Indonesian Journal of Electrical Engineering and Computer Science
Vol. 18, No. 3, June 2020, pp. 1607~1615
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v18.i3.pp1607-1615

New deterministic initialization method for soft computing global optimization algorithms

Norazlan Hashim[1], Zainal Salam[2], Nik Fasdi Nik Ismail[3], Dalina Johari[4]
[1,3,4]Faculty of Electrical Engineering, Universiti Teknologi MARA (UiTM), Malaysia
[2]Centre of Electrical Energy Systems, School of Electrical Engineering, Universiti Teknologi Malaysia (UTM), Malaysia

Article Info

Article history:

Received Oct 30, 2019
Revised Dec 31, 2019
Accepted Jan 14, 2020

Keywords:

Deterministic initialization
Global optimization
Initialization method
Pseudo-random number
Soft computing

ABSTRACT

The initialization stage of a Soft Computing (SC) algorithm is vital as it affects the success rate of algorithms in solving multi-peak global optimization problems. The individuals in an initial population, which are known as search agents, are often generated randomly using pseudo-random number generator (PRNG) due to unavailability of prior information on the location of global peak (GP). The random nature of the generated search agents causes uneven distribution of the initial population over the search space (SS), which may lead the search towards unpromising regions from the very beginning. This paper proposes a new deterministic initialization method (DIM) for SC algorithms where search agents are evenly fixed in the SS by using a simple deterministic formulation. The performance of the proposed DIM is then compared to the conventional PRNG and more recent quasi-random number generator (QRNG). An optimization case study is carried out using two popular SC algorithms which are the Particle Swarm Algorithm (PSO) and the Evolutionary Programming (EP), and three relatively new SC algorithms which are the Whale Optimization Algorithm (WOA), the Elephant Herding Optimization (EHO), and the Butterfly Optimization Algorithm (BOA). The optimization is done on various one-dimensional (1D) benchmark functions, as well as practical problems such as partial shading condition (PSC). Simulation results show that the proposed DIM successfully improved the performance of each SC algorithm under study in solving almost all tested functions with 99% success rate compared to 88.7% and 80.2% for the QRNG and PRNG approaches respectively. Furthermore, the WOA is the most reliable and robust among the five SC techniques under study with a success rate of 93.8% when all initialization methods are employed.

Corresponding Author:

Norazlan Hashim,
Faculty of Electrical Engineering,
Universiti Teknologi MARA,
40450 Shah Alam, Malaysia.
Email: azlan4477@uitm.edu.my

1. INTRODUCTION

Soft Computing (SC) is defined as a collection of problem solving methods that fall under the field of computer science. In contrast to conventional hard computing methods, SC exploits tolerant of imprecision, partial truth, uncertainty, robustness, and low-cost solutions [1]. Nowadays, SC algorithms have been successfully applied to many different problem-solving techniques [2]. One of them is the solution to the multi-peak global optimization problem that features multiple local peaks (LPs) and one global peak (GP). In general, all SC algorithms share a common stage at the beginning of the algorithm which is population initialization [3]. In the open literature, there is limited research regarding the effects of population initialization methods on the improvement of SC algorithms. In fact, the initialization stage plays an important role in determining the success of the GP search. The purpose of initialization is to generate initial individuals known as search agents within the population. Then, the initially guessed population will be improved iteratively through an optimization process until a stopping criterion is met. The initial population is typically generated at random using uniform pseudo-random number generator (PRNG), when no a priori information about the location of global peak (GP) is available [4-6]. PRNG has been shown to generate uniformly distributed initial population [6] and cover promising regions of the search space (SS). This can be achieved by increasing the population size of the SC algorithms large enough to enable a more extensive exploration of the SP. However, larger population size requires more CPU computational time and increases the program's memory requirement for the GP search. Therefore, a small population size is preferable to accommodate low-cost hardware implementation. On the other hand, the chance of covering all promising regions of the SS reduces with small population size especially when the dimension of the SP increases. Furthermore, the initial search agents in the population is not evenly distributed over the SS, which may lead the search towards unpromising regions right from the beginning. Generally, a good initial population can facilitate SC algorithms to locate the GP [6]. On the contrary, starting from bad initial population may result in the SC algorithms getting stuck at a LP or require substantial amount of iteration/time in order to converge towards the GP.

