mathematics

Article

Design of Aquila Optimization Heuristic for Identification of Control Autoregressive Systems

Khizer Mehmood [1], Naveed Ishtiaq Chaudhary [2,*], Zeshan Aslam Khan [1], Muhammad Asif Zahoor Raja [2], Khalid Mehmood Cheema [3] and Ahmad H. Milyani [4]

[1] Department of Electrical and Computer Engineering, International Islamic University, Islamabad 44000, Pakistan; khizer.mehmood@iiu.edu.pk (K.M.); zeeshan.aslam@iiu.edu.pk (Z.A.K.)
[2] Future Technology Research Center, National Yunlin University of Science and Technology, 123 University Road, Section 3, Douliou, Yunlin 64002, Taiwan; rajamaz@yuntech.edu.tw
[3] Department of Electronic Engineering, Fatima Jinnah Women University, Rawalpindi 46000, Pakistan; khalid.mehmood@fjwu.edu.pk
[4] Department of Electrical and Computer Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia; ahmilyani@kau.edu.sa
* Correspondence: chaudni@yuntech.edu.tw

Abstract: Swarm intelligence-based metaheuristic algorithms have attracted the attention of the research community and have been exploited for effectively solving different optimization problems of engineering, science, and technology. This paper considers the parameter estimation of the control autoregressive (CAR) model by applying a novel swarm intelligence-based optimization algorithm called the Aquila optimizer (AO). The parameter tuning of AO is performed statistically on different generations and population sizes. The performance of the AO is investigated statistically in various noise levels for the parameters with the best tuning. The robustness and reliability of the AO are carefully examined under various scenarios for CAR identification. The experimental results indicate that the AO is accurate, convergent, and robust for parameter estimation of CAR systems. The comparison of the AO heuristics with recent state of the art counterparts through nonparametric statistical tests established the efficacy of the proposed scheme for CAR estimation.

Keywords: swarm intelligence; parameter estimation; controlled autoregressive; aquila optimizer

MSC: 90C31; 93-10

check for updates

Citation: Mehmood, K.; Chaudhary, N.I.; Khan, Z.A.; Raja, M.A.Z.; Cheema, K.M.; Milyani, A.H. Design of Aquila Optimization Heuristic for Identification of Control Autoregressive Systems. *Mathematics* **2022**, *10*, 1749. https://doi.org/10.3390/math10101749

Academic Editor: Petr Stodola

Received: 17 April 2022
Accepted: 18 May 2022
Published: 20 May 2022

1. Introduction

1.1. Literature Review

In recent years, system identification has gained significant attention in various areas such as signal processing, parameter estimation, multiple-input multiple-output systems, etc. [1–4]. Parameter estimation refers to the determination of the best fitness values for each parameter using local or global optimization techniques. Parameter estimation consists of three steps: First, construct a mathematical model for a given system such that it replicates exact behaviour under the same conditions. Second, define a fitness function for a given set of parameters using various approximations such as least square, weighted least square, and generalized least square. Third, select an optimization technique for finding the best fitness values through iteration [5].

The research community has shown great interest in parameter estimation of control autoregressive (CAR) systems because of their importance and significance in effectively modelling a variety of engineering problems including power system optimization [6] electricity load prediction [7] battery charge estimation [8] forecasting groundwater flooding [9] and CO_2 emission forecasting [10]. Various methods for parameter estimation of control autoregressive (CAR) models are proposed in the literature. Ding et al. [11] decompose a

CAR model into two subsystems and derive a two-stage multi-innovation gradient-based iterative algorithm for parameter estimation. Raja et al. [12] use genetic algorithms (GA) for parameter estimation of a nonlinear Hammerstein controlled autoregressive system by minimizing the error function between true and estimated parameters. Mehmood et al. [13] explore the strengths of evolutionary and swarm intelligence for parameter estimation of a controlled autoregressive moving average model by minimizing mean absolute error and other measures for parameter estimation. Tariq et al. [14] apply differential evolution-based algorithms for parameter estimation of Hammerstein systems minimizing the actual and predicted responses of cost function output errors.

Metaheuristic methods have made significant progress in solving optimization problems [15–17]. These optimization methods can be classified into four categories. Category one includes evolutionary algorithms, which involve natural biological behaviour such as mutation and crossover operation. Numerous algorithms are proposed in this category, such as genetic algorithms, GAs [18] differential evolution, DE [19] fuzzy evolution [20] maximum likelihood adaptive differential evolution [21] and tree growth algorithms [22]. Category two includes human-based algorithms, which are inspired from human behaviour such as collective decision optimization [23] imperialist competitive algorithms [24] and teaching-learning-based optimization [25]. Category three includes physics-based methods which use physical laws for solution of optimization problems. A few of the methods in this area are gravitational search algorithms [26] thermal exchange optimization [27] and multi-verse optimizers [28]. The final category includes swarm intelligence and animal-inspired methods for optimization solution. Significant progress has made in this domain as well, and various methods are proposed such as particle swarm optimization (PSO) [29] artificial bee colonies [30] lion optimization algorithms [31] and whale optimization algorithms [32].

Among swarm intelligence techniques, the Aquila optimizer (AO) [33] has been recently proposed for global solutions and applied in solving various real-world optimization problems. Elaziz et al. [34] proposed an image classification hybrid framework for COVID-19 images by combining deep learning and AO for feature selection and dimensionality reduction for CT and X-ray images. Hussan et al. [35] applied AO in harmonic parameter estimation for an H-bridge inverter by minimizing error with real-time verification on digital signal processing launchpad. Khamees et al. [36] applied AO in Weibull distribution parameter estimation for low error and high correlation coefficients in a wind energy system.

1.2. Research Contribution

In the current study, the swarm intelligence of the Aquila optimizer, AO, is exploited for parameter estimation of control autoregressive (CAR) systems. The AO is evaluated in terms of robustness, correctness, and convergence for different noise levels in the CAR model. The noticeable contributions are as follows:

- The strength of a swarm intelligence-based Aquila optimizer (AO) heuristic is exploited for solving parameter estimation in a control autoregressive (CAR) model.
- The convergence, accuracy, and robustness analyses of the AO are conducted for different noise levels considered in the CAR model.
- Statistical analyses for parameter tuning of the AO as well as for reliability and stability assessment are conducted for different generations and population sizes.

1.3. Paper Organization

The rest of the paper is prepared as follows: the CAR system model is given in Section 2. The AO-based methodology is presented in Section 3. The performance analysis of the CAR model is provided in Section 4. The main conclusions and some future research directions are listed in Section 5.

2. Mathematical Model of CAR Systems

Consider the second-order CAR model presented in (1):

$$\mathrm{R(z)q(t) = S(z)\theta(t) + \omega(t)}, \tag{1}$$

where θ(t) is the input of the model, q(t) is the output of the model, and ω(t) is zero mean white noise. R(z) and S(z) are polynomials given in (2) and (3):

$$\mathrm{R(z) = 1 + r_1 z^{-1} + r_2 z^{-2} + \ldots + r_{n_r} z^{-n_r}}, \tag{2}$$

$$\mathrm{S(z) = s_1 z^{-1} + s_2 z^{-2} + \ldots + s_{n_s} z^{-n_s}}. \tag{3}$$

Assume that $\theta(t) = 0$, $q(t) = 0$, $\omega(t) = 0$ for $t < 0$, and n_r and n_s are known. The parameter vectors are given in (4) and (5):

$$\mathrm{r = [r_1, r_2, \ldots, r_{n_r}]^T \in R^{n_r}}, \tag{4}$$

$$\mathrm{s = [s_1, s_2, \ldots, s_{n_s}]^T \in R^{n_s}}. \tag{5}$$

The corresponding information vectors are given in (6) and (7):

$$\epsilon_r(t) = [-q(t-1), -q(t-2), \ldots, -q(t-n_r)]^T \in R^{n_r}, \tag{6}$$

$$\epsilon_s(t) = [\theta(t-1), \theta(t-2), \ldots, \theta(t-n_s)]^T \in R^{n_s}. \tag{7}$$

The model presented in (1) can be rewritten as given in (8) and simplified in (9).

$$q(t) = [1 - R(z)]q(t) + S(z)\,\theta(t) + \omega(t), \tag{8}$$

$$q(t) = \epsilon_r(t)r + \epsilon_s(t)s + \ \omega(t). \tag{9}$$

The overall information and parameter vectors of the CAR model are given as

$$\epsilon(t) = [\epsilon_r(t)\ \epsilon_s(t)] \tag{10}$$

$$\alpha = [r\ s]. \tag{11}$$

Then, the identification for CAR system becomes:

$$q(t) = \epsilon^T(t)\alpha + \omega(t). \tag{12}$$

3. Methodology

In this section, an AO-based methodology for parameter estimation of a CAR model is presented. The graphical abstract of the proposed methodology for CAR model is presented in Figure 1. It provides the overview of the proposed study, which includes the parameter estimation of a CAR model by applying a swarm intelligence-based Aquila optimizer. The optimum parameters are evaluated on the basis of the square of the difference between estimated and true values along with the number of generations as termination criteria.

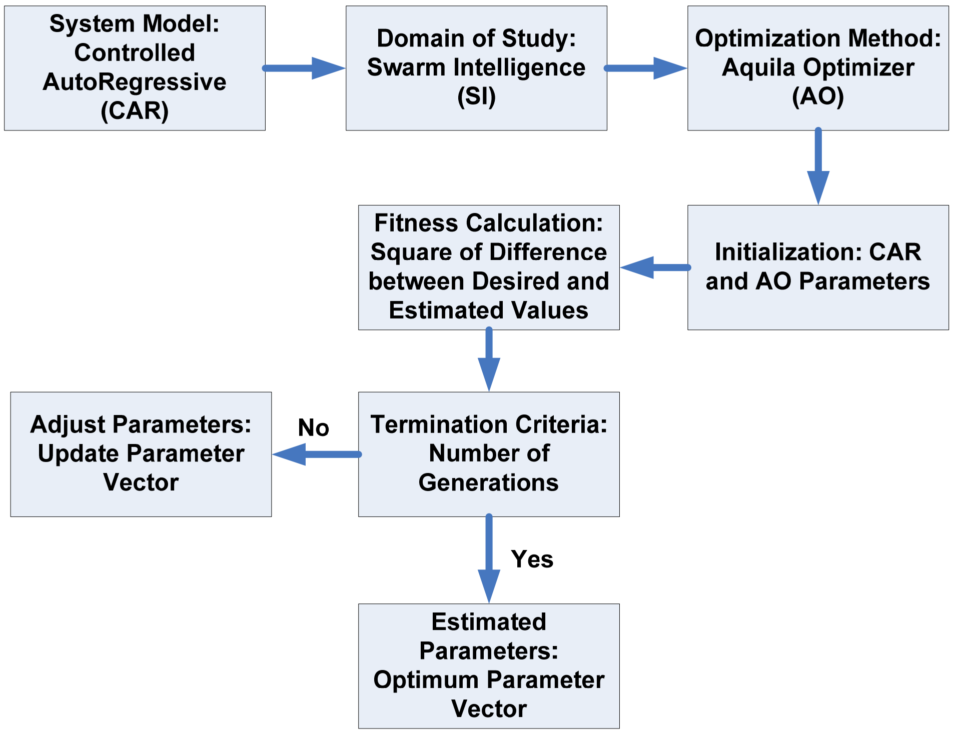

System Model: Controlled AutoRegressive (CAR)
Domain of Study: Swarm Intelligence (SI)
Optimization Method: Aquila Optimizer (AO)
Initialization: CAR and AO Parameters
Fitness Calculation: Square of Difference between Desired and Estimated Values
Termination Criteria: Number of Generations
No
Adjust Parameters: Update Parameter Vector
Yes
Estimated Parameters: Optimum Parameter Vector

Figure 1. Graphical abstract of the proposed study.

