algorithms

MDPI

Article

Pendulum Search Algorithm: An Optimization Algorithm Based on Simple Harmonic Motion and Its Application for a Vaccine Distribution Problem

Nor Azlina Ab. Aziz [1,*] and Kamarulzaman Ab. Aziz [2]

[1] Faculty of Engineering & Technology, Multimedia University, Melaka 75450, Malaysia
[2] Faculty of Business, Multimedia University, Melaka 75450, Malaysia; kamarulzaman.aziz@mmu.edu.my
* Correspondence: azlina.aziz@mmu.edu.my

Abstract: The harmonic motion of pendulum swinging centered at a pivot point is mimicked in this work. The harmonic motion's amplitude at both side of the pivot are equal, damped, and decreased with time. This behavior is mimicked by the agents of the pendulum search algorithm (PSA) to move and look for an optimization solution within a search area. The high amplitude at the beginning encourages exploration and expands the search area while the small amplitude towards the end encourages fine-tuning and exploitation. PSA is applied for a vaccine distribution problem. The extended SEIR model of Hong Kong's 2009 H1N1 influenza epidemic is adopted here. The results show that PSA is able to generate a good solution that is able to minimize the total infection better than several other methods. PSA is also tested using 13 multimodal functions from the CEC2014 benchmark function. To optimize multimodal functions, an algorithm must be able to avoid premature convergence and escape from local optima traps. Hence, the functions are chosen to validate the algorithm as a robust metaheuristic optimizer. PSA is found to be able to provide low error values. PSA is then benchmarked with the state-of-the-art particle swarm optimization (PSO) and sine cosine algorithm (SCA). PSA is better than PSO and SCA in a greater number of test functions; these positive results show the potential of PSA.

Keywords: harmonic motion; pendulum; optimization; search

Citation: Ab. Aziz, N.A.; Ab. Aziz, K. Pendulum Search Algorithm: An Optimization Algorithm Based on Simple Harmonic Motion and Its Application for a Vaccine Distribution Problem. *Algorithms* **2022**, *15*, 214. https://doi.org/10.3390/a15060214

Academic Editor: Lorenzo Salas-Morera

Received: 15 May 2022
Accepted: 13 June 2022
Published: 17 June 2022

1. Introduction

Optimization problems occur in all branches of studies, from engineering to health science to logistics planning to computer science to finance, and many others. Optimization problems often involve maximization of gain or/and minimization of lost with respect to some constraints. Many of these problems are complex and solving them using exact optimization algorithms is impractical due to the computational cost and time taken.

Metaheuristics provide the answer of this problem. Metaheuristics are approximation algorithms that provide optimal or near-optimal solutions within reasonable time and computational constraints. Metaheuristics are general purpose and can be adapted to various problems.

Much research has been conducted in this field and many algorithms have been proposed. The majority of the algorithms proposed are nature-inspired. In fact, the state of the art and most researched algorithms–namely the genetic algorithm (GA) [1], ant colony optimization (ACO) algorithm [2], and particle swarm optimization (PSO) [3]–are all inspired by nature. The GA is inspired by genetic evolution like the selection of the fittest, mutation, and crossover to generate a new and superior generation. Meanwhile, the ACO is inspired by the foraging behavior of ants where a trail of pheromones is left as information for the colony's members. PSO, on the other hand, is inspired by the social behavior observed among flock of birds, swarm of bees, and school of fishes. PSO mimics how the cognitive and social factors influence the individuals' decision.

Although many metaheuristic algorithms have been proposed in recent years–for example the sine cosine algorithm (SCA) [4], gravitational search algorithm (GSA) [5], bat algorithm (BA) [6], artificial bee colony (ABC) [7], grey wolf optimization (GWO) [8], and many others–the no free lunch theorem (NFL) motivates researchers to keep introducing new algorithms or improving those existing. NFL states that no supreme algorithm that can provide the best solution for all optimization problems exists. An algorithm might be able to give the best solution for a set of problems but not for another set.

In this work, a new metaheuristic named the Pendulum Search Algorithm (PSA) is proposed. PSA is a population-based algorithm where the solution of an optimization problem is searched for by a group of agents. The agents in PSA move around the search space looking for the optimal solution according to pendulum harmonic motion. A pendulum centered at a pivot point swings in a harmonic motion, with the amplitude of the swing decreasing with time. The proposed algorithm is validated using 13 multimodal test functions and benchmarked with PSO and SCA. The findings show that PSA is a good optimization algorithm, outperforming PSO in 8 out of 13 functions and 12 out of 13 for SCA. PSA is then applied for the optimization of vaccine distribution using the case of Hong Kong's 2009 H1N1 influenza endemic. The distribution percentage found by PSA is better in lowering the number of infections when compared with three traditional strategies usually used in health science.

This paper is divided into five sections. The following section investigates existing algorithms with similar framework. The PSA is introduced in Section 3. Section 4 presents the experiment conducted including details of the H1N1′s extended SEIR and the findings obtained. Finally, the work is concluded in Section 5.

2. Related Works

Metaheuristic algorithms are iterative procedure where search agents repeatedly evaluate and improve their solution in each iteration. The success of a metaheuristic algorithm depends on the ability of the search agents to explore or expand their search area and to exploit or fine-tune the search around the current search area [9]. Many research show that exploration is desired at the early stage of the iterative procedure while exploitation is important towards the end of the iteration [10]. Exploration widens the search area of an agents so that a region containing an optimal solution is identified, whereas exploitation narrows down the search within the identified region so that the location with an optimal solution is found.

A time decreasing inertia weight is an example of a mechanism frequently adopted by researchers to control agents' behaviour to favour exploration at the start of the search before switching to exploitation [11–13]. The time decreasing inertia weight reduces the step size as the iteration increases. Among the common patterns applied by researchers are linearly decreasing and exponentially decreasing inertia weight [12].