To mitigate these problems, authors in [7, 8] proposed a quasi-random number generator (QRNG) embedded in real-coded genetic algorithm. In the literature, QRNG is described as a low-discrepancy technique [6] and designed to have a high level of uniformity in multidimensional SS. In contrast to PRNG, the points in QRNG do not imitate genuine random points. Instead, the points are designed to cover the whole SS yet maintaining a certain degree of randomness [9]. This gives the advantage of eliminating unnecessary search spaces when generating search agents for an initial population and consequently increases the chance of finding the GP. Despite the advantages of QRNG, this approach is numerically more complex than the PRNG and therefore demands more CPU time. QRNG is generally iterative which consumes a large amount of memory and requires long computational time on high-dimensional problems. Furthermore, the excellent distribution properties diminish when the dimension of the problem increases [7, 8]. These shortcomings make the QRNG approach less widely used in the domain of optimization [10]. With respect to this limitation, this paper proposes a new deterministic initialization method (DIM) for SC algorithms. It is based on a simple deterministic formulation, where search agents are evenly distributed in the SP with equal distance between one agent and the other. The proposed DIM is applied on many SC algorithms in solving various 1D benchmark functions as well as practical problems, namely partial shading condition (PSC). The performance of the proposed DIM is then compared with the conventional PRNG and more recent QRNG methods.

2. OVERVIEW OF SOFT COMPUTING (SC) ALGORITHMS

The optimization process of the population-based SC algorithms can be categorised into three major stages which are initialization, reproduction, and selection [3] as shown in Figure 1.

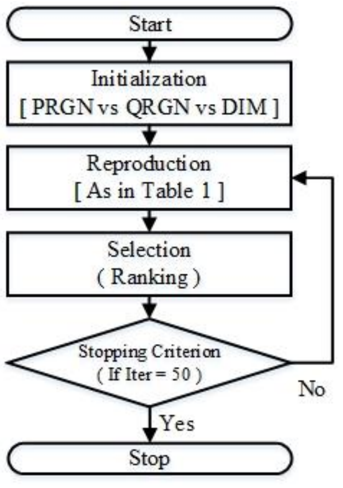


Figure 1. The general methodology for population-based SC algorithms

At the initialization stage, the initial number of search agents in the population (NP) known as parent is generated. A small NP leads to low-accuracy solutions. On the other hand, a large NP increases the computational time per iteration [11]. Therefore, a trade-off is typically made between high accuracy and reasonable computational time. In this work, the optimum value of NP is found to be 5 and are obtained by trial and error process. Hence, the NP is kept constant at 5 for all SC algorithms and problems under consideration. As mentioned earlier, the most frequently used method at this stage is random initialization using PRNG. DIM is proposed in this work to mitigate the problems encountered in the PRNG and QRNG techniques.

At the reproduction stage, a new population known as offspring is generated from the parent population through a uniquely formulated equation, according to the various distinctive SC algorithms. Five conventional SC algorithms are chosen for the optimization case study which are Particle Swarm Algorithm (PSO) [12, 13], Evolutionary Programming (EP) [14, 15], Whale Optimization Algorithm (WOA) [16], Elephant Herding Optimization (EHO) [17] and Butterfly Optimization Algorithm (BOA) [18, 19]. The reproduction equations and input parameters for each algorithm are tabulated in Table 1. Since the present study is focused on the effect of initialization towards the success of algorithms in finding the GP, tedious parameter tuning process for improving the accuracy of final solutions is avoided.

At the selection stage, a discriminatory process is conducted to select the best search agents in the population that could survive for the next generation. The main objective is to accept the good search agents and remove the bad search agents within the population while maintaining a constant population size for all generations. The most common selection methods in the literature are tournament, roulette wheel, ranking, boltzmann, elitism and steady state selection. A comparative analysis of all methods can be found in [20-23]. In the present work, the ranking selection method is chosen due to its consistency (no randomness) which was proven to produce good results. In the ranking selection method, both parent and offspring are combined together before being sorted in descending order (for maximization problems) based on their corresponding fitness function value. Next, NP best individuals (search agents) are chosen to survive in the next generation.

Finally, the reproduction and selection stages are repeated iteratively until a pre-defined stopping criterion is met. The stopping criterion is the method used to terminate the algorithm. Typically, the algorithm is terminated after the specified maximum number of generation is reached or after the accuracy of final solutions reached a pre-defined threshold value. Termination could also happen once the best solution has not changed over a specified number of generations. In this work, the algorithm is stopped when the iteration reaches the maximum number of 50 generations. This value is sufficient in order to allow all algorithms to reach the final solution. Generally, a higher maximum generation count produces more accurate final solutions though at the expense of longer computational time.

In this work, the entire simulation process is repeated by 1000 independent trials to reduce the statistical errors that arises from uncertainty of randomness which exists in the uniquely formulated reproduction equation. An algorithm is considered successful in locating the GP when the final solutions is within 5% of the true GP.