3.1. Aquila Optimization (AO) Method

The AO is a swarm intelligence-based method for finding solutions to optimum global problems [33]. It is applied in various domains such as internet of things (IoT) [37], power electronics [35], image processing [38], oil production forecasting [39], Francis turbines [40], hybrid solid oxide fuel cell (SOFC) [41], wind energy [42], and population forecasting [43]. AO is a population-based optimization method inspired by Aquila's prey-hunting ability. It uses four hunting techniques and has the ability to switch between these techniques. The mathematical model, pseudocode, and flowchart are presented below.

3.1.1. Population Initialization

AO starts with the initialization of the population for candidate solutions (W) as given in (13):

$$W = \begin{bmatrix} w_{1,1} & \cdots & w_{1,D} \\ \vdots & \ddots & \vdots \\ w_{N_p,1} & \cdots & w_{N_p,D} \end{bmatrix}. \tag{13}$$

The population is randomly generated using (14):

$$W_{k,l} = \text{rand}(UB_l - LB_l) + LB_l,\ k = 1,2\ldots N_p, l = 1,2\ldots D \tag{14}$$

where N_p is the total population size, and D is the number of decision variables.

3.1.2. The Mathematical Model

The mathematical formulation is divided into four steps which are presented below.

Expanded Exploration (W_1)

In the first method, W_1, the Aquila explores the prey area by involving a high soar and a vertical stoop. It is presented in (15):

$$W_1(t+1) = W_{best}(t) \times \left(1 - \frac{t}{T}\right) + (W_M(t) - W_{best}(t) * \text{rand}) \tag{15}$$

where $W_1(t+1)$ is the next-iteration solution for W_1, $W_{best}(t)$ is the best solution, $\left(1-\frac{t}{T}\right)$ is used to control the search space, and $W_M(t)$ is the mean of the current solution, which is calculated using (16):

$$W_M(t) = \frac{1}{N_p}\sum_{k=1}^{N_p} W_k(t), \forall\, l = 1, 2 \ldots D \tag{16}$$

where N_p is the total population size, and D is the number of decision variables.

Narrowed Exploration (W_2)

In the second method (W_2), upon finding the prey area, the Aquila circles around the target and uses a method called contour fight with a short glide attack. In W_2, AO narrowly explores the area for preparation of the attack on the target, which is calculated using (17):

$$W_2(t+1) = W_{best}(t) \times LEF(DI) + W_R(t) - (y - w) * rand \tag{17}$$

where $W_2(t+1)$ is the next-iteration solution for W_2, DI is the space dimension, $W_R(t)$ is the random solution from [1 N_p], and LEF(DI) is the distribution function calculated in (18):

$$LEF(DI) = d \times \frac{e \times \sigma}{|f|^{\frac{1}{\delta}}} \tag{18}$$

where d is constant equals 0.01, e and f are random numbers between 0–1. σ is calculated using (19):

$$\sigma = \left(\frac{\Gamma'(1+\delta) \times \sin\left(\frac{\pi\delta}{2}\right)}{\Gamma'\left(\frac{1+\delta}{2}\right) \times \delta \times 2^{\left(\frac{\delta-1}{2}\right)}} \right) \tag{19}$$

where δ is fixed at 1.5. y and w from (17) are calculated as follows.

$$y = g \times \cos(\theta), \tag{20}$$

$$w = g \times \sin(\theta) \tag{21}$$

where g is between 1–20.

$$g = g_1 + V \times DI_1, \tag{22}$$

$$\theta = -\varepsilon \times DI_1 + \theta_1, \tag{23}$$

$$\theta_1 = \frac{3\pi}{2} \tag{24}$$

where V is fixed to 0.00565 and ε is fixed to 0.005.

Expanded Exploitation (W_3)

In the third method (W_3), AO exploits the search space by descending vertically to discover prey reaction and to land and attack. It is given in (25):

$$W_3(t+1) = (W_{best}(t) - W_M(t)) \times \beta - rand + ((UB - LB) \times rand + LB) \times \mu \tag{25}$$

where $W_3(t+1)$ is the next-iteration solution for W_3, β and μ are exploitation adjustment factors, $W_{best}(t)$ is the best solution, UB and LB are problem-dependent parameters, and $W_M(t)$ is the mean of the current solution, which is calculated using (16).

Narrowed Exploitation (W_4)

In the fourth method (W_4), AO uses the method of walking and grabbing by getting closer to prey and attacking based on stochastic movement as presented in (26):

$$W_4(t+1) = QYF \times W_{best}(t) - (O_1 \times W(t) \times rand) - O_2 \times LEF(DI) + rand \times O_1 \quad (26)$$

where $W_4(t+1)$ is the next-iteration solution for W_4 and QYF is the quality factor, calculated using (27):

$$QYF = t^{\frac{2rand-1}{(1-T)^2}} \quad (27)$$

O_1 and O_2 indicate the variations of motion, which are calculated using (28) and (29):

$$O_1 = 2 \times rand - 1, \quad (28)$$

$$O_2 = 2 \times \left(1 - \frac{t}{T}\right) \quad (29)$$

The flowchart for AO is shown in Figure 2.

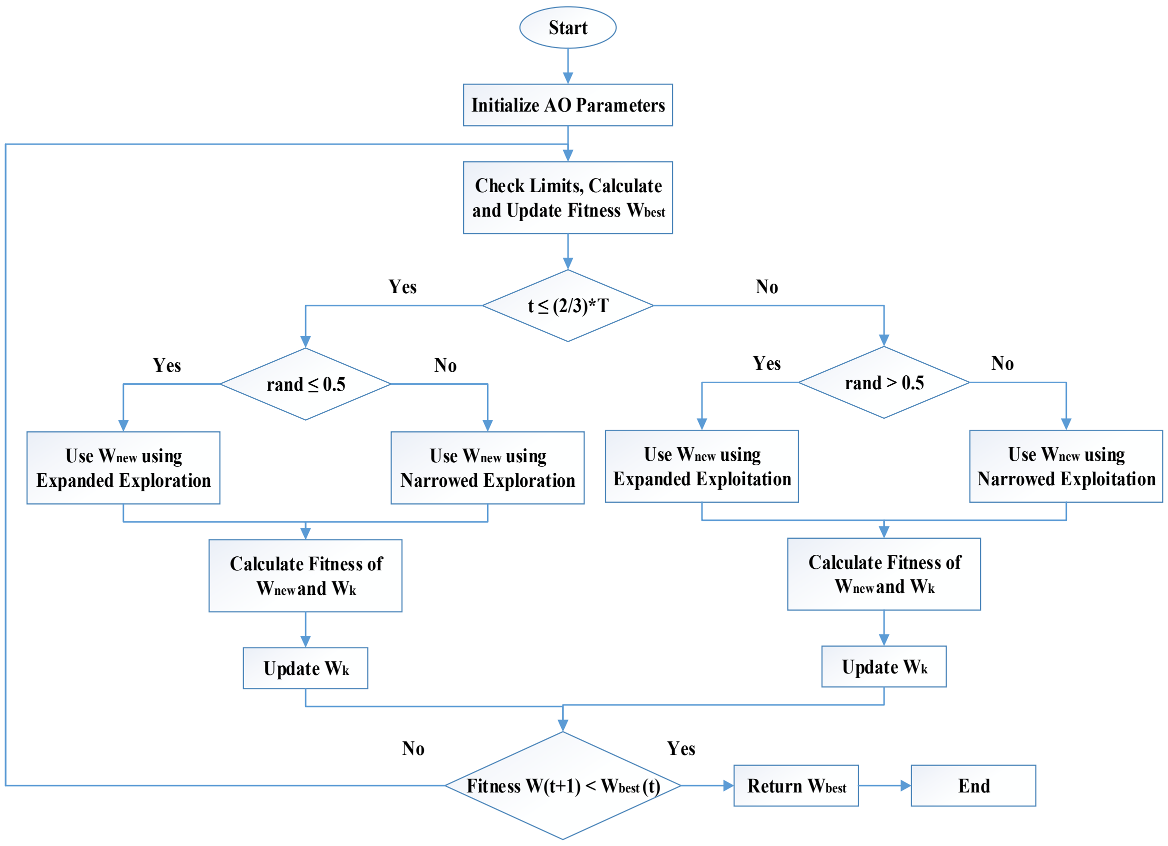

Start
Initialize AO Parameters
Check Limits, Calculate and Update Fitness Wbest
t ≤ (2/3)*T
Yes
No
rand ≤ 0.5
Yes
No
rand > 0.5
Yes
No
Use Wnew using Expanded Exploration
Use Wnew using Narrowed Exploration
Use Wnew using Expanded Exploitation
Use Wnew using Narrowed Exploitation
Calculate Fitness of Wnew and Wk
Calculate Fitness of Wnew and Wk
Update Wk
Update Wk
Fitness W(t+1) < Wbest (t)
No
Yes
Return Wbest
End

Figure 2. AO Flowchart.

The pseudocode of the MLADE is presented in Algorithm 1.

Algorithm 1: Pseudo-code of AO

```
Initialization:
Initialize the population W and parameters of AO such as $\sigma$, $\beta$, etc.
WHILE do
Calculate fitness values
Determine the best obtained solution $W_{best}$.
for $k = 1 : N_p$
Update mean value $W_M(t)$.
Update $w, y, O_1, O_2$ and $LEF(DI)$.
if $t \le \left(\frac{2}{3}\right) * T$
if rand $\le 0.5$
Expanded Exploration ($W_1$)
Update solution using (15).
If (Fitness $W_1(t+1)$ < Fitness $W(t)$)
                                        $W(t) = W_1(t+1)$
If (Fitness $W_1(t+1)$ < Fitness $W_{best}(t)$)
                                        $W_{best}(t) = W_1(t+1)$
end
end
else
Narrowed Exploration ($W_2$)
Update solution using (17).
If (Fitness $W_2(t+1)$ < Fitness $W(t)$)
                                        $W(t) = W_2(t+1)$
If (Fitness $W_2(t+1)$ < Fitness $W_{best}(t)$)
                                        $W_{best}(t) = W_2(t+1)$
end
end
end
else if rand $\le 0.5$
Expanded Exploitation ($W_3$)
Update solution using (25).
If (Fitness $W_3(t+1)$ < Fitness $W(t)$)
                                        $W(t) = W_3(t+1)$
If (Fitness $W_3(t+1)$ < Fitness $W_{best}(t)$)
$W_{best}(t) = W_3(t+1)$
end
end
else
Narrowed Exploitation ($W_4$)
Update solution using (26).
If (Fitness $W_4(t+1)$ < Fitness $W(t)$)
                                        $W(t) = W_4(t+1)$
If (Fitness $W_4(t+1)$ < Fitness $W_{best}(t)$)
                                        $W_{best}(t) = W_4(t+1)$
end
end
end
end
end
end
return $W_{best}$
```

4. Performance Analysis

In this section, the performance analysis of AO for the CAR model is presented. The identification of the CAR model is conducted on various noise levels, generations, and population sizes. The algorithm is weighed in terms of accuracy, which is measured by the fitness function presented in (30):

$$\text{Fitness Function} = \text{mean}\left(\mathbf{z} - \hat{\mathbf{z}}\right)^2 \tag{30}$$

where $\hat{z}$ is the estimated response through the proposed evolutionary algorithms and z is the desired response. For the simulation study, we considered the second-order CAR model from [6] as presented in (31) and (32):

$$R(z) = 1 + 1.35z^{-1} + 0.75z^{-2}, \tag{31}$$

$$S(z) = 1.68z^{-1} + 2.32z^{-2}. \tag{32}$$

4.1. Parameter Tuning of AO

Learning optimal weights plays a significant role in boosting the performance of the AO method. Hence, the aim is to use the best values for the exploitation adjustment factors (β and μ) for learning the optimum weights W_3 using the update rule given in (25). The best values for both parameters (β and μ) are achieved through hyper-parameter tuning. Using grid search, various combinations of both β and μ are split into different cases like case 1 to case 9, and the chosen values of β and μ are presented in Table 1. Hyper-parameter tuning is performed for different generations and population sizes in a noise-free environment (zero noise). Each case is executed for three generations, i.e., 1000, 1500, and 2000, and populations, i.e., 30, 40, and 50, whereas the simulations for a combination of one generation and one population are performed for 15 runs to accomplish the average fit, best fit, worst fit, and standard deviation.