Metaheuristic search agents often face the challenge of premature convergence where the agents are trapped in local optima [14]. In [15], crossover operation is introduced to the cat swarm optimization algorithm to overcome the local optima trap and increase diversity. Meanwhile, in [16] a simple re-initialization method is proposed for PSO. Mutation is also a popular method to avoid premature convergence, where various mutation strategies have been adopted by researchers [17,18]. Overall, all these strategies are introduced to create a disturbance to the agents' convergence so that the premature convergence can be avoided.

The most relevant algorithm to this work is SCA. Mirjalili adopted the sine cosine function to guide the search for the optimal solution by fluctuating around the best solution. To ensure convergence, SCA envelopes the sine and cosine function using a linearly decreasing function mirrored by the time axis. However, despite the fluctuation and linearly decreasing envelope, both [19,20] listed premature convergence as the drawback of SCA. Additionally, [21] highlighted that the solution update equations of SCA result in a bias towards the origin, causing SCA to work very well for optimization problems where the

global solution is located at the origin. The performance declines with shifted problems. Three random numbers are needed in SCA.

In this work, a physical phenomenon–namely the harmonic motion of a pendulum–is mimicked to move the search agents so that an optimal solution is achieved. Unlike SCA, the harmonic motion of the pendulum slows down with exponential function. Exponential function had been observed to provide good exploration–exploitation balance in metaheuristics [22,23]. The PSA agents do not randomly select between sine and cosine functions; rather, they follow the harmonic motion function. The harmonic motion function determines the maximum limit for the random numbers that controls the PSA agents' stochastic search, with respect to current solution and best solution.

3. Pendulum Search Algorithm

3.1. Source of Inspiration

An idealized pendulum swings left and right with equal height endlessly. However, this is unrealistic, as fluid drag causes by air dampens the pendulum's movement, which eventually reach to a standstill [24].

Pendulum damped harmonic motion is the inspiration for PSA. The weight swings back and forth, while the oscillation amplitude declines with time until equilibrium is reached. The air resistance dampens and slows down the pendulum motion. Figure 1 illustrate a swinging pendulum hanged by a string and its typical harmonic oscillation. The harmonic oscillation equation is also shown in the figure. The maximum displacement from the equilibrium is represented as A, the angular frequency is ω, and φ is the initial phase.

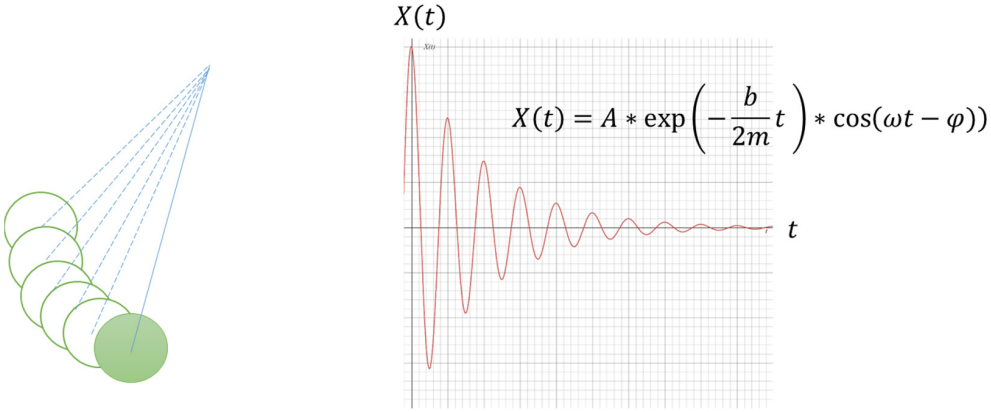

Figure 1. Pendulum Harmonic Motion.

The pendulum harmonic motion is chosen here to control the search step of PSA agents due to the oscillating pattern observed. This pattern that expands and contracts alternately while decreasing the amplitude throughout the time is desirable. This behaviour is predicted to support agents' exploration at the beginning, and to be balanced by exploitation as the search progress.

3.2. The Algorithm

PSA is a population-based metaheuristic, where the search for the optimal solution is driven by a group of agents. In PSA, each of the agents acts like a pendulum with independent parameters. They move around the search space looking for the optimal solution, driven by the pendulum movement that centres around their own current position.

The agent's position updated equation is shown in Equation (1):

$$P_i^d(t) = P_i^d(t-1) + pend_i^d(t)\left(Best^d - P_i^d(t-1)\right) \quad (1)$$

where $P_i^d(t)$ represents position of agent ith at dimension dth for time tth, while $Best^d$ is the best solution found so far since the start of the search by the entire population. The agent moves towards or away from the best solution based on $pend_i^d(t)$, which is a random number with maximum amplitude oscillates obeying the pendulum harmonic equation as illustrated in Figure 2. Different agents have a different $pend_i^d(t)$ value. The $pend_i^d(t)$ is calculated according to Equation (2).

$$pend_i^d(t) = 2\exp(-t/tmax)(\cos(2\pi * rand)) \quad (2)$$

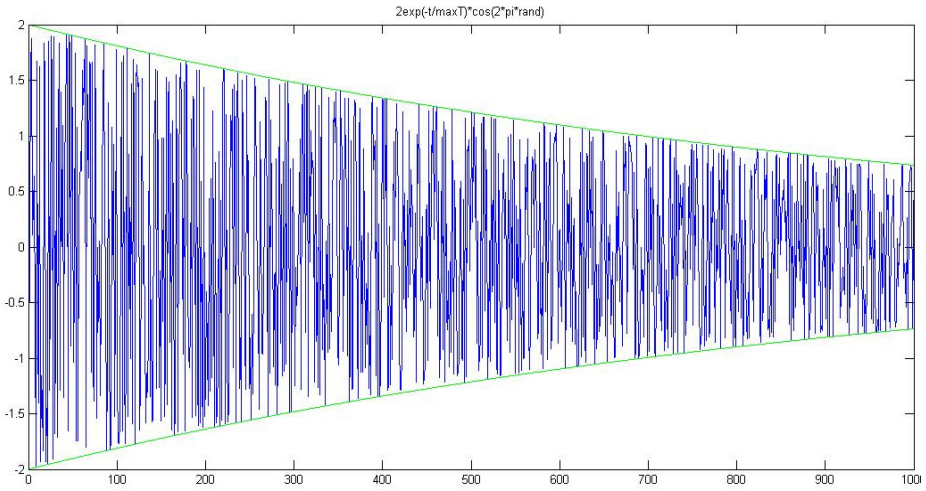

Figure 2. $pend_i^d(t)$ Magnitude of Oscillation.