Table 1. Reproduction Equation and Input Parameters for Each Sc Algorithm

SC Algorithms	Reproduction Equation	Input Parameters
PSO	$Vel = K \cdot [Vel + C_1 \cdot rand\left(P_{best} - Parent\right) + C_2 \cdot rand\left(G_{best} - Parent\right)]$ $Offspring = Parent + Vel$	K= 0.5, C_1= 1.5, C_2= 1.5
EP	$\alpha = \beta \times \frac{\sum_{i=1}^{NP} f_i}{f_i} .* randn(0,1)$ $Offspring = Parent + \alpha$	β=0.1
WOA	$A = 2 \cdot \alpha \cdot rand - \alpha$ $C = 2 \cdot rand$ $D = \left\|C \cdot X_{best} - Parent\right\|$ $Offspring = X_{best} - A \cdot D$	$\alpha \in [2,0]$
EHO	$\Delta X = \alpha \cdot \left(X_{best} - Parent\right) \cdot rand$ $Offspring = Parent + \Delta X$ $Offspring_{best} = \beta \cdot X_{center}$	α=0.8, β=0.1
BOA	$f = cI^{\alpha}$ $\Delta X = \left(rand^2 \cdot g_{best} - Parent\right) \cdot f$ $Offspring = Parent + \Delta X$	α=0.1, c=0.01

3. THE PROPOSED INITIALIZATION METHODS

This section introduces the formulation of the proposed new deterministic initialization method (DIM). The distribution of the proposed DIM is compared with the conventional PRGN and more recent QRNG. Each method is implemented in Matlab with a lower bound (LB) of 0, upper bound (UB) of 1, problem dimension (D) of 1, and number of search agents in the population (NP) of 5. The existence of randomness in each method is observed by running 100 trials and the probability of search agents at each run is plotted in Figure 2 to Figure 4. CPU times are measured in seconds using Matlab's functions tic and toc to evaluate the complexity of each formulation.

3.1. Pseudo-Random Number Generator (PRNG)

PRNG is the most commonly used method for initialization of SC algorithms for many years. The employment of PRNG to produce uniformly random numbers within a pre-defined range within Matlab is formulated as;

```
$for \quad i = 1 : NP$
    $X(i) = LB + (UB - LB) \cdot * rand$
$end$                                                    (1)
```

where NP is the number of search agents in the population, LB and UB are the lower bound and the upper bound of the SS respectively. Meanwhile, rand is the Matlab function which returns random numbers uniformly between 0 and 1. Whether the size of SS increases or the NP decreases, (1) is unable to produce a perfectly uniform distribution of points [5]. The probability plot of the search agents in the SS for 100 trials is shown in Figure 2. For analysis purposes, the SS is divided into two sections which is the first half and the second half as illustrated in Figure 2 though in actual problems, the location of the GP is unknown and could lie within the first half or second half of the SS. Due to randomness, different search agents are generated at each trial as shown in Figure 2. On average, the probability of search agents scattered in either half of the SS is about 3%. Meanwhile, the probability for the search agents located in both sides of the SP is about 94%. This indicates that, PRNG initialization method causes approximately 3% possibility of SC algorithm getting trapped at LP. This happens when the GP exist outside of the half search area. The average CPU time taken to execute (1) is approximately 2.667 ms.

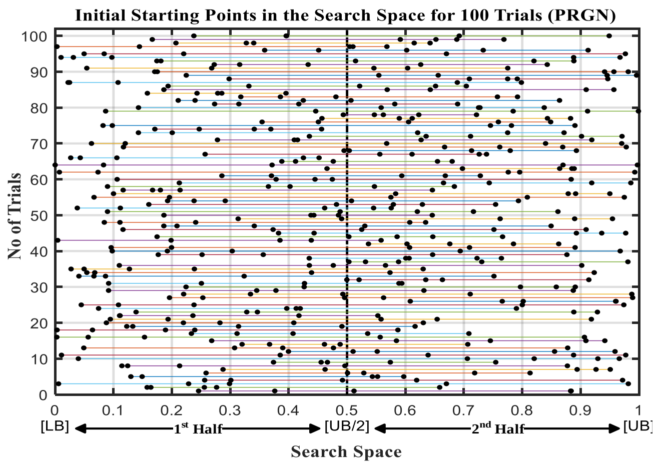


Figure 2. Probability plot of the search agents for 100 trials (PRNG)