Table 1. Parameter-tuning cases for AO analysis.

Case No.	β Value	μ Value
1	0.9	0.9
2	0.9	0.5
3	0.9	0.1
4	0.5	0.9
5	0.5	0.5
6	0.5	0.1
7	0.1	0.9
8	0.1	0.5
9	0.1	0.1

Different scenarios (cases) reflecting the tuning of β and μ for the optimal weight update mechanism and the average fit, best fit, worst fit, and standard deviation with different generations and populations are computed and represented in Tables 2–10. Varying β and μ causes the fit to vary with regard to a change in a generation or a population size, and the tables show the fitness trends. It is observed from the results given in Tables 2–10 that the optimal fit for different generations and populations is achieved with case 9, i.e., $\beta = 0.1$ and $\mu = 0.1$.

Apart from the fitness results presented in Tables 2–10, the mean fit values achieved with nine (β and μ) variations, three generations, and three populations are presented in Table 11. It is observed from the mean fit values in Table 11 that AO obtained the minimum mean fit for case 9, i.e., 440×10^{-6}. Therefore, for optimal AO performance, the remaining simulation results are presented with the best hyper-parameter values, i.e., $\beta = 0.1$ and $\mu = 0.1$.

Table 2. AO parameter analysis for case 1.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	1.2×10^{-3}	1.2×10^{-4}	3.3×10^{-3}	8.7×10^{-4}
	40	1.3×10^{-3}	1.8×10^{-4}	4.0×10^{-3}	1.0×10^{-3}
	50	6.9×10^{-4}	1.4×10^{-4}	2.4×10^{-3}	6.1×10^{-4}
1500	30	1.5×10^{-3}	4.1×10^{-4}	4.1×10^{-3}	1.2×10^{-3}
	40	8.8×10^{-4}	6.2×10^{-5}	2.2×10^{-3}	6.7×10^{-4}
	50	7.0×10^{-4}	9.0×10^{-5}	1.6×10^{-3}	4.7×10^{-4}
2000	30	6.4×10^{-4}	1.5×10^{-4}	2.0×10^{-3}	4.9×10^{-4}
	40	7.7×10^{-4}	1.7×10^{-4}	2.5×10^{-3}	6.4×10^{-4}
	50	4.5×10^{-4}	1.2×10^{-4}	1.1×10^{-3}	3.1×10^{-4}

Table 3. AO parameter analysis for case 2.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	4.4×10^{-3}	3.9×10^{-4}	12×10^{-3}	3.5×10^{-3}
	40	2.6×10^{-3}	4.3×10^{-5}	16×10^{-3}	3.9×10^{-3}
	50	2.8×10^{-3}	8.9×10^{-5}	11×10^{-3}	2.9×10^{-3}
1500	30	2.9×10^{-3}	1.3×10^{-4}	9.0×10^{-3}	3.0×10^{-3}
	40	1.1×10^{-3}	1.9×10^{-5}	2.7×10^{-3}	8.3×10^{-4}
	50	1.1×10^{-3}	1.2×10^{-4}	3.1×10^{-3}	7.9×10^{-4}
2000	30	1.0×10^{-3}	1.9×10^{-4}	2.8×10^{-3}	7.4×10^{-4}
	40	1.1×10^{-3}	3.3×10^{-4}	3.1×10^{-3}	7.3×10^{-4}
	50	1.6×10^{-3}	2.2×10^{-4}	6.5×10^{-3}	1.7×10^{-3}

Table 4. AO parameter analysis for case 3.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	1.9×10^{-3}	1.4×10^{-4}	5.3×10^{-3}	1.5×10^{-3}
	40	1.4×10^{-3}	1.5×10^{-4}	3.9×10^{-3}	1.2×10^{-3}
	50	6.9×10^{-4}	4.1×10^{-5}	1.8×10^{-3}	4.9×10^{-4}
1500	30	1.0×10^{-3}	1.0×10^{-4}	5.2×10^{-3}	1.3×10^{-3}
	40	8.5×10^{-4}	1.7×10^{-4}	2.5×10^{-3}	6.7×10^{-4}
	50	6.4×10^{-4}	1.1×10^{-4}	1.8×10^{-3}	5.2×10^{-4}
2000	30	8.5×10^{-4}	1.2×10^{-4}	2.2×10^{-3}	5.6×10^{-4}
	40	5.2×10^{-4}	3.0×10^{-5}	1.0×10^{-3}	3.3×10^{-4}
	50	5.6×10^{-4}	8.2×10^{-5}	2.4×10^{-3}	5.7×10^{-4}

Table 5. AO parameter analysis for case 4.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	4.4×10^{-3}	3.9×10^{-4}	12×10^{-3}	3.5×10^{-3}
	40	2.6×10^{-3}	4.3×10^{-4}	16×10^{-3}	3.9×10^{-3}
	50	2.8×10^{-3}	8.9×10^{-5}	11.5×10^{-3}	2.9×10^{-3}
1500	30	2.9×10^{-3}	1.3×10^{-4}	9.0×10^{-3}	3.0×10^{-3}
	40	1.1×10^{-3}	1.9×10^{-5}	2.7×10^{-3}	8.3×10^{-4}
	50	1.1×10^{-3}	1.3×10^{-4}	3.1×10^{-3}	7.9×10^{-4}
2000	30	1.0×10^{-3}	1.9×10^{-4}	2.8×10^{-3}	7.4×10^{-4}
	40	1.1×10^{-4}	3.3×10^{-4}	3.1×10^{-3}	7.3×10^{-4}
	50	1.6×10^{-3}	2.2×10^{-4}	6.5×10^{-3}	1.7×10^{-3}

Table 6. AO parameter analysis for case 5.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	1.1×10^{-3}	1.7×10^{-4}	3.4×10^{-3}	8.9×10^{-4}
	40	5.4×10^{-4}	7.2×10^{-5}	1.4×10^{-3}	3.6×10^{-4}
	50	5.7×10^{-4}	2.3×10^{-5}	1.2×10^{-3}	2.9×10^{-4}
1500	30	6.8×10^{-4}	6.7×10^{-5}	1.7×10^{-3}	5.5×10^{-4}
	40	2.2×10^{-4}	7.4×10^{-6}	6.6×10^{-4}	2.1×10^{-4}
	50	3.7×10^{-4}	1.3×10^{-4}	1.2×10^{-3}	3.3×10^{-4}
2000	30	3.2×10^{-4}	3.9×10^{-5}	9.6×10^{-4}	3.1×10^{-4}
	40	2.6×10^{-4}	3.1×10^{-5}	6.1×10^{-4}	1.8×10^{-4}
	50	2.1×10^{-4}	2.1×10^{-5}	5.6×10^{-4}	1.5×10^{-4}

Table 7. AO parameter analysis for case 6.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	1.2×10^{-3}	1.2×10^{-4}	3.3×10^{-3}	8.7×10^{-4}
	40	1.3×10^{-3}	1.8×10^{-4}	4.0×10^{-3}	1.0×10^{-3}
	50	6.9×10^{-4}	1.4×10^{-4}	2.4×10^{-3}	6.1×10^{-4}
1500	30	1.5×10^{-3}	4.1×10^{-4}	4.1×10^{-3}	1.2×10^{-3}
	40	8.8×10^{-4}	6.2×10^{-5}	2.2×10^{-3}	6.7×10^{-4}
	50	7.0×10^{-4}	9.0×10^{-5}	1.6×10^{-3}	4.7×10^{-4}
2000	30	6.4×10^{-4}	1.5×10^{-4}	2.0×10^{-3}	4.9×10^{-4}
	40	7.7×10^{-4}	1.7×10^{-4}	2.5×10^{-3}	6.4×10^{-4}
	50	4.5×10^{-4}	1.2×10^{-4}	1.1×10^{-3}	3.1×10^{-4}

Table 8. AO parameter analysis for case 7.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	1.9×10^{-3}	1.7×10^{-4}	5.3×10^{-3}	1.5×10^{-3}
	40	1.4×10^{-3}	7.2×10^{-5}	3.9×10^{-3}	1.2×10^{-3}
	50	6.9×10^{-4}	2.3×10^{-5}	1.8×10^{-3}	4.9×10^{-4}
1500	30	1.0×10^{-3}	6.7×10^{-5}	5.2×10^{-3}	1.3×10^{-3}
	40	8.5×10^{-4}	7.4×10^{-6}	2.5×10^{-3}	6.7×10^{-4}
	50	6.4×10^{-4}	1.3×10^{-4}	1.8×10^{-3}	5.2×10^{-4}
2000	30	8.5×10^{-4}	3.9×10^{-5}	2.2×10^{-3}	5.6×10^{-4}
	40	5.2×10^{-4}	3.1×10^{-5}	1.0×10^{-3}	3.3×10^{-4}
	50	5.6×10^{-4}	2.1×10^{-5}	2.4×10^{-3}	5.7×10^{-4}

Table 9. AO parameter analysis for case 8.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	7.3×10^{-4}	2.6×10^{-5}	3.0×10^{-3}	9.0×10^{-4}
	40	9.1×10^{-4}	2.4×10^{-4}	2.5×10^{-3}	7.4×10^{-4}
	50	6.9×10^{-4}	4.3×10^{-5}	1.7×10^{-3}	5.3×10^{-4}
1500	30	5.5×10^{-4}	3.0×10^{-5}	1.8×10^{-3}	4.7×10^{-4}
	40	5.3×10^{-4}	1.0×10^{-4}	1.1×10^{-3}	2.5×10^{-4}
	50	4.8×10^{-4}	7.8×10^{-5}	1.6×10^{-3}	4.6×10^{-4}
2000	30	4.2×10^{-4}	8.9×10^{-5}	1.5×10^{-3}	3.5×10^{-4}
	40	5.1×10^{-4}	6.8×10^{-5}	2.3×10^{-3}	5.5×10^{-4}
	50	3.7×10^{-4}	4.8×10^{-5}	8.6×10^{-4}	2.1×10^{-4}

Table 10. AO parameter analysis for case 9.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	6.9×10^{-4}	1.1×10^{-4}	2.6×10^{-3}	6.2×10^{-4}
	40	4.7×10^{-4}	4.3×10^{-5}	1.5×10^{-3}	3.8×10^{-4}
	50	5.2×10^{-4}	3.5×10^{-5}	2.2×10^{-3}	5.7×10^{-4}
1500	30	6.7×10^{-4}	1.4×10^{-4}	2.3×10^{-3}	5.7×10^{-4}
	40	4.6×10^{-4}	2.8×10^{-5}	1.4×10^{-3}	4.3×10^{-4}
	50	3.0×10^{-4}	7.7×10^{-5}	7.1×10^{-3}	2.1×10^{-4}
2000	30	3.3×10^{-4}	3.6×10^{-5}	9.6×10^{-4}	2.3×10^{-4}
	40	1.9×10^{-4}	3.9×10^{-5}	5.0×10^{-4}	1.4×10^{-4}
	50	2.8×10^{-4}	4.5×10^{-5}	1.1×10^{-3}	2.5×10^{-4}

Table 11. AO parameter tuning mean value analysis.

