PDPSO: THE FUSION OF PRIMAL-DUAL INTERIOR POINT METHOD AND PARTICLE SWARM OPTIMIZATION ALGORITHM

***Emmanuel Gbenga Dada*[1] *and Effirul Ikhwan Ramlan*[2]**

[1]Department of Computer Engineering, Faculty of Engineering, University of Maiduguri, P.M.B 1069, Maiduguri, Borno State, Nigeria.
[2]Natural Computing Laboratory, Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, 50603, Kuala Lumpur, Malaysia

Email: gbengadada@siswa.um.edu.my[1], effirul@um.edu.my[2]

DOI: https://doi.org/10.22452/mjcs.vol31no1.2

ABSTRACT

Particle Swarm Optimization (PSO) is a meta-heuristic algorithm that has been used to solve a variety of complex optimization problems. In spite of the acceptance of the algorithm in various fields, PSO still suffers from common issues such as premature convergence and local minima. This provides a platform for generating a variety of PSO variants. Although these variants are successful in addressing issues specific to a directed domain, they are still unable to resolve the issues effectively. The Interior-Point Methods (IPMs) are efficient tools for solving nonlinear optimization problems. On the one hand, the method is depicted as the most robust algorithm for solving large scale nonlinear optimization problems. On the other, similar to PSO, the methods are still plagued with several issues. We propose Primal-Dual Interior Point Particle Swarm Optimization (pdPSO) to resolve the shortcomings of a standard PSO without the limitations of the IPM methods. We applied the Primal Dual procedure to each particle in a finite number of iterations, and fed the PSO with the its output. We compared the performance of our new algorithm (pdPSO) with IPM and PSO using nine different dynamic benchmark functions. Our results revealed that pdPSO performed better than both the independent PSO algorithm and the IPM method. The proposed algorithm is not susceptible to premature convergence, and can better avoid local minima than conventional PSO, hence hypothetically it has the potential to perform better than many variants of PSO.

Keywords: ***Particle Swarm Optimization (PSO), Interior Point Method, Primal-Dual, Unimodal functions, Multimodal functions***

1.0 INTRODUCTION

PSO is a stochastic population based algorithm [1] that operates on the optimization of a candidate solution (or particle) aimed at optimizing a directed performance measure [2]. The first PSO algorithm (proposed by Kennedy and Eberhart [3]) was based on the social behaviours exemplified by a flock of birds, a school of fish, and herds of animals. The algorithm uses a set of candidates called particles that undergo gradual changes through collaboration and contest among the particles from one generation to the other. PSO have been used to solve non-differentiable [4], non-linear [4, 5] and non-convex engineering problems [6]. Several state of the art meta-heuristic algorithms are available (e.g., Genetic Algorithm [7], Flower Pollination Algorithms [8], Cuckoo Search Algorithm [9, 10], Artificial Bee Colony Algorithm [11, 12], Fireworks Algorithm [13], Ringed Sealed Search Optimization Algorithm [14], Chicken Swarm Optimization [15, 16], Bat Algorithm [17], and Firefly Algorithm [18]), however, PSO is the focus of our work as it is theoretically straightforward and with limited computational requirements [19]. PSO uses only a few parameters, which have minimal influence on the results compared to the other optimization algorithms. This negates the variance of results generated by the algorithm. This property also applies to the initial generation of the algorithm. The randomness of the initial generation will not be reflected in the output produced. Despite these advantages, PSO faces shortcomings similar to the other optimization algorithms. Specifically, the PSO algorithm suffers from premature convergence, the inability to solve dynamic optimization problems, the tendency of particles to be trapped in local minima and partial optimism (which degrades the regulation of its speed and direction).

Several attempts have been made to increase the efficiency of PSO algorithms. Liang et al [20] proposed the comprehensive learning strategy PSO (CLPSO) with the aim of achieving better performance in comparison to the present PSO modifications for multimodal functions. Zhao [21] presented the Perturbed Particle Swarm

Optimisation for numerical optimisation algorithm (pPSA). The algorithm uses a scheme for overcoming premature convergence by engaging a particle updating method that focuses on the inspiration of disconcerted global best particles. Akbari and Ziarati [22] developed the rank based particle swarm optimisation algorithm with dynamic adaptation (PSO_{rank}). The algorithm utilizes the cooperative behaviour of particles to achieve a significant boost in efficiency compared to a conventional PSO algorithm. Zhan et al. [23] proposed orthogonal learning PSO (OLPSO-G). This algorithm uses a perpendicular learning approach to establish a conducive and efficient model to direct particles to move in the most appropriate path. Huang et al. [24] developed Example-based learning PSO (ELPSO) for continuous optimisation. Their aim is to use an example-based learning scheme to proffer a superior performance for multimodal functions. Adaptive parameter tuning of PSO centred on velocity information (APSO-VI) was proposed by Xu in [25]. Diversity enhanced PSO with neighbourhood (DNSPSO) was presented in [26]. This algorithm employs the diversity improvement method and neighbourhood search approaches to achieve an optimum point between exploration and exploitation. Multi-objective sorting-based learning PSO for continuous optimisation (MLPSO) proposed in [27] uses the MSL approach to direct particles to move in the most suitable path by creating a direction paradigm that has superior fitness value and variety in swarm populations. In spite of all the efforts, the presence of premature convergence, the inability of particles to escape from being trapped in global optima, and the incompatibility of PSO to handle dynamic tasks are still evident in different variants of PSO algorithms that have been proposed.

Recently, the state-of-the-art Interior-Point algorithm has gained popularity as the most preferred approach for providing solutions to large-scale linear programming problems [28]. They are however limited due to their inability to solve problems that are unstable in nature. This is because contemporary Interior-Point algorithms are unable to cope with the ever-increasing need of the constraints. Efforts to improve the efficiency of the Interior-Point algorithm have led to the development of more variants of this algorithm that can handle unstable linear programming problems (similar steps were observed in the meta-heuristic field). These algorithms lower the number of work per iteration by using a small number of constraints; hence drastically reducing the computational processing time [29]. Over the years, some progress has been made in overcoming the challenges associated with interior point algorithms. The Inner-iteration Krylov subspace methods for least square problems was proposed in [30]. The GMRES method for rank deficient least square problems were used for convergence of the inner-iteration in [31] and an interior-point method for LP based on Krylov Subspace Iterative Solvers with Inner-Iteration Preconditioning was proposed in [32]. A new direction in polynomial time interior-point methods for monotone linear complementarity problems was proposed in [33]. The major shortcoming among the different approaches mentioned above is that the variants of IPM proposed are still plagued with the problem of solving the derivatives system of linear equations as the iterations progress (which is known as ill-conditioning) to the end of the interior-point. There is no iterative solver that has succeeded in resolving this ill-conditioned problem.

In this study, we proposed a fusion of conventional PSO with the Primal-Dual Interior-Point method to resolve those common issues relevant to the PSO algorithm. Two key components of this implementation are the explorative capacity of PSO, and the exploitative capability of the Primal-Dual Interior-Point algorithm. On one hand, exploration is key in searching (i.e., traversing the search landscape) to provide reliable approximation values of the global optimal [34]. On the other, exploitation is critical to focus the search on ideal solutions resulting from exploration to produce more refined results [35]. The speed at which an algorithm attains the global optimum is a very important parameter for assessing the performance of the algorithm. Since the Primal Dual method is a robust optimisation algorithm, it is expected that *pdPSO* will produce superior results in comparison to the traditional PSO algorithm, the Primal Dual algorithm alone, and the other state of the art PSO algorithms.