3.2. Quasi-Random Number Generator (QRNG)

QRNG is proposed to address the issue of low uniformity of PRNG. It is designed to cover the whole search space and to produce points that maximally avoid each other while maintaining a certain degree of randomness [9]. The most common sequences are Hammersley, Halton, Sobol, Faure and Niederreiter. In this work, QRNG based on Sobol sequence is chosen for analysis due to its popularity, simplicity, and efficiency in implementation [24]. The distribution properties of a Sobol sequence are described in detail in [25, 26]. In Matlab, the formulation of Sobol sequence is given as follows;

```
for   $i = 1 : NP$
    $p = sobolset(D,'Skip',1e3,'Leap',1e2)$
    $p = scramble(p,'MatousekAffineOwen')$
    $X(i) = LB + (UB - LB) \cdot *net(p, ND)$
end                                                    (2)
```

Further details corresponding to this formulation can be found elsewhere [27]. Figure 3 shows the probability plot of search agents for 100 trials. Based on Figure 3, the search agents are scattered in both halves of the SS for all 100 trials. This results in a high probability of success when implemented by SC algorithms. However, the computation involves high CPU time of approximately 177.97 ms which reflects the complexity of the formulation. Therefore, the need for expensive memory components is inevitable, which does not allow low-cost hardware implementation.

3.3. The New Deterministic Initialization Method (DIM)

The present work proposes a simple deterministic formulation for initialization to eliminate the uncertainty caused by randomness and to reduce the equation's complexity. The formulation is presented as follows;

```
for   $i = 0 : (NP - 1)$
    $X(i+1) = \left[LB + 0.1 \cdot (UB - LB)\right] + \left[0.8 \cdot (UB - LB) / (NP - 1)\right] \cdot i$
end                                                    (3)
```

As shown in (3) is designed based on the fact that in most real-life problems, the GP lies between the lower boundary (LB) and the upper boundary (UB). Furthermore, the GP would not lie exactly at the LB and the UB line. Thus, to optimize the use of search agents which covers all regions that most likely contain the GP, the search area is limited to 90% of the SS size. By this strategy, the chance of finding the GP increases. Furthermore, to eliminate randomness, the search agents are fixed with evenly distributed between one and the other in the SS. Figure 4 presents the probability plot of the search agents produced by (3). It is observed that the same search agents are generated at each trial which excludes the lines of LB and UB where the GP is unlikely to be found. In this manner, the algorithm is avoided from moving towards unpromising regions right from the beginning. Meanwhile, a low average CPU time of about 2.243 ms indicates a simple formulation.

4. BENCHMARK AND PRACTICAL TEST FUNCTIONS

To investigate the effects of initialization methods towards the success of SC algorithms, various multi-peak benchmark functions with different SS size and real-world problems are selected in this work as shown in Figure 5. All test functions are divided into three different groups based on the location of their GP, which are at the left side of the SS (F1, F4 & F7), the middle of the SS (F2, F5 & F8), and the right side of the SS (F3, F6 & F9). Also, the GP for each function tested in this work is depicted in Figure 5. The real-world problems (F7, F8 & F9) used in this work is known as partial shading condition (PSC) [28], which is very common in photovoltaic (PV) system. It is a condition where the entire PV modules of an array do not receive the same solar irradiance (G). Under PSC, Power-Voltage (P-V) characteristic curve contains multiple local peaks (LPs) with a single global peak (GP) to operate in maximum power point of the PV system. Generally, Maximum Power Point Tracking (MPPT) algorithm that is incorporated within a DC-DC converter is implemented to maintain the operating point of the PV system at GP. To produce the functions of F7, F8, and F9 shown in Figure 5, a lookup table is employed in this work. The P-V input data for the lookup table is based on MSX-60 modules and is generated using a MATLAB/Simulink simulator developed in reference [29]. In the present work, the PV array shown in Figure 6 is partially shaded with different values of irradiance patterns as described in Table 2.

Table 2. Module Irradiance for Shading Patterns

Shading Pattern	Module Irradiance (G = 1.0 = 1000 W/m^2)					Max. Power (GP)
	G_A	G_B	G_C	G_D	G_E	
Pattern 1 (F7)	0.25	0.25	0.35	0.93	0.93	100.011 W
Pattern 2 (F8)	0.30	0.30	0.57	0.57	0.98	100.007 W
Pattern 3 (F9)	0.33	0.33	0.48	0.48	0.82	100.002 W