Cases	Mean Fitness Value
Case 1	908×10^{-6}
Case 2	2.06×10^{-3}
Case 3	938×10^{-6}
Case 4	2.06×10^{-3}
Case 5	479×10^{-6}
Case 6	908×10^{-6}
Case 7	938×10^{-6}
Case 8	581×10^{-6}
Case 9	440×10^{-6}

The fitness plots for case 9 with zero noise for three generations and populations are shown in Figures 3 and 4. The fitness curves with fixed generation size and varying population size are given in Figure 3a–c, and Figure 4a–c include fitness curves for the fixed population size and changing generation size. Figures 3 and 4 show that with the best parameter setting of $\beta = 0.1$ and $\mu = 0.1$, the AO strategy achieves minimum fit for a large number of generations and populations.

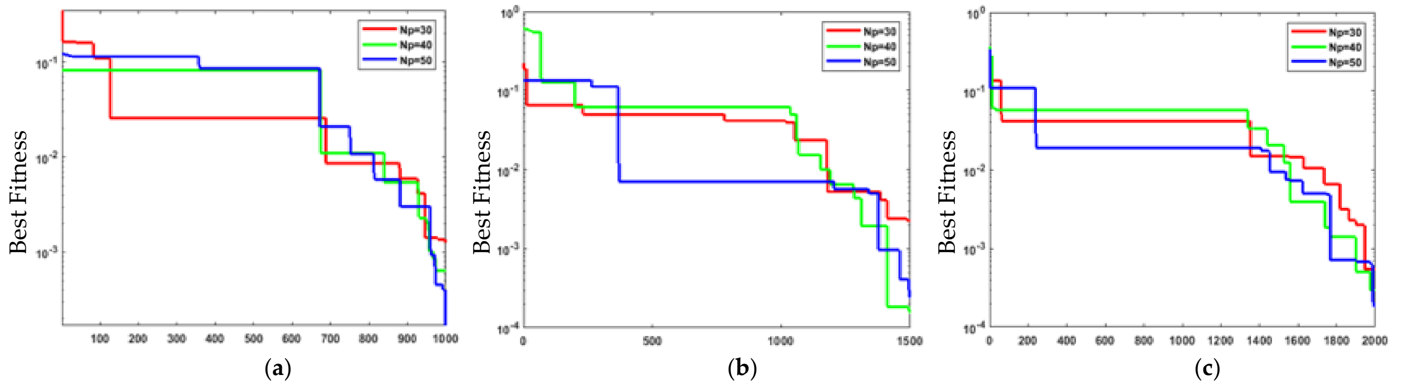
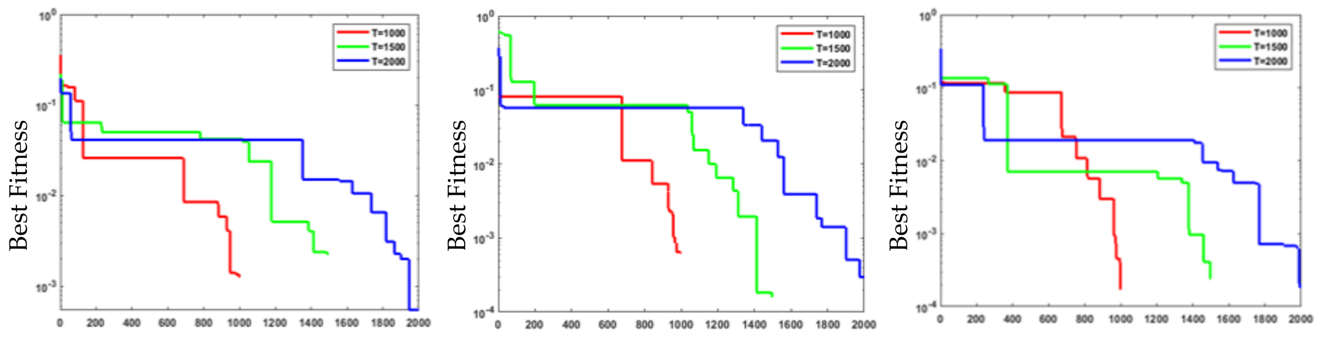


Figure 3. Fitness plots for AO with respect to population sizes. (**a**) T = 1000. (**b**) T = 1500. (**c**) T = 2000.



(**a**) (**b**) (**c**)

Figure 4. Fitness plots for AO with respect to generations. (**a**) Np = 30. (**b**) Np = 40. (**c**) Np = 50.

4.2. Statistical Convergence Analysis

In this section, the performance of AO algorithm is assessed by introducing three noise levels to the CAR model. Moreover, the fit of AO is estimated through three variations of generation [1000, 1500, 2000] and population [30, 40, 50]. The evaluation metrics used to assess the performance of AO for CAR are average fit, best fit, worst fit, and standard deviation (STD).

The performance in terms of fit variations and standard deviations for the three noise levels, i.e., 0.04, 0.06, and 0.08, is demonstrated in Tables 12–14, respectively. It is witnessed from Tables 12–14 that the AO fit decreases by increasing population and generation size. It is noticed from Table 12 that the minimum average fit, best fit, and worst fit achieved for noise level = 0.04 are 1.7×10^{-3}, 1.0×10^{-3}, and 3.0×10^{-3}, respectively. Similarly, the three best fit values for noise levels 0.06 and 0.08, given in Tables 13 and 14, are 2.7×10^{-3}, 2.3×10^{-3}, and 3.2×10^{-3} and 4.5×10^{-3}, 4.1×10^{-3}, and 4.9×10^{-3}, respectively.

Table 12. AO analysis with respect to generation and population sizes at 0.04 noise variance.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	3.8×10^{-3}	1.0×10^{-3}	9.3×10^{-3}	2.7×10^{-3}
	40	2.0×10^{-3}	1.1×10^{-3}	3.7×10^{-3}	7.4×10^{-4}
	50	1.8×10^{-3}	1.1×10^{-3}	4.0×10^{-3}	8.9×10^{-4}
1500	30	2.0×10^{-3}	1.1×10^{-3}	5.3×10^{-3}	1.1×10^{-3}
	40	1.7×10^{-3}	1.1×10^{-3}	3.0×10^{-3}	6.0×10^{-4}
	50	1.7×10^{-3}	1.2×10^{-3}	2.4×10^{-3}	4.5×10^{-4}
2000	30	1.9×10^{-3}	1.1×10^{-3}	3.2×10^{-3}	4.8×10^{-4}
	40	1.7×10^{-3}	1.0×10^{-3}	3.7×10^{-3}	8.4×10^{-4}
	50	1.9×10^{-3}	1.1×10^{-3}	5.6×10^{-3}	1.3×10^{-3}

Table 13. AO analysis with respect to generation and population sizes at 0.06 noise variance.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	4.6×10^{-3}	2.3×10^{-3}	10×10^{-3}	2.4×10^{-3}
	40	3.3×10^{-3}	2.3×10^{-3}	4.4×10^{-3}	6.9×10^{-4}
	50	3.3×10^{-3}	2.3×10^{-3}	5.9×10^{-3}	1.0×10^{-3}
1500	30	3.6×10^{-3}	2.4×10^{-3}	6.5×10^{-3}	1.3×10^{-3}
	40	3.4×10^{-3}	2.5×10^{-3}	8.1×10^{-3}	1.4×10^{-3}
	50	2.9×10^{-3}	2.3×10^{-3}	4.1×10^{-3}	6.0×10^{-4}
2000	30	3.1×10^{-3}	2.3×10^{-3}	7.1×10^{-3}	1.2×10^{-3}
	40	2.8×10^{-3}	2.3×10^{-3}	3.8×10^{-3}	4.6×10^{-4}
	50	2.7×10^{-3}	2.3×10^{-3}	3.2×10^{-3}	2.7×10^{-4}

Table 14. AO analysis with respect to generation and population sizes at 0.08 noise variance.

Generations (T)	Population (Np)	Average Fitness	Best Fitness	Worst Fitness	STD
1000	30	6.4×10^{-3}	4.1×10^{-3}	13.1×10^{-3}	2.5×10^{-3}
	40	5.0×10^{-3}	4.1×10^{-3}	7.0×10^{-3}	7.8×10^{-4}
	50	5.0×10^{-3}	4.1×10^{-3}	7.1×10^{-3}	8.6×10^{-4}
1500	30	5.1×10^{-3}	4.2×10^{-3}	8.2×10^{-3}	1.0×10^{-3}
	40	4.7×10^{-3}	4.1×10^{-3}	6.3×10^{-3}	6.4×10^{-4}
	50	4.8×10^{-3}	4.0×10^{-3}	6.8×10^{-3}	7.1×10^{-4}
2000	30	4.6×10^{-3}	4.1×10^{-3}	4.9×10^{-3}	2.2×10^{-4}
	40	4.6×10^{-3}	4.1×10^{-3}	5.6×10^{-3}	3.9×10^{-4}
	50	4.5×10^{-3}	4.1×10^{-3}	5.0×10^{-3}	2.5×10^{-4}

The performance of the AO method in terms of best fit for three noise levels, i.e., 0.04, 0.06, and 0.08, is evaluated for the three variations in a generation, 1000, 1500, and 2000, and population size, 30, 40, and 50. Figure 5 shows the fitness plots. The fitness curves in Figure 5a–c represent the best fit of the AO algorithm for noise variance = 0.04. In contrast, Figure 5d–e show the best fit curves for noise variance = 0.06. Likewise, the best fit plots for noise variance = 0.08 are given in Figure 5g–i. Figure 5a–i shows that the AO fit for the three noise levels, i.e., 0.04, 0.06, and 0.08, decreases significantly with increasing generation and population size as well. However, better fit results are achieved for smaller values of noise with a greater number of generations and populations.

