Global Swarm Optimization Algorithms with hybrid search strategies

By

Po-Chun CHANG

Principle Supervisor

Longbing Cao

Submitted in partial fulfilment for the degree of

Master of Analytics by Research

In

The University of Technology, Sydney

2014

ABSTRACT

In decades, global optimization algorithm with nature-inspired technique has become an important research topic. The aim is to find the optimal solutions for given problems without knowing the characteristics of solutions beforehand. In particular, Swarm Intelligence is a population-based meta-heuristic methodology belonging to Soft Computing. The collective behaviour of swarm members is often inspired by the biological system and behaviours of nature. For instance, Particle Swarm Optimization is inspired from bird flocking. Evolutionary algorithms such as Genetic Algorithm and Differential Evolution are inspired from biological evolution. These algorithms try to iteratively improve the discovered solutions by employing specially designed formulae to synthesize new solution candidates. However, sometimes algorithms present low performance in some problems. Possible reasons could be an algorithm itself is not specialized for particular types of problems; an algorithm is with the inappropriate selection of control parameters, or an inappropriate way to perform evaluation.

To address the above issues, this research is to design swarm optimization algorithms to operate in the black-box scenario where objective functions are the only direct source of information. Different optimization methods are specialized for solving different types of problems, but they may not achieve good results in other problem classes. Hybridization of different algorithms and incorporating their knowledge may combine the strength of different optimization approaches and cancel out their weaknesses. Therefore, the two swarm optimization algorithms are developed with this manner. The optimization performance is verified by public benchmark mathematical functions.

The proposed methods in the thesis are: 1) Simplified Swarm Optimization with Differential Evaluation mutation strategy (SSODE) and 2) Macroscopic Indeterminacy Swarm Optimization (MISO). SSODE is an experimental method which is developed to verify the proposed hybrid principle in the thesis. SSODE hybridizes Simplified Swarm Optimization (SSO) algorithm structure with multiple mutation strategies from DE. The experiment results of SSODE indicate that the hybridization of different algorithms and mutation strategies is able to achieve general efficiency. By continuing the research of SSODE, MISO presents a well-structured memetic algorithm with new evaluation

schema. Substantial experiments have shown that the performance of MISO is significantly superior to many well-known algorithms in many objective functions.

ACKNOWLEDGE

First of all, I would like to express my gratitude to my previous principle supervisor Assoc. Prof. Wei-Chang YEH. He has played the most important role in my two years research journey. I truly appreciate for his stimulating suggestions in the area of research on soft computing and swarm intelligence systems.

I am thankful to my current principle supervisor Prof. Longbing Cao. Although he supervised me less than three months, he gave me useful advices for finalizing this thesis. Besides, during my research journey in my master degree, I have attended many discussion groups and seminars which arranged by him and his students. Their creative thinking and interesting ideas exert a subtle influence on me that I can probe into the the subject from different point of views.

I would like to thank my parents and my elder sister for their never ending encouragement and support. Therefore, I can fully concentrate on my study.

CERTIFICATE OF ORIGINAL AUTHORSHIP

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Student:

Date:

PUBLICATIONS ARISING FROM THIS THESIS

Chang, P.-C. and W.-C. Yeh (2013). Simplified swarm optimization with differential evolution mutation strategy for parameter search. Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication, ACM.

Wei-Chang Yeh, Yun-Chih Ke, Po-Chun Chang, Yuan-Ming Yeh, and Vera Chung, "Forecasting Wind Power in the Mai Liao Wind Farm based on the Multi-Layer Perceptron Artificial Neural Network Model with Improved Simplified Swarm Optimization", International Journal of Electrical Power & Energy Systems (JEPE2518), DOI: 10.1016/j.ijepes.2013.10.001, 2012/10/08 (SCI).

Po-Chun Chang and Xiangjian He (2014). Macroscopic Indeterminacy Swarm Optimization (MISO) for Real-Parameter Search, 2014 IEEE World Congress on Computational Intelligence

Chun-Hua Chou, Chia-Ling, and Po-Chun Chang (2014). A RFID Network Design Methodology for Decision Problem in Health Care, 2014 IEEE World Congress on Computational Intelligence

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

"*Life is problems. Living is solving problems*"(Baker Jr and Baker Jr, 2001). Dealing with problem solving and decision-making are part of human life. Even if people remain silence, no comment, or go with random guessing, these are the answers they provided. There are no wrong answers, the question is whether these solutions can be efficiently concluded and effectively solves the objective problems.

1.1 Background

Many real world problems can be analysed and formularized into problem models known as objective functions (sometimes called fitness functions) with a set of constraints. Optimization algorithm is a process for selecting better solutions over poor solutions. In general, optimization is to discover the best parameters which can maximize or minimize the outputs of objective functions. There are two common types of optimization problems: function optimization and combinatorial optimization problems. Function optimization problem is concerned about finding the best values, either discrete number or continuous number, within predefined range. For instance, researchers want to discover the best value or vector to achieve minimum/maximum result of a concave/convex function. Combinatorial optimization problem is concerned about the best combination or ordering of the given variables. Some practical applications are finding the shortest routes, time scheduling, etc. Now a day, optimization problems can be handled by computer.

Swarm intelligence is a population based meta-heuristic computational intelligence based on soft computing (SC). Unlike the conventional hard computing (HC) looking for the precise solution, SC is trying to discover the approximate and imprecise optimal solution(s)(Karaboga, 2011). The idea of HC is to design exact methods for the target problems. By giving the same inputs, HC algorithm is always returning the same

outputs. In contrast, SC is somewhat like human mind that tolerance for imprecision, uncertainty, partial truth, and approximation. Instead of focusing on one optimization agent, SI is an artificial intelligence emerged from the population, which is a set of agents. Similar to human societies, people may make a conclusion based on intuition which influenced by experience, other people's opinions, lessons of history, etc. How these kinds of knowledge accomplish the final decision is hard to articulate (probably, go with the answers they "feel" right or correct). Stochastic method, random variables appeared in formulae, illustrates human minds of making decisions or random walk behaviour of living creatures.

As its name implies, Swarm intelligence Optimization in computer science refers to the concept of swarm, which is a group or groups of simple agents. Many powerful and efficient algorithms have been proposed. Some of these algorithms are Differential Evolution (DE) by Storn and Price(Storn and Price, 1995), Particle Swarm Optimization (PSO) algorithm by Eberhart and Kennedy (Shi and Eberhart, 1999, Poli et al., 2007), Simplified Swarm Optimization (SSO) by Yeh (Yeh et al., 2009, Yeh, 2009, Wei-Chang, 2012, Yeh, 2013), and Artificial Bee Colony (ABC) by Karaboga and Basturk (Karaboga and Basturk, 2007). The simple agents are called differently in different algorithms, such as particles in PSO, chromosomes in genetic algorithm (GA), bees in ABC, parameter vectors or solution candidates in DE. This is because the inspirations often come from nature behaviours or phenomena. Based on the predefined rules, individual agents iteratively generate new candidate solutions by interacting with other agents. The interaction between agents leads to the emergence of intelligence of the entire system.

Maintaining a certain level of diversity between agents is important. Exploration is the process that finding the points far from the current ones. High diversity between agents provides strong global optimization ability, which increases the possibility to avoid the areas that have been investigated. Nevertheless, it may cause the optimization process never improve the solution candidates. In contrast, exploitation process, also known as

local search, synthesis new solution candidates by performing small changes. This provides the steady improvement of current solution candidates. However, losing diversity is almost equal to losing search ability in solution space. There may be better solutions in distant areas from the current positions. For solving the optimization problem, individual may not be able to discover the global best solution, but the system. The knowledge discovered by individuals contributes to the overall output.

1.2 Research Objectives and Contributions

There is no one optimization algorithm which can efficiently and successfully solve problems in any cases. According to No Free Lunch Theorem (NFL), if an algorithm achieves satisfied results in some problem classes, these good performances have to be paid for with inferior solution qualities in different situations (Wolpert and Macready, 1997). NFL implies that the performances of algorithms can be compared while giving a limited number of objective functions. Optimization practitioners often refer to public benchmark functions and toy problems for demonstrating the utility of their algorithms, in order to achieve credible results. By comparing the performances of optimizers with one another, it is able to find the weaknesses of optimization algorithms for the particular problems. Using minimal numbers of optimizers to solve maximal kinds of problems would be one research focus.

1.2.1 *Review of existing population-based methods*

At the very beginning, this thesis presents the comprehensive review of some existing SI based methods and summarizes the comment features of different optimization strategies. Different algorithms achieve superior results in different problem domains.

1.2.2 *Hybrid different search strategies increases generalization ability*

By integrating different search operation techniques incorporate their knowledge, the strengths of different methods can be combined and compensate for one another's weaknesses. Therefore, the first contribution is to successively present two new hybrid swarm optimization algorithms. Each proposed algorithm introduces a new principle to concurrently use different optimization strategies.

1.2.3 *The effect of performing evaluation process*

Moreover, another concept that less concerned about in SI optimization is evaluation phase. Many researches are focused on implementing the systems that can efficiently discover better solutions. Thinking of it differently, whether the quality of solution is better or not is dependent on the result of evaluation process. It does not matter what principles and strategies been used in an algorithm, so long as its evaluation outcomes are often pleasing the system. Consequently, the second contribution introduces a new evaluation schema. Assume an exploitation mechanism samples a new point from the current one, it performs local search and generates a similar solution candidate. While recursively executing the mechanism a number of times, the random region will be extended and will be able to synthesis new point in distant areas. By manipulating the moment of applying evaluation, it will influence the balance between exploration and exploitation.

1.3 Research issues

- Black-box optimization – solving problems without knowing any background knowledge.
- Swarm Intelligence – optimization methods are often inspired from nature phenomena, but should not stick on examples from nature.

- Global optimization algorithm – the optimization techniques which can efficiently and effectively solve a wide range of problems are often interested by stakeholders.
- Efficiency does not always mean effectiveness – a good trade-off between computational complexity and performance.

1.4 Significance of the research issues

Many real-world problems cannot be optimized analytically and can only be evaluated by objective functions. Another main issue is the curse of dimensionality (Li et al., 2013). The solution space of a problem increases exponentially with the size of the problem domain. Besides, for the same objective function with different size of decision variables should be considered as different optimization problems. One classical example is Rosenbrock function. Despite of Rosenbrock function is a unimodal function in 2D, it becomes multimodal when the dimension is higher than 3 (Shang and Qiu, 2006). Thus, the distributions of outcomes, also known as the problem landscapes, are varying due to not only different objective functions, but also the size of decision variables. It is often time consuming for well-analysing the specific problem before implement the precise model with hard computing (HC). Beside, many real-world problems are typically ill-defined systems with imperfect information, which is difficult to model and with large-scale solution spaces. Moreover, for solving complex problem, although HC technique is often stable, it may take a lot of computation time. Instead of specifically design optimization approaches for particular types of problems, it is practical to design a general-purpose algorithm which is able to handle multiple kinds of problems.

Swarm intelligence (SI) is a research branch in SC and it has become a popular approach to solve optimization problems. The concept of a swarm refers to swarm behaviour in the real world such as bird flocks, gene pool, or quantum mechanical phenomena. Intelligence in machine learning usually refers to the method which can

solve the problem somehow successfully. In decades, researchers have been looking for examples in nature and seeking for inspiration. For example, Particle Swarm Optimization (PSO) is inspired from bird flocking. Ant Colony Optimization (ACO) is inspired from ant colony. Genetic Algorithm (GA) and Differential Evolution (DE) are evolutionary algorithms inspired from biological evolution. Nevertheless, these biological behaviours and phenomena are only some parts of the whole universe in nature. Swarm optimization research should not cease to mimic particular examples in nature, but comprehend their knowledge and techniques to design an algorithm for the given problems.

Due to the fact that computer technologies are advancing with each passing day, the complexity and dimension of problem domain increase. To recap the curse of dimensionality issues, the previous successful approaches may no longer be capable for new problems. Besides, objective functions can be changed due to different problems or different suppositions for building computational models. A method which specifically designed for a particular problem may not achieve satisfied results for other ones. For these reasons, stakeholders are often interested in the method which is not only achieving maximum output with the minimum resource usage (i.e. time, memory), but also achieve high generalization ability. In SI system, the running time and required memory space, which an algorithm needs, grow with increasing the size of swarm (the number of search agents or optimizers). Although a good algorithm should keep the processing time and the memory space as small as possible, the quality of output results is always the key concern in optimization issues. Small swarm size may result in local convergence, whereas, large size will increase computational efforts. The trade-off between computational complexity and the quality of discovered solutions is always the research topic.

1.5 Research methodology and Justification (Approach)

By reviewing many scientific literatures on various types of optimization algorithm, they have the certain characteristics as follow: 1) the population structure or topology; 2) the trade-off between population size and iteration; and 3) updating strategy or equation for improving candidate solutions. As the thesis is designed to propose new meta-heuristic optimization algorithms, the study is carried in these three focuses with new ideas and inspirations. The second focus in this thesis is to verify and compare the algorithms performances with other existing algorithms. The public benchmark functions and toy problems are used for evaluating algorithms in a systematic and creditable manner.

1.5.1 *Benchmarks and toy problems*

Benchmark and toy problems are used to demonstrate the performance of optimization algorithms. In spite of these problems may not relevant to the real-world problems, they are widely used for testing optimization algorithms. Many practitioners use them for measuring and comparing the utility of algorithms, verifying theories, examining hypotheses, and understanding the concept of optimization problems(Weise, 2009).

Mathematical benchmark functions are designed for measuring and comparing the algorithm based on real number vectors with n elements $V \in R^n$. Such vectors usually refer to parameters or candidate solutions. The optima of the mathematical functions are predefined which is similar to the answers to riddles. While applying the optimization algorithms to the functions, they are trying to find the solutions as close as possible to the predefined answers. Mathematical optimization functions can be categorized into two types: Single-Objective (SO) and Multi-Objective (MO). SO optimization is to find the one and only one optimum for a single objective function, whereas, MO optimization is to find a set of compromised, trade-off, or Pareto-optimal solutions. For different objective functions or in different dimensional space, the distributions of

fitness landscape are varying. Functions could be unimodal, multimodal, or even lack of useful structure. IEEE Congress of Evolutionary Computation (CEC) has released the mathematical benchmark test suite for researchers to download (Suganthan et al., 2005, Tang et al., 2007, Liang et al., 2013).

1.6 Limitation of the study

There is always a risk that the proposed strategy cannot result satisfactory outcome. As the thesis is planning to develop new optimization algorithms, the main bottlenecks might be the new algorithms cannot outperform the existing well-known algorithms in:

- Overall performance on average
- The quality of discovered solutions
- Computation complexity
- Generalization ability

Soft computing based optimization algorithm is always tolerant of imprecision, uncertainty, partial truth and approximation. Thus, it can efficiently locate the approximately optimal result, but may take long times and almost impossible to approach the global best optima. The balance between efficiency and effectiveness needs to be justified properly. At the end, there is no point to implement a system that cannot provide any improvement; the proposed approach needs to be beneficial in some concepts

1.7 Thesis organization

In the thesis, the detail literature review of optimization problem and comprehensive summary of some representative swarm optimization algorithms are discussed in chapter 2. There are two hybrid swarm optimization algorithms are successively proposed. In Chapter 3, the preliminary experimental hybrid algorithm: Simplified

Swarm Optimization with Differential Evolution mutation strategy (SSODE) is proposed. The experiment designs and results for SSODE are shown in subsection 3.4. Chapter 4 presents the improved version of SSODE with new evaluation scheme called Macroscopic Indeterminacy Swarm Optimization (MISO). The benchmark test for MISO is shown in subsection 4.4. The conclusion and final remarks are given in Chapter 5. Moreover, all the mathematical benchmark functions provided by Congress on Evolutionary Computation conference are referenced in Appendix A. The additional experiment results for SSODE and MISO are listed in Appendix B, Appendix C, and Appendix D.

2 LITERATURE REVIEW

2.1 Optimization problem

Global optimization algorithms are methods to discover the optimal solutions from all feasible solutions in problem domain. With the purpose of decision analysis or optimization studies, real-world problems are expressed into mathematical functions called objective functions $\boldsymbol{F(x)}$ (Boyd and Vandenberghe, 2004, Burges, 1998, Weise, 2009).

Single-objective (SO) function: $F = \{f: X \to Y \subseteq \mathbb{R}\}$

Multi-objective (MO) function: $F = \{f_i: X \to Y_i : 1 < i \leq n, Y_i \subseteq \mathbb{R}\}$

$$subject\ to: \begin{cases} g_i(X) \leq 0, & i = 1, \dots, m \\ h_i(X) = 0, & i = 1, \dots, p \\ X^L \leq X \leq X^U \end{cases} \quad (1)$$

where X is a set of all possible solutions in the solution space that satisfy the given constraints. Y is a set of the possible outputs of function F. $g_i(X)$ are the inequality constraints and $h_i(X)$ are the equality constraints. The role of optimization algorithm is to efficiently select better solutions from X. X^L and X^U indicate the lower and upper bounds for X. Discovering the best solutions for an objective function equates to finding the approximately best solutions for a real world problem.

The aim of optimization is to discover the best solution(s) to the target problem. Assuming the $F(x^*)$ is the target result that the system wants to achieve, the optimization problem would be to discover the best solution x from all possible solutions X. The goal is to achieve $F(x^*) \cong F(x)$.

$$
\begin{aligned}
&x \in X \\
&\min eval(x) \\
&where\ \ error = eval(x) = F(x) - F(x^*)
\end{aligned}
\tag{2}
$$

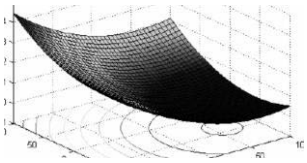
Due to the solution space of optimization problems being in either continuous or discrete, they can be categorized into two types: Continuous Optimization Problem (COP) and Discrete Optimization Problem (DOP).

2.1.1 *Continuous Optimization Problem (COP)*

The continuous search space of candidate solution means that there are infinite many possible solutions that satisfy the given constraints. The candidate solutions for an objective function are subset of real numbers $X \subseteq R^D$ in D-dimensional space. The objective function $F(x),\ x \in X$ is continuous means if a point x^*(global optimum) in space$\lim_{x \to x^*} F(x)\ exist$. Thus, $\lim_{x \to (x^*)^+} F(x) = \lim_{x \to (x^*)^-} F(x)$, then objective is to discover x where $\lim_{x \to x^*} F(x) = F(x^*)$. In other words, while x is close to x^*, the output of function $F(x)$ should be close to the output $F(x)$. One approach for solving COP is to perform real-parameters optimization (Liang et al., 2013). According to the design guideline of benchmark functions (Liang et al., 2013), a benchmark function $f(x)$ is mathematical formula which simulates a complex search space. As the dimensionality of the problem domain increases, finding the optimal solutions becomes more challenging. One approach for solving a large scale problem is to break it into sub-components and solve them individually. However, many problems in real-world are not fully separable. Even more, distribution of possible solutions in search space are varying according to objective functions. A function could be uni-modal with a single extremum, or multi-modal with a number of local extrema. Examples are shown in below:

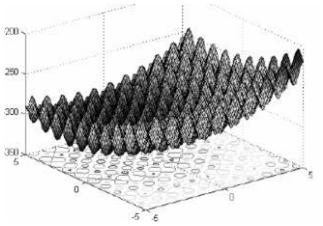
- **Separable Function examples**

Sphere Function (Uni-modal)



$$f(x) = \sum_{i=1}^{D} x_i^2 \tag{3}$$

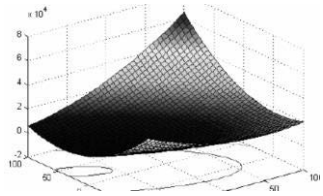
Rastrigin's Function (Multi-modal)



$$f(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10) \tag{4}$$

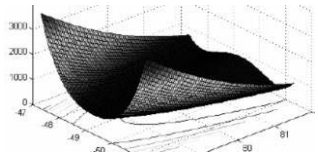
- **Non-separable Function examples**

Schwefel's Problem 1.2 (Uni-modal)



$$f(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{5}$$

Rosenbrock's function (Multi-modal)



$$f(x) = \sum_{i=1}^{D-1} (100 \times (x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \tag{6}$$

2.1.2 *Discrete Optimization Problem (DOP)*

In comparison, the search space of DOP is discontinuous that the number of possible solutions is finite. DOP often refers to Combinatorial Optimization Problems (ComOP), such as assignment problems(Maniezzo and Colorni, 1999), schedule problems(Blum and Sampels, 2004), and routing problems(Reimann et al., 2004). The formal definition of ComOP (Papadimitriou and Steiglitz, 1998) is shown as below (7):

$$
\begin{aligned}
&P = (S, \Omega, f) \\
&\Omega\text{: set of constraints among variables} \\
&X = \{X_i\}, \qquad i = 1, \dots, n \\
&v_i^j \in D_i = \left\{v_i^1, \dots, v_i^{|D_i|}\right\} \\
&S = \left\{s = \left\{\left(X_1, v_1^j\right), \dots, \left(X_n, v_n^j\right)\right\}\right\}, \qquad s \in S \\
&f\text{: objective function } f(s) \epsilon R^+ \\
&s^*\text{: global optimum where } f(s^*) \leq f(s)\ \forall\ \mathrm{s}\ \in\ \mathrm{S}
\end{aligned}
\tag{7}
$$

There is a search space $\boldsymbol{S}$ which contains all combinations for a finite set of distinct nodes $\boldsymbol{X}$. Each node $\boldsymbol{X_i}$ is combined with another node $\boldsymbol{v_i^j}$ from a particular set A_i into a solution component $\left(\boldsymbol{X_1}, \boldsymbol{v_1^j}\right)$ according the predefined constraints and rules $\boldsymbol{\Omega}$. If there are no constraints $\boldsymbol{\Omega}$ applied, then each discrete decision variable can choose any value $\boldsymbol{v_i^j}$ from domain $\boldsymbol{D_i}$. However, in the real world practices, constraints are varying depended on different type of COPs. The goal is to discover the best solution $\mathbf{s}^*$ from all possible solutions $\boldsymbol{s} \in \boldsymbol{S}$. One of the famous COPs is Travelling Salesman Problem (TSP) (Reinelt, 1994). Giving a set of n nodes and path costs for each pair of nodes, the goal is to find the optimal trip (with the lowest total path cost) that visits every node exactly once. TSP problem can be categorized into two types: Symmetric and Asymmetric TSP. Assuming V is a set of n cities $V = \{v_1. \dots, v_n\}$, A is the edge set

between city r and s overall cities $A = \{(r,s): r,s \in V\}$. The path cost from city r to city s is d_{rs}, whereas from city s to city r is d_{sr}. Symmetric TSP (STSP) means the path costs for any two nodes are the same for both directions $d_{rs} = d_{sr}$. Thus, there are $\frac{(n-1)!}{2}$ possible routes for TSP. In contrast, Asymmetric TSP (ATSP) means for the different directions between any two nodes are assigned with different path costs $d_{rs} \neq d_{sr}$. Therefore, there are $(n-1)!$ possible solutions in search space.

2.1.3 *No Free Lunch Theorem*

In fact, it is most likely no optimization algorithm can be efficient and effective on all problems. There are specialized algorithms which implemented for specific types of problems. Likewise, there are also general methods which are outperformed by highly specialized algorithms, but they are able to achieve acceptable results in many kinds of problems. Wolpert and Macready have proposed No Free Lunch Theorem (NFL) (Wolpert and Macready, 1997) for search and optimization algorithms. For a given problem ϕ, the conditional probability for an algorithm alg to find global optima $gbest$ with iteration time $iter$ is set as $P(gbest\ |\phi, iter, alg\ )$.NFL proves that the sum of all conditional probabilities over all possible problems on finite domain is always identical for all optimization algorithms. The average performance over all given problems is independent of applied algorithm. That is, for two optimizers alg_1 and alg_2:

$$\sum_{\forall \phi} P(gbest\ |\phi, iter, alg_1\ ) = \sum_{\forall \phi} P(gbest\ |\phi, iter, alg_2\ )$$

For alg_1 outperforms alg_2 in one optimization problem, alg_1 has to be inferior in another problem. Figure 1 shows a crude sketch of NFL theorem. It is impossible for any method to always outperform non- repeating random walks (Weise, 2009). Algorithms are able to achieve good results in certain types of problems, but underperform in others.

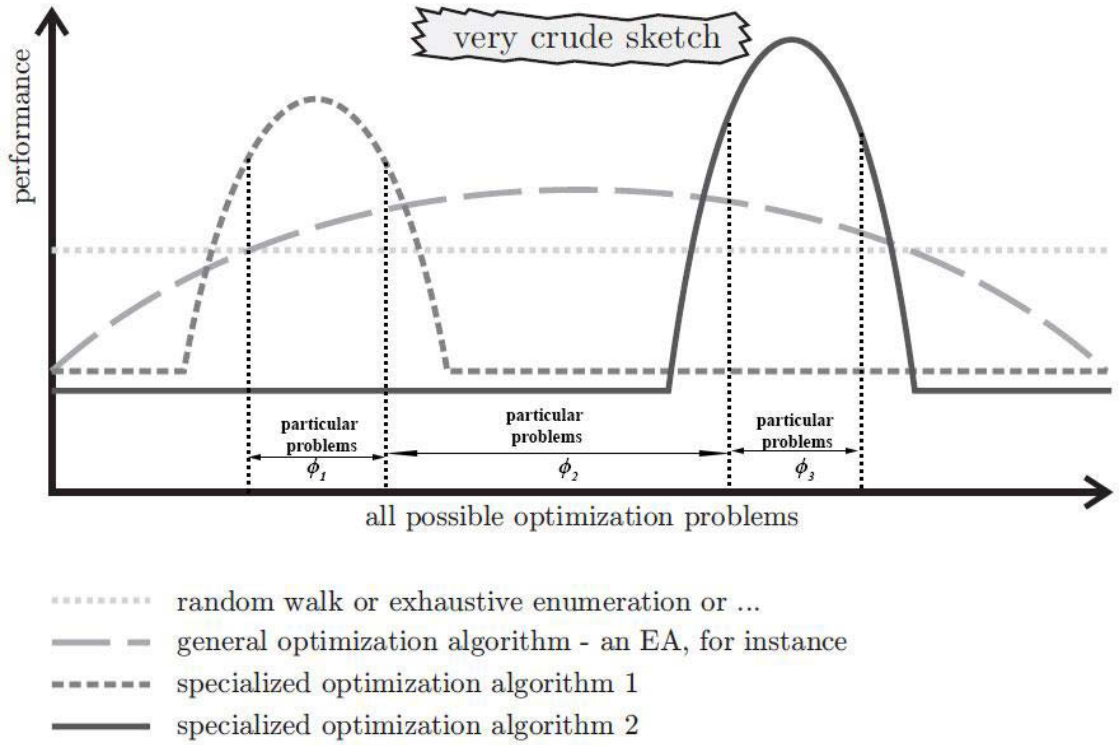


Figure 1 illustration of no free lunch theorem

Moreover, not all of the problems can be formulized with quadratic or analytic derivatives. Many of them are complicated and even considered as black boxes, which their analytic forms are not available. Instead of finding the precise solution(s) for a given problem, many researchers are trying to maximize the quality of discovered solutions. In practice, by only given some problems (e.g. $\phi_1, \phi_2,$ or ϕ_3 in Figure 1), it is possible to compare the performance between different optimization algorithms.

2.1.4 *Comments*

Optimization problems are diverse in forms and many optimization algorithms are required. However, specially designing optimizers for individual objective functions seems to be impractical due problems are constantly changing all the time. For a given problem, the objective function can be modelled in different ways due to different

aspects of problem representation. Besides, for one objective function with different dimension of input parameters, its problem landscapes and the global optimal regions may different. The ideal situation is to design and implement one approach which has the ability for solving all kinds problems, but it is little possible due to NFL theorem.

Think differently, for the black-box optimization that given problem(s) is unknown. If concurrently running different types of optimization algorithms, some of them may able to discover the optimal solutions, so the problems could be solved efficiently. This approach is known as memetic and hybrid algorithm that specific evolutionary and population-based problem optimizers are combined (Moscato et al., 2004). The individual agents in a population are cooperated and competed to one another to achieve individual improvement of the solutions. Hybridization of different algorithms does not mean to be hodgepodge. It is important to understand the theories behind optimizers, their nature inspired knowledge, and how to incorporate the ideas to solve the given problems. The review of some representative swarm optimization algorithms are shown in the next section 2.2. The principles of hybridizing different optimization techniques are discussed in Chapter 3 and 4.

2.2 Reviews for Optimization Algorithms

Although the inspirations often come from living creatures or nature phenomena, the constraints and objectives do not have to stick on these examples from nature. They are often re-implemented for fitting targeted problem domains.

2.2.1 *Differential Evolution (DE)*

DE is a population based evolutionary algorithm proposed by Storn and Price (Storn and Price, 1995). The key concept of DE is to generate new candidate solutions by calculating vector differences between other randomly selected solutions from population. DE takes four control parameters: population size NP, mutation factor

(Scaling rate) F, Crossover rate CR, and strategy st for choosing update equations. The pseudo-code of DE algorithm is shown as below:

Table 1. Pseudo-code for DE

1. Initialize a population of candidate solution vectors in D-dimensional search space.
2. Loop $G = 0 \sim maxIter$
   a. Perform evaluation on each candidate solution
      $X_G = \{X_{i,G}\},\ i = 1, \ldots, NP$
   b. Generate the trial candidate solution vectors U_G by given X_G
      # Update Equations
   c. Compare the quality of $U_{i,G}$ with $X_{i,G}$
      $$X_{i,G} = \begin{cases} U_{i,G} & if\ eval(U_{i,G})\ better\ than\ eval(X_{i,G}) \\ X_{i,G} & else \end{cases}$$
   d. Update $gbest$ with the best success in X_G
3. Exit loop if criterion is met

The update equations for DE are based on the framework of evolutionary algorithm that involves mutation and crossover phases. In mutation phase, one of the DE schemes is applied to create a set of new mutant vectors $V_G = \{V_{i,G}\}, i = 1, \ldots, NP$. During the mutation phase, mutated vectors V_G are synthesised from *G-1* generation's population X_{G-1}. The mutation strategy plays important role in DE algorithm. Many mutation strategies have been proposed and they all implemented for reasons. The following shows the list of frequently used strategies:

Table 2 DE mutation schemes

Strategy	**equation**
DE/best/1 (Storn, 1996)	$V_{i,G} = gbest + F * (X_{r1^i,G} - X_{r2^i,G})$
DE/rand/1 (Storn, 1996)	$V_{i,G} = X_{r3^i,G} + F * (X_{r1^i,G} - X_{r2^i,G})$
DE/best/2 (Storn, 1996)	$V_{i,G} = gbest + F * (X_{r1^i,G} + X_{r2^i,G} - X_{r3^i,G} - X_{r4^i,G})$

DE/rand/2 (Qin et al., 2009)	$V_{i,G} = X_{r5^i,G} + F * \left(X_{r1^i,G} - X_{r2^i,G}\right) + F * \left(X_{r3^i,G} - X_{r4^i,G}\right)$
DE/rand-to-best/1 (Storn, 1996)	$V_{i,G} = X_{i,G} + F * \left(gbest - X_{i,G}\right) + F * \left(X_{r1^i,G} - X_{r2^i,G}\right)$
DE/rand-to-best/2 (Qin et al., 2009)	$V_{i,G} = X_{i,G} + F * \left(gbest - X_{i,G}\right) + F * \left(X_{r1^i,G} - X_{r2^i,G}\right) + F * \left(X_{r3^i,G} - X_{r4^i,G}\right)$
DE/current-to-rand/1 (Iorio and Li, 2005)	$V_{i,G} = X_{i,G} + K * \left(X_{r1^i,G} - X_{i,G}\right) + F * \left(X_{r2^i,G} - X_{r3^i,G}\right)$
$where \begin{cases} r_{1^i,G} = rand(1,NP), r_{1^i,G} \neq i \\ r_{2^i,G} = rand(1,NP), r_{2^i,G} \neq i \\ r_{3^i,G} = rand(1,NP), r_{3^i,G} \neq i \\ r_{4^i,G} = rand(1,NP), r_{4^i,G} \neq i \\ r_{5^i,G} = rand(1,NP), r_{5^i,G} \neq i \\ r_{1^i,G} \neq r_{2^i,G} \neq r_{3^i,G} \neq r_{4^i,G} \neq r_{5^i,G} \\ K = rand(0,1) \end{cases}$	

Crossover Phase:

$$U_{i,j,G} = \begin{cases} V_{i,j,G} & if \left(rand_j(0,1) \leq CR\right) OR \left(j = j_{rand}\right) \\ X_{i,j,G} & otherwise \end{cases} \quad (8)$$
$$j = 1, \ldots, D$$

However, DE performance is highly dependent on its control parameters such as crossover rate CR, scaling rate F, and selection of mutation scheme. As it can be seen from Table 2, there are at least seven most frequently used mutation strategies used in DE algorithm (Mallipeddi et al., 2011). With the different configuration settings, DE can be used for increasing either exploration, or exploitation. For solving a specific optimization problem, trial-and-error search for appropriate strategy is not only time consuming, but also increases the computational cost. Here, in order to achieve balancing exploration and exploitation, the thesis suggests to concurrently using different mutation strategies. The detail discussion will be presented in Chapter 3.

2.2.2 *Particle Swarm Optimization (PSO)*

PSO (Kennedy and Eberhart, 1995) is population based search algorithm with a set of random solutions called particles. The inspiration comes from the nature behaviours such as bird flocking and fish schooling. Particles are flying through the search space. At each time point, particles migrate to new positions and their velocities change over time. The new positions and velocities for individual particles are dynamically adjusted according to their last positions, previous successes (*pbest*), and previous successes of its neighbours. Assume the population size *NP*, the population of *D*-dimensional solution vectors $X_{i,G}$, $i = 1, \dots, NP$ at time G with velocity vectors $V_{i,G}$. The pseudo-code of PSO algorithm is shown as below:

Table 3. Pseudo-code for PSO

1. Initialize a population of particles with random positions and velocities in D-dimensional search space.
2. Loop $G = 0 \sim maxIter$
   a. Perform evaluation on each particle $X_{i,G},\ i = 1, \dots, NP$
   b. Each particle move toward its best previous position ($pbest$) and towards the best particle in the whole swarm ($gbest$).
      Compare $X_{i,G}$ with $pbest_i$
      $$pbest_i = \begin{cases} X_{i,G} & if\ eval(X_{i,G})\ better\ than\ eval(pbest_i) \\ pbest_i & otherwise \end{cases}$$
   c. Update $gbest$ with the best success in $pbest$
   d. Update the velocity and position of the particles
      #Update mechanism
3. Exit loop if criterion is met

The update mechanism is the heart of the entire algorithm. Since 1995 the original PSO has been proposed, many continuous research are carried on enhancing the learning strategy. The original version PSO is shown as below (Kennedy and Eberhart, 1995):

Original PSO update mechanism

$$\begin{cases} V_{i,t+1} = V_{i,t} + r_1 * \left(pbest_{i,t} - X_{i,t}\right) + r_2 * \left(gbest - X_{i,t}\right) \\ X_{i,t+1} = X_{i,t} + V_{i,t+1} \end{cases} \quad (9)$$

Where $r_1 = rand(0, \phi_1)$ and $r_2 = rand(0, \phi_2)$. ϕ_1 and ϕ_2 are predefined control parameters. $V_{i,t}$ is kept in the range between $[-V_{max}, +V_{max}]$ where V_{max} is the control parameter.

In 1998, Shi and Eberhart (Shi and Eberhart, 1998a) modified the update mechanism by adding a control parameter, inertia weight ω :

Modified PSO update mechanism

$$\begin{cases} V_{i,t+1} = \omega * V_{i,t} + r_1 * \left(pbest_{i,t} - X_{i,t}\right) + r_2 * \left(gbest - X_{i,t}\right) \\ X_{i,t+1} = X_{i,t} + V_{i,t+1} \end{cases} \quad (10)$$

Based on the empirical study by Shi and Eberhart (Shi and Eberhart, 1999), The recommended inertia weight selection is $\omega \in [0.8, 1.2]$ if ω is not reduced with time. Otherwise, ω is suggested to decay from 0.9 to 0.4 during the optimization process.

Clerc and Kennedy implemented the constriction coefficients for controlling the convergence of the particle and eliminating the arbitrary V_{max} parameter(Clerc and Kennedy, 2002):

Constriction Coefficients update mechanism

$$\begin{cases} V_{i,t+1} = C * \left(V_{i,t} + r_1 * \left(pbest_{i,t} - X_{i,t}\right) + r_2 * \left(gbest - X_{i,t}\right)\right) \\ X_{i,t+1} = X_{i,t} + V_{i,t+1} \\ r_1 = rand(0, \phi_1) \\ r_2 = rand(0, \phi_2) \\ \phi = \phi_1 + \phi_2 > 4 \\ C = \dfrac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}} \end{cases} \quad (11)$$

Commonly, ϕ is set to 4.1 and $\phi_1 = \phi_2$. The constraint multiplier C is set to approximately 0.7298. $0.7298 * \left(\frac{4.1}{2}\right) \approx 1.49618$. The random numbers r_1 and r_2 are limited within the range [0, 1.49618].

Based on the neighbourhood structure, there are two common versions of PSO. One is global version that each particle is neighboured to the entire population. The other is local version in which an individual particle is neighboured to limited numbers of particles on its sides (Eberhart and Shi, 2004). In general, local version PSO with small neighbourhoods may achieve better performance with slow convergence on complex problems. In contrast, global version PSO (or local version PSO with large neighbourhoods) is able to achieve faster convergence, but often suffer from local optimal regions.

By reviewing the factors that impacting optimization performance in PSO and DE, the neighbourhood structure is to the update mechanism in PSO; mutation strategy selection is to the entire population in DE. These two concepts bring a new hypothesis of designing hybrid algorithm structure: the entire population is vaguely structured into groups and applied with different update equations. Each equation presents an independent way to form neighbour relation with other particles. The more detail explanation will be presented in Chapter 3 and chapter 4.

2.2.3 *Simplified Swarm Optimization (SSO)*

SSO was proposed by Yeh (Yeh et al., 2009). Initially, it was implemented for solving discrete data optimization problems that the shortcoming of PSO (Particle Swarm Optimization) algorithm (Yeh, 2009). PSO is an efficient optimization method for the problem within continuous space, but it easily suffers from local minima in discrete space. SSO originally named as discrete PSO and it solves the optimal problem by applying the particle swarm update strategy. For the i_{th} particle, its new candidate is synthesised from either the i_{th} particle itself, the currently best solution overall particles,

the generation best solution of i_{th} particle, or random variables. The Pseudo-code of SSO algorithm is shown as below:

Table 4. Pseudo-code for SSO

1. Initialize a population of candidate solution vectors in D-dimensional search space.
2. Loop $G = 0 \sim maxIter$
   a. Perform evaluation on each candidate solution vector $X_{i,G},\ i = 1, \dots, NP$
   b. Compare $X_{i,G}$ with $pbest_i$, and update $pbest_i$

$$pbest_i = \begin{cases} X_{i,G} & if\ eval(X_{i,G})\ better\ than\ eval(pbest_i) \\ pbest_i & otherwise \end{cases}$$

   c. Update $gbest$ with the best success in $pbest$
   d. Update the candidate solutions X_G

#Update mechanism

3. Exit loop if criterion is met

Although the principle of SSO is similar to the crossover technique in EA (Evolution Algorithm), the concepts are different. In addition to the parameters *NP* and *g*, *SSO* takes four extra parameters c_w, c_p, c_g, c_r to declare the thresholds for four crossover strategies:

SSO update mechanism:

$$c_w + c_p + c_g + c_r = 1$$
$$C_{\mathrm{w}} = c_{\mathrm{w}}, C_{\mathrm{p}} = C_{\mathrm{w}} + c_{\mathrm{p}}, C_{\mathrm{g}} = C_{\mathrm{p}} + c_{\mathrm{g}}$$
$$dice = rand(0,1)$$
$$X_{i,j,G} = \begin{cases} X_{i,j,G} & if\ \ 0 \le dice \le C_{\mathrm{w}} \\ pbest_{i,j,G} & if\ \ C_{\mathrm{w}} \le dice \le C_{\mathrm{p}} \\ gbest_j & if\ \ C_{\mathrm{p}} \le dice \le C_{\mathrm{g}} \\ lb_j + rand(0,1) * (ub_j - lb_j) & if\ \ C_{\mathrm{g}} \le dice \le 1 \end{cases} \quad (12)$$

where $pbest_i$ is the best solution for i_{th} candidate solution overall iterations. $gbest$ is the current best solution of all candidates. SSO employs the crossover operation to produce new candidate solutions. The current candidate solutions crossover with $pbest$,

$gbest$, random, or remaining the same based on the random number $dice$, where $dice \in \{0, \dots, 1\}$.

Similar to Roulette Wheel Selection (RWS) in genetic algorithm (GA) (Brindle, 1981), SSO uses four condition rules which can be imagined as four slots with different sizes in roulette wheel. However, SSO update mechanism is different from RWS, which does not calculate the Probability p_i and cumulative probability q_i see equation (13). In other words, the slots in SSO update mechanism are statically defined. Besides, SSO update mechanism syntheses new solution candidates, whereas GA uses RWS to select parent vectors.

Roulette Wheel Selection

Population Size: NP
Problem Dimension:D
Fitness cost:$f_i \;\; i = 1, \dots, NP$
population: $X = \{X_{i,j}\} \; where \; i = 1, \dots, NP, j = 1, \dots, D$
uninform random number : $r \in (0,1]$
Probability(slot size): $p_i = \frac{f_i}{\sum_{j=1}^{NP} f_j}$ (13)
Cumulative Probability: $q_i = \sum_{j=1}^{i} p_j$
Selection: $seItem = \begin{cases} x_1, \;\; if \; r < q_1 \\ x_i, \;\; else \; if \; q_{i-1} < r \leq q_i \end{cases}$

In short, SSO update mechanism (12) uses four control parameters $c_w, c_p, c_g,$ and c_r to dynamically select different update equations for generating diverse solution candidates from the current ones. Its operation mechanism can be applied to different use. The swarm optimization algorithms proposed in this thesis are developed based on SSO framework. As it can be seen in Chapter 3 and 4, both two algorithms extends SSO algorithm framework to randomly partition the entire population into sub-groups and apply different mutation strategies dynamically.

2.2.4 *Artificial Bee Colony (ABC)*

ABC is a population based optimization algorithm inspired from the nature behaviour of honey bee swarm proposed by Karaboga and Basturk (Karaboga and Basturk, 2007). The agents in ABC algorithm is called bees and the food sources are candidate solutions. One of the characteristics of ABC is the population structure that there are three types of bees in the colony: employed bees, onlookers, and scouts. Assumed the food sources can only be discovered by employed bees, so the number of employed bees is equal to the number of food sources. Employed bees pass the knowledge about food sources to onlookers by dancing. Then onlookers pick up one food source respectively and try to discover the honeys from their food sources. That is, perform local search on the food sources in order to find the local optima. The employed bees whose food sources have been abandoned, which means they cannot discover better local optima after a period of time, become scouts to start finding new food sources. The Pseudo-code of ABC is shown as below:

Table 5. Pseudo-code for ABC

1. Initialize food sources discovered by employed bees $X_G,\ G = 0$.
2. Loop $G = 0 \sim maxIter$
   2.1. Perform evaluation on each food source $X_{i,G},\ i = 1, \dots, NP$
   2.2. Employed bees try to discover better food sources neighbouring to the current ones.
      2.2.1. Update mechanism
      2.2.2. $X_{i,G} = \begin{cases} U_{i,G} & if\ eval(X_{i,G})\ better\ than\ eval(pbest_i) \\ X_{i,G} & otherwise \end{cases}$
   2.3. Onlookers try to perform local search based on the food sources X_G. Each food source has its own probability value $p_{i,G}$. The selection of food source depends on $p_{i,G}$ (see **probability equations**).
      $If\ (rand(0,1) > p_{i,G})$
      2.3.1. Update mechanism
      2.3.2. $X_{i,G} = \begin{cases} U_{i,G} & if\ eval(X_{i,G})\ better\ than\ eval(pbest_i) \\ X_{i,G} & otherwise \end{cases}$
      $endif$
   2.4. While a employed bee cannot update $X_{i,G}$ after predefined a period of times

2.4.1. The employed bee becomes scout by randomly search for new food source

2.5. Update $gbest$ with the best success in X_G

3. Exit loop if criterion is met

Each food source $X_{i,G}$ has a corresponding probability value $p_{i,G}$ and it is calculated by the following equation:

Probability equation:

$$p_{i,G} = \frac{fit_i}{\sum_{n=1}^{NP} fit_n} \tag{14}$$

Where fit_i is the fitness value of food source $X_{i,G}$ which indicates the amount of nectar near the position. If $rand(0,1) > p_{i,G}$, the $X_{i,G}$ will be selected for performing local search by onlooker.

ABC uses the following equations to update the trial food sources:

ABC update mechanism:

$$U_{i,j} = X_{i,j} + \phi_{i,j} * \left(X_{i,j} - X_{k,j}\right)$$
$$where\ k\ and\ j\ are\ are\ randomly\ chosen\ indexes$$
$$k \in \{1, \dots, NP\}\ and\ k \neq i,$$
$$j \in \{1, \dots, D\}$$
$$\phi_{i,j} = rand\ number\ between\ [-1, +1] \tag{15}$$

In brief, the idea behind ABC algorithm is to perform multiple times local search in the suspicious fields where may contain better solutions. The fields that cannot find better solutions after a certain number of iterations will be replaced with new regions in space.

