(IJACSA) International Journal of Advanced Computer Science and Applications,
Vol. 10, No. 8, 2019

# Empirical Study of Segment Particle Swarm Optimization and Particle Swarm Optimization Algorithms

Mohammed Adam Kunna Azrag[1], Tuty Asmawaty Abdul Kadir[2]
Faculty of Computer Science & Software Engineering
Universiti Malaysia Pahang, Kuantan, Malaysia

***Abstract*—In this paper, the performance of segment particle swarm optimization (Se-PSO) algorithm was compared with that of original particle swarm optimization (PSO) algorithm. Four different benchmark functions of Sphere, Rosenbrock, Rastrigin, and Griewank with asymmetric initial range settings (upper and lower boundaries values) were selected as the test functions. The experimental results showed that, the Se-PSO algorithm achieved better results in terms of faster convergences in all the testing cases compared to the original PSO algorithm. However, the experimental results further showed the Se-PSO as a promising optimization algorithm method in some other different fields.**

***Keywords*—Se-PSO; PSO; sphere; Rosenbrock; Rastrigin; Griewank**

## I. Introduction

Within the last two decades, optimization algorithms with mathematical programing have proved to be effective in solving large complex optimization problems. Recently, swarm intelligence techniques have gained popularity because of their capacity to locate partially optimal solutions for combinatorial optimization problems [1, 2]. These techniques have been applied in various areas, such as economics, engineering, bioinformatics, and industry. These problems are better solved using swarm intelligence techniques because they are usually very hard to solve accurately due to the lack of any precise algorithm to solve them [1, 2]. The swarm intelligence algorithms mainly depend on updating the population of individuals by applying some operators according to the fitness information obtained from the environment. With these updates, the individuals in a population are expected to move towards an optimum solution.

The Particle Swarm optimization (PSO) is one of the popular swarm algorithms which were formulated in 1995 by Dr. Kennedy and Eberhart. The PSO algorithm simulates the flocking behavior of birds and the schooling of fishes in order to achieve complex solutions [3-5]. The PSO algorithm is easy to execute and requires few parameters to be adjusted; it is computationally proficient and has a faster speed and premature convergence towards optima compared to Genetic Algorithm (GA) and Simulating Annealing (SA) algorithm [4]. It also has a flexible and well-balanced mechanism of improving its exploration capabilities [6]. The scenario of PSO is started by initializing a population of random solutions. Each potential solution is assigned with a randomized velocity, and the potential solutions are called particles. Each particle has its own position, velocity, and fitness used to decide its best or bad positions in the solution space. The particle search depends on the personal best position (pbest) and the global best position (gbest) of each particle [7]. Moreover, the ability of PSO to find an optimum solution in reasonable time creates the need for its continuous improvement [8]. However, a segmentation of the PSO is implemented in this study to improve its convergence and accuracy using 4 optimization functions problem.

The rest of this paper is organized thus: the second section discusses the methodology of the PSO and the implemented Se-PSO algorithms. The third section explains the experimental setup and discussion of the results. The fourth section provides the conclusions drawn from the study.

## II. Methodology

This paper presents the implementation method of Se-PSO with the comparison of PSO algorithms. The description of Se-PSO and PSO is given in *Pseudo code* in Algo (1 and 2). Se-PSO algorithm is developed according to conventional PSO algorithm. Thus, it has to go through the same procedures initialization of the bird's step, number of birds, iteration, the problem dimension, and the position/velocity updating evaluation processes. However, the segmentation of the problem is added to PSO algorithm in order to reduce the time with a few iterations. The segmentation technique is always providing a fast convergence that able to achieve the best initial local position.