2.0 PARTICLE SWARM OPTIMIZATION (PSO) ALGORITHM

In PSO, (X_i) represents the position of a particle, and (V_i) is the velocity of the particle. The particle's number is i, where ($i = 1,...,N$), and N is the number of particles in the swarm. The i^{th} particle is represented as $X_i = (X_{i1}, X_{i2},..., X_{iN})$. While the velocity is the rate at which the next position is changing with respect to the current position. Variable $V_i = (V_{i1}, V_{i2},..., V_{iN})$ represents the velocity for the particle i. At the start of the algorithm, initial numerical values of the position and velocity of the particles are assigned randomly. Equations (1) and (2) will then update the position and velocity of the particles for subsequent iterations during the search process.

$$v_{i,m}^{(t+1)} = w * v_{i,m}^{(t)} + c_1 * rand_1() * (pbest_{i,m} - x_{i,m}^{(t)} + c_2 * rand_2() * (gbest_m - x_{i,m}^{(t)}) \quad (1)$$

$$x_{i,m}^{(t+1)} = x_{i,m}^{(t)} + v_{i,m}^{(t+1)} \quad (2)$$

According to Shi and Eberhart [36], to avert eruption, the value $v_{i,m}^{(t+1)}$ is fixed at ± v_{max} . This is because the value of $v_{\max}$ will be too large if the search range is very wide. If $v_{\max}$ is too small, the scope of the search will be excessively limited, thereby forcing the particles to support local exploration. "*w*" is the weight of inertia (constriction factor); this regulates the algorithm search properties. Shi and Eberhart [36] suggested a large inertia value (a more global search) initiation that is dynamically reduced towards the end of the optimization (a more local search). The use of small inertia weights usually guarantees quick convergence as the time spent on the exploration of the global space is reduced [37].

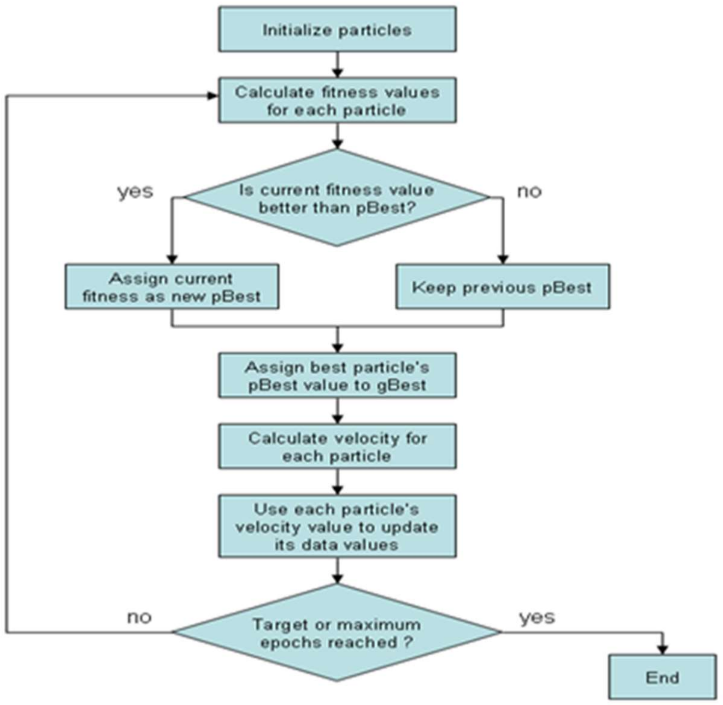
Initialize particles
Calculate fitness values for each particle
yes
Is current fitness value better than pBest?
no
Assign current fitness as new pBest
Keep previous pBest
Assign best particle's pBest value to gBest
Calculate velocity for each particle
Use each particle's velocity value to update its data values
no
Target or maximum epochs reached ?
yes
End

Fig 1: The flowchart of a conventional PSO Algorithm (Kennedy and Eberhart [3]).

The inclusion of *w* in the equation is to provide an equilibrium between the global and local search capabilities of the particles. The positive constant c_1 and c_2 represent the cognitive scaling and social scaling factors which are set to the value of 2 in [38]. Both the cognitive and social scaling factors assist PSO in successfully building the local bests into the global best [39]. The stochastic variables $rand_1()$ and $rand_2()$ have a uniform distribution. These random variables are independent functions that provide energy to the particles. The best position found so far by the particle is represented as $pbest_{i,m}$. The best position attained by the neighbouring particles is represented as $gbest_m$. There are two types of particle neighbourhoods in PSO, and the type of neighbourhood determines the value of $gbest_m$. The two types of neighbourhoods are:

1. *gBest* (Global neighbourhood) – There is a full connection among the particles, and the exploration of swarm is controlled by the best particle in the swarm.

2. *lBest* (Local neighbourhood) – There is no full connection among the particles in the swarm, rather they are connected only to their neighbours.

Equation (2) is used in updating the position of the particles whereby the velocity is added together with the earlier position and a new search is started from its former position. Eberhart and Shi [38] bounded $x_{i,m}^{(t+1)}$ to avoid a situation whereby particles are spending too much time in infeasible region. A problem dependent fitness function is used to evaluate the superiority of $x_{i,m}^{(t+1)}$. Assuming the present solution is superior to the fitness of $pbest_{i,m}$ or $gbest_m$ then the new position will replace $pbest_{i,m}$ or $gbest_m$ accordingly. Unless the condition for ending the search is met (either the iteration has reached its peak or we have obtained the desired solution), the update process will continue. The optimal solution is the best particle found when the stopping criterion is satisfied [37]. The flowchart for the original PSO algorithm for collective robot search is shown in fig. 1.

3.0 PRIMAL DUAL INTERIOR POINT METHOD

The primal-dual interior-point (PDIP) method is an excellent example of an algorithm that uses constraint-reduction methods. Mehrotra [40] developed the Mehrotra's Predictor-Corrector PDIP algorithm, which has been executed in the majority of the interior-point software suites for solving both linear and convex-conic problems [41]. The primal-dual methods are a new category of interior-point methods that have been practically employed for solving large-scale nonlinear optimization problems recently [42]. Contrary to the traditional primal method, the primal-dual method evaluates both the primal variables x and the dual Lagrange multipliers λ related to the constraints concurrently. The disconcerted Karush-Kuhn-Tucker (KKT) equations below can be solved using the precise primal-dual solution (x^*_μ,λ^*_μ) at a given parameter μ

$$\begin{cases} \nabla F(x) - C^T\lambda = 0 \\ \lambda_i C_i(x) = \mu, i = 1, \ldots, m \end{cases} \text{with the constraint } (C(x), \lambda) \geq 0. \quad (3)$$

The Newton's algorithm and the line search approach are employed to recursively solve any primal or primal-dual sub-problem for a given μ value [43]. Feasibility and convergence is enforced in the algorithm by selecting the size of each step in the iteration. This can be achieved by appropriately reducing the merit function used in gauging the degree of advancement to the solution. The dual variables of primal dual can be protected by using the $F\mu$ as a function that can incorporate the primal and dual variables [44]. At the same time, it measures the harmony between the data and the fitting model for a particular choice [45]. The major setback of the barrier functions is the ineffectiveness of traditional line exploration methods thereby necessitating the development of more efficient line searches [46].