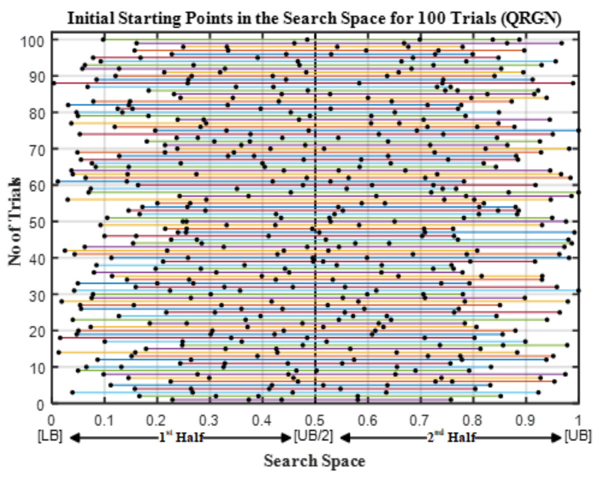


Figure 3. Probability plot of the search agents for 100 trials (QRNG)

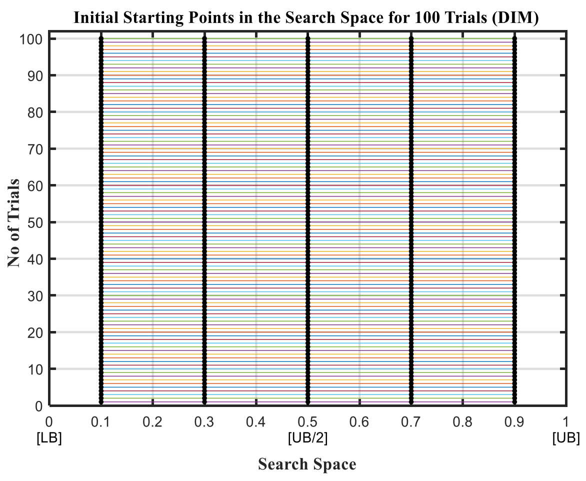
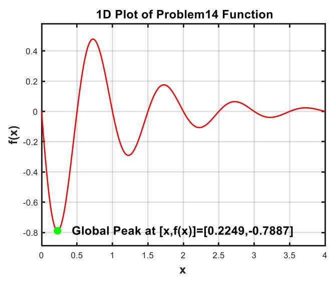


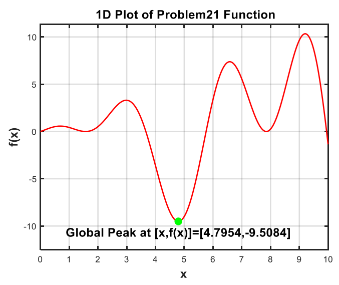
Figure 4. Probability plot of the search agents for 100 trials (DIM)

GP at the Left Sided of SP
F1: Problem14



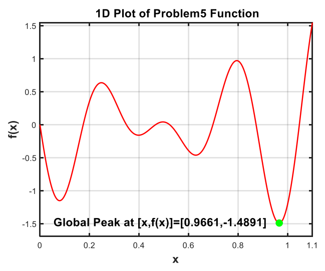
$f(x) = -e^{-x} \cdot \sin(2\pi x)$; x∈[0, 4]

GP at the middle of SP
F2: Problem21



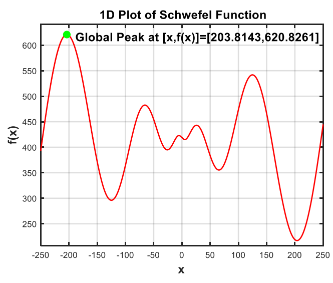
$f(x) = x \cdot sin(x) + x \cdot \cos(2x)$; x∈[0, 10]

GP at the Right Sided of SP
F3: Problem5



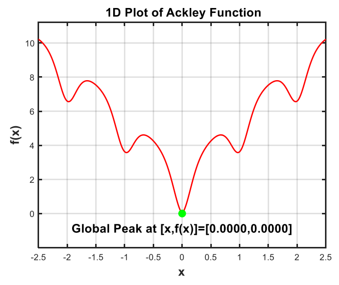
$f(x) = -(1.4 - 3x) \cdot \sin(18x)$; x∈[0, 1.1]

F4: Schwefel Function



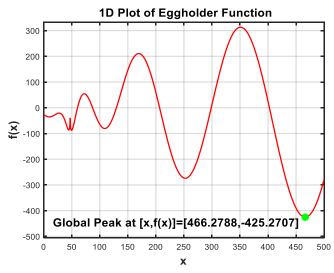
$f(x) = 418.9829 - x \cdot \sin(\sqrt{|x|})$; x∈[-250 250]

