}^{D}\left(y_i \sin\left(\sqrt{\lvert y_i \rvert}\right)\right)$	5	[−500,500]	-2094.914
$F_9(Y)=\sum_{i=1}^{D}\left[y_i^2 - 10\cos\left(2\pi y_i\right)+10\right]$	5	[−5.12,5.12]	0
$F_{10}(Y)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} y_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi y_i\right)\right)+20+e$	5	[−32,32]	0
$F_{11}(Y)=\frac{1}{4000}\sum_{i=1}^{D} y_i^2 - \prod_{i=1}^{D}\cos\left(\frac{y_i}{\sqrt{i}}\right)+1$	5	[−600,600]	0
$F_{12}(Y)=\frac{\pi}{D}\left\{10\sin\left(\pi z_1\right)+\sum_{i=1}^{D-1}\left(z_i-1\right)^2\left[1+10\sin^2\left(\pi z_{i+1}\right)\right]+\left(z_D-1\right)^2\right\}+\sum_{i=1}^{D} u\left(y_i,10,100,4\right)$ *where* $u\left(y_i,a,k,m\right)=\begin{cases} k\left(y_i-a\right)^m, & y_i > a, \\ 0, & -a \le y_i \le a, \\ k\left(-y_i-a\right)^m, & y_i < -a \end{cases}$ $z_i = 1+\frac{1}{4}\left(y_i+1\right)$	5	[−50,50]	0

$F_{13}(Y)=0.1\left\{\sin^2(3\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2\left[1+\sin^2(3\pi y_{i+1})\right]+(y_D-1)^2\left[1+\sin^2(2\pi y_D)\right]\right\}+\sum_{i=1}^{D}u(y_i,5,100,4)$	5	$[-30,30]$	0

Table A.3 Multimodal benchmark test functions with fixed dimension

Objective Function	Dim	Range	F_{min}
$F_{14}(Y)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[–65, 65]	1
$F_{15}(Y)=\sum_{i=1}^{11}\left[a_i-\frac{y_1(b_i^2+b_iy_2)}{b_i^2+b_iy_3+y_4}\right]^2$	4	[–5, 5]	0.00030
$F_{16}(Y)=4y_1^2-2.1y_1^4+\frac{1}{3}y_1^6+y_1y_2-4y_2^2+4y_2^4$	2	[–5, 5]	-1.0316
$F_{17}(Y)=\left(y_2-\frac{5.1}{4\pi^2}y_1^2+\frac{5}{\pi}y_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos y_1+10$	2	[–5, 5]	0.398
$F_{18}(Y)=\left[1+(y_1+y_2+1)^2(19-14y_1+6y_1y_2+3y_2^2)\right]\times\left[30+(2y_1-3y_2)^2\times(18-32y_1+12y_1^2+48y_2-36y_1y_2+27y_2^2)\right]$	2	[–2, 2]	3
$F_{19}(Y)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3	[1,3]	-3.86
$F_{20}(Y)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_{ij}(x_j-p_{ij})^2\right)$	6	[0,1]	-3.32
$F_{21}(Y)=-\sum_{i=1}^{5}\left[(Y-a_i)(Y-a_i)^T+c_i\right]^{-1}$	4	[0,10]	-10.1532
$F_{22}(Y)=-\sum_{i=1}^{7}\left[(Y-a_i)(Y-a_i)^T+c_i\right]^{-1}$	4	[0,10]	-10.4028
$F_{23}(Y)=-\sum_{i=1}^{10}\left[(Y-a_i)(Y-a_i)^T+c_i\right]^{-1}$	4	[0,10]	-10.5363