This equation mimics the harmonic equation, $tmax$ is the maximum number of iterations, and $rand$ is a random number between 0 to 1.0 drawn from uniform distribution. The maximum displacement amplitude 2 is chosen so that exploration is favoured during the early iterations. Once the $pend_i^d(t)$ is below 1, the agents switch to favour exploitation. Additionally, $pend_i^d(t)$ oscillates between the positive and negative range. A positive value $pend_i^d(t)$ encourages an agent to move towards the direction of the best solution, while a negative $pend_i^d(t)$ drives the agents towards the opposite direction from the best solution.

There are only two simple mathematical equations employed in PSA with two parameters to be selected, namely the number of agents and maximum number of iterations. This makes PSA a very simple optimization algorithm with low computational cost. The pseudocode of PSA is shown in Algorithm 1.

Algorithm 1 Pseudocode of PSA

```
Initialize the agents' parameters and positions randomly.
For i = 1: maximum iteration
    For each agent
        Update agents using Equation (1) & (2)
        Evaluate agent's fitness
    End
    Identify the best agent
End
Solution: best agent
```

4. Experiment, Results & Discussion

4.1. Optimization of Benchmark Problems

The performance of the proposed PSA is studied using 13 multimodal functions from the CEC2014 benchmark test suite, function 4–16. These functions are minimization problems that are shifted and rotated from their original basic functions. The shift and

rotation avoids the issue of origin bias. Multimodal functions have many local optima and one ultimate global optimum. Multimodal functions are suitable to observe whether an algorithm is able to perform and avoid premature convergence or not, as well as observing the algorithm's exploration and exploitation ability. Therefore, only the 14 multimodal functions are adopted here. The names and ideal fitness of the functions are listed in Table 1; other details of the 13 benchmark functions such as their mathematical functions can be found in [25]. In this work, the dimension of the functions is set to 10.

Table 1. List of Benchmark Functions.

	Function Name	Ideal Fitness
f4	Shifted and Rotated Rosenbrock's Function	400
f5	Shifted and Rotated Ackley's Function	500
f6	Shifted and Rotated Weierstrass Function	600
f7	Shifted and Rotated Griewank's Function	700
f8	Shifted Rastrigin's Function	800
f9	Shifted and Rotated Rastrigin's Function	900
f10	Shifted Schwefel's Function	1000
f11	Shifted and Rotated Schwefel's Function	1100
f12	Shifted and Rotated Katsuura Function	1200
f13	Shifted and Rotated HappyCat Function	1300
f14	Shifted and Rotated HGBat Function	1400
f15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
f16	Shifted and Rotated Expanded Scaffer's F6 Function	1600

4.1.1. Number of Agents

There are only two parameters to be selected for PSA. The first is the number of iterations. This is related to the computational ability of the computing platform and the complexity of the problems. Hence, for this experiment the number of iteration is set to 1000. The second parameter is the number of agents. In this section the effect of the number of agents is investigated. The number of agents tested are 10, 20, 30, 40, and 50. All the settings are run 30 times; the best results of the run are tabulated in Table 2. Since the benchmark functions are minimization problems, the best results are the minimum value from the 30 runs of each setting. It can be seen that there is no one number of agents that works best for all test functions. However, the number of agents equal to 50 obtained a higher number of best results compared to the others.

The Friedman signed rank test of 1 × N is then conducted to identify the best number of agents using this set of benchmark functions. The Friedman signed rank test is a statistical method commonly used to provide an unbiased analysis of metaheuristics' performance [26–28]. The lower the Friedman rank of an algorithm, the better the algorithm is. A 1 × N test compares the best-ranked algorithm with the other algorithms. If the Friedman statistic value–which is distributed according to χ^2 with $k-1$ degrees of freedom, where k is the number of test (here $k = 5$, i.e.,: 5 number of agents tested)–has p-value of lesser than the significance level adopted, α, then the null hypothesis that states that all settings tested are on par with each other is rejected. In order to identify the significant difference, a post hoc procedure known as the Holm post hoc procedure is adopted. In this work, $\alpha = 0.05$ is adopted. Other than 0.05, 0.1

is also commonly used by researchers. The smaller the value, the stricter the statistical test in accepting the null hypothesis.

Table 2. Effect of Number of Agents.

	Number of Agents				
Function Name	**10**	**20**	**30**	**40**	**50**
f4	400.0557	400.0003	400.001	400.0008	400.0013
f5	519.997	519.996	519.9923	519.983	519.9913
f6	601.5922	601.0678	600.3217	600.3587	600.0771
f7	700.0595	700.027	700.0418	700.0591	700.0517
f8	801.9904	800	800	800	800
f9	905.9708	905.9698	904.9748	905.9698	904.9748
f10	1007.031	1000.375	1000.25	1003.602	1000.25
f11	1247.431	1226.78	1140.238	1222.493	1131.949
f12	1200.059	1200.006	1200.012	1200.01	1200.017
f13	1300.138	1300.132	1300.1	1300.043	1300.137
f14	1400.171	1400.144	1400.123	1400.079	1400.052
f15	1500.94	1500.575	1500.499	1500.578	1500.417
f16	1602.263	1602.072	1602.032	1601.511	1601.108
Friedman Rank	5	2.9615	2.3462	2.5769	2.1154

In this work, the statistical test is carried using Knowledge Extraction based on Evolutionary Learning (KEEL), which is an open-source tool with statistical analysis modules [26]. KEEL is a user friendly software developed by a group of researchers for research and academic purposes [26–28].

