Experimental analysis on the operation of Particle Swarm Optimization

Naeemeh Yadollahpour

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF ARTS

GRADUATE PROGRAM IN
INFORMATION SYSTEMS AND TECHNOLOGY
YORK UNIVERSITY
TORONTO, ONTARIO
APRIL 2021

Abstract

In Particle Swarm Optimization, it has been observed that swarms often “stall” as opposed to “converge”. A stall occurs when all of the forward progress that could occur is instead rejected as Failed Exploration. Since the swarm’s particles are in good regions of the search space with the potential to make more progress, the introduction of perturbations to the *pbest* positions can lead to significant improvements in the performance of standard Particle Swarm Optimization. The *pbest* perturbation has been supported by a line search technique that can identify unimodal, globally convex, and non-globally convex search spaces, as well as the approximate size of attraction basin.

A deeper analysis of the stall condition reveals that it involves clusters of particles that are performing exploitation, and these clusters are separated by individual particles that are performing exploration. This stall pattern can be identified by a newly developed method that is efficient, accurate, real-time, and search space independent. A more targeted (heterogenous) modification for stall is presented for globally convex search spaces.

Contents

List of Tables

List of Figures

1 Introduction and motivation

With the rise of big data, Large Scale Global Optimization (LSGO) is an important part of many real-world applications such as the Internet of Things, bioinformatics, and logistics. However, more data will often increase the dimensionality of the problem domain, and the performance of optimization algorithms can deteriorate under these circumstances -- an effect known as the curse of dimensionality. Specifically, when dimensions increase, the volume of the search space grows exponentially [1]. Regarding limits in resources, computational effort can not keep up easily with the demands of LSGO. Hence, it must be noted that we continue to need better and more efficient optimization algorithms for a broad range of problems.

Exact algorithms require (in the worst case) exponential time to solve an NP-hard problem. For this class of problems, the use of approximate search techniques is justified. These techniques are categorized into two main groups: specific heuristics, and metaheuristics. As the name would suggest, specific heuristics are subject dependent and designed to solve certain problems. In contrast, metaheuristics apply to a large set of hard optimization problems [2].

Metaheuristics are used in both combinatorial and continuous domains. Popular metaheuristics for combinatorial problems include scatter search, simulated annealing, genetic algorithm (GA), and Tabu search [3]. Although some of them can be effectively adapted for continuous optimization, this adaptation process is more involved for some metaheuristics than for others. Particle swarm optimization [4] and an evolutionary approach called differential evolution [5] are designed primarily for continuous domains.

If a technique has a memory-less search, it behaves more like a trajectory-based method. Simple search methods like Hooks and Jeeves [6] and Nelder-Mead [7] are operator-based techniques, and they can follow gradients in non-differentiable search spaces. These types of methods and other single-point search

techniques are all trajectory-based methods, and there is no concept of targeting different parts of a search space in any of them.

Alternatively, a stored solution, which we will call a reference solution, is an essential component in a large class of metaheuristics. Reference solutions have different names in different techniques such as parent solutions in GA, personal best solutions (*pbest*) in PSO, and target solutions in differential evolution. However, in all of the mentioned methods, a reference solution is a solution against which the new search point is compared [8]. Reference solutions represent a form of accumulated knowledge about the search process. They help the metaheuristic direct future search solutions towards areas that are perceived to be the most promising regions of the search space. Progress to the search process thus depends on updating these reference solutions; otherwise, the same areas of the search space will be targeted repeatedly.

In the broad concept of the reference solutions, we particularly focus on the PSO technique's reference solution and its characteristics. Its reference solutions are the personal best (*pbest*) positions of each particle. New positions are based on a movement (influenced by reference solutions) from their current positions. The effect of attraction and momentum leads the particles in PSO to have oscillatory search trajectories around their reference solutions [9]. The movement/updating of reference solutions is thus critical to the ability of PSO to guide its particles towards better regions of the search space.

We define several key concepts for continuous domains based on dividing a multi-modal search space into attraction basins which each have a single local optimum. Each point in an attraction basin has a monotonic path of increasing (for maxima) or decreasing (for minima) fitness to its local optima. A new search point is then defined to be performing "exploration" if it is in a different attraction basin than its reference solution(s), and it is defined to be performing "exploitation" if it is in the same attraction basin as (one of) its reference solution(s) [10].

Most population-based metaheuristics have convergent search trajectories which lead all of the population members towards a single (ideally global) optimum. Generally, it is assumed that all PSO particles are converged when reference solutions have no progress in finding better solutions. For a true convergence condition in PSO, all *pbest* positions are located in the same attraction basin. Leading the particles' search trajectories to no longer generate any (exploratory) search points in different attraction basins. However, as it has been shown in [10], this is not always the case for PSO.

As an alternative to convergence, it has been shown that PSO can enter a pattern of stall. In stall, the *pbest* positions still represent many different attraction basins, and the particles that have attractors in different attraction basins can still explore new attraction basins as part of their search trajectories. However, the selective process of reference solutions is based on search solution fitness, and it does not take into account attraction basin fitness, which leads to many unfortunate rejections of these new (and beneficial) search points. Therefore, no progress can be seen during stall in terms of finding and keeping a search solution from a more promising area.

Contemporary research has "only presented experimental results and did not provide adequate discussion (neither from theoretical perspective, nor general discussions) on merits of the proposed approach" [11]. In this thesis, we have designed several experiments to discover the conditions under which stall occurs, and we then use key properties of these conditions to design tailored remedies. The specific information we collect includes the nature of reference solutions, how they are updated, and how they affect the individual and group behaviour of particle motions. We then show the effects of our remedies on the previously measured features in addition to the standard performance improvements on benchmark functions.

Our experiments demonstrate the occurrence of stall in some search spaces for PSO. We further show that stall can involve having multiple clusters of particles in different attraction basins. Building upon

these insights, extensive analysis of particle location and velocity data have led to different methods to identify stall, different classification for stall patterns, and insights into how different search spaces lead to these different stall patterns for PSO.

We will start Section 1 with reviews on popular population-based and single-point metaheuristics and important information about minimization problems. We will also review some modifications of population-based metaheuristic, different definitions of exploration in metaheuristics, and a precise definition for exploration with a specific example for this definition on the Rastrigin function. In section 3, various experimental analyses on the operation of PSO will be carried out leading to the formal introduction of the concept of "stall" (which is specifically contrasted against "convergence"). A refined line search technique will also be used to provide detailed information about the structure of different search spaces and the stall patterns that can occur in them. The details of implementation for each remedy and experimental results for the new modified versions of PSO on a set of benchmark problems will be presented in Section 4. Finally, in Section 5 my contributions in the process of upgrading the PSO performance will be specified and in Section 6 more suggestions for improving the PSO performance will be discussed.

2 Literature Review

A metaheuristic algorithm is applicable for hard optimization problems when there is no specific algorithm to solve them. The prefix “meta” means “higher level”, which implies that metaheuristics algorithms do not need to adapt to any problem deeply. Some standard features of metaheuristics include 1- the algorithms incorporate stochastic components (including random variables), 2- there is no gradient or Hessian matrix for the objective functions, and 3- there are tunable parameters (e.g., population size) which can greatly affect the performance of a technique on a specific (instance of a) problem.

A formal classification of metaheuristic can begin with their population size, which classifies them into population-based and single-point techniques. It can be said that single-point metaheuristics have a more exploitative structure while population-based techniques have a more explorative structure. In each metaheuristic, different operators and parameters, and combinations of the two, have been considered for addressing the balance between exploration and exploitation.

2.1 Single-point metaheuristic

Single-point metaheuristics, which are also known as trajectory methods, include simulated annealing, tabu search, GRASP, variable neighborhood, guided local search, iterated local search, and modifications of them [12].

Simulated annealing: It is applicable in both continuous and discrete search space but mainly focuses on non-continuous cost functions [13]. The algorithm first initializes a solution (S) and temperature (T); for which there are different initialization methods including random or by using a heuristic. Then in each iteration, a new solution (S′) is selected in the (S) neighborhood. Based on a comparison between F(S) and F(S′), two outcomes are possible. If the latter one is fitter, the new solution is accepted. Otherwise,

the probability operator decides whether the worse solution will get accepted or not [12]. Usually, the probability component in simulated annealing is computed by Boltzmann distribution as follows:

Equation 1:

$$P(S' \mid T_k, S) = e^{-\frac{f(s') - f(s)}{T_k}}$$

Simulated Annealing has a proof for convergence to the global optimum, but the encompassed conditions are too slow and thus not feasible for practical purposes [14]. The cooling schedule in SA decreases the temperature factor over time. It means that explorative searches decline over time since, as it can be inferred from the Equation 1, there is an implicit direct relationship between the temperature and the probability of accepting worse solutions. As a result, with a decrease in temperature, the explorative search state of SA decreases, and exploitative behaviors increase.

Tabu Search: this method mimics human memory, which means it can learn from its past experiences or searches. This technique has a list of visited locations (tabu list) in memory, which helps to prevent seeing these solutions in future iterations. The larger the tabu list is, the more exploratory the search technique can be because it puts greater pressure on leaving the most recent search areas. On the contrary, the smaller tabu list allows the search process to stay within smaller regions meaning the method can be more exploitative [12].

2.2 Population-based metaheuristics

In population-based methods a number of solutions (i.e., a population) is used. Evolutionary Computation (EC) and Swarm Intelligence (SI) are two notable groups of algorithms within the class of population-based methods [12]. Overall, all EC algorithms inspired by Darwinian evolution encompass

the three steps of selection, recombination, and mutation to produce new solutions for various optimization problems. In SI, techniques mimic social interaction rather than individual adaptations.

2.2.1 Evolution computation (EC)

Evolutionary computation: Evolutionary Computation (EC) (also called Evolutionary Algorithms (EAs)), is a general class that includes genetic algorithms, evolution strategies [15], evolutionary programming [16], and genetic programming [17]. In every generation, the population of individuals undergoes a natural selection procedure. Two or more selected individuals (the so-called parents) are recombined and create new individuals (the children or offspring), and these offspring can be further mutated to promote diversity. Based on solution fitness and a proper selection strategy, the next generation of parent solutions is determined. After a limited number of generations or some other stopping criterion, the process will finish and return the fittest individual(s) as the result.

Exploration is the process of visiting entirely new regions of a search space, while exploitation is the process of staying within the neighborhood of previously visited points. Exploration and exploitation lack clear and formal definitions, but it is generally acknowledged that the balance between them is important. The importance of a good balance between exploration and exploitation comes more into attention when metaheuristic techniques are dealing with multi-modal search spaces. In each EA, different operators or combinations of them are considered for addressing this balance. One common belief is that EAs should start with intense exploration and then gradually change it into a full exploitation state [18].

Evolution strategy: for the first time, the population concept was introduced in this method; the idea was inspired by nature and projecting the extinctions of individuals with deficits. From one individual, which is coded in real numbers, several individuals are created by adding gaussian noise to the original

solution (i.e., asexual reproduction). The fittest solution(s) survive selection to be the parent(s) for the next generation.

Genetic Algorithm (GA): Many variations of GA have been developed and applied to different optimization problems. There are essential genetic operators in a basic GA that can be implemented differently to address various optimization problems, including, 1- problem representation (chromosomes), 2- selection strategy, 3- type of crossover and 4- type of mutation. Based on [19], the balance between exploration and exploitation in GA can be achieved either by selection plan (e.g., roulette-wheel selection, tournament selection, ranking selection, etc.) or by the recombination (e.g., n-point and uniform) operators. Further, mutation introduces some randomness into the search to prevent the optimization process from getting trapped into local optima.

Differential Evolution (DE): It is a well-known metaheuristic technique for continuous search space, capable of doing local and global searches. In DE methods, a population of candidate solutions are randomly initialized. For each individual, some other individuals are randomly chosen within the population to calculate difference vectors. A *new* solution is then created by adding the weighted difference vector to a *base* solution with crossover applied thereafter. Finally, the *target* individual is compared against the *new* solution, and the fitter individual will be maintained for the next generation [12].

A premature loss of diversity is also problematic for DE because it generates a solution based on the differences between solutions in the solution space. As the population converges the magnitude of exploratory moves also decreases - such convergence is typically irreversible, which means the algorithm can never again explore widely. Indeed, DE can suffer from a cascading collapse in diversity -- when some solutions cluster in a small area, they produce short difference vectors that may ‘recruit’ other solutions to that same area [20].

2.2.2 Swarm Intelligence (SI)

Figure 1. Movement of bird flocks covers the search space [21]

Particle swarm optimization: fits into the family of algorithms based on the concept of swarm intelligence [22]. The primary purpose of the particle swarm concept is to simulate the graceful and unpredictable dance of bird flocks (see in Figure 1) with the aim of mimicking the patterns which govern the ability of birds to make simple individual decisions on their flight paths which can lead to complex group dynamics. From this initial goal, the concept became a simple and efficient optimization algorithm.

In PSO, individuals, referred to as particles, "fly" through a hyperdimensional search space. Changes to the position of particles within the search space are based on the social-psychological tendency of individuals to imitate the success of other individuals. The changes to a particle within the swarm are therefore influenced by the experience, or knowledge, of its neighbors. The search behavior of a particle is thus affected by that of other particles within the swarm (PSO is therefore a kind of symbiotic cooperative algorithm). The consequence of modeling this social behavior is that the search process is such that particles stochastically return toward previously successful regions in the search space. Individuals in a particle swarm follow a very simple behavior: to emulate the success of neighboring

individuals and their own successes. The collective behavior that emerges from this simple behavior is that of discovering optimal regions of a high dimensional search space [23]. However, selection is an important factor to adapt the swarm motion into an optimization technique, but selection also changes the (modelled) particle motions and as a result the overall swarm motion.

This global search technique includes a population of particles for which each particle maintains a location in the search space and a velocity. Each particle also has a memory, which is used for storing the personal based experience of each solution. Particles are connected with a topology. Two key structures are star (*gbest*) and ring (*lbest*) topologies. In the first structure, all particles get updated with respect to the best particle's experience, so the topology is a fully connected graph. In the second structure, called a ring topology, each particle is updated based on its two neighbors. Ring topology makes PSO more explorative than the *gbest* topology since it takes the population more time to converge. However, it should be noted here that reaching convergence is not always the case. In some search spaces, particles can experience the stall condition [12].

A larger value for inertia weight parameter in updating velocity causes larger jumps of particles in PSO, which, as a result, gives more opportunity for performing global optimization. On the other side, smaller values for inertia weight would result in local searches [24]. A swarm's diversity can be represented by how far particles are from each other in the search space. One measure for diversity is the variance of all particles from the center in each time step. When diversity is higher, particles are more scattered, which indicate a more explorative behavior for the swarm [25].

2.3 A benchmark set of problems for real-parameter optimization

Based on the CEC2013 [26] definition, all of these problems should be treated as black-box problems. The explicit equations of the problems are not allowed to be used. However, the dimensionality of the problems and the total number of available function evaluations can be considered as known values that can be used to design dynamic or adaptive approaches in an algorithm. For benchmarking algorithm results, all of the problems of Section 2.3.2 are treated without previous knowledge. In addition to the CEC2013 benchmark set, the Rastrigin function has been used in several experiments based on its well-known structure to give us more insight into the design of various algorithms.

2.3.1 Rastrigin

A common method to create a multi-modal function is to super-impose a sinusoid on top of a base function. A popular example is the Rastrigin function (Equation 2) which has a search range of [-5.12 , 5.12] in every dimension. The periodic nature of the sinusoid adds many (local) optima to the unimodal base function (i.e., x^2) which creates a globally convex multi-modal search space. As first described in [27], the Rastrigin function has a regular fitness landscape in which every point with integer values in all dimensions is a local optimum, and all other points belong to the attraction basin of the local optimum that is determined by rounding each solution term to its nearest integer value. These features make it possible to quickly identify the attraction basin for a solution and to easily calculate the fitness of its local optimum.

Equation 2:

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

2.3.2 Benchmark set

The CEC2013 problems, summarised in Table 1, are used to evaluate the performance of our modifications to PSO. This benchmark includes 28 minimization problems divided into three groups: unimodal functions (F1–F5), basic multi-modal functions (F6–F20), and composition functions (F21–F28). For all of these minimization problems, the search range is set to [-100,100] in every dimension. A maximum of 10,000 * d function evaluations is permitted, where d is the dimension of the problem. All problems have the global optimum within the given bounds and there is no need to perform search outside of the given bounds for these problems.

The basic unimodal function in CEC 2013 is the sphere function. In its simplest form, its single (global) optimum will be at the origin, and it will also be separable so that it can be solved one dimension at a time. In general, higher dimensional functions are created by replicating the base function in each dimension. To avoid favouring methods which exploit separability (e.g., Artificial Bee Colony [28]) or center-bias [29]), benchmark functions are often shifted and rotated. However, these modifications generally do not affect the unimodal, globally convex, or non-globally convex nature of the search spaces created by their base functions.

In benchmark function sets, it is easy to determine if a search space is unimodal, globally convex, or non-globally convex through analysis of the generating function. This determination is more difficult from an analysis of the search space alone. In particular, it is impossible to determine a global feature (e.g., globally vs. non-globally convex) from a local analysis (e.g., [30] [31]).

Category	F	Name
Unimodal Functions	1	Sphere Function
	2	Rotated High Conditioned Elliptic Function
	3	Rotated Bent Cigar Function
	4	Rotated Discus Function
	5	Different Powers Function
Basic Multimodal Functions	6	Rotated Rosenbrock's Function
	7	Rotated Schaffers F7 Function
	8	Rotated Ackley's Function
	9	Rotated Weierstrass Function
	10	Rotated Griewank's Function
	11	Rastrigin's Function
	12	Rotated Rastrigin's Function
	13	Non-Continuous Rotated Rastrigin's Function
	14	Schwefel's Function
	15	Rotated Schwefel's Function
	16	Rotated Katsuura Function
	17	Lunacek bi-Rastrigin Function
	18	Rotated Lunacek bi-Rastrigin Function
	19	Expanded Griewank's plus Rosenbrock's Function
	20	Expanded Scaffer's F6 Function
Composition Functions	21	Composition Function 1 (n=5,Rotated)
	22	Composition Function 2 (n=3,Unrotated)
	23	Composition Function 3 (n=3,Rotated)
	24	Composition Function 4 (n=3,Rotated)
	25	Composition Function 5 (n=3,Rotated)
	26	Composition Function 6 (n=5,Rotated)
	27	Composition Function 7 (n=5,Rotated)
	28	Composition Function 8 (n=5,Rotated)

Table 1. CEC2013 Benchmarks

2.4 Precise definition of exploration and exploitation

The lack of explicit definitions for exploration (and exploitation) hinders the ability to measure them correctly. A broad survey of over 100 papers in [18] indicted an unexpected conclusion that "The fact that until now exploration and exploitation have only been implicitly defined in EAs comes as a big surprise."

