Article

Optimal Neural Tracking Control with Metaheuristic Parameter Identification for Uncertain Nonlinear Systems with Disturbances

Roxana Recio-Colmenares, Kelly Joel Gurubel-Tun * and Virgilio Zúñiga-Grajeda *

School of Engineering and Technological Innovation, University of Guadalajara, Campus Tonalá, Jalisco 45425, Mexico; roxana.recio@academicos.udg.mx

* Correspondence: joel.gurubel@academicos.udg.mx (K.J.G.-T.); virgilio.zuniga@academicos.udg.mx (V.Z.-G.)

Received: 29 August 2020; Accepted: 9 October 2020; Published: 12 October 2020

Abstract: In this paper, we propose an inverse optimal neural control strategy for uncertain nonlinear systems subject to external disturbances. This control strategy is developed based on a neural observer for the estimation of unmeasured states and inverse optimal control theory for trajectory tracking. The stabilization of states along the desired trajectory is ensured via a control Lyapunov function. The optimal parameters of the control law are identified by different nature-inspired metaheuristic algorithms, namely: Ant Lion Optimizer, Grey Wolf Optimizer, Harris Hawks Optimization, and Whale Optimization Algorithm. Finally, a highly nonlinear biological system subject to parameter uncertainties and external disturbances (Activated Sludge Model) is proposed to validate the control strategy. Simulation results demonstrate that the control law with Ant Lion Optimizer outperforms the other optimization methods in terms of trajectory tracking in the presence of disturbances. The control law with Harris Hawks Optimization shows a better convergence of the neural states in presence of parameter uncertainty.

Keywords: optimal control; artificial neural network; metaheuristic optimization; nonlinear systems

1. Introduction

Researchers solving problems in engineering, computer science, biological, and physical sciences are developing advanced mathematical methods for control purposes. Technological advances have had a significant impact on the implementation of new analytical methods for solving nonlinear problems [1]. In most cases involving nonlinear control systems, states may not be completely measurable, which makes it difficult to solve complex control engineering problems. Other major difficulties in the area of control design are the use of a range of different models and concepts, a lack of standardization in the parameters, a lack of adequate control techniques, external disturbances, and the high level of nonlinearity of the equations that govern processes. A further issue is a lack of knowledge of important variables, since the states of the system can directly reflect the nature of the control design process allowing for high performance to be achieved [2]. It can therefore be stated that it is necessary to use advanced estimation, control, and optimization techniques to guarantee optimal nonlinear system performance. Knowledge of the system is important to understand its control requirements; however, the nonlinearities are often so complex that the control design for an adequate performance of the system is difficult [3–6]. Over time, new control strategies have emerged to ensure optimal system performance, and the estimation of states is vitally important for the development of feedback laws that allow for the avoidance of interruptions, stops, or process failures. For these reasons, it is important to develop devices called observers to estimate the missing states, and at the same time, to address the lack of online sensor measurements, the high cost of sensors, and the long laboratory analysis times [7,8]. Artificial

neural networks have successfully demonstrated their feasibility and excellent performance in terms of describing highly nonlinear systems [9,10]. This is mainly due to their simplicity of deployment flexible architecture, robustness, and online learning. Several researchers have focused on the applications of neural networks applications to the identification, performance evaluation, and control of nonlinear plants [11,12]. However, most of these schemes require a complete knowledge of the variables of the mathematical model, which is not always possible. For this reason, Recurrent High Order Neural Networks (RHONN) have proven to be very useful for identification and control problems, and artificial neural networks and control strategies have been proposed to increase the system performance [13,14]. One of the main challenges in optimal control of nonlinear systems is tuning its parameters for an optimal solution. This problem has been addressed in the past by using metaheuristic techniques, since they have proved to be effective when dealing with complex systems. Metaheuristic optimization algorithms are well suited for problems with unknown search spaces, unlike deterministic algorithms. These optimization techniques find an optimal solution from a set of candidate solutions and have been used in almost every branch of science, technology, and engineering. In the last few years, a subset of metaheuristic algorithms inspired from nature, called swarm intelligence, have become more popular due to their simplicity, versatility, and flexibility [15]. Ant Lion Optimizer (ALO) has been successfully used in modeling, control, scheduling, and management with application in distributed generation, mechanics, thermal systems, and renewable energy, among others [16–19]. In [20,21], an ALO was used for parameter identification of the controller gains in a two-area thermal power system. The objective was to obtain optimal values for the control law, speed regulation, and frequency for both areas in presence of disturbances. A comparison of ALO performance with other algorithms based on standard deviations and objective functions was reported. Results showed that ALO algorithm outperforms the other optimization methods for a real problem, by obtaining precise values for the regulator design. Another nature-inspired technique, namely, Grey Wolf Optimizer (GWO) [22], has been used in [23] to control a quadrotor dynamic model and stability prediction of power systems after a large disturbance [24]. Whale Optimization Algorithm (WOA) [25] is another technique used to fine-tune neural networks to obtain more reliable models [26]. Another recently invented algorithm, Harris Hawks Optimization (HHO) [27], has been applied to optimize a controller to dampen power system oscillations [28]. Although there are many examples where these algorithms have been studied in the past, there are fewer studies that have tackled the problem of external disturbances and parameter uncertainty.

In this paper, an optimal tracking neural control strategy with metaheuristic parameter optimization is proposed for highly nonlinear, uncertain systems subject to external disturbances. In Section 2, a neural observer based on a RHONN is proposed to estimate the unmeasured states needed to develop a state feedback control law. Section 3 describes the stabilizing control law based on the inverse optimal control law theory. In Section 4, the novelty of this research is presented, the use of the newest metaheuristic algorithms, namely: ALO, GWO, HHO, and WOA for optimal parameter estimation in order to automatically find the optimal parameters of the control law. Section 5 describes the optimal control strategy architecture. To validate the efficacy of this control strategy, Section 6 presents a highly nonlinear biological system subject to disturbances as a case study. This model is applied to describe the aerobic digestion of organic matter in a continuous stirred tank reactor (CSTR). In Section 7, simulation results of the control strategy with metaheuristic algorithms are presented. Two cases are analyzed: the system subject to load disturbances and subject to uncertainties in the composition of the incoming wastewater. The total chemical oxygen demand (TCOD) is the variable of interest to be controlled by an optimal oxygen transfer (KLA) rate. Optimal operation of the wastewater treatment plant can improve energy efficiency and increase the treatment capacity. Finally, relevant conclusions are given in Section 8.

2. Neural Observer Based on RHONN

A discrete-time nonlinear system is considered as follows

$$\begin{aligned} z_{k+1} &= f(z_k) + g_s(z_k)u_k + d_k \\ y_k &= h(z_k) \end{aligned} \quad (1)$$

where $z_k \in R^n$ is the state variable of the system at time $k \in N_+$, $u_k \in R^n$ is the input control, $d_k \in R^n$ is the disturbance vector representing modeling errors and uncertain parameters, and $f(z_k) \in R^n$ and $g_s(z_k) \in R^{n \times m}$ are smooth mapping functions with $f(0) = 0$ and $\text{rank}\{g_s(z_k)\} = m, \forall z_k \neq 0$. y_k is the output vector and $h(z_k) \in R^{p \times n}$ is a nonlinear function of the system states.