According to [46], the primal-dual method can efficiently handle large linear programming problems (the bigger the problem size, the more noticeable the efficiency of the primal-dual algorithm). The algorithm is not susceptible to degradation and the number of iterations does not depend on the number of vertices in the feasible search space [47]. The primal-dual algorithm uses considerably less iterations compared to the simplex method and the algorithm is able to generate ideal solutions for a linear programming problem in less than 100 iterations irrespective of the huge number of variables involved in nearly all its implementations [47]. However, the primal-dual method is hindered by its inability to detect the possibility of a problem having an unbounded status (to a certain extent, the method is labelled as incomplete). This issue has been addressed sufficiently using undiversified model as suggested in [48, 44]. In addition, the computational cost for each iteration in primal-dual is higher than that of the simplex algorithm. In spite of this issue, the primal-dual method is able to handle a large linear programming problem better than the simplex algorithm. This is because the total work required in providing solution to a large linear programming problem is comprised of the multiplication of the number of iterations with the work executed for each iteration. The primal-dual steps have the inclination of producing inferior steps that defile the boundaries $s > 0$ and $z > 0$ extensively, causing the progress to dwindle. The Interior Point Method algorithm below was adapted from Paul, Jean, and Sophie [44]. The algorithm is depicted in Fig. 2.

```
Step 1: Initialize iteration count, particles with randomly chosen positions, and velocities within
          the limits of the search space.
Step 2: Initialize the number of primal variables (n), the number of constraints (m), and the total
          number of primal-dual optimization variables (nv). Initialize the Lagrange multipliers.
          Initialize second-order information.
Step 3: Primal Dual phase activated
Step 4: Reset the random number generator
Step 5: Initialize some of the algorithm parameters (such as the maximum centering parameter,
          maximum forcing number, minimum barrier parameter, maximum step size, minimum
          step size, granularity of backtracking search, and amount of actual decrease we will
          accept in line search).
Step 6: Compute the responses of the unperturbed Karush-Kuhn-Tucker optimality conditions
Step 7: Check for convergence
Step 8: Update the BFGS approximation to the Hessian of the objective.
Step 9: Find Solution to perturbed KKT system.
Step 10: Perform backtracking line search.
Step 11: Compute the response of the merit function and the directional gradient at the current
          point and search direction.
Step 12: Compute the candidate point, the constraints, and the response of the objective function
          and merit function at the candidate point.
```

Fig. 2: Pseudo code of the Primal Dual algorithm.

4.0 PRIMAL DUAL INTERIOR POINT PARTICLE SWARM OPTIMIZATION (*PDPSO*)

We propose a new algorithm called Primal-Dual-PSO algorithm (*pdPSO*) which combines the best features of both conventional PSO and the primal-dual method. The new algorithm works by first initiating the particles' positions randomly. We then pass the particles to the Primal-Dual method, which gives us its initial optimization result after a number of iterations. The result of the Primal-Dual optimization is then inserted into the PSO, which creates a perturbation in the population and also maintain diversity in the population until there is either convergence to the global optimal or the termination criteria is reached. PSO was fused into the primal dual algorithm in order to combine the strengths of the two approaches and compensate for the drawbacks of each of them. The hybrid algorithm integrating the Primal Dual Interior Point Method and PSO is depicted in Fig. 3.

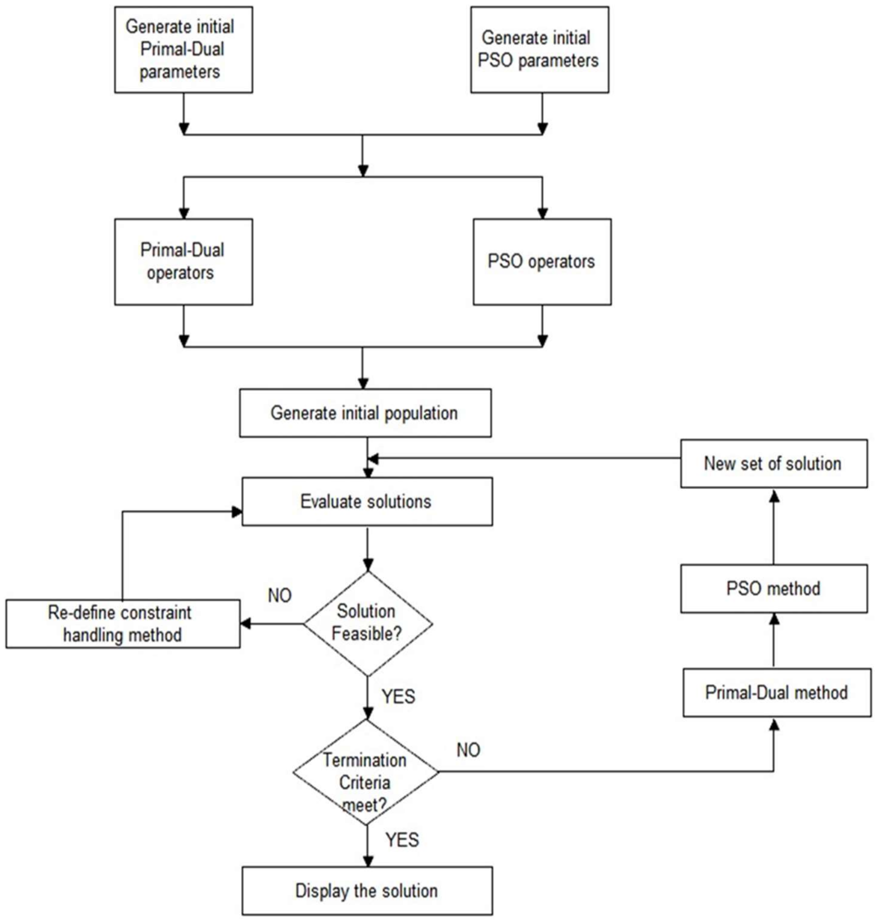
Generate initial Primal-Dual parameters
Generate initial PSO parameters
Primal-Dual operators
PSO operators
Generate initial population
New set of solution
Evaluate solutions
PSO method
NO
Re-define constraint handling method
Solution Feasible?
YES
Primal-Dual method
Termination Criteria meet?
NO
YES
Display the solution

Fig. 3: Primal- Dual-PSO (*pdPSO*) algorithm.

5.0 RESULTS AND DISCUSSION

It is evident from our previous studies, that due to the shortcomings of the conventional PSO and the need of applying the algorithm to address multi-faceted domains, there is a need to develop (or customize) various versions of PSO algorithms (custom PSO variants). This has contributed towards the ever-expanding pool of PSO algorithms. The algorithm has some fundamental problems to function as a modular multi-faceted problem solver. This problem is prominent when handling dynamic optimizations. The problem is fundamental at its core, and we hypothesize that by introducing a hybrid algorithm (with the integration of the primal-dual method), we would be able to address the fundamental issues of conventional PSO, thus transforming the algorithm into a modular multi-faceted problem solver practical to any domain or problem. This section provides a comparative analysis of the performance of *pdPSO* against a multitude of optimization problems and benchmarking data from state of the art algorithms.

5.1 Parameter settings

A dimension value of 10 is assigned for each function (i.e. n = 10). For each of the PSO mentioned above, a swarm of 30 particles was generated with a global best topology. We carried out 400 iterations for each of the algorithms we tested using 9 benchmark functions running on MATLAB R2012a. The cognitive scaling c_1 which influences local search is set to the value of 2. Similarly, social scaling, c_2, which influences the global search, is identically set to the value of 2. Functions $rand_1$ and $rand_2$ are stochastic variables that have the uniform distribution U (0, 1). They are independent functions that provide energy to the particles. To avert eruption during the particles' exploration of the search space, the value of the velocity is fixed at $\pm V_{max}$ and the value of V_{max} is set to be equal to the value of X_{max}. This helps in controlling the search range. The range of the searches will become wide if the value assigned is large, thus limiting the algorithm to only global exploration. In contrast, if the value of V_{max} is small the scope of the search will be excessively limited thereby forcing the particles to support only local exploration. The inertia weight *w* called (constriction factor) is the inertia parameter; this regulates the algorithm's searching properties. The initial value is 0.9 and this value decreases to a final value of 0.4. We started with a larger inertia value (a more global search) that is dynamically reduced towards the end of the optimization (a more local search). A small inertia weight guarantees quick convergence of the algorithms due to the reduction of time for exploration in the global space. The inertia weight *w* is used to provide equilibrium between the global and local search capabilities of the particles in the swarm.