Precise definition for exploration (and exploitation) is possible with accurate definition of attraction basins (in continuous domain search spaces). An attraction basin is introduced as part of a search space in which all points can reach the local optimum by following a monotonic path. Based on above definition, it can be said that multimodal search spaces contain a countable number of these attraction basins, while a unimodal search space includes exactly one attraction basin. Moreover, the fitness of an attraction basin is defined to be the same as the fitness of its local optimum.

Hereby, a search point (e.g., the current position of a particle) is performing "exploration" if it is in a different attraction basin than its reference solution(s) (e.g., a particle's *pbest* position), and it is defined to be performing "exploitation" if it is in the same attraction basin as (one of) its reference solution(s). In exploratory actions, comparison of current solution and *pbest* would result in four different outcomes, which shows the importance of selection in performance of PSO. The concise definitions of these categories accompanied by Figure 2 are as bellow:

"**Successful Exploration**: A fitter attraction basin represented by a fitter exploratory search solution is accepted to replace the less fit reference solution from a less fit attraction basin.

Successful Rejection: A less fit attraction basin represented by a less fit exploratory search solution is rejected to keep the fitter reference solution from a fitter attraction basin.

Deceptive Exploration: A less fit attraction basin represented by a fitter exploratory search solution is accepted to replace the less fit reference solution from a fitter attraction basin.

Failed Exploration: A fitter attraction basin represented by a less fit exploratory search solution is rejected to keep the fitter reference solution from a less fit attraction basin."

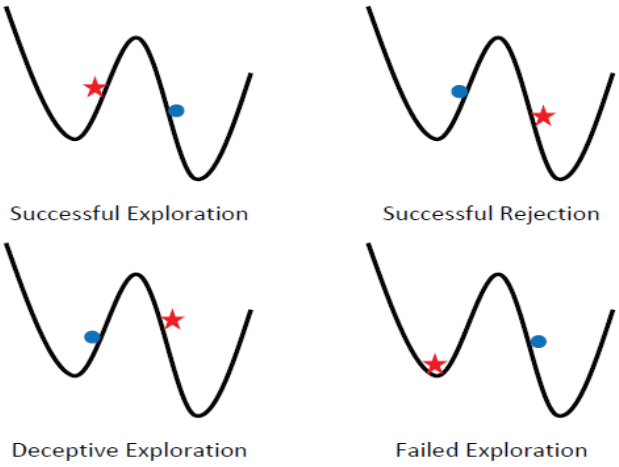


Figure 2. Four categories of exploration after selection. In each sub-figure, the red star represents the reference solution, and the blue dots represents the exploratory search solution [10]

In most other research, the importance of selection on exploration has not been considered. However, rejected exploratory search solutions often have no effect on the overall search process, so selection plays a pivotal role in moderating the effects of exploratory activities. Another key problem in other research is related to the misuse of the term exploration when "potential for exploration" could be more accurate. The lack of a clear definition of exploration and not considering the effect of selection can obscure the important effects of Failed Exploration.

3 Experimental analyses and observation on the operation of PSO

The demonstration of Failed Exploration in [10] leads to our differentiation of "convergence" and "stall" in PSO. At convergence, all *pbest* positions are located in the same attraction basin, and the search trajectories of the particles no longer generate any (exploratory) search points in other attraction basins. In stall, the *pbest* positions still represent many different attraction basins and the particles which have attractors in different attraction basins can still explore new attraction basins as part of their search trajectories. However, all of these search solutions are rejected, including many under the category of Failed Exploration.

In order to accurately label every search solution as exploration (with four possible categories) or exploitation, we need to determine the attraction basin associated with every solution in the search space. For general search spaces (e.g., real-world problems), this can often be infeasible. Initial experiments were thus limited to artificial search spaces such as the Rastrigin function shown in Equation 2. Based on Rastrigin's structure (Section 2.3.1), it is possible to clearly identify the category of exploration or exploitation for all of the particle motions during every iteration.

Our experiments use a version of standard particle swarm optimization [32] with a ring topology. The key parameters specified from this standardization are $\chi = 0.72984$, and $C_1 = C_2 = 2.05$ for the velocity updates given in Equation 3. Additional implementation details are the use of p = 50 particles [32], zero initial velocities [33], and "Reflect-Z" for particles that exceed the boundaries of the search space (i.e., reflecting the position back into the search space and setting the velocity to zero) [34].

Equation 3:

$$\boldsymbol{v_{i+1,d}} = \boldsymbol{\chi}\{\boldsymbol{v_{i,d}} + \boldsymbol{c_1}\,\boldsymbol{\epsilon_1}\,(Pbest_{i,d} - \boldsymbol{x_{i,d}}) + \boldsymbol{c_2}\,\boldsymbol{\epsilon_2}\,(lbest_{i,d} - \boldsymbol{x_{i,d}})\}$$

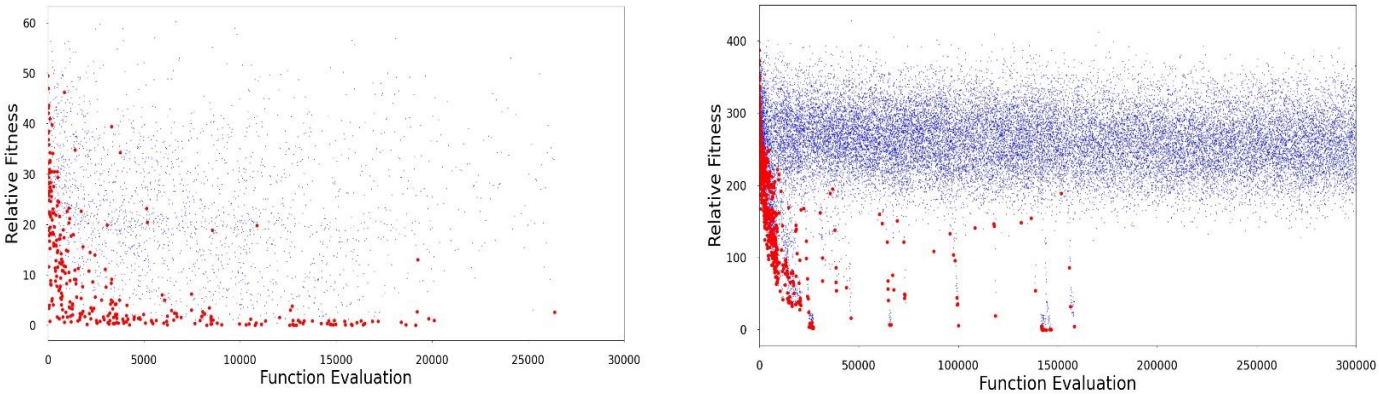
Many modified versions of PSO exist [11]. Some of these versions modify weights (e.g. [9], [35]), communication topologies (e.g. [36], [37]), velocity update rules (e.g. [38], [39]), or population size (e.g. [40], [41]). However, the broad survey in [11] indicates no significant research activity to address the effects of Failed Exploration. The results of this thesis should thus be equally relevant to these types of modified versions of PSO which have similar selection strategies to the standard version of PSO (i.e., updating *pbest* only when fitter current positions are found). More generally, any swarm intelligence method which can experience stall (as a result of its elitist selection) may be able to benefit from the methods and insights we will present.

3.1 Stall detection in Rastrigin in PSO

The common assumption in PSO is that if the fitness of the best overall particle has not improved for a large number of iterations, then the swarm has converged. However, from a linguistic perspective, the word “converge” in English implies that a converged swarm should have all of the particles reach the same location of the search space. The current experiment using the Rastrigin function clearly shows that this is not the typical case for the operation of PSO (in higher dimensions for multi-modal search spaces).

3.1.1 PSO convergence vs stall in different dimensions

This experiment applies standard PSO to the Rastrigin function in d = 3 and d = 30 dimensions. A total of 10,000d function evaluations is used.



(a) D = 3 (b) D = 30

Figure 3. Convergence vs. stall in PSO. In d = 3 dimensions (a), the swarm has converged after the last red circle representing Successful Exploration. In d = 30 dimensions (b), large amounts of (failed) exploration continue to occur in more promising regions of the search space as indicated by the blue dots. However, swarm progress has largely stalled as indicated by the lack of red circles which represent Successful Exploration [42].

The plots in Figure 3 demonstrate a key difference between "convergence" and "stall". The plots show the relative fitness of exploratory search positions from fitter attraction basins. Relative fitness refers to the difference between the fitness of the current position and the fitness of its attraction basin (which is also the nearest local optimum in Rastrigin). Exploratory search solutions are with respect to the *pbest* and *lbest* attraction basins as search solutions in these attraction basins would be labelled as performing exploitation. The small blue dots represent Failed Exploration (i.e., rejection), and the large red dots represent Successful Exploration.

In Figure 3(a) for d = 3 dimensions, PSO converges. The first (i.e., *gbest*) particle reaches the global optimum after about 15% of the allocated function evaluations, and all of the other particles reach that attraction basin by the end of the run. After the last large red dot for Successful Exploration on the far right, there are no more dots at all. As the run progresses, the density of red and blue dots decreases as fewer and fewer particles produce exploratory search solutions (in fitter attraction basins). The lack of any

dots at the end of the run is a sign of convergence – all of the trajectories of the particles stay within the attraction basins of their respective *pbest* positions.

In Figure 3(b) for d = 30 dimensions, PSO stalls. The first (i.e., *gbest*) particle often reaches the final fitness level after about 15% of the allocated function evaluations, but then the swarm stalls. There are still large amounts of exploration in fitter attraction basins that could lead to better overall solutions being found, but most of it is rejected in the form of Failed Exploration.

In both sub-plots, it is worth noting that the relative fitness of the red dots covers a large range during the first few iterations. An initial position with poor relative fitness can be improved upon by an exploratory search solution in a fitter attraction basin with similar relative fitness. However, extensive studies in [10], [27] show that *pbest* positions rapidly approach their local optima which has a relative fitness of zero. This improving relative fitness of the *pbest* positions forces the relative fitness of the exploratory search solution to similarly improve in order to achieve Successful Exploration.

A swarm that has converged (e.g., Figure 3 (a)) requires techniques to increase exploration since no additional exploratory search positions are being generated. However, a swarm that has stalled (e.g., Figure 3 (b)) is still producing many useful exploratory search positions. The problem is that (almost) all of these useful exploratory search positions are being rejected as Failed Exploration. Our strategy is to reduce the rates of Failed Exploration through the use of *pbest* perturbations. These perturbations are distinctly different from restarts which will be analyzed as part of the performance-based analysis in Section 4.1.

3.1.2 Stall detection by explicit category information in Rastrigin

This experiment applies standard PSO to the Rastrigin function in d = 30 dimensions. A total of 300,000 function evaluations is allowed which leads to the 6000 iterations shown on the y-axis. The x-axis

represents each particle's index in PSO, and it is noted that each particle's index is fixed during the whole process of standard PSO.

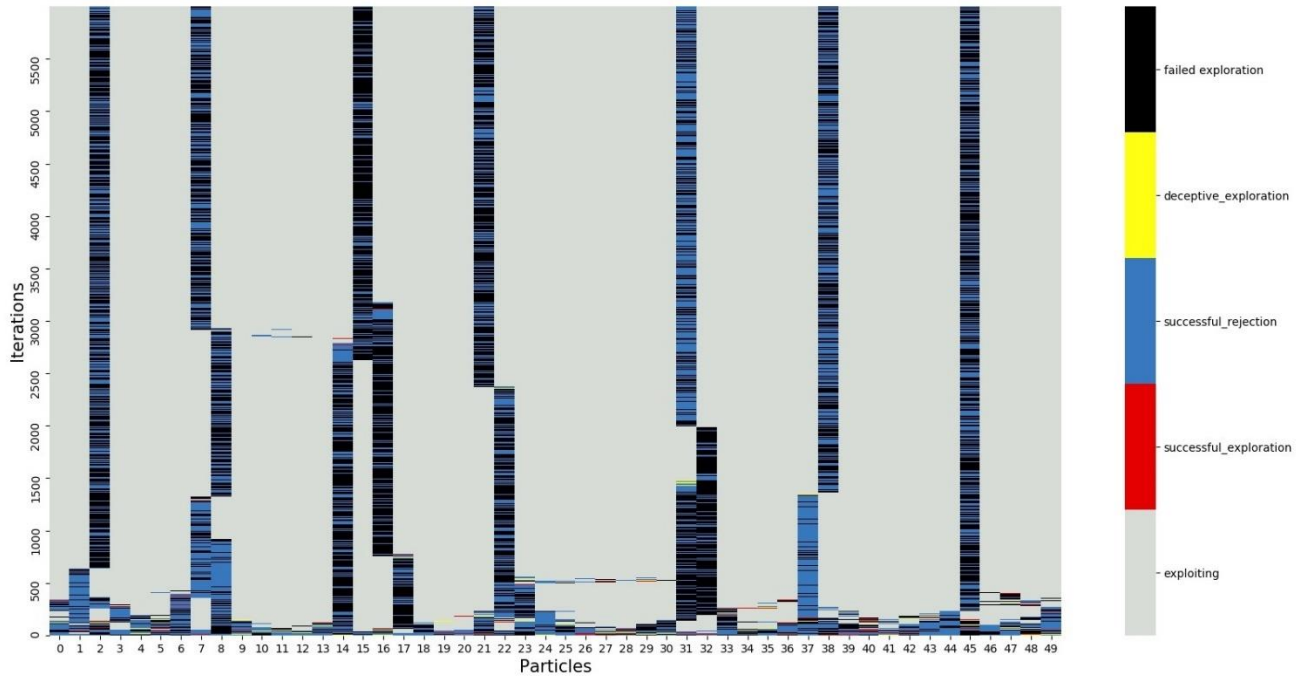


Figure 4. Particles' search categories for all generations in standard PSO in Rastrigin function, x-axis and y-axis represents particles indices and generation number, respectively.

As shown in Figure 4, after a few iterations, clusters of (exploitative) particles appear – particles performing exploitation are shown in gray color. Obviously, despite what was expected from PSO, there is not one converged cluster in the picture; instead, there are usually 3 to 8 clusters of particles (with 7 in the example shown in Figure 4) doing exploitations in different attraction basins. Furthermore, many particles are doing exploration until the end of iterations, but they are unable to lead the swarm to better (or even different) areas of the search space since they all have resulted in either Failed Exploration or Successful Rejection.

The colors red, yellow, blue, black, and gray represent Successful Exploration, Deceptive Exploration, Successful Rejection, Failed Exploration, and exploitation, respectively. Red and yellow colors occur mostly during the early stages of PSO. The initial high rates of Successful Exploration and Deceptive Exploration is related to having reference solutions that are not yet locally optimized. These types of reference solutions which are conducive to the acceptance of exploratory search solutions do not last long, and after an early phase of activity, rates of these two categories of exploration diminish very fast.

Blue and black colors are seen more frequently and even in an alternating pattern. For several particles, this pattern lasts until the end of the allowed iterations. Another interesting observation here is that there is a halt of blue and black colors in particle 1 and a sudden coincidental start for them in particle 2, and this phenomenon happened for several other adjacent particles as well. This structure indicates that there is always at least one “flying” particle between the clusters of particles, and this particle separates different clusters from each other. It means that if these particles' roles get changed during an iteration from explorative particles to exploitative particles (being part of a group of particles that do exploitative searches), their exploratory role will need to be transferred to a neighboring particle.

The proportion of blue and black colors is similar in the first and the second half of iterations, but the difference is that they are continually occurring in some specific particles in the second half, which in this case are the particles with indexes 2, 7, 15, 21, 31, 38, and 45. Generally, these particles are trapped between two clusters of particles, and this leads them to explore the regions of the search space that are between and around the two clusters. However, since the *pbest*s for these exploring particles are locally optimized, these particles' current locations have a vanishingly small chance to have a better fitness.

In Rastrigin, when the reference solutions for a set of adjacent/communicating particles are in the same attraction basin, a cluster of particles is created. Based on this definition of clustered particles, their

detection can be observed in each iteration. The rate of clustered particles and non-clustered particles are presented as percentages for a full run of PSO.

Particle type	exploitation	successful exploration	successful rejection	failed exploration	deceptive exploration
Clustered	80.04	0.02	0.04	0.05	0.01
Non-clustered	2.07	0.17	8.10	9.39	0.11
All	82.11	0.19	8.14	9.44	0.12

Table 2. Aggregated category information in one run on rastrigin for all particles and clustered particles

As shown in Table 2, the aggregated information shows that clustered particles have a very small chance of exploring and it becomes the primary responsibility for non-clustered particles to perform exploration. The reason is that particles in clusters have short attraction vectors (as all of their neighbors are very close), so they very rarely move large distances in the search space. Therefore, even though position and velocity updating in PSO have stochastic components, the chance for these particles to perform exploratory movements is minimal.

3.2 Line search technique for identifying attraction basins in search spaces

The previous analysis of stall shows that the failure of PSO to reach the global optimum is not about the lack of exploratory solutions, but rather that (almost) all of these useful exploratory search positions are being rejected as Failed Exploration. One strategy we have applied to reduce the rates of Failed Exploration is the use of *pbest* perturbations.

It is noted that knowledge about a search space's global structure (e.g., unimodal, globally convex, or non-globally convex) can be helpful. To increase the rates of Successful Exploration in globally convex

search spaces, where it can be expected that around half of the adjacent attraction basins are fitter, particles should search near the attraction basin of their *pbest*s. Hence, some degree of awareness about the size of attraction basins can lead to more effective perturbations and consequently more progressive motions in the particles.