For system (1), a RHONN is used to design a reduced order Luenberger-like observer with the following structure [29]:

$$\begin{aligned} \hat{z}_{i,k+1} &= w_{i,k}^T \theta_i(\hat{z}_k) + w_i'^T \varphi_i(\hat{z}_k, u_k) + g_i e_k \\ \hat{y}_k &= h(\hat{z}_k) \end{aligned} \quad (2)$$

where $\hat{z}_k = [\hat{z}_{1,k} \dots \hat{z}_{i,k} \dots \hat{z}_{n,k}]^T$ is the estimated state vector, $w_{i,k} \in R^n$ are the online adapted weight vectors, $w_i'^T$ is the fixed weight vector associated with the control inputs to ensure controllability of the observer, φ_i denotes a function of $\hat{z}_k$ or u_k corresponding to the plant structure (1) or external inputs to the network, respectively. $g_i \in R^p$ are the Luenberger gains, e_k is the output error, and θ_i is an L_p dimensional vector defined as

$$\theta_i(z_k, u_k) = \begin{bmatrix} \prod_{t \in I_1} S_{i_t}^{n_{i_t}(1)} \\ \prod_{t \in I_2} S_{i_t}^{n_{i_t}(2)} \\ \vdots \\ \prod_{t \in I_{L_p}} S_{i_t}^{n_{i_t}(L_p)} \end{bmatrix} \quad (3)$$

where n_{i_t} are nonnegative integers, L_p is the number of high-order connections, and I_{L_p} is a collection of nonordered subsets of $\{1, 2, \dots, n + m\}$. S_i is a sigmoid function vector defined as:

$$s(z_k) = \frac{\alpha_i}{1 + e^{-\beta_i}}; \alpha_i, \beta_i > 0 \quad (4)$$

The output error is defined by:

$$e_k = y_k - \hat{y}_k. \quad (5)$$

The output error can be minimized by an appropriate selection of L_p . w_i^* and $w_i'^*$ are assumed ideal weight matrices for analytical purposes, and these are constant but unknown. The optimal unknown weights $w_{i,k}^*$ are approximated by online adjustable versions $w_{i,k}$.

For online learning of the RHONN weights, an Extended Kalman Filter (EKF) is used [30]. EKF training of recurrent neural networks has proven its applicability to extensive difficult modeling and control problems. The sequential nature of the EKF provides a stochastic component that allows for more effective search of the weight space. The EKF methods are easily implemented in software and hardware. Perhaps the most significant limitation of EKF training is its limited applicability to cost functions other than minimizing sum of squared error. For the proposed neural network training, the cost function criterion is satisfied. The weights of the neural network become the estimated states. The neural network training to find the optimal weight values is as follows:

$$\begin{aligned} w_{i,k+1} &= w_{i,k} + \eta_i K_{i,k} e_k, \\ K_{i,k} &= P_{i,k} H_{i,k}^T M_{i,k}^{-1} \\ P_{i,k+1} &= P_{i,k} - K_{i,k} H_{i,k}^T P_{i,k} Q_{i,k} \\ M_{i,k} &= [R_{i,k} + H_{i,k}^T P_{i,k} H_{i,k}]^{-1} \end{aligned} \quad (6)$$

where $w_{i,k} \in R^{L_i}$ is the weight vector, η_i is the learning rate, $K_{i,k} \in R^{L_i \times p}$ is the Kalman gain matrix, $P_{i,k} \in R^{L_i \times L_i}$ is the prediction error associated covariance matrix, $Q_{i,k} \in R^{L_i \times L_i}$ is the state noise associated covariance matrix, $M_{i,k} \in R^{p \times p}$ is a global scaling matrix, $R_{i,k} \in R^{p \times p}$ is the measurement noise associated covariance matrix, and $H_{i,k} \in R^{L_i \times n}$ is a matrix of the derivatives of the network outputs with respect to all trainable neural network weights as follows:

$$H_{i,k} = \left[\frac{\partial \hat{y}_k}{\partial w_{i,k}} \right] \quad (7)$$

L_i is the total number of neural network weights. P_i , Q_i , and R_i are initialized as diagonal matrices. During training, $K_{i,k}$, $P_{i,k}$, and $H_{i,k}$ are ensured to be bounded [31].

3. Discrete-Time Inverse Optimal Control

Consider the following cost function related to trajectory tracking of the nonlinear system in (1)

$$J(\delta_k) = \sum_{n=k}^{\infty} (l(\delta_n) + u_n^T R_z u_n) \quad (8)$$

where $\delta = z_k - z_{r,k}$; $z_{r,k}$ is the desired trajectory for z_k , $J(\delta_k) : R^n \rightarrow R^+$; $l(\delta_k) : R^n \rightarrow R^+$ is a positive semidefinite function, and $R_z : R^n \rightarrow R^+$ is a real weighting matrix, symmetric and positive definite for control efforts. The cost function in (8) is a performance measurement. Availability of system states is necessary for the state feedback control design. Using the optimal value function $J^*(\delta_k)$ for (8) as a Lyapunov function $V_L(\delta_k)$, the cost function can be rewritten as [32]

$$\begin{aligned} V_L(\delta_k) &= l(\delta_k) + u_k^T R_z u_k + \sum_{n=k+1}^{\infty} l(\delta_n) + u_n^T R_z u_n \\ &= l(\delta_k) + u_k^T R_z u_k + V_L(\delta_{k+1}) \end{aligned} \quad (9)$$

From Bellman's principle of optimality [33], for the case of infinite time horizon optimization, the value function $V_L^*(\delta_k)$ becomes time invariant and satisfies the discrete-time Bellman equation. The boundary condition $V(0) = 0$ must be satisfied so that $V_L(\delta_k)$ becomes a Lyapunov function. The following discrete-time Hamiltonian [32] is defined

$$H(\delta_k, u_k) = l(\delta_k) + u_k^T R_z u_k + V_L(\delta_{k+1}) - V_L(\delta_{k+1}) - V_L(\delta_k) \quad (10)$$

A necessary condition for the optimal control law u_k is $\partial H(\delta_k, u_k) / \partial u_k = 0$. Then,

$$0 = 2R u_k + \frac{\partial V_L(\delta_{k+1})}{\partial u_k} = 2R u_k + g_s^T(z_k) \frac{\partial V_L(\delta_{k+1})}{\partial \delta_k} \quad (11)$$

The optimal control law to achieve trajectory tracking is therefore formulated as follows

$$u_k^* = \frac{1}{2} R^{-1} g_s^T(z_k) \frac{\partial V_L(\delta_{k+1})}{\partial \delta_{k+1}} \quad (12)$$

By substituting (12) in (10), the Hamilton–Jacobi–Bellman (HJB) equation is obtained. Solving this equation is a challenging task, and to address this problem, inverse optimal control based on Lyapunov stability is proposed [32].