5.2 Benchmark problems

For the purpose of our experimental study, nine benchmark functions were selected. They can be classified as either Unimodal or Multimodal (both can either be static or dynamic) for the experiment. The selected functions are Sphere [49], Tripod [50], NDParabola [50], Griewank [51, 52], Rastrigin [51], Ackley [53], Shaffer f6 [54], Shaffer f6 modified [55], and f6 Bubble Dynamic [55].

The first function is Ackley (results are depicted in Fig. 4 and Table 1). It is a multi-modal function with deep local minima. It has several local minima. It is commonly used to test the ability of the optimization algorithm to escape local minima. It also used to test the presence of premature convergence in the algorithm. Based on the simulation results for IPM, PSO, and *pdPSO*, there are many local minima generated by the function for PSO and *pdPSO*. PSO and *pdPSO* converged to global optimum, while IPM got trapped in local minima. The convergence rates of PSO and *pdPSO* are almost identical. For the *pdPSO* algorithm, the first 200 iterations were handled by the IPM, while the last 200 iterations were executed by the PSO (thereby combining the exploitative power of IPM and the explorative ability of the PSO). We observed a sharp drop in the *gbest* of *pdPSO* from the first to the 50th iteration, and the convergence rate is doubled when the output of the IPM is inserted into the PSO algorithm. In our comparisons, we used the values of the best fitness, mean fitness, and standard deviation because they are some of the performance measures mentioned in [56]. When we compared the performance of the three algorithms in terms of the numerical values of best fitness, mean fitness, and standard deviation, we can deduce that the performance of PSO is better than the other two algorithms for the Ackley function.

The second function is the Sphere function which is a simple unimodal function with no communication between the variables. Optimization algorithms would commonly be able to solve the function efficiently. The function can also be used to test the presence of premature convergence in optimization algorithms. From the result of our simulation (depicted in Fig. 5 and Table 2), *pdPSO* and IPM converged faster than PSO. Within 250 iterations, both PSO and *pdPSO* converged successfully. PSO underwent more iterations before convergence because it seems to be trapped at a local minimum. This suggests the superiority of the *pdPSO* (in terms of convergence speed) compared to the PSO algorithm. We compared the performance of the three algorithms based on the values of best fitness, mean fitness and standard deviation. From the numerical results, the performance of *pdPSO* was better in terms of the best fitness, mean fitness and standard deviation.

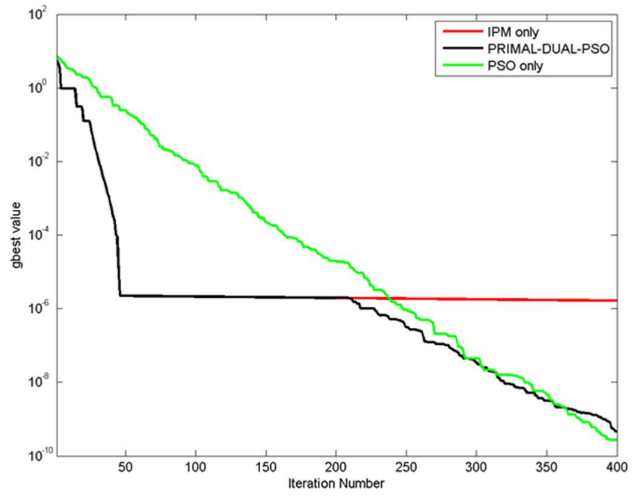

Fig 4: Graph of Ackley function for Primal-Dual, PSO and *pdPSO*

Table 1: Result Comparison for Ackley Function

Algorithm	**Best Fitness**	**Worst Fitness**	**Mean Fitness**	**Std. Deviation**
Primal-Dual-PSO	5.79464e-10	2.28608e-05	+9.00249e-07	4.16412e-06
Primal-Dual	1.64406e-06	3.57445e+00	+1.44513e+00	1.35517e+00
PSO	2.72194e-10	2.20607e-08	+1.94965e-09	4.12310e-09

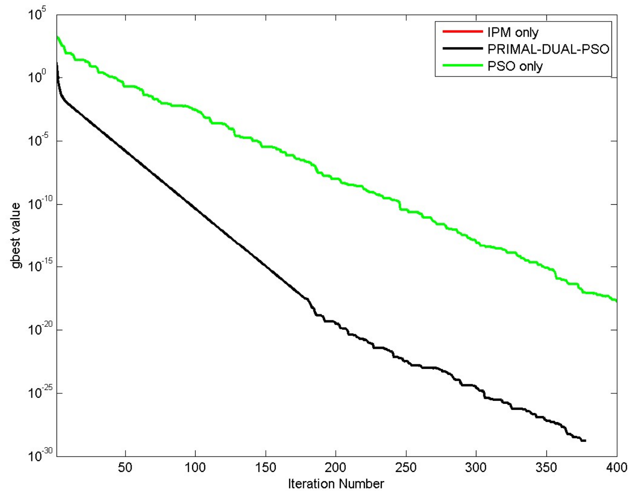

Fig 5: Graph of Sphere function for Primal-Dual, PSO and *pdPSO*

Table 2: Result Comparison for Sphere Function

Algorithm	**Best Fitness**	**Worst Fitness**	**Mean Fitness**	**Std. Deviation**
Primal-Dual-PSO	1.87349e-29	9.11685e-27	+1.19780e-27	2.20741e-27
Primal-Dual	3.05475e-18	1.00235e-17	+6.33670e-18	1.71551e-18
PSO	1.80186e-18	2.76436e-14	+1.05297e-15	5.02818e-15

The third function is the Griewank function. It is a non-linear multimodal function. It is highly multimodal because of the cosine modulation that produces many widespread local minima. The Griewank function has characteristics that are relatively similar to that of the Rastrigin function except for the number of local optima, which is more frequent in the Griewank function. The numerous local minima have complex structure, and only multi-initialization optimization algorithms can converge to the global minimum (which increases based on the dimension of the problem). The function tests the ability of the optimization algorithm to escape local minima. In Fig. 6 and Table 3, we have the simulation results of the Griewank function for IPM, PSO, and *pdPSO*. PSO encounters many local optima for this function. However, all the algorithms successfully converged to the global optimum. The performance of IPM and *pdPSO* for Griewank function is far better than that of PSO based on their speed of convergence. In terms of the numerical value of best fitness, the *pdPSO* is superior to the other two algorithms. However, the performance of primal-dual is better in terms of mean fitness and standard deviation. *pdPSO* however has a better convergence rate, thereby giving it an edge over Primal-Dual and PSO.

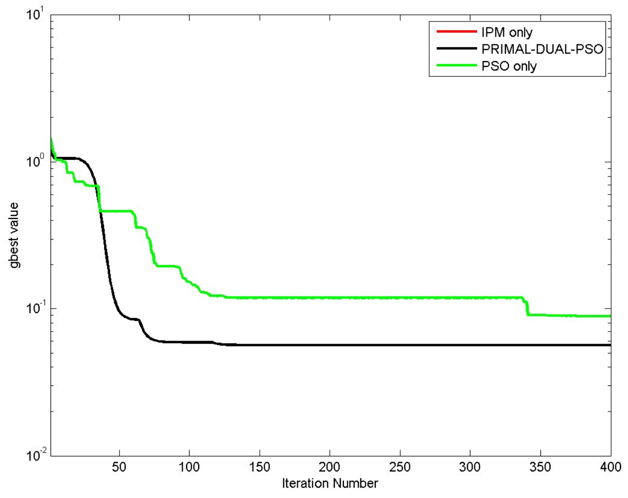

Fig 6: Graph of Griewank function for Primal-Dual, PSO and *pdPSO*

Table 3: Result Comparison for Griewank Function

Algorithm	**Best Fitness**	**Worst Fitness**	**Mean Fitness**	**Std. Deviation**
Primal-Dual-PSO	5.65695e-02	3.45688e+00	+8.22243e-01	7.07663e-01
Primal-Dual	5.65695e-02	1.08272e+00	+2.73931e-01	3.61745e-01
PSO	8.86239e-02	1.91516e+00	+4.59212e-01	5.75298e-01