### *A. PSO Algorithm*

The PSO algorithm is simulating the behavior of birds flocking and fish schooling in order to solve optimization problem in D-dimensional search space. This algorithm was proposed in 1995 [9]. The initialization of PSO is start by a group of random particles (solutions) and then searches for the optimum solution by updating generations. Each particle is flown through the solution problem space, having its position based on the information from its own personal best position and the best particle of the swarm. The performance of each particle and how close from the global optimum is calculated using a fitness function of the optimization problem. Each particle $i$ flies through the D-dimensional solution space and maintains the current position $x_i$ of particle $i$, the personal best position $p_i$ of the particle $i$, and the current velocity $v_i$ of particle $i$. The particle keeps track of its coordinates in the solution space which are associated with best solution (fitness) it has achieved so far which stored at each iteration. This value is called pbest (personal best position). Another value of the (best) called gbest (global best position) tracked by the global version of the particle swarm optimizer is the overall best value, and its location obtained so far by any particle in the population. During the particle swarm optimization searching calculation for solution, at each time step, changing the velocity (accelerating) each particle toward its pbest and gbest. Were this calculation mathematically described by this equation:

$$v_i(t+1) = \omega v_i(t) + c_1 r_1 \left(p_i^{best}(t)\right) - c_2 r_2 \left(g_i^{best}(t)\right)$$

$$X_i(t+1 = X_{ir}(t) + v_{ir}(t+1)$$

Where $v_i$ is the velocity of particle $i$ it iteration $t$, $X_i$ is the position of particle $i$ at iteration $t$, $p_i^{best}$= ($p_{i1}^{best}$, $p_{i2}^{best}$, …, $p_{ir}^{best}$) is the personal best position of each particle $i$ till the particle number $r$ and $g_i^{best}$=($g_{i1}^{best}$, $g_{i2}^{best}$, …, $g_{ir}^{best}$) is the global best position of the entire particle $i$ till the particle number $r$ ; $\omega$ is an inertia weight parameter, $c_1c_2$ are acceleration coefficients, $r_1r_2$ are random number between 0 and 1, and $r$ is the dimension in the solution space. The PSO algorithm procedure can be summarized as shown below in Algo 1.

Algo 1

1. Start
2. Initialize the $nbird$, $bird\_s$, $d$, $\omega$;
3. Initialize the $v_i$, $X_i$, $c_1c_2$, $r_1r_2$;
4. Calculate the $fitness\ function$
5. If $fitness\ function\ > 0$
6. For each $Q$;
   a. Iter $Q$=1, $Q++$;
   b. Updating the velocity $v_i$ towards fitness:
   c. $v_i(t+1) = \omega v_i(t) + c_1 r_1 \left(p_i^{best}(t)\right) - c_2 r_2 \left(g_i^{best}(t)\right)$;
   d. Update the position $X_i$ towards fitness:
   e. $X_i(t+1 = X_{ir}(t) + v_{ir}(t+1)$;
7. If $fitness\ function\ \leq 0$;
   a. Print $g_i^{best}$ of each particles;
8. If $fitness\ function > 0$ return step 2 till the iteration found highly solution or finished.
9. End

## B. Se-PSO Algorithm

The Segment Particle Swarm algorithm idea is to divide the PSO particles to searching groups that can be considered as segments [10]. The segmentation means to separate the problem into parts or segments which reach the solution easily. In addition, the idea of Segmentation PSO algorithm is to divide the initial values into segments to help PSO particles during the search for the optimal values finding a local best position that may the global best position is around it. The segmentation can be divided into more than two groups based on the dimension.

Considering Fig. 1 is the scenario of parameters segmentation. This scenario contains 3 parameters in search space. Each parameter has its own boundaries. Based on the significance parameter the segment is proposed to find the best position in the parameters boundaries. After the best position is found, the new optimum segment is divided by 2 so the PSO algorithm starts searching from the new segments as described in Fig. 2.

Each group of particles considered a segment while the procedures for finding the optimal solution (optimal segments) is following PSO algorithm then the optimal segment for the initial parameters will be used as the new initial parameters later PSO search in that range toward the optimal solution. The segmentation can be described in Fig. 3.