Definition: Tracking Inverse Optimal Control

For the control law in (12) and tracking error $\delta_k = z_k - z_{r,k}$, inverse optimal stabilization along the desired trajectory is guaranteed if:

- (i) (global) asymptotic stability of z_k along reference $z_{r,k}$ is achieved
- (ii) $V_L(\delta_k)$ is a (radially unbounded) positive definite function and satisfies the inequality

$$\bar{V}(\delta_k) : V_L(\delta_{k+1}) - V_L(\delta_k) + u_k^{T*} R_z u_k^* \leq 0 \quad (13)$$

The inverse optimal control law for trajectory tracking is based on a knowledge of $V_L(\delta_k)$, and a Control Lyapunov Function (CLF) $V_L(\delta_k)$ is proposed to ensure that requirements (i) and (ii) are satisfied. Hence, instead of solving the HJB equation for $V_L(\delta_k)$ a CLF is proposed as follows:

$$V_L(\delta_k) = \frac{1}{2} \delta_k^T P_L \delta_k, P_L = P^T > 0 \quad (14)$$

By substituting (14) in (12), the following control law is obtained:

$$u_k^* = \frac{1}{2} \left[R_z + \frac{1}{2} g_s^T(z_k) P_L g_s(z_k) \right]^{-1} g_s^T(z_k) P_L (f(z_k) - z_{r,k+1}) \quad (15)$$

where P_L and R_z are positive definite symmetric matrices that guarantee the existence of the inverse function in (15). The proof that the control law in (15) ensures stability and optimality for (1) is given in [32].

4. Metaheuristic Optimization Algorithms

In this section, the four optimization algorithms used in this paper are briefly described. It is worth noting that the versions used are the original proposed by their respective author, thus variants, hybrids, or improved versions are not considered.

4.1. Ant Lion Optimizer

The main inspiration for the ALO algorithm was the foraging behavior of antlion larvae [16]. The algorithm mimics the interactions between antlions and their prey (ants) in traps. To model these interactions, ants move around the search space and antlions try to hunt them and become more sophisticated at using traps. Because the movements of ants when searching for food are stochastic in nature, a random walk or move is chosen to model this movement, as follows:

$$X(t) = [0, \text{cumsum}(2r(t_1) - 1), \text{cumsum}(2r(t_2) - 1), \dots, \text{cumsum}(2r(t_n) - 1)] \quad (16)$$

where *cumsum* computes the cumulative sum, n represents the number of iterations, t is the step of the random walk (or iteration), and $r(t)$ is a function that is defined as follows:

$$r(t) = \begin{cases} 1 & \text{if } rand > 0.5 \\ 0 & \text{if } rand \leq 0.5 \end{cases} \quad (17)$$

where t is the step of the random walk and *rand* is a random number produced with a uniform distribution on the interval $[0, 1]$.

The random moves are based on Equation (17). Ants update their position with a random move at every step of the process. Given that every search space has a boundary, Equation (16) is not directly used to update the position of an ant. In order to keep the random moves inside the search space, they need to be normalized using the following equation, which is a min–max normalization:

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i - c_i^t) + c_i}{(b_i^t - a_i)} \quad (18)$$

where a_i is the minimum random move for the i th variable, b_i is the maximum random move for the i th variable, c_i^t is the minimum for the i th variable at the t th iteration, and d_i^t is the maximum for the i th variable at the t th iteration. Equation (18) should be applied at each iteration to ensure that random moves are confined to the search space.

When the prey is close to the trap, antlions hurl grains of sand outwards from the pit, causing the ant to slide further down. This behavior is modeled by reducing the radius of the ants' random walk using the following equations:

$$c^t = \frac{c^t}{I}, d^t = \frac{d^t}{I} \quad (19)$$

where I is a ratio, c^t is the minimum of all the variables in the t th iteration and d^t is a vector that includes the maximum of all variables in the t th iteration.

When an ant falls to the bottom of the pit, the antlion updates its position to the last position of the ant in order to increase the chances of catching the prey. For this, the following equation is used:

$$Antlion_j^t = Ant_i^t, \text{ if } (Ant_i^t) > f(Antlion_j^t) \quad (20)$$

where t is the current iteration, $Antlion_j^t$ is the position of the chosen j th antlion at t th iteration, and Ant_i^t is the position of the i th ant at t th iteration.

At the end, the fittest antlion for each iteration is kept and considered elite. Given that the elite is the best antlion, it affects the movements of all the ants during the iterations. Thus, every ant walks randomly around a selected antlion by means of a roulette wheel and the elite as follows:

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \quad (21)$$

where R_A^t is the random move around the selected antlion by the roulette wheel at the t th iteration, R_E^t is the random move around the elite at the t th iteration, and Ant_i^t is the position of the t th ant at the t th iteration. Figure 1 shows the pseudo code for the ALO algorithm.

```

Initialize the first population of ants and antlions randomly
Calculate the fitness of ants and antlions
Find the best antlion and consider it as the elite (determined optimum)
while the end criterion is not satisfied
    for every ant
        Select an antlion using the roulette wheel
        Update c and d using equation 19
        Create a random walk and normalize it using equations 16 and 18
        Update the position of the ant using equation 21
    end for
    Calculate the fitness of all ants
    Replace an antlion with its corresponding ant if it becomes fitter using equation 20
    Update elite if an antlion becomes fitter than the elite
end while
Return elite

```

Figure 1. Pseudo code of the Ant Lion Optimizer algorithm.

4.2. Grey Wolf Optimizer

The Grey Wolf Optimizer is inspired in the collective behavior of a pack of wolves hunting together [23]. Grey wolves show a very well-defined social hierarchy, where the leader of the pack is called dominant or alpha (α). The alpha wolf is in charge of making decisions about moving from one area to another, when to sleep, hunt, etc. The next level in their hierarchy is the beta wolf (β), which is the group of wolves that help the alpha to discipline the pack. The beta wolf is normally the best

candidate to lead the pack when the alpha dies or becomes too old. The next level is the delta wolf (δ) and they serve as scouts or sentinels. Lastly, the omega wolf (ω) is the lowest in rank, it serves as the scapegoat and they are the last ones to eat. The omega wolves are actually very useful to the pack, since without them the pack would fight with each other.

Grey wolves work as a team and their hunting strategy is described in three steps: Finding and chasing the prey, surrounding and frightening until it stops moving, and attack. In the GWO algorithm, the search for prey corresponds to the exploration of the space and the attacking step to the exploitation phase. The encircling behavior is governed by the following equations:

$$D = |C \cdot X_p(t) - X(t)| \quad (22)$$

$$X(t+1) = X_p(t) - A \cdot D \quad (23)$$

where t is the current iteration, X_p is a vector that represents the position of the prey, A and C are coefficient vectors, and X the position of a grey wolf.