The fourth function is the Schaffer f6 Function. It is a complex multimodal function. Most hill-climbing and reactive search methods find it very difficult due to its circular local maxima. It is considered a Genetic Algorithm which is a hard function to optimize. It is used to test the ability of the optimization algorithm to escape local minima as well as premature convergence. The challenge that this function poses to optimization algorithms is the rise in the magnitude of prospective maxima which must be overcome to get to a minimum as one moves nearer to the global minimum. The simulation results for the Schaffer f6 function with IPM, PSO, and *pdPSO* is depicted in Fig. 7 and Table 4. Apparently, this function poses a challenge to both PSO and *pdPSO* (presence of several local minima). There was a sharp and steady fall in the value of gbest of IPM and *pdPSO* up till the 150th iteration. *pdPSO* seems to be trapped in a local minimum from the 150th to the 275th iteration after which it experienced a fall in the value of its gbest again. PSO however converged faster under this function when compared to IPM and *pdPSO*. While *pdPSO* and PSO have better performance in terms of the numerical value of best fitness, PSO is better in terms of mean fitness and standard deviation.

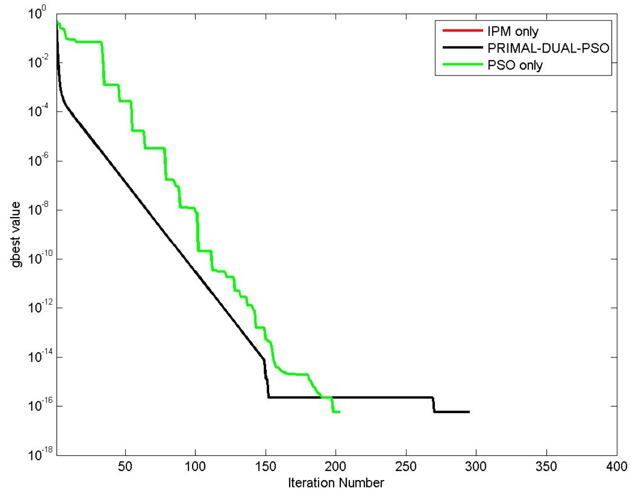

Fig 7: Graph of Schaffer f6 function for Primal-Dual, PSO and *pdPSO*

Table 4: Result Comparison for Schaffer f6 Function

Algorithm	**Best Fitness**	**Worst Fitness**	**Mean Fitness**	**Std. Deviation**
Primal-Dual-PSO	0.00000e+00	2.42012e-01	+8.06720e-03	4.41851e-02
Primal-Dual	2.22045e-16	4.24477e-01	+3.33300e-01	1.13677e-01
PSO	0.00000e+00	4.99600e-16	+2.40548e-17	9.29832e-17

The fifth function is the Schaffer f6 Modified Function which is the sum of five Schaffer f6 functions with different focal points to reach local minimum. It also tests the ability of the optimization algorithm to escape local minima and to check for the presence of premature convergence. IPM and *pdPSO* converged faster than PSO (refer Fig. 8 and Table 5). After the 30th iteration, PSO experienced a sharp fall in the value of its *gbest* and from that point, it is unable to escape a local. *pdPSO* performs better in terms of its best fitness value. When we compare the mean fitness and standard deviation, PSO performs better. *pdPSO* is however superior to PSO and Primal-dual because it was able to overcome the problem of premature convergence. *pdPSO* was able to escape being trapped in local minima whereas PSO and Primal-dual were not able to do so.

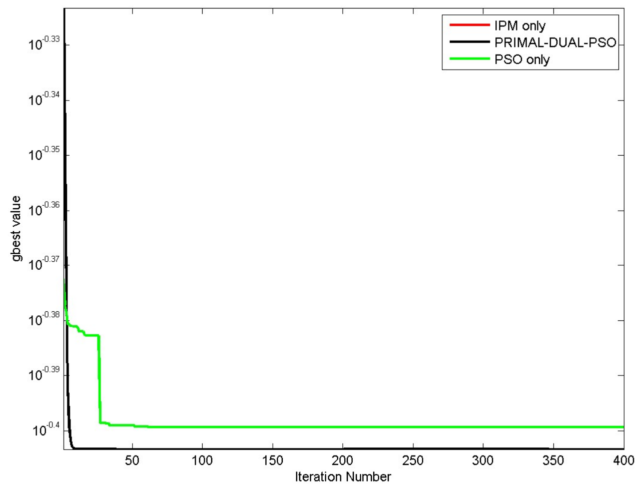

Fig 8: Graph of Schaffer f6 modified function for Primal-Dual, PSO and *pdPSO*

Table 5: Result Comparison for Schaffer f6 modified Function

Algorithm	**Best Fitness**	**Worst Fitness**	**Mean Fitness**	**Std. Deviation**
Primal-Dual-PSO	3.95063e-01	5.66018e-01	+4.03145e-01	3.34147e-02
Primal-Dual	3.95063e-01	4.80710e-01	+4.59061e-01	2.29854e-02
PSO	3.98750e-01	4.84612e-01	+4.01617e-01	1.56753e-02

The sixth function is the Schaffer f6 Bubble Dynamic Function, which comprises of Schaffer f6 where each goes on bubble magnitude cycles up and down. They are 180 degrees out of phase with each other. The function tests the ability of the algorithm to work effectively in a dynamic environment. The simulation results for the Schaffer f6 Bubble Dynamic function with IPM, PSO, and *pdPSO* is depicted in Fig. 9. Details are presented in Table 6. The bubble magnitude cycle was repeated specifically for both IPM and *pdPSO*. PSO has a number of local minima while IPM and *pdPSO* have lesser local minima peaks. *pdPSO* and IPM converged faster than conventional PSO. The performance of *pdPSO* is better in terms of mean fitness, Primal-dual was better in terms of best fitness, and PSO performs better in terms of standard deviation. We can deduce that the overall performance of *pdPSO* is better for this function than that of the other two algorithms because it demonstrates its ability to handle a dynamic environment. This means that *pdPSO* is more suitable in solving problems that are dynamic in nature compared to standard PSO. PSO and Primal-dual were static and got trapped in local minima while *pdPSO* was not.

The seventh function is NDParabola. It is commonly used to test for global minimization problems in Clerc's "semi-continuous challenge." It works well with gradient methods but it is incompatible with PSO which is a stochastic method. This function tests the ability of the algorithm to converge to a global optima after escaping from being trapped in local minima. The simulation results are depicted in Fig. 10 and Table 7. The IPM, PSO and *pdPSO* algorithms converged to global optima. Both PSO and *pdPSO* have several local minima as shown in the figure. There was a significant reduction in the gbest value of IPM, PSO, and *pdPSO* from the beginning of the iteration to the end with the convergence speed favouring *pdPSO* and IPM as compared to the conventional PSO. Based on the numerical values of best fitness, mean fitness and standard deviation, *pdPSO* performs better for this function. Our new algorithm (*pdPSO*) also demonstrates its ability to escape being trapped in local minima and to evade premature convergence in this function. In Fig. 10, the red line depicting the convergence of Primal Dual was overlapped by *pdPSO*.