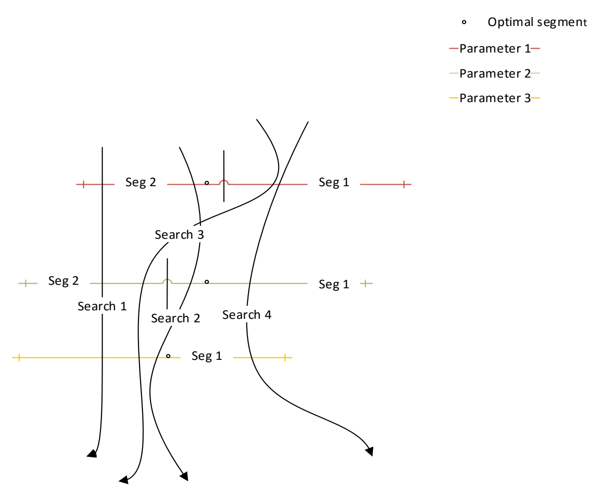


Fig. 1. Parameter Segment.

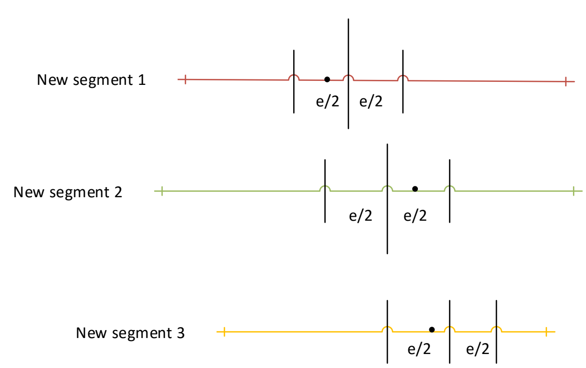


Fig. 2. New Segment.

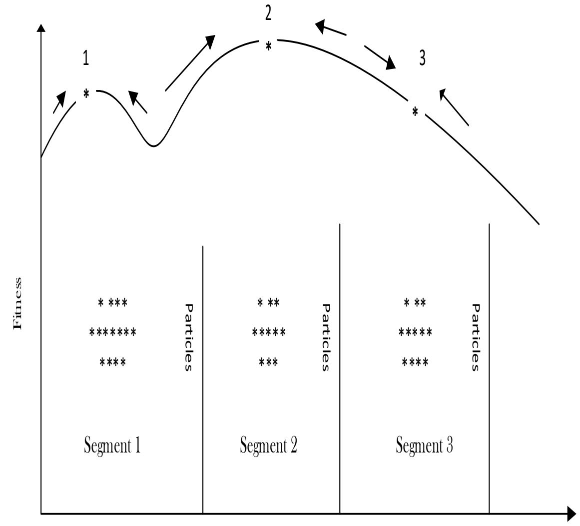


Fig. 3. Segment Particles.

In the above Fig. 3, the segments easily locate many local points where point 2 is the best local points, as it's the best individual position of the particle and the best position of the entire swarm should be modified to achieve the best optimal segment as a fellow in Eq. (1).

$$Seg\ length = \frac{initial\ limits}{nu\ of\ segments} \tag{1}$$

Then eq. (3) should be modified according to the segmentation changes and described as follow in Eq. (2) and (3).

$$v_i\ (t+1,j) = \omega v_i\ (t,j) + c_1\, r_1\ (p_i\ (t,j) - X_i\ (t,j)) + c_2\, r_2\ (G_i\ (t,j) - X_i\ (t,j)) \tag{2}$$

$$X_i(t+1,j) = X_i(t,j) + v_i(t+1,j) \tag{3}$$

Hence the optimal segment can be described in Eq. (4) below:

$$optimal\ segment = \frac{optimal\ X_{ij} \mp segemnt\ length}{2} \tag{4}$$

Where $j$ is the number of segments.