2.2.5 *Ant Colony Optimization (ACO)*

Ant Colony Optimization(ACO), also known as Ant Colony System (ACS), which is a meta-heuristic algorithm inspired from the foraging behaviour of real ants (Dorigo and Gambardella, 1997). Ants are searching for food source via random paths. Once they

discover the food source, they go back to the nest and deposit pheromone on their return routes. Higher quantities of pheromone deposited on the paths means more ants use them to find food source. As a result, the most preferable path to food source would be the one with the highest quantity of pheromone on the ground. ACO was initially proposed for solving the optimization issue of TSP.

- $\boldsymbol{InitialPheromonValue}(\,)$

In TSP, there are n nodes which have to be visited exactly once. The population of the ant colony is NP. The solution components $C = \{c_{i,j}\}$ indicate the edges between pairs of cities to be visited one after another (see Figure 2 (Dorigo et al., 2006)). $c_{i,j}$ means the node i to node j. Individual ants discover the food source according to the pheromones on the edge $T = \{t_{i,j}\}$. Initially, ants have no knowledge of path from their nest to food source, so the pheromone on each path is the same $t_{i,j} = h, h > 0$ where h is the predefined constant parameter.

Table 6. ACO framework:

Initialize
$InitialPheromonValue(T)$
While ($\boldsymbol{termination\ conditions\ not\ met}$)

- **Loop (Ant 1~NP)**

$s = ConstructAntSolution(T, H)$
$if\ (s ==\ valid\ solution)$ {
$s = ApplyLocalSearch(s)$ ---- (optional)

#Evaluation and update
$if\ (f(s) < f(gbest)\ OR\ gbest == NULL)${
$gbest = s$
}
$S = S \cup s$
}

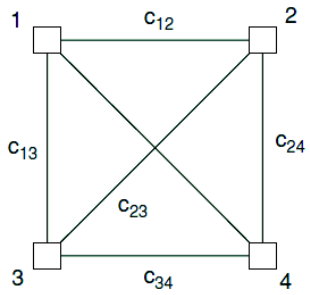
- **End-loop**

$UpdatePheromone(T)$
End-while

Figure 2. construction

- $\boldsymbol{ConstructAntSolution(T)}$

graph for 4 nodes tsp



Individual ants construct the solution s from the set of solution components C. Initially, ants have no knowledge about the path, so their knowing solution is empty at the beginning $s^p = \{\} = \emptyset$. At each iteration of the algorithm, individual ants move from one node to another and extended the partial solution s^p. How an ant moves to next node is depended on the transition probability :

Transition Probability	$p_{i,j,t}^k = \begin{cases} \frac{(\tau_{i,j,t})^\alpha * (\eta_{i,j})^\beta}{\sum_{c_{i,u} \in N(s^p)} (\tau_{i,u,t})^\alpha * (\eta_{i,u})^\beta} & else\ if\ c_{i,j} \in N(s^p) \\ 0 & otherwise \end{cases}$ $where \quad \eta_{i,j} = \begin{cases} \frac{1}{d_{i,j}} & if\ d_{i,j} > 0 \\ \frac{1}{0.0001} & otherwise \end{cases}$ (16)

Table 7. ACO decision rule

$if\ (q \leq q_0)\ then$
$\quad j = \arg\max_{c_{i,u} \in N(s^p)} \{\tau_{i,l} * \eta_{i,j}^\beta\}$
$elseif$
$\quad sumProb = sum(p_{i,u,t}^k), \quad u \in \arg(N(s^p))$
$\quad prob = rand(0,1] * sumProb$
$\quad j = 1$
$\quad p = \frac{p_{i,1,t}^k}{sumProb}$
$\quad while\ (p < prob\ AND\ j \leq |N(s^p)|)$
$\quad\quad j++$
$\quad\quad p += \frac{p_{i,j,t}^k}{sumProb}$
$\quad endwhile$
$endif$
$nextNodeIdx = j$

ACS uses pseudo-random proportional rules that $p_{i,j,t}^{k}$ is the probability of ant k moves from node i to node j at iteration t. q is the random variable uniformly distributed between 0 and 1. q_0 is the predefined control parameter by user. $N(s^p) \subseteq C$ is the set of solution components which can be added in the current partial solution s^p. u is the one of the nodes which is not yet visited by ant k. α and β are the control parameters for controlling the relative importance between pheromone and edge distance $d_{i,j}$. The construction process will continuously perform until individual ants' s^p are generated.

- $\boldsymbol{ApplyLocalSearch(s)}$

Local search is an optional step in ACO. The solutions discovered by $ConstructAntSolution(T)$ can further improved by the ants through a local search. The algorithm is usually varying for different problem.

- $\boldsymbol{UpdatePheromone(T)}$

There are two ways to update pheromone level on each edge. One is to increase the pheromone level for the edges which are associated with good solutions. Another is called pheromone evaporation, which is to decrease the pheromone level on all edges. Thus, the solution components for satisfactory results will be relatively clear than others. The pheromone update is performed by all ants after $ConstructAntSolution(T)$, evaluation and solution update.

Pheromone update formula	$\tau_{i,j} = \begin{cases} (1-\rho) * \tau_{i,j} + \rho * \Delta\, \tau_{i,j} & if\ (i,j)\ \in\ gbest \\ \tau_{i,j} & otherwise \end{cases}$ $\Delta\, \tau_{i,j}^{k} = \begin{cases} \frac{1}{L_{best}} & if\ ant\ k\ use\ edge\ (i,j) in\ solution \\ 0 & otherwise \end{cases}$	(17)

Where ρ is pheromone evaporation rate, commonly $0 < \rho \leq 1$ (Dorigo and Blum, 2005). $\Delta\, \tau_{i,j}^{k}$ means ant k leaves a quantity of pheromone on the edge from node i to

node j. L_{best} is the length of the best solution. The best solution can be either for iteration best solution, best-so-far solution, or combination of both.

The movement behaviour of real ants is often simulated as Biased Correlated Random Walk (BCRW). When ants move from one node to another, they often select the direction with higher pheromone level along its edge. Meanwhile, there is still a chance to choose any other track. Besides, an ant changes its direction gradually from step to step. The next destination chosen to move on is related to its current position. Thus, based on the shared knowledge in the population, individual ants make their own decisions to find food sources with independent trajectories.

2.3 Nature-inspired mechanism

In SI system, many researchers seeking for the inspiration form nature. Up to now, many reprehensive Swarm optimization algorithms have been reviewed. Many of them are inspired from nature phenomena such as ant colony system. However, ant foraging behaviour is merely an example of biological behaviours in nature. For different purposes and different species, animals present various ways and rules to achieve their goals (i.e. to survive in their living environments). Before understanding and mimicking a particular example, it is important to investigate their common fundamental feature. That is, the basic concept of motion behaviour and random walk. The more detail review is shown in subsubsection 2.3.1.

Moreover, the world, reality, and nature that everyone lives and sees every day are governed by unfamiliar laws. The laws which cannot be fully described, observed, or interpreted, but it is believed they are real. There must be hidden variables between inputs and outcomes. Although discovering the hidden rules behind the system is one research branch, it might be a debate whether the deterministic rules exist, especially for the study which is incomplete. One example is quantum mechanics. In great debate between Bohr and Einstein, Einstein wrote: *"God does not play dice*", which means

determinism that the future is completely determined by the present. Nevertheless, Bohr replied: "*Einstein, stop telling God what to do*". Bohr and Copenhagen interpretation believe indeterminism can be applied on everything due to the nature itself is fundamentally fuzzy. Only the observation can collapse the uncertainty (with many probabilities) into one possible result. Similarly, swarm optimization algorithms often treat the given problems as black-boxes. The problem landscape is unknown (or not necessary to be knowable) and, in any instant of time, individual search agents stochastically move to somewhere in space. Only the evaluation results can present the meaningful information. A brief review of quantum mechanics is shown in subsubsection 2.3.2.

2.3.1 *Motion behaviour and random walk*

The moving way of an object which can be observed by naked eyes is often continuous in space-time. It follows certain laws, such as Newton's laws of motion. For instance, while observing a bird foraging behaviour in a period of time, a bird flies toward the food source with a particular trajectory. However, an object can only exist in one place in any one instant of time. The movement of an object consists of series of infinitesimal interval states, which can be considered as a continuous point set with infinite innumerable positions in space. According to the point set theory, the precise positions for point set cannot be found without measurement. They are discontinuous somewhere in space, even if the law for motion is known (Shan, 2001). That is, by giving a time-interval, the moving way of an object is continuous which is constituted with infinite discontinuous random positions, a sequence of hops, in space.

For animal locomotion, an animal performs random walk at every instant of time (Codling et al., 2008). Its motion could be uncorrelated random walk (URW) or correlated random walk (CRW). URW means each hop performed by an organism is independent to prior hops. In contrast, the animal movement described by CRW is that there are certain dependencies (i.e., number of hops, move length, and distribution of

random turning angels) between individual hop and its successive hops. For animal swarm, the movement of an individual random walker is not only correlated with its previous moving pattern, but may also depend on its neighbours moving behaviours. Meanwhile, animals often respond to the environmental stimulus and introduce bias into their movement (e.g., bacteria movement is biased by the concentration gradient of a particular chemical).

In addition, for complex animal societies (i.e., human societies), or symbiosis between different species (i.e., human-E.coli, ant-plant), the distinct types of behavioural mechanisms cooperate and interact with one another to achieve the common goals e.g. for better living condition and survival. In the research of nature-inspired optimization algorithm, the behavioural mechanisms can be illustrated as random walk strategies, as well as the recombination operators, for dealing with all sorts of strange shapes in the given problem landscape. In this sense, the hybridization between different strategies is very important, since every technique has a chance to cause a search agent stuck in local optima. The search agents that suffered from local optima regions are expecting to be "rescued" by other recombination operators. More detail discussion will be presented in Chapter 3 and 4.

The object which can be observed by naked eyes is in macro-scale. Macroscopic phenomena often can be described by classical mechanics (Newton's mechanics). In short, the motion of a particle can be described by a definite trajectory. Its position and momentum (or velocity) can be measured precisely and the future path can be predicted correctly. However, the mechanics in microscopic phenomena such as electrons is completely different story.

2.3.2 *The basic quantum mechanics and uncertainty*

Quantum mechanics is used to describe the nature phenomena in microscopic world. In quantum mechanics, particles are wave-particle duality in nature. Each particle can be

illustrated as a wave packet, the composite of many waves, and it makes quantum particles have wave and particle-like properties. The motion of particles can be described by wave function, which is a mathematical function, determines the probability density of the particles appearing in the certain positions. According to the Heisenberg uncertainty principle, there is an uncertainty relation between position and momentum. While the uncertain range for a quantum particle position is narrow, its uncertain range for momentum is wide, and vice versa (Hameka, 2004, Wimmel, 1992). The quantum objects exist in superposition, which is the combination of all possible states, until the measurement takes place. According to Copenhagen interpretation of quantum mechanics, the measurement is not passing the observation, is interaction between a particle and result in both way. Even more, one of the interesting quantum phenomena is quantum entanglement.

Similar to the evaluation process in SI system, the quality of solution cannot be certain due to search agents (particles) are constantly moving through the solution space via stochastic recombination operators. Assume a well-designed operator which is able to guide particles approaching to global optimal regions. Although the discovered solution(s) is attempted to be refined iteratively, the convergence to better results is not always guaranteed. The system may experience several numbers of iterations with unsatisfied evaluation results, and then occasionally find better solution candidates. Therefore, how to increase the probability of gathering satisfactory evaluation results is important. The more specific discussion regarding to evaluation process will be presented in chapter 4.

2.4 Evaluating optimization algorithms

In order to evaluate and compare different optimization algorithms, the public mathematical benchmark functions are used as black-box objective functions (see Appendix A). The performance is measured by the difference of evaluation result between discovered solution and predefined global best solution, called error value. See

equation (2). There are two algorithms are presented in this thesis (Chapter 3 and 4). The more specific experiment designs are explained individually (see subsection 3.4 and 4.4).

The boundary handling method called Periodic mode (Zhang et al., 2004) is applied to solve beyond boundary problem (18).

$$range = ub - lb$$
$$u_{i,j} = \begin{cases} ub - mod\left((lb - u_{i,j}), range\right), & u_{i,j} < lb \\ lb + mod\left((u_{i,j} - ub), range\right), & u_{i,j} > ub \\ u_{i,j}, & u_{i,j} \in [lb, ub] \end{cases} \quad (18)$$
$$* \ mod = modulus$$

By definition, modulus is the amount by which a number exceeds the largest integer multiple of the divisor that is not greater than the number. However, in many programming languages, such as Java or C, modulus function is implemented as remainder function. Therefore, by replacing modulus with remainder, the equation (18) can be written as follow (19):

$$range = ub - lb$$
$$u_{i,j} = \begin{cases} ub + rem\left((u_{i,j} - lb), range\right), & u_{i,j} < lb \\ lb + rem\left((u_{i,j} + ub), range\right), & u_{i,j} > ub \\ u_{i,j}, & u_{i,j} \in [lb, ub] \end{cases} \quad (19)$$
$$* \ rem = remainder$$

2.5 Chapter summary

By reviewing many scientific literatures on various types of optimization algorithms, their researches are carried on following areas:

- Ideas or techniques are inspired from nature or existing algorithms.
- Update mechanism for synthesis new candidate solutions.
- High-level Algorithm framework implementation.

Table 8 population based algorithm common framework

4. Initialize Population *X* 5. Initial Evaluation: *eval(X)* 6. Update *the best solutions* 7. Repeat until ($G = 0 \sim maxIter$) a) Reproduction process (i) Emerge trial population *U* (ii) Evaluation: *eval(U)* (iii) Re-constitute population *X* b) Update *the best solutions* discovered so far 8. END

Some algorithms are implemented base on common principles or well-known approaches, but not in detail. For instance, SSO, DE, and PSO are implemented based on the common population based algorithm procedures (Storn and Price, 1997, Kennedy, 2006) (Table 8). However, they all use different update mechanisms to generate new candidate solutions. In DE, there is no $pbest$ matrix for storing the best solutions discovered so far by individual particles. The current particles' positions in the solution space are continuously updated to the best positions. In contrast, SSO and PSO iteratively update the current particles into new positions in solution space, which is different from the ones in $pbest$ matrix. The update mechanism for DE involves the calculation of vector difference and biological concept, mutation and crossover phase, from Genetic Algorithm. The update mechanism for PSO involves the calculation of position and velocity which is inspired from bird flocking or fish schooling. ABC and ACO algorithms present distinct algorithm frameworks and update mechanisms that simulate the intelligent of foraging behaviours of honey bee swarm and ant colony respectively. In addition, although different algorithms present different methods and principles to solve optimization problems, the techniques and formulae they use often refer to fundamental mathematical concepts. E.g. the equations used in DE, PSO, and ABC update mechanisms involve the calculation of vector difference.

In this thesis, the research is focused on designs and implements new methods to efficiently and effectively solve black box optimization problem with swarm

intelligence. Since the characteristics of the solutions are unknowable beforehand, the iterative optimization procedures can be illustrated as a set of search agents moving their positions in solution space. Agents make biased random walk move toward to global optima according to the predefined rules or strategies. In the following chapter (chapter 3), the hybrid algorithm with different mutation strategies is proposed.

3 SIMPLIFIED SWARM OPTIMIZATION WITH DIFFERENTIAL EVOLUTION MUTATION STRATEGY FOR PARAMETER SEARCH

3.1 Introduction

Many soft computing based optimization algorithms have been proposed for locating the global and local optima parameters for functions within predefined search space. For instance, evolutionary computation methods such as Genetic Algorithm (GA)(Dong Seong et al., 2005), particle swarm optimization (PSO)(Poli et al., 2007), Differential Evaluation (DE) (Chakraborty, 2008), and so forth are well known algorithms for finding optima. Researchers propose many different concepts and assumptions for enhancing the usage of existing algorithms or introducing the new approaches. At the time point their papers been published, their ideas were guaranteed to solve particular optimization problems based on their hypotheses. Therefore, solving any optimization problems with existing methods seems to be a good starting point.

However, according to the No Free Lunch Theorem (NFL) (Wolpert and Macready, 1997), it is little possible to find a uniformly best search or optimization algorithm for solving all possible problems. An algorithm A might outperform algorithm B in some problems, then there must be some other problems algorithm B outperforms algorithm A (see subsubsection 2.1.3). Even more, due to the fact that focuses can be varying according to different methods and their settings, the ways of balancing exploitation and exploration are different. Hybridization of different algorithms' operators and incorporating their knowledge would be increase the general efficiency of searching methods (Grosan and Abraham, 2007). Hybrid evolutionary algorithm is able to handle the problems involving complexity, noisy environment, imprecision, uncertainty, and vagueness.

The proposed new global optimization algorithm is Simplified Swarm Optimization (SSO) with Differential Evolution (DE) mutation strategy SSODE. SSODE is developed to overcome the difficulty of choosing mutation strategy and scaling rate in DE. One drawback of DE is the mutation strategy and scaling rate need to be predefined constantly. They are not adaptively selected according to the quality measures of current particles. Many mutation strategies have been proposed (see Table 2) and they all implemented for reasons. E.g., DE/best/2/bin strategy provides good convergence, whereas, DE/rand/2/bin strategy provides good diversity (Lampinen and Zelinka, 2000). Although fast convergence leads to efficiently discover the optimal solution, it is easy to suffer from local optima; diversity ability too high may cause the function never converge to global optima. If the predicted global best result is closed to global optima, DE/best/2/bin strategy is probably more preferred than DE/rand/2/bin.

3.2 Inspiration

Assume the solution space of an objective function has one global extremum and many local extrema. Swarm Intelligence (SI) approach is used for discovering the best solutions. Many well-known SI-based algorithms have certain characteristics. First, algorithms search the areas of interest in solution space which possibly return good evaluation results (exploration). Second, algorithms tend to iteratively improve the current solutions by small changes, which try to make them approach to the extrema in these good regions (exploitation). Third, there is a trade-off to be had between exploitation and exploration. Algorithms that favour to exploitation are able to achieve fast convergence, but may be suffered from premature convergence. Whereas, algorithms that favour to exploration can possibly avoid failing into local optima, but may never form convergence.

During the processing of a SI-based optimization algorithm, particles are flying over the solution space according to the predefined rules. The new solution particles are emerged from a group of existing particles in every iteration time. The evaluation process is

relied on objective function for measuring the quality of discovered solutions. Although the quality of solution often reflects the importance of particle position in solution space, however, it does not tell which regions (in solution space) are worth to perform exploitation process.

One of the simple clustering methods is k-nearest neighbour (KNN) algorithm (Tan, 2007). Based on the Nearest Neighbour (NN) problem that determining the closest data points near the given query point (Beyer et al., 1999), KNN assigns the query point as the same class as the majority group of its nearest neighbours. However, with the increase of dimensionality, the different in distance from the query point to its nearest neighbour and other points becomes insignificant (Beyer et al., 1999). The clustering result becomes unstable. Besides, the algorithm requires longer computation time to process high dimensional data.

3.3 SSODE algorithm

SSODE algorithm assumes particles in solution spaces can be formed as clusters, or will form clusters, since their movements are coordinated by other members in the population. Different clusters of particles indicate they are in different regions of solution space. Therefore, the different update mechanism rules should be applied for individual regions. The clustering principle in SSODE is based on the evaluation results of particles. Inspired from the nearest neighbour concept in KNN, the particles that return similar evaluation results can be clustered together. The evaluation process can be considered as the projection of individual particles from high dimensional solution space to low dimensional space, with the aspect of the distances between individual particles to the real target position.

SSODE fuzzily clusters and ranks the particles into four different classes according to their evaluation results with descending order Table 9) :

Table 9 One population has four classes of particles

Good Region	Class A	Particles are almost formed convergence to the extrema in global best or local best regions (exploitation)
	Class B	Particles in the global or local best regions, but have not been formed convergence yet. (exploitation)
Hesitant region	Class C	Particles are approaching to good regions (exploration)
	Class D	Random-like particles in the solution space (exploration)

Based on the fitness function evaluation, the particles with the lower error or higher accuracy are probably the ones close to global optima or local optima. Therefore, they belong to Class A (the best solution group) and Class B (the second best group). The particles in hesitant region are probably meaningful particles, cannot be seen from evaluation result. The particles with normal evaluation results are belong to Class C (normal group) and the rest of particles belong to Class D.

For different classes, the different formulas for mutation (see Table 10) are used to generate new mutated vectors $v^g = \{v_{i,j}^g \dots, v_{NP,D}^g\}$. The formulae used in SSODE are inspired by DE mutation strategies with slightly change.

While more in-depth study of DE mutation strategies (Table 2), the mutation vectors are generated in the following ways:

- Synthesised from the best particle with randomly selected other particles.
- Synthesised from the current particle with randomly selected other particles.
- Synthesised from one randomly selected particle with randomly selected other particles.

Table 10 SSODE four formulas for mutation

Formula 1: $\boldsymbol{Func1}(\boldsymbol{x}, \boldsymbol{F})$ $v_i^g = x_i^g + F * \left(x_i^g - x_{r1}^g\right) + F * \left(x_{r2}^g - x_{r3}^g\right)$
Formula 2: $\boldsymbol{Func2}(\boldsymbol{x}, \boldsymbol{F})$ $v_i^g = x_i^g + F * (pbest_{r1} - pbest_{r2}) + F * (pbest_{r3} - pbest_{r4})$
Formula 3: $\boldsymbol{Func3}(\boldsymbol{x}, \boldsymbol{F})$ $v_i^g = gbest + F * \left(x_{r1}^g - x_{r2}^g\right) + F * \left(x_{r3}^g - x_{r4}^g\right)$
Formula 4: $\boldsymbol{Func4}(\boldsymbol{x}, \boldsymbol{F})$ $v_i^g = x_i^g + K * \left(x_{r1}^g - x_i^g\right) + F * \left(x_{r2}^g - x_{r3}^g\right)$

Formula 1 is inspired by DE/current-to-rand/1 by replace K with F (Iorio and Li, 2005); Formula 2 is inspired by DE/best/2 (Storn, 1996) and DE/rand/2 (Qin et al., 2009) by replacing p_{best}^g with current p_i^g and $p_{r_{1\sim4,i,g}}^g$ are replaced with $pbest_{r_{1\sim4,i,g}}$; Formula 3 refers to DE/best/2 (Storn, 1996); and Formula 4 refers to DE/current-to-rand/1 (Iorio and Li, 2005). Based on the predefined Crossover rate CR, the candidate solutions elements $x_{i,k}^g$ are crossover with $v_{i,k}^g$ and generate new particle position X^{g+1} for next generation. The crossover operation is the same as the one in DE algorithm (eq (8)). SSODE takes parameters NP (population size), max_gen (maximum generation), and CR (Crossover Rate). Besides, inspired from the update mechanism used in Simplified Swarm Optimization (SSO), SSODE takes four additional control parameters gm, cp,gr, and cr to define the proportion of particles sequentially assigned to Class A, Class B, Class C, and Class D (see Table 1). The algorithm is shown as bellow:

Table 11 SSODE algorithm

Algorithm SSODE algorithm

Input: gm, cp, gr, cr, $maxIter$, NP, D
$gm + cp + gr + cr = 1$
Output: $pbest$, $gbest$

1: g=0
2: $M_{gm} = gm$, $M_{cp} = M_{gm} + cp$, $M_{gr} = M_{cp} + gr$
3: $X^g = \{x^g_{i,j}, ..., x^g_{NP,D}\}$ ▷ for g^{th} generation
4:
5: **repeat**
6: $X^g = sort(X^g, 'desc')$
7: **for** $i \leftarrow 0, round(NP * M_{gm})$ **do**
8: $F = M_{gm}$
9: $v^g_i = \textbf{Func1}(X^g, F)$
10: **for** $i \leftarrow round(NP * M_{gm}) + 1, round(NP * M_{cp})$ **do**
11: $F = M_{cp}$
12: $v^g_i = \textbf{Func2}(X^g, F)$
13: **for** $i \leftarrow round(NP * M_{cp}) + 1, round(NP * M_{gr})$ **do**
14: $F = M_{gr}$
15: $v^g_i = \textbf{Func3}(X^g, F)$
16: **for** $i \leftarrow round(NP * M_{gr}) + 1, NP$ **do**
17: $F = rand(0, 1]$
18: $v^g_i = \textbf{Func4}(X^g, F)$
19:
20: **if** $rand(0, 1] \leq CR$ **then**
21: $x^{g+1}_{i,j} = v^g_{i,j}$ ▷ $j = 1, ..., D$
22: **else**
23: $x^{g+1}_{i,j} = x^g_{i,j}$
24:
25: **for** $i \leftarrow 0, NP$ **do**
26: **if** $\textbf{eval}(x^g_i) < pbest_i$ **then**
27: $pbest_i = x_i$
28: **if** $\textbf{eval}(pbest_i) < gbest$ **then**
29: $gbest = pbest_i$
30: g++
31: **until** ($g == maxIter$ **OR** *meet termination condition*)

When the particles is sorted with descending order according to the evaluation result based on fitness function, from top to bottom, the first group ($NP \times gm$ particles) belongs to Class A with the smallest scaling rate (M_{gm}); the second group ($NP \times cp$ particles) belongs to Class B with slightly larger scaling rate (M_{cp}); the third group ($NP \times gr$ particles) belongs to Class C with the largest scaling rate (M_{gr}); and the last group ($NP \times cr$ particles) belongs to Class D with random scaling rate. The mutation formulas for individual classes are listed in (Table 10). Figure 3 shows the detail steps of SSODE algorithm.

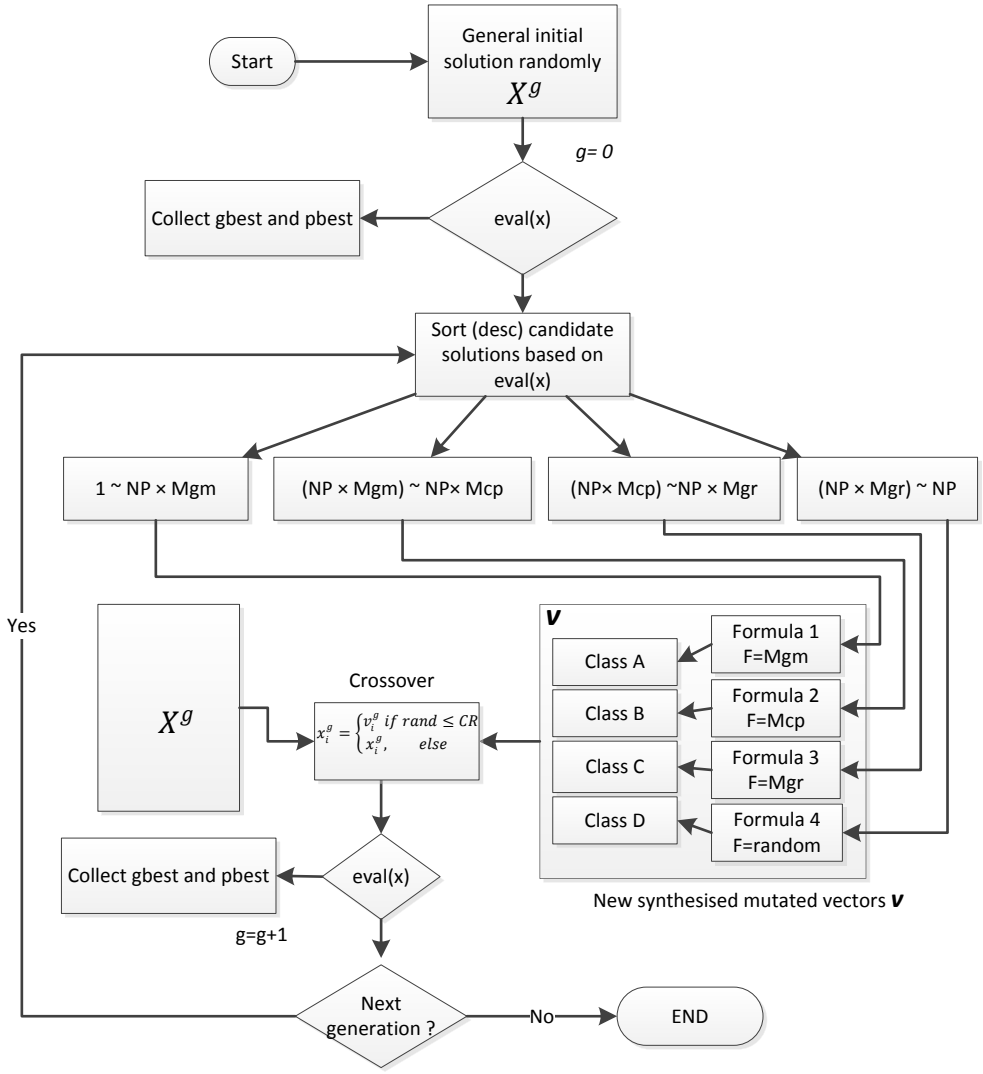


Figure 3 the flow diagram of SSODE algorithm

3.4 Experiment

Two Experiments are carried on to verify the performance of SSODE algorithm. Here, the proposed algorithm SSODE is compared with original SSO and various types of DE algorithms. For SSO(2-3-3-2) algorithm, the selection of control parameters Cwpgr=[0.2, 0.3, 0.3, 0.2] is based on trial and error. These parameter values are also been selected for SSODE(2-3-3-2) [gm, cp gr, cr] =[0.2, 0.3, 0.3, 0.2] and SSODE(4-2-

2-2) [gm, cp gr, cr] =[0.4, 0.2, 0.2, 0.2]. For DE algorithm, the source code is downloaded from Differential Evolution Homepage in Matlab code section (Price and Storn, 1996). Based on the demo source code, the scaling rate $F = 0.8$ and crossover rate CR=0.8 are chosen. Similarly, SSODE also use crossover rate $CR = 0.8$ for manipulate diversity, but the calculation for scaling rates Fs are calculated by [gm, cp gr, cr] according to Table 11. The population size is 10*D = 20, which is the recommended setting by Storn and Price (Storn and Price, 1997).

Due to the DE and SSODE algorithms may generate the variables that exceeded the boundary, the boundary handling method called Periodic mode (Zhang et al., 2004) is applied to solve beyond boundary problem (18) .

3.4.1 *Hyper-parameter selection for Support Vector Machine (SVM)*

The aim of this experiment is to test the performance of SSODE for solving hyper-parameter tuning problem. The data mining toy problem is based on the supervised learning algorithm Support vector machine (SVM) (Vapnik, 1995) and public benchmark dataset. The public SVM package, LIBSVM (Chang and Lin, 2011) is selected and the tool C-Support Vector Classification (C-SVC) with Radial Basis Function (RBF) kernel is used for this experiment. The benchmark dataset "spambase" is downloaded from UCI machine Learning Repository created by Hopkins, et al.(Blake and Merz, 1998). The dataset consist 4601 email instances with 1813 spams and 2788 non-spam emails with 57 attributes. The goal is to find the best hyper parameters: cost value 2^{log2c} for C-SVC and the best gamma value 2^{log2g} for RBF kernel functions. Both parameters $log2c$ and $log2g$ are within the range -10~10. The spam e-mail database 'spambase' is used for the experiment which gatheredThe 5-fold stratified cross-validation (cv=5) (Witten et al., 2011) technique is used and in each iteration, the original data is randomly partitioned into non-overlapped 50% training set and 20% testing set. The statistical analysis will based on minimum, maximum and mean of

accuracy, and the standard deviation overall 30 times experiment. The comparison of the optimization result is based on 100 times of generation.

The results, as shown in Figure 4 and Table 12, is quite revealing in several ways. The medians of all algorithms are almost the same and close to 87% accuracy. The Interquartile range (IQR) for SSO is more spread than other algorithms. The DE with DE/rand/1 has the second large IQR. There is an extreme outlier in SSO, which may be the indication of suffer from strong local optima. Although DE is able to achieve appropriate results, with the different mutation strategy, the distributions of results are different. Thus, choosing appropriate mutation strategy plays important role for solving a particular problem. The prior knowledge or trial-and-error search for finding appropriate strategy is not only time consuming, but also increases the computational cost. In contrast, the IQR for SSODE is small and there is no significant skewness in the distribution. Besides, the amounts of outlier points are small. To sum up, the experiment shows SSODE and DE achieve better performance than SSO.

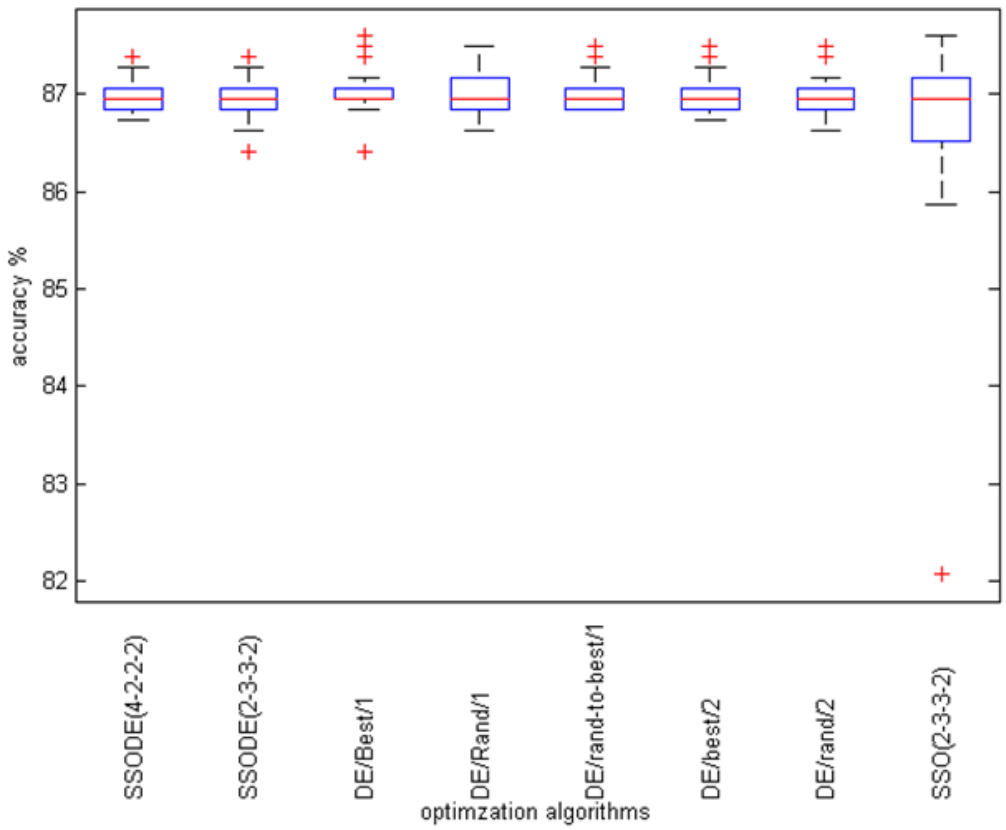


Figure 4 boxplot results among different approaches

Table 12 summary results among different algorithms and settigns

Over 30 times Experiment with cv=5		SSODE (4-2-2-2)	SSODE (2-3-3-2)	DE/best/1	DE/rand/1	DE/rand-to-best/1	DE/best/2	DE/rand/2	SSO (2-3-3-2)
Accuracy %	max	87.39	87.39	87.61	87.5	87.5	87.5	87.5	87.61
	min	86.74	86.41	86.41	86.63	86.85	86.74	86.63	82.07
	avg	86.98	86.96	87.01	87	87.03	87.04	87	86.65
	std	0.19	0.19	0.23	0.21	0.18	0.22	0.25	1.14

3.4.2 *Performance test based on public benchmark functions*

In this experiment, SSODE is compare with SSO and DE algorithms based on some public benchmark functions from CEC 2005 special session on real-parameter optimization (Suganthan et al., 2005) (see Table 13).

Table 13 benchmark functions

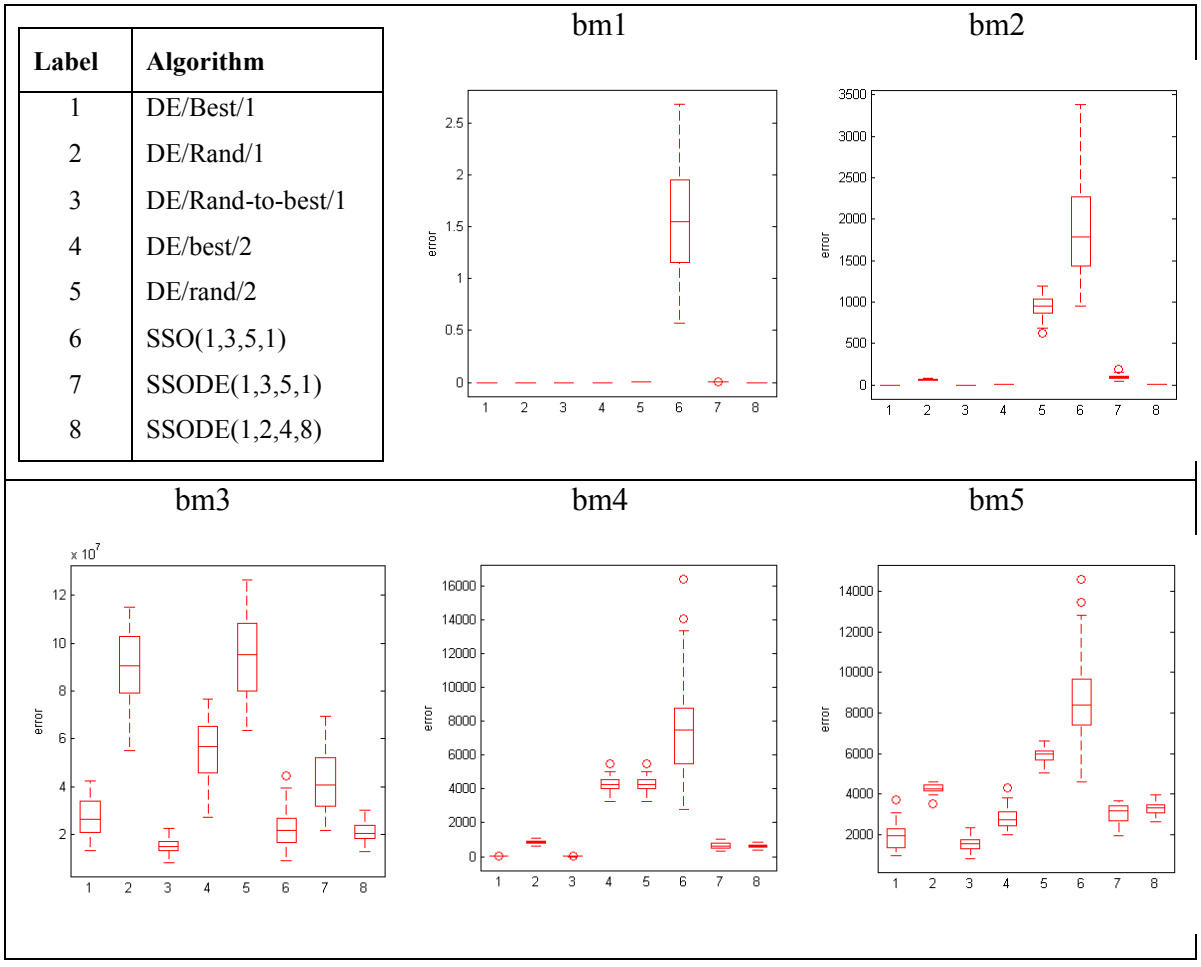
	*f(x)=f(x)+f_bias			
No.	**Benchmark function f(x)**		**boundary**	**f_bias**
Unimodal function				
bm1	Shifted Sphere Function	separable	[-100,100]	-450
bm2	Shifted Schwefel's Problem 1.2	non-separable	[-100,100]	-450
bm3	Shifted Rotated High Conditioned Elliptic Function	non-separable	[-100,100]	-450
bm4	Shifted Schwefel's Problem 1.2 with Noise in Fitness	non-separable	[-100,100]	-450
bm5	Schwefel's Problem 2.6 with Global Optimum on Bounds	non-separable	[-100,100]	310
Multimodal function				
bm6	Shifted Rosenbrock's Function	non-separable	[-100,100]	390
bm7	Shifted Rotated Griewank's Function without Bounds	non-separable	[-600, 600]	-180
bm8	Shifted Rotated Ackley's Function with Global Optimum on Bounds	non-separable	[-32,32]	-140
bm9	Shifted Rastrigin's Function	separable	[-5,5]	-330
bm10	Shifted Rotated Rastrigin's Function	non-separable	[-5,5]	-330
bm11	Shifted Rotated Weierstrass Function	non-separable	[-0.5,0.5]	90
bm12	Schwefel's Problem 2.13	non-separable	[-100,100]	-460
bm13	Expanded Extended Griewank's plus Rosenbrock's Function (F8F2)	non-separable	[-3,1]	-130
bm14	Expanded Rotated Extended Scaffe's F6	non-separable	[-100,100]	-300

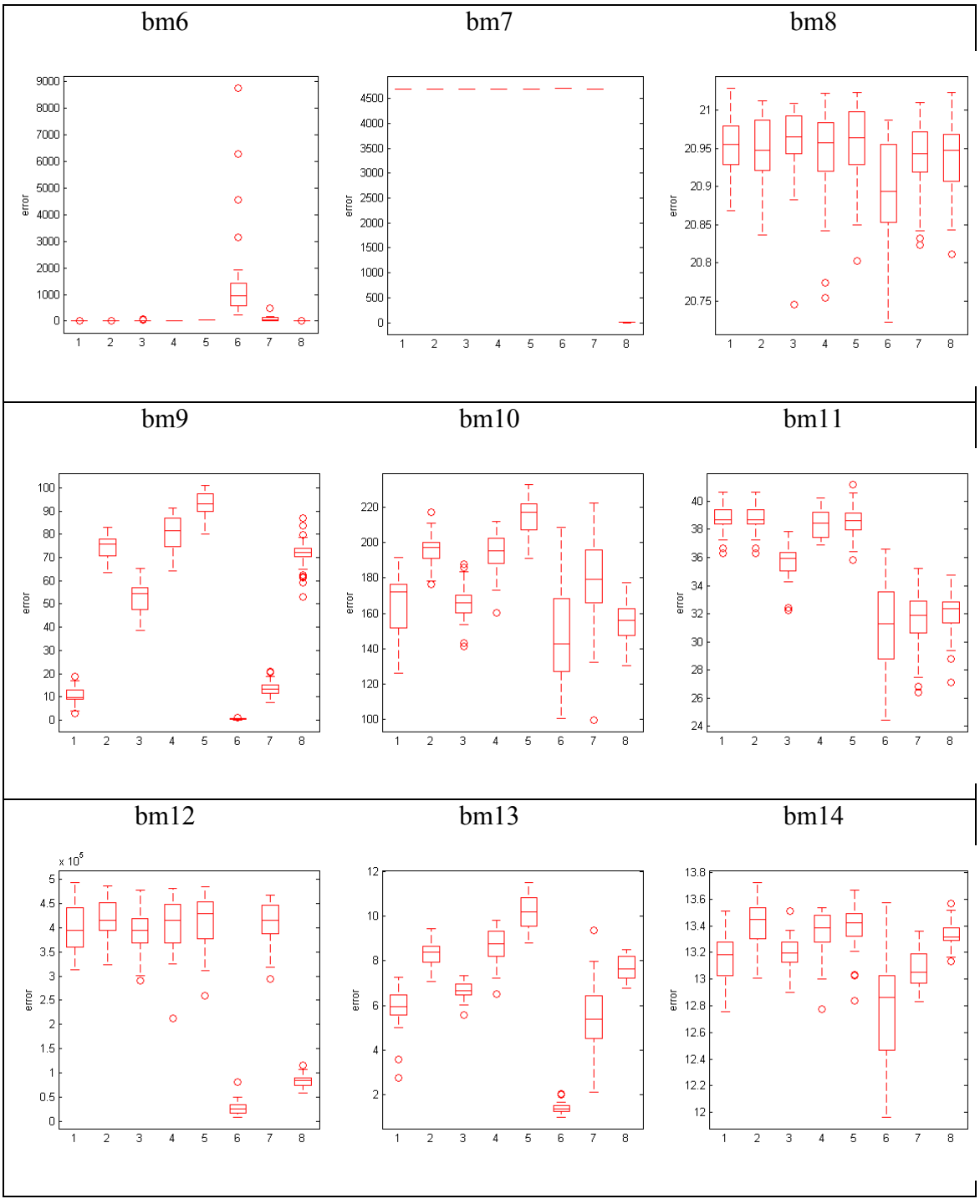
This time, the Dimension of parameter search domain is $D = 30$; the population size $NP = 10 \times D$; and the maximum generation time is $max\_gen = 1000$. The control parameters for DE is set: $F = 0.5$ and $CR = 0.3$, which is the common satisfactory setting for unimodal and multimodal functions (Storn and Price, 1997). SSO control

parameter is set Cwpgr=[0.1, 0.3, 0.5,0.1], which is the same setting as Yeh et al. experiment (Yeh et al., 2009). SSODE (1,3,5,1) uses for $gm = 0.1, cp = 0.3, gr = 0.5$, and cr $= 0.1$ (the same setting as SSO) with the $CR = 0.3$ (the same setting as DE). Even more, the custom control parameters for $gm = \frac{1}{15}, cp = \frac{2}{15}, gr = \frac{4}{15}$, and $cr = \frac{8}{15}$ with $CR = 0.3$ are selected for testing the performance of SSODE (1,2,4,8). The statistical analysis is based on minimum, maximum and mean of error values overall 30 times experiment. The results are shown in Table 14.