The Friedman ranks are listed in the last row of Table 2. The Friedman statistical value (distributed according to χ^2 with 4 degrees of freedom) is 28.030769 and its p-value is 0.000012, which is less than a significance level of 0.05. Therefore, a significant difference exists among the five settings. The Holm post-hoc is used to determine which setting is on par or worse than the best-ranked number of agents. Table 3 shows the p-value and Holm value of the post hoc test. The Holm's procedure rejects those hypotheses that have a p-value < Holm. Hence, a number of agents equal to 10 is significantly worse than 50, and the others are on par with each other. Thus, it can be concluded that a number of agents above 20 is recommended for PSA.

Table 3. Holm Post Hoc for Number of Agents.

i	Number of Agents	p	Holm
4	10	0.000003	0.0125
3	20	0.172447	0.016667
2	40	0.45675	0.025
1	30	0.709815	0.05

Figure 3 further illustrates the results of the experiment. The results are converted into error values for better graph representation. The error value is the difference between the fitness of the solution found with the ideal fitness. At the top left of the figure is the close-up of the graph. Similar to what is observed in Table 2, it can be seen that the number of agent equal to 50 gives the least error in most of the functions, but not in all functions.

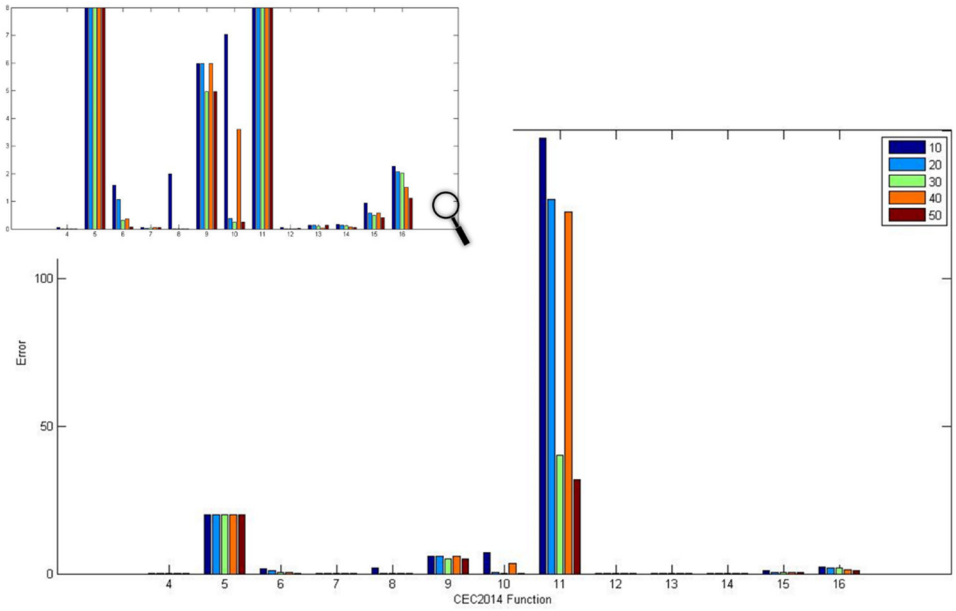

Figure 3. Fitness error of the benchmark functions with different number of agents.

4.1.2. PSA vs PSO and SCA

The PSA is benchmarked with PSO and SCA. PSO is selected due to its reputation as the state-of-the-art metaheuristic algorithm. Additionally, like PSA, PSO also only has two mathematical equations to be updated at each iteration. Thus, both algorithms have similar computational complexity. The SCA is chosen due to its similarity with PSA as discussed before. Based on the findings from the previous section, all three algorithms are run with 50 agents and for 1000 iterations. Since the algorithms are stochastic algorithms with randomness incorporated into them, the algorithms are run 30 times. In addition to the best result found–i.e., the minimum value; the lower the value, the better the result–the maximum, mean and standard deviation of the algorithms are recorded and tabulated in Table 4, and their convergence curves are presented in Figure 4. Similar to the last section, statistical analysis is conducted using the best results.

Table 4. Performance of PSA vs PSO and SCA.

	PSA				PSO				SCA			
	Min	**Max**	**Mean**	**Std Dev**	**Min**	**Max**	**Mean**	**Std Dev**	**Min**	**Max**	**Mean**	**Std Dev**
f4	400.01	438.72	421.30	17.32	400.15	442.94	429.58	16.54	431.09	490.65	458.46	14.61
f5	519.83	520.00	519.99	0.03	520.11	520.45	520.29	0.08	517.81	520.56	520.32	0.48
f6	600.03	606.29	602.33	1.46	600.00	606.87	601.35	1.51	604.51	609.45	606.50	1.30
f7	700.03	700.66	700.22	0.16	700.03	700.23	700.13	0.06	706.04	718.41	711.08	3.57
f8	800.00	802.98	800.83	0.79	800.99	803.98	802.50	0.96	827.23	855.40	839.57	7.99
f9	902.98	924.87	911.18	5.96	903.98	916.47	909.30	3.21	935.95	957.31	943.53	6.08
f10	1003.41	1140.72	1026.81	40.76	1003.72	1365.82	1133.64	113.59	1744.35	2388.44	1994.61	173.71
f11	1225.66	2166.99	1574.51	214.47	1106.89	1881.69	1482.65	198.60	1658.50	2876.79	2400.37	279.23
f12	1200.03	1200.24	1200.11	0.06	1200.08	1201.40	1200.61	0.38	1200.91	1201.72	1201.30	0.19
f13	1300.13	1300.48	1300.29	0.11	1300.06	1300.22	1300.13	0.04	1300.50	1300.84	1300.62	0.08
f14	1400.07	1400.84	1400.32	0.20	1400.07	1400.32	1400.15	0.06	1400.38	1401.52	1400.92	0.34
f15	1500.45	1503.31	1501.54	0.76	1500.49	1502.69	1501.16	0.45	1505.01	1513.12	1507.42	1.64
f16	1601.24	1603.22	1602.39	0.56	1600.74	1603.14	1602.15	0.48	1602.72	1603.79	1603.37	0.22
Friedman Rank	1.3846				1.7692				2.8462			