Vectors A and C are obtained as follows:

$$A = 2a \cdot r_1 - a \quad (24)$$

$$C = 2 \cdot r_2 \quad (25)$$

where r_1 and r_2 are random vectors in $[0, 1]$ and a is a vector with components that decrease linearly from 2 to 0 during iterations.

The alpha wolf usually leads the hunt and the beta and delta join. However, in an optimization search space, the location of the best position is not known; therefore, the GWO algorithm assumes that these wolves are closer to the location of the prey. Their three positions are saved, and the rest of the pack is encouraged to follow them updating their positions according to the following formulas:

$$D_\alpha = |C_1 \cdot X_\alpha - X|, D_\beta = |C_2 \cdot X_\beta - X|, D_\delta = |C_3 \cdot X_\delta - X| \quad (26)$$

$$X_1 = X_\alpha - A_1 \cdot (D_\alpha), X_2 = X_\beta - A_2 \cdot (D_\beta), X_3 = X_\delta - A_3 \cdot (D_\delta) \quad (27)$$

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3} \quad (28)$$

The pseudo code in Figure 2 describes the GWO algorithm:

```

Initialize the grey wolf population  $X_i$  ( $i = 1, 2, 3, \dots, n$ )
Initialize  $a$ ,  $A$ , and  $C$ 
Calculate the fitness of each search agent
 $X_\alpha$  = the best search agent
 $X_\beta$  = the second best search agent
 $X_\delta$  = the third best search agent
while ( $t < \text{Maximum number of iterations}$ )
    for each search agent
        Update the position of the current search agent using equation (28)
    end for
    Update  $a$ ,  $A$ , and  $C$ 
    Calculate the fitness of all search agents
    Update  $X_\alpha$ ,  $X_\beta$ , and  $X_\delta$ 
     $t = t + 1$ 
end while
Return  $X_\alpha$ 

```

Figure 2. Pseudo code of the Grey Wolf Optimizer algorithm.

4.3. Harris Hawks Optimization

The Harris Hawks Optimization relatively new algorithm proposed by Heidari et al. [27] in 2019. It is a nature-inspired population-based technique that mimics the cooperative behavior and hunting style of Harris' hawks. These raptors pounce a prey from different angles, attempting to surprise it and they display different chasing patterns depending on the particular natural scenarios and escaping patterns of the pray (usually rabbits). The HHO algorithm, as many of the other nature-inspired techniques, has both exploration and exploitation phases. For the exploration phase, they have mainly two strategies: one of them consists in perching based on the locations of other hawks, in order to be close to them at the moment of the attack, and the second strategy is perching based on random tall trees, still close to other members. Both strategies are equally probable and denoted by q in Equation (29), which models this behavior:

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 |X_{rand}(t) - 2r_2 X(t)| & \text{if } q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3 (LB + r_4 (UB - LB)) & \text{if } q < 0.5 \end{cases} \quad (29)$$

where $X(t+1)$ is the position vector of the hawks for the next iteration t . X_{rabbit} is the position of the rabbit, $X(t)$ is the current position of the hawks, r_1, r_2, r_3, r_4 , and q are random numbers in the range of $(0, 1)$ and are updated in each iteration, LB and UB are the upper and lower bounds of variables, $X_{rand}(t)$ is a hawk selected randomly from the current population, and X_m is the average position of the current population of hawks.

The HHO algorithm transfers from exploration to exploitation and changes between different exploitative strategies based on the energy of the prey. This energy decreases during the escaping behavior, which is modeled by the following equation:

$$E = 2E_0(1 - \frac{t}{T}) \quad (30)$$

where E is the escaping energy of the rabbit, T is the maximum number of iterations, and E_0 is the initial state of the energy. E_0 randomly changes in the interval $(-1, 1)$ at each iteration. When its value decreases from 0 to -1 , the prey is physically weak, and when it increases from 0 to 1, the prey strengthens. The dynamic escaping energy E decreases during the iterations.

The next phase is called "soft besiege" when the rabbit still has energy and tries to escape performing random misleading jumps. This behavior is modeled by the following rules:

$$X(t+1) = \Delta X(t) - E |J X_{rabbit}(t) - X(t)| \quad (31)$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \quad (32)$$

where $\Delta X(t)$ is the difference between the position vector of the prey and the current location at iteration t , $J = 2(1 - r_5)$ represents the random hop strength of the rabbit when trying to escape, and r_5 is a random number in $(0, 1)$. The value of J changes to simulate the random movement of the rabbit.

"Hard besiege" is when the prey is very tired and has low escaping energy. The Harris's hawks surround the prey to finally execute the surprise pounce. Their current positions are updated by Equation (33):

$$X(t+1) = X_{rabbit}(t) - E |\Delta X(t)| \quad (33)$$

The pseudo code for the HHO algorithm is described in Figure 3.


```

Inputs: The population size  $N$  and maximum number of iterations  $T$ 
Outputs: The location of the rabbit and its fitness value
Initialize the random population  $X_i (i=1,2,...,N)$ 
while (stopping condition is not met)
    Calculate the fitness values of the hawks
    Set  $X_{rabbit}$  as the location of the rabbit (best location)
    for (each hawk ( $X_i$ ))
        Update the initial energy  $E_0$  and jump strength  $J$  //  $E_0 = 2rand() - 1, J = 2(1 - rand())$ 
        Update the  $E$  using equation 30
        if ( $|E| \geq 1$ ) then //Exploration phase
            Update the location vector using equation 29
        if ( $|E| < 1$ ) then //Exploitation phase
            if ( $|E| \geq 0.5$ ) then //Soft besiege
                Update the location vector using equation 31
            elseif ( $|E| < 0.5$ ) then //Hard besiege
                Update the location vector using equation 33
    Return  $X_{rabbit}$ 

```

Figure 3. Pseudo code of the Harris Hawks Optimization algorithm.

4.4. Whale Optimization Algorithm

The Whale Optimization Algorithm is inspired by the foraging behavior of humpback whales [25]. These mammals have a complex social behavior, which helps them to hunt in groups to feed on their favorite prey: krill. Humpback whales hunt krill and small fishes that swim close to the surface. They create distinctive bubbles along a circle path around the school, creating what is called a bubble-net to trap their prey.

Because the solution of an optimization problem on the search space is not known, the whale optimization algorithm assumes that the optimal candidate is the target prey. The behavior of the whales or search agents is given by the following equations:

$$D = |CX^*(t) - X(t)| \quad (34)$$

$$X(t+1) = X^*(t) - A \cdot D \quad (35)$$

where t indicates the current iteration, X is the position vector, A and C are coefficient vectors, and X^* is the position vector of the best solution so far. X^* is to be updated in each iteration when a better solution is found. The vectors A and C are determined as follows:

$$A = 2a \cdot r - a, C = 2r \quad (36)$$

where a linearly decreases from 2 to 0 during iterations and r is a random vector in $[0, 1]$.