Table 14 boxplot results for 14 benchmark functions among different algorithms

Label	Algorithm
1	DE/Best/1
2	DE/Rand/1
3	DE/Rand-to-best/1
4	DE/best/2
5	DE/rand/2
6	SSO(1,3,5,1)
7	SSODE(1,3,5,1)
8	SSODE(1,2,4,8)





Based on these 14 benchmark functions, the results can be analysed in many ways. For SSO algorithm, it can achieve good performance in bm9, bm11, bm12, bm13, and bm14 functions. However, except bm7, bm9, bm12, and bm13, SSO has larger box size and

longer whiskers length than DE and SSODE in many benchmark functions. This indicates the best and the worst solutions discovered by SSO may have significant difference. For DE algorithm, the selection of mutation strategy does influence the DE performance. With appropriate mutation strategy, DE can often achieve satisfactory results with small variance distributions, whereas it would be suffered from local optima if the inappropriate mutation strategy is applied. In comparison, SSODE can achieve satisfactory performance in bm1, bm2, bm4, bm6, and bm7 functions. The box size is often smaller than SSO, which indicates the performance of SSODE would be steadier than SSO.

3.5 Chapter summary

SSODE provides the ability for particles to apply the suitable mutation strategies adaptively in each generation according to the output of fitness function. These two experiments indicate that within small dimensional problem (e.g. 2 parameters), the performance of SSODE is superior to SSO and is non-inferior to DE algorithm. In the large search dimension, the selection of satisfactory optimization algorithms is dependent on problem domains. Similar superior result can also be achieved based on the benchmark functions tests.

In conclusion, SSODE is a practical algorithm in certain according to the experiments results. It achieves satisfactory performance for unimodal, multimodal, separable and non-separable functions. However, the results also show that SSODE may not significantly outperform other classic optimization algorithms, such as DE and SSO. There are many factors which can affect the efficiency and effectiveness of an algorithm. For instance, hyper-parameter selections, the equations used in update mechanism, evaluation procedures, and so forth. The further study based on the notion of SSODE is represented in the next chapter.

4 MACROSCOPIC INDETERMINACY SWARM OPTIMIZATION (MISO) ALGORITHM

In the previous chapter, the SI based algorithm SSODE has been proposed. This chapter extends previous research and focuses on the update mechanism and evaluation strategy in SSODE. The new algorithm Macroscopic Indeterminacy Swarm Optimization (MISO) is proposed in this chapter.

4.1 Introduction

Swarm Intelligence (SI) has become robust methods for dealing with global optimization issues. Swarm behaviour is a collective motion of a set of self-propelled particles (O'Loan and Evans, 1999). Particles propose solutions to problem and iteratively refine them by interacting with neighbours. How efficient movement toward to global optima is important. Many well-known algorithms, such as DE(Storn and Price, 1997), PSO(Poli et al., 2007), and SSO(Yeh, 2009), proposed different update mechanisms for manipulating the particles' movements in solution space. They often mimic the phenomena in nature, such as collective behaviour, emergent behaviour, self-organization, etc.. One characteristic that all SI-based algorithms have in common is that though particles emerge from the interaction of other members, they behave chaotically and independently. Similar to the reproduction process in nature, without violating the rules of reproduction, the offspring of individuals in a population are independent and variant to one another. Another example is animal flocking phenomenon, though the individuals self-propel to unique positions at any point in time, they follow the certain interaction rules to manipulate their movements to form a group.

Although update mechanism and its formulae and equations play important roles in optimization algorithm, they do not guarantee success. Different update mechanisms model different particles movement patterns in solution space. Before the evaluation process taken place, the exact positions of particles are indeterminate. Due to the fact

that new particles often resampled with stochastic techniques, they are spreading out in space and can only be assure that they probably resided in certain regions. Similar to the microscopic indeterminacy in quantum mechanics (Wimmel, 1992). Quantum particles have both wave and particle properties known as wave-particle duality. Each quantum particle is guided by a composite of many waves. Since the measurement can only observe one of them, the momentum before the observation is uncertain. Thus, the quantum objects exist in superposition, which is the combination of all possible states, until the observation takes place.

Imagine, a SI system discovers satisfactory solutions after a number of iterations. The evaluation results are recorded and plot with line chart (quality against evaluation time interval). Then the curve in the plot can be illustrated as the movement of swarm in continuous space time. Every time the algorithm runs will plot different curves due to the stochastic method is used in the algorithm. The positions of particles before evaluation take place are indeterminate. Therefore, how individual particles move in search space cannot be directly known from the evaluation.

For the optimization problems which the characteristics of solutions are unknowable beforehand, like a black box, the observation for solution quality can only be evaluated by objective function. The aim of optimization algorithm is to find the solutions which can maximize or minimize the function outcome. E.g. for iterative method, the wanted evaluation outcome is significantly smaller or larger than the one in previous iteration time. Think differently, the algorithm is expecting the occurrence that a satisfactory solution is collected because the objective function been applied. This idea brings out a new thinking: the evaluation take place at different time will possibly gather different result. In what situation at what time, the objective function should be applied on a solution vector and probability return satisfied outcome.

All in all, SI based optimization algorithm is a method to discover the approximate ideal solutions for the given problem. Although its update mechanism is often inspired from swarm behaviour in nature, the root of matter is particles' motion behaviour in solution

space. A swarm can be considered as a macro object, which is the projection of its individual members on a micro scale. There is a relationship between the movement of particles and the motion behaviour of an entire swarm.

4.2 Inspiration

While a SI system discovers satisfactory solutions after a number of iterations, the whole progression is similar to animals' food finding behavior in nature. Individual swarm particles move toward to global optima through random walk biased by the predefined evaluation criteria and update mechanism. In macro scales, the movement of particles is continuous in solution space over iteration. Similar to the moving object which can be observed by naked eyes is in continuous space-time, which can be explained by classical mechanics. On the other hand, an object can only exist in one place in any one instant of time. In micro scale, the movement of particles consists of series of infinitesimal interval states. That is, the infinite numbers of discontinuous points in space summates to a continuous point set. Before the evaluation process taken place, the exact positions of particles are indeterminate due to new particle positions often resampled with stochastic techniques. Similar to the uncertainty principle in quantum mechanics, the precise positions of point set cannot be established without observation even if the predicted law of motion is known (Shan, 2001).

When millions of particles clumped together into a macroscopic object, classical physics such as Newton's laws can be used to describe the motion of an object. In general, the indeterminacy of quantum phenomenon still exists, albeit with little effects to macroscopic world. Nevertheless, the thought experiment Schrodinger's Cat (Gribbin, 1984) shows the indeterminacy in microscopic world may leak to the uncertain macro-state. A living cat is put into a sealed box for an hour, with a poison gas device which can be triggered by radioactive substance (atom decays). Since there is a 50% chance the radioactive substance decays within that an hour and triggers the poison gas been released, there is an equal chance the cat is either dead or alive. Without opening the

box and observe it, the cat is in superposition both dead and alive as well as the radioactive substance decayed and not decayed.

4.3 Macroscopic Indeterminacy Swarm Optimization (MISO) algorithm

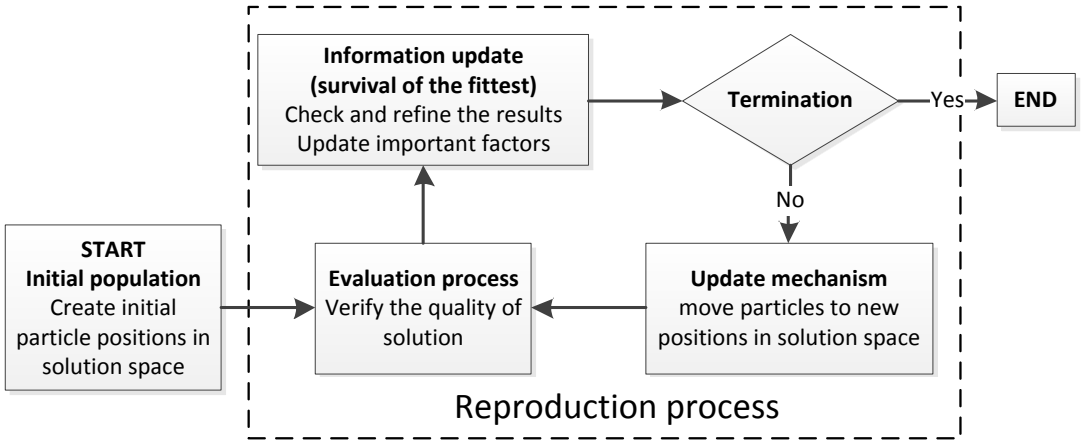


Figure 5 main stages of swarm optimization algorithm

MISO believes algorithm efficiency and effectiveness are depended on the regulatory coordination between update mechanism and evaluation process. The common steps of swarm optimization algorithm are shown in Figure 5. An update mechanism, which consists of well-designed formulae and equations, is able to fairly address the exploration and exploitation of a search space. Every time the system wants to collect and integrate useful information, such as $gbest$ or other variables required by update mechanism, the evaluation process needs to be performed. Based on the concept of survival of the fittest, only the better solution candidates can trigger the information update phase. An execution of evaluation process appears to be superfluous, if it cannot return positive feedback. However, without evaluation taking place, the quality of a

solution candidate is indeterminate. In order to improve the performance of optimization algorithm, MISO considers well-designed update mechanism and suitable evaluation strategy need to be auxiliary to each other.

The new reproduction process is proposed. The algorithm is shown as below:

Table 15 MISO algorithm

1. Initialize Population *X* with *NP* particles
2. Initial Evaluation: *eval(X)*
3. Update *pbest* and *gbest*
4. **Repeat until (*G*=0~*maxIter*)**
   a) **Reproduction process**

$iterBest = \boldsymbol{UpdateIterBest}(gbest)$

$se = \boldsymbol{UpdateSeeds}(X, iterBest)$

$re = rand(1, D) \in \mathbb{N}_1$

$for\ i = 1 \sim re\ \{$

$U = \boldsymbol{PUO}(X, se)$

$for\ j = 1 \sim D\ \{$

#*Recombination*

$$\mathrm{U_{i,j}} = \begin{cases} U_{i,j} & if\ \left(rand_{i,j} \le 0.5\right) \\ X_{i,j}^{G} & otherwise \end{cases}$$

$\}$

$\}$

   (i) **Evaluation: *eval(U)***

   (ii) **Re-constitute population $\boldsymbol{X^{G+1}}$**

$$X_i^{G+1} = \begin{cases} U_i & if\ eval(U_i) > f\left(X_{i,G}\right) \\ X_{i,G} & otherwise \end{cases}, \qquad i = 1, \ldots, NP$$

   b) **Update *gbest***

5. **END**

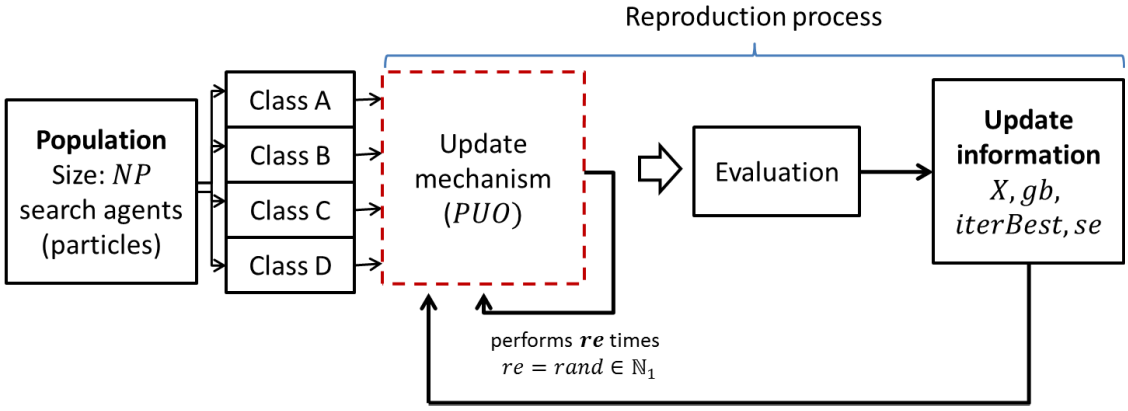


Figure 6 MISO algorithm structure

The MISO algorithm is shown inFigure 6. By extending the previous study in SSODE, the population is divided into four groups (class A to D). A new update mechanism (PUO) controls the mutation strategies of individual classes. In order to simulate the uncertain moment in time for performing evaluation, each particle executes PUO a random number of times (re) prior before evaluation takes place. More detail explanation will be shown in the following subsubsections.

4.3.1 *Crucial vectors for particle interaction*

Where $\boldsymbol{iterBest}$ and $\boldsymbol{se}$ are two crucial matrix which contains important solution vectors for emerging new solution candidates. The mechanisms are shown in (20) and (21):

Function: $\boldsymbol{UpdateIterBest}(gbest)$

IF ($eval(gbest) > eval(iterBest_{iBIdx}) * 0.95$)
  $iBIdx = iBIdx + 1;$
  $iterBest_{iBIdx} = gbest$
END (20)

The significant difference is measured by the variance interval 0.05. $\boldsymbol{gbest}$ is updated every time the better solution is discovered, whereas, $\boldsymbol{iterBest}$ only records $\boldsymbol{gbest}$ if it

is significantly better than the last element in $\boldsymbol{iterBest}$ array. This prevents the $iterBest$ array are contaminated by the latest generations' results. After many iteration times, particles would be formed convergence to local or global optima. Therefore, the diversities of particles with their neighbours are decreased and less likely to produce significantly different particles. In order to maintain a certain level diversity, a certain number of vectors are taken from $\boldsymbol{iterBest}$ in descending order and are added into $\boldsymbol{se}$.

Function: $\boldsymbol{UpdateSeeds}(X, iterBest)$

IF $(iBIdx < A_{size})$
$se_{1\sim iBIdx} = iterBest_{iBIdx\sim 1}$
$se_{iBIdx+1\sim NP} = x_{iBIdx+1\sim NP}$
ELSE
$se_{1\sim A_{size}} = iterBest_{A_{size}\sim 1}$
$se_{A_{size}+1\sim NP} = x_{A_{size}+1\sim NP}$
END (21)

$\boldsymbol{se}$ is a set of vectors which stores the important knowledge learnt from previous generations. $\boldsymbol{se}$ is composed by two types of particles:

- A number of particles from X
- A number of particles from significantly different $gbest$ which have learnt so far in $\boldsymbol{iterBest}$ array.

4.3.2 *Update mechanism: Position Updates Operator (PUO)*

Based on literature review of SSO(Yeh et al., 2009), DE(Storn and Price, 1995), PSO(Kennedy and Eberhart, 1995), and ABC(Karaboga and Basturk, 2007) algorithms, the formulae and equations they used in update mechanisms share the similar features. Table 16 lists partial equations used in these algorithms.

Table 16 some equations used in DE, PSO, ABC, and SSO

DE/best/1	$V_{i,G} = gbest + F * \left(X_{r1^i,G} - X_{r2^i,G}\right)$	(see Table 2)

PSO	$\begin{cases} V_{i,t+1} = V_{i,t} + r_1 * (pbest_{i,t} - X_{i,t}) + r_2 * (gbest - X_{i,t}) \\ X_{i,t+1} = X_{i,t} + V_{i,t+1} \end{cases}$	(see Eq. (9), (10), and (11))
ABC	$U_{i,j} = X_{i,j} + \phi_{i,j} * (X_{i,j} - X_{k,j})$	(see Eq. (15))
SSO	$X_{i,j,G} = lb_j + rand(0,1) * (ub_j - lb_j)$	(see Eq. (12))

The emergence of a new vector is calculated using vector difference. The pattern of these equations can be summarized as (22):

$$newV = Vec_0 + \sum_{i=1}^{n} amp_i * (Vec_{2i} - Vec_{2i-1}) \tag{22}$$

Where $Vec_{1\sim 2n}$ are crucial vectors for synthesis new vector $newV$ and $amp_{1\sim n}$ are the amplifiers to increase or decrease the vector differences. The idea can be interpreted by classical mechanics. The average of velocity over given time is: $velocity = \frac{displacement(S)}{during\ time\ intrval\ (\triangle t)}$. For the computer simulation of motion behavior in solution space, $\triangle t = 1$ is assigned for a change of position. The equation can be rewrite as: $S = velocity = position_{new} - position_{old}$. Accordingly, different algorithms propose different position updates mechanisms and they all implemented for reasons. No matter what theories behind their designs, the essence is the vector difference equations of displacement in solution space.

To recall the concept of SSODE, the entire swarm is partitioned into four different proportions of sub-swarms, corresponding to four different mutation strategies to synthesis new set of solution candidates. However, based on the benchmark experiments, the results (see Table 14), the performance on of SSODE is often superior to SSO(Yeh et al., 2009) and is non-inferior to DE(Storn and Price, 1995) algorithm. This indicates that SSODE framework is efficacious in dealing with many general optimization problem, but further research needs to be carried on to improve algorithm

performance. Here, the new update mechanism Position Update Operator (PUO) is proposed (see Table 17).

Table 17 update mechanism: PUO

Control parameters: na, nb, nc, nd
$na + nb + nc + nd = 1$ $M_{\mathrm{na}} = \mathrm{na}, M_{nb} = M_{\mathrm{na}} + nb\ , M_{\mathrm{nc}} = M_{\mathrm{nb}} + nc$ $A_{end} = round(M_{\mathrm{na}} * NP)$ $B_{end} = round(M_{nb} * NP)$ $C_{end} = round(M_{\mathrm{nc}} * NP)$ $D_{end} = NP$
 # Class A $F = M_{na} + rand(0,1) * (nb)$ $for\ p = 1 \sim A_{end}\ \{$ $\quad U_p = X_{r1^p,G} + F * \left(X_{r2^p,G} - se_{r1^p,G}\right)$ $\}$ ***# Class B*** $F = M_{na} + rand(0,1) * (F + nc)$ $for\ p = (A_{end} + 1) \sim B_{end}\ \{$ $\quad U_p = X_{r1^p,G} + F * \left(se_{p,G} - X_{r2^p,G}\right)$ $\}$ ***# Class C*** $F = M_{nb} + rand(0,1) * (F + nc)$ $for\ p = (B_{end} + 1) \sim C_{end}\ \{$ $\quad U_p = X_{r1^p,G} + F * \left(se_{p,G} - X_{r2^p,G}\right)$ $\}$ ***# Class D*** $F = M_{nc} + rand * \left(F + (1 - M_{nc})\right)$ $for\ p = (C_{end} + 1) \sim NP\ \{$ $\quad U_p = gbest + F * \left(se_{p,G} - X_{r1^p,G}\right)$ $\}$ * $r_{1^p}$and $r_{2^p}$are non-repeated random sequence where $r_{1^p} \neq r_{2^p} \neq p$.

Where the four ratios $\boldsymbol{na}$,$\boldsymbol{nb}$, $\boldsymbol{nc}$, and $\boldsymbol{nd}$ control the number of particles in individual four classes: Class A, Class B, Class C, and Class D. That is, 3 thresholds separate the population into four groups. The scaling rate $\boldsymbol{F}$ amplifies the differential variation between vectors which is dynamically generated according to different equations for different classes. Three equations (Table 18) are introduced in PUO and they are applied to four classes in dynamic ways.

Table 18 formulae used in PUO

Formula 1: (for class A) $U_p = X_{r1^p,G} + F * \left(X_{r2^p,G} - se_{r1^p,G}\right)$
Formula 2: (for class B & C) $U_p = X_{r1^p,G} + F * \left(se_{p,G} - X_{r2^p,G}\right)$
Formula 3: (for class D) $U_p = gbest + F * \left(se_{p,G} - X_{r1^p,G}\right)$

4.3.3 *Uncertain time-interval for performing evaluation*

- ***One hop per generation***

By reviewing many algorithms (e.g. SSO(Yeh et al., 2009), DE, PSO(Kennedy and Eberhart, 1995), and ABC(Karaboga and Basturk, 2007)), a trial solution vector is synthesised by one-round update mechanism, then immediately followed by evaluation process. That is, particles may perform only one hop before evaluation. It was more like interpreting the motion behaviour on a micro scale than mimicking the swarming behaviour in nature. According to the general presuppositions about motion in continuous space-time(Shan, 2001), the motion of particle can be illustrated as infinitesimal interval state of particle, which is the point set in time and space. Similarly, while a particle can only move one hop per generation, then it represents an

instant state of particle. The movement of one particle in solution space is the set of its positions overall generations.

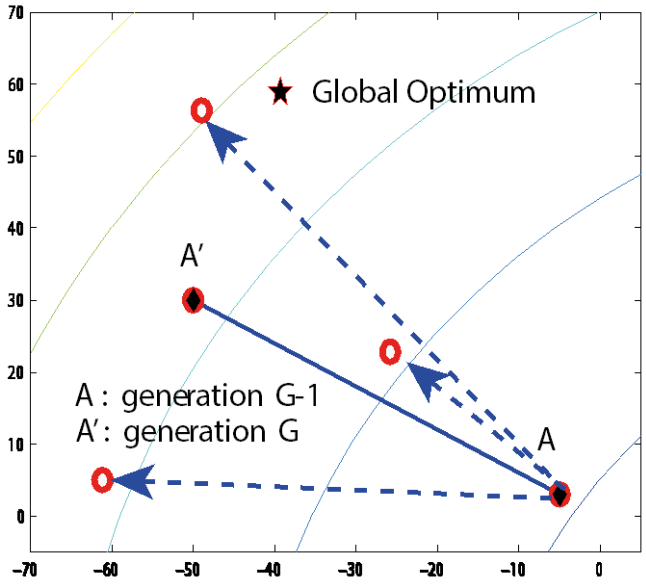


Figure 7 one hop per generation

Figure 7 illustrates a particle movement from the previous position $\boldsymbol{A}$ to current one $\boldsymbol{A'}$. Due to the fact that update mechanism stochastically update the particle's position, $\boldsymbol{A'}$ cannot be certain until the evaluation takes place. While performing the same process on the same particle position $\boldsymbol{A}$, it may appear on other position, denote as hollow point in the figure. Some hollow points may be closer to the global optimum than the actual position $\boldsymbol{A'}$.

In General, iterative methods often perform evaluation process immediately after individual particles shifting their positions. However, though the desired outcome of evaluation is locating better solution, it does not always presents positive feedback. It brings a new area to think about: is the evaluation process necessary to be applied every time particles update their position?

- ***Multi-hops per generation***

While a particle is able to perform multi-hops in one generation, it moves a number of displacements within a time interval (number of hop count). That is, in every generation, a particle performs an action instead of merely change one position in space. The entire optimization process is a goal-directed movement consist of sequencing actions. In other words, Individual particles are interacting with one another and performing random walk. The algorithm is able to randomly choose exploitation and exploration in each generation. If particles move a small number of hops away from their original positions, the new positions vectors might be near to the current ones. Else, if particles move a large number of hops, their final positions could be anywhere in solution space.

As an example shown in Figure 8, from position $\boldsymbol{A}$, the particle can arrive A' with four hops. However, during its journey toward to A', it may visit position A'_2, which is even closer to global optimum, with only two hops. If the evaluation is performed on $A_2{}'$ instead of A', the result would be better. This is the key concept of uncertain time-interval for evaluation.

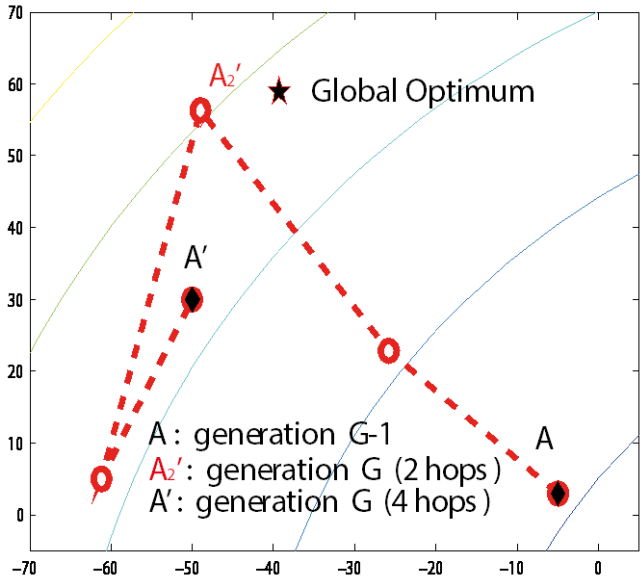


Figure 8 multi-hops per generation

From equation (23), $\boldsymbol{re}$ is randomly generated in every iteration time. It simulates the uncertain number of instants of time between current and previous evaluations performed on individual particles. That is, the simulation of uncertain time-interval for performing evaluation**.** The related code gathered from (23) is shown below

(23):

$$
\begin{aligned}
&re = rand(1, D) \in \mathbb{N}_1 \\
&for\ i = 1 \sim re\ \{ \\
&\quad U = \boldsymbol{PUO}(X, se) \\
&\quad for\ j = 1 \sim D\ \{ \\
&\quad\quad \underline{\boldsymbol{\#Recombination}} \\
&\quad\quad \mathrm{U_{i,j}} = \begin{cases} U_{i,j} & if\ \left(rand_{i,j} \leq 0.5\right) \\ X_{i,j}^{G} & otherwise \end{cases} \\
&\quad \} \\
&\}
\end{aligned}
\tag{23}
$$

In every iteration, individual particles perform $\boldsymbol{re = 1 \sim D}$ times movements and recombination to arrive their final positions. This simulates the movement of living creatures. One type of animal orientation behaviour is kinesis, which is the random change in the animal's speed or direction caused by nature stimuluses. Instead of moving with straight line, they often follow nonlinear paths to arrive their next positions, which is the moment observation performed.

4.4 Experiment

All algorithms are implemented in C language. The computer specification is: Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz with 4GB memory. The operating system is 64-bit windows 7.

The purpose of this experiment is to evaluate the performance of proposed algorithm MISO. The default MISO algorithm with no control parameter setting is compared with DE (Price and Storn, 1996, Storn and Price, 1997), ABC(Karaboga and Basturk, 2007) , SSO (Yeh, 2009, Wei-Chang, 2012, Yeh, 2013) and PSO(Kennedy and Eberhart, 1995).

Here, DE/best/1 and DE/rand/1 mutation strategies are selected. Three recommended setting for DE parameters are chosen (Storn and Price, 1997, Mallipeddi et al., 2011). The benchmark ABC source code is downloaded from ABC algorithm homepage (Karaboga, 2011). SSO control parameter is set $Cwpgr$=[0.1, 0.3, 0.5,0.1], which is the same setting as Yeh et al. experiment (Yeh et al., 2009). The setting for PSO is $c_1 = c_2 = 2.0$ with decaying weight $w = 0.9 \sim 0.4$, which is able to perform global search at the beginning and local search at the end (Shi and Eberhart, 1998b, Shi and Eberhart, 1999). The population size $NP = 10 \times D$ for all algorithms, except ABC $NP = 40$ (Karaboga, 2011). The equation (18) is applied to solve beyond boundary problem.

The experiment is carried on 28 benchmark functions from CEC 2013 Special Session on Real-Parameter Optimization (Liang et al., 2013). Table 19 and Table 20 list all 28 benchmark functions and more detail information are explained in Appendix A.

Table 19 benchmark functions

No.	Function name	fbias
Bm1	Sphere Function	-1400
Bm2	Rotated High Conditioned Elliptic Function	-1300
Bm3	Rotated Bent Cigar Function	-1200
Bm4	Rotated Discus Function	-1100
Bm5	Different Powers Function	-1000
Bm6	Rotated Rosenbrock's Function	-900
Bm7	Rotated Schaffers F7 Function	-800
Bm8	Roated Ackley's Function	-700
Bm9	Rotated Weierstrass Function	-600
Bm10	Rotated Griewank's Function	-500
Bm11	Rastrigin's Function	-400
Bm12	Rotated Rastrigin's Function	-300
Bm13	Non-Continuous Rotated Rastrigin's Function	-200
Bm14	Schwefel's Function	-100
Bm15	Rotated Schwefel's Function	100
Bm16	Rotated Katsuura Function	200
Bm17	Lunacek Bi_Rastrigin Function	300
Bm18	Rotated Lunacek Bi_Rastrigin Function	400
Bm19	Expanded Griewank's plus Rosenbrock's Function	500
Bm20	Expanded Scaffer's F6 Function	600

Table 20 composition benchmark functions

No.	Function name	elements	fbias
Bm21	Composition Function 1 (n=5,Rotated)	Bm1,Bm3,Bm4,Bm5,Bm6	700
Bm22	Composition Function 2 (n=3,Unrotated)	Bm14	800
Bm23	Composition Function 3 (n=3,Rotated)	Bm15	900
Bm24	Composition Function 4 (n=3,Rotated)	Bm9,Bm12,Bm15	1000
Bm25	Composition Function 5 (n=3,Rotated)	Bm9,Bm12,Bm15	1100
Bm26	Composition Function 6 (n=5,Rotated)	Bm2,Bm9,Bm10,Bm12,Bm15	1200
Bm27	Composition Function 7 (n=5,Rotated)	Bm1,Bm9,Bm10,Bm12,Bm15	1300
Bm28	Composition Function 8 (n=5,Rotated)	Bm1,Bm7,Bm15,Bm19,Bm20	1400

Three measure criteria are examined: 1) performance within limited Function Evaluation times, 2) algorithm complexity, and 3) overall real-time performance in seconds.

4.4.1 *Performance test in limited number of function evaluations*

The maximum function evaluation times $MaxFEs = 300{,}000$ is set for all algorithms. The main consideration for optimization algorithm is the quality of discovered solutions. The experiment is based on the CEC evaluation criteria. Assume the objective function $F(x)$, the optimum discovered by algorithm x^*, and the predefined global optimum o, and then the evaluation is based on the error value $err = eval(x) = |F(x^*) - F(o)|$. The statistical analysis is based on minimum, maximum and mean of error values, and the standard deviation of 30 applications of experiment. The detail experiment result is shown in Appendix C. According to the No Free Lunch Theorem (Wolpert and Macready, 1997), there is no algorithm can outperform all others in all problems. Therefore, the algorithm which can achieve the best performance overall given problems would be the most generalized optimization method. Overall 28 benchmark functions, MISO can achieve better performance and generalization ability than DE, ABC, SSO, and PSO algorithms in 14 benchmark functions (Table 21).

Table 21 summary of algorithms performances over 28 benchmark functions

		F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
	MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
max	13	4						11	0	0
min	13	3						8	3	1
mean	14	3						10	1	0
std	8	12						8	0	0

Table 22 The Average of the discovered results for all 28 functions

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
Error value		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
Average	max	6.86E+06	1.52E+08	4.92E+08	4.28E+08	5.26E+08	5.21E+08	1.22E+09	7.87E+07	4.78E+08	1.96E+09
	min	2.43E+04	4.41E+07	2.00E+08	9.55E+07	1.94E+08	2.67E+08	4.84E+08	8.81E+06	1.46E+07	2.81E+08
	mean	6.57E+05	8.75E+07	3.93E+08	2.44E+08	3.64E+08	3.89E+08	8.63E+08	3.06E+07	1.35E+08	9.00E+08
	std	1.86E+06	4.51E+07	1.45E+08	1.34E+08	1.66E+08	1.49E+08	3.18E+08	3.01E+07	1.86E+08	6.55E+08

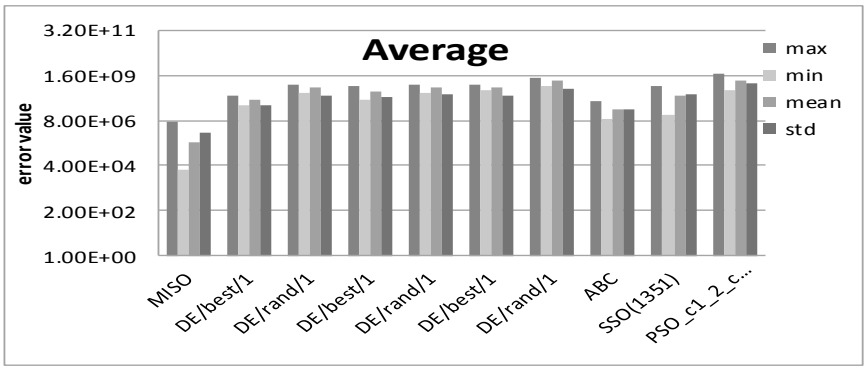


Figure 9 efficiency comparison over 28 functions: error vs. algorithm

From Table 22 and Figure 9, the average evaluation results of the worst (maximum), mean, and the best (minimum) solutions discovered by MISO are significantly smaller than other algorithms. This means even though MISO cannot achieve better results than other algorithms in some benchmark problems, it can still discover acceptable solutions in many of them. Table 23 lists all benchmark functions that MISO outperforms other algorithms.

Table 23 Summary results obtained by MISO, DE, ABC, SSO, & PSO

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
bm2	mean	**1.82E+03**	2.54E+07	3.34E+07	3.01E+07	3.03E+07	5.49E+07	6.62E+07	1.50E+07	2.85E+07	5.02E+07
	std	4.51E+03	6.24E+06	5.92E+06	6.47E+06	5.72E+06	1.12E+07	1.14E+07	3.14E+06	1.34E+07	2.14E+07
bm3	mean	**1.84E+07**	2.42E+09	1.10E+10	6.81E+09	1.02E+10	1.08E+10	2.41E+10	8.42E+08	3.76E+09	2.51E+10
	std	3.23E+07	6.78E+08	2.02E+09	2.11E+09	2.47E+09	2.01E+09	4.25E+09	5.22E+08	3.33E+09	1.08E+10
bm4	mean	**6.69E-04**	7.17E+04	7.56E+04	7.97E+04	7.75E+04	5.86E+04	6.94E+04	7.12E+04	1.28E+04	1.28E+04
	std	7.66E-04	9.06E+03	8.02E+03	1.03E+04	8.55E+03	8.78E+03	8.00E+03	8.22E+03	2.79E+03	7.50E+03
bm6	mean	**1.15E+01**	3.00E+01	1.12E+02	3.80E+01	1.04E+02	7.45E+01	3.24E+02	1.51E+01	7.14E+01	6.13E+02
	std	1.60E+01	1.25E+01	1.45E+01	7.18E+00	1.13E+01	9.73E+00	3.82E+01	2.66E+00	3.33E+01	2.42E+02
bm7	mean	**1.40E+01**	9.07E+01	1.13E+02	1.11E+02	1.13E+02	1.08E+02	1.42E+02	1.19E+02	1.17E+02	1.40E+02
	std	8.62E+00	1.06E+01	1.19E+01	1.26E+01	1.29E+01	1.03E+01	1.49E+01	1.79E+01	2.90E+01	4.55E+01
bm9	mean	**2.79E+01**	2.97E+01	2.98E+01	2.93E+01	2.93E+01	3.30E+01	3.30E+01	2.97E+01	2.92E+01	3.43E+01
	std	3.11E+00	1.35E+00	1.19E+00	1.72E+00	1.72E+00	1.18E+00	1.36E+00	1.83E+00	3.11E+00	4.23E+00
bm10	mean	**1.54E-02**	3.03E+01	1.83E+02	7.18E+01	1.54E+02	2.03E+02	6.74E+02	3.86E-01	2.58E+01	7.49E+02
	std	2.09E-02	7.27E+00	2.60E+01	1.35E+01	2.33E+01	3.00E+01	8.49E+01	1.22E-01	1.11E+01	2.43E+02
bm12	mean	**1.19E+02**	1.53E+02	2.50E+02	2.10E+02	2.47E+02	2.27E+02	3.09E+02	2.63E+02	1.40E+02	2.35E+02
	std	2.66E+01	1.79E+01	1.71E+01	2.03E+01	1.90E+01	1.39E+01	1.74E+01	3.43E+01	4.52E+01	5.44E+01
bm13	mean	**1.62E+02**	1.85E+02	2.84E+02	2.41E+02	2.78E+02	2.36E+02	3.23E+02	3.12E+02	2.16E+02	3.02E+02
	std	2.55E+01	1.63E+01	1.66E+01	1.83E+01	1.50E+01	1.55E+01	1.98E+01	3.18E+01	4.71E+01	4.91E+01
bm18	mean	**2.06E+02**	2.33E+02	3.26E+02	2.84E+02	3.33E+02	2.70E+02	3.86E+02	3.40E+02	2.57E+02	2.20E+02
	std	**1.07E+01**	1.53E+01	1.75E+01	1.69E+01	1.75E+01	1.28E+01	2.24E+01	2.82E+01	4.42E+01	5.01E+01
bm20	mean	**1.19E+01**	1.41E+01	1.40E+01	1.43E+01	1.43E+01	1.39E+01	1.39E+01	1.45E+01	1.22E+01	1.35E+01
	std	3.05E-01	3.64E-01	4.76E-01	3.65E-01	4.19E-01	2.56E-01	2.96E-01	2.80E-01	4.90E-01	8.74E-01
bm24	mean	**2.26E+02**	2.77E+02	2.86E+02	2.83E+02	2.85E+02	2.87E+02	2.96E+02	2.87E+02	2.87E+02	3.04E+02
	std	7.59E+00	5.60E+00	4.55E+00	5.75E+00	5.31E+00	4.69E+00	4.74E+00	4.96E+00	1.06E+01	1.40E+01
bm25	mean	**2.34E+02**	2.98E+02	3.03E+02	3.02E+02	3.04E+02	3.10E+02	3.15E+02	3.07E+02	3.01E+02	3.45E+02
	std	2.02E+01	4.46E+00	4.07E+00	4.25E+00	3.51E+00	3.55E+00	2.81E+00	4.30E+00	8.59E+00	1.05E+01
bm26	mean	**2.00007E+02**	2.02E+02	2.03E+02	2.03E+02	2.03E+02	2.06E+02	2.06E+02	2.0084E+02	2.43E+02	2.01E+02
	std	6.75498E-03	5.46E-01	6.15E-01	5.83E-01	6.69E-01	1.36E+00	1.50E+00	2.0228E-01	7.57E+01	9.95E-01

To sum up, MISO would be the recommended method for solving black-box optimization problem due to it can achieve the satisfactory results within limited function evaluation times (FEs).

4.4.2 *Algorithm complexity measurement*

The Performance measurement based on the solution quality within $MaxFEs$ is not always a fair approach way for testing algorithm. One of the key issues is algorithmic complexity, which commonly refers to the amount of time for an algorithm to run. For

solving the same problems within $MaxFEs$, some algorithms finish earlier than others. Based on the evaluation criteria for algorithm complexity in CEC 2013 Special Session (Liang et al., 2013), the results is shown in Table 24.

Table 24 summary of algorithm time complexity over 28 benchmark functions

Time (sec)	Miso	F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		ABC	SSO	PSO
		DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1			
mean	1.48E+01	3.46E+00	3.31E+00	3.22E+00	4.10E+00	3.79E+00	3.35E+00	3.35E+00	1.03E+01	8.51E+00
sum	4.13E+02	9.69E+01	9.27E+01	9.03E+01	1.15E+02	1.06E+02	9.37E+01	9.38E+01	2.87E+02	2.38E+02

The experiment result shows MISO has the highest algorithm complexity, which means MISO takes the longest time to run. It is reasonable because every iteration MISO executes the update mechanism multiple times before an evaluation takes place. However, inefficiency does not always mean to be ineffectiveness. In the next section, the real-time analysis is presented for verifying the true convergence performance based on the benchmark functions.

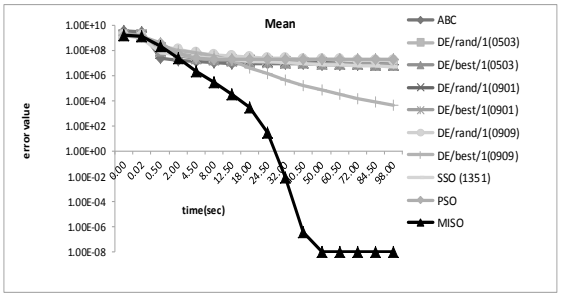
4.4.3 *Real-time performance measurement*

The function error value is recorded in varying pause time $T = \{t_i\}$, $t_i = 0.5 * (i)^2, i = 0 \sim 14$ (see Table 25).

Table 25 sequence of seconds

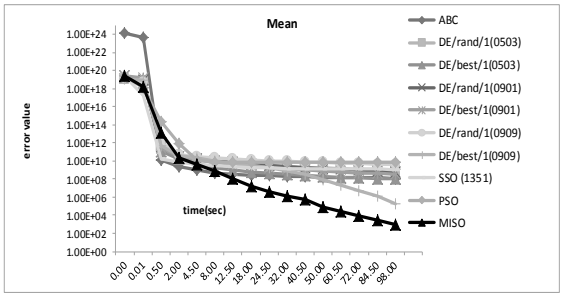
Time (sec) $t_i = 0.5 \times i^2$												
0.5	2.0	4.5	8.0	12.5	18.0	24.5	40.5	50.0	60.5	72.0	84.5	98.0

The statistical analysis is based on of error values of 30 applications of experiment. The experiment results are shown in Appendix D. Here, we display the convergence graphs for some descriptive benchmark functions (See Figure 10 ~21). In the graphs, the solid bold triangle line represents MISO algorithm. If error value ≤ 1.00E-08, than set it to 1.00E-08.



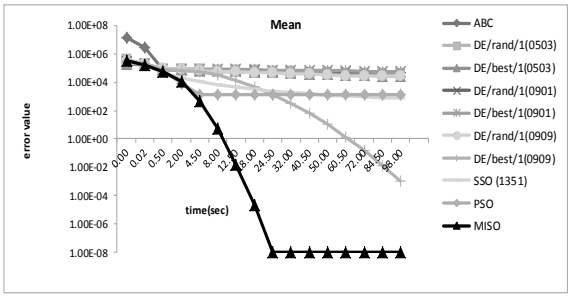
Bm2

Figure 10 Rotated High Conditioned Elliptic Function



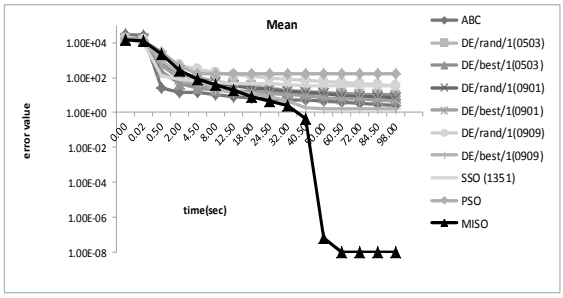
Bm3

Figure 11 Rotated Bent Cigar Function



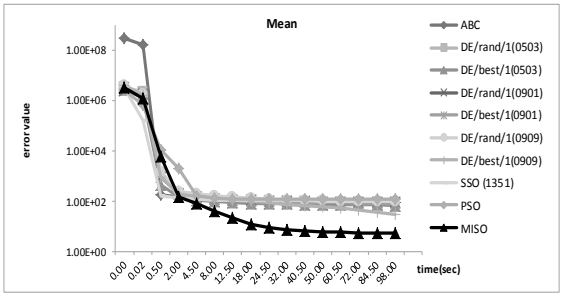
Bm4

Figure 12 Rotated Discus Function



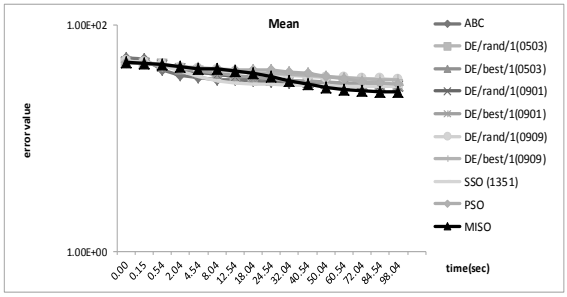
Bm6

Figure 13 Rotated Rosenbrock's Function



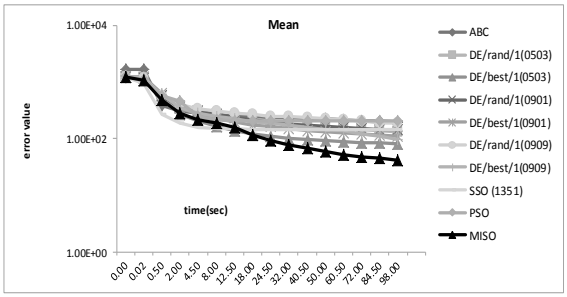
Bm7

Figure 14 Rotated Schaffers F7 Function



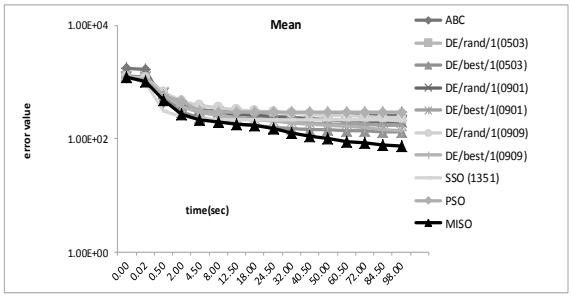
Bm9

Figure 15 Rotated Weierstrass Function



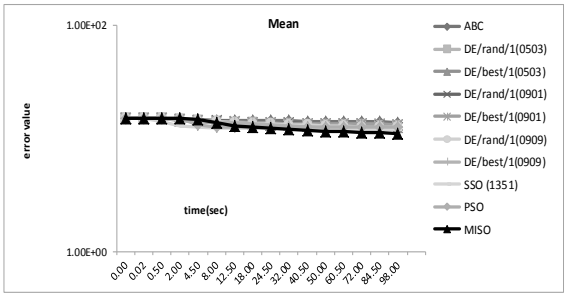
Bm12

Figure 16 Rotated Rastrigin's Function



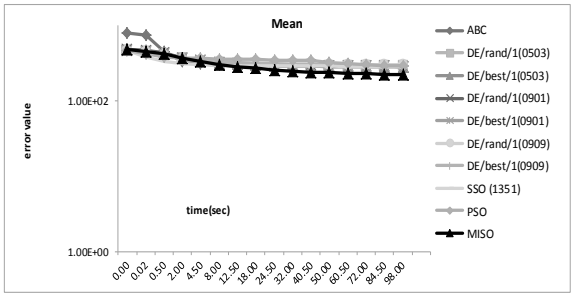
Bm13

Figure 17 Non-Continuous Rotated Rastrigin's Function



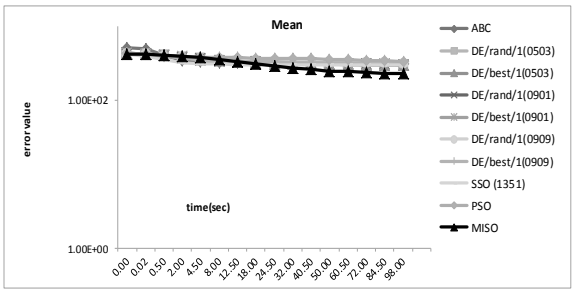
Bm20

Figure 18 Expanded Scaffer's F6 Function



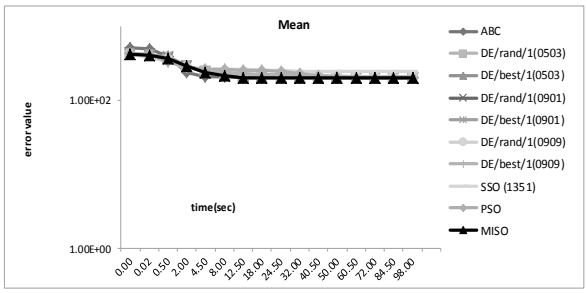
Bm23

Figure 19 Composition Function 3



Bm25

Figure 20 Composition Function 5



Bm26

Figure 21 Composition Function 6

The graphs indicate that many algorithms like SSO, DE, ABC, and PSO are able to form fast convergence at the beginning. However, they may have premature optimization issue and suffer from local optima. Besides, as it can be seen from the experiment results of benchmarks: bm2, bm3, bm4, bm6, bm7 & bm13. After algorithms running a period of time, the performance of MISO is gradually improving. Finally, MISO achieves either fast convergence, or discovers the better optimal solutions than others. From the boxplot graphs (see Appendix D: Boxplot section), within approximate 98 seconds, MISO is able to perform effective search in many benchmark problems. Except the benchmark functions bm14, bm15, bm16, bm17, bm22 & bm23, MISO is able to discover satisfactory results with small standard deviation range.