**Algo 2**

1 BEGIN

2 Initialize $S, D, \omega$;

3 initialize $v_i$, $X_i$, $c_1$, $c_2$, $r_1$, $r_2$, number_segment;

4 Segment length=initial value/number_segment;

5 Adopt the $b_{e1}$: $b_{en}$ parameters boundaries;

6 For j= 1 to number of segment;

7 Determine initial $fitness$ for segment $J$;

8 Assume $Best\ fitness = initial\ fitness$;

9 If $fitness$ > Best $fitness$;

10 For each $S$;

11 iter $S$=1, $S$++;

12 Updating the velocity $V_i$ toward fitness:

$$v_i\ (t+1,j) = \omega v_i\ (t,j) + c_1\, r_1\ (p_i\ (t,j) - X_i\ (t,j)) + c_2\, r_2\ (G_i\ (t,j) X_i\ (t,j));$$

13 Update the position $X_{i,}j$ toward fitness:

$$X_i(t+1,j) = X_i(t,j) + v_i(t+1,j);$$

14 End if,

15 If fitness ≤ Best $fitness$;

Print $G_i best$ of each particles;

Update the $b_{e1}$: $b_{en}$ based of $G_i best$ of each kinetic parameters;

16 If fitness > Best $fitness$ return step 2 till the iteration is finished or discover high-quality solution;

17 End if,

18 End if,

19 Global point(j)= $G_i best$;

20 Next, j;

21 Optimal_segment=max (Global point) ± segment length/2;

22 Repeat algorithm 1 for the new initial values;

23 End.

## III. Experimental Setup

For comparison, for nonlinear functions are used here. The first function is the Sphere function described by equation ($f(x)$):

$$f(x) = \sum_{s=1}^{n} X^2$$

Where $X = (X_1, X_2, \ldots, X_n)$ is an n-dimensional real-valued vector. The second function is the Rosenbrock function described by equation ($f_1(x)$):

$$f_1(x) = \sum_{i=1}^{n} (100(X_i - X_i^2)^2 + (X_i - 1)^2)$$

The third function is generalized Rastrigrin function described by equation ($f_2(X)$):

$$f_2(X) = \sum_{i=1}^{n} (X_i^2 - 10\cos(2\pi X_i) + 10)$$

The last function is the generalized Griewank function described by equation ($f_3(X)$):

$$f_3(X) = \frac{1}{4000}\sum_{i=1}^{n} X_i^2 - \prod_{i=1}^{n}\cos\left(\frac{X_i}{\sqrt{i}}\right) + 1$$

TABLE I. LOWER AND UPPER BOUNDARIES

| Function | Lower and upper values |
|---|---|
| $f$ | [-5, 5] |
| $f_1$ | [-5, 10] |
| $f_2$ | [-5.12, 5.12] |
| $f_3$ | [-600, 600] |

Following the suggestion in [11] and for the purpose of comparison, the lower and upper values are selected to be as the original values. Table I lists the initialization of the upper and lower values of the four functions.

As in the above Table I, for each function, the maximum number of iteration is set to 50, 100, 150 in the both algorithms. The number of birds is set to 20, 40, 60, and 80. In order to compare the both algorithms inertia weight is adapted to 0.9. The learning factors $c_1\, c_2 \approx 4$. The dimension is 10 for both. Each algorithm was tested 10 times to get the Mean global best position.

## IV. RESULTS AND DISCUSSION

In Sphere function, the global best position of Se-PSO showed a very improved result in short time when compared to PSO. Moreover, as shown in Tables II and III the convergence speed of the Se-PSO towards the optimum values was faster when compared to PSO. It's however, noticeable that the convergence of Se-PSO was quick in the all function but slowed down searching large space for the global best position before to be decided by Se-PSO algorithm as it approaches the optimal. The Se-PSO it took 0.213s to reach the best global position but the PSO reached the global best position in 0.033s. Where, Se-PSO takes 0.004s to decide the global best position while PSO takes 0.013s are depicted in Tables II and III in the self –time column. Moreover, Se-PSO searching large space almost 7 times of PSO as it described in the Calls column (4920 and 650) respectively.