F5: Ackley Function



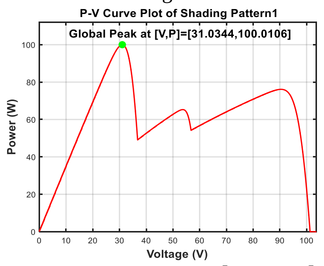
$f(x) = -20 \cdot e^{-0.2 \cdot \sqrt{x^2}} - e^{\cos(2\pi x)} + 20 + e^1$; x∈[-2.5, 2.5]

F6: Eggholder Function



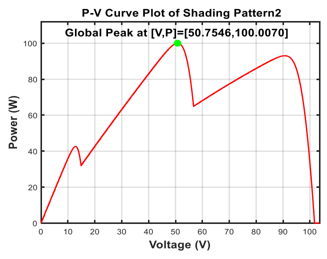
$f(x) = -47 \cdot sin\left(\sqrt{\left|\frac{x}{2} + 47\right|}\right) - x \cdot sin\left(\sqrt{|x - 47|}\right)$; x∈[0, 500]

F7: Shading Pattern 1



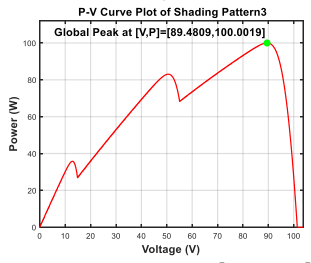
f(x) = Lookup Table; x∈[0, 101.73]

F8: Shading Pattern 2



f(x) = Lookup Table; x∈[0, 101.73]

F9: Shading Pattern 3



f(x) = Lookup Table; x∈[0, 101.73]

Figure 5. 1D Plot of Benchmark and Practical Test Functions

Table 3. Successful Optimization Trials and Successful Initializations

	Test Functions	Initialization Method	Successful of SC Algorithm on Each Test Function (out of 1000 trials)					Successful of Initialization Method (out of 5000 trials)		
			PSO	EP	WOA	EHO	BOA	PRNG (1)	QRNG (2)	DIM (3)
GP on the left side of SP	F1	PRNG - (1)	950	985	899	440	585	3859	–	–
		QRNG - (2)	986	993	997	533	785	–	4294	–
		DIM - (3)	1000	993	1000	536	1000	–	–	4529
	F4	PRNG - (1)	903	999	841	836	887	4466	–	–
		QRNG - (2)	936	1000	959	984	987	–	4866	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
	F7	PRNG - (1)	669	832	585	327	599	3012	–	–
		QRNG - (2)	756	849	741	525	699	–	3570	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
GP on the center of SP	F2	PRNG - (1)	723	491	775	645	682	3316	–	–
		QRNG - (2)	827	588	807	703	782	–	3707	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
	F5	PRNG - (1)	998	895	1000	1000	871	4764	–	–
		QRNG - (2)	1000	995	1000	1000	971	–	4966	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
	F8	PRNG - (1)	877	534	881	798	530	3620	–	–
		QRNG - (2)	922	759	986	840	830	–	4337	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
GP on the right side of SP	F3	PRNG - (1)	693	913	988	844	981	4419	–	–
		QRNG - (2)	776	968	977	870	989	–	4580	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
	F6	PRNG - (1)	746	893	888	544	856	3927	–	–
		QRNG - (2)	898	945	994	864	996	–	4697	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
	F9	PRNG - (1)	997	931	1000	988	791	4707	–	–
		QRNG - (2)	1000	999	1000	992	921	–	4912	–
		DIM - (3)	1000	1000	1000	1000	1000	–	–	5000
TOTAL			Total trials for each algorithm = 27000					Total trials for each method = 45000		
			24657^{2} (91.3 %)	24562^{3} (91 %)	25318^{1} (93.8 %)	22269^{5} (82.5 %)	237424^{4} (87.9 %)	36090^{3} (80.2 %)	39929^{2} (88.7 %)	44529^{1} (99 %)