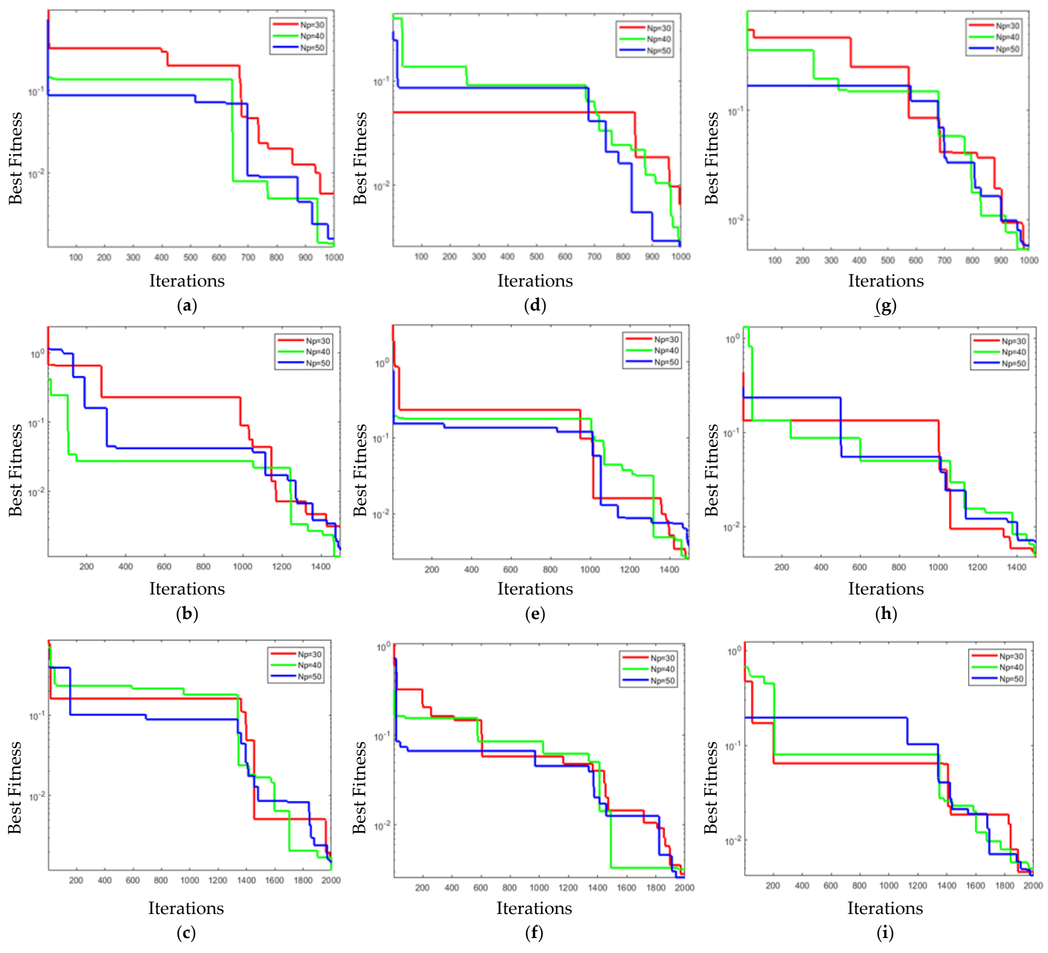


Figure 5. Fitness plots for AO w.r.t population sizes. (**a**) T = 1000. (**b**) T = 1500. (**c**) T = 2000. (**d**) T = 1000. (**e**) T = 1500. (**f**) T = 2000. (**g**) T = 1000. (**h**) T = 1500. (**i**) T = 2000.

To confirm the natural behaviour of the AO strategy for different noise values, the performance of the AO method is also verified by fixing the population size (30, 40, or 50) and changing the generation size (1000, 1500, or 2000) for three values of noise variance (0.04, 0.06, and 0.08), and the fitness-based learning curves are presented in Figure 6. Figure 6a–c represent the AO fit with population = 30, and the fitness plots for population = 40 are given in Figure 6d,e. Figure 6g–i denotes the fitness plots for population = 50. It is realized from the fitness curves given in Figure 6a–i that for a fixed population and generation size, the AO fit for low levels of noise, i.e., 0.04 and 0.06, is quite lower than the fit for high noise, i.e., 0.08. Nevertheless, AO accomplishes the minimum fit for the smallest noise level, i.e., 0.04, for fixed population size. Therefore, it is confirmed from the curves in Figure 6 that the performance of AO degrades noticeably for higher noise values.

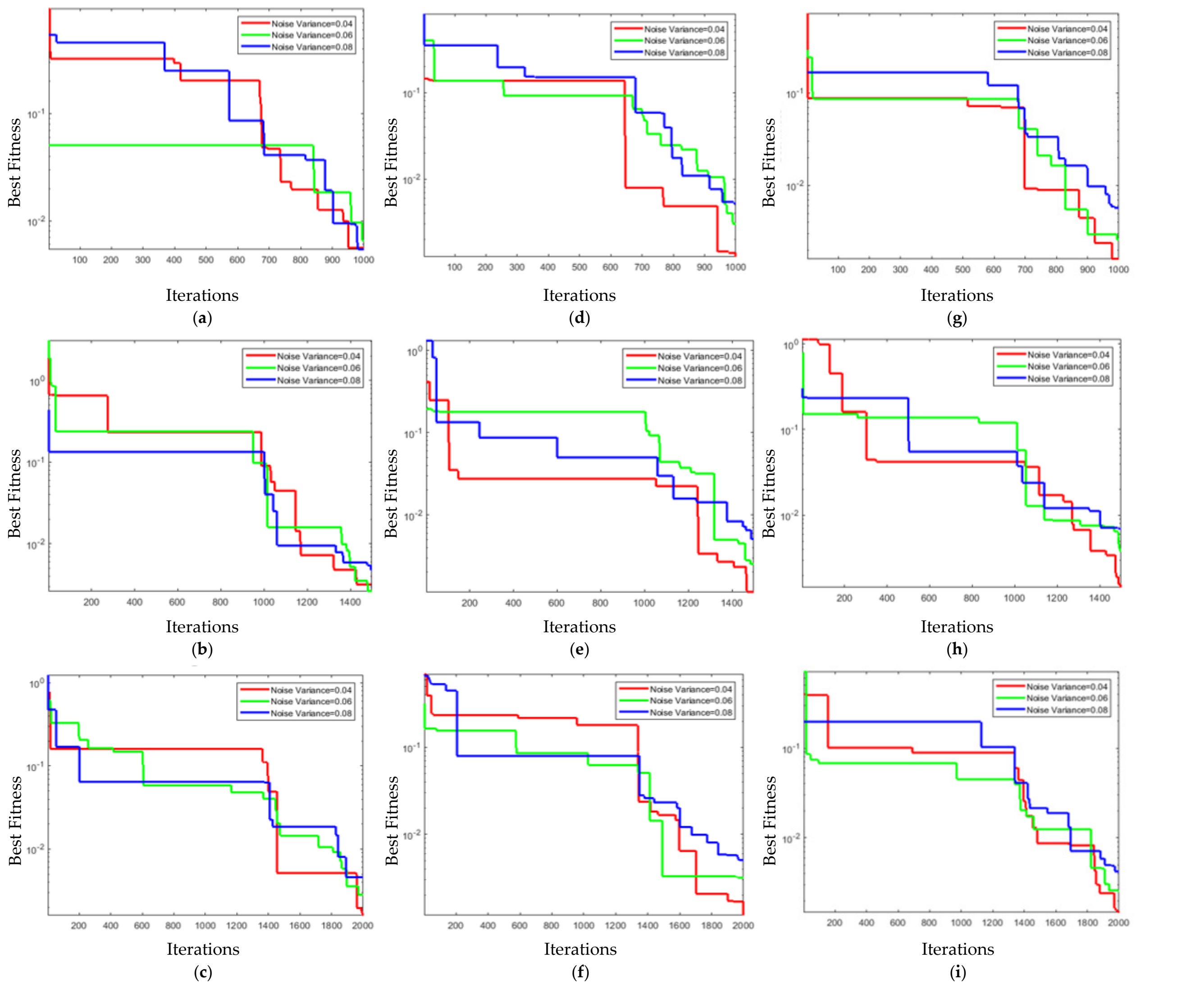


Figure 6. Fitness plots for AO with respect to population sizes. (**a**) Np = 30. (**b**) Np = 30. (**c**) Np = 30. (**d**) Np = 40. (**e**) Np = 40. (**f**) Np = 40. (**g**) Np = 50. (**h**) Np = 50. (**i**) Np = 50.

4.3. Results Comparison with other Heuristics

To further investigate the exploration and exploitation phase of the AO, it is compared with the arithmetic optimization algorithm (AOA) [44], the sine cosine algorithm (SCA) [45], and the reptile search algorithm (RSA) [46] for 15 independent runs with 3 variations of generation (1000, 1500, 2000) and population (30, 40, 50). Tables 15–17 shows the performance of all algorithms in terms of estimated weights and best fit for 0.04, 0.06, and

0.08 noise levels. It is seen that the algorithm gives better results at low noise, i.e., 0.04, than at high noise. Moreover, for low noise, the estimated weights are closer to the true values with minimum fit.

Table 15. Comparison of AO with AOA, SCA, and RSA against the true values for the CAR model at 0.04 noise variance.

Algorithm	Generations (T)	Population (Np)	Design Parameters				Best Fit
			r_1	r_2	s_1	s_2	
AO	1000	30	1.362	0.767	1.692	2.338	1.0×10^{-3}
		40	1.357	0.765	1.706	2.308	1.1×10^{-3}
		50	1.353	0.764	1.712	2.298	1.1×10^{-3}
	1500	30	1.361	0.759	1.702	2.319	1.1×10^{-3}
		40	1.356	0.768	1.704	2.316	1.1×10^{-3}
		50	1.357	0.759	1.703	2.302	1.2×10^{-3}
	2000	30	1.366	0.771	1.723	2.321	1.1×10^{-3}
		40	1.359	0.760	1.686	2.329	1.0×10^{-3}
		50	1.348	0.757	1.717	2.284	1.1×10^{-3}
RSA	1000	30	1.190	0.599	1.740	1.855	5.3×10^{-2}
		40	1.376	0.769	1.619	2.418	3.1×10^{-3}
		50	1.201	0.569	1.371	2.042	1.0×10^{-1}
	1500	30	1.281	0.734	1.663	2.178	1.9×10^{-2}
		40	1.294	0.649	1.384	2.381	2.5×10^{-2}
		50	1.127	0.579	1.704	1.767	8.3×10^{-2}
	2000	30	1.272	0.704	1.174	2.648	5.4×10^{-2}
		40	1.381	0.769	1.526	2.510	7.5×10^{-3}
		50	1.227	0.708	1.719	2.051	3.3×10^{-2}
SCA	1000	30	1.341	0.715	1.646	2.278	2.9×10^{-3}
		40	1.354	0.774	1.725	2.332	3.4×10^{-3}
		50	1.394	0.766	1.678	2.397	2.6×10^{-3}
	1500	30	1.376	0.759	1.662	2.443	4.7×10^{-3}
		40	1.364	0.779	1.572	2.497	2.7×10^{-3}
		50	1.347	0.742	1.626	2.355	9.5×10^{-4}
	2000	30	1.350	0.753	1.694	2.334	2.5×10^{-3}
		40	1.326	0.782	1.600	2.389	4.1×10^{-3}
		50	1.312	0.704	1.691	2.205	2.8×10^{-3}
AOA	1000	30	1.346	0.866	1.494	2.613	1.9×10^{-2}
		40	1.449	0.881	1.465	2.833	1.7×10^{-2}
		50	1.353	0.769	1.555	2.519	8.3×10^{-3}
	1500	30	1.394	0.762	1.791	2.250	6.6×10^{-3}
		40	1.345	0.726	1.649	2.358	6.2×10^{-3}
		50	1.385	0.800	1.613	2.512	3.5×10^{-3}
	2000	30	1.331	0.677	1.455	2.409	1.5×10^{-2}
		40	1.383	0.724	1.569	2.433	7.8×10^{-3}
		50	1.346	0.742	1.763	2.166	8.6×10^{-3}
	True Values		1.350	0.750	1.680	2.320	0

Table 16. Comparison of AO with AOA, SCA, and RSA against the true values for the CAR model at 0.06 noise variance.