Table A.4 Composite benchmark test functions

Objective Function	Dim	Range	F_{min}
$F_{24}(CF1)$: $f_1, f_2, f_3, ..., f_{10} = Sphere\ Function$ $[\sigma_1, \sigma_2, \sigma_3, ..., \sigma_{10}] = [1,1,1, ..., 1]$ $[\lambda_1, \lambda_2, \lambda_3, ..., \lambda_{10}] = [5/100,\ 5/100,\ 5/100,\ ....., 5/100]$	10	[–5, 5]	0
$F_{25}(CF2)$: $f_1, f_2, f_3, ..., f_{10} = Griewank's\ Function$ $[\sigma_1, \sigma_2, \sigma_3, ..., \sigma_{10}] = [1,1,1, ..., 1]$ $[\lambda_1, \lambda_2, \lambda_3, ..., \lambda_{10}] = [5/100,\ 5/100,\ 5/100,\ ....., 5/100]$	10	[–5, 5]	0
$F_{26}(CF3)$: $f_1, f_2, f_3, ..., f_{10} = Griewank's\ Function$ $[\sigma_1, \sigma_2, \sigma_3, ..., \sigma_{10}] = [1,1,1, ..., 1]$ $[\lambda_1, \lambda_2, \lambda_3, ..., \lambda_{10}] = [1,1,1, ..., 1]$	10	[–5, 5]	0
$F_{27}(CF4)$: $f_1, f_2 = Ackley's\ Function, \quad f_3, f_4 = Rastrigin's\ Function$ $f_5, f_6 = Weierstrass\ Function, \quad f_7, f_8 = Griewank's\ Function$ $f_9, f_{10} = Sphere\ Function$ $[\sigma_1, \sigma_2, \sigma_3, ..., \sigma_{10}] = [1,1,1, ..., 1]$ $[\lambda_1, \lambda_2, \lambda_3, ..., \lambda_{10}] = [5/32, 5/32, 1, 1, 5/0.5, 5/0.5, 5/100, 5/100, 5/100,\ 5/100]$	10	[–5, 5]	0
$F_{28}(CF5)$: $f_1, f_2 = Rastrigin's\ Function, \quad f_3, f_4 = Weierstrass\ Function$ $f_5, f_6 = Griewank's\ Function, \quad f_7, f_8 = Ackley's\ Function$ $f_9, f_{10} = Sphere\ Function$ $[\sigma_1, \sigma_2, \sigma_3, ..., \sigma_{10}] = [1,1,1, ..., 1]$ $[\lambda_1, \lambda_2, \lambda_3, ..., \lambda_{10}] = [1/5, 1/5, 5/0.5, 5/0.5, 5/100, 5/100, 5/32, 5/32, 5/100,\ 5/100]$	10	[–5, 5]	0
$F_{29}(CF6)$: $f_1, f_2 = Rastrigin's\ Function, \quad f_3, f_4 = Weierstrass\ Function$ $f_5, f_6 = Griewank's\ Function, \quad f_7, f_8 = Ackley's\ Function$ $f_9, f_{10} = Sphere\ Function$ $[\sigma_1, \sigma_2, \sigma_3, ..., \sigma_{10}] = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]$ $[\lambda_1, \lambda_2, \lambda_3, ..., \lambda_{10}] = \begin{bmatrix} 0.1\times 1/5, 0.2\times 1/5, 0.3\times 5/0.5, 0.4\times 5/0.5, 0.5\times 5/100, \\ 0.6\times 5/100, 0.7\times 5/32, 0.8\times 5/32, 0.9\times 5/100,\ 1\times 5/100 \end{bmatrix}$	10	[–5, 5]	0