The specific challenge of multi-modal search spaces is of course the existence of local optima. These local optima can exist within globally convex and non-globally convex search spaces. In a globally convex search space, there is a path from any local optimum to the global optimum passing through neighbouring optima that have a monotonic improvement in fitness. Subsequently, heuristic search techniques with highly localized search patterns can be very effective in these search spaces. Non-globally convex search spaces require different search behaviours, so it would be useful to be able to identify this search space characteristic.

The proposed line search method for the analysis of continuous domain search spaces (which uses some of the allotted function evaluations) can provide any search technique with useful information about the shape and features of a fitness landscape. Our primary objective with line searches is to determine the size of the search space's attraction basins. We have defined the concept of an attraction basin in continuous domain search spaces based on dividing a multi-modal search space into a countable number of regions which each have a single local optimum. Assuming some degree of symmetry, the size of an attraction basin can be measured by the distance between two adjacent local minima. However, it needs to be remembered that two considerations limit this method. First, a comprehensive analysis of the search space is akin to exhaustive search, a computationally infeasible task. Second, the goal of the analysis is to produce results that can be used in real-time to guide the search activity of PSO, so it must use a restricted number of function evaluations.

```
for i = 1: d do
    //set a⃗
    x_i = -5:12
    for j = 1: d do
        if i ≠ j then
            x_j = [-5.12; 5.12]
        end if
    end for
    //set b⃗
    x_i = 5.12
    for j = 1: d do
        if i ≠ j then
            x_j = [-5.12; 5.12]
        end if
    end for
   for j = 0:999 do
       x⃗_j = a⃗ + j /999 * (b⃗ - a⃗)
       record f(x⃗_j)
   end for
end for
```

Algorithm 1. General line search for Rastrigin

The developed method of line searches satisfies the above conditions since they use 1000d function evaluations (or 10% of the allotted amount). In particular, we perform d line searches (one in each dimension of a d-dimensional search space), and each search uses 1000 function evaluations. For each line search, we traverse a parameterized line that starts at a random point on one bounding hyperplane and ends at a second random point on the opposite hyperplane of the (hypercube) search space (See Algorithm 1).

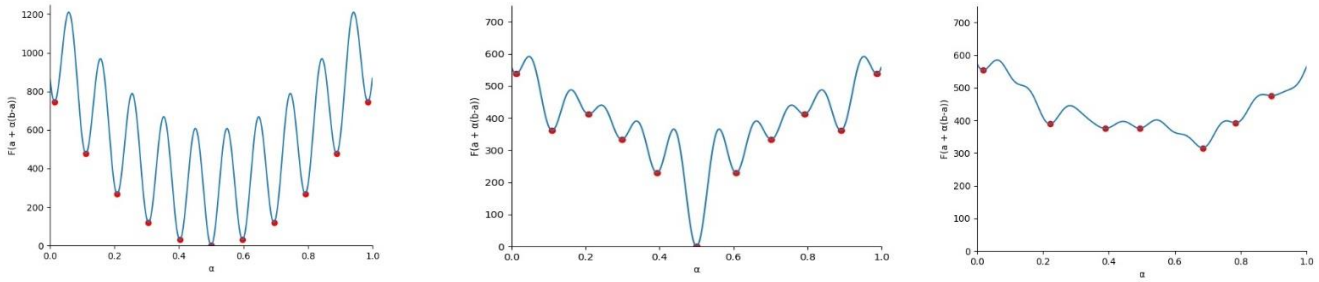
3.2.1 Line search on Rastrigin

A line search along the axis of a separable function will naturally reveal the size and shape of attraction basins for that search space. For generating functions such as Rastrigin (Equation 2), ideal profiles are

often shown as in Figure 5 (a) which is a line search of the main diagonal (e.g., $\vec{a}$ = -5.12 and $\vec{b}$ = 5.12). Test functions are often rotated, and transposed, and similar effects can be observed with a rotated and transposed line search. Figure 5 (b) shows a random line through the origin (i.e., $\vec{b} = -\vec{a}$), and Figure 5 (c) shows a random line (e.g., as generated by Algorithm 1).

Each of the sub-plots in Figure 5 show all 1000 points for a line search. The larger red dots indicate the observed local minima – any point which is less than both of its neighbouring two points. The main diagonal in Figure 5 (a) shows the symmetry and perfect global convexity of the Rastrigin function. In random diagonals, the correct number of local optima is usually observed, but the perfect convexity of the search space is often distorted – see the selected example in Figure 5 (b). A random line search often sees neither. The (typical) example shown in Figure 5 (c) has identified only 7 of the 11 expected attraction basins, and the (regular) shape of the search space is almost completely obscured.

The ideal case of a line search is to be along the axis for a separable function as this will reveal the exact size of the attraction basins in the search space. However, for this task, the minima identified by a random line search are still useful. In particular, the smallest distance between minima in Figure 5 (c) seems to be quite close to the actual distance. This measurement has been applied to 30 independent trials each in d = 30 dimensions (for 900 total measurements). The known distance between local optima along an axial direction in Rastrigin is 1, and the measured distance is 0.884 with a standard deviation of 0.192. This measurement appears accurate enough for use on a broader set of benchmark functions as will be presented part of the remedy in Section 4.4.1.

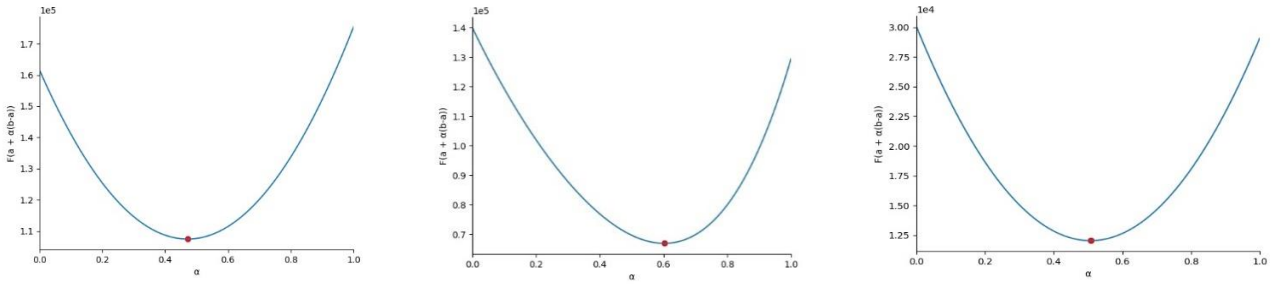


(a) main diagonal (b) random diagonal (c) random line

Figure 5. Examples of line searches for Rastrigin. Although random lines (c) often do not identify all optima (see (a) and (b) for examples), useful estimates.

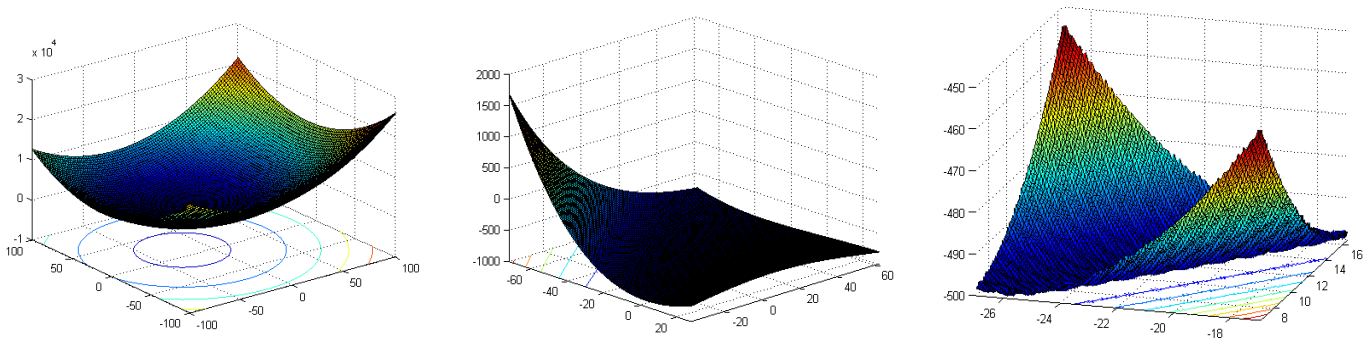
3.2.2 Line search on benchmark functions

Line searches reveal a unimodal landscape when they find only one minimum in the line. In Figure 6, we show three examples of line searches with a single minimum. However, it is noted that Figure 6 (a) shows the sphere function, which is unimodal, and Figure 6 (b) and Figure 6 (c) show F6 (Rotated Rosenbrock's Function) and F10 (Rotated Griewank's Function) which are multimodal functions. It is completely within expectations to find that the sphere function has one minimum in any direction/dimension (see Figure 7 (a) which presents F1 in 2D), but it is an odder result for the two multimodal functions (see Figure 7 (b) and Figure 7 (c) which present F6 and F10 respectively in 2D). We label these search spaces as "deceptive" since the overall search space has more than one attraction basin, and as result more than one local minimum, but the line search tool is not able to detect this.



(a) F1 (b) F6 (c) F10

*Figure 6. Line searches in benchmark functions; sphere (a), function 6 (multimodal) (b) and, function 10 (*multimodal*) (c) indicates unimodal shape.*

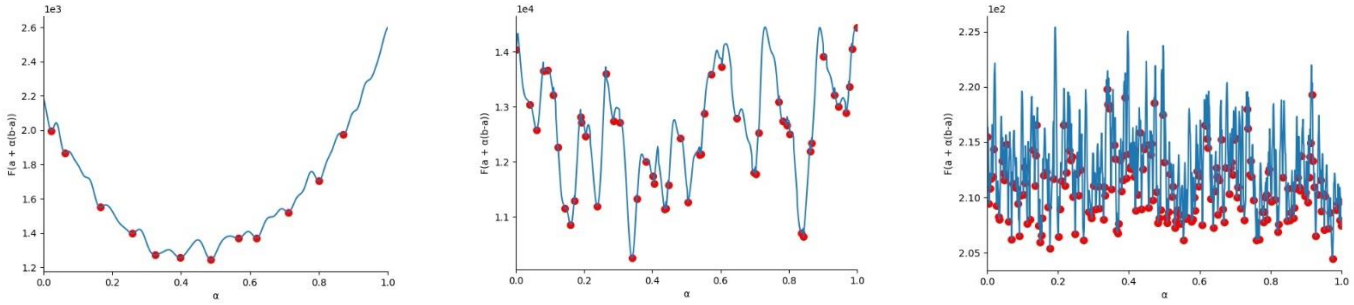


(a) F1 (b) F6 (c) F10

Figure 7. 3-D map for 2-D function [26]

Focusing on the rest of main multi-modal functions (F6-F20) in the benchmark set [26], there are three major patterns for the line searches as shown in Figure 8. The first sub-plot for F12 (Rotated Rastrigin's Function) in Figure 8 (a) shows a globally convex search space where the identified local optima reveal the key structure of the search space. The second sub-plot for F15 (Rotated Schwefel's Function) in Figure 8 (b) shows a partially structured search space where the identified local optima appear to show some structure within the sub-regions of the search space, but there is otherwise no information gleaned about

the size and shape of these sub-regions. The third sub-plot for F16 (Rotated Katsuura Function) in Figure 8 (c) shows a noisy search space where the identified local optima indicate that there is no structure to the search space.

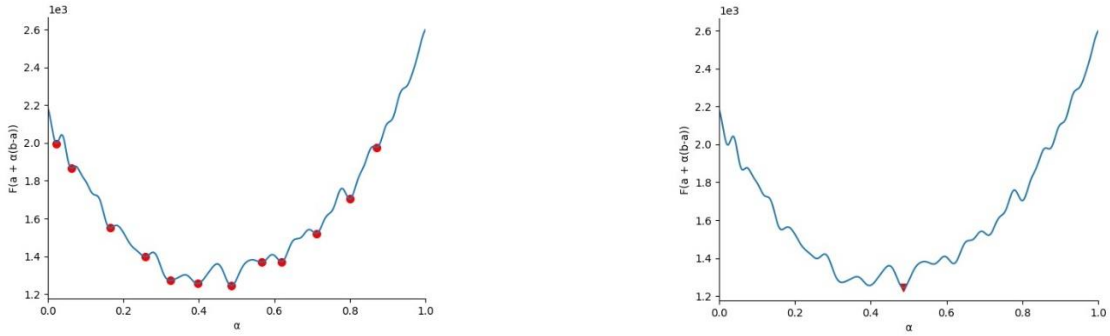


(a) F12 (b) F15 (c) F16

Figure 8. Examples of line searches on the CEC2013 benchmark functions. In addition to the spacing between the local optima, insights into the structure of the search space can be gleaned, Among the multi-modal functions, three major profiles emerge: globally convex (a), deceptive (b), and noisy (c).

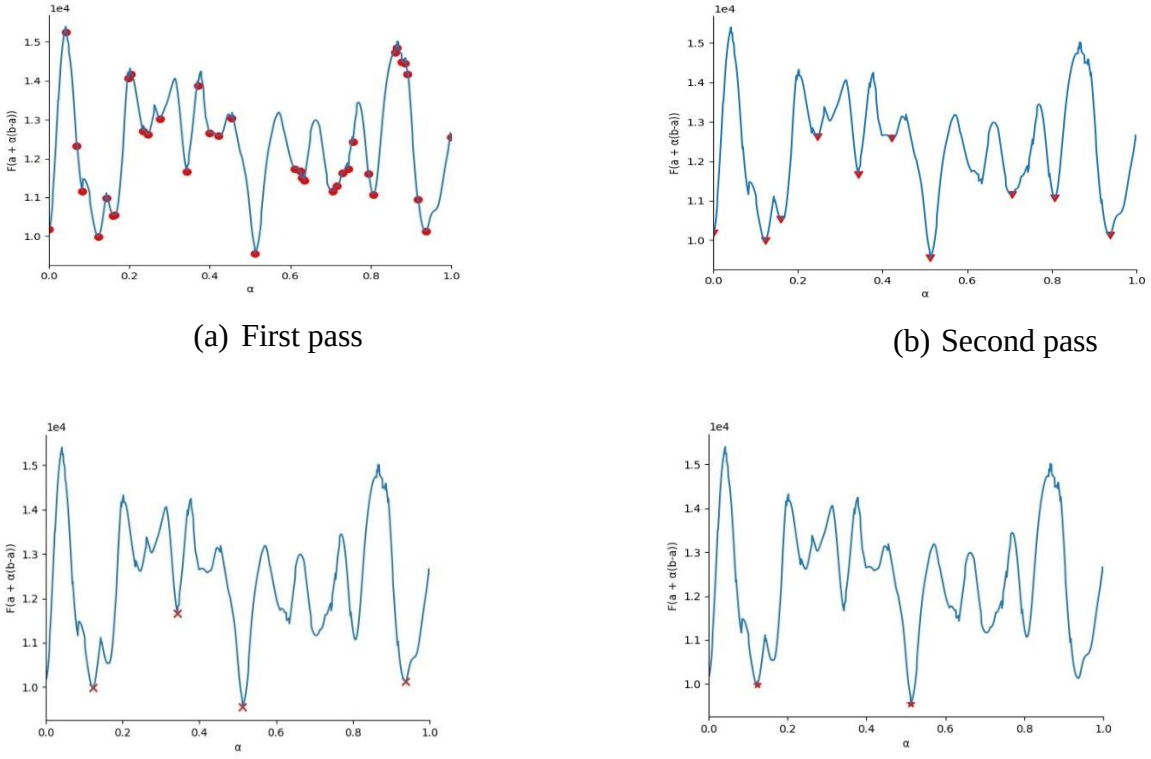
3.2.3 Structural analysis of search spaces with line search minima passes

All three functions in Figure 8 have multiple local minima. These are easily identified on a line search by finding an index i such that $f(\vec{x}_i) < f(\vec{x}_{i-1})$ and $f(\vec{x}_i) < f(\vec{x}_{i+1})$. However, these local minima can exist in globally or non-globally convex search spaces. By repeating the same procedure to find minima among the previously found minima (each procedure is called a pass), it can be determined if the minima exist in a single "meta-basin" or multiple "meta-basins". F12 (Rotated Rastrigin's Function) has one optimum (Figure 9 (b), second pass) among its previously found optima (Figure 9 (a), first pass) which presents a globally convex search space. F15 (Rotated Schwefel's Function) has two optima (Figure 10 (d), forth pass) among its previously found optima (Figure 10 (c), third pass) which presents a non-globally



(a) First pass (b) Second pass

Figure 9. The two levels of optima among optima (for two total passes) along a general line for function 12 (Rotated Rastrigin's Function).



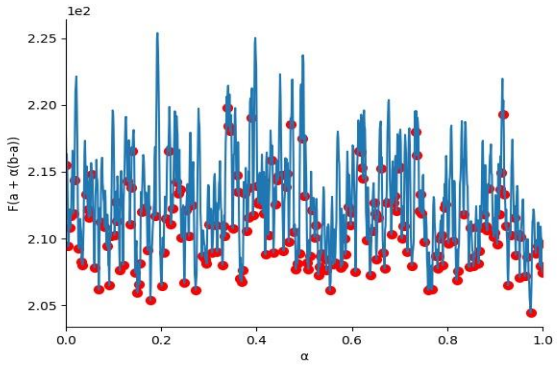
(a) First pass (b) Second pass

(c) Third pass (d) Fourth pass

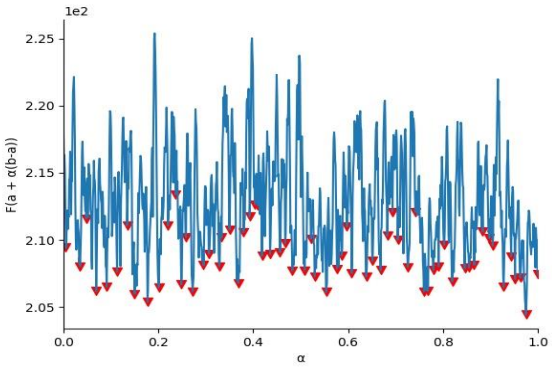
Figure 10. The four levels of optima among optima (for five total passes) along a general line for function 15 (Rotated Schwefel's Function).

convex search space. Insight into the structure of the search space is shown in Figure 10 (c) – there are four main meta-basins whose optima are at least 169 indices apart. We believe that this information could in the future be used by a metaheuristic during its exploration phase. In particular, being able to find the meta-basin with the global optimum should greatly improve later stages of the search process.