The exploitation phase is called “Bubble-net attacking” where the whales circle their prey performing a spiral shape, which is described by the following equation:

$$X(t+1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t) \quad (37)$$

where $D' = |X^*(t) - X(t)|$ is the distance of the i th whale to the prey, b is a constant to define the shape of the logarithmic spiral, and l is a random number in $[-1, 1]$.

The exploration phase consists of a random search by each whale with $A \geq 1$ or $A \leq 1$, where A is a random value in order to force each whale to move away from a reference whale. The mathematical model is as follows:

$$D = |C \cdot X_{rand} - X| \quad (38)$$

$$X(t+1) = X_{rand} - A \cdot D \quad (39)$$

where X_{rand} is a random position vector chosen from the current population. The pseudo code for the WOA algorithm is shown in Figure 4.

```

Initialize the whale's population  $X_i(i=1,2,...,N)$ 
Calculate the fitness of each search agent
 $X^*$  = the best search agent
while (t < maximum number of iterations)
  for each search agent
    Update a, A, C, l and p
    if1 ( p < 0.5 )
      if2 (  $|A| < 1$  )
        Update the position of the current search agent using equation 34
      else if2 (  $|A| \geq 1$  )
        Select a random search agent (  $X_{rand}$  )
        Update the position of the current search agent using equation 39
      end if2
    else if1 ( p  $\geq$  0.5 )
      Update the position of the current search using equation 37
    end for
  Check if any search agent goes beyond the search space and amend it
  Calculate the fitness of each search agent
  Update  $X^*$  if there is a better solution
  t = t + 1
end while
Return  $X^*$ 

```

Figure 4. Pseudo code of the Whale Optimization Algorithm algorithm.

5. Optimal Control Strategy Architecture

Nonlinear systems can have a complex dynamic behavior based on the stability of their equilibrium points. Convergence of nonlinear system states to one operating point can be unpredictable depending on the nature and the initial state of the system, such that robust control strategies are necessary. Figure 5 displays the optimal neural control strategy for uncertain nonlinear systems subject to disturbances. The RHONO estimates unmeasured states of the nonlinear system needed to develop a state feedback control law. The controlled system could work properly in the presence of small disturbances but, for large disturbances and parameter uncertainties, the system could be unstable affecting the performance and operating safety. In order to address this issue, metaheuristic algorithms are used to identify optimal tuning parameters for controller. The metaheuristic algorithms propose a P_L matrix for the CLF to reduce the mean square tracking error between the neural model state and its given trajectory reference. Optimization strategy stops when the reference trajectory is reached, and the CLF matrix satisfies the asymptotic stability condition given in Section 3. The optimization strategy is able to find optimal parameters for reference trajectories tracking of different operating points.

Figure 5. Optimal neural tracking control strategy for nonlinear systems.

6. Case Study

The mathematical model is derived from Activated Sludge Model No. 1 (ASM1) [34] and describes the aerobic digestion process that occurs in a CSTR, consisting of the material balances of the components, the kinetics, and the stoichiometry of the major processes in the plant. The TCOD is the variable of interest that will be controlled by regulating the KLA. The following nonlinear discrete-time Equations (40)–(49) describe the biological transformation processes that occur in the reactor, and Figure 6 illustrates the plant configuration.

Figure 6. Wastewater plant configuration.

Readily biodegradable soluble substrate

$$X_{1,k+1} = D(X_{1,in} - X_{1,k}) - \frac{1}{Y_H} \cdot \mu_{\max,H} \cdot \mu_1 \cdot (\mu_3 + \mu_6 \cdot \mu_4 \cdot n_g) X_{3,k} + k_h \cdot \mu_7 (\mu_3 + n_h \cdot \mu_6 \cdot \mu_4) X_{3,k} \quad (40)$$

Slowly biodegradable particulate substrate

$$X_{2,k+1} = D(X_{2,in} - X_{2,k}) + D_r(b-1)X_{2,k} + (1-fp)(b_H X_{3,k} + b_A X_{4,k}) - k_h \cdot \mu_7 (\mu_3 + n_h \cdot \mu_6 \cdot \mu_4) X_{3,k} \quad (41)$$

Active heterotrophic particulate biomass

$$X_{3,k+1} = D(X_{3,in} - X_{3,k}) + D_r(b-1)X_{3,k} + \mu_{\max,H} \cdot \mu_1 \cdot \mu_3 \cdot X_{3,k} + \mu_{\max,H} \cdot \mu_1 \cdot \mu_6 \cdot \mu_4 \cdot n_g \cdot X_{3,k} - b_H \cdot X_{3,k} \quad (42)$$

Active autotrophic particulate biomass

$$X_{4,k+1} = D(X_{4,in} - X_{4,k}) + D_r(b-1)X_{4,k} + \mu_{\max,A} \cdot \mu_5 \cdot \mu_2 \cdot X_{4,k} - b_A \cdot X_{4,k} \quad (43)$$

Soluble oxygen

$$X_{5,k+1} = D(X_{5,in} - X_{5,k}) + KLA(X_{5,max} - X_5) - \frac{(1 - Y_H)}{Y_H} \cdot \mu_{max,H} \cdot \mu_1 \cdot \mu_3 \cdot X_{3,k} - \frac{(4.57 - Y_A)}{Y_A} \cdot \mu_{max,A} \cdot \mu_5 \cdot \mu_2 \cdot X_{4,k} \quad (44)$$

Soluble nitrate and nitrite nitrogen

$$X_{6,k+1} = D(X_{6,in} - X_{6,k}) - \frac{(1 - Y_H)}{2.86Y_H} \cdot \mu_{max,H} \cdot \mu_1 \cdot \mu_6 \cdot \mu_4 \cdot n_g \cdot X_3 + \frac{1}{Y_A} \cdot \mu_{max,A} \cdot \mu_5 \cdot \mu_2 \cdot X_4 \quad (45)$$

Soluble ammonium (NH₄⁺) and NH₃ nitrogen

$$X_{7,k+1} = D(X_{7,in} - X_{7,k}) - i_{XB} \mu_{max,H} \cdot \mu_1 (\mu_3 + \mu_6 \cdot \mu_4 \cdot n_g) X_3 - \left(i_{XB} + \frac{1}{Y_A} \right) \mu_{max,A} \cdot \mu_5 \cdot \mu_2 \cdot X_{4,k} + k_A \cdot X_{8,k} \cdot X_{3,k} \quad (46)$$

Soluble biodegradable organic nitrogen

$$X_{8,k+1} = D(X_{8,in} - X_{8,k}) - k_A \cdot X_{8,k} \cdot X_{3,k} + k_h \cdot \mu_7 (\mu_3 + n_h \mu_6 \mu_4) X_{3,k} \frac{X_{9,k}}{X_{2,k}} \quad (47)$$