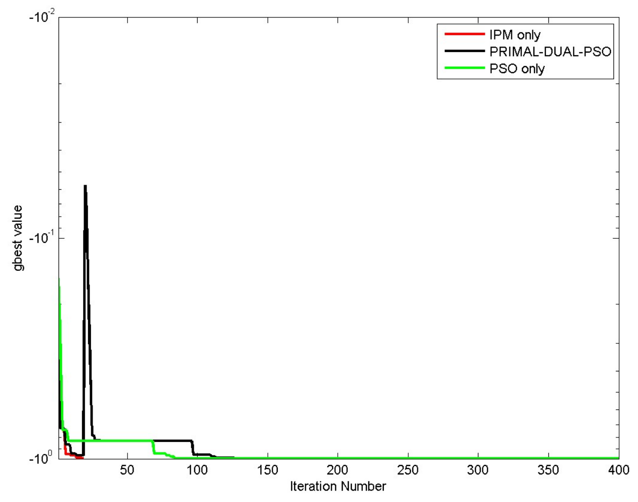

Fig 9: Graph of Schaffer f6 Bubble Dynamic function for Primal-Dual, PSO and *pdPSO*

Table 6: Result Comparison for Schaffer f6 Bubble Dynamic Function

Algorithm	**Best Fitness**	**Worst Fitness**	**Mean Fitness**	**Std. Deviation**
Primal-Dual-PSO	5.18555e-02	8.98753e-01	-6.49279e-01	5.05210e-01
Primal-Dual	4.58057e-02	4.43780e-01	-2.89311e-01	9.56216e-02
PSO	9.02080e-01	9.02179e-01	-9.02131e-01	1.50446e-05

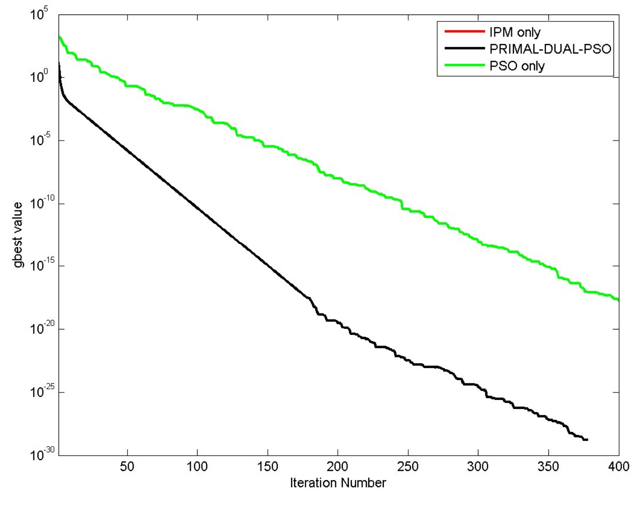

Fig 10: Graph of NDParabola function for Primal-Dual, PSO and *pdPSO*

Table 7: Result Comparison for NDParabola Function

Algorithm	Best Fitness	Worst Fitness	Mean Fitness	Std. Deviation
Primal-Dual-PSO	1.87351e-29	9.11685e-27	+1.19780e-27	2.20741e-27
Primal-Dual	3.05475e-18	1.00235e-17	+6.33670e-18	1.71551e-18
PSO	1.80186e-18	2.76436e-14	+1.05297e-15	5.02818e-15

The eighth function is the Rastrigin function which is a non-convex, multi-modal version of the sphere function with the addition of cosine modulation to produce frequent local minima. It contains millions of local optima (organized in a systematic lattice). This function is a moderately problematic function because of the large search space and the large number of local minima. This highly multimodal function has several local minima which are regularly distributed throughout the iteration for all three algorithms (depicted in Fig. 11 and Table 8). IPM, PSO and *pdPSO* converged to a global optimum. IPM, PSO and *pdPSO* (because of the nature of the Rastrigin function) have several local minima and were trapped in those minima for the rest of the iteration. The performances of IPM and *pdPSO* were slightly better as compared to conventional PSO. Judging from the results of the numerical values of best fitness and mean fitness, *pdPSO* performs better. If we consider the standard deviation, the performance of PSO seems to be better. The rate of convergence of *pdPSO* is however superior to that of the other two algorithms.

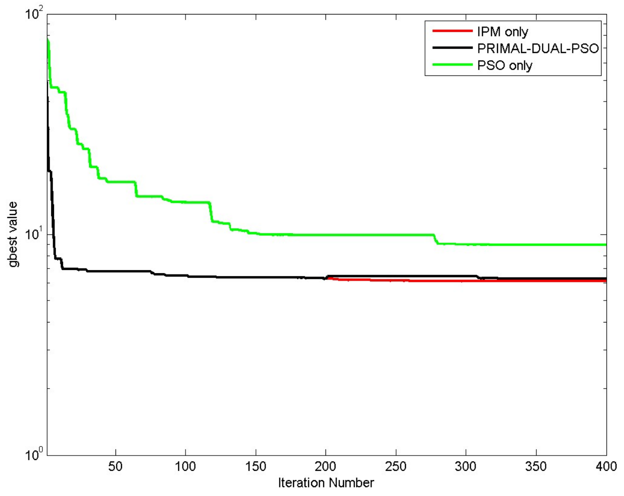

Fig 11: Graph of Rastrigin function for Primal-Dual, PSO and *pdPSO*

Table 8: Result Comparison for Rastrigin Function

Algorithm	Best Fitness	Worst Fitness	Mean Fitness	Std. Deviation
Primal-Dual-PSO	6.29026e+00	1.81783e+02	+7.29321e+01	3.88201e+01
Primal-Dual	6.16813e+00	2.69526e+01	+1.60688e+01	5.13242e+00
PSO	8.95462e+00	8.96635e+00	+8.95527e+00	2.32294e-03

The ninth function is Tripod, which is a semi-continuous function. This function is commonly hard to solve compared to many other optimization algorithms because of the tendency of a stagnant plateau at local minima. The simulation results for the Tripod function with IPM, PSO, and *pdPSO* are depicted in Fig. 12 and Table 9. The *gbest* value of PSO reduced sharply from the start of the iteration up till the 10th iteration and decreased steadily from there until it got trapped in a local minimum and was unable to escape throughout the iterations. Using the numerical values of best fitness and mean fitness as parameters for our judgment, there was not much difference between the performances of PSO and *pdPSO*. The performance of *pdPSO* was better based on the standard deviation when compared to the other two algorithms. From our experiment, *pdPSO* was able to achieve our aim of designing an algorithm that can overcome the problem of premature convergence that usually characterizes standard PSO. In Fig. 12, the red line denoting the convergence of Primal Dual algorithm is not available because the algorithm failed to generate any gbest value. This is because the algorithm (which is deterministic in nature) is unable to cope with such benchmark functions like tripod.

Based on the analysis of our results for the nine benchmarking functions used, *pdPSO* performs better than the other algorithms in 7 out of 9 test cases. In the remaining 2 cases, PSO was superior to the other algorithms. The performance of *pdPSO* was superior for the Sphere, Griewank, Schaffer f6 modified, Schaffer f6 Bubble dynamic, NDParabola, and Tripod functions. The functions where PSO performs better are Ackley and Schaffer f6. The ability of *pdPSO* to overcome the problem of premature convergence and escape being trapped in local minima was demonstrated in the Sphere, Griewank, Schaffer f6 modified, Schaffer f6 Bubble dynamic, and NDParabola functions. The Tripod function further confirmed the capacity of our algorithm to avoid premature convergence. The results from Schaffer f6 Bubble dynamic shows that *pdPSO* has the capability to handle optimization problems in a dynamic environment.