Algorithm	Generations (T)	Population (Np)	Design Parameters				Best Fit
			r_1	r_2	s_1	s_2	
AO	1000	30	1.371	0.771	1.701	2.339	2.3×10^{-3}
		40	1.369	0.777	1.715	2.336	2.3×10^{-3}
		50	1.368	0.773	1.700	2.340	2.3×10^{-3}
	1500	30	1.366	0.771	1.744	2.300	2.4×10^{-3}
		40	1.362	0.761	1.733	2.291	2.5×10^{-3}
		50	1.356	0.769	1.718	2.305	2.3×10^{-3}
	2000	30	1.367	0.770	1.711	2.327	2.3×10^{-3}
		40	1.364	0.776	1.723	2.317	2.3×10^{-3}
		50	1.363	0.772	1.706	2.333	2.3×10^{-3}
RSA	1000	30	1.200	0.657	1.159	2.531	8.1×10^{-2}
		40	1.202	0.600	1.789	1.882	5.9×10^{-2}
		50	1.183	0.580	1.425	2.118	6.1×10^{-2}
	1500	30	1.120	0.582	1.584	1.850	9.7×10^{-2}
		40	1.253	0.645	1.091	2.681	7.2×10^{-2}
		50	1.265	0.683	1.431	2.396	2.4×10^{-2}
	2000	30	1.102	0.600	1.888	1.557	1.3×10^{-1}
		40	1.353	0.762	1.341	2.726	3.0×10^{-2}
		50	1.342	0.758	1.349	2.610	2.7×10^{-2}
SCA	1000	30	1.341	0.715	1.646	2.278	6.1×10^{-3}
		40	1.354	0.774	1.725	2.332	4.5×10^{-3}
		50	1.394	0.766	1.678	2.397	3.3×10^{-3}
	1500	30	1.376	0.759	1.662	2.443	3.2×10^{-3}
		40	1.364	0.779	1.572	2.497	2.3×10^{-3}
		50	1.347	0.742	1.626	2.355	2.8×10^{-3}
	2000	30	1.350	0.753	1.694	2.334	2.5×10^{-3}
		40	1.326	0.782	1.600	2.389	2.1×10^{-3}
		50	1.312	0.704	1.691	2.205	2.3×10^{-3}
AOA	1000	30	1.427	0.870	1.470	2.808	1.0×10^{-2}
		40	1.430	0.811	1.295	2.854	2.3×10^{-2}
		50	1.315	0.737	1.738	2.167	6.0×10^{-3}
	1500	30	1.419	0.789	1.446	2.662	1.2×10^{-2}
		40	1.339	0.779	1.702	2.328	4.5×10^{-3}
		50	1.369	0.794	1.077	3.000	1.1×10^{-2}
	2000	30	1.281	0.789	1.599	2.357	1.4×10^{-2}
		40	1.351	0.684	1.579	2.307	1.2×10^{-2}
		50	1.399	0.806	1.754	2.346	7.6×10^{-3}
	True Values		1.350	0.750	1.680	2.320	0

Table 17. Comparison of AO with AOA, SCA, and RSA against the true values for the CAR model at 0.08 noise variance.

Algorithm	Generations (T)	Population (Np)	Design Parameters				Best Fit
			r_1	r_2	s_1	s_2	
AO	1000	30	1.357	0.772	1.754	2.271	4.1×10^{-3}
		40	1.370	0.772	1.715	2.322	4.1×10^{-3}
		50	1.364	0.771	1.712	2.318	4.1×10^{-3}
	1500	30	1.380	0.784	1.737	2.337	4.2×10^{-3}
		40	1.371	0.778	1.707	2.344	4.1×10^{-3}
		50	1.372	0.776	1.722	2.328	4.0×10^{-3}
	2000	30	1.367	0.776	1.741	2.312	4.1×10^{-3}
		40	1.372	0.775	1.746	2.304	4.1×10^{-3}
		50	1.358	0.776	1.763	2.272	4.1×10^{-3}
RSA	1000	30	1.143	0.529	1.575	1.928	1.0×10^{-1}
		40	1.148	0.604	2.135	1.461	1.4×10^{-1}
		50	1.141	0.562	1.283	2.256	1.0×10^{-1}
	1500	30	1.120	0.598	1.752	1.709	1.1×10^{-1}
		40	1.277	0.690	1.318	2.498	3.6×10^{-2}
		50	1.293	0.694	1.210	2.656	4.7×10^{-2}
	2000	30	1.205	0.661	1.549	2.053	7.1×10^{-2}
		40	1.323	0.706	1.275	2.556	4.2×10^{-2}
		50	1.304	0.733	1.536	2.333	2.3×10^{-2}
SCA	1000	30	1.416	0.831	1.623	2.574	7.9×10^{-3}
		40	1.359	0.806	1.688	2.380	6.4×10^{-3}
		50	1.350	0.738	1.705	2.289	4.1×10^{-3}
	1500	30	1.364	0.801	1.602	2.486	4.7×10^{-3}
		40	1.375	0.788	1.679	2.379	4.0×10^{-3}
		50	1.317	0.755	1.695	2.255	5.3×10^{-3}
	2000	30	1.344	0.773	1.623	2.366	3.9×10^{-3}
		40	1.399	0.778	1.658	2.429	4.5×10^{-3}
		50	1.340	0.713	1.667	2.250	5.1×10^{-3}
AOA	1000	30	1.427	0.870	1.470	2.808	2.1×10^{-2}
		40	1.430	0.811	1.295	2.854	1.0×10^{-2}
		50	1.315	0.737	1.738	2.167	1.1×10^{-2}
	1500	30	1.419	0.789	1.446	2.662	1.4×10^{-2}
		40	1.339	0.779	1.702	2.328	1.3×10^{-2}
		50	1.369	0.794	1.077	3.000	1.3×10^{-2}
	2000	30	1.281	0.789	1.599	2.357	7.6×10^{-3}
		40	1.351	0.684	1.579	2.307	5.7×10^{-3}
		50	1.399	0.806	1.754	2.346	9.8×10^{-3}
	True Values		1.350	0.750	1.680	2.320	0

Tables 18–20 show the performance of the AO, AOA, RSA, and SCA algorithms in terms of average fit for all noise variances. It is seen that for all noise variances, the AO algorithm gives better results as compared with RSA, SCA, and AOA for generations and populations.

Table 18. Comparison of AO with RSA, SCA, and AOA against average fit for the CAR model at 0.04 noise variance.

Generations (T)	Population (Np)	AO	RSA	AOA	SCA
1000	30	3.8×10^{-3}	6.1×10^{-1}	4.1×10^{-2}	1.0×10^{-1}
	40	2.0×10^{-3}	2.5×10^{-1}	4.7×10^{-2}	6.4×10^{-2}
	50	1.8×10^{-3}	2.7×10^{-1}	2.8×10^{-2}	1.0×10^{-1}
1500	30	2.0×10^{-3}	5.0×10^{-1}	3.0×10^{-2}	1.2×10^{-1}
	40	1.7×10^{-3}	2.9×10^{-1}	3.3×10^{-2}	5.9×10^{-2}
	50	1.7×10^{-3}	2.5×10^{-1}	3.0×10^{-2}	3.7×10^{-2}
2000	30	1.9×10^{-3}	3.7×10^{-1}	2.9×10^{-2}	1.2×10^{-1}
	40	1.7×10^{-3}	2.8×10^{-1}	2.4×10^{-2}	8.7×10^{-2}
	50	1.9×10^{-3}	2.1×10^{-1}	2.2×10^{-2}	6.5×10^{-2}

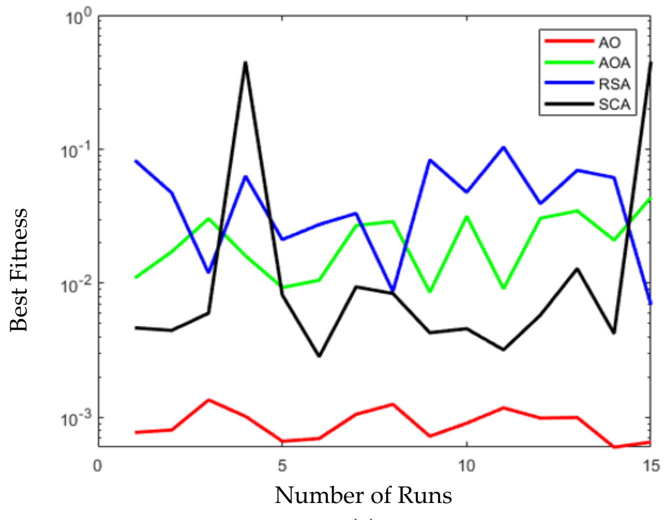
Table 19. Comparison of AO with RSA, SCA, and AOA against average fit for the CAR model at 0.06 noise variance.

Generations (T)	Population (Np)	AO	RSA	AOA	SCA
1000	30	4.6×10^{-3}	3.9×10^{-1}	5.0×10^{-2}	6.9×10^{-2}
	40	3.3×10^{-3}	2.5×10^{-1}	5.1×10^{-2}	1.0×10^{-1}
	50	3.3×10^{-3}	2.7×10^{-1}	3.2×10^{-2}	6.5×10^{-2}
1500	30	3.6×10^{-3}	6.5×10^{-1}	3.6×10^{-2}	1.7×10^{-1}
	40	3.4×10^{-3}	2.6×10^{-1}	3.5×10^{-2}	1.9×10^{-1}
	50	2.9×10^{-3}	2.3×10^{-1}	2.8×10^{-2}	7.8×10^{-2}
2000	30	3.1×10^{-3}	3.4×10^{-1}	4.0×10^{-2}	1.6×10^{-1}
	40	2.8×10^{-3}	5.4×10^{-1}	3.5×10^{-2}	7.0×10^{-2}
	50	2.7×10^{-3}	2.1×10^{-1}	2.3×10^{-2}	7.1×10^{-3}

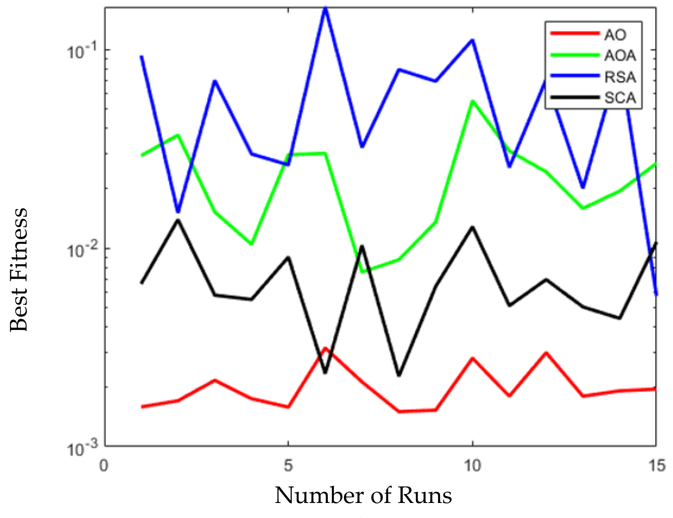
Table 20. Comparison of AO with RSA, SCA, and AOA against average fit for the CAR model at 0.08 noise variance.