After all, according to these three experiments, we can conclude that an algorithm with high time complexity can be complemented by small number of function evaluation times. Despite of the fact that every iteration MISO takes longer time to execute, it can

achieve satisfactory result within less number of function evaluations. Consuming lots of time in every iteration might pay off in the end.

4.4.4 *Chapter summary*

The update mechanism, which is a strategy or set of steps and equations to produce new solutions, plays important role for discovering new solution candidates. Many representative algorithms such as SSO, DE, PSO, and ABC, have been proposed to solve global optimization problems. Their ideas were assured to solve the optimization problems based on their hypotheses. However, they do not guarantee success. Even more, they seem to be too ambitious, and try to synthesis better solution candidates in every iteration.

Macroscopic Indeterminacy Swarm Optimization (MISO) algorithm is proposed in this chapter. In order to improve the performance of optimization algorithm, MISO considers well-designed update mechanism and suitable evaluation strategy need to be auxiliary to each other. Macroscopic Indeterminacy refers to biased random walk phenomena in nature. Lives find ways out to survive. Living organisms constantly change their positions in response to the stimuli from colleagues and their environment. Similarly, particle agents constantly change their positions in response to the stimuli from colleagues, and frequently response to conditions (solution quality) in solution space. That is, particles may update their positions multiple times before evaluation take place.

Based on benchmark functions provided by CEC conference (Liang et al., 2013), the experiments show MISO is superior SSO, DE, PSO, and ABC in many of them. In spite of the fact that algorithm complexity for MISO is significantly higher than others, it can achieve good results within small number of function evaluations. By given enough of processing time, MISO is able to come from behind and achieve efficient and effective

results in many benchmark problems. The overall performance of MISO is significantly better than others.

5 CONCLUSION AND FINAL REMARKS

This thesis has demonstrated the research of SI based system for solving global optimization problem. Before solving any problem, it is essential to understand and formulize a problem into objective function. Then the second consideration is the method for minimizing or maximizing the outcome(s) of objective function(s). The literature review is demonstrated in chapter 2. By reviewing existing SI-based algorithms, many of them focus on implementing new formulae and updating equations to achieve emerging new solutions. It is believe the appropriate equations can efficiently and effectively form convergence to global optima.

However, it is not practical to design an algorithm which is uniformly best in all possible problems due to No Free Lunch Theorem. An algorithm may be well suited for solving particular types of problems, but inappropriate for others. Hybridization of different algorithms' operators and incorporation of their knowledge would complement one another's strength and weakness. In Chapter 3, the new hybrid evolutionary algorithm SSODE is proposed. SSODE uses four update formulae which are summarized from many well-known mutation strategies from DE. Inspired from SSO algorithm structure, these four update formulae are dynamically applied for synthesis new solution candidates. The experiment results show that the general efficiency of SSODE is better than SSO and DE. Although the performance of SSODE may not be superior to SSO and DE, SSODE is able to present similar results in many benchmark problems.

In chapter 4, the further research is carried on SSODE and new algorithm MISO is proposed. There are two research focuses. One is to improve the efficiency of SSODE. A well-defined update mechanism can speed up the convergence while avoiding local optima. The other is to present a new schema for evaluation process. Inspired from phenomena in nature such as living creatures find ways out to survive or quantum

particles movement in space, everything in the universe is considered to be moving. Evaluation or observation is the moment that a system interacts or reflects to the environment. The moment for performing evaluation can be considered as sudden occurrences to make decisions and conclusions. That is, it seems to be unnecessary to evaluate a particle every time it moves one position.

In addition, the time complexity is of an algorithm is the amount of the time it took to run. In general, the most effective algorithm is the one which takes the shortest time to execute. However, effectiveness and efficiency do not always equate. Some algorithms can complete all processing steps in a short time, but they may not achieve good results. In comparison, a high algorithm complexity method often takes longer time to run, but if it can achieve the similar or even better results with less processing steps, it may meet the termination requirement earlier than others. The implication is that by given a certain period time, an algorithm which can achieve the more accurate results will be more preferable one to use.

5.1 Contributions

The thesis offers several contributions to swarm intelligence in optimization algorithm research. First, a multi-groups hybrid strategies algorithm is proposed. Particles (a set of search agents) are partitioned into different groups. Based on the special designed algorithms, different mutation strategies are applied stochastically. Secondly, a new update mechanism for generating new solution candidates is proposed. A well-designed position update mechanism is able to provide both global and local search abilities. Moreover, the selection of parent vectors for emerging new solution candidates is a key factor affecting algorithm performance. Apart from interacting with neighbouring particle vectors, some of the best solutions discovered in previous iterations provide important information. Finally, the moment to evaluate new solution candidates is another key factor affecting algorithm effectiveness, especially for iterative method. While a method that possible solves problems, then the matter is when to verify

discovered solutions. In the proposed hybrid algorithm MISO, the position update mechanism (PUO) dynamically applied different search strategies for different groups of particles. The evaluation process is performed after uncertain time of PUO has been executed for each individual particle.

6 BIBLIOGRAPHY

BAKER JR, R. B. & BAKER JR, R. 2001. *Who Said That?*, Writers Club Press.

BEYER, K., GOLDSTEIN, J., RAMAKRISHNAN, R. & SHAFT, U. 1999. When is "nearest neighbor" meaningful? *Database Theory—ICDT'99.* Springer.

BLAKE, C. & MERZ, C. J. 1998. *UCI Repository of machine learning databases* [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.htm.

BLUM, C. & SAMPELS, M. 2004. An ant colony optimization algorithm for shop scheduling problems. *Journal of Mathematical Modelling and Algorithms,* 3**,** 285-308.

BOYD, S. & VANDENBERGHE, L. 2004. *Convex optimization*, Cambridge university press.

BRINDLE, A. 1981. Genetic algorithms for function optimization.

BURGES, C. J. C. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery,* 2**,** 121-167.

CHAKRABORTY, U. K. 2008. *Advances in differential evolution*, Springer Verlag.

CHANG, C. C. & LIN, C. J. 2011. LIBSVM: A Library for support vector machines. *ACM Transactions on Intelligent Systems and Technology,* 2.

CLERC, M. & KENNEDY, J. 2002. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on,* 6**,** 58-73.

CODLING, E. A., PLANK, M. J. & BENHAMOU, S. 2008. Random walk models in biology. *Journal of the Royal Society Interface,* 5**,** 813-834.

DONG SEONG, K., HA-NAM, N. & JONG SOU, P. Genetic algorithm to improve SVM based network intrusion detection system. Advanced Information Networking and Applications, 2005. AINA 2005. 19th International Conference on, 28-30 March 2005 2005. 155-158 vol.2.

DORIGO, M., BIRATTARI, M. & STUTZLE, T. 2006. Ant colony optimization. *Computational Intelligence Magazine, IEEE,* 1**,** 28-39.

DORIGO, M. & BLUM, C. 2005. Ant colony optimization theory: A survey. *Theoretical Computer Science,* 344**,** 243-278.

DORIGO, M. & GAMBARDELLA, L. M. 1997. Ant colonies for the travelling salesman problem. *Biosystems,* 43**,** 73-81.

EBERHART, R. C. & SHI, Y. 2004. Guest Editorial Special Issue on Particle Swarm Optimization. *Evolutionary Computation, IEEE Transactions on,* 8**,** 201-203.

GRIBBIN, J. 1984. *In Search of Schrodinger's Cat: Quantam Physics And Reality*, Bantam.

GROSAN, C. & ABRAHAM, A. 2007. Hybrid evolutionary algorithms: methodologies, architectures, and reviews. *Hybrid evolutionary algorithms.* Springer.

HAMEKA, H. F. 2004. *Quantum mechanics: a conceptual approach*, Wiley-Interscience.

HANSEN, N., FINCK, S., ROS, R. & AUGER, A. 2009. Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions.

IORIO, A. & LI, X. 2005. Solving rotated multi-objective optimization problems using differential evolution. *AI 2004: Advances in Artificial Intelligence***,** 861-872.

KARABOGA, D. 2011. *Artificial bee colony (ABC) algorithm homepage* [Online]. Available: http://mf.erciyes.edu.tr/abc/publ.htm.

KARABOGA, D. & BASTURK, B. 2007. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization,* 39**,** 459-471.

KENNEDY, J. 2006. Swarm Intelligence

Handbook of Nature-Inspired and Innovative Computing. *In:* ZOMAYA, A. (ed.). Springer US.

KENNEDY, J. & EBERHART, R. Particle swarm optimization. Neural Networks, 1995. Proceedings., IEEE International Conference on, Nov/Dec 1995 1995. 1942-1948 vol.4.

LAMPINEN, J. & ZELINKA, I. 2000. On stagnation of the differential evolution algorithm.

LI, X., TANG, K., OMIDVAR, M. N., YANG, Z., QIN, K. & CHINA, H. 2013. Benchmark Functions for the CEC'2013 Special Session and Competition on Large-Scale Global Optimization. *gene,* 7**,** 33.

LIANG, J., QU, B. & SUGANTHAN, P. 2013. Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization.

MALLIPEDDI, R., SUGANTHAN, P., PAN, Q. & TASGETIREN, M. 2011. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing,* 11**,** 1679-1696.

MANIEZZO, V. & COLORNI, A. 1999. The ant system applied to the quadratic assignment problem. *Knowledge and Data Engineering, IEEE Transactions on,* 11**,** 769-778.

MOSCATO, P., COTTA, C. & MENDES, A. 2004. Memetic algorithms. *New optimization techniques in engineering.* Springer.

O'LOAN, O. & EVANS, M. 1999. Alternating steady state in one-dimensional flocking. *Journal of Physics A: Mathematical and General,* 32**,** L99.

PAPADIMITRIOU, C. H. & STEIGLITZ, K. 1998. *Combinatorial optimization: algorithms and complexity*, Courier Dover Publications.

POLI, R., KENNEDY, J. & BLACKWELL, T. 2007. Particle swarm optimization. *Swarm intelligence,* 1**,** 33-57.

PRICE, K. V. & STORN, R. M. 1996. *Differential Evolution Homepage* [Online]. International Computer Science Institute (ICSI). Available: http://www1.icsi.berkeley.edu/~storn/code.html.

QIN, A., HUANG, V. & SUGANTHAN, P. 2009. Differential evolution algorithm with strategy adaptation for global numerical optimization. *Evolutionary Computation, IEEE Transactions on,* 13**,** 398-417.

REIMANN, M., DOERNER, K. & HARTL, R. F. 2004. D-Ants: savings based ants divide and conquer the vehicle routing problem. *Computers & Operations Research,* 31**,** 563-591.

REINELT, G. 1994. *The traveling salesman: computational solutions for TSP applications*, Springer-Verlag.

SHAN, G. 2001. From quantum motion to classical motion-seeking the lost reality. *Physics Essays,* 14**,** 37-48.

SHANG, Y.-W. & QIU, Y.-H. 2006. A note on the extended Rosenbrock function. *Evolutionary Computation,* 14**,** 119-126.

SHI, Y. & EBERHART, R. A modified particle swarm optimizer. Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on, 1998a. IEEE, 69-73.

SHI, Y. & EBERHART, R. Parameter selection in particle swarm optimization. Evolutionary Programming VII, 1998b. Springer, 591-600.

SHI, Y. & EBERHART, R. C. Empirical study of particle swarm optimization. Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, 1999. IEEE.

STORN, R. On the usage of differential evolution for function optimization. 1996. IEEE, 519-523.

STORN, R. & PRICE, K. 1995. Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. *International computer science institute-publications-TR*.

STORN, R. & PRICE, K. 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization,* 11**,** 341-359.

SUGANTHAN, P. N., HANSEN, N., LIANG, J. J., DEB, K., CHEN, Y., AUGER, A. & TIWARI, S. 2005. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *KanGAL Report,* 2005005.

TAN, P.-N. 2007. *Introduction to data mining*, Pearson Education India.

TANG, K., YAO, X., SUGANTHAN, P. N., MACNISH, C., CHEN, Y. P., CHEN, C. M. & YANG, Z. 2007. Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. *Nature Inspired Computation and Applications Laboratory, USTC, China*.

VAPNIK, V. N. 1995. *The Nature of Statistical Learning Theory,* NY, Springer Verlag.

WEI-CHANG, Y. 2012. Optimization of the Disassembly Sequencing Problem on the Basis of Self-Adaptive Simplified Swarm Optimization. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* 42**,** 250-261.

WEISE, T. 2009. Global optimization algorithms–theory and application. *Self-Published*.

WIMMEL, H. 1992. *Quantum physics & observed reality: a critical interpretation of quantum mechanics*, World Scientific Publishing Company Incorporated.

WITTEN, I. H., FRANK, E. & HALL, M. A. 2011. *Data Mining: Practical machine learning tools and techniques*, Morgan Kaufmann.

WOLPERT, D. H. & MACREADY, W. G. 1997. No free lunch theorems for optimization. *Evolutionary Computation, IEEE Transactions on,* 1**,** 67-82.

YEH, W.-C. 2013. New Parameter-Free Simplified Swarm Optimization for Artificial Neural Network Training and Its Application in the Prediction of Time Series.

YEH, W. C. 2009. A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems. *Expert Systems with Applications,* 36**,** 9192-9200.

YEH, W. C., CHANG, W. W. & CHUNG, Y. Y. 2009. A new hybrid approach for mining breast cancer pattern using discrete particle swarm optimization and statistical method. *Expert Systems with Applications,* 36**,** 8204-8211.

ZHANG, W.-J., XIE, X.-F. & BI, D.-C. Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space. Evolutionary Computation, 2004. CEC2004. Congress on, 2004. IEEE, 2307-2311.

APPENDIX A

Benchmark functions

The benchmark functions used in this paper are presented by Congress of evolutionary computation (Tang et al., 2007, Liang et al., 2013)

$\boldsymbol{f(x)}$	Objective function
$\boldsymbol{D}$	Dimension of solution space
$\boldsymbol{x}$	Decision vector/solution vector/position vector in solution space $x \in \mathbb{R}^D, x = [x_1, x_2, \dots, x_D], x \in [lb, ub]^D$
$\boldsymbol{o}$	Shifted distance vector $o = [o_1, o_2, \dots, o_D]$
$\boldsymbol{z}$	Shifted solution vector $z = [z_1, z_2, \dots, z_D], z = x - o,\ z \in [lb, ub]^D$
$\boldsymbol{f_{bias}}$	the bias in the function value
$\boldsymbol{T_{osz}(x_i)}$	Transformation function for creating smooth local irregularities (Hansen et al., 2009) $T_{osz}(x_i) = sign(x_i) \times \exp\left(\widehat{x_i} + 0.049 \times (\sin(c_1\widehat{x_i}) + \sin(c_2\widehat{x_i}))\right)$ $for\ i = 1, \dots, D$ $where\ \widehat{x_i} = \begin{cases} \log(\lvert x_i \rvert), & x \neq 0 \\ 0, & otherwise \end{cases}, \ sign(x_i) = \begin{cases} -1, & x_i < 0 \\ 0, & x_i = 0 \\ 1, & x_i > 0 \end{cases}$ $c_1 = \begin{cases} 10, & x_i > 0 \\ 5.5, & x_i \leq 0 \end{cases}, \quad c_2 = \begin{cases} 7.9, & x_i > 0 \\ 3.1, & x_i \leq 0 \end{cases}$
$\boldsymbol{T_{asy}^{\beta}}$	Transformation function to break the symmetry of the functions (Hansen et al., 2009). $T_{asy}^{\beta}: \mathbb{R}^D \to \mathbb{R}^D, \ x_i = \begin{cases} x_i^{1+\beta\frac{t-1}{D-1}\sqrt{x_i}}, & if\ x_i > 0 \\ x_i, & otherwise \end{cases} for\ i = 1, \dots, D$
$\boldsymbol{M_1, M_2, \dots, M_{10}}$	Rotation matrix generated from standard normally distributed entries

	by Gram-Schmidt orthonormalization.
$\boldsymbol{\Lambda^{\alpha}}$	D-dimensional diagonal matrix with diagonal elements $\lambda_{ii} = \alpha^{\frac{1}{2}\times\frac{i-1}{D-1}}$. For creating ill-conditioning (Hansen et al., 2009).

Shifted Sphere Function

$$f(x) = \sum_{i=1}^{D} z_i^2 + f_{bias}$$

Schwefel's Function

$$f(z) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i) + f_{bias}$$

$$z = \Lambda^{10}\left(\frac{1000(x-o)}{100}\right) + 4.209687462275036e{+}002$$

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

Rotated Schwefel's Function

$$f(z) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i) + f_{bias}$$

$$z = \Lambda^{10} M_1\left(\frac{1000(x-o)}{100}\right) + 4.209687462275036e{+}002$$

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

Shifted Schwefel's Problem 1.2

$$f(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2 + f_{bias}$$

Shifted Schwefel's Problem 1.2 with Noise in Fitness

$$f(x) = \left(\sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2 \right) \times (1 + 0.4 \times |rand(0,1)|) + f_{bias}$$

Schwefel's Problem 2.6 with Global Optimum on Bounds

For 2-dimensional problem

$$f(x) = \max\{|x_1 + 2x_2 - 7|, |2x_1 + x_2 - 5|\}$$

For D-dimensional problem

$$f(x) = \max\{|A_i x - B_i|\} + f_{bias}, i = 1, \dots, D$$

$$A = \{a_{i,j}\}, i = j = D, a_{i,j} \in \mathbb{Z} \in [-500,500], \ \det(A) \neq 0$$

$$B_i = A_i \times o$$

Schwefel's Problem 2.13

$$f(x) = \sum_{i=1}^{D} (A_i - B_i(x))^2 + f_{bias}$$

$$A_i = \sum_{j=1}^{D} (a_{ij} sin\ \alpha_j + b_{ij} cos\ \alpha_j)$$

$$B_i(x) = \sum_{j=1}^{D} (a_{ij} \sin x_j + b_{ij} \cos x_j), for\ i = 1, \dots, D$$

$A, B\ are\ D \times D\ matrix\ , a_{ij}, b_{ij}\ are\ random\ integer\ in\ [-100,100]$

$\alpha = [\alpha_1, \dots, \alpha_D], are\ random\ number\ in\ [-\pi, \pi]$

Schwefel's Problem 2.21

$$f(x) = \max_i \{|z_i|, 1 \leq i \leq D\} + f_{bias}$$

Shifted Elliptic Function

$$f(x) = f(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} \times z_i^2 + f_{bias}$$

Rotated High Conditioned Elliptic Function

$f(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} \times z_i^2 + f_{bias}\,,\ \ z = T_{osz}\big(M_1(x-o)\big)$
Rotated Rosenbrock's Function
$f(x) = \sum_{i=1}^{D-1} (100 \times (z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + f_{bias}$ $z = M_1\left(\frac{2.048(x-o)}{100}\right) + 1$
Shifted Rosenbrock's Function
$f(x) = \sum_{i=1}^{D-1} (100 \times (z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + f_{bias}$
Rotated Griewank's Function
$f(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f_{bias},\ \ z = \Lambda^{100} M_1 \frac{600(x-o)}{100}$
Shifted Rotated Griewank's Function without Bounds
$f(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f_{bias},\ \ z = (x-o) \times M$ M': linear transformation matrix, condition number 3 $M = M'(1 + 0.3 \times \lvert N(0,1) \rvert)$
Rotated Expanded Griewank's plus Rosenbrock's Function
Basic Griewank's function: $g_1(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1$ Basic Rosenbrock's function: $g_2(x) = \sum_{i=1}^{D-1} (100 \times (z_i^2 - z_{i+1})^2 + (z_i - 1)^2)$ $f(x) = g_1\big(g_2(z_1, z_2)\big) + g_1\big(g_2(z_2, z_3)\big) + \cdots + g_1\big(g_2(z_{D-1}, z_D)\big) + g_1\big(g_2(z_D, z_1)\big) + f_{bias}$ $z = M_1\left(\frac{5(x-o)}{100}\right) + 1$
Rotated Ackley's Function

$$f(x) = -20 \times \exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20 + e + f_{bias}$$

$$z = \Lambda^{10} M_2 T_{asy}^{0.5}\left(M_1(x - o)\right)$$

Shifted Rotated Ackley's Function with Global Optimum on Bounds

$$f(x) = -20 \times \exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20 + e + f_{bias}$$

$$z = (x - o) \times M, \qquad o_{2j-1} = -32 o_{2j}, \qquad j = 1,2,\ldots,\left\lfloor\frac{D}{2}\right\rfloor$$

$$M: linear\ transformation\ matrix\ condition\ number = 100$$

Shifted Rastrigin's Function (Suganthan et al., 2005)

$$f(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias}, \qquad z = x - o$$

Shifted Rotated Rastrigin's Function (Suganthan et al., 2005)

$$f(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias}, \qquad z = (x - o) \times M$$

Rastrigin's Function

$$f(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias}$$

$$z = \Lambda^{10} T_{asy}^{0.2}\left(T_{osz}\left(\frac{5.12(x - o)}{100}\right)\right)$$

Rotated Rastrigin's Function

$$f(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias}$$

$$z = M_1 \Lambda^{10} M_2 T_{asy}^{0.2}\left(T_{osz}\left(M_1 \frac{5.12(x - o)}{100}\right)\right)$$

Lunacek Bi_Rastrigin Function

$$f(x) = \min\left(\sum_{i=1}^{D}(\hat{x}_i - \mu_0)^2, dD + s\sum_{i=1}^{D}(\hat{x}_i - \mu_1)^2\right) + 10\left(D - \sum_{i=1}^{D}\cos(2\pi\hat{z}_i)\right) + f_{bias}$$

$$\mu_0 = 2.5, \mu_1 = -\sqrt{\frac{\mu_0^2 - d}{s}}, s = 1 - \frac{1}{2\sqrt{D+20} - 8.2}, d = 1, y = \frac{10(x-o)}{100}$$

$$\hat{x}_i = 2sign(x_i)y_i + \mu_0, for\ i = 1, \dots, D, z = \Lambda^{100}(\hat{x} - \mu_0)$$

Rotated Lunacek Bi_Rastrigin Function

$$f(x) = \min\left(\sum_{i=1}^{D}(\hat{x}_i - \mu_0)^2, dD + s\sum_{i=1}^{D}(\hat{x}_i - \mu_1)^2\right) + 10\left(D - \sum_{i=1}^{D}\cos(2\pi\hat{z}_i)\right) + f_{bias}$$

$$\mu_0 = 2.5, \mu_1 = -\sqrt{\frac{\mu_0^2 - d}{s}}, s = 1 - \frac{1}{2\sqrt{D+20} - 8.2}, d = 1, y = \frac{10(x-o)}{100}$$

$$\hat{x}_i = 2sign(x_i)y_i + \mu_0, for\ i = 1, \dots, D, z = \mathrm{M}_2\Lambda^{100}(M_1\hat{x} - \mu_0)$$

Non-Continuous Rotated Rastrigin's Function

$$f(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias}$$

$$\hat{x} = M_1\frac{5.12(x-o)}{100}, y_i = \begin{cases} \hat{x}, & if\ |\hat{x}_i| \le 0.5 \\ round\dfrac{(2\hat{x}_i)}{2}, & if\ |\hat{x}_i| > 0.5 \end{cases} for\ i = 1, \dots, D$$

$$z = M_1\Lambda^{10}M_2T_{asy}^{0.2}\left(T_{osz}(y)\right)$$

Rotated Schaffers F7 Function

$$f(x) = \left(\frac{1}{D-1}\sum_{i=1}^{D-1}\left(\sqrt{z_i} + \sqrt{z_i}\sin^2(50z_i^{0.2})\right)\right)^2 + f_{bias}$$

$$z_i = \sqrt{y_i^2 + y_{i+1}^2}\ for\ i = 1, \dots, D, \qquad y = \Lambda^{10}M_2T_{asy}^{0.5}\left(M_1(x-o)\right)$$

Rotated Bent Cigar Function

$$f(x) = z_1^2 + 10^6\sum_{i=2}^{D}z_i^2 + f_{bias}, \qquad z = M_2T_{asy}^{0.5}\left(M_1(x-o)\right)$$

Rotated Discus Function

$f(x) = 10^6 z_1^2 + \sum_{i=2}^{D} z_i^2 + f_{bias}, \qquad z = T_{osz}\left(M_1(x-o)\right)$
Different Powers Function
$f(x) = \sqrt{\sum_{i=1}^{D} \lvert z_i \rvert^{2+4\frac{i-1}{D-1}}} + f_{bias}, \qquad z = x - o$
Rotated Weierstrass Function
$f(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{kmax}\left[a^k cos\left(2\pi b^k(z_i+0.5)\right)\right]\right) - D\sum_{k=0}^{kmax}[a^k \cos(2\pi b^k * 0.5)] + f_{bias}$ $a = 0.5, b = 3, kmax = 20, \qquad z = \Lambda^{10} M_2 T_{asy}^{0.5}\left(M_1 \frac{0.5(x-o)}{100}\right)$
Shifted Rotated Weierstrass Function
$f(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{kmax}\left[a^k cos\left(2\pi b^k(z_i+0.5)\right)\right]\right) - D\sum_{k=0}^{kmax}[a^k \cos(2\pi b^k * 0.5)] + f_{bias}$ $a = 0.5, b = 3, kmax = 20, \qquad z = (x-o) * M$
Rotated High Conditioned Elliptic Function
$f(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} \times z_i^2 + f_{bias}, \qquad z = T_{osz}\left(M_1(x-o)\right)$
Shifted Rotated High Conditioned Elliptic Function (Suganthan et al., 2005)
$f(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} \times z_i^2 + f_{bias}, \qquad z = (x-o) * M$
Rotated Katsuura Function
$f(x) = \frac{10}{D^2}\prod_{i=1}^{D}\left(1 + i\sum_{j=1}^{32}\frac{\lvert 2^j z_i - round(2^j z_i)\rvert}{2^j}\right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} + f_{bias}$ $z = M_2 \Lambda^{100}\left(M_1 \frac{5(x-o)}{100}\right)$

Rotated Expanded Scaffer's F6 Function

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

$$f(x) = g(z_1,z_2) + g(z_2,z_3) + \cdots + g(z_{D-1},Z_D) + g(z_D,z_1) + f_{bias}$$

$$z = M_2(T_{asy}^{0.5}\left(M_1(x-o)\right)$$

Shifted Expanded Griewank's plus Rosenbrock's Function (F8F2)

Griewank's Function: $F_8(x) = \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$

Rosenbrock's Function: $F_2(x) = \sum_{i=1}^{D-1}\left(100 \times \left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right)$

F8F2: $F_8F_2(x) = F_8\left(F_2(x_1,x_2)\right) + F_8\left(F_2(x_2,x_3)\right) + \cdots + F_8\left(F_2(x_{D-1},x_D)\right) + F_8\left(F_2(x_D,x_1)\right)$

$$f(x) = F_8\left(F_2(z_1,z_2)\right) + F_8\left(F_2(z_2,z_3)\right) + \cdots + F_8\left(F_2(z_{D-1},z_D)\right) + F_8\left(F_2(z_D,z_1)\right) + f_{bias}$$

$$z = x - o + 1$$

Composition benchmark functions

$$f(x) = \sum_{i=1}^{n} \{\varpi_i * [\lambda_i g_i(x) + bias_i]\} + f_{bias}$$

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

$$\varpi_i = \frac{w_i}{\sum_{i=1}^{n} w_i}$$

Composition Function 1
$n = 5, \sigma = [10,20,30,40,50], \lambda = [1,1e\text{-}6, 1e\text{-}26, 1e\text{-}6, 0.1]$
$bias = [0,100,200,300,400]$
$g_1(x)$: Rotated Rosenbrock's Function
$g_2(x)$: Rotated Different Powers Function
$g_3(x)$: Rotated Bent Cigar Function
$g_4(x)$: Rotated Discus Function
$g_5(x)$: Sphere Function
Composition Function 2
$n = 3, \sigma = [20,20,20], \lambda = [1,1,1], bias = [0,100,200]$
$g_1(x)$: Schwefel's Function
$g_2(x)$: Schwefel's Function
$g_3(x)$: Schwefel's Function
Composition Function 3
$n = 3, \sigma = [20,20,20], \lambda = [1,1,1], bias = [0,100,200]$
$g_1(x)$: Rotated Schwefel's Function
$g_2(x)$: Rotated Schwefel's Function
$g_3(x)$: Rotated Schwefel's Function

Composition Function 4
$n = 3, \sigma = [20,20,20], \lambda = [0.25,1,2.5], bias = [0,100,200]$ $g_1(x)$: Rotated Schwefel's Function $g_2(x)$: Rotated Rastrigin's Function $g_3(x)$: Rotated Weierstrass Function
Composition Function 5
$n = 3, \sigma = [10,30,50], \lambda = [0.25,1,2.5], bias = [0,100,200]$ $g_1(x)$: Rotated Schwefel's Function $g_2(x)$: Rotated Rastrigin's Function $g_3(x)$: Rotated Weierstrass Function
Composition Function 6
$n = 5, \sigma = [10,10,10,10,10], \lambda = [0.25,1,1e\text{-}7,2.5, 10]$ $bias = [0,100,200,300,400]$ $g_1(x)$: Rotated Schwefel's Function $g_2(x)$: Rotated Rastrigin's Function $g_3(x)$: Rotated High Conditioned Elliptic Function $g_4(x)$: Rotated Weierstrass Function $g_5(x)$: Rotated Griewank's Function
Composition Function 7
$n = 5, \sigma = [10,10,10,20,20], \lambda = [100,10,2.5,25,0.1]$ $bias = [0,100,200,300,400]$ $g_1(x)$: Rotated Griewank's Function $g_2(x)$: Rotated Rastrigin's Function $g_3(x)$: Rotated Schwefel's Function $g_4(x)$: Rotated Weierstrass Function $g_5(x)$: Sphere Function

Composition Function 7
$n = 5, \sigma = [10,20,30,40,50], \lambda = [2.5,2.5e\text{-}3,2.5,5e\text{-}4,0.1]$ $bias = [0,100,200,300,400]$ $g_1(x)$: Rotated Expanded Griewank's plus Rosenbrock's Function $g_2(x)$: Rotated Schaffers F7 Function $g_3(x)$: Rotated Schwefel's Function $g_4(x)$: Rotated Expanded Scaffer's F6 Function $g_5(x)$: Sphere Function

APPENDIX B

Benchmark test for SSODE

Benchmark (error value)		DE/Best/1	DE/Rand/1	DE/ rand-to-best/1	DE/best/2	DE/rand/2	SSO (1-3-5-1)	SSODE (1-3-5-1)	SSODE (1-2-4-8)
Bm1	max	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.31E-03	5.71E+03	3.53E-03	0.00E+00
	min	0.00E+00	0.00E+00	0.00E+00	0.00E+00	5.51E-04	1.44E+04	1.37E-04	0.00E+00
	mean	0.00E+00	0.00E+00	0.00E+00	0.00E+00	9.31E-04	2.74E+04	9.92E-04	0.00E+00
	std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.82E-04	5.96E+03	6.57E-04	0.00E+00
Bm2	max	0.00E+00	8.38E+01	0.00E+00	1.13E+01	1.19E+03	6.74E+01	1.83E+02	8.85E+00
	min	0.00E+00	4.91E+01	0.00E+00	2.81E+00	6.27E+02	7.65E+01	4.73E+01	3.55E+00
	mean	0.00E+00	6.28E+01	0.00E+00	7.35E+00	9.46E+02	8.49E+01	9.36E+01	6.15E+00
	std	0.00E+00	8.62E+00	0.00E+00	1.70E+00	1.29E+02	4.87E+00	2.95E+01	1.35E+00
Bm3	max	4.22E+07	1.15E+08	2.26E+07	8.08E+07	1.27E+08	4.47E+07	6.94E+07	3.02E+07
	min	9.28E+06	5.49E+07	8.44E+06	2.71E+07	6.36E+07	8.90E+06	2.17E+07	1.28E+07
	mean	2.56E+07	8.75E+07	1.49E+07	5.52E+07	9.49E+07	2.30E+07	4.24E+07	2.12E+07
	std	8.84E+06	1.43E+07	3.11E+06	1.15E+07	1.70E+07	8.79E+06	1.36E+07	4.21E+06
Bm4	max	5.62E-03	1.06E+03	1.05E-03	5.49E+03	5.49E+03	6.47E+02	1.00E+03	8.29E+02
	min	2.82E-04	6.23E+02	1.90E-04	3.27E+03	3.27E+03	6.75E+02	3.29E+02	3.86E+02
	mean	1.26E-03	8.56E+02	4.27E-04	4.30E+03	4.30E+03	7.13E+02	6.28E+02	5.91E+02
	std	1.05E-03	1.00E+02	1.50E-04	4.68E+02	4.68E+02	1.68E+01	1.90E+02	1.01E+02
Bm5	max	3.73E+03	4.63E+03	2.35E+03	4.30E+03	6.63E+03	6.76E+02	3.68E+03	3.96E+03
	min	9.52E+02	3.54E+03	8.46E+02	1.98E+03	5.04E+03	6.89E+02	1.96E+03	2.62E+03
	mean	1.96E+03	4.28E+03	1.55E+03	2.82E+03	5.94E+03	7.08E+02	3.05E+03	3.30E+03
	std	6.32E+02	2.31E+02	3.48E+02	5.23E+02	3.38E+02	1.07E+01	5.20E+02	2.91E+02
Bm6	max	3.99E+00	2.09E+01	7.91E+01	4.91E+00	5.10E+01	8.75E+03	4.76E+02	2.84E+01
	min	0.00E+00	1.94E+01	1.42E+01	2.31E-01	3.53E+01	2.40E+02	2.19E+01	2.27E+01
	mean	6.65E-01	2.01E+01	2.87E+01	2.69E+00	4.10E+01	1.57E+03	9.00E+01	2.43E+01
	std	1.51E+00	3.73E-01	2.20E+01	1.03E+00	3.18E+00	1.97E+03	1.09E+02	1.47E+00
Bm7	max	4.70E+03	4.70E+03	4.70E+03	4.70E+03	4.70E+03	4.72E+03	4.70E+03	5.52E-01
	min	4.70E+03	4.70E+03	4.70E+03	4.70E+03	4.70E+03	4.71E+03	4.70E+03	2.77E-01
	mean	4.70E+03	4.70E+03	4.70E+03	4.70E+03	4.70E+03	4.71E+03	4.70E+03	4.29E-01
	std	9.25E-13	9.25E-13	9.25E-13	9.25E-13	2.93E-06	3.04E+00	2.65E-05	7.72E-02
Bm8	max	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01
	min	2.09E+01	2.08E+01	2.07E+01	2.08E+01	2.08E+01	2.07E+01	2.08E+01	2.07E+01
	mean	2.10E+01	2.09E+01	2.09E+01	2.09E+01	2.10E+01	2.09E+01	2.09E+01	2.09E+01
	std	4.42E-02	4.17E-02	5.90E-02	5.17E-02	5.19E-02	7.34E-02	5.12E-02	6.05E-02
Bm9	max	1.89E+01	8.30E+01	6.54E+01	9.22E+01	1.01E+02	1.09E+00	2.10E+01	8.69E+01
	min	2.98E+00	6.34E+01	3.88E+01	6.41E+01	8.02E+01	2.15E-01	7.59E+00	5.30E+01
	mean	1.08E+01	7.47E+01	5.28E+01	8.04E+01	9.25E+01	5.49E-01	1.36E+01	7.13E+01
	std	3.78E+00	5.56E+00	5.59E+00	7.11E+00	5.64E+00	2.09E-01	3.16E+00	6.60E+00
Bm10	max	1.91E+02	2.17E+02	1.88E+02	2.16E+02	2.33E+02	2.20E+02	2.23E+02	1.77E+02
	min	1.26E+02	1.77E+02	1.36E+02	1.60E+02	1.91E+02	9.30E+01	9.97E+01	1.30E+02
	mean	1.66E+02	1.96E+02	1.66E+02	1.95E+02	2.15E+02	1.50E+02	1.78E+02	1.55E+02
	std	1.73E+01	9.53E+00	1.18E+01	1.15E+01	1.07E+01	2.95E+01	2.69E+01	1.09E+01
Bm11	max	4.06E+01	4.06E+01	3.78E+01	4.07E+01	4.12E+01	3.66E+01	3.52E+01	3.48E+01
	min	3.07E+01	3.61E+01	3.23E+01	3.45E+01	3.58E+01	2.45E+01	2.64E+01	2.71E+01
	mean	3.76E+01	3.87E+01	3.58E+01	3.82E+01	3.86E+01	3.10E+01	3.13E+01	3.20E+01
	std	1.98E+00	1.06E+00	1.18E+00	1.18E+00	1.17E+00	3.08E+00	2.18E+00	1.58E+00
Bm12	max	4.94E+05	4.86E+05	4.78E+05	4.82E+05	4.85E+05	8.11E+04	4.68E+05	1.16E+05
	min	3.13E+05	3.23E+05	2.90E+05	2.12E+05	2.59E+05	9.25E+03	2.94E+05	5.85E+04
	mean	4.02E+05	4.19E+05	3.93E+05	4.03E+05	4.13E+05	2.80E+04	4.07E+05	8.38E+04
	std	4.84E+04	3.91E+04	4.11E+04	5.60E+04	5.58E+04	1.43E+04	4.48E+04	1.46E+04
Bm13	max	7.44E+00	9.46E+00	7.64E+00	1.00E+01	1.15E+01	2.04E+00	9.38E+00	8.49E+00
	min	2.74E+00	7.09E+00	5.56E+00	6.53E+00	8.80E+00	9.93E-01	2.11E+00	6.77E+00
	mean	5.76E+00	8.28E+00	6.59E+00	8.64E+00	1.03E+01	1.40E+00	5.41E+00	7.64E+00
	std	1.08E+00	5.67E-01	4.29E-01	7.75E-01	7.33E-01	2.37E-01	1.67E+00	5.37E-01
Bm14	max	1.35E+01	1.37E+01	1.35E+01	1.35E+01	1.37E+01	1.36E+01	1.36E+01	1.34E+01
	min	1.26E+01	1.29E+01	1.28E+01	1.28E+01	1.28E+01	1.20E+01	1.31E+01	1.28E+01
	mean	1.32E+01	1.34E+01	1.32E+01	1.33E+01	1.34E+01	1.28E+01	1.33E+01	1.31E+01
	std	1.91E-01	1.67E-01	1.43E-01	1.67E-01	1.76E-01	4.03E-01	9.99E-02	1.45E-01

APPENDIX C

Benchmark test for MISO

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
Total Mean	max	6.86E+06	1.52E+08	4.92E+08	4.28E+08	5.26E+08	5.21E+08	1.22E+09	7.87E+07	4.78E+08	1.96E+09
	min	2.43E+04	4.41E+07	2.00E+08	9.55E+07	1.94E+08	2.67E+08	4.84E+08	8.81E+06	1.46E+07	2.81E+08
	mean	6.57E+05	8.75E+07	3.93E+08	2.44E+08	3.64E+08	3.89E+08	8.63E+08	3.06E+07	1.35E+08	9.00E+08

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
	max	13			4				11	0	0
	min	13			3				8	3	1
	mean	14			3				10	1	0
	std	8			12				8	0	0

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
max	bm1	3.15E-05	1.00E-08	2.26E+02	7.00E-08	1.39E+02	4.38E-03	3.83E+03	0.00E+00	2.19E+00	5.69E+03
	bm2	3.15E+04	3.99E+07	4.38E+07	4.15E+07	4.02E+07	7.49E+07	8.50E+07	2.14E+07	5.88E+07	1.10E+08
	bm3	1.92E+08	4.22E+09	1.37E+10	1.20E+10	1.47E+10	1.45E+10	3.42E+10	2.18E+09	1.33E+10	5.49E+10
	bm4	2.95E-03	8.84E+04	9.13E+04	1.01E+05	9.07E+04	7.42E+04	8.61E+04	8.92E+04	2.01E+04	4.03E+04
	bm5	5.99E-03	1.00E-08	5.35E+01	4.35E-05	3.59E+01	6.18E-02	1.09E+03	0.00E+00	1.04E+00	7.97E+02
	bm6	7.41E+01	8.15E+01	1.35E+02	5.83E+01	1.29E+02	9.93E+01	3.90E+02	1.88E+01	1.43E+02	1.46E+03
	bm7	3.98E+01	1.12E+02	1.38E+02	1.39E+02	1.40E+02	1.24E+02	1.71E+02	1.47E+02	1.89E+02	3.16E+02
	bm8	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01	2.10E+01
	bm9	3.44E+01	3.21E+01	3.20E+01	3.20E+01	3.18E+01	3.49E+01	3.55E+01	3.31E+01	3.72E+01	4.11E+01
	bm10	1.27E-01	5.33E+01	2.38E+02	9.75E+01	2.06E+02	2.83E+02	8.69E+02	8.09E-01	5.69E+01	1.37E+03
	bm11	1.37E+02	1.00E-08	5.35E+01	1.93E-06	4.42E+01	2.88E+01	1.93E+02	0.00E+00	2.98E+00	3.66E+02
	bm12	1.72E+02	1.84E+02	2.83E+02	2.49E+02	2.79E+02	2.50E+02	3.48E+02	3.20E+02	2.86E+02	3.57E+02
	bm13	1.93E+02	2.22E+02	3.17E+02	2.87E+02	3.04E+02	2.69E+02	3.59E+02	3.73E+02	3.09E+02	4.20E+02
	bm14	7.37E+03	1.37E+00	5.81E+02	6.99E+00	4.98E+02	1.95E+03	2.61E+03	7.55E+00	1.32E+01	4.06E+03
	bm15	7.76E+03	5.98E+03	6.02E+03	6.06E+03	5.80E+03	7.24E+03	7.19E+03	4.70E+03	5.63E+03	7.79E+03
	bm16	2.95E+00	2.57E+00	2.68E+00	2.42E+00	2.43E+00	2.90E+00	2.85E+00	2.00E+00	2.99E+00	3.08E+00
	bm17	2.12E+02	3.04E+01	1.02E+02	3.31E+01	8.63E+01	9.62E+01	2.64E+02	3.06E+01	3.71E+01	4.38E+02
	bm18	2.27E+02	2.58E+02	3.56E+02	3.13E+02	3.64E+02	2.93E+02	4.15E+02	3.89E+02	3.87E+02	3.49E+02
	bm19	1.67E+01	2.80E+00	1.45E+01	3.64E+00	1.23E+01	8.71E+00	1.67E+02	8.39E-01	2.74E+00	3.22E+03
	bm20	1.25E+01	1.48E+01	1.47E+01	1.50E+01	1.50E+01	1.43E+01	1.44E+01	1.50E+01	1.32E+01	1.48E+01
	bm21	4.44E+02	4.44E+02	5.13E+02	4.49E+02	4.93E+02	4.45E+02	9.68E+02	3.00E+02	4.44E+02	1.33E+03
	bm22	7.48E+03	3.17E+02	9.39E+02	4.05E+02	7.63E+02	2.86E+03	3.16E+03	1.39E+02	2.34E+02	6.80E+03
	bm23	8.01E+03	6.68E+03	6.86E+03	6.67E+03	6.73E+03	7.81E+03	7.72E+03	5.75E+03	6.64E+03	8.38E+03
	bm24	2.47E+02	2.87E+02	2.93E+02	2.94E+02	2.94E+02	2.97E+02	3.04E+02	2.97E+02	3.05E+02	3.33E+02
	bm25	2.81E+02	3.07E+02	3.10E+02	3.11E+02	3.10E+02	3.17E+02	3.20E+02	3.15E+02	3.19E+02	3.63E+02
	bm26	2.00028E+02	2.03E+02	2.04E+02	2.04E+02	2.04E+02	2.09E+02	2.10E+02	2.0144E+02	3.90E+02	2.04E+02
	bm27	7.93E+02	4.39E+02	4.82E+02	4.32E+02	4.46E+02	1.24E+03	1.26E+03	4.00E+02	1.31E+03	1.43E+03
	bm28	3.00E+02	3.08E+02	1.26E+03	6.35E+02	1.20E+03	3.39E+02	2.08E+03	3.00E+02	9.95E+02	3.57E+03
	MEAN	**6.86E+06**	**1.52E+08**	**4.92E+08**	**4.28E+08**	**5.26E+08**	**5.21E+08**	**1.22E+09**	**7.87E+07**	**4.78E+08**	**1.96E+09**
	Total	**1.92E+08**	**4.26E+09**	**1.38E+10**	**1.20E+10**	**1.47E+10**	**1.46E+10**	**3.43E+10**	**2.20E+09**	**1.34E+10**	**5.50E+10**