Avneet Thesis

ORIGINALITY REPORT

20%	12%	14%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	uvb.nrel.colostate.edu Internet Source	4%
2	Nitin Mittal, Urvinder Singh, Balwinder Singh Sohi. "Modified Grey Wolf Optimizer for Global Engineering Optimization", Applied Computational Intelligence and Soft Computing, 2016 Publication	1%
3	Narinder Singh, S. B. Singh. "Hybrid Algorithm of Particle Swarm Optimization and Grey Wolf Optimizer for Improving Convergence Performance", Journal of Applied Mathematics, 2017 Publication	1%
4	www.docstoc.com Internet Source	1%
5	goes.ngdc.noaa.gov Internet Source	1%
6	Gaurav Dhiman, Vijay Kumar. "Emperor Penguin Optimizer: A Bio-inspired Algorithm	<1%

for Engineering Problems", Knowledge-Based Systems, 2018
Publication

7	wise-obs.tau.ac.il Internet Source	<1%
8	ph.academicdirect.org Internet Source	<1%
9	Yuanyang Zou. "The whirlpool algorithm based on physical phenomenon for solving optimization problems", Engineering Computations, 2019 Publication	<1%
10	"Soft Computing for Problem Solving", Springer Science and Business Media LLC, 2019 Publication	<1%
11	"Harmony Search and Nature Inspired Optimization Algorithms", Springer Science and Business Media LLC, 2019 Publication	<1%
12	papers.ssrn.com Internet Source	<1%
13	www.pmmh.espci.fr Internet Source	<1%
14	A. Lajtai. "Energy spectrum measurements of neutrons for energies 30 keV - 4 meV from thermal fission of main fuel elements",	<1%

Radiation Effects and Defects in Solids, 3/1/1986
Publication

15	Gaurav Dhiman, Vijay Kumar. "Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications", Advances in Engineering Software, 2017 Publication	<1%
16	Jun He. "A mixed strategy of combining evolutionary algorithms with multigrid methods", International Journal of Computer Mathematics, 2008 Publication	<1%
17	202.141.81.85 Internet Source	<1%
18	"Decision Science in Action", Springer Nature America, Inc, 2019 Publication	<1%
19	Z. Mustaffa, M.H. Sulaiman, B. Yusob, F. Ernawan. "Integration of GWO-LSSVM for time series predictive analysis", 4th IET Clean Energy and Technology Conference (CEAT 2016), 2016 Publication	<1%
20	mcf.gsfc.nasa.gov Internet Source	<1%

21	jnusrv01.kek.jp Internet Source	<1%
22	link.springer.com Internet Source	<1%
23	Majdi Mafarja, Ibrahim Aljarah, Ali Asghar Heidari, Hossam Faris, Philippe Fournier-Viger, Xiaodong Li, Seyedali Mirjalili. "Binary dragonfly optimization for feature selection using time-varying transfer functions", Knowledge-Based Systems, 2018 Publication	<1%
24	Seema, Vijay Kumar. "Modified Grey Wolf Algorithm for optimization problems", 2016 International Conference on Inventive Computation Technologies (ICICT), 2016 Publication	<1%
25	Gaurav Dhiman, Vijay Kumar. "Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems", Knowledge-Based Systems, 2018 Publication	<1%
26	Prescilla, K., and A. Immanuel Selvakumar. "Modified Binary Particle Swarm optimization algorithm application to real-time task assignment in heterogeneous multiprocessor",	<1%

Microprocessors and Microsystems, 2013.
Publication

27	www.bolsinov.com Internet Source	<1%
28	hal.archives-ouvertes.fr Internet Source	<1%
29	Vijay Kumar, Dinesh Kumar. "Binary whale optimization algorithm and its application to unit commitment problem", Neural Computing and Applications, 2018 Publication	<1%
30	"Intelligent Computing Methodologies", Springer Science and Business Media LLC, 2018 Publication	<1%
31	www.mdpi.com Internet Source	<1%
32	Philip Vickerman, Janine Kim Coates. "Trainee and recently qualified physical education teachers' perspectives on including children with special educational needs", Physical Education & Sport Pedagogy, 2009 Publication	<1%
33	Ajay Kumar, Seema Bawa. "Generalized Ant Colony Optimizer: swarm-based meta-heuristic algorithm for cloud services execution",	<1%