* The overall performance rank is indicated by superscripts numbers

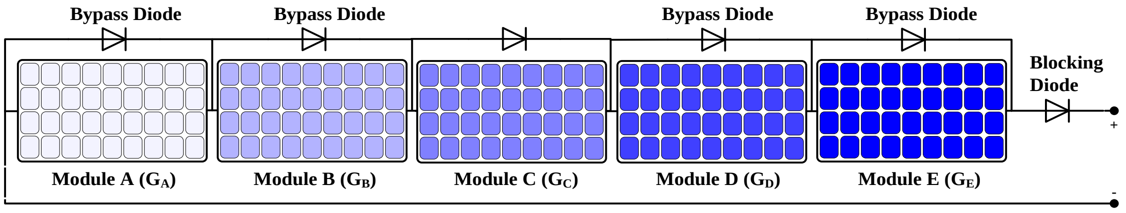


Figure 6. Schematic of partially shaded PV array with varying irradiance pattern

5. SIMULATION RESULTS AND DISCUSSIONS

Simulations are carried out on a PC running on a 64-bit Windows 10 Professional operating system with 8GB DDR3 Memory and with Intel(R) Core(TM) i5−3470 processor (3.20 GHz CPU). All algorithms are implemented using Matlab m-file code. Five SC algorithms are chosen in this work which are PSO, EP, WOA, EHO and BOA. The proposed DIM approach, together with the PRNG and the QRNG techniques are applied on all SC algorithms under study consecutively to solve each of the tested functions (F1-F9). To reduce statistical error, 1000 independent runs of each SC algorithm are performed. Successful computations of each SC algorithm and each initialization method are recorded in Table 3. The algorithm is considered successful in locating the GP when the final solutions is within 5% of the true GP. Based on Table 3, the proposed DIM is 100% successful within 5000 trials in finding GP for functions F2 to F9. For the function F1, the DIM approach recorded 4529 successes out of 5000 trials which is still a considerably high success rate. In total, the proposed DIM achieves 44529 successes out of 45000 trials (99% success rate) which is superior above the rest of the initialization methods. The second best initialization method is QRNG which achieves 39929 successes or approximately 88.7% success rate and the worst initialization method is PRNG with 360903 successful trials (about 80.2% success rate) in finding the GP.

This observation emphasizes the objective of the QRNG that aims at improving the performance of PRNG. Of all SC algorithms investigated, the WOA method produces the best performance with 25318 successful trials out of 27000 trials (93.8% success rate). This indicates that the WOA is a reliable and robust algorithm for solving various global optimization problems. The worst SC algorithm in this study is the EHO, with 22269 successes out of 27000 trials (82.5% success rate).

6. CONCLUSION

In the present study, a new but simple deterministic initialization method (DIM) is proposed to increase the success rate of various SC algorithms under study in finding the global peak (GP) within a P-V curve. DIM is designed to eliminate randomness while optimizing the use of search agents. The search domain in the DIM covers 90% of the search space region that has high possibility of locating the GP. The effectiveness of the proposed DIM is then compared with the conventional PRNG and more recent QRNG methods. Simulation results show that the proposed DIM definitely outperform both the PRNG and QRNG initialization techniques. DIM has successfully improved the performance of each SC algorithm under study in solving almost all tested functions. The success rate of the DIM is investigated using various SC algorithms including PSO, EOP, WOA, EHO, and BOA. Out of the five SC algorithms, WOA showed the highest success rate of 93.8% with 25318 successes out of 27000 trials. WOA has demonstrated the characteristics of a robust and reliable algorithm which is highly recommended to be used for solving many practical optimization problems. In order to increase the performance of other SC algorithms, it is suggested that the formulation of reproductive is enhanced to be more dynamic and effective in solving various optimization problems.

ACKNOWLEDGEMENTS

This research is supported and funded by the Institute of Quality and Knowledge Advancement (InQKA), Universiti Teknologi MARA (UiTM), Shah Alam, Selangor, Malaysia.

REFERENCES

[1] L. A. Zadeh, "Fuzzy logic, neural networks, and soft computing," *Communications of the ACM,* vol. 37, no. 3, pp. 77-85, 1994.

[2] I. Ismail and A. H. Halim, "Comparative study of meta-heuristics optimization algorithm using benchmark function," *International Journal of Electrical and Computer Engineering (IJECE),* Article vol. 7, no. 3, pp. 1643-1650, 2017.

[3] N. Hashim and Z. Salam, "Critical evaluation of soft computing methods for maximum power point tracking algorithms of photovoltaic systems," *International Journal of Power Electronics and Drive Systems (IJPEDS),* vol. 10, no. 1, p. 548, 2019.

[4] D. Bajer, G. Martinović, and J. Brest, "A population initialization method for evolutionary algorithms based on clustering and Cauchy deviates," *Expert Systems with Applications,* vol. 60, pp. 294-310, 2016.

[5] B. Kazimipour, X. Li, and A. K. Qin, "*Initialization methods for large scale global optimization,*" in 2013 IEEE Congress on Evolutionary Computation*, CEC 2013*, 2013, pp. 2750-2757.

[6] B. Kazimipour, X. Li, and A. K. Qin, "*A review of population initialization techniques for evolutionary algorithms,*" in Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014, 2014, pp. 2585-2592.