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
min	bm1	5.10E-07	1.00E-08	1.05E+02	2.00E-08	6.27E+01	2.01E-03	2.04E+03	0.00E+00	4.29E-01	4.36E+02
	bm2	1.05E+01	1.13E+07	2.01E+07	1.75E+07	1.95E+07	2.92E+07	4.49E+07	9.50E+06	5.42E+06	1.56E+07
	bm3	6.54E+05	1.22E+09	5.58E+09	2.66E+09	5.41E+09	7.45E+09	1.35E+10	2.37E+08	4.04E+08	7.86E+09
	bm4	2.59E-05	4.47E+04	5.90E+04	5.63E+04	4.60E+04	3.57E+04	4.88E+04	4.99E+04	8.05E+03	3.67E+03
	bm5	3.11E-04	1.00E-08	1.52E+01	1.64E-05	1.99E+01	2.86E-02	2.76E+02	0.00E+00	2.37E-01	8.13E+01
	bm6	1.40E-03	1.86E+01	7.30E+01	2.71E+01	7.81E+01	5.53E+01	2.27E+02	8.17E+00	1.58E+01	8.94E+01
	bm7	2.73E+00	6.18E+01	8.53E+01	7.67E+01	7.44E+01	8.64E+01	1.15E+02	5.70E+01	4.81E+01	7.64E+01
	bm8	2.09E+01	2.08E+01	2.08E+01	2.08E+01	2.08E+01	2.08E+01	2.07E+01	2.08E+01	2.08E+01	2.08E+01
	bm9	2.02E+01	2.63E+01	2.72E+01	2.39E+01	2.39E+01	3.05E+01	2.97E+01	2.42E+01	2.25E+01	2.36E+01
	bm10	1.12E-04	1.76E+01	1.20E+02	3.90E+01	9.82E+01	1.34E+02	4.91E+02	1.95E-01	7.73E+00	2.42E+02
	bm11	5.51E+01	1.00E-08	3.53E+01	4.20E-07	2.97E+01	1.85E+01	1.45E+02	0.00E+00	3.91E-01	1.31E+02
	bm12	6.22E+01	1.13E+02	2.08E+02	1.65E+02	2.01E+02	1.92E+02	2.65E+02	1.63E+02	6.06E+01	1.24E+02
	bm13	7.95E+01	1.46E+02	2.45E+02	1.92E+02	2.32E+02	2.00E+02	2.78E+02	2.21E+02	1.10E+02	1.96E+02
	bm14	5.76E+03	1.30E-01	3.37E+02	4.83E-01	2.96E+02	1.25E+03	1.82E+03	3.10E-01	3.07E+00	1.89E+03
	bm15	6.44E+03	4.48E+03	4.85E+03	4.37E+03	4.58E+03	5.94E+03	5.52E+03	3.28E+03	3.10E+03	3.71E+03
	bm16	1.53E+00	1.36E+00	1.52E+00	1.32E+00	1.33E+00	1.52E+00	1.72E+00	1.06E+00	1.05E+00	1.67E+00
	bm17	1.36E+02	3.04E+01	7.28E+01	3.12E+01	6.91E+01	7.72E+01	2.20E+02	3.04E+01	3.33E+01	1.72E+02
	bm18	1.71E+02	1.89E+02	2.85E+02	2.43E+02	2.90E+02	2.32E+02	3.25E+02	2.66E+02	1.58E+02	1.41E+02
	bm19	8.65E+00	1.52E+00	8.71E+00	2.36E+00	8.18E+00	6.08E+00	4.22E+01	2.83E-01	1.28E+00	5.12E+01
	bm20	1.11E+01	1.31E+01	1.29E+01	1.32E+01	1.29E+01	1.33E+01	1.27E+01	1.37E+01	1.11E+01	1.18E+01
	bm21	2.00E+02	2.00E+02	4.69E+02	2.71E+02	4.27E+02	2.04E+02	6.60E+02	1.01E+02	1.26E+02	5.40E+02
	bm22	5.32E+03	1.44E+02	5.55E+02	2.28E+02	3.93E+02	1.95E+03	2.12E+03	1.85E+01	1.51E+01	2.16E+03
	bm23	5.71E+03	5.15E+03	5.30E+03	5.26E+03	4.99E+03	5.77E+03	6.45E+03	4.04E+03	3.50E+03	4.45E+03
	bm24	2.10E+02	2.63E+02	2.72E+02	2.68E+02	2.62E+02	2.78E+02	2.83E+02	2.75E+02	2.59E+02	2.72E+02
	bm25	2.07E+02	2.85E+02	2.88E+02	2.90E+02	2.94E+02	3.02E+02	3.09E+02	2.98E+02	2.77E+02	3.14E+02
	bm26	2.00001E+02	2.01E+02	2.01E+02	2.01E+02	2.01E+02	2.03E+02	2.03E+02	2.0052E+02	2.01E+02	2.00E+02
	bm27	4.96E+02	4.08E+02	4.17E+02	4.13E+02	4.15E+02	1.08E+03	1.08E+03	4.00E+02	8.73E+02	9.42E+02
	bm28	1.00E+02	2.86E+02	7.72E+02	3.93E+02	7.49E+02	3.21E+02	1.67E+03	1.35E+02	1.23E+02	1.44E+03
	MEAN	**2.43E+04**	**4.41E+07**	**2.00E+08**	**9.55E+07**	**1.94E+08**	**2.67E+08**	**4.84E+08**	**8.81E+06**	**1.46E+07**	**2.81E+08**
	Total	**6.80E+05**	**1.24E+09**	**5.61E+09**	**2.67E+09**	**5.43E+09**	**7.48E+09**	**1.35E+10**	**2.47E+08**	**4.09E+08**	**7.88E+09**

			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
mean	bm1	5.55E-06	1.00E-08	1.58E+02	4.27E-08	1.07E+02	3.04E-03	3.01E+03	0.00E+00	1.01E+00	2.26E+03
	bm2	1.82E+03	2.54E+07	3.34E+07	3.01E+07	3.03E+07	5.49E+07	6.62E+07	1.50E+07	2.85E+07	5.02E+07
	bm3	1.84E+07	2.42E+09	1.10E+10	6.81E+09	1.02E+10	1.08E+10	2.41E+10	8.42E+08	3.76E+09	2.51E+10
	bm4	6.69E-04	7.17E+04	7.56E+04	7.97E+04	7.75E+04	5.86E+04	6.94E+04	7.12E+04	1.28E+04	1.28E+04
	bm5	2.24E-03	1.00E-08	3.73E+01	2.95E-05	2.55E+01	4.38E-02	6.95E+02	0.00E+00	6.36E-01	2.34E+02
	bm6	1.15E+01	3.00E+01	1.12E+02	3.80E+01	1.04E+02	7.45E+01	3.24E+02	1.51E+01	7.14E+01	6.13E+02
	bm7	1.40E+01	9.07E+01	1.13E+02	1.11E+02	1.13E+02	1.08E+02	1.42E+02	1.19E+02	1.17E+02	1.40E+02
	bm8	2.09E+01	2.10E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01
	bm9	2.79E+01	2.97E+01	2.98E+01	2.93E+01	2.93E+01	3.30E+01	3.30E+01	2.97E+01	2.92E+01	3.43E+01
	bm10	1.54E-02	3.03E+01	1.83E+02	7.18E+01	1.54E+02	2.03E+02	6.74E+02	3.86E-01	2.58E+01	7.49E+02
	bm11	1.02E+02	1.00E-08	4.70E+01	1.05E-06	3.76E+01	2.42E+01	1.70E+02	0.00E+00	1.22E+00	2.28E+02
	bm12	1.19E+02	1.53E+02	2.50E+02	2.10E+02	2.47E+02	2.27E+02	3.09E+02	2.63E+02	1.40E+02	2.35E+02
	bm13	1.62E+02	1.85E+02	2.84E+02	2.41E+02	2.78E+02	2.36E+02	3.23E+02	3.12E+02	2.16E+02	3.02E+02
	bm14	6.55E+03	2.88E-01	4.93E+02	2.44E+00	3.98E+02	1.67E+03	2.28E+03	3.48E+00	7.55E+00	2.75E+03
	bm15	7.17E+03	5.31E+03	5.55E+03	5.37E+03	5.32E+03	6.76E+03	6.71E+03	4.11E+03	4.57E+03	6.32E+03
	bm16	2.50E+00	1.99E+00	2.01E+00	2.01E+00	2.00E+00	2.39E+00	2.37E+00	1.61E+00	1.97E+00	2.46E+00
	bm17	1.81E+02	3.04E+01	8.95E+01	3.20E+01	8.08E+01	8.64E+01	2.44E+02	3.05E+01	3.53E+01	2.47E+02
	bm18	2.06E+02	2.33E+02	3.26E+02	2.84E+02	3.33E+02	2.70E+02	3.86E+02	3.40E+02	2.57E+02	2.20E+02
	bm19	1.38E+01	2.32E+00	1.22E+01	3.05E+00	1.03E+01	7.60E+00	9.51E+01	5.18E-01	1.91E+00	4.28E+02
	bm20	1.19E+01	1.41E+01	1.40E+01	1.43E+01	1.43E+01	1.39E+01	1.39E+01	1.45E+01	1.22E+01	1.35E+01
	bm21	3.46E+02	3.19E+02	4.95E+02	4.33E+02	4.65E+02	3.46E+02	8.25E+02	2.25E+02	2.94E+02	9.37E+02
	bm22	6.54E+03	2.14E+02	7.71E+02	2.91E+02	6.27E+02	2.51E+03	2.73E+03	1.04E+02	1.35E+02	3.69E+03
	bm23	7.40E+03	5.98E+03	6.25E+03	6.03E+03	6.21E+03	7.13E+03	7.15E+03	5.07E+03	4.99E+03	7.25E+03
	bm24	2.26E+02	2.77E+02	2.86E+02	2.83E+02	2.85E+02	2.87E+02	2.96E+02	2.87E+02	2.87E+02	3.04E+02
	bm25	2.34E+02	2.98E+02	3.03E+02	3.02E+02	3.04E+02	3.10E+02	3.15E+02	3.07E+02	3.01E+02	3.45E+02
	bm26	2.00007E+02	2.02E+02	2.03E+02	2.03E+02	2.03E+02	2.06E+02	2.06E+02	2.0084E+02	2.43E+02	2.01E+02
	bm27	6.07E+02	4.14E+02	4.47E+02	4.21E+02	4.31E+02	1.17E+03	1.19E+03	4.00E+02	1.10E+03	1.14E+03
	bm28	2.96E+02	3.00E+02	1.02E+03	5.58E+02	9.51E+02	3.27E+02	1.90E+03	2.60E+02	3.44E+02	2.38E+03
	MEAN	**6.57E+05**	**8.75E+07**	**3.93E+08**	**2.44E+08**	**3.64E+08**	**3.89E+08**	**8.63E+08**	**3.06E+07**	**1.35E+08**	**9.00E+08**
	Total	**1.84E+07**	**2.45E+09**	**1.10E+10**	**6.84E+09**	**1.02E+10**	**1.09E+10**	**2.42E+10**	**8.57E+08**	**3.79E+09**	**2.52E+10**

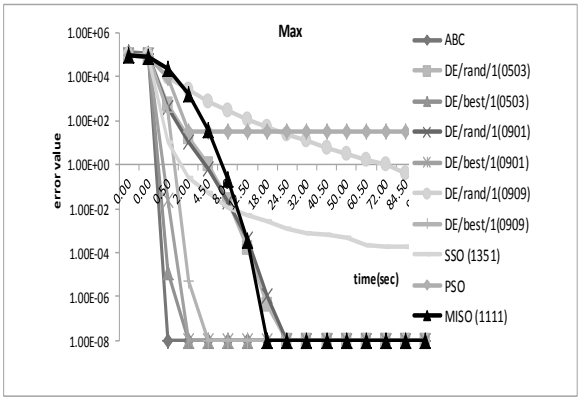
			F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		NP=40		w=0.9~0.4
		MISO	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	ABC	SSO(1351)	PSO_c1_2_c2_2
std	bm1	5.82E-06	5.05E-24	2.41E+01	1.11E-08	1.71E+01	5.87E-04	4.31E+02	0.00E+00	3.71E-01	1.14E+03
	bm2	4.51E+03	6.24E+06	5.92E+06	6.47E+06	5.72E+06	1.12E+07	1.14E+07	3.14E+06	1.34E+07	2.14E+07
	bm3	3.23E+07	6.78E+08	2.02E+09	2.11E+09	2.47E+09	2.01E+09	4.25E+09	5.22E+08	3.33E+09	1.08E+10
	bm4	7.66E-04	9.06E+03	8.02E+03	1.03E+04	8.55E+03	8.78E+03	8.00E+03	8.22E+03	2.79E+03	7.50E+03
	bm5	1.33E-03	5.01E-24	9.02E+00	6.50E-06	3.97E+00	8.05E-03	1.56E+02	0.00E+00	1.87E-01	1.63E+02
	bm6	1.60E+01	1.25E+01	1.45E+01	7.18E+00	1.13E+01	9.73E+00	3.82E+01	2.66E+00	3.33E+01	2.42E+02
	bm7	8.62E+00	1.06E+01	1.19E+01	1.26E+01	1.29E+01	1.03E+01	1.49E+01	1.79E+01	2.90E+01	4.55E+01
	bm8	3.81E-02	5.81E-02	4.93E-02	5.35E-02	4.38E-02	4.69E-02	5.47E-02	5.30E-02	5.38E-02	5.05E-02
	bm9	3.11E+00	1.35E+00	1.19E+00	1.72E+00	1.72E+00	1.18E+00	1.36E+00	1.83E+00	3.11E+00	4.23E+00
	bm10	2.09E-02	7.27E+00	2.60E+01	1.35E+01	2.33E+01	3.00E+01	8.49E+01	1.22E-01	1.11E+01	2.43E+02
	bm11	2.37E+01	5.01E-24	3.85E+00	3.25E-07	3.25E+00	2.56E+00	1.13E+01	0.00E+00	5.26E-01	5.27E+01
	bm12	2.66E+01	1.79E+01	1.71E+01	2.03E+01	1.90E+01	1.39E+01	1.74E+01	3.43E+01	4.52E+01	5.44E+01
	bm13	2.55E+01	1.63E+01	1.66E+01	1.83E+01	1.50E+01	1.55E+01	1.98E+01	3.18E+01	4.71E+01	4.91E+01
	bm14	3.97E+02	2.19E-01	5.78E+01	1.32E+00	4.58E+01	1.58E+02	1.73E+02	1.67E+00	2.46E+00	4.93E+02
	bm15	2.83E+02	4.13E+02	2.66E+02	3.54E+02	2.98E+02	2.87E+02	2.98E+02	3.30E+02	6.59E+02	1.32E+03
	bm16	2.77E-01	2.29E-01	2.80E-01	2.40E-01	2.45E-01	3.03E-01	2.33E-01	2.00E-01	4.52E-01	3.21E-01
	bm17	1.57E+01	3.50E-06	5.38E+00	4.32E-01	3.86E+00	4.50E+00	1.02E+01	4.27E-02	8.21E-01	5.09E+01
	bm18	1.07E+01	1.53E+01	1.75E+01	1.69E+01	1.75E+01	1.28E+01	2.24E+01	2.82E+01	4.42E+01	5.01E+01
	bm19	1.63E+00	2.60E-01	1.20E+00	3.27E-01	9.07E-01	5.50E-01	3.01E+01	1.43E-01	2.99E-01	5.29E+02
	bm20	3.05E-01	3.64E-01	4.76E-01	3.65E-01	4.19E-01	2.56E-01	2.96E-01	2.80E-01	4.90E-01	8.74E-01
	bm21	7.11E+01	1.04E+02	9.57E+00	4.59E+01	1.62E+01	1.07E+02	5.55E+01	4.25E+01	8.00E+01	2.01E+02
	bm22	5.13E+02	3.68E+01	8.46E+01	3.46E+01	8.03E+01	2.10E+02	1.69E+02	2.72E+01	6.93E+01	7.95E+02
	bm23	3.94E+02	3.79E+02	3.85E+02	3.22E+02	3.81E+02	4.03E+02	3.02E+02	3.81E+02	8.06E+02	1.14E+03
	bm24	7.59E+00	5.60E+00	4.55E+00	5.75E+00	5.31E+00	4.69E+00	4.74E+00	4.96E+00	1.06E+01	1.40E+01
	bm25	2.02E+01	4.46E+00	4.07E+00	4.25E+00	3.51E+00	3.55E+00	2.81E+00	4.30E+00	8.59E+00	1.05E+01
	bm26	6.75498E-03	5.46E-01	6.15E-01	5.83E-01	6.69E-01	1.36E+00	1.50E+00	2.0228E-01	7.57E+01	9.95E-01
	bm27	6.10E+01	5.93E+00	1.42E+01	4.76E+00	7.31E+00	3.60E+01	4.01E+01	1.14E-02	9.57E+01	1.20E+02
	bm28	2.80E+01	2.36E+00	1.18E+02	4.73E+01	1.15E+02	4.43E+00	8.99E+01	5.66E+01	1.07E+02	4.82E+02
	MEAN	**1.15E+06**	**2.44E+07**	**7.23E+07**	**7.54E+07**	**8.84E+07**	**7.22E+07**	**1.52E+08**	**1.87E+07**	**1.19E+08**	**3.88E+08**
	Total	**3.23E+07**	**6.85E+08**	**2.02E+09**	**2.11E+09**	**2.47E+09**	**2.02E+09**	**4.26E+09**	**5.25E+08**	**3.34E+09**	**1.09E+10**

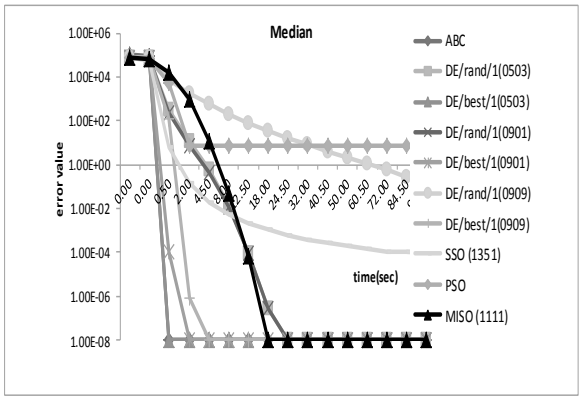
APPENDIX D

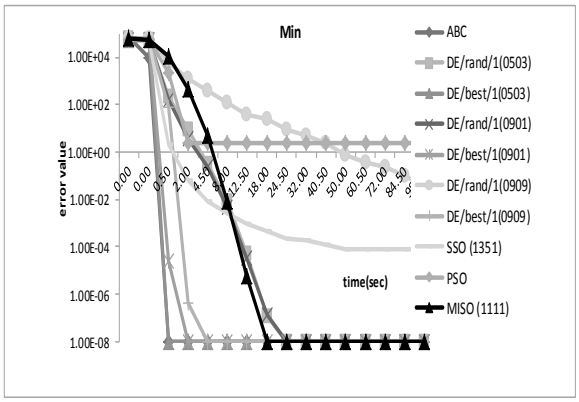
Convergence graphs

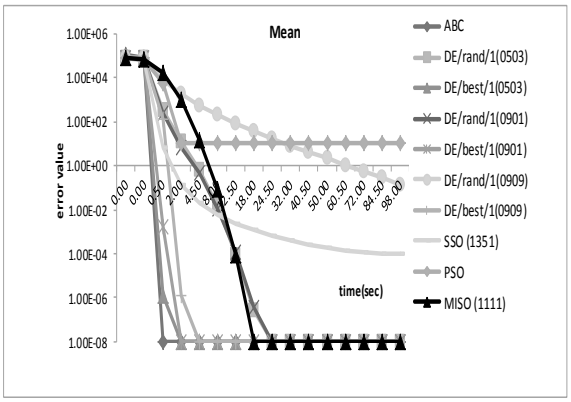
In the graphs, if error value ≤ 1.00E-08, than set it to 1.00E-08.

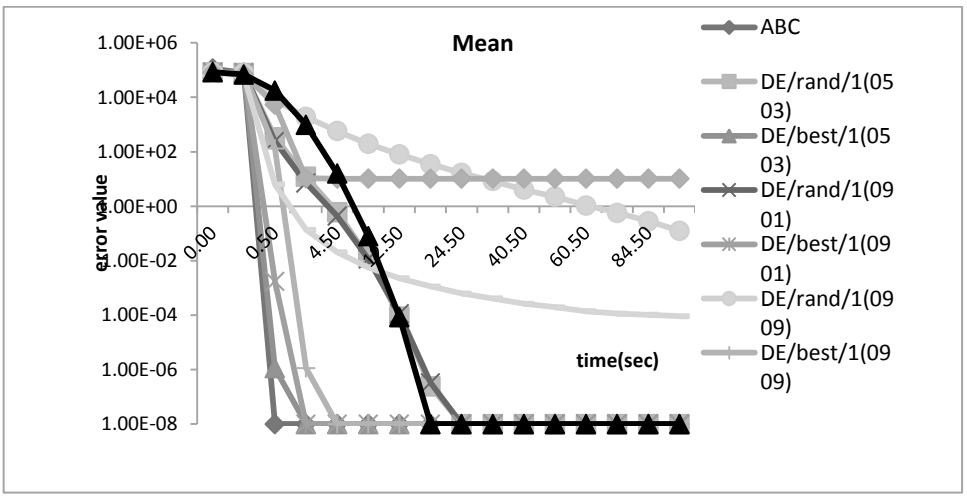
BM1



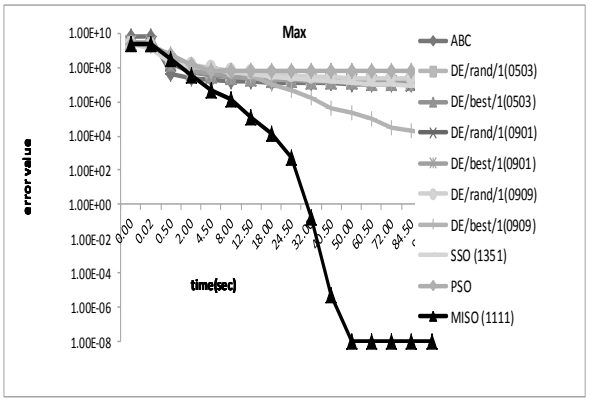


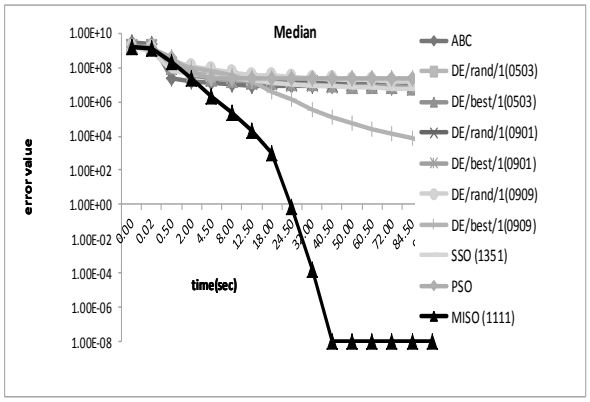


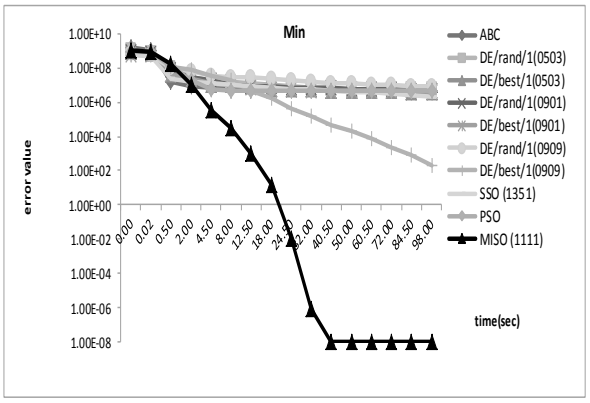


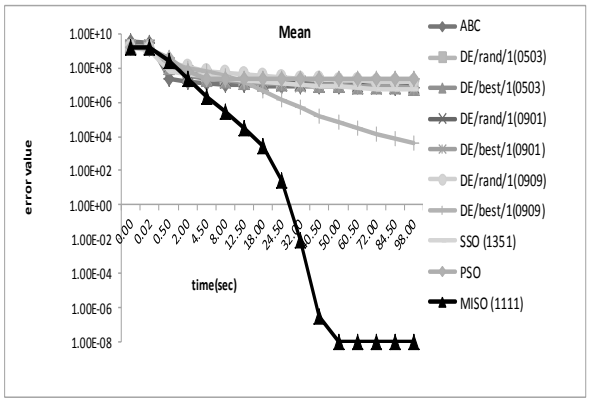


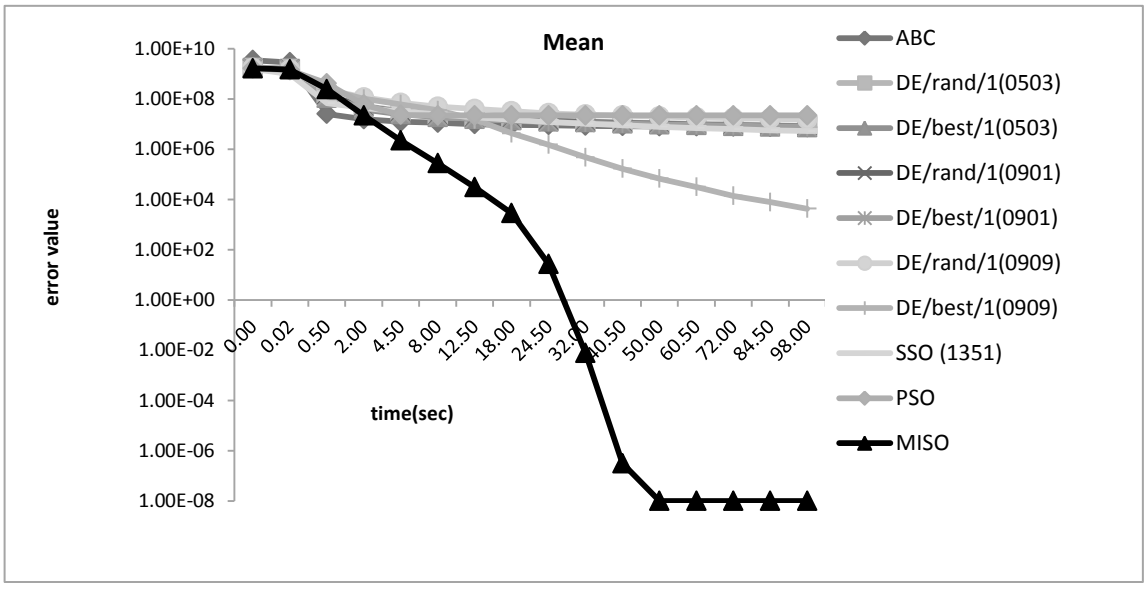
BM2


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

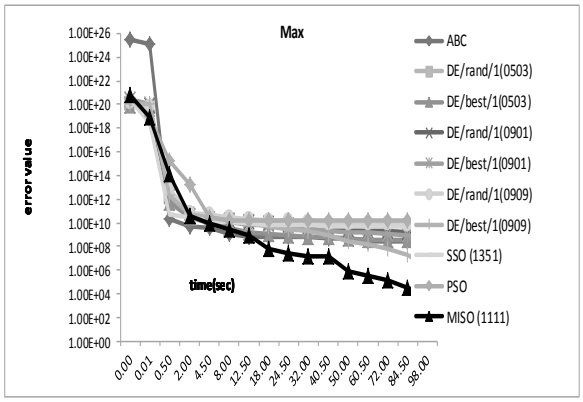

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

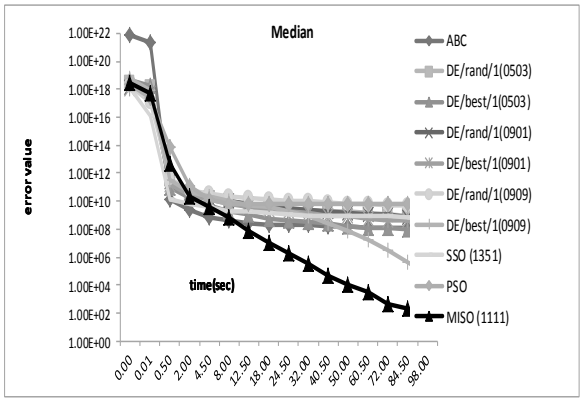

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

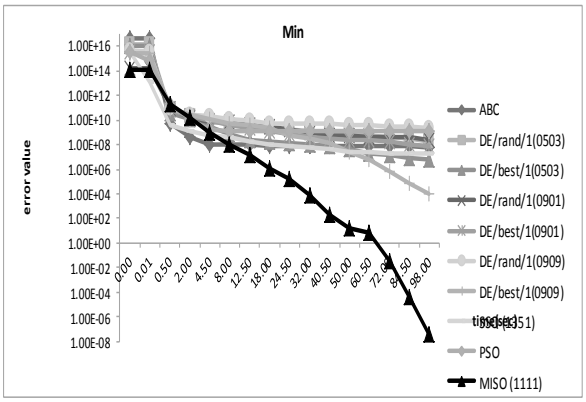

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

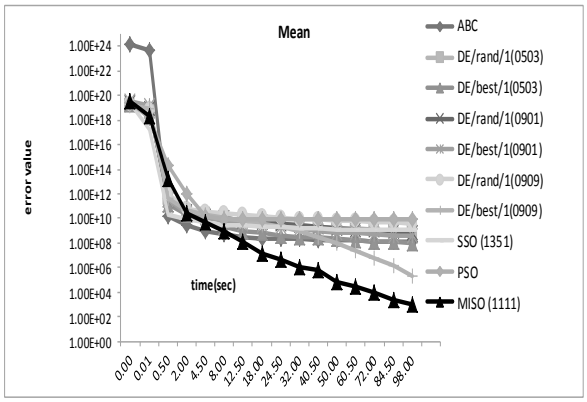

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

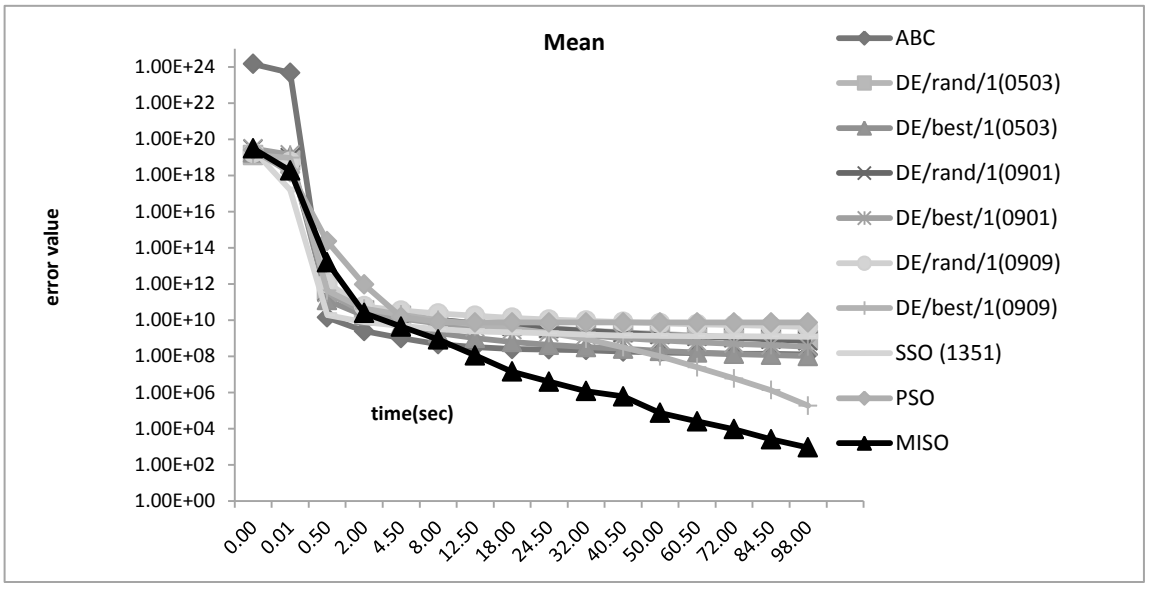
BM3


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

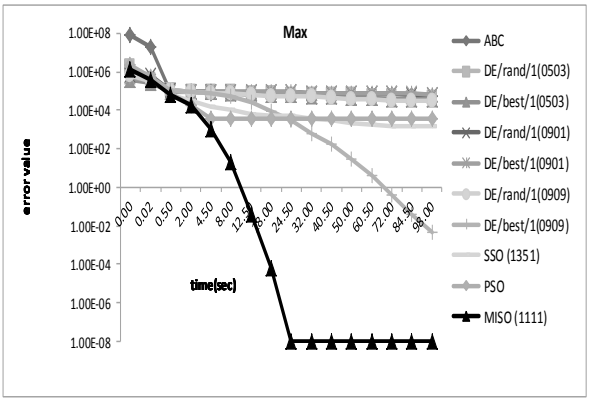

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

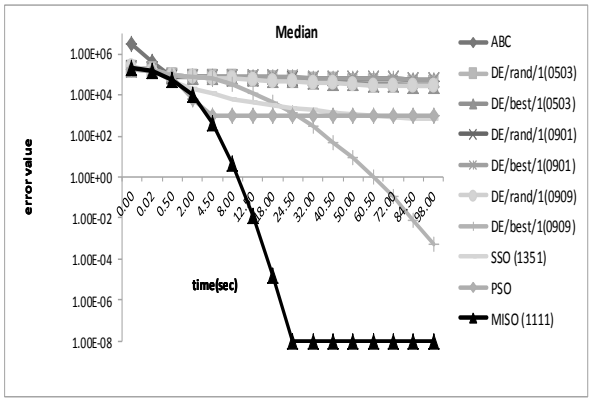

Min
error value
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
PSO
MISO (1111)

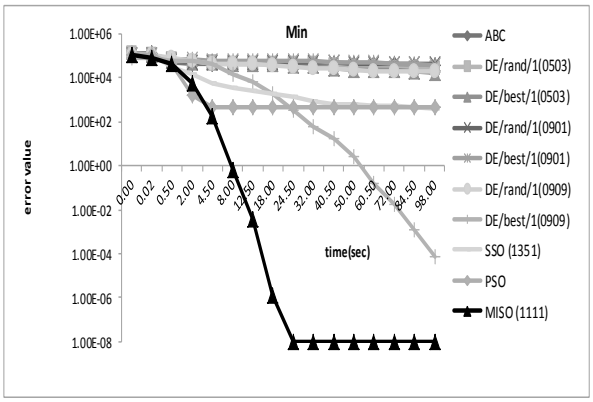

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

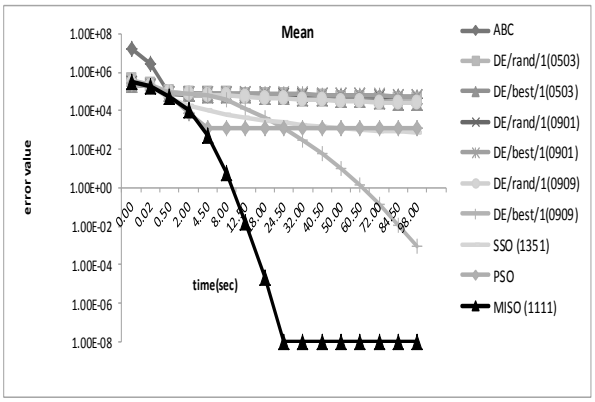

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

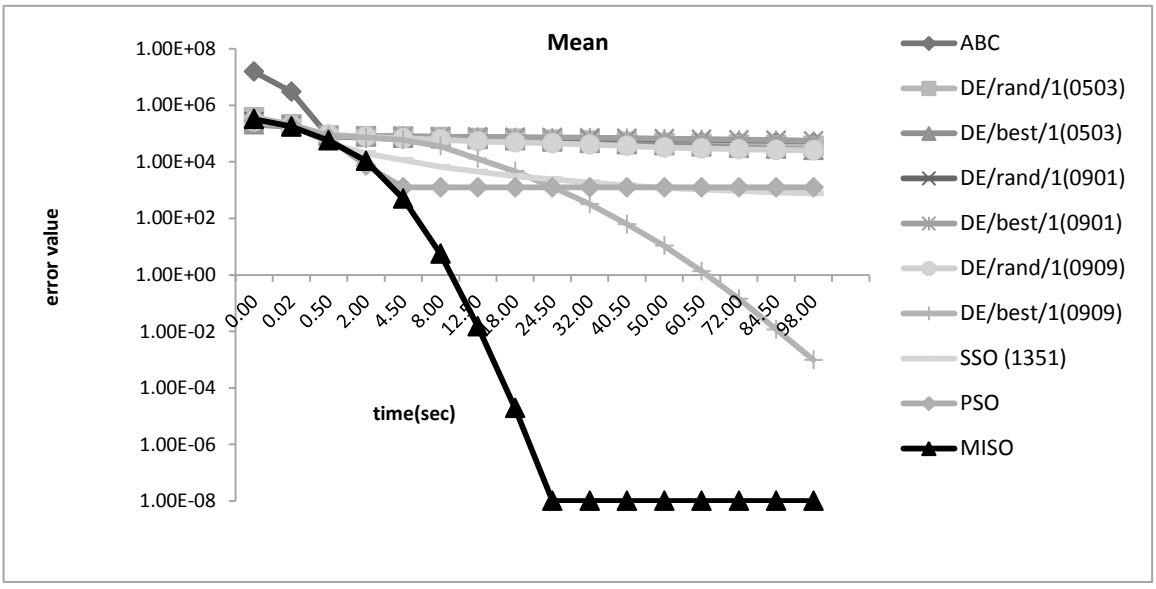
BM4


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

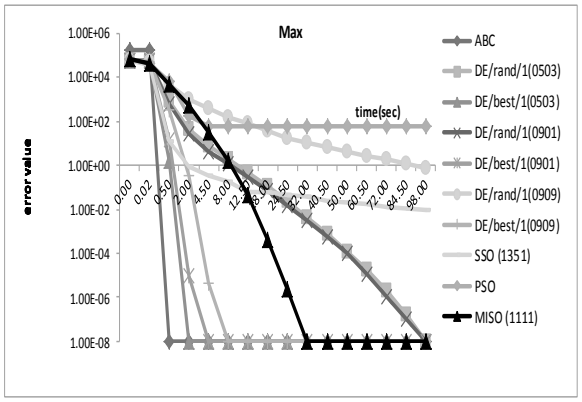

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

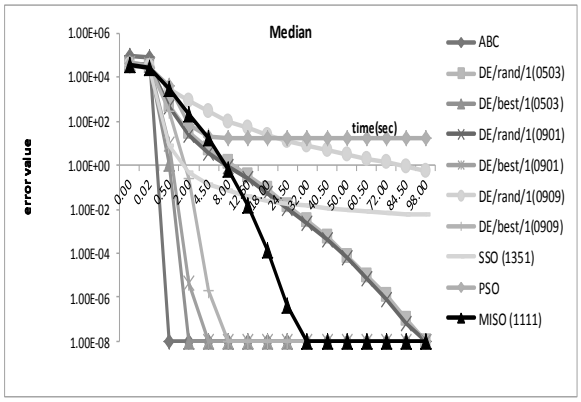

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

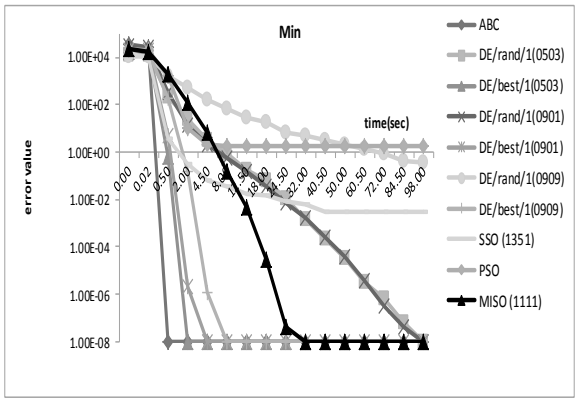

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

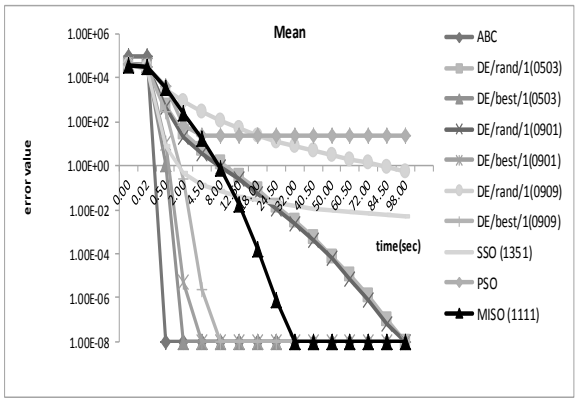

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

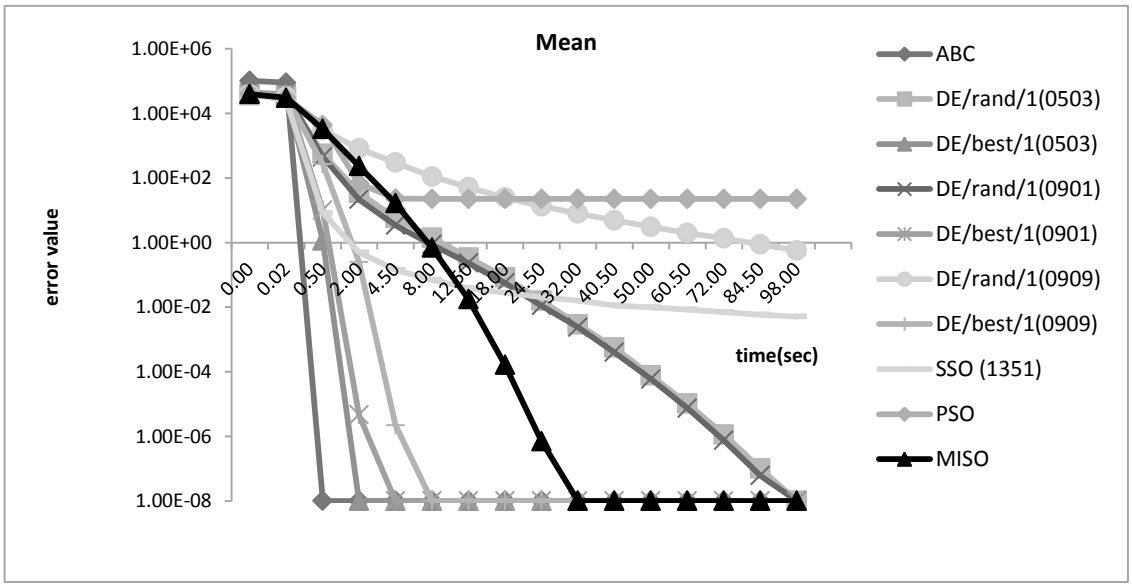
BM5


Max
error value
time(sec)
1.00E+06
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