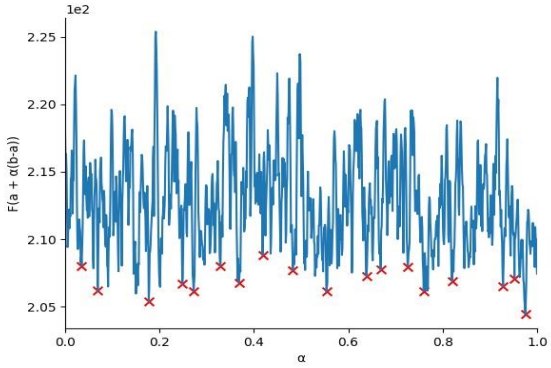
F16 (Rotated Katsuura Function) in Figure 11 shows a much higher than expected number of passes. Since this function has a large plateau with highly irregular (i.e., mostly random) local optima. This situation creates lots of optima among the optima, but it is not indicative of any global patterns among these optima. Tools to identify and remedies address this search space characteristic is also an area for future research.



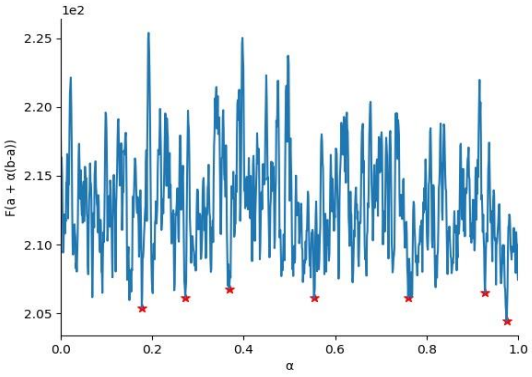
(a) First pass



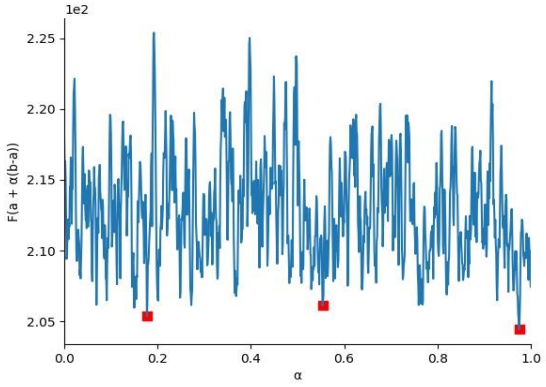
(b) Second pass



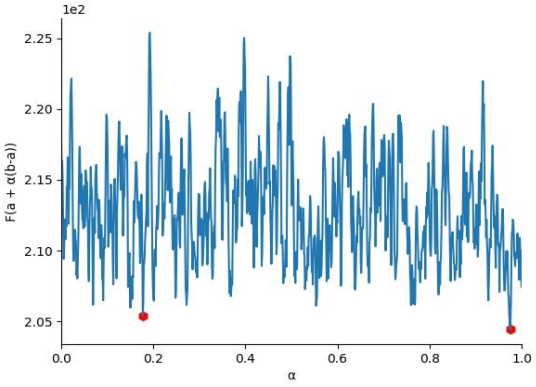
(c) Third pass



(d) Forth pass



(E) Fifth pass



(F) Sixth pass

Figure 11. The six levels of optima among optima (for seven total passes) along a general line for function 16 (Rotated Katsuura Function).

Functions	passes						
	1	2	3	4	5	6	7
Rastrigin	0	10	20	0	0	0	0
1	30	0	0	0	0	0	0
2	28	2	0	0	0	0	0
3	30	0	0	0	0	0	0
4	30	0	0	0	0	0	0
5	30	0	0	0	0	0	0
Unimodal							
6	30	0	0	0	0	0	0
7	0	0	0	15	15	0	0
8	0	0	0	0	0	8	22
9	0	0	0	0	14	14	2
10	30	0	0	0	0	0	0
11	0	1	29	0	0	0	0
12	0	13	17	0	0	0	0
13	0	0	0	30	0	0	0
14	0	0	10	20	0	0	0
15	0	0	0	12	18	0	0
16	0	0	0	0	0	24	6
17	0	0	0	24	6	0	0
18	0	0	2	24	4	0	0
19	30	0	0	0	0	0	0
20	28	0	1	1	0	0	0
Multimodal							
21	30	0	0	0	0	0	0
22	0	0	0	12	18	0	0
23	0	0	0	10	19	1	0
24	0	0	7	20	3	0	0
25	0	0	1	19	10	0	0
26	3	11	9	6	1	0	0
27	0	0	4	20	6	0	0
28	0	0	1	15	11	3	0
Composite							

Table 3. Structural analysis of search spaces with number of passes on benchmark set

To further study the operational characteristics of the proposed line searches, they have been applied to all of the functions in the CEC 2013 benchmark. 30 line searches are used on these 30-dimensional functions, and it then takes up to 7 passes to get to a single optimum. The number of line searches using from 1 to 7 passes is shown in Table 3.

The proposed line search technique is mostly successful in identifying the unimodal search spaces. Out of the 150 line searches on the 5 unimodal functions, 148 took only one pass to process, and the other 2 took two passes. Similar to the misidentification of Rastrigin as non-globally convex (see Figure 5 (c)), it is possible for a straight line through the search space to see a (curved) contour as a local optimum.

Conversely, several multi-modal search spaces were (mis) identified as being unimodal search spaces. Function F6 (Rotated Rosenbrock's) is (see Figure 6 (b)), so it is mostly unimodal. Like function F20 (Expanded Scaffer's Function) and (composition) function F21, it is only multi-modal in a small area of the search space which is easily missed by random line searches. A more interesting (mis) identification of unimodal search spaces occur for functions F10 (Rotated Griewank's Function) (see Figure 6 (c)), and F19 (Expanded Griewank's plus Rosenbrock's Function). The period of the sinusoid in Griewank's function is so short that these functions are effectively unimodal for the step size used in the line searches.

3.2.4 Application of Line searches in metaheuristics' initialization

Other than measuring the approximate size of attraction basins and estimating the global structure of the search spaces, line searches can also be used for improving any technique's performance in a more direct manner. After the line searches, the minimum from each line in each dimension can be assigned to a particle as its initial point (providing 30 initial particle locations out of 50). As it has been mentioned earlier, 10% of the allotted computation effort has been dedicated to the line searches, and the best location of each line can offer a better position in comparison to the typically used random initialization.

3.3 Cluster Detection in PSO

In Section 3.1.2 the emergence of particle clusters in Rastrigin during PSO has been discussed. However, the detection of clusters on Rastrigin used perfect knowledge of the search space, and search would not be necessary with this level of information. Therefore, practical methods must be developed for cluster detection in PSO. These techniques must be able to work correctly without having any information about the objective functions and/or the structure of the search space. In the next subsections we propose two methods to identify these clusters.

3.3.1 Cluster detection based on particles' location

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is proposed as an option to recognize groups of particles within PSO based on an individual's position information. This method identifies clusters of points in space based on density (points adjacent to many neighbors) and finds outliers in less dense regions. Two parameters must be addressed using this technique: Epsilon (specified radius) and MinPts (a specified number of points). Three different categories of points can be identified: core points; with an equal or larger number of points in the epsilon distance neighborhood, border points; have fewer points in the specified distance but are in the neighborhood of at least one core point, and lastly, outliers have none of the mentioned features [43].

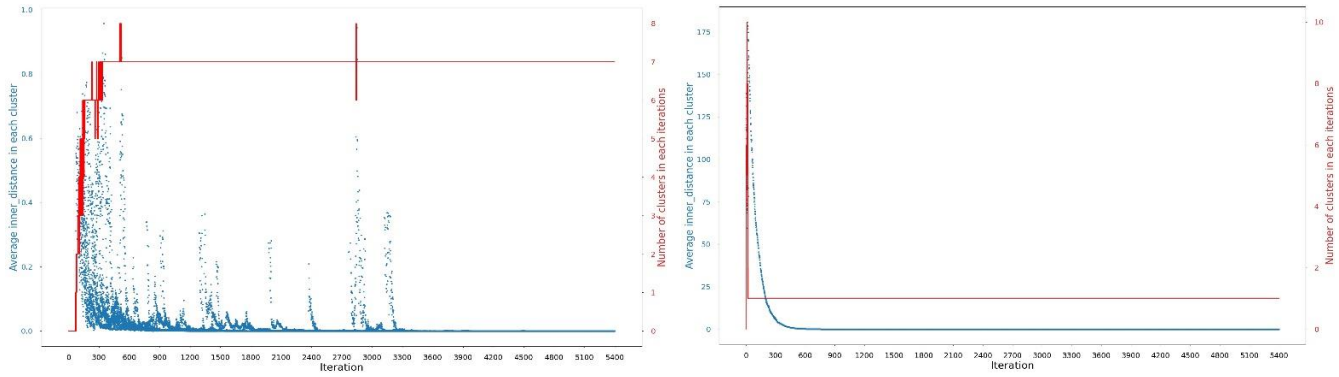
The reason for choosing the DBSCAN method is because the detection of outliers is important. It is likely that these outliers are particles that are not stuck in one attraction basin with other particles and that they are still performing exploration. Secondly, the clustering algorithm must be able to detect the clusters without knowing the number of clusters that it is going to find. These two important features led us to choose the DBSCAN option in order to find groups of particles in the search space.

It is noted that clustered particles are defined as a group of particles which are all located in the same attraction basin. Therefore, the estimated attraction basin size from the line search technique can be used as the radius in the DBSCAN method. The accuracy of cluster detection by the DBSCAN is tested on Rastrigin which has known and easily measurable attraction basins [27], and the results are shown in Table 4. We show the accuracy at several specific iterations (e.g., at every iteration 600 which is 10% of the allotted function evaluations), and also the overall accuracy averaged for all iterations from 600 to 6000.

Iterations	**Accuracy**
600	94.4%
1200	100 %
1800	100 %
2400	100 %
3000	100 %
3600	100 %
4200	100 %
4800	100 %
5400	100 %
6000	100 %
Overall	99.5%

Table 4. Accuracy of DBSCAN method in clusters identification

We begin by testing our DBSCAN analysis on the basic functions of Sphere and Rastrigin. It is noted that PSO encompasses 5400 generations, shown in Figure 12 on the x-axis. As mentioned in Section 3.2, 1000 function evaluations are needed for each linesearch, and this consumes 600 generations (30,000 function evaluations) for 30 line searches.



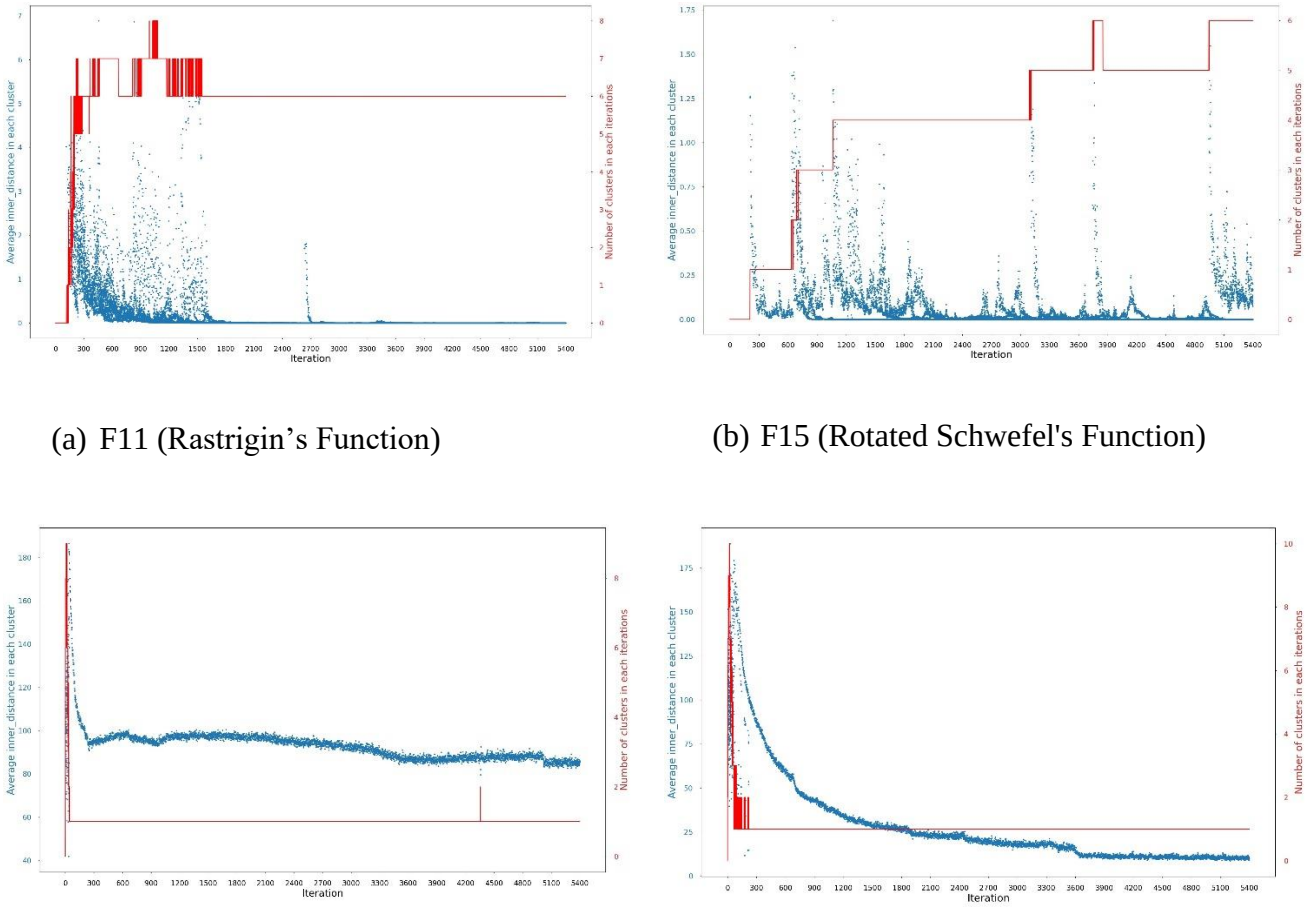
(a) Rastrigin (b) Sphere

Figure 12. Cluster detection by DBSCAN methods, Rastrigin (a), and F1 Sphere (b) shows stall and convergence respectively. Both functions have been considered in 30 dimensions. X-axis, left y-axis and right y-axis represents iterations, average inter distance of each cluster (blue dot) and number of clusters in each iteration (red line) respectively.

The average distance in a cluster is calculated based on the average distances of all particles within a cluster from its centroid, shown in Figure 12 on the left y-axis. As expected, convergence occurs quickly on the sphere function (see in Figure 12 (b)), and after a few iterations there is only one cluster of particles in which all of the particles are very close to each other. On the other hand, after a few iterations of PSO on the Rastrigin function, Figure 12 (a) shows a (semi) stable state with seven detected clusters. It should be noted that in the first half of iterations the number of clusters is steady, but they are not internally stable. Figure 12 (a) shows that there are several sporadic iterations in which clusters are getting rearranged and displaying very unstable clusters with high average inter distance – the particles are far away from each other. In the second half of the iterations, the stability in PSO is increased by having all of the clusters with very close inter-cluster distances. This situation represents our previously identified pattern for stall.

The stall pattern is also detected in several functions from the CEC2013 benchmark set, e.g., F11 (Rastrigin's Function) in Figure 13 (a). However, in Figure 13 (b) for F15 (Rotated Schwefel's Function), different formations of clusters can be inferred. In this semi-stall structure, both the number of clusters

and their inter distances never get settled. That can also be related to the function's structure because it is less close to the globally convex search space.



(a) F11 (Rastrigin's Function)

(b) F15 (Rotated Schwefel's Function)

(c) F6 (Rotated Rosenbrock's Function)

(d) F10 (Rotated Griewank's Function)

Figure 13 Cluster detection by DBSCAN methods, F11(Rastrigin's Function) (a), and F15(Rotated Schwefel's Function) (b) shows stall and semi stall, respectively. F6(Rotated Rosenbrock's Function) (c) and F10(Rotated Griewank's Function) (d) showing false convergence. X-axis, left y-axis, and right y-axis represents iterations, average inter distance of each cluster (blue dot) and number of clusters in each iteration (red line) respectively.

F6 (Rotated Rosenbrock's Function) and F10 (Rotated Griewank's Function) have been identified as deceptive by the line search technique. Therefore, the size of attraction basin is measured to be the full range (a full range is [-100,100] in each dimension) of the search space. As a result, 200 is considered as

the radius size for the DBSCAN method for these types of functions. The limitation of line search for approximating the proper size of attraction basins results in DBSCAN poor performance. Overall, it can be said the achieved states of F6 (Rotated Rosenbrock's Function) (see Figure 13 (c)) and F10 (Rotated Griewank's Function) (see Figure 13 (d)) are false representatives of convergence which can be seen through the large inter cluster distances. Moreover, for other functions in benchmarks sets, including F8 (Rotated Ackley's Function) and F16 (Rotated Katsuura Function) which have a noisy structure, the actual basin sizes are too small to be measured by the line searches; thus, the parameter value used by DBSCAN leads to an inability to detect any clusters. The Dependence of the DBSCAN method on more accurate information on the size of attraction basins led us to another element within the PSO to identify these clusters.