Generations (T)	Population (Np)	AO	RSA	AOA	SCA
1000	30	6.4×10^{-3}	4.3×10^{-1}	4.3×10^{-2}	2.3×10^{-1}
	40	5.0×10^{-3}	6.7×10^{-1}	2.5×10^{-2}	1.0×10^{-1}
	50	5.0×10^{-3}	4.3×10^{-1}	3.1×10^{-2}	1.0×10^{-1}
1500	30	5.1×10^{-3}	3.0×10^{-1}	3.5×10^{-2}	9.2×10^{-2}
	40	4.7×10^{-3}	2.5×10^{-1}	3.1×10^{-2}	6.5×10^{-2}
	50	4.8×10^{-3}	2.9×10^{-1}	3.9×10^{-2}	1.1×10^{-1}
2000	30	4.6×10^{-3}	4.3×10^{-1}	2.9×10^{-2}	1.2×10^{-1}
	40	4.6×10^{-3}	2.7×10^{-1}	2.6×10^{-2}	1.5×10^{-1}
	50	4.5×10^{-3}	2.6×10^{-1}	2.5×10^{-2}	1.3×10^{-1}

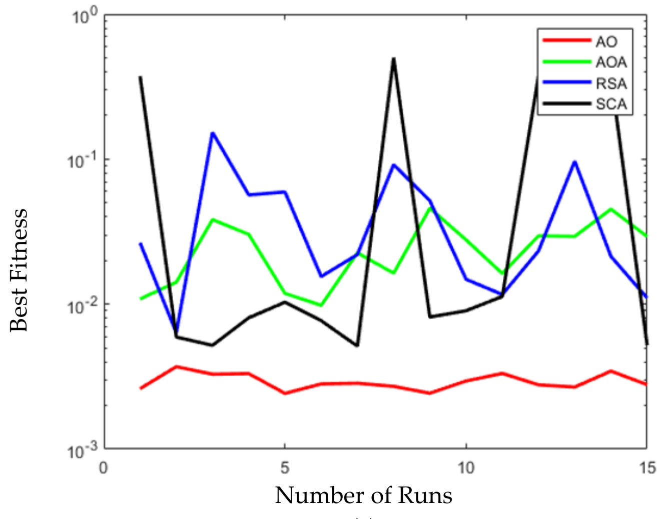
The statistical analysis of AO, SCA, AOA, and RSA for multiple runs, noise variances, and population sizes and constant generation size are shown in Figure 7. It is witnessed that for all noise variances, the AO achieves worse fit as compared with RSA, AOA, and SCA. It is also observed that by increasing the noise level degrades the performance of all algorithms. However, AO achieves optimal fit in all scenarios.



(**a**)



(**b**)



(**c**)

Figure 7. Statistical analyses plots of AO, AOA, SCA, and RSA for Np = 50, T = 2000. (**a**) Noise level = 0.04. (**b**) Noise level = 0.06. (**c**) Noise level = 0.08.

To further investigate the performance of AO vs. RSA, AO vs. AOA, and AO vs. SCA, a nonparametric Mann-Whitney U test [47] is performed on average fit values for all algorithms on noise variances 0.04, 0.06, and 0.08 with generations 1000, 1500, and 2000 and populations 30, 40, or 50. The Mann-Whitney U test is a parametric equivalent of two sample *t* test. The significance level is 0.01 and one tailed hypothesis is used. The computed z-score is −6.29719 and *p*-value is less than 0.00001. Moreover, the result is significant at $p < 0.01$ as presented in Figures 8–10.

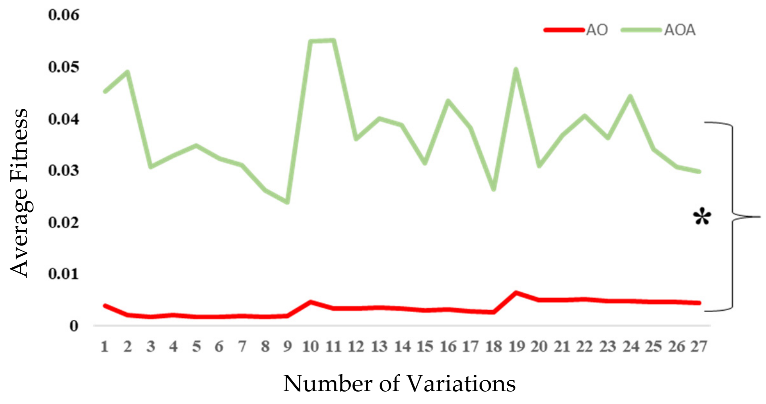


Figure 8. Mann-Whitney U test between AO and AOA where * $p < 0.01$.

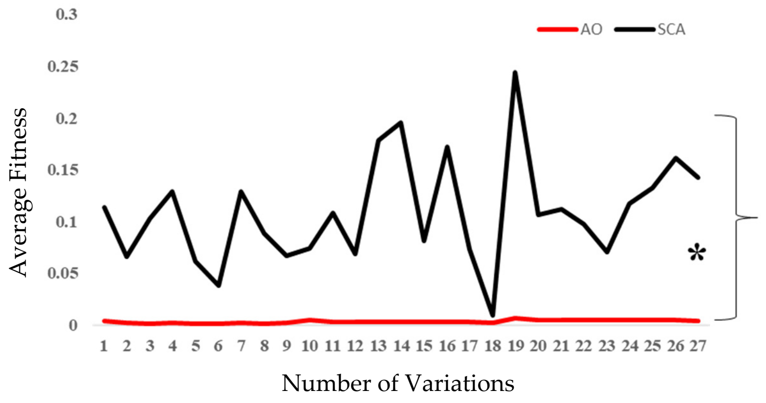


Figure 9. Mann-Whitney U test between AO and SCA where * $p < 0.01$.

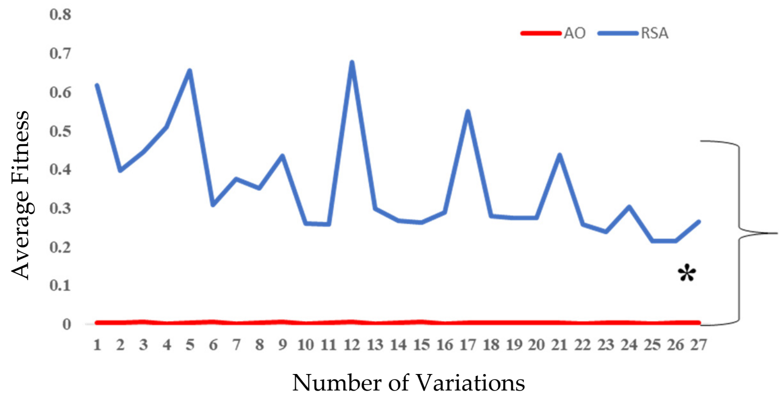


Figure 10. Mann-Whitney U test between AO and RSA where * $p < 0.01$.

The results of detailed simulations and the statistics indicate that the AO based swarming optimization heuristics effectively estimates the parameters of the CAR systems. However, the real time implementation of the swarm optimization algorithms for practical system identification problems require further investigation.

5. Conclusions

Following are the conclusions drawn from the extensive simulation results presented in the last section:

The strength of swarm intelligence of the Aquila optimizer, AO, is effectively exploited for parameter estimation of control autoregressive, CAR, systems. Performance of the AO is enhanced by escalating the population and generation at the expense of computational cost. While, the optimal fitness for different generations and populations is achieved for $\beta = 0.1$ and $\mu = 0.1$ exploitation adjustable factors. The robustness and accuracy of the AO decreases by varying the noise level. The comparative study of the AO with other heuristics based on AOA, SCA and RSA established the efficacy of the proposed scheme. The statistical analysis through Mann-Whitney U test endorsed the reliability of the AO scheme for CAR system identification The current study expands the application domain of swarm intelligence based optimizers by exploiting the strength of AO approach for system identification. However, future work may consider applying the proposed methodology of the for solving complex problems [48–54].

Author Contributions: Conceptualization, K.M. and N.I.C.; methodology, K.M.; software, K.M.; validation, N.I.C., M.A.Z.R. and Z.A.K.; formal analysis, Z.A.K.; writing—original draft preparation, K.M.; writing—review and editing, N.I.C., Z.A.K. and M.A.Z.R.; project administration, K.M.C. and A.H.M.; funding acquisition, K.M.C. and A.H.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

References

1. Mehmood, A.; Raja, M.A.Z.; Shi, P.; Chaudhary, N.I. Weighted differential evolution-based heuristic computing for identification of Hammerstein systems in electrically stimulated muscle modeling. *Soft Comput.* **2022**, 1–17. [CrossRef]
2. Chaudhary, N.I.; Raja, M.A.Z.; Khan, Z.A.; Mehmood, A.; Shah, S.M. Design of fractional hierarchical gradient descent algorithm for parameter estimation of nonlinear control autoregressive systems. *Chaos Solitons Fractals* **2022**, *157*, 111913. [CrossRef]
3. Chaudhary, N.I.; Raja, M.A.Z.; Khan, Z.A.; Cheema, K.M.; Milyani, A.H. Hierarchical Quasi-Fractional Gradient Descent Method for Parameter Estimation of Nonlinear ARX Systems Using Key Term Separation Principle. *Mathematics* **2021**, *9*, 3302. [CrossRef]
4. Shen, Q.; Ding, F. Least squares identification for Hammerstein multi-input multi-output systems based on the key-term separation technique. *Circuits Syst. Signal Process.* **2016**, *35*, 3745–3758. [CrossRef]
5. Begum, N.; Dadashpour, M.; Kleppe, J. Subchapter 1.6—A case study of reservoir parameter estimation in Norne oil field, Norway by using Ensemble Kalman Filter (EnKF). In *Innovative Exploration Methods for Minerals, Oil, Gas, and Groundwater for Sustainable Development*; Moitra, A.K., Kayal, J.R., Mukerji, B., Bhattacharya, J., Eds.; Elsevier: Amsterdam, The Netherlands, 2022; pp. 61–78. [CrossRef]
6. Hwang, J.K.; Shin, J. Identification of interarea modes from ambient data of phasor measurement units using an autoregressive exogenous model. *IEEE Access* **2021**, *9*, 45695–45705. [CrossRef]
7. Javed, U.; Ijaz, K.; Jawad, M.; Ansari, E.A.; Shabbir, N.; Kütt, L.; Husev, O. Exploratory Data Analysis Based Short-Term Electrical Load Forecasting: A Comprehensive Analysis. *Energies* **2021**, *14*, 5510. [CrossRef]
8. Dong, G.; Chen, Z.; Wei, J. Sequential monte carlo filter for state-of-charge estimation of lithium-ion batteries based on auto regressive exogenous model. *IEEE Trans. Ind. Electron.* **2019**, *66*, 8533–8544. [CrossRef]
9. Basu, B.; Morrissey, P.; Gill, L.W. Application of nonlinear time series and machine learning algorithms for forecasting groundwater flooding in a lowland karst area. *Water Resour. Res.* **2022**, *58*, e2021WR029576. [CrossRef]
10. Shabani, E.; Ghorbani, M.A.; Inyurt, S. The power of the GP-ARX model in CO2 emission forecasting. In *Risk, Reliability and Sustainable Remediation in the Field of Civil and Environmental Engineering*; Elsevier: Amsterdam, The Netherlands, 2022; pp. 79–91. [CrossRef]
11. Ding, F.; Lv, L.; Pan, J.; Wan, X.; Jin, X.-B. Two-stage gradient-based iterative estimation methods for controlled autoregressive systems using the measurement data. *Int. J. Control Autom. Syst.* **2020**, *18*, 886–896. [CrossRef]
12. Raja, M.A.Z.; Shah, A.A.; Mehmood, A.; Chaudhary, N.I.; Aslam, M.S. Bio-inspired computational heuristics for parameter estimation of nonlinear Hammerstein controlled autoregressive system. *Neural Comput. Appl.* **2018**, *29*, 1455–1474. [CrossRef]
13. Mehmood, A.; Zameer, A.; Raja MA, Z.; Bibi, R.; Chaudhary, N.I.; Aslam, M.S. Nature-inspired heuristic paradigms for parameter estimation of control autoregressive moving average systems. *Neural Comput. Appl.* **2019**, *31*, 5819–5842. [CrossRef]