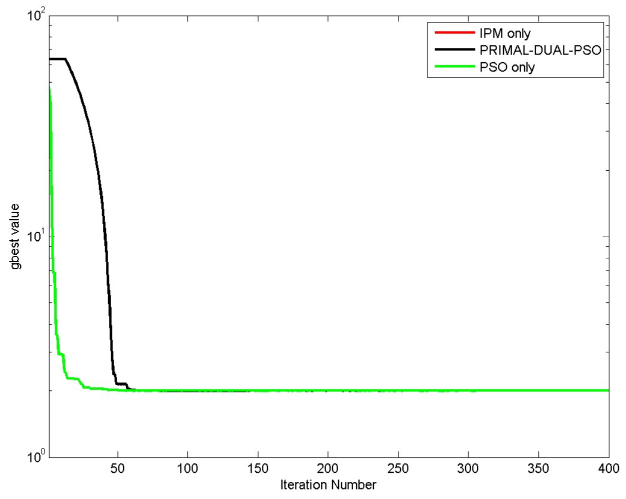

Fig 12: Graph of Tripod function for Primal-Dual, PSO and *pdPSO*

Table 9: Result Comparison for Tripod Function

Algorithm	Best Fitness	Worst Fitness	Mean Fitness	Std. Deviation
Primal-Dual-PSO	2.00000e+00	2.00000e+00	+2.00000e+00	5.94551e-15
Primal-Dual	2.00000e+00	2.00000e+00	+2.00000e+00	1.76109e-08
PSO	2.00000e+00	2.00000e+00	+2.00000e+00	1.96425e-12

With reference to convergence speed, *pdPSO* was faster than PSO and Primal-dual in 5 out of 9 functions that we considered. We can therefore consider *pdPSO* as a fast algorithm that can be used to solve complex numerical optimization problems. *pdPSO* also has a higher level of steadiness in comparison to the other two algorithms. The mean fitness and standard deviation values for the Sphere, Schaffer f6 Bubble dynamic, NDParabola and Tripod functions were very small when compared to those of PSO and Primal-dual. We can therefore conclude that *pdPSO* is a stable algorithm that has the capacity to produce rational results that are reliable. Finally we can deduce that *pdPSO* is a robust algorithm as it performs better than PSO and Primal-dual in its ability to successfully find the global optimum on all the benchmarking functions we used, especially on the Griewank, Schaffer f6, NDParabola and Rastrigin functions which many of the most recent optimization algorithms find very problematic to solve. Consequently, *pdPSO* can be considered as an algorithm that can withstand adverse conditions.

6.0 CONCLUSION

This paper presents a new hybrid optimization algorithm named Primal Dual Interior Point Method Particle Swarm Optimization (*pdPSO*). This algorithm combines the explorative ability of PSO with the exploitative capacity of the Primal Dual Interior Point Method, thereby possessing a strong capacity for avoiding premature convergence. A comparative study of the proposed algorithm has been conducted with conventional PSO and Primal Dual using nine benchmark functions. It is very clear that our algorithm performs better in terms of precision, rate of convergence, steadiness and robustness. The desirable behaviour of *pdPSO* under the unimodal and multimodal functions shows that the algorithm is a suitable tool for solving complex optimization problems that PSO or Primal Dual alone cannot solve efficiently. Our future works include the application of the proposed algorithm for swarm robotics such as flocking and pattern formation. This would be able to validate the capability of the proposed algorithm in handling dynamic optimisation tasks. The applicability of the algorithm can also be extended to solve real world problems such are retina vessel image segmentation, big data optimisation, and economic dispatch.

REFERENCES

[1] Chun-Feng Wang, Kui Liu. A Novel Particle Swarm Optimization Algorithm for Global Optimization. *Computational Intelligence and Neuroscience*, Volume 2016 (2016), Article ID 9482073, 9 pages. Accessed from http://dx.doi.org/10.1155/2016/9482073 on July 28, 2016.

[2] Y. Del Valle, G. K. Venayagomoorthy, .S Mohagheghi, J. C. Hernandez, R. G. Harley, “Particle swarm optimization: basic concepts, variants and applications in power systems”. *IEEE Trans Evol Comput (2008)* 12(2):171 – 195.

[3] J. Kennedy, R. C. Eberhart, “Particle swarm optimization”, in: *Proceedings of the International Conference on Neural Networks, vol. 4, IEEE Press*, Piscataway, NJ, (1995) pp. 1942-1948.

[4] F. A. Guerra, L. D. S. Coelho, “Applying Particle Swarm Optimization to Adaptive Controller”. A. Saad et al. (Eds.): *Soft Computing in Industrial Applications, ASC*, vol. 39, (2007), pp. 82–91.

[5] Y. Shi, R. C. Eberhart, “Parameter Selection in particle swarm optimization”, In Proceedings of the 7th International Conference on Evolutionary Programming, (1998) pp. 591 – 600.

[6] H. Rusman. An Efficient Particle Swarm Optimization Technique for Solving the Non-convex Economic Dispatch Problems. *International Journal of Engineering Sciences*, (2013) 2(5), pp. 137 – 144.

[7] Keisam Thoiba Meetei. A Survey: Swarm Intelligence vs. Genetic Algorithm. *International Journal of Science and Research (IJSR),* (2014), ISSN (Online): 2319-7064.

[8] Yang X.-S., Karamanoglu M., He X. S. Flower pollination algorithm: a novel approach for multiobjective optimization. *Engineering Optimization*. (2014); 46 (9):1222–1237. doi: 10.1080/0305215X.2013.832237.

[9] Yang X. S. *Cuckoo Search and Firefly Algorithm: Theory and Applications*. Vol. 516. Heidelberg, Germany: Springer; (2014). (Studies in Computational Intelligence).

[10] Yang X.-S., Deb S. Cuckoo search: recent advances and applications. *Neural Computing and Applications*. (2014); 24 (1):169–174. doi: 10.1007/s00521-013-1367-1.

[11] Basturk, B., Karaboga, D. A powerful and Efficient Algorithm for Numerical Function Optimisation: Artificial Bee Colony (ABC) algorithm, *J. Glob Optim,* (2007),vol. 39, 459-471.

[12] Besdok, E., & Civicioglu, P., A conceptual comparison of the Cuckoo-search, particle swarm optimisation, differential evolution and artificial bee colony algorithms *Artif Intell Rev,* (2013), 39, (pp. 315-346). DOI 10.1007/s10462-011-9276-0.

[13] Gong, C. Opposition-Based Adaptive Fireworks Algorithm. *Algorithms* (2016), *9*, 43.

[14] Saadi Y, Yanto ITR, Herawan T, Balakrishnan V, Chiroma H, Risnumawan A. Ringed Seal Search for Global Optimization via a Sensitive Search Model (2016) *PLoS ONE*, 11(1): e0144371. doi:10.1371/journal.pone.0144371.

[15] Xianbing Meng, Yu Liu, Xiaozhi Gao, Hengzhen Zhang. A New Bio-inspired Algorithm: Chicken Swarm Optimization. 5th International Conference, ICSI 2014, Hefei, China, October 17-20, (2014), *Proceedings, Part I.* pp 86-94, DOI 10.1007/978-3-319-11857-4_10.

[16] Dinghui Wu, Fei Kong ; Wenzhong Gao ; Yanxia Shen ; Zhicheng Ji. Improved chicken swarm optimization. 2015, pp 681 – 686, *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), (2015), IEEE International Conference*, DOI: 10.1109/CYBER.2015.7288023.

[17] Yang X.-S. Bat algorithm: literature review and applications. *International Journal of Bio-Inspired Computation*. (2013);5 (3):141–149. doi: 10.1504/IJBIC.2013.055093.

[18] Yu, S. H.; Zhu, S. L.; Ma, Y. Enhancing firefly algorithm using generalized opposition-based learning. Computing (2015), 97, 741–754

[19] A. Abraham, A. Konar, S. Das, "Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives". *Studies in Computational Intelligence, (SCI)* 116, (2008), pp. 1–38.

[20] Liang, J. J., Qin, A. K., Suganthan, P. N. & Baskar, S. Comprehensive Learning Particle Swarm Optimizer for Global Optimisation of Multimodal Functions, *IEEE transactions on evolutionary computation*, Vol. 10, No. 3, June, (2006). pp. 281 – 295.

[21] Zhao X. A perturbed particle swarm algorithm for numerical optimization. *Appl Soft Comput.,* (2010). vol. 10, issue 1, pp. 119–124.