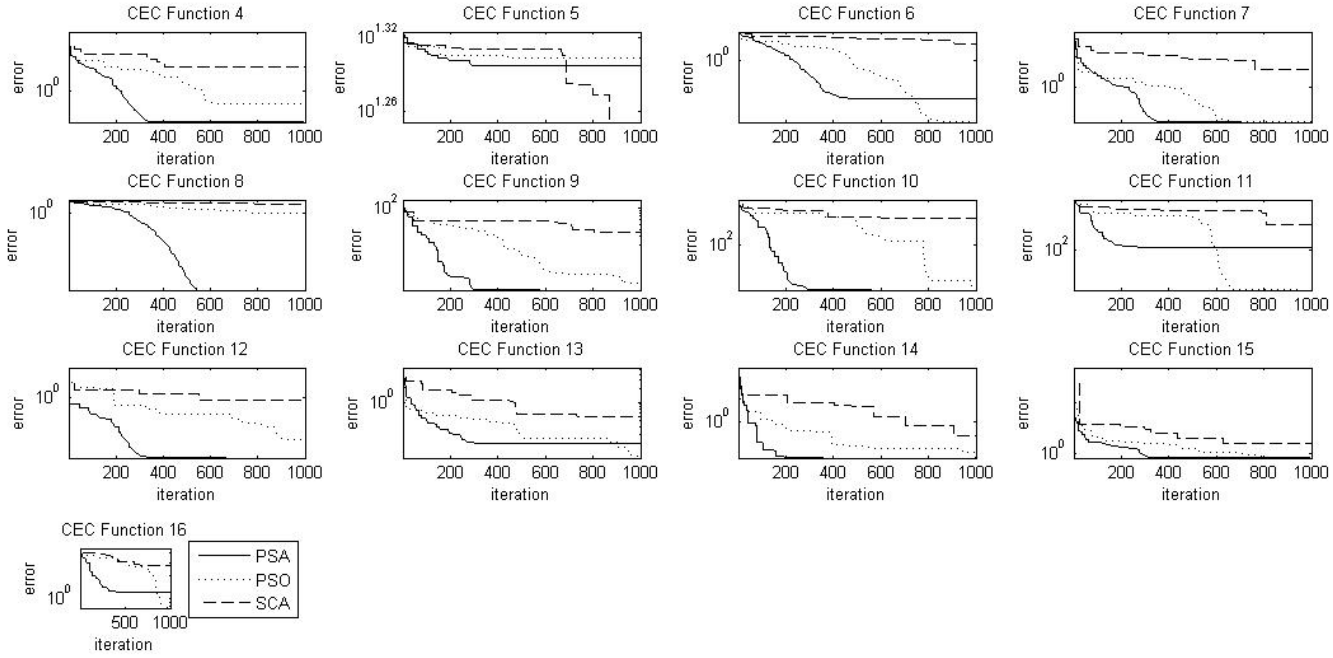

Figure 4. Convergence Curves.

The results from the 13 functions found that the best solution of PSA is better than PSO and SCA in 8 functions. This is followed by PSO (= 4) and SCA (= 1). The Friedman test ranked PSA the best with the lowest average rank; this is followed by PSO and SCA. The ranks are listed in the last row of Table 4. Friedman statistic (distributed according to χ^2 with 2 degrees of freedom) is obtained at 14.923077 and the p-value computed by the Friedman Test is 0.000575. Hence, with a significance level of 0.05, it is found that there is significant different between the algorithms. Further analysis with Holm post hoc shows that PSA is on par with PSO and significantly better than SCA. This is indicated by the p-value and Holm value in Table 5.

Table 5. Holm Post Hoc for Number of Agents.

i	Algorithm	*p*	Holm
2	SCA	0.000194	0.025
1	PSO	0.3268	0.05

The standard deviation of PSA is less than 1 for 8 out of the 13 test functions. Low standard deviation shows the consistency of the algorithm's performance. This is also seen for the state-of-the-art PSO. On the other hand, SCA has a greater number of standard deviations bigger than 1. The same findings are observed through the small difference between the maximum and minimum solution of PSA.

The convergence curves of PSA show that PSA converges gradually at the beginning and found its best solution faster than PSO and SCA. This pattern shows that further improvement focusing on enhancing agents' exploration is needed

4.2. PSA for Vaccine Distribution Optimization

During a pandemic, an efficient vaccine distribution strategy is important to ensure that objectives such as achieving heard immunity and lowering the transmissibility rate is achieved. As observed during the current COVID-19 pandemic as well as the 2009 H1N1 outbreak, vaccination is an important measure to combat the outbreak of infectious diseases. However, vaccine production is frequently costly and time-consuming, causing limited supply [29]. Therefore, the distribution of vaccines to susceptible persons must be optimized in order to achieve the optimum effects [30].