Computing, 2018
Publication

---

34 www.oecdnea.org <1%
Internet Source

---

35 Laizhong Cui, Genghui Li, Zexuan Zhu, Zhong Ming, Zhenkun Wen, Nan Lu. "Differential evolution algorithm with dichotomy-based parameter space compression", Soft Computing, 2018 <1%
Publication

---

36 Long, Wen, Ximing Liang, Shaohong Cai, Jianjun Jiao, and Wenzhuan Zhang. "A modified augmented Lagrangian with improved grey wolf optimization to constrained optimization problems", Neural Computing and Applications, 2016. <1%
Publication

---

37 Li Wang, Jiguang Yue, Yongqing Su, Feng Lu, Qiang Sun. "A Novel Remaining Useful Life Prediction Approach for Superbuck Converter Circuits Based on Modified Grey Wolf Optimizer-Support Vector Regression", Energies, 2017 <1%
Publication

---

38 Juan Barraza, Luis Rodríguez, Oscar Castillo, Patricia Melin, Fevrier Valdez. "A New Hybridization Approach between the Fireworks <1%

Algorithm and Grey Wolf Optimizer Algorithm", Journal of Optimization, 2018
Publication

39	Lyuyang Tong, Minggang Dong, Chao Jing. "An improved multi-population ensemble differential evolution", Neurocomputing, 2018 Publication	<1%
40	Ramit Sawhney, Roopal Jain. "Modified Binary Dragonfly Algorithm for Feature Selection in Human Papillomavirus-Mediated Disease Treatment", 2018 International Conference on Communication, Computing and Internet of Things (IC3IoT), 2018 Publication	<1%
41	Lecture Notes in Computer Science, 2015. Publication	<1%
42	www.iiap.res.in Internet Source	<1%
43	Xiang Feng, Hanyu Xu, Huiqun Yu, Fei Luo. "Particle state change algorithm", Soft Computing, 2017 Publication	<1%
44	Studies in Computational Intelligence, 2014. Publication	<1%
45	Heming Jia, Jinduo Li, Wenlong Song, Xiaoxu Peng, Chunbo Lang, Yao Li. "Spotted Hyena	<1%

Optimization Algorithm with Simulated Annealing for Feature Selection", IEEE Access, 2019
Publication

---

46 Mira Komala, Ida Wahidah, Istikmal. "LTE networks BTS location optimation with double step grey wolf optimizer", 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA), 2017 — <1%
Publication

---

47 Roberto Irizarry. "LARES: An Artificial Chemical Process Approach for Optimization", Evolutionary Computation, 12/2004 — <1%
Publication

---

48 jpg.sbccom.army.mil — <1%
Internet Source

---

49 Weiguo Zhao, Liying Wang, Zhenxing Zhang. "Supply-demand-based Optimization: a Novel Economics-inspired Algorithm for Global Optimization", IEEE Access, 2019 — <1%
Publication

---

50 www.tandfonline.com — <1%
Internet Source

---

51 "Cuckoo Search and Firefly Algorithm", Springer Science and Business Media LLC, 2014 — <1%

Publication

52	Gaurav Dhiman, Amandeep Kaur. "STOA: A bio-inspired based optimization algorithm for industrial engineering problems", Engineering Applications of Artificial Intelligence, 2019 Publication	<1%
53	Shailendra S. Aote, M. M. Raghuwanshi, L. G. Malik. "Improved Particle Swarm Optimization Based on Natural Flocking Behavior", Arabian Journal for Science and Engineering, 2015 Publication	<1%
54	Qian Zhang, Huiling Chen, Jie Luo, Yueting Xu, Chengwen Wu, Chengye Li. "Chaos Enhanced Bacterial Foraging Optimization for Global Optimization", IEEE Access, 2018 Publication	<1%
55	E. Emary, Hossam M. Zawbaa, Crina Grosan. "Experienced Gray Wolf Optimization Through Reinforcement Learning and Neural Networks", IEEE Transactions on Neural Networks and Learning Systems, 2018 Publication	<1%
56	Gurmukh Singh, Munish Rattan, Sandeep Singh Gill, Nitin Mittal. "Hybridization of water wave optimization and sequential quadratic programming for cognitive radio system", Soft Computing, 2018	<1%