Particulate biodegradable organic nitrogen

$$X_{9,k+1} = D(X_{9,in} - X_{9,k}) + D_r(b - 1)X_{9,k} + (i_{XB} - f_p \cdot i_{XP}) (b_H \cdot X_{3,k} + b_A \cdot X_{4,k}) - k_h \cdot k_{sat} \cdot (\mu_3 + n_h \cdot \mu_6 \cdot \mu_4) X_{3,k} \cdot \frac{X_{9,k}}{X_{2,k}} \quad (48)$$

Parameters of the model and Monod reaction rates μ_i , $i = 1, \dots, 7$ [35] are given in Appendix A. The inputs of the plant are the dilution rates (D , D_r), KLA, biomass, and substrate concentrations ($X_{i,in}$). The output of the plant is the TCOD given by (49), which is composed of the readily biodegradable soluble substrate ($X_{1,k}$), the slowly biodegradable particulate substrate ($X_{2,k}$), and the inert soluble and particulate organic material (I_s). The steady-state value for I_s is given by the corresponding constant value of the influent:

$$TCOD = X_{1,k} + X_{2,k} + I_s \quad (49)$$

7. Results

The performance of the control strategy for trajectory tracking is verified with the biological nonlinear model (40)–(49), in the presence of load disturbances and parameter uncertainty. Reference trajectories are proposed based on the hydraulic retention time (HRT) of the wastewater treatment plant, and the maximal daily rate of organic biodegradability per biomass unit. Neural network structure for the biological model consists of five neurons in the hidden layer and one neuron at the output layer, with a logistic and a linear activation function, respectively. The neural network estimates the $X_{1,k}$ and $X_{2,k}$ states of the system in order to calculate the TCOD output. Parameter of the neural structure are given in Appendix A. The optimization consists of using each of the metaheuristic algorithms (ALO, GWO, HHO, and WOA) to find the best set of parameters to configure the control law (15) for optimal trajectory tracking. In order to configure the control law, matrix P_L is used as the variable to be optimized. P_L is a symmetric matrix, where all values must be real positive numbers. Because the matrix is symmetric, values P_{L12} and P_{L21} are the same, thus the vector to be optimized consists of three variables: $[P_{L11}, P_{L12}, P_{L23}]$. The fitness function is the mean square tracking error between the neural model state and its given trajectory reference; thus, the objective is to minimize the area between desired trajectory and the actual output. In this section, two cases are presented where tracking error is compared when the system is subject to load disturbances and parameter uncertainty, respectively. In the following tests, the metaheuristic algorithms were configured as follows: 50 search agents, 100 iterations, a lower boundary of 0.00001, and an upper boundary of 1. MATLAB R2016a (Mathworks Inc., Natick, MA, USA) was used for these mathematical simulations.

7.1. External Disturbance

In all of the case studies, reference paths are used to comply with the TCOD requirements. The allowable TCOD concentration leaving a wastewater treatment plant has a maximum value of 50 mg COD/L. A step disturbance with an amplitude equal to 40% of the steady-state value X_{1in} (the input concentration of readily biodegradable substrate) is applied at $t = 5$ d. The tracking error comparison is displayed in Figures 7 and 8.

Figure 7. Tracking performance comparison in presence of external disturbance: (a) $X_{1,k}$ and (b) $X_{2,k}$.

Figure 8. Tracking performance comparison in presence of external disturbance: (a) TCOD control and (b) KLA signal.

Figure 7 displays the tracking error for substrates along their respective desired trajectories. For the case of the neural model control with ALO algorithm, the tracking performance for both substrates is better than the other methods. For both sceneries, the neural model presents a slightly transient state error when disturbance is applied at $t = 5$ days. Figure 8a shows the TCOD tracking error as function of the substrates defined by (49). As in the previous test, the TCOD with ALO algorithm presents a good convergence along the desired trajectory as compared with the other methods. Figure 8b portrays the KLA signal behavior to reach the reference trajectories of the system. A time variant behavior of KLA is observed before the disturbance is applied, and then the KLA signal shows an incremental response over the remaining simulation time to reject the disturbance. For the case of the neural model control with ALO algorithm, the KLA signal illustrates a stable increasing behavior over the simulation time up to a value of 5 d^{-1} , maintaining the system along the reference trajectories. Thus, the control law with ALO outperforms the other methods in terms of tracking trajectory in a nonlinear disturbed system.

7.2. Parameter Uncertainty

Active heterotrophs are known to govern TCOD removal in activated sludge systems, and the biodegradation rates are expressed in terms of the active biomass. In this test, the biodegradation rate is investigated by increasing the maximum specific growth rate $\mu_{\max,H}$ by 20% for active heterotrophic biomass at $t = 5$ d. Figures 9 and 10 illustrate the tracking error comparison in the presence of parameter uncertainty.

Figure 9. Tracking performance comparison in presence of parameter uncertainty: (a) $X_{1,k}$ and (b) $X_{2,k}$.

Figure 10. Tracking performance comparison in presence of parameter uncertainty: (a) TCOD control and (b) KLA signal.

Figure 9 illustrates the tracking error performance for the substrates in the presence of parameter uncertainty. The neural model control using metaheuristic algorithms shows a good tracking performance with a neglected transient state error. For this test, HHO shows a better convergence along the trajectory reference in presence of parameter uncertainty. Figure 10a shows the TCOD tracking error as a result of the transient error of the substrates. Figure 10b displays the KLA signal behavior to reach reference trajectories of the system. The KLA shows a time variant behavior after the parameter uncertainty. For the case of the control law with WOA algorithm, KLA shows an increasing stable performance over simulation time up to a value of 4.8 d^{-1} to reject the disturbance. Thus, the control strategy with WOA demonstrates a more stable behavior for KLA, but control law with HHO shows a better convergence for the neural states. Thus, the control law combined with these metaheuristic algorithms is effective in terms of tracking trajectory in presence of parameter uncertainty.

8. Conclusions

This paper proposes an optimal neural tracking control strategy for nonlinear uncertain systems subject to external disturbances. The novelty of this strategy is the use of the newest metaheuristic

algorithm (ALO, GWO, HHO, and WOA) implementation to find optimal tuning of the control law. Efficacy of the coupled strategy is proven in a highly nonlinear biological model subject to disturbances, uncertainties, and hard to measure state variables. A neural observer based on a RHONN is proposed to estimate the unmeasured states of the system necessary to develop a state feedback control law. Inverse optimal control theory for trajectory tracking based on a neural model is used to design a stabilizing control law. In order to evaluate the efficiency of the coupled strategy, four configurations with metaheuristic algorithms are used. Two case studies are examined, system subject to load disturbances, and parameter uncertainty, respectively. Results via simulations show that the optimal control law combined with the ALO metaheuristic algorithm is a robust method of stabilizing the system states along the reference trajectories in presence of external disturbances. Optimal control law combined with the HHO is effective in presence of parameter uncertainty. Control strategy with the other methods shows a significant transient state error due to a limited tuning. Thus, optimal operation of the plant can improve energy efficiency and increase the treatment capacity. As future work, the control strategy will be proven in a real scale pilot plant for wastewater treatment to validate its efficacy.