[22] Akbari R & Ziarati K. A rank based particle swarm optimization algorithm with dynamic adaptation. *J Comput Appl Math*, (2011). vol. 235, pp. 2694–2714

[23] Zhan Z. H, Zhang J, Li Y & Shi Y. H. Orthogonal learning particle swarm optimisation. *IEEE Trans Evol Comput*, (2011). vol 15, pp. 832–847.

[24] Huang H, Qin H, Hao Z & Lim A. Example-based learning particle swarm optimisation for continuous optimisation. *Inf Sci*, (2012). vol. 182 pp. 125–138.

[25] Xu G. An adaptive parameter tuning of particle swarm optimization algorithm. *Appl Math Comput,* (2013). vol. 219, pp.4560–4569

[26] Wang H, Sun H, Li C. H., Rahnamayan S, Pan J. S. Diversity enhanced particle swarm optimization with neighborhood search. *Inf Sci,* (2013), vol. 223, pp. 119–135

[27] Gang Xu, Binbin Liu, Jun Song, Shuijing Xiao & Aijun Wu. Multiobjective sorting-based learning particle swarm optimisation for continuous optimisation. *Nat Comput*, (2016). DOI 10.1007/s11047-016-9548-3

[28] C. D. Laird. "Structured Large-Scale Nonlinear Optimization Using Interior-Point Method: Applications in Water Distribution Systems". *PhD Thesis Department of Chemical Engineering, Carnegie Mellon University*, Pittsburgh, Pennsylvania (2006).

[29] L. M. Blohm Winternitz, "Primal-Dual Interior-Point Algorithms for Linear Programs with Many Inequality Constraints". *Dissertation submitted to the Faculty of the Graduate School of the University of Maryland,* College Park in partial fulfillment of the requirements for the degree of Doctor of Philosophy (2010).

[30] K. Morikuni and K. Hayami, Inner-iteration Krylov subspace methods for least squares problems, *SIAM J. Matrix Anal. Appl.,* 34 (2013), pp. 1–22, http://dx.doi.org/10.1137/ 110828472.

[31] K. Morikuni and K. Hayami, Convergence of inner-iteration GMRES methods for rank deficient least squares problems, *SIAM J. Matrix Anal. Appl*., 36(1) (2015), pp. 225–250, http://dx.doi.org/10.1137/130946009.

[32] Yiran Cui, Keiichi Morikuni, Takashi Tsuchiya, Ken Hayami. Implementation of interior-point methods for LP Based on Krylov Subspace Iterative Solvers with Inner-Iteration Preconditioning (2016). *arXiv:1604.07491v1 [math.oc].* Access from https://arxiv.org/pdf/1604.07491v1.pdf on July 28 2016.

[33] Tangi Migot, Mounir Haddou. A new direction in polynomial time interior-point methods for monotone linear complementarity problem, Mode-Smai (2016), *Inp-Enseeiht* Toulouse, 23-24 et 25 Mars 2016. Accessed from http://mode2016.sciencesconf.org/85064/document on July 28 2016.

[34] A. Abraham, M. Pant, P. Bouvry, R. Thangaraj, "Particle swarm optimization: Hybridization perspectives and experimental illustrations", *Appl. Math. Comput,* vol. 217, pp. 5208 – 5226, (2011) doi:10.1016/j.amc.2010.12.053

[35] A. Z. Torn, "Global Optimization", *Lecture Notes in Computer Science*, vol. 350, Springer-Verlag (1989).

[36] Y. Shi and R. C. Eberhart, "Parameter Selection in particle swarm optimization", *In Proceedings of the 7th International Conference on Evolutionary Programming*, pp. 591 – 600, (1998).

[37] N. A. A. Aziz, Z Ibrahim, "Asynchronous Particle Swarm Optimization for Swarm Robotics", *International Symposium on Robotics and Intelligent Sensors (IRIS 2012)*. Procedia Engineering 41 pp. 951 – 957.

[38] J. Kennedy, R. Eberhart, Y. Shi, "Swarm Intelligence", *Morgan Kaufmann Publishers Inc*. San Francisco, CA, USA, (2001).

[39] P. Civicioglu, E. Besdok. "A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms". *Artif Intell Rev* (2013) 39:315-346, DOI 10.1007/s10462-011-9276-0.

[40] S. Mehrotra. "On the implementation of a primal-dual interior point method," *SIAM Journal on Optimization,* (1992) vol. 2, pp. 575-601.

[41] K. R. Frisch, "The logarithmic potential method of convex programming", *Technical Report,* University Institute of Economics, Oslo, Norway, 1955.

[42] S. J. Wright, "Primal-dual interior-point methods", *SIAM* (1997) Philadelphia, PA, 1st edition.

[43] S. Boyd, L. Vandenberghe, "Convex Optimization"*, Cambridge University Press* (2004) New York, 1st edition.

[44] V. H. Quintana, G. L. Torres. "Introduction to Interior-Point Methods", *IEEE PES task Force on Interior-Point Methods Applications to Power Systems,* 1997. Available online at http://wathvdc3.uwaterloo.ca/~iee-ipm.

[45] A. Sofer, C. A. Johnson, J. Seidel, "Interior-point methodology for 3-D PET reconstruction", *IEEE Trans. Medical Imaging,* vol. 19, no. 4, (2000), pp. 271-285.

[46] E. Chouzenoux, J. Idier, S. Moussaoui, "Efficiency of Line Search Strategies in Interior Point Methods for Linearly Constrained Signal Restoration", in: *Proceedings of the IEEE Workshop on Statistical Signal Processing (SSP)*, Nice, France, 28-30 June (2011), pp. 101-104.

[47] M. Glavic, L. Wehenkel, "Interior Point Methods: A Survey, Short Survey of Applications to Power Systems, and Research Opportunities", *Technical Report. University of Liège Electrical Engineering and Computer Science Department Sart Tilman* B-28 4000 Liege, Belgium (2004).

[48] J. C. Gilbert, P. Armand, S. Jan-Jégou, "A feasible BFGS interior point algorithm for solving strongly convex minimization problems", *SIAM J. Optimization*, vol. 11, (2000), pp. 199 – 222.

[49] B. K. Panigrahi, Y. Shi, M. H. Lim. "Handbook of Swarm Intelligence: Concepts, Principles and Applications", *Springer-Verlag Berlin Heidelberg*, (2011) ISBN 978-3-642-17389-9, pp. 119-132.

[50] M. Clerc, "Semi-Continuous Challenge", 2004.
www.clerc.maurice.free.fr/pso /semi-continuous_challenge/semi-continuous_challenge.htm

[51] J. B. Park, J. R. Shin, K. Y. Lee, Y. W. Jeong, "An Improved Particle Swarm Optimization for Nonconvex Economic Dispatch Problems", *IEEE Transactions on Power Systems* (2010) Vol. 25, No. 1, pp. 156- 166.

[52] Locatelli, M. "A note on the Griewank test function", *Journal of Global Optimization*, (2003) 25 (2), pp. 169-174, doi:10.1023/A:1021956306041

[53] B. Basturk, D. Karaboga, "A powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) algorithm", *J. Glob Optim,* (2007) vol. 39, pp. 459-471.

[54] J. M. Dieterich and B. Hartke, "Empirical review of standard benchmark functions using evolutionary global optimization", (2012) arXiv:1207.4318v1 [cs.NE] 18 Jul 2012.

[55] Matlab Central (2014). *http://www.mathworks.com/matlabcentral/fileexchange/7506-particle-swarm-optimization-toolbox/content/testfunctions/*

[56] M. R. Chena, X. Lia, X. Zhanga, Y. Z. Lu. "A novel particle swarm optimizer hybridized with extremal optimization", *Applied Soft Computing*, vol. 10, Issue 2, March (2010), pp. 367–373.