Publication

57	Jiehao Guo, Xingbao Gao, Mengnan Tian. "A Gravitation-Based Chaos Water Cycle Algorithm for Numerical Optimization", 2017 13th International Conference on Computational Intelligence and Security (CIS), 2017 Publication	<1%
58	R Greenwood. "Collective and two-quasiparticle states in 158Gd observed through study of radiative neutron capture in 157Gd", Nuclear Physics A, 1978 Publication	<1%
59	archive.epa.gov Internet Source	<1%
60	www.epa.gov Internet Source	<1%
61	mirlabs.org Internet Source	<1%
62	Seyedali Mirjalili, Seyed Mohammad Mirjalili, Andrew Lewis. "Grey Wolf Optimizer", Advances in Engineering Software, 2014 Publication	<1%
63	www.alrc.doe.gov Internet Source	<1%

Sathish Babu Pandu, Kamaraj Nagappan. "A

64	Novel Multiobjective Control of DVR to Enhance Power Quality of Sensitive Load", The Scientific World Journal, 2015 Publication	<1%
65	Yi, Jin, Liang Gao, Xinyu Li, and Jie Gao. "An efficient modified harmony search algorithm with intersect mutation operator and cellular local search for continuous function optimization problems", Applied Intelligence, 2015. Publication	<1%
66	Gaurav Dhiman, Amandeep Kaur. "Spotted Hyena Optimizer for Solving Engineering Design Problems", 2017 International Conference on Machine Learning and Data Science (MLDS), 2017 Publication	<1%
67	www.ii.pwr.wroc.pl Internet Source	<1%
68	repository.ntu.edu.sg Internet Source	<1%
69	Rizk M. Rizk-Allah. "An improved sine–cosine algorithm based on orthogonal parallel information for global optimization", Soft Computing, 2018 Publication	<1%

70	C. Wang, K. Z. Gao, J. Guo. "An improved gravitational search algorithm based on neighbor search", 2013 Ninth International Conference on Natural Computation (ICNC), 2013 Publication	<1%
71	Bo Xing, Wen-Jing Gao. "Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms", Springer Nature, 2014 Publication	<1%
72	Qingke Zhang, Weiguo Liu, Xiangxu Meng, Bo Yang, Athanasios V. Vasilakos. "Vector coevolving particle swarm optimization algorithm", Information Sciences, 2017 Publication	<1%
73	"Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016", Springer Science and Business Media LLC, 2017 Publication	<1%
74	Amandeep Kaur, Sushma Jain, Shivani Goel. "SP-J48: a novel optimization and machine-learning-based approach for solving complex problems: special application in software engineering for detecting code smells", Neural Computing and Applications, 2019 Publication	<1%

75	mdpi.com Internet Source	<1%
76	msa.lib.ohio-state.edu Internet Source	<1%
77	Devidas G. Jadhav, Shyam S. Pattnaik, Sanjoy Das. "Memetic Algorithm with Local Search as Modified Swine Influenza Model-Based Optimization and Its Use in ECG Filtering", Journal of Optimization, 2014 Publication	<1%
78	esdocs.com Internet Source	<1%
79	Patidar, K.C.. "@e-Uniformly convergent non-standard finite difference methods for singularly perturbed differential difference equations with small delay", Applied Mathematics and Computation, 20060401 Publication	<1%
80	S. Saikumar, M. S. Shunmugam. "Parameter Selection Based on Surface Finish in High-Speed End-Milling Using Differential Evolution", Materials and Manufacturing Processes, 2006 Publication	<1%
81	openaccess.city.ac.uk Internet Source	<1%