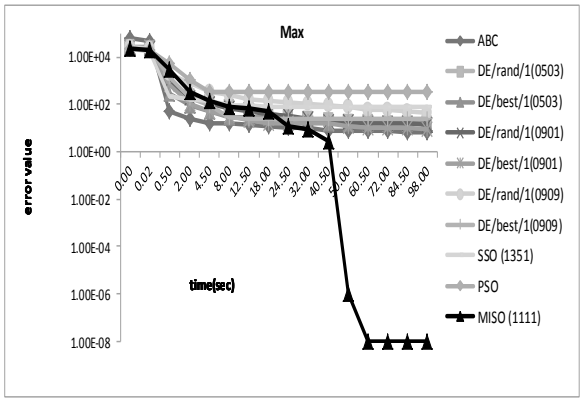

Median
error value
time(sec)
1.00E+06
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

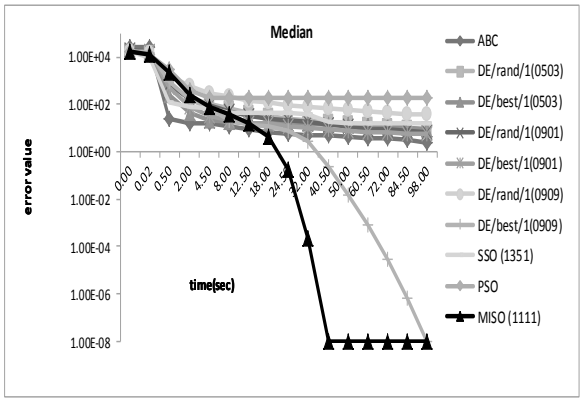

Min
error value
time(sec)
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

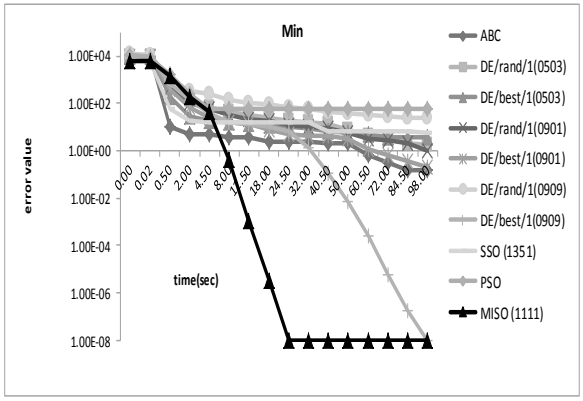

Mean
error value
time(sec)
1.00E+06
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

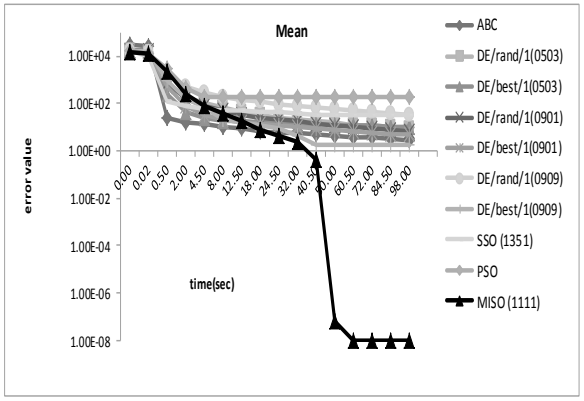

Mean
error value
time(sec)
1.00E+06
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

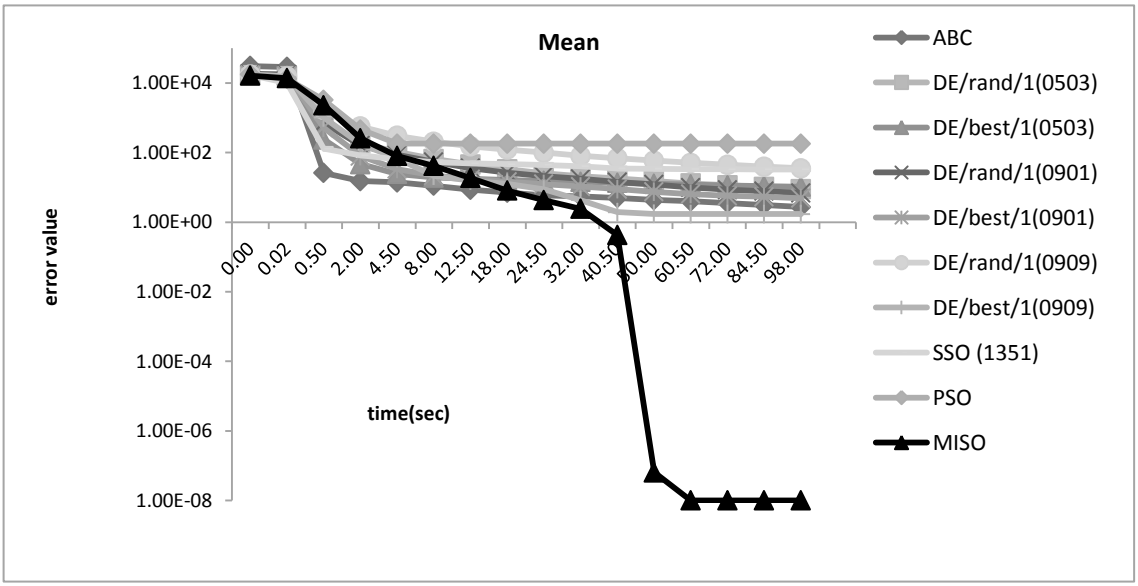
BM6


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

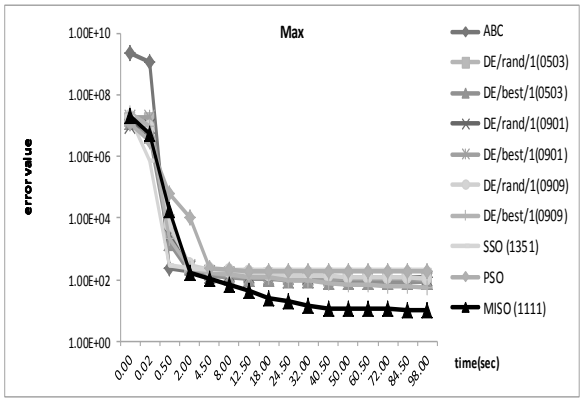

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

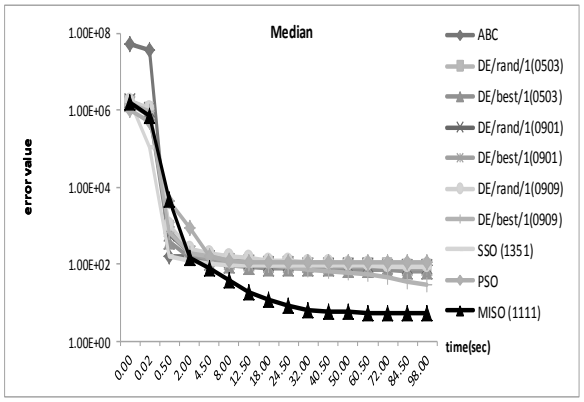

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

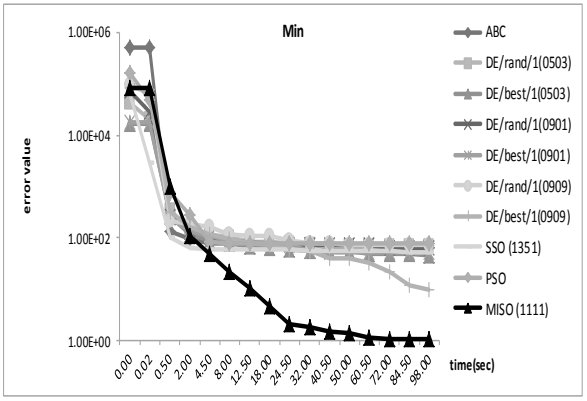

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

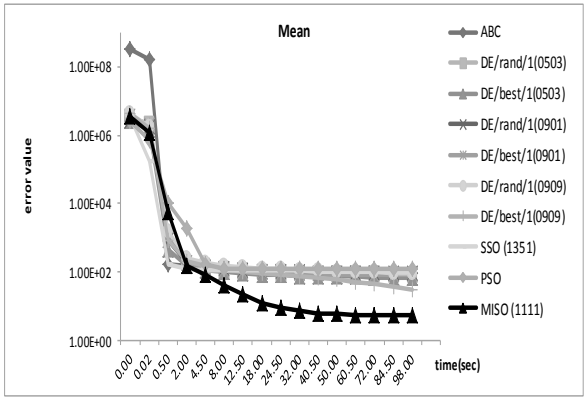

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

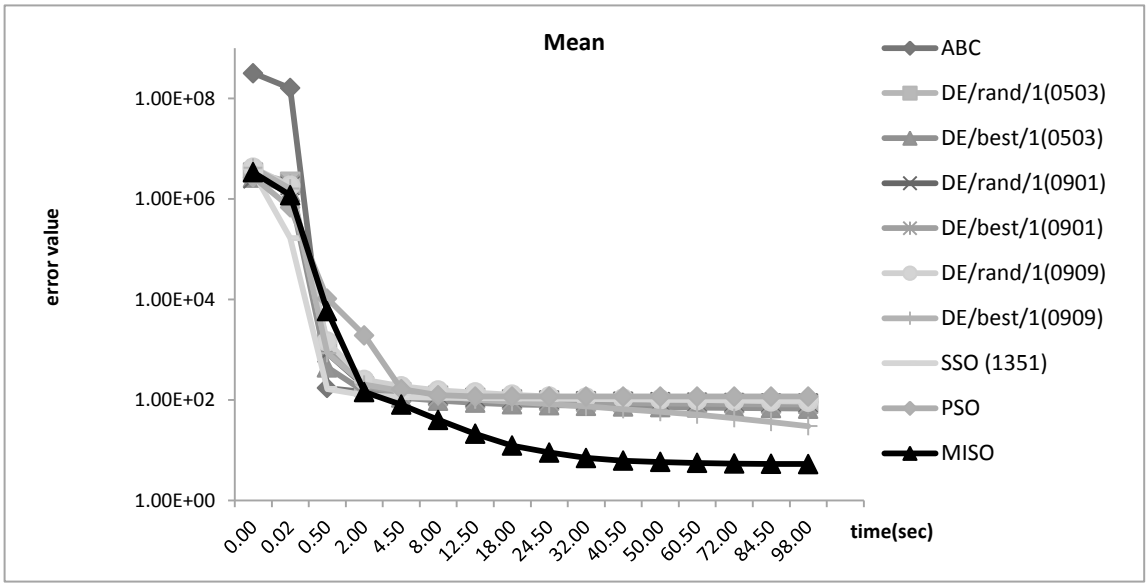
BM7


Max
error value
1.00E+10
1.00E+08
1.00E+06
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

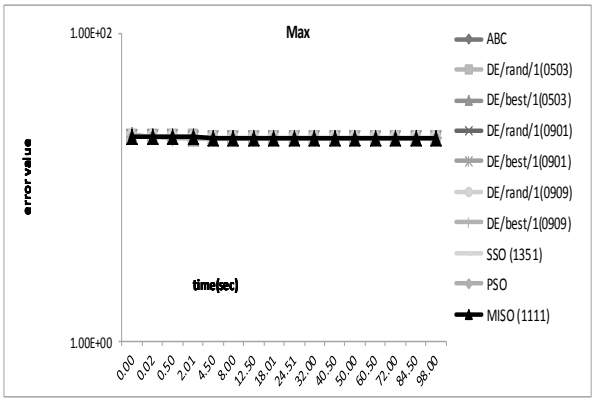

Median
error value
1.00E+08
1.00E+06
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

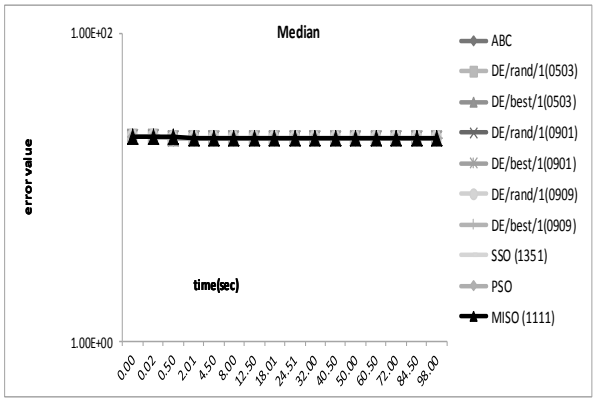

Min
error value
1.00E+06
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

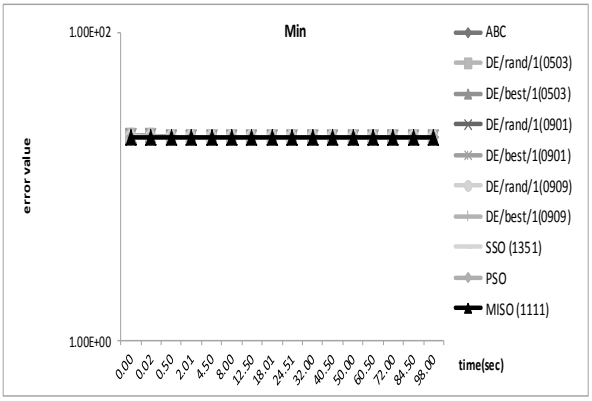

Mean
error value
1.00E+08
1.00E+06
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

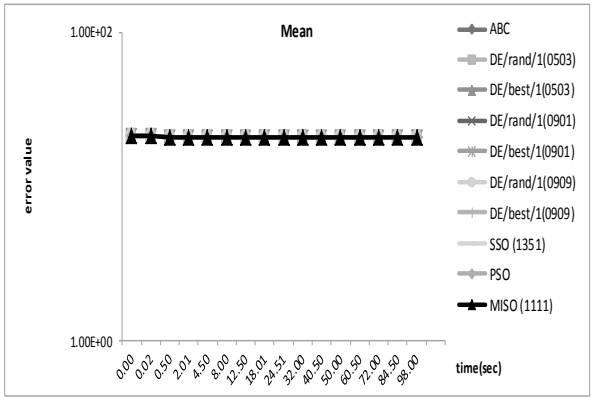

Mean
error value
1.00E+08
1.00E+06
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

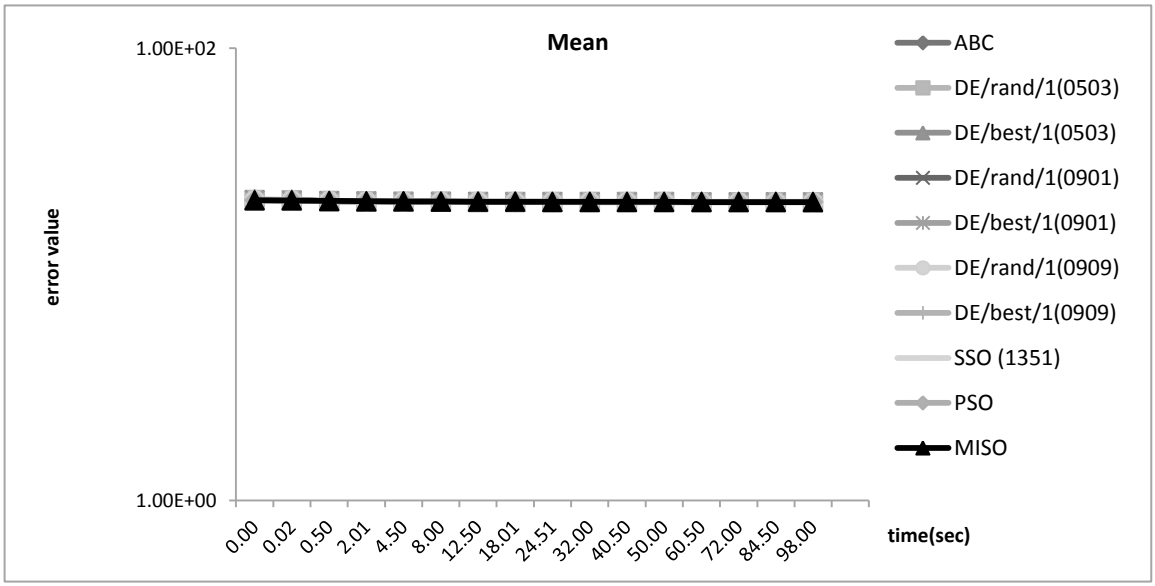
BM8


Max
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.01 4.50 8.00 12.50 18.01 24.51 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

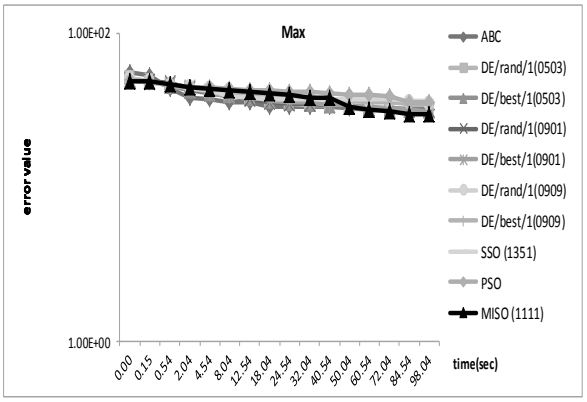

Median
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.01 4.50 8.00 12.50 18.01 24.51 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

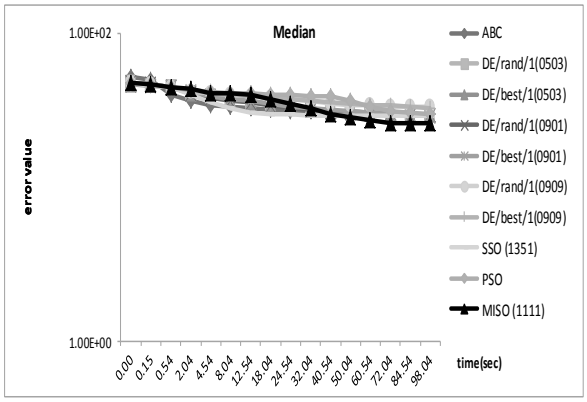

Min
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.01 4.50 8.00 12.50 18.01 24.51 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

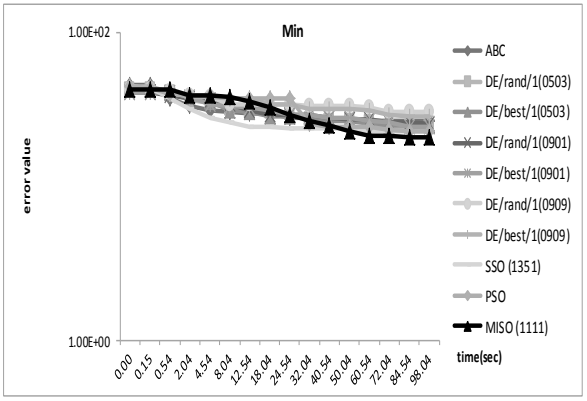

Mean
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.01 4.50 8.00 12.50 18.01 24.51 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

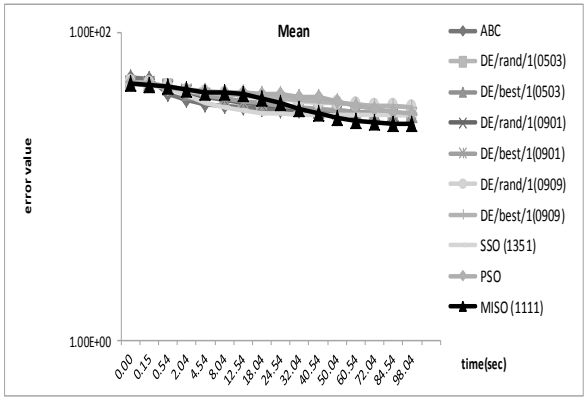

Mean
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.01 4.50 8.00 12.50 18.01 24.51 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

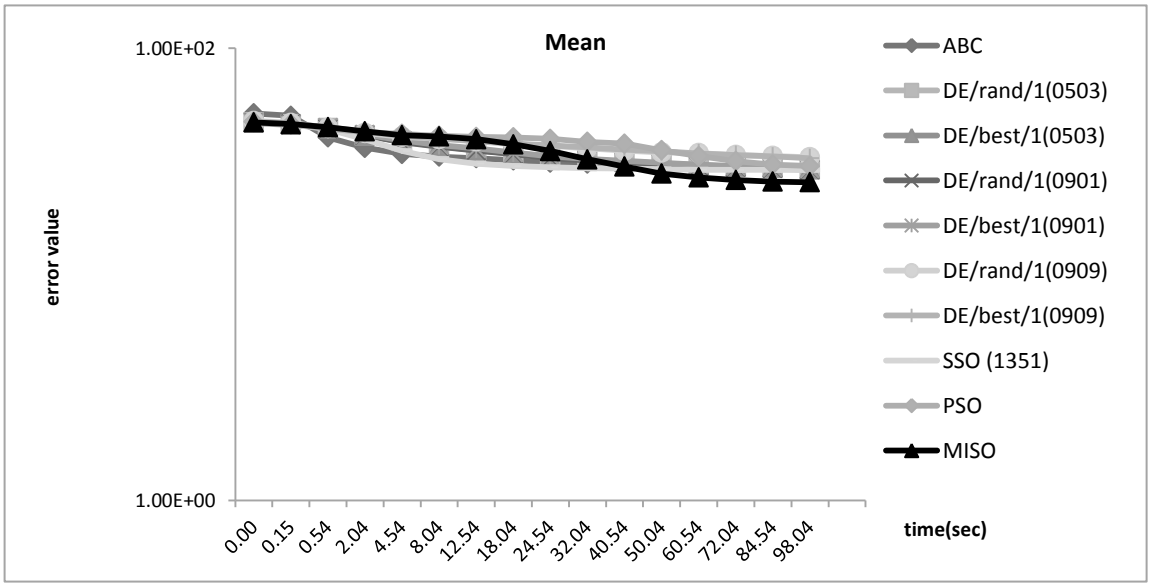
BM9


Max
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.15 0.54 2.04 4.54 8.04 12.54 18.04 24.54 32.04 40.54 50.04 60.54 72.04 84.54 98.04

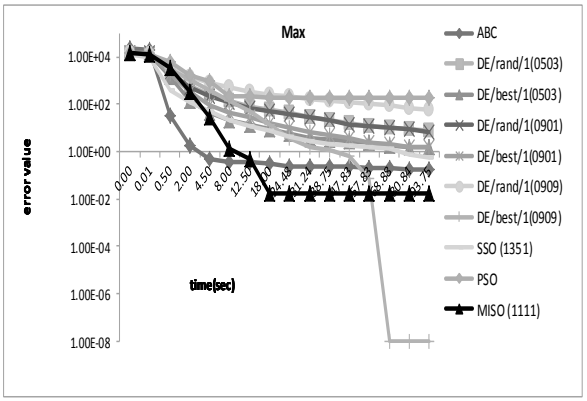

Median
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.15 0.54 2.04 4.54 8.04 12.54 18.04 24.54 32.04 40.54 50.04 60.54 72.04 84.54 98.04

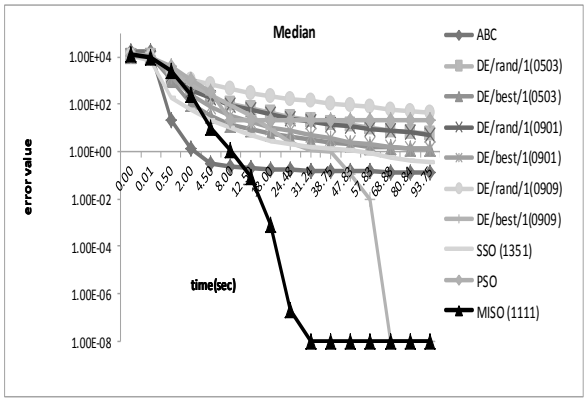

Min
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.15 0.54 2.04 4.54 8.04 12.54 18.04 24.54 32.04 40.54 50.04 60.54 72.04 84.54 98.04

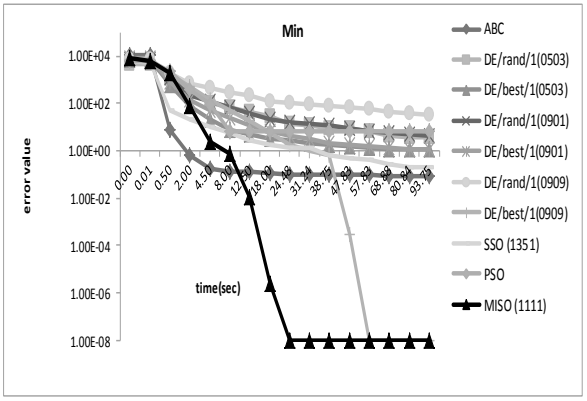

Mean
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.15 0.54 2.04 4.54 8.04 12.54 18.04 24.54 32.04 40.54 50.04 60.54 72.04 84.54 98.04

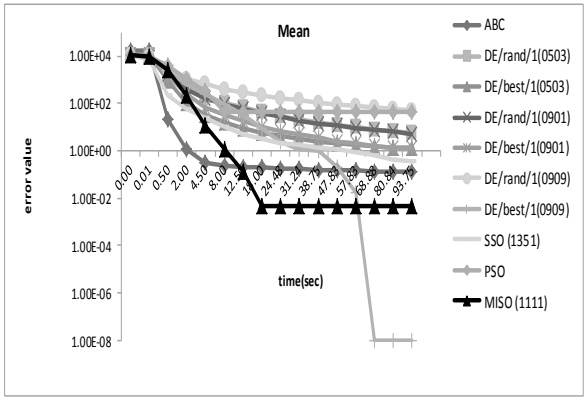

Mean
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO
0.00 0.15 0.54 2.04 4.54 8.04 12.54 18.04 24.54 32.04 40.54 50.04 60.54 72.04 84.54 98.04

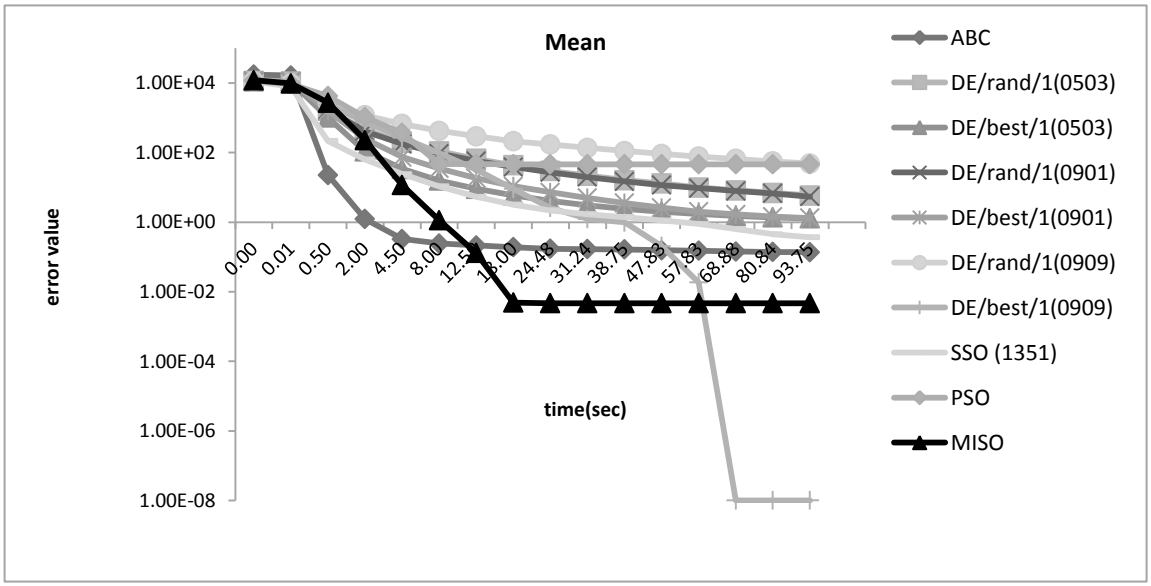
BM10


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

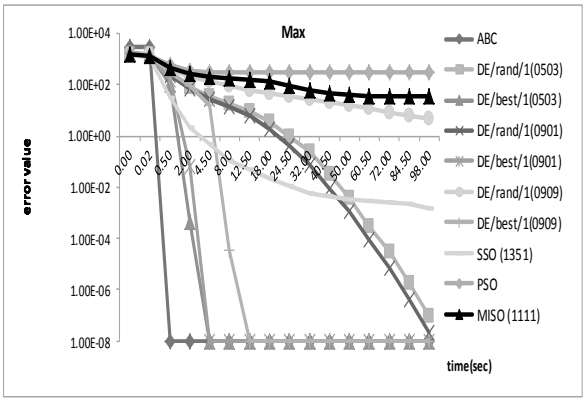

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

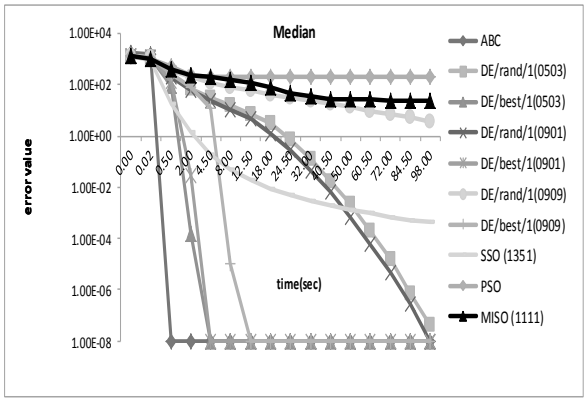

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

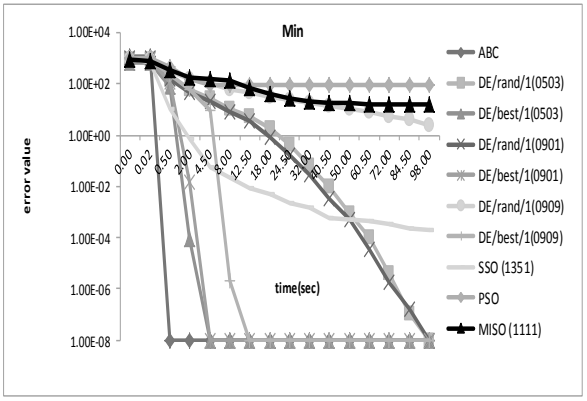

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

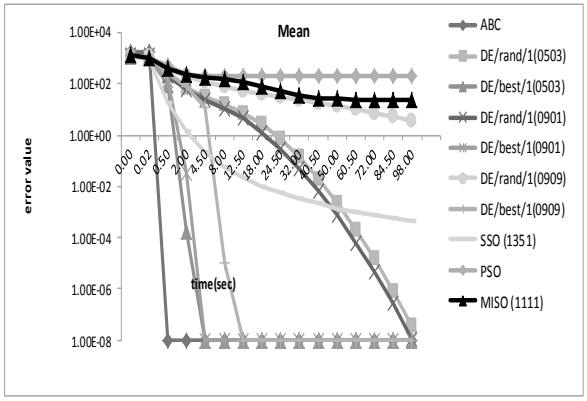

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

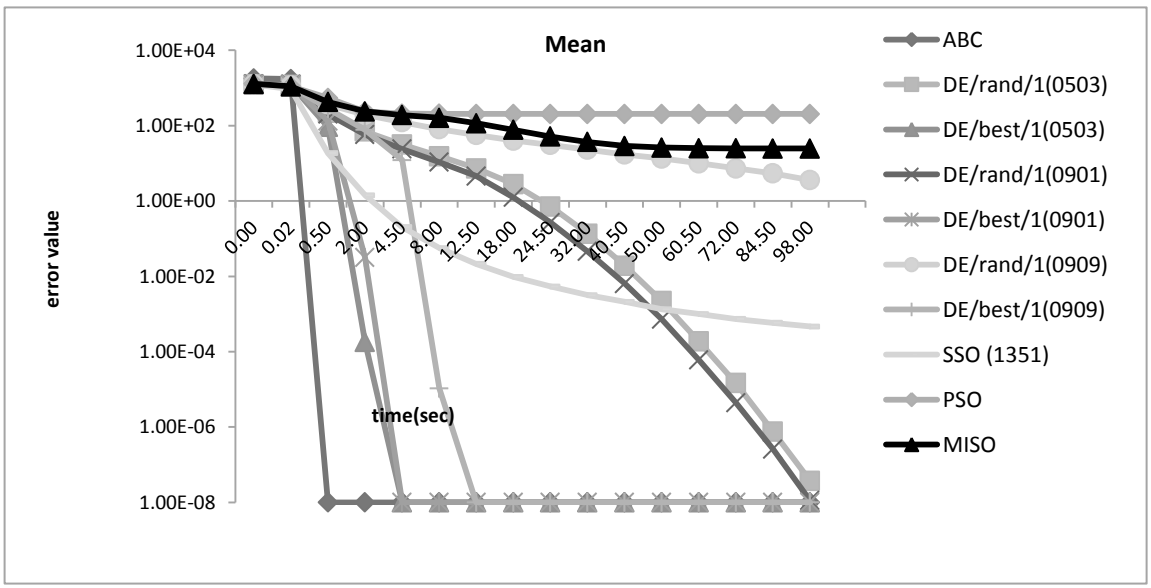
BM11


Max
error value
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

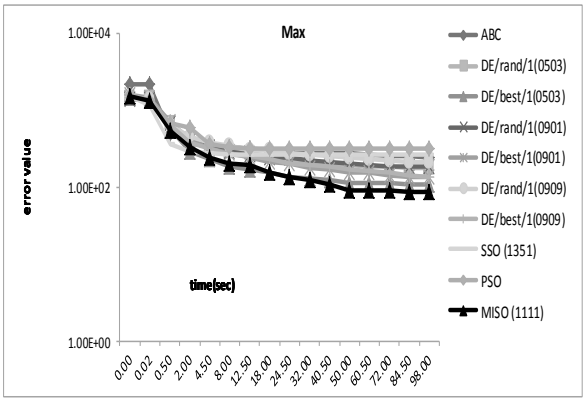

Median
error value
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

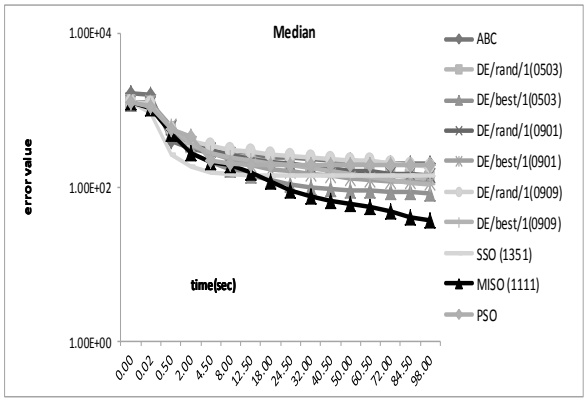

Min
error value
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

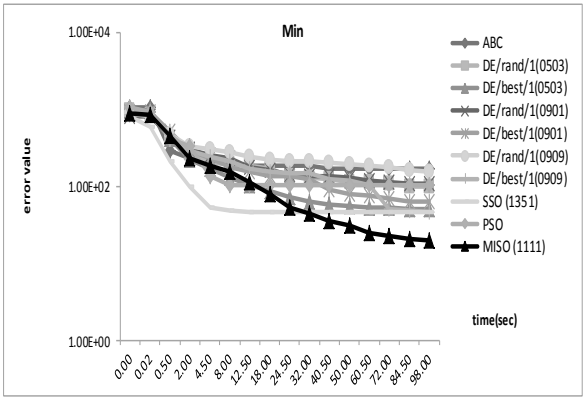

Mean
error value
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

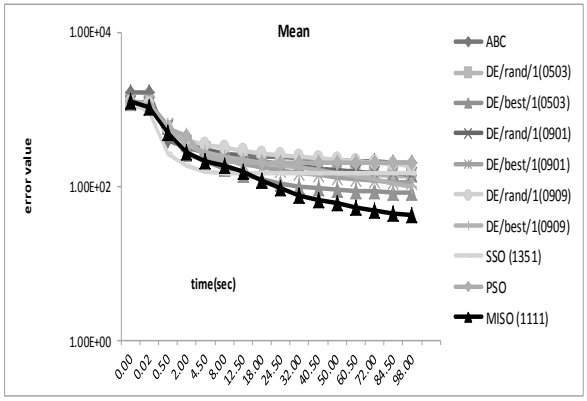

Mean
error value
1.00E+04
1.00E+02
1.00E+00
1.00E-02
1.00E-04
1.00E-06
1.00E-08
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