Looking at Table IV where Sphere function was tested using Se-PSO and PSO on a 10-dimensional space with 5, 15, and 25 bird-step and 5, 15, and 25 iterations, the global best position of Se-PSO has very good result that is almost 4 times as good as the result of PSO. Moreover, the convergence speed of Se-PSO toward the optimum values are faster than those of PSO as showed in Tables II and III. Moreover, it is easy to observe that Se-PSO convergences quickly but slows its convergence speed down when reaching the optimum. It takes 0.004s to get the best global position while PSO takes 0.013s as can be seen clearly in Tables II and III. Finally, the total performance speed of the both algorithm is 0.213s with 25 iterations for Se-PSO and 0.033s only for PSO.

In Rastrigin function, the global best position of Se-PSO showed a very improved result in short time when compared to PSO in Table VII. Moreover, as shown in Tables V and VI the convergence speed of the Se-PSO towards the optimum values was faster when compared to PSO. It's however, noticeable that the convergence of Se-PSO was quick in the all function but slowed down searching large space for the global best position before to be decided by Se-PSO algorithm as it approaches the optimal. The Se-PSO it took 0.363s to reach the best global position but the PSO reached the global best position in 0.048s. Where, Se-PSO takes 0.001s to decide the global best position while PSO takes 0.013s are depicted in Tables V and VI in the self –time column. Moreover, Se-PSO searching large space almost 7 times of PSO as it described in the Calls column (4920 and 650), respectively.

TABLE II. SE-PSO CONSUMPTION FOR SPHERE FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **Se-PSO** | 1 | 0.213s | 0.004s |
| **PSO** | 3 | 0.209s | 0.030s |
| **Sphere** | 4920 | 0.179s | 0.179s |

TABLE III. PSO CONSUMPTION FOR SPHERE FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **PSO** | 1 | 0.033s | 0.013s |
| **Sphere** | 650 | 0.020s | 0.020s |

TABLE IV. SE-PSO AND PSO RESULTS FOR SPHERE FUNCTION

| Bird step | Dimension | Iteration | $g^{Best}$ of Se-PSO | $g^{Best}$ of PSO |
|---|---|---|---|---|
| 5 | 10 | 5 | 1.01335e-5 | 0.0701 |
| | | 15 | 2.12225e-06 | 0.0517 |
| | | 25 | 4.25101e-7 | 0.0074 |
| 15 | 10 | 5 | 3.25661e-07 | 0.0158 |
| | | 15 | 4.25861e-08 | 0.0132 |
| | | 25 | 1.05843e-08 | 0.0013 |
| 25 | 10 | 5 | 1.00035e-08 | 0.0001 |
| | | 15 | 2.03589e-09 | 0.0011 |
| | | 25 | 1.23565e-0.9 | 0.00002 |

TABLE V. SE-PSO CONSUMPTION FOR RASTRIGIN FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **Se-PSO** | 1 | 0.363s | 0.001s |
| **PSO** | 3 | 0.362s | 0.028s |
| **Rastrigin function** | 4920 | 0.334s | 0.334s |

TABLE VI. PSO CONSUMPTION FOR RASTRIGIN FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **PSO** | 1 | 0.048s | 0.013s |
| **Rastrigin Function** | 650 | 0.035s | 0.035s |

TABLE VII. SE-PSO AND PSO RESULTS FOR RASTRIGIN FUNCTION

| Bird step | Dimension | Iteration | $g^{Best}$ of Se-PSO | $g^{Best}$ of PSO |
|---|---|---|---|---|
| 5 | 10 | 5 | 1.1237e-05 | 1.0835 |
| | | 15 | 8.3919e-06 | 0.2655 |
| | | 25 | 3.5864e-06 | 0.9891 |
| 15 | 10 | 5 | 1.6515e-06 | 0.0028 |
| | | 15 | 2.2586e-07 | 0.0359 |
| | | 25 | 4.2056e-07 | 0.0026 |
| 25 | 10 | 5 | 3.0125e-08 | 0.0085 |
| | | 15 | 2.0123e-08 | 0.0064 |
| | | 25 | 1.0213e-09 | 0.003 |