Author Contributions: Methodology, R.R.-C.; formal analysis, K.J.G.-T., V.Z.-G.; writing—original draft preparation, R.R.-C.; writing—review and editing, K.J.G.-T., V.Z.-G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Basic Science—CONACYT Mexico through Project 258960.

Acknowledgments: A graduate scholarship for Roxana Recio-Colmenares was provided by the National Council for Science and Technology (CONACyT) of México.

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

Appendix A

Table A1. Mathematical model parameters.

Parameter	Values	Stoichiometric Coefficient	
		Units	Description
Y_A	0.24	g COD/g N	Autotrophic yield coefficient
Y_H	0.67	g CDO/g COD	Heterotrophic yield coefficient
$\mu_{\max,A}$	0.8	d ⁻¹	Maximum specific growth rate for autotrophs
$\mu_{\max,H}$	6	d ⁻¹	Maximum specific growth rate for heterotrophs
μ_1	-	-	Monod kinetics for readily biodegradable soluble substrate
μ_2	-	-	Monod kinetics for the component $X_{5,k}$ as function of $X_{4,k}$
μ_3	-	-	Monod kinetics for the component $X_{5,k}$ as function of $X_{3,k}$
μ_4	-	-	Monod kinetics for soluble nitrate and nitrite nitrogen
μ_5	-	-	Monod kinetics for soluble ammonium nitrogen
μ_6	-	-	Inhibition kinetics for $X_{5,k}$
μ_7	-	-	Saturation kinetics
b	0.9	-	Fraction coefficient for D_r
b_A	0.05	d ⁻¹	Autotrophic decay coefficient
b_H	0.22	d ⁻¹	Heterotrophic decay coefficient
f_P	0.08	(-)	Fraction of biomass yielding particulate products
i_{XB}	0.086	mg N/mg SS	Nitrogen content in biomass
i_{XP}	0.06	mg N/mg SS	Nitrogen content in inert particulate
k_A	0.081	L/g COD. d	Ammonification coefficient
k_h	3	d ⁻¹	Hydrolysis coefficient
η_g	0.8	(-)	Correction factor for anoxic growth of heterotrophs

Table A1. Cont.

Stoichiometric Coefficient			
Parameter	Values	Units	Description
η_h	0.4	(-)	Correction factor for anoxic hydrolysis
I_S	5	mg COD/L	Concentration of soluble and particulate inert organic matter
CSTR Parameters			
Parameter	Values	Units	Description
D	2	(d ⁻¹)	Dilution rate
D_r	1	(d ⁻¹)	Dilution recycle rate
KLA	-	d ⁻¹	Oxygen transfer coefficient
$X_{i,in}$	200	mg COD/L	Initial condition of $X_{1,k}$
$X_{2,in}$	100	mg COD/L	Initial condition of $X_{2,k}$
$X_{3,in}$	0	mg COD/L	Initial condition of $X_{3,k}$
$X_{4,in}$	0	mg COD/L	Initial condition of $X_{4,k}$
$X_{5,in}$	2	mg/L	Initial condition of $X_{5,k}$
$X_{6,in}$	1	mg N/L	Initial condition of $X_{6,k}$
$X_{7,in}$	15	mg N/L	Initial condition of $X_{7,k}$
$X_{8,in}$	9	mg N/L	Initial condition of $X_{8,k}$
$X_{9,in}$	0	mg COD/L	Initial condition of $X_{9,k}$
$X_{5,max}$	10	mg/L	Maximum concentration of soluble oxygen
V	15	L	Tank volume

Table A2. Control law parameters.

Parameter	Values	Units	Description
R_Z	0.2	-	Weighting on control efforts
P_L	$a_{11} = 0.007444$ $a_{12} = a_{21} = 0.001412$ $a_{22} = 0.000546$	-	Symmetric matrix of CLF (found by metaheuristic algorithm)

Table A3. Neural observer parameters.

Parameter	Values	Units	Description
$P1,(0)$	1400	-	prediction error associated covariance matrix for $X_{1,k}$ estimation
$Q1,(0)$	1	-	state noise associated covariance matrix for $X_{1,k}$ estimation
$R1,(0)$	61	-	measurement noise associated covariance matrix for $X_{1,k}$ estimation
$P2,(0)$	1500	-	prediction error associated covariance matrix for $X_{2,k}$ estimation
$Q2,(0)$	1.3	-	state noise associated covariance matrix for $X_{2,k}$ estimation
$R2,(0)$	63	-	measurement noise associated covariance matrix for $X_{2,k}$ estimation
η_1	2	-	EKF learning rate for $X_{1,k}$ estimation
η_2	1.07	-	EKF learning rate for $X_{2,k}$ estimation
$W1$	$[6 \times 1] = 0.1$	-	Weight vector for $X_{1,k}$ estimation (online learned)
$W2$	$[6 \times 1] = 0.5$	-	Weight vector for $X_{2,k}$ estimation (online learned)
$g1, g2$	$[1 \times 2] = 0.12$	-	Luenberger gains
$K_{i,k}$	-	-	Kalman gain matrix
$M_{i,k}$	-	-	Global scaling matrix