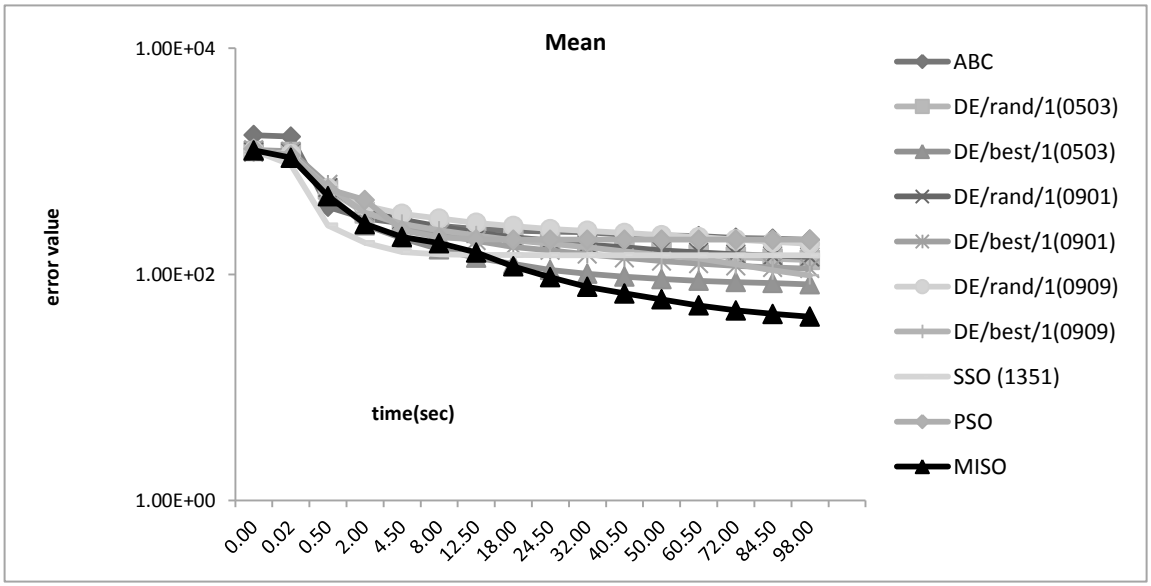
BM12


Max
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

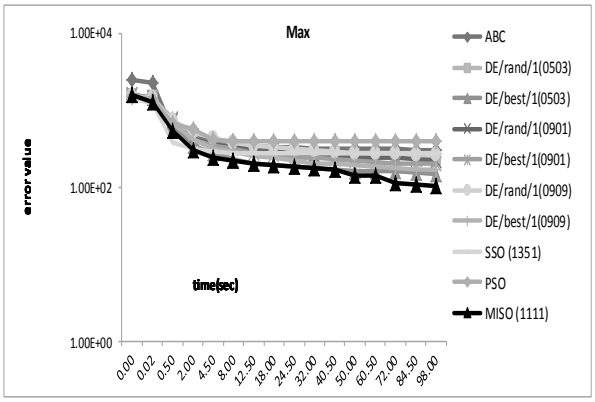

Median
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
MISO (1111)
PSO

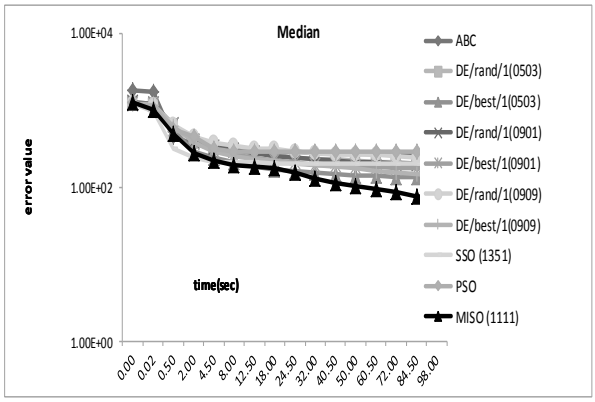

Min
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

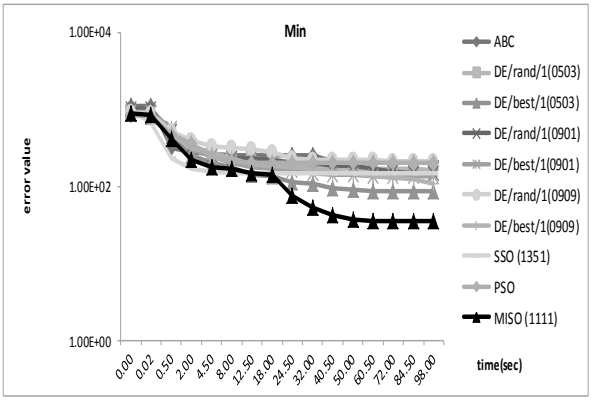

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

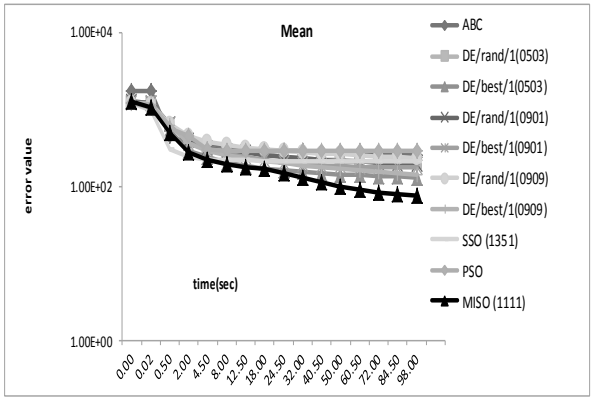

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

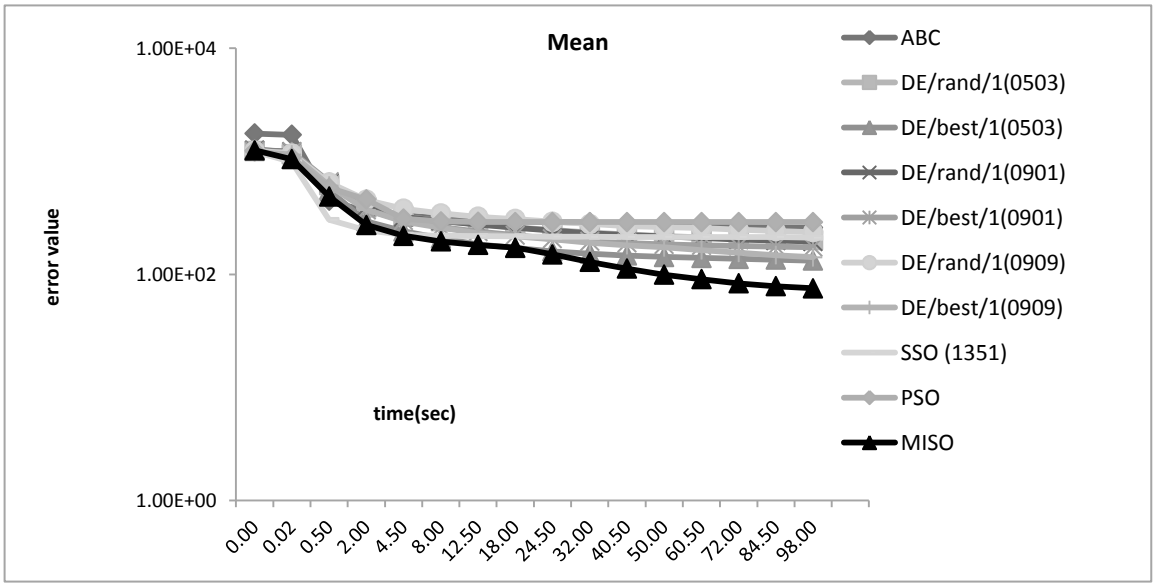
BM13


Max
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

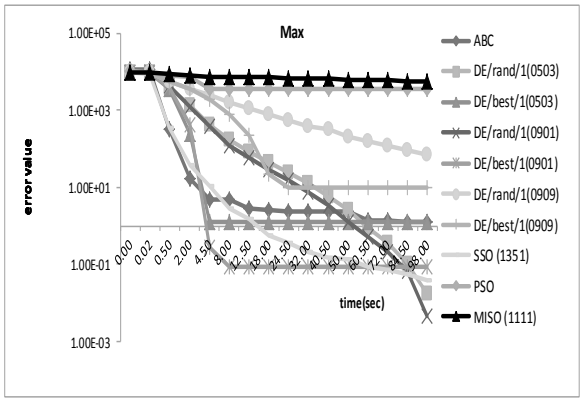

Median
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

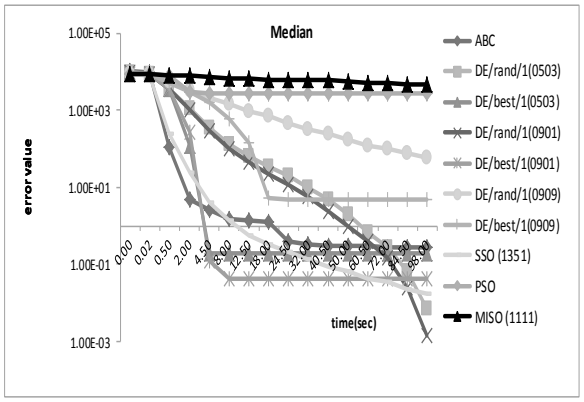

Min
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

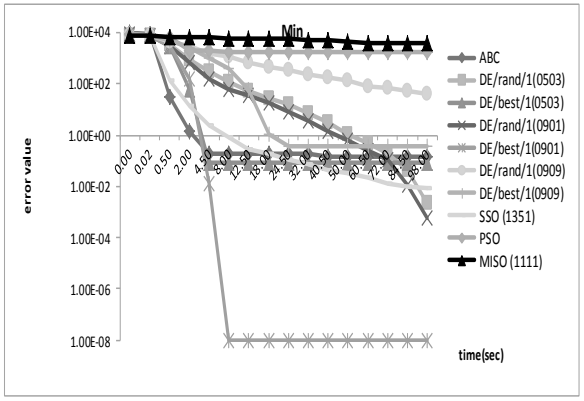

Mean
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

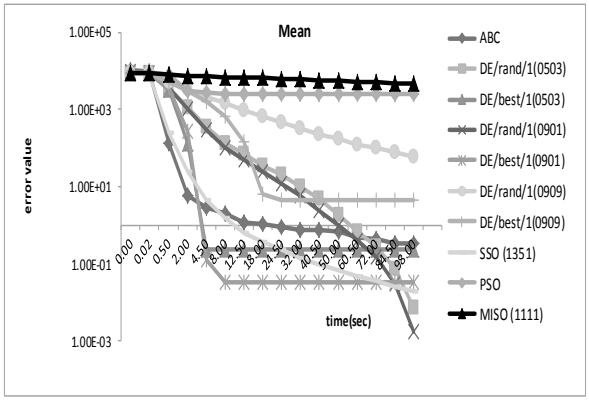

Mean
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

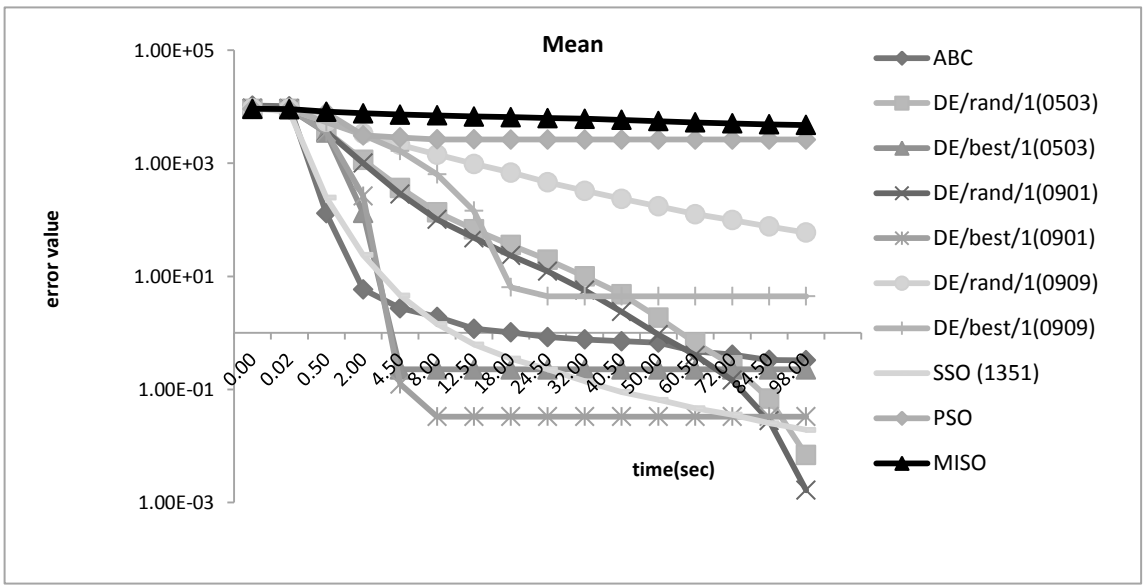
BM14


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

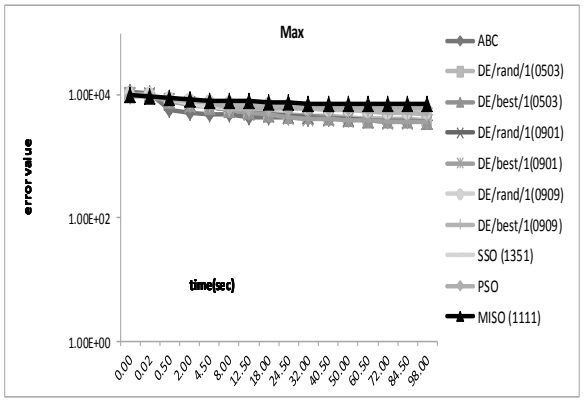

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

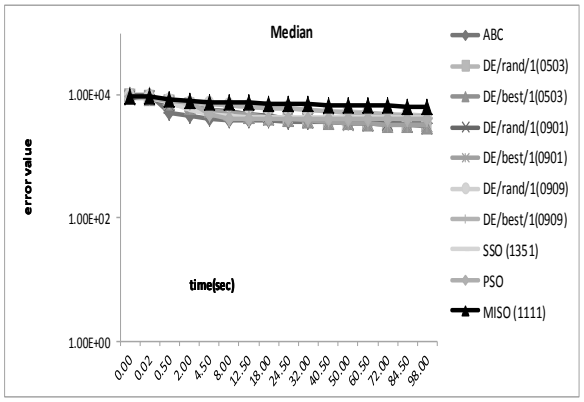

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

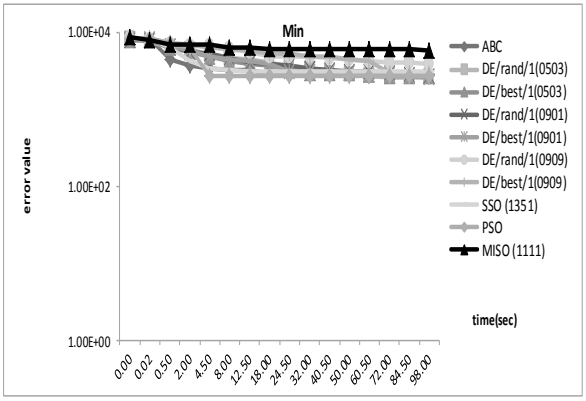

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

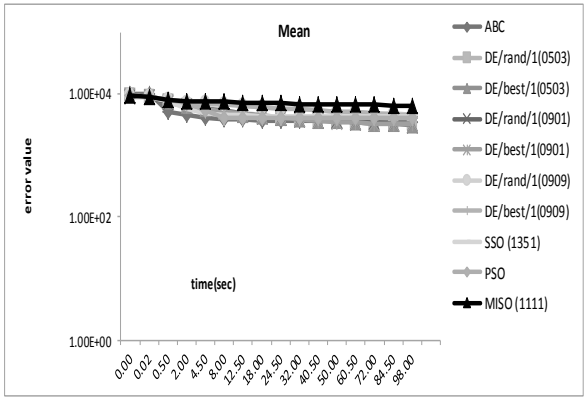

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

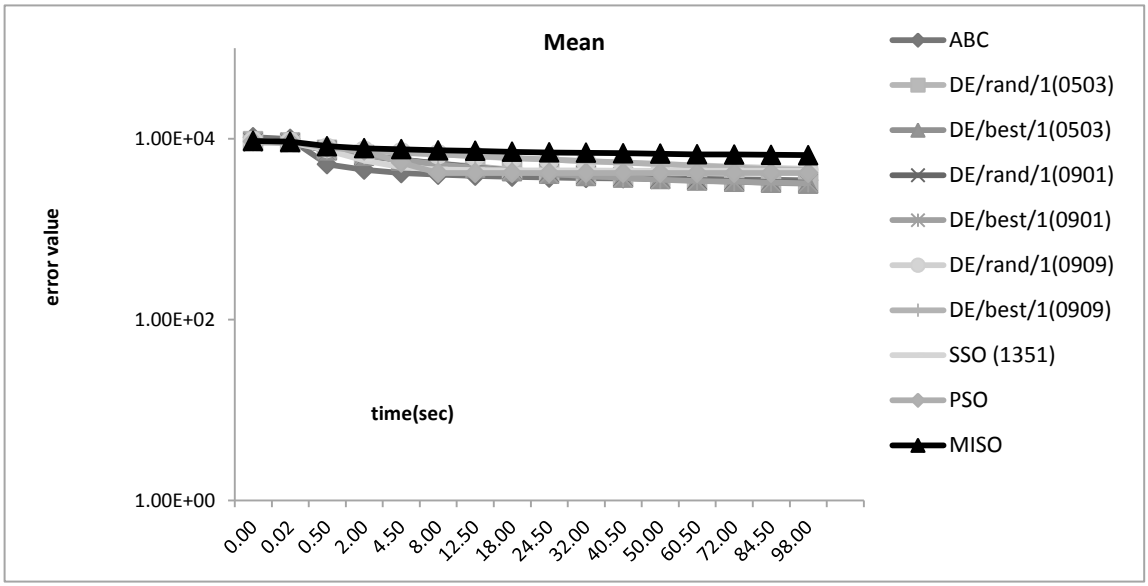
BM15


Max
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

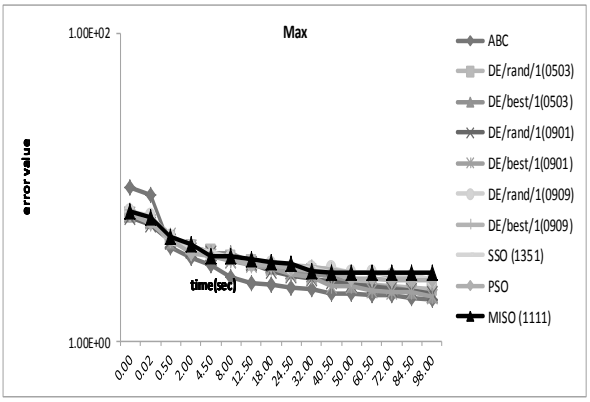

Median
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

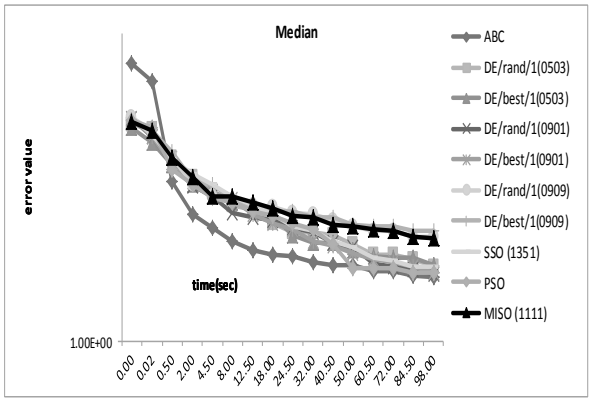

Min
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

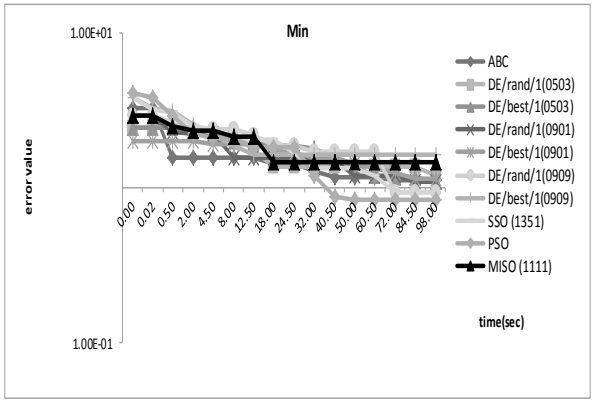

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

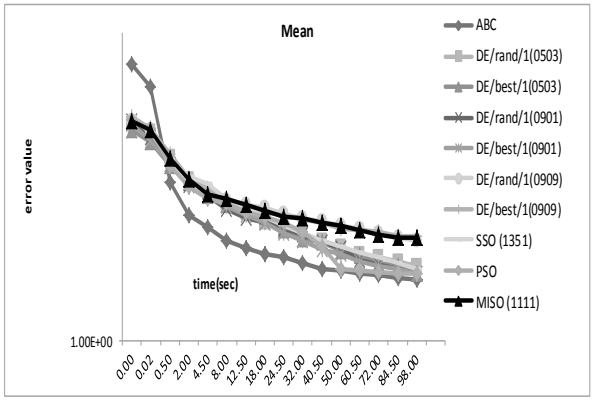

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

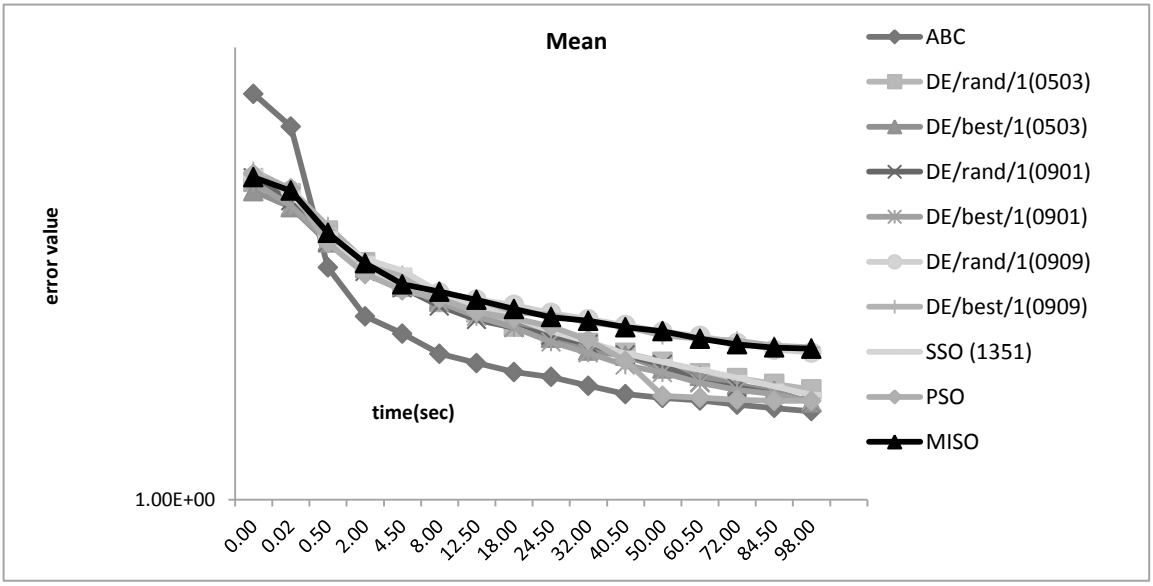
BM16


Max
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

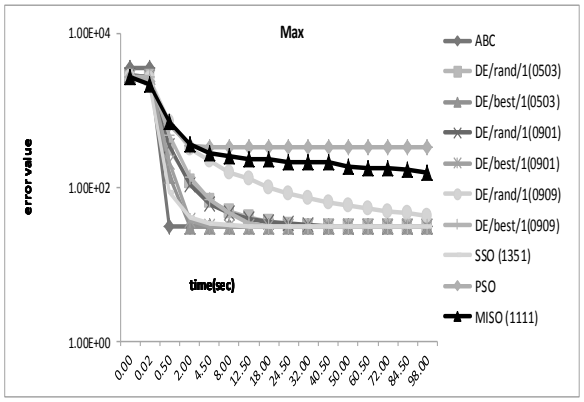

Median
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

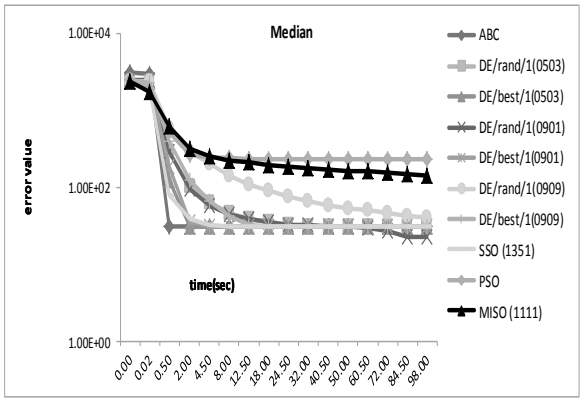

Min
1.00E+01
1.00E-01
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

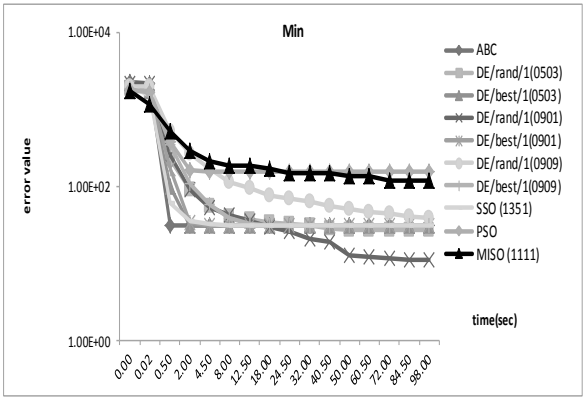

Mean
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

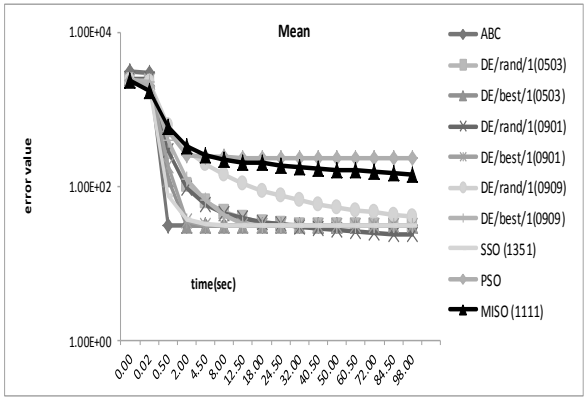

Mean
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

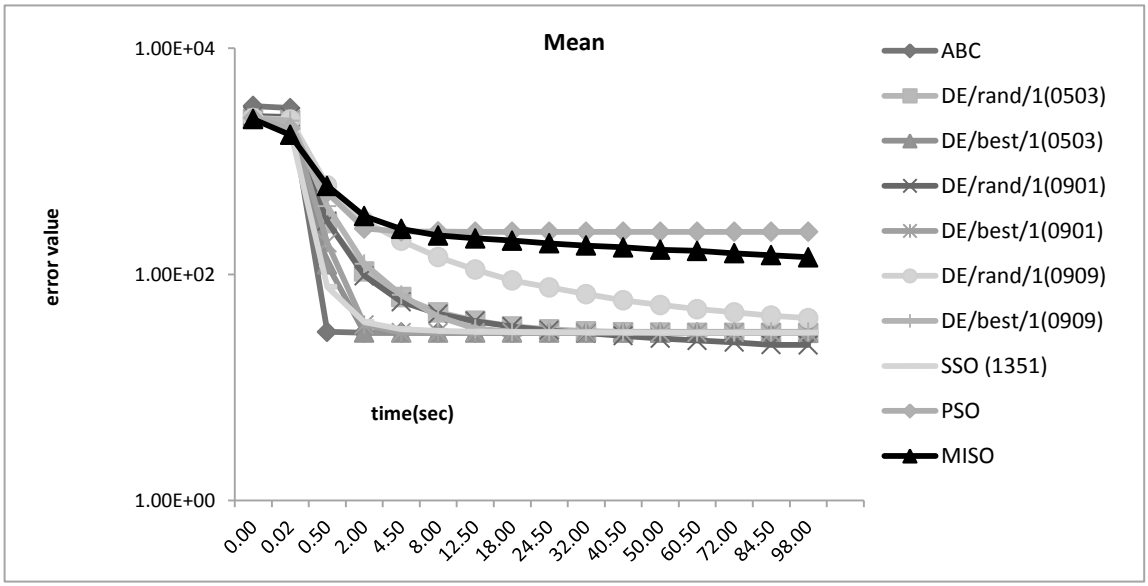
BM17


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

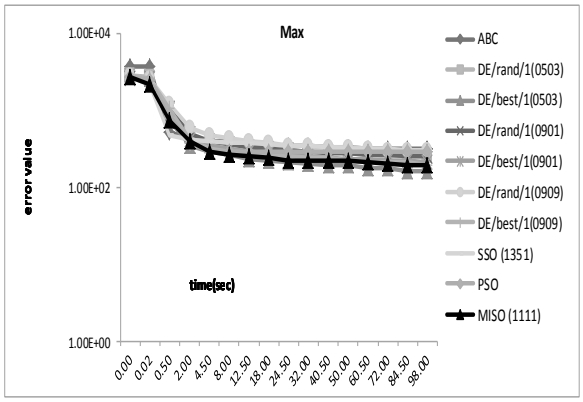

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

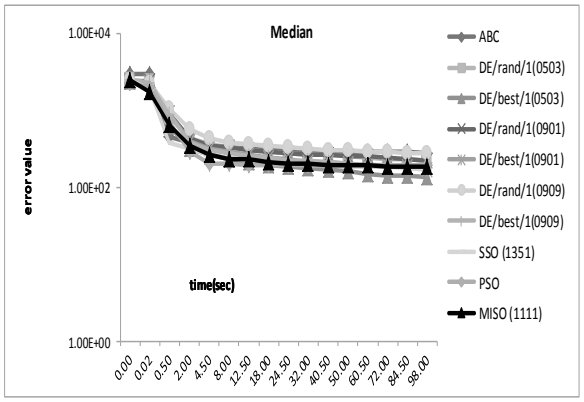

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

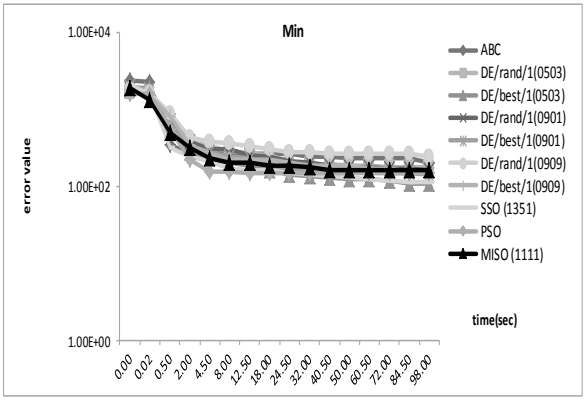

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

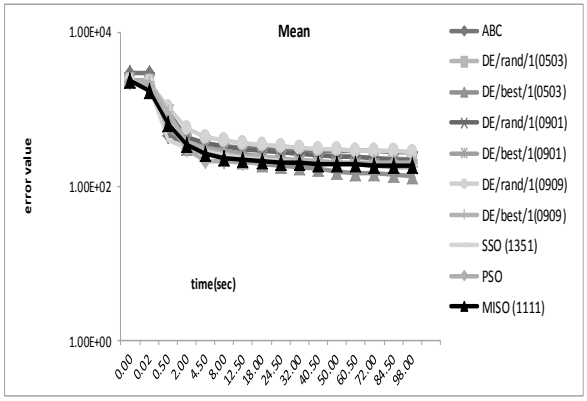

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

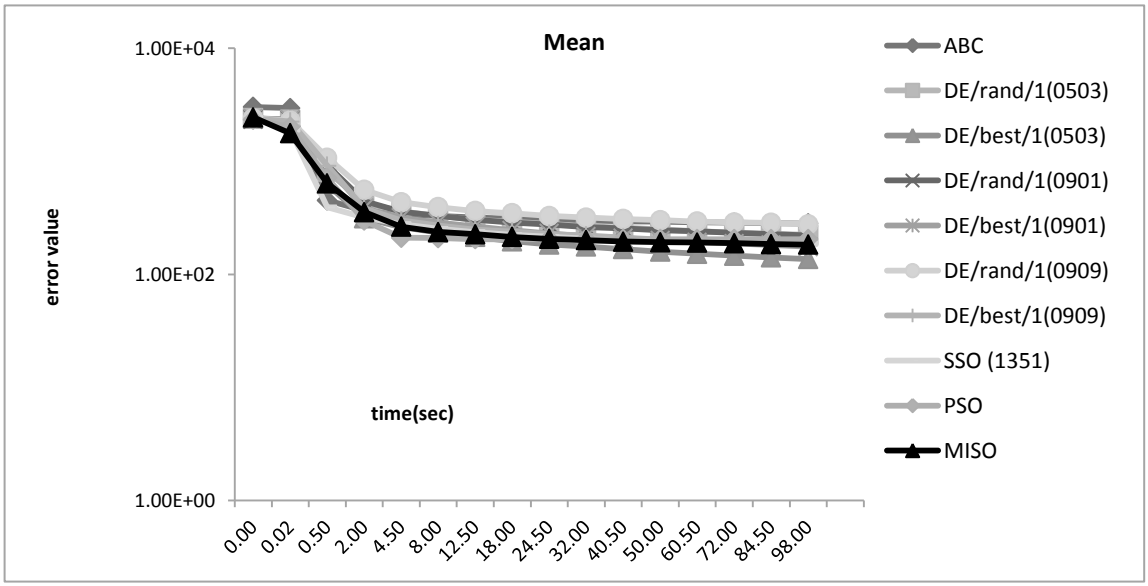
BM18


Max
error value
time(sec)
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

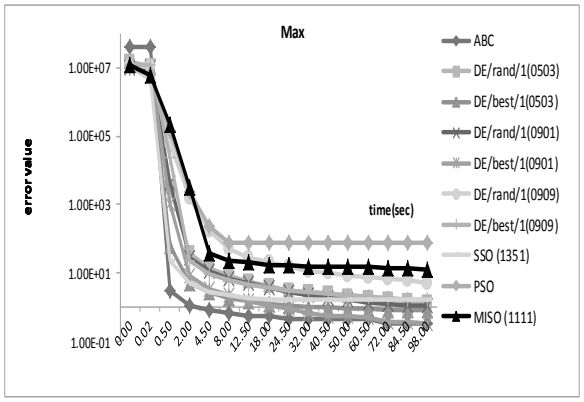

Median
error value
time(sec)
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

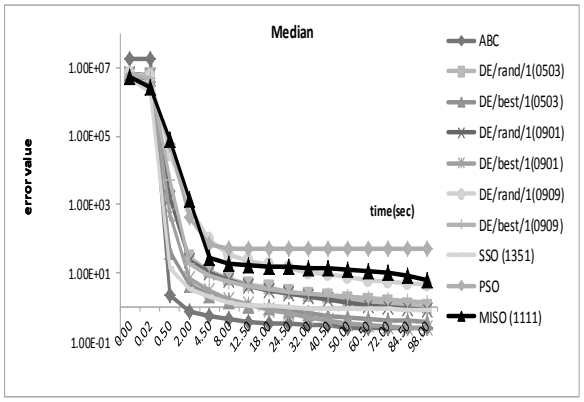

Min
error value
time(sec)
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

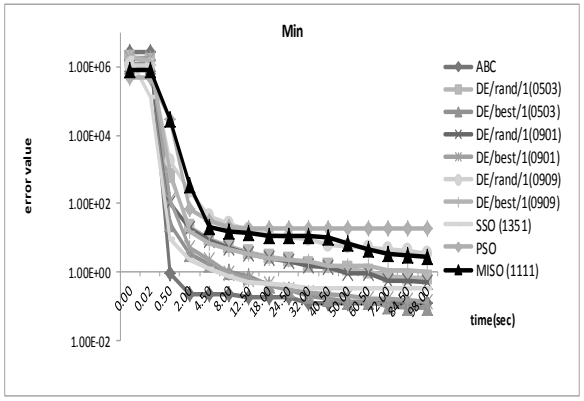

Mean
error value
time(sec)
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

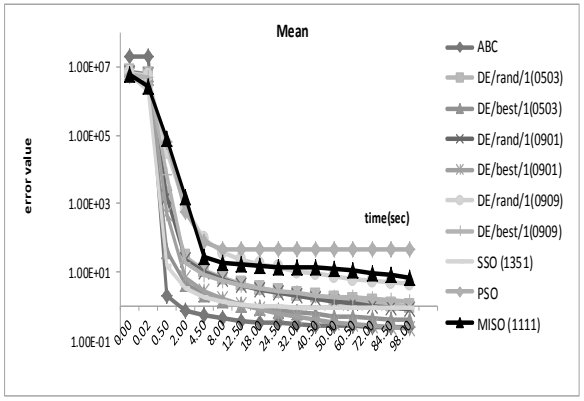

Mean
error value
time(sec)
1.00E+04
1.00E+02
1.00E+00
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

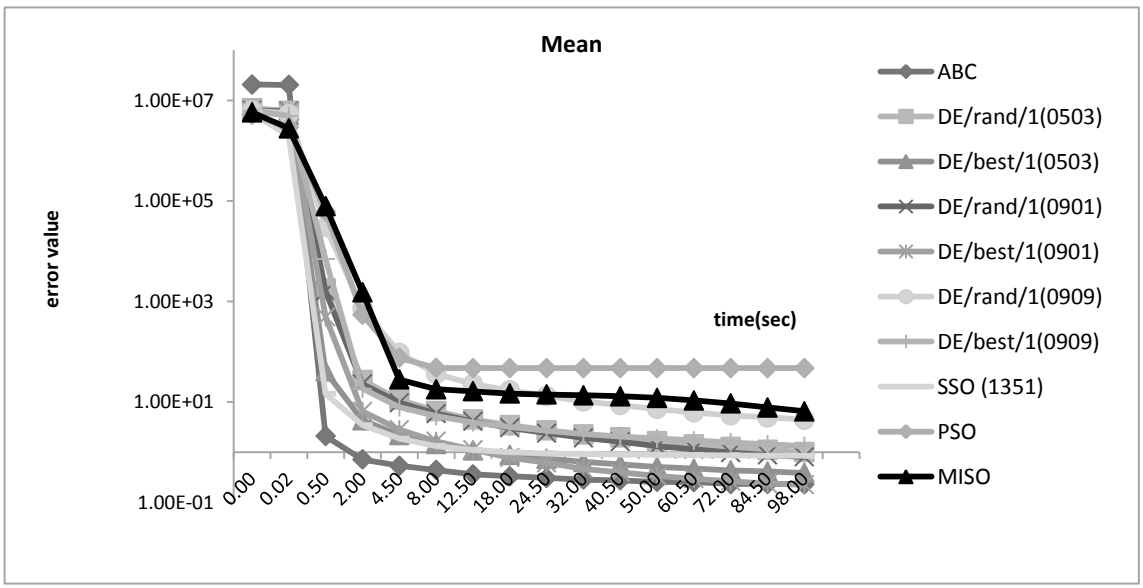
BM19


Max
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

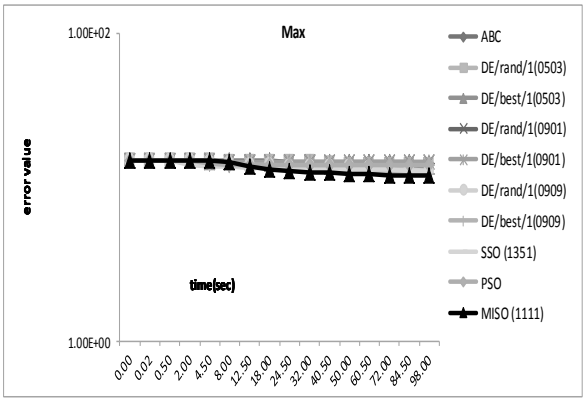

Median
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

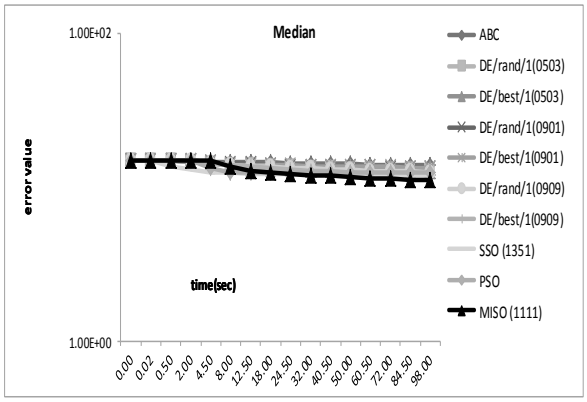

Min
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

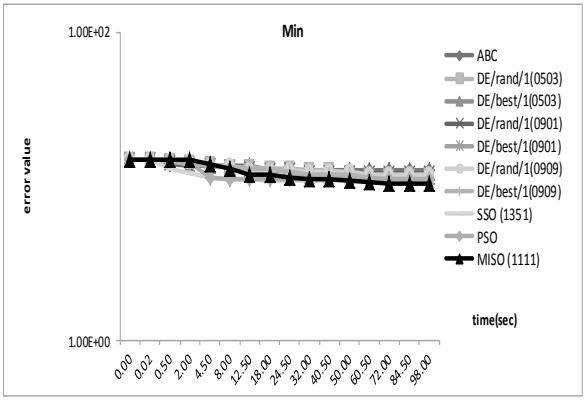

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

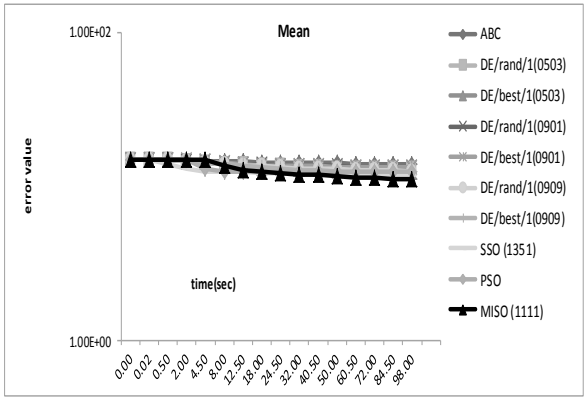

Mean
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

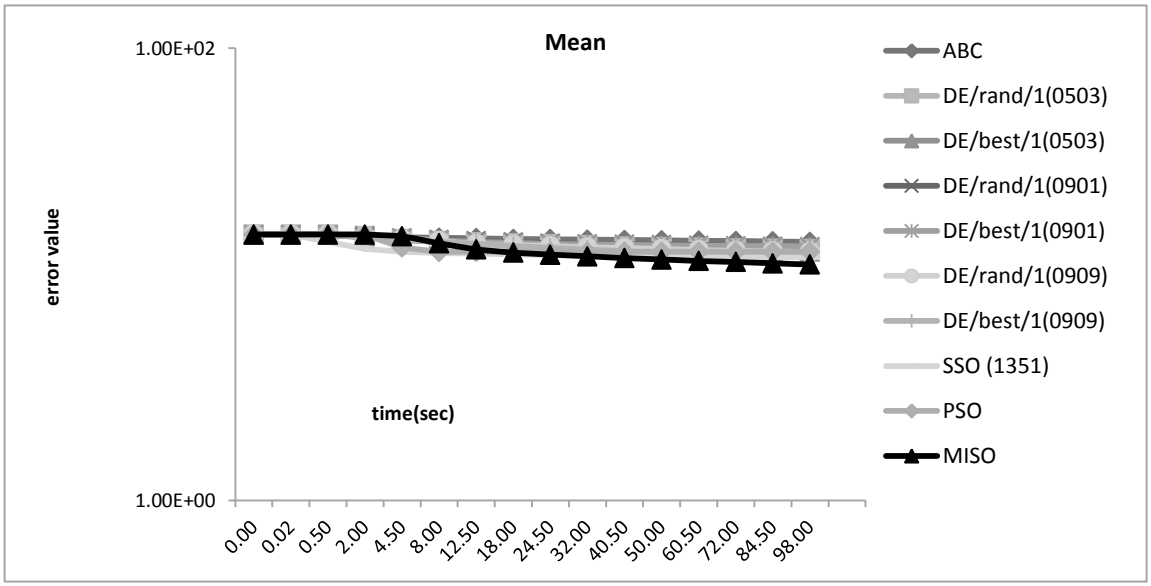
BM20


Max
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00

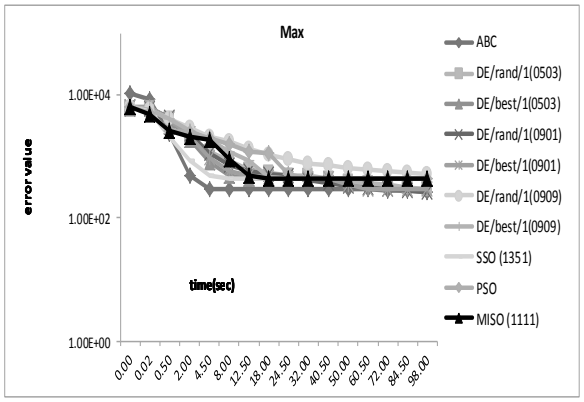

Median
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00

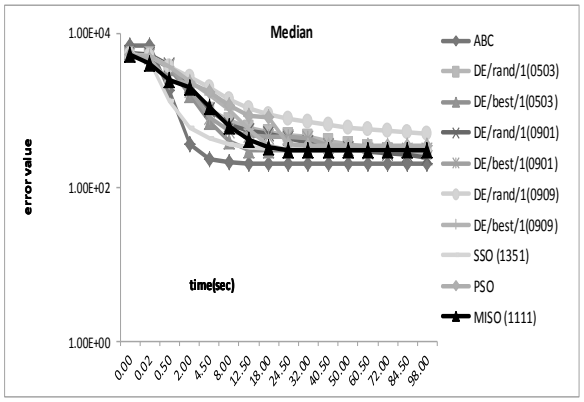

Min
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00

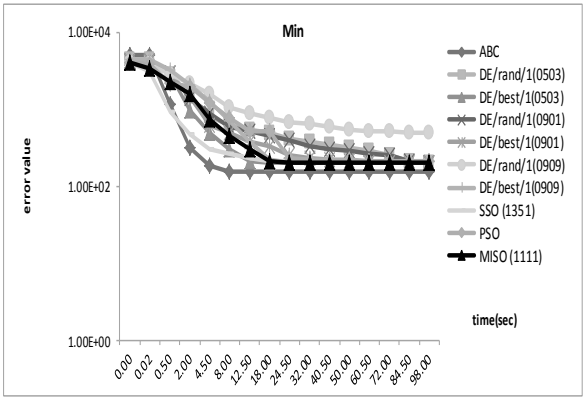

Mean
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00

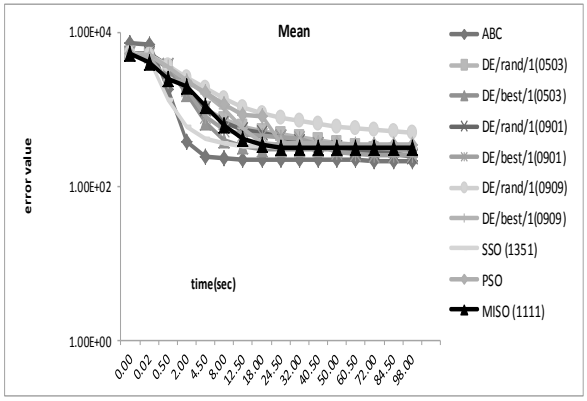

Mean
1.00E+02
1.00E+00
error value
time(sec)
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00

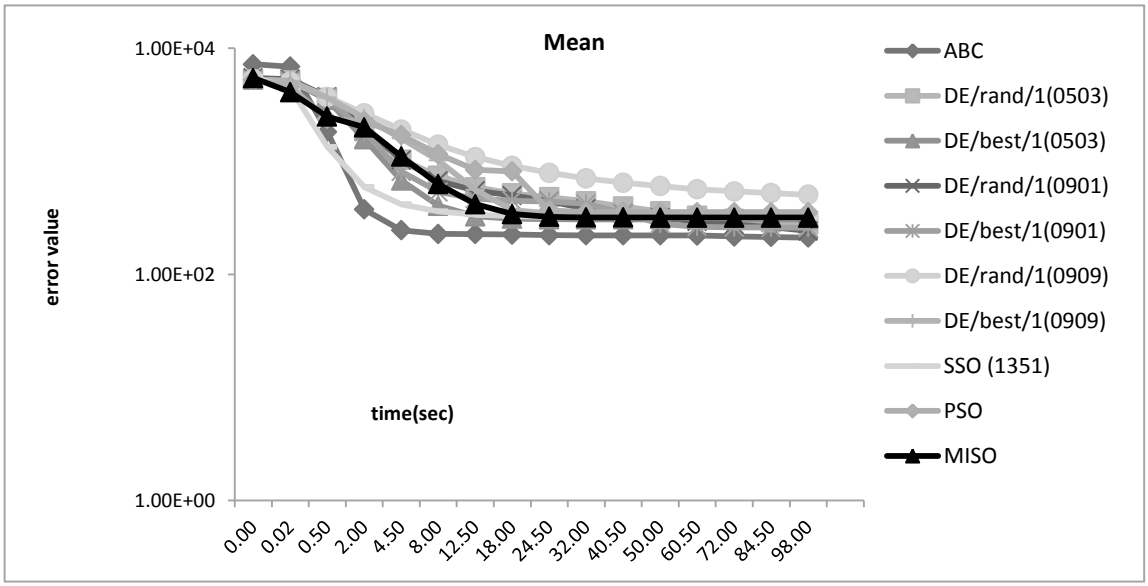
BM21


Max
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

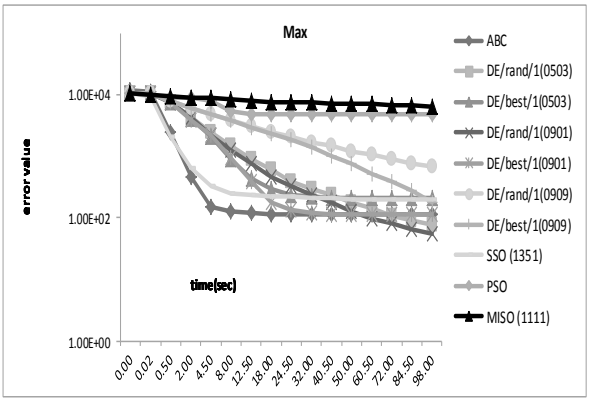

Median
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

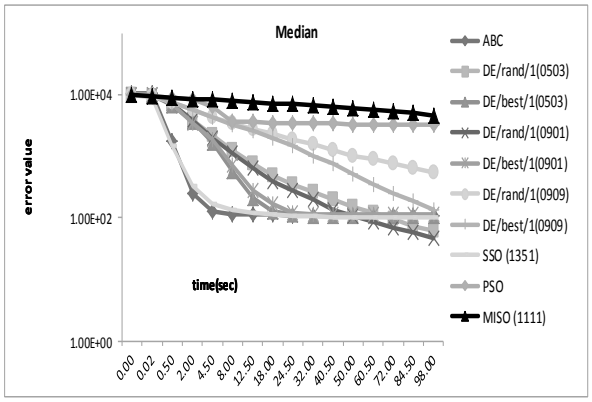

Min
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

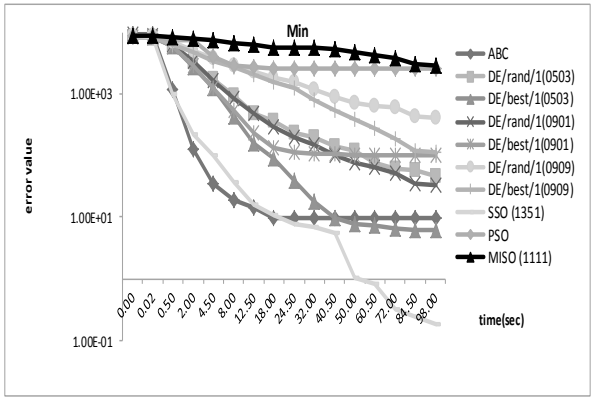

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

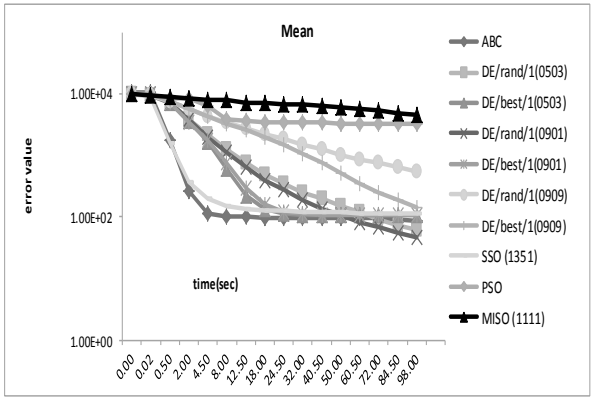

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

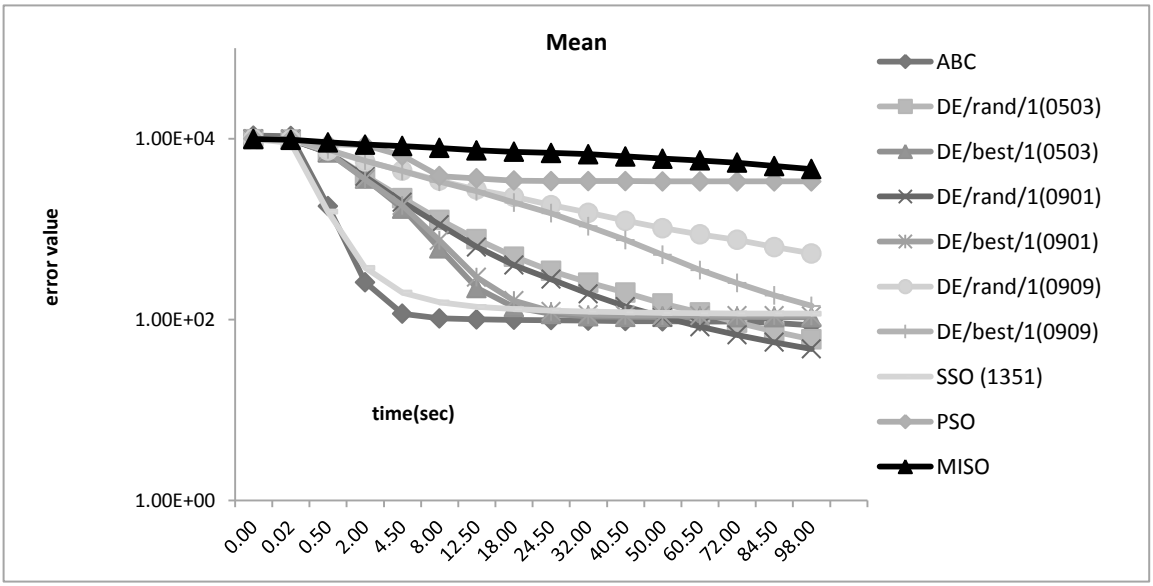
BM22


Max
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

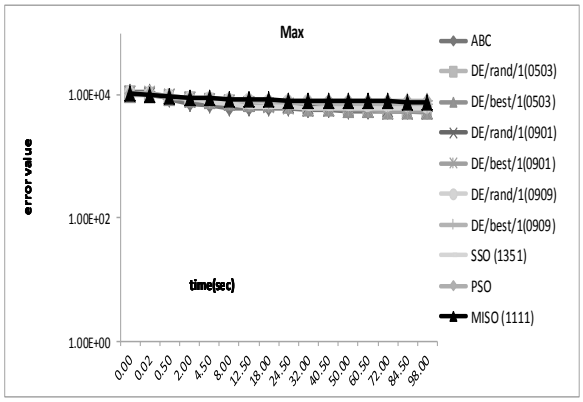

Median
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

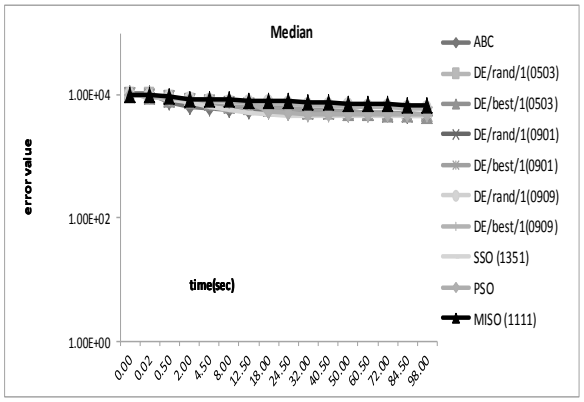

Min
error value
1.00E+03
1.00E+01
1.00E-01
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