In this work, we proposed the application of PSA for vaccine distribution optimization. The 2009 Hong Kong's H1N1 epidemic [31] is used here as the case study. In [31], an extended SEIR model with vaccination (SEIR-V) is proposed. The SEIR model where, S—susceptible, E—exposed, I—infected, and R—recovered, is a compartmental mathematical model frequently adopted to model dynamics of infectious diseases. The SEIR model can be extended to include other aspects such as vaccination effect (V), death (D), and hospitalization (H). The model adopted here characterizes the infection dynamics according to five age groups and the contact rate of the individuals. The SEIR-V model is represented by nonlinear differential Equations (3)–(7).

$$\frac{dS_g}{dt} = (-\lambda_g)\cdot(S_g - \Delta v_g) + (-\Delta v_g) \tag{3}$$

$$\frac{dE_g}{dt} = (-\gamma)\cdot E_g + \lambda_g\cdot(S_g - \Delta v_g) \tag{4}$$

$$\frac{dI_g}{dt} = (-\tau)\cdot I_g + \gamma\cdot E_g \tag{5}$$

$$\frac{dR_g}{dt} = \tau\cdot I_g \tag{6}$$

$$\frac{dV_g}{dt} = \Delta v_g \tag{7}$$

The subscript g in the equations represents age group, where $g = \{1,2,3,4,5\}$. The population is divided to five age groups as follows; $A_1(5-14)$, $A_2(15-24)$, $A_3(25-44)$, $A_4(45-64)$, and $A_5(65+)$. Children between the age of 0–4 are not included as they do not have independent social contact. The contact frequency matrix, $C = \{c_{g1,g2} | g1, g2 \in (1,5)\}$, between individuals from different and same age groups is presented in Table 6. The numbers are decided based on the work conducted in [32].

Table 6. Contact frequency matrix.

	A_1	A_2	A_3	A_4	A_5
A_1	8.27	1.395	4.165	1.51	0.715
A_2	1.395	5.65	2.385	1.83	0.895
A_3	4.167	2.385	6.55	3.425	1.383
A_4	1.51	1.83	3.425	4.2	2.055
A_5	0.715	0.895	1.383	2.055	2.66

In Equation (3), (4), and (7), Δv_g is the amount of vaccine released in the current step. Meanwhile, λ_g represents infection risk of the susceptible individuals in group A_g which is formulated in Equation (8).

$$\lambda g = \frac{1}{5}\cdot\left(\sum_{j=1}^{5}\left(c_{g,\,g2}\cdot\frac{Ig2}{Pg2}\right)\right)\cdot\frac{Sg}{Pg}\cdot\beta g \tag{8}$$

The size of population according to age group is represented by Pg while βg is the age-based infection rate. γ and τ represent the incubation and recovery rate. These parameters are set according to the setting of [31], which is shown in Table 7.

The objective of the PSA is to minimize the total infections by minimizing the peak of the infection curve. The position of PSA's agent, P_i^g, represents the distribution percentage of the vaccine according to age group, g. The dimension of the agent follows the number of age group. The objective function is as shown in Equation (9).

Table 7. Parameters of 2009 Hong Kong's H1N1 epidemic.

Age Group	Population P_i	Infection Rate β_i	Incubation Rate γ	Recovery Rate τ
A_1	0.94 m	0.434	0.25	0.334
A_2	0.94 m	0.158	0.25	0.334
A_3	2.30 m	0.118	0.25	0.334
A_4	1.86 m	0.046	0.25	0.334
A_5	0.85 m	0.046	0.25	0.334

$$\begin{gathered} fit_i = min\left(\sum_{i=1}^{5} I_g(peak)\right) \\ \text{subject to : } \sum_{g=1}^{5} V_g = V_{max}, \quad V_g = P_i^g \times V_{max} \\ \sum_{g=1}^{5} P_i^g = 1 \\ 1 \leq peak \leq end_{ob} \end{gathered} \tag{9}$$

where fit_i is fitness of agent i, $I_g(peak)$ is the peak of infection number from the start of the outbreak to the end of the outbreak, end_{ob}. V_{max} is the total number of vaccines available. The infection number is calculated using Equation (5). Since day of the outbreak, t, is a discrete value ($t \in (1, end_{ob})$), the differential equation is approached using the discrete calculus method, where the I is calculated using summation.

The performance of the PSA in solving the vaccine distribution problem is compared with the three traditional strategies implemented in [31], which are based on transmissibility (S1), vulnerability (S2), and infection risk (S3). Three conditions are used here, where the number of vaccines is set to small, medium, and large quantity which are 5%, 10%, and 20% of the total population. The vaccine is assumed to be given during the infection dispersing stage, which is at day 50. The duration of the outbreak is end_{ob} = 300 days and the initial exposed individual are 30 from A_2. The experimental parameter settings are tabulated in Table 8.

Table 8. Experimental parameter settings.

Parameter	Value
No. of vaccine	(0.3 m = 5%, 0.6 m = 10%, 1.2 m = 20%)
Administered day	50
Outbreak duration	300
Initial exposed individuals $\{E_1E_2,\ E_3,\ E_4,\ E_5\}$	{0, 30, 0, 0, 0}

The results of vaccine distribution using PSA are illustrated in Figures 5–7. The results show that for small and medium vaccination coverage, PSA can find much better vaccination distribution percentages that the other three traditional strategies. The peaks of the infection outbreak are lowest using the percentage determined by PSA compared to the other strategy. Meanwhile, when the number of vaccines is large, the performance of PSA, S2, and S3 are close to one another.

Overall, the findings show that PSA is a good optimization algorithm that can solve a complex nonlinear differential problem like the vaccination distribution problem. In addition, it is observed that increasing the number of vaccines lowers and flattens the infection curve faster.