[7] H. Maaranen, K. Miettinen, and M. M. Mäkelä, "Quasi-random initial population for genetic algorithms," *Computers & Mathematics with Applications,* vol. 47, no. 12, pp. 1885-1895, 2004.

[8] W. J. Morokoff and R. E. Caflisch, "Quasi-random sequences and their discrepancies," *SIAM Journal on Scientific Computing,* vol. 15, no. 6, pp. 1251-1279, 1994.

[9] H. Maaranen, K. Miettinen, and A. Penttinen, "On initial populations of a genetic algorithm for continuous optimization problems," *Journal of Global Optimization,* vol. 37, no. 3, p. 405, 2007.

[10] K. Ding and Y. Tan, "*Comparison of random number generators in Particle Swarm Optimization algorithm,*" in Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014, 2014, pp. 2664-2671.

[11] T. Weise, Y. Wu, R. Chiong, K. Tang, and J. Lässig, "Global versus local search: the impact of population sizes on evolutionary algorithm performance," *Journal of Global Optimization,* vol. 66, no. 3, pp. 511-534, 2016.

[12] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm intelligence,* vol. 1, no. 1, pp. 33-57, 2007.

[13] J. Kennedy and R. Eberhart, "*Particle swarm optimization,*" in IEEE International Conference on Neural Networks - Conference Proceedings, 1995, vol. 4, pp. 1942-1948.

[14] J. Yuryevich and K. P. Wong, "Evolutionary programming based optimal power flow algorithm," *IEEE transactions on Power Systems,* vol. 14, no. 4, pp. 1245-1250, 1999.

[15] N. Hashim, Z. Salam, and S. M. Ayob, "*Maximum Power Point Tracking for stand-alone Photovoltaic system using Evolutionary Programming,*" in Proceedings of the 2014 IEEE 8th International Power Engineering and Optimization Conference, PEOCO 2014, 2014, pp. 7-12.

[16] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in engineering software,* vol. 95, pp. 51-67, 2016.

[17] G. G. Wang, S. Deb, and L. D. S. Coelho, "*Elephant Herding Optimization,*" in Proceedings-2015 3rd International Symposium on Computational and Business Intelligence, ISCBI 2015, 2016, pp. 1-5.

[18] S. Arora and S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," *Soft Computing,* vol. 23, no. 3, pp. 715-734, 2019.

[19] M. Ghetas, C. H. Yong, and P. Sumari, "*Harmony-based monarch butterfly optimization algorithm,*" in Proceedings - 5th IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2015, 2016, pp. 156-161.

[20] S. Gandhi, D. Khan, and V. S. Solanki, "A comparative analysis of selection scheme," *International Journal of Soft Computing and Engineering,* vol. 2, no. 4, pp. 131-134, 2012.

[21] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," in *Foundations of genetic algorithms,* vol. 1: Elsevier, 1991, pp. 69-93.

[22] S. L. Yadav and A. Sohal, "Comparative study of different selection techniques in genetic algorithm," *Journal Homepage: http://www. ijesm. co. in,* vol. 6, no. 3, 2017.

[23] P. J. B. Hancock, "An empirical comparison of selection methods in evolutionary algorithms," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 865 LNCS, ed, 1994, pp. 80-94.

[24] B. Kazimipour, X. Li, and A. K. Qin, "*Effects of population initialization on differential evolution for large scale optimization,*" in Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014, 2014, pp. 2404-2411.

[25] N. A. Mohammed, "Comparing Halton and Sobol Sequences in Integral Evaluation," *ZANCO Journal of Pure and Applied Sciences,* vol. 31, no. 1, pp. 32-39, 2019.

[26] G. Levy, "An introduction to quasi-random numbers," *Numerical Algorithms Group Ltd., http://www. nag. co. uk/IndustryArticles/introduction_to_quasi_random_numbers. pdf (last accessed in April 10, 2012),* p. 143, 2002.

[27] "https://au.mathworks.com/help/stats/sobolset.html."

[28] Z. Salam, Z. Ramli, J. Ahmed, and M. Amjad, "Partial shading in building integrated PV system: Causes, effects and mitigating techniques," *International Journal of Power Electronics and Drive Systems (IJPEDS),* Article vol. 6, no. 4, pp. 712-722, 2015.

[29] K. Ishaque, Z. Salam, and Syafaruddin, "A comprehensive MATLAB Simulink PV system simulator with partial shading capability based on two-diode model," *Solar Energy,* Article vol. 85, no. 9, pp. 2217-2227, 2011.