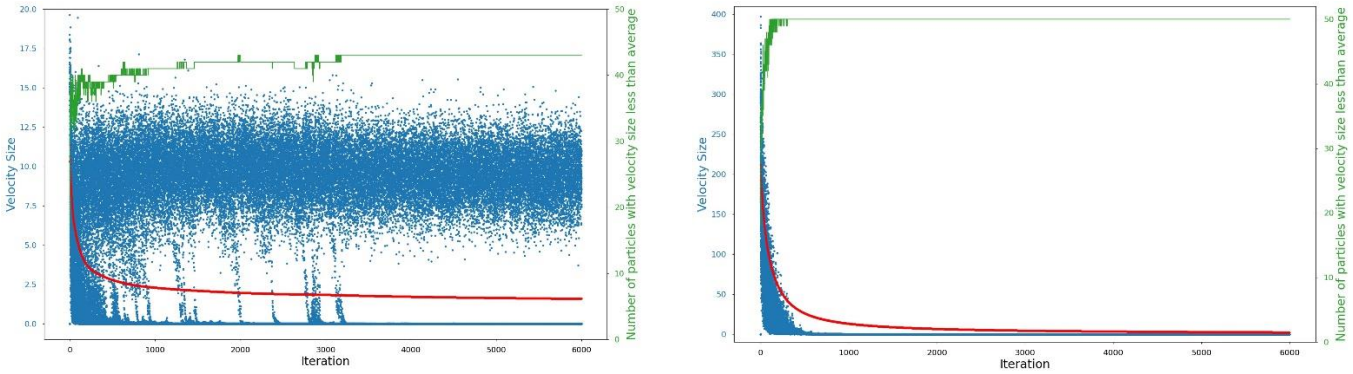
3.3.2 Cluster detection based on particle speed

We propose a computationally efficient, search space independent method to identify a stalled swarm. It is shown in Figure 4 that a stalled swarm can have a characteristic pattern of "clusters" of particles in the ring topology. Within each cluster, the particles tend to perform exploitation around the same local optimum. At the edge between neighbouring clusters on the ring topology, there is one particle with its *pbest* position in one cluster and its *lbest* position in another cluster, and this particle can perform a large amount of exploration. A simple velocity measurement allows us to identify these particles, and thus the clusters of exploiting particles that they separate.

The stall pattern shown in Figure 4 has some particles in clusters that are performing exploitation and some particles between the clusters that are performing exploration. The particles that perform only exploitation clearly have velocities which never lead them to escape the confines of their current attraction

basin – i.e., they are small in magnitude. The particles which perform exploration instead require velocities with sufficiently large magnitudes to allow them to visit different attraction basins. We believe we can thus identify exploring and exploiting particles based on their speed.

We measure the speed of every particle for our implementation of standard PSO when applied to the Rastrigin function in d = 30 dimensions. Figure 14 (a) shows the speed of each particle (blue dots), the cumulative average (mean) speed of all particles (red dots), and the total number of particles in each iteration with a speed below the cumulative average (green line). The cumulative average speed is used because the average speed cannot differentiate between stall and convergence. For example, Figure 14 (b) shows the data for Sphere in d = 30 dimensions, and all of the particles rapidly slow below the cumulative average speed while the average speed at a given iteration would obviously require some particles to have greater speeds. A converged swarm still has one cluster, but it is otherwise noted that the number of particles with above average speeds will match the number of clusters in the swarm. Note: the green line shows the number of particles with below average speed (the right y-axis). This leads to the following speed-based method for cluster identification.



(a) Rastrigin Function

(b) Sphere Function

Figure 14. Convergence vs stall based on velocity plots. Blue dots represent particle speed (n = 50 per iteration), red dots indicate cumulative average speed (one per iteration), and the green line shows the number of particles in each iteration that have a slower speed than the cumulative average.

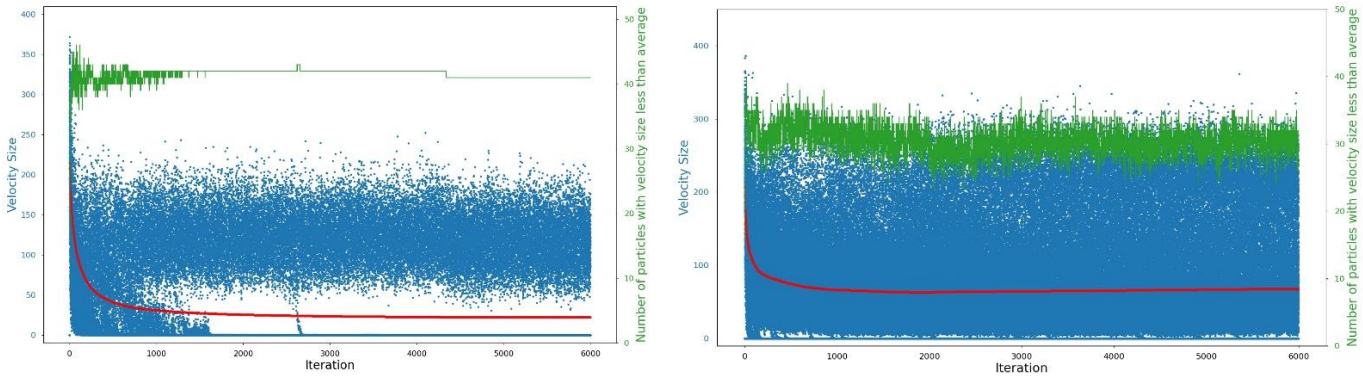
We first label the particles with below average speed to be exploitative and the particles with above average speed to be explorative. The higher speed explorative particles break the ring topology into clusters of particles which each perform exploitation around the same optimum. The identification of explorative and exploitative particles and the clusters of exploitative particles can both be determined in linear time (with respect to the number of particles n). The accuracy of the new speed-based method is tested on Rastrigin which has known and easily measurable attraction basins [27], and the results are shown in a Table 5. We show the accuracy at several specific iterations (e.g., there is 100% accuracy at iteration 600 which is 10% of the allotted function evaluations), and also the overall accuracy averaged for all iterations from 600 to 6000.

Iterations	Accuracy
600	100 %
1200	100 %
1800	100 %
2400	100 %
3000	100 %
3600	100 %
4200	100 %
4800	100 %
5400	100 %
6000	100 %
Overall	99.7 %

Table 5. Accuracy of speed-based practice identification for exploration and exploitation

The data presented in Figure 14 provides another observable difference between convergence and stall. The convergence condition shown in Figure 14 (b) for Sphere has all of the particle speeds drop below the cumulative average after about 10% of the iterations. The zero velocities indicate that all of the particles have reached the same location, and these are the two conditions which define convergence. The

stall condition shown in Figure 14 (a) for Rastrigin also enters a stable pattern after about 50% of the iterations. There are 43 particles with near zero velocity that are performing exploitation and seven particles with large velocities that are performing exploration. The lack of particle speeds between these two distinct sets (as seen sporadically between iterations 1000 and 3000) implies that there is no movement in *pbest* positions that could cause previously (near) stationary particles to begin moving.



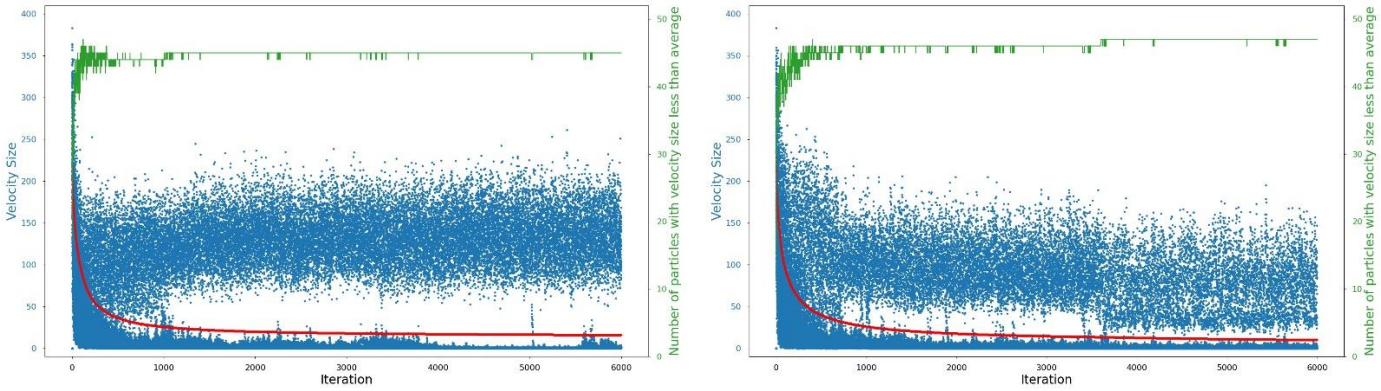
(a) Globally Convex Function (F11)

(b) Noisy Function (F16)

Figure 15. Velocity analysis in functions F11(Rastrigin's Function) (a), and F16(Rotated Katsuura Function) (b). Blue dots represent particle speed (n = 50 per iteration), red dots indicate cumulative average speed (one per iteration), and the green line shows the number of particles in each iteration that have a slower speed than the cumulative average. The clear gap around the red dots in (a) indicate the formation of large and stable clusters.

Two stall patterns that appear for two classes of fitness landscapes are highlighted in Figure 15. Figure 15 (a) shows the stall pattern for a "globally convex" function – F11 (Rastrigin's Function) from the CEC2013 benchmark. The clear separation between high speed (exploring) particles and low speed (exploiting) particles implies that the particles moving between the clustered particles find explorative solutions that are all rejected. Figure 15 (b) shows a stall pattern for a "noisy" function – F16 (Rotated

Katsuura Function) from the CEC2013 benchmark which has a large number of deep and narrow attraction basins (see Figure 8 (c)).



(a) F6 (Rotated Rosenbrock's Function) (b) F10 (Rotated Griewank's Function)

Figure 16. Velocity analysis in functions F6(a), and F10 (b). Blue dots represent particle speed (n = 50 per iteration), red dots indicate cumulative average speed (one per iteration), and the green line shows the number of particles in each iteration that have a slower speed than the cumulative average. All mentioned functions have shown unimodal structure in line search technique

The speed-based method is also able to identify stall conditions in the deceptive search spaces of F6 (Rotated Rosenbrock's Function) and F10 (Rotated Griewank's Function) (see Figure 13 (c), Figure 13 (d)) for which the DBSCAN method was unable to. As it can be seen in Figure 16, after 15% of the whole iterations, there are around five particles that are classified as explorative because they have above average velocity (the total number of particles in each iteration with a speed below the cumulative average is shown in green line). Although the stall pattern for these two functions is not as clean, the figure is obviously closer to stall (Figure 14 (a)) than to convergence (Figure 14(b)).

4 Stall Remedies and Results

It is shown in the previous section that the typical operation of PSO involves an initial phase where large amounts of Successful Exploration can occur. After this phase, PSO often enters a stall pattern where large amounts of Failed Exploration occur. We introduce new mechanisms to standard PSO which allow replication of the initial phase with large amounts of Successful Exploration. We then show that these modifications can lead to large improvements in PSO performance on multi-modal functions. All of the suggested remedies will be applied to a version of standard PSO that has been described in the beginning of Section 3.

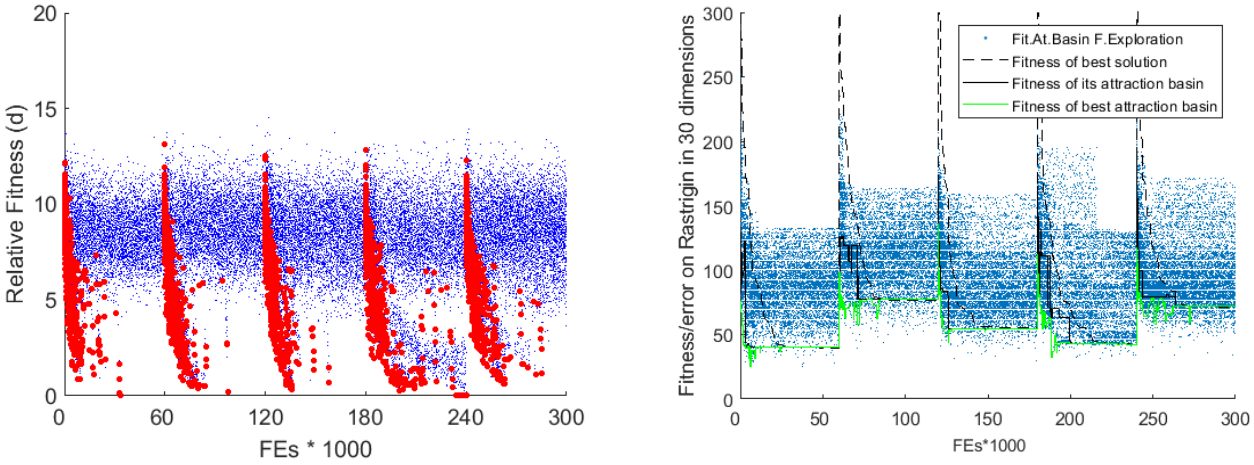
4.1 The effect of restarts in Stall

The simplest restart strategy is to perform k sets of PSO for 10,000d/k function evaluations each. Although more advanced restart strategies exist (e.g., [44] – [45]), the simplicity of this restart strategy helps our experimental analysis to highlight several general characteristics of all restart strategies. The time to converge/stall is approximately 15% of the allotted function evaluations. Setting k = 5 allows each sub-run 20% of the originally allocated function evaluations which is usually sufficient for the best *pbest* (i.e., *gbest*) position to reach a local optimum.

PSO	Dimension	mean	Std dev
Standard	3	0.0	0.0
	30	66.6	14.0
5 Restarts	3 (best of five)	0.0	0.0
	30 (first sub-run)	69.8	14.8
	30 (best of five)	51.5	10.7

Table 6. PSO with restarts

The results in Table 6 show that restarts have a minuscule effect for d = 3 dimensions – all sub-runs in all 30 trials reach the attraction basin of the global optimum, but some *gbest* positions are a negligible amount away from a 0 fitness and not all particles have converged to the basin with the global optimum. The effects of restarts for d = 30 dimensions are highlighted by recording results after the first sub-run and for the best overall of the k = 5 sub-runs. As can be expected, the result of the first sub-run, which is equivalent to PSO with fewer allotted function evaluations, is worse. However, PSO has largely stalled, and the improvements achieved by restarts show that it is more effective to completely restart (even randomly) than to hope for fortuitous occurrences of Successful Exploration.



(a) effects of selection

(b) swarm performance

Figure 17. The effects of selection are similar for each restart (a). The performance of each sub-run is also similar in that the best particle (dashed line) quickly reach the fitness of its local optimum (solid black line), and then there is little improvement thereafter (b). Essentially, each sub-run has stalled, and this is indicated by both the lack of red circles in the relative fitness plot (a) and by the lack of improvement in the performance plot (b). Each sub-run is independent, and this can lead to large variations in performance, especially in a negative direction [42].

Additional insights into the effects of restarts are available through our observations of Successful Exploration and Failed Exploration. Figure 17 shows two perspectives. In Figure 17 (a), the relative fitness

of exploratory search solutions which represent Successful Exploration (large red dots), and Failed Exploration (small blue dots) are shown. It can be seen that each restart leads to a large burst of Successful Exploration, and that each of the k = 5 sub-runs are largely similar.

Figure 17 (b) shows the actual/absolute fitness for each sub-run. The dashed line is the actual fitness of the best overall (*gbest*) position, and the solid line is the fitness of its attraction basin. Another solid line below (in green) shows the fitness of the best attraction basin for any *pbest* position. The light blue dots show the fitness for the attraction basins that are rejected under the category of Failed Exploration.

One observation to highlight is that each of the restarts/sub-runs operates at a different actual fitness (even though the relative fitness is quite similar). Through this variation, the best of the five sub-runs leads to a better overall result than a single sub-run and a single standard execution of PSO (see Table 6). Conversely, all of the sub-runs have blue dots that have better fitness (i.e., are below) the bottom solid line. These dots indicate that useful exploration is still occurring, and that the restarts can also relocate the swarm into a less promising region of the search space.

These two observations lead us towards the goal of achieving increased (bursts of) Successful Exploration with less disruption than (random) restarts. It can be seen in Figure 17 (b) that the dotted line has converged with the first solid line, and this indicates that the *pbest* position has approached its local optimum. The studies in [10], [27] show that local optimization of (*pbest*) reference solutions lead to drastically increased rates of Failed Exploration. What is necessary is not a restart of the swarm, but a reversing of the local optimization of the *pbest* position.

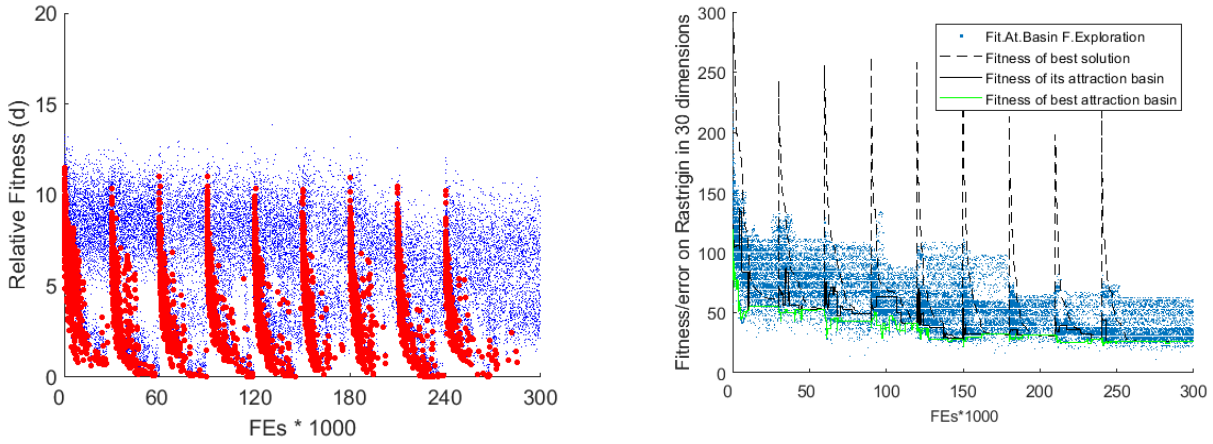
4.2 Perturbation around *pbest* with line search support

As presented in Figure 4, some particles are busy with exploitation for most of their time. After about 10 percent of iterations, most of the particles start exploiting. Instead, these wasted resources could be used for scouting more areas of the search space. The problem is that the velocity of the exploitative particles is low. So, they are unable to move to newer attraction basins to perform exploration. As a result, any explosive motion, even a random one, can add more motion to the system. However, large movements of the current positions can cause inefficient exploration since search solutions are highly likely to get rejected when they are compared to existing *pbest* (reference) solutions that have approached their local optima (i.e., Failed Exploration is likely to occur).