In the Rosenbrock function, the global best position of Se-PSO showed a very improved result in short time when compared to PSO in Table X. Moreover, as shown in Tables VIII and IX, the convergence speed of the Se-PSO towards the optimum values was faster when compared to PSO. It's however, noticeable that the convergence of Se-PSO was quick in the all function but slowed down searching large space for the global best position before to be decided by Se-PSO algorithm as it approaches the optimal. The Se-PSO it took 0.158s to reach the best global position but the PSO reached the global best position in 0.03s. Where, Se-PSO takes 0.004s to decide the global best position while PSO takes 0.013s are depicted in Tables VIII and IX in the self –time column. Moreover, Se-PSO searching large space almost 7 times of PSO as it described in the Calls column (4920 and 650) respectively.

Looking at Table X above, where Rosenbrock function was tested using Se-PSO and PSO on a 10-dimensional space with 5, 15, and 25 bird-step and 5, 15, and 25 iterations, the global best position of Se-PSO has very good result that is almost 4 times as good as the result of PSO. Moreover, the convergence speed of Se-PSO toward the optimum values are faster than those of PSO as showed in Tables VIII and IX. Moreover, it is easy to observe that Se-PSO convergences quickly but slows its convergence speed down when reaching the optimum. It takes 0.004s to get the best global position while PSO takes 0.013s as can be seen clearly Tables VIII and IX. Finally, the total performance speed of the both algorithm is 0.158s with 25 iterations for Se-PSO and 0.030s only for PSO.

The global best position of Se-PSO showed a very improved result in short time when compared to PSO in Griewank function depicted in Table XIII. Moreover, as shown in Tables XI and XII the convergence speed of the Se-PSO towards the optimum values was faster when compared to PSO. It's however, noticeable that the convergence of Se-PSO was quick in the all function but slowed down searching large space for the global best position before to be decided by Se-PSO algorithm as it approaches the optimal. The Se-PSO it took 0.037s to reach the best global position but the PSO reached the global best position in 0.014s. Where, Se-PSO takes 0.002s to decide the global best position while PSO takes 0.013s are depicted in Tables XI and XII in the self –time column. Moreover, Se-PSO searching large space almost 7 times of PSO as it described in the Calls column (4920 and 650), respectively.

TABLE VIII. SE-PSO CONSUMPTION FOR ROSENBROCK FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **Se-PSO** | 1 | 0.158s | 0.004s |
| **PSO** | 3 | 0.154s | 0.026s |
| **Rosenbrock function** | 4920 | 0.128s | 0.128s |

TABLE IX. PSO CONSUMPTION FOR ROSENBROCK FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **PSO** | 1 | 0.030s | 0.013s |
| **Rosenbrock function** | 650 | 0.016s | 0.016 |

TABLE X. SE-PSO AND PSO RESULTS FOR ROSENBROCK FUNCTION

| Bird step | Dimension | Iteration | $g^{Best}$ of Se-PSO | $g^{Best}$ of PSO |
|---|---|---|---|---|
| 5 | 10 | 5 | 0 | 0.8743 |
| | | 15 | 0 | 0.3272 |
| | | 25 | 0 | 0.0545 |
| 15 | 10 | 5 | 0 | 0.6533 |
| | | 15 | 0 | 0.942 |
| | | 25 | 0 | 0.7522 |
| 25 | 10 | 5 | 0 | 0.539 |
| | | 15 | 0 | 0.6945 |
| | | 25 | 0 | 0.2477 |

TABLE XI. SE-PSO CONSUMPTION FOR GRIEWANK FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **Se-PSO** | 1 | 0.037s | 0.002s |
| **PSO** | 3 | 0.035s | 0.022s |
| **Griewank function** | 4920 | 0.013s | 0.013s |

TABLE XII. PSO CONSUMPTION FOR GRIEWANK FUNCTION

| Function | Calls | Total time | Self-time |
|---|---|---|---|
| **PSO** | 1 | 0.014s | 0.013s |
| **Griewank function** | 650 | 0.001s | 0.001 |