82	"Applications of Evolutionary Computation", Springer Science and Business Media LLC, 2013 Publication	<1%
83	Zhiming Li, Yongquan Zhou, Sen Zhang, Junmin Song. "Lévy-Flight Moth-Flame Algorithm for Function Optimization and Engineering Design Problems", Mathematical Problems in Engineering, 2016 Publication	<1%
84	Majdi Mafarja, Seyedali Mirjalili. "Whale optimization approaches for wrapper feature selection", Applied Soft Computing, 2018 Publication	<1%
85	Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Abdolreza Hatamlou. "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization", Neural Computing and Applications, 2015. Publication	<1%
86	Miguel Leon, Ning Xiong. "Adapting Differential Evolution Algorithms For Continuous Optimization Via Greedy Adjustment Of Control Parameters", Journal of Artificial Intelligence and Soft Computing Research, 2016 Publication	<1%

87	Mirjalili, Seyedali. "SCA: A Sine Cosine Algorithm for solving optimization problems", Knowledge-Based Systems, 2016. Publication	<1%
88	"Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2017", Springer Science and Business Media LLC, 2018 Publication	<1%
89	W.B. Wilson, T.R. England. "Delayed neutron study using ENDF/B-VI basic nuclear data", Progress in Nuclear Energy, 2002 Publication	<1%
90	oecd-nea.org Internet Source	<1%
91	Majdi Mafarja, Ibrahim Aljarah, Hossam Faris, Abdelaziz I. Hammouri, Ala' M. Al-Zoubi, Seyedali Mirjalili. "Binary Grasshopper Optimisation Algorithm Approaches for Feature Selection Problems", Expert Systems with Applications, 2018 Publication	<1%
92	"Massively Parallel Evolutionary Computation on GPGPUs", Springer Nature, 2013 Publication	<1%
93	Handbook of Combinatorial Optimization, 2013. Publication	

		<1%
94	tel.archives-ouvertes.fr Internet Source	<1%
95	www.cs.uoi.gr Internet Source	<1%
96	Amolkumar Narayan Jadhav, N. Gomathi. "WGC: Hybridization of exponential grey wolf optimizer with whale optimization for data clustering", Alexandria Engineering Journal, 2017 Publication	<1%
97	alliance-carton-nature.org Internet Source	<1%
98	stemcellcommons.org Internet Source	<1%
99	Rizk M. Rizk-Allah, Aboul Ella Hassanien, Mohamed Elhoseny, M. Gunasekaran. "A new binary salp swarm algorithm: development and application for optimization tasks", Neural Computing and Applications, 2018 Publication	<1%
100	Natural Computing Series, 2015. Publication	<1%
101	dblp.dagstuhl.de	

	Internet Source	<1%
102	www.cs.ubbcluj.ro Internet Source	<1%
103	es.scribd.com Internet Source	<1%
104	www.atlantis-press.com Internet Source	<1%
105	"Advances in Swarm Intelligence", Springer Science and Business Media LLC, 2017 Publication	<1%
106	150.214.191.180 Internet Source	<1%
107	Shail Kumar Dinkar, Kusum Deep. "Opposition based Laplacian Ant Lion Optimizer", Journal of Computational Science, 2017 Publication	<1%
108	"The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018)", Springer Science and Business Media LLC, 2018 Publication	<1%
109	Wei-zhen Sun, Jie-sheng Wang, Xian Wei. "An Improved Whale Optimization Algorithm Based on Different Searching Paths and Perceptual	<1%

Disturbance", Symmetry, 2018
Publication

110	journals.sagepub.com Internet Source	<1%
111	"Soft Computing in Data Analytics", Springer Nature America, Inc, 2019 Publication	<1%
112	insightsociety.org Internet Source	<1%
113	www.coursehero.com Internet Source	<1%
114	iridia.ulb.ac.be Internet Source	<1%

Exclude quotes	On	Exclude matches	< 8 words
Exclude bibliography	On		