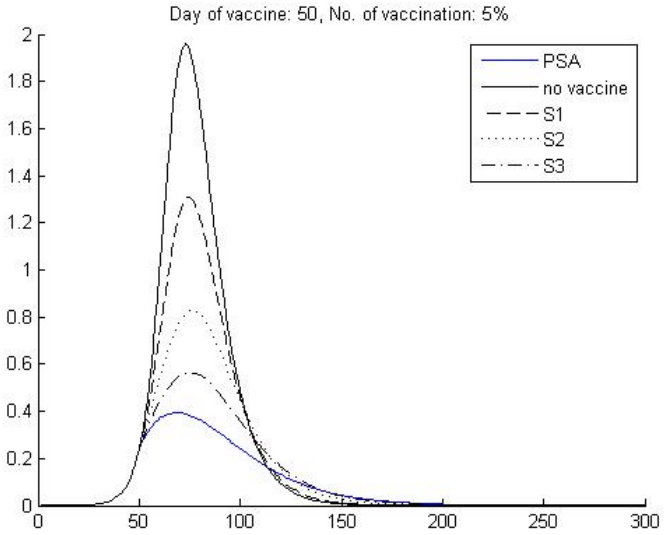

Figure 5. Infection dynamic for 5% vaccination coverage.

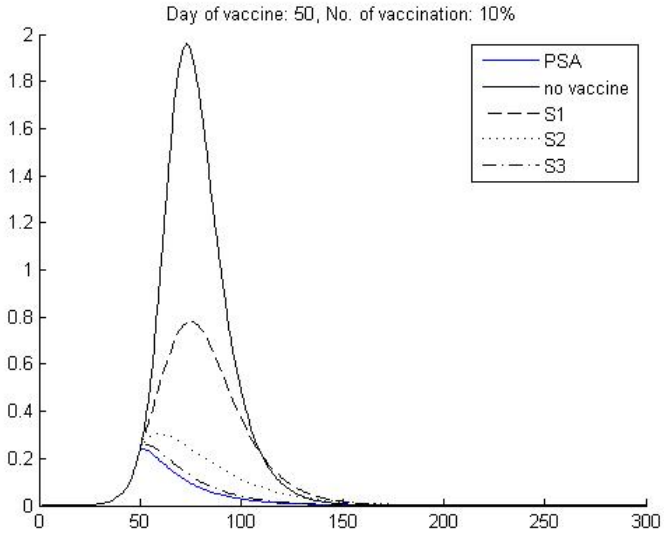

Figure 6. Infection dynamic for 10% vaccination coverage.

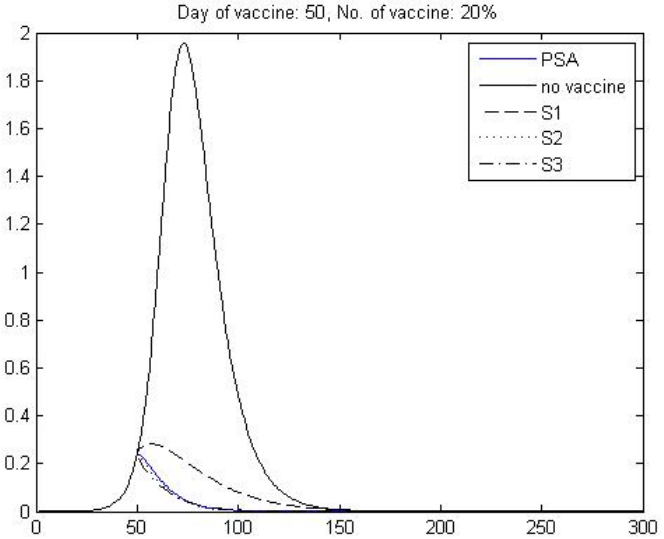

Figure 7. Infection dynamic for 20% vaccination coverage.

5. Conclusions

A new metaheuristic algorithm, PSA, is proposed in this work. PSA is a population-based algorithm where the agents move in the search space by mimicking pendulum movement. The performance of the algorithm is studied using 13 multimodal mathematical functions. The findings found that the algorithm is able to look for a good solution and performs as well as the state-of-the-art PSO and significantly better than SCA. The PSA is then applied to the vaccine distribution optimization problem. Three conditions are tested and the results of PSA are better than that of traditional strategies. Nonetheless, convergence curves show that PSA converges faster than PSO. This contributes to PSO obtaining better results in the functions where it outperformed PSA. Hence, further work should focus on improving the exploration ability of PSA so that a better performance can be achieved. For example, a move towards a randomly selected agent by some agents may be incorporated to encourage exploration. In addition, a mechanism that allows the algorithm to self-adapt the number of agents as well as the iteration number is desirable. Finally, the work also points to the potential benefits in developing smart decision support systems that can enhance authorities' ability to take the most optimum strategic decisions when addressing complex problems such as vaccine distribution to combat a pandemic.

Author Contributions: Conceptualization, N.A.A.A. and K.A.A.; Formal analysis, K.A.A.; Funding acquisition, N.A.A.A.; Investigation, N.A.A.A. and K.A.A.; Methodology, N.A.A.A. and K.A.A.; Writing–original draft, N.A.A.A.; Writing—review & editing, K.A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Ministry of Higher Education, Malaysia under the Fundamental Research Grant Scheme, FRGS/1/2019/ICT02/MMU/02/15.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Holland, J.H. Genetic Algorithms. *Sci. Am.* **1992**, *267*, 66–73. Available online: http://www.jstor.org/stable/24939139 (accessed on 15 March 2022). [CrossRef]
2. Dorigo, M.; Birattari, M.; St, T. Ant Colony Optimization. *IEEE Comput. Intell. Mag.* **2006**, *1*, 28–39. [CrossRef]
3. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Houston, TX, USA, 12 June 1997; Volume 4, pp. 1942–1948. [CrossRef]
4. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
5. Rashedi, E.; Nezamabadi-pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
6. Yang, X.S. A New Metaheuristic Bat-Inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Gonzalez, J.R., Ed.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74.
7. Karaboga, D. *An Idea Based on Honey bee Swarm for Numerical Optimisation*; Technical Report; Computer Engineering Department, Erciyes University: Kayseri, Turkey, 2005.
8. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
9. Yang, X.S.; Deb, S.; Fong, S. Metaheuristic Algorithms: Optimal Balance of Intensification and Diversification. *Appl. Math. Inf. Sci.* **2014**, *8*, 977–983. [CrossRef]
10. Olorunda, O.; Engelbrecht, A.P. Measuring exploration/exploitation in particle swarms using swarm diversity. In Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–6 June 2008; pp. 1128–1134. [CrossRef]
11. Eberhart, R.; Shi, Y. Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. In Proceedings of the 2000 Congress on Evolutionary Computation, La Jolla, CA, USA, 16–19 July 2000; pp. 84–88.
12. Bansal, J.C.; Singh, P.K.; Saraswat, M.; Verma, A.; Jadon, S.S.; Abraham, A. Inertia weight strategies in particle swarm optimization. In Proceedings of the 2011 Third World Congress on Nature and Biologically Inspired Computing, Salamanca, Spain, 19–21 October 2011; pp. 633–640. [CrossRef]
13. Elkhateeb, N.A.; Badr, R.I. Employing Artificial Bee Colony with dynamic inertia weight for optimal tuning of PID controller. In Proceedings of the 2013 5th International Conference on Modelling, Identification and Control (ICMIC), Cairo, Egypt, 31 August–2 September 2013; pp. 42–46.