References

1. Kumar, A.; Jain, T. Sub-optimal Control Design for Second Order Non-linear Systems using Krotov Sufficient Conditions. *IFAC Pap.* **2020**, *53*, 272–276. [\[CrossRef\]](#)
2. Yu, Q.; Hou, Z.; Bu, X.; Yang, J. Observer-based data-driven constrained norm optimal iterative learning control for unknown non-affine non-linear systems with both available and unavailable system states. *J. Frankl. Inst.* **2020**, *357*, 5852–5877. [\[CrossRef\]](#)
3. Ballesteros, M.; Chairez, I.; Poznyak, A. Robust optimal feedback control design for uncertain systems based on artificial neural network approximation of the Bellman's value function. *Neurocomputing* **2020**, *413*, 134–144. [\[CrossRef\]](#)
4. Wu, X.; Shen, J.; Wang, M.; Lee, K.Y. Intelligent predictive control of large-scale solvent-based CO₂ capture plant using artificial neural network and particle swarm optimization. *Energy* **2020**, *196*, 117070. [\[CrossRef\]](#)
5. Gurubel, K.J.; Sanchez, E.N.; Coronado-Mendoza, A.; Zuniga-Grajeda, V.; Sulbaran-Rangel, B.; Breton-Deval, L. Inverse optimal neural control via passivity approach for nonlinear anaerobic bioprocesses with biofuels production. *Optim. Contr. Appl. Methods* **2019**, *40*, 848–858. [\[CrossRef\]](#)
6. Mcclamroch, N.H.; Kolmanovsky, I. Performance benefits of hybrid control design for linear and nonlinear systems. *Proc. IEEE* **2000**, *88*, 1083–1096. [\[CrossRef\]](#)
7. Gurubel, K.J.; Sanchez, E.N.; González, R.; Coss y León, H.; Recio, R. Artificial Neural Networks Based on Nonlinear Bioprocess Models for Predicting Wastewater Organic Compounds and Biofuel Production. In *Artificial Neural Networks for Engineering Applications*; Academic Press: St. Louis, MI, USA, 2019; pp. 79–96. [\[CrossRef\]](#)
8. Alanis, A.Y.; Sanchez, E.N. *Discrete-Time Neural Observers: Analysis and Applications*; Academic Press: London, UK, 2017.
9. Alanis, A.Y.; Sanchez, E.N.; Loukianov, A.G.; Hernandez, E.A. Discrete-time recurrent high order neural networks for nonlinear identification. *J. Frankl. Inst.* **2010**, *347*, 1253–1265. [\[CrossRef\]](#)
10. Siregar, S.P.; Wanto, A. Analysis of Artificial Neural Network Accuracy Using Backpropagation Algorithm In Predicting Process (Forecasting). *Int. J. Inf. Syst. Technol.* **2017**, *1*, 34–42. [\[CrossRef\]](#)
11. Pisa, I.; Morell, A.; Vicario, J.L.; Vilanova, R. Denoising Autoencoders and LSTM-Based Artificial Neural Networks Data Processing for Its Application to Internal Model Control in Industrial Environments-the Wastewater Treatment Plant Control Case. *Sensors* **2020**, *20*, 3743. [\[CrossRef\]](#)
12. Schmitt, F.; Banu, R.; Yeom, L.-T.; Do, K.-U. Development of artificial neural networks to predict membrane fouling in an anoxic-aerobic membrane bioreactor treating domestic wastewater. *Biochem. Eng. J.* **2018**, *133*, 47–58. [\[CrossRef\]](#)
13. Sadeghassadi, M.; Macnab, C.J.B.; Gopaluni, B.; Westwick, D. Application of neural networks for optimal-setpoint design and MPC control in biological wastewater treatment. *Comput. Chem. Eng.* **2018**, *115*, 150–160. [\[CrossRef\]](#)
14. Leon, B.S.; Alanis, A.Y.; Sanchez, E.N.; Ornelas-Tellez, F.; Ruiz-Velazquez, E. Neural Inverse Optimal Control via Passivity for Subcutaneous Blood Glucose Regulation in Type 1 Diabetes Mellitus Patients. *Intell. Autom. Soft. Comput.* **2014**, *20*, 279–295. [\[CrossRef\]](#)
15. Mirjalili, S.; Dong, J.S.; Lewis, A. *Nature-Inspired Optimizers: Theories, Literature Reviews and Applications*; Springer: Cham, Switzerland, 2019; Volume 811. [\[CrossRef\]](#)
16. Heidari, A.A.; Faris, H.; Mirjalili, S.; Aljarah, I.; Mafarja, M. Ant lion optimizer: Theory, literature review, and application in multi-layer perceptron neural networks. In *Nature-Inspired Optimizers*; Springer: Cham, Switzerland, 2020; pp. 23–46. [\[CrossRef\]](#)
17. Fathy, A.; Kassem, A.M. Antlion optimizer-ANFIS load frequency control for multi-interconnected plants comprising photovoltaic and wind turbine. *ISA Trans.* **2019**, *87*, 282–296. [\[CrossRef\]](#)
18. Pradhan, R.; Majhi, S.K.; Pradhan, J.K.; Pati, B.B. Antlion optimizer tuned PID controller based on Bode ideal transfer function for automobile cruise control system. *J. Ind. Inf. Integr.* **2018**, *9*, 45–52. [\[CrossRef\]](#)
19. Verma, S.; Mukherjee, V. Optimal real power rescheduling of generators for congestion management using a novel ant lion optimiser. *IET Gener. Transm. Dis.* **2016**, *10*, 2548–2561. [\[CrossRef\]](#)
20. Gupta, E.; Saxena, A. Performance evaluation of antlion optimizer based regulator in automatic generation control of interconnected power system. *J. Eng.* **2016**. [\[CrossRef\]](#)

21. Saikia, L.C.; Sinha, N. Automatic generation control of a multi-area system using ant lion optimizer algorithm based PID plus second order derivative controller. *Int. J. Electr. Power.* **2016**, *80*, 52–63. [[CrossRef](#)]
22. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [[CrossRef](#)]
23. Oliveira, J.; Oliveira, P.M.; Boaventura-Cunha, J.; Pinho, T. Evaluation of Hunting-Based Optimizers for a Quadrotor Sliding Mode Flight Controller. *Robotics* **2020**, *9*, 22. [[CrossRef](#)]
24. Arefi, M.; Chowdhury, B. Post-fault transient stability status prediction using Grey Wolf and Particle Swarm Optimization. In Proceedings of the SoutheastCon 2017, Charlotte, NC, USA, 30 March–2 April 2017; pp. 1–8. [[CrossRef](#)]
25. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [[CrossRef](#)]
26. Tien Bui, D.; Abdullahi, M.M.; Ghareh, S. Fine-tuning of neural computing using whale optimization algorithm for predicting compressive strength of concrete. *Eng. Comput.* **2019**. [[CrossRef](#)]
27. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Generat. Comput. Syst.* **2019**, *97*, 849–872. [[CrossRef](#)]
28. Devarapalli, R.; Bhattacharyya, B. Application of Modified Harris Hawks Optimization in Power System Oscillations Damping Controller Design. In Proceedings of the 8th International Conference on Power Systems (ICPS), Jaipur, India, 20–22 December 2019; pp. 1–6. [[CrossRef](#)]
29. Alanis, A.Y.; Sanchez, E.N.; Ricalde, L.J. Discrete-time reduced order neural observers for uncertain nonlinear systems. *Int. J. Neural Syst.* **2010**, *20*, 29–38. [[CrossRef](#)] [[PubMed](#)]
30. Haykin, S. *Kalman Filtering and Neural Networks*; John Wiley & Sons: New York, NY, USA, 2004; Volume 47.
31. Sanchez, E.N.; Alanis, A.Y.; Loukianov, A.G. *Discrete-Time High Order Neural Control Trained with Kalman Filtering*; Springer: Berlin, Germany, 2008.
32. Sanchez, E.N.; Ornelas-Tellez, F. *Discrete-Time Inverse Optimal Control for Nonlinear Systems*; CRC Press: Boca Raton, FL, USA, 2017.
33. Wang, D.; Liu, D.; Wei, Q.; Zhao, D.; Jin, N. Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. *Automatica* **2012**, *48*, 1825–1832. [[CrossRef](#)]
34. Henze, M.; Gujer, W.; Mino, T.; van Loosdrecht, M.C. *Activated Sludge Models ASM1, ASM2, ASM2d and ASM3*; IWA Publishing: London, UK, 2000.
35. Nelson, M.I.; Sidhu, H.S. Analysis of the activated sludge model (number 1). *Appl. Math. Lett.* **2009**, *22*, 629–635. [[CrossRef](#)]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).