Based on the provided insights about the random restart, a more suitable search trajectory can be created by perturbing the particles in the neighborhood of their *pbest*s. Perturbation of current solutions around their *pbest*s increases the amount of exploration in the clustered particles. In this remedy, the fitness of references solutions also "resets" to the current perturbed location, and this "reverse optimization" leads to increased rates of Successful Explorations.

The local optima in the Rastrigin function are located at the points in the search space with integer values in all dimensions. The size of the attraction basin is thus ±0.5 in each dimension around each local optimum. The experiments in [27] show that a random solution in the $\mathbf{x} = -3$ basin can achieve Successful Exploration over 75% of the time whereas a solution that has moved even half of this distance towards its local optimum will achieve less than 1% Successful Exploration. To improve the rates of Successful Exploration without losing forward progress of the swarm, we "perturb" the *pbest* (and current position) of each particle to become a random point within its current attraction basin. Assuming that the *pbest* position has reached its local optimum, the perturbation is to add a random uniform value in the range of [-0.5, 0.5] to each term of the solution.

The perturbations occur within a single execution of PSO, so the swarm does not have to converge after the interim perturbations. Thus, rather than using the restart level of 20% of the function evaluations which allows the fitness of the best overall position to reach a local optimum (see Figure 17 (b)), perturbations occur every 10% of the function evaluations as this allows the swarm to mostly reach the end of the initial wave of Successful Exploration (see Figure 17 (a)). Perturbations thus occur at 10, 20, 30, 40, 50, 60, 70, and 80 percent of the allotted function evaluations (which gives the final perturbation a good chance to reach its local optimum).



(a) effects of selection

(b) swarm performance

Figure 18. Perturbations have similar effects on selection as restarts (a). These bursts of Successful Exploration are also visible in the performance of the swarm (b) – especially in the movements of the solid lines. There is also a steadier trajectory of continuous improvement with perturbation compared to the more random changes caused by restarts.

PSO	Dimensions	mean	Std dev
Standard	3	0.0	0.0
	30	66.6	14.0
5 restarts	3 (best of five)	0.0	0.0
	30 (best of five)	51.5	10.7
Pbest perturbation	3	0.0	0.0
	30	29.0	4.9

Table 7. PSO with pbest perturbations

The addition of perturbations leads to much larger improvements than the addition of restarts – see Table 7. The actual effects of perturbations are better observed in Figure 18. First, Figure 18 (a) shows that the desired replication of bursts of Successful Exploration has been achieved. Figure 18 (b) then shows that these bursts occur within the context of a progressing, single search trajectory. In particular, the "deteriorating" step after the first restart in Figure 17 (b) is replaced by steadier progress as seen in Figure 18 (b).

The introduction of perturbations increases both the probability of Successful Exploration and Deceptive Exploration. The effects of Deceptive Exploration can be observed by the upward movements of the solid lines in Figure 18 (b) which indicate that *pbest* positions have moved into worse attraction basins. Further, the swarm still stalls short of the global optimum (in d = 30 dimensions). Nonetheless, these promising initial results warrant a further analysis on the effects of perturbations across a broader range of functions. The results of *pbest* perturbation remedy on the CEC2013 benchmark set will be provided in Section 4.4.1, and the results get compared with the 5 random restart solution.

4.3 Adaptive topology based on detected clusters in PSO

The standard PSO algorithm, *gbest* or *lbest* topology, is considered a homogeneous swarm. In these PSO types, all particles have the same number of neighbors, interact according to the same model of influence, and modify their velocities applying the same update rules. Furthermore, the parameters of the velocity update rule are the same for all particles (see Equation 3) [46]. In these two remedies, whether in 5 random restarts of PSO or *pbest* perturbation on all particles of PSO, a homogeneous approach has been used to address the stall situation.

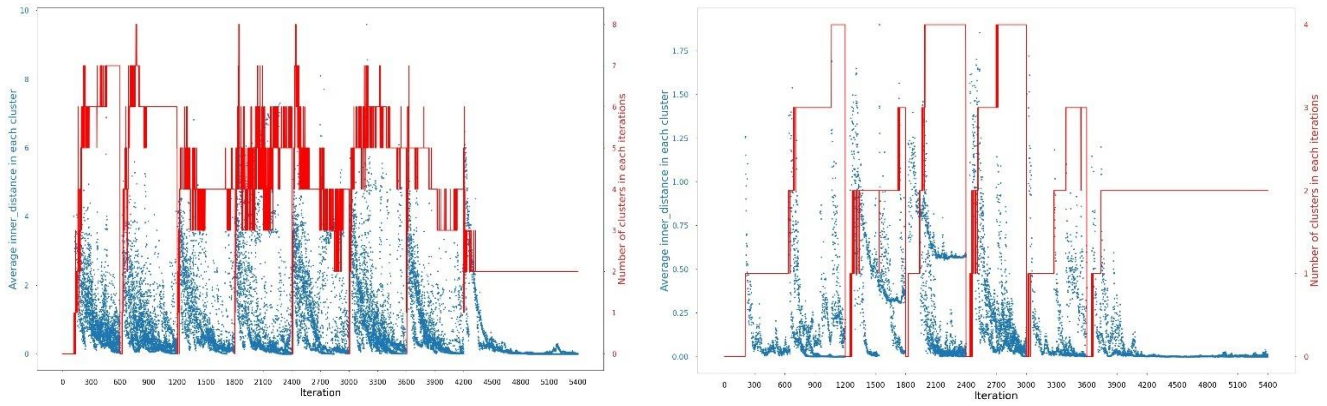
In addition to the many homogenous approaches for improving PSO, there have also been extensive studies on heterogeneous approaches. PSO is heterogeneous when at least two particles differ in any of neighborhood size, model of influence, update rule, or their parameters. If particles change configuration over time, the resulting heterogeneity is qualified as dynamic, and it is static otherwise. Adaptive particle swarms at the individual level build on dynamic heterogeneity by triggering configuration changes as a response to some event caused by the swarm's behaviour [46].

In the following remedies, adaptive topologies are applied on standard PSO. The standard PSO implementation and how it is updated are described at the beginning of Section 3. Every 600 iterations (which is 10% of the allotted function evaluations), we apply the following procedures to address an assumed stall condition. (Note: we do not apply the remedy at the 90% mark which gives the last modification at the 80%-mark sufficient time to settle). The identification of the particle clusters is performed as described in Section 3.3.1 (cluster detection based on particles location) and Section 3.3.2 (cluster detection based on particles speed). We discard all clusters with fewer than three particles. The rest of the details for each remedy with adaptive topology is described in the next two subsections.

4.3.1 Remedy when detected clusters are by DBSCAN

Based on Figure 12(a), the particles which are together as a cluster, are doing exploitation and not communicating to particles within the other clusters. In the second half of iterations, the average inter distance of particles in each cluster is very close to zero, so these separated clusters are effectively acting like different swarms. As a result, forcing them closer together to make them share information with each other can help the particles to explore more promising regions.

Every 600 iterations, clusters are detected by the DBSCAN method. If there are at least three clusters, the fittest member of each cluster is chosen, and they are put together as neighbors (in both ring topology and location). We assign perturbed locations of the centroid of clusters (see *O* in Figure 21) as the new locations of the chosen particles. All the rest of the particles are perturbed around their *pbests*. This perturbation is exactly the same as the one that has been introduced in the previous remedy. It must be noted here that the whole number of generations is 5400 since 600 out of them have been used for line searches. The initial velocity of each particle is zero, so the overall procedure can be viewed as seven targeted restarts. Elitism is added by copying the *gbest* position from before the reset to replace the worst particle after the reset.



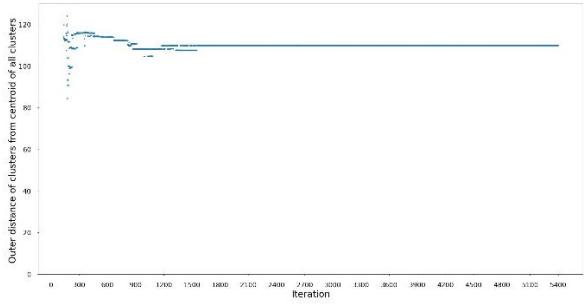
(a) F11 (Rastrigin's Function) (b) F15 (Rotated Schwefel's Function)

Figure 19. Cluster detection by DBSCAN methods after applying adaptive topology on PSO for F11(Rastrigin's Function) (a), and F15 (Rotated Schwefel's Function) (b). X-axis, left y-axis, and right y-axis represents iterations, average inter distance of each cluster (blue dot) and number of clusters in each iteration (red line) respectively.

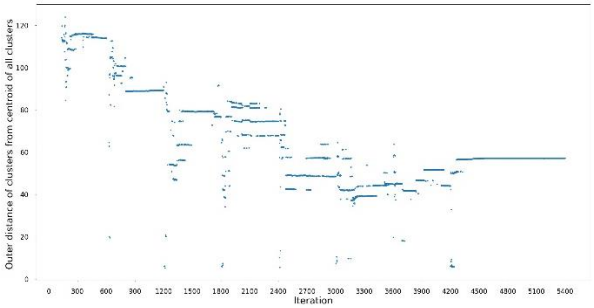
Based on the comparison between Figure 19 and Figure 13, it can be inferred that F11(Rastrigin's Function) and F15 (Rotated Schwefel's Function) show a more stable state after applying the remedy. Clusters got closer, which means they have been merged into larger and fitter clusters. Every 600 iterations, they will be given another opportunity to explore more promising areas the of search space and perturbation also gives them greater opportunity to accept their current solution in these promising areas as a new *pbest* solution. This idea can be further observed by the average outer distance of clusters from the centroid of all clusters.

As it is shown on Figure 20, after applying adaptive topology on PSO, clusters are getting closer and offer a better state to be convergent toward a fitter area. Figure 20 (c) and Figure 20 (d) indicate that the convergence of clusters occurs less in F15 (Rotated Schwefel's Function) by applying the remedy since the average outer distance of clusters from the centroid of all clusters is so high. It can be said that one state of a stall has been transformed into another stall condition. However, there is some trace of improvement in distances of clusters from each other (in the second half of iterations, the average outer

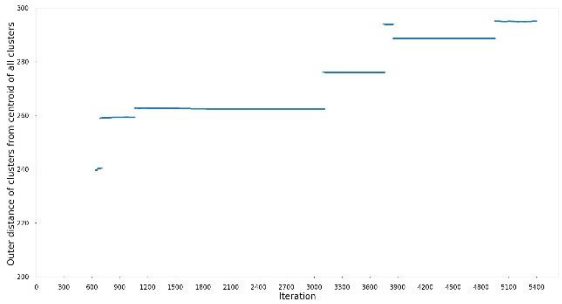
distance of clusters has been changed from 300 to 250). In globally convex search space, which the stall was recognized well by DBSCAN method including F11 (Rastrigin's Function) (see Figure 19 (a)), the comparison between Figure 20 (a) and Figure 19(b) reveals that in F11 (Rastrigin's Function), clusters got merged and created bigger clusters in a fitter area of the search space (see Table 10).



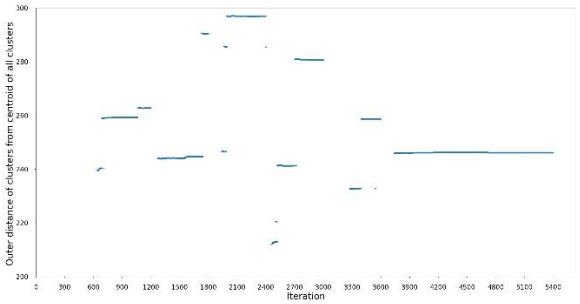
(a) F11-- PSO



(b) F11--PSO with adaptive topology



(c) F15 --PSO



(c) F15-- PSO with adaptive topology

Figure 20. Cluster detection by DBSCAN methods before and after applying adaptive topology on PSO for F11 (Rastrigin's Function) (a), and F15 (Rotated Schwefel's Function) (b). X-axis, y-axis represents iterations, average outer distance of all clusters from centroid of all clusters (blue dot).

4.3.2 Remedy when detected clusters are by speed data

Figure 15 (b) displays a swarm with a high average speed and a relatively low number of particles with below average speed (about 30), and these features are consistent with clusters that have a small number of particles which will be able to perform very little exploitation. This lack of exploitation can also be inferred from the number of updates to *pbest* positions during a run of PSO. Our measurements show an average of 11.0 updates per iteration for F11 (Rastrigin's Function) and 0.2 for F16 (Rotated Katsuura Function). The following cluster based remedy (artificially) creates clusters of particles, and these clusters also support exploitation.

The procedure starts with an identification of clustered particles which is a signature feature of the stall condition, and it then creates new artificial clusters which are designed to increase the opportunity for Successful Exploration. These modifications which concentrate exploration around previously found local optima are primarily designed for structured (e.g., globally convex) search spaces.

Every 600 iterations, clusters are detected based on the speed method. We discard all clusters with fewer than three particles, and we then keep only the fittest particle from the remaining clusters. If there are more than nine clusters, only nine of them are chosen (randomly). Using the saved particles/clusters, a centroid is determined. If there are fewer than nine initial clusters/positions, we create additional new clusters around (randomly chosen) "reflection" points (see Figure 21). If there are fewer than five clusters, we also create new clusters around (randomly chosen) "projection" points (see Figure 21). If there are fewer than three clusters, we do not apply the remedy (and this has occurred only for the unimodal functions in [26]).

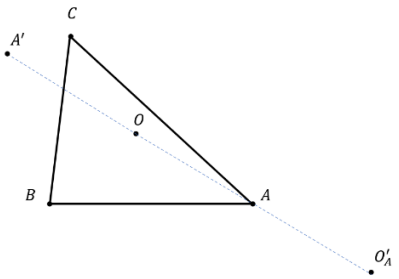
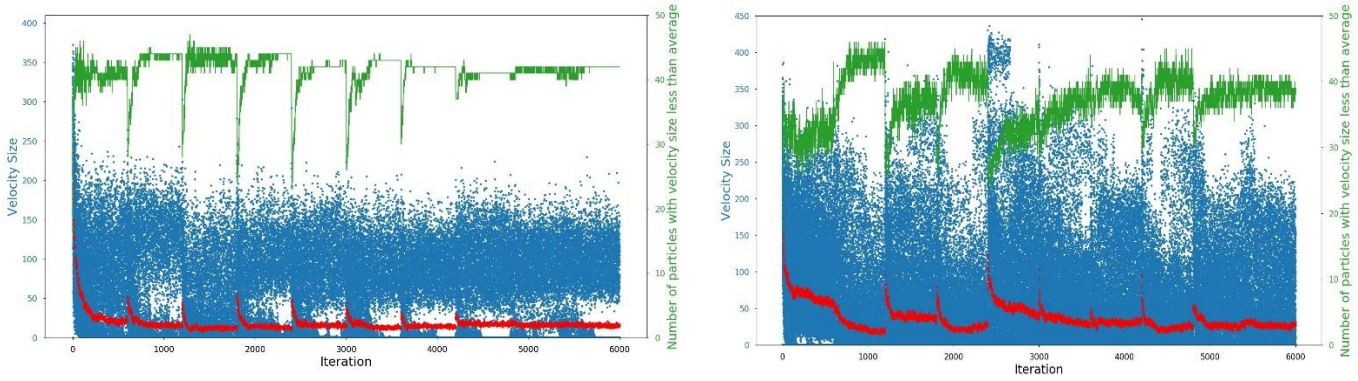


Figure 21. Each vertex represents the fittest particle from a cluster (shown in two dimensions for simplicity). O, O'_A, and A' are the centroid of the clusters, an example of reflection point (of A with respect to O), and an example of a projection point (of A through O) $A' = 2 * O - A$, $O'_A = 2 * A - O$.

There are three components to the proposed modification to address stall in PSO: the clusters are rearranged to promote new particle paths between neighbouring clusters, new clusters are created in promising areas (e.g., the centroid), and perturbations are used within the new clusters to promote Successful Exploration. Every 600 iterations, we create ten new clusters of particles around existing particle positions (which we expect to be local optima), the centroid of these clusters, and if necessary, reflection and projection points. The ten initial particle positions are assigned in random order to the indices 2, 7, 12, 17, 22, 27, 32, 37, 42, and 47. Artificial clusters are then created around these particles using perturbations (which increase the chances for "Successful Exploration" [42]). For example, particles 0, 1, 2, 3, and 4 are all perturbed a small amount from the previous position of the particle assigned to index 2.



(a) Globally Convex Function (F11)

(b) Noisy Function (F16)

Figure 22. Speed data after applying our stall remedy on (a) Globally Convex Function – F11 (Rastrigin's Function) and (b) Noisy Function – F16(Rotated Katsuura Function). The bursts of activity when the modifications are applied every 600 iterations are clearly visible. Compared to Figure 15, the average speed (red dots) is lower for both functions. This indicates closer clusters in (a) which supports more focused exploration in a globally convex search space and larger clusters in (b) which supports more exploitation.

Perturbations based on the measured size of the attraction basins from line searches [42] *pbest* have been compared against generic sizes of 0.5, 1, and 2 percent of the range of the search space. The results were generally consistent, so line searches have been excluded from the current implementation for simplicity. Of the generic sizes, a maximum perturbation of 0.5 percent of the search space led to the best results, so this value is used for the presented results. Each particle in a new cluster is given an initial position that is the seed position plus or minus a uniform random perturbation in each dimension up to the maximum amount. The initial velocity of each particle is zero, so the overall procedure can be viewed as eight targeted restarts. Elitism is added by copying the *gbest* position from before the reset to replace the worst particle after the reset.

The effect of our modifications to PSO can be observed in Figure 22. Compared to the stable stall pattern in Figure 15 (a), Figure 22 (a) shows the desired additional activity with each restart. This restart-based activity is also present in Figure 22 (b), but the new stall pattern is generally more stable than the

initial stall pattern shown in Figure 15 (b) (e.g., more particles with below average speeds in green and lower overall average speeds in red). The artificial formation of clusters for F16 (Rotated Katsuura Function) leads to more opportunity for exploitation, and we now record an average of 3.5 *pbest* updates per iteration. The benefits from these increased opportunities for exploration and exploitation are now presented for the full set of CEC2013 benchmark functions.

4.4 Remedies benchmark results reports

This section presents a computational study of the previously developed stall remedies. The test problems of the CEC2013 special session on real-parameter optimization [26] (CEC2013 benchmark) are used to evaluate the performance of the algorithms. All of the functions are considered in d = 30 dimensions.