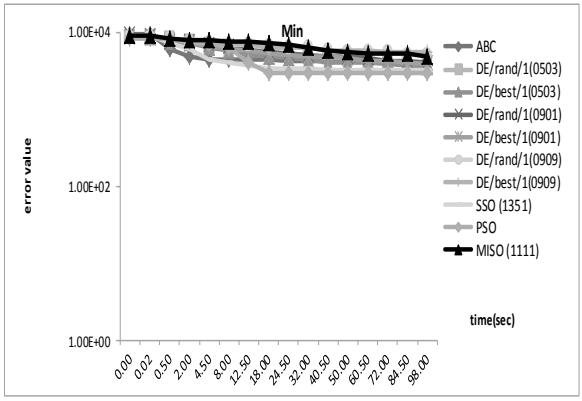

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

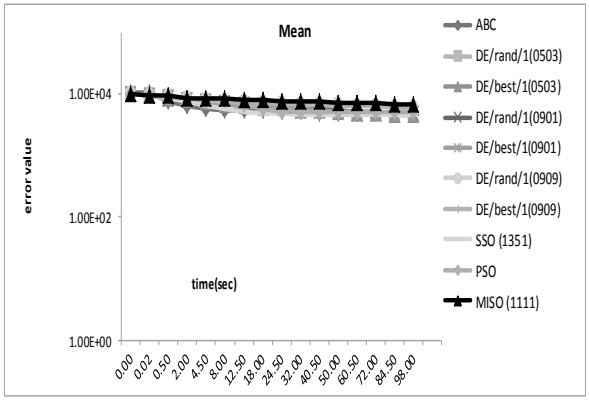

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

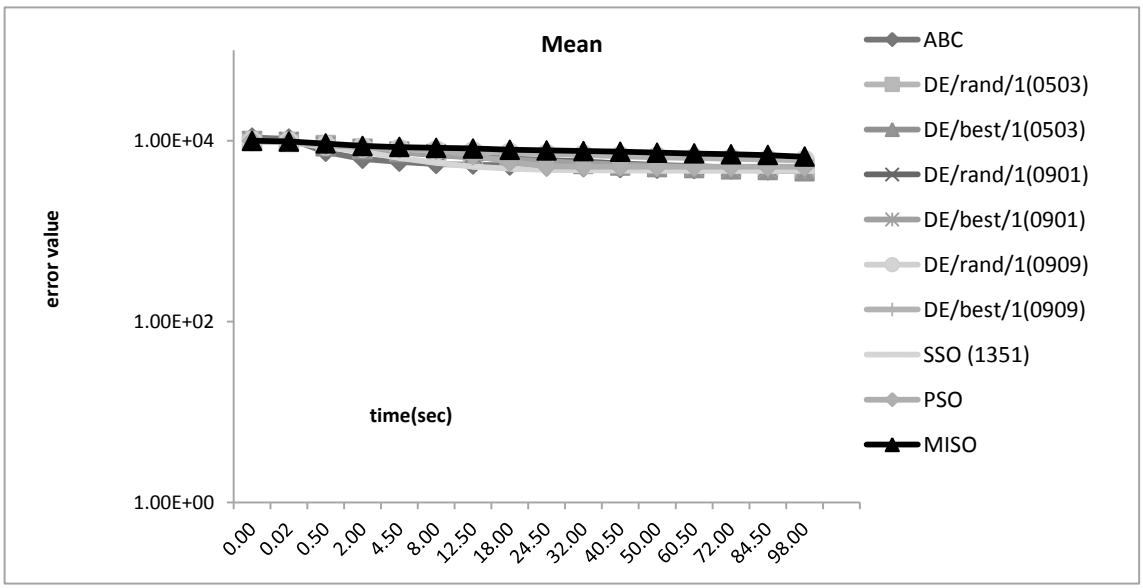
BM23


Max
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

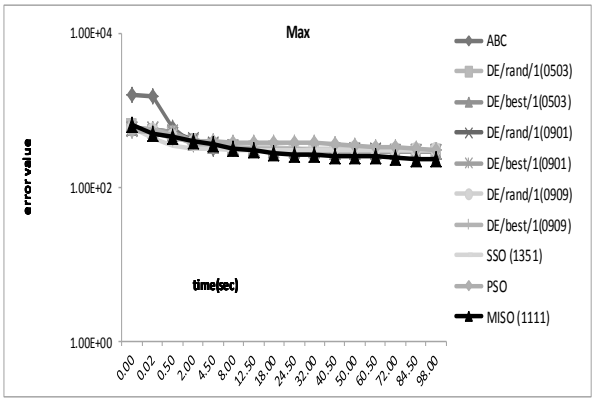

Median
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

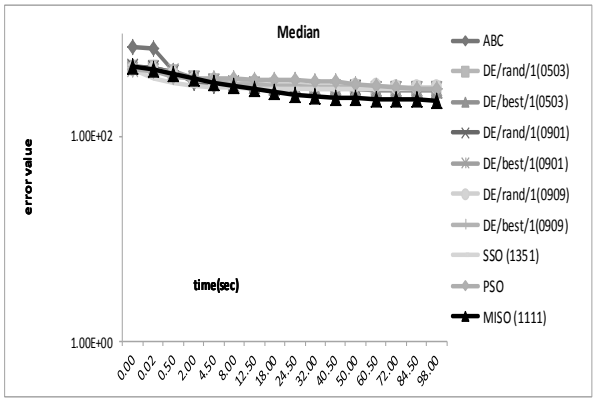

Min
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

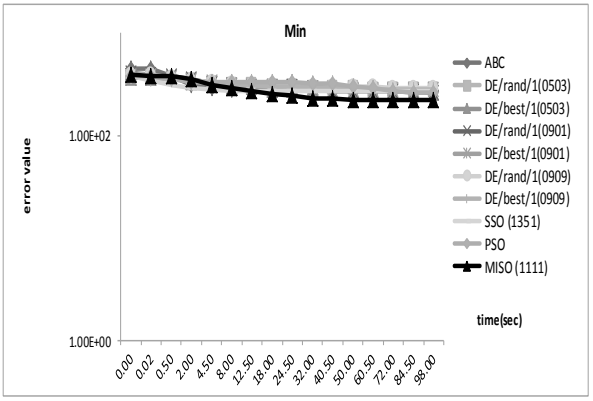

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

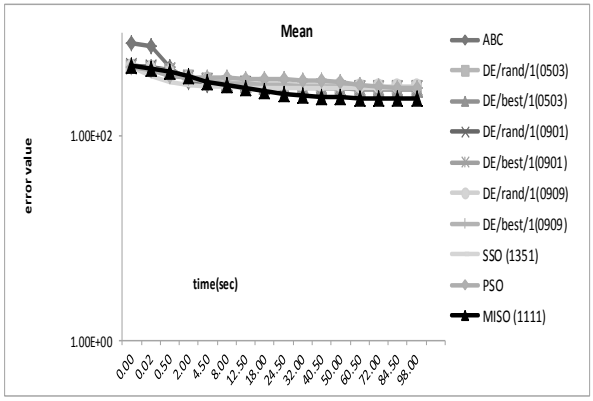

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

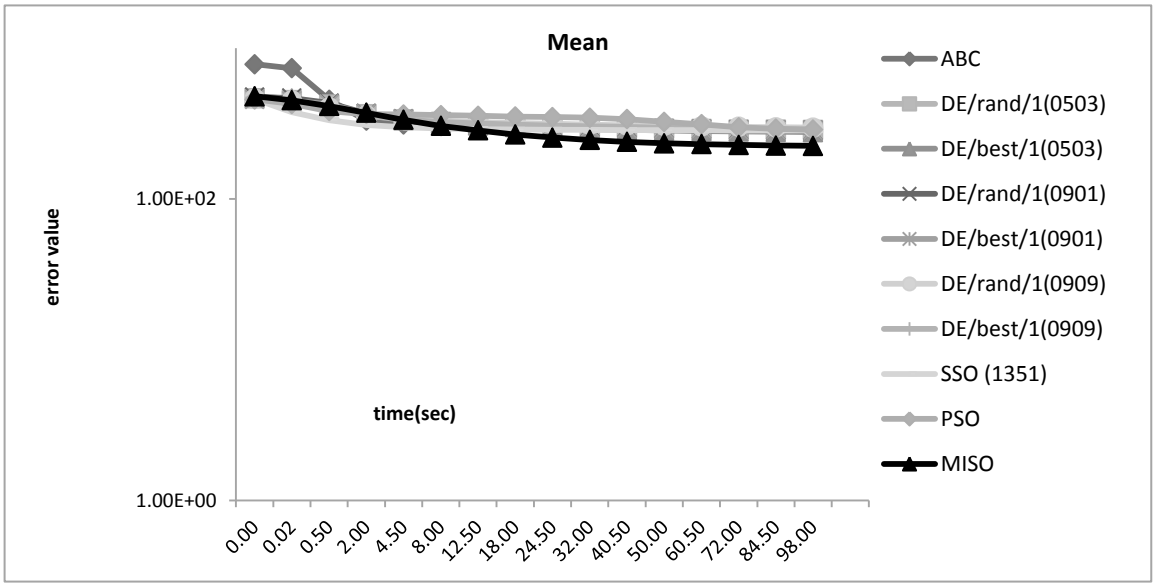
BM24


Max
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

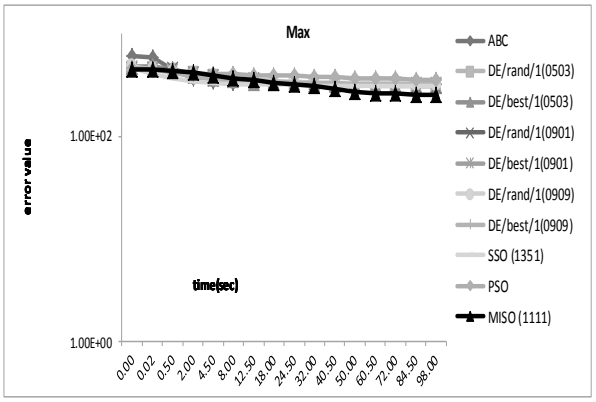

Median
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

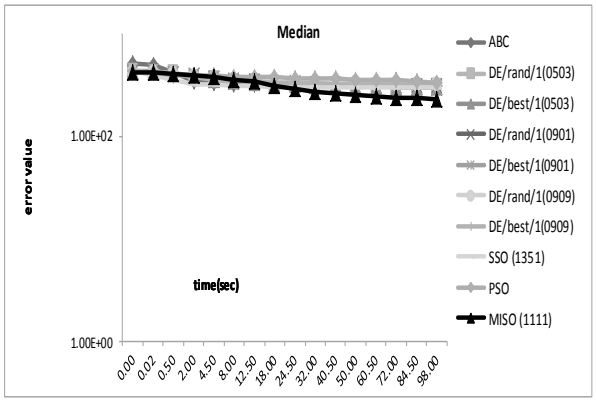

Min
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

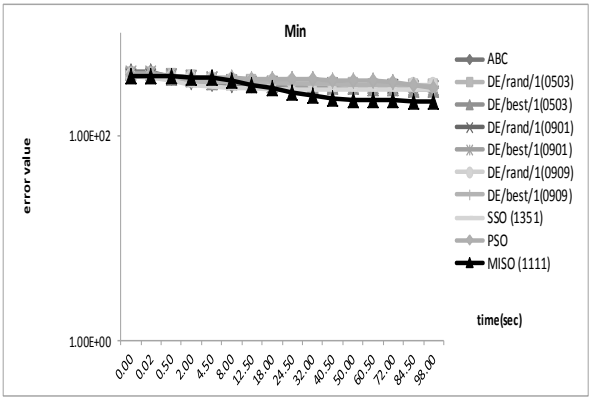

Mean
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

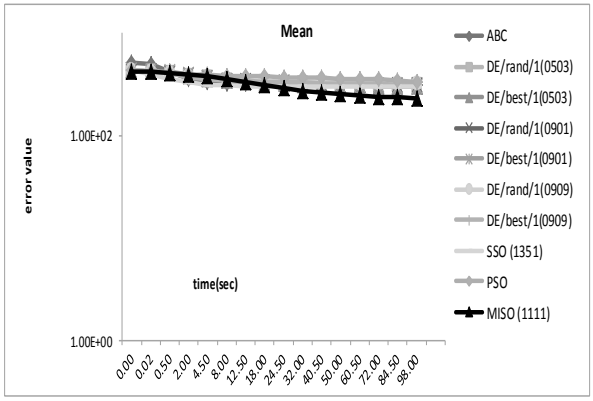

Mean
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

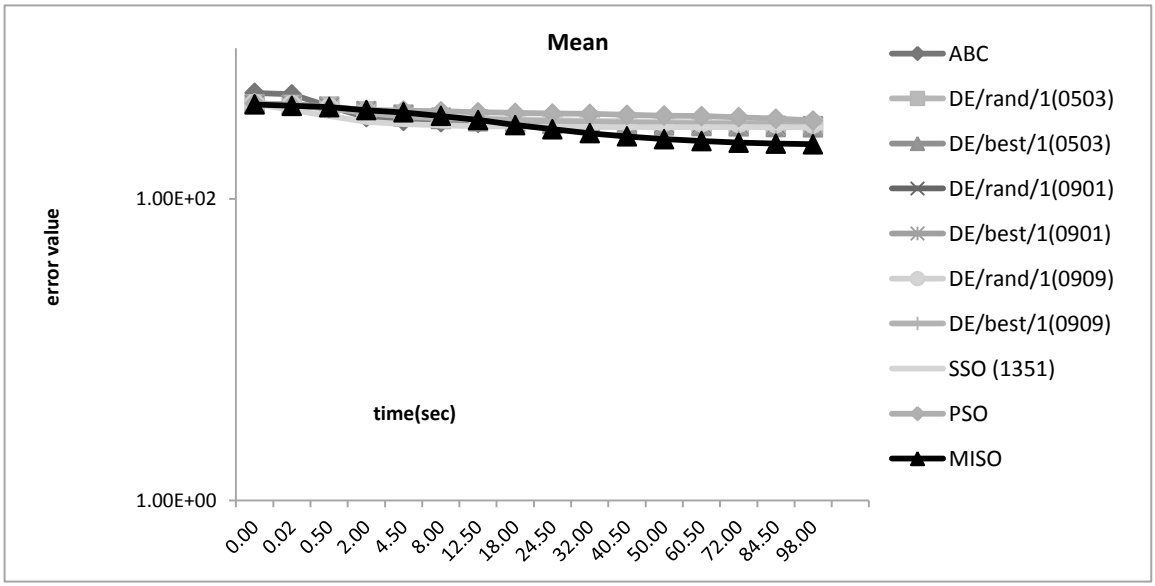
BM25


Max
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

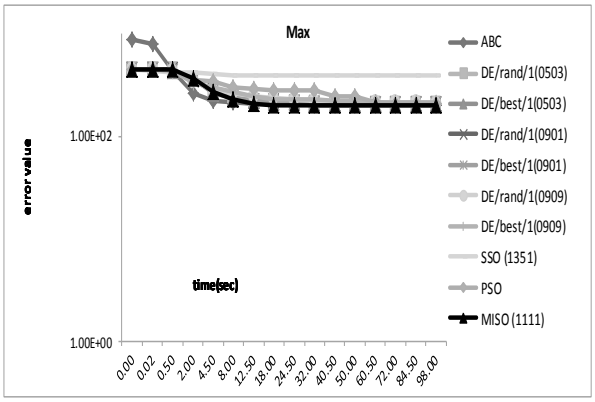

Median
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

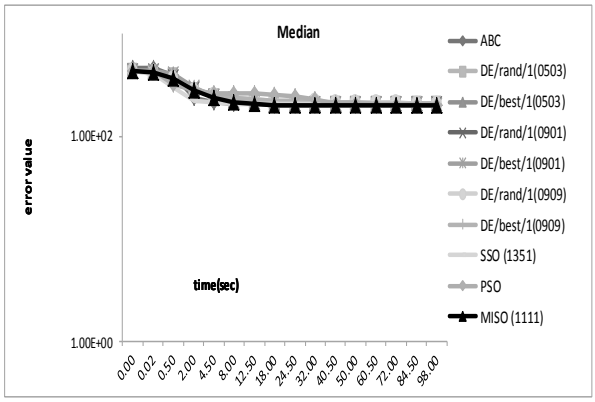

Min
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

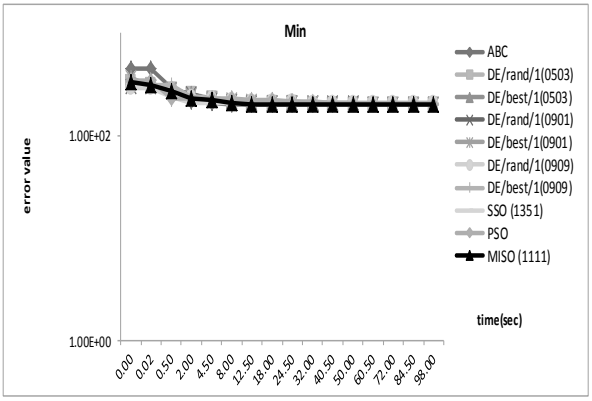

Mean
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

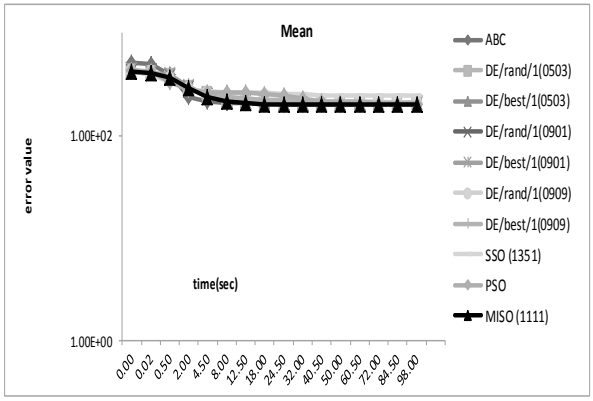

Mean
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

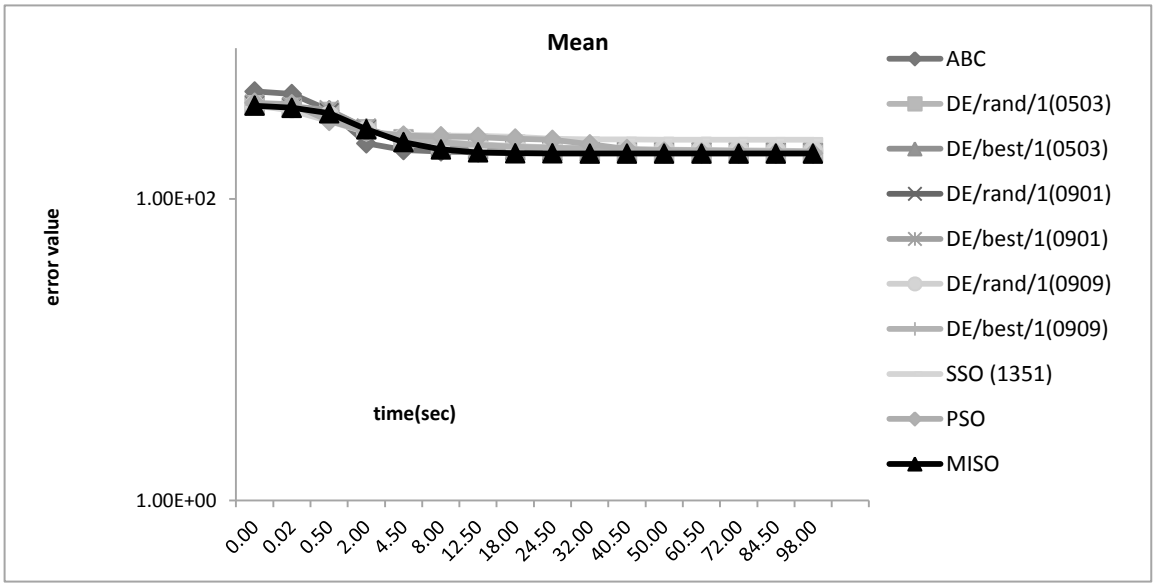
BM26


Max
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

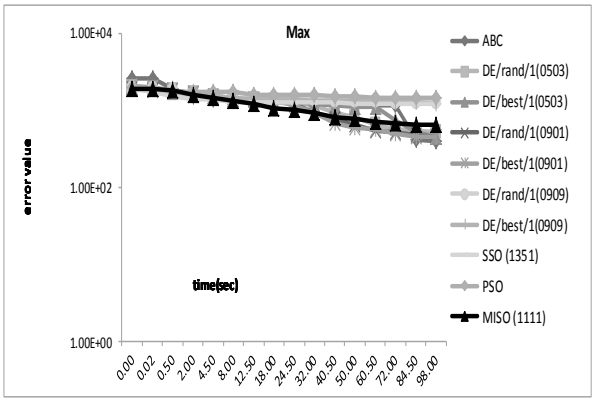

Median
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

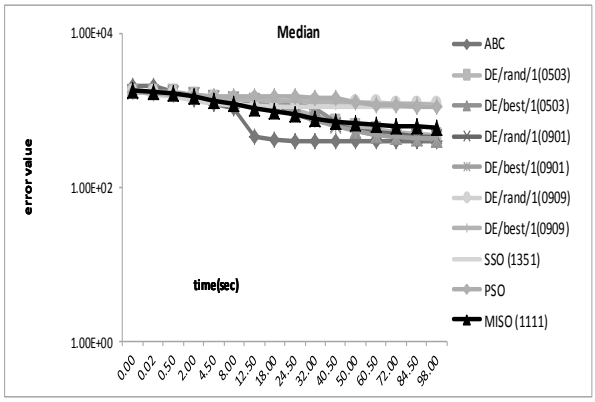

Min
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

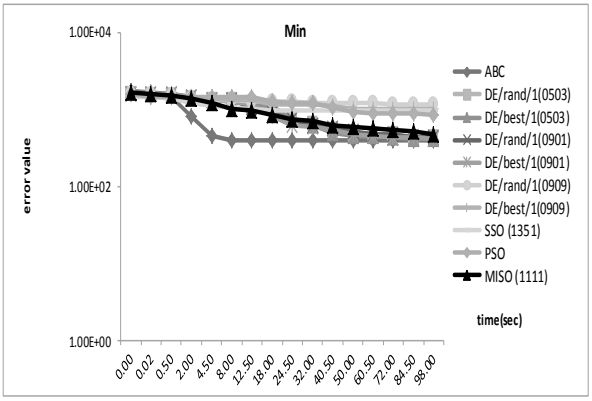

Mean
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

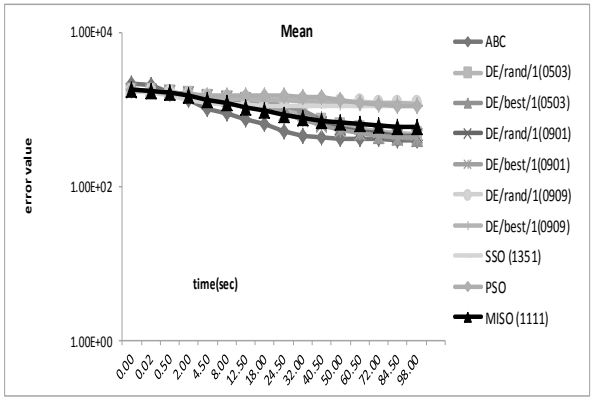

Mean
error value
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

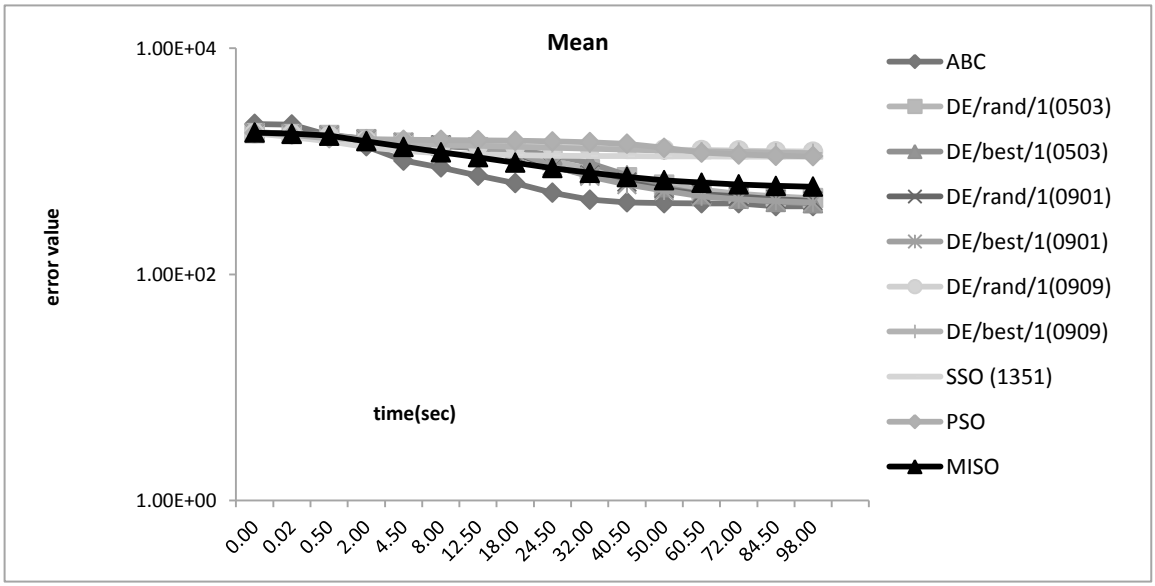
BM27


Max
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

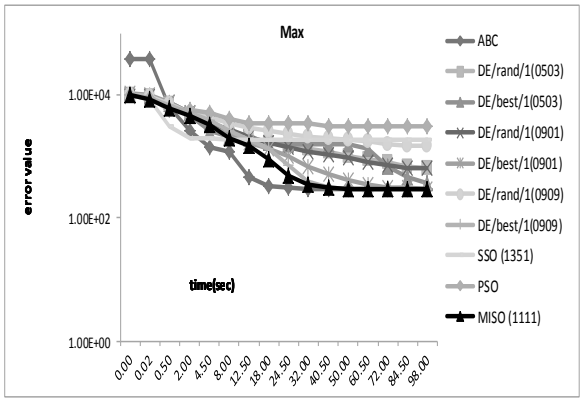

Median
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

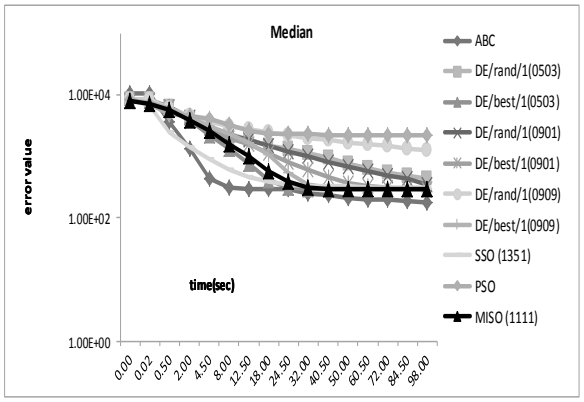

Min
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

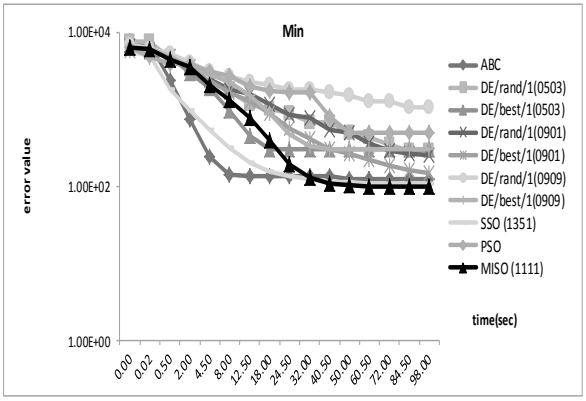

Mean
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

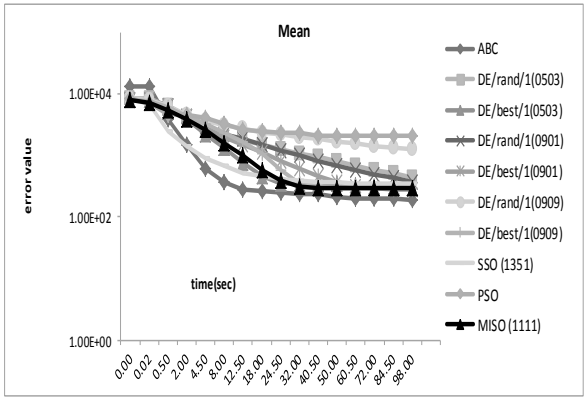

Mean
1.00E+04
1.00E+02
1.00E+00
error value
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

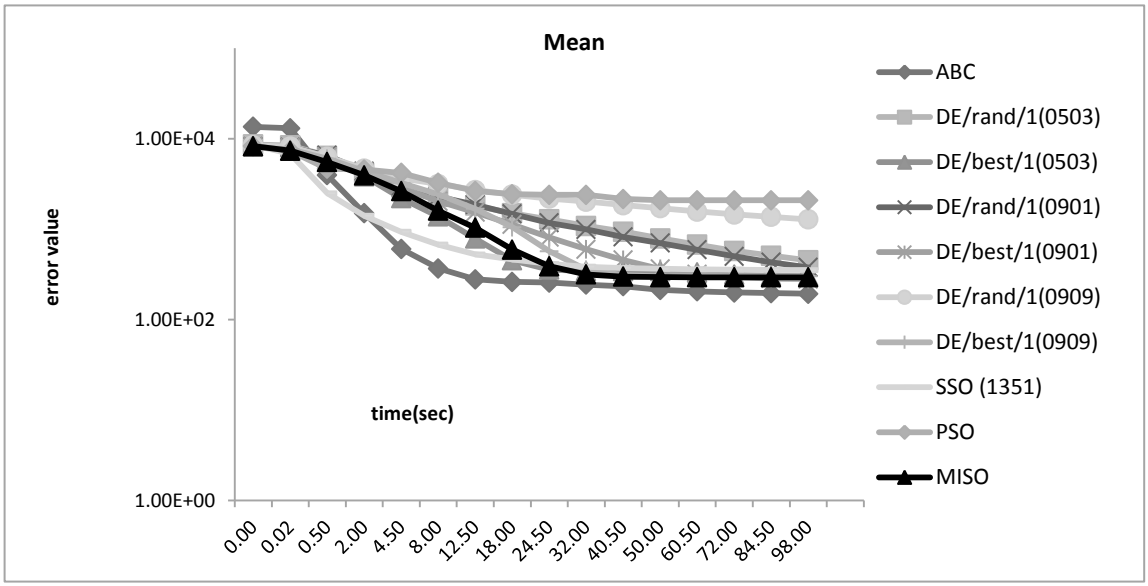
BM28


Max
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)


Median
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)


Min
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)

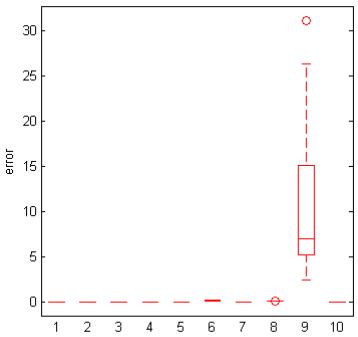

Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO (1111)


Mean
error value
1.00E+04
1.00E+02
1.00E+00
time(sec)
0.00 0.02 0.50 2.00 4.50 8.00 12.50 18.00 24.50 32.00 40.50 50.00 60.50 72.00 84.50 98.00
ABC
DE/rand/1(0503)
DE/best/1(0503)
DE/rand/1(0901)
DE/best/1(0901)
DE/rand/1(0909)
DE/best/1(0909)
SSO (1351)
PSO
MISO

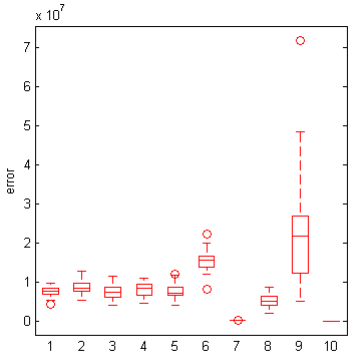
Boxplot

Label	Algorithm
1	ABC
2	DE/Rand/1(0503)
3	DE/Best/1(0503)
4	DE/Rand/1(0901)
5	DE/Best/1(0901)
6	DE/Rand/1(0909)
7	DE/Best/1(0909)
8	SSO(1351)
9	PSO
10	MISO(1111)

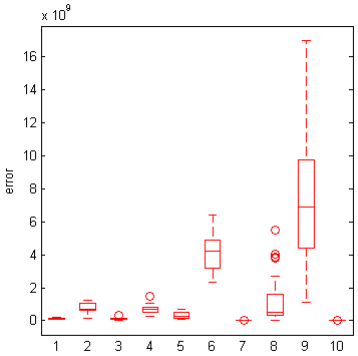
bm1



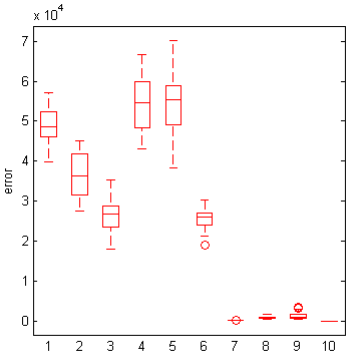
bm2



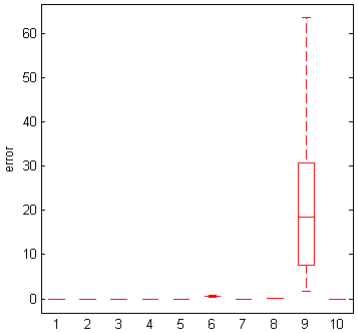
bm3



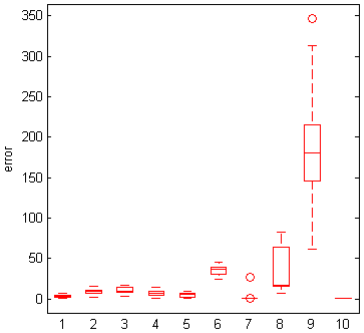
bm4



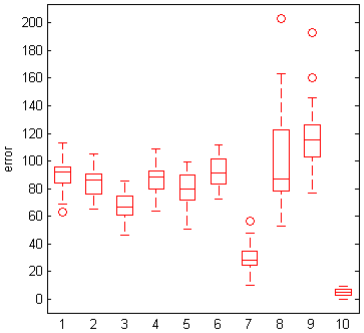
bm5



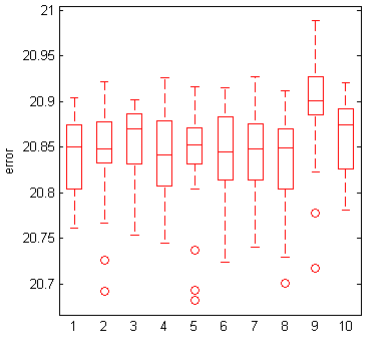
bm6



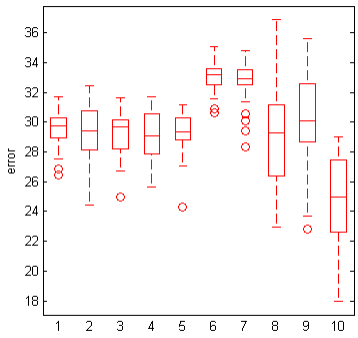
bm7



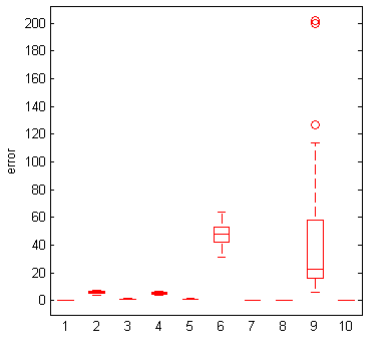
bm8



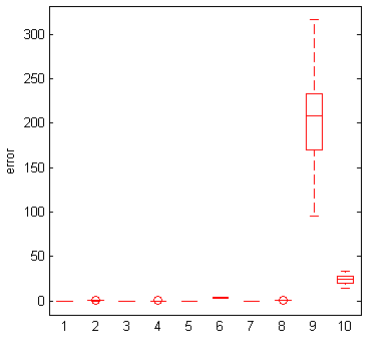
bm9



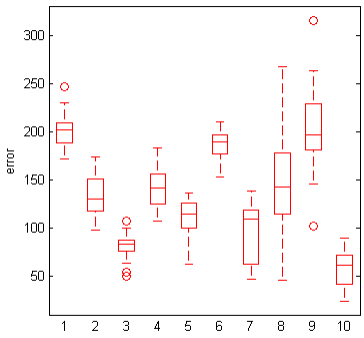
bm10



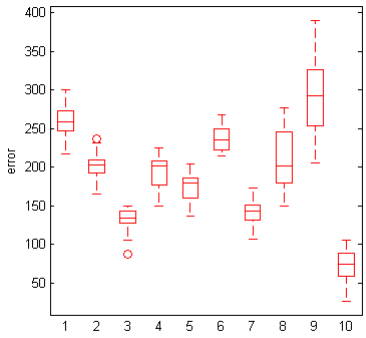
bm11



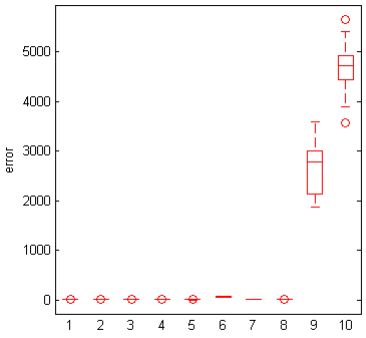
bm12



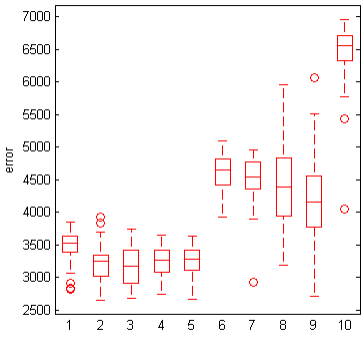
bm13



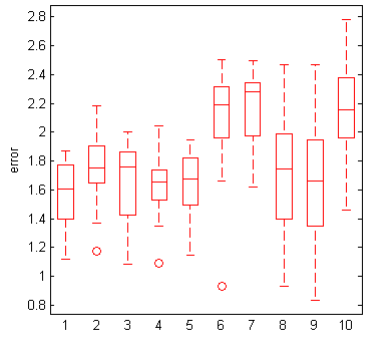
bm14



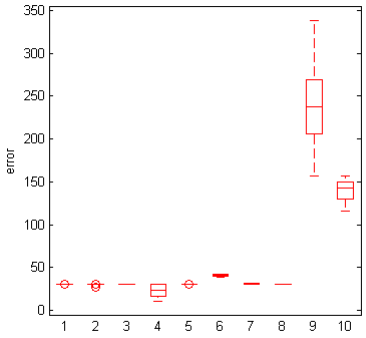
bm15



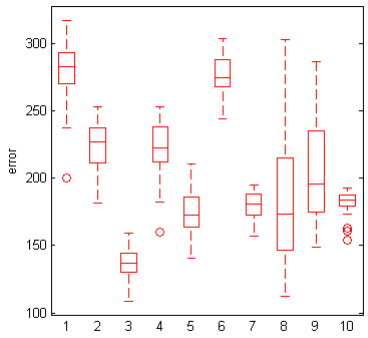
bm16



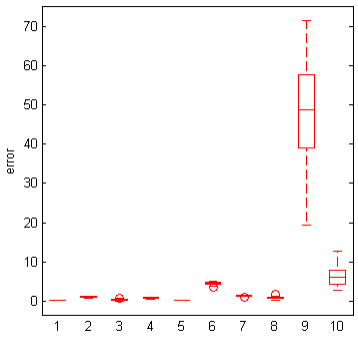
bm17



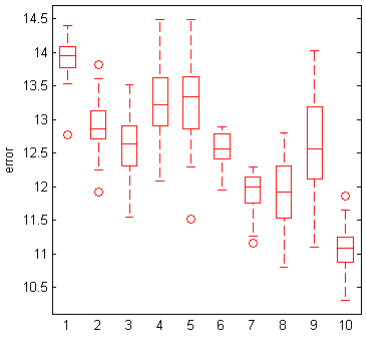
bm18



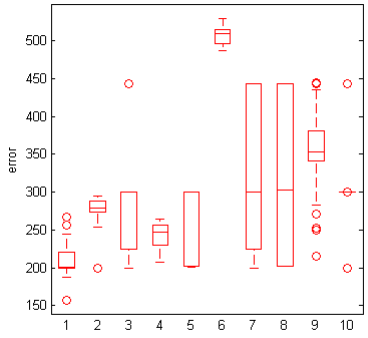
bm19



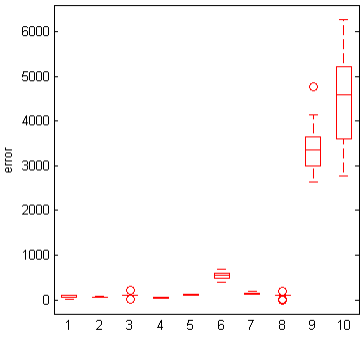
bm20



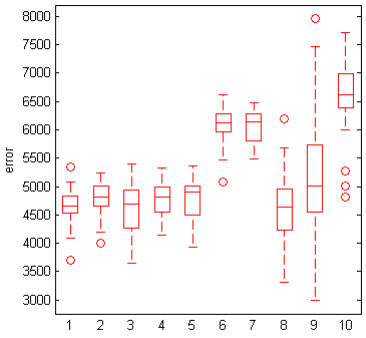
bm21



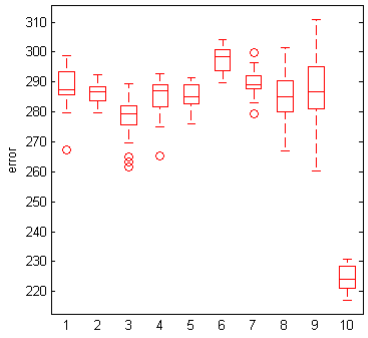
bm22



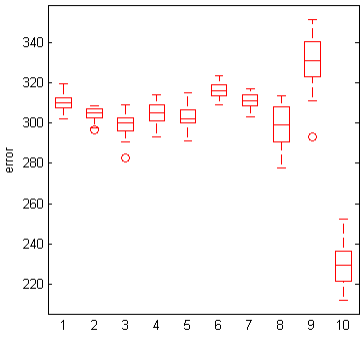
bm23



bm24



bm25



bm26	bm27	bm28