14. Tariq, H.B. Maximum-Likelihood-Based Adaptive and Intelligent Computing for Nonlinear System Identification. *Mathematics* **2021**, *9*, 3199. [CrossRef]
15. Dong, J.; Wang, Z.; Mo, J. A phase angle-modulated bat algorithm with application to antenna topology optimization. *Appl. Sci.* **2021**, *11*, 2243. [CrossRef]
16. Abualigah, L.; Elaziz, M.A.; Khasawneh, A.M.; Alshinwan, M.; Ibrahim, R.A.; Al-qaness, M.A.; Mirjalili, S.; Sumari, P.; Gandomi, A.H. Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: A comprehensive survey, applications, comparative analysis, and results. *Neural Comput. Appl.* **2022**, *34*, 4081–4110. [CrossRef]
17. Wang, S.; Jia, H.; Abualigah, L.; Liu, Q.; Zheng, R. An improved hybrid aquila optimizer and harris hawks algorithm for solving industrial engineering optimization problems. *Processes* **2021**, *9*, 1551. [CrossRef]
18. Altaf, F.; Chang, C.L.; Chaudhary, N.I.; Raja, M.A.Z.; Cheema, K.M.; Shu, C.M.; Milyani, A.H. Adaptive Evolutionary Computation for Nonlinear Hammerstein Control Autoregressive Systems with Key Term Separation Principle. *Mathematics* **2022**, *10*, 1001. [CrossRef]
19. Storn, R.; Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
20. Malik, M.F.; Chang, C.L.; Aslam, M.S.; Chaudhary, N.I.; Raja, M.A.Z. Fuzzy-Evolution Computing Paradigm for Fractional Hammerstein Control Autoregressive Systems. *Int. J. Fuzzy Syst.* **2022**, 1–29. [CrossRef]
21. Cui, T.; Xu, L.; Ding, F.; Alsaedi, A.; Hayat, T. Maximum likelihood-based adaptive differential evolution identification algorithm for multivariable systems in the state-space form. *Int. J. Adapt. Control Signal Process.* **2020**, *34*, 1658–1676. [CrossRef]
22. Cheraghalipour, A.; Hajiaghaei-Keshteli, M.; Paydar, M.M. Tree Growth Algorithm (TGA): A novel approach for solving optimization problems. *Eng. Appl. Artif. Intell.* **2018**, *72*, 393–414. [CrossRef]
23. Zhang, Q.; Wang, R.; Yang, J.; Ding, K.; Li, Y.; Hu, J. Collective decision optimization algorithm: A new heuristic optimization method. *Neurocomputing* **2017**, *221*, 123–137. [CrossRef]
24. Atashpaz-Gargari, E.; Lucas, C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 4661–4667. [CrossRef]
25. Rao, R.V.; Savsani, V.J.; Vakharia, D.P. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput. Des.* **2011**, *43*, 303–315. [CrossRef]
26. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
27. Kaveh, A.; Dadras, A. A novel meta-heuristic optimization algorithm: Thermal exchange optimization. *Adv. Eng. Softw.* **2017**, *110*, 69–84. [CrossRef]
28. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [CrossRef]
29. Malik, N.A.; Chang, C.L.; Chaudhary, N.I.; Raja, M.A.Z.; Cheema, K.M.; Shu, C.M.; Alshamrani, S.S. Knacks of Fractional Order Swarming Intelligence for Parameter Estimation of Harmonics in Electrical Systems. *Mathematics* **2022**, *10*, 1570. [CrossRef]
30. Karaboga, D.; Basturk, B. On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.* **2008**, *8*, 687–697. [CrossRef]
31. Yazdani, M.; Jolai, F. Lion optimization algorithm (LOA): A nature-inspired metaheuristic algorithm. *J. Comput. Des. Eng.* **2016**, *3*, 24–36. [CrossRef]
32. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
33. Abualigah, L.; Yousri, D.; Elaziz, M.A.; Ewees, A.A.; Al-qaness, M.A.A.; Gandomi, A.H. Aquila Optimizer: A novel meta-heuristic optimization Algorithm. *Comput. Ind. Eng.* **2021**, *157*, 107250. [CrossRef]
34. Elaziz, M.A.; Dahou, A.; Alsaleh, N.A.; Elsheikh, A.H.; Saba, A.I.; Ahmadein, M. Boosting COVID-19 Image Classification Using MobileNetV3 and Aquila Optimizer Algorithm. *Entropy* **2021**, *23*, 1383. [CrossRef] [PubMed]
35. Hussan, M.R.; Sarwar, M.I.; Sarwar, A.; Tariq, M.; Ahmad, S.; Shah Noor Mohamed, A.; Khan, I.A.; Ali Khan, M.M. Aquila Optimization Based Harmonic Elimination in a Modified H-Bridge Inverter. *Sustainability* **2022**, *14*, 929. [CrossRef]
36. Khamees, K.; Abdelaziz, A.Y.; Eskaros, M.R.; Alhelou, H.H.; Attia, M.A. Stochastic Modeling for Wind Energy and Multi-Objective Optimal Power Flow by Novel Meta-Heuristic Method. *IEEE Access* **2021**, *9*, 158353–158366. [CrossRef]
37. Fatani, A.; Dahou, A.; Al-Qaness, M.A.; Lu, S.; Elaziz, M.A. Advanced Feature Extraction and Selection Approach Using Deep Learning and Aquila Optimizer for IoT Intrusion Detection System. *Sensors* **2022**, *22*, 140. [CrossRef]
38. Rajinikanth, V.; Aslam, S.M.; Kadry, S.; Thinnukool, O. Semi/Fully-Automated Segmentation of Gastric-Polyp Using Aquila-Optimization-Algorithm Enhanced Images. *CMC-Comput. Mater. Contin.* **2022**, *70*, 4087–4105. [CrossRef]
39. AlRassas, A.M.; Al-qaness, M.A.; Ewees, A.A.; Ren, S.; Abd Elaziz, M.; Damaševičius, R.; Krilavičius, T. Optimized ANFIS model using Aquila Optimizer for oil production forecasting. *Processes* **2021**, *9*, 1194. [CrossRef]
40. Vashishtha, G.; Kumar, R. Autocorrelation energy and aquila optimizer for MED filtering of sound signal to detect bearing defect in Francis turbine. *Meas. Sci. Technol.* **2021**, *33*, 15006. [CrossRef]
41. Wang, S.; Ma, J.; Li, W.; Khayatnezhad, M.; Rouyendegh, B.D. An optimal configuration for hybrid SOFC, gas turbine, and Proton Exchange Membrane Electrolyzer using a developed Aquila Optimizer. *Int. J. Hydrogen Energy* **2022**, *47*, 8943–8955. [CrossRef]
42. Khamees, K.; Abdelaziz, A.Y.; Eskaros, M.R.; El-Shahat, A.; Attia, M.A. Optimal Power Flow Solution of Wind-Integrated Power System Using Novel Metaheuristic Method. *Energies* **2021**, *14*, 6117. [CrossRef]

43. Ma, L.; Li, J.; Zhao, Y. Population Forecast of China's Rural Community Based on CFANGBM and Improved Aquila Optimizer Algorithm. *Fractal Fract.* **2021**, *5*, 190. [CrossRef]
44. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The arithmetic optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113609. [CrossRef]
45. Mirjalili, S. SCA: A sine cosine algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
46. Abualigah, L.; Abd Elaziz, M.; Sumari, P.; Geem, Z.W.; Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. *Expert Syst. Appl.* **2022**, *191*, 116158. [CrossRef]
47. MacFarland, T.W.; Yates, J.M. Mann–whitney u test. In *Introduction to Nonparametric Statistics for the Biological Sciences Using R*; Springer: Cham, Switzerland, 2016; pp. 103–132. [CrossRef]
48. Stodola, P. Using metaheuristics on the multi-depot vehicle routing problem with modified optimization criterion. *Algorithms* **2018**, *11*, 74. [CrossRef]
49. Stodola, P.; Michenka, K.; Nohel, J.; Rybanský, M. Hybrid algorithm based on ant colony optimization and simulated annealing applied to the dynamic traveling salesman problem. *Entropy* **2020**, *22*, 884. [CrossRef]
50. Stodola, P.; Mazal, J. Model of optimal cooperative reconnaissance and its solution using metaheuristic methods. *Def. Sci. J.* **2017**, *67*, 529. [CrossRef]
51. Stodola, P.; Mazal, J. Applying the ant colony optimisation algorithm to the capacitated multi-depot vehicle routing problem. *Int. J. Bio-Inspired Comput.* **2016**, *8*, 228–233. [CrossRef]
52. Stodola, P.; Mazal, J. Tactical models based on a multi-depot vehicle routing problem using the ant colony optimization algorithm. *Int. J. Math. Models Methods Appl. Sci.* **2015**, *9*, 330–337. Available online: https://www.naun.org/main/NAUN/ijmmas/2015/a782001-387.pdf (accessed on 10 February 2022).
53. Chaudhary, N.I.; Raja, M.A.Z.; He, Y.; Khan, Z.A.; Machado, J.T. Design of multi innovation fractional LMS algorithm for parameter estimation of input nonlinear control autoregressive systems. *Appl. Math. Model.* **2021**, *93*, 412–425. [CrossRef]
54. Chaudhary, N.I.; Latif, R.; Raja, M.A.Z.; Machado, J.T. An innovative fractional order LMS algorithm for power signal parameter estimation. *Appl. Math. Model.* **2020**, *83*, 703–718. [CrossRef]