The experimental setup follows the directions given in the CEC2013 benchmark. A total of 30 randomized trials with a maximum allocation of 300,000 function evaluations were performed on each function, and the mean and standard deviation of the fitness errors are presented in the following subsections (see Table 8, Table 10 and Table 12). The relative difference (%-diff) of the mean errors for m_1, "improved PSO" (with the stall remedy), versus m_2 , our benchmark for standard PSO, is calculated as 100*(m_2 - m_1) / max (m_2, m_1). Positive values indicate by what amount (percent) m_1 outperforms m_2 – negative values indicate the opposite. A standard *Man_Whitney*-test is also performed to measure statistical significance for the two data sets which are independent and ordinal. Results with statistically significant differences (p − value < 0.05) and improvements of at least ten percent are highlighted in bold. Under the null hypothesis H0, the distributions of both populations are indistinguishable. The alternative hypothesis H1 is that the distributions are different [47].

4.4.1 Perturbation PSO vs restarts PSO

We now apply PSO with *pbest* perturbations to the CEC2013 Real-Parameter Optimization Benchmark Functions [26]. The updated algorithm proceeds as follows. The first 10% of function evaluations are now used to perform the line searches. To further leverage these function evaluations, the initial swarm is seeded with the best solution from each of these d = 30 line searches and 20 additional uniform random solutions (as opposed to 50 uniform random solutions). During the operation of PSO, perturbations occur after 20, 30, 40, 50, 60, 70, and 80% of the allotted function evaluations. Each perturbation adds in each dimension a uniform random value of ±0.5 times the distance between local optima as measured by the line search in Section 3.2.1. Further, to reduce the deteriorating fluctuations after perturbations, the *gbest* position from before the perturbations is copied to replace the worst *pbest* position produced after the perturbations. The results of 5-restart remedy and pbest perturbation remedy are presented and compared together in Table 8.

Perturbations are better than restarts on unimodal functions (F1-F5) because they are less disruptive to the swarm's convergent trajectory to the global optimum. Perturbations still perform worse than standard PSO (without restarts), and this makes sense under No Free Lunch [48] since perturbations have been added to achieve improvements on multi-modal functions. On the targeted multi-modal functions (F6-F20), perturbations achieve just over 25% average improvement compared to restarts. It is further noted that 10 of the 15 functions in this set achieve the threshold for a large improvement (as shown in bold Table 8). Lastly, the composite functions (F21-F28) show no discernible pattern.

An observation worth highlighting is the relationship between the profile of the line search and the performance of PSO with *pbest* perturbations. The globally convex functions, e.g., F18 (Rotated Lunacek Bi_Rastrigin Function), tend to have the best performance. The partially structured functions with large sub-structures (e.g., F15 (Rotated Schwefel's Function) as shown in Figure 8 (b)) lead to smaller but still

meaningful performance improvements. The noisy functions (e.g., F8 (Rotated Ackley's Function)) can lead to noisy results. In general, perturbations have been designed to help PSO escape from the current local optima. Implicitly, PSO will then be able to search nearby attraction basins to find their (local) optima, and this will be most effective if this type of localized hopping among nearby attraction basins can lead to the global optimum as it could in a globally convex search space.

It is noted that PSO is a convergent algorithm, so it should not be expected to perform well on noisy functions (in which convergence to one good solution/local optimum does not help to find another/the next good solution). Matching a suitable metaheuristic (and/or one of its variations/modifications) to measured search space characteristics is a promising direction for further research.

More detailed information about the comparison of *pbest* PSO and restart PSO on multimodal functions (F6-F20) are presented in Table 9. The overall best result is happening in functions with Deceptive and Noisy structures. The average outperformance is 40.1% and 40%, respectively, in the two mentioned groups of structures. Globally convex search spaces show 21.1% improvement, and partially structured search spaces are in the last place with 13.3% outperformance.

No.	5 Restarts		Perturbation		%_diff	*Man_whitney_test*
1	1.73e--8	1.07e--8	2.27e--13	0.00e+0	**100**	**0.00**
2	5.86e+6	2.32e+6	3.00e+6	1.18e+6	**48.8**	**0.00**
3	2.33e+8	1.48e+8	4.87e+7	3.22e+7	**79.1**	**0.00**
4	4.15e+4	8.25e+3	2.49e+4	5.87e+3	**40.1**	**0.00**
5	3.50e--5	1.61e--5	1.79e--10	1.64e--10	**100**	**0.00**
1-5					73.6	
6	1.92e+1	7.14e+0	1.78e+1	4.68e+0	7.6	0.03
7	5.45e+1	1.12e+1	6.30e+1	1.68e+1	-15.6	0.06
8	2.10e+1	5.39e--2	2.09e+1	5.62e--2	0.4	**0.00**
9	2.80e+1	3.05e+0	2.77e+1	2.15e+0	1.2	0.74
10	3.21e+0	1.29e+0	2.09e--1	1.08e--1	**93.5**	**0.00**
11	6.03e+1	1.31e+1	4.57e+1	1.08e+1	**24.1**	**0.00**
12	9.07e+1	2.52e+1	7.26e+1	1.44e+1	**20.0**	**0.01**
13	1.51e+2	2.74e+1	1.31e+2	2.41e+1	**13.3**	**0.00**
14	2.54e+3	4.41e+2	2.35e+3	4.40e+2	7.6	0.14
15	4.78e+3	6.80e+2	3.29e+3	3.85e+2	**31.1**	**0.00**
16	2.09e+0	3.39e--1	4.25e--1	1.32e--1	**79.6**	**0.00**
17	1.17e+2	1.54e+1	8.78e+1	1.27e+1	**24.7**	**0.00**
18	2.18e+2	2.21e+1	8.70e+1	1.11e+1	**60.0**	**0.00**
19	7.05e+0	1.55e+0	3.48e+0	7.46e--1	**50.7**	**0.00**
20	1.23e+1	3.35e--1	1.12e+1	4.17e--1	8.7	0.00
6-20					27.1	
21	2.48e+2	4.07e+1	2.27e+2	6.39e+1	8.7	0.00
22	2.79e+3	3.66e+2	2.66e+3	2.99e+2	4.8	0.23
23	5.23e+3	7.15e+2	3.73e+3	3.90e+2	**28.6**	**0.00**
24	2.71e+2	6.96e+0	2.69e+2	6.65e+0	0.8	0.32
25	2.86e+2	7.11e+0	2.86e+2	6.87e+0	0.0	0.84
26	2.18e+2	5.18e+1	2.52e+2	7.50e+1	-15.9	0.09
27	1.02e+3	5.42e+1	9.98e+2	5.71e+1	1.9	0.20
28	3.00e+2	1.29e--2	2.93e+2	3.65e+1	2.2	0.00
21-28					3.9	

Table 8. Comparison of 5 restarts and perturbation in PSO

No.	Structure	Diff %	Man_whitney_test
7	Globally Convex	-15.6	0.06
11	Globally Convex	**24.1**	**0.00**
12	Globally Convex	**20.0**	**0.01**
13	Globally Convex	**13.3**	**0.00**
17	Globally Convex	**24.7**	**0.00**
18	Globally Convex	**60.0**	**0.00**
		21.1	
6	Deceptive	7.6	0.03
10	Deceptive	**93.5**	**0.00**
19	Deceptive	**50.7**	**0.00**
20	Deceptive	8.7	0.00
		40.1	
9	Partially Structured	1.2	0.74
14	Partially Structured	7.6	0.14
15	Partially Structured	**31.1**	**0.00**
		13.3	
8	Noisy	0.4	0.00
16	Noisy	**79.6**	**0.00**
		40.0	

Table 9. Results of PSO with Pbest vs 5 restarts PSO based on structure

4.4.2 Results of PSO with adaptive topology (by DBSCAN) vs PSO

The suggested remedy hasn't been useful for unimodal functions (F1-F5) since DBSCAN is not able to detect the proper state of a stall for unimodal functions or deceptive functions with unimodal shape (revealed by line search including F6 (Rotated Rosenbrock's Function), F10 (Rotated Griewank's Function), F19 (Expanded Griewank's plus Rosenbrock's Function) and F20 (Expanded Scaffer's F6 Function) in Table 3). As it has been discussed earlier, no information is revealed in terms of attraction basin size for deceptive functions, so no information is passed to the DBSCAN method for radius data (by default it is the full range of search space). These shortcomings in adaptive PSO have resulted in either no progress or even an adverse situation in these functions.

Furthermore, this adaptive topology for PSO is primarily intended for globally-convex, multi-modal search spaces. On the multimodal functions (F6-F20 – which also include multi-modal functions without global convexity), the average improvement of 14.74% compared to standard PSO can be seen in Table 10. Within this subset, our method achieves meaningful improvements (%-diff greater than 10% and p-value less than 0.05) on 6 of the 15 functions.

Another weakness of the proposed adaptive PSO is related to its performance on noisy functions. In these types of functions with small attraction basin, the radius size which is passed to the DBSCAN method are too small, thus most of the time no clusters have been detected and as a result, no improvement can be seen.

No.	Standard PSO		Adaptive topology		%_diff	*Man_Whitney_test*
1	0.00e+0	0.00e+0	1.32e+1	3.74e+0	NC	NC
2	2.18e+6	9.67e+5	0.00e+0	8.62e--14	-29.5	0.04
3	7.29e+7	9.87e+7	2.83e+6	1.22e+6	-35.3	0.45
4	1.64e+4	4.09e+3	9.87e+7	1.13e+8	-34.9	0.00
5	0.00e+0	0.00e+0	2.21e+4	5.49e+3	NC	NC
1-5					-33.2	
6	1.63e+1	2.93e+0	1.64e+1	3.73e+0	-0.5	0.22
7	6.14e+1	1.33e+1	6.06e+1	1.58e+1	1.4	0.41
8	2.09e+1	5.10e--2	2.09e+1	5.70e--2	0.1	0.52
9	2.7e+1	1.97e+0	2.77e+1	2.47e+0	0.4	0.92
10	1.30e--1	4.78e--2	1.22e--1	4.56e--2	5.7	0.54
11	6.64e+1	1.49e+1	2.75e+1	1.10e+1	**58.5**	**0.00**
12	8.37e+1	1.76e+1	3.03e+1	9.10e+0	**63.8**	**0.00**
13	1.37e+2	2.93e+1	9.99e+1	2.64e+1	**26.9**	**0.00**
14	2.62e+3	3.71e+2	2.19e+3	3.63e+2	**16.1**	**0.00**
15	3.97e+3	6.42e+2	3.45e+3	6.20e+2	**13.1**	**0.01**
16	1.66e+0	3.43e--1	1.60e+0	3.66e--1	3.7	0.71
17	1.03e+2	1.38e+1	6.14e+1	1.04e+1	**40.5**	**0.00**
18	1.67e+2	3.08e+1	1.72e+2	3.35e+1	-2.9	0.29
19	5.61e+0	1.32e+0	6.03e+0	1.57e+0	-7.4	0.43
20	1.18e+1	5.95e--1	1.16e+1	4.86e--1	1.8	0.14
6-20					14.7	
21	2.30e+2	4.66e+1	2.57e+2	5.04e+1	-11.6	0.04
22	2.91e+3	4.71e+2	2.71e+3	5.08e+2	6.9	0.16
23	4.31e+3	7.70e+2	4.22e+3	4.96e+2	2.3	0.66
24	2.78e+2	7.56e+0	2.68e+2	5.29e+0	3.6	0.00
25	2.89e+2	6.44e+0	2.86e+2	7.44e+0	1.3	0.09
26	2.22e+2	5.76e+1	2.25e+2	5.77e+1	-1.4	0.12
27	1.04e+03	6.65e+1	1.00e+3	5.78e+1	3.7	0.02
28	3.00e+2	0.00e+0	2.93e+2	3.65e+1	2.2	0.68
21-28					**0.8**	

Table 10. Comparison of PSO and improved PSO (adaptive topology by DBSCAN cluster detection)

No.	Structure	Diff %	*Man_whitney_test*
7	Globally Convex	1.4	0.41
11	Globally Convex	**58.5**	**0.00**
12	Globally Convex	**63.8**	**0.00**
13	Globally Convex	**26.9**	**0.00**
17	Globally Convex	**40.5**	**0.00**
18	Globally Convex	-2.9	0.29
		31.4	
6	Deceptive	-0.5	0.22
10	Deceptive	5.7	0.54
19	Deceptive	-7.4	0.43
20	Deceptive	1.8	0.14
		-0.1	
9	Partially Structured	0.4	0.92
14	Partially Structured	**16.2**	**0.00**
15	Partially Structured	**13.1**	**0.01**
		9.9	
8	Noisy	0.1	0.52
16	Noisy	3.7	0.71
		1.9	

Table 11. Results of PSO and improved PSO (adaptive topology by DBSCAN cluster detection)based on structure

More detailed information about the comparison of adaptive PSO and PSO on multimodal functions (F6-F20) is presented in Table 11. The overall best result is happening in functions with globally convex structures; the average performance is 31.4%. Some trace of improvement also can be seen in search spaces with partial structures around 9.9%. However, other types of structures like noisy and deceptive indicate no improvements.

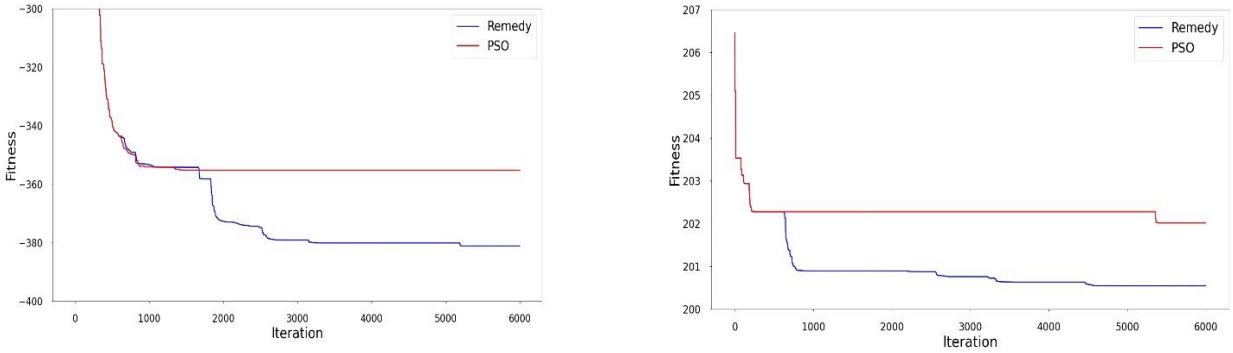
4.4.3 Results of PSO with adaptive topology (by velocity) vs PSO

The second adaptive PSO is also primarily intended for globally-convex, multi-modal search spaces. As it has been shown on Table 12, On the multimodal functions (F6-F20 – which also include multi-modal functions without global convexity), we are able to achieve an average improvement of 34.9% compared to standard PSO. Within this subset, our method achieves meaningful improvements (%-diff greater than 10% and p-value less than 0.05) on 12 of the 15 functions. Our modifications designed for globally convex search spaces perform poorly on F6 (Rotated Rosenbrock's Function) which is deceptive (line searches appear unimodal, but each line finds a different local optimum), but they can perform well on noisy functions like F16 (Rotated Katsuura Function). The short-range exploration within the artificial clusters can be useful for noisy functions with lots of nearby local optima, but it can be less useful on search spaces with large distances between local optima.

The stall remedy can also be useful on unimodal functions that have curved contours. Particle trajectories are aligned to the line segment between *pbest* and *lbest* positions, and this line segment can have non-monotonic fitness even if there is a path between these two points that has a monotonically changing fitness (e.g., F3 (Rotated Bent Cigar Function) and F4 (Rotated Discus Function)). Conversely, our speed-based classification method determined that the swarm had converged on F1 (Sphere Function) and F5 (Different Powers Function), so no remedy was applied, and the results are identical (same initial seeds for each set of trials). The modifications have little effect on the composition functions (F21-F28).

The effect of remedy to address the stall condition in PSO can be further analyzed through the performance plots shown in Figure 23. In these plots, the red and blue lines are identical for the first 600 iterations since the same random seeds are used. After 600 iterations, the performance curves for standard PSO have large plateaus of stall where the fitness does not improve (since all exploratory solutions are rejected). The performance curves for improved PSO have many noticeable movements at the moments

the remedy is applied every 600 iterations. The steady and consistent improvements on F11 (Rastrigin's Function) in Figure 23 (a) imply that the location of new clusters and the perturbations of the positions within all of the clusters is supporting the intended short-range exploration which should be useful in structured/globally convex search spaces.



(a) Globally Convex Function (F11)

(b) Noisy Function (F16)

Figure 23. Performance comparison for standard PSO with and without the stall remedy on (a) Globally Convex Function – F11(Rastrigin's Function) and (b) Noisy Function – F16 (Rotated Katsuura Function)

More detailed information about the comparison of adaptive PSO and PSO on multimodal functions (F6-F20) is presented in Table 13. The overall best result is in functions with globally convex structures; the average performance is 52.1%. Functions with Noisy structures come into second place with 30.2% improvement. After that, search spaces with deceptive and partially structures indicate 22.0% and 20.8% progress. It is further noted that the remedy was specifically designed for globally convex search spaces, and the most consistent results are achieved in this subset followed by the partially structured subset which has locally convex components.