TABLE XIII. SE-PSO AND PSO RESULTS FOR GRIEWANK FUNCTION

| Bird step | Dimension | Iteration | $g^{Best}$ of Se-PSO | $g^{Best}$ of PSO |
|---|---|---|---|---|
| 5 | 10 | 5 | 4.5368e-05 | 0.1176 |
| | | 15 | 8.1222e-07 | 0.0446 |
| | | 25 | 4.1117e-09 | 0.0023 |
| 15 | 10 | 5 | 3.6206e-09 | 0.0075 |
| | | 15 | 8.4574e-10 | 0.3777 |
| | | 25 | 5.2106e-11 | 0.3637 |
| 25 | 10 | 5 | 9.2952e-12 | 0.0137 |
| | | 15 | 3.5258e-13 | 0.0024 |
| | | 25 | 4.2586e-14 | 0.0019 |

## V. CONCLUSION

The Se-PSO algorithm was introduced in this study through the incorporation of a segmentation solution during the searching process by the particles into the original version of the PSO. After this, the best segment position of Se-PSO algorithm was projected as the new search dimension to find the optimum values. To investigate the proposed method, four functions were employed. The results of the experiments showed that the Se-PSO exhibited fast convergence when compared with the ABO. Although it was observed that the Se-PSO algorithm worked efficiently, the need for further experiments on more complex multimodal, separable and non-separable functions cannot be ruled out in order to further validate the search capacity of the Se-PSO. Moreover, there is a need to apply the proposed Se-PSO to other optimization search landscapes such as urban transportation problems, job scheduling, dynamic modeling, and vehicle routing.

## ACKNOWLEDGMENT

The authors gratefully acknowledge the financial support from the Universiti Malaysia Pahang and FSKKP under PGRS190312 and FRGS under Grant RDU 160101. The authors would like to thank Dr. Tuty Asmawaty Abdul Kadir, who supervised this work.

## REFERENCES

[1] Soliman, Soliman Abdel-Hady, and Abdel-Aal Hassan Mantawy. Modern optimization techniques with applications in electric power systems. Springer Science & Business Media, 2011.

[2] Venter, Gerhard. "Review of optimization techniques." Encyclopedia of aerospace engineering (2010).

[3] Kennedy, James. "Particle swarm optimization." Encyclopedia of machine learning. Springer, Boston, MA, 2011. 760-766.

[4] Eberhart, Russell C., and Yuhui Shi. "Comparison between genetic algorithms and particle swarm optimization." International conference on evolutionary programming. Springer, Berlin, Heidelberg, 1998.

[5] Khare, Anula, and Saroj Rangnekar. "A review of particle swarm optimization and its applications in solar photovoltaic system." Applied Soft Computing 13.5 (2013): 2997-3006.

[6] Banks, Alec, Jonathan Vincent, and Chukwudi Anyakoha. "A review of particle swarm optimization. Part I: background and development." Natural Computing 6.4 (2007): 467-484.

[7] Banks, Alec, Jonathan Vincent, and Chukwudi Anyakoha. "A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications." Natural Computing 7.1 (2008): 109-124.

[8] Shi, Yuhui. "Particle swarm optimization: developments, applications and resources." evolutionary computation, 2001. Proceedings of the 2001 Congress on. Vol. 1. IEEE, 2001.

[9] Poli, Riccardo, James Kennedy, and Tim Blackwell. "Particle swarm optimization." Swarm intelligence 1.1 (2007): 33-57.

[10] Jaber, Aqeel S., Abu Zaharin Ahmad, and Ahmed N. Abdalla. "A new parameters identification of single area power system based LFC using Segmentation Particle Swarm Optimization (SePSO) algorithm." Power and Energy Engineering Conference (APPEEC), 2013 IEEE PES Asia-Pacific. IEEE, 2013.

[11] Shi, Yuhui, and Russell C. Eberhart. "Empirical study of particle swarm optimization." Evolutionary computation, 1999. CEC 99. Proceedings of the 1999 congress on. Vol. 3. IEEE, 1999.