14. Yang, X.S. Review of Metaheuristics and Generalized Evolutionary Walk Algorithm. *arXiv* **2011**, arXiv:1105.3668. Available online: http://arxiv.org/abs/1105.3668 (accessed on 6 November 2014).
15. Sarangi, A.; Sarangi, S.K.; Panigrahi, S.P. An approach to identification of unknown IIR systems using crossover cat swarm optimization. *Perspect. Sci.* **2016**, *8*, 301–303. [CrossRef]
16. Binkley, K.J.; Hagiwara, M. Balancing Exploitation and Exploration in Particle Swarm Optimization: Velocity-based Reinitialization. *Trans. Jpn. Soc. Artif. Intell.* **2008**, *23*, 27–35. [CrossRef]
17. Jancauskas, V. Empirical Study of Particle Swarm Optimization Mutation Operators. *Balt. J. Mod. Comput.* **2014**, *2*, 199–214.
18. Li, C.; Yang, S.; Korejo, I. An Adaptive Mutation Operator for Particle Swarm Optimization. In Proceedings of the 2008 UK Workshop on Computational Intelligence, Leicester, UK, 10–12 September 2008; pp. 165–170.
19. Abualigah, L.; Diabat, A. *Advances in Sine Cosine Algorithm: A Comprehensive Survey*; Springer: Berlin/Heidelberg, Germany, 2021; Volume 54.
20. Gabis, A.B.; Meraihi, Y.; Mirjalili, S.; Ramdane-Cherif, A. *A Comprehensive Survey of Sine Cosine Algorithm: Variants and Applications*; Springer: Berlin/Heidelberg, Germany, 2021; Volume 54.
21. Askari, Q.; Younas, I.; Saeed, M. Critical evaluation of sine cosine algorithm and a few recommendations. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, Cancún, Mexico, 8–12 July 2020; pp. 319–320. [CrossRef]
22. Aziz, N.H.A.; Ibrahim, Z.; Aziz, N.A.A.; Mohamad, M.S.; Watada, J. Single-solution Simulated Kalman Filter algorithm for global optimisation problems. *Sādhanā* **2018**, *43*, 103. [CrossRef]
23. Rahman, T.A.B.; Ibrahim, Z.; Aziz, N.A.A.; Zhao, S.; Aziz, N.H.A. Single-Agent Finite Impulse Response Optimizer for Numerical Optimization Problems. *IEEE Access* **2018**, *6*, 9358–9374. [CrossRef]
24. Mongelli, M.; Battista, N.A. A swing of beauty: Pendulums, fluids, forces, and computers. *Fluids* **2020**, *5*, 48. [CrossRef]
25. Liang, J.J.; Qu, B.Y.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical report; Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2013.
26. Alcalá-Fdez, J.; Sánchez, L.; García, S.; del Jesus, M.J.; Ventura, S.; Garrell, J.M.; Otero, J.; Romero, C.; Bacardit, J.; Rivas, V.M.; et al. KEEL: A software tool to assess evolutionary algorithms for data mining problems. *Soft Comput.* **2009**, *13*, 307–318. [CrossRef]
27. Triguero, I.; González, S.; Moyano, J.M.; García, S.; Alcalá-Fdez, J.; Luengo, J.; Fernández, A.; del Jesús, M.J.; Sánchez, L.; Herrera, F. KEEL 3.0: An Open Source Software for Multi-Stage Analysis in Data Mining. *Int. J. Comput. Intell. Syst.* **2017**, *10*, 1238. [CrossRef]
28. Alcalá-Fdez, J.; Fernández, A.; Luengo, J.; Derrac, J. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *J. Mult.-Valued Log. Soft Comput.* **2011**, *17*, 255–287.
29. Ulmer, J.B.; Valley, U.; Rappuoli, R. Vaccine manufacturing: Challenges and solutions. *Nat. Biotechnol.* **2006**, *24*, 1377–1383. [CrossRef] [PubMed]
30. Hu, X. Optimizing Vaccine Distribution for Different Age Groups of Population Using DE Algorithm. In Proceedings of the 2013 Ninth International Conference on Computational Intelligence and Security, Emeishan, China, 14–15 December 2013; pp. 21–25. [CrossRef]
31. Liu, J.; Xia, S. Toward effective vaccine deployment: A systematic study. *J. Med. Syst.* **2011**, *35*, 1153–1164. [CrossRef] [PubMed]
32. Mossong, J.; Hens, N.; Jit, M.; Beutels, P.; Auranen, K.; Mikolajczyk, R.; Massari, M.; Salmaso, S.; Tomba, G.S.; Wallinga, J. Social contacts and mixing patterns relevant to the spread of infectious diseases. *PLoS Med.* **2008**, *5*, 381–391. [CrossRef] [PubMed]