No.	Standard PSO		Adaptive topology		%_diff	*Man_whitney_test*
1	0.00e+0	0.00e+0	0.00e+0	0.00e+0	NC	NC
2	2.18e+6	9.67e+5	2.31e+6	1.00e+6	5.9	0.61
3	7.29e+7	9.87e+7	1.79e+7	1.93e+7	**75.5**	0.00
4	1.64e+4	4.09e+3	1.29e+4	5.50e+3	**21.5**	0.00
5	0.00e+0	0.00e+0	0.00e+0	0.00e+0	NC	NC
1-5					30.3	
6	1.63e+1	2.93e+0	1.89e+1	6.50e+0	-16.4	0.37
7	6.14e+1	1.33e+1	4.40e+1	1.14e+1	**28.3**	**0.00**
8	2.09e+1	5.10e--2	2.09e+1	5.93e--2	0.2	0.00
9	2.7e+1	1.97e+0	2.38e+1	2.67e+0	**14.4**	**0.00**
10	1.30e--1	4.78e--2	7.83e--2	6.49e--2	**39.7**	**0.00**
11	6.64e+1	1.49e+1	2.60e+1	8.14e+1	**60.8**	**0.00**
12	8.37e+1	1.76e+1	3.27e+1	1.12e+1	**60.9**	**0.00**
13	1.37e+2	2.93e+1	7.22e+1	2.81e+1	**47.2**	**0.00**
14	2.62e+3	3.71e+2	1.79e+3	2.97e+2	**31.6**	**0.00**
15	3.97e+3	6.42e+2	3.32e+3	4.84e+2	**16.3**	**0.00**
16	1.66e+0	3.43e--1	6.59e--1	1.29e--1	**60.3**	**0.00**
17	1.03e+2	1.38e+1	4.83e+1	4.89e+0	**53.2**	**0.00**
18	1.67e+2	3.08e+1	6.30e+1	9.02e+0	**62.2**	**0.00**
19	5.61e+0	1.32e+0	2.42e+0	4.38e--1	**56.9**	**0.00**
20	1.18e+1	5.95e--1	1.09e+1	6.93e--1	7.8	0.00
6-20					34.9	
21	2.30e+2	4.66e+1	2.30e+2	4.66e+1	0.0	0.23
22	2.91e+3	4.71e+2	2.07e+3	5.13e+2	**28.7**	**0.00**
23	4.31e+3	7.70e+2	3.76e+3	5.48e+2	**12.8**	**0.00**
24	2.78e+2	7.56e+0	2.67e+2	7.35e+0	4.2	**0.00**
25	2.89e+2	6.44e+0	2.81e+2	7.32e+0	3.0	**0.00**
26	2.22e+2	5.76e+1	2.32e+2	6.42e+1	-4.2	0.46
27	1.04e+03	6.65e+1	9.06e+2	7.24e+1	**12.7**	**0.00**
28	3.00e+2	0.00e+0	3.00e+2	0.00e+0	0.0	0.65
21-28					7.3	

Table 12. Comparison of PSO and improved PSO (adaptive topology by velocity cluster detection)

No.	Structure	Diff %	*Man_whitney_test*
7	Globally Convex	**28.3**	**0.00**
11	Globally Convex	**60.8**	**0.00**
12	Globally Convex	**60.9**	**0.00**
13	Globally Convex	**47.2**	**0.00**
17	Globally Convex	**53.2**	**0.00**
18	Globally Convex	**62.2**	**0.00**
		52.1	
6	Deceptive	-16.4	0.37
10	Deceptive	**39.7**	**0.00**
19	Deceptive	**56.9**	**0.00**
20	Deceptive	7.8	0.00
		22.0	
9	Partially Structured	**14.4**	**0.00**
14	Partially Structured	**31.6**	**0.00**
15	Partially Structured	**16.3**	**0.00**
		20.8	
8	Noisy	0.2	0.00
16	Noisy	**60.3**	**0.00**
		30.2	

Table 13. Results of PSO and improved PSO (adaptive topology by speed method)based on structure

4.5 Discussion

It is important to note the difference between a swarm that has "converged" versus a swarm that has "stalled". In a converged swarm that has had all of its particles reach the same region of the search space, particle movement/speed and diversity can both approach zero. The inability of a converged swarm to perform any exploration at all makes a restart (like) disruption necessary to obtain any value from the remaining function evaluations. Conversely, a stalled swarm merely needs to have its particles (or just their *pbest* positions) escape from their current local optima. The perturbed *pbest* positions can increase the chances that these solutions can survive selection and lead to Successful Exploration.

The stall condition also provides useful insight into a potential mode of failure for PSO in multi-modal search spaces [42]. However, the initial identification of stall depended on precise definitions for attraction basins, exploration, and exploitation. The new speed based cluster detection [10] makes it possible to observe stall outside of simple, artificial search spaces such as Rastrigin. The ability to recognize stall in PSO allows specific remedies to be applied. The offered adaptive topology PSO in this thesis is primarily designed for globally convex search spaces as it targets the centroid and the neighbourhood around the current local optima found by the swarm.

Methods to increase exploration have of course been successful in improving the performance of PSO [11]. It is noted, however, that increasing exploration and increasing the survival rates of highly promising exploratory search solutions are largely orthogonal activities. The perturbation methods introduced in this thesis could be added to any PSO modification that alters particle trajectories (e.g., by changing swarm topologies). The altered search trajectories (proposed by other researchers) can find solutions from new and more promising regions of the search space, and the introduced perturbations in this thesis can improve the odds that these solutions are accepted by selection.

One popular method to help particles to escape from local optima is to change the topology of the swarm (e.g., [45], [49]). A new swarm topology can allow particles to acquire new neighbours/attractors from different regions of the search space. The resulting trajectories can allow exploration of new and different attraction basins. However, implicit in the ability of topological changes (only) to alter particle trajectories is the realization that the swarm has stalled, not converged. Changing the topology of particles which have *pbest* positions near their local optima does nothing to address Failed Exploration which is presented as a key factor in causing a stalled swarm.

5 Contributions

Benchmark function sets are designed to include many search space characteristics – e.g., unimodal, multi-modal, globally convex, ill-conditioned, etc. Different search techniques often display different strengths and weaknesses across these various search spaces. Real-time identification of search space features can lead to adaptive selection and/or behaviour of the metaheuristics used for optimization. One contribution of this thesis is the introduction of line searches as a method to analyze the search space. This tool can help to reveal the global structure of the search space as opposed to just the local structure (e.g., ruggedness [50]). In addition, the tool provides us with information on the approximate size of attraction basins in multi-modal functions, and this has been deployed as part of "PSO with *pbest* perturbations" to increase the rate of Successful Explorations.

Another contribution of this thesis is related to the cluster detection method based on velocity in standard PSO. Before that, it was questioned if the results from Rastrigin were generally applicable and thus if the stall condition was present in PSO for other search spaces. The new speed-based methods to observe stall make it clear that stall occurs in PSO across a broad range of search spaces and that its existence can be easily detected.

The last contribution is related to the noisy type of functions in which PSO cannot perform extensive exploitation. The reason is that the shape of the attraction basins is small in width and tall in height (i.e., needle shaped -- see Figure 8 (c)). Therefore, the forced cluster approach allows groups of particles to do exploitation and thus find fitter locations of attraction basins.

The first contribution has been published in [42], and the last two contributions have been submitted to the selection workshop 2021 IEEE Congress on Evolutionary Computation.

6 Future Research

In Threshold Convergence [51], [52], convergence is "held" back by a threshold function. Since search steps smaller than the threshold are disallowed, a threshold size greater than or equal to the size of attraction basins will eliminate the possibility of local search. The ability to separate the processes of exploration (i.e., global search) and exploitation (i.e., local search) has led to improved performance for metaheuristics such as Particle Swarm Optimization [51] and Differential Evolution [52]. However, these early results were limited by the use of generic threshold functions which produced poor estimates for the size of the attraction basins in the search space. The developed line search technique can be used to provide a good estimate of the size of attraction basins and meta-basins for techniques like Threshold convergence.

Another possibility for future research is to have the number of phases in the search process match the number of passes of a line search to reach a single optimum. With the ability of line searches to identify unimodal, globally convex, and non-globally convex search spaces, metaheuristics can be meaningfully adapted to have one, two, or more phases to match this important global characteristic of a search space.

The line search method described in Section 3.2.2 can identify deceptive search spaces (e.g., F6 (Rotated Rosenbrock's Function) and F10 (Rotated Griewank's Function) in [8]) and the typical size and shape of attraction basins. The addition of speed and cluster information can build upon the insights provided by the line searches. For example, it is easier to distinguish between a noisy search space (e.g., F16 (Rotated Katsuura Function)) and a globally convex search space (e.g., F11 (Rastrigin's Function)) from the speed and cluster data than from the line search data. The lack of clusters is specifically insightful to identify a search space in which neighbouring particles often fly over the deep and narrow attraction basins found in a noisy search space. This information could be used to develop a specific remedy for this class of search spaces. The current remedy is primarily designed for globally convex search spaces in that

it looks for new attraction basins close to the existing attraction basins. The forced clusters and perturbations also help on noisy problems, but there are surely more specific remedies that could be more effective. The current remedy can also be ineffective on deceptive problems (e.g., F6 (Rotated Rosenbrock's Function)). The development of specific remedies for more classes and characteristics of search spaces is a promising direction for further research.

7 Bibliography

[1] A. Bolufé-Röhler, S. Chen and D. Tamayo-Vera, "An analysis of minimum population search on large scale global optimization," in *IEEE Congress on Evolutionary Computation (CEC)*, 2019, June.

[2] E.-G. Talbi, Metaheuristics: from design to implementation, John Wiley & Sons, 2009.

[3] "wikipedia," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Metaheuristic.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, 1995.

[5] S. Rainer and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization,* vol. 11(4), pp. 341-359, 1997.

[6] R. Hooke and T. A. Jeeves, "Direct Search"Solution of Numerical and Statistical Problems," *Journal of the ACM (JACM),* pp. 8, no. 2 (1961): 212-229, 1961.

[7] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal,* pp. 7, no. 4 : 308-313, 1965.

[8] A. Bolufé-Röhler, S. Chen and . D. Tamayo-Vera, "An analysis of minimum population search on large scale global optimization," in *IEEE Congress on Evolutionary Computation (CEC),* 2019, June.

[9] F. Van Den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Information sciences,* pp. vol. 176, no. 8, pp. 937–971, 2006.

[10] S. Chen, B.-R. Antonio , J. Montgomery and T. Hendtlass, "An analysis on the effect of selection on exploration in particle swarm optimization and differential evolution.," in *IEEE Congress on Evolutionary Computation (CEC)*, 2019.

[11] M. R. Bonyadi and M. Zbigniew , "Particle swarm optimization for single objective continuous space problems: a review.," *MIT Press,* pp. 25(1): 1-54, 2017.

[12] I. BoussaïD, J. Lepagnot and . P. Siarry, "A survey on optimization metaheuristics," *Information sciences 237,* pp. 82-117, 2013.

[13] V. Laarhoven, P. JM and E. HL Aarts, "Simulated annealing," *Statistical science,* pp. 8(1), 10-15, 1993.

[14] B. Christian, A. Roli and A. Enrique, An introduction to metaheuristic techniques, Parallel Metaheuristics: A New Class of Algorithms, 2005.

[15] I. Rechenberg, "Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution," in *frommann–holzboog*, 1973.

[16] L. J. Fogel, . A. J. Owens and M. J. Walsh., "Artificial intelligence through simulated evolution.," 1966.

[17] J. Koza, "Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)," *The MIT Press,* vol. first ed, 1992.

[18] M. Črepinšek, . S.-H. Liu and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM computing surveys (CSUR)*, pp. 45(3), 1-33., 2013.

[19] H. Abid and Y. Shad Muhammad, "Trade-off between exploration and exploitation with genetic algorithm using a novel selection operator," *Complex & Intelligent Systems*, pp. 1-14, 2019.

[20] J. Montgomery, S. Chen and Y. Gonzalez-Fernandez, "Identifying and exploiting the scale of a search space in differential evolution," in *Congress on Evolutionary Computation (CEC)*, 2014.

[21] [Online].Available: https://www.google.com/imgres?imgurl=https://ww2.mathworks.cn/matlabcentral/mlc-downloads/downloads/submissions/57286/versions/8/screenshot.jpg&imgrefurl=https://ww2.mathworks.cn/matlabcentral/fileexchange/57286-video-tutorial-of-particle-swarm-optimizati.

[22] F. Marini and B. Walczak, "Particle swarm optimization (PSO)," *Chemometrics and Intelligent Laboratory Systems*, pp. 149, 153-165., 2015.

[23] [Online]. Available: https://web2.qatar.cmu.edu/~gdicaro/15382/.

[24] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *International conference on evolutionary programming*, Berlin, Heidelberg, 1998.

[25] B. Phlippie and . P. E. Andries, "Diversity rate of change measurement for particle swarm optimisers," in *In International Conference on Swarm Intelligence*, Cham, 2014, September.

[26] J. J. Liang, B. Y. Qu, P. N. Suganthan and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization.," *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report*, vol. 201212(34), pp. 281-295, 2013.

[27] Y. Gonzalez-Fernandez and S. Chen, "Leaders and followers—a new metaheuristic to avoid the bias of accumulated information," in *IEEE Congress on Evolutionary Computation (CEC)*, 2015.

[28] M. El-Abd, "Black-box optimization benchmarking for noiseless function testbed using artificial bee colony algorithm," in *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, 2010.

[29] M. Davarynejad, J. van den Berg and J. Rezaei, "Evaluating center-seeking and initialization bias: The case of particle swarm and gravitational search algorithms," *Information Sciences*, pp. 802-821, 2014.

[30] K. M. Malan and A. P. Engelbrecht, "Quantifying ruggedness of continuous landscapes using entropy.," in *IEEE Congress on evolutionary computation*, 2009.

[31] W. Beaudoin, S. Verel, P. Collard and C. Escazut, "Deceptiveness and neutrality the nd family of fitness landscapes," in *In Proceedings of the 8th annual conference on Genetic and evolutionary computation*, 2006.

[32] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *IEEE swarm intelligence symposium*, 2007.

[33] A. Engelbrecht, "Particle swarm optimization: Velocity initialization," in *IEEE congress on evolutionary computation*, 2012.

[34] S. Helwig, J. Branke and S. Mostaghim, "Experimental analysis of bound handling techniques in particle swarm optimization," *IEEE Transactions on Evolutionary computation,* pp. 17(2), 259-271, 2012.

[35] M. R. Bonyadi and Z. Michalewicz, "Analysis of stability, local convergence, and transformation sensitivity of a variant of the particle swarm optimization algorithm," *IEEE Transactions on Evolutionary Computation,* pp. 20(3), 370-385, 2015.

[36] J. Kennedy, " Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in *Proceedings of the 1999 congress on evolutionary computation-CEC99, IEEE*, 1999, July.

[37] Z. Chenggong and Z. Yi, "Scale-free fully informed particle swarm optimization algorithm," in *Information Sciences, 181(20), 4550-4568.*, 2011.

[38] J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions.," *IEEE transactions on evolutionary computation,* vol. 10(3), pp. 281-295, 2006.

[39] W. Hui, H. Sun, C. Li, S. Rahnamayan and J.-S. Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Informatic Science,* vol. 223, pp. 119-135, 2013.

[40] D. Chen and C. Zhao, "Particle swarm optimization with adaptive population size and its application," *Applied Soft Computing,* vol. 9(1), pp. 39-48, 2009.

[41] M. A. M. De Oca, T. Stutzle, K. Van den Enden and M. Dorigo, "Incremental social learning in particle swarms," *EEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 41(2), pp. 368-384, 2010.

[42] S. Chen, I. Abdulselam, N. Yadollahpour and Y. Gonzalez-Fernandez, "Particle Swarm optimization with pbest Perturbations," in *IEEE Congress on Evolutionary Computation (CEC)*, 2020, July.

[43] E. Schubert, J. Sander, M. Ester, H. P. Kriegel and X. Xu, "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN," *ACM Transactions on Database Systems (TODS),* pp. 42(3), 1-21, 2017.

[44] T. Hendtlass, "a multi-optima particle swarm algorithm," in *IEEE Congress on Evolutionary Computation* , 2005.

[45] J.-J. Liang and P. N. Suganthan, "Dynamic multi-swarm particle swarm," in *IEEE Swarm Intelligence Symposium*, 2005.

[46] M. A. M. De Oca, J. Pena, T. Stutzle, C. Pinciroli and M. Dorigo, "Heterogeneous particle swarm optimizers.," in *IEEE congress on evolutionary computation*, 2009.

[47] [Online]. Available: https://en.wikipedia.org/wiki/Mann%E2%80%93Whitney_U_test.

[48] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation,* vol. 1(1), pp. 67-82, 1997.

[49] Y.-X. Wang, and Q.-L. Xiang, "Particle swarms with dynamic ring topology.," in *IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence),* 2008.

[50] K. M. Malan and A. P. Engelbrecht, "Fitness landscape analysis for metaheuristic performance prediction," *Recent advances in the theory and application of fitness landscapes,* pp. 103-132, 2014.

[51] S. Chen and J. Montgomery, "Particle swarm optimization with thresheld convergence.," in *IEEE congress on evolutionary computation*, 2013.

[52] A. Piad-Morffis, S. Estévez-Velarde, A. Bolufé-Röhler, J. Montgomery and S. Chen, "Evolution strategies with thresheld convergence," in *IEEE congress on evolutionary computation (CEC)*, 2015, May.

[53] B. Leonora, M. Dorigo, L. M. Gambardella and . W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, pp. 8(2), 239-287, 2009.

[54] D. Tamayo-Vera, S. Chen, A. Bolufé-Röhler, J. Montgomery and T. Hendtlass, "Improved exploration and exploitation in particle swarm optimization," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 2018, June.

[55] F. Glover, M. Laguna and R. Martí, "Fundamentals of scatter search and path relinking," *Control and cybernetics*, pp. 29(3), 653-684, 2000.

[56] A. Modiri and K. Kiasaleh, "Modification of Real-Number and Binary PSO Algorithms for Accelerated Convergence," *IEEE Transactions on Antennas*, pp. 214-224, 2010.

[57] A. Bolufé-Röhler and S. Chen, "Minimum population search-lessons from building a heuristic technique with two population members.," in *IEEE Congress on Evolutionary Computation* , 2013.

[58] J. Montgomery and S. Chen, "A Simple Strategy for Maintaining Diversity and Reducing Crowding in Differential Evolution," in *IEEE World Congress on Computational Intelligence*, 2012.

[59] S. Chen and J. Montgomery, "A Simple Strategy to Maintain Diversity and Reduce Crowding in Particle Swarm Optimization," in *In Australasian Joint Conference on Artificial Intelligence*, 2011.

[60] "https://www.researchgate.net/publication/259643342_source_code,"June,2017 [Online].Available: https://www.researchgate.net/publication/259643342_source_code_for_an_implementation_of_standard_particle_swarm_optimization –revised?ev